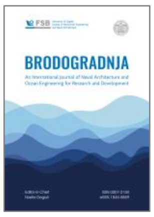
journal homepage: www.brodogradnja.fsb.hr

Brodogradnja

An International Journal of Naval Architecture and Ocean Engineering for Research and Development



Adaptive trajectory controller design for unmanned surface vehicles based on SAC-PID

Wei Guan[*], *Zhaoyong Xi, Zhewen Cui, Xianku Zhang*

Navigation College, Dalian Maritime University, Dalian, China

ARTICLE INFO

Keywords:
deep reinforcement learning
unmanned surface vehicle
soft actor critic
PID tuning

ABSTRACT

An adaptive proportional integral derivative (PID) controller based on the soft actor-critic (SAC) algorithm for trajectory control of unmanned surface vehicles (USV) is proposed in this paper. The gains of the PID controller need to be manually adjusted based on experience in the original formulation. Furthermore, once tuned, these gains remain fixed and making further modifications becomes time-consuming and labor-intensive. To address these limitations, the SAC algorithm is introduced, enabling online tuning of PID gains through agent-environment interaction. Additionally, the strategy of combining SAC algorithm with PID controller mitigates concerns regarding interpretability and security often associated with DRL. In this study, stability analysis of the adaptive trajectory controller based on the SAC-PID algorithm is conducted. This paper horizontally compares the proposed method with traditional PID tuning methods, genetic algorithms (GA), and deep deterministic policy gradient (DDPG) algorithm to highlight the superiority of the SAC-PID approach. Finally, experiments in different scenarios are performed to compare generalization capabilities between DDPG and SAC algorithms. Results demonstrate that the proposed SAC-PID algorithm exhibits excellent stability properties, fast convergence speed, and strong generalization ability.

1. Introduction

Trajectory tracking and control are key technologies for autonomous and safe navigation of unmanned surface vehicles (USVs). The voyage needs to resist the interference of wind, waves, and currents, and follow the path proposed by the trajectory planning system to successfully complete the corresponding unmanned tasks.

In recent years, ship trajectory control has emerged as a prominent research area among scholars and scientists [1-3]. The proportional integral derivative (PID) control algorithm and subsequent developed fuzzy adaptive PID algorithm have been validated on real ships and can effectively achieve USV trajectory control [4-5]. To address the issue of unmeasurable states in a class of high-order nonlinear systems, Chen et al. [6] proposed an observer based adaptive consensus tracking control strategy using backstepping technology. In order to address the issue of trajectory tracking for USV that encounter system uncertainty and time-varying

[*] Corresponding author.

E-mail address: gwwtxdy@dlmu.edu.cn

http://dx.doi.org/10.21278/brod76206

Received 07 January 2025; Received in revised form 09 March 2025; Accepted 19 March 2025
Available online 28 March 2025
ISSN 0007-215X; eISSN 1845-5859

external disturbances, a novel control scheme utilizing finite time sliding mode control and a nonlinear disturbance observer was proposed [7]. This method combines backstepping design technology with second-order sliding mode control, ensuring the finite time stability of the system. The exceptional characteristics of sliding mode control, including its prompt response, robustness, and straightforward physical implementation [8], have led to its increasing adoption by researchers in the field of trajectory control research. With the emergence of artificial intelligence, Zhao et al. [9], Paulig and Okhrin [10] utilized deep Q-networks and deep Q-learning algorithms for the navigation and control of USVs, demonstrating their exceptional adaptability and ability to handle unprecedented scenarios with precise control accuracy. These studies serve as excellent examples of applying deep reinforcement learning (DRL) in trajectory tracking control. However, implementing them practically on USVs remains challenging.

The widely adopted PID algorithm in industrial systems is known for its simplicity and effectiveness in addressing automatic control issues [11, 12]. Nevertheless, it still presents notable constraints, including the requirement for manual tuning of PID gains and the inability to dynamically modify the adjusted gains [13]. The enormous growth in computing power provides the possibility of using more data, rather than relying solely on limited experimental data. Therefore, through the utilization of optimization techniques on the complete dataset, it becomes feasible to decrease the duration of experiments while attaining a more accurate model, ultimately resulting in improved controller performance [14]. Sahib and Ahmed [15] designed a novel time domain performance criterion based on multi-objective Pareto front solutions, and demonstrated the superiority of the proposed objective function by employing the application of particle swarm optimization (PSO) algorithm in automatic voltage regulation systems. Zhang et al. [16] combined unmodeled dynamics data-driven compensation with multi-step ahead of optimal control strategy to construct a nonlinear PID controller for pulp neutralization process, reducing the fluctuation range of pulp pH value. In addition, the exponential weighted error squared function is also used for PID controller gain tuning, thereby enhancing the adjustability of maximum sensitivity during the process [17]. Considering the limitations associated with PID controllers, a novel indirect design approach is proposed [18], which not only enables online fine-tuning of the PID controller gains but also provides an additional feature of online robustness of the controller.

Many above-mentioned expert systems developed for the tuning of adaptive PID controllers entail a significant high computational burden, requiring not only past expert knowledge but also efficient algorithms and software execution. To address this issue, Carlucho et al. [19] developed an expert system based on the Double Q-learning algorithm to enable adaptive control of multiple low-level PID controllers in mobile robots. Udekwe et al. [20] employed soft actor-critic (SAC), deep deterministic policy gradient (DDPG), and twin delayed deep deterministic (TD3) policy gradient algorithms to optimize the parameters of PID controllers, and compared their trajectory tracking performances on ball plate systems, contributing to the establishment of a solid theoretical basis for future research in this domain. A non-integer PID controller based on the DDPG algorithm is proposed [21] for supplementary learning control of wheeled mobile robots. This method improves control accuracy, anti-interference performance, and uncertainty suppression performance. However, the issues of wheel slip and lateral sliding are not considered. A hybrid control strategy utilizing the actor-critic framework was proposed by Carlucho et al. [22]. This approach demonstrates effective control over MIMO PID systems with diverse dynamics and multiple set point requirements, including applications in underwater vehicles and terrestrial robots. Yu et al. [23] proposed a hierarchical structure of SAC algorithm and incremental PID control method, using SAC algorithm to compensate for path error of patrol robots and determine the optimal PID control parameters.

With the advancement of DRL algorithm, researchers have made enhancements to the DRL-PID algorithm from the perspectives of information input method [24], action space exploration encouragement [25], activation function and hyperparameter selection [26], and neural network replacement [27]. At the same time, this control scheme has been successfully implemented in various industrial systems [28, 29]. In terms of intelligent ship autonomous control, Chu et al. [30] combined imitation learning with TD3 algorithm to design motion controllers for unmanned underwater vehicles, leveraging the data of PID algorithm as expert data for pre-training to accelerate convergence speed. However, this control method heavily relies on the use of thrusters. Lee et al. [31] developed an adaptive PID controller for the ship dynamic positioning system based on the DDPG algorithm. The controller considers the slow response characteristics of big vessels and limits the adjustable gains within a predetermined range through pre-tuning. Unfortunately, the generalization effect in different environments is not excellent enough. In addition, an intelligent adaptive PID controller based on proximal policy optimization (PPO) was also proposed [32] to achieve course-keeping of USV. The prospects of applying deep reinforcement learning in the process industry mainly focus on stability, interpretability, sample efficiency, and practicality [33-35]. The combination of DRL and PID addresses the issue of limited interpretability in DRL, and the challenges associated with fine-tuning gains in conventional PID controllers, showcasing considerable promise for engineering applications. Moreover, Lawrence et al. [36] developed corresponding industrial DRL-PID control software to avoid expensive hardware deployment, offering promising prospects for the industrial implementation of DRL. The development of control technology and multi-agent deep reinforcement learning has brought bright prospects to this research field [37-39].

According to the above research, this paper proposes an adaptive trajectory controller for USV based on the SAC-PID algorithm. The superiority of the proposed method has been verified through horizontal comparison with traditional PID algorithm, genetic algorithm (GA), and DDPG algorithm. The experimental results in different environments show that the proposed method has good generalization ability.

The subsequent sections of this manuscript are structured as follows. Section 2 introduces preliminary knowledge. Section 3 presents the algorithm framework and details, including stability analysis. In Section 4, the simulation experiment results are analyzed to verify the performance of the proposed SAC-PID algorithm. Section 5 gives a summary and outlook.

2. Methodology

2.1 PID Algorithm

The PID controller, a widely used feedback loop component in industrial control applications, comprises proportional, integral, and differential units. It possesses the advantages of straightforward underlying principle, easy implementation, broad applicability, and independent control parameters. The mathematical expression for PID control is as follows:

$$u(t) = k_p e(t) + k_i \int e(t)\mathrm{d}t + k_d \frac{\mathrm{d}e(t)}{\mathrm{d}t} \tag{1}$$

where $u(t)$ means the output of the controller, $e(t)$ means the error at time t. k_p, k_i, k_d are the proportional, integral and derivative tuning gains of the PID controller respectively.

Given that computer control is a sampling control, researchers put forward a digital PID algorithm known as the positional PID algorithm and the incremental PID algorithm. In this paper, we adopt the incremental PID algorithm, which can be expressed as:

$$\Delta u(t) = k_p[e(t) - e(t-1)] + k_i e(t) + k_d[e(t) - 2e(t-1) + e(t-2)] \tag{2}$$

where $\Delta u(t)$ is the control increment, $e(t)$, $e(t-1)$, $e(t-2)$ are the errors of the time t, $t-1$ and $t-2$ respectively.

2.2 Ship Motion Mathematical Model

The mathematical model of ship motion serves as the foundation for investigating ship motion control and is extensively employed in the design of ship motion controllers as well as the development of ship motion simulators. The variables of the mathematical model of ship motion are shown in Figure 1.

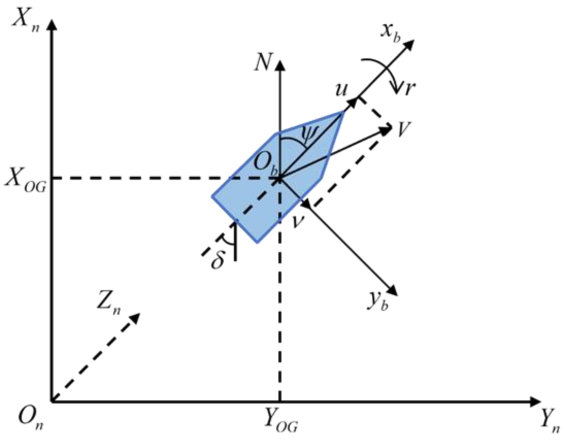


Fig. 1 Mathematical model of ship motion

$O_n - X_nY_nZ_n$ is the inertial coordinate system, where O_nX_n axis points due north, O_nY_n axis points due east, and O_nZ_n axis is perpendicular to the horizontal plane and points towards the bottom of the ship. $O_b - x_by_bz_b$ is the body-fixed coordinate system, where O_b is usually chosen at the center of gravity of the ship, O_bx_b axis points towards the bow along the centerline of the ship, O_by_b axis points towards the starboard side, and O_bz_b axis points towards the center of the earth.

For most ship motion control problems, the heave motion, pitch motion, and roll motion can be ignored. Only the motion variables of the other three degrees of freedom need to be discussed, namely the surge velocity u, sway velocity v, and yaw rate r as depicted in Figure 1. ψ represents the heading angle, taking 0° due north and 0°~360° clockwise. δ is the rudder angle, taking the starboard side rudder turning as the positive position. The mathematical model for ship plane motion is expressed as follows:

$$\begin{bmatrix} (m - X_{\dot{u}}) & 0 & 0 \\ 0 & (m - Y_{\dot{v}}) & (mx_{OG} - Y_{\dot{r}}) \\ 0 & (mx_{OG} - N_{\dot{v}}) & (I_{zz} - N_{\dot{r}}) \end{bmatrix} \begin{bmatrix} \Delta\dot{u} \\ \dot{v} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} X_u & 0 & 0 \\ 0 & Y_v & (Y_r - mu_0) \\ 0 & N_v & (N_r - mx_{OG}u_0) \end{bmatrix} \begin{bmatrix} \Delta u \\ v \\ r \end{bmatrix} + \begin{bmatrix} 0 \\ Y_\delta \\ N_\delta \end{bmatrix} \delta \tag{3}$$

where u_0 is the component of the ship motion velocity V in the O_nX_n direction, m is the mass of the ship, x_{OG} means the coordinate value of the center of mass of the ship in the inertial coordinate system. I_{zz} is the moment of inertia of the ship about the vertical axis passing through the centroid point O_b. $X_{\dot{u}}$, $Y_{\dot{v}}$, $Y_{\dot{r}}$, $N_{\dot{v}}$, $N_{\dot{r}}$, X_u, Y_v, Y_r, Y_δ, N_v, N_r and N_δ are all hydrodynamic coefficients [40,41].

The Nomoto model serves as the predominant transfer function representation in marine control systems for characterizing the ship's yaw dynamics in response to rudder inputs. As an essential mathematical tool in naval architecture, this second-order model provides critical theoretical support for control system parameter optimization and facilitates comparative analysis through dynamic response curve simulations.

$$G_{\psi\delta}(s) = \frac{\psi(s)}{\delta(s)} = \frac{K}{s(1 + Ts)} \tag{4}$$

$$T\ddot{\psi} + \dot{\psi} = K\delta. \tag{5}$$

In the simulation experiment of this study, the gain K of the ship model is 0.52448, and the time constant T is 0.169.

2.3 Deep reinforcement learning

RL tasks often use Markov decision process to describe the decision-making process of agents in the environment. The RL agent receives a state s_t at each time, maps the state to the corresponding action through policy $\pi(a_t|s_t)$, and selects actions from the action set A. According to the performance of the action in the environment, the RL agent receives a reward r_t and moves to the next state s_{t+1}. The purpose of RL is to find an optimal policy through continuous trial and error, with the aim of attaining the maximum total cumulative reward $G_t = \sum_{n=0}^{\infty} \gamma^n r_{t+n+1}$, where γ is a discount factor. The following two value functions can be used to evaluate states and actions respectively.

$$v_\pi(s) = E_\pi(G_t|s_t = s), \forall s \in S \tag{6}$$

$$q_\pi(s,a) = E_\pi(G_t|s_t = s, a_t = a), \forall s \in S, a \in A. \tag{7}$$

When encountering a continuous and complex set of states, traditional RL will face challenges. The introduction of deep learning (DL) technology provides a new possibility for reinforcement learning. The DRL method employs deep neural networks to create predictive models for the environment and rewards, enabling approximation of both the value and policy functions. These models are then trained through iterative interactions with the environment. The Actor-Critic is a fundamental framework of DRL, the structures of actor network and critic network are illustrated in Figure 2, where α is the learning rate. The actor network employs a policy approximation function to generate actions and updates its network parameters θ through the utilization of policy gradients. The critic network uses a value approximation function to generate action values for evaluating the performance of the actor and guiding the subsequent actions. Additionally, the critic network utilizes an auxiliary loss function to update the network parameters ω, with the mean squared error loss function being the most employed.

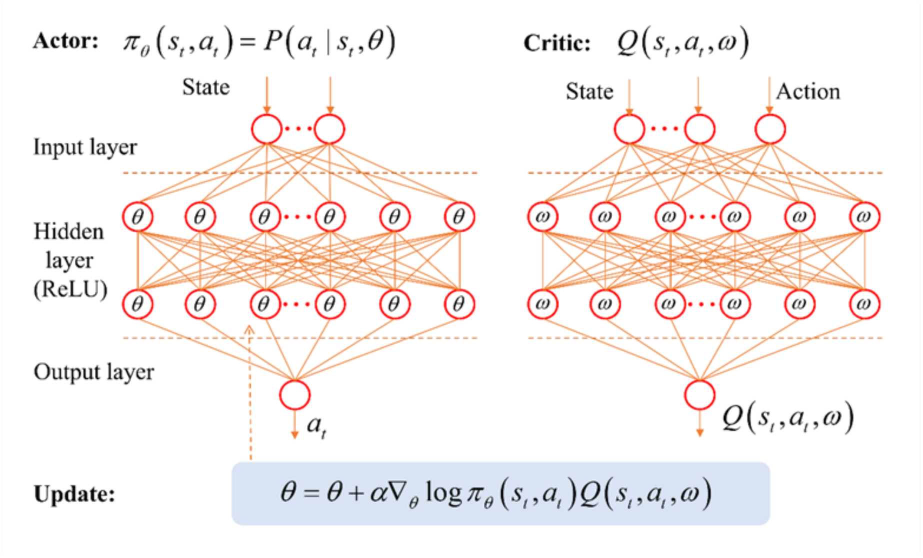

Actor: $\pi_\theta(s_t,a_t) = P(a_t | s_t,\theta)$
Critic: $Q(s_t,a_t,\omega)$
State
State
Action
Input layer
Hidden layer (ReLU)
Output layer
a_t
$Q(s_t,a_t,\omega)$
Update:
$\theta = \theta + \alpha\nabla_\theta \log \pi_\theta(s_t,a_t) Q(s_t,a_t,\omega)$

Fig. 2 Structures of actor network and critic network

3. Proposed Method and Stability Analysis

3.1 Design of SAC-PID Controller

The SAC algorithm [42] is a model-free DRL algorithm based on maximum entropy, employing off-policy approach. The SAC algorithm introduces the concept of maximum entropy on the basis of maximizing future cumulative rewards, thereby enhancing the robustness and the exploration ability of agents. To strike a balance between maximizing future cumulative rewards and maximizing entropy, it is crucial for the policy to exhibit randomness in order to generate a broader distribution of state-action pairs. Consequently, this ensures that the probability of each action output is dispersed rather than concentrated on a single action.

The objective function of maximum entropy reinforcement learning is as follows:

$$J(\pi) = \sum_{t=0}^{T} \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}\left[r(s_t,a_t) + \mu H\big(\pi(a_t|s_t)\big)\right] \tag{8}$$

where T represents the total number of time steps undertaken by the intelligent agent to interact with the environment. ρ_π represents the distribution of (s_t, a_t) under policy π. $H\big(\pi(a_t|s_t)\big)$ is the entropy value. The temperature parameter μ weighs the relative importance of the entropy term on the reward and thus regulates the level of stochasticity in the optimal policy.

In order to evaluate the value of the policy, soft Q value is:

$$Q(s_t, a_t) = \mathrm{E}_{s_{t+1}\sim D}[r(s_t, a_t) + \gamma V(s_{t+1})] \tag{9}$$

where D means the replay buffer. The soft state value function is defined as Equation (10), and it represents the expected reward in a certain state:

$$V(s_t) = \mathrm{E}_{a_t\sim\pi}\big[Q(s_t, a_t) + \mu H\big(\pi(a_t|s_t)\big)\big] = \mathrm{E}_{a_t\sim\pi}[Q(s_t, a_t) - \mu \log \pi\,(a_t|s_t)]. \tag{10}$$

The overall framework of the proposed SAC-PID controller is shown in Figure 3, which is divided into two layers, namely SAC agent and PID controller block. The SAC agent contains 5 networks, namely actor network, Q_1 critic network, Q_2 critic network, V critic network, and target V critic network. They are parameterized by ϕ, θ_1, θ_2, ζ, and $\bar{\zeta}$ respectively. The SAC agent maintains continuous interaction with the control environment, acquiring states and rewards from the environment. It then utilizes the actor network to generate actions, which in this study refers to PID gains. The replay buffer stores tuples (s_t, a_t, r_t, s_{t+1}), which plays a crucial role in eliminating the correlation of experiences, especially in DRL, where there is often a strong correlation between consecutive current and past actions. Then these tuples can be disentangled and stored within the replay buffer. Subsequently, during neural network training, a batch of experiences is randomly sampled from this buffer to optimize the effect of training.

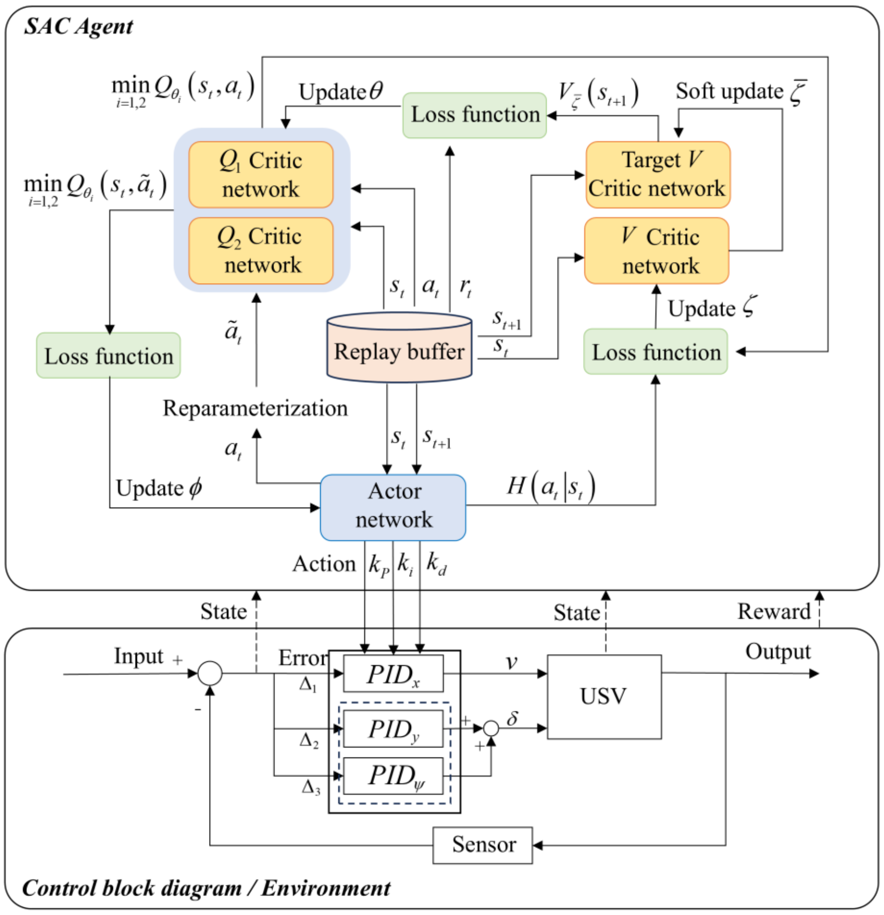


Fig. 3 Diagram of the proposed SAC-PID controller

The SAC algorithm essentially serves as an intelligent decision-making core for multi-parameter cooperative optimization within the PID controller framework. By continuously evaluating the environmental interaction states in real-time, it dynamically assesses the effectiveness of different parameter combinations (actions) and intelligently determines optimal parameter adjustment strategies, thereby realizing a closed-loop self-optimization mechanism for controller parameters.

The critic network V is used to evaluate the value of actions, updating the network parameter ζ by minimizing the squared error between the current value and the target value with entropy. The objective function of critic network V can be defined as follows:

$$J_V(\zeta) = \mathrm{E}_{s_t \sim D}\left[\frac{1}{2}\left(V_\zeta(s_t) - \mathrm{E}_{a_t \sim \pi_\phi}\left[\min_{i=1,2} Q_{\theta_i}(s_t, a_t) - \mu \log \pi_\phi\,(a_t|s_t)\right]\right)^2\right]. \tag{11}$$

where, SAC implemented a dual Q network structure, wherein the selection of the lower Q value between critical network Q_1 and critic network Q_2 was made to prevent overestimation and enhance the speed of convergence.

The target critic network V adopts soft update policy to update parameters. In contrast to hard update, the soft update employs a convex combination of the current network parameters and the target network parameters to facilitate parameter updates in each iteration, and it can be expressed as:

$$\bar{\zeta} = \tau\zeta + (1-\tau)\bar{\zeta} \tag{12}$$

where the soft interval update coefficient τ, typically set between 0 and 1, is usually chosen to be a small value. This ensures that the parameters of the target critic network V undergo smooth changes and that the target values of the network output remain relatively moderate. Such an approach aids in improving algorithm stability. It is worth noting that if the value of τ is too small, it will cause the algorithm to converge slowly. Therefore, researchers should carefully choose an appropriate soft update parameter τ to make SAC algorithm training both stable and fast.

The critic networks Q_1 and Q_2 specifically possess the identical network structure, serving the purpose of avoiding potential overestimation. Their parameters are updated in the same way, and the objective function is as follows:

$$J_Q(\theta) = \mathrm{E}_{(s_t, a_t) \sim D}\left[\frac{1}{2}\left(Q_\theta(s_t, a_t) - \hat{Q}(s_t, a_t)\right)^2\right] \tag{13}$$

with

$$\hat{Q}(s_t, a_t) = r(s_t, a_t) + \gamma \mathrm{E}_{s_{t+1} \sim D}\left[V_{\bar{\zeta}}(s_{t+1})\right]. \tag{14}$$

The actor network can be updated through the minimization of Kullback-Leibler (KL) divergence, while the policy can be re-parameterized using a neural network transformation:

$$a_t = f_\phi(\varepsilon_t; s_t) \tag{15}$$

where ε_t is the input noise vector, sampled from some fixed distribution, such as a spherical Gaussian. The objective function of the actor network is defined as follows:

$$J_\pi(\phi) = \mathrm{E}_{s_t \sim D, \varepsilon_t \sim N}\left[\log \pi_\phi\left(f_\phi(\varepsilon_t; s_t)|s_t\right) - \min_{i=1,2} Q_\theta(s_t, a_t)\right]. \tag{16}$$

The proposed method involves the identification of three points located near the USV along a specified route, followed by consecutive computations of the vertical coordinate discrepancies Δx_i and horizontal coordinate discrepancies Δy_i between these points and the current position point.

Remark 1: *The selection of the three points is shown in Figure 4. The point nearest to the USV satisfies the requirement that the center of gravity of the USV is at the center of the trajectory, and it is also convenient for calculating the successful travelled distance for the reward function. The farthest point enhances the controller's anticipatory control capability, thereby improving tracking precision for high-curvature paths. The middle point balances both the requirements of preview control mechanism and the necessity to maintain the USV's center of gravity close to the trajectory center. Therefore, this point is used to calculate the tracking error in the reward function.*

Additionally, it takes into account the linear velocity v and angular velocity w as well as the deviation $\Delta\psi_i$ between the expected heading and the current heading of USV at each point. These variables collectively form an 11-dimensional state vector s_t.

$$s_t = [\Delta x_1, \Delta y_1, \Delta\psi_1, \Delta x_2, \Delta y_2, \Delta\psi_2, \Delta x_3, \Delta y_3, \Delta\psi_3, v, w]. \tag{17}$$

In this paper, three incremental PID controllers are used to control USV sailing speed and rudder angle, as shown in Figure 5. To ensure the modularization and scalability of the proposed control system, this study adopts a hierarchical control structure that prioritizes the linear velocity and angular velocity regulation before allocating the propulsion force and steering torques. Therefore, the action vector of SAC algorithm has nine dimensions, namely nine PID gains.

$$a_t = [k_{xp}, k_{xi}, k_{xd}, k_{yp}, k_{yi}, k_{yd}, k_{\psi p}, k_{\psi i}, k_{\psi d}]. \tag{18}$$

In order to achieve excellent training results, this study designs a reward function that integrates factors including the distance sailed, USV velocity, and destination attainment success rate. The reward function is formulated with three branches, corresponding to three scenarios: successful arrival at the target point, deviation from the designated route, and failure to reach the intended destination despite remaining on track. The specific design of the reward function is outlined as follows:

$$r_t = \begin{cases} \rho_d d(i) + \rho_v v(i) + p, & \text{arrival} \\ \rho_d d(i) + \rho_v v(i) - p, & \text{off-course} \\ \rho_e e(i) + 1, & \text{otherwise} \end{cases} \tag{19}$$

where $d(i)$ means the distance traveled by the USV in the ith episode, $v(i)$ is the average velocity of the USV in the ith episode, and $e(i) = \sqrt{(\Delta x_i)^2 + (\Delta y_i)^2}$ means the distance between the next point and the current position. ρ_d, ρ_v and ρ_e are the coefficients corresponding to $d(i)$, $v(i)$ and $e(i)$ respectively. The distance traveled and average velocity are treated as reward function terms based on different considerations. In the early stage of training, the agent is unable to complete the tracking control of the entire trajectory. The distance traveled is a key factor influencing the reward value, which helps the agent to learn its early strategies. While speed is the key factor affecting the efficiency of trajectory tracking. In addition, p is a penalty item to motivate the SAC agent to complete decision-making tasks. When the USV reaches the destination, the reward function receives a positive reward $+p$. However, if the USV deviates from the course, the reward function will receive a negative reward $-p$. A visual representation of these variables is provided in Figure 4 and the hyper parameters of the SAC algorithm are given in Table 1.

Table 1 Settings of the SAC

Hyper Parameter	Value
Discount factor γ	0.99
Learning rating λ	0.0003
Mini-batch size	17
Replay buffer size	100000
Soft target update τ	0.005
Temperature parameter α	0.05

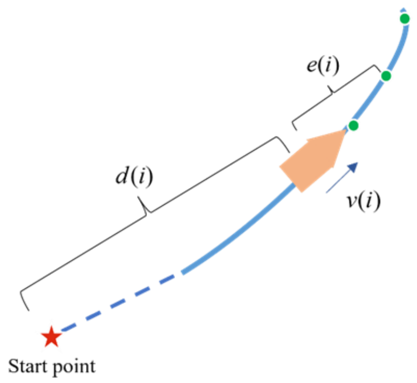


Fig. 4 The environmental information observed by the agent and the representation of reward function variables

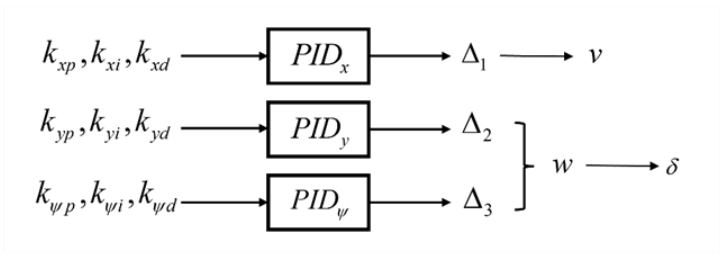


Fig. 5 Description of incremental PID controllers

3.2 Stability Analysis

To verify the stability of the SAC-PID controller with variable gains within a limited range, the following analysis is made.

The mathematical model of underactuated surface ship motion can be expressed as follows:

$$\begin{cases} \dot{x} = u\cos\psi - v\sin\psi \\ \dot{y} = u\sin\psi + v\cos\psi \\ \dot{\psi} = r \\ \dot{u} = f_u(\bar{v}) + \dfrac{1}{m_u}\tau_u + d_{\omega u} \\ \dot{v} = f_v(\bar{v}) + d_{\omega v} \\ \dot{r} = f_r(\bar{v}) + \dfrac{1}{m_r}\tau_r + d_{\omega r} \end{cases} \tag{20}$$

with

$$\begin{cases} f_u(\bar{v}) = \dfrac{m_v}{m_u}vr - \dfrac{d_u}{m_u}u - \dfrac{d_{u2}}{m_u}|u|u - \dfrac{d_{u3}}{m_u}u^3 \\ f_v(\bar{v}) = -\dfrac{m_u}{m_v}ur - \dfrac{d_v}{m_v}v - \dfrac{d_{v2}}{m_v}|v|v - \dfrac{d_{v3}}{m_v}v^3 \\ f_r(\bar{v}) = \dfrac{(m_u - m_v)}{m_r}uv - \dfrac{d_r}{m_r}r - \dfrac{d_{r2}}{m_r}|r|r - \dfrac{d_{r3}}{m_r}r^3 \end{cases} \tag{21}$$

where x, y and ψ are the vertical position coordinate, horizontal position coordinate and heading angle of the ship in the geographic coordinate system. u, v and r are the surge velocity, sway velocity, and yaw rate respectively. τ_u and τ_r represent the longitudinal main propulsion force and turning moment of the ship's control device. $d_{\omega u}$, $d_{\omega v}$ and $d_{\omega r}$ are used to describe the interference force and torque generated by external environmental interference. m_u, m_v, m_r, d_u, d_v, d_r, d_{u2}, d_{v2}, d_{r2}, d_{u3}, d_{v3}, d_{r3} are unknown or time-varying model parameters used to describe the inherent mass, added mass, and hydrodynamic damping of a ship. The parameters of nonlinear functions $f_u(\cdot)$, $f_v(\cdot)$, and $f_r(\cdot)$ are uncertain, and the function structures are also unknown, which are used to describe any uncertainty in the model.

For the convenience of stability analysis, this article makes the following assumptions.

Assumption 1: Assume that the environmental interference term satisfies that $d_{\omega u} \leq d_{u\max}$, $d_{\omega v} \leq d_{v\max}$, $d_{\omega r} \leq d_{r\max}$. $d_{u\max}$, $d_{v\max}$, and $d_{r\max}$ are all unknown positive constants.

Assumption 2: Assume that sway motion of the ship automatically satisfies the uniformly dissipative bounded property, namely the sway velocity v is passive-bounded.

Remark 2: *From Equations (35) and (38), it can be seen that the environmental disturbance terms need to comply with Assumption 1 to meet stability requirements. Considering the extreme case, when the environmental disturbance is sufficiently large, the ship may even capsize. Regarding Assumption 2, passive-boundedness of sway dynamic has been systematically analyzed [43]. This assumption is reasonable and highly realistic in practice because the hydrodynamic damping force plays a dominant role in the direction of sway, and the sway velocity is suppressed by this force. Natural factors such as water resistance and wave damping will cause the energy to gradually dissipate. Furthermore, Assumption 2 is helpful in determining the stability of the ship in the sway motion, thereby simplifying the analysis process and enabling the design of more effective control strategies. In general, the uniformly dissipative bounded property provides a theoretical basis for controller design.*

Define the following error variables:

$$\begin{cases} x_e = x_d - x \\ y_e = y_d - y \\ z_e = \sqrt{x_e^2 + y_e^2} \\ \psi_e = \psi_r - \psi \end{cases} \tag{22}$$

where x_d and y_d represent the vertical and horizontal coordinates of the expected track point in the designated path. ψ_r represents the deviation of the actual ship azimuth angle relative to the tangent line at the expected tracking point.

$$\psi_r = \frac{\pi}{2} \cdot [1 - \operatorname{sgn}(x_e)] \operatorname{sgn}(y_e) + arctan\left(\frac{y_e}{x_e}\right). \tag{23}$$

According to the relationship between the actual ship position and the expected track point, the combination of Equations (20) and (23) leads to

$$\begin{cases} \dot{z}_e = \dot{x}_d \cos\psi_r + \dot{y}_d \sin\psi_r - u\cos\psi_e - v\sin\psi_e \\ \dot{\psi}_e = \dot{\psi}_r - r \end{cases} \tag{24}$$

The surge velocity error and rudder angle error u_e, r_e are defined as:

$$\begin{cases} u_e = \alpha_u - u \\ r_e = \alpha_r - r \end{cases} \tag{25}$$

where α_u and α_r are ideal values obtained by the virtual control law.

Then, Equation (24) can be rewritten as follows:

$$\begin{cases} \dot{z}_e = \dot{x}_d \cos\psi_r + \dot{y}_d \sin\psi_r - (\alpha_u - u_e)\cos\psi_e - v\sin\psi_e \\ \dot{\psi}_e = \dot{\psi}_r - \alpha_r + r_e \end{cases} \tag{26}$$

The virtual control law α_u, α_r are designed as

$$\begin{cases} \alpha_u = (\cos\psi_e)^{-1}[\dot{x}_d \cos\psi_r + \dot{y}_d \sin\psi_r - v\sin\psi_e + k_{ze}z_e] + u_e \\ \alpha_r = k_{\psi e}\psi_e + \dot{\psi}_r \end{cases} \tag{27}$$

where $k_{ze} > 0$, $k_{\psi e} > 0$ are positive constants. According to the Equation (27), the virtual control signals can be only defined when $|\psi_e| < 0.5\pi$.

The combination of Equations (26) and (27) leads to

$$\begin{cases} \dot{z}_e = -k_{ze}z_e \\ \dot{\psi}_e = -k_{\psi e}\psi_e + r_e \end{cases} \tag{28}$$

Considering Equations (20) and (27), taking the derivative of Equation (25) yields

$$\begin{cases} \dot{u}_e = \dot{\alpha}_u - \dot{u} = B_u(\dot{x}_d, \ddot{x}_d, \dot{y}_d, \ddot{y}_d, \psi_r, \dot{\psi}_r, \psi_e, \dot{\psi}_e, \ddot{\psi}_e, v, \dot{v}) - f_u(\bar{v}) - \dfrac{1}{m_u}\tau_u - d_{\omega u} \\ \dot{r}_e = \dot{\alpha}_r - \dot{r} = B_r(\dot{\psi}_e, \ddot{\psi}_r) - f_r(\bar{v}) - \dfrac{1}{m_r}\tau_r - d_{\omega r} \end{cases} \tag{29}$$

where $B_i(\cdot)$ $(i = u, r)$ are continuous functions. Considering the hydrodynamic damping effect encountered by ships during sea voyages, and given that u, v, and r are all bounded variables, there must be a positive parameter M_i $(i = u, r)$ satisfies $|B_i(\cdot)| \le M_i$, indicating that $B_i(\cdot)$ has an upper bound.

According to Equation (1), the dynamic control law is designed as

$$\begin{cases} \tau_u = k_{pu}u_e + k_{iu}\displaystyle\int u_e dt + k_{du}\dot{u}_e \\ \tau_r = k_{pr}r_e + k_{ir}\displaystyle\int r_e dt + k_{dr}\dot{r}_e \end{cases} \tag{30}$$

where k_{pj}, k_{ij}, k_{dj} $(j = u, r)$ are PID gains.

Next, this paper will prove the stability of the system through Lyapunov method, and the relevant analysis is reflected in Theorem 1.

Theorem 1: If we assume that the conditions stated in Assumptions 1 and 2 are met for the underactuated vessel (20), it can be concluded that control laws (30) can ensure that all signals within the closed-loop system exhibit semi-global uniform ultimate boundedness (SUUB).

Define the following Lyapunov function

$$V = \frac{1}{2}u_e^2 + \frac{1}{2}r_e^2 + \frac{1}{2}z_e^2 + \frac{1}{2}\psi_e^2 \tag{31}$$

Differentiating Equation (31) yields

$$\dot{V} = u_e\dot{u}_e + r_e\dot{r}_e + z_e\dot{z}_e + \psi_e\dot{\psi}_e \tag{32}$$

Combination of Equations (29) and (30) lead to

$$\begin{aligned} u_e\dot{u}_e &= \frac{m_u u_e}{m_u + k_{du}}[B_u(\cdot) - f_u(\bar{v}) - d_{\omega u}] - \frac{k_{pu}}{m_u + k_{du}}u_e^2 - \frac{k_{iu}u_e}{m_u + k_{du}}\int u_e \mathrm{d}t \\ r_e\dot{r}_e &= \frac{m_r r_e}{m_r + k_{dr}}[B_r(\cdot) - f_r(\bar{v}) - d_{\omega r}] - \frac{k_{pr}}{m_r + k_{dr}}r_e^2 - \frac{k_{ir}r_e}{m_r + k_{dr}}\int r_e \mathrm{d}t \end{aligned} \tag{33}$$

From Young's inequality, the following inequalities hold

$$\begin{aligned} u_e\dot{u}_e &\le \left(\frac{1}{4} - \frac{k_{pu}}{m_u + k_{du}}\right)u_e^2 + D_u^2 - \frac{k_{iu}}{m_u + k_{du}}u_e\int u_e \mathrm{d}t \\ r_e\dot{r}_e &\le \left(\frac{1}{4} - \frac{k_{pr}}{m_r + k_{dr}}\right)r_e^2 + D_r^2 - \frac{k_{ir}}{m_r + k_{dr}}r_e\int r_e \mathrm{d}t \end{aligned} \tag{34}$$

where D_u and D_r are defined as

$$\begin{aligned} D_u &= \frac{m_u}{m_u + k_{du}}[B_u(\cdot) - f_u(\bar{v}) - d_{\omega u}] \\ D_r &= \frac{m_r}{m_r + k_{dr}}[B_r(\cdot) - f_r(\bar{v}) - d_{\omega r}] \end{aligned} \tag{35}$$

Considering Equation (28), we can obtain

$$\begin{aligned} z_e\dot{z}_e &= -k_{ze}z_e^2 \\ \psi_e\dot{\psi}_e &= -k_{\psi e}\psi_e^2 + \psi_e r_e \end{aligned} \tag{36}$$

From Young's inequality, the following inequality holds

$$\psi_e \dot{\psi}_e \leq \left(\frac{1}{4} - k_{\psi e}\right) \psi_e^2 + r_e^2 \tag{37}$$

On the basis of the above analysis, it can be obtained that

$$\begin{aligned}\dot{V} &\leq -\left(\frac{k_{pu}}{m_u + k_{du}} - \frac{1}{4}\right) u_e^2 + D_u^2 - \frac{k_{iu}}{m_u + k_{du}} u_e \int u_e \mathrm{d}t \\ &\quad -\left(\frac{k_{pr}}{m_r + k_{dr}} - \frac{5}{4}\right) r_e^2 + D_r^2 - \frac{k_{ir}}{m_r + k_{dr}} r_e \int r_e \mathrm{d}t - k_{ze} z_e^2 - \left(k_{\psi e} - \frac{1}{4}\right) \psi_e^2 \\ &\leq -2aV + \sigma\end{aligned} \tag{38}$$

where a and σ are defined as

$$\begin{aligned} a &= \min\left\{\frac{k_{pu}}{m_u + k_{du}} - \frac{1}{4}, \frac{k_{pr}}{m_r + k_{dr}} - \frac{5}{4}, k_{ze}, k_{\psi e} - \frac{1}{4}\right\} \\ \sigma &= D_u^2 + D_r^2 - \frac{k_{iu}}{m_u + k_{du}} u_e \int u_e \mathrm{d}t - \frac{k_{ir}}{m_r + k_{dr}} r_e \int r_e \mathrm{d}t \end{aligned} \tag{39}$$

Integrating Equation (38) yields

$$V(t) \leq \frac{\sigma}{2a} + \left[V(0) - \frac{\sigma}{2a}\right] e^{-2at} \tag{40}$$

Note that V is bounded by $\sigma/2a$ as $t \to \infty$. It can be concluded that u_e, r_e, z_e and ψ_e are bounded if the design constants and PID gains are appropriately selected. Therefore, all the error signals are SGUUB under the above design.

4. Results and Discussion

The PyBullet simulation platform is designed for robotics technology, simulating continuous physical states such as gravity and collisions. Considering the scalability of the physical model, the simplicity of data acquisition and the authenticity of the physical simulation, this study selects PyBullet for simulation experiments. Also, a systematic numerical validation experiment for trajectory tracking control was performed using the ship dynamics model defined in Equation 3, with the experimental parameters configured in the study [44].

The training path of DRL algorithm is a curve, whose function expression is as follows:

$$x = y \sin(0.2y)/4 + y \cos(0.6y)/5 + 2\sin(y). \tag{41}$$

In the field of navigation, x represents the vertical axis and y represents the horizontal axis, both in units of hectometer. The number of sampling points on a trajectory is set to 200.

In terms of research methods, this study uses traditional tuning methods, genetic algorithm, DDPG algorithm, and SAC algorithm to tune the PID gains. The traditional tuning method is mainly the Ziegler-Nichols (Z-N) method, which is further divided into the constant amplitude oscillation method and the response curve method. However, the equal amplitude oscillation method cannot be directly applied to second-order systems, and the response curve method cannot be applied to systems with integral components. Therefore, this paper first approximates the second-order ship model to a first-order model, and then uses the constant amplitude oscillation method for PID gains tuning. Because genetic algorithm is self-organizing, self-adaptive and self-learning, and can avoid falling into local optimal, it is chosen as the representative of heuristic algorithm. In the experiment of genetic algorithm tuning PID gain, the population size is set to 100, the number of iterations is set to 1000, the crossover probability is 0.9, and the mutation probability is 0.4. As the predominant DRL algorithms, DDPG and SAC are both off-policy. The difference is that the former trains a deterministic strategy, while the latter trains a stochastic strategy.

The average reward curve and success rate curve of DDPG are presented in Figure 6(a) and (b). As a comparison, Figure 7 shows the training curve of SAC. Comparing Figure 6(a) and Figure 7(a), it is evident that DDPG converges to a reward value of -5 after 2000 episodes, while SAC converges after just 400 episodes, with a reward value fluctuating around 90. From Figure 6(b) and Figure 7(b), it can be seen that the success rate of DDPG has been fluctuating around 40 %, while the success rate of SAC has stabilized around 90 % as the algorithm converges. It can be concluded that SAC has a faster convergence speed, higher reward value and success rate, which is undoubtedly superior to DDPG.

Remark 3: *DDPG is more sensitive to hyperparameters, and SAC has stronger exploration ability due to the presence of entropy term, which may be the reason for the comparison result of curves in Figure 6(a) and Figure 7(a). The definition of success in this experiment is that the USV successfully reaches the destination along a given trajectory within a certain error range. Therefore, improving the success rate is more difficult than increasing the average reward value, and even if the average reward gradually converges, the success rate may not necessarily converge. For example, in all sampling points, a single significant error may have negligible impact on the average reward value, however, it can result in the episode being judged as a failure.*

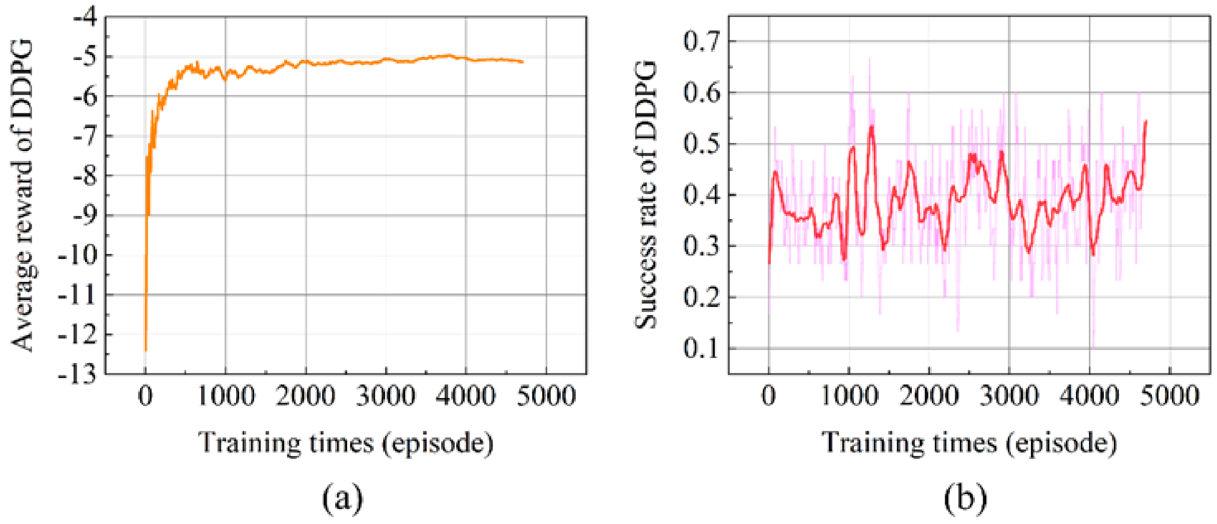


Fig. 6 Average reward curve (a) and success rate curve (b) of DDPG

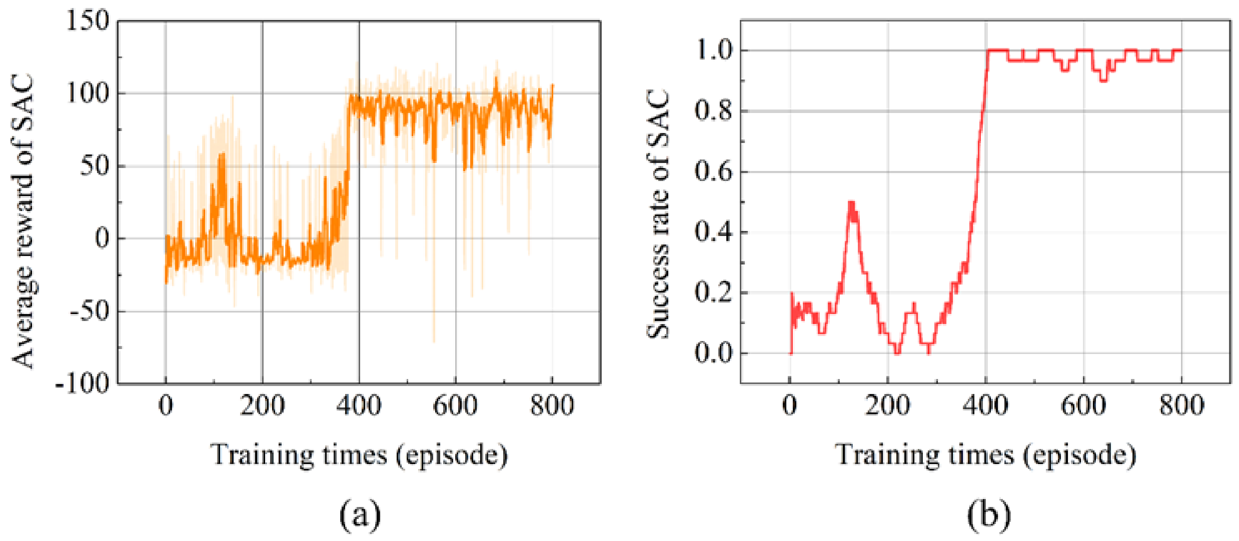


Fig. 7 Average reward curve (a) and success rate curve (b) of SAC

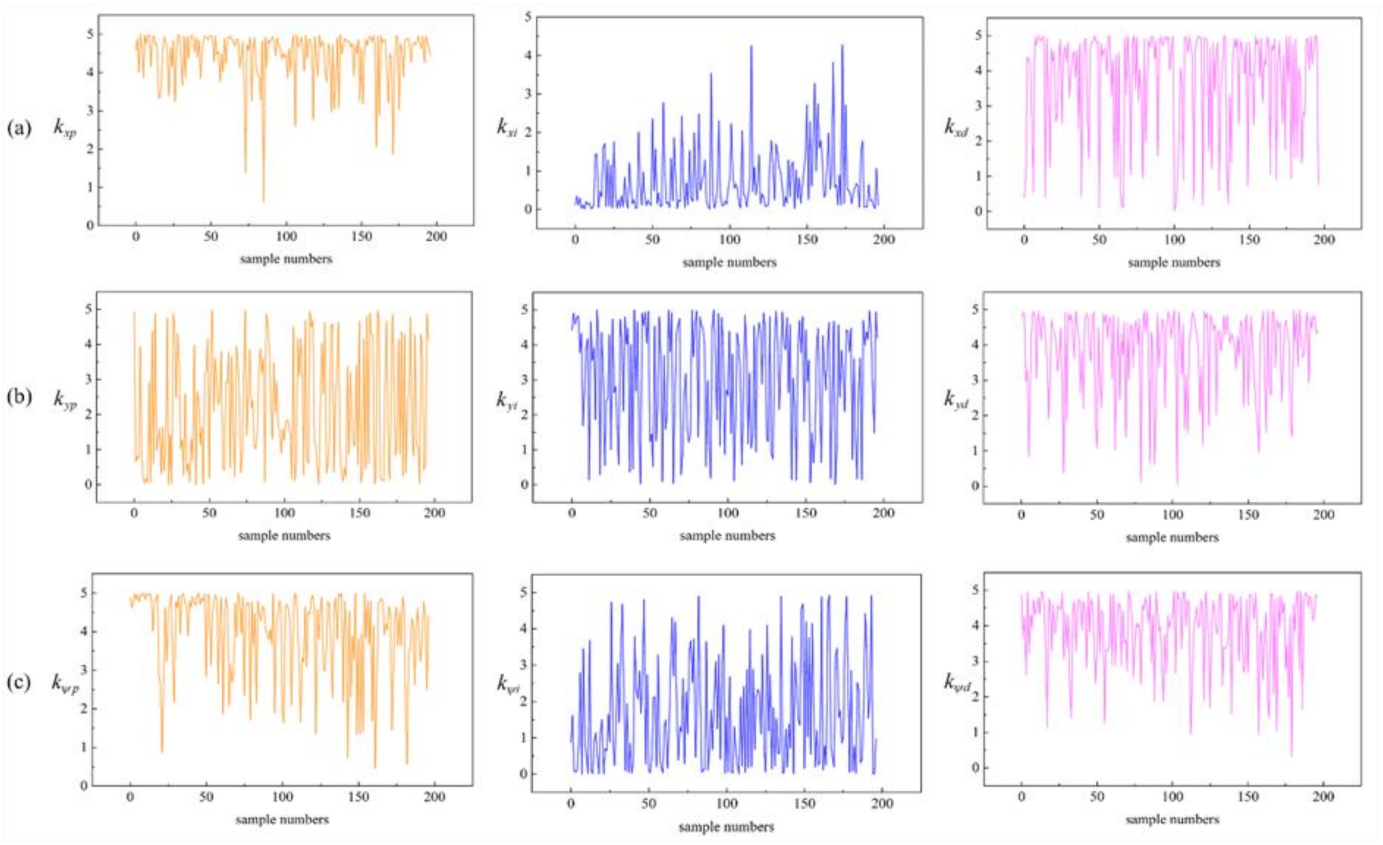


Fig. 8 The output curves of actions (k_p, k_i, k_d variables of (a) PID$_x$, (b) PID$_y$ and (c) PID$_\psi$ in Equation (18))

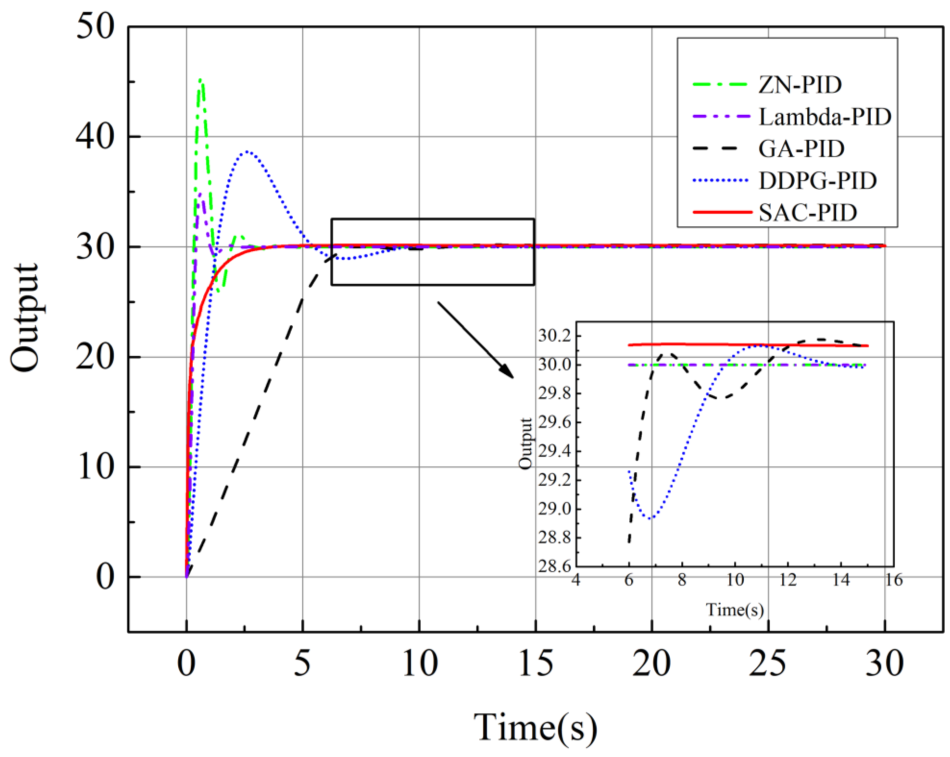


Fig. 9 Comparison of response curves for different methods

The output curves of all action variables in Equation (18) are illustrated in Figure 8, representing the selection of 9 PID gains during the trajectory tracking process. Based on the stability analysis in Section 3, it is imperative to restrict the PID gains within a specific range to ensure controller stability. Consequently, through preliminary experiments, we establish an interval [0,5] for selecting appropriate PID gains. From Figure 8, it can be seen that unlike traditional PID methods where the gains are fixed after selection, the PID gains output by the SAC agent vary at 200 sampling points during trajectory tracking. It means that the SAC-PID method has superior adaptability in addressing potential environmental challenges.

Table 2 Average Trajectory Tracking Errors of Different Methods

Method	Average error (hm)
ZN-PID	0.264865
Lambda-PID	0.304641
GA-PID algorithm	0.479417
DDPG-PID algorithm	0.335141
SAC-PID algorithm	0.260949

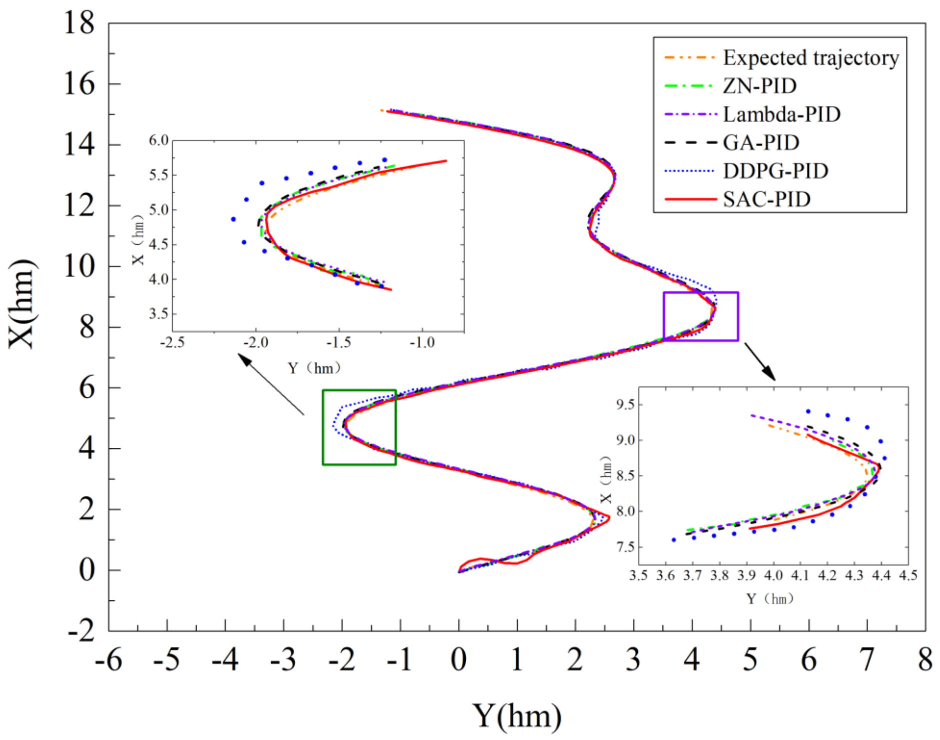


Fig.10 Comparison of trajectory tracking performance of different PID gains tuning methods

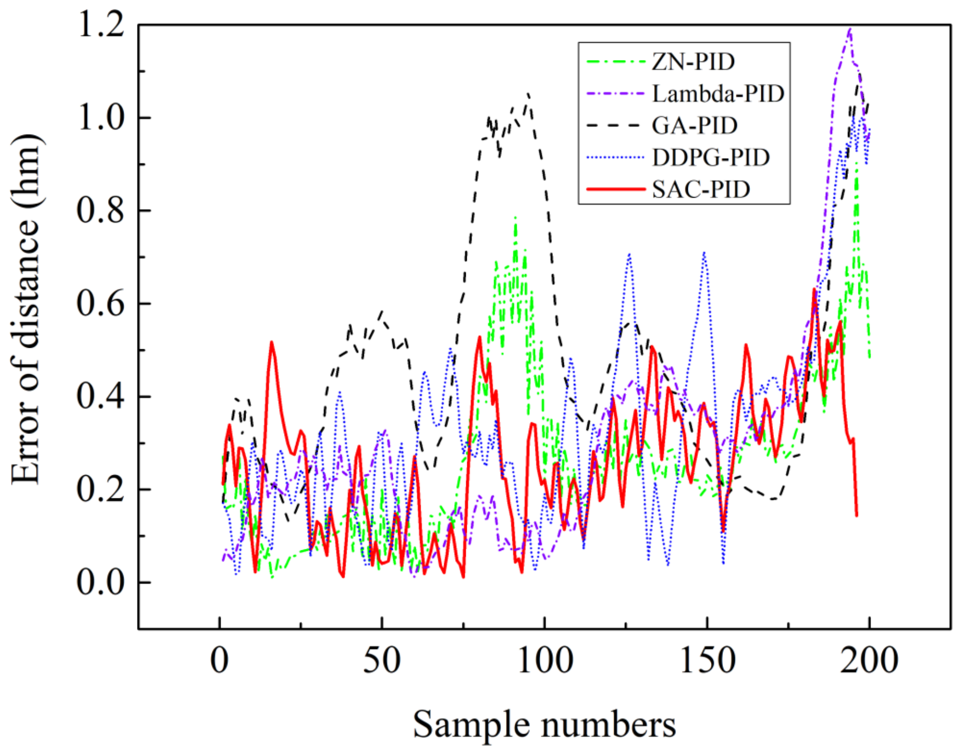


Fig.11 Error curves of trajectory tracking by different methods

Then, this study takes the rudder angle controller as an example to compare the response curves of five PID tuning methods. The command course signal is set to 30° and the comparison result is shown in Figure 9. In terms of dynamic performance, the main considerations are settling time and overshoot. The traditional methods of ZN-PID and Lambda-PID have relatively short regulation time, but they have relatively large overshoot, which is likely to cause system output oscillation and affect system stability. The overshoot of DDPG-PID is 26.7 %, which is the second largest and the adjustment time is relatively long. As for GA-PID, the overshoot is small enough, but the settling time is not satisfactory. The proposed SAC-PID not only ensures a small overshoot but also has a sufficiently short settling time. In terms of steady-state performance, it can be seen from the local magnification diagram that the traditional PID tuning method exhibits the smallest steady-state error, rather than SAC-PID. However, the steady-state error of SAC-PID is only 0.4 %, which is within the allowable range 2~5 %. On the other hand, it can be argued that despite this set of actions leading to control effects, the agent still receives substantial rewards, suggesting that this error is inconsequential for the reward function evaluation system. Therefore, it can be inferred that the SAC-PID trajectory controller outperforms the traditional tuning method, the optimization-based tuning method, and the DDPG-PID in terms of dynamic performance and steady-state performance.

Figure 10 illustrates the effects of different PID tuning methods on trajectory tracking control. The expected trajectory is obtained from Equation (41), with the starting point set to (0, 0) and the red pentagram representing the destination. While all five methods can reach the destination, the locally enlarged images in Figure 10 reveal that the trajectory tracking control faces challenges at the turning point, with DDPG-PID exhibiting the largest trajectory tracking error among all methods.

In order to conduct quantitative research, 200 expected trajectory points are sampled to calculate the error between the actual and expected trajectory points under different method. The trajectory tracking error variation curve is depicted in Figure 11. It is viewed that the five methods had significant trajectory tracking errors near the 100th and 200th sampling points, with GA-PID having the highest trajectory tracking error of 110 m. The trajectory tracking error of SAC-PID exhibits relatively gentle fluctuations. The average trajectory tracking errors of different PID tuning methods are given in Table 2. It can be seen that GA-PID has the highest average track error of 0.479417, while SAC-PID has the lowest average track error of 0.260949.

Therefore, it can be concluded that the proposed SAC-PID not only enables real-time tuning of PID gains but also achieves the best trajectory tracking effect.

Remark 4: *The most intuitive effect of trajectory tracking error is illustrated in Figure 10, which depicts the degree of deviation between routes guided by different methods and the desired route. However, a deeper analysis of the tracking errors through 200 expected track points reveals an interesting situation: although the average track tracking error is largest, it does not correspond to the overall largest deviation from the expected route. This discrepancy arises because although there may be significant errors between actual and desired trajectory points for the USV, these actual trajectory points happen to fall on the subsequent expected route and thus cannot be intuitively reflected in terms of overall route deviations. Fortunately, quantitative analysis compensates for such deficiency.*

Considering the distribution of obstacles in the Setouchi Sea of Japan, a simulation environment was constructed as Scene 1. This study initially planned the expected trajectory and imposed wind-wave interference at the second turning point. The effective wave height was 0.15 m, the wave period was 5.5 s and the average wind speed was 0.45 m/s. The tracking control effect of the trajectory under Scene 1 is shown in Figure 12, and the trajectory tracking error is presented in Table 3. In addition, based on the actual sea conditions in the waters near Ise Bay and Takashi Island in Japan, this article has set up Scene 2 and planed the expected trajectory. Afterwards, the trajectory tracking control effects of different methods are compared, as shown in Figure 13. The black area represents land, the gray area represents shallow water, and the white area represents deep water.

Table 3 Average Trajectory Tracking Errors of Different Methods in actual sea conditions

Method	Average Error of Scene 1 (hm)	Average Error of Scene 2 (hm)
ZN-PID	0.700442	0.442924
Lambda-PID	0.278640	0.264314
GA-PID algorithm	0.642305	0.476531
DDPG-PID algorithm	0.179155	0.178038
SAC-PID algorithm	0.145078	0.142113

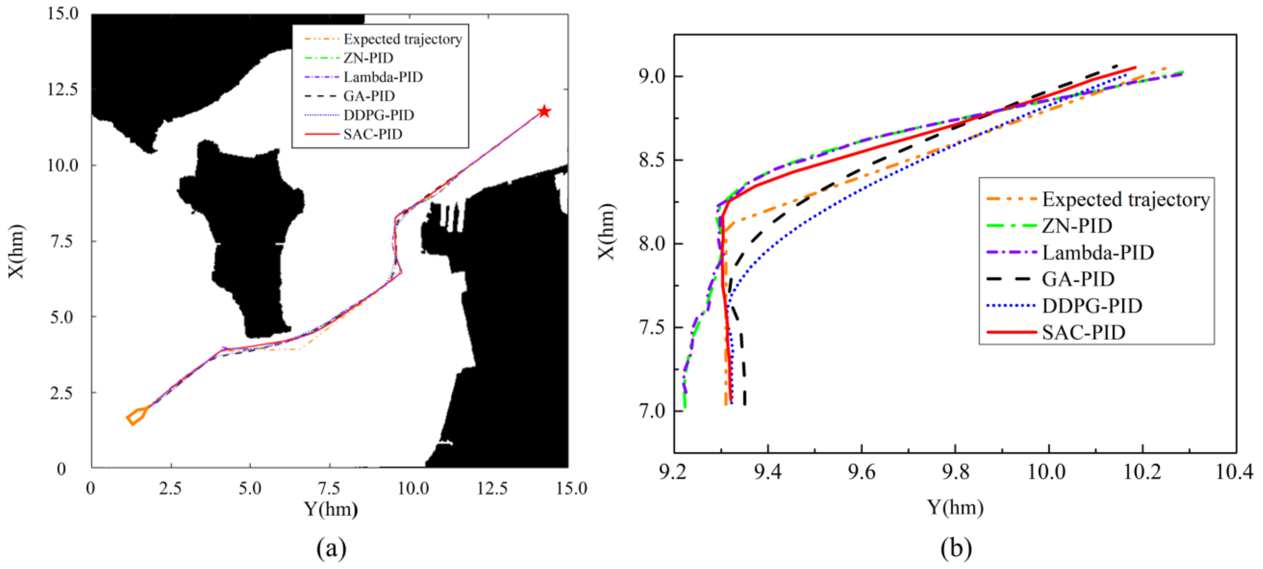


Fig.12 Comparison of trajectory control effects of different methods in Scene 1, (a) real sea conditions and (b) partial enlarged image

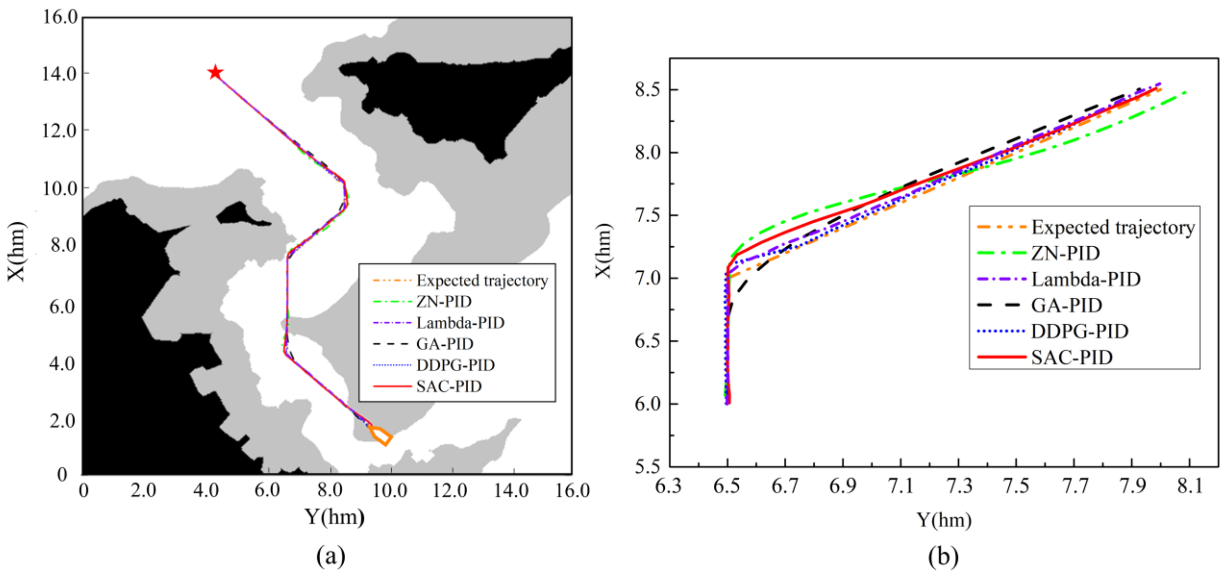


Fig.13 Comparison of trajectory control effects of different methods in Scene 2, (a) real sea conditions and (b) partial enlarged image

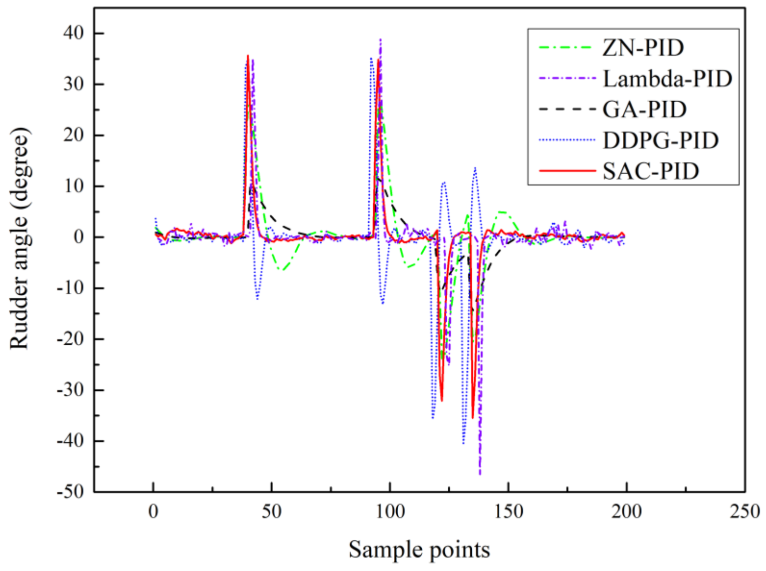


Fig.14 Comparison of ship rudder angle variation curves under the guidance of various methods in Scene 2

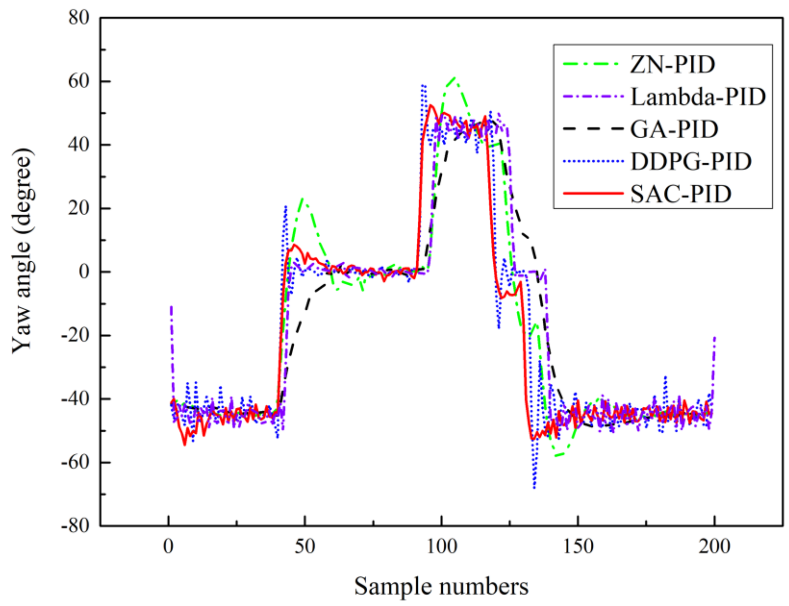


Fig.15 Comparison of yaw angle variation curves under the guidance of various methods in Scene 2

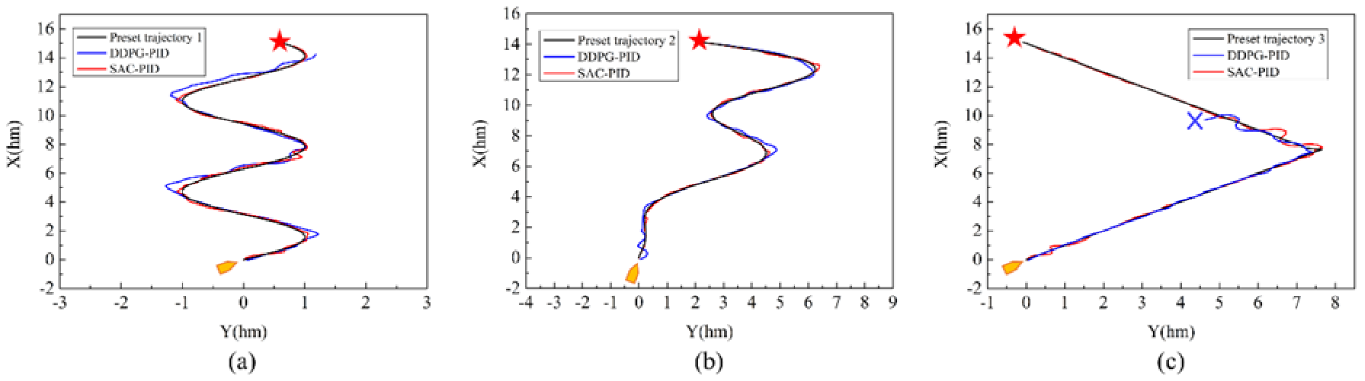


Fig.16 Generalization experiments of DDPG-PID and SAC-PID

From Figure 12(b) and Figure 13(b), it can be seen that trajectory control faces challenges at turning points of the route. Combined with Table 3, it can be seen that SAC-PID has the minimum average trajectory tracking error. In order to deeply explore the superiority of SAC-PID, this paper takes the experimental results in Scene 2 as an example to analyze the changes of rudder angle and yaw angle in the process of trajectory control. To further investigate the changes in rudder angle during the trajectory tracking process, its variation curve is depicted in Figure 14, with four large rudder angle changes corresponding to four turning points of the route. Due to the large trajectory tracking errors of traditional PID and GA-PID, our focus lies in comparing the rudder angle change curves of the remaining two deep reinforcement learning methods. From Figure 14, it can be seen that the rudder angle curve of DDPG-PID fluctuates frequently, which is not conducive to safe navigation of ships. Therefore, the trajectory control effect of SAC-PID is optimal.

In addition, the yaw angle variation during the trajectory control process of Scene 2 is analyzed in Figure 15. It can be seen that SAC-PID performs better in maintaining the heading during the straight-line segment trajectory tracking. Meanwhile, at the turning points, it basically does not have the excessive heading variation, while other methods have frequent or even excessive heading variations. Therefore SAC-PID can enhance the safe and stable navigation of unmanned vessels.

In order to verify the generalization of the two deep reinforcement learning algorithms discussed in this study, multiple generalization experiments were conducted, and three typical cases were selected and presented in Figure 16. Unlike the training scenario in Figure 10, the trajectories illustrated in Figure 16 can be deemed as new environments that have not been exposed to the DRL agents. In Figure 16(a), a regular sinusoidal curve is employed to depict the preset trajectory, while in Figure 16(b), the preset trajectory is an irregular curve. In both generalization scenarios, DDPG-PID and SAC-PID successfully reach the destination symbolized by the red pentagram, indicating proficient generalization. However, SAC-PID exhibits a smaller deviation error compared to DDPG-PID. When the preset trajectory transforms into a broken line as shown in Figure 16(c), it can be intuitively observed that both DRL methods face challenges at sharp corners where their ability to track the trajectory begins to deteriorate. Nevertheless, the SAC-PID successfully adjusts its trajectory and reaches the destination point, whereas DDPG-PID fails to do so. Consequently, it can be inferred that SAC-PID outperforms DDPG-PID in terms of generalization capacity.

5. Conclusion

In this study, a USV adaptive trajectory controller based on the SAC-PID algorithm was developed, which combines deep reinforcement learning with the control algorithm to address the limitations of relying on manual experience for PID tuning and maintaining fixed gains once tune. The primary contributions of this article encompass: (1) conducting a comparative analysis on conventional methods, GA-PID, DDPG-PID and SAC-PID in terms of response curve and trajectory tracking error; (2) investigating the stability of the trajectory controller under varying gains within a specific range; (3) exploring the generalization capacities of DDPG-PID and SAC-PID. The final results demonstrate that the proposed SAC-PID algorithm not only ensures controller stability, but also achieves superior gain sequences, enhances trajectory tracking performance, and exhibits exceptional generalization capability.

This research has important practical significance in the application of ocean engineering. As one of DRL algorithms, SAC algorithm can adjust PID parameters in real time according to system state and environmental changes, improve the dynamic performance and robustness of the control system, and reduce the tuning cost. In addition, this method can expand its application scope and be applied to ocean engineering fields such as offshore wind turbine system, ship roll stability control and autonomous underwater vehicles.

However, this study still has some limitations. The training of the SAC-PID algorithm has certain requirements for computer configuration, and when facing complex environments, the SAC agent is difficult to find the optimal policy within a short period of time. When tracking trajectories with significant steering turns, the tracking effect is largely influenced by the environmental information observed by the agent. Future research will mainly focus on reducing the computational complexity of the algorithm, optimizing the state space and reward function, and studying the anti-interference performance, in order to deploy it on actual ships for real-scene ship experiments.

ACKNOWLEDGEMENTS

The paper is partially supported by National Natural Science Foundation of China (51409033, 52171342), Dalian Innovation Team Support Plan in the Key Research Field (2020RT08), and the Fundamental Research Funds for the Central Universities (3132023502). The authors would like to thank the reviewers for their valuable comments.

REFERENCES

[1] Li, Y., Tang, Z., Gong, J., 2023. The effect of PID control scheme on the course-keeping of ship in oblique stern waves. *Brodogradnja*, 74(4), 155-178. https://doi.org/10.21278/brod74408

[2] Guan, W., Peng, H., Zhang, X., Sun, H., 2022. Ship steering adaptive CGS control based on EKF identification method. *Journal of Marine Science & Engineering*, 10(2), 294-307. https://doi.org/10.3390/jmse10020294

[3] He, Y., Zhao, X., Huang, L., Xu, L., Liu, J., 2024. Control method for the ship track and speed in curved channels. *Brodogradnja*, 75(3), 1-27. https://doi.org/10.21278/brod75307

[4] Fan, Y., Sun, X., Wang, G., Guo, C., 2015. On Fuzzy Self-adaptive PID Control for USV Course. 34th Chinese Control Conference (CCC), Hangzhou, CN.

[5] Xiang, X., Yu, C., Zhang, Q., 2017. Robust fuzzy 3D path following for autonomous underwater vehicle subject to uncertainties. *Computers & Operations Research*, 84, 165-177. https://doi.org/10.1016/j.cor.2016.09.017

[6] Chen, C. L. P., Wen, G.-X., Liu, Y.-J., Liu, Z., 2016. Observer-Based Adaptive Backstepping Consensus Tracking Control for High-Order Nonlinear Semi-Strict-Feedback Multiagent Systems. *IEEE Transactions on Cybernetics*, 46, 1591-1601. https://doi.org/10.1109/TCYB.2015.2452217

[7] Liu, S., Liu, Y., Wang, N., 2017. Nonlinear disturbance observer-based backstepping finite-time sliding mode tracking control of underwater vehicles with system uncertainties and external disturbances. *Nonlinear Dynamics*, 88, 465-476. https://doi.org/10.1007/s11071-016-3253-8

[8] Gonzalez-Garcia, A., Castaneda, H., 2021. Guidance and Control Based on Adaptive Sliding Mode Strategy for a USV Subject to Uncertainties. *IEEE Journal of Oceanic Engineering*, 46, 1144-1154. https://doi.org/10.1109/JOE.2021.3059210

[9] Zhao, Y., Qi, X., Ma, Y., Li, Z., Malekian, R., Sotelo, M. A., 2021. Path Following Optimization for an Underactuated USV Using Smoothly-Convergent Deep Reinforcement Learning. *IEEE Transactions on Intelligent Transportation Systems*, 22, 6208-6220. https://doi.org/10.1109/TITS.2020.2989352

[10] Paulig, N., Okhrin, O., 2024. Robust path following on rivers using bootstrapped reinforcement learning. *Ocean Engineering*, 298, 117207. https://doi.org/10.1016/j.oceaneng.2024.117207

[11] Pan, Y., Li, X., Yu, H., 2019. Efficient PID Tracking Control of Robotic Manipulators Driven by Compliant Actuators. *IEEE Transactions on Control Systems Technology*, 27, 915-922. https://doi.org/10.1109/TCST.2017.2783339

[12] Gao, J., Zhang, Y., 2024. Ship collision avoidance decision-making research in coastal waters considering uncertainty of target ships. *Brodogradnja*, 75(2), 1-16. https://doi.org/10.21278/brod75203

[13] Somefun, O. A., Akingbade, K., Dahunsi, F., 2021 . The dilemma of PID tuning. *Annual Reviews in Control*, 52, 65-74. https://doi.org/10.1016/j.arcontrol.2021.05.002

[14] Berner, J., Soltesz, K., Hagglund, T., Astrom, K. J., 2018. An experimental comparison of PID autotuners. *Control Engineering Practice*, 73, 124-133. https://doi.org/10.1016/j.conengprac.2018.01.006

[15] Sahib, M. A., Ahmed, B. S., 2016. A new multiobjective performance criterion used in PID tuning optimization algorithms. *Journal of Advanced Research*, 7, 125-134. https://doi.org/10.1016/j.jare.2015.03.004

[16] Zhang, Y., Jia, Y., Chai, T., Wang, D., Dai, W., Fu, J., 2018. Data-Driven PID Controller and Its Application to Pulp Neutralization Process. *IEEE Transactions on Control Systems Technology*, 26, 828-841. https://doi.org/10.1109/TCST.2017.2695981

[17] Verma, B., Padhy, P. K., 2018. Optimal PID controller design with adjustable maximum sensitivity. *IET Control Theory and Applications*, 12, 1156-1165. https://doi.org/10.1049/iet-cta.2017.1078

[18] Verma, B., Padhy, P. K., 2020. Robust Fine Tuning of Optimal PID Controller With Guaranteed Robustness. *IEEE Transactions on Industrial Electronics*, 67, 4911-4920. https://doi.org/10.1109/TIE.2019.2924603

[19] Carlucho, I., De Paula, M., Acosta, G. G., 2019. Double Q-PID algorithm for mobile robot control. *Expert Systems with Applications*, 137, 292-307. https://doi.org/10.1016/j.eswa.2019.06.066

[20] Udekwe, D., Ajayi, O.-O., Ubadike, O., Ter, K., Okafor, E., 2024. Comparing actor-critic deep reinforcement learning controllers for enhanced performance on a ball-and-plate system. *Expert Systems with Applications*, 245, 123055. https://doi.org/10.1016/j.eswa.2023.123055

[21] Gheisarnejad, M., Khooban, M. H., 2021. An Intelligent Non-Integer PID Controller-Based Deep Reinforcement Learning: Implementation and Experimental Results. *IEEE Transactions on Industrial Electronics*, 68, 3609-3618. https://doi.org/10.1109/TIE.2020.2979561

[22] Carlucho, I., De Paula, M., Acosta, G. G., 2020. An adaptive deep reinforcement learning approach for MIMO PID control of mobile robots. *ISA Transactions*, 102, 280-294. https://doi.org/10.1016/j.isatra.2020.02.017

[23] Yu, X., Fan, Y., Xu, S., Ou, L., 2022. A self-adaptive SAC-PID control approach based on reinforcement learning for mobile robots. *International Journal of Robust and Nonlinear Control*, 32, 9625-9643. https://doi.org/10.1002/rnc.5662

[24] Shi, Q., Lam, H.-K., Xuan, C., Chen, M., 2020. Adaptive neuro-fuzzy PID controller based on twin delayed deep deterministic policy gradient algorithm. *Neurocomputing*, 402, 183-194. https://doi.org/10.1016/j.neucom.2020.03.063

[25] Chowdhury, M. A., Lu, Q., 2023. A Novel Entropy-Maximizing TD3-based Reinforcement Learning for Automatic PID Tuning. *In Proceedings of the American Control Conference*, San Diego, CA. https://doi.org/10.23919/ACC55779.2023.10156246

[26] Dogru, O., Velswamy, K., Ibrahim, F., Wu, Y., Sundaramoorthy, A. S., Huang, B., Xu, S., Nixon, M., Bell, N., 2022. Reinforcement learning approach to autonomous PID tuning. *Computers & Chemical Engineering*, 161, 107760. https://doi.org/10.1016/j.compchemeng.2022.107760

[27] Li, T., Cui, W., Liu, X., Li, X., Xie, N., Wang, Y. 2023. A Physiological Control System for Pulsatile Ventricular Assist Device Using an Energy-Efficient Deep Reinforcement Learning Method. *IEEE Transactions on Instrumentation and Measurement*, 72, 1-10. https://doi.org/10.1109/TIM.2023.3277993

[28] Wang, X., Wang, R., Jin, M., Shu, G., Tian, H., Pan, J., 2020. Control of superheat of organic Rankine cycle under transient heat source based on deep reinforcement learning. *Applied Energy*, 278, 115637. https://doi.org/10.1016/j.apenergy.2020.115637

[29] Shuprajhaa, T., Sujit, S. K., Srinivasan, K., 2022. Reinforcement learning based adaptive PID controller design for control of linear/nonlinear unstable processes. *Applied Soft Computing*, 128, 109450. https://doi.org/10.1016/j.asoc.2022.109450

[30] Chu, Z., Sun, B., Zhu, D., Zhang, M., Luo, C., 2020. Motion control of unmanned underwater vehicles via deep imitation reinforcement learning algorithm. *IET Intelligent Transport Systems*, 14, 764-774. https://doi.org/10.1049/iet-its.2019.0273

[31] Lee, D., Lee, S. J., Yim, S. C., 2020. Reinforcement learning-based adaptive PID controller for DPS. *Ocean Engineering*, 216, 108053. https://doi.org/10.1016/j.oceaneng.2020.108053

[32] Lai, P., Liu, Y., Zhang, W., Xu, H., 2023. Intelligent controller for unmanned surface vehicles by deep reinforcement learning. *Physics of Fluids*, 35, 037111. https://doi.org/10.1063/5.0139568

[33] Nian, R., Liu, J., Huang, B., 2020. A review on reinforcement learning: Introduction and applications in industrial process control. *Computers & Chemical Engineering*, 139, 106886. https://doi.org/10.1016/j.compchemeng.2020.106886

[34] Vouros, G. A., 2022. Explainable deep reinforcement learning: state of the art and challenges. *ACM Computing Surveys*, 55(5), 1-39. https://doi.org/10.1145/3527448

[35] Mo, K., Ye, P., Ren, X., Wang, S., Li, W., Li, J., 2024. Security and privacy issues in deep reinforcement learning: Threats and countermeasures. *ACM Computing Surveys*, 56(6), 1-39. https://doi.org/10.1145/3640312

[36] Lawrence, N. P., Forbes, M. G., Loewen, P. D., Mcclement, D. G., Backstrom, J. U., Gopaluni, R. B., 2022. Deep reinforcement learning with shallow controllers: An experimental application to PID tuning. *Control Engineering Practice*, 121, 105046. https://doi.org/10.1016/j.conengprac.2021.105046

[37] Ma, Z., Pan, T., 2023. Adaptive Weight Tuning of EWMA Controller via Model-Free Deep Reinforcement Learning. *IEEE Transactions on Semiconductor Manufacturing*, 36, 91-99. https://doi.org/10.1109/TSM.2022.3225480

[38] Wang, Y., Fang, S., Hu, J., 2023. Active Disturbance Rejection Control Based on Deep Reinforcement Learning of PMSM for More Electric Aircraft. *IEEE Transactions on Power Electronics*, 38, 406-416. https://doi.org/10.1109/TPEL.2022.3206089

[39] Wang, Z., Xiang, X., Xiong, X., Yang, S., 2025. Position-based acoustic visual servo control for docking of autonomous underwater vehicle using deep reinforcement learning. *Robotics and Autonomous Systems*, 186, 104914. https://doi.org/10.1016/j.robot.2024.104914

[40] Fossen, T. I., 2011. Handbook of Marine Craft Hydrodynamics and Motion Control, *John Wiley and Sons*, Sussex, UK. https://doi.org/10.1002/9781119994138

[41] Cui, Z., Guan, W., Luo, W., Zhang, X., 2023. Intelligent navigation method for multiple marine autonomous surface ships based on improved PPO algorithm. *Ocean Engineering*, 287, 115783. https://doi.org/10.1016/j.oceaneng.2023.115783

[42] Haarnoja, T., Zhou, A., Abbeel, P., Levine, S., 2018. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. *35th International Conference on Machine Learning (ICML)*, 10-15 July, Stockholm, Sweden.

[43] Li, J.-H., Lee, P.-M., Jun, B.-H., Lim, Y.-K., 2008. Point-to-point navigation of underactuated ships. *Automatica*, 44, 3201-3205. https://doi.org/10.1016/j.automatica.2008.08.003

[44] Cui Z, Guan W, Zhang X., 2024. Collision avoidance decision-making strategy for multiple USVs based on Deep Reinforcement Learning algorithm. *Ocean Engineering*, 308, 118323. https://doi.org/10.1016/j.oceaneng.2024.118323