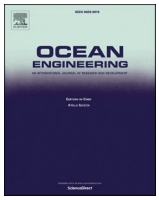
Ocean Engineering 281 (2023) 115018

Contents lists available at ScienceDirect

Ocean Engineering

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



Real-time planning and collision avoidance control method based on deep reinforcement learning

Xinli Xu [a], Peng Cai [b], Yunlong Cao [a], Zhenzhong Chu [a], Wenbo Zhu [a], Weidong Zhang [c,d,*]

[a] School of Mechanical Engineering, University of Shanghai for Science and Technology, Shanghai, 200093, China
[b] School of Mechanical Engineering, Tsinghua University, Beijing, 100084, China
[c] School of Information and Communication Engineering, Hainan University, Haikou, 570228, Hainan, China
[d] Department of Automation, Shanghai Jiao Tong University, Shanghai, 200240, China

ARTICLE INFO

Keywords:
Unmanned surface vehicle
Real-time planning
Collision avoidance control
Deep reinforcement learning
Neural network
Experience playback

ABSTRACT

Real time planning and collision avoidance is an important part of the motion planning for USVs, its core is to design effective planning and collision avoidance methods. This paper proposes a real-time planning and collision avoidance method based on deep reinforcement learning for USVs in complex environments. Firstly, a novel reward function is designed, which transforms the collision avoidance problem into a deep reinforcement learning strategy solution problem that includes distance, direction, speed, and GOLREGs constraints, it ensures the smooth and safe operation of the USV. Secondly, inefficient exploration and exploitation methods make it difficult for intelligent agents to discover key information in the environment, high value experience is obtained through the designed layered sampling exploration mechanism in this article, the intelligent agent learns effective control strategies through this mechanism, which accelerates the convergence of the strategies and reduces the waste of computing resources. Finally, a real-time collision avoidance simulation platform for the USV is established, and experimental results of different exploration mechanisms are compared, the efficiency of the layered sampling exploration mechanism has been verified. The simulation scenarios from easy to difficult are designed, and the results indicate that the USV can complete real-time planning and collision avoidance tasks in complex environments.

1. Introduction

Real-time path planning and collision avoidance require the unmanned surface vehicle (USV) to avoid obstacles in a reasonable way and complete the navigation task safely and efficiently. According to whether the environmental information is fully utilized, collision avoidance methods can be divided into two categories: The first type is the pre-generation method, typical examples include A* algorithm (Duchoň et al., 2014) and Dijkstra algorithm (Hwang et al., 2003). This method uses the whole environment model to seek the best safety track, which is a global planning but it cannot be adjusted in real time (González et al., 2015); The second type is reactive method, such as artificial potential field method, rapid expansion random tree, etc. For this kind of method, agent only uses the detection data to find the safety track, which is a local planning, and it is easy to implement online (Ji et al., 2017; Wu et al., 2021). For example, Khatib (1985) implemented real-time operations in a complex environment by constructing a potential field environment model. Kothari et al. (Kothari and Postlethwaite, 2013) proposed a real-time probabilistically robust path planner based on the rapidly-exploring random tree (RRT) algorithm. Alvarez et al. (2004) proposed a genetic algorithm (GA) for path planning of an autonomous underwater vehicle in an ocean environment characterized by strong currents and enhanced space-time variability. Belkhouche et al. (Belkhouche, 2009) dealt with the problem of path planning in a dynamic environment through velocity obstacle method.

Obviously, in special application scenarios, reactive method has greater advantages because it provides real-time feedback. However, most of the reported reactive methods usually rely on prior knowledge and rules of artificial design, and the robustness and generalization of the algorithm may be affected (Guoqing et al., 2018). In particular, the design of collision avoidance controllers is more challenging in complex nonlinear environments.

With the development of deep learning and reinforcement learning techniques, gradually some scholars have applied deep reinforcement

* Corresponding author. School of Information and Communication Engineering, Hainan University, Haikou, 570228, Hainan, China.
E-mail address: wdzhang@sjtu.edu.cn (W. Zhang).

https://doi.org/10.1016/j.oceaneng.2023.115018
Received 27 January 2023; Received in revised form 20 May 2023; Accepted 31 May 2023
Available online 9 June 2023

learning techniques to collision avoidance systems for USVs (Wang et al., 2019a; Bellemare et al., 2020). Learning through interaction with the environment does not require expert experience or tagged data, the agent truly learn autonomously from experience and can be well suited to a variety of application scenarios. Based on the MDP model, a large amount of data could be used for learning and training, and neural networks could directly map state information to decision-making and control systems. The MDP framework allows implicit associations to be established by data, which not only weakens the impact of errors on decision results in local tasks, but also reduces computational redundancy caused by excessive division of task in traditional control algorithms. Mirowski et al. (2016) performed decision making by deep reinforcement learning to accomplish navigation tasks in static obstacle environments. Do et al. used a reinforcement learning strategy gradient proximal strategy optimization algorithm to generate collision avoidance paths (Chun et al., 2021), which has better performance compared to traditional A * methods. Li et al. made targeted improvements to the DQN method using the artificial potential field method (Li et al., 2021), optimizing the reward settings for DQN. Xie et al. designed collision avoidance algorithms based on the A3C method (Xie et al., 2020), addressing the drawbacks of slow learning speed and non convergence. Wu et al. optimized the performance of the DQN algorithm based on the duel network structure (Wu et al., 2020). Woo et al. used deep reinforcement learning method to design ship collision avoidance decision-making (Woo and Kim, 2020), and combined with the velocity obstacle method, achieved good collision avoidance decision-making results in a grid environment. Zhao et al. divided the surrounding area of the ship into four regions and implemented a multi vessel collision avoidance algorithm based on DQN by considering the nearest obstacle ship (Zhao and Roh, 2019). Zhang et al. divided collision avoidance scenarios into scene layer and autonomous navigation decision layer (Zhang et al., 2019), and designed collision avoidance algorithms using deep reinforcement learning DQN method. Weidong Zhang's team (Yin and Weidong, 2018; Wang et al., 2019b; Xinli et al., 2020a) and Shen et al. (Haiqing and Chen, 2018) accomplished collision avoidance tasks in dynamic environments through discrete action spaces based on Deep Q Network (DQN), respectively. Du et al. (2022) used reinforcement learning methods to accomplish automatic collision avoidance of a single obstacle ship in an unknown environment.

Many reinforcement learning algorithms have been used in the design of autonomous collision avoidance system for USVs, but their research is still in the initial stage, and there are some problems.

(1) The reward function is the feedback signal of the interaction between the agent and the environment. If the USV successfully reaches the destination, a positive reward will be given. If it cannot reach the destination or collides, a negative reward will be given. However, the difference in maritime navigation is that it needs to consider its navigation method, such as avoiding drift and overturning, and also considering COLREGs constraints during collision avoidance. Therefore, the difficulty in designing efficient reward functions is a hindrance to the application of reinforcement learning algorithms in collision avoidance systems.
(2) Ineffective exploration and exploitation methods lead to difficulties in the discovery of critical information in the environment by the agent. The agent cannot learn effective control strategies, and the strategy convergence is too slow, resulting in a waste of computing resources. Excellent exploration methods can not only improve the learning and convergence speed of agents, but also improve the performance of agents in complex tasks. Therefore, the research of exploration methods is another key issue.
(3) The collision avoidance problem of USVs is a reinforcement learning problem of multi-objective optimization. At present, most of the algorithms based on deep reinforcement learning do not consider the random scene of the target (destination and obstacle ship), which leads to the need to train different model parameters for different targets. Therefore, the existing methods are difficult to be directly applied to the collision avoidance system of USVs.

This paper studies the collision avoidance of USVs, proposes a real-time planning and collision avoidance control method, which is based on the deep deterministic policy gradient (DDPG) of continuous action space. Real-time planning and collision avoidance control are realized by designing reasonable state space, action space, reward function, network structure and exploration methods. The main contributions are.

(1) A novel reward function is designed, including daily rewards and settlement rewards. Thus, the collision avoidance problem is transformed into a strategy solution problem of deep reinforcement learning including distance, direction, speed and GOLREGs constraints.
(2) In this paper, a layered sampling exploration method is proposed, which can improve the playback rate of high-value samples, prevent the problem of overfitting, and shorten the training time compared with the random sampling algorithm.
(3) A virtual simulation experiment platform is developed. The position of the destination, the initial direction of the USV and the initial position and direction of the obstacle are set as random to fully test the robustness of the algorithm in different scenarios. Simulation training and testing are conducted on the self-designed platform. The testing environment is divided into four scenarios from easy to difficult, and comparative evaluation is conducted to verify the intelligence of the algorithm.

The subsequent part of this paper is organized as: Section 2 describes the collision avoidance problem and completes the transformation of reinforcement learning control problem. Section 3 introduces the construction of Markov decision process, including the design of state space, action space and reward function. Section 4 is the solution of strategy for real-time planning and collision avoidance strategy based on deep reinforcement learning. Section 5 verifies the effectiveness of the method through four scenario simulations and performs a comparative evaluation. Finally, Section 6 summarizes the whole paper and gives research prospects.

2. Description and mathematical model of collision avoidance for USVs

2.1. Description and assumption of the problem

In complex waters, the operating environment of USV is complex. In order to perform tasks safely and efficiently, the USV needs to have the ability of perception and decision-making (Xinli et al., 2020b). Assumptions.

(1) The USV senses the surrounding environment with its own sensor system, and can obtain the information of random destinations and surrounding obstacles in real time;
(2) The USV can obtain its own running speed $\vec{v} = (v_U, \omega_U)$, v_U, ω_U is linear speed and angular speed respectively, and the maximum speed of the USV is $\vec{v_{max}} = (v_{Umax}, \omega_{Umax})$, $v_U \in [0, v_{Umax}]$, $\omega_U \in [0, \omega_{Umax}]$.

Real-time planning and collision avoidance can be expressed as a sequential decision problem. At the beginning time Ts ($t = 0$), the USV detects and calculates dangerous obstacles and takes them as collision avoidance objects, and the USV enters the collision avoidance situation. At each time t ($t = 0, 1, 2, \ldots$), during collision avoidance, the USV receives the environmental information perceived by sensors, including

the position, running speed and other information of itself, destination and obstacles. USV generates and executes control instructions based on the above information to change its own situation. After each decision is completed, enter the next moment $t = t+1$. The task is completed when the USV reaches the destination without collision. At each moment, state is used as input and control command is used as output. The problem to be solved is to obtain real-time planning and collision avoidance strategies, which need to meet the following requirements.

(1) The USV will not collide with obstacles during navigation.
(2) The collision avoidance process should take as little time as possible.
(3) Collision avoidance actions should meet the COLREGs.
(4) The navigation is stable and consistent with human driving habits.

2.2. Mathematical representation in the process of collision avoidance

2.2.1. Motion model of the USV

In the real marine environment, the motion of a ship can be regarded as the superposition of moving motion and rotating motion (Lu et al., 2020; Cheng et al., 2021). The nonlinear dynamic equations can be expressed as follows:

$$\dot{\eta} = T(\psi)\nu$$

$$M\dot{\nu} = -N(\nu)\nu - g(\nu) + \tau + \tau_w \tag{1}$$

Where $T(\psi)$ is the transformation matrix, $T^{-1}(\psi)T(\psi) = I_{3\times3}$. The position vector $\eta \triangleq [x\ y\ \psi]^T \in \mathbb{R}^3$, and the velocity vector $\nu \triangleq [u\ v\ r]^T \in \mathbb{R}^3$. The surge velocity, sway velocity and the yaw velocity are u, v and r. $M = M^T = M_A + M_{RB} > 0$, which is the inertia matrix. m is the mass of the ship, I_z is the rotary inertia of z-axis. $X_{\dot{u}}$, $Y_{\dot{v}}$, $Y_{\dot{r}}$, $N_{\dot{v}}$, $N_{\dot{r}}$ are coefficients of added mass. $N(\nu)\nu = C(\nu)\nu + D(\nu)$. $C(\nu) = C_A(\nu) + C_{RB}(\nu)$, $N(\nu)\nu$ is the hydrodynamic force. In Table 1, X_u,$X_{|u|u}$, X_{uuu}, Y_r, Y_v, $Y_{|v|v}$, $Y_{|r|v}$, $Y_{|v|r}$, $Y_{|r|r}$, N_r, N_v, $N_{|v|r}$, $N_{|r|v}$, $N_{|v|v}$, $N_{|r|r}$ are the hydrodynamic coefficient. $g(\nu) = [g_u, g_v, g_r]^T \in \mathbb{R}^3$ is the unmodeled dynamics, the value of its parameters is in the literature (Li et al., 2021). $\tau = [\tau_u \quad \tau_v \quad \tau_r]^T \in \mathbb{R}^3$, τ_u is the thrust, τ_r is the moment, $\tau_r = \frac{1}{2}C_L\rho AV^2L$, C_L is the lift coefficient, which is related to the geometry of the rudder blade and changes with rudder angle δ. ρ is the density of water, A is the area of rudder blade. V is the velocity of water at the rudder blade. L is the distance from the pressure center of rudder to the center of the USV. For the underactuated USV, $\tau_v = 0$. $\tau_w = [\tau_{wu} \quad \tau_{wv} \quad \tau_{wr}] \in \mathbb{R}^3$ is the time-varying environment interference vector.

2.2.2. The relationship model of two ships based on COLREGs

In the process of the encounter between the USV and the obstacle ship, the variables that need to be concerned are the relative position of the two ships, speed and direction. Obstacle ship is abbreviated as O. The position of the USV is (x_U, y_U), course angle is φ_U, speed is v_U. The position of O is (x_O, y_O), course angle is φ_O, speed is v_{Obs}. The included angle between v_U and v_O is C_T, and $C_T = \varphi_O - \varphi_U$. When $C_T < 0$, $C_T = C_T + 360°$. The relative orientation of the USV and obstacle ship are θ_U and θ_O respectively. v_R is the relative velocity. They are shown in Fig. 1.

Collision avoidance needs to comply with the International Regulations for Preventing Collisions at Sea (COLREGs) (Deseck, 1983). According to COLREGs, this paper quantifies the encounter situation as head-on, starboard crossing (small-angle), starboard crossing (large--angle), port crossing, overtaking (left) and overtaking (right). For the convenience of viewing, the situation is shown in Fig. 2.

(1) Head-on (χ_1). The obstacle ship is located within (355°,5°] in front of the USV, obstacle ship is in the green area, namely the relative orientation of obstacle ship is $0° \le \theta_O \le 5°$ or

Table 1
Specific parameters of variables.

$$T(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad M_A = \begin{bmatrix} -X_{\dot{u}} & 0 & 0 \\ 0 & -Y_{\dot{v}} & -Y_{\dot{r}} \\ 0 & -N_{\dot{v}} & -N_{\dot{r}} \end{bmatrix} \quad M_{RB} = \begin{bmatrix} m & 0 & 0 \\ 0 & m & mx_g \\ 0 & mx_g & I_Z \end{bmatrix}$$

$$C_A(\nu) = \begin{bmatrix} 0 & 0 & Y_{\dot{v}}v + Y_{\dot{r}}r \\ 0 & 0 & -X_{\dot{u}}u \\ -Y_{\dot{v}}v - Y_{\dot{r}}r & X_{\dot{u}}u & 0 \end{bmatrix} \quad C_{RB}(\nu) = \begin{bmatrix} 0 & 0 & -m(x_g r + v) \\ 0 & 0 & mu \\ m(x_g r + v) & -mu & 0 \end{bmatrix}$$

$$D(\nu) = \begin{bmatrix} -X_u - X_{u|u|}|u| - X_{uuu}u^2 & 0 & -Y_r - Y_{v|r|}|v| - Y_{r|r|}|r| \\ 0 & -Y_v - Y_{v|v|}|v| - Y_{r|v|}|r| & -Y_r - Y_{v|r|}|v| - Y_{r|r|}|r| \\ 0 & -N_v - N_{v|v|}|v| - N_{r|v|}|r| & -N_r - N_{v|r|}|v| - N_{r|r|}|r| \end{bmatrix}$$

$$g(\nu) = \begin{bmatrix} 0.0279uv^2 + 0.0342v^2r, \\ 0.0912u^2v, 0.0156uv^2 + 0.0278uvr^2 \end{bmatrix}^T \quad \tau_w = [2\cos(0.5t)\cos(t) + 0.3\cos(0.5t)\sin(0.5t) - 3, 0.01\sin(0.1t), 0.6\sin(1.1t)\cos(0.3t)]^T$$

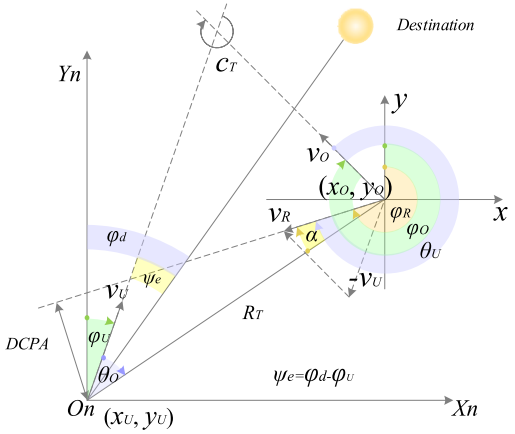


Fig. 1. Relationship model of two ships.

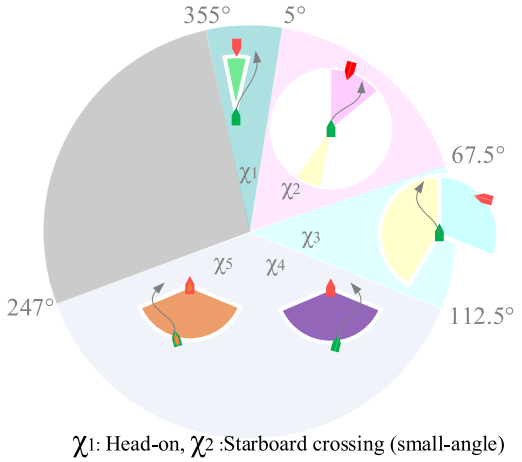


Fig. 2. Quantification of the encounter situation and action for collision avoidance.

$355° < \theta_O \leq 360°$, and $|180° - |\varphi_O - \varphi_U|| \leq 5°$, the USV should turn right.

(2) Starboard crossing (χ_2, χ_3). If the obstacle ship is on the starboard side (5°, 112.5°] of the USV, it is the starboard crossing. When the obstacle ship is on the starboard side (5°, 67.5°] of the USV, obstacle ship is in the pink area, and its speed direction falls in the yellow area, it is starboard crossing (small-angle) (χ_2), that is, the relative orientation of the obstacle ship is $5° < \theta_{Obs} \leq 67.5°$ and $|180° - |\varphi_O - \varphi_U|| \leq 5°$, the USV should turn right. When the obstacle ship is on the starboard side (67.5°, 112.5] of the USV, obstacle ship is in the blue area, and its speed direction falls in the yellow area, it is starboard crossing (large-angle) (χ_3), that is, the relative orientation of the obstacle ship is $67.5° \leq \theta_O < 112.5°$ and $|180° - |\varphi_O - \varphi_U|| > 5°$, the USV should turn left.

(3) Overtaking. One ship is directly behind the other ship (112.5°, 247.5°], and the two ships are sailing in the same direction. If the relative orientation of the obstacle ship is $112.5° < \theta_O \leq 247.5°$ and the velocity component of v_U on the $v_O{}'$ s direction is greater than v_O, i.e. $v_U > v_O * cos\ C_T$,

a) If $\alpha < 90°$, DCPA > 0, obstacle ship is in the purple area, the situation is overtaking (right) $\chi 4$, the USV should turn right.

b) If $270° < \alpha \leq 360°$, DCPA≤0, obstacle ship is in the orange area, the situation is overtaking (left) χ_5, and USV should turn left.

(4) Port crossing. The obstacle ship is on the port side (247.5°,355°] of the USV, the USV is the straight-ship, and no collision avoidance action is required. According to the COLREGs, the USV is a given-way ship in the first three cases, and it should take collision avoidance action. Therefore, this paper only considers the case that the USV is a given-way ship (χ_1, χ_2, χ_3, χ_4, χ_5).

2.2.3. Model of collision avoidance process

Safe navigation should ensure that the distance to closest point of approach (DCPA) is greater than a safe distance. This safety distance can be defined as e_A. The obstacle ship shall not enter this area (the centre of the circle: the barycenter of the USV, radius: e_A) of the USV. The value of e_A is related to the length of the ship and environment. The value is defined as follows.

$$e_A = (e_U + e_O) + 2 \times W + (e_U \times \pi / 135 + e_o \times \pi / 45) \tag{2}$$

e_U is the length of the USV, e_O is the length of the obstacle ship. $2 \times W$ is the error caused by environmental interference such as wind, waves and current to change the ship's position. $e_U \times \pi/135$, $e_O \times \pi/45$ is half of the band width of the track.

In the multi-obstacle environment, the number of obstacles is recorded as O_m. The USV can only take action after detecting and calculating the obstacle ship. Based on this, the three regions of the USV are defined, as shown in Fig. 3. The three regions from the outside to the inside are:

Free area ($\mathscr{F}_i$). When the obstacle ship enters this area of the USV, there is no collision between them, and the USV does not need to take collision avoidance actions, it continues to sail to the destination.

Avoidance area ($\mathscr{A}_i$). When the obstacle ship enters this area, the two will collide, and the USV needs to take collision avoidance action.

Obstacle area ($\mathscr{O}_i$). That is the dangerous area. Once the obstacle enters this area or any collision occurs, the task fails.

Therefore, the three regions of the USV should be $\mathscr{F}_i$, $\mathscr{A}_i$, $\mathscr{O}_i$. $\mathscr{F} = \cup_{i\in O_m}\mathscr{F}_i$, $\mathscr{A} = \cup_{i\in O_m}\mathscr{A}_i$, $\mathscr{O} = \cup_{i\in O_m}\mathscr{O}_i$.

According to ship maneuvering, the calculation formula of free area is as follows:

$$e_F = v_R\left(T_n + \frac{\gamma \times V \times e_A}{v_R}\right) \tag{3}$$

where, T_n is the time required for the ship to turn 90° with full rudder, γ is the coefficient, and the value is between (1.5, 2). V is the visibility coefficient.

Risk is the basis for taking collision avoidance actions, and it needs to be quantified in real time. Therefore, the risk degree is defined as follows.

$$\rho = min(e_{UOi} / e_F) \tag{4}$$

e_{UOi} is the distance between the USV and the obstacle O_i ($i = 1, 2, \ldots, n$). When $\rho > 1$, the free area of USV has no obstacles, and the USV sails towards the destination. When $\rho \leq 1$, there is at least one obstacle in the

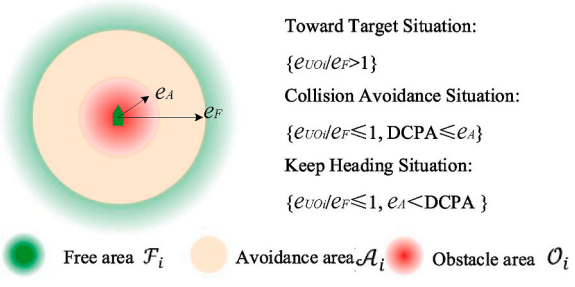


Fig. 3. Quantification of avoidance area and navigational situation for USVs.

free area. When there are multiple obstacles, the danger level of the obstacles is determined through ρ for each obstacle, so as to determine the priority of avoiding obstacles.

The situation of the USV is divided into Toward Target situation, Collision Avoidance situation and Keep Heading Situation. When the USV is in the Toward Target situation, its mission is to sail toward the destination. When the USV is in the Collision Avoidance situation, corresponding collision avoidance actions should be taken according to the requirements of COLREGs. After the USV takes the collision avoidance action, it sometimes enters the avoidance area again, to avoid that, the course angle of the USV should remain unchanged to ensure its safe navigation until it reaches the free area, we define it as Keep Heading Situation. The identification of each situation is shown in Fig. 3.

The USV collects and calculates the relevant information through the self-carried sensing system, including e_A, e_F, ρ, DCPA, TCPA, etc. Which are used to determine whether the USV has arrived at the destination. If it arrives, the process is over, otherwise, the program enters the next step. The collision avoidance process is developed as shown in Fig. 4.

3. Markov decision process

In order to solve the optimal strategy, this paper constructs the collision avoidance problem as a Markov decision process, which includes S, A, R (Mnih et al., 2015). S is the state space of the environment, which are all possible states in the operating environment. A is the action space, which is the control command for the USV. R is the reward function, which is used to evaluate the state transition process. In the established model, the USV determines the control instructions according to the state of the environment. The state space, action space and reward function are established as follows.

3.1. State space

The status information s_t at time t includes: own data of the USV and data of the surrounding environment. These data are generated by real-time detection of the sensor system carried by the USV.

The data of the USV include linear velocity $s_1 = v_U$, angular velocity $s_2 = \omega_U$, course angle $s_3 = \psi_U$.

The environment data contains the relationship information between the USV and the obstacle. That is, the distance between them $s_4 = e_{UOi}$, the azimuth of obstacle ship relative to the USV $s_5 = \theta_O$, the azimuth of the USV relative to obstacle ship $s_6 = \theta_U$, the value of the velocity component of v_{USV} on the $v_{Obs}{}'$ s direction $s_7 = RO$, the encounter situation $s_8 = \chi_i$.

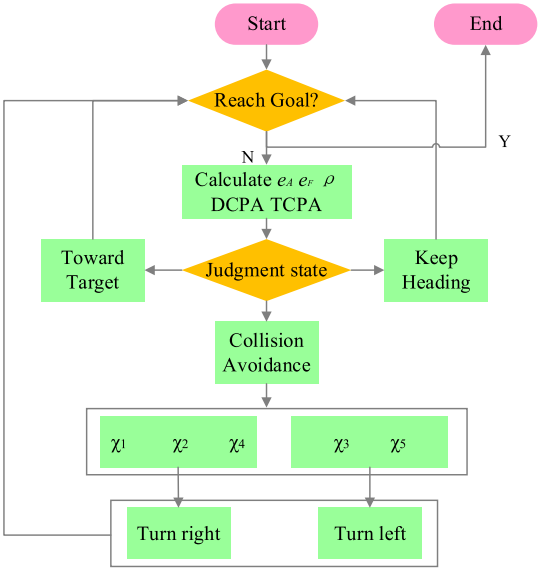


Fig. 4. Flow chart of real-time planning and collision avoidance for the USV.

The environmental data also contains the relationship information between the USV and the destination. It includes the distance from USV to destination $s_9 = e_{dist}$, course angle error $s_{10} = \psi_e$ (as shown in Fig. 1), the flag bit of destination $s_{11} = GO$, that is, GO is 1 when the USV reaches the destination, otherwise it is 0.

Therefore, the environment state is composed of the above 11 tuples $s_1 \times s_2 \ldots s_{11}$.

In order to improve the effectiveness of decision-making, dynamic information in the environment needs to be utilized. Therefore, the consecutive states sequence $\vec{s_t} = (s_{t+(-N+1)}, s_{t+(-N+2)}, \ldots, s_t)$ is used to infer the speed, movement trend and other information of obstacles. The more states in the sequence, the more adequate environmental information is included, which can improve the accuracy of decision making, but too much information will affect the speed of calculation and reduce the timeliness of decision making. Therefore, $N = 3$ is taken in this paper, namely $\vec{s_t} = (s_{t-2}, s_{t-1}, s_t)$ is the state sequence. At the beginning of collision avoidance $t = 0$, the sequence $\vec{s_0} = (s_0, s_0, s_0)$.

3.2. Action space

Action space is the operation that the ship can take at a certain moment. The thrust and rudder angle of the USV are used as control instructions, that is $a_t = [\tau_u, \delta]$, τ_u is the thrust, $\tau_u \in [-50N, 50N]$. δ is the rudder angle, $\delta \in [-35°, 35°]$. Therefore, the actions can control the USV acceleration, deceleration, turn left and turn right.

3.3. Reward function

Path exploration is a complex process, in order to make the agent reach the set destination efficiently, the control strategy should be taken into account when designing the reward function. The task of this paper is to control the USV so that it can sail faster from the initial position to the target position. There are two signs that indicate the end of the training: one is when the USV enters the target position area. The other one, if the total number of explored steps exceeds the allowable value, or its actions violate COLREGs, the training of this episode will end regardless of whether the USV reaches the target position.

In the training process of reinforcement learning, the problem of sparse reward (Mathis et al., 2018; Lillicrap et al., 2016) has been very difficult. Sparse reward will lead to slow convergence of training algorithm, and even non-convergence. Therefore a novel reward function is designed in this paper, the design of reward function considers two parts, the daily reward in the training process and the settlement reward at the end of the episode.

3.3.1. Daily rewards during training

Daily reward is the reward given to USV for each action taken in an episode. The USV's task is to sail from the initial point to the random destination. The ideal process is to adjust the course angle of the USV to the target course angle, and then sail quickly to the destination, while avoiding the dynamic obstacle ship with COLREGs. The following aspects should be considered in the formulation of the reward function.

(1) Directional reward. The whole task needs to be viewed with human thinking, the first thing to consider is the direction of navigation. If the course is wrong, it is impossible to reach the destination no matter how the USV move, so it is necessary to set a directional reward. The greater the deviation between the real-time course angle and the target course angle (as shown in Fig. 1), the greater the negative reward should be. When the deviation between the real-time course angle and the target course angle is within the allowable range, the reward value is zero.

$$R_{angle}=\begin{cases}-\dfrac{|\psi_e|}{\pi}, if\ |\psi_e|\geq|\psi|\\ 0, otherwise\end{cases} \tag{5}$$

The $|\psi_e|$ represents the absolute value of the deviation between the real-time course angle and the target course angle, normalized to get the final direction reward. ψ is a small value.

(2) Stop reward. When the USV stops moving during training, a negative reward is given to encourage the USV to move.

$$R_v=\begin{cases}0 & v_U\geq 0.05\\ -r_v & otherwise\end{cases} \tag{6}$$

where v_U represents the real-time speed of the USV. A fixed negative reward r_v should be given when the speed is lower than 0.05 m/s.

(3) Steering angular velocity reward. Too high steering speed is harmful, for example, the USV overturns, and it is also difficult for the USV to stop at the correct course angle, which is not conducive to the control of the USV. Therefore a negative reward of the steering angular velocity requires to be added. When the maximum acceptable steering angular velocity is exceeded, a negative reward for overspeed should be given.

$$R_\omega=\begin{cases}-r_\omega & \omega_U\geq 2^\circ/s\\ 0 & \omega_U<2^\circ/s\end{cases} \tag{7}$$

ω_U represents the steering angular velocity, a fixed negative reward should be given when the steering angular velocity exceeds 2°/s.

(4) Deceleration reward. To ensure that the USV slows down near the destination and reaches the destination accurately and smoothly, the deceleration reward is set. As the USV nears its destination, it begins to slow down.

$$R_{slow}=\begin{cases}-\dfrac{v_U}{v_{Umax}}, & e_{dist}\leq e_{slow}\\ 0, & e_{dist}>e_{slow}\end{cases} \tag{8}$$

when the distance between the USV and the destination e_{dist} is e_{slow}, the USV starts to slow down so that it can stop at the destination accurately.

(5) Collision avoidance reward. Safety is regarded as the core element of collision avoidance. The distance between the USV and the surrounding ship is the most intuitive and important measurement index to avoid collision. Therefore, the free area e_F is used as the design standard in the collision avoidance reward function. Collision avoidance rewards are designed as follows.

$$R_{ca}=-\frac{1}{\sqrt{2\pi}\sigma}e^{-\left(1-\frac{e_{UO}}{e_F}\right)^2} \tag{9}$$

σ is the scale parameter and e_{UO} is the distance between USV and the obstacle ship.

(6) Sailing mode reward. It makes more sense for the USV to sail in the surge mode during navigation. This reward will ensure that the USV stays on the correct course and avoids drifting (sailing in the sway mode). When the USV sails in the sway mode, the corresponding punishment shall be given. The established sailing mode reward is as follows.

$$R_{mode}=\begin{cases}-r_{mode}, & if\ |v_{sway}|\geq \mathscr{M}\\ 0, otherwise\end{cases} \tag{10}$$

r_{mode} and $\mathscr{M}$ are positive constants and v_{sway} is the speed of sway.

The final result of daily reward is:

$$R_s=\xi_{angle}R_{angle}+\xi_vR_v+\xi_{\omega_U}R_{\omega_U}+\xi_{slow}R_{slow}+\xi_{ca}R_{ca}+\xi_{mode}R_{mode} \tag{11}$$

where ξ_{angle}, ξ_v, ξ_{ω_U}, ξ_{slow}, ξ_{ca}, ξ_{mode} is the weight vector.

3.3.2. Settlement reward

Settlement reward is the reward given at the end of a training episode. There are two conditions for the end of the episode (as mentioned above), two types of settlement rewards should be defined.

(1) The settlement reward for completing the task. The task is completed when the USV enters the circle (with the destination as the center, the allowable ε_{end} is the radius). The reward function is as follows:

$$R_{dist}=\begin{cases}10 & e_{dist}\leq\varepsilon_{end}\ and\ s\leq s_{max}\\ -\dfrac{e_{dist}}{D} & otherwise\end{cases} \tag{12}$$

e_{dist} is the distance between USV and the destination at time t, s is the number of steps, s_{max} is the maximum number of steps allowed in this episode. D is the distance between USV and the destination at initial moment. When the distance between USV and destination is less than ε_{end} and the number of steps is less than the allowable maximum number of steps, the task is completed, the episode ends and the maximum reward value is obtained. Here the ε_{end} is 5m. When the USV has not reached the destination in this episode, or has consumed the maximum steps but has not reached the destination, the corresponding punishment will be given. In addition, the negative reward setting by distance can better promote the USV to approach the destination.

(2) GOLREGs Awards. In the training process, when USV evades the dynamic obstacle ship, its action violates the GOLREGs, at this time, the episode ends and the mission is judged to be failed. The reward function is as follows:

$$R_{COLREGs}=\begin{cases}0, if\ action\ complies\ with\ \ COLREGs\\ -r_{COLREGs}, else\end{cases} \tag{13}$$

$r_{COLREGs}$ is a positive constant.

The settlement reward function is as follows:

$$R_c=\xi_{dist}R_{dist}+\xi_{COLREGs}R_{COLREGs} \tag{14}$$

ξ_{dist}, $\xi_{COLREGs}$ are weight vectors.

The reward function of the whole reinforcement learning task is as follows.

$$R=R_s+R_c \tag{15}$$

4. Solution of the strategy for real-time planning and collision avoidance based on reinforcement learning

The problem of real-time planning and collision avoidance ultimately needs to solve the optimal collision avoidance strategy. In the established model, the state $\vec{s_t}$ belongs to high-dimensional input, and the output action a_t is continuous. Therefore, the deep deterministic policy gradient algorithm (Wu et al., 2020) in reinforcement learning is adopted to train the deterministic strategy and determine the optimal action at each step.

4.1. Network structure

The DDPG algorithm includes two neural networks: actor and critic (Silver et al., 2016). Actor network taking state sequence $\vec{s_t}$ as input and action a_t as the output, controls the speed and steering of the USV, and collision avoidance strategy is $\mu_\theta(a_t|\vec{s_t})$. Critic network taking state sequence $\vec{s_t}$ and action a_t as input, the value $Q(\vec{s_t},a_t)$ of both is output,

and which can evaluate the decision-making results of actor network so that it can be continuously optimized.

The state at the decision time and the first two moments is selected as the state sequence, and the multi-frame sequence data is used as the input of the decision. In order to solve the decision problem of multi-frame sequence input, the framework is designed which is shown in Figs. 5 and 6 for actor and critic networks. Fig. 5 shows the ctor network. The state sequence $\vec{s}_t = (s_{t-2}, s_{t-1}, s_t)$ connects the fully connected network by stacking them into one-dimensional vectors in order. The intermediate layer and the final output layer are all fully connected networks. The specific parameters of the actor network are shown in Table 2.

Fig. 6 shows the critic network. Similar to the actor network, critical network also uses the method of data stacking to process the input of multi-frame sequence. Since the critic network evaluates the state-action $(\vec{s}_t, a)$, the action processing layer is also added. The specific parameters of the critic network are shown in Table 2. The activation functions used in the two networks include the arctangent function tanh(x) and relu(x) = max (0, x).

4.2. Exploration mechanism of layered sampling

The experience playback mechanism breaks the correlation between samples and solves the problem of difficult convergence in network training. However, the experience playback mechanism only replays the samples at the same frequency (Vinyals et al., 2019; Morris et al., 2006), the importance of the sample is not considered which makes the training process of the network very long, and the agent cannot learn effective control strategies. To solve this problem, a layered sampling exploration mechanism is proposed to avoid updating the TD error of all samples before each training. By improving the playback frequency of samples with large TD error value, the algorithm learning is accelerated.

TD error (temporal difference error δ) is used to indicate the importance of the sample. The greater the absolute value of TD error, the greater the range of network weight update. In this case, experience with higher TD error has more important learning value.

The TD-error between the current state action value function $Q^{\mu}(s_t, a_t)$ and the pre estimated state action value function $r_{t+1} + \gamma Q^{\mu}(s_{t+1}, a_{t+1})$ is calculated

$$\delta_t = r_{t+1} + \gamma Q^{\mu}(s_{t+1}, a_{t+1}) - Q^{\mu}(s_t, a_t) \tag{16}$$

where $Q^{\mu}(s_t, a_t)$ is the state action value function, that is, when the agent follows the policy, the expectation of return obtained by taking action a_t in the current state s_t:

$$Q^{\mu}(s, a) = E_{\mu}\left[r_{t+1} + \gamma Q^{\mu}(s_{t+1}, a_{t+1})|_{s_t=s, a_t=a}\right] \tag{17}$$

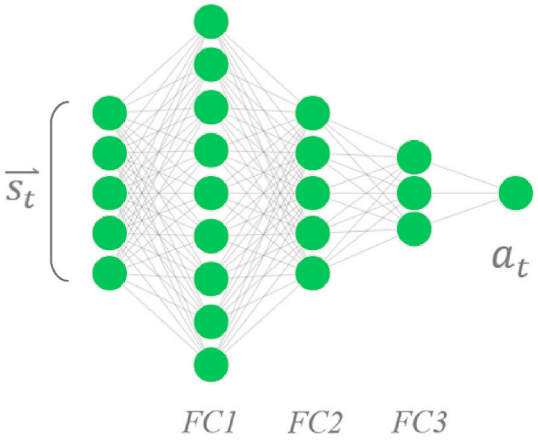


Fig. 5. The network structure of actor network.

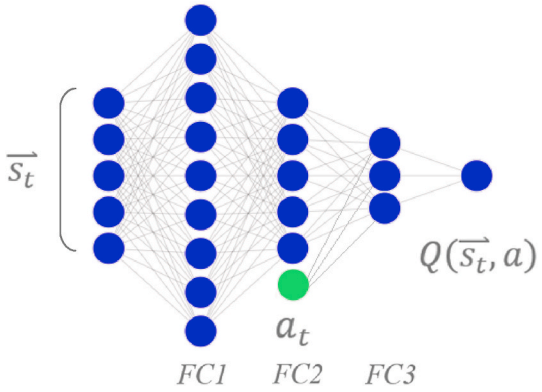


Fig. 6. The network structure of critic network.

(1) The newly generated samples from the environment are added to the small batch sample pool S, it is used for iterative updating, which can ensure that the samples are trained at least once, this effectively reduces the over-fitting problem.
(2) Randomly selected batches of samples (from experience pool B) form sample pool G. The samples in G are stratified into E-1 layers, 1≤i≤E-1. i is any layer, E is the number of batch samples.
(3) According to the sampling probability of the samples in each layer, one sample is extracted from each layer and placed into the sample pool S. This allows the final sample to be taken without duplication and ensures the diversity of samples. The sampling probability of the sample P_{ij} is

$$\omega_{ij} = |\delta_{ij}| + \beta \tag{18}$$

$$P_{ij} = \frac{e^{\omega_{ij}^{\lambda}}}{\sum_{g} e^{\omega_{ij}^{\lambda}}} \tag{19}$$

where $|\delta_{ij}|$ is the absolute value of TD error, β is a minimal positive number. It can prevent the sampling probability from approaching 0 when the TD error approaches 0. The exponent λ is used to control the degree of priority, and its value range is [0,1]. The exponential form can make the difference of sampling probability larger, and the agent can also find the optimal strategy faster.

Samples are sampled from each layer according to the sampling probability. Samples with large $|\delta_{ij}|$ are played back with a greater probability, and samples with small $|\delta_{ij}|$ are played back with a small probability. This mechanism avoids updating the TD error of all samples in the experience pool, and only needs to update the TD error of the samples in the G sample pool. This reduces the amount of calculation, time and complexity. In this paper, the above mechanism is applied to the network, and an LSDDPG algorithm based on the layered sampling is constructed, and the overall structure and pseudocode of the algorithm are as follows (see Fig. 7).

Initialization: The capacity of experience pool B is N. The size of minibatch is E. The size of training experience pool S is E. Number of samples in each layer is g. The environment is initialized and the experience buffer is set to zero. The parameters of actor network $\mu(s;\theta^{\mu})$ and critic network $Q(s,a;\theta^{Q})$ is initialized randomly, and the actor target network and critic target network are given corresponding parameters, i.e $\theta^{\mu'} \leftarrow \theta^{\mu}$, $\theta^{Q'} \leftarrow \theta^{Q}$

1 For episode = 1:M
2 The initial state s_1 is obtained
3 For step = 1: T
4 According to the policy and the interference, the input is s_t, the output is a_t, $a_t \sim N(\mu(s_t|\theta^{\mu}), var)$

(continued on next page)

Table 2
Parameters of the network structure.

Actor network parameters				Critic network parameters			
Network structure		Dimension	Activation function	Network structure		Dimension	Activation function
Input $\vec{o}_t$		11	–	Input $\vec{o}_t$	$\vec{o}_t$	11	–
					a_t	2	–
Fully connected layer 1	FC1	128	relu	Fully connected layer 1	FC 1	128	relu
Fully connected layer 2	FC2	256	relu	Fully connected layer 2	FC 2.1	256	relu
					FC 2.2	128	relu
Fully connected layer 3	FC3	128	relu	Fully connected layer 3	FC 3	128	relu
Output	a_t	2	relu	Output	$Q(\vec{o}_t,a)$	2	tanh

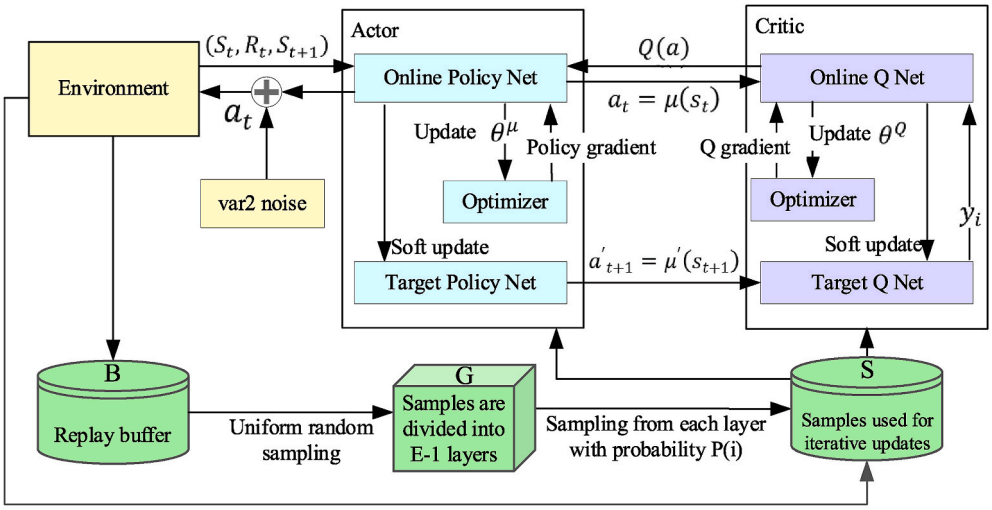


Fig. 7. Flow chart of LSDDPG algorithm.

(*continued*)

5 The agent makes the action a_t and gets the return r_t and the subsequent state s_{t+1}
6 (s_t, a_t, r_t, s_{t+1}) is stored in experience pool B, S as a sample
7 Randomly sample (E-1)*g samples from B, and they are evenly divided into E-1 parts on average
8 For i = 1: E-1
9 For j = 1: g
10 Calculate TD error of sample in Gi : $\delta_t = r_{t+1} + \gamma Q^{\mu}(s_{t+1}, a_{t+1}) - Q^{\mu}(s_t, a_t)$
11 Calculate the weight of samples in Gi $\omega_{ij} = |\delta_{ij}| + \beta$
12 Calculate the sampling probability of samples in Gi: $P_{ij} = \frac{e^{\omega_{ij}^i}}{\sum_g e^{\omega_{ij}^i}}$
13 According to the sampling probability P_{ij}, select a sample from Gi and put it into S
14 End for
15 End for
16 Calculate $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu})|\theta^{Q'})$ with the sample in S
17 Update the Q network by minimizing the loss function: $L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$
18 Update the policy network through the policy gradient: $\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_i \nabla_a Q(s, a|\theta^Q)\Big|_{s=s_t, a=\mu(s_t)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)|_{s_t}$
19 Update target network : $\begin{cases} \theta^{\mu'} \leftarrow \tau\theta^{\mu} + (1-\tau)\theta^{\mu'} \\ \theta^{Q'} \leftarrow \tau\theta^{Q} + (1-\tau)\theta^{Q'} \end{cases}$
20 End for
21 End for

5. Simulation

In this paper, a real-time planning and collision avoidance training platform is built in the Python environment, and simulation scenarios of different difficulty are designed. In each episode of simulation, a random disturbance is added to change the initial direction of USV. The initial positions of obstacles are randomly generated to fully test the robustness of the algorithm in different scenarios. Different obstacle ships in the picture use different colors, from the beginning to the end, and the color of their track changes from light to deep. The setting and parameters of algorithm are shown in Tables 3 and 4.

5.1. Scenario 1: Real-time planning (No obstacle ship)

This USV is put into a simple environment for real-time planning learning in this section. In the environment, the initial course of the USV is random and the speed is 3.5 m/s. It can be seen from Fig. 8(a).(d) that the USV cannot grasp the environmental information at the beginning of learning, cannot take the most effective actions, and the path is messy, as shown in figure (a). Gradually, it tries to sail towards its destination, but there are some improper movements, as shown in figure (b). With full interaction with the environment, the USV's movements stabilize and successfully reach its destination, as shown in figure (c). With continuous training and the update of network parameters, the accumulated learning experience of the decision-making system can effectively plan the path, and the planned path is more and more excellent, as shown in figure (d).

To show the application performance of the proposed method, the training time and the average cumulative reward value are used as evaluation indicators. During the experiment, to avoid contingency, the training process is repeated 100 times, and the average mean is figured out. The abscissa represents the number of episodes, and the ordinate

Table 3
Parameters and performance of algorithm.

Parameter	Content	Value	Parameter	Content	Value
α^{μ}	Learning rate of actor network	1×10^{-4}	T	Maximum time steps	500
α^{Q}	Learning rate of critic network	1×10^{-3}	v_{Umax}	Maximum linear speed of USV	3.5 m/s
γ	Discount factor	0.99	a_{max}	Maximum linear acceleration of USV	0.4 m/s^2
τ	Update of target network	0.01	e_U	Length of USV	1.8m
B	Number of samples	10000	e_O	Length of Obs	1.8m
b	Size batch	32	v_{Omax}	Maximum speed of Obs	3.5 m/s
E	Number of layers	16	ω_{max}	Maximum angular velocity of USV	0.2 rad/s
δ	Range of rudder angle	35°	α_{max}	Maximum angular acceleration of USV	0.05 rad/s^2

Table 4
Setting of simulation environment with different difficulties.

Difficulty	Number of obstacles	Purpose of the test
1	0	Real-time planning
2	1	Real-time planning and collision avoidance according to COLREGs
3	multiple	Real-time planning and collision avoidance according to COLREGs

represents the time steps and average cumulative reward.

From Fig. 9, it can be seen that the first 300 episodes, the time steps used by both algorithms are basically the same. However, after 300 episodes, LSDDPG takes significantly fewer time steps than DDPG, and the LSDDPG algorithm tends to be stable quickly.

From Fig. 10, it can be found that in the first 300 episodes, the reward value of DDPG and LSDDPG algorithms are unstable, but after 300 episodes, the average cumulative reward value obtained by LSDDPG algorithm is higher than that of DDPG algorithm, and the LSDDPG algorithm tends to be stable quickly.

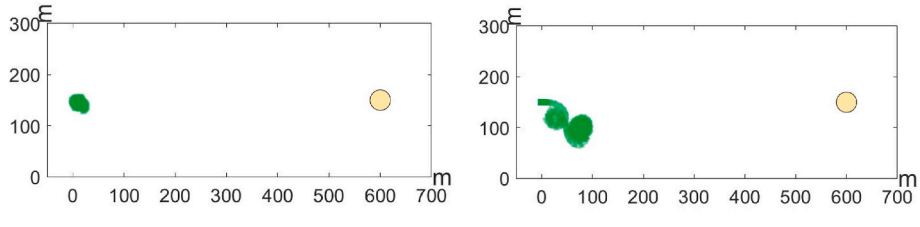
The above research fully confirms that the proposed LSDDPG algorithm has a good real-time planning effect.

5.2. Scenario 2: Single collision avoidance test (real-time planning and single collision avoidance according to COLREGs)

To verify the effect of the proposed algorithm (real-time planning and single collision avoidance according to COLREGs), the simulation is set to five cases, and the relevant parameters are shown in Table 4. The relationship between position, angle and speed can be known through kinematics formula, the thrust and rudder angle can be solved through the dynamic formula. The above parameters can be solved through the formula (1) in the second part. Fig. 11 shows the tracks of USV and obstacle ship in the case of COLREGs. Fig. 12 shows the speed and rudder angle of the USV during collision avoidance. Where, the solid line represents the track of the USV or obstacle ship respectively, the USV is green, and the obstacle ship is red. At the beginning, the obstacle ship is far away from the USV. The USV is in the Toward Target situation, which is sailing towards the destination. With the passage of time, the obstacle ship enters the free area of the USV and meets the conditions $\rho \leq 1$, DCPA$\leq e_A$, they belong to the head-on situation, and the USV turns right to avoid collision. Then the condition $\rho \leq 1$, $e_A <$ DCPA is met and the USV is in the Keep Heading Situation. After the collision avoidance is completed, the obstacle ship keeps away from the USV. When the condition $\rho > 1$ is met, the USV is in the Toward Target situation, and the USV continues to sail towards the destination and successfully reaches the destination. Figs 13–20 shows other situations of COLREGs. Each step is 1 second.

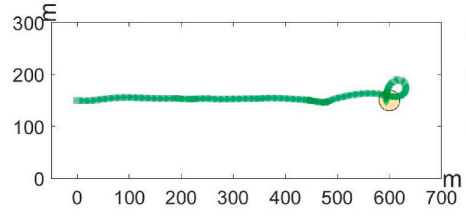
In order to evaluate the effect of the algorithm fairly and accurately, the success rate is selected as the evaluation index of the experiment. Success rate means that if the USV complies with the COLREGs and finally reaches the destination, the navigation will be deemed as successful, otherwise it will be deemed as failure. Compare the success rate of 5000 episodes. The success rates of five situations for the COLREGs are shown in Fig. 21.

It can be seen that in the first 500 episodes, the success rate of collision avoidance in five cases is not high. This is because the USV constantly explores the environment, learns collision avoidance strategies in the environment, and immediately carries out the next training after a collision. In the 1000–4000 episodes, after a period of learning and trial and error, the success rate of the algorithm increased rapidly. In the 5000 episodes, the success rates of several cases in the convergence

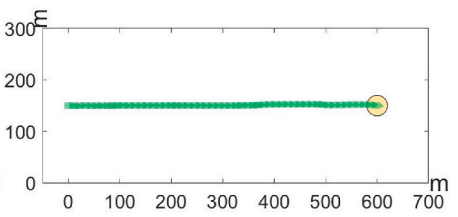


(a) Training effect after 20 episodes.

(b) Training effect after 200 episodes.

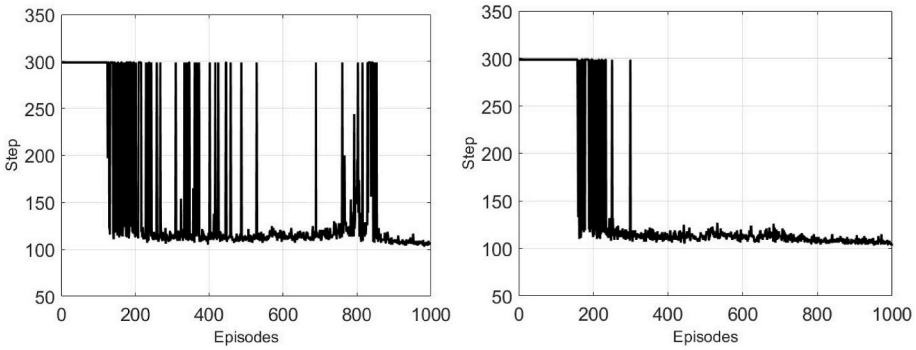


(c) Training effect after 500 episodes.



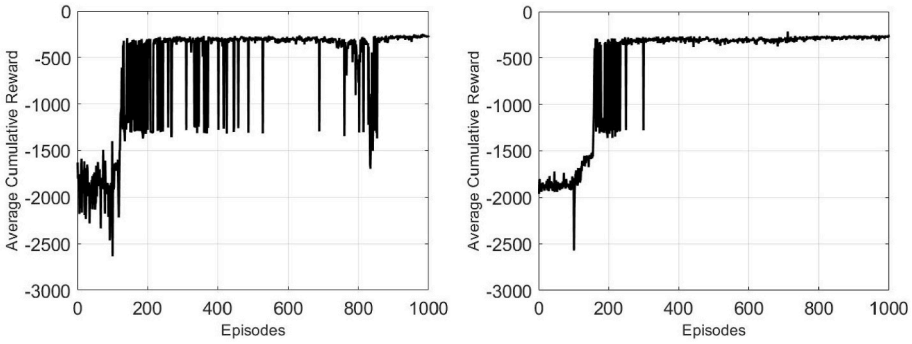
(d) Training effect after 1000 episodes.

Fig. 8. Learning process of real-time planning.



(a) DDPG algorithm. (b) LSDDPG algorithm.

Fig. 9. The curve of training time.



(a) DDPG algorithm. (b) LSDDPG algorithm.

Fig. 10. The curve of the average cumulative reward value.

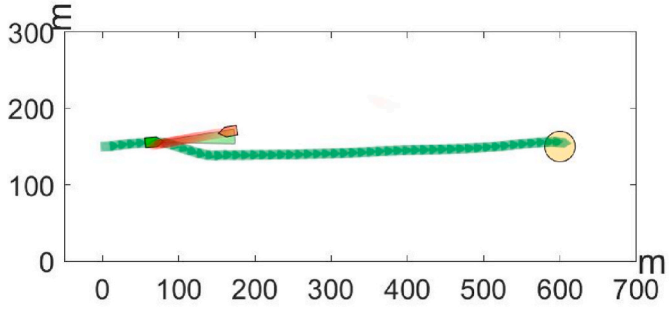


Fig. 11. Head-on.

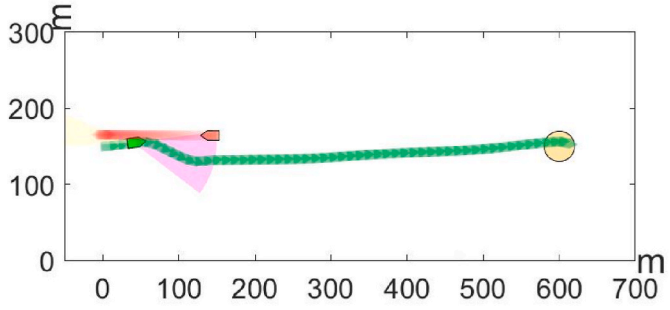


Fig. 13. Starboard crossing (small-angle).

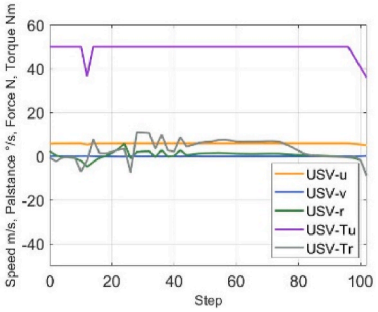


Fig. 12. Speed and rudder angle of the USV.

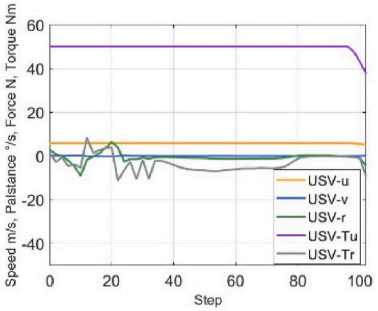


Fig. 14. Speed and rudder angle of the USV.

state tend to be stable, reaching about 96%. It can be seen that the LSDDPG algorithm can control the USV to plan the path reasonably and avoid real-time obstacle ships in accordance with the requirements of COLREGs.

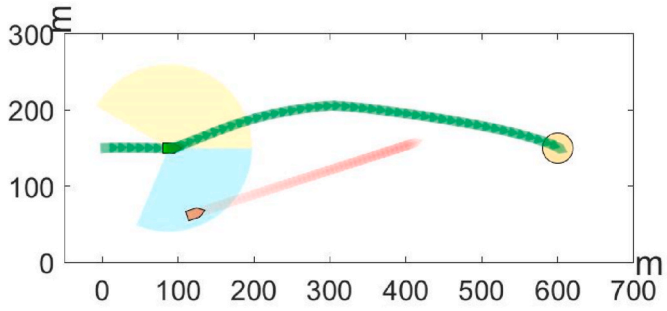


Fig. 15. Starboard crossing (large-angle).

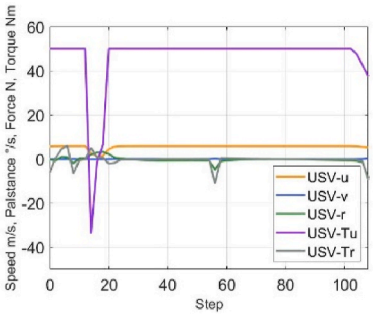


Fig. 16. Speed and rudder angle of the USV.

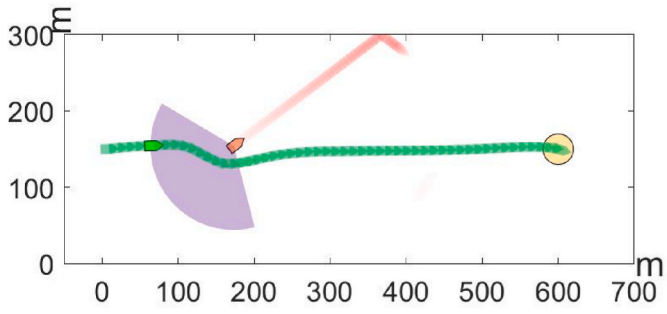


Fig. 17. Overtaking (right).

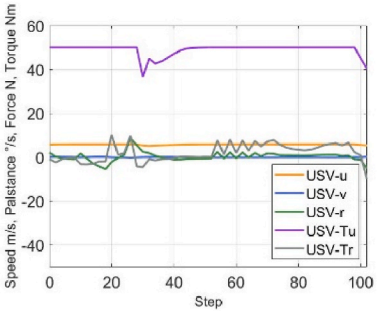


Fig. 18. Speed and rudder angle of the USV.

5.3. Scenario 3: Multiple collision avoidance test (three obstacle ships)

The above simulation only verifies the autonomous collision avoidance capability of the USV when dealing with a single obstacle ship. This section carries out real-time planning and collision avoidance simulation in the case of continuous encounter of multiple ships. The following are the results of offline testing. In each episode of training, we will

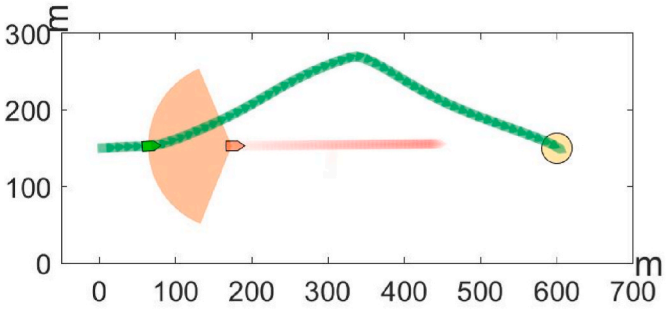


Fig. 19. Overtaking (left).

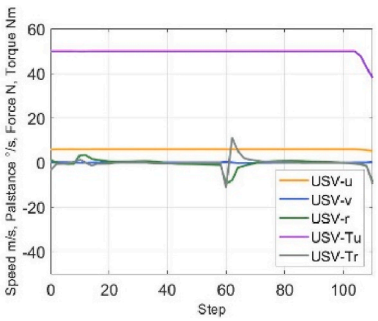


Fig. 20. Speed and rudder angle of the USV.

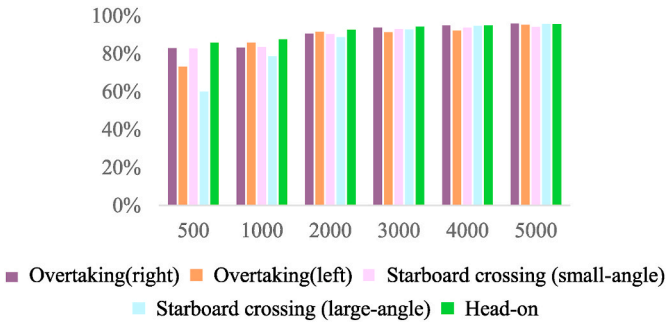
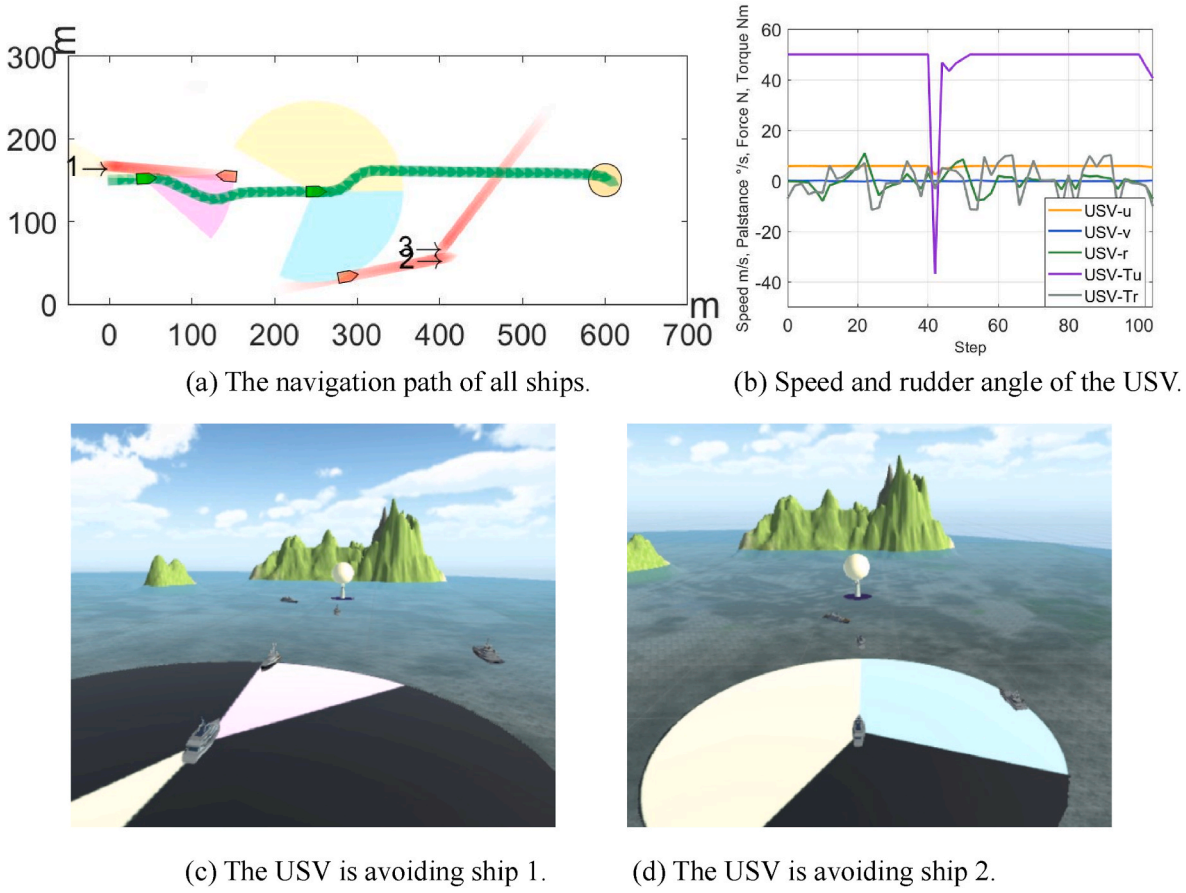


Fig. 21. Success rate of five cases of the COLREGs.

continuously generate obstacle ships that form various encounter situations with the USV, which increases the probability of various encounter situations occurring.

In this simulation, the initial speed of the USV is 5.6 m/s, the initial course is due east, and the initial position is (5.6, 150.2). The speed of obstacle ship 1, 2 and 3 are 2.2 m/s, 1.9 m/s and 2.0 m/s respectively, and the course is 179.6° (the positive direction of the x-axis is the reference, and the counterclockwise direction is positive), 10.8°, 232.5°, and the initial position is (147.7 km, 154.9 km), (205.0 km, 17.7 km), (531.5 km, 235.1 km). The track of the USV and three obstacle ships is shown in Fig. 22.

In figure (a), after the USV detects ship 1, the distance between the two ships is getting closer and closer, meeting the requirements of $\rho \leq 1$, DCPA$\leq e_A$, the USV is in the Collision avoidance situation in the 7th step, and the two ships are in the situation of starboard crossing (small-angle). According to the COLREGs, the USV turns right, and then meets the condition $\rho > 1$. The USV is in the Keep Heading Situation in the 12th step. As the distance between the two is getting farther and farther, the USV enters the Toward Target situation in the 24th step and continues to carry out the task of sailing to the destination. During the navigation, the USV detects ship 2 in the 43rd step, and the encounter situation is overtaking (left), and $\rho \leq 1$, DCPA$\leq e_A$. In order to avoid



(a) The navigation path of all ships.

(b) Speed and rudder angle of the USV.

(c) The USV is avoiding ship 1.

(d) The USV is avoiding ship 2.

Fig. 22. Collision avoidance simulation for three obstacle ships.

collision, the USV takes a left turn to avoid collision. Then USV approaches ship 3. In the process of approaching, the condition $\rho \leq 1$ is met, but DCPA > e_A. It is determined that there is no danger of collision between them, so they continue to sail along the desired path, finally the USV arrives at the destination. Python and Unity 3D software are used for joint simulation figures (c) and (d) are 3D simulation images of the USV during avoiding obstacle ship 1 and 2.

During the testing process, we selected another scenario where the number of obstacle ships is five. The speeds of obstacle ships 1, 2, 3, 4 and 5 are 2.9 m/s, 2.8 m/s, 3.3 m/s, 0.1 m/s, 1.0 m/s, and their courses are 187.7°, 332.3°, 4.3°, 164.5°, 155.3°. The initial positions are (301.8 km, 139.8 km), (100.3 km, 285.8 km), (265.4 km, 119.3 km), (168.8 km, 142.1 km), (389.7 km, 279.1 km). The navigation path of the USV and five obstacle ships is shown in Fig. 23.

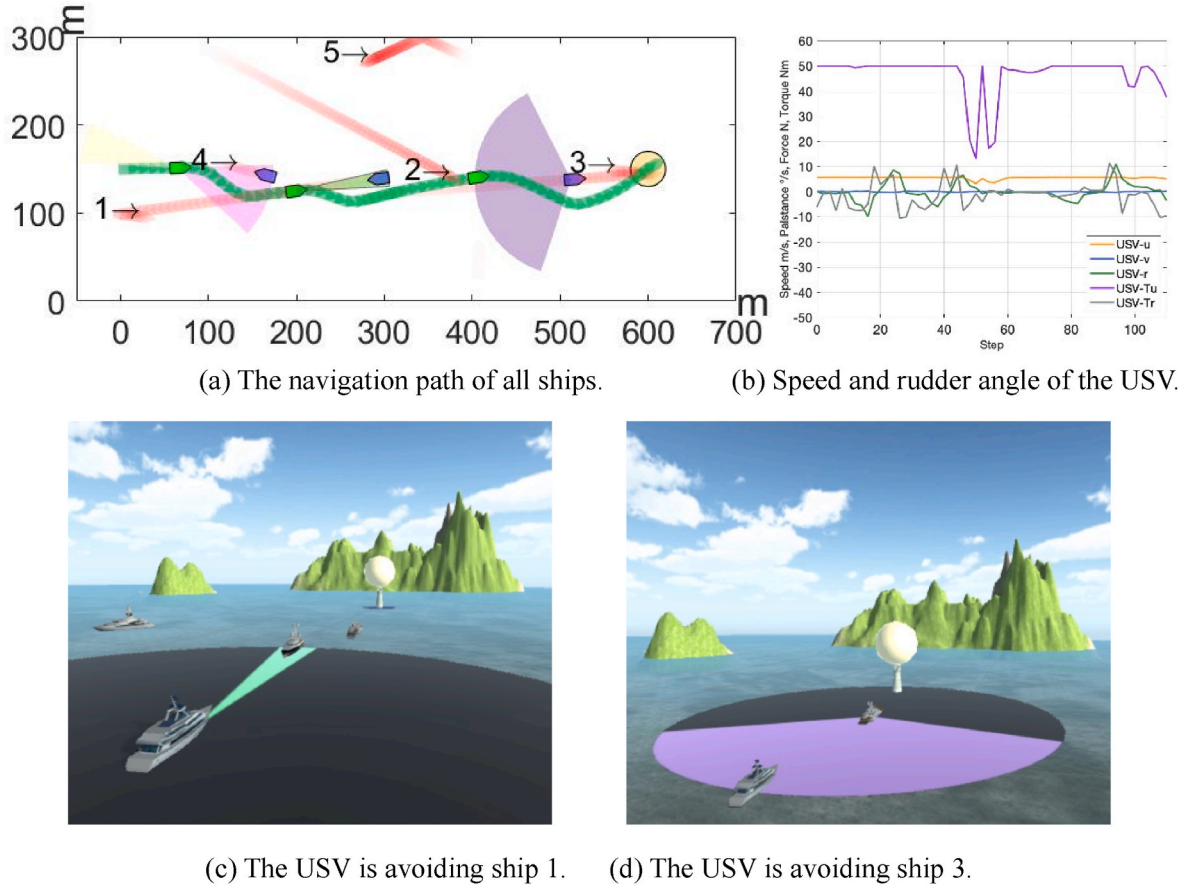
In figure (a), after discovering the ship 4, the USV is in the Collision avoidance situation with $\rho \leq 1$, DCPA$\leq e_A$ in the 11th step. The two ships belong to the starboard crossing (small-angle) encounter situation. According to the COLREGs, the USV first turn right, and then meet the conditions $\rho > 1$, the USV is in the Keep Heading Situation in the 18th step. As the distance between the two becomes farther and farther, satisfying the condition $\rho > 1$, it enters the Toward Target situation, and the USV continues to perform the task of sailing to the destination. Then the USV approaches ship 1. In the process of approaching, the two ships belong to the head-on encounter situation in the 35th step. According to the COLREGs, the USV turn right, and enters the Keep Heading Situation in the 41st step. Then the USV is in the Toward Target situation in the 47th step, continues to sail to the destination. Then the USV approaches ship 2, the condition $\rho \leq 1$ is satisfied, but DCPA $> e_A$. It is determined that there is no risk of collision between the two, so it continues to sail along the desired path. Since the USV and ship 3 belong to overtaking (right), and $\rho \leq 1$, DCPA$\leq e_A$, in order to avoid collision, the USV turns right in the 75th step. The USV discoveres ship 5, but they do not meet the collision condition, and the USV does not need to take collision avoidance action. Finally, the USV escapes several obstacle ships and successfully arrives at the destination. Figures (c) and (d) are 3D simulation images of the USV during avoiding obstacle ship 1 and 3.

Through the above collision avoidance simulation with multiple obstacle ships, it is fully verified that the designed collision avoidance algorithm has good collision avoidance effect in multiple collision avoidance tests. Under the guidance of LSDDPG algorithm, the USV has the ability of real-time planning and collision avoidance decision-making and behavior execution, so it can be concluded that the USV has obtained certain intelligent navigation ability.

6. Conclusion

Compared with the traditional analytical algorithm, the deep reinforcement learning method does not depend on the artificially set rules, and directly adds the collision avoidance constraint to the reward function, by learning the optimization strategy through trial and error, it has stronger ability to cope with complex environment and high-dimensional input. In this paper, the intelligent avoidance of dynamic obstacles by the USV in complex environments is taken as the research background, in the absence of global map information, the real-time planning and collision avoidance simulation system of USVs and the deep reinforcement learning control algorithm are studied by using the sensing device carried by USV itself to sense environmental information.

In this study, a real-time environment with strong applicability is built by randomly initializing the position and speed of obstacle ships. The position and navigation situation of the ship are abstracted and constructed into a Markov decision process. According to the characteristics of the task, a novel reward function is designed, including daily



(a) The navigation path of all ships. (b) Speed and rudder angle of the USV.

(c) The USV is avoiding ship 1. (d) The USV is avoiding ship 3.

Fig. 23. Collision avoidance simulation for five obstacle ships.

rewards and settlement rewards. Thus, the collision avoidance problem is transformed into a strategy solving problem with distance, direction, speed and GOLREGs constraints, which ensure USV operates in a smooth and safe manner. Aiming at the exploration and exploitation scheme that makes it difficult for agents to find key information in the environment, this paper proposes a layered sampling exploration method, which can improve the playback rate of high-value samples and prevent overfitting. Compared with random sampling, this algorithm has achieved better learning effect in average cumulative reward and training time. In this paper, a simulation training platform is built to carry out real-time planning and collision avoidance simulation in a multi-obstacle ship environment. Through the test of four simulation scenarios from easy to difficult, it is proved that the algorithm has good effect and has certain application value.

In the future, more uncertainties of maritime navigation need to be considered. In order to promote the practical application of algorithm in this field, it is necessary to develop hardware equipment and carry out verification and optimization.

CRediT authorship contribution statement

Xinli Xu: Ideas, Evolution of overarching research goals and aims, Creation of models

Peng Cai: Designing computer programs, Implementation of the computer code and supporting algorithms

Yunlong Cao: Methodology, Software, Investigation, Writing – review & editing

Zhenzhong Chu: Validation, Investigation, Writing – review & editing

Wenbo Zhu: Validation, Investigation, Writing – review & editing

Weidong Zhang: Supervision, Resources, Funding acquisition, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgments

This paper is partly supported by the National Key R&D Program of China(2022ZD0119900) , Shanghai Science and Technology program (22015810300),Hainan Province Science and Technology Special Fund (ZDYF2021GXJS041), and the National Natural Science Foundation of China(U2141234).

References

Alvarez, A., Caiti, A., Onken, R., 2004. Evolutionary path planning for autonomous underwater vehicles in a variable ocean. IEEE J. Ocean. Eng. 29 (2), 418–429.

Belkhouche, F., 2009. Reactive path planning in a dynamic environment. IEEE Trans. Robot. 25 (4), 902–911.

Bellemare, M.G., Candido, S., Castro, P.S., 2020. Autonomous navigation of stratospheric balloons using reinforcement learning. Nature 588 (7836), 77–82.

Cheng, X., Lu, Y., Qiao, L., Hu, Z., Zhang, W., 2021. Multitask cooperative formation control of autonomous surface vehicles with interception of moving objects. IEEE J. Ocean. Eng. 47 (2), 271–281.

Chun, D.H., Roh, M.I., Lee, H.W., 2021. Deep reinforcement learning-based collision avoidance for an autonomous ship. Ocean Eng. 234 (5), 109216.

Deseck, Pierre, 1983. International Regulations for Preventing Collisions at Sea. Barker & Howard.
Du, B., Lin, B., Zhang, C., Dong, B., Zhang, W., 2022. Safe deep reinforcement learning-based adaptive control for USV interception mission. Ocean Eng. 246, 110477.
Duchoň, F., Babinec, A., Kajan, M., Beňo, P., Florek, M., Fico, T., 2014. Path planning with modified a star algorithm for a mobile robot. Procedia Eng. 96 (1), 59–69.
González, D., Pérez, J., Milanés, V., Nashashibi, F., 2015. A review of motion planning techniques for automated vehicles. IEEE Trans. Intell. Transport. Syst. 17 (4), 1135–1145.
Guoqing, Z., Yingjie, D., Weidong, Z., 2018. Novel DVS guidance and path-following control for underactuated ships in presence of multiple static and moving obstacles. Ocean Eng. 170 (11), 100–110.
Haiqing, S., Chen, G., 2018. An intelligent collision avoidance and navigation approach of unmanned surface vessel considering navigation experience and rules. J. Harbin Eng. Univ. 39 (6), 998–1005.
Hwang, J.Y., Kim, J.S., Lim, S.S., Park, K.H., 2003. A fast path planning by path graph optimization. IEEE Trans. Syst. Man Cybern. Syst. Hum. 33 (1), 121–129.
Ji, J., Khajepour, A., Melek, W.W., 2017. Path planning and tracking for vehicle collision avoidance based on modelpredictive control with multiconstraints. IEEE Trans. Veh. Technol. 66 (2), 952–964.
Khatib, O., 1985. Real-time obstacle avoidance for manipulators and mobile robots. In: Proceedings. IEEE International Conference on Robotics and Automation, vol. 2, pp. 500–505, 2.
Kothari, M., Postlethwaite, I., 2013. A probabilistically robust path planning algorithm for UAVs using rapidly-exploring random trees. J. Intell. Rob. Syst. 71 (2), 231–253.
Li, L., Wu, D., Huang, Y., 2021. A path planning strategy unified with a COLREGS collision avoid-ance function based on deep reinforcement learning and artificial potential field. Appl. Ocean Res. 113 (7), 102759.
Lillicrap, T.P., Cownden, D., Tweed, D.B., Akerman, C.J., 2016. Random synaptic feedback weights support error backpropagation for deep learning. Nat. Commun. 7 (1), 1–10.
Lu, L., Dan, W., Zhouhua, P., 2020. Predictor-based LOS guidance law for path following of underactuated marine surface vehicles with sideslip compensation. Ocean Eng. 24 (1), 340–348.
Mathis, A., Mamidanna, P., Cury, K.M., Abe, T., Murthy, V.N., Mathis, M.W., Bethge, M., 2018. DeepLabCut: markerless pose estimation of user-defined body parts with deep learning. Nat. Neurosci. 21 (9), 1281–1289.
Mirowski, P., Pascanu, R., Viola, F., Soyer, H., Ballard, A.J., Banino, A., Hadsell, R., 2016. Learning to navigate in complex environments. arXiv preprint arXiv:1611.03673.
Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Hassabis, D., 2015. Human-level control through deep reinforcement learning. Nature 518 (7540), 529–533.
Morris, G., Nevet, A., Arkadir, D., Vaadia, E., Bergman, H., 2006. Midbrain dopamine neurons encode decisions for future action. Nat. Neurosci. 9 (8), 1057–1063.
Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van, G., Hassabis, D., 2016. Mastering the game of Go with deep neural networks and tree search. Nature 529 (7587), 484–489.
Vinyals, O., Babuschkin, I., Czarnecki, W.M., Mathieu, M., Dudzik, A., Chung, J., Silver, D., 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. Nature 575 (7782), 350–354.
Wang, C., Wang, J., Shen, Y., 2019a. Autonomous navigation of UAVs in large-scale complex environments: a deep reinforcement learning approach. IEEE Trans. Veh. Technol. 68 (3), 2124–2136.
Wang, X., Liu, X., Shen, T., 2019b. A greedy navigation and subtle obstacle avoidance algorithm for. USV using reinforcement learning 39 (1), 770–775.
Woo, J., Kim, N., 2020. Collision avoidance for an unmanned surface vehicle using deep reinforcement learning. Ocean Eng. 199 (20), 107001.
Wu, X., Chen, H., Chen, C., 2020. The autonomous navigation and obstacle avoidance for USVs with ANOA deep reinforcement learning method. Knowl. Base Syst. 196 (8), 105201.
Wu, J., Wang, H., Wang, Y., 2021. UAV reactive interfered fluid path planning. Acta Autom. Sin. 47 (2), 1–16.
Xie, S., Chu, X., Zheng, M., 2020. A composite learning method for multi-ship collision avoidance based on reinforcement learning and inverse control. Neurocomputing 411 (2), 375–392.
Xinli, X., Yu, L., Xiaocheng, L., Weidong, Z., 2020a. Intelligent collision avoidance algorithms for USVs via deep reinforcement learning under COLREGs. Ocean Eng. 217 (1), 1–14.
Xinli, X., Wei, P., Yubo, H., Weidong, Z., 2020b. Dynamic collision avoidance algorithm for USVs via layered APF with collision cone. J. Navig. 73 (6), 1306–1325.
Yin, C., Weidong, Z., 2018. Concise deep reinforcement learning obstacle avoidance for underactuated unmanned marine vessels. Neurocomputing 272 (5), 63–73.
Zhang, X., Wang, C., Liu, Y., 2019. Decision-making for the autonomous navigation of maritime au-tonomous surface ships based on scene division and deep reinforcement learning. Sensors 19 (18), 4055.
Zhao, L., Roh, M.I., 2019. COLREGs-compliant multiship collision avoidance based on deep reinforce-ment learning. Ocean Eng. 191 (18), 106436.