Neurocomputing 411 (2020) 375–392

Contents lists available at ScienceDirect

Neurocomputing

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

# A composite learning method for multi-ship collision avoidance based on reinforcement learning and inverse control

Shuo Xie, Xiumin Chu, Mao Zheng *, Chenguang Liu

*National Engineering Research Center for Water Transport Safety, Wuhan University of Technology, 430063 Wuhan, Hubei Province, PR China*

ARTICLE INFO

*Article history:*
Received 18 June 2019
Revised 12 April 2020
Accepted 28 May 2020
Available online 4 June 2020
Communicated by W. Gao

*Keywords:*
Ship collision avoidance
Asynchronous advantage actor-critic
Long short-term memory neural network
Inverse control

ABSTRACT

Model-free reinforcement learning methods have potentials in ship collision avoidance under unknown environments. To defect the low efficiency problem of the model-free reinforcement learning, a composite learning method is proposed based on an asynchronous advantage actor-critic (A3C) algorithm, a long short-term memory neural network (LSTM) and Q-learning. The proposed method uses Q-learning for adaptive decisions between a LSTM inverse model-based controller and the model-free A3C policy. Multi-ship collision avoidance simulations are conducted to verify the effectiveness of the model-free A3C method, the proposed inverse model-based method and the composite learning method. The simulation results indicate that the proposed composite learning based ship collision avoidance method outperforms the A3C learning method and a traditional optimization-based method.



## 1. Introduction

The automation of vehicles is an effective approach to reduce the risk caused by dangerous driving behaviors [20,71]. Due to the safety requirements of the autonomous surface vessels (ASVs), a collision avoidance system becomes more and more important [4]. With the recent development of artificial intelligence, deep reinforcement learning (DRL) methods [34,44] have profound effects on complex decision-making tasks like multi-ship collision avoidance, for which the DRL method provides an alternative approach compared with the traditional methods. As a branch of DRL methods, the policy gradient-based methods (e.g., deep deterministic policy gradient (DDPG) [25], asynchronous advantage actor-critic (A3C) [33], proximal policy optimization (PPO) [41], etc) can solve continuous action problems with a deterministic policy parameterized by neural networks. At present, the DRL has made breakthroughs in a variety of domains, such as control [38], path planning [29], collision avoidance [10] and so on. In this section, the existing ship collision avoidance methods are introduced, including the traditional methods and DRL based methods.

### 1.1. A survey on traditional ship collision avoidance methods

Traditional ship collision avoidance methods can be briefly divided to two categories: path generation methods and intelligent optimization methods.

Path generation methods mainly include global grid based methods (e.g., A* [47,26]) and local path generation methods (e.g., artificial potential field (APF) [62,63]). As a heuristic search algorithm, A* considers both the origin and the destination, which has a global optimality. Hierarchical planning [53,8] and additional constraints [11,48] are the commonly used approaches in A* to improve the search efficiency and path smoothness. Compared with A*, APF [31,24] has smaller computation and smoother paths by using artificial gravitational and repulsive field to model the navigation environment.

With the development of intelligent optimization algorithms, the optimization based methods, e.g., fuzzy mathematics, neural networks and swarm intelligence, have attracted more attention in ship collision avoidance [49] than before. Fuzzy mathematics has been applied in the fuzzy classification [13,12] and reasoning [39,40] of ship collision risk for a long time. Generally, the output of fuzzy mathematics relies on the membership functions set in advance, which needs more prior knowledge. Besides, neural networks [17] is another powerful approach to model the uncertain factors in reasoning the ship collision risk. The neural networks are commonly combined with fuzzy mathematics [2] and expert system (ES) [46] to realize reliable collision avoidance.

Recently, swarm intelligence and evolution optimization methods have become hot topics in ship collision avoidance. Ant colony optimization (ACO) [22,23,51] and particle swarm optimization (PSO) [32,35,7,28] are the most commonly used algorithms in ship collision avoidance, which can obtain good results with an appropriate fitness function.

* Corresponding author.
*E-mail address:* zhmao2018@126.com (M. Zheng).

https://doi.org/10.1016/j.neucom.2020.05.089
0925-2312/

### 1.2. Related works on DRL based methods

At present, several DRL methods have been applied in ship collision avoidance. The key issues in the DRL methods are the definitions of the state, action and reward function.

Deep Q-network is firstly applied in ship collision avoidance with a low-dimensional state-action space. In Ref. [42], the discrete ship heading changes and a set of distances measured by the fixed interval detection lines around the ship are defined as the actions and states in deep Q-network, respectively. The proposed deep Q-network is verified in both simulations [42] and model ship experiments [43]. In Ref. [65], a so-called constrained deep Q-network is proposed to reduce the complexity of the state-action space by adding constrains based on international regulations for preventing collisions at sea (COLREGs), which also obtains good collision avoidance results.

In addition, using high-complex deep neural networks is an effective approach to train the value function in a high-dimensional state-action space. In Ref. [9], a convolutional neural network (CNN) is used to obtain a more reliable policy. This CNN model uses the perception information, the motion state and the actions of the ship in a certain horizon to produce a chain lumped state matrix for reinforcement learning.

Although deep Q-network based methods have achieved good results, a large memory space is still required for the discrete actions. With the development of policy gradient-based methods (e.g., DDPG, A3C, etc), a deterministic policy from ship states to continuous actions can be obtained with less memory space. In Ref. [19] the DDPG algorithm is applied in a simplified state-action space, in which the vertical distances away from the target course are defined as the continuous actions. In Xu et al. [59], the DDPG algorithm is also applied for ship collision avoidance with the same state-action space defined in Cheng and Zhang [9]. The simulation results have indicated the effectiveness and advantages of DDPG algorithm in continuous decisions for ship collision avoidance.

Although existing DRL methods have obtained rich achievements in ship collision avoidance, the relative low efficiency is the main barrier in application. At present, the asynchronous computing framework (e.g., A3C) and model-based for model-free (i.e., MB-MF) learning can both improve the learning efficiency. To reduce the exploring time and memory costs, an on-policy A3C method [33] updates the obtained gradients through asynchronous parallel computing, which has greater potentials than the experience reply technique [37] in DDPG. For MB-MF learning methods, the main idea is to establish a short-term model-based optimizer for combination with the long-term learner. In Ref. [5], a linear-quadratic regulator (LQR) optimizer is adopted for data-efficient learning in both simulations and real-world experiments. This LQR optimizer uses a time-varying linear-Gaussian (TVLG) model for optimization, which is established by fitting the samples. In Ref. [36], the widely used model predictive control (MPC) approach [70,67–69] is applied for model-based learning, which can initialize the networks for accelerating the model-free learning. The model used in MPC is a dynamic model that predicts the state changes over the time step duration.

### 1.3. Motivation and contributions

Surveying from the ship collision avoidance research, DRL methods obtain an optimal policy by maximizing future rewards through interactions, which have better potentials than the traditional methods in uncertain environments. In spite of this, the main drawback in existing DRL based collision avoidance methods is the low learning efficiency problem, especially the off-line methods such as DDPG.

#### 1.3.1. Motivations

As denoted in the related DRL works, the asynchronous computing and MB-MF learning can be considered to improve the learning efficiency for ship collision avoidance. Motivated by Refs. [33,5,36], the main works of this paper are as follows:

(1) To reduce the learning time and memory costs, the A3C method [33] is applied for ship collision avoidance in this study.
(2) The MB-MF learning is considered to further improve the learning efficiency. Besides, traditional model-based method [5,36] needs to establish an accurate model before designing the controller. While the inverse control [55,3,14] method directly uses the inverse model between the desired outputs and inputs as the controller, which has more concise strategy and potentials in MB-MF learning.

#### 1.3.2. Contributions

In order to improve the learning efficiency for ship collision avoidance, a composite learning method is proposed in this study. A simple framework of the proposed composite learning is shown in Fig. 1 for brief explanation. Instead of the simple initialization in Ref. [36], the main idea of this method is to use Q-learning for

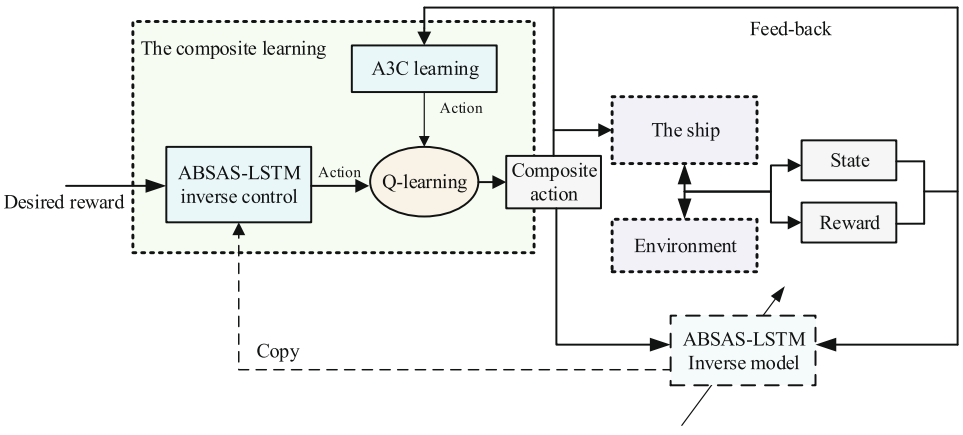


**Fig. 1.** A concise framework of the proposed composite learning.

adaptive decisions between the actions of a model-free A3C and a model-based inverse controller in the entire learning process. The contributions of this study are:

(1) Compared with traditional ship collision avoidance methods, the proposed method has efficient learning ability by integrating A3C reinforcement learning, which performs better with limited perception and states.
(2) Compared with pure model-free A3C method, the proposed composite method uses a feed-forward LSTM controller and Q-learning to generate supervised trajectories for A3C, which has higher learning efficiency.

Therefore, in addition to A3C applications, the originality of the proposed composite learning method is reflected in two aspects: the inverse controller for ship collision avoidance and decisions based on Q-learning.

The remainder of this article is organized as follows. In Section 2, the ship hydrodynamic model and collision risk model are described. In Section 3, the A3C learning based ship collision avoidance method is described. In Section 4, the composite learning method is proposed. In Section 5, simulation experiments under multi-ship encounters are carried out to assess the effectiveness of the proposed methods. In Section 6, conclusions and future research are presented.

## 2. Ship hydrodynamic model and collision risk model

In consideration of the nonlinear characteristics of ship motion and different collision states (e.g., distance of the closest point of approach (DCPA) and time to the closest point of approach (TCPA)), the three degree-of-freedom (3-DOF) ship hydrodynamic model and the collision risk model are used to calculate the collision risk index (CRI).

### 2.1. Ship hydrodynamic model

Abkowitz model [1] and MMG (math model group) model [64] have been widely used for the ship motion modeling. Generally, the 3-DOF ship motion coordinate system is shown in Fig. 2.

In Fig. 2, $O_o - x_o y_o z_o$ is the inertial coordinate system of the ship; $O - xyz$ is the co-rotational coordinate system of the ship; $\delta$ and $\psi$ are the rudder and heading angle of the vessel, respectively; $u$ and $v$ are the velocities in surge (body-fixed $x$) and sway (body-fixed $y$) respectively, and $r$ is the velocity of the heading angle $r = \dot{\psi}$, i.e., the yaw direction (body-fixed $z$); $\beta$ is the drift angle. Then the 3-DOF form of the Abkowitz model and the MMG model can be denoted as [64,30]:

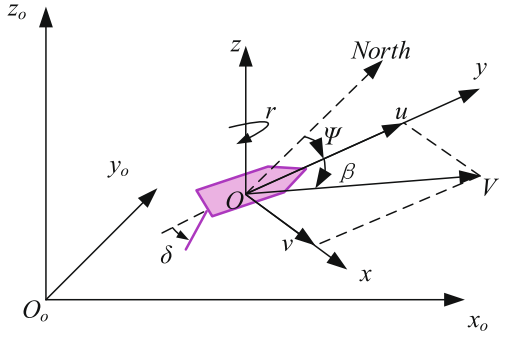


**Fig. 2.** Ship motion coordinate system.

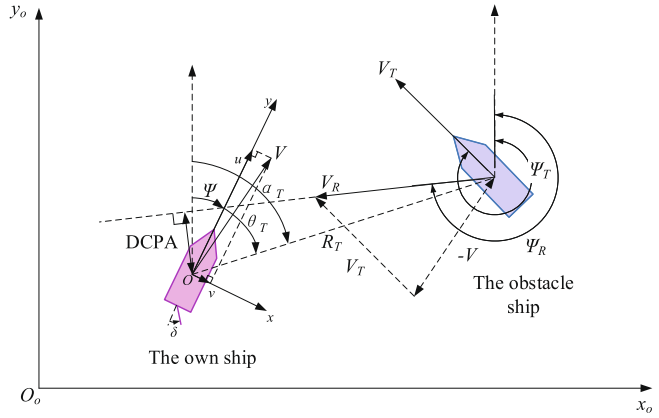


**Fig. 3.** Typical two-ship encounter.

$$\dot{\boldsymbol{\eta}} = \boldsymbol{R}(\psi)\boldsymbol{\upsilon}, \boldsymbol{M}\dot{\boldsymbol{\upsilon}} = \boldsymbol{f}(u, v, r, \delta, n), \tag{1}$$

where $\boldsymbol{\eta} = [x \quad y \quad \psi]^{\mathrm{T}}$ and $\boldsymbol{\upsilon} = [u \quad v \quad r]^{\mathrm{T}}$ are the position and velocity vectors of the ship, respectively. $\delta$ is the rudder angle and $n$ is the engine speed. $\boldsymbol{f}$ is the nonlinear lumped force and moment matrix of the ship with respect to $\boldsymbol{\upsilon}, \delta$ and $n$. $\boldsymbol{R}(\psi)$ is the rotation matrix between $\dot{\boldsymbol{\eta}}$ and $\boldsymbol{\upsilon}$. $\boldsymbol{M}$ is the inertia matrix of the ship:

$$\begin{aligned} \boldsymbol{R}(\psi) &= \begin{bmatrix} \sin(\psi) & \cos(\psi) & 0 \\ -\cos(\psi) & \sin(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \\ \boldsymbol{M} &= \begin{bmatrix} m - X_{\dot{u}} & 0 & 0 \\ 0 & m - Y_{\dot{v}} & mx_G - Y_{\dot{r}} \\ 0 & mx_G - N_{\dot{v}} & I_z - N \end{bmatrix}, \end{aligned} \tag{2}$$

where $m$ is the total mass of the vessel; $x_G$ is the longitudinal coordinate of the gravity center of the vessel in surge direction; $I_Z$ is the moment of the inertia; $X_{\dot{u}}$, $Y_{\dot{v}}$, $Y_{\dot{r}}$, $N_{\dot{v}}$ and $N_{\dot{r}}$ are the inertia coefficients.

### 2.2. Collision risk model

As shown in Fig. 3, assuming that the lumped state matrix of the own ship is $\boldsymbol{X} = [\boldsymbol{\eta}^{\mathrm{T}}, \boldsymbol{\upsilon}^{\mathrm{T}}]^{\mathrm{T}}$, and the state matrix of the obstacle ship is $\boldsymbol{X_T} = [\boldsymbol{\eta}_T^{\mathrm{T}}, \boldsymbol{\upsilon}_T^{\mathrm{T}}]^{\mathrm{T}} = [x_T, y_T, \psi_T, u_T, v_T, r_T]^{\mathrm{T}}$, the relative motion parameters, i.e., DCPA, TCPA, the relative distance, position direction and the relative speed, can be used to evaluate the CRI between two ships in typical encounters [49]:

$$\begin{aligned} DCPA &= R_T \sin(\psi_R - \alpha_T - \pi), TCPA \\ &= R_T \cos(\psi_R - \alpha_T - \pi)/V_R, R_T \\ &= \sqrt{(x_T - x)^2 + (y_T - y)^2}, \theta_T = \alpha_T - \psi \pm 2\pi, \end{aligned} \tag{3}$$

where $R_T$ is the relative distance between two ships; $\psi_R$ is the relative course of the obstacle ship; $\alpha_T$ is the true relative position direction of the obstacle ship; $\theta_T$ is converted from $\alpha_T$ in body-fixed coordinate system of the own ship; $V_T = \sqrt{u_T^2 + v_T^2}$ and $V = \sqrt{u^2 + v^2}$ are the speeds of the obstacle ship and the own ship, respectively; $V_R$ is the relative speed of the obstacle ship. The $V_R$, $\psi_R$ and $\alpha_T$ are calculated as:

$$
\begin{aligned}
&v_{x_R} = u_T \sin(\psi_T) + v_T \cos(\psi_T) - (u \cdot \sin(\psi) + v\cos(\psi)),\\
&v_{y_R} = u_T \cos(\psi_T) - v_T \sin(\psi_T) - (u\cos(\psi) - v\sin(\psi)),\\
&V_R = \sqrt{v_{x_R}^2 + v_{y_R}^2},\\
&\psi_R = \arctan\tfrac{v_{x_R}}{v_{y_R}} +\\
&\begin{cases} 0 & v_{x_R} \geqslant 0 \cup v_{y_R} \geqslant 0\\ \pi & (v_{x_R} < 0 \cup v_{y_R} < 0) || (v_{x_R} \geqslant 0 \cup v_{y_R} < 0)\\ 2\pi & (v_{x_R} < 0 \cup v_{y_R} \geqslant 0) \end{cases},\\
&\alpha_T = \arctan\tfrac{(x_T - x)}{(y_T - y)} +\\
&\begin{cases} 0 & (x_T - x) \geqslant 0 \cup (y_T - y) \geqslant 0\\ \pi & ((x_T - x) < 0 \cup (y_T - y) < 0) || ((x_T - x) \geqslant 0 \cup (y_T - y) < 0)\\ 2\pi & ((x_T - x) < 0 \cup (y_T - y) \geqslant 0) \end{cases},
\end{aligned}
\tag{4}
$$

where $v_{x_R}$ and $v_{y_R}$ are the relative speed components of the obstacle ship on the $X$ and $Y$ axis, respectively. Then the membership functions of DCPA, TCPA, $R_T, \theta_T$ and velocity ratio $K = V_T/V$ can be used to calculate the final CRI based on the fuzzy comprehensive evaluation method in Ref. [6]:

$$
\begin{aligned}
&f_{CRI} = \boldsymbol{\lambda}_{CRI}\boldsymbol{u}_{CRI},\\
&\boldsymbol{\lambda}_{CRI} = [\lambda_{DCPA} \quad \lambda_{TCPA} \quad \lambda_{R_T} \quad \lambda_{\theta_T} \quad \lambda_K],\\
&\boldsymbol{u}_{CRI} = [u_{DCPA} \quad u_{TCPA} \quad u_{R_T} \quad u_{\theta_T} \quad u_K]^{\mathrm{T}},
\end{aligned}
\tag{5}
$$

where $u_{DCPA}$ , $u_{TCPA}$ ,$u_{R_T}$, $u_{\theta_T}$ and $u_K$ are the membership function values of DCPA, TCPA, $R_T$, $\theta_T$ and $K$, respectively. $\lambda_{DCPA}$ , $\lambda_{TCPA}$ ,$\lambda_{R_T}$, $\lambda_{\theta_T}$ and $\lambda_K$ are the set weights of $u_{DCPA}$ , $u_{TCPA}$ ,$u_{R_T}$, $u_{\theta_T}$ and $u_K$, respectively. In this study, the membership function values are calculated based on the method in Refs. [6,60]. Referring to [60], the weights can be set as $\boldsymbol{\lambda}_{CRI} = [0.400, 0.367, 0.133, 0.067, 0.033]^{\mathrm{T}}$ for collision risk estimation.

## 3. A3C learning based ship collision avoidance method

In this section, the basic A3C algorithm is applied to ship collision avoidance by regarding the collision avoidance process as a typical Markov decision-making process (MDP). The definitions of the state and reward function are the key issues in the application of A3C.

### 3.1. State definition

The goal of ship collision avoidance is to reach the destination and prevent collisions with all obstacle ships. A set of relative motion states (e.g., DCPA, TCPA, relative distance $R_T$, relative position direction $\theta_T$, etc) between the own ship and the obstacle ships can represent multiple sets of different original motion states ($\boldsymbol{X}$, $\boldsymbol{X}_T$), which are more suitable for policy learning. Therefore, the state with respect to the $i$th obstacle ship is defined as:

$$
\begin{aligned}
\boldsymbol{s}_t^i = \Big[ &DCPA^i(t) \quad TCPA^i(t) \quad R_T^i(t)\\
&\theta_T^i(t) \quad f_{CRI}^i(t) \quad C_T^i(t) \quad K^i \Big]^{\mathrm{T}}
\end{aligned}
\tag{6}
$$

In order to learn a general policy for different obstacle ships, the multi-ship collision avoidance process is divided into a set of sub-MDP processes for each obstacle ship. At each step, the own ship takes a rudder action $a = \delta$ to avoid the collisions and receives different reward $r^i$ with respect to the $i$th obstacle ship. Then the final action is obtained by a weighting method based on the collision risks as follows:

$$
u_{op} = \frac{f_{CRI}^i}{\sum_{i=1}^n f_{CRI}^i}\pi(\boldsymbol{s}^i),
\tag{7}
$$

where $u_{op}$ is the final action result for collision avoidance; $\pi$ is the learned policy; $f_{CRI}^i$ is the collision risk between the own ship and the $i$th obstacle ship. After the own ship takes the action $u_{op}$, the collision states with different obstacle ships are updated and collected for learning of the policy $u = \pi_{op}^{\theta}(\boldsymbol{s}^i)$. The framework of the multi-ship collision avoidance strategy is shown in Fig. 4

### 3.2. Reward design

A reasonable reward function $r$ is very important in reinforcement learning. The reward function for collision avoidance is designed from the aspects of safety and economy.

(1) Safety

In multi-ship encounters, the maximum collision risk and the average collision risk are both considered for safety:

$$
r_s = -\mu_1 \max\left\{ f_{CRI}^1 \quad f_{CRI}^2 \quad \cdots \quad f_{CRI}^n \right\} - \mu_2 \frac{1}{n}\sum_{i=1}^n f_{CRI}^i,
\tag{8}
$$

where $r_s$ is the reward for safety; $\mu_1 > 0$ and $\mu_2 > 0$ are the set weights for the maximum and average collision risk, respectively.

(2) Economy

During the voyage, steering with a large rudder angle will reduce the ship surge speed and increase the time for re-sailing. Generally, the squares of the rudder angle can be used to establish the economic reward with respect to the energy loss caused by large rudder angle:

$$
r_\delta = -\frac{1}{2}\delta^2.
\tag{9}
$$

In addition, the own ship also needs to avoid excessive deviations from the original paths. Thus, the widely used line-of-sight (LOS) guidance strategy is adopted in this study. Fig. 5 shows the LOS guidance strategy when the own ship is sailing between the adjacent path nodes $P_k(x_k, y_k)$ and $P_{k+1}(x_{k+1}, y_{k+1})$.

In LOS guidance, the ship is guided by minimizing the error $\tilde{\psi}_{\mathrm{LOS}}$ between the actual heading angle $\psi$ and the LOS angle $\psi_{\mathrm{LOS}}$. The LOS angle $\psi_{\mathrm{LOS}}$ can be calculated by solving the following equations:

$$
\begin{aligned}
(x_{\mathrm{LOS}} - x)^2 + (y_{\mathrm{LOS}} - y)^2 = R_{\mathrm{LOS}}^2, &\frac{y_{\mathrm{LOS}} - y_k}{x_{\mathrm{LOS}} - x_k} = \frac{y_{k+1} - y_k}{x_{k+1} - x_k}, \psi_{\mathrm{LOS}}\\
&= \arcsin\left(\frac{x_{\mathrm{LOS}} - x}{R_{\mathrm{LOS}}}\right),
\end{aligned}
\tag{10}
$$

where $P_{\mathrm{LOS}}(x_{\mathrm{LOS}}, y_{\mathrm{LOS}})$ is the LOS guidance point; $R_{\mathrm{LOS}}$ is the radius of the acceptance circle in Fig. 5. Then, the square of the error $\tilde{\psi}_{\mathrm{LOS}}$ is used to construct the economic reward with respect to the re-sailing as follows:

$$
r_{\mathrm{LOS}} = -\tfrac{1}{2}\tilde{\psi}_{\mathrm{LOS}}^2 = -\tfrac{1}{2}(\psi_{\mathrm{LOS}} - \psi)^2.
\tag{11}
$$

Then, the final reward function for collision avoidance $r$ can be obtained by weighting the safety reward $r_s$ and economy rewards $r_e$ and $r_LOS$:

$$
r = \lambda_s r_s + \lambda_\delta r_\delta + \lambda_{\mathrm{LOS}} r_{\mathrm{LOS}},
\tag{12}
$$

where $\lambda_\delta$, $\lambda_{\mathrm{LOS}}$ and $\lambda_s$ are the setting weights for $r_\delta$, $r_{\mathrm{LOS}}$ and the collision risk $r_s$, respectively.

### 3.3. Reinforcement learning based on A3C

To realize collision avoidance based on A3C, parallel asynchronous threads are created for learning. In each thread $j$, a parametric actor policy network $a = \pi^j(\boldsymbol{s}, \theta^{\pi j})$ is used to generate the optimal collision avoidance actions in each thread which maps the states $\boldsymbol{s}$ in Eq. (6) to the rudder action $a = \delta$. Then, a critic net-

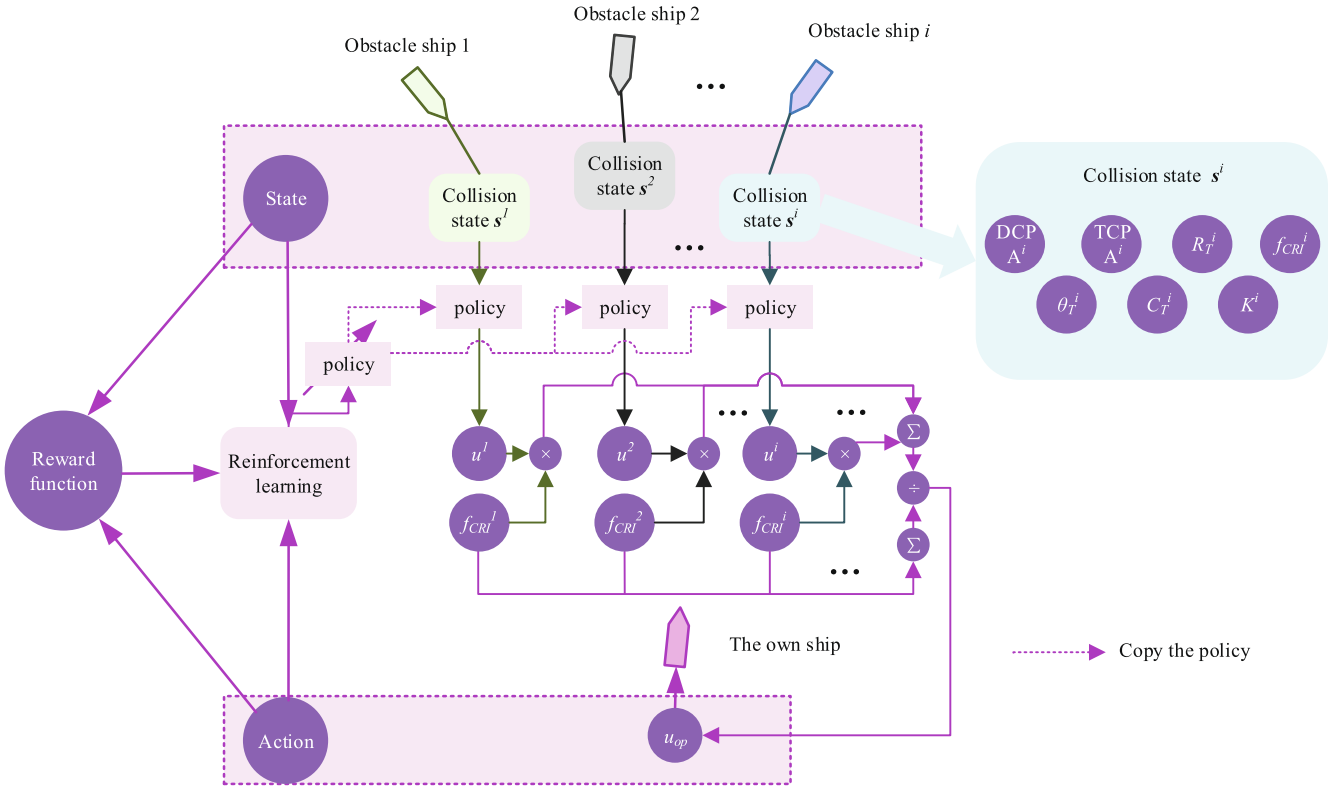


**Fig. 4.** The framework of the multi-ship collision avoidance strategy based on reinforcement learning.

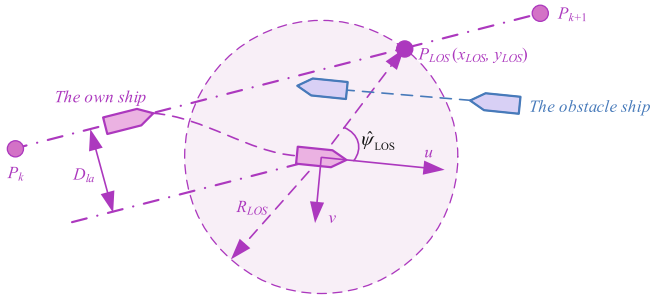


**Fig. 5.** The LOS guidance strategy and LOS tracking error.

work $V_{\theta^j}^j(\boldsymbol{s})$ is used to estimate the state value function $V^j$ with respect to the state $\boldsymbol{s}$. $\theta^{\pi j}$ and $\theta^j$ are the parameters of the actor and critic networks in $j$th thread, respectively. Moreover, a pair of global actor $\pi^g$ and critic $V^g$ networks are parametrized by $\theta^{\pi g}$ and $\theta^g$ to update the policy with all the obtained gradients in each thread. The reinforcement learning process of A3C is described as follows:

(1) Actor network

In each thread $j$, the state-actor value function of the actor $\pi^j(\boldsymbol{s}, \theta^{\pi j})$ can be defined as:

$$Q_\theta(\boldsymbol{s}_t, a_t) = \mathbb{E}\left[\sum_{k=t}^{\infty} \gamma^{k-t} r(\boldsymbol{s}_k, a_k) \middle| a_k = \pi^j\left(\boldsymbol{s}_k, \theta^{\pi j}\right)\right], \tag{13}$$

where $\gamma$ is the set discount of the reward $r$ denoted in Eq. (12), $\mathbb{E}[\cdot]$ is the expectation operator. Then the following Bellman equation can be obtained for the deterministic policy $\pi^j(\boldsymbol{s}, \theta^{\pi j})$:

$$Q_\theta(\boldsymbol{s}_t, a_t) = \mathbb{E}\left[r(\boldsymbol{s}_t, a_t) + \gamma Q_\theta\left(\boldsymbol{s}_{t+1}, \pi^j\left(\boldsymbol{s}_{t+1}, \theta^{\pi j}\right)\right)\right]. \tag{14}$$

Based on the Bellman equation, the actor $a = \pi^j(\boldsymbol{s}, \theta^{\pi j})$ can be updated by maximizing the expected reward with respect to the actor parameters:

$$\begin{aligned} \theta^{\pi j} &= \arg\max_{\theta^{\pi j}} J\left(\theta^{\pi j}\right) \\ &= \arg\max_{\theta^{\pi j}} \left\{\mathbb{E}_{\boldsymbol{s}\sim E_{\boldsymbol{s}}}\left[Q_\theta\left(\boldsymbol{s}, \pi^j\left(\boldsymbol{s}, \theta^{\pi j}\right)\right) - b(\boldsymbol{s})\right]\right\}, \end{aligned} \tag{15}$$

where $\boldsymbol{s} \sim E_{\boldsymbol{s}}$ denotes observing state $\boldsymbol{s}$ from the environment $E_{\boldsymbol{s}}$; $b(\boldsymbol{s})$ is the set baseline to reduce the valuation variance of the state-action value function $Q_\theta$. In order to solve the maximization problem, an advantage function is defined as the differential between the state-action value $Q_\theta$ and the estimated value by the critic $V_{\theta^j}^j(\boldsymbol{s})$:

$$A(\boldsymbol{s}, a) = Q_\theta\left(\boldsymbol{s}, \pi^j\left(\boldsymbol{s}, \theta^{\pi j}\right)\right) - V_{\theta^j}^j(\boldsymbol{s}), \tag{16}$$

where $A(\boldsymbol{s}, a)$ is a scalar advantage of the action $a = \pi^j\left(\boldsymbol{s}, \theta^{\pi j}\right)$. Since the only term with respect to the actor parameter $\theta^{\pi j}$ in Eq. (16) is the state-action value $Q_\theta$, the maximization problem of Eq. (15) can be transferred to:

$$\theta^{\pi j} = \arg\max_{\theta^{\pi j}} \left\{J\left(\theta^{\pi j}\right)\right\} = \arg\max_{\theta^{\pi j}} \left\{A\left(\boldsymbol{s}, \pi^j\left(\boldsymbol{s}, \theta^{\pi j}\right)\right)\right\}. \tag{17}$$

To solve the optimization problem in Eq. (17), the widely used policy gradient method [45] is applied to obtain the gradient of the actor policy based on the chain rule:

$$\begin{aligned} \nabla_{\theta^{\pi j}} J\left(\theta^{\pi j}\right) &= A(\boldsymbol{s}, a) \cdot \nabla_{\theta^\pi} \log \pi^j\left(\boldsymbol{s}, \theta^{\pi j}\right) \\ &= A(\boldsymbol{s}, a) \cdot \frac{\partial \log \pi^j\left(\boldsymbol{s}, \theta^{\pi j}\right)}{\partial \theta^{\pi j}}, \end{aligned} \tag{18}$$

where $\nabla$ is the gradient operator, $\nabla_{\theta^{\pi j}} \log \pi^j(\boldsymbol{s}, \theta^{\pi j})$ is the gradient of the likelihood of the actor $\pi^j$ with respect to the parameters $\theta^{\pi j}$, which is the policy gradient of the actor $\pi^j$ [1]. Then the obtained gradients in all threads are applied to the global actor network $\pi^g$ asynchronously:

$$\theta^{\pi g} = \theta^{\pi g} - \alpha_\pi \nabla_{\theta^{\pi j}} J\left(\theta^{\pi j}\right) = \theta^{\pi g} - \alpha_\pi \cdot A(\boldsymbol{s}, a) \cdot \frac{\partial \log \pi^j\left(\boldsymbol{s}, \theta^{\pi j}\right)}{\partial \theta^{\pi j}}, \quad (19)$$

where $\alpha_\pi$ is the learning rate of the actor policy. After applying the gradient, the parameters of the global actor will be pulled immediately to the actor in each thread as $\theta^{\pi j} = \theta^{\pi g}$ for the next learning epoch.

(2) Critic network

In each thread $j$, the critic network in the value function $V^j_{\theta^j}(\boldsymbol{s})$ is updated by the Bellman equation as in Q-learning [54], which can be optimized by minimizing the $n$-step loss between the output of $V^j_{\theta^j}(\boldsymbol{s})$ and real accumulated rewards as:

$$\begin{aligned} &L\left(\theta^j\right) = \mathbb{E}_{\boldsymbol{s}_t \sim E_{\boldsymbol{s}}}\left[\left(V^j_{\theta^j}\left(\boldsymbol{s}_t \middle| \theta^j\right) - y_t\right)^2\right], \\ &y_t = r(\boldsymbol{s}_t, a_t) + \gamma r(\boldsymbol{s}_{t+1}, a_{t+1}) + \ldots \\ &+\gamma^{n-1} r(\boldsymbol{s}_{t+n-1}, a_{t+n-1}) + \gamma V^j_{\theta^j}\left(\boldsymbol{s}_{t+n} \middle| \theta^j\right), \end{aligned} \quad (20)$$

where $n$ is the set step number for updating the critic network. Then the gradient of the loss $L\left(\theta^j\right)$ with respect to the critic parameter $\nabla_\theta L\left(\theta^j\right) = \frac{\partial L(\theta^j)}{\partial \theta^j}$ can be obtained in each thread. After that, the global critic network $V^g$ is also updated based on all the obtained gradients:

$$\theta^g = \theta^g - \alpha \nabla_{\theta^j} L\left(\theta^j\right) = \theta^g - \alpha \frac{\partial L\left(\theta^j\right)}{\partial \theta^j}, \quad (21)$$

where $\alpha$ is the learning rate of the critic network. After applying the gradient, the parameters of the global critic will be also pulled immediately to the critic in each thread as $\theta^j = \theta^g$ for the next learning epoch.

## 4. The proposed composite Learning method

In this section, a composite learning method is proposed by using Q-learning to make adaptive decisions between the A3C learning and an inverse model-based controller. The inverse controller and Q-learning decisions are described as follows.

### 4.1. Inverse control for ship collision avoidance

As a branch of recurrent neural network (RNN), the LSTM [15] has the advantage of approximating time sequence nonlinear systems with high precision [56], which has potentials in inverse system modeling.

Firstly, the Markov decision process of collision avoidance can be described by the following approximated function:

$$r(t) = F(\boldsymbol{s}(t), a(t)), \quad (22)$$

where $F(\cdot)$ represents the relationship between the reward $r(t)$, the action $a(t)$ and the state $\boldsymbol{s}(t)$. Due to the great hysteresis of ship motion, the steering effect of the current rudder action will be reflected in future rewards. Therefore, we consider the obtained discounted reward vector in a certain horizon $\boldsymbol{R}^H$ as:

$$\begin{aligned} \boldsymbol{R}^H(t) &= \left[\gamma^{H-1} r(t-H) \quad \gamma^{H-2} r(t-H+1) \quad \cdots \quad r(t)\right] \\ &= \boldsymbol{F}^H(\boldsymbol{s}(t-H+1), a(t-H+1), \boldsymbol{s}(t-H+2), a(t-H+2), \cdots, a(t)), \end{aligned} \quad (23)$$

where $\boldsymbol{R}^H(t)$ is the discounted reward vector, $\gamma$ is the discount factor, $\boldsymbol{F}^H(\cdot)$ is the extension function of $F(\cdot)$. Then, the action at time $t$ can be output by the following inverse model:

$$\begin{aligned} a(t) = \boldsymbol{F}^{H^{-1}}(&\boldsymbol{s}(t-H+1), a(t-H+1), \boldsymbol{s}(t-H+2), cdots, \\ &\boldsymbol{s}(t), \gamma^{H-1} r(t-H), \gamma^{H-2} r(t-H), r_d(t)), \end{aligned} \quad (24)$$

where $\boldsymbol{F}^{H^{-1}}(\cdot)$ is the approximated inverse function of $\boldsymbol{F}^H(\cdot)$; $r_d(t)$ is the target reward at time $t$, which is used as an input of $\boldsymbol{F}^{H^{-1}}(\cdot)$.

Based on the defined inputs and outputs, the LSTM neural network is used to train the inverse model $\boldsymbol{F}^{H^{-1}}(\cdot)$ in Eq. (31) with a long-term dependence [50]. The feed-forward step of a LSTM cell is based on the following equations:

$$\begin{aligned} \boldsymbol{i}_t &= \sigma\left(\boldsymbol{w}^i_x \boldsymbol{x}_t + \boldsymbol{w}^i_h \boldsymbol{i}^h_{t-1}\right), \\ \boldsymbol{f}_t &= \sigma\left(\boldsymbol{w}^f_x \boldsymbol{x}_t + \boldsymbol{w}^f_h \boldsymbol{f}^h_{t-1}\right), \\ \tilde{\boldsymbol{c}}_t &= \phi\left(\boldsymbol{w}^c_x \boldsymbol{x}_t + \boldsymbol{w}^c_h \boldsymbol{f}^h_{t-1}\right), \\ \boldsymbol{c}_t &= \boldsymbol{f}_t \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \tilde{\boldsymbol{c}}_t \\ \boldsymbol{o}_t &= \sigma\left(\boldsymbol{w}^o_x \boldsymbol{x}_t + \boldsymbol{w}^o_h \boldsymbol{o}^h_{t-1}\right), \\ \boldsymbol{y}_t &= \boldsymbol{o}_t \phi(\boldsymbol{c}_t), \end{aligned} \quad (25)$$

where $\boldsymbol{i}_t$, $\boldsymbol{f}_t$ and $\boldsymbol{o}_t$ are the outputs of the input, forget and output gates, respectively; $\boldsymbol{c}_t$ and $\boldsymbol{y}_t$ are the outputs of the cell and the LSTM unit, respectively; $\boldsymbol{w}^i_*$, $\boldsymbol{w}^f_*$, $\boldsymbol{w}^o_*$ and $\boldsymbol{w}^c_*$ represent the weights of the input, forget, output gates and the cell, respectively; $\boldsymbol{w}^*_x$, $\boldsymbol{w}^*_h$ and $\boldsymbol{w}^*_c$ represent the weights of the input state $\boldsymbol{x}$, the history output of each gate and the cell, respectively; $\sigma$ and $\phi$ are sigmoid and tanh activation function.

After training the LSTM network, the optimal action can be output directly from $\boldsymbol{F}^{H^{-1}}(\cdot)$ by setting an empirical maximum reward $r_d(t) = \hat{r}_{\max}(t)$. Then the LSTM model $\boldsymbol{F}^{H^{-1}}(\cdot)$ can be regarded as a typical inverse controller. Besides, to defect the local optimum in complex model training, we adopt a simple beetle antenna search algorithm (BAS) [57] and elite opposition based learning (EOBL) technique [66] to initialize the weights in LSTM networks for fine-tuning. The convergence analyze and optimality proof of the beetle antenna search algorithm are given in Refs. [57,58].

### 4.2. Decisions based on Q-learning

**Algorithm 1**. Collision avoidance based on the composite learning method

**Input:** The trained inverse model $\boldsymbol{F}^{H^{-1}}$; The initial state of the own ship and the other ship at the current time, $\boldsymbol{X}_s(0)$ and $\boldsymbol{X}_T(0)$;

**Output:** The optimal collision avoidance policy, $a = \pi(\boldsymbol{s}, \theta^\pi)$;

1: Initialize the parameters of A3C, i.e., the maximum step in an episode $T$, the maximum episodes for learning $N_{ep}$, the learning rates for actor and critic $\alpha_\pi$ and $\alpha$, reward discount in A3C $\gamma$, the step number for updating the critic $n$;

2: Initialize the parameters of the upper Q-learning layer, i.e., the Q-table, the learning rate and reward discount $\alpha_Q$ and $\gamma_Q$, the weights in reward $\xi_1$ and $\xi_2$.

3: Load the inverse model $\boldsymbol{F}^{H^{-1}}$ and start A3C learning with multiple threads.

[1] The policy gradient is a widely used method in policy-based reinforcement learning, of which the details can be seen in Ref. [45].

**Algorithm 1**. Collision avoidance based on the composite learning method

4: **for** Each thread **do**
5: **while** $k < k_{max}$ **do**
6: Reset the state of the own ship and the obstacle ships, set $t = 0$;
7: **while** $t < T$ **do**
8: Choose the best action in Table 1 with the Q-table in each thread.
9; **switch** the best state **do**
10: **case 1**
11: Generate the action by the actor in A3C and add the exploratory noise.
12: **case 2**
13: Generate the action by the inverse model $\boldsymbol{F}^{H^{-1}}$ directly.
14: **case 3**
15: Generate the action by the composite model based on Eq. (26).
16: **case 4**
17: Generate the action the actor in A3C directly without exploration.
18: Execute the action and obtain the reward.
19: $t = t + 1$
20: **end while**
21: $k = k + 1$
22: **end while**
23: **end for**
24:

In order to make full use of the prior model knowledge during the entire learning process, the Q-learning method is adopted to realize adaptive combination of the model-free A3C learner and the inverse model-based controller.

Firstly, a composite output is obtained by weighting the outputs of the inverse model and the actor policy in A3C:

$$a_{co}(k) = \lambda_{a3c}(k)\pi(\boldsymbol{s}, \theta^{\pi}) + (1 - \lambda_{a3c}(k))a_{lstm}(k), \tag{26}$$

where $a_{co}(k)$ is the composite output in $k$ th step; $\lambda_{a3c}(k)$ is the setting weight for the output of the A3C actor $\pi(\boldsymbol{s}, \theta^{\pi})$ in $k$ th step.

In the early stage of A3C learning, the obtained reward is relatively low and the inverse model is expected to provide more supervision for A3C. When the reward in A3C becomes stable, the actor is considered to obtain a better collision avoidance policy. At this time, the weight of A3C output can be enlarged to learn an optimal policy in an unknown environment. Therefore, an adaptive weight is set based on the reward changes in A3C:

$$\lambda_{a3c}(k) = e^{-\Delta R(k)} = e^{R(k-1)-R(k)}, R(k) = \sum_{t=1}^{T} r_k(t), \tag{27}$$

where $R(k)$ is the current sum rewards of all steps in $k$ th episode; $T$ is the set maximum steps in each episode. Besides, if the inverse model cannot generate better actions in long-term learning, the composite output may be worse than that of the A3C. Therefore, a Q-learning upper layer is established to make adaptive decisions between the original A3C, the proposed inverse model and the proposed composite control. The state and action in Q-learning are defined as shown in Table 1.

In fact, this Q-learning layer can be regarded as the meta-learning layer. In Ref. [61], a meta-reward is defined by the difference between the current reward and the previous reward. In this study, the reward function in the Q-learning layer for $k$ th episode is defined as:

$$r_Q(k) = \xi_1 \cdot (R(k) - R(k-1)) + \xi_2 \cdot (R(k) - \max\{R(1), R(2), \cdots, R(k-1)\}), \tag{28}$$

where $r_Q(k)$ is the reward in $k$ th step; $\xi_1, \xi_2 \in (0, 1]$ are two setting weights. Besides, $\xi_2 > \xi_1$ is always set to give greater reward for the action with the best reward in history. Note that the policy in A3C is learned in multi-threads, independent Q-tables are created for the threads in A3C and updated based on Q-learning:

$$Q_{k+1}(s_k^Q, a_k^Q) = Q_k(s_k^Q, a_k^Q) + \alpha_Q\left[r_Q(k+1) + \gamma_Q \max Q_k(s_{k+1}^Q, a^Q) - Q_k(s_k^Q, a_k^Q)\right], \tag{29}$$

where $Q_k(s_k^Q, a_k^Q)$ is the Q value of each state-action pair in Table 1; $\alpha_Q$ and $\gamma_Q$ are the learning rate and reward discount, respectively. Then the final composite learning method can be denoted in Algorithm 1. To build the LSTM networks for inverse control, a basic 3-layer network including an input layer, a hidden layer composed of LSTM units and a fully connected dense output layer is used in this study. The work flow of the composite learning is shown in Fig. 6.

## 5. Case study

The widely used KVLCC2 ship model [27] is adopted for multi-ship collision avoidance in the following case studies to verify the effectivenesses of the proposed composite learning method. Case studies are conducted from the following aspects for comprehensive verification: 1) Ship collision avoidance based on the basic A3C learning; 2) Comparisons between the proposed method and A3C learning; 3) Comparisons between the proposed method and traditional optimization method.

### 5.1. Set up of the A3C learning and the LSTM networks

The details of the networks in A3C and LSTM are described in this subsection. Throughout all the case studies, we build the networks in TensorFlow and run the tests in a 4-cores i5 CPU with 8 threads.

**Table 1**
The state and action in Q-learning.

| State $s^Q$ | | Action $a^Q$ | | | |
|---|---|---|---|---|---|
| | | 2 | 3 | 4 | |
| 1 | A3C | Remain state 1 | Jump to state 2 | Jump to state 3 | Jump to state 4 |
| 2 | IC[1] | Jump to state 1 | Remain state 2 | Jump to state 3 | Jump to state 4 |
| 3 | CL[2] | Jump to state 1 | Jump to state 2 | Remain state 3 | Jump to state 4 |
| 4 | Actor[3] | Jump to state 1 | Jump to state 2 | Jump to state 3 | Remain state 4 |

[1] IC: the inverse control method.
[2] CL: the composite learning method.
[3] Actor: directly output by the actor in A3C.

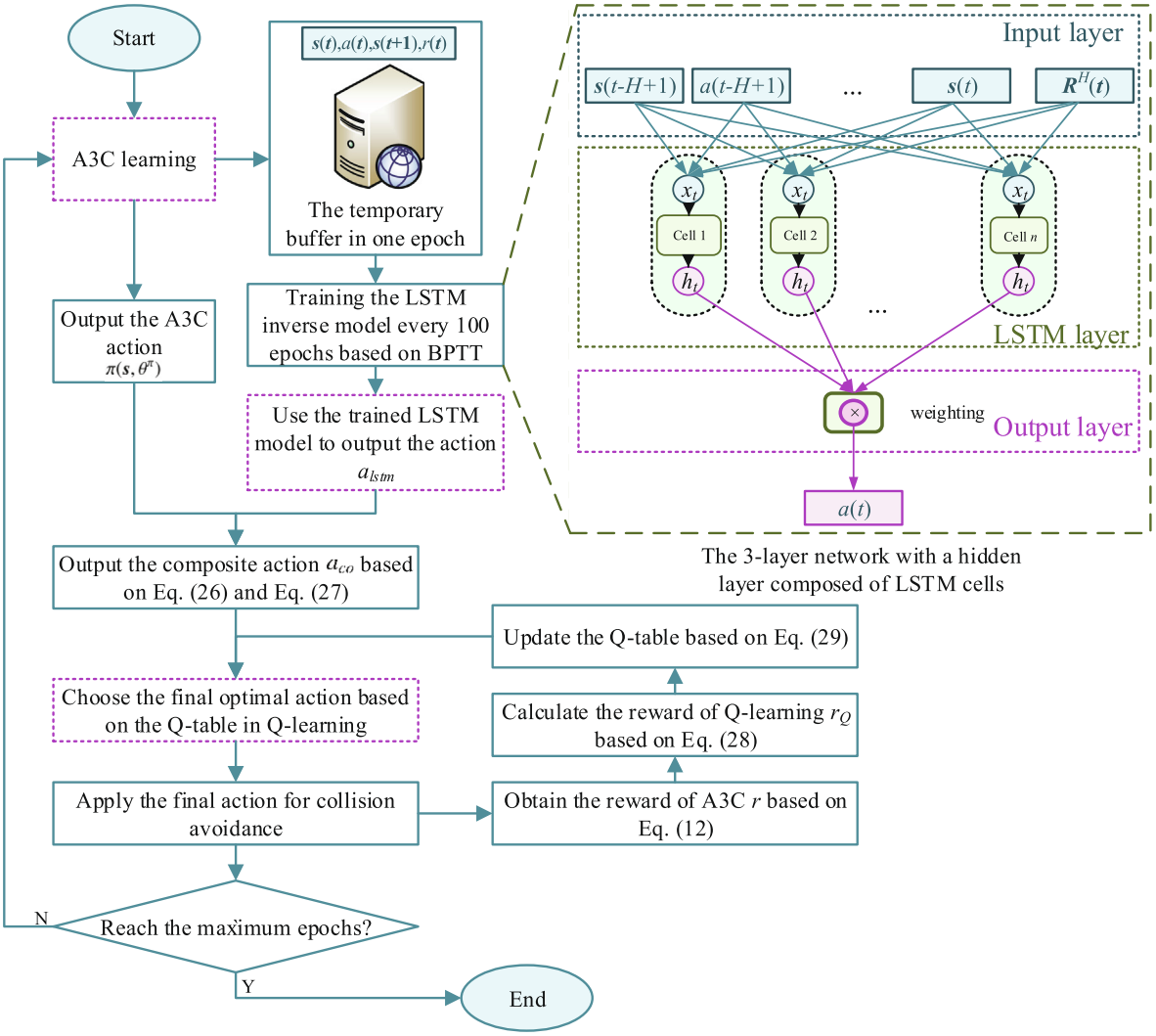


**Fig. 6.** The work flow of the composite learning.

*5.1.1. Networks in A3C learning*

Referring to the related works on actor-critic reinforcement learning [25,33], all actor and critic networks are built with 2 hidden layers with 32 units and *ReLU* activations in each layer.(1) The input and output of the actor and critic networks.

Since the actor in A3C is used to map the states and the actions, the input dimension of the actor network is set to 7 for the state vector $\boldsymbol{s}_t^i$ in Eq. (13). For better exploration, a normal distribution $\mathcal{N}(a,\sigma^2)$ is used to sample the rudder action $a=\delta$ with a standard deviation $\sigma$. Therefore, the outputs of the actor network are the rudder action $a$ and the standard deviation $\sigma$, which are produced by 2 independent dense layers with *tanh* activations.

The critic in A3C is used to estimate the state value $V_\theta(\boldsymbol{s}_t)$ with respect to the state $\boldsymbol{s}_t^i$, thus the input dimension of the critic is set to 7 and the output of the critic is the state value $V_\theta$ with 1 dimension, which are produced by a dense layer with *tanh* activations.(2) The learning parameters and process.

Referring to Refs. [25,33], the learning rates $\alpha_\pi$ and $\alpha$ are set as $1\times10^{-3}$ and $2\times10^{-3}$, respectively. The discount factor of the reward is set as $\gamma=0.99$ [25]. The updating of the actor and critic networks follows the A3C baseline in Ref. [33]:

Step 1: In each thread $j$ at time step $t$, the actor network $\pi^j_{\theta^\pi}$ output the rudder action $a_t^i$ with respect to the $i$th obstacle ship based on the state $\boldsymbol{s}_t^i$. After combining all actions $a_t^i$ based on Eq. (14), the final action $a_{op}$ is adopted for collision avoidance and the immediate rewards $r_t^i$ are obtained based on Eq. (15)∼(19). Then, the tuples $\{\boldsymbol{s}_t^i, a_{op}, r_t^i, \boldsymbol{s}_{t+1}^i\}$ are restored in a temporary buffer $B$.

Step 2: After every $n=5$ steps, the advantage values $A_\theta$ with respect to the states in $B$ are calculated by the temporal difference (TD) errors denoted in Eq. (27) as $A_\theta(s_t,a_t)=r_t+\gamma r_{t+1}+\cdots+\gamma^{n-1}r_{t+n-1}-V_\theta(s_t,a_t)$ where $V_\theta(s_t,a_t)$ is the state value output by the current critic network $V_\theta^j$.

Step 3: The gradients of the actor $\pi^j_{\theta^\pi}$ and the critic $V_\theta^j$ in Eqs. (25) and (28) are obtained by maximizing the partial derivatives of $J(\theta^\pi)$ and $L(\theta)$ with respect to the parameters $\theta^\pi$ and $\theta$, respectively.[2] Then, all gradients are applied to update the global actor network $\pi^g_{\theta^\pi}$ and critic network $V^g_\theta$.

Step 4: After updating, the actor and critic networks in each thread are drawn from the global actor and critic networks for the next exploration. The temporary buffer $B$ in each thread is also cleared.

Step 5: Run the loop in Step 1–4 until the maximum epoch is reached.

[2] We use the normal gradient calculation function in Tensorflow to calculate the gradients.

*5.1.2. Networks in LSTM*

As denoted in Section 4.1, the basic 3-layer LSTM network is used to establish the inverse controller. Considering that the inverse model is trained based on the learning results of A3C, the horizon $H$ in Eq. (30) is set the same as the step interval in A3C, i.e., $H = n = 5$. The mean square error is used as the loss function in LSTM and the *Adam* optimizer is used to update the LSTM parameters [16] based on back propagation through time (BPTT).

In case study 5.3, the trajectory of the A3C learning in the last epoch is used for training of the inverse model every 100 epochs to track the inverse dynamics.

*5.2. Case study 1: A3C based ship collision avoidance*

Since the weight $\lambda_{LOS}$ of the LOS tracking error $\hat{\psi}_{\mathrm{LOS}}$ in the reward function has influences on the re-sailing ability after collision avoidance, different $\lambda_{LOS}$ are set to analyze the learning results of A3C.

*5.2.1. Set up*

The weights of the rudder and collision risk are set equally as $r_\delta = r_s = 1$. To ensure safety, the reward for the collision risk should be larger than that for the re-sailing when the actual collision risk is larger than the threshold $f_{CRI} > f_{CR_{\min}}$. Therefore, the weight $\lambda_{LOS}$ is set based on the minimum risk threshold as $r_{LOS} = r_s \cdot \frac{f^2_{CRI_{\min}}}{\pi^2}$, where $\pi$ is the maximum LOS tracking error and $f_{CRI_{\min}}$ is the minimum risk threshold. Since the collision risk $f_{CRI} \in [0, 1]$, we set the empirical risk threshold range as $f_{CRI_{\min}} \in [0, 0.5]$. Then, the range of $\lambda_{LOS}$ is set as $\lambda_{LOS} \in [0, 0.05]$ for A3C learning.

Ship collision avoidance simulations with 4 obstacle ships are conducted. In order to form the multi-ship encounter situation, the initial state of different obstacle ships are set based on Eq. (30) to ensure the collisions if no measures are taken.

$$\begin{aligned} x^i_{T0} &= x_0 + R^i \sin(\psi_0) + R^i \sin(\psi_0 + \theta^i), y^i_{T0} \\ &= y_0 + R^i \cos(\psi_0) + R^i \cos(\psi_0 + \theta^i), \psi^i_{T0} = \psi_0 + \theta^i + \pi, u^i_{T0} \\ &= u_0 = U, v^i_{T0} = v_0 = 0, r^i_{T0} = r_0 = 0, \end{aligned} \tag{30}$$

where $[x^i_{T0}, y^i_{T0}, \psi^i_{T0}, u^i_{T0}, v^i_{T0}, r^i_{T0}]$ is the initial state of the $i$th obstacle ship, $R^i$ is the set distance before the collision between the own ship and the $i$th obstacle ship, and $\theta^i$ is the set collision angle of the $i$th obstacle ship. By setting different $R^i$ and $\theta^i$, the obstacle ships will have collisions with the own ship in sequences if no avoidance measures are taken. A head-on ship with $R^1 = 60$ m, $\theta^1 = 0°$, two starboard crossing ships with $R^2 = 70$ m, $\theta^2 = 30°, R^3 = 80$ m, $\theta^3 = 60°$ and one larboard crossing ship with $R^4 = 90$ m, $\theta^4 = 325°$ are set in this case.

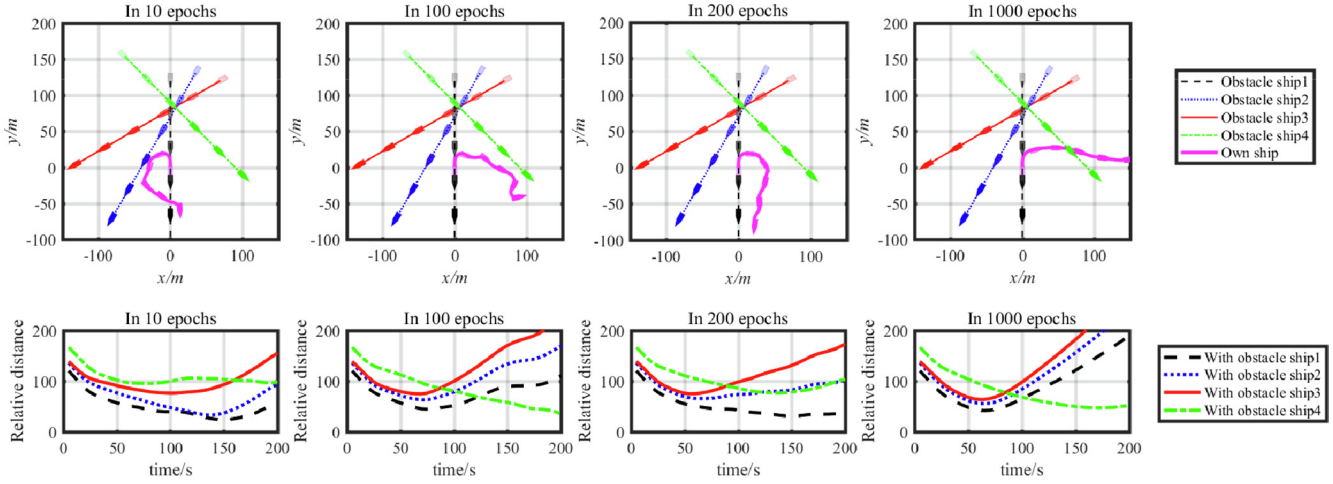


**Fig. 7.** Collision avoidance results of A3C without $\lambda_{LOS}$ in reward function.

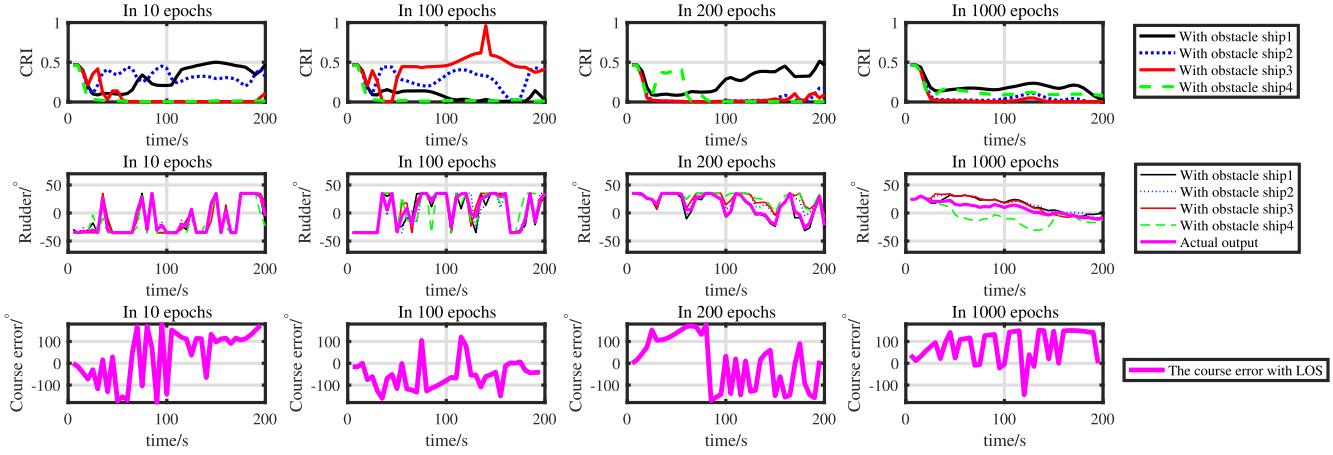


**Fig. 8.** The CRI, rudder and $\hat{\psi}_{\mathrm{LOS}}$ results without $\lambda_{LOS}$ in reward function.

*5.2.2. Results*

The distances $R^i_{T\min}$ and the lateral error $D_{la}$ results with different $\lambda_{LOS}$ are shown in Fig. 13, and the detailed collision avoidance trajectories and relative distances in 10, 100, 200 and 1000 epochs with $\lambda_{LOS} = 0, 0.02$ and 0.05 are shown in Figs. 7, 9 and 11 for examples. The corresponding CRIs, rudder angle and the heading error $\hat{\psi}_{\text{LOS}}$ in collision avoidance are shown in Figs. 8, 10 and 12, respectively. The calculated mean CRIs, the minimum distances $R^i_{T\min}$, the heading error $\hat{\psi}_{\text{LOS}}$ and lateral error $D_{la}$ of the own ship with $\lambda_{LOS} = 0, 0.02$ and 0.05 are shown in Table 2.

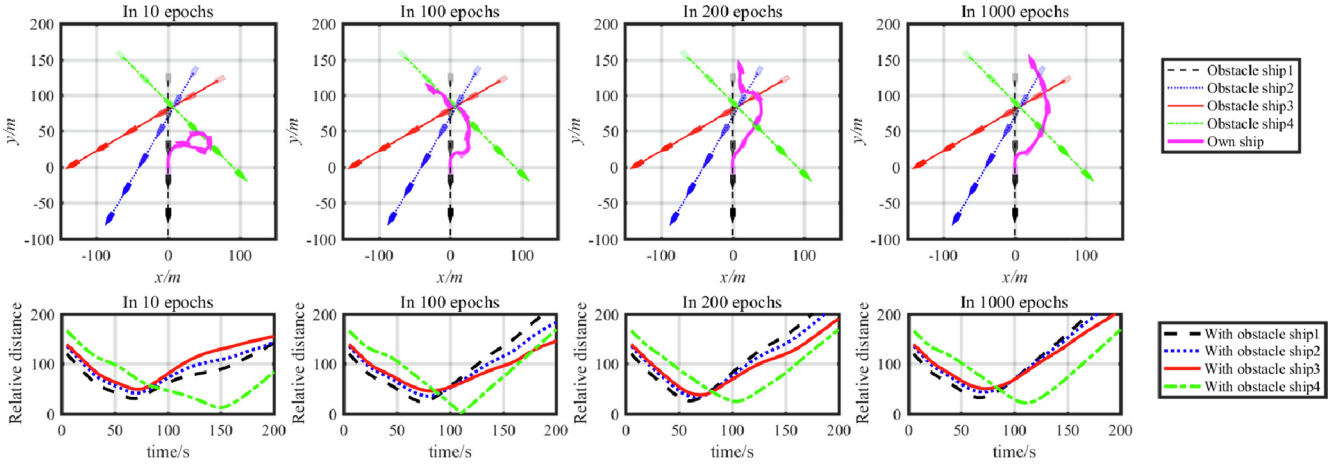


**Fig. 9.** Collision avoidance results of A3C with $\lambda_{LOS} = 0.02$ in reward function.

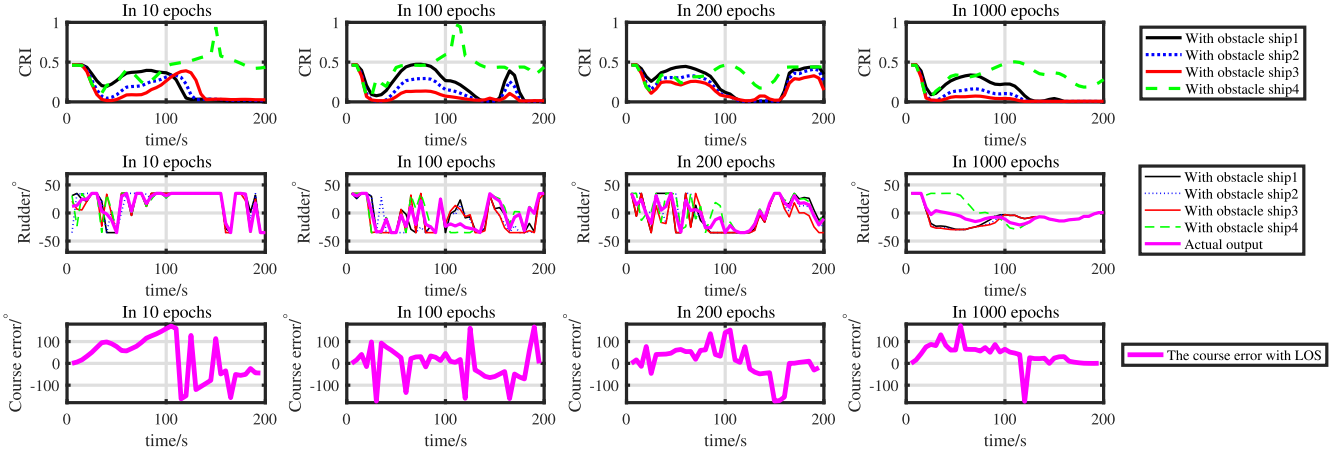


**Fig. 10.** The CRI, rudder and $\hat{\psi}_{\text{LOS}}$ results with $\lambda_{LOS} = 0.02$ in reward function.

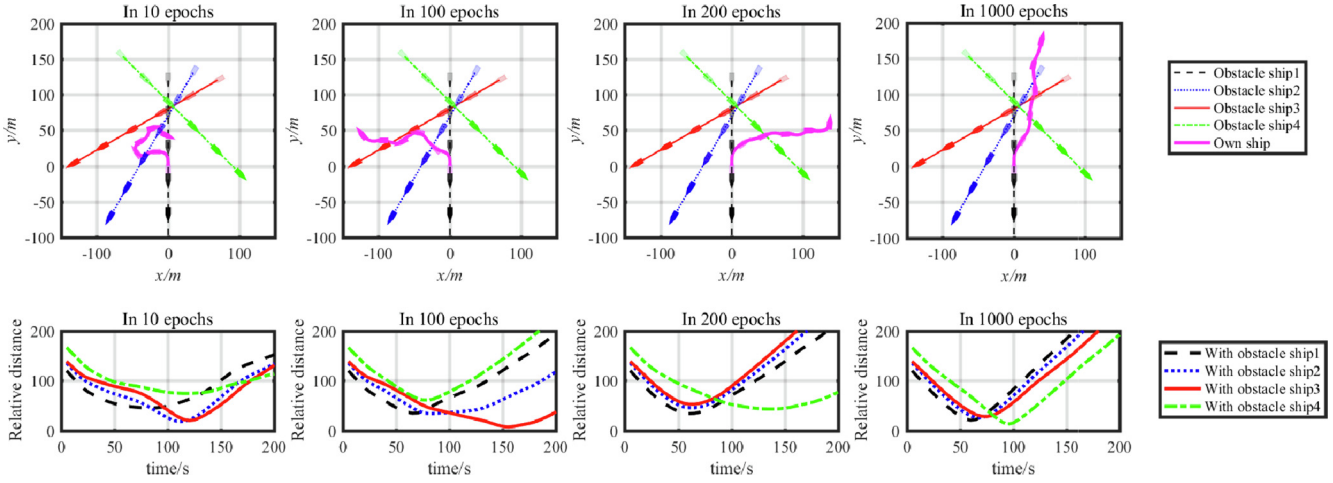


**Fig. 11.** Collision avoidance results of A3C with $\lambda_{LOS} = 0.05$ in reward function.

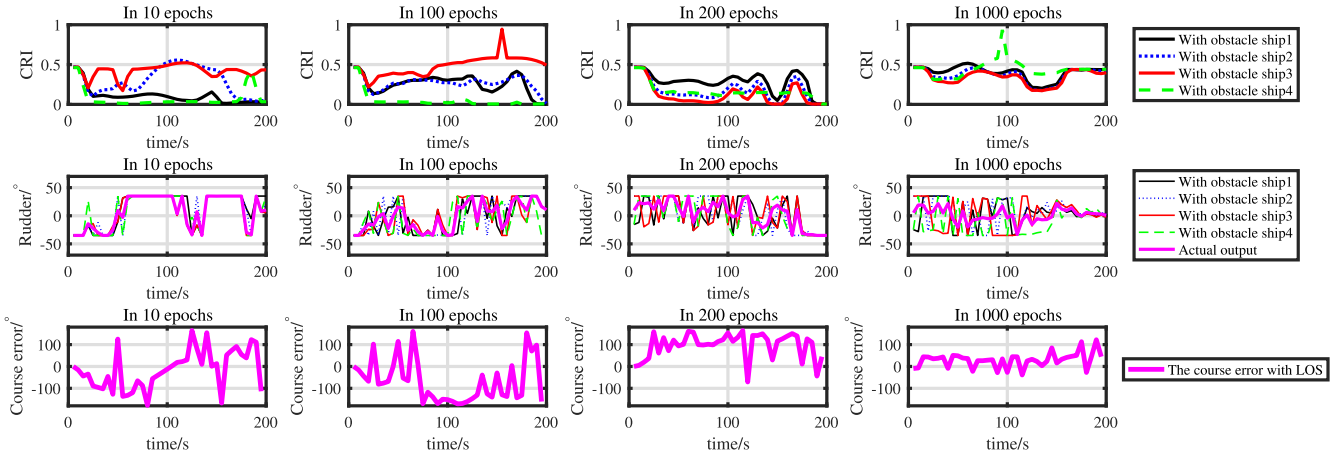


**Fig. 12.** The CRI, rudder and $\hat{\psi}_{\mathrm{LOS}}$ results with $\lambda_{LOS} = 0.05$ in reward function.

**Table 2**
The CRIs and $R^i_{T\,\min}$ between different obstacle ships and the $\hat{\psi}_{\mathrm{LOS}}$ and $D_{la}$ results with different $\lambda_{LOS}$.

| | Ship 1 | | | Ship 2 | | | Ship 3 | | | Ship 4 | | | $\hat{\psi}_{\mathrm{LOS}}$ | $D_{la}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $CRI_{mean}$ | $CRI_{\min}$ | $R_{T\,\min}$ | $CRI_{mean}$ | $CRI_{\min}$ | $R_{T\,\min}$ | $CRI_{mean}$ | $CRI_{\min}$ | $R_{T\,\min}$ | $CRI_{mean}$ | $CRI_{\min}$ | $R_{T\,\min}$ | | |
| 0(without) | 0.195 | 0.023 | 41.645 | 0.074 | 0.005 | 55.227 | 0.050 | 0.004 | 63.453 | 0.134 | 0.064 | 46.944 | 15.503 | 68.389 |
| 0.02 | 0.158 | 0.004 | 32.494 | 0.086 | 0.005 | 42.865 | 0.058 | 0.007 | 50.363 | 0.327 | 0.055 | 21.319 | 13.333 | 30.214 |
| 0.05 | 0.401 | 0.202 | 20.744 | 0.368 | 0.175 | 25.672 | 0.343 | 0.172 | 28.070 | 0.444 | 0.312 | 12.888 | 24.779 | 22.376 |

The proposed collision avoidance strategy, the A3C learning performance and the designed reward function in A3C are both discussed as follows:

1) The collision avoidance strategy
Take the optimal learning results of A3C in Figs. 7–12 for example. When the LOS tracking error $\hat{\psi}_{\mathrm{LOS}}$ is not considered, the initial decision results for all obstacle ships are large starboard rudders. After avoiding ship 1, 2 and 3, the continues starboard steering increases the CRI with ship 4. Then, the decision result for ship 4 (the green dash line in Fig. 8) turns to a larboard steering, which makes the final weighting result gradually turns to a middle rudder. After considering the error $\hat{\psi}_{\mathrm{LOS}}$ properly (Figs. 9 and 10), the A3C can already output larboard rudder actions for re-sailing after avoiding obstacle ship 1, 2 and 3. Therefore, it can be concluded that the collision risk based weight method is suitable for multi-ship collision avoidance.
2) The A3C learning
It can be seen from Figs. 7–12 that the own ship cannot effectively avoid all the obstacle ships before 100 epochs due to the random exploration and the under-convergent policy in the early learning stage. Over time, the sum rewards in reinforcement learning process gradually increases and an effective collision avoidance policy can be obtained within 1000 epochs.
3) The designed reward function

It can be seen from the results in Table 2 and Fig. 13 that the error $\hat{\psi}_{\mathrm{LOS}}$ in the reward function has great influences on the actual and re-sailing results($R^i_{T\,\min}$ and $D_{la}$). When the error $\hat{\psi}_{\mathrm{LOS}}$ is not considered in the reward function (i.e., $\lambda_{LOS} = 0$), the optimal policy adopts large steering rudders to avoid all the obstacle ships, which obtain safe avoidance results. While large $\hat{\psi}_{\mathrm{LOS}}$ and $D_{la}$ are obtained as shown in Figs. 8 and 13. On the contrary, when $\lambda_{LOS} = 0.05$, the steering rudders in Fig. 11 are much more smaller than those in Fig. 7, which obtain smaller $\hat{\psi}_{\mathrm{LOS}}$ and $D_{la}$ results. While the CRIs are also higher (especially with ship 4) as shown in Figs. 12 and 13. When an appropriate $\lambda_{LOS}$ (e.g., 0.02) is taken, the optimal policy can effectively reduce the $\hat{\psi}_{\mathrm{LOS}}$ and $D_{la}$ with small impacts on the CRIs and $R^i_{T\,\min}$, which is shown in Fig. 13.

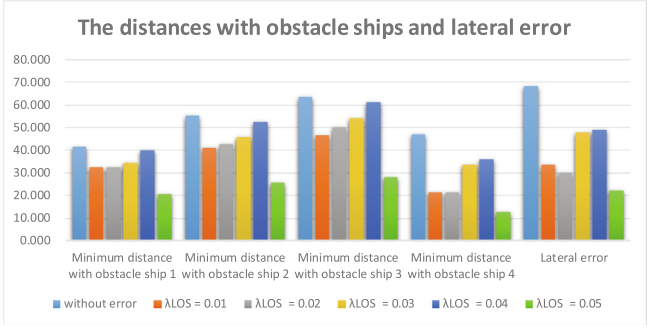


**Fig. 13.** Distances with obstacle ships and lateral errors.

Therefore, it can be concluded that an appropriate weight $\lambda_{LOS}$ of the LOS error $\hat{\psi}_{\mathrm{LOS}}$ in reward function can effectively reduce the lateral tracking error caused by excessive steering angle without much impacts on the collision avoidance safety.

Besides, the mean CRIs, the minimum distances $R^i_{T\,\min}$, the heading error $\hat{\psi}_{\mathrm{LOS}}$, lateral error $D_{la}$ and the obtained maximum sum rewards in 100, 300, 600 and 1000 epochs with $\lambda_{LOS} = 0.02$ are shown in Table 3. With the increase of the maximum sum rewards, the minimum distances between different ships $R^i_{T\,\min}$ are increased and the maximum lateral error $D_{la}$ is controlled at about 30 m. Therefore, the maximum sum reward can be used as the indicator to evaluate the performance of the A3C learning in collision avoidance.

**Table 3**
The CRIs and $R^i_{T\min}$ between different obstacle ships, the $\hat{\psi}_{\mathrm{LOS}}$, $D_{la}$ and sum rewards in different epochs.

| Epochs | Ship 1 | | Ship 2 | | Ship 3 | | Ship 4 | | $\hat{\psi}_{\mathrm{LOS}}$ | $D_{la}$ | Rewards |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | $CRI_{mean}$ | $R_{T\min}$ | $CRI_{mean}$ | $R_{T\min}$ | $CRI_{mean}$ | $R_{T\min}$ | $CRI_{mean}$ | $R_{T\min}$ | | | |
| 100 | 0.224 | 23.885 | 0.138 | 35.148 | 0.085 | 45.572 | 0.445 | 2.112 | 12.247 | 15.664 | −56.009 |
| 300 | 0.267 | 21.169 | 0.227 | 27.458 | 0.201 | 31.530 | 0.292 | 24.984 | 15.334 | 30.610 | −35.729 |
| 600 | 0.176 | 35.416 | 0.096 | 45.983 | 0.064 | 52.675 | 0.209 | 36.418 | 18.530 | 51.160 | −25.710 |
| 1000 | 0.158 | 32.494 | 0.086 | 42.865 | 0.058 | 50.363 | 0.327 | 21.319 | 13.333 | 30.214 | −23.422 |

### 5.3. Case study 2: Comparison between the proposed method and A3C method

To comprehensively verify the performance of the proposed composite learning method, simulations are conducted from the following two aspects: 1) the effectiveness verification of the integrated inverse controller; 2) the comparison between the proposed method and the A3C learning.

#### 5.3.1. Verification of the inverse controller

(1) Set up

In this case, two different scenarios are defined to verify the inverse control performance as shown in Table 4. As mentioned in Section 4, we use a basic 3-layer LSTM network in this section. After fine-tuning, the cells number $N_{LSTM}$ in the hidden layer is set to 12. Then the trained LSTM network is used for inverse control, and the collision avoidance results are compared with the A3C method.

(2) Results

The inverse control results and the optimal A3C learning results in the setting two scenarios are shown in Figs. 14 and 15, respectively. Indicators, i.e., the CRIs and minimum distances with different obstacle ships $R^i_{T\min}$, the LOS tracking error $\hat{\psi}_{\mathrm{LOS}}$ and the lateral error $D_{la}$ are shown in Table 5. It can be seen that the inverse controller can obtain effective collision avoidance trajectories and indicators, which are similar to the A3C method. Since a time sequence model is used in LSTM, the rudder actions of the inverse control are smoother than those of the A3C, which are shown in Figs. 14 and 15.

**Table 4**
Initial $R^i$ and $\theta^i$ for different obstacle ships with inverse control.

| Ship | Scenario 1 | | | | Scenario 2 | | | |
|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| $\theta^i$/° | 5 | 35 | 65 | 330 | −5 | 325 | 295 | 40 |
| $R^i$/m | 60 | 70 | 80 | 90 | 60 | 70 | 80 | 90 |

#### 5.3.2. Comparison between the proposed method and the A3C learning

(1) Set up

In this case, the scenarios defined in Table 4 are still used for collision avoidance. Both the Q-learning factor $\alpha_Q$ and discount $\gamma_Q$ in the proposed composite learning are set as 0.8. The original A3C is used for comparisons, and the maximum sum reward is taken as the indicator of the learning performance. To reduce contingency, repeated tests are conducted for calculating the average sum rewards.

(2) Results

The average sum rewards of the composite learning and A3C in two scenarios are shown in Figs. 16 and 17, respectively. The collision avoidance results of the composite learning and A3C after 10, 20, 200 and 1000 epochs are shown in Figs. 18 and 20. Moreover, the relative distances with obstacle ships and the heading error $\hat{\psi}_{\mathrm{LOS}}$ after 10, 100, 200 and 1000 epochs are shown in Figs. 19 and 21.

It can be seen that both the proposed composite learning method and A3C method can realize effective collision avoidance results after 1000 epochs. The learning efficiency and final learning results of the proposed method and A3C are discussed as follows:

1) The learning efficiency

It can be seen from Figs. 16, 17 and Table 6 that the proposed composite learning method can obviously obtain higher sum

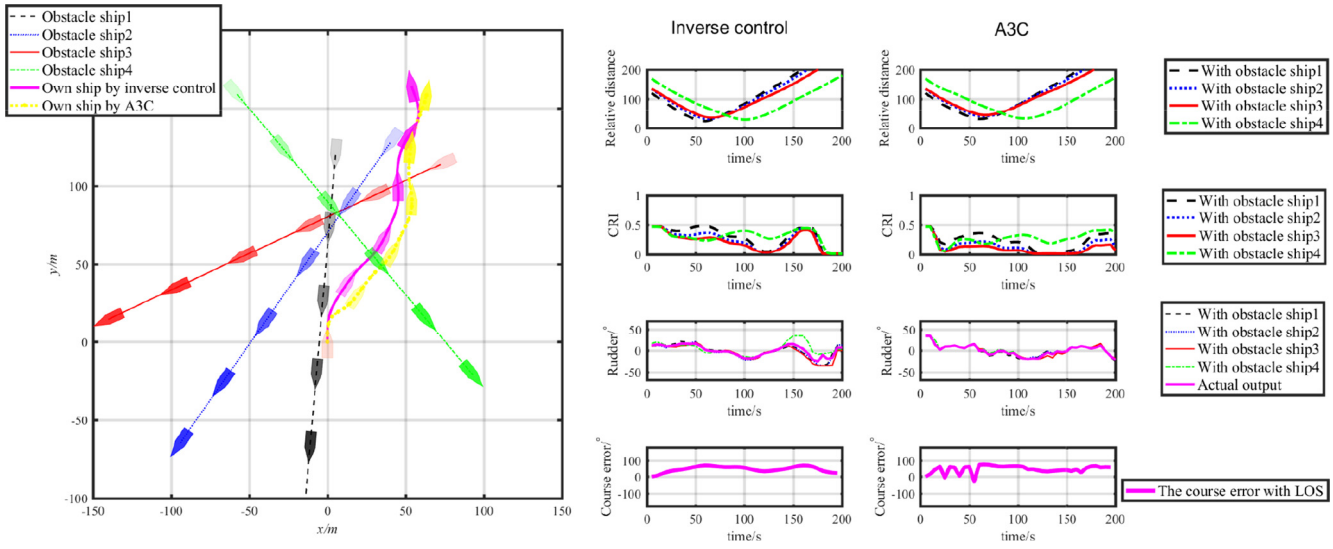


**Fig. 14.** The collision avoidance results in Scenario 1 based on LSTM inverse control.

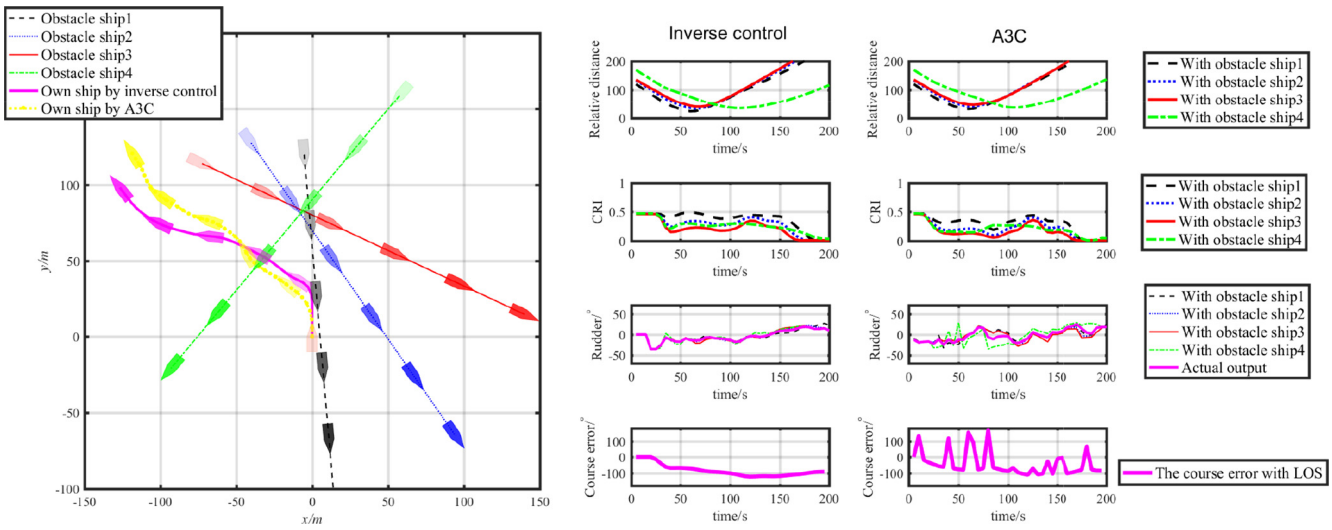


**Fig. 15.** The collision avoidance results in Scenario 2 based on LSTM inverse control.

**Table 5**
The CRIs and $R^i_{T\min}$ between different obstacle ships and the $\dot{\psi}_{\text{LOS}}$ and $D_{la}$ results of the inverse control and A3C.

| Scenario | | Ship 1 | | Ship 2 | | Ship 3 | | Ship 4 | | $\dot{\psi}_{\text{LOS}}$ | $D_{la}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $CRI_{mean}$ | $R_{T\min}$ | $CRI_{mean}$ | $R_{T\min}$ | $CRI_{mean}$ | $R_{T\min}$ | $CRI_{mean}$ | $R_{T\min}$ | | |
| 1 | IC[1] | 2.94E−01 | 2.37E+01 | 2.43E−01 | 3.12E+01 | 2.11E−01 | 3.60E+01 | 3.02E−01 | 2.87E+01 | 7.29E−01 | 3.50E+01 |
| | A3C | 2.20E−01 | 3.12E+01 | 1.40E−01 | 3.97E+01 | 9.95E−02 | 4.51E+01 | 2.72E−01 | 3.28E+01 | 7.96E−01 | 3.96E+01 |
| 2 | IC[1] | 3.20E−01 | 2.83E+01 | 2.20E−01 | 3.80E+01 | 1.58E−01 | 4.50E+01 | 1.96E−01 | 4.00E+01 | 2.91E+00 | 6.02E+01 |
| | A3C | 2.87E−01 | 3.13E+01 | 2.04E−01 | 4.08E+01 | 1.55E−01 | 4.62E+01 | 1.96E−01 | 3.65E+01 | 2.31E+00 | 5.73E+01 |

[1] IC: the inverse control method,

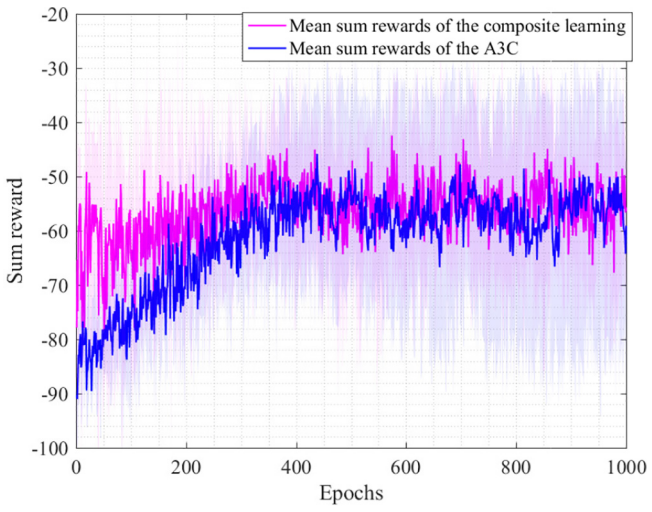


**Fig. 16.** The sum rewards of the composite learning and A3C in scenario 1.

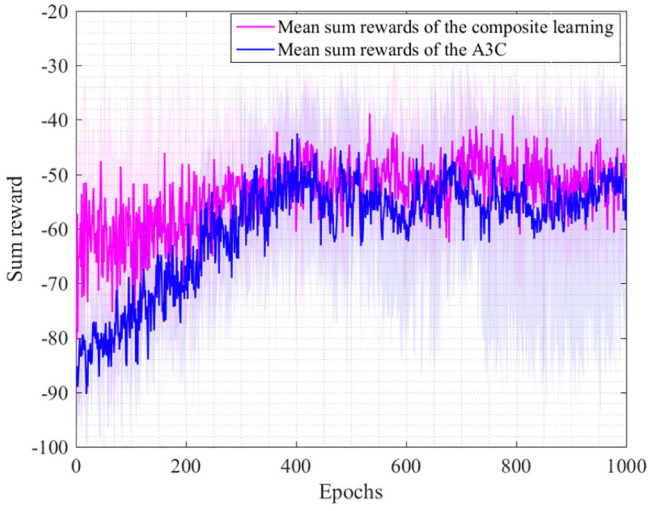


**Fig. 17.** The sum rewards of the composite learning and A3C in scenario 2.

rewards than the original A3C method in the early stage of learning (before 200 epochs). With the rapid increase of the sum rewards, the collision avoidance trajectories in two scenarios obtained by the proposed method become stable after 200 epochs, while the trajectories obtained by the original A3C are still tortuous. Therefore, the proposed composite learning method has higher learning efficiency than the original A3C method.

2) The final learning results

Since the target of reinforcement learning is to obtain the sum rewards as large as possible, the average values of the obtained maximum sum rewards in repeated tests are compared, which are shown in Table 6. In scenario 1, the average reward of the composite learning method is −42.9, which is larger than that of the original A3C learning, i.e., −46.4. In scenario 2, the similar results can be obtained. Therefore, the proposed composite learning method can obtain higher maximum sum rewards and performs better than the original A3C learning.

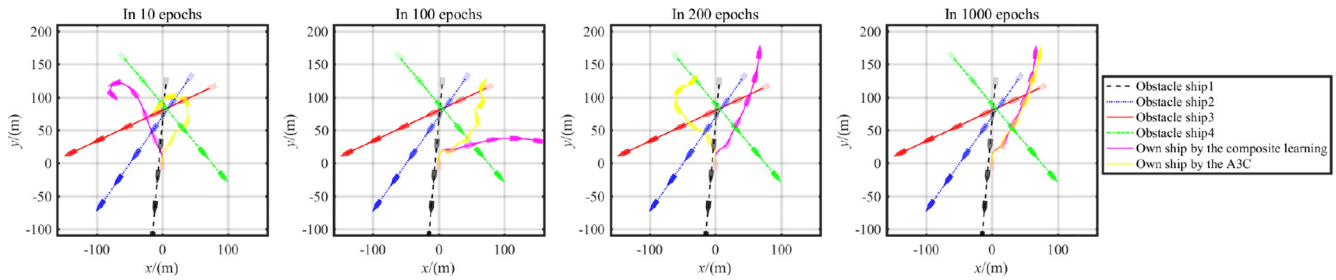


**Fig. 18.** The collision avoidance results of composite learning and A3C in scenario 1.

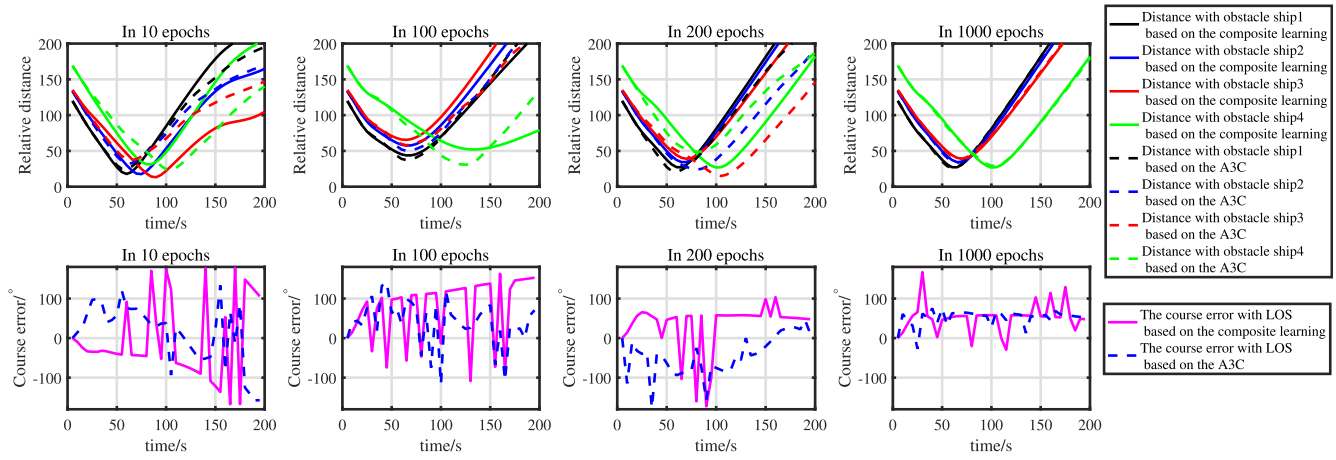


**Fig. 19.** The relative distances with obstacle ships and the heading error $\hat{\psi}_{LOS}$ in scenario 1.

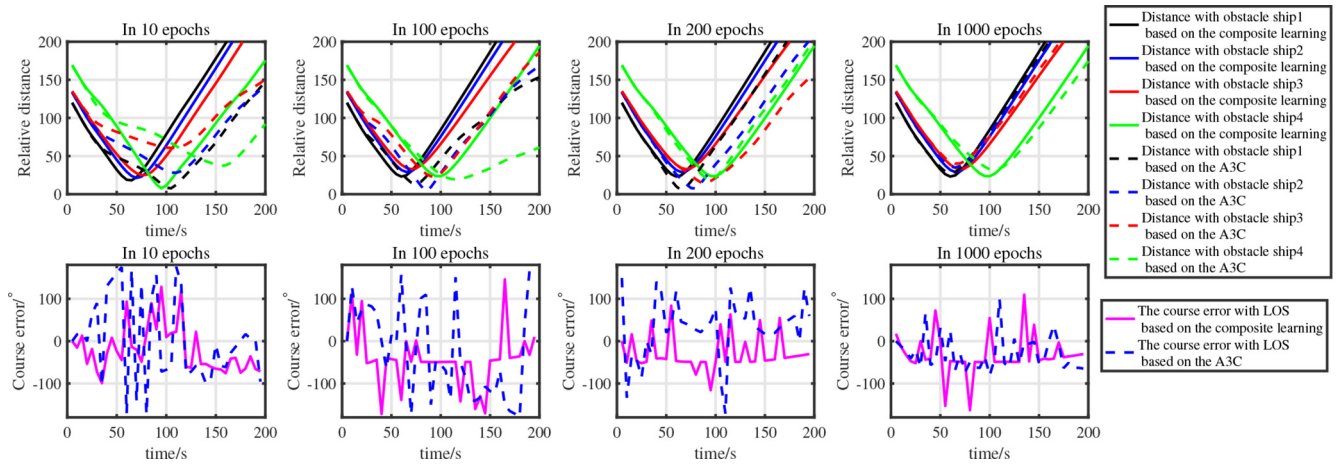


**Fig. 20.** The collision avoidance results of composite learning and A3C in scenario 2.

### *5.4. Case study 3: Comparisons between the proposed method and traditional method*

Since the proposed method is derived from the A3C reinforcement learning, which has greater potentials than the traditional method with limited perception. In this section, the proposed method is compared with a traditional optimization-based method. Recently, the swarm intelligence (SI) and evolutionary algorithms (EA) are widely used in ship collision avoidance, including the genetic algorithm [21,52], the ant colony algorithm [23,51] and especially the PSO algorithm [32,18,28,7], we apply the PSO to optimize the same reward function denoted in Eq. (19) as the traditional method for comparisons.

#### *5.4.1. Set up*

The perception range of the own ship is limited to 70 m, i.e., the obstacle ships located out of the perception range cannot be observed. The distances before collisions with obstacle ship 2, 3 and 4 in scenario 1 are enlarged to 80 m, 100 m, and 120 m to verify the performance of the proposed method in this case.

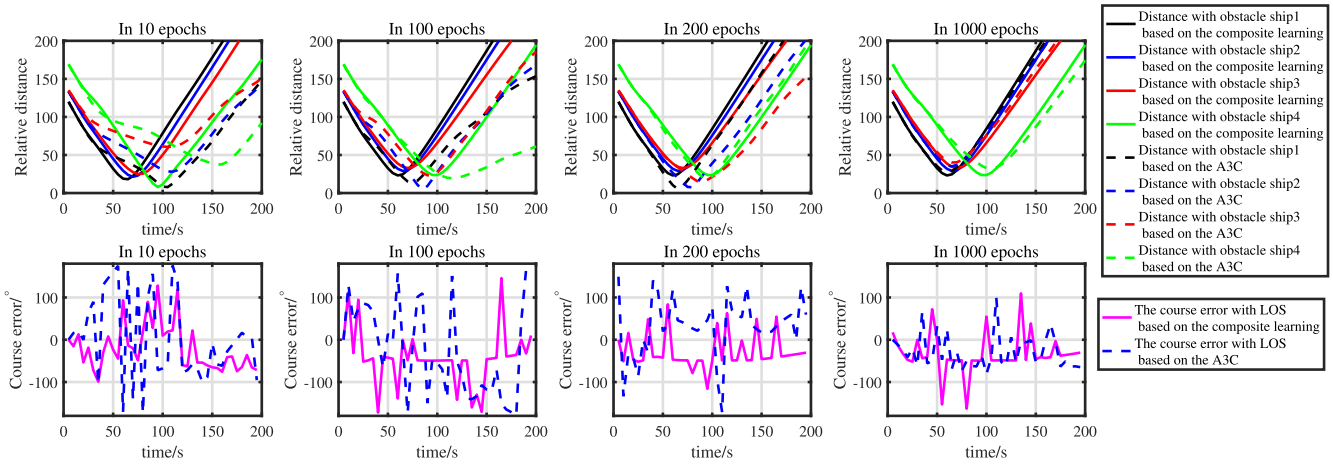


**Fig. 21.** The relative distances with obstacle ships and the heading error $\dot{\psi}_{LOS}$ in scenario 2.

**Table 6**
The maximum values of the sum rewards after 10, 20, 200 and 1000 epochs.

| epochs | Scenario 1 | | | | Scenario 2 | | | |
|---|---|---|---|---|---|---|---|---|
| | 10 | 100 | 200 | 1000 | 10 | 100 | 200 | 1000 |
| The propose method | −57.203 | −50.691 | −50.691 | −42.903 | −51.13 | −46.03 | −46.03 | −38.915 |
| A3C | −80.543 | −72.195 | −61.829 | −46.37 | −80.61 | −67.801 | −63.356 | −42.53 |

Moreover, we also consider the limited states including only the relative distance $R_T$ and relative position angle $\theta_T$, which can be easily detected by ship-side sensors (such as the radar). With the limited states, the risk model in Eq. (5) is simplified as:

$$f_{CRI} = \boldsymbol{\lambda}_{CRI}\boldsymbol{u}_{CRI}, \boldsymbol{\lambda}_{CRI} = [\lambda_{R_T} \quad \lambda_{\theta_T}], \quad \boldsymbol{u}_{CRI} = [u_{R_T} \quad u_{\theta_T}]^{\mathrm{T}}, \tag{31}$$

and the weights $u_{R_T}$ and $u_{\theta_T}$ are scaled equally as $u_{R_T} = 0.82$, $u_{R_T} = 0.18$. Then the state space in the proposed method and PSO in Eq. (6) is simplified as:

$$\boldsymbol{s}_t^i = \left[R_T^i(t) \quad \theta_T^i(t) \quad f_{CRI}^i(t)\right]. \tag{32}$$

Therefore, the input dimensions of the actor and critic networks in the proposed method are reduced to 3, other parameters remain the same as denoted in Section 5.1.

*5.4.2. Results*

To simplify the labels, the obstacle ships are represented as S1–S4, and the own ship is represented as OS. The final collision avoidance results of the proposed method and PSO-based method with all states in Eq. (13) and limited states in Eq. (32) are shown in Figs. 22 and 23, respectively. The light pink and light blue solid circles in Figs. 22 and 23 show the perception range of the own ship based on the proposed method and PSO, respectively.

1) With all states and limited perception

It can be seen from Fig. 22 that the CRIs between the own ship and the obstacle ships are 0 in the beginning since all the obstacle ships are out of the perception range of the own ship. After the own ship has detected the collision risks, the proposed method generate a small starboard steering to avoid all collisions successfully, which is similar to the results in Fig. 14. While the PSO-based method

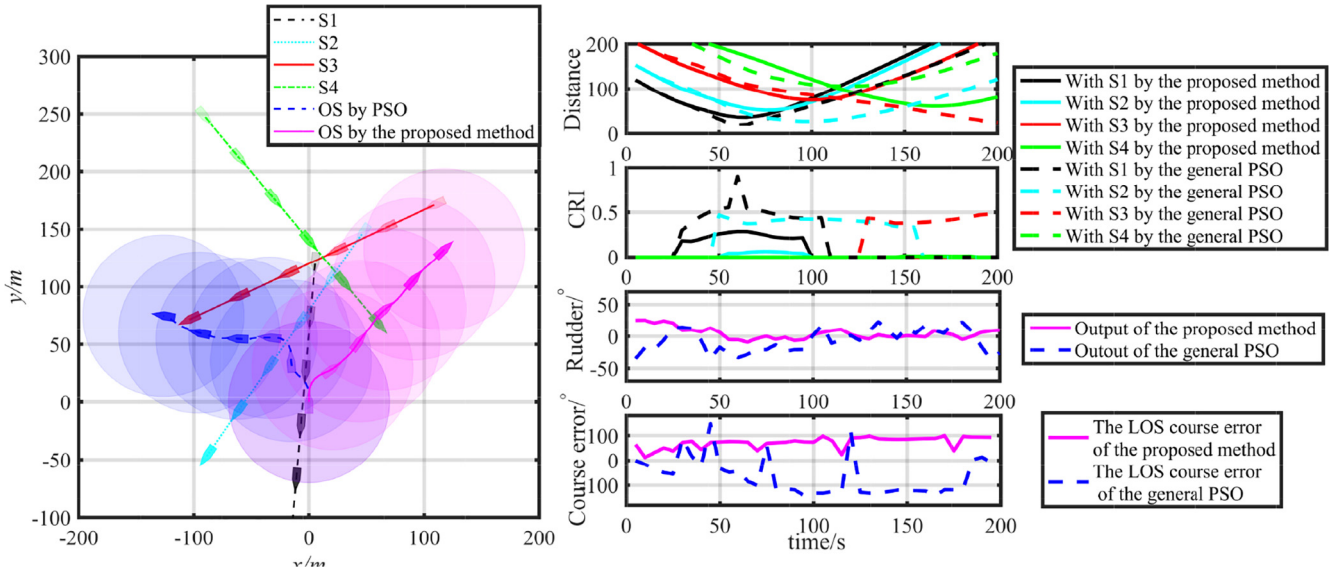


**Fig. 22.** The collision avoidance results of the proposed method and traditional method with all states under limited perception.

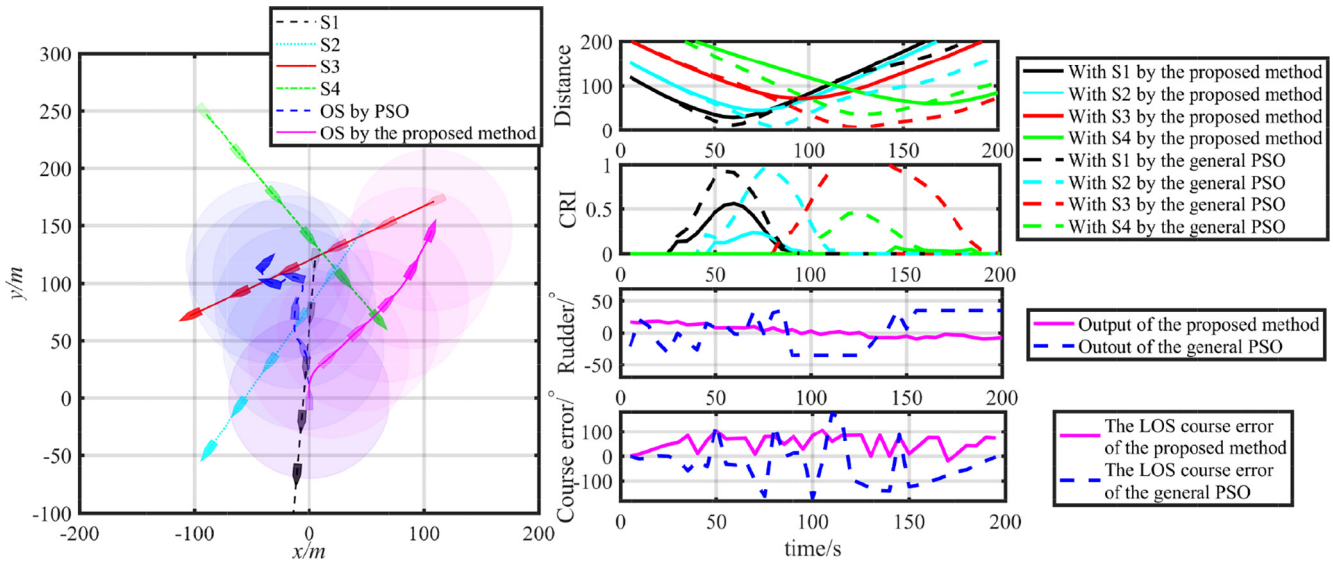


**Fig. 23.** The collision avoidance results of the proposed method and traditional method with limited states under limited perception.

**Table 7**
The CRIs, $R^i_{T\min}$ between different obstacle ships, the $\hat{\psi}_{\text{LOS}}$ and $D_{la}$ results and the rewards of the proposed method and PSO-based traditional method.

| | Methods | Ship 1 | | Ship 2 | | Ship 3 | | Ship 4 | | $\hat{\psi}_{\text{LOS}}$ | $D_{la}$ | Reward |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $CRI_{mean}$ | $R_{T\min}$ | $CRI_{mean}$ | $R_{T\min}$ | $CRI_{mean}$ | $R_{T\min}$ | $CRI_{mean}$ | $R_{T\min}$ | | | |
| Full state | PSO | 0.422 | 18.367 | 0.364 | 26.496 | 0.400 | 23.555 | 0.060 | 105.293 | 13.040 | 53.763 | −50.614 |
| | The proposed method | 0.271 | 35.557 | 0.093 | 51.643 | 0.043 | 74.692 | 0.041 | 60.779 | 18.845 | 55.767 | −25.990 |
| Limited state | PSO | 0.216 | 11.974 | 0.247 | 8.370 | 0.447 | 7.200 | 0.187 | 36.300 | 12.409 | 17.068 | −52.882 |
| | The propose method | 0.136 | 29.587 | 0.089 | 45.203 | 0.093 | 71.380 | 0.072 | 60.493 | 17.194 | 54.148 | −25.013 |

generates a larboard steering, which avoids S1 and S2 but increases the CRI between S3. From the final CRIs and minimum distance results in Table 7, it can be also seen that the proposed method obtains lower CRIs and larger minimum distances than the PSO method for all obstacle ships, which results in a higher final reward.

2) With limited states and limited perception

Due to the simplification of the state space, the outputs and CRIs of the proposed method in Fig. 23 change more smoothly than those in Fig. 22. It can be seen from Fig. 23 that the results of the proposed method with limited states are similar to those with all states. While the PSO-based method is fail to generate stable steering actions after avoiding S1 and S2, which results in a larger CRI with S3 as shown in Fig. 23 and Table 7.

In summary, the proposed composite learning method also outperforms the traditional PSO optimization-based method under limited perception, which has the advantage of learning a better policy in an uncertain environment.

## 6. Conclusions and future research

To realize efficient learning of multi-ship collision avoidance policy, a model based for model-free (MB-MF) composite learning method is proposed in this study. The main originality of the proposed method is to use the LSTM model-based controller to accelerate the model-free A3C learning by adaptive Q-learning decisions in the entire learning process. The following conclusions are drawn from the simulation experiments using the model of KVLCC2 ship:

(1) In terms of the application of A3C learning method, with the designed collision avoidance strategy and reward function, the optimal multi-ship collision avoidance policy can be learned by A3C effectively.

(2) In terms of the proposed composite learning method, with the Q-learning based combination of the A3C and the proposed inverse controller, the composite learning method learns the ship collision avoidance policy faster and better than the original model-free A3C method and traditional optimization method.

Future works can be carried out on the following aspects:

(1) In this study, only the rudder is taken as the action of collision avoidance for ships sailing at sea. In future research, both the engine speed and the rudder will be considered as the actions in more complex environment.
(2) Similar to most traditional studies, only the own ship is taken as the agent for the ship collision avoidance learning. Multi-agent learning will be considered in future research.
(3) This study focuses on applying the A3C algorithm and MB-MF idea on ship collision avoidance to achieve efficient learning. The final learning performance is related to several design parameters, e.g., the learning factor in the Q-learning layer. Depth analysis on the main parameters that affect the efficiency of composite learning will be conducted in future research.
(4) The weighting method is used in the reward function to consider different rewards in this study, more convincing method, e.g., the multi-object reinforcement learning can be used in future research.

## 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.

## CRediT authorship contribution statement

**Shuo Xie:** Conceptualization, Methodology, Writing - original draft. **Xiumin Chu:** Funding acquisition, Supervision. **Mao Zheng:** Software, Validation. **Chenguang Liu:** Writing - review & editing.

## Acknowledgements

This research is supported by the National Key Research and Development Program of China (No. 2018YFB1600400), the Fundamental Research Funds for the Central Universities (WUT:203144003), the National Natural Science Foundation of China (No. 51709220), the Open Project Program of Fujian University Engineering Research Center of Marine Intelligent Ship Equipment (No. 322031010602), the Project of Science and Technology Bureau of Fuzhou (No. 2018-G-92) and the Key Project of Science and Technology of Wuhan (No. 201701021010132).

## References

[1] M.A. Abkowitz, Measurement of hydrodynamic characteristics from ship maneuvering trials by system identification, Maneuverability (1980).
[2] J.-H. Ahn, K.-P. Rhee, Y.-J. You, A study on the collision avoidance of a ship using neural networks and fuzzy logic, Applied Ocean Research 37 (2012) 162–173.
[3] T. Awad, M.A. elfatah Elgohary, T.E. Mohamed, Ship roll damping via direct inverse neural network control system, Alexandria Engineering Journal 57 (4) (2018) 2951–2960..
[4] 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 (2) (2012) 267–283.
[5] Y. Chebotar, K. Hausman, M. Zhang, G. Sukhatme, S. Schaal, S. Levine, Combining model-based and model-free updates for trajectory-centric reinforcement learning, in: Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, 2017, pp. 703–711..
[6] D. Chen, C. Dai, X. Wan, J. Mou, A research on AIS-based embedded system for ship collision avoidance, in: 2015 International Conference on Transportation Information and Safety (ICTIS), IEEE, 2015, pp. 512–517.
[7] L. Chen, L. Huang, Ship collision avoidance path planning by pso based on maneuvering equation, in: Future Wireless Networks and Information Systems, Springer, 2012, pp. 675–682.
[8] L. Cheng, C. Liu, B. Yan, Improved hierarchical A-star algorithm for optimal parking path planning of the large parking lot, in: 2014 IEEE International Conference on Information and Automation (ICIA), IEEE, 2014, pp. 695–698..
[9] Y. Cheng, W. Zhang, Concise deep reinforcement learning obstacle avoidance for underactuated unmanned marine vessels, Neurocomputing 272 (2018) 63–73.
[10] M. Duguleana, G. Mogan, Neural networks based reinforcement learning for mobile robots obstacle avoidance, Expert Systems with Applications 62 (2016) 104–115.
[11] E. Fernandes, P. Costa, J. Lima, G. Veiga, Towards an orientation enhanced A-star algorithm for robotic navigation, 2015 IEEE International Conference on Industrial Technology (ICIT) IEEE (2015) 3320–3325.
[12] K. Hara, A. Hammer, A safe way of collision avoidance maneuver based on maneuvering standard using fuzzy reasoning model, in: Proceedings of the 1993 International Conference on Marine Simulation & Ship Manoeuvrability, 1993, pp. 163–170.
[13] K. Hasegawa, A. Kouzuki, T. Muramatsu, H. Komine, Y. Watabe, Ship auto-navigation fuzzy expert system (safes), Journal of the Society of Naval Architects of Japan 1989 (166) (1989) 445–452.
[14] G. Hernandez-Mejia, A.Y. Alanis, E.A. Hernandez-Vargas, Neural inverse optimal control for discrete-time impulsive systems, Neurocomputing 314 (2018) 101–108.
[15] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Computation 9 (8) (1997) 1735–1780.
[16] Y. Hu, X. Sun, X. Nie, Y. Li, L. Liu, An enhanced lstm for trend following of time series, IEEE Access 7 (2019) 34020–34030.
[17] M. Inaishi, H. Matsumura, Basic research on a collision avoidance system using neural networks, Journal of Navigation 112 (1992) 22–28.
[18] Y. Kang, W. Chen, D. Zhu, J. Wang, Q. Xie, Collision avoidance path planning for ships by particle swarm optimization, Journal of Marine Science and Technology 26 (6) (2018) 777–786.
[19] D.-H. Kim, S.-U. Lee, J.-H. Nam, Y. Furukawa, Determination of ship collision avoidance path using deep deterministic policy gradient algorithm, Journal of the Society of Naval Architects of Korea 56 (1) (2019) 58–65.
[20] D.S. Kim, Analysis of causes of collision caused by human error of captain and oow in ship collision accidents, Journal of the Ergonomics Society of Korea 37 (1) (2018) 1–13.
[21] H. Kim, S.H. Kim, M. Jeon, J.H. Kim, S. Song, K.J. Paik, A study on path optimization method of an unmanned surface vehicle under environmental loads using genetic algorithm, Ocean Engineering 142 (2017) 616–624.
[22] A. Lazarowska, Safe ship control method with the use of ant colony optimization, in: Solid State Phenomena, vol. 210, Trans Tech Publ, 2014, pp. 234–244..
[23] A. Lazarowska, Ship's trajectory planning for collision avoidance at sea based on ant colony optimisation, Journal of Navigation 68 (2) (2015) 291–307.
[24] 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..
[25] T.P. Lillicrap, J.J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, Continuous control with deep reinforcement learning, 2015, arXiv preprint arXiv:1509.02971..
[26] C. Liu, Q. Mao, X. Chu, S. Xie, An improved A-star algorithm considering water current, traffic separation and berthing for vessel path planning, Applied Sciences 9 (6) (2019) 1057–1074.
[27] J. Liu, F. Quadvlieg, R. Hekkenberg, Impacts of the rudder profile on manoeuvring performance of ships, Ocean Engineering 124 (2016) 226–240.
[28] L. Liu, D. He, M. Ying, T. Li, J. Li, Research on ships collision avoidance based on chaotic particle swarm optimization, in: Proceedings of the International Conference on Smart Vehicular, 2017, pp. 230–239.
[29] E.S. Low, P. Ong, K.C. Cheah, Solving the optimal path planning of a mobile robot using improved Q-learning, Robotics and Autonomous Systems 115 (2019) 143–161.
[30] W. Luo, X. Li, Measures to diminish the parameter drift in the modeling of ship manoeuvring using system identification, Applied Ocean Research 67 (2017) 9–20.
[31] H. Lyu, Y. Yin, Fast path planning for autonomous ships in restricted waters, Applied Sciences 8 (12) (2018) 2592–2616.
[32] Y. Ma, M. Hu, X. Yan, Multi-objective path planning for unmanned surface vehicle with currents effects, ISA Transactions 75 (2018) 137–156.
[33] V. Mnih, A.P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, K. Kavukcuoglu, Asynchronous methods for deep reinforcement learning, in: Proceedings of the International Conference on Machine Learning, 2016, pp. 1928–1937.
[34] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.
[35] W. Naeem, S.C. Henrique, L. Hu, A reactive COLREGs-compliant navigation strategy for autonomous maritime navigation, IFAC-PapersOnLine 49 (23) (2016) 207–213.
[36] A. Nagabandi, G. Kahn, R.S. Fearing, S. Levine, Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning, in: Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2018, pp. 7559–7566.
[37] J. ONeill, B. Pleydell-Bouverie, D. Dupret, J. Csicsvari, Play it again: reactivation of waking experience and memory, Trends in Neurosciences 33 (5) (2010) 220–229..
[38] Y.P. Pane, S.P. Nageshrao, J. Kober, R. Babuka, Reinforcement learning based compensation methods for robot manipulators, Engineering Applications of Artificial Intelligence 78 (2019) 236–247.
[39] L.P. Perera, J.P. Carvalho, C.G. Soares, Fuzzy logic based decision making system for collision avoidance of ocean navigation under critical collision conditions, Journal of Marine Science & Technology 16 (1) (2011) 84–99.
[40] L.P. Perera, V. Ferrari, F.P. Santos, M.A. Hinostroza, C.G. Soares, Experimental evaluations on ship autonomous navigation and collision avoidance by intelligent guidance, IEEE Journal of Oceanic Engineering 40 (2) (2014) 374–387.
[41] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, 2017, arXiv preprint arXiv:1707.06347..
[42] H. Shen, C. Guo, T. Li, An intelligent collision avoidance and navigation approach of unmanned surface vessel considering navigation experience and rules, Journal of Harbin Engineering University 39 (6) (2017) 1–7.
[43] H. Shen, H. Hashimoto, A. Matsuda, Y. Taniguchi, D. Terada, C. Guo, Automatic collision avoidance of multiple ships based on deep Q-learning, Applied Ocean Research 86 (2019) 268–288.
[44] D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al., Mastering the game of Go with deep neural networks and tree search, Nature 529 (7587) (2016) 484–489..
[45] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, M. Riedmiller, Deterministic policy gradient algorithms, 2014..
[46] U. Simsir, M. Bal, S. Ertugrul, Decision support system for collision avoidance of vessels, Applied Soft Computing Journal 25 (C) (2014) 369–378.
[47] Y. Singh, S. Sharma, R. Sutton, D. Hatton, A. Khan, A constrained A* approach towards optimal path planning for an unmanned surface vehicle in a maritime environment containing dynamic obstacles and ocean currents, Ocean Engineering 169 (2018) 187–201.
[48] R. Song, Y. Liu, R. Bucknall, Smoothed A* algorithm for practical unmanned surface vehicle path planning, Applied Ocean Research 83 (2019) 9–20.
[49] R. Szlapczynski, J. Szlapczynska, Review of ship safety domains: models and applications, Ocean Engineering 145 (2017) 277–289.
[50] Y. Tian, K. Zhang, J. Li, X. Lin, B. Yang, LSTM-based traffic flow prediction with missing data, Neurocomputing 318 (2018) 297–305.

[51] M.-C. Tsou, C.-K. Hsueh, The study of ship collision avoidance route planning by ant colony algorithm, Journal of Marine Science and Technology 18 (5) (2010) 746–756.
[52] M.-C. Tsou, S.-L. Kao, C.-M. Su, Decision support from genetic algorithms for ship collision avoidance route planning and alerts, The Journal of Navigation 63 (1) (2010) 167–182.
[53] H. Wang, J. Zhou, G. Zheng, Y. Liang, HAS: Hierarchical A-star algorithm for big map navigation in special areas. In: 2014 5th International Conference on Digital Home, IEEE, 2014, pp. 222–225..
[54] C.J. Watkins, P. Dayan, Q-learning, Machine learning 8 (3–4) (1992) 279–292.
[55] B. Widrow, Adaptive inverse control, in: Proceedings of the Adaptive Systems in Control and Signal Processing 1986, Elsevier, 1987, pp. 1–5..
[56] J. Woo, J. Park, C. Yu, N. Kim, Dynamic model identification of unmanned surface vehicles using deep learning network, Applied Ocean Research 78 (2018) 123–133.
[57] S. Xie, X. Chu, M. Zheng, C. Liu, Ship predictive collision avoidance method based on an improved beetle antennae search algorithm, Ocean Engineering 192 (2019), 106542.
[58] S. Xie, V. Garofano, X. Chu, R.R. Negenborn, Model predictive ship collision avoidance based on q-learning beetle swarm antenna search and neural networks, Ocean Engineering 193 (2019), 106609.
[59] H. Xu, N. Wang, H. Zhao, Z. Zheng, Deep reinforcement learning-based path planning of underactuated surface vessels, Cyber-Physical Systems 5 (1) (2019) 1–17.
[60] L. Xu, Study of ship collision avoidance based on optimal control, 2016..
[61] T. Xu, Q. Liu, L. Zhao, J. Peng, Learning to explore with meta-policy gradient, 2018, arXiv preprint arXiv:1803.05044..
[62] Y. Xue, D. Clelland, B.S. Lee, D. Han, Automatic simulation of ship navigation, Ocean Engineering 38 (17) (2011) 2290–2305.
[63] Y.Z. Xue, Y. Wei, Y. Qiao, The research on ship intelligence navigation in confined waters, Advanced Materials Research 442 (2012) 398–401.
[64] H. Yasukawa, Y. Yoshimura, Introduction of MMG standard method for ship maneuvering predictions, Journal of Marine Science and Technology 20 (1) (2015) 37–52.
[65] R. Zhang, X. Wang, K. Liu, X. Wu, T. Lu, C. Zhaohui, Ship collision avoidance using constrained deep reinforcement learning, in: Proceedings of the IEEE 2018 5th International Conference on Behavioral, Economic, and Socio-Cultural Computing (BESC), IEEE, 2019, pp. 115–120.
[66] J. Zhao, L. Lv, T. Fan, H. Wang, C. Li, P. Fu, Particle swarm optimization using elite opposition-based learning and application in wireless sensor network, Sensor Letters 12 (2) (2014) 404–408.
[67] 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.
[68] H. Zheng, R.R. Negenborn, G. Lodewijks, Closed-loop scheduling and control of waterborne agvs for energy-efficient inter terminal transport, Transportation Research Part E: Logistics and Transportation Review 105 (2017) 261–278.
[69] H. Zheng, R.R. Negenborn, G. Lodewijks, Fast ADMM for distributed model predictive control of cooperative waterborne AGVs, IEEE Transactions on Control Systems Technology 25 (4) (2017) 1406–1413.
[70] H. Zheng, R.R. Negenborn, G. Lodewijks, Robust distributed predictive control of waterborne agvsa cooperative and cost-effective approach, IEEE Transactions on Cybernetics 48 (8) (2017) 2449–2461.
[71] T. Zhou, J. Zhang, Analysis of commercial truck drivers potentially dangerous driving behaviors based on 11-month digital tachograph data and multilevel modeling approach, Accident Analysis & Prevention 132 (2019), 105256.

**Shuo Xie** is a PhD candidate of Wuhan University of Technology and National Engineering Research Center for Water Transport Safety. He received his master degree majoring in Transportation Engineering in the School of Energy and Power Engineering, Wuhan University of Technology in 2017. He has published more than 10 academic papers including 4 SCI journal papers. His research interests include ship model identification, ship control and collision avoidance.

**Xiumin Chu** is a professor in the National Engineering Research Center for Water Transport Safety, Wuhan University of Technology, Wuhan, China. He received the PhD degree (2002) and M.S. degree (1998) majoring in Automobile Application Engineering in Jilin University. He has published 2 books and more than 70 papers. His research topics include waterway transportation intelligence, smart ship, and ship motion simulation.

**Mao Zheng** is a senior engineer at National Engineering Research Center for Water Transportation Safety. He received his PhD degree majoring in Ship Engineering college, Harbin Engineering University in 2014. He has published more than 10 academic papers on ship hydrodynamics and machine learning. His current research interests include ship collision avoidance, ship maneuvering motions simulation and tests, etc.

**Chenguang Liu** is an assistant professor in the National Engineering Research Center for Water Transport Safety, Wuhan University of Technology, Wuhan, China. He received his M.S. degree and Ph.D. degree in the School of Energy and Power Engineering, Wuhan University of Technology, China in 2014 and 2017, respectively. He finished his post-doctoral research in Wuhan University in 2019. He has published more than 10 academic papers. His current research interests include ship intelligence, ship motion control, and model predictive control.