Robotica (2024), 1–25
doi:10.1017/S0263574724001802

RESEARCH ARTICLE

Research on obstacle avoidance of underactuated autonomous underwater vehicle based on offline reinforcement learning

Tao Liu[1,2], Junhao Huang[1] and Jintao Zhao[1]

[1]School of Ocean Engineering and Technology, Sun Yat-sen University & Southern Marine Science and Engineering Guangdong Laboratory (Zhuhai), Zhuhai, China
[2]Guangdong Provincial Key Laboratory of Information Technology for Deep Water Acoustics, Zhuhai, China
Corresponding author: Tao Liu; Email: liutao55@mail.sysu.edu.cn

Received: 20 June 2024; **Revised:** 14 August 2024; **Accepted:** 26 September 2024

Keywords: AUVs; offline rL; CQL; dynamic obstacle avoidance

Abstract
The autonomous navigation and obstacle avoidance capabilities of autonomous underwater vehicles (AUVs) are essential for ensuring their safe navigation and long-term, efficient operation. However, the complexity of the marine environment poses significant challenges to safe and effective obstacle avoidance. To address this issue, this study proposes an AUV obstacle avoidance control algorithm based on offline reinforcement learning. This method adopts the Conservative Q-learning (CQL) algorithm, which is based on the Soft Actor-Critic (SAC) framework. It learns from obtained historical obstacle avoidance data and ultimately achieves a favorable obstacle avoidance control strategy. In this method, PID and SAC control algorithms are utilized to generate expert obstacle avoidance data to construct a diversified offline database. Additionally, based on the line-of-sight (LOS) guidance method and artificial potential field (APF) method, information regarding the distance and orientation of targets and obstacles is incorporated into the state space, and heading and obstacle avoidance reward terms are integrated into the reward function design. The algorithm successfully guides the AUV in autonomous navigation and dynamic obstacle avoidance in three-dimensional space. Furthermore, the algorithm exhibits a certain degree of anti-interference capability against uncertain disturbances and ocean currents, enhancing the safety and robustness of the AUV system. Simulation results fully demonstrate the feasibility and effectiveness of the intelligent obstacle avoidance method based on offline reinforcement learning. This study highlights the profound significance of offline reinforcement learning in enabling robust and reliable control systems for AUVs, paving the way for enhanced operational capabilities in challenging marine environments.

1. Introduction

Autonomous underwater vehicles (AUVs), as advanced marine exploration tools, possess capabilities such as autonomous navigation, environmental perception, and data processing. These capabilities enable them to conduct precise and efficient detection tasks in complex and variable underwater environments. Currently, AUVs have emerged as the core system platform for marine resource exploration. With the advancements in science and technology, the levels of autonomy, intelligence, and safety of AUVs have also increased significantly [1, 2]. They are playing an increasingly crucial role in various fields, including marine resource exploration, marine environmental data collection, and marine security defense [3–5].

In AUV systems, the path planning and obstacle avoidance capabilities of the control system are pivotal technologies that ensure safe navigation and enable long-term, efficient operations under limited resource payloads. However, due to the strong coupling between various motion freedoms during its underwater movement, the inherent high nonlinearity of the AUV system, and the significant uncertainty of environmental loads imposed by the complex marine environment, enabling the AUV to safely and efficiently accomplish tasks such as obstacle avoidance and path tracking has become a formidable challenge [6, 7].

In the realm of AUV motion control algorithms, the proportional-integral-derivative (PID) control method is widely used in practical engineering due to its simplicity and rapid response speed. However, traditional PID algorithms face challenges such as difficult parameter tuning and poor robustness when dealing with AUV systems characterized by strong nonlinearity, strong coupling, and uncertainties in model parameters [8, 9]. Given these limitations, various control technologies are developing rapidly [10–14]. Reinforcement Learning (RL), as a cutting-edge artificial intelligence algorithm, has demonstrated significant advantages in the field of robotic motion control [15–17]. The core idea of RL lies in its ability to continuously learn and acquire optimal control strategies through the interaction between the controlled object and the environment, employing a trial-and-error and feedback-based approach. By continuously acquiring and computing feedback information regarding state transitions and rewards, the algorithm can continuously update its control strategies to adapt to the latest environmental conditions. This adaptive learning and data-driven nature enables reinforcement learning algorithms to handle nonlinear, time-varying, and uncertain systems, thus achieving more robust and efficient control in complex and variable underwater environments.

Currently, reinforcement learning algorithms have been widely applied in the fields of path tracking and obstacle avoidance decision-making for AUVs [18–21]. Wu H et al. proposed a model-free reinforcement learning algorithm for learning a state-feedback controller from AUV's sampled trajectories based on the deterministic policy gradient theorem and neural network approximation. This algorithm outperformed nonlinear model-based controllers in AUV's depth control [22]. Cui R et al. combined reinforcement learning with adaptive neural networks and proposed a control method for solving AUV's horizontal trajectory tracking problem. Through simulation experiments and comparisons with traditional PD control and neural network methods, the effectiveness and robustness of this method in trajectory tracking under external disturbances, model parameter uncertainties, and nonlinear effects of control inputs were verified [23]. Carlucho I et al. designed an adaptive controller based on deep reinforcement learning for AUV's low-level control. In addition to simulation experiments, they also conducted pool tests using the Nessie VII vehicle, demonstrating the feasibility of applying deep learning-based control methods to AUV's actual low-level control [24]. Jiang P et al. designed a novel control algorithm based on meta-RL, verifying its effectiveness in trajectory tracking control for AUVs in unknown and time-varying dynamic environments [25]. Ma D et al. integrated a neural network model with the traditional Actor-Critic structure and designed a Model Proximal Policy Optimization (Model PPO) algorithm for AUV's trajectory tracking in the presence of ocean current interference. The neural network model adopted by this algorithm can explore the spatiotemporal variation patterns of the AUV and its surrounding environment, outperforming PPO and MPC algorithms in terms of tracking accuracy and anti-interference capabilities [26]. Yuan J et al. proposed a horizontal autonomous obstacle avoidance method for AUVs based on the Double Deep Q-network (Double DQN) algorithm. This method utilizes deep reinforcement learning algorithms to learn from processed sonar information, enabling autonomous navigation for AUVs in complex dynamic obstacle environments [27]. Hadi B et al. presented an adaptive motion planning and obstacle avoidance method for AUVs based on Twin-delayed Deep Deterministic Policy (TD3). This method employs two deep learning agents for AUV's depth control and heading control. Simulation results show that this method can accurately guide the AUV towards the target while avoiding potential obstacles [28].

Although the advantages of reinforcement learning algorithms are evident, most control strategies are obtained through extensive online training in simulation environments. Due to the significant differences between complex marine environments and simulation environments, the strategies ultimately obtained may not achieve the desired performance in real-world environments. When training in real-world environments, online reinforcement learning algorithms face issues such as insufficient sampling efficiency, high trial-and-error costs, difficulties in deploying the algorithms to practical settings, and convergence challenges in high-dimensional spaces. Meanwhile, in complex marine environments, there are also problems such as the difficulty of data collection, high safety risks, and significant environmental uncertainties.

Offline reinforcement learning (Offline RL) is a reinforcement learning algorithm that can directly utilize offline data for policy training. Due to the existence of extrapolation error, online reinforcement learning algorithms cannot directly learn from offline data [29]. In offline RL, the algorithm does not need to interact with the environment in real-time, and if the offline data is sufficient and the adopted algorithm has strong exploration capabilities, it is enough to learn an excellent policy [30]. Nowadays, researchers have proposed various offline reinforcement learning algorithms, such as Batch-constrained Q-learning (BCQ), CQL, and model-based offline policy optimization [31–33]. The BCQ algorithm proposed in paper [29] even surpasses traditional reinforcement learning offline policy algorithms and imitation learning algorithms in some environments.

In the field of robotic control, offline reinforcement learning algorithms have unique advantages [30]. Besides avoiding safety risks brought by real-time interactions, they also allow for model training and optimization on powerful computing resources, without affecting the real-time performance of the robot system during actual operation. When the data samples are sufficiently diverse, the algorithm can learn more general and robust control strategies, thus better adapting to the complexity and variability of underwater environments. This generalization ability enables offline reinforcement learning algorithms to demonstrate better adaptability and stability when facing unknown or changing underwater environments. Currently, the research on AUV obstacle avoidance strategies based on offline RL is still in its initial stages.

To better address the obstacle avoidance problem of AUVs in complex marine environments, this study proposes an end-to-end obstacle avoidance control algorithm for AUVs based on the conservative Q-learning (CQL) algorithm. The algorithm adopts the Actor-Critic framework which is consistent with the Soft Actor-Critic (SAC) algorithm, leveraging deep neural networks to learn expert obstacle avoidance data from an offline database, ultimately yielding a high-performing obstacle avoidance policy. In this method, to construct a diversified offline database, PID control algorithms and SAC control algorithms are employed to generate expert-level obstacle avoidance data. Additionally, the state space and reward function are meticulously designed based on the line-of-sight guidance method and artificial potential field method, ultimately enabling AUVs to achieve autonomous navigation and dynamic obstacle avoidance. Furthermore, this obstacle avoidance policy exhibits a certain degree of resilience against external disturbances and ocean currents, demonstrating robust algorithmic capabilities. Simulation results comprehensively validate the feasibility and effectiveness of the proposed AUV intelligent obstacle avoidance method based on offline reinforcement learning.

The main contributions of this paper can be summarized as follows:

1) An AUV dynamic obstacle avoidance algorithm based on offline reinforcement learning is designed. By employing the CQL algorithm to train from suboptimal obstacle avoidance data, a well-performing obstacle avoidance policy is obtained, enabling autonomous navigation and dynamic obstacle avoidance for AUVs in three-dimensional space. This method adopts an offline training approach, avoiding safety risks associated with real-time interactions, and thus possesses high practical application value.

2) For AUV autonomous navigation and obstacle avoidance, information regarding the distance and orientation of targets and obstacles is introduced into the state space. Additionally, a heading reward term based on the LOS guidance method and an obstacle avoidance reward term based on the artificial potential field method are incorporated into the reward function design. This approach enhances the stability and robustness of the policy. Simulation results demonstrate that the proposed method possesses excellent anti-interference capabilities, achieving three-dimensional dynamic obstacle avoidance in complex environments.

The remainder of this paper is organized as follows: Section II describes the AUV models and formulates the reinforcement learning problem for obstacle avoidance, providing a comprehensive understanding of the problem domain and modeling approach. Section III elaborates on the CQL algorithm and the tailored design of the reward function for obstacle avoidance, elucidating the core

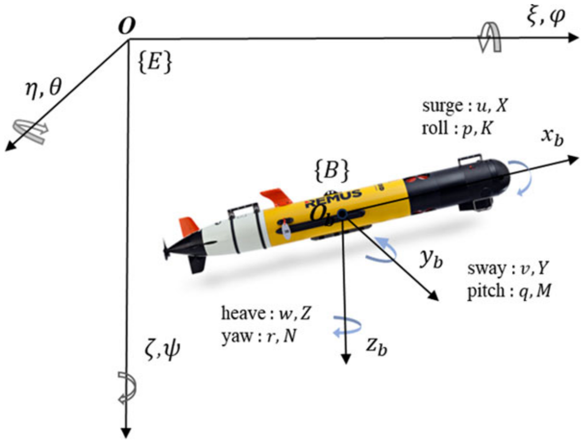


Figure 1. *Coordinate system of AUV.*

methodological aspects of the proposed solution. In Section IV, extensive simulations are conducted to evaluate the proposed obstacle avoidance method, with detailed results and analysis presented to validate its performance and highlight its advantages over existing approaches. Finally, Section V concludes the paper, summarizing the key findings and contributions, while also discussing potential future research directions.

2. Problem formulation

2.1. AUV motion model

To accurately describe the motion state of an AUV, it is necessary to select an appropriate reference frame to represent its movement. Commonly used reference frames include the geodetic coordinate system and the body coordinate system. The coordinate system is shown in Figure 1.

According to Figure 1, in the geodetic coordinate system, the positions of the underwater vehicle are denoted as ξ, η, ζ. The attitude of AUV (including roll, pitch, and yaw) is expressed as φ, θ, ψ. The motion of the AUV in six degrees of freedom is described by the following vectors:

$$\boldsymbol{\eta} = \begin{bmatrix} \boldsymbol{\eta}_1 & \boldsymbol{\eta}_2 \end{bmatrix}^T ; \boldsymbol{\eta}_1 = \begin{bmatrix} x & y & z \end{bmatrix}^T ; \boldsymbol{\eta}_2 = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T \tag{1}$$

$$\boldsymbol{v} = \begin{bmatrix} \boldsymbol{v}_1 & \boldsymbol{v}_2 \end{bmatrix}^T ; \boldsymbol{v}_1 = \begin{bmatrix} u & v & w \end{bmatrix}^T ; \boldsymbol{v}_2 = \begin{bmatrix} p & q & r \end{bmatrix}^T \tag{2}$$

where $\boldsymbol{\eta}_1 = \begin{bmatrix} x & y & z \end{bmatrix}^T$ and $\boldsymbol{\eta}_2 = [\phi \quad \theta \quad \psi]^T$ represent the position vector and attitude angle vector in the geodetic coordinate system respectively, $\boldsymbol{v}_1 = \begin{bmatrix} u & v & w \end{bmatrix}^T$ and $\boldsymbol{v}_2 = [p \quad q \quad r]^T$ represent the velocity vector and angular velocity vector in the body coordinate system respectively.

The kinematic equation is:

$$\dot{\boldsymbol{\eta}} = \boldsymbol{J}(\boldsymbol{\eta}_2)\boldsymbol{v} \tag{3}$$

with

$$\boldsymbol{J}(\boldsymbol{\eta}_2) = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{T} \end{bmatrix} \tag{4}$$

$$\boldsymbol{R} = \begin{bmatrix} \cos\psi\cos\theta & -\sin\psi\cos\phi + \cos\psi\sin\theta\sin\phi & \sin\psi\sin\phi + \cos\psi \\ \sin\psi\cos\theta & \cos\psi\cos\phi + \sin\phi\sin\theta\sin\psi & -\cos\psi\sin\phi + \sin\theta\sin\psi\cos\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix} \tag{5}$$

$$\boldsymbol{T} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \dfrac{\sin\phi}{\cos\theta} & \dfrac{\cos\phi}{\cos\theta} \end{bmatrix} \tag{6}$$

The dynamics equation can be expressed as below [34, 35]:

$$\boldsymbol{M}\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{v})\boldsymbol{v} + \boldsymbol{D}(\boldsymbol{v})\boldsymbol{v} + \boldsymbol{g}(\boldsymbol{\eta}) = \boldsymbol{\tau} + \boldsymbol{w} \tag{7}$$

where $\boldsymbol{\tau} = [X, Y, Z, K, M, N]^{\mathrm{T}}$ refers to the forces and moments generated by the AUV propulsion system, $\boldsymbol{w}$ stands for external disturbance. $\boldsymbol{M} = \mathrm{diag}(m - X_{\dot{u}}, m - Y_{\dot{v}}, m - Z_{\dot{w}}, I_x - K_{\dot{p}}, I_y - M_{\dot{q}}, I_z - N_{\dot{r}})$ is inertia coefficient matrix, $\boldsymbol{C}(\boldsymbol{v})$ is the Coriolis force matrix, $\boldsymbol{D}(\boldsymbol{v}) = \mathrm{diag}(X_u + X_{u|u|}|u|, Y_v + Y_{v|v|}|v|, N_r + N_{r|r|}|r|)$ is the fluid damping matrix, $\boldsymbol{g}(\boldsymbol{\eta})$ is restoring force vector produced by gravity and buoyancy and

$$\boldsymbol{C}(\boldsymbol{v}) = \begin{bmatrix} 0 & 0 & 0 & 0 & (m - Z_{\dot{w}})w & -(m - Y_{\dot{v}})v \\ 0 & 0 & 0 & -(m - Z_{\dot{w}})w & 0 & (m - X_{\dot{u}})u \\ 0 & 0 & 0 & (m - Y_{\dot{v}})v & -(m - X_{\dot{u}})u & 0 \\ 0 & (m - Z_{\dot{w}})w & -(m - Y_{\dot{v}})v & 0 & I_z\psi - N_{\dot{r}}r & -I_y\theta + M_{\dot{q}}q \\ -(m - Z_{\dot{w}})w & 0 & (m - X_{\dot{u}})u & -I_z\psi + N_{\dot{r}}r & 0 & I_x\varphi - K_{\dot{p}}p \\ (m - Y_{\dot{v}})v & -(m - X_{\dot{u}})u & 0 & I_y\theta - M_{\dot{q}}q & -I_x\varphi + K_{\dot{p}}p & 0 \end{bmatrix} \tag{8}$$

$$\boldsymbol{g}(\boldsymbol{\eta}) = \begin{bmatrix} (W - B)\sin\theta \\ -(W - B)\cos\theta\sin\phi \\ -(W - B)\cos\theta\cos\phi \\ -\left(y_g W - y_b B\right)\cos\theta\cos\phi + \left(z_g W - z_b B\right)\cos\theta\sin\phi \\ \left(z_g W - z_b B\right)\sin\theta + \left(x_g W - x_b B\right)\cos\theta\cos\phi \\ -\left(x_g W - x_b B\right)\cos\theta\sin\phi - \left(y_g W - y_b B\right)\sin\theta \end{bmatrix} \tag{9}$$

where m, W, and B represent the mass, gravity, and buoyancy of the AUV, respectively. I_x, I_y and I_z represent the inertial tensor.

This study will adopt the REMUS AUV model designed in ref. [36], whose excellent maneuverability and stability have been verified in both simulation and real-world environments. Since the roll angle ϕ is generally small compared to the pitch angle θ and yaw angle ψ when the AUV performs obstacle avoidance tasks, we assume $\phi = p = 0$ in the simulation experiments conducted in this paper to simplify the calculations.

2.2. Reinforcement learning

In RL, the process of an agent interacting with its environment is modeled as a Markov Decision Process (MDP). The interaction loop between the agent and the environment is illustrated in Figure 2. In the MDP framework, the model is typically defined by the tuple $< S, A, P, r, \gamma >$. Specifically, $\boldsymbol{S}$ represents the state space, $\boldsymbol{A}$ denotes the action space, $P(\boldsymbol{s}'|\boldsymbol{s}, \boldsymbol{a})$ is the state transition function, $r(\boldsymbol{s}, \boldsymbol{a})$ is the reward function, and $\gamma \in [0, 1]$ is the discount factor. Starting from a state $\boldsymbol{S_t}$ at time t, the sum of all discounted rewards is referred to as the return, which is expressed as:

$$G_t = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k} \tag{10}$$

The objective of a MDP is to find an optimal policy π^* for an agent to maximize the return obtained from the initial state. We utilize the state-value function $V^{\pi}(\boldsymbol{s})$ and the action-value function $Q^{\pi}(\boldsymbol{s}, \boldsymbol{a})$

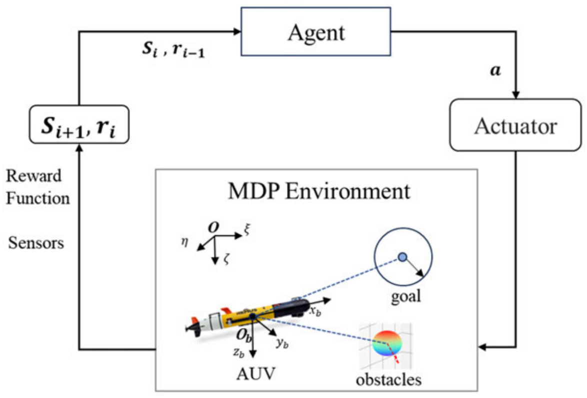


Figure 2. *The agent interacts with the MDP environment.*

to measure the value of a state and an action, respectively. The state-value function is defined as the expected return obtained when starting from state $\boldsymbol{s}$ and following policy π, which can be expressed as

$$V^{\pi}(s) = E_{\pi}[G_t | s_t = s] \tag{11}$$

$Q^{\pi}(\boldsymbol{s}, \boldsymbol{a})$ is defined as the expected return obtained by executing action $\boldsymbol{a}$in the current state $\boldsymbol{s}$ while following policy π. The mathematical expression is as follows:

$$Q^{\pi}(\boldsymbol{s}, \boldsymbol{a}) = E_{\pi}[R_t | \boldsymbol{s}_t = \boldsymbol{s}, \boldsymbol{a}_t = \boldsymbol{a}] \tag{12}$$

The relationship between $V^{\pi}(\boldsymbol{s})$ and $Q^{\pi}(\boldsymbol{s}, \boldsymbol{a})$ can be expressed as:

$$V^{\pi}(\boldsymbol{s}) = \sum_{a \in \mathcal{A}} \pi(\boldsymbol{a} \mid \boldsymbol{s})\, Q^{\pi}(\boldsymbol{s}, \boldsymbol{a}) \tag{13}$$

$V^{\pi}(\boldsymbol{s})$ and $Q^{\pi}(\boldsymbol{s}, \boldsymbol{a})$ can be iteratively updated through the Bellman Expectation Equation to find the π^*. The Bellman Expectation Equation is:

$$Q^{\pi}(\boldsymbol{s}, \boldsymbol{a}) = E_{\pi}\left[R_t + \gamma Q^{\pi}(\boldsymbol{s}', \boldsymbol{a}') \mid \boldsymbol{S}_t = \boldsymbol{s}, \boldsymbol{A}_t = \boldsymbol{a}\right]$$

$$= r(\boldsymbol{s}, \boldsymbol{a}) + \gamma \sum_{s' \in \mathcal{S}} P(\boldsymbol{s}' \mid \boldsymbol{s}, \boldsymbol{a}) \sum_{a' \in \mathcal{A}} \pi(\boldsymbol{a}' \mid \boldsymbol{s}')\, Q^{\pi}(\boldsymbol{s}', \boldsymbol{a}')$$

$$V^{\pi}(\boldsymbol{s}) = \mathrm{E}_{\pi}\left[R_t + \gamma V^{\pi}(\boldsymbol{s}') \mid \boldsymbol{S}_t = \boldsymbol{s}\right] \tag{14}$$

$$= \sum_{a \in \mathcal{A}} \pi(\boldsymbol{a} | \boldsymbol{s}) \left(r(\boldsymbol{s}, \boldsymbol{a}) + \gamma \sum_{s' \in \mathcal{S}} P(\boldsymbol{s}' | \boldsymbol{s}, \boldsymbol{a})\, V^{\pi}(\boldsymbol{s}') \right) \tag{15}$$

In RL, many algorithms are designed based on the Actor-Critic framework. In this framework, the Actor and Critic are typically implemented as deep neural networks, where the Actor computes the policy function $\pi(\boldsymbol{s})$ and the Critic computes the value function $Q^{\pi}(\boldsymbol{s}, \boldsymbol{a})$. The relationship between the Actor network and Critic networks is illustrated in Figure 3.

2.3. Offline reinforcement learning

Different from Online RL, Offline RL can directly use offline data for policy training. The differences between offline RL, online policy RL, and offline policy RL are shown in Figure 4.

Both on-policy algorithms and off-policy algorithms are online reinforcement learning algorithms. They collect data through real-time interaction with the environment. The difference is that on-policy algorithms immediately utilize this data to update their target policy, while off-policy algorithms store

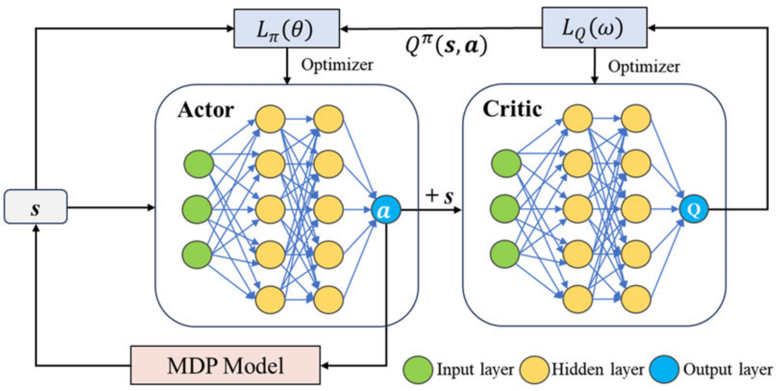


Figure 3. *The relationship between Actor and Critic.*

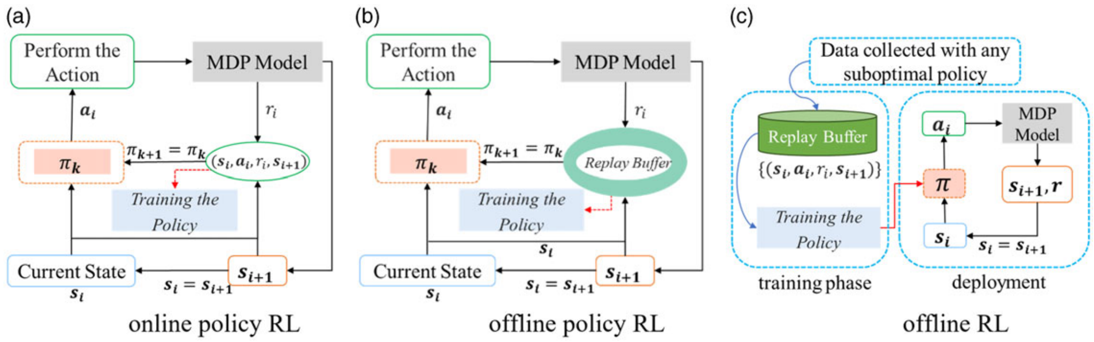


Figure 4. *The differences between offline RL, online policy RL, and offline policy RL.*

the data in a database called replay buffer for secondary training. This mechanism allows off-policy algorithms to make more use of historical information and improve the generalization capability of the policy. However, due to the presence of extrapolation errors, off-policy algorithms cannot be directly applied to offline environments, even if the replay buffer contains expert data [29]. The advantage of offline reinforcement learning lies in its ability to directly utilize existing offline data for policy training by reducing extrapolation errors, thus reducing the high cost of trial-and-error that results from direct interaction with the environment.

3. The proposed method

In this controller, we employ the CQL algorithm based on the SAC framework as our baseline method to realize the obstacle avoidance control of AUV. By initializing the environmental model, policy, and value networks, and collecting data through interactions with the environment, the conservative term mechanism of CQL is utilized to constrain the overestimation of the value function, thereby enhancing the stability of the algorithm. During the training process, the value network and policy network are alternately updated, incorporating the entropy regularization term of SAC to encourage exploration. The AUV selects actions based on the updated policy and detects obstacles in real time through sensors to avoid collisions. Through continuous iterative training and testing, the policy network is optimized, ultimately achieving efficient and safe obstacle avoidance control for AUVs. The specific implementation flowchart of this algorithm is shown in Figure 5.

Due to the complexity of the 3D dynamic obstacle avoidance model for AUVs, many reinforcement learning algorithms fail to learn effectively. SAC, as an off-policy algorithm, introduces the concept

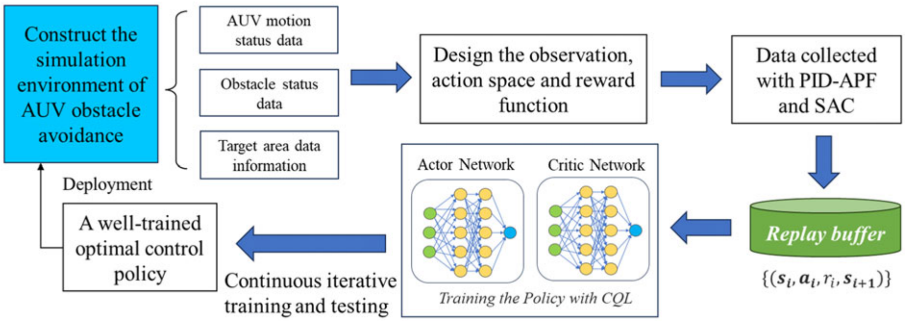

Construct the simulation environment of AUV obstacle avoidance
AUV motion status data
Obstacle status data
Target area data information
Design the observation, action space and reward function
Data collected with PID-APF and SAC
Replay buffer
$\{(s_i, a_i, r_i, s_{i+1})\}$
Actor Network
Critic Network
Training the Policy with CQL
Continuous iterative training and testing
A well-trained optimal control policy
Deployment

Figure 5. *Implementation flowchart of AUV obstacle avoidance algorithm based on CQL.*

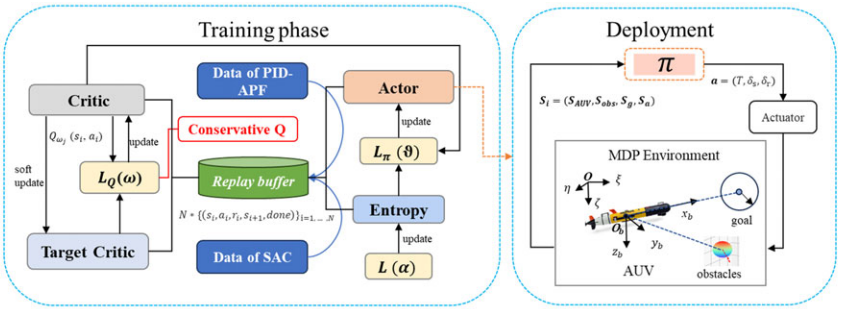


Figure 6. *Structure of AUV obstacle avoidance controller based on CQL.*

of entropy into the Actor-Critic framework, enhancing the exploration capability of the policy. This exploration mechanism improves the adaptability of SAC in complex environments, prevents the policy from prematurely converging to local optima, and thus enhances the stability of the model. The CQL algorithm, which utilizes SAC as its foundation, inherits SAC's exploration capabilities and exhibits superior performance in complex multi-dimensional environments.

During the training phase of CQL, we generate expert obstacle avoidance data using both SAC-based obstacle avoidance algorithms and PID-APF-based obstacle avoidance algorithms. The SAC-based algorithm is an end-to-end approach that directly controls the actuators based on environmental information obtained from sensors. This algorithm employs the same state space, action space, and reward function as the CQL algorithm, and achieves the final obstacle avoidance policy through training in a simulation environment. The PID-APF algorithm, a more traditional approach, utilizes the APF algorithm to generate safe paths based on real-time environmental information, which are then tracked and controlled by the PID algorithm. We incorporate obstacle avoidance data obtained from both algorithms into the offline database to increase data diversity. This diversified data is conducive to obtaining policies with stronger generalization capabilities.

In the 3D obstacle avoidance environment for AUVs, the agent's primary objective can be summarized as reaching the predefined target safely. To achieve this goal, we design a reward function that includes a goal-oriented reward and a LOS navigation reward to assist the AUV in reaching the desired target. Additionally, we introduce an APF-based obstacle avoidance reward to evade obstacles. Furthermore, the reward function incorporates terms related to angular velocity to enhance training stability and a control signal limitation term to reduce unnecessary energy consumption.

The structure of the proposed AUV obstacle avoidance controller grounded in the CQL approach is illustrated in Figure 6.

3.1. Soft Actor-Critic

The Soft Actor-Critic (SAC) algorithm forms the backbone of the methodology, comprising one Actor network and four Critic networks. Among these, two Critic networks are designated as training networks, each accompanied by a target Q-network to enhance training stability and convergence.

The loss functions for the two Critic training networks, denoted by $j = 1, 2$, are defined as:

$$L_Q(\omega) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - Q_{\omega_j}(\boldsymbol{s}_i, \boldsymbol{a}_i)\right)^2 \tag{16}$$

where y_i represents the target Q-value obtained from the Bellman Equation, and Q_{ω_j} is the current Q-value estimate produced by the jth Critic network for the state-action pair $(\boldsymbol{s}_i, \boldsymbol{a}_i)$.

The loss function of the Actor network, responsible for learning the optimal policy, is formulated as:

$$L_\pi(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left(\alpha \log \pi_\theta(\tilde{\boldsymbol{a}}_i|\boldsymbol{s}_i) - \min_{j=1,2} Q_{\omega_j}(\boldsymbol{s}_i, \tilde{\boldsymbol{a}}_i)\right) \tag{17}$$

where $\tilde{a}_i$ is the action obtained through reparametrized sampling from the Actor network, and π_θ represents the Actor network with parameters θ, aiming to maximize the minimum Q-value over the two Critic networks.

To encourage exploration of the state space and prevent premature convergence to suboptimal policies, SAC introduces an entropy regularization term. By minimizing the following loss function, the entropy regularization coefficient α can be adapted automatically, striking a balanced tradeoff between exploration and value function improvement:

$$L(\alpha) = \mathrm{E}_{\boldsymbol{s}_t \sim R, \boldsymbol{a}_t \sim \pi(\cdot|\boldsymbol{s}_t)}\left[-\alpha \log \pi(\boldsymbol{a}_t|\boldsymbol{s}_t) - \alpha H_0\right] \tag{18}$$

where H_0 is the target entropy, typically set to the negative of the action space dimensionality to encourage maximum entropy exploration.

The loss functions $L_Q(\omega)$, $L_\pi(\theta)$, and $L(\alpha)$ are minimized, and the training network parameters and the entropy regularization coefficient α are iteratively updated using stochastic gradient descent or other optimization techniques.

To ensure stable training and mitigate the divergence of the target Q-networks from their respective training networks, a soft update approach is adopted:

$$\begin{aligned} \omega_1^- &\leftarrow \tau\omega_1 + (1-\tau)\,\omega_1^- \\ \omega_2^- &\leftarrow \tau\omega_2 + (1-\tau)\,\omega_2^- \end{aligned} \tag{19}$$

where $\tau \in (0, 1)$ is a hyperparameter controlling the rate at which the target networks are updated, typically set to a small value (e.g., 0.005) to facilitate smooth transitions.

3.2. Conservative Q-learning based on SAC

While the adopted CQL algorithm inherits the Actor-Critic architecture from SAC, a key distinction lies in the introduction of a conservative term and a compensation term into the Q-function optimization objective Q. These enhancements are designed to mitigate the detrimental effects of extrapolation errors, a common issue in value-based reinforcement learning methods when encountering out-of-distribution state-action pairs.

The general iterative equation for the Q in CQL can be expressed as:

$$\tilde{Q}^{k+1} \leftarrow \arg\min_{\mathrm{Q}} \mathrm{E}_{(\boldsymbol{s},\boldsymbol{a})\sim D}\left[\left(Q(\boldsymbol{s},\boldsymbol{a}) - \hat{\boldsymbol{B}}^{\pi}\hat{Q}^k(\boldsymbol{s},\boldsymbol{a})\right)^2\right] \tag{20}$$

where $\hat{\boldsymbol{B}}^{\pi}$ represents the Bellman operator for the policy π, and D is the offline dataset containing expert demonstrations or previously collected experience.

To prevent the overestimation of Q-values, which can lead to suboptimal or erratic behavior, CQL imposes a constraint on the Q-values for actions following a specific distribution $\mu(\boldsymbol{a}, \boldsymbol{s})$. This distribution is typically chosen to be a broad, data-covering distribution, such as a Gaussian or a uniform distribution. Additionally, a compensation term is applied to the Q-values of data conforming to the behavior policy π_b that generated the offline dataset. This compensation term prevents excessive restriction of the Q-values, ensuring sufficient exploration and policy improvement. Furthermore, a regularization term $R(\mu)$ is introduced to mitigate overfitting to the specific distribution $\mu(\boldsymbol{a}, \boldsymbol{s})$, promoting generalization to unseen state-action pairs. The iterative CQL Q-function update equation, incorporating these enhancements, becomes:

$$\hat{Q}^{k+1} \leftarrow \arg\min_{\mathrm{Q}} \beta\big(\mathrm{E}_{s\sim D, \boldsymbol{a}\sim\mu(\boldsymbol{a}|\boldsymbol{s})}\left[Q(\boldsymbol{s}, \boldsymbol{a}) - \mathrm{E}_{s\sim \boldsymbol{D}, \boldsymbol{a}\sim\hat{\pi}_b(\boldsymbol{a}|\boldsymbol{s})}\left[Q(\boldsymbol{s}, \boldsymbol{a})\right]\right]$$

$$+\frac{1}{2}\mathrm{E}_{(s,\boldsymbol{a})\sim D}\left[\left(Q(\boldsymbol{s}, \boldsymbol{a}) - \hat{B}^{\pi}\hat{Q}^{k}(\boldsymbol{s}, \boldsymbol{a})\right)^2\right] + R(\mu)\Big) \tag{21}$$

where β is a balancing factor that controls the tradeoff between the conservative constraint and the compensation term, and $\hat{\pi}_b$ approximates the behavior policy π_b that generated the offline data.

The regularization term $R(\mu)$ is adopted as the Kullback-Leibler divergence from a prior policy $\rho(\boldsymbol{a}|\boldsymbol{s})$, where $\rho(\boldsymbol{a}|\boldsymbol{s})$ is typically chosen as a uniform distribution to promote broad exploration. This formulation yields the final iterative CQL Q-function update equation:

$$\hat{Q}^{k+1} \longleftarrow \arg\min_{Q} \beta\mathrm{E}_{s\sim D}\left[\log\sum_{a}\exp\left(Q(\boldsymbol{s}, \boldsymbol{a})\right) - \mathrm{E}_{\boldsymbol{a}\sim\hat{\pi}_{\mathrm{b}}(\boldsymbol{a}|\boldsymbol{s})}\left[Q(\boldsymbol{s}, \boldsymbol{a})\right]\right]$$

$$+\frac{1}{2}\mathrm{E}_{(s,\boldsymbol{a})\sim D}\left[\left(Q(\boldsymbol{s}, \boldsymbol{a}) - \hat{B}^{\pi}\hat{Q}^{k}(\boldsymbol{s}, \boldsymbol{a})\right)^2\right] \tag{22}$$

This formulation, combining the conservative constraint, compensation term, and Bellman error minimization, enables obtaining a robust and reliable Q-function estimate that mitigates overestimation biases. Subsequently, this estimated Q-function facilitates the generation of an effective obstacle avoidance policy for AUVs operating in complex environments.

The pseudo-code of the training process of AUV obstacle avoidance algorithm based on CQL is as follows:

3.3. State-action space and reward function

Next, we meticulously design the three-dimensional dynamic obstacle avoidance problem of AUV in the training phase, which mainly includes the design of the state space, action space, and reward function formulations.

a) State space and action space

The state space refers to the set of all possible system states, where each state encapsulates all relevant information about the environment or system at the current moment. In reinforcement learning, the agent makes decisions based on the current state; therefore, a clear definition of the state space is a prerequisite to ensure that the agent can effectively learn and solve problems. For the three-dimensional dynamic obstacle avoidance problem of AUV within the MDP model, comprehensive considerations are given to the AUV's motion, navigation, obstacle avoidance, and energy consumption control. The information of the state space S mainly includes the motion state information of the AUV $\boldsymbol{S}_{AUV}$, obstacle information $\boldsymbol{S}_{obs}$, target information $\boldsymbol{S}_{g}$, and control signal information $\boldsymbol{S}_{a}$.

The research object of this paper is the underactuated REMUS AUV, whose control inputs for the actuator units are $u = [T, \delta_s, \delta_r]$, where $T\epsilon(3\mathrm{N}, 6\mathrm{N})$ represents the thrust of the propeller. $\delta_s\epsilon\left(-\frac{\pi}{6}, \frac{\pi}{6}\right)$ denotes the horizontal rudder angle, and $\delta_r\epsilon\left(-\frac{\pi}{6}, \frac{\pi}{6}\right)$ represents the directional rudder angle. To ensure the smooth output of the control signals as much as possible, we define the action space as an incremental

Algorithm 1. *AUV obstacle avoidance algorithm based on CQL.*

The training process of the AUV obstacle avoidance algorithm based on CQL:

```
1.Obtain the obstacle avoidance data of APF and SAC to generate replay buffer $D$
2.Randomly initialize the Actor network $\pi_\vartheta(\boldsymbol{s})$, the Critic networks $Q_{\omega_1}(\boldsymbol{s},\boldsymbol{a})$ and $Q_{\omega_2}(\boldsymbol{s},\boldsymbol{a})$, as well as the entropy regularization coefficient $\alpha$.
3. Copy the parameters $\omega_1^- \leftarrow \omega_1$, $\omega_2^- \leftarrow \omega_2$, and initialize the target networks $Q_{\omega_1^-}$ and $Q_{\omega_2^-}$ accordingly.
4. for $k = 1 \to K$ do
5. Sample $N * \{(\boldsymbol{s}_i, \boldsymbol{a}_i, , r_i, \boldsymbol{s}_{i+1}, done)\}_{i=1,\ldots,N}$ from $D$.
6. Update the Critic networks for $\omega = \omega_1, \omega_2$:
   $\omega_t \leftarrow \omega_{t-1} - \eta_Q \nabla_\omega \Bigg( \alpha \cdot \mathrm{E}_{s\sim D}\Bigg[\log \sum_{\boldsymbol{a}} \exp(Q_\omega(\boldsymbol{s},\boldsymbol{a})) - \mathrm{E}_{\boldsymbol{a}-\hat{\pi}_{\boldsymbol{b}}(\boldsymbol{a}|\boldsymbol{s})}\left[Q_\omega(\boldsymbol{s},\boldsymbol{a})\right]\Bigg] + \frac{1}{2}\mathrm{E}_{(\boldsymbol{s},\boldsymbol{a})-D}\left[\left(Q(\boldsymbol{s},\boldsymbol{a}) - \hat{B}^\pi Q_\omega(\boldsymbol{s},\boldsymbol{a})\right)^2\right]\Bigg)$
7. Update the Actor networks:
   $\vartheta_t \leftarrow \vartheta_{t-1} - \eta_\pi \nabla_\vartheta \mathrm{E}_{s\sim D,\boldsymbol{a}\sim\pi_\vartheta(\boldsymbol{a}|\boldsymbol{s})}[\alpha \log \pi_\vartheta(\boldsymbol{a}|\boldsymbol{s}) - \min_{j=1,2} Q_{\omega_j}(\boldsymbol{s},\boldsymbol{a})]$
8. Update $\alpha$:
   $\alpha_t \leftarrow \alpha_{t-1} - \eta_\alpha \nabla_\alpha E_{s\sim D,\boldsymbol{a}\sim\pi_\vartheta(a|s)}[-\alpha_{t-1}\log \pi_\vartheta(\boldsymbol{a}|\boldsymbol{s}) - \alpha_{t-1}H_0]$
9. Soft update the Critic target networks:
   $\omega_1^- \leftarrow \tau\omega_1 + (1-\tau)\omega_1^-$
   $\omega_2^- \leftarrow \tau\omega_2 + (1-\tau)\omega_2^-$
10. end for
```

form, specifically as follows:

$$\boldsymbol{a} = [\Delta T, \Delta\delta_s, \Delta\delta_r] \tag{23}$$

with $\Delta T\epsilon(-1\text{N}, 1\text{N})$, $\Delta\delta_s$, $\Delta\delta_r\epsilon(-1^\circ, 1^\circ)$.

The input to the actuator is updated as:

$$u(t) = u(t-1) + \boldsymbol{a} \tag{24}$$

Considering the control of AUV's position and velocity, the observation of the AUV is set as follows:

$$\boldsymbol{S_{AUV}} = \left[u, v, w, q, r, \theta, \psi\right] \tag{25}$$

To comprehensively account for the shape, size, velocity, and position of obstacles, and to ensure the transferability of the obstacle avoidance policy to different environments, the AUV's observation of obstacles is set as:

$$\boldsymbol{S_{obs}} = \left[\theta_o^i, \psi_o^i, d_o^i, r_o^i, \boldsymbol{v}_o\right] \tag{26}$$

where r_o^i and $\boldsymbol{v}_o = [u_o^i, v_o^i, w_o^i]^T$ represent the size and velocity of obstacle, respectively. The positions of the center of gravity of the AUV P_G and the obstacle's centroid P_o in the geodetic coordinate system are given by (ξ, η, ζ) and (ξ_o, η_o, ζ_o), respectively. $\theta_o = -\arctan\left(\frac{\zeta_o - \zeta}{\sqrt{(\xi_o-\xi)^2+(\eta_o-\eta)^2}}\right)$ and $\psi_o = \arctan 2(\eta_o - \eta, \xi_o - \xi)$ denote the angles from P_o to P_G, respectively. $d_o = \sqrt{(\xi_o-\xi)^2 + (\eta_o-\eta)^2 + (\zeta_o-\zeta)^2}$ is the distance between P_o and P_G. $i = 1, 2$ refer to the two obstacles closest to the AUV.

Assuming the target position is $p_g = (\xi_g, \eta_g, \zeta_g)$, the target information $\boldsymbol{S_g}$ can be set as:

$$\boldsymbol{S_g} = \left[\theta_g, \psi_g, d_g\right] \tag{27}$$

To avoid frequent rudder operations, the output of the actuator from the previous step is included in the state space, represented as:

$$\boldsymbol{S_a} = u(t-1) = \left[T^{t-1}, \delta_s^{t-1}, \delta_r^{t-1}\right] \tag{28}$$

Ultimately, the state $\boldsymbol{S}$ is the concatenation of all components:

$$\boldsymbol{S} = \boldsymbol{S_{AUV}} + \boldsymbol{S_{obs}} + \boldsymbol{S_g} + \boldsymbol{S_a} \tag{29}$$

b) Reward function

Within the framework of reinforcement learning, the reward function plays an indispensable and pivotal role. It not only serves as a fundamental cornerstone that constitutes the learning cycle of the agent but also directly shapes and determines the behavioral decision-making trajectory of the agent as well as its ultimate performance outcomes. The robustness of algorithmic convergence and the quality of the trained strategy, both hinge crucially on the meticulous design of the reward function. In the MDP model for three-dimensional obstacle avoidance, the reward function r can be designed to comprise five components: the goal-oriented reward r_g, the line-of-sight (LOS) navigation reward $r_{los} = r_\psi + r_\theta$, the obstacle avoidance reward r_{obs}, the angular velocity reward r_q and r_r, and the control signal smoothness reward r_T, r_{δ_s}, and r_{δ_r}.

1. Goal-oriented reward r_g

$$r_g = \begin{cases} 1000, & d_g < \varepsilon_g \\ -k_1 d_g, & d_g \geq \varepsilon_g \end{cases} \tag{30}$$

where ε_g represents the allowable error, and k_1 is a positive constant.

2. LOS navigation reward r_ψ and r_θ

The target angle calculation formula for LOS guidance is:

$$\psi_g = \arctan 2\left(\eta_g - \eta, \xi_g - \xi\right) + \beta \tag{31}$$

$$\theta_g = -\arctan\left(\frac{\zeta_g - \zeta}{\sqrt{\left(\xi_g - \xi\right)^2 + \left(\eta_g - \eta\right)^2}}\right) \tag{32}$$

where $\beta = \arctan\left(\frac{v}{\sqrt{u^2+w^2}}\right)$ is the sideslip angle of the AUV. And the LOS reward is defined as:

$$r_{los} = r_\psi + r_\theta = -k_{21}{\varepsilon_\psi}^2 - k_{22}{\varepsilon_\theta}^2 \tag{33}$$

where $\varepsilon_\psi = \psi_d - \psi$ and $\varepsilon_\theta = \theta_d - \theta$, k_{21} and k_{22} are positive constants.

c) Obstacle avoidance reward r_{obs}

The obstacle avoidance reward function is based on artificial potential field method, which is expressed as follows:

$$r_{obs} = \begin{cases} \max\left\{-\frac{1}{2}k_3 R_o\left(\frac{1}{d_o + 0.1} - \frac{1}{\rho_o}\right)^2 - \right. \\ \left. k_3 R_o\left(\frac{1}{d_o + 0.1} - \frac{1}{\rho_o}\right)\frac{d_g}{d_o}, -1000\right\}, 0 \leq d_o \leq \rho_o \\ 0, d_o > \rho_o \end{cases} \tag{34}$$

where R_o is a factor measuring the size of the obstacle, and for spherical obstacles, R_o is the radius. d_o is the minimum distance between the AUV and the obstacle surface. k_3 and ρ_o are positive constants.

d) Angular velocity reward r_q and r_r

The underactuated AUV system is a complex nonlinear model with coupled degrees of freedom. Due to the influence of the restoring torque, the AUV is prone to getting stuck in circular motion during the training process. Therefore, angular velocity rewards are implemented, specifically as follows:

$$r_q = \begin{cases} -k_{41}q^2, \varepsilon_\theta \le 1 \textit{ and } |q| > 0.1 \\ \dfrac{-1}{(|q| + 0.1)}, \varepsilon_\theta > 1 \textit{ and } |q| < 0.1 \end{cases} \tag{35}$$

$$r_r = \begin{cases} -k_{42}r^2, \varepsilon_\psi \le 1 \textit{ and } |r| > 0.1 \\ \dfrac{-1}{(|r| + 0.1)}, \varepsilon_\psi > 1 \textit{ and } |r| < 0.1 \end{cases} \tag{36}$$

where k_{41} and k_{42} are positive constants.

e) Control signal smoothness reward r_T, r_{δ_s}, and r_{δ_r}

To smoothen the control signal output, the reward function penalizes sudden changes in the control signal, represented by the difference between the current action and the moving average of the actions over the previous τ_1 steps. Specifically, this is achieved as follows:

$$\begin{cases} r_T = -k_{51}\left|T^{t-\tau_1\,:\,t-1} - T^t\right| \\ r_{\delta_s} = -k_{52}\left|\delta_s^{t-\tau_1\,:\,t-1} - \delta_s^t\right| \\ r_{\delta_r} = -k_{53}\left|\delta_r^{t-\tau_1\,:\,t-1} - \delta_r^t\right| \end{cases} \tag{37}$$

where k_{51}, k_{52}, and k_{53} are positive constants. Additionally, a small positive constant k is introduced to limit the magnitude of the reward function, accelerating the algorithm's convergence speed. The overall reward function is given as:

$$r = k \times \left(r_g + r_{los} + r_{obs} + r_q + r_r + r_T + r_{\delta_s} + r_{\delta_r}\right) \tag{38}$$

This comprehensive formulation of the state space, action space, and reward function tailored for the AUV's three-dimensional dynamic obstacle avoidance task provides a solid foundation for the CQL algorithm to learn an effective obstacle avoidance policy. The careful design considerations, such as incorporating relevant information, addressing underactuation challenges, and promoting smooth control signals, aim to enhance the controller's performance, stability, and transferability to complex environments.

4. Numerical simulations and analysis

On the basis of the theoretical foundation, the subsequent section is dedicated to numerical simulations and result analysis, aiming to validate the control performance of the proposed method. The primary objective of these simulations is to evaluate the feasibility and effectiveness of the offline reinforcement learning framework in switching training and implementing obstacle avoidance functions. By carefully examining the obtained simulation results, valuable insights will be generated to further validate the applicability of this method in complex underwater environments. This evaluation is crucial for determining the method's suitability and drawing scientifically sound conclusions about its performance in

Table I. *Specific values of the AUV's inertia and hydrodynamic parameters.*

Parameters	Value	Parameters	Value
m	30.48 kg	$X_{\dot{u}}$	−0.93 kg
I_{zz}	3.45 kg·m^2	$Y_{\dot{v}}$	−35.5 kg
Y_δ	9.64 kg/(m · rad)	$Y_{\dot{r}}$	1.93 kg·m/rad
N_δ	−6.15 kg/rad	$N_{\dot{r}}$	−4.88 kg·m^2/rad
X_{vr}	35.5 kg/rad	$N_{\dot{v}}$	1.93 kg·m
X_{rr}	−1.93 kg·m/rad	$X_{u\|u\|}$	−1.62 kg/m
Y_{ur}	5.22 kg/rad	$Y_{v\|v\|}$	−1310 kg/m
Y_{uv}	−28.6 kg/m	$Y_{r\|r\|}$	0.632 kg·m/rad^2
N_{ur}	−2 kg·m/rad	$N_{v\|v\|}$	−3.18 kg
N_{uv}	−24 kg	$N_{r\|r\|}$	−94 kg·m^2/rad^2

various scenarios. In this study, the classical RUMUS was employed as our simulation model. And Table I furnishes us with precise values concerning the AUV's inertia and hydrodynamic parameters. To comprehensively evaluate the trained agent, several modes were considered.

4.1. Parameter setting and policy training

In this study, a three-dimensional obstacle avoidance simulation environment was established for the REMUS AUV using Python. Within this simulation environment, the AUV performs autonomous navigation towards randomly selected target points in a three-dimensional space while simultaneously avoiding potential obstacles. The position, velocity, and heading of the AUV are updated using equations (3) and (7) which model the AUV's kinematic and dynamic behavior.

Firstly, a SAC-based obstacle avoidance control algorithm was designed, adopting the same state space, action space, and reward function as the previously designed CQL obstacle avoidance algorithm. This SAC algorithm was then trained in the simulation environment, resulting in a suboptimal obstacle avoidance control policy. Secondly, to construct an offline database storing expert obstacle avoidance data, the suboptimal policy obtained from the SAC algorithm training was employed along with another control algorithm based on PID control and APF. These algorithms were used to conduct numerous obstacle avoidance simulations. During the data acquisition process, the starting point of the AUV was set at(0m, 0m, 0m)in the geodetic coordinate system, with an initial heading angle ranging from $-\pi$ to π radians. AUV thrust is limited to 3N to 6N. The obstacles were randomly positioned with a spherical shape and radius ranging from 0mto 4m. The goal of each simulation mission was for the AUV to safely reach a randomly set target region G, where the target position ranged from −30m to 30m on each axis, and the target region had a radius of 2m. The duration of each mission was limited to within 800 time steps, terminating when the AUV reached the target point or the step limit was exceeded. During each simulation, detailed logs of the AUV's observation, actions taken, rewards received, and obstacle encounters were recorded. This data was critical for analyzing the performance of the SAC and PID-APF algorithms and later training the CQL algorithm in an offline manner.

Lastly, the simulation data procured from the SAC and PID-APF algorithms is employed to train the CQL algorithm, and the network architecture and training parameters are summarized in Table II. After 3000 iterations of training, an optimal policy was obtained, outperforming both the SAC algorithm and the PID-APF algorithm used to generate the original obstacle avoidance data. The reward curve illustrating the training process is shown in Figure 7. To demonstrate the obstacle avoidance effect of the trained policy, visualizations were provided, where the blue square represents the starting point of the AUV, the green triangle represents the target position, the red solid line represents the movement trajectory of the AUV, and the blue dashed line represents the direct line between the starting and ending points.

Table 2. *Training parameters of the AUV obstacle avoidance algorithm based on CQL.*

Parameters	**Value**
Actor learning rate	0.0001
Critic learning rate	0.0001
α learning rate	0.0001
Hidden layers of Actor and Critic	4
Number of neurons in each hidden layer	128 units
Discount factor γ	0.99
Soft update parameterτ	0.01
Target entropy H_0	-3
Equilibrium factor β	5
Batch size N	256
The number of trainings per epoch	256
Time step	0.1 s

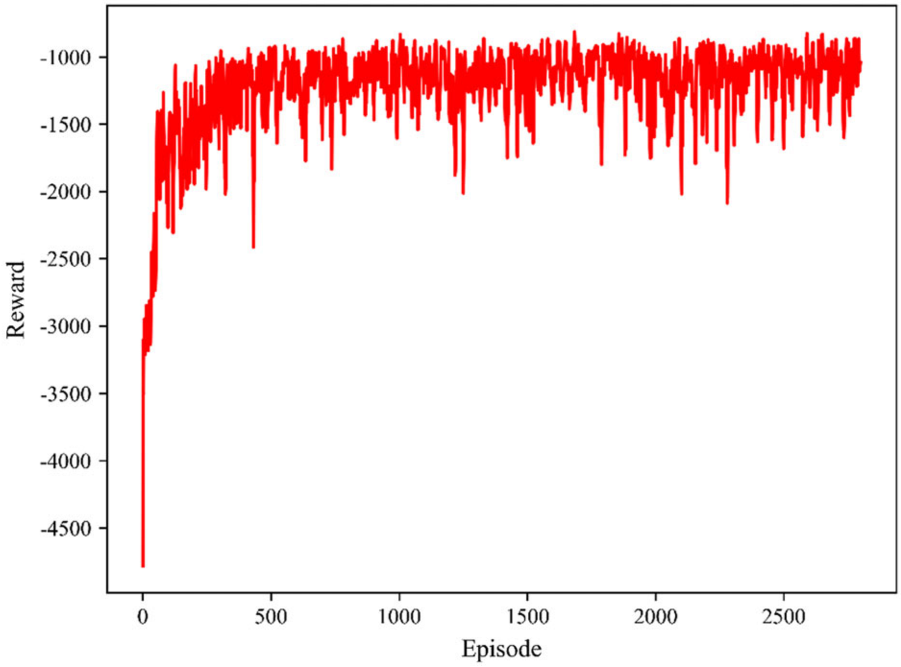


Figure 7. *The reward curve of CQL for AUV obstacle avoidance.*

4.2. Performance validation

To comprehensively evaluate the performance of the trained intelligent agent's control algorithm, we conducted simulations and tests from multiple aspects. Firstly, in static environments, we simulated the motion planning process of the agent navigating from different starting points to various target points and assessed the system's performance by achieving successful navigation to multiple consecutive targets. To make the simulation more realistic, we introduced the factor of ocean currents and tested the agent's path planning and control capabilities in complex environments. Specifically, we focused on the agent's obstacle avoidance planning ability in the presence of dynamic obstacles to thoroughly evaluate its adaptability in complex scenarios. Finally, we validated the anti-interference capability of the algorithm

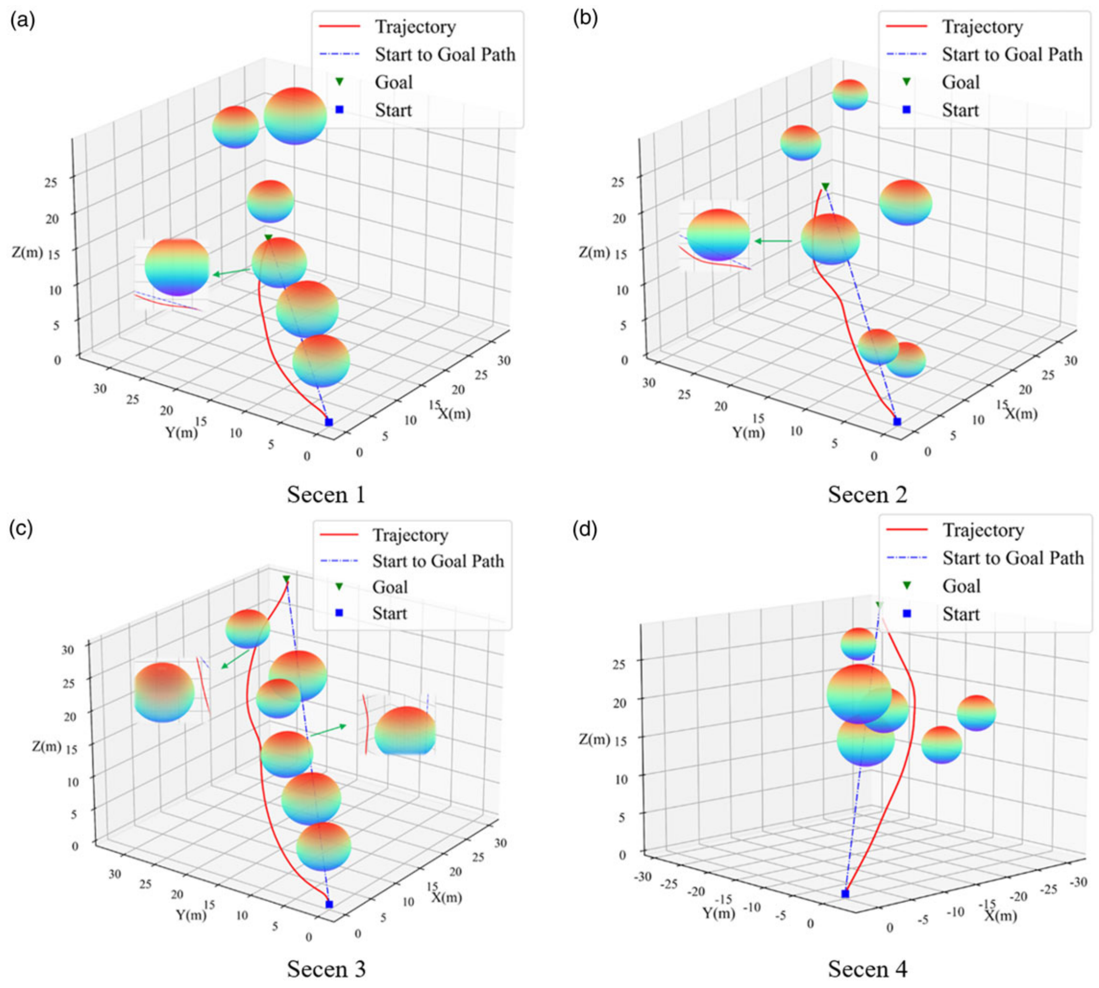


Figure 8. *Results of AUV obstacle avoidance in environments with 6 obstacles.*

by introducing Gaussian noise to simulate external disturbance forces and ocean currents and conducted obstacle avoidance tests separately in these disturbed environments.

4.2.1. Simulation of static obstacle avoidance

To rigorously evaluate the obstacle avoidance performance of the trained policy, a series of tests were conducted in environments populated with static obstacles. The operating space for the AUV was defined as a $30\text{m} \times 30\text{m} \times 30\text{m}$ cube, with the AUV's initial position fixed at the origin point $(0\text{m}, 0\text{m}, 0\text{m})$ within the geodetic coordinate system. The target positions were systematically assigned to $(22\text{m}, 25\text{m}, 10\text{m})$,$(26\text{m}, 27\text{m}, 15\text{m})$, $(30\text{m}, 30\text{m}, 30\text{m})$, and $(-30\text{m}, -27\text{m}, 29\text{m})$. Obstacles were randomly placed within the operating space, with a fixed quantity of six obstacles.

As illustrated in Figure 8, the trajectories of the AUV in these different environments clearly demonstrate the efficacy of the proposed algorithm in avoiding obstacles. The algorithm adeptly adjusts the AUV's rudder angle and thrust based on real-time distance and orientation data relative to the obstacles. This enables the AUV to navigate safely and efficiently, successfully reaching the predetermined three-dimensional waypoints.

To further substantiate the robustness of the algorithm, the complexity of the obstacle environment was heightened by increasing the number of obstacles to nine, as depicted in Figure 9. The figure

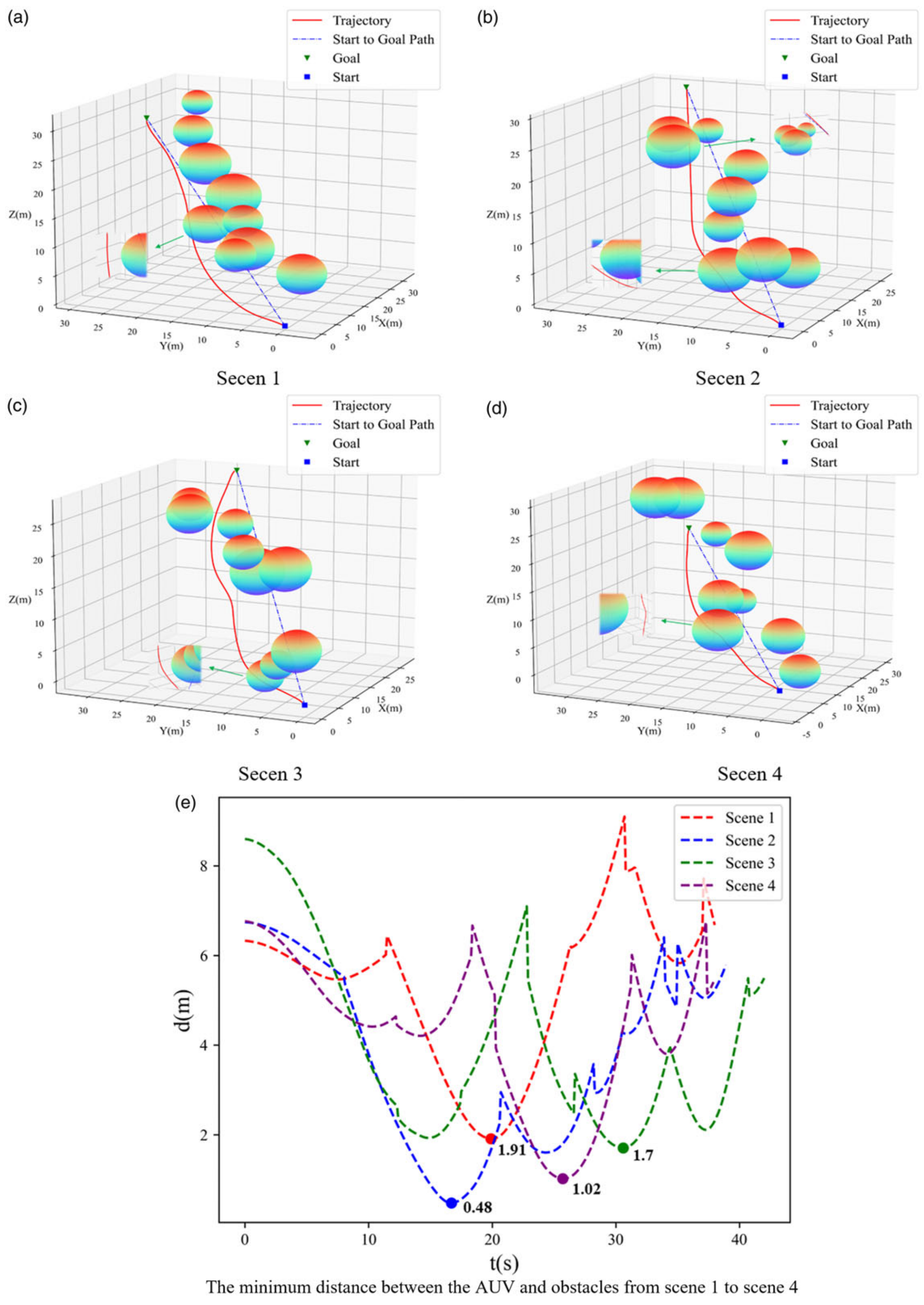


Figure 9. *Results of AUV obstacle avoidance in environments with 9 obstacles.*

shows the AUV's trajectories along with the variation curves representing the minimum distances between the AUV and the obstacles during the last four obstacle avoidance missions. In these scenarios, the AUV commenced from the origin point(0m, 0m, 0m), with target positions at (25m, 20m, 23m), (24m, 25m, 25m), (30m, 30m, 24m), and (27m, 28m, 24m). The recorded minimum distances between the AUV and the nearest obstacles were 1.91m, 0.48m, 1.7m, and 1.02m, respectively, indicating that the AUV maintained a safe distance from obstacles throughout its navigation.

The trajectories presented in these figures underscore the algorithm's superior obstacle avoidance capabilities, even in more challenging environments with increased obstacle density. The AUV successfully avoids obstacles within its sensing range and consistently reaches its target points. These results validate the effectiveness and reliability of the proposed method for autonomous navigation and obstacle avoidance in three-dimensional underwater environments.

4.2.2. Simulation of dynamic obstacle avoidance

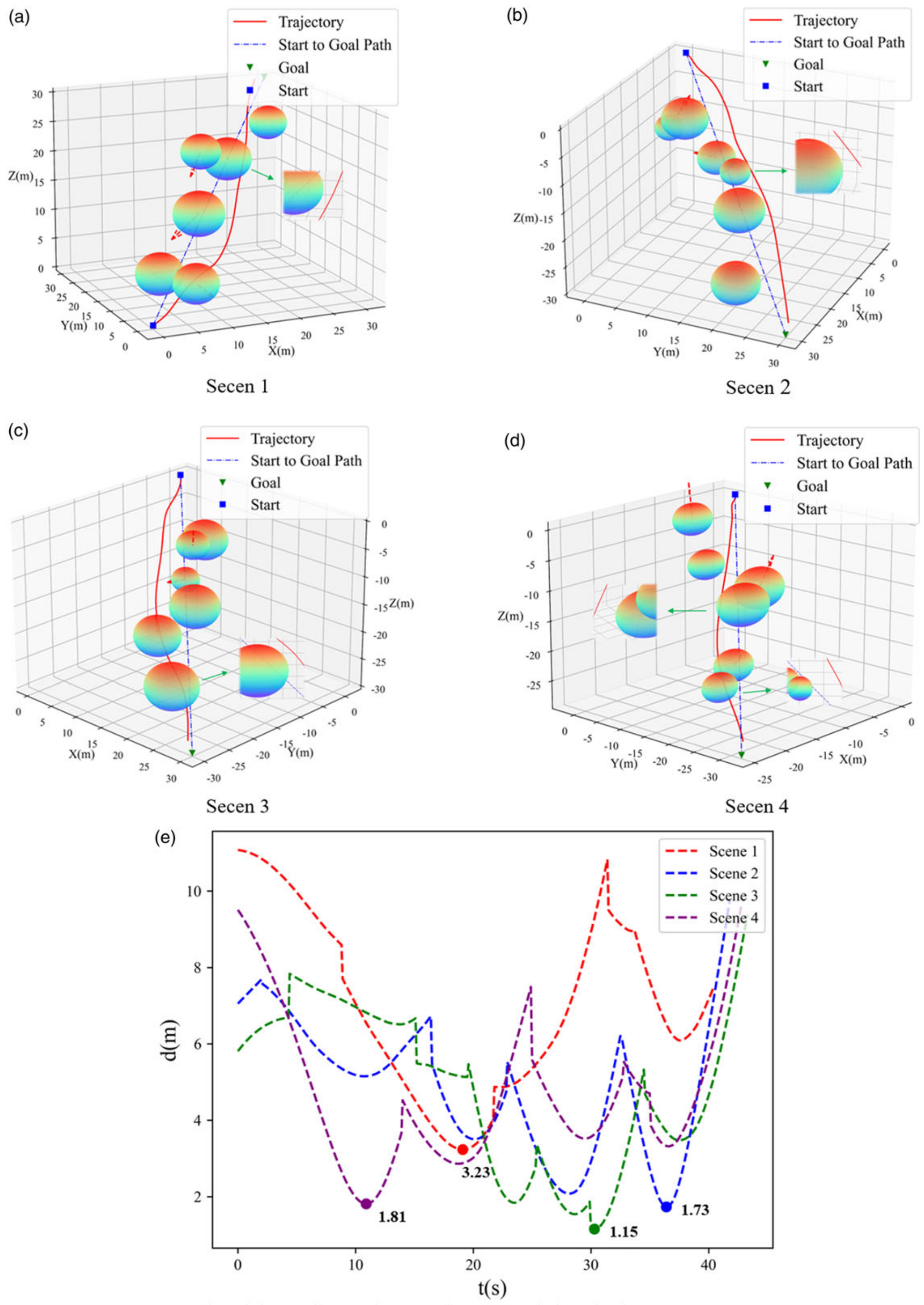
To evaluate the dynamic obstacle avoidance performance of the proposed algorithm, we randomly configured a simulation environment with three dynamic obstacles and three static obstacles. The dynamic obstacles performed uniform rectilinear motion back and forth within the AUV's workspace, with their velocities and directions randomly assigned. The speeds of these dynamic obstacles were capped at 1.2 m/s, and their initial moving directions were indicated by red dashed arrows. The obstacle radius was randomly set between 2 m and 4 m.

In the four obstacle avoidance tasks depicted in Figure 10, the AUV's initial positions were all (0 m, 0 m, 0 m), and the target positions were (30 m, 30 m, 30 m), (30 m, 30 m, −30m), (30 m, −30 m, −30 m), and (−25 m, 32 m, −29 m), respectively. Through calculations, the minimum distances between the AUV and obstacles were 3.23 m, 1.73 m, 1.15 m, and 1.81 m, respectively. The AUV's movement trajectories in Figure 10 demonstrate that the AUV successfully completed autonomous navigation towards random targets and achieved real-time dynamic obstacle avoidance, validating the efficacy of the CQL-based obstacle avoidance strategy for AUV.

To provide a comprehensive comparison, we evaluated the performance of three obstacle avoidance algorithms—PID-APF, SAC, and CQL—within the same environment. Figure 11 presents the results of these algorithms. The minimum distances between the AUV and obstacles were 3.23 m, 1.45 m, and 2.50 m for the PID-APF, SAC, and CQL algorithms, respectively. The time steps required were 405, 458, and 449, and the lengths of the movement paths were 66.34m, 55.28 m, and 54.59 m, respectively. The execution time of the system is 1.15 s, 0.56 s, and 0.66 s respectively, and the average execution time of each step is 0.0028 s, 0.0012 s, and 0.0015 s respectively, which indicates that the three algorithms all meet the real-time requirements of the system without online training. As observed in Figure 11(a), the PID-APF algorithm exhibited significant trajectory deviations due to the necessity for real-time path updates and insufficient robustness in dynamic environments, resulting in the longest path. In contrast, the trajectories based on SAC and CQL algorithms, depicted in Figure 11(b) and Figure 11(c), were smoother, reflecting the flexibility of reinforcement learning algorithms. Notably, the CQL algorithm required fewer steps, achieved a greater minimum distance from obstacles, and resulted in a shorter movement path, thereby demonstrating superior obstacle avoidance performance.

To further evaluate the performance of the three algorithms, a fixed random seed was used to generate obstacles and targets, and the dynamic obstacle avoidance effects of these three algorithms were tested in the most recent 100 obstacle avoidance tasks. Figure 12 illustrates the minimum distances between the AUV and obstacles for each of these 100 tasks. It can be seen from Figure 12 that in the most recent 100 tasks, the collision frequencies of the PID-APF, SAC, and CQL strategies were 20, 8, and 5 times, respectively, corresponding to success rates of 80%, 92%, and 95%. These full results unequivocally demonstrate the safety and effectiveness of the CQL-based obstacle avoidance method.

Based on the comprehensive simulation results, it can be concluded that the end-to-end obstacle avoidance control algorithm for AUVs designed based on the CQL algorithm can effectively guide the AUV to perform autonomous navigation and avoid possible dynamic obstacles in real time. Compared



The minimum distance between the AUV and obstacles from scene 1 to scene 4

Figure 10. *Results of AUV obstacle avoidance in dynamic obstacle environment.*

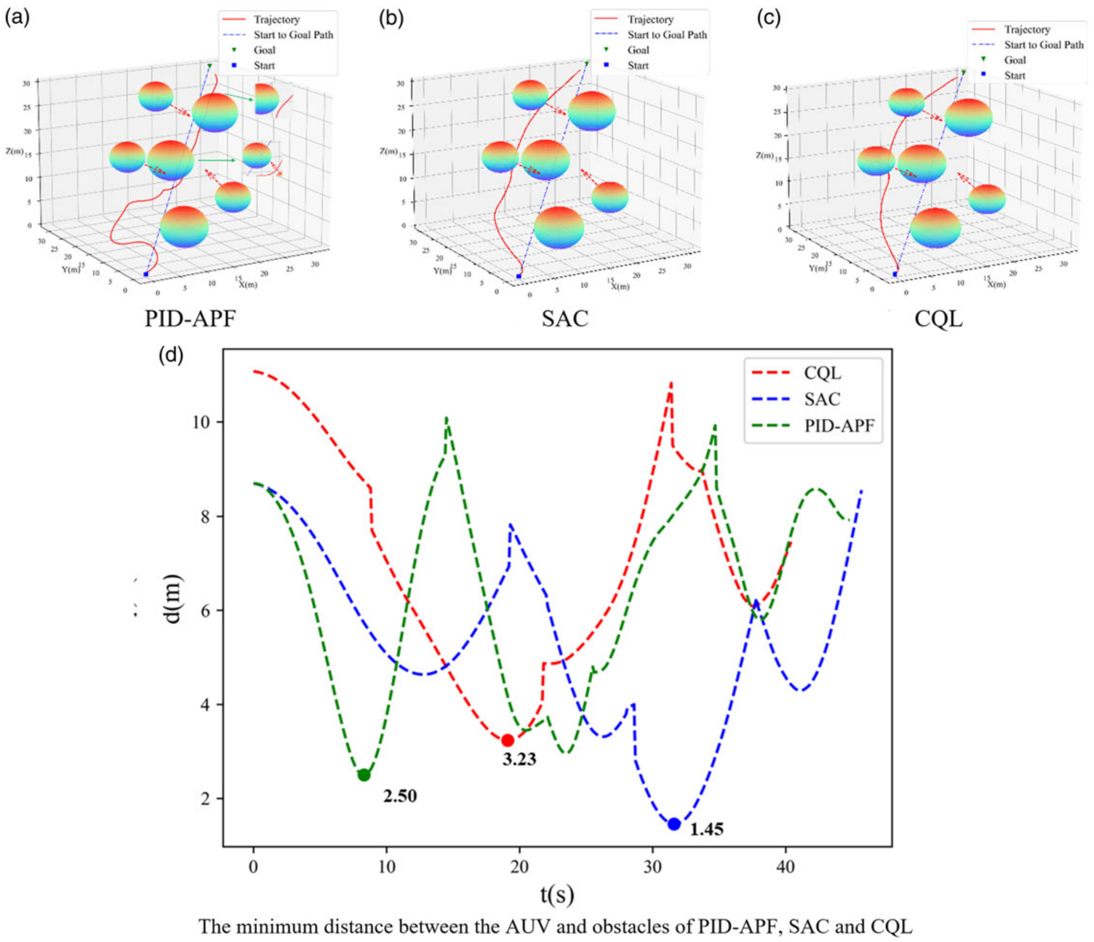


Figure 11. *The obstacle avoidance results of PID-APF, SAC, and CQL in the same environment.*

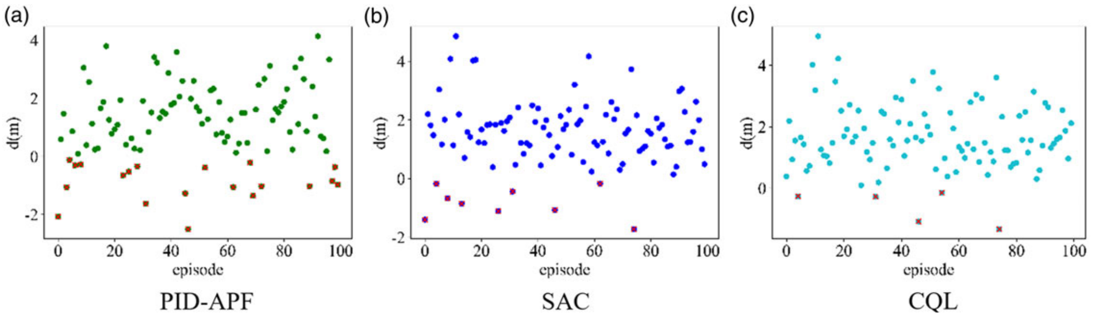


Figure 12. *The minimum distance distribution between AUV and obstacle in the last 100 obstacle avoidance tasks.*

to PID-APF and SAC obstacle avoidance algorithms, the CQL algorithm exhibits enhanced safety and efficiency, confirming the feasibility and effectiveness of this intelligent obstacle avoidance method for AUVs based on offline reinforcement learning.

To verify the motion planning capability of the controller, we utilized this obstacle avoidance strategy to accomplish navigation of multiple consecutive targets in Figure 13 In these two missions,

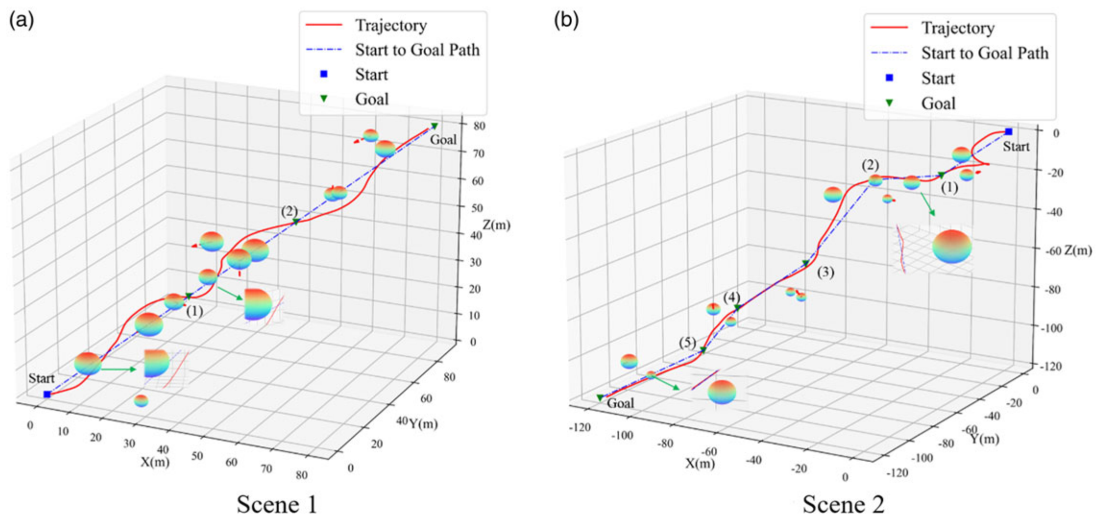


Figure 13. *Results of AUV obstacle avoidance of multiple consecutive targets.*

the target point sets are [(30,30,30), (50,60,50), (80,90,80)] and [(−20,-20,-20), (−40,−40,−20), (−60,−60,−60), (−80,−80,−80), (−90,−90,−100), (−120,−120,−120)], respectively. As depicted in Figure 13, the AUV successfully reached the predetermined targets, further demonstrating the autonomous navigation and obstacle avoidance capabilities of this method.

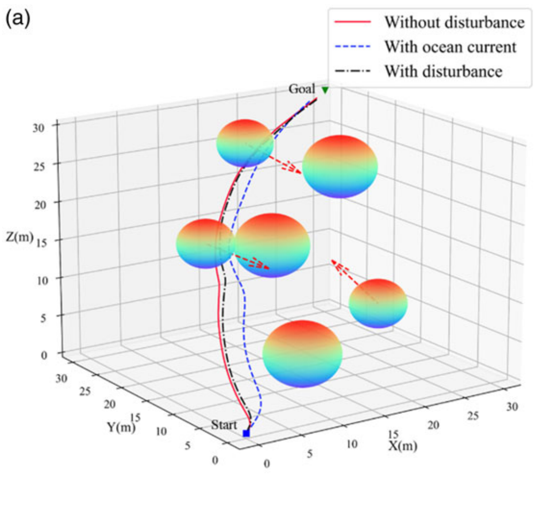
4.2.3. Simulation of obstacle avoidance with disturbances

To validate the anti-interference capability of this method, we employed Gaussian noise to simulate the random changes of disturbing forces and ocean currents. Subsequently, obstacle avoidance tests were conducted separately in the presence of external force disturbances and ocean currents. The results are presented in Figure 14.

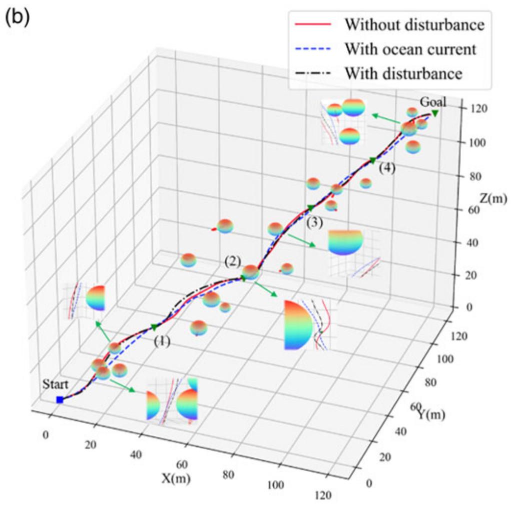
Figure 14 exhibits the obstacle avoidance outcomes of this method during navigation to one and multiple consecutive target points, with target positions set as (30,30,30) and [(30, 30, 30), (60, 55, 50), (80, 80, 80), (100, 100, 100), (120, 120, 120)], respectively. The red, blue, and black lines represent the AUV's motion trajectories in an ideal environment, under external force disturbances, and under the influence of ocean currents.

Through detailed calculations and analysis, we have drawn the following conclusions. During single-target navigation Figure 14(a), the minimum distance between the AUV and obstacles in an ideal environment was 2.50 m, while under external force disturbances and the influence of ocean currents, this distance decreased to 1.93 m and 0.98 m, respectively. Despite the reduction in minimum distance due to interference and currents, the AUV was still able to safely reach the target point. Meanwhile, we noticed that under these three conditions, the AUV's motion path lengths were 54.59 m, 54.40 m, and 53.88 m, indicating that the interference and currents had limited effects on the overall motion path length of the AUV.

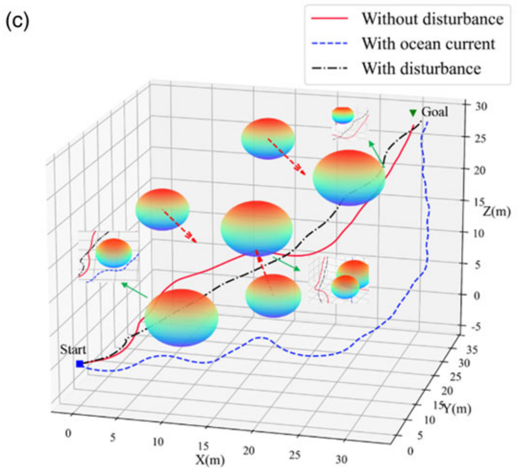
During continuous multi-target navigation (Figure 14(b)), we observed similar phenomena. In an ideal environment, the minimum distance between the AUV and obstacles was 1.40 m, while under external force disturbances and the influence of ocean currents, this distance decreased to 1.05 m and 0.79 m, respectively. Despite the reduction in minimum distance due to interference and currents, the AUV was still able to reach each target point sequentially according to the preset path. Simultaneously, under these three conditions, the AUV's motion path lengths were 216.35 m, 212.26 m, and 215.99 m, demonstrating that in the process of continuous navigation, the interference and currents had similarly limited effects on the overall motion path length of the AUV.



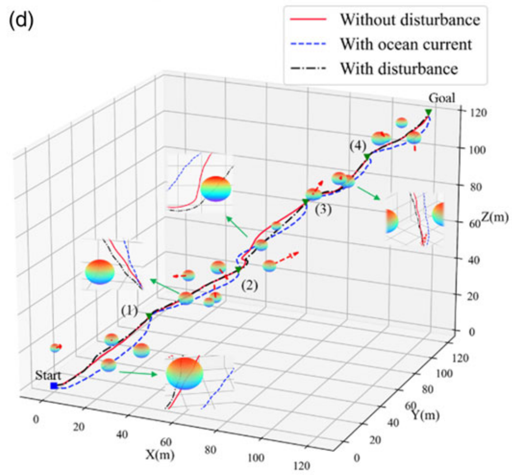
Result of one target using the CQL-based method



Result of multiple consecutive targets using the CQL-based method



Result of one target using adaptive PID



Result of multiple consecutive targets using adaptive PID

Figure 14. *Results of AUV obstacle avoidance in a disturbed environment.*

To further elucidate the outstanding anti-interference capability of the CQL-based method, we showcase the performance of the adaptive PID-APF method in executing the same navigation tasks in Figure 14 (c) and 14(d). In this approach, the APF updates the path every five steps based on the current obstacle environment, and subsequently, the adaptive PID method takes over for tracking and controlling the safe path. As depicted in Figure 14(c), during a single-target-point navigation task, the minimum distance between the AUV and obstacles under ideal conditions is 2.47 m. However, under the influence of external force disturbances and ocean currents, this minimum distance alters to 4.60 m and −0.78 m, respectively. Concurrently, the AUV's movement path lengths in these three scenarios are 65.01 m, 76.08 m, and 64.97 m, respectively. Figure 14(d) illustrates the AUV's movement path when navigating towards multiple consecutive target points. Here, the minimum distances between the AUV and obstacles are −1.49 m, 0.47 m, and −2.65 m, with the corresponding movement path lengths being 221.45 m, 232.73 m, and 221.55 m, respectively. This demonstrates that the adaptive PID-APF algorithm has limited anti-interference ability, particularly under the influence of ocean currents, where the control efficacy diminishes significantly. Consequently, substantial deviations are observed in the overall movement trajectory, with longer travel paths and increased collision risks.

In summary, we can conclude that the CQL-based method possesses strong anti-interference capabilities and robustness. Even in environments with external force disturbances and the influence of ocean currents, the AUV is still able to reach the preset target points safely and accurately, and its overall motion path does not exhibit significant deviations. This result not only verifies the effectiveness of this method but also provides strong support for its application in actual marine environments.

5. Conclusions

This paper presents a novel end-to-end obstacle avoidance control algorithm for AUVs based on CQL. The proposed algorithm has been rigorously validated through extensive numerical simulations, demonstrating its ability to effectively learn a superior obstacle avoidance policy from historical data without necessitating real-time interaction with the environment. This approach mitigates the risks and costs associated with trial-and-error learning in actual system operations. During the simulation process, the CQL-based algorithm consistently achieved real-time dynamic obstacle avoidance, showcasing exceptional performance. To further substantiate its effectiveness, we compared the CQL-based obstacle avoidance strategy with those based on PID-APF and SAC within the same simulation environment. Both the CQL and SAC algorithms employed identical network structures, state spaces, action spaces, and reward functions. The comparative results revealed that the CQL-based strategy had a higher success rate in obstacle avoidance tasks. Specifically, in a series of random obstacle avoidance scenarios, the success rates for the PID-APF, SAC, and CQL algorithms were 80%, 92%, and 95%, respectively. Additionally, the algorithm's robustness was tested under conditions involving irregular external force disturbances and ocean currents. The results demonstrated that the CQL-based algorithm could still effectively guide the AUV to its preset target points while avoiding obstacles, thereby proving its robust anti-interference capabilities. The simulation results unequivocally demonstrate the effectiveness, safety, and robustness of the CQL-based method in intelligent dynamic obstacle avoidance for AUVs. The algorithm's offline training characteristics, coupled with its safe and efficient obstacle avoidance strategy, significantly enhance the safety and reliability of AUV systems. This advancement provides robust technical.

The current study on AUV obstacle avoidance utilizing CQL algorithm, while demonstrating promising results, is limited by data dependency for generalization, computational complexity for real-time applications, and adaptability to highly dynamic environments. Future research should explore data augmentation, computational optimization, hybrid online-offline learning, multi-agent collaboration, integration with traditional navigation techniques, enhanced interpretability, as well as comprehensive task planning and energy management to address these limitations and advance the field.

Author contributions. The contributions of the authors to this paper are as follows: Tao Liu conceived and designed the study; JunHao Huang performed the simulations, data analysis, and wrote the initial draft of the manuscript; Jintao Zhao contributed to the study design and manuscript revision. All authors reviewed and approved the final version of the manuscript.

Financial support. The work is supported by Sun Yat-sen University 2024 Basic Scientific Research Fund Youth Faculty Development Program (24qnpy032), and Southern Marine Science and Engineering Guangdong Laboratory (Zhuhai).

Competing interests. The authors declare no conflicts of interest exist.

Ethical approval. None.

References

[1] J. Zhou, Y. Si and Y. Chen, "A review of subsea AUV technology," *J Mar Sci Eng* **11**(6), 1119 (2023).

[2] F. S. Tabatabaee-Nasab, S. A. A. Moosavian and A. K. Khalaji, "Adaptive fault-tolerant control for an autonomous underwater vehicle," *Robotica* **40**(11), 4076–4089 (2022).

[3] E. Taheri, M. H. Ferdowsi and M. Danesh, “Design boundary layer thickness and switching gain in SMC algorithm for AUV motion control,” *Robotica* **37**(10), 1785–1803 (2019).
[4] H. Wang and B. Su, “Event-triggered formation control of AUVs with fixed-time RBF disturbance observer,” *Appl Ocean Res* **112**, 102638 (2021).
[5] C. Thomas, E. Simetti and G. Casalino, “A unifying task priority approach for autonomous underwater vehicles integrating homing and docking maneuvers,” *J Mar Sci Eng* **9**(2), 162 (2021).
[6] P. Gong, Z. Yan, W. Zhang and J. Tang, “Trajectory tracking control for autonomous underwater vehicles based on dual closed-loop of MPC with uncertain dynamics,” *Ocean Eng* **265**, 112697 (2022).
[7] D. Li and L. Du, “AUV trajectory tracking models and control strategies: A review,” *J Mar Sci Eng* **9**(9), 1020 (2021).
[8] M. M. Hammad, A. K. Elshenawy and M. I. El Singaby, “Position Control and Stabilization of Fully Actuated AUV Using PID Controller,” **In:** *Proceedings of SAI Intelligent Systems Conference (IntelliSys) 2016*, (Springer International Publishing, 2018) pp. 517–536.
[9] M. H. Khodayari and S. Balochian, “Modeling and control of autonomous underwater vehicle (AUV) in heading and depth attitude via self-adaptive fuzzy PID controller,” *J Mar Sci Technol* **20**(3), 559–578 (2015).
[10] H. Peng, B. Huang, M. Jin, C. Zhu, J. Zhuang, “Distributed finite-time bearing-based formation control for underactuated surface vessels with Levant differentiator,” *ISA Transactions*, **147**, 239–251 (2024).
[11] B. Zhou, B. Huang, Y. Su and C. Zhu, “Interleaved periodic event-triggered communications-based distributed formation control for cooperative unmanned surface vessels,” *IEEE Trans Neur Net Learn Syst* 1–13 (2024).
[12] B. Huang, S. Song, C. Zhu, J. Li and B. Zhou, “Finite-time distributed formation control for multiple unmanned surface vehicles with input saturation,” *Ocean Eng* **233**, 109158 (2021).
[13] B. Zhou, B. Huang, Y. Su, W. Wang and E. Zhang, “Two-layer leader-follower optimal affine formation maneuver control for networked unmanned surface vessels with input saturations,” *Int J Robust Nonlin Cont* **34**(5), 3631–3655 (2024).
[14] C. Wang, W. Cai, J. Lu, X. Ding and J. Yang, “Design, modeling, control, and experiments for multiple AUVs formation,” *IEEE Trans Autom Sci Eng* **19**(4), 2776–2787 (2021).
[15] N. Khlif, K. Nahla and B. Safya, “Reinforcement learning with modified exploration strategy for mobile robot path planning,” *Robotica* **41**(9), 2688–2702 (2023).
[16] M. Q. Zaman and H.-M. Wu, “An improved fuzzy inference strategy using reinforcement learning for trajectory-tracking of a mobile robot under a varying slip ratio,” *Robotica* **42**(4), 1134–1152 (2024).
[17] C. Sancak, F. Yamac and M. Itik, “Position control of a planar cable-driven parallel robot using reinforcement learning,” *Robotica* **40**(10), 3378–3395 (2022).
[18] S. Liu, C. Ma and R. Juan, “AUV obstacle avoidance framework based on event-triggered reinforcement learning,” *Electronics* **13**(11), 2030 (2024).
[19] Y. Fang, Z. Huang, J. Pu and J. Zhang, “AUV position tracking and trajectory control based on fast-deployed deep reinforcement learning method,” *Ocean Eng* **245**, 110452 (2022).
[20] C. Cheng, Q. Sha, B. He and G. Li, “Path planning and obstacle avoidance for AUV: A review,” *Ocean Eng* **235**, 109355 (2021).
[21] Y. Sun, C. Zhang, G. Zhang, H. Xu and X. Ran, “Three-dimensional path tracking control of autonomous underwater vehicle based on deep reinforcement learning,” *J Mar Sci Eng* **7**(12), 443 (2019).
[22] H. Wu, S. Song, K. You and C. Wu, “Depth control of model-free AUVs via reinforcement learning,” *IEEE Trans Syst Man Cybern: Syst* **49**(12), 2499–2510 (2018).
[23] R. Cui, C. Yang, Y. Li and S. Sharma, “Adaptive neural network control of AUVs with control input nonlinearities using reinforcement learning,” *IEEE Trans Syst Man Cybern Syst* **47**(6), 1019–1029 (2017).
[24] I. Carlucho, M. De Paula, S. Wang, Y. Petillot and G. G. Acosta, “Adaptive low-level control of autonomous underwater vehicles using deep reinforcement learning,” *Robot Auton Syst* **107**, 71–86 (2018).
[25] P. Jiang, S. Song and G. Huang, “Attention-based meta-reinforcement learning for tracking control of AUV with time-varying dynamics,” *IEEE Trans Neur Net Lear Syst* **33**(11), 6388–6401 (2021).
[26] D. Ma, X. Chen, W. Ma, H. Zheng and F. Qu, “Neural network model-based reinforcement learning control for auv 3-d path following,” *IEEE Trans Intell Veh*, **9**(1), 893–904 (2023).
[27] J. Yuan, H. Wang, H. Zhang, C. Lin, D. Yu and C. Li, “AUV obstacle avoidance planning based on deep reinforcement learning,” *J Mar Sci Eng* **9**(11), 1166 (2021).
[28] B. Hadi, A. Khosravi and P. Sarhadi, “Deep reinforcement learning for adaptive path planning and control of an autonomous underwater vehicle,” *Appl Ocean Res* **129**, 103326 (2022).
[29] S. Fujimoto, D. Meger and D. Precup, “Off-Policy Deep Reinforcement Learning Without Exploration,” **In:** *International Conference on Machine Learning*, (PMLR, 2019) pp. 2052–2062.
[30] R. Agarwal, D. Schuurmans and M. Norouzi, “An Optimistic Perspective on Offline Reinforcement Learning,” **In:** *International Conference on Machine Learning*, (PMLR, 2020) pp. 104–114.
[31] S. Levine, A. Kumar, G. Tucker and J. Fu, “Offline reinforcement learning: Tutorial, review, and perspectives on open problems (2020) arxiv preprint arxiv: 2005.
[32] A. Kumar, A. Zhou, G. Tucker and S. Levine, “Conservative q-learning for offline reinforcement learning,” *Adv Neur Inf Process Syst* **33**, 1179–1191 (2020).
[33] T. Yu, G. Thomas, L. Yu, S. Ermon, J. Y. Zou, S. Levine and T. Ma, “Mopo: Model-based offline policy optimization,” *Adv Neur Inf Process Syst* **33**, 14129–14142 (2020).

[34] T. I. Fossen, "Marine control systems-guidance. navigation, and control of ships, rigs and underwater vehicles," (2002). Marine Cybernetics, Trondheim, Norway, Org. Number NO 985 195 005 MVA, www. marinecybernetics. com, ISBN: 82 92356 00 2.

[35] T. I. Fossen. *Handbook of Marine Craft Hydrodynamics and Motion Control,* John Wiley & Sons Ltd, Chichester, UK (2011).

[36] T. T. J. Prestero, Verification of a six-degree of freedom simulation model for the REMUS autonomous underwater vehicle (Doctoral dissertation, Massachusetts institute of technology), (2001).

Cite this article: T. Liu, J. Huang and J. Zhao, "Research on obstacle avoidance of underactuated autonomous underwater vehicle based on offline reinforcement learning", Robotica. https://doi.org/10.1017/S0263574724001802