Ocean Engineering 260 (2022) 112038

Contents lists available at ScienceDirect

Ocean Engineering

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

A learning method for AUV collision avoidance through deep reinforcement learning☆

Jian Xu, Fei Huang *, Di Wu, Yunfei Cui, Zheping Yan, Xue Du

College of Intelligent Systems Science and Engineering, Harbin Engineering University, Harbin, 150001, china

ARTICLE INFO

Keywords:
Autonomous underwater vehicle
Collision avoidance
Deep reinforcement learning
Soft actor–critic
Event-triggered

ABSTRACT

This paper investigates the collision avoidance problem for autonomous underwater vehicle (AUV) with limited detection range via deep reinforcement learning method in an unknown underwater environment. Firstly, to deal with both unknown static and dynamic obstacles, we introduce two different event-triggered mechanism to generate obstacle-related states and reward the safe collision avoidance action of AUV, respectively. Secondly, the complete state space is customized by combining the state of AUV itself and the state related to a target area. In addition, considering the task requirement of reaching a target area with the shortest path, the complete reward function is designed. Then, we propose a novel event-triggered soft actor–critic (ET-SAC) algorithm for AUV collision avoidance, and give its detailed pseudo code and implementation architecture. Finally, training and evaluation are carried out in the simulation platform we designed, including unknown static obstacle environment and unknown dynamic obstacle environment in 2D/3D space, and the results confirm that the proposed algorithm of this paper is feasible and effective for AUV to avoid collision safely.

1. Introduction

With the development of computer engineering, material engineering and mechanical engineering, the design of autonomous underwater vehicle (AUV) system has become more and more sophisticated, and AUV has been playing an increasingly important role in both civil and military fields. AUV can help humans accomplish many complex underwater tasks, such as marine hydrological monitoring (Wynn et al., 2014), marine biological exploration (Sagala and Bambang, 2011), submarine pipeline inspection (Liu et al., 2018) and mine detection (Hagen et al., 2003), etc. Furthermore, it deserves to be mentioned that the autonomy of AUV has always been a crucial influential factor for its high-quality and high-efficiency task completion in the harsh underwater environment, which has been widely investigated by the academic and engineering circles. In recent years, owing to the increasing exploration of the ocean, various underwater tasks have become more and more complicated, which also puts forward higher requirements for the autonomy of AUV. Besides, it should be pointed out that autonomous collision avoidance is the key technology to ensure AUV safety and achieve autonomy (Cheng et al., 2021) in an unknown underwater environment, which determines the quality and efficiency of task completion. Therefore, the autonomous collision avoidance capability of AUV needs to meet the following requirements:

(1) Reach a designated target area from the starting point;
(2) Safely avoid all unknown static obstacles and unknown dynamic obstacles;
(3) Minimize the path length of AUV from the starting point to the target area.

For the collision avoidance problem of AUV, the existing research methods can be divided into two categories: (*i*) AUV realizes collision avoidance by tracking a planned collision-free path; (*ii*) AUV directly avoids the detected obstacles through collision avoidance algorithm (collision avoidance controller).

For the first kind of methods mentioned above, there have been a large number of research results. For example, to avoid known static obstacles through global path planning, Li and Zhang (2020) proposed a multi-directional A^* algorithm, which overcame defects of the traditional A^* algorithm and reduced the number of search nodes to get the optimal path. To enhance the convergence speed of AUV collision-free path planning using genetic algorithm, Cao et al. (2016) improved the initial population generation method and designed the tangential operator. In order to accomplish the path planing mission of AUV in the complex environment, MahmoudZadeh et al. (2018) used a differential evolution algorithm to optimize the control points generated by the B-spline, so that the AUV can effectively deal with various

☆ This work was supported by the National Natural Science Foundation of China under grant No. 5217110503, No. 51909044 and No. 5217110332.
* Corresponding author.
E-mail address: huangfei@hrbeu.edu.cn (F. Huang).

https://doi.org/10.1016/j.oceaneng.2022.112038
Received 26 March 2022; Received in revised form 18 June 2022; Accepted 16 July 2022
Available online 3 August 2022

obstacles in the three-dimensional space. Additionally, Lim et al. (2020) combined differential evolution with particle swarm optimization, and proposed a selective differential evolution particle swarm optimization algorithm for AUV path planning, which greatly reduced the amount of computation by selecting the most suitable particles for differential evolution hybridization. For the sake of making AUV find the best path in the complex seabed, Che et al. (2020) put forward a particle swarm optimization ant colony algorithm. Moreover, there are also some other methods, such as D^* Lite algorithm (Koenig and Likhachev, 2005), rolling window optimization algorithm (Wang et al., 2016), vector–polar histogram method (Wang et al., 2013) and so on, which will not be introduced in detail here. As a conclusion, the above methods all achieve collision avoidance by planning an optimal collision-free path, but they have the following two main disadvantages:

(1) Since the path tracking controller of the AUV needs to be used in the collision avoidance process, the stability of this method is poor;
(2) The time complexity of the path planning algorithm is high. It is difficult to meet the requirements of real-time collision avoidance in the face of complex underwater environments.

Compared with the first type of method mentioned above, the research results of the second type of method are relatively less. Taheri et al. (2019) proposed a closed-loop rapid exploration random tree (CL-RRT) algorithm to address the kinematic constraints caused by obstacles and the characteristics of AUV, which used three fuzzy proportional differential controllers and an AUV model to evaluate whether the range and vertices of the search tree meet the nonholonomic dynamic constraints of AUV. Based on the hydrodynamic model of AUV and Lyapunov theory, Khalaji and Tourajizadeh (2020) proposed a novel nonlinear controller, which used an improved potential field algorithm (adding an adjustable obstacle avoidance gain into the potential function) to avoid stationary or moving obstacles. Their simulation results confirm that AUV can avoid all obstacles with the minimum deviation from the reference trajectory. Moreover, Abbasi et al. (2010) used line of sight (LOS) guidance to move the AUV toward a target area. By considering the experience of experts and taking the speed of moving obstacles as one of the inputs of fuzzy controller, AUV can turn left or right when encountering obstacles. Their method enables AUV to bypass fixed or moving obstacles and reach the target area effectively. Ding et al. (2014) proposed a leader follower biologically inspired neural network for multi-AUV obstacle avoidance. Specifically, the velocity and trajectory of the virtual AUV were obtained by using the position of the leading AUV, and then the formation control law of multi-AUV was designed by backstepping method. Although the above methods can directly control AUV to realize real-time collision avoidance, they either need a lot of prior knowledge (collision avoidance method based on fuzzy theory) or accurate AUV hydrodynamic model (method based on artificial potential field, CL-RRT, etc.). Therefore, the robustness of these methods will be greatly reduced in the face of more complex environment.

However, the great success of reinforcement learning (RL) in recent years has brought us a new way to design AUV collision avoidance algorithm. RL obtains experience through interaction with the environment and learns policy from experience (Sutton and Barto, 2018). Using RL to design AUV collision avoidance algorithm, we do not need to know the hydrodynamic model of AUV, nor do we need to have complete prior knowledge. At present, there have been a small number of research results. For example, Huang et al. (2014) used single beam sonar to measure obstacle information in turn, and applied RL method to select steering action. A negative reward is given when the AUV approaches an obstacle, and a positive reward is given when the AUV is away from an obstacle. The simulation results show that AUV can realize real-time collision avoidance. Nevertheless, RL has a fatal disadvantage, that is, it cannot deal with the larger dimension of state space. In order to tackle this problem, Mnih et al. (2015) miraculously combined deep learning with RL and proposed deep Q-network (DQN), where they used deep neural network to represent state value function, state–action value function and policy, and perfectly solved the dimensional disaster. Since a large number of deep RL algorithms with strong learning ability have been proposed, such as deep deterministic policy gradient (DDPG) (Lillicrap et al., 2015), twin delayed deep deterministic policy gradient (TD3) (Fujimoto et al., 2018), trust region policy optimization (TRPO) (Schulman et al., 2015) and so on, more and more researchers have applied it into the collision avoidance problem of AUV. Wu et al. (2019) proposed an end-to-end AUV motion control framework based on the proximal policy optimization algorithm (PPO) (Schulman et al., 2017), which directly took the original sonar perception information as the input without considering the dynamic characteristics of AUV. The well-designed reward function enables AUV to avoid collision safely in complex underwater environment. Moreover, Havenstrøm et al. (2021) directly used the two-dimensional sonar image as the input of the deep neural network, and selected the control signal of the fin as the action. Then, a collision avoidance algorithm based on deep RL was proposed. Its output action was processed by low-pass filter, and can complete the tasks of path tracking and obstacle avoidance at the same time. It can be seen from the above research results that deep RL can effectively deal with the collision avoidance problem of AUV. In addition, in the field of deep RL, some algorithms based on maximum entropy RL (Ziebart, 2010; Ziebart et al., 2008) have shown more powerful performance, such as soft Q-learning (SQL) (Haarnoja et al., 2017) and soft actor–critic (SAC) (Haarnoja et al., 2018). They have obtained excellent results on the standard deep RL algorithm test platform. Inspired by these, in order to solve the main challenges of AUV collision avoidance: (a) The processing of the limited perception range of AUV; (b) How to make AUV avoid all dynamic obstacles and static obstacles; (c) Meet the indicators of safe collision avoidance; (d) Customization of state space, action space and reward function, this paper will propose a novel learning method for AUV collision avoidance based on deep RL theory.

The main novelties and contributions of this paper are summarized as follows:

- A method is proposed to generate obstacle-related states through an event-triggered mechanism, in which the event is triggered on the basis of whether the AUV detects an obstacle or not. Then, a complete state space is customized by combining the states of the AUV itself and the states related to the target area.
- A method of giving rewards related to safe collision avoidance action through another event-triggered mechanism is proposed, in which the condition of event triggering is controlled by whether the AUV enters an unsafe area near the obstacle. In addition, the reward functions related to the target area and the shortest path are also designed.
- Combining SAC and event-triggered mechanism, a novel ET-SAC algorithm suitable for AUV collision avoidance in an unknown underwater environment is proposed, and we give its detailed pseudo code and implementation architecture.

The rest of the paper is organized as follows: the basic theory and necessary knowledge for AUV hydrodynamic model, environmental state model and maximum entropy RL are introduced in Section 2. In Section 3, firstly, the state space based on event-triggered mechanism is customized. Secondly, a reward function based on event-triggered mechanism is designed. Finally, the ET-SAC algorithm for AUV collision avoidance is proposed, and the detailed pseudo code and implementation architecture are given. In the following Section 4, the proposed algorithm is trained and evaluated on the simulation platform we designed. Section 5 summarizes the relevant contributions of this paper and points out the direction of future work.

2. Preliminaries

In this section, we will give three basic knowledge, which are necessary to derive the learning method proposed in this paper. It mainly includes AUV hydrodynamic model, environmental state model and maximum entropy RL.

2.1. AUV hydrodynamic model

The 5-degree-of-freedom (DOF) mathematical model of an underactuated AUV proposed by Do and Pan (2009) has been widely used in the field of AUV motion control. Generally, we make use of Fig. 1 to describe the motion coordinate system of an underactuated AUV. In Fig. 1, $O-XYZ$ is the earth-fixed coordinate system and $O'-X'Y'Z'$ is the body-axis coordinate system of the AUV. Then, x, y and z denote the position of the AUV in the earth-fixed coordinate system. θ and ψ represent pitch angle and yaw angle of the AUV in the earth-fixed coordinate system respectively, and their positive direction follows the right-hand rule. u, v and w are the velocities in surge (X of the body-axis coordinate system), sway (Y of the body-axis coordinate system) and heave (Z of the body-axis coordinate system), respectively. Finally, q and r refer to pitch angular velocity and yaw angular velocity in the body-axis coordinate system, and their positive direction is the same as θ and ψ.

Consider 5-DOF kinematics equation of an underactuated AUV as

$$\dot{\boldsymbol{\eta}} = \boldsymbol{T}(\theta,\psi)\,\boldsymbol{\nu} \tag{1}$$

where $\boldsymbol{\eta} = [x, y, z, \theta, \psi]^T \in \mathbb{R}^5$ is the position vector, and $\boldsymbol{\nu} = [u, v, w, q, r]^T \in \mathbb{R}^5$ is the velocity vector. The transformation matrix between $\boldsymbol{\eta}$ and $\boldsymbol{\nu}$ can be written as

$$\boldsymbol{T}(\theta,\psi) = \begin{bmatrix} cos(\theta)cos(\psi) & -sin(\psi) & sin(\theta)cos(\psi) & 0 & 0 \\ cos(\theta)sin(\psi) & cos(\psi) & sin(\theta)sin(\psi) & 0 & 0 \\ -sin(\theta) & 0 & cos(\theta) & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & cos^{-1}(\theta) \end{bmatrix} \tag{2}$$

The dynamics equation of 5-DOF is expressed as follows

$$\begin{cases} m_{11}\dot{u} = m_{22}vr - m_{33}wq - f_u + d_u + \tau_u \\ m_{22}\dot{v} = -m_{11}ur - f_v + d_v \\ m_{33}\dot{w} = m_{11}uq - f_w + d_w \\ m_{55}\dot{q} = (m_{33} - m_{11})uw - f_q - \rho_\eta g \nabla GM_L sin(\theta) + d_q + \tau_q \\ m_{66}\dot{r} = (m_{11} - m_{22})uv - f_r + d_r + \tau_r \end{cases} \tag{3}$$

with m_{ii}, $i = 1,2,3,5,6$ denoting the added mass parameters and the inertia of an underactuated AUV. f_j, $j = u,v,w,q,r$ represent unknown nonlinear dynamics of the AUV, which are mainly hydrodynamic damping and friction terms. d_u, d_v, d_w, d_q and d_r stand for unknown and bounded time-varying disturbances including waves, ocean currents and so on. In this paper, we set the value of the external disturbance terms to zero, that is, the disturbances are not considered. $\boldsymbol{\tau} = [\tau_u, \tau_q, \tau_r]^T \in \mathbb{R}^3$ is the control force and control torque provided by propellers and rudders. $\rho_\eta g \nabla GM_L$ indicates buoyancy related items.

2.2. Environment state model

In this paper, in order to obtain an efficient representation of the environmental state features, we adopt the environmental state model to model the environment around AUV. The main goal of this paper is to make the AUV avoid unknown static obstacles and unknown dynamic obstacles, and then reach a designated target area. Therefore, we only consider the relative position relationship between AUV and detected obstacles or a known target area for the environment state, including the relative distance, the relative pitch angle, and the relative yaw angle. Here, we only take dynamic obstacles as an example. Fig. 2 shows the environmental state model when AUV detects a dynamic obstacle. The situation that AUV detects static obstacles and the relative positional relationship between AUV and a known target area are the same as the model shown in Fig. 2. Moreover, in this paper, we simplify each unknown obstacle and a known task area as a sphere.

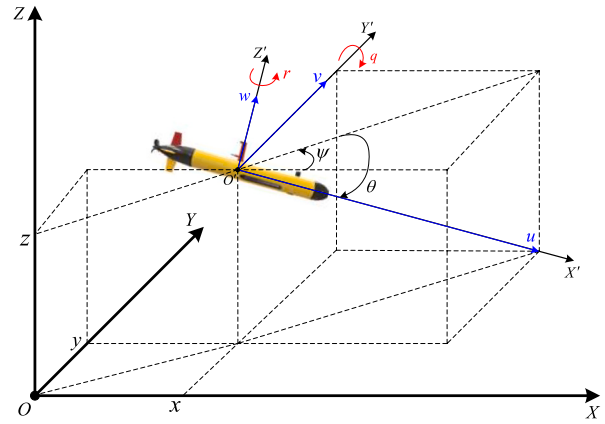

Fig. 1. The motion coordinate system of an underactuated AUV.

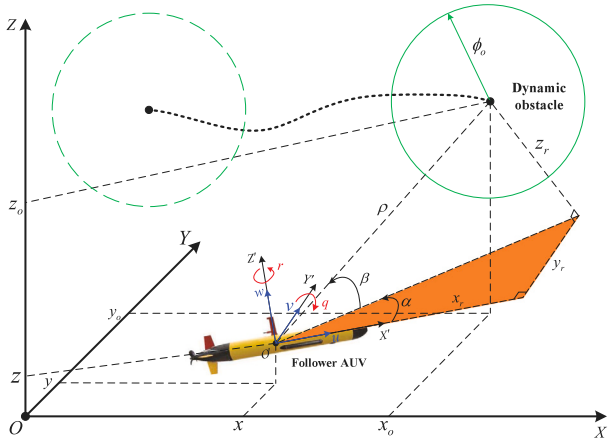

Fig. 2. The relative position relationship between AUV and a dynamic obstacle.

We consider the position (x_o, y_o, z_o) of a detected dynamic obstacle is known, and rewrite Eq. (2) as

$$\boldsymbol{T}(\theta,\psi) = \begin{bmatrix} \boldsymbol{T}_1(\theta,\psi) & \boldsymbol{O}_{3\times2} \\ \boldsymbol{O}_{2\times3} & \boldsymbol{T}_2(\theta) \end{bmatrix} \tag{4}$$

Then, by analyzing the three-dimensional geometric relationship in Fig. 2, we can easily get

$$\begin{bmatrix} x_r \\ y_r \\ z_r \end{bmatrix} = \boldsymbol{T}_1^T(\theta,\psi) \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} \tag{5}$$

with $\Delta x = x_o - x$, $\Delta y = y_o - y$ and $\Delta z = z_o - z$. Finally, the relative distance ρ, the relative pitch angle β and the relative yaw angle α can be derived as

$$\rho = \sqrt{x_r^2 + y_r^2 + z_r^2} \tag{6}$$

$$\beta = arctan2\left(z_r, \sqrt{x_r^2 + y_r^2}\right) \tag{7}$$

$$\alpha = arctan2\left(y_r, x_r\right) \tag{8}$$

where $arctan2(b,a)$ returns the arc tangent of b/a within the bound $(-\pi, \pi]$.

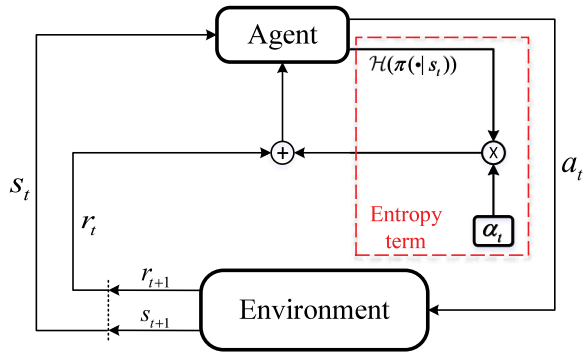

Fig. 3. Maximum entropy reinforcement learning.

2.3. Maximum entropy reinforcement learning

In this article, we consider applying RL to solve the learning problem of AUV collision avoidance policies in continuous state space and action space. Due to the limited detection range of AUV, this problem can be modeled as policy search in partially observable Markov decision process (POMDP), which is defined by a four-tuple $\left(\mathcal{S}, \mathcal{A}, p(\boldsymbol{s}, \boldsymbol{s}', \boldsymbol{a}), r(\boldsymbol{s}, \boldsymbol{a})\right)$. Under the setting of this paper, the observable state space $\mathcal{S}$ and action space $\mathcal{A}$ are defined as continuous. The state transition probability $p(\boldsymbol{s}, \boldsymbol{s}', \boldsymbol{a}) : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \rightarrow [0, \infty)$ denotes the probability density of the next state $\boldsymbol{s}' \in \mathcal{S}$ given the current state $\boldsymbol{s} \in \mathcal{S}$ and action $\boldsymbol{a} \in \mathcal{A}$ of the agent. The environment emits a reward $r(\boldsymbol{s}, \boldsymbol{a}) : \mathcal{S} \times \mathcal{A} \rightarrow [r_{min}, r_{max}]$ on each transition. In addition, we use $\rho_\pi(\boldsymbol{s})$ and $\rho_\pi(\boldsymbol{s}, \boldsymbol{a})$ to stand for the state and state–action marginals of the trajectory distribution included by a policy $\pi(\boldsymbol{a}|\boldsymbol{s})$.

For standard RL, our objective is to learn a policy $\pi(\boldsymbol{a}|\boldsymbol{s})$ to maximize the expected sum of reward, that is

$$\pi^*_{std} = \arg\max_\pi \sum_t \mathbb{E}_{(\boldsymbol{s}_t, \boldsymbol{a}_t) \sim \rho_\pi} [r(\boldsymbol{s}, \boldsymbol{a})] \tag{9}$$

However, almost all standard RL methods suffer from the problem of balancing exploration and exploitation. To deal with the problem, maximum entropy RL augments the reward of standard RL with an entropy term $\mathcal{H}(\pi(\cdot|\boldsymbol{s}))$ of the policy, such that the optimal policy additionally aims to maximize its entropy at each visited state. As shown in Fig. 3.

$$\pi^*_{MaxEnt} = \arg\max_\pi \sum_t \mathbb{E}_{(\boldsymbol{s}_t, \boldsymbol{a}_t) \sim \rho_\pi} \left[r(\boldsymbol{s}, \boldsymbol{a}) + \alpha_t \mathcal{H}(\pi(\cdot|\boldsymbol{s}))\right] \tag{10}$$

where α_t is an adaptive temperature parameter that determines the relative importance of the entropy term versus the reward, and thus controls the randomness of the optimal policy, that is, the degree of exploration. Although the maximum entropy objective differs from the standard maximum expected return objective used in conventional RL, the conventional objective can be recovered in the limit as $\alpha_t \rightarrow 0$.

3. AUV collision avoidance method via SAC and event-triggered mechanism

In this section, the basic SAC framework is applied to solve the problem of AUV collision avoidance in an unknown underwater environment, and its performance heavily depends on the definition of input state and the design of feedback reward. Considering the limited detection range of AUV and the need for the proposed learning method to deal with both unknown static obstacles and unknown dynamic obstacles, we creatively introduce two different event-triggered mechanism to customize the state space and design the reward function, respectively.

3.1. State space and action space

For general methods based on deep RL, the definition of state space needs to follow the following three criteria:

- *Task-oriented*: According to the task requirements in Section 1, select the states related to the tasks.
- *Orthogonalization*: Ensure that the selected states are as independent as possible from each other.
- *Normalization*: Make states of different dimensions in the same numerical magnitude, such as mapping the state value to the interval $[-1, 1]$.

Follow the above guidelines, in order to fully describe the states of AUV itself and the environmental states within the detection range, the observable state space S will be composed of the following parts.

(1) The states of AUV itself

In Section 2.1, we establish the hydrodynamic model of AUV, and obtain its kinematics equation (Eq. (1)) and dynamics equation (Eq. (3)). However, the linear velocity $[u, v, w]^T$, the angular velocity $[q, r]^T$, the position $[x, y, z]^T$ and the angle $[\theta, \psi]^T$ of AUV are highly correlated, so they are not suitable to be directly used as input states in the SAC framework. We will orthogonalize and normalize these states of AUV separately. First, in order to orthogonalize $[u, v, w]^T$, we project it onto the X-axis, Y-axis and Z-axis of $O - XYZ$. Through, Eqs. (1), (2) and (4), we can get that the three components after projection are

$$\begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \boldsymbol{T}_1(\theta, \psi) \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{11}$$

Then, the Min-Max scaling of $\left[v_x, v_y, v_z\right]^T$ can be obtained

$$\begin{bmatrix} \bar{v}_x \\ \bar{v}_y \\ \bar{v}_z \end{bmatrix} = \begin{bmatrix} v_{max} & 0 & 0 \\ 0 & v_{max} & 0 \\ 0 & 0 & v_{max} \end{bmatrix}^{-1} \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \tag{12}$$

where v_{max} is the maximum linear velocity of AUV. Next, since both $[q, r]^T$ and $[x, y, z]^T$ are orthogonal, we only need to perform Min-Max scaling on them.

$$\begin{bmatrix} \bar{q} \\ \bar{r} \end{bmatrix} = \begin{bmatrix} q_{max} & 0 \\ 0 & r_{max} \end{bmatrix}^{-1} \begin{bmatrix} q \\ r \end{bmatrix} \tag{13}$$

$$\begin{bmatrix} \bar{x} \\ \bar{y} \\ \bar{z} \end{bmatrix} = \begin{bmatrix} x_{max} & 0 & 0 \\ 0 & y_{max} & 0 \\ 0 & 0 & z_{max} \end{bmatrix}^{-1} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{14}$$

with k_{max}, $k = q, r$ representing the maximum value of the corresponding states of AUV. x_{max}, y_{max} and z_{max} denote the maximum value of a task area in the X-axis, Y-axis and Z-axis of $O - XYZ$, respectively. Finally, for the orthogonalization and normalization of $[\theta, \psi]^T$, it is undoubtedly the best choice to directly calculate their sine and cosine values.

$$\begin{bmatrix} \bar{\theta} \\ \bar{\psi} \end{bmatrix} = \begin{bmatrix} sin(\theta) & cos(\theta) & sin(\psi) & cos(\psi) \end{bmatrix}^T \tag{15}$$

(2) Relative to obstacles

In this section, in order to uniformly represent the states related to static obstacles and dynamic obstacles within the limited detection range of AUV, including relative distance ρ, relative pitch angle β and relative yaw angle α, we introduce an event-triggered mechanism and combine it with the environmental state model established in Section 2.2. In the collision avoidance tasks, we define the physical meaning of the event triggering as whether the AUV detects obstacles. First, an event-triggered flag E_d is given to indicate whether the event

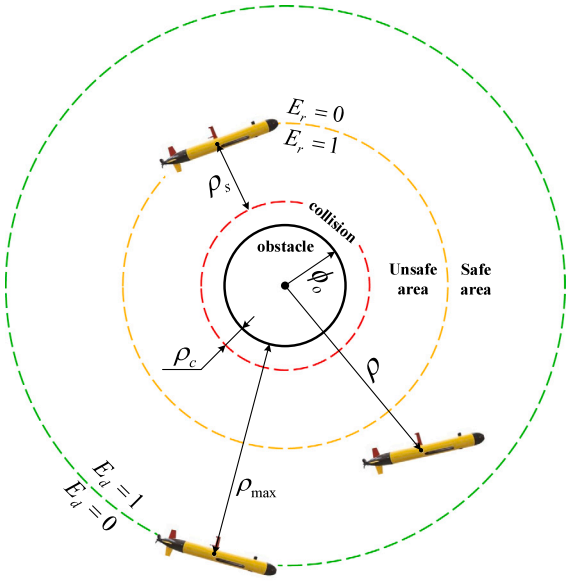

Fig. 4. Schematic diagram of the state triggering, the reward triggering, the relative distance relationship between AUV and an obstacle.

is triggered or not, that is, $E_d = 0$ means that the AUV has not detected any obstacles and $E_d = 1$ means that the AUV has detected obstacles.

$$E_d = \begin{cases} 0, & \rho > \phi_o + \rho_{max} \\ 1, & \rho \le \phi_o + \rho_{max} \end{cases} \tag{16}$$

where ϕ_o is the radius of a obstacle, and ρ_{max} represents the maximum detection range of the sonar equipped with AUV, as shown in Fig. 4. In addition, since the dimension of the input states of the SAC framework is fixed after the neural network structure is designed, so as to ensure that the dimension of the states related to the obstacles remain unchanged, we only select the relevant states of a nearest obstacle that can be detected by AUV at any time. Then, combined with Eq. (6), we can define the distance generated by the event trigger.

$$\rho_o = E_d\left(\rho - \phi_o - \rho_c\right) + \left(1 - E_d\right)\left(\rho_{max} - \rho_c\right) \tag{17}$$

where ρ_c denotes the collision radius (when $\rho \le \phi_o + \rho_c$, AUV will collide with an obstacle). In the same way, combining Eqs. (7) and (8), we define

$$\beta_o = E_d\beta + \left(1 - E_d\right)\frac{\pi}{2} \tag{18}$$

$$\alpha_o = E_d\alpha + \left(1 - E_d\right)\pi \tag{19}$$

Finally, perform Min-Max scaling on ρ_o to obtain the states we need.

$$\bar{\rho}_o = \frac{\rho_o}{\rho_{max} - \rho_c} \tag{20}$$

For the processing of β_o and α_o, the sine and cosine values are calculated as in Eq. (15).

$$\begin{bmatrix} \bar{\boldsymbol{\beta}}_o \\ \bar{\boldsymbol{\alpha}}_o \end{bmatrix} = \begin{bmatrix} sin(\beta_o) & cos(\beta_o) & sin(\alpha_o) & cos(\alpha_o) \end{bmatrix}^T \tag{21}$$

(3) Relative to a target area

As in (2), regarding the states of a target area, we only select the relative distance ρ_b, relative pitch angle β_b and relative yaw angle α_b between AUV and a target area, and the calculation method is the same as Eqs. (6)–(8). Since the target area in the AUV collision avoidance mission is specified, its center point and radius are known, so we do not need to introduce event-triggered mechanism when obtaining the state related to a target area. For ρ_b, after performing Min-Max scaling, we can get

$$\bar{\rho}_b = \frac{\rho_b}{\sqrt{x_{max}^2 + y_{max}^2 + z_{max}^2} - \phi_b} \tag{22}$$

where ϕ_b is the radius of a target area. Besides, the orthogonalized and normalized relative angle $\bar{\beta}_b$ and $\bar{\alpha}_b$ can be written directly as

$$\begin{bmatrix} \bar{\boldsymbol{\beta}}_b \\ \bar{\boldsymbol{\alpha}}_b \end{bmatrix} = \begin{bmatrix} sin(\beta_b) & cos(\beta_b) & sin(\alpha_b) & cos(\alpha_b) \end{bmatrix}^T \tag{23}$$

(4) Terminal states

We use a two-dimensional vector $\boldsymbol{A}$ to represent the terminal states, which is mainly used to determine the end states of an episode, so as to assist the training of the learning algorithm that will be proposed later. Table 1 lists the meaning of the four terminal states and their corresponding two-dimensional vectors.

At last, the final state $\boldsymbol{s}$ is defined by combining the above four parts.

$$\begin{aligned} \boldsymbol{s} \doteq \Big[& \bar{v}_x \quad \bar{v}_y \quad \bar{v}_z \quad \bar{q} \quad \bar{r} \quad \bar{x} \quad \bar{y} \quad \bar{z} \quad \bar{\boldsymbol{\theta}}^T \quad \bar{\boldsymbol{\psi}}^T \\ & E_d \quad E_r \quad \bar{\rho}_o \quad \bar{\boldsymbol{\beta}}_o^T \quad \bar{\boldsymbol{\alpha}}_o^T \quad \bar{\rho}_b \quad \bar{\boldsymbol{\beta}}_b^T \quad \bar{\boldsymbol{\alpha}}_b^T \quad \boldsymbol{A}^T \Big]^T \end{aligned} \tag{24}$$

where E_r is an event-triggered flag related to collision avoidance safety and will be given in the next section. For the continuous action space $\mathcal{A}$, we directly define the continuous action $\boldsymbol{a}$ as the control force and control moment in AUV dynamics Eq. (3).

$$\boldsymbol{a} \doteq \boldsymbol{\tau} = \begin{bmatrix} \tau_u, \tau_q, \tau_r \end{bmatrix}^T \tag{25}$$

3.2. Reward

In the RL method, the design of the reward function r is very important, which determines whether the agent can complete the specified goal. For the AUV collision avoidance task, we design the reward function from the following four aspects according to the task requirements in Section 1.

(1) Safe collision avoidance

In order to make the AUV avoid the detected obstacles safely, a safe distance ρ_s shown in Fig. 4 is given. Our goal is to ensure that the relative distance between AUV and an obstacle is always not less than the safe distance throughout the mission, that is, $\rho - \phi_o - \rho_c \ge \rho_s$. Secondly, like E_d in Section 3.1 (2), an event-triggered flag E_r is given to indicate whether AUV is in a safe area or an unsafe area, as shown in Fig. 4.

$$E_r = \begin{cases} 0, & \rho - \phi_o - \rho_c > \rho_s \\ 1, & \rho - \phi_o - \rho_c \le \rho_s \end{cases} \tag{26}$$

We set ς_{oe} as the weight of the safety collision avoidance reward, which can be defined as

$$\varsigma_{oe} = \varsigma_{os} + E_r\varsigma_{ou} \tag{27}$$

where ς_{os} and ς_{ou} are constants. Finally, considering that the goal of RL is to maximize the expected sum of reward (refer to Eqs. (9) and (10)), the reward function for safe collision avoidance can be defined as

$$r_o = -\varsigma_{oe}\left(1 - \bar{\rho}_o + 1 - \frac{2|\beta_o|}{\pi} + 1 - \frac{|\alpha_o|}{\pi}\right) \tag{28}$$

Table 1
Description of terminal states and corresponding vectors.

A	Description
$[0,\ 0]^T$	AUV navigates normally within the mission area without collision.
$[0,\ 1]^T$	The number of iterations reaches the set value.
$[1,\ 0]^T$	AUV droves out of a mission area or AUV collides with an obstacle.
$[1,\ 1]^T$	AUV reaches the designated target area without collision.

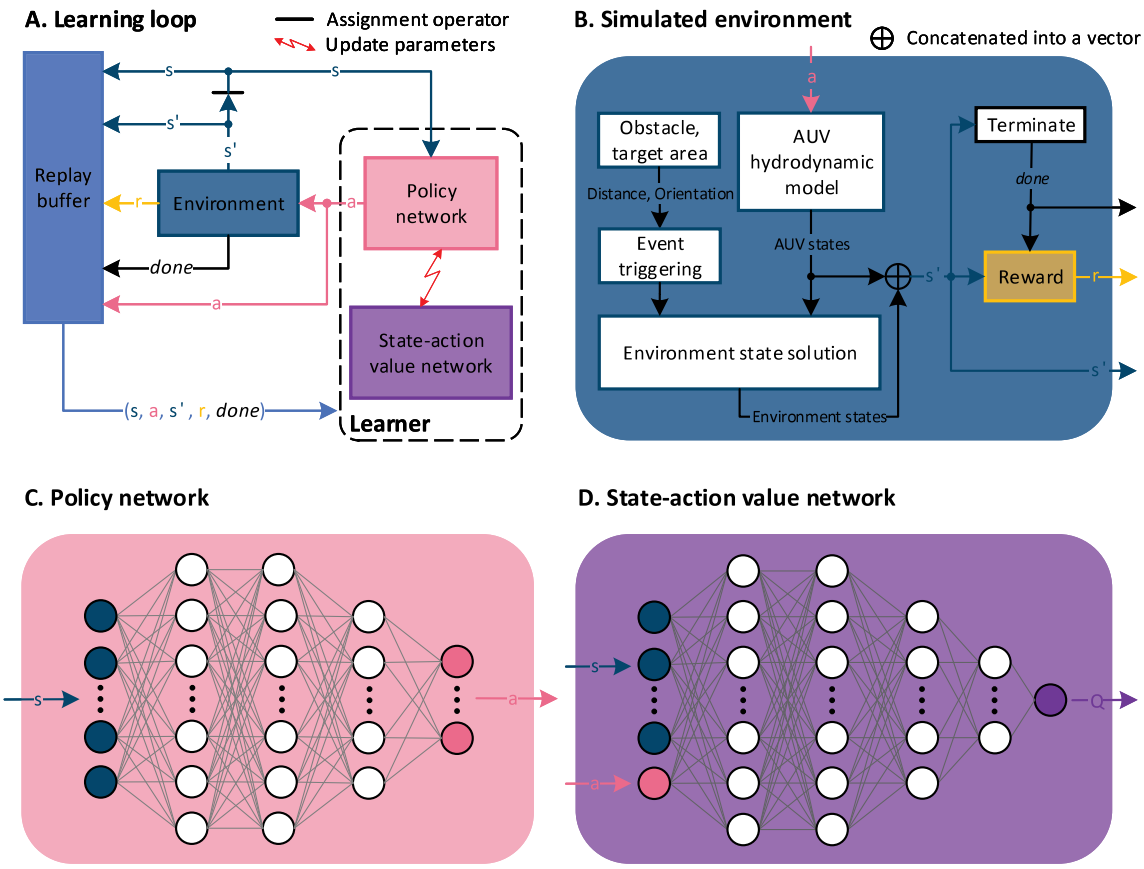

Fig. 5. Component representation of ET-SAC collision avoidance architecture. **A.** Interaction between environment and a learner. A learner outputs actions to the environment according to the current state. Then, the environment generates the states of next moment, rewards and a terminal states flag, and sends these data to the replay buffer. Finally, the learner samples data from the replay buffer to update the network parameters. **B.** Simulation environment model, consisting of obstacles and target area, event-triggered mechanism, AUV hydrodynamic model (Section 2.1), environment state solution module (Section 3.1), reward module and terminal module (Section 3.2). **C.** Policy network is a fully connected network with two hidden layers whose inputs are states and outputs are actions. **D.** State–action value network is a fully connected network with three hidden layers. Its input are states and a action, and its output is the evaluation value of state–action.

The first item $1-\bar{\rho}_o$ indicates that the closer AUV is to an obstacle, the smaller the reward. Moreover, the second item $1-\frac{2|\beta_o|}{\pi}$ and the third item $1-\frac{|\alpha_o|}{\pi}$ mean that the AUV's forward direction (including pitch angle and yaw angle) is toward the direction of an obstacle, the smaller the reward.

(2) Reach the target area

As in (1), in order to enable the AUV to reach a target area accurately, we also use relative distance, relative pitch angle and relative yaw angle to define the reward function. The difference is that the center point and radius of a target area are known. We do not need to introduce event trigger, and our goal has become to reach rather than to stay away. The reward function r_b for reaching the specified target area can be written as

$$r_b = -\varsigma_b\left(\bar{\rho}_b + \frac{2|\beta_b|}{\pi} + \frac{|\alpha_b|}{\pi}\right) \tag{29}$$

where ς_b is the setting weight for r_b. Moreover, to ensure that the action of AUV close to a target area is always rewarded rather than punished, $\varsigma_b < \varsigma_{os} + \varsigma_{ou}$ must be satisfied when setting the value of ς_b, ς_{os} and ς_{ou}.

(3) Shortest path

We will ensure the shortest total path of AUV from the starting point to a target area from the following two aspects. On the one hand, the shortest path is equivalent to the minimum total energy consumed by AUV. Therefore, we can naturally define that the shortest path reward r_e decreases with the energy consumption of AUV at each time step t.

$$r_e = -\varsigma_e\left[\left(\frac{\tau_u}{\tau_{u-max}}\right)^2 + \left(\frac{\tau_q}{\tau_{q-max}}\right)^2 + \left(\frac{\tau_r}{\tau_{r-max}}\right)^2\right] \tag{30}$$

where ς_e is the setting weight for r_e. τ_{u-max}, τ_{q-max} and τ_{r-max} represent the maximum value of the three control signals. On the other hand, we have implemented that the reward value of each time step is set to a negative value, which means that the AUV can get a larger reward with a shorter time steps to the specified target area.

(4) Terminal reward

According to the terminal states $\boldsymbol{A}$ in Section 3.1 (4), we can define the reward function r_f to evaluate whether the AUV has successfully completed the collision avoidance task in an episode.

$$r_f = \begin{cases} 0.001, & \boldsymbol{A} = [0,\ 0]^T \\ -1, & \boldsymbol{A} = [0,\ 1]^T \\ -1, & \boldsymbol{A} = [1,\ 0]^T \\ 1, & \boldsymbol{A} = [1,\ 1]^T \end{cases} \tag{31}$$

As a conclusion, the final reward function r can be obtained by simply combining the above four parts.

$$r = r_o + r_b + r_e + r_f \tag{32}$$

3.3. The proposed learning method

In this section, a novel learning method for AUV collision avoidance is proposed by introducing an event-triggered mechanism into SAC, and we call it event-triggered SAC (ET-SAC). Firstly, combined with the maximum entropy RL in Section 2.3, some network parameter update laws of SAC algorithm are given (Haarnoja et al., 2018). In the SAC architecture, the interaction between two independent state–action

value network $Q_{\theta_k}(s_t, a_t), k = 1, 2$ and policy network $\pi_\omega(a_t|s_t)$ make the policy converge gradually and close to the optimal π^*_{MaxEnt}, where θ_1, θ_2 and ω are network parameters. The state–action value function is mainly used to evaluate the current policy, which can be defined as

$$Q_{\theta_k}(s_t, a_t) \doteq r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1}\sim p}\left[V(s_{t+1})\right], k = 1, 2 \tag{33}$$

where

$$V(s_t) \doteq \mathbb{E}_{a_t\sim\pi_\omega}\left[Q_{\theta_k}(s_t, a_t) - \alpha_t \log \pi_\omega(a_t|s_t)\right] \tag{34}$$

is the soft state value function, and γ is the discount factor. To stabilize training, two target state–action value network $Q_{\bar{\theta}_k}(s_t, a_t), k = 1, 2$ are introduced to calculate the objective function of $Q_{\theta_k}(s_t, a_t)$.

$$\begin{aligned} y = r(s_t, a_t) + \gamma \Big(& \min_{k=1,2} Q_{\bar{\theta}_k}(s_{t+1}, a_{t+1}) \\ & -\alpha_t \log\left(\pi_\omega(a_{t+1}|s_{t+1})\right) \Big) \end{aligned} \tag{35}$$

Therefore, the update objective of the parameters θ_k can be defined as

$$J_Q(\theta_k) = \mathbb{E}_{(s_t,a_t)\sim D}\left[\frac{1}{2}\left(Q_{\theta_k}(s_t, a_t) - y\right)^2\right] \tag{36}$$

where D is the replay buffer used to store s_{i-1}, a_{i-1}, s_i, $done_{i-1}$, $i \in [t - B, t]$. The boolean value $done_{i-1}$ indicates whether the state at time $i-1$ is a terminal state, and B is the capacity of the replay buffer. Then, θ_k can be optimized with Stochastic Gradients Descent (SGD)

$$\begin{aligned} \hat{\nabla}_{\theta_k} J_Q(\theta_k) = \nabla_{\theta_k} Q_{\theta_k}(s_t, a_t) \Big(& Q_{\theta_k}(s_t, a_t) - \Big(r(s_t, a_t) + \gamma \\ & \times \Big(\min_{k=1,2} Q_{\bar{\theta}_k}(s_{t+1}, a_{t+1}) - \alpha_t \log\left(\pi_\omega(a_{t+1}|s_{t+1})\right) \Big) \Big) \Big) \end{aligned} \tag{37}$$

Finally, we get the update law of θ_k as

$$\theta_k \leftarrow \theta_k - \lambda_\theta \hat{\nabla}_{\theta_k} J_Q(\theta_k), k = 1, 2 \tag{38}$$

where λ_θ is the learning rate, and the parameter $\bar{\theta}_k$ is updated slowly through θ_k and a weight ϱ

$$\bar{\theta}_k \leftarrow \varrho\theta_k + (1 - \varrho)\bar{\theta}_k, k = 1, 2 \tag{39}$$

In addition, the parameter ω of the policy network $\pi_\omega(a_t|s_t)$ can be learned by directly minimizing the following performance function

$$\begin{aligned} J_\pi(\omega) = \mathbb{E}_{s_t\sim D}\Big[& \mathbb{E}_{a_t\sim\pi_\omega}\left[\alpha_t \log \pi_\omega(a_t|s_t)\right. \\ & \left. - \min_{k=1,2} Q_{\theta_k}(s_t, a_t)\right]\Big] \end{aligned} \tag{40}$$

In order to update the parameter ω in the policy network through back propagation, we need to reparameterize the action

$$a_t = f_\omega(\chi_t; s_t) = \mu_\omega + \sigma_\omega \chi_t \tag{41}$$

where μ_ω and $\ln(\sigma_\omega)$ are the output of the policy network, and χ_t represents the input noise vector obeying spherical Gaussian distribution. Therefore, Eq. (40) can be rewritten as

$$\begin{aligned} J_\pi(\omega) = \mathbb{E}_{s_t\sim D, \chi_t\sim\mathcal{N}}\Big[& \alpha_t \log \pi_\omega\left(f_\omega(\chi_t; s_t)|s_t\right) \\ & - \min_{k=1,2} Q_{\theta_k}(s_t, f_\omega(\chi_t; s_t))\Big] \end{aligned} \tag{42}$$

Then, the gradient estimation of Eq. (42) can be obtained

$$\begin{aligned} \hat{\nabla}_\omega J_\pi(\omega) = & \nabla_\omega \alpha_t \log\left(\pi_\omega(a_t|s_t)\right) + \Big(\nabla_{a_t} \alpha_t \log \pi_\omega(a_t|s_t) \\ & - \nabla_{a_t} \min_{k=1,2} Q_{\theta_k}(s_t, a_t) \Big) \hat{\nabla}_\omega f_\omega(\chi_t; s_t) \end{aligned} \tag{43}$$

Finally, the update law of ω can be written as

$$\omega \leftarrow \omega - \lambda_\omega \hat{\nabla}_\omega J_\pi(\omega) \tag{44}$$

where λ_ω stands for the learning rate. Moreover, We can automatically adjust the weight α_t of entropy term by optimizing the following objective function.

$$J(\alpha_t) = \mathbb{E}_{a_t\sim\pi_\omega}\left[-\alpha_t \log \pi_\omega(a_t|s_t) - \alpha_t \mathcal{H}_0\right] \tag{45}$$

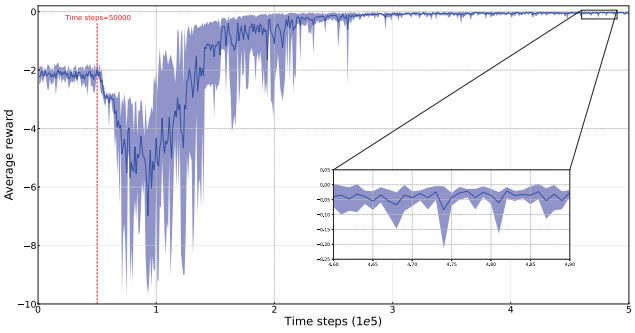

Fig. 6. Average reward obtained during training in a 2D unknown static obstacle environment (10 evaluations every 1000 time steps).

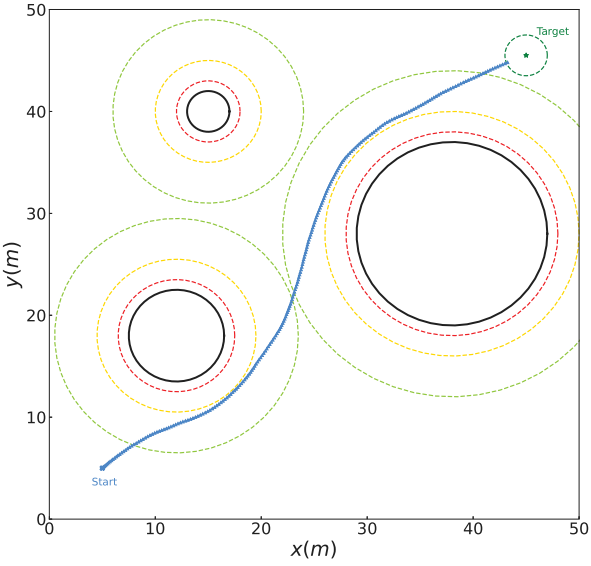

Fig. 7. The trajectory of AUV in a 2D unknown static obstacle environment.

Then, the gradient estimation of Eq. (45) can be written as

$$\hat{\nabla}_{\alpha_t} J(\alpha_t) = -\log \pi_\omega(a_t|s_t) - \mathcal{H}_0 \tag{46}$$

where $\mathcal{H}_0$ is the entropy target. The update law of α_t is

$$\alpha_t \leftarrow \alpha_t - \lambda_{\alpha_t} \hat{\nabla}_{\alpha_t} J(\alpha_t) \tag{47}$$

where λ_{α_t} is the learning rate. Secondly, combining the state space, action space and reward function designed based on the event-triggered mechanism in Sections 3.1 and 3.2, and using the event-triggered flags E_d and E_r to control the update logic of SAC, we can get a novel event-triggered SAC (ET-SAC) algorithm for AUV collision avoidance. As a conclusion, the complete pseudo code of ET-SAC is described by Algorithm 1, which contains detailed update logic controlled by the event-triggered mechanism. In addition, Fig. 5 shows the component representation of ET-SAC collision avoidance architecture in detail.

4. Case verification

According to different AUV collision avoidance environments, four simulation platforms for 2D/3D unknown static and dynamic obstacle environments are developed respectively, and the effectiveness of the proposed learning method is verified through case studies. Although all the work in this paper is carried out based on the 3D space, the collision avoidance method for the 2D space can be obtained by simply removing the items related to the z-axis. The 5-DOF hydrodynamic

Algorithm 1 Pseudo code of ET-SAC for AUV collision avoidance

Initialize the AUV collision avoidance environment
Randomly initialize state–action value networks Q_{θ_1}, Q_{θ_2} and policy network $\pi_{\boldsymbol{\omega}}$ with parameter vectors θ_1, θ_2, $\boldsymbol{\omega}$
Initialize target state–action value networks $Q_{\bar{\theta}_1}$, $Q_{\bar{\theta}_2}$ with parameter vectors $\bar{\theta}_1 \leftarrow \theta_1$, $\bar{\theta}_2 \leftarrow \theta_2$
Initialize replay buffer D to capacity $\boldsymbol{B}$ and empty it
Initialize global shared step counter $C \leftarrow 0$
Set the maximum number of iterations C_{max} and some other necessary parameters (refer to Section 4)
for $C \in \{0, 1, 2, \cdots, C_{max}\}$ **do**
    Reset step counter for each episode $t \leftarrow 0$
    Reset the environment, including $\boldsymbol{A} \leftarrow [0,0]^T$, $E_d \leftarrow 0$, $E_r \leftarrow 0$, $done \leftarrow False$ and so on
    Get the initial state $\boldsymbol{s} \leftarrow \boldsymbol{s}_0$
    repeat
        Calculation $\mu_{\boldsymbol{\omega}}, \ln(\sigma_{\boldsymbol{\omega}}) \leftarrow \pi_{\boldsymbol{\omega}}(\boldsymbol{s})$
        Perform action with the input noise $\boldsymbol{a} \leftarrow \mu_{\boldsymbol{\omega}} + \sigma_{\boldsymbol{\omega}} \chi_t$, $\chi_t \sim \mathcal{N}(0, \sigma)$
        Get partial states $\bar{v}_x, \bar{v}_y, \bar{v}_z, \bar{q}, \bar{r}, \bar{x}, \bar{y}, \bar{z}, \bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\psi}}, \bar{\rho}_b, \bar{\boldsymbol{\beta}}_b, \bar{\boldsymbol{\alpha}}_b, \boldsymbol{A}$ and partial rewards r_b, r_e, r_f
        if $\boldsymbol{A} \neq [0,0]^T$ **then**
            $done \leftarrow True$
        end if
        if $\rho \leq \phi_o + \rho_{max}$ **then**
            $E_d \leftarrow 1$
        end if
        if $\rho - \phi_o - \rho_c \leq \rho_s$ **then**
            $E_r \leftarrow 1$
        end if
        Get partial states $E_d, E_r, \bar{\rho}_o, \bar{\boldsymbol{\beta}}_o, \bar{\boldsymbol{\alpha}}_o$ and partial reward r_o
        Merge to get the state of the next moment $\boldsymbol{s}'$ (Eq. (24)) and reward r (Eq. (32))
        Store samples in the replay buffer $D \leftarrow D \cup \left\{\left(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}', r\right)\right\}$
        if $|D| == \boldsymbol{B}$ **then**
            Sample b_s samples $\left(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}', r, done\right)$ from D
            Update $\boldsymbol{\theta}_k \leftarrow \boldsymbol{\theta}_k - \lambda_\theta \hat{\nabla}_{\boldsymbol{\theta}_k} J_Q(\boldsymbol{\theta}_k)$, for $k = 1, 2$ (Eq. (38))
            Update $\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} - \lambda_{\boldsymbol{\omega}} \hat{\nabla}_{\boldsymbol{\omega}} J_\pi(\boldsymbol{\omega})$ (Eq. (44))
            Update $\alpha_t \leftarrow \alpha_t - \lambda_{\alpha_t} \hat{\nabla}_{\alpha_t} J(\alpha_t)$ (Eq. (47))
            Update $\bar{\theta}_k \leftarrow \varrho \boldsymbol{\theta}_k + (1 - \varrho)\bar{\theta}_k$, for $k = 1, 2$ (Eq. (39))
        end if
        $\boldsymbol{s} \leftarrow \boldsymbol{s}'$
        $t \leftarrow t + 1$
    until $done == True$
end for

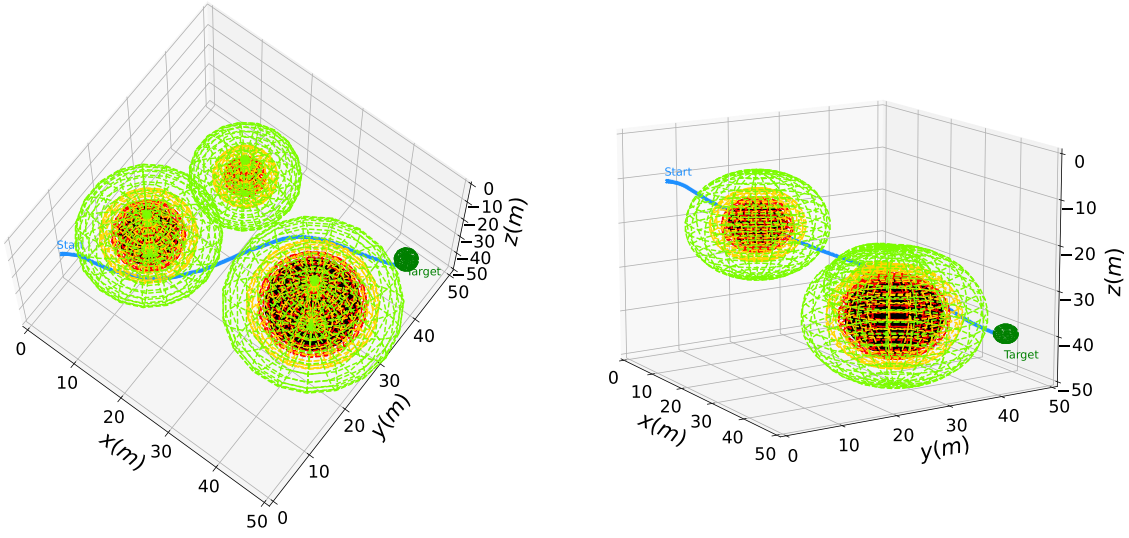

Fig. 8. The trajectory of AUV in a 3D unknown static obstacle environment. The left and right figures are different perspectives of a same figure.

model introduced in Section 2.1 is used to simulate the motion of AUV. In addition, Table 2 gives the values and brief description of some parameters that will be uniformly used in the following two case studies.

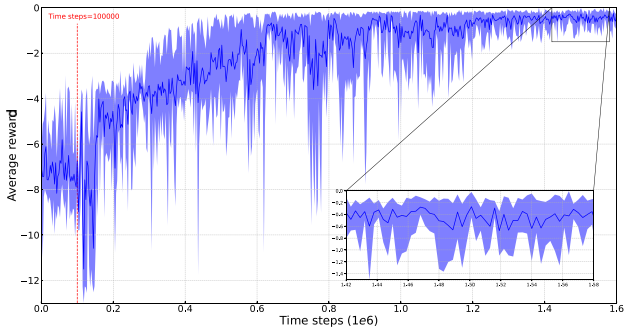

Fig. 9. Average reward obtained during training in a 3D unknown static obstacle environment (10 evaluations every 3000 time steps).

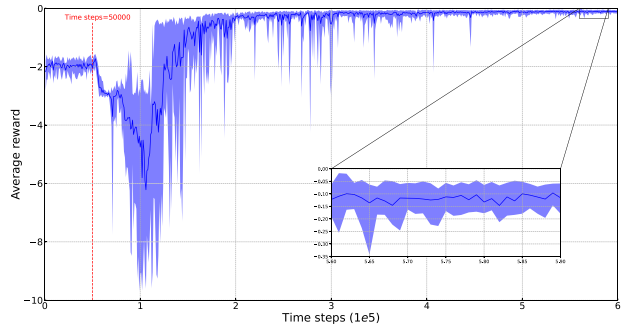

Fig. 10. Average reward obtained during training in a 2D unknown dynamic obstacle environment (10 evaluations every 1000 time steps).

Table 2
The value of shared parameters.

Parameter	*Value and description*
m_{11}, m_{22}, m_{33}, m_{55}, m_{66}	25.0, 17.5, 30.0, 22.5, 15.0
f_u	$0.31u + 0.11u\|u\| + 0.08u^3$
f_v	$20.23v + 15.56v\|v\| + 12.17v^3$
f_w	$10.51w + 9.29w\|w\| + 7.16w^3$
f_q	$11.13q + 10.69q\|q\| + 9.48q^3$
f_r	$3.11r + 2.31r\|r\| + 0.91r^3$
$\rho_\eta g \nabla G M_L$	9.7
ϕ_b	2.0
x_{max}, y_{max}, z_{max}	50.0
v_{max}, q_{max}, r_{max}, ρ_{max}	3.0, 0.3, 1.0, 7.0
ρ_c, ρ_s	1.0, 3.0
τ_{u-max}, τ_{q-max}, τ_{r-max}	100.0
Policy network	Fully connected network with three hidden layers. The number of nodes in the hidden layer is 320, 320 and 160, respectively.
State–action value network	Fully connected network with four hidden layers. The number of nodes in the hidden layer is 320, 320, 160 and 80, respectively.
ς_{os}, ς_{ou}, ς_b, ς_e	0.02, 0.01, 0.01, 0.01
λ_θ, λ_ω, λ_{α_i}	0.0003
ϱ	0.005
γ	0.99
σ	1.0
$\boldsymbol{B}$	2D environment: 50000; 3D environment: 100000
b_s	2D environment: 64; 3D environment: 128
$\mathcal{H}_0$	2D environment: −0.5; 3D environment: −3.0

4.1. Case 1: Static obstacle environment

In order to verify the collision avoidance ability of the proposed algorithm in an unknown static obstacle environment, training and evaluation are carried out on 2D and 3D simulation platforms, respectively. During training, we evaluate the learned policy 10 times every 1000 or 3000 time steps and calculate the average reward for each episode. Figs. 6 and 9 show the average reward obtained through evaluation at different time steps, in which the upper and lower ends of the shaded area represent the maximum and minimum values of the average reward in 10 evaluations respectively. In addition, the red lines in Figs. 6 and 9 represent the time steps for starting learning, which are 50000 and 100000. On the left side of the red line, we only collect experience. On the right side of the red line, we learn policy while collecting experience. It can be seen from the simulation in the 2D/3D environment that the average reward of the random strategy without learning is only around −2.3/−7.5 (left of the red line), while the average reward of the strategy after learning is about −0.03/−0.45 (right of the red line). Finally, we evaluate visually the learned policy, and the results are shown in Figs. 7 and 8. It can be seen that the AUV safely avoids all static obstacles within the detection range (green circle) and reaches a target area with almost the shortest path, where "safety" means that the AUV does not enter any unsafe area (yellow circle). To sum up, the learning method proposed in this paper is very effective for AUV to avoid collision safely in an unknown static obstacle environment.

4.2. Case 2: Dynamic obstacle environment

To further verify the performance of the proposed learning method in handling unknown dynamic obstacles, we conduct experiments similar to Case 1 in a 2D/3D unknown dynamic obstacle environment. And so in the same way, we show the average reward obtained through evaluation during the training process in Figs. 10 and 13. It can be seen that the average rewards obtained by random policy in the 2D/3D environment are around −2.1/−3.1, while the average rewards of trained policy are around −0.12/−0.65. In addition, Figs. 11 and 12 depict the results of the visual evaluation, and it is observed that the AUV can safely reach the target area with almost the shortest path from the starting position. As a conclusion, the above simulation results strongly confirm that our algorithm can make AUV avoid unknown dynamic obstacles efficiently and safely.

5. Conclusion

In this paper, a novel learning method for AUV collision avoidance has been proposed by combining deep RL and event-triggered mechanism. In order to enable AUV to avoid both unknown static obstacles and unknown dynamic obstacles under the condition of limited detection range, we have introduced two different event-triggered mechanisms when customizing the state space and the reward function. Firstly, we have proposed a method to generate obstacle-related states through an event-triggered mechanism, in which the condition for an event to be triggered is controlled by whether AUV detects obstacles. Secondly, so as to reward the safe collision avoidance action of AUV, a method has been proposed to give reward related to collision avoidance through another event-triggered mechanism, and the condition for the event to be triggered have become whether AUV enters an unsafe area near the obstacles. In addition, the states of AUV itself, the states and reward related to a target area and the reward related to the path length have been all considered to customize the whole state space and the reward function. Finally, combined with the SAC algorithm and the above state space as well as reward function based on event-triggered mechanism, a novel ET-SAC algorithm suitable for AUV collision avoidance has been proposed, and we have given its detailed pseudo code and implementation architecture. To train and evaluate our algorithm, we have developed simulation platforms for four different environments, including 2D/3D unknown static and dynamic obstacle environments. A large number of simulation experiments have been carried out on the above four platforms. As a conclusion, the average reward obtained by evaluation during training and the actual trajectory of AUV have confirmed that the algorithm proposed in this paper is very effective for AUV to avoid collision safely in an unknown environment.

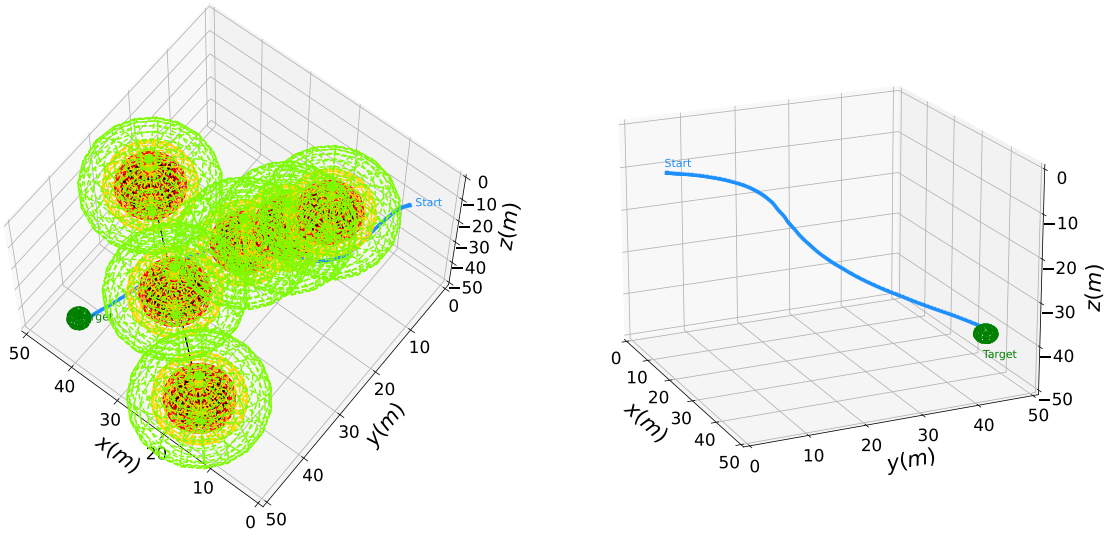

Fig. 11. The trajectory of AUV in a 3D unknown dynamic obstacle environment. The left and right figures are different perspectives of a same figure, and the obstacles are removed in the right figure in order to observe the AUV trajectory. Two black arrows represent the moving directions of the two obstacles, respectively.

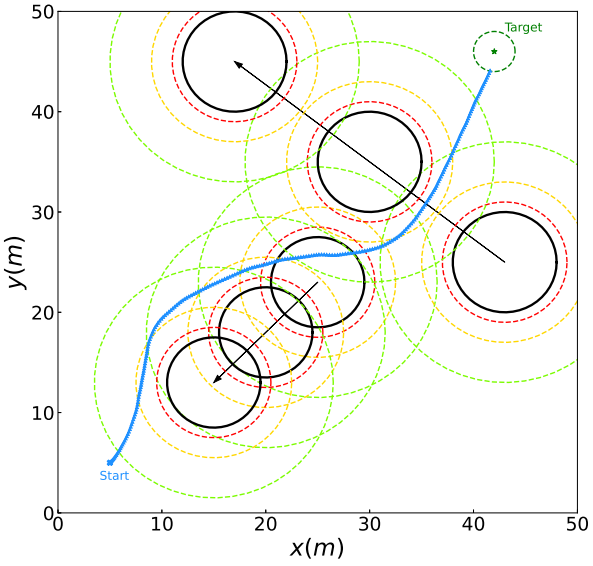

Fig. 12. The trajectory of AUV in a 2D unknown dynamic obstacle environment. Two black arrows represent the moving directions of the two obstacles, respectively.

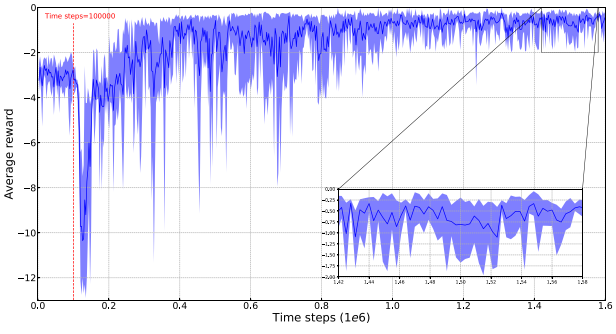

Fig. 13. Average reward obtained during training in a 3D unknown dynamic obstacle environment (10 evaluations every 3000 time steps).

Actually, we mainly realize the autonomous collision avoidance of AUV in theory, and verify the performance of the proposed learning method through simulation. However, for the actual AUV, there are still some restrictions that have not been taken into account, such as time delay, actuator failures, ocean current and so on. In the future, we will deeply consider these constraints to improve the security and robustness of the proposed method, and apply it into the actual AUV.

CRediT authorship contribution statement

Jian Xu: Methodology, Investigation, Review. **Fei Huang:** Investigation, Writing – original draft, Software. **Di Wu:** Project administration, Resources. **Yunfei Cui:** Writing – review & editing. **Zheping Yan:** Supervision, Funding acquisition. **Xue Du:** Resources, Data curation.

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

The authors do not have permission to share data.

References

Abbasi, M., Danesh, M., Ghayour, M., 2010. A path fuzzy planner for autonomous underwater vehicles to avoid moving unknown obstacles. In: 2010 IEEE International Conference on Mechatronics and Automation. IEEE, pp. 1264–1269.

Cao, J., Li, Y., Zhao, S., Bi, X., 2016. Genetic-algorithm-based global path planning for AUV. In: 2016 9th International Symposium on Computational Intelligence and Design. ISCID, IEEE, pp. 79–82.

Che, G., Liu, L., Yu, Z., 2020. An improved ant colony optimization algorithm based on particle swarm optimization algorithm for path planning of autonomous underwater vehicle. J. Ambient Intell. Humaniz. Comput. 11 (8), 3349–3354.

Cheng, C., Sha, Q., He, B., Li, G., 2021. Path planning and obstacle avoidance for AUV: A review. Ocean Eng. 235, 109355.

Ding, G., Zhu, D., Sun, B., 2014. Formation control and obstacle avoidance of multi-AUV for 3-D underwater environment. In: Proceedings of the 33rd Chinese Control Conference. IEEE, pp. 8347–8352.

Do, K.D., Pan, J., 2009. Control of Ships and Underwater Vehicles: Design for Underactuated and Nonlinear Marine Systems. Springer.

Fujimoto, S., Hoof, H., Meger, D., 2018. Addressing function approximation error in actor-critic methods. In: International Conference on Machine Learning. PMLR, pp. 1587–1596.

Haarnoja, T., Tang, H., Abbeel, P., Levine, S., 2017. Reinforcement learning with deep energy-based policies. In: International Conference on Machine Learning. PMLR, pp. 1352–1361.

Haarnoja, T., Zhou, A., Hartikainen, K., et al., 2018. Soft actor-critic algorithms and applications. arXiv preprint arXiv:1812.05905.

Hagen, P.E., Størkersen, N., Vestgård, K., Kartvedt, P., Sten, G., 2003. Operational military use of the HUGIN AUV in Norway. In: Proc. UDT Europe 2003. pp. 123–130.

Havenstrøm, S.T., Rasheed, A., San, O., 2021. Deep reinforcement learning controller for 3D path following and collision avoidance by autonomous underwater vehicles. Front. Robot. AI 7, 211.

Huang, C.X., Pan, W., Chen, J., Wu, H.T., Wu, D.X., Xu, S.X., 2014. Simulation research on obstacle avoidance of autonomous underwater vehicle based on single beam ranging sonar. J. Xiamen Univ. (Nat. Sci.) 53 (4), 484–489.

Khalaji, A.K., Tourajizadeh, H., 2020. Nonlinear lyapounov based control of an underwater vehicle in presence of uncertainties and obstacles. Ocean Eng. 198, 106998.

Koenig, S., Likhachev, M., 2005. Fast replanning for navigation in unknown terrain. IEEE Trans. Robot. 21 (3), 354–363.

Li, M., Zhang, H., 2020. AUV 3D path planning based on A* algorithm. In: 2020 Chinese Automation Congress. CAC, IEEE, pp. 11–16.

Lillicrap, T.P., Hunt, J.J., Pritzel, A., et al., 2015. Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971.

Lim, H.S., Fan, S., Chin, C.K., Chai, S., Bose, N., 2020. Particle swarm optimization algorithms with selective differential evolution for AUV path planning. Int. J. Robot. Autom. 9 (2), 94–112.

Liu, Y., Wang, F., Lv, Z., Cao, K., Lin, Y., 2018. Pixel-to-action policy for underwater pipeline following via deep reinforcement learning. In: 2018 IEEE International Conference of Intelligent Robotic and Control Engineering. IRCE, IEEE, pp. 135–139.

MahmoudZadeh, S., Powers, D.M., Yazdani, A.M., Sammut, K., Atyabi, A., 2018. Efficient AUV path planning in time-variant underwater environment using differential evolution algorithm. J. Mar. Sci. Appl. 17 (4), 585–591.

Mnih, V., Kavukcuoglu, K., Silver, D., et al., 2015. Human-level control through deep reinforcement learning. Nature 518 (7540), 529–533.

Sagala, F., Bambang, R.T., 2011. Development of sea glider autonomous underwater vehicle platform for marine exploration and monitoring. Indian J. Geo-Mar. Sci. 40 (2), 287–295.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., Moritz, P., 2015. Trust region policy optimization. In: International Conference on Machine Learning. PMLR, pp. 1889–1897.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O., 2017. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347.

Sutton, R.S., Barto, A.G., 2018. Reinforcement Learning: An Introduction. MIT Press.

Taheri, E., Ferdowsi, M.H., Danesh, M., 2019. Closed-loop randomized kinodynamic path planning for an autonomous underwater vehicle. Appl. Ocean Res. 83, 48–64.

Wang, H., Wang, L., Li, J., Pan, L., 2013. A vector polar histogram method based obstacle avoidance planning for AUV. In: 2013 MTS/IEEE OCEANS-Bergen. IEEE, pp. 1–5.

Wang, H., Zhou, H., Yao, H., 2016. Research on autonomous planning method based on improved quantum Particle Swarm Optimization for Autonomous Underwater Vehicle. In: OCEANS 2016 MTS/IEEE Monterey. IEEE, pp. 1–7.

Wu, H., Song, S., Hsu, Y., You, K., Wu, C., 2019. End-to-end sensorimotor control problems of auvs with deep reinforcement learning. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS, IEEE, pp. 5869–5874.

Wynn, R.B., Huvenne, V.A., Le Bas, T.P., et al., 2014. Autonomous Underwater Vehicles (AUVs): Their past, present and future contributions to the advancement of marine geoscience. Mar. Geol. 352, 451–468.

Ziebart, B.D., 2010. Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy. Carnegie Mellon University.

Ziebart, B.D., Maas, A.L., Bagnell, J.A., Dey, A.K., 2008. Maximum entropy inverse reinforcement learning. In: Aaai. Chicago, IL, USA, pp. 1433–1438.