Hindawi
Complexity
Volume 2021, Article ID 6649625, 25 pages
https://doi.org/10.1155/2021/6649625

Research Article

Deep Reinforcement Learning for Vectored Thruster Autonomous Underwater Vehicle Control

Tao Liu,[1] Yuli Hu,[2] and Hui Xu[1]

[1]*School of Marine Science and Technology, Northwestern Polytechnical University, Xi'an, China*
[2]*Key Laboratory for Unmanned Underwater Vehicle, Northwestern Polytechnical University, Xi'an 710072, China*

Correspondence should be addressed to Yuli Hu; yulihu001@gmail.com

Received 2 December 2020; Revised 2 March 2021; Accepted 31 March 2021; Published 23 April 2021

Academic Editor: Shenggang Li

Autonomous underwater vehicles (AUVs) are widely used to accomplish various missions in the complex marine environment; the design of a control system for AUVs is particularly difficult due to the high nonlinearity, variations in hydrodynamic coefficients, and external force from ocean currents. In this paper, we propose a controller based on deep reinforcement learning (DRL) in a simulation environment for studying the control performance of the vectored thruster AUV. RL is an important method of artificial intelligence that can learn behavior through trial-and-error interactions with the environment, so it does not need to provide an accurate AUV control model that is very hard to establish. The proposed RL algorithm only uses the information that can be measured by sensors inside the AUVs as the input parameters, and the outputs of the designed controller are the continuous control actions, which are the commands that are set to the vectored thruster. Moreover, a reward function is developed for deep RL controller considering different factors which actually affect the control accuracy of AUV navigation control. To confirm the algorithm's effectiveness, a series of simulations are carried out in the designed simulation environment, which is a method to save time and improve efficiency. Simulation results prove the feasibility of the deep RL algorithm applied to the control system for AUV. Furthermore, our work also provides an optional method for robot control problems to deal with improving technology requirements and complicated application environments.

1. Introduction

Since oceans are the most important source in terms of marine life, all kinds of scarce minerals, marine chemical, ocean energy and transportation, and human societies are increasingly dependent on the oceans. Therefore, exploring, developing, exploiting, and protecting the ocean have become a hot issue of global development and technical equipment. Thus, an enormous amount of research effort goes into research and development of all kinds of instruments and equipment, such as large numbers of underwater robots. Unmanned underwater vehicles are a kind of ideal platform to carry out ocean surveying and monitoring [1]. Because the natural environment of the ocean is so harsh for human beings to investigate, autonomous underwater vehicles (AUVs) as ideal platforms for the significant improvement in their performance are widely used for exploring and utilizing resources by carrying different detecting and operating instruments [2, 3]. Although the performance of AUVs has obtained a huge development, there are still a lot of challenging problems appealing to scientists and engineers immensely in this field. For example, the conventional AUVs are unable to perform detailed inspection missions at zero and low forward speed because the control surfaces become ineffective in this condition for control force depends on the forward speed [4–7]. These disadvantages greatly limit the application of AUVs. An important and effective approach to overcome this restriction is to use a vectored thruster, which can use the control force produced by the vectored thrust for controlling the AUVs [8–10]. To perform underwater tasks, it is necessary to design a control system for the vectored thruster AUV to perform precise trajectory tracking control. However, the AUVs are highly complex

and coupled nonlinear systems with all kinds of unknown, structured, and unstructured uncertainties corresponding to underwater environment [11, 12]; therefore, it is difficult to establish a precise control model for the designed AUV. Therefore, the control of AUVs has attracted considerable attention in recent years, which needs to satisfy the demand for an accurate trajectory tracking control for AUV against the variations in hydrodynamic coefficients and external forces from ocean currents [13, 14].

Over a few decades, various control methods have been proposed for AUVs to solve vehicle control issues while considering the aforementioned difficulties. The representative methods for AUV control such as proportional integral derivate have been developed for the low-level AUV control. In the early work, Jalving [15] designed a PID controller for AUV with steering, diving, and speed subsystems. A controller based on PID was proposed for the position and attitude tracking of the AUVs, and the authors also proved the global convergence of the proposed algorithm [16]. Herman [17] proposed a decoupled PD set-point controller for underwater vehicles on the basis of previous study [18, 19]. To solve the problem of windup due to uncertain dynamics together with the actuator saturation, many researchers have devoted themselves to this aspect and have achieved many results in theory and application [20–23]. Furthermore, considering the hydrodynamics of underwater vehicle which is highly nonlinear and the uncertainty in the model, some research about the adaptive controller is proposed for controlling underwater vehicles to track the desired trajectory [24–26]. Besides, many researchers have deployed some research on the controller for AUVs combining with other algorithms and have achieved some progress [27–33].

In addition, other different techniques have also been used for controlling AUVs to accomplish tasks, such as sliding model control, backstepping, and model predictive control. Sliding model control (SMC) is one of the most efficient and robust methods to deal with some nonlinear uncertainties and external disturbance [34, 35]. In the earlier literature, Young et al. proposed a controller based on SMC for robust trajectory tracking control of the AUVs [36] and carried out related experiments using adaptive SMC [37]. Cristi et al. designed a decoupled controller using adaptive SMC for AUV diving control [38]. Healey and Lienard proposed multivariable sliding mode control for autonomous diving and steering of unmanned underwater vehicles [39]. Furthermore, in order to improve the performance of SMC, researchers designed a controller based on a higher order sliding model control for diving control [40]. An adaptive robust control system was proposed by employing fuzzy logic, backstepping, and sliding mode control theory [41]. Zain and Harun proposed a nonlinear control method for stabilizing all attitudes and positions of an underactuated X4-AUV with four thrusters and six degrees-of-freedom (DOFs) according to the Lyapunov stability theory and using backstepping control method [42]. In this work, Steenson et al. developed a depth and pitch controller using the model predictive control method to manoeuvre the vehicle within the constraints of the AUV actuators [43]. Shen et al. proposed a nonlinear model predictive control scheme to control the depth of AUV and to have a friendly interaction with the dynamic path planning method [44]. These research studies evidence a growing need for designing a better controller for underwater vehicles to complete a variety of tasks in different complex unknown environmental conditions.

However, the traditional nonlinear controller is still significantly dependent on the model, and the performance of the model-based controller will seriously degrade due to a lack of precise knowledge about nonlinearities, uncertainties, and unknown disturbances. Therefore, it is obviously difficult to obtain an accurate dynamic model; the conventional control method is hard to ensure the accurate and automatic control of the AUV [1]. In order to develop real autonomous systems, researchers have turned their attention to artificial intelligence methods, such as using artificial neural networks in AUV control formulations [45]. Fujii developed a self-organizing neural-net-controller system as an adaptive motion control system, which can generate autonomously an appropriate controller according to some evaluations of motion of the vehicle [46]. A neural network adaptive controller for diving control of an AUV is presented in this paper [47]. Based on neural network, Shojaei addressed a control formation for underactuated AUVs with limited torque input under environmental disturbances [48]. Many other researchers also carried out a lot of research and also achieved fruitful results from different perspectives [49–51].

In the current study, Zhang et al. proposed an approach-angle-based three-dimensional path-following control scheme for underactuated AUV which experiences unknown actuator saturation and environmental disturbance [52]. This paper investigates three-dimensional target tracking control problem of underactuated AUVs by using coordinate transformation and multilayer neural networks [53]. The authors address the problem of reachable set estimation for continuous-time Takagi–Sugeno (T-S) fuzzy systems subject to unknown output delays, and a new controller design method is also discussed based on the reachable set concept for AUVs [54]. In this paper, neural network- (NN-) based adaptive trajectory tracking control scheme has been designed for underactuated AUVs which are subjected to unknown asymmetrical actuator saturation and unknown dynamics [55]. This paper investigates neural network estimators-based fault-tolerant tracking control problem for AUV with rudder faults and ocean current disturbance [56]. A robust neural network approximation-based output-feedback tracking controller is proposed for autonomous underwater vehicles (AUVs) in six degrees-of-freedom in this paper [57].

Reinforcement learning (RL) is another important method of artificial intelligence for designing control systems [58]. RL algorithms are able to learn behavior through trial-and-error interactions with a dynamic environment [59]. RL can learn a control policy directly without requiring a model [60]. Chris Gaskett and Wettergreen developed an autonomous underwater vehicle for exploration and inspection with on-board intelligent control, which can learn

to control its thrusters in response to the command and sensor inputs [61]. A hybrid coordination method is proposed for behavior-based control architectures, where the behaviors are learned online by reinforcement learning [62]. Carreras et al. presented a hybrid behavior-based scheme using reinforcement learning for high-level control of autonomous underwater vehicles [63]. In the paper of El-Fakdi, a high-level RL control system using a Direct Policy Search method is proposed for solving the action selection problem of an autonomous robot in cable tracking task [64]. Fjerdingen analyzed the application of several reinforcement learning techniques for continuous state and action spaces to pipeline following for an AUV [65]. Wu proposed an RL algorithm that learns a state-feedback controller from sampled trajectories of the AUV for tracking the desired depth trajectories. In the work of Frost et al. a behavior-based architecture using a natural actor-critic RL algorithm is presented for forming the foundation of the system with an extra layer, which uses experience to learn a policy for modulating the behaviors' weights [66]. In this article, El-Fakdi and Carreras proposed a control system based on actor-critic algorithm for solving the action selection problem of an autonomous robot in a cable tracking task [67]. On the other hand, the performance and application scope of RL algorithm is increasing rapidly, due to the development of deep learning [68]. Based on deep reinforcement learning (DRL), a lot of research studies have been carried out and achieved fruitful accomplishments, such as autonomous vehicle control [69–71]. Yu et al. proposed an underwater motion control system through a modified deep deterministic policy gradient, and it is proved that this algorithm is more accurate than traditional PID control in solving the trajectory tracking of AUV [72]. Two reinforcement learning schemes, which include deep deterministic policy gradient and deep Q network, were investigated to control the docking of an AUV onto a fixed platform in a simulation environment [73]. In the works of Carlucho et al. a deep RL framework based on an actor-critic goal-oriented deep RL architecture is developed for controlling the AUV's thrusters directly using the sensory information as input parameter, and experiments on a real AUV demonstrate the applicability of the proposed deep RL approach [74].

Based on research and literature review, we proposed a deep RL based on deep deterministic policy gradient algorithm for low-level velocity control of the vectored thruster AUV. In the proposed control scheme, the input parameters are the data that can be measured by the on-board sensors directly, and the outputs of the designed controller are set to the actions of the vectored thruster. Moreover, a reward function is developed for deep RL controller considering different factors which actually affect the accuracy of AUV navigation control. To confirm the algorithm's effectiveness, a series of simulations are carried out in the designed simulation environment, which is a method to save time and improve efficiency. The simulation results demonstrated the feasibility of the proposed deep RL applied on an AUV navigation control. Our work based on reinforcement learning algorithm provides an optional method for AUV control problems to deal with improving technology requirements and complicated application environments. This method based reinforcement learning significantly improves the control performance of AUVs. Furthermore, the simulation results also open up a vast range of prospects for the application of the deep RL method in complex engineering system.

The organization of this paper is as follows. In Section 2, we have briefly introduced the related configuration of a vectored thruster AUV, investigated the kinematic and dynamic of the AUV, and designed a control system based on PID algorithm. In Section 3, we introduce the related knowledge of deep reinforcement learning. In Section 4, we develop our proposed controller based on deep RL. In Section 5, we carry out a series of simulations to confirm the algorithm's effectiveness. In Section 6, we conclude this paper and look to the future work.

2. The Vectored Thruster AUV Model and Control Problems

The tilt angles of the ducted propeller in the AUVs' yaw and pitch plane are limited in $\pm 15^{\circ}$. The vectored thruster AUVs have the ability to perform missions at zero or low forward speeds for the control force provided by vectored thruster. To achieve reliable and accurate control of the AUVs, there are high demands on the autonomous control system design. And the kinematic and dynamic of the AUV are fundamental to design a control system.

The study of the AUVs about modeling and control problems involves many theories and methods of statics and dynamics. Generally, the motion study of AUVs can be divided into two major parts: one is the kinematics analysis model and the other is the dynamics analysis model of AUV. The kinematics analysis model of AUVs is used to complete the study of position and orientation of motion, while the dynamics model deals with the motion of the vehicle caused by the forces and moments.

Generally, the motions of AUVs in underwater environment are related to six degrees-of-freedoms (6 DOFs). For analyzing the motion of the vectored thruster AUV in 6 DOFs concisely and efficiently, it is convenient to define two commonly used frames, namely, earth-fixed frame and body-fixed frame, as shown in Figure 1. These DOFs usually refer to the motions about the three coordinate axes of AUVs, including surge, sway, heave, roll, pitch, and yaw, respectively. And the motions mentioned above determine the position and orientation of AUVs in the ocean corresponding to the six DOFs.

In this study, the earth-fixed frame is a global-coordinate system that can be considered to be inertial and fixed to its origin. The body-fixed frame is a moving frame fixed to the AUV, whose origin is coinciding with the center of mass of the AUVs. To make it convenient in investigating the vectored thruster AUV, the standard notations used to describe the motion of AUV are defined in Table 1.

The general kinematics transformation of AUV between the two independent coordinate systems can be represented as follows:

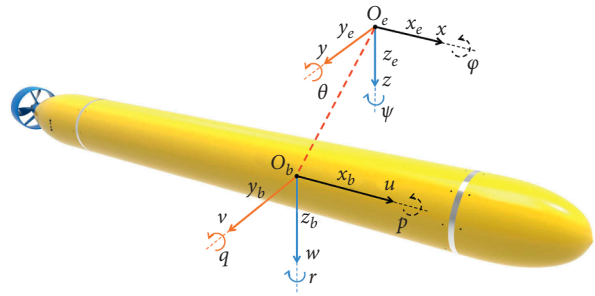

Figure 1: The earth-fixed and body-fixed frame of an AUV.

Table 1: The notation of SNAME for AUVs.

DOF	Forces and moments	Positions and Euler angles	Linear and angular velocities
Surge	X	x	u
Sway	Y	y	v
Heave	Z	z	w
Roll	K	ϕ	p
Pitch	M	θ	q
Yaw	N	ψ	r

$$\dot{\boldsymbol{\eta}} = \mathbf{J}(\boldsymbol{\eta})\mathbf{v}, \tag{1}$$

where $\boldsymbol{\eta} = [\,x\ \ y\ \ z\ \ \phi\ \ \theta\ \ \psi\,]^T$ denotes the vector of position and orientation, $\mathbf{J} \in R^{6\times 6}$ represents the transformation matrices from the body-fixed frame to the earth-fixed frame, and $\mathbf{v} = [\,u\ \ v\ \ w\ \ p\ \ q\ \ r\,]^T$ represents the corresponding vectors of linear and angular velocity.

The kinematics description of the nonlinear equations of the AUV above can be described separately for linear and angular parameters as follows:

$$\begin{aligned} \dot{\boldsymbol{\eta}}_1 &= \mathbf{J}_1(\boldsymbol{\eta}_2)\mathbf{v}_1, \\ \dot{\boldsymbol{\eta}}_2 &= \mathbf{J}_2(\boldsymbol{\eta}_2)\mathbf{v}_2, \end{aligned} \tag{2}$$

where $\boldsymbol{\eta} = [\,\boldsymbol{\eta}_1\ \ \boldsymbol{\eta}_2\,]^T$, $\eta_1 = [\,x\ \ y\ \ z\,]^T$, and $\eta_2 = [\,\phi\ \ \theta\ \ \psi\,]^T$ denote the vectors of position and orientation; $\mathbf{v} = [\,\mathbf{v}_1\ \ \mathbf{v}_2\,]^T$, $\mathbf{v}_1 = [\,u\ \ v\ \ w\,]^T$, and $\mathbf{v}_2 = [\,p\ \ q\ \ r\,]^T$ represent the vectors of linear velocity and angular velocity; and $\mathbf{J}_1(\eta_2), \mathbf{J}_2(\eta_2)$ denote the linear and angular velocity transformation matrix between the body-fixed frame and earth-fixed frame, respectively. $\mathbf{J}_1(\eta_2), \mathbf{J}_2(\eta_2)$ are defined as follows:

$$\begin{aligned} \mathbf{J}_1(\boldsymbol{\eta}_2) &= \begin{bmatrix} c\psi c\theta & -s\psi c\theta + c\psi s\theta s\phi & -s\psi s\theta + c\phi c\psi s\theta \\ s\psi c\theta & c\psi c\theta + s\psi s\theta s\phi & -c\psi s\theta + c\phi c\psi s\theta \\ -s\theta & c\theta c\phi & c\theta c\phi \end{bmatrix}, \\ \mathbf{J}_2(\boldsymbol{\eta}_2) &= \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & \dfrac{s\phi}{c\theta} & \dfrac{c\phi}{c\theta} \end{bmatrix}, \end{aligned} \tag{3}$$

where $s(\cdot)$, $c(\cdot)$, and $t(\cdot)$ denote $\sin(\cdot)$, $\cos(\cdot)$, and $\tan(\cdot)$, respectively.

The dynamic equations of motion for the underwater vehicles are derived from Newton–Euler equation using the principle of virtual work and D'Alembert's principle. The equations of motion of underwater vehicle are established based on the traditional six-DOF model in the earth-fixed frame, and it can be expressed in the following form:

$$\mathbf{M}\dot{\nu} + \mathbf{C}(\mathbf{v})\mathbf{v} + \mathbf{D}(\mathbf{v})\mathbf{v} + \mathbf{g}(\boldsymbol{\eta}) = \boldsymbol{\tau} + \Delta\boldsymbol{\tau}, \tag{4}$$

where $\mathbf{v} \in R^{6\times 1}$, $\dot{\nu} \in R^{6\times 1}$ denote the vectors of velocity and acceleration related to the body-fixed frame. $\mathbf{M} \in R^{6\times 6}$ iss the matrix of inertia of the vehicle, consisting of the rigid body inertia matrix $\mathbf{M}_{RB}$ and the added mass$\mathbf{M}_A$. The terms $\mathbf{M}_{RB}$ and $\mathbf{M}_A$ are listed in (A.1) and (A.2) in supplementary file, and the developed coefficients of the vehicle used in this paper are listed in Appendix Bin supplementary file.

$\mathbf{C}(\mathbf{v}) \in R^{6\times 6}$ represents the Coriolis-centripetal matrix related to the Coriolis forces and the centripetal effects. The Coriolis-centripetal matrix $\mathbf{C}(\mathbf{v})$ also includes the rigid body term $\mathbf{C}_{RB}(\mathbf{v})$ and the added mass term $\mathbf{C}_A(\mathbf{v})$, as defined in the following equation:

$$\mathbf{C}(\mathbf{v}) = \mathbf{C}_{RB}(\mathbf{v}) + \mathbf{C}_A(\mathbf{v}), \tag{5}$$

where $\mathbf{D}(\mathbf{v}) \in R^{6\times 6}$ refers to the hydrodynamic damping matrix of vehicle, which is mainly composed of linear damping matrix $\mathbf{D}$ and nonlinear damping matrix $\mathbf{D}_n(\mathbf{v})$. Hence, the terms of the hydrodynamic damping matrix of the underwater vehicle can be described by the following:

$$\mathbf{D}(\mathbf{v}) = \mathbf{D} + \mathbf{D}_n(\mathbf{v}), \tag{6}$$

where $\mathbf{D}$ denotes the damping matrix due to linear skin friction, $\mathbf{D}_n(\mathbf{v})$ represents the nonlinear damping matrix mainly generated from potential damping, wave drift damping, damping due to vortex shedding, and lifting

forces. The calculating formulas of terms $\mathbf{D}$ and $\mathbf{D}_n(\mathbf{v})$ in equation (5) are given as (A.3) and (A.4)in supplementary file.

$\mathbf{g}(\eta)$ is the vector restoring forces and moments related by gravity and buoyancy of the vehicle.

$\tau \in R^{6\times1}$ defines the resultant vector of applied forces and moments on the vehicle in the body-fixed frame. $\Delta\tau \in R^{6\times1}$ represents the vector of forces and moments produced by environment disturbances including ocean currents and waves.

In general, the thrust T_p is generated by a propeller mounted at the stern of AUV, and the direction of thrust is collinear with the cylindrical axis of the vehicle's hull. Hence, the applied forces and moments τ on the vehicle can be expressed as

$$\boldsymbol{\tau} = \begin{bmatrix} T_p & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T. \tag{7}$$

On the other hand, the direction of thrust T_p provided by the designed vectored thruster can be adjusted according to the need of control AUV. The resultant vector τ which represents the applied forces and moments acting on AUVs can be expressed as follows:

$$\boldsymbol{\tau} = \begin{bmatrix} \mathbf{F} & \mathbf{M} \end{bmatrix}^T = \begin{bmatrix} X & Y & Z & K & M & N \end{bmatrix}^T. \tag{8}$$

By comparison with the conventional AUV, there is difference in controlling because the direction of thrust is controlled by the thrust-vectoring mechanism. The deflection angle of the duct is a combination of the rudder angle α and the elevator angle β in the body-fixed frame, as presented in Figure 2.

The vector of thrust applied on the AUV along with axis in the body-fixed frame is defined as

$$\mathbf{F} = \begin{bmatrix} F_x & F_y & F_z \end{bmatrix}^T. \tag{9}$$

Besides, since the thrust T_p of this vectored thruster AUV is provided by the propeller, according to the theory of standard propeller, the thrust can be described as

$$T_p = K_T \rho n_p^2 D^4, \tag{10}$$

where ρrepresents the density of water and K_T, n_p, and D denote the thrust coefficient, rotation speed, and diameter of the propeller, respectively. Referring to the definition of deflection angles of the duct α and β in Figure 2 and equation (10), the vector of thrust $\mathbf{F}$ can be calculated by

$$\mathbf{F} = T_p \begin{bmatrix} \cos\beta\cos\alpha \\ \cos\beta\sin\alpha \\ -\sin\beta \end{bmatrix}. \tag{11}$$

In order to study the relations between the vector of thrust $\mathbf{F}$ and deflection angle of the duct α and β, the unit vectorδ is defined as follows:

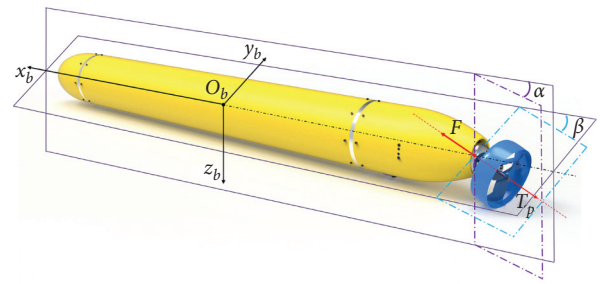

Figure 2: Thrust decomposition of the vectored thruster AUV.

$$\boldsymbol{\delta} = \begin{bmatrix} \delta_x \\ \delta_y \\ \delta_z \end{bmatrix} = \begin{bmatrix} \cos\beta\cos\alpha \\ \cos\beta\sin\alpha \\ -\sin\beta \end{bmatrix}. \tag{12}$$

Figure 3 shows the 3D graph of the factor δ with the tilt angles $-(\pi/12) \le a \le (\pi/12)$ and$-(\pi/12) \le \beta \le (\pi/12)$. The linear motion of the vehicle is controlled by the vector of thrust $\mathbf{F}$ according to adjusting the thrust T_p and deflection angle of the duct α and β. Due to the particularity of the vectored thruster AUV, the vehicle's motions of pitch and yaw are controlled by moments produced by components of thrust vector $\mathbf{F}$. The moments $\mathbf{M}$ acting on the AUV are generated when the thrust T_p is not coincident with the axis of the vehicle's hull. Referring to Figure 2, the moments $\mathbf{M}$ due to the thrust $\mathbf{F}$ acting on the center of mass can be expressed as

$$\mathbf{M} = \mathbf{r}_p \times \mathbf{F} = \mathbf{r}_p \times T_p \cdot \boldsymbol{\delta}, \tag{13}$$

where $\mathbf{r}_p = \begin{bmatrix} x_p & y_p & z_p \end{bmatrix}$ denotes the position vector from the point of action of the thrust $\mathbf{F}$ to the vehicle center of gravity. The motions of pitch and yaw of this AUV are controlled by moment vector$\mathbf{M}$. Because the moment $\mathbf{M}$ is determined by thrust T_p and deflection angle of the duct α and β and value of thrust T_p only depends on the rotation speed of the propellern_p, the value of moment $\mathbf{M}$ is independent of the forward speed and attitude of AUV.

Due to having 6 motional DOFs, the dynamics model of AUVs with highly nonlinear characteristics is a big challenge to design a controller. In order to realize underwater vehicle function designed completely, control system plays an important role in the process of design of AUV [75]. In general, the overall control process of AUV can be represented as shown in Figure 1. In the control process, the controller design is essential for manipulating the AUV. In our vectored thruster AUV, the control system consists of three control loops that represent its surge, pitch, and yaw motions. The inputs of the AUV controller are velocity errors, and the outputs from the controller are the control actions provided by vectored thruster. In the surge control loop, the input to the controller is the linear velocity u and the output is the thrust T_p referring to Figure 1. Similarly, the input and output of the pitch controller are angular velocity q and the elevatorβ; the input and output of the yaw controller are angular velocity r and rudder angle α. In order to meet the demand of real applications, controllers for AUVs are

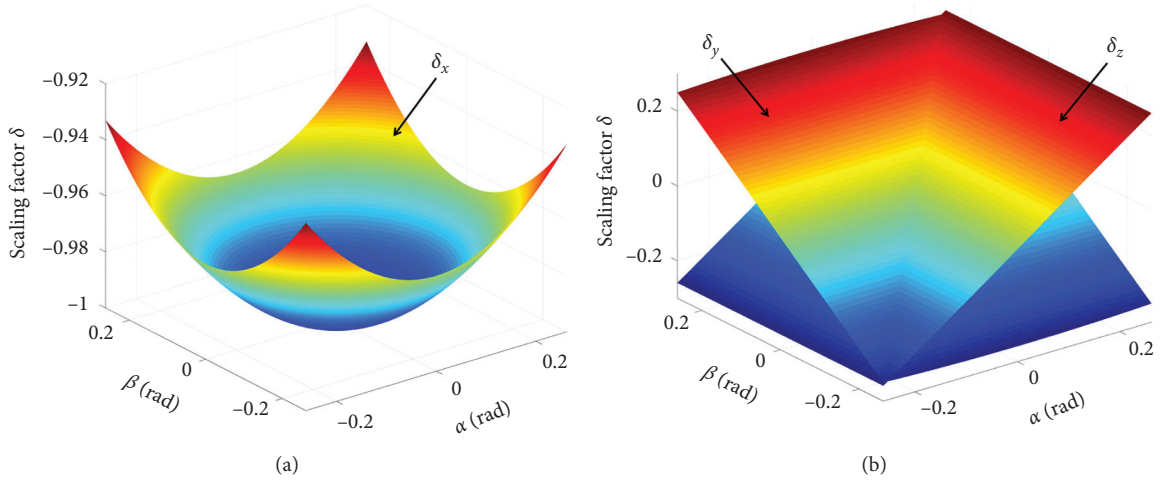

Figure 3: Scaling factors varying with tilt angle α and β.

usually designed based on the proportional-integral-derivative (PID) algorithm. PID algorithm can be expressed as

$$U(t) = k_p e(t) + k_i \int_0^t e(t)\mathrm{d}t + k_d \frac{\mathrm{d}e(t)}{\mathrm{d}t}, \tag{14}$$

where k_p, k_i, and k_d are the proportional factor, integral factor, and differential factor, respectively, and $e(t)$ is defined as an error value from the difference between the desired set point and a measured process variable.

Based on the control process in Figure 4 and PID algorithm in equation (10), a controller is designed aiming at the vectored thruster AUV. To verify control performance of this system, a series of simulations on the performance are carried out, according to a series of analysis and research on AUV mentioned above. Before simulation analysis, the reference velocity is defined by $\mathbf{S_t^r} = (v_x^r, \omega_y^r \omega_z^r)$, and the output thrust for AUV is$0 \le T_p \le 10\,\mathrm{N}$, $\alpha, \beta \in [-(\pi/12), (\pi/12)]$ ($[-15°, 15°]$). When the reference velocities $v_x^r = 1\,(\mathrm{m/s}), \omega_y^r = 0, \omega_z^r = 0$, the simulation result is obtained as shown in Figure 5.

The simulation results in Figure 5 show that the proposed controller based on PID algorithm is practicable and effectively applied in this reference velocity. When the reference velocities $v_x^r = 1\,(\mathrm{m/s}), \omega_y^r = 0.1\,(\mathrm{rad/s}), \omega_z^r = 0$, the simulation result is obtained as shown in Figure 6.

As shown in Figure 6, the reference linear velocity $v_x = v_x^r$ in a short time, but the angular velocity ω_y is different from the reference velocityω_y^r. On the basis of analyses, it is shown that the thrust T_p is inadequate to meet the need of achieving the reference angular velocity. As described earlier, the thrust T_p is determined by the surge controller, and hence the thrust T_p does not increase although the angular velocity ω_y did not reach the referenceω_y^r.

When a new requirement is introduced, such as the reference velocities $\omega_y^r = 0.1\,(\mathrm{rds/s}), \omega_z^r = 0$, with neglect of the velocity v_x^r setting, this designed AUV controller becomes difficult to implement. In order to solve this problem, the thrust T_p for the AUV is set to a high value $T_{p-\mathrm{set}}$ in advance. When$T_{p-\mathrm{set}} = 6$, the simulation results are shown in Figure 7.

According to the above, the simulation results and analysis show the inadequacies of the designed AUV controller based on PID algorithm. In order to improve the performance and reduce energy consumption, it is essential to find a new method to design the AUV controller.

3. Deep Reinforcement Learning Control for AUV

3.1. Reinforcement Learning Statement. Reinforcement learning (RL) is a part of machine learning that focuses on studying how an agent optimizes its behavior for a task by interacting with the environment. Then, the environment produces new stats to respond to the executed action in some state. At the same time, the agent receives a new reward value from the environment, which can be seen as the indexes to evaluate the advantages and disadvantages of action. A series of data are generated by the agent and the environment through continuing loop iterations. The basic principle of reinforcement learning is presented in Figure 8.

The environment for agent training in RL can be described as a Markov Decision Process (MDP), where the environment is assumed as fully observable. An MDP can be defined as a 5-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}_{sa}, \gamma\mathcal{R})$, where $\mathcal{S} \in \mathbb{R}^d$ is the d-dimensional state space, $\mathcal{A}$ defines the action space, $\mathcal{P}_{sa}$ is the probability of transition to state s' by taking an action a in states, $\gamma \in (0, 1]$ denotes the discount factor for future rewards, and $\mathcal{R}$: $\mathcal{S} \times \mathcal{A} \longrightarrow \mathbb{R}$ is the function expressing the reward for taking action in a particular state. The policy function π represents a mapping from states to actions and denotes a mechanism for choosing action **a** in current state **s**.

The goal of the agent is to maximize the total amount of reward it receives [76, 77]. When a strategy π is given, the discounted sum of immediate rewards is defined as returnR_t:

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}. \tag{15}$$

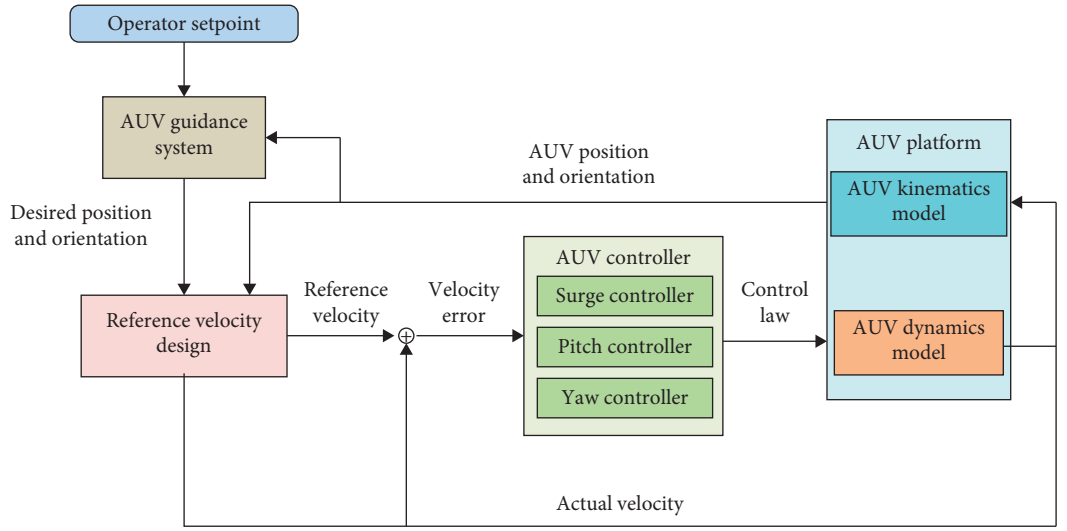

Figure 4: A block diagram of the AUV's control process.

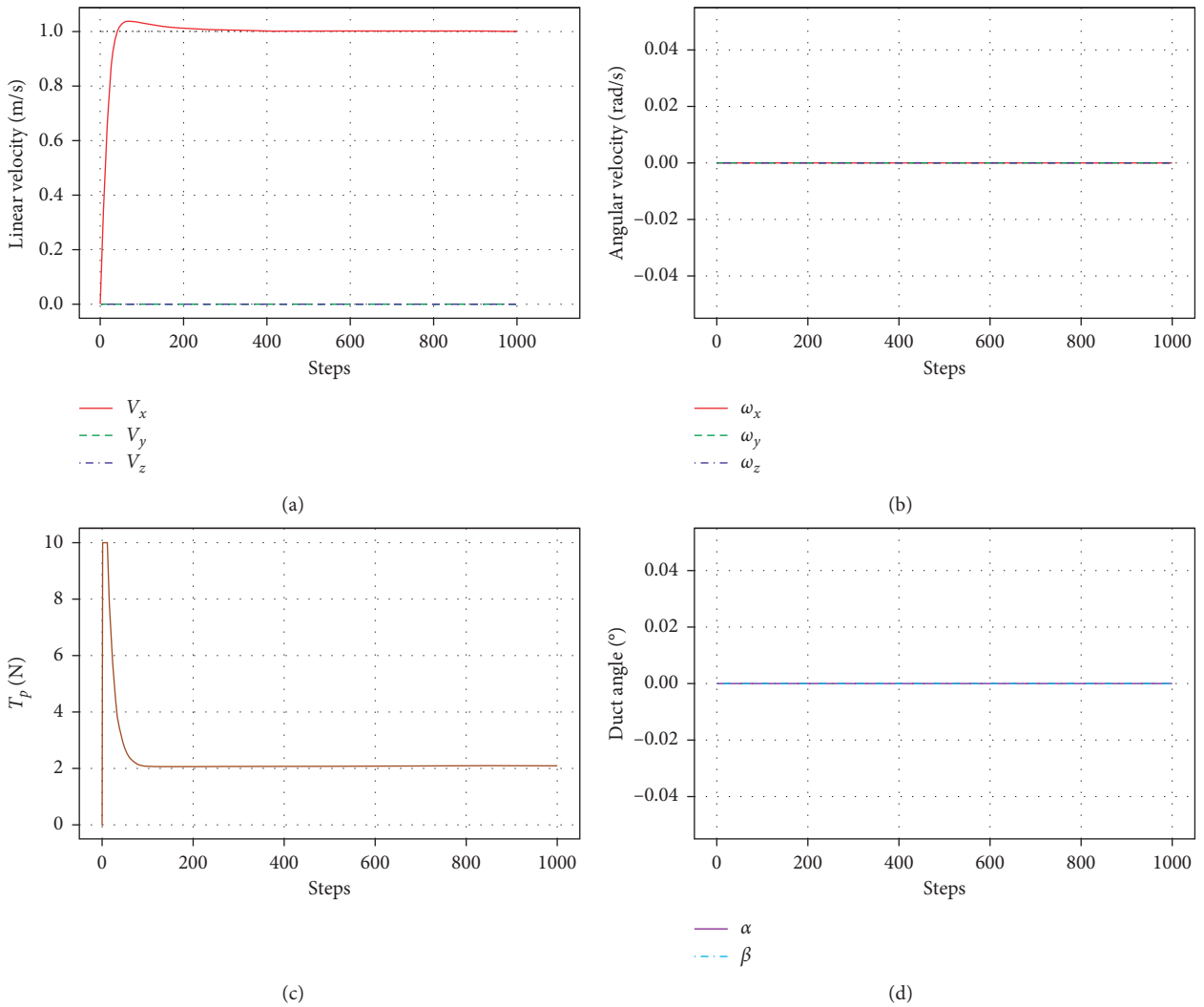

Figure 5: Simulation results of the reference velocity $v_x^r = 1\,(\mathrm{m/s})$, $\omega_y^r = 0$, $\omega_z^r = 0$. (a) Linear velocities. (b) Angular velocities. (c) Thrust for AUV. (d) Deflection angles of the duct.

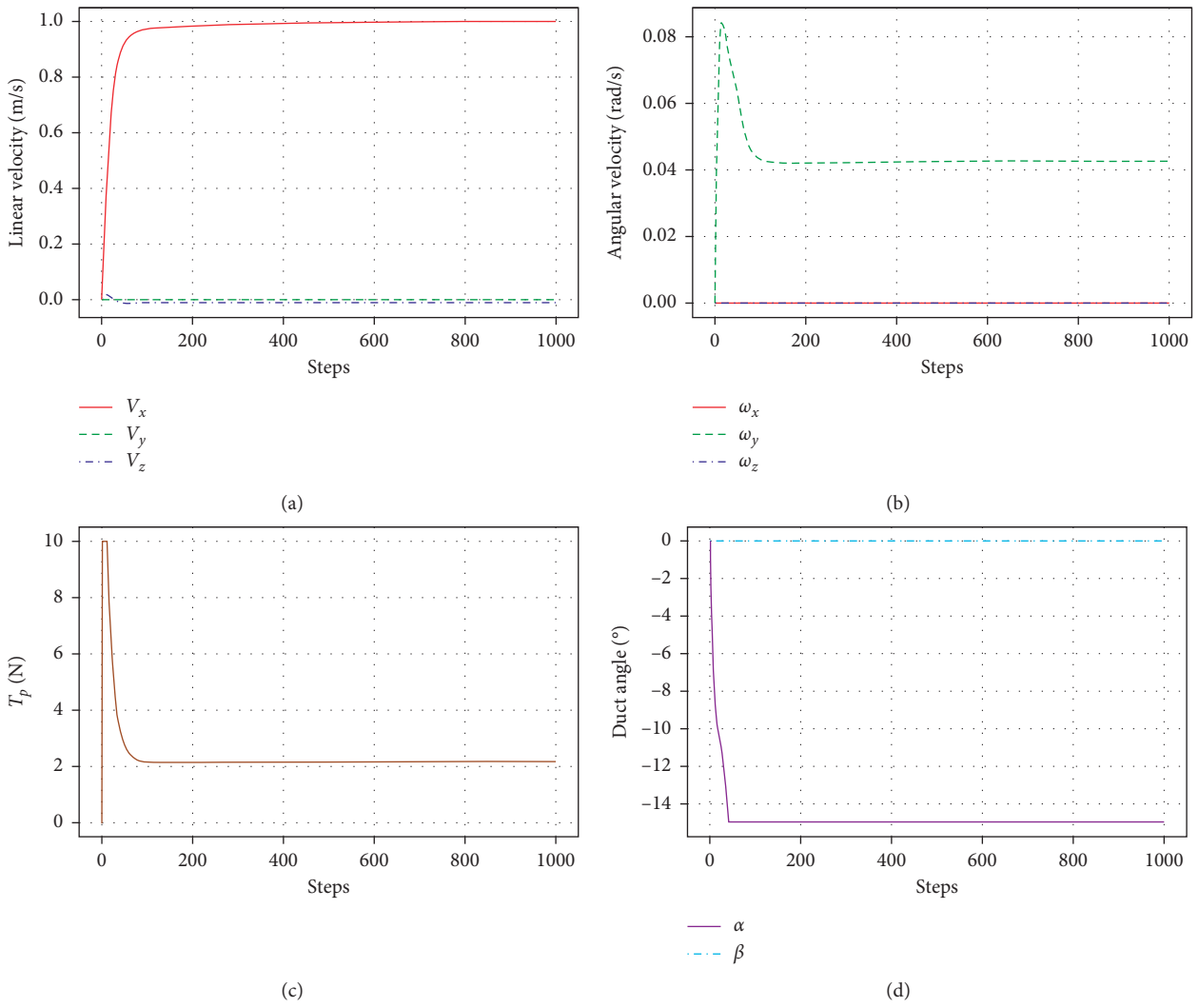

FIGURE 6: Simulation results of the reference velocity $v_x^r = 1\,(\mathrm{m/s}), \omega_y^r = 0.1\,(\mathrm{rad/s}), \omega_z^r = 0$. (a) Linear velocities. (b) Angular velocities. (c) Thrust for AUV. (d) Deflection angles of the duct.

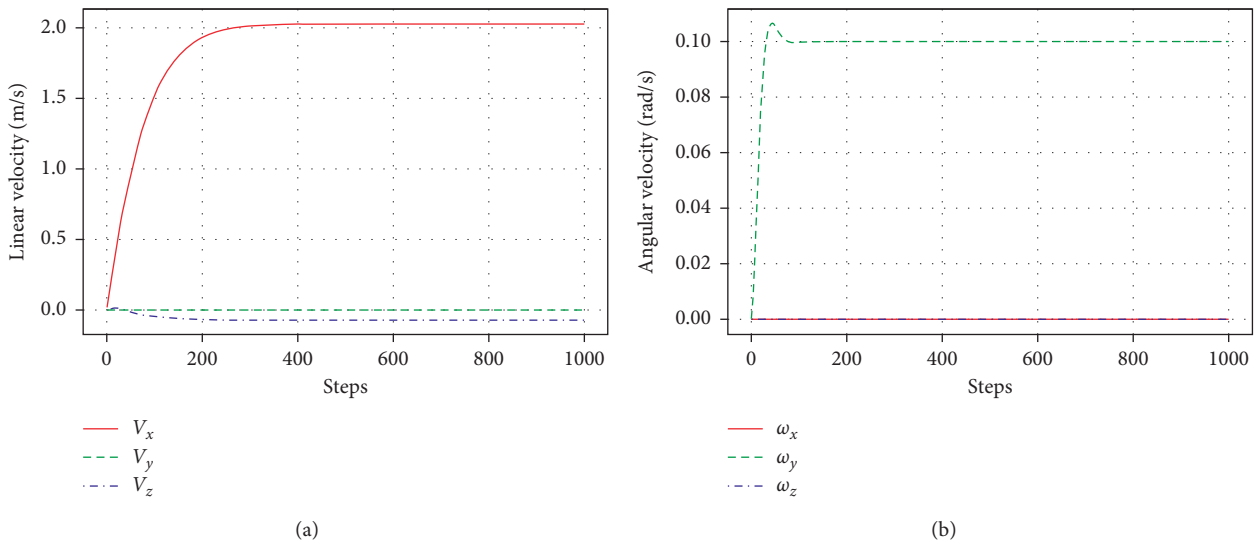

FIGURE 7: Continued.

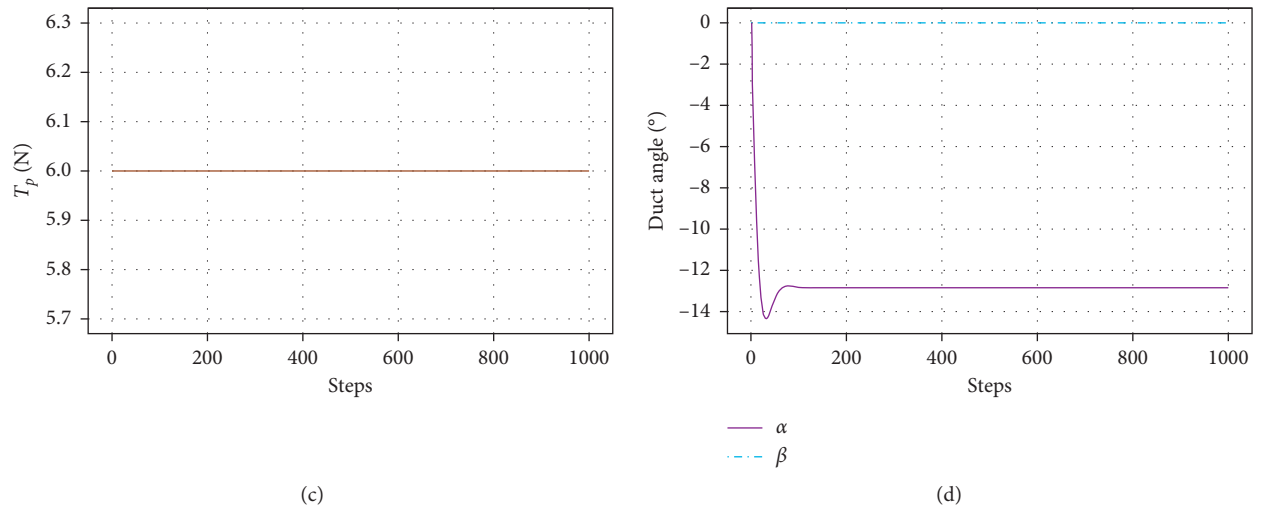

Figure 7: Simulation results of the reference velocity $T_{p-\text{set}} = 6, \omega_y^r = 0.1\,(\text{rds/s}), \omega_z^r = 0$. (a) Linear velocities. (b) Angular velocities. (c) Thrust for AUV. (d) Deflection angles of the duct.

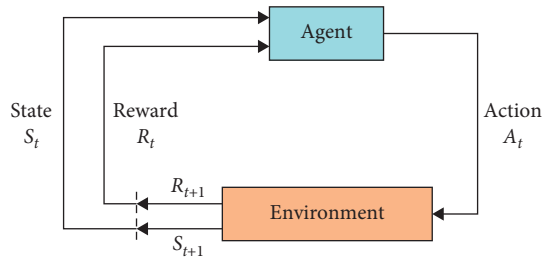

Figure 8: The overview of reinforcement learning process.

The purpose of the reinforcement learning method is to find the optimal policy π^*, which maximizes the return R_t by following the policy. The optimal policy π^* satisfies the function as

$$J_\pi^* = \max_\pi J_\pi = \max_\pi E_\pi\{R_t \mid \mathbf{s}_t = \mathbf{s}\}, \tag{16}$$

where the performance objective $J_\pi = E[R_t|\pi]$ denotes the expected total reward under the policy π and γ is the discount factor.

The state-value function is defined as the expected value of cumulative discounted rewards from the state $\mathbf{s}$ corresponding to the policy.

$$V_\pi(\mathbf{s}) = E_\pi[R_t \mid s_t = s] = E_\pi\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid \mathbf{s}_t = \mathbf{s}\right]. \tag{17}$$

Similar to the state-value function, the action-value function, also known as the Q function, can be defined as

$$Q_\pi(\mathbf{s}, \mathbf{a}) = E_\pi[R_t \mid s_t = s] = E_\pi\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid \mathbf{s}_t = \mathbf{s}, \mathbf{a}_t = \mathbf{a}\right]. \tag{18}$$

The state-value function $V^\pi(\mathbf{s})$ and the action-value function $Q^\pi(\mathbf{s}, \mathbf{a})$ satisfy the Bellman equation.

$$\begin{aligned} V_\pi(\mathbf{s}) &= E_\pi[r_t + \gamma V_\pi(\mathbf{s}_{t+1}) | \mathbf{s}_t = \mathbf{s}], \\ Q_\pi(\mathbf{s}, \mathbf{a}) &= E_\pi[r_t + \gamma Q_\pi(\mathbf{s}_{t+1}, \mathbf{a}_{t+1}) | \mathbf{s}_t = \mathbf{s}, \mathbf{a}_t = \mathbf{a}]. \end{aligned} \tag{19}$$

When the agent utilizes strategy optimal policy π^*, the optimal state-value function $V^*(\mathbf{s})$ and the action-value function $Q^\pi(\mathbf{s}, \mathbf{a})$ achieve the highest return. The optimal functions satisfy the following Bellman equation:

$$\begin{aligned} V^*(\mathbf{s}) &= \max_a E_\pi[r_t + \gamma V^*(\mathbf{s}_{t+1}) | \mathbf{s}_t = \mathbf{s}, \mathbf{a}_t = \mathbf{a}], \\ Q_\pi^*(\mathbf{s}, \mathbf{a}) &= E_\pi\left[r_t + \gamma \max_{a_{t+1}} [Q_\pi(\mathbf{s}_{t+1}, \mathbf{a}_{t+1})]\right] \end{aligned} \tag{20}$$

The purpose of RL problems is to learn an optimal policy π^*. A greedy policy π is derived from $Q(\mathbf{s}, \mathbf{a})$ by taking the state and action to get the highest return. Once $Q_\pi^*$is obtained by interactions, the optimal policy π^* can be obtained directly by

$$\pi^*(\mathbf{s}) = \text{aramax}_a Q^*(\mathbf{s}, \mathbf{a}). \tag{21}$$

3.2. Reinforcement Learning in Continuous Domain. Existing RL algorithms mainly consist of value-based and policy-based methods. The first proposed value-based method is Q-learning, which has become one of the most popular and widely used RL algorithms. In the use process, Q-learning needs to calculate the Q-value of each state and action and store it in a table. It is precise because of looking up the table in each iterative calculation, so this value-based algorithm is suitable for those applications where the space of state and action are discrete and the dimension is not too high. In order to resolve the problem about the spaces of state and action being too large, the function approximation to estimate the value function is proposed. Along with the deepening of research, deep neural networks are used to develop a novel artificial agent, which is named deep Q-network (DQN), which can learn successful policies from high-dimensional state [78]. Due to the use of deep neural

networks, this kind of value-based algorithm has been successfully applied to all sorts of games and achieved good results. Along with depth research into the RL method theory and the extensive application of DQN, it is natural for the emergence of various varieties, which are Double DQN, Dueling DQN, and Rainbow [79].

However, while it could resolve problems with high-dimensional state spaces, value-based methods can only tackle the discrete actions applications but fail in continuous action space. This kind of RL algorithms cannot be applied to the continuous domain directly because it depends on looking up the action that maximizes the action-value function, which needs to compete the process of iterative optimization at every step. Besides, if the rough discretization of state action is made, the results will become unacceptable; if the discretization is made so thin, then the results will be difficult to solve. Hence, it may be impracticable when applying this value-based method to a continuous control domain, such as our control of vectored thruster AUV, while another important RL algorithm, named Policy Gradient (PG) reinforcement learning method, has a wide range of applications in the areas of continuous behavior control. PG methods perform gradient ascent on the policy objective function $J(\theta)$; with respect to the parameters θ of policyπ, the policy objective function $J(\theta)$ can be defined as

$$\begin{aligned} J(\theta) &= \int_{\mathcal{S}} \rho^{\pi}(s) \int_{\mathcal{A}} \pi_{\theta}(s,a) r(s,a) \mathrm{d}a \, \mathrm{d}s, \\ &= E_{s\sim\rho^{\pi}, a\sim\pi_{\theta}}[r(s,a)], \end{aligned} \tag{22}$$

where π_θ is a stochastic policy and $\rho^{\pi}(s)$ is the state distribution. The basic idea behind the PG methods is to adjust the parameters θ of the policy in the direction of the performance gradient $\nabla_\theta J(\theta)$. Hence, the corresponding gradient theorem of the policy objective function $J(\theta)$ is

$$\begin{aligned} \nabla_{\theta} J(\theta) &= \int_{\mathcal{S}} \rho^{\pi}(s) \int_{\mathcal{A}} \nabla_{\theta} \pi_{\theta}(a \mid s) Q^{\pi}(s,a) \mathrm{d}a \, \mathrm{d}s, \\ &= E_{s\sim\rho^{\pi}, a\sim\pi_{\theta}}[\nabla_{\theta} \log \pi_{\theta}(a \mid s) Q^{\pi}(s,a)]. \end{aligned} \tag{23}$$

Then, the policy-based methods update the parameter θ as follows:

$$\theta^{t+1} = \theta^{t} + \alpha \widehat{\nabla_{\theta} J(\theta)}, \tag{24}$$

where α is the learning rate and $\widehat{\nabla_\theta J(\theta)}$ denotes the stochastic expectation approximations of the gradient of the objective performance.

This equation above shows that the gradient is an exception of possible states and actions. Rather than approximating a value function, the PG methods approximate a stochastic policy using an independent function approximator with its own parameters that maximize the future expected reward. The main advantage of the PG method against value-based function is using an approximator to represent the policy directly. In the process of PG learning, it should consider the probability distribution of the states and actions simultaneously. Hence, the PG method integrates over both state and action spaces during the training process. There can be no doubt that it consumes a large amount of computing resources for the high-dimensional state and action spaces [80]. To solve this drawback, the deterministic policy gradient algorithms for reinforcement learning are presented [81]. Because the map from state spaces to action spaces is fixed in deterministic policy gradient, it is not needed to integrate over all action spaces. Consequently, the deterministic policy gradient needs much fewer samples to train compared with the stochastic policy gradient. This meant that the deterministic policy gradient can be estimated much more efficiently than the stochastic version. With a deterministic policy μ_θ: $\mathcal{S} \longrightarrow \mathcal{A}$ with a parameterθ and a discounted state distribution $\rho^{\mu}(s)$, the performance objective as an expectation can be defined as

$$\begin{aligned} J(\mu_{\theta}) &= \int_{\mathcal{S}} \rho^{\mu}(s) (s, \mu_{\theta}(s)) \mathrm{d}s, \\ &= E_{s\sim\rho^{\mu}}[r(s, \mu_{\theta}(s))]. \end{aligned} \tag{25}$$

The gradient for deterministic policy is

$$\begin{aligned} \nabla_{\theta} J(\mu_{\theta}) &= \int_{\mathcal{S}} \rho^{\mu}(s) \nabla_{\theta} \mu_{\theta}(s) \nabla_{a} Q^{\mu}(s,a) = |a = \mu_{\theta}(s) \mathrm{d}s, \\ &= E_{s\sim\rho^{\mu}}[\nabla_{\theta} \mu_{\theta}(s) \nabla_{a} Q^{\mu}(s,a) = |a = \mu_{\theta}(s)]. \end{aligned} \tag{26}$$

In order to explore the environment fully, a stochastic policy is often necessary. To ensure the deterministic policy gradient algorithm's adequate exploration, an off-policy actor-critic learning algorithm is proposed subsequently. The actor-critic algorithms consist of two components in policy gradient, an actor and a critic, respectively. Actor and critic are two different networks and have different policies to realize different functions. The critic estimates the value function which could be the action value (the Q-value) or state value (the V value). The actor updates the policy distribution in the direction suggested by the critic (such as with policy gradients). Actor is a policy network to produce actions by space exploration, while the critic is a value function to evaluate the actions made by the actor [82]. The critic network is updated by temporal-difference learning, and the actor network is updated by policy gradient. The performance objective functions over the state distribution by this behavior policy:

$$J_{\beta}(\theta) = \int_{\mathcal{S}} \rho^{\beta}(s) Q^{\mu}(s, \mu_{\theta}(s)) \mathrm{d}s, \tag{27}$$

where $\rho^{\beta}(s)$ is the stationary distribution of the behavior policy β and $Q^{\mu}$is the action-value function. Then, the off-policy deterministic policy gradient can be presented as

$$\nabla_{\theta} J_{\beta}(\theta) = E_{s\sim\rho^{\beta}}[\nabla_{a} Q^{\mu}(s,a) \nabla_{\theta} \mu_{\theta}(s) | a = \mu_{\theta}(s)]. \tag{28}$$

Given the policy gradient direction, the update process of actor-critic off-policy DPG can be presented as

$$\begin{aligned} \delta_t &= r_t + \gamma Q^{w}(s_{t+1}, \mu_{\theta}(s_t + 1)) - Q^{w}(s_t, a_t), \\ w_{t+1} &= w_t + \alpha \delta_t \nabla_{w} Q^{w}(s_t, a_t), \\ \theta_{t+1} &= \theta_t + \alpha_{\theta} \delta_t \nabla_{\theta} \mu_{\theta}(s_t) \nabla_{w} Q^{w}(s_t, a_t) | a = \mu_{\theta}(s). \end{aligned} \tag{29}$$

The advantage of the actor-critic algorithm is the ability to implement the single-step update, which makes make it more efficient. The performance of actor-critic algorithm is decided by critic's value judgment; nevertheless, it is very difficult to realize convergence, particularly when the actor also needs to upgrade its parameters. To overcome those problems mentioned above, Deep Deterministic Policy Gradient (DDPG) has been presented.

3.3. DDPG in Continuous Domain. Deep reinforcement learning is composed of deep neural network and reinforcement learning. The algorithm structure makes it directly learn control policies from high-dimensional state-action spaces. Due to the excellent performance over a wide range of applications, deep neural network (DNN), which is an artificial neural network (ANN) with several layers between the input and output layers, has arisen and become a very popular research topic in machine learning recently. Thanks to the huge success in a variety of fields such as medical imaging analysis, artificial neural networks have attracted great interest in deep learning. With the development of these neural network structures, it has been used in different areas such as solving engineering control problems.

The basic unit of neural networks is neuron, which is a mathematical function that models the functioning of a biological neuron within an artificial neural network. Because it consists of a large number of layers and neurons in each layer, DNN can always find the right mathematical operation to convert the inputs to outputs, whether linear or nonlinear relationship. In the neural networks, it can be called fully connected layers when each neuron in a layer is connected to all neurons in the next layer. In the network form, each connection contains parameters weight w and bias b and applies the activation function σ to the weighted sum $z = \sigma(b + \sum_i w_i \quad x_i)$, where x is the input vector. The learning process of the neural networks is the process that continuously regulates relative parameters, which mainly include the weights and biases of the network, to reduce the errors generated by the real and prediction results. Combining deep neural networks with reinforcement learning algorithms, deep reinforcement learning (DRL) can be created to resolve previously unresolved problems [83]. In DRL, artificial neural networks can be used as universal function approximator to estimate value function or policy-based methods.

4. Control Based on DDPG for Vectored Thruster AUV

Deep Deterministic Policy Gradient (DDPG) is a model-free off-policy actor-critic algorithm using deep function approximator that can be used to solve the problems of high-dimensional, continuous motion spaces. Because it is proposed based on the concept of DQN, DDPG also uses the deep neural networks as function approximator, which makes it feasible in complex action-space applications [84]. DDPG contains two independent networks, which are actor network and critic network, respectively. With the parameter θ^{μ}, the actor network represents the deterministic policy $a = \mu(s|\theta^{\mu})$, which is used to update the policy that corresponds to the actor in the actor-critic framework. The critic network with parameter θ^{Q} is used to estimate the action-value function $Q(s, a|\theta^{Q})$ of the state-cation pair and calculate the gradient parameters. In order to make it stable and robust, DDPG algorithm adopts experience replay and the target network.

In order to achieve stable learning, DDPG deploys experience replay and target networks like DQN. Experience replays are a key technology behind many of the latest advancements in deep reinforcement learning [85]. The experience replay was applied to avoid such situations where the training samples are not independent and identically distributed. In the training process, samples are generated by sequential explorations in an environment. In order to break the data correlation, experience replay is implemented by sampling transitions (s_t, a_t, r_t, s_{t+1}), which are historical data samples from the environment and stored in the replay buffer. And the replay buffer is continually updated by replacing old samples with new ones when the buffer is full. The actor and critic are trained with minibatches sampled from the reply buffer randomly in the training process. The main effect of experience replay is to overcome the problem of correlated data and nonstationary distribution of empirical data. This random sampling method greatly increases the utilization of samples and improves the stability of the algorithm [85].

In order to optimize the state-value function (critic function) neural network, a loss function based on mean squared error is proposed to carry out the backpropagation. In DDPG, the parameters of the deep neural network for the critic are updated by minimizing the loss function L defined as

$$L = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - Q\left(s_i, a_i|\theta^{Q}\right)\right)^2, \tag{30}$$

where y_i is the target value function generated by the target neural network θ^{Q_t} and can be defined as

$$y_i = r_i + \gamma Q'\left(s_{i+1}, \mu'\left(s_{i+1}|\theta^{\mu'}\right)|\theta^{Q'}\right), \tag{31}$$

Then, the gradient of the loss function L is defined as

$$\nabla_{\theta^{Q}} L = -\frac{2}{N}\sum_{i=1}^{N}\left(y_i - Q\left(s_i, a_i|\theta^{Q}\right)\right)\frac{\partial Q\left(s_i, a_i|\theta^{Q}\right)}{\partial \theta^{Q}}. \tag{32}$$

The actor policy function is presented with the network parameter θ^{μ}, which is updated using the critic network parameter θ^{Q} to optimize the expected function. The objective function $J(\theta)$ is the expected R_t under the policy $\mu(s \mid \theta^{\mu})$ which can be defined as

$$J\left(\mu^{\theta}\right)E[R_t] = E_{S\sim\rho_{\theta}^{\mu}, a\sim\mu^{\theta}}[Q(s, a)]. \tag{33}$$

A policy gradient method is generally obtained by the deterministic policy gradient with respect to network parameter θ^{μ}, with the deterministic strategy $a = \mu(s \mid \theta^{\mu})$. The gradient of the cost function with respect to θ^{μ} can be expressed as

$$\frac{\partial \nabla_{\theta^{\mu}} J}{\partial \theta^{\mu}} = \frac{\partial Q\left(s, a \mid \theta^{Q}\right)}{\partial a} \cdot \frac{\partial \mu(s \mid \theta) \theta^{\mu}}{\partial \theta^{\mu}}. \tag{34}$$

Hence, the parameter of the actor online policy network for the actor in DDPG can be updated by using the sampled policy

$$\nabla_{\theta^{\mu}} J \approx \frac{1}{N} \sum_{i} \left(\nabla_{a} Q\left(s, a \mid \theta^{Q}\right)|_{s=s_i, a=\mu(s_i)} \cdot \nabla_{\theta^{\mu}} \mu(s \mid \theta^{\mu})|_{s=s_i} \right). \tag{35}$$

In order to avoid the divergence of the algorithm, separate target networks are created as copies of the original actor network and critic network. In the DDPG algorithm, two target networks $Q\prime(s, a \mid \theta^{Q'})$ and $\mu\prime(s \mid \theta_{\mu'})$ are created for the main critic and actor networks, respectively. The two target networks have the same architecture with the main networks except that the target networks have different parameters θ'.

To improve the stability of the learning, we use the "soft" update method to update the parameters as illustrated by Mnih et al. The weights of the target network are constrained to update slowly by tracking the main networks: $\theta\prime = \tau\theta + (1 - \tau)\theta'$, where parameter $\tau \ll 1$. With the update like the proposed approach, the algorithm improves the stability of the network significantly. In addition, noise needs to be added samples from a noise process $\mathcal{N}$ to the actions for improving the efficiency of exploration because the actor policy is deterministic.

According to the above, the vectored thruster has one thruster, and two deflection angles need to be controlled. Hence, it is implied that the control system needs to produce the continuous control outputs of the three parts to achieve the designed reference dynamic state, such as the velocities of AUV. According to the dynamics of AUV, the designed control system must be able to complete the operational task of a nonlinear continuous control of the vectored thruster AUV in a complicated and changeable underwater environment. To solve the continuous control problem, an adaptive control system for the vectored thruster AUV based on DDPG algorithm and the study of AUV is proposed in this work. The aim of this study is to develop a new control algorithm, which has the ability to solve the problem of the vectored thruster AUV with different operative conditions. In our study of the AUV, the architecture of the control system based on DDPG algorithm can be illustrated as in Figure 9. As it can be seen, the designed control architecture can be divided into four factors: AUV reference generator unit, reward function unit, DDPG controller unit, and AUV environment unit.

As shown in Figure 9, the control architecture has an important part named as AUV reference speed generator, which can be used to generate a set of control data points for training AUV more effectively in the process of training. According to the use of the reference generator, the designed controller obtains the operating instructions $\mathbf{s}_t^r$ provided by the reference generator to manipulate the AUV. With this approach, the AUV controller can deal with different setting conditions for the AUV. Another import dynamic information is the measurement data $\mathbf{s}_t^m$ generated by the sensor system of the AUV. This item of measured data $\mathbf{s}_t^m$ is merged with $\mathbf{s}_t^r$ to produce an instantaneous error vector $\mathbf{e}_t$. This item presented the difference between the setting parameters and the practical measurement results that provide instantaneous information to the merged information $\boldsymbol{s}_t$. The reward function unit is the main indicator of evaluating the advantage and disadvantages of this algorithm. The input parameter of the reward function model is the instantaneous error vector e_t. In this way, the immediate reward r_t is defined by the reference state $\mathbf{s}_t^r$ and the error vector $\boldsymbol{e}_t$ to evaluate the operation situation of present activity and error in feedback. The AUV controller receives the information summarized in the system state $\boldsymbol{s}_t$ of AUV and the immediate reward $\boldsymbol{r}_t$ of the current states. According to the input parameters s_t and r_t, the AUV controller produces the action a_t to the AUV simulation environment based on a lot of studies and iterative computation. In practical application, the state information can be measured by the sensor system in AUV, such as DVL and IMU. In our work, the AUV simulation environment is established based on the study mentioned in Section 2; therefore, the state information can be obtained directly by the kinematic and dynamic analysis of AUV. This simulation method ensures the accuracy of development, increases productivity, and shortens development cycles. In addition, the state information based on simulation environment is more widely used at the earlier stage of AUV control based on DDPG algorithm. This method can be used to avoid making extraordinary efforts, especially at a great cost in experiments and computation, to complete the AUV controller training process.

Based on the presented DDPG algorithm architecture, the AUV controller for low-level control of vectored thruster AUV is developed. In order to help us represent the control system better, the algorithmic representation is developed to make our code more readable. Therefore, the algorithm for the vectored thruster AUV control can be summarized in the pseudocode as shown in Algorithm 1, and the algorithm workflow is shown in Figure 10.

In line 1 of Algorithm 1, the input parameters are the maximum training episodes M, iteration times of each episode T, the minibatch size of N, the soft target update factor for the deep target networks τ, the minimum and the maximum sizes of the replay buffer$\mathscr{R}$, and γ the discount factor. Since Algorithm 1 needs to learn from a continuous control problem, the actor network and critic network are initialized randomly, and the target networks are initialized with the same parameters $\theta^{Q'} = \theta^{Q}$, $\theta^{\mu'} = \theta^{\mu}$, as shown in line 2 and in line 3. In addition, the replay buffer $\mathscr{R}$ is also needed to initialize at the beginning stage (line 4). As a result of the analysis, this control problem of vectored thruster AUV can be seen as a continuous control task. In Algorithm 1, each episode of the learning process contains a loop

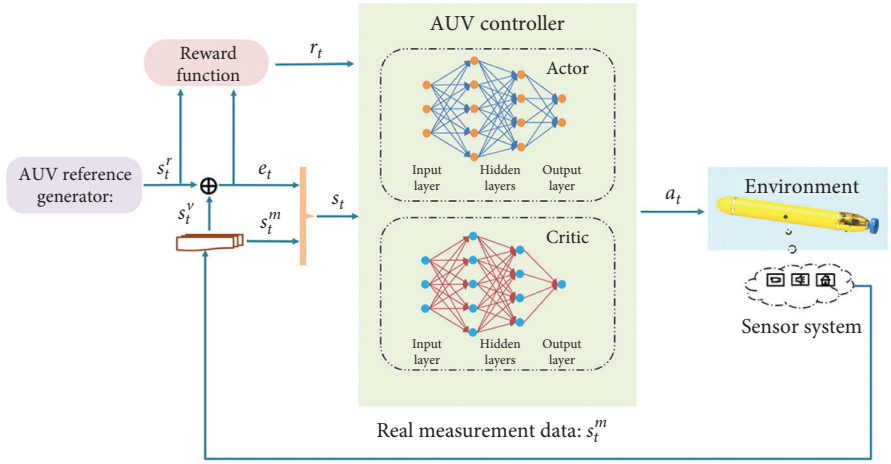

Figure 9: The control architecture based on DDPG algorithm.

(1) Input parameters M, T, N, θ, γ $m_{\min}$, $m_{\max}$
(2) Randomly initialize critic network $Q(s,a|\theta^Q)$ and actor $\mu(s|\theta^\mu)$ with weights θ^Q and θ^μ
(3) Initialize target network Q and μ' with weights $\theta^{Q\prime} \leftarrow \theta^Q$
(4) Initialize replay buffer $\mathscr{R}$
(5) For $episode = 0$ to M do
(6) Initialize a random process $\mathscr{N}$ for action exploration
(7) Initialize the AUV simulation environment
(8) Receive initial observation state s_1 from the AUV simulation environment
(9) For step = 0 to T do
(10) Select action $a_t = \mu(s_t, \theta^\mu) + \mathscr{N}_t$ according to the current policy and exploration noise
(11) Execute action a_t in the AUV simulation environment
(12) If $|\mathscr{R}| \geq m_{\min}$ then
(13) Sample a random minibatch of N transitions (s_i, a_i, r_i, s_{i+1}) from $\mathscr{R}$
(14) Set $y_i = r_i + \gamma Q\prime(s_{i+1}, \mu\prime(s_{i+1}|\theta^{\mu\prime})|\theta^{Q\prime})$
(15) Update critic by minimizing the loss: $L = (1/N)\sum_i (y_i - Q(s_i, a_i)|\theta^Q)^2$
(16) Update the actor policy using the sampled policy gradient:$\nabla_{\theta^\mu} \approx 1/N \sum_i \nabla Q(s,a|\theta^Q)|_{s=s,a=\mu(s_i)} \nabla_{\theta^\mu}(s|\theta^\mu)$
(17) Update the target Q networks: $\theta^{Q\prime} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q\prime}$
(18) Update the target policy networks: $\theta^{\mu\prime} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu\prime}$
(19) {End if}
(20) If $|\mathscr{R}| \geq m_{\max}$
(21) Remove the oldest stored data from the reply buffer
(22) End if
(23) Obtain the new state s_{t+1}
(24) Obtain reward r_t
(25) Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathscr{R}$
(26) End for
(27) End for
(28) Output parameters $Q(s,a|\theta^Q)$ and $\mu(s|\theta^\mu)$ with weights θ^Q and θ^μ

Algorithm 1: DDPG algorithm for AUV low-level control.

task with a fixed number of steps T. In training, the algorithm process, which is shown as line 5 to line 28 of Algorithm 1, is carried out as a loop with the max number of cycles M. In line 6, the random process $\mathscr{N}$, which refers to the random Ornstein–Uhlenbeck stochastic process, is carried out for the action exploration. In lines 7 and 8, the AUV simulation environment is initialized with the setting parameters at time 0 s, and the observation state s_1 can be obtained directly from the AUV simulation environment. In the inner loop from line 9 to line 27, the core part of our algorithm is performed to control the AUV. In the process of this algorithm, a fixed sample time of each step in the inner loop is set to dt, taking the practical application of the real AUV system and the efficiency into consideration. Thus, the training processes develop over time to meet the control system needs with the increased number of cycles. In the

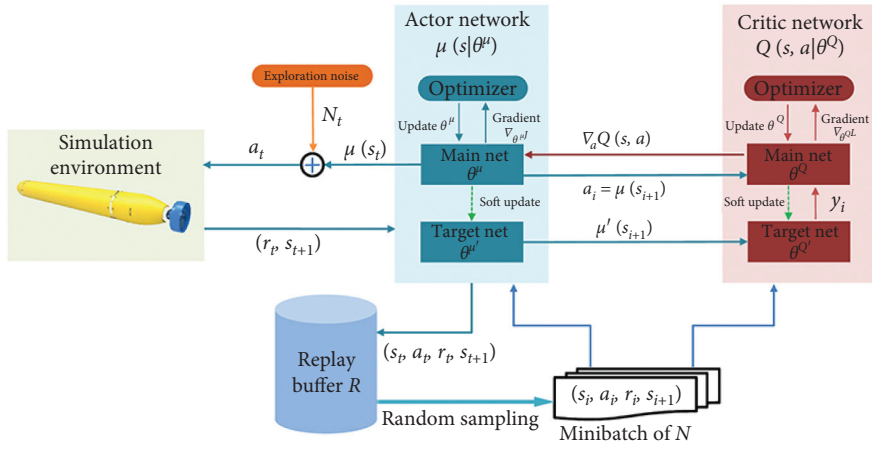

Figure 10: The algorithm flow based on DDPG for vectored thruster AUV.

establishment of the parameter, the time interval dt must be chosen effectively and reasonably. The control action a_t is obtained when the state s_t is given because the actor policy is determined (line 10). The action a_t is immediately sent to the AUV simulation environment to complete the corresponding motion control (line 11).

In order to improve the efficiency and reliability of the training process, the experience replay buffer $\mathscr{R}$ is used to update the actor and critic networks in the process of training. To ensure the normal operation of experience replays, the buffer $\mathscr{R}$ must have a sufficient number of transitions $m_{\min}$ stored to train the networks (line 12). When these conditions are fulfilled, a random minibatch N of transitions is sampled from the buffer $\mathscr{R}$ (line 13 and 14), and then the target state-action value y_i can be calculated, where $Q'(\cdot|\theta^{\mu'})$ and $\mu'(\cdot|\theta^{\mu'})$ can be obtained by the target Q network and target policy network, separately. The critic network can be updated by minimizing the loss function L, and the critic network parameter θQ (line 15) is obtained. In line 16, the actor network is updated by using the sampled policy gradient $\nabla_{\theta^{\mu}} J$, and the actor network parameter θ^{μ} is obtained. According to the network parameters $\theta^{Q}$and θ^{μ} calculated in line 14 to 16, the critic target Q network and the actor target policy network are updated in lines 17 and 18. In addition, when the size of the reply buffer $\mathscr{R}$ stored the experience for training reaches a maximum, then the earliest stored experience needs to be removed to improve efficiency and reduce costs (lines 20–22). The AUV receives the action a_t, and then the new state s_{t+1} can be obtained directly from the sensor system on the AUV in a real application, while the state information can be calculated by the simulation environment (line 23). Then, the reward function can calculate the immediate reward r_t to evaluate the effects of the action (line 24). Subsequently, with the combination of above the data obtained, the transition (s_t, a_t, r_t, s_{t+1}) is stored in $\mathscr{R}$ for training the networks. At the end of this algorithm, the critic network represented by $Q(s, a|\theta^{Q})$ and the actor policy network represented by $\mu(s|\theta^{\mu})$ are output.

5. Simulation Results

In our designed control system, the environment simulation is used to simulate the real underwater physics of AUV. During the simulations, the parameter sampling time used in Algorithm 1 is set to $dt = 0.1s$ with the concern about the actual application. In this vectored thruster AUV, the control commands are applied to the three functions, including propulsive force T_p, rudder angle α, and elevator angle β representing the deflection angles of the duct. The AUV-received commands can be defined by setting a vector $\mathbf{a}_t = (a_t^1, a_t^2, a_t^3)$, where a_t^1, a_t^2, a_t^3 are the force T_p, rudder angle α, and elevator angle β, respectively. Those commands are generated by the actor policy network of the designed controller.

In this control algorithm, the state $\mathbf{s}_t$ in the Markov process represents the current state of the vectored thruster in the underwater environment. In our AUV simulation environment, the state parameters, which can be expressed as $\mathbf{s}_t(\mathbf{v}_t, \omega_t, \dot{\mathbf{v}}_t, \dot{\omega}_t, \mathbf{e}_t)$, are defined by the instantaneous measurements from the sensors in AUV. The terms $\mathbf{v}_t = (v_x, v_y, v_z)$, $\omega_t(\omega_x, \omega_y, \omega_z)$ are the linear and angular velocities, which can be measured by DVL and IMU. $\dot{\mathbf{v}}_t = (\dot{v}_x, \dot{v}_y, \dot{v}_z)$, $\dot{\omega}_t(\dot{\omega}_x, \dot{\omega}_y, \dot{\omega}_z)$ are the linear and angular accelerations corresponding to the linear and angular velocities. $\mathbf{e}_t$ is the velocity error obtained by the real measured velocity $\mathbf{s}_v^r$ and the setting reference velocity at time t. The ultimate goal of this controller is to minimize the deviations of the real measured variable from the reference settings while minimizing the use of the vectored thruster to reduce the energy consumption. In addition, the fluctuation of the controlled dynamic variable of the AUV is not expected to be large enough, which will make it difficult to use in practical control. In order to accomplish this purpose, the reward function used in Algorithm 1 is essential for evaluating the effects of the executed action $\mathbf{a_t}$ for the system performance. In order to more fully evaluate the advantages and disadvantages of reward functions, a kind of reward function is proposed with different considerations in our study. Therefore, this immediate reward function $\mathbf{r_t}$ is defined as follows:

$$\mathbf{r_t} = -\zeta\left(\mathbf{s_t^v} - \mathbf{s_t^r}\right)^T\left(\mathbf{s_t^v} - \mathbf{s_t^r}\right) - \kappa\sum_{i=1}^{3}\left|a_i\right| - \xi\left|\overline{\mathbf{a}}_{t-1} - \overline{\mathbf{a}}_t\right|\mathrm{V} - \sigma\int e_t \mathrm{d}t\ , \tag{36}$$

where the first item evaluates the square error between the real measurement values $\mathbf{s_t^v}$ from the references$\mathbf{s_t^r}$. Due to the motion characteristics of AUV, a scale factor Λ, which can be defined as $\Lambda = ([\lambda_1^2, \lambda_2^2, \lambda_3^2])$, needs to be added to represent the error more efficiently. In the process of training, the parameters λ_i in factor Λ are changed according to the motion characteristics of this AUV. The second term is utilized to describe the actual use degree of the vectored thruster. The third term is added to avoid the vectored thruster producing sudden changes in propulsive force and duct deflection angles. In this term, it can be obtained by calculating the norm between the average of past executed actions $\overline{\mathbf{a}}_{t-1}$ and the current action $\mathbf{a}_t$. The average of past executed actions $\overline{\mathbf{a}}_{t-1}$ is obtained by computing the mean of action over a certain time period $t-1$ for each iteration. The last term presents the error accumulation between the real measured dynamic variable $\mathbf{s_t^v}$ from the references $\mathbf{s_t^r}$. This term is inspired by PID algorithm to reduce eliminate the steady error. The parameter ζ, κ, ξ and σ are scale factor $\in\in(0, 1]$.

In order to verify the feasibility of the proposed Algorithm 1, a numerical simulation is implemented in *Python* with TensorFlow. According to the related content in Section 4, the policy network uses a deep fully connected neural network with five layers, including three hidden layers, one input layer, and one output layer. The size of the input layer is 18, the sizes of hidden layers are 600 and 400, and the size of the output layer is 3. In addition, in the aspect of the selection of activation function, three hidden layers choose ReLU as activation functions, and the output layer chooses Tanh activation function. The state-action value networks use a similar network architecture apart from the size of the output layer. In addition, all parameters are set up before carrying out a series of numerical simulations. The maximum episode and step were fixed as M and T. During the process of training, the sampling time is set as dt, which is full in consideration of the calculation speed and accuracy of the designed simulation environment and the practical application of AUV.

The maximum and minimum sizes of the experience replay buffer $\mathscr{R}$ were set as $m_{\max}$ and $m_{\min}$. The learning rate for actor and critic networks is L_{R-A} and L_{R-C}. The discount rate and the soft updating rate for the target networks are γ and τ, respectively. The size of state transitions for the minibatch was defined as N. The parameter setting of the DDPG controller is shown in Table 2.

According to aforementioned proposed Algorithm 1 and related parameters, a series of simulations are carried out to study the effects of each term in reward function equation (36). In order to make a better comparison, the following related simulations are accomplished with the same reference state. The reference state is defined by velocities for the vectored thruster AUV $\mathbf{s_t^r} = (v_x^r, \omega_y^r, \omega_z^r)$. When the reference velocities $\mathbf{s_t^r} = (v_x^r, \omega_y^r, \omega_z^r) = (1\,(\mathrm{m/s}), 0, 0)$, the parameters in the scale factor Λ, $\lambda_1 = 1$, $\lambda_2 = \lambda_3 = 50$ Then, the performance of the reward function with $\zeta = 1\kappa = 0$, $\xi = 0$ and $\sigma = 0$ can be simulated, and the results are shown in Figure 11.

Table 2: The hyperparameters used for training the DDPG controller.

Hyperparameters	Value
M	3000
T	1000
dt	0.1 s
$m_{\max}$	500000
$m_{\min}$	1000
L_{R-A}	0.0001
L_{R-C}	0.0001
γ	0.99
τ	0.0001
N	64

As we can see in Figures 11(a) and 11(b), the linear and angular velocities are very close to the desired set reference velocities $\mathbf{s_t^r} = (1\ (\mathrm{m/s}), 0, 0, 0, 0, 0)$. The simulation results have proved the feasibility and correctness of the scheme. The results also illustrate the sample function reward, where only the errors between the real measurement values and the references are taken into consideration and have achieved good practical performance. According to Figures 11(a)–11(d), the linear and angular velocities have the phenomenon of higher volatility because the change extent of the thrust T_p and deflection angles of the duct α, β are great. In addition, the linear and angular velocities are bigger than the references, which will result in the unnecessary loss of power. Considering that the reference velocities are set to zero except the velocity in x-direction, the position and orientation of the AUV should be zero except the displacement in x-direction. Those deviations are also needed to be considered in the reward function for improving the performance of Algorithm 1. In order to achieve the goal of reducing energy consumption, the factor κ in the reward function is set to $\kappa = 0.001$, and the other parameters remain unchanged, as described above. The relative simulations are carried out, and the results are shown in Figure 12.

As we can see in Figures 12(a)–12(d), the simulation results are obtained based on the reward function with the new factors. By comparing the two results in Figures 11 and 12, the usage of the vectored thruster has declined significantly. The simulation results as shown in those pictures also illustrate the second term of the reward function and the reasonability and validity, which can smooth out the velocity fluctuation and enhance the reference tracking performance of algorithm effectively. Through the comparison of the amplitude variations of the thrust and duct angle with the same parameters, it is shown that the reward function considering energy consumption penalizes the usage of the vectored thruster while reducing its fluctuation range. Meanwhile, this method also reduces the deviations of position and orientation to provide more accurate control for the AUV according to the comparison of two simulation results.

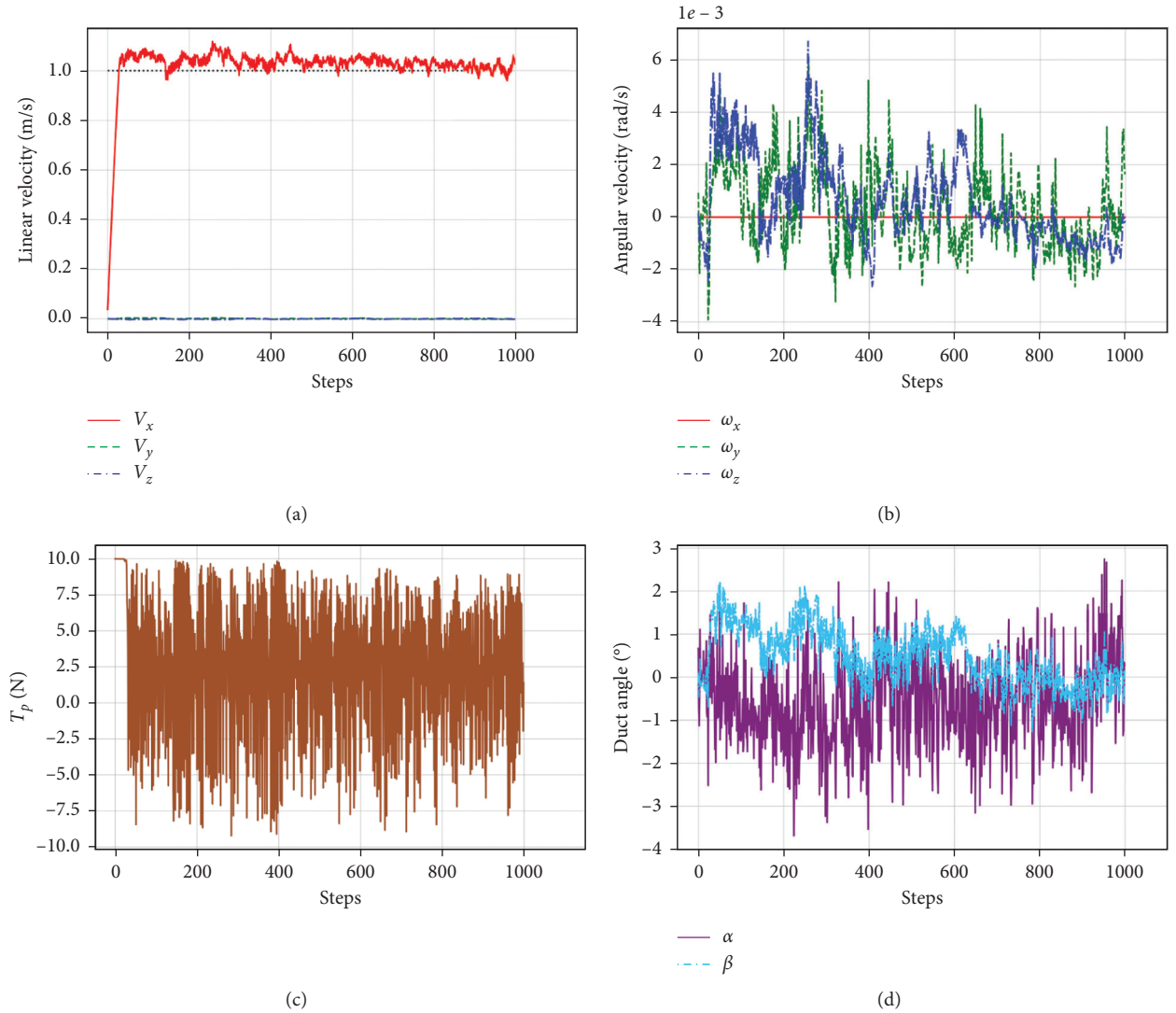

Figure 11: Simulation results with $\zeta = 1$, $\kappa = 0$, $\xi = 0$, and $\sigma = 0$. (a) Linear velocities. (b) Angular velocities. (c) Thrust for AUV. (d) Deflection angles of the duct.

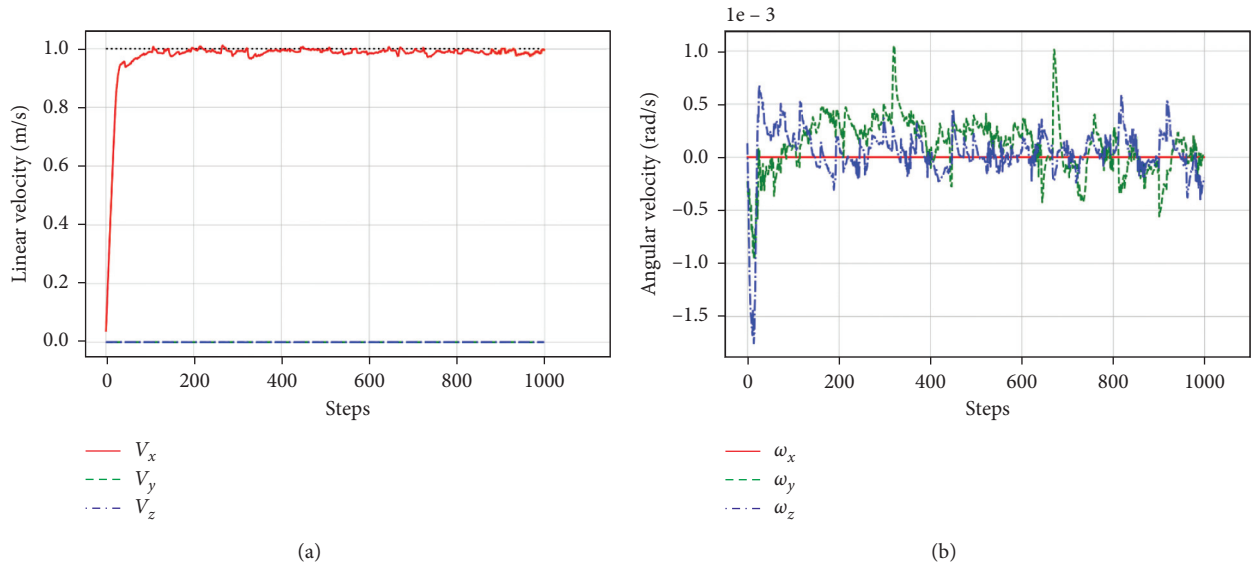

Figure 12: Continued.

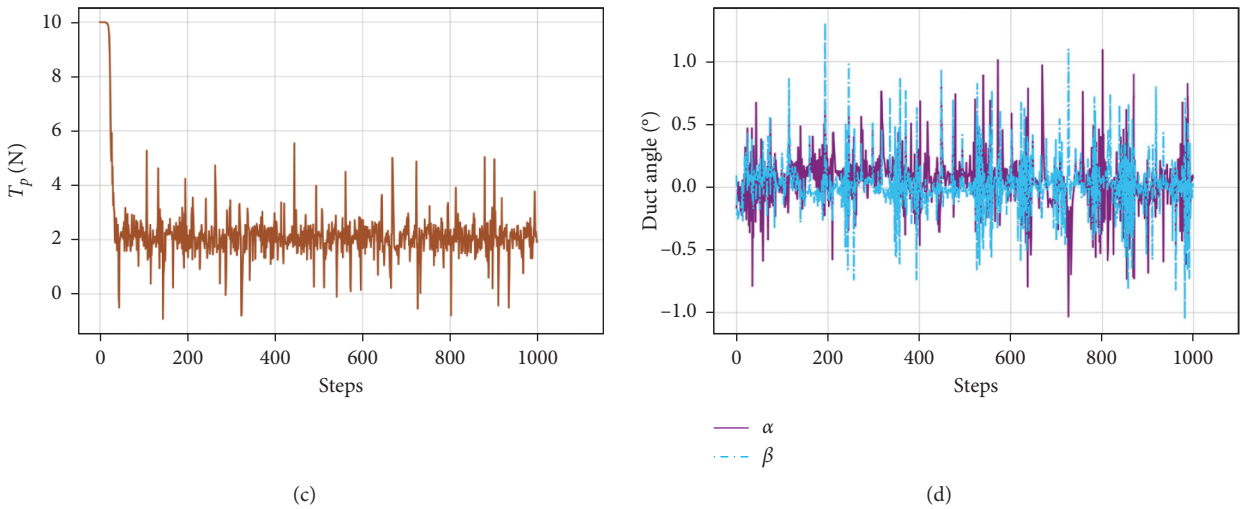

Figure 12: Simulation results with $\zeta = 1$, $\kappa = 0.001$, $\xi = 0$, and $\sigma = 0$. (a) Linear velocities. (b) Angular velocities. (c) Thrust for AUV. (d) Deflection angles of the duct.

However, the great change extent of the thrust T_p and the duct angles α, β will make it difficult for controlling vectored thruster to take advantage of the algorithm for AUV in the real application, while the results have proved that such reward function has achieved a good result. Hence, the third term with the factor $\xi = 0.02$, which presents the punishment term of the fluctuation of action outputs, is added in the reward function to evaluate the performance of actions. In addition, in order to further evaluate the reasonable existence of the second term in the reward function, another simulation is carried out with the factor $\kappa = 0$. The simulation results are obtained as shown in Figure 13.

Comparing the simulations of Figures 11–13, the current result considering the third term of the reward function proves that it is so useful to reduce the change ranges of action outputs. According to Figures 12 and 13, it should be noted that the second term of the reward function plays an important part in reducing the energy consumption for improving the performance of this AUV.

In order to improve the performance of the control system greatly, the first three terms are adopted in the reward function to further take advantage of the algorithm for AUV in the real application. Hence, in the next simulation, the factor is set to $\zeta = 1$, $\kappa = 0.001$, and $\xi = 0.02$, and the results are obtained as shown in Figure 14.

As we can see in Figure 14, the change ranges of the thrust T_p and the duct angles α and β are smaller than before, which makes this algorithm easier to use in the real application for AUV. Performance provides potent proof for the reward function with the second term used to stabilize and smooth the action outputs, while this term is applied in reducing energy consumption in the original design. Although simulations above indicate that this algorithm may gain better results in the vectored thruster AUV, the bias between the control deflection angles of the duct and the goal is large. Meanwhile, the biases of the duct angles α, β and thrust T_p lead to the large deviation in the position and orientation of the AUV. In order to further improve the performance of the algorithm, this bias about the thrust and duct angles is needed to be considered in the reward function. Based on the above comparison and consideration, the last term of the reward function, which is inspired by the integral item of error of PID algorithm, is added to reduce the effect of error propagation. The new simulation is carried out with the reward function with all the four terms considered, and the results can be obtained as shown in Figure 15.

The results of simulations are obtained and shown in Figure 15. As it can be seen, the improving performance indicates the effects of using the reward function with the punishment term of error accumulation. As shown in Figures 15(e) and 15(f), the position and orientation of the AUV also can be obtained. In particular, the biases of duct angles, position, and orientation of this AUV decrease effectively. It can be proved that the result of the reward function considering all aspects is good and stable from the comparison between current results in Figure 15 and other results above. After comparison with the simulation results between Figures 5 and 15, it has been found that a high coincidence rate is found between the designed controller based on Algorithm 1 and the traditional PID method. The results of simulation comparing RL and PID are obtained and shown in Figure 16.

As it can be seen, the results of simulation indicate that the controller based on DDPG makes a good performance in controlling the vectored thruster AUV problem. In contrast with the simulation results, the designed controller based on DDPG is better than PID controller in dynamic performance. In order to further research the performance of the designed controller under conditions of greater uncertain factors, the simulations are carried out to study anti-jamming performance with Gauss white noise excitations. The simulation results are shown in Figure 17.

Under the Gauss white noise disturbances presented in the simulation environment, the controller based on DDPG

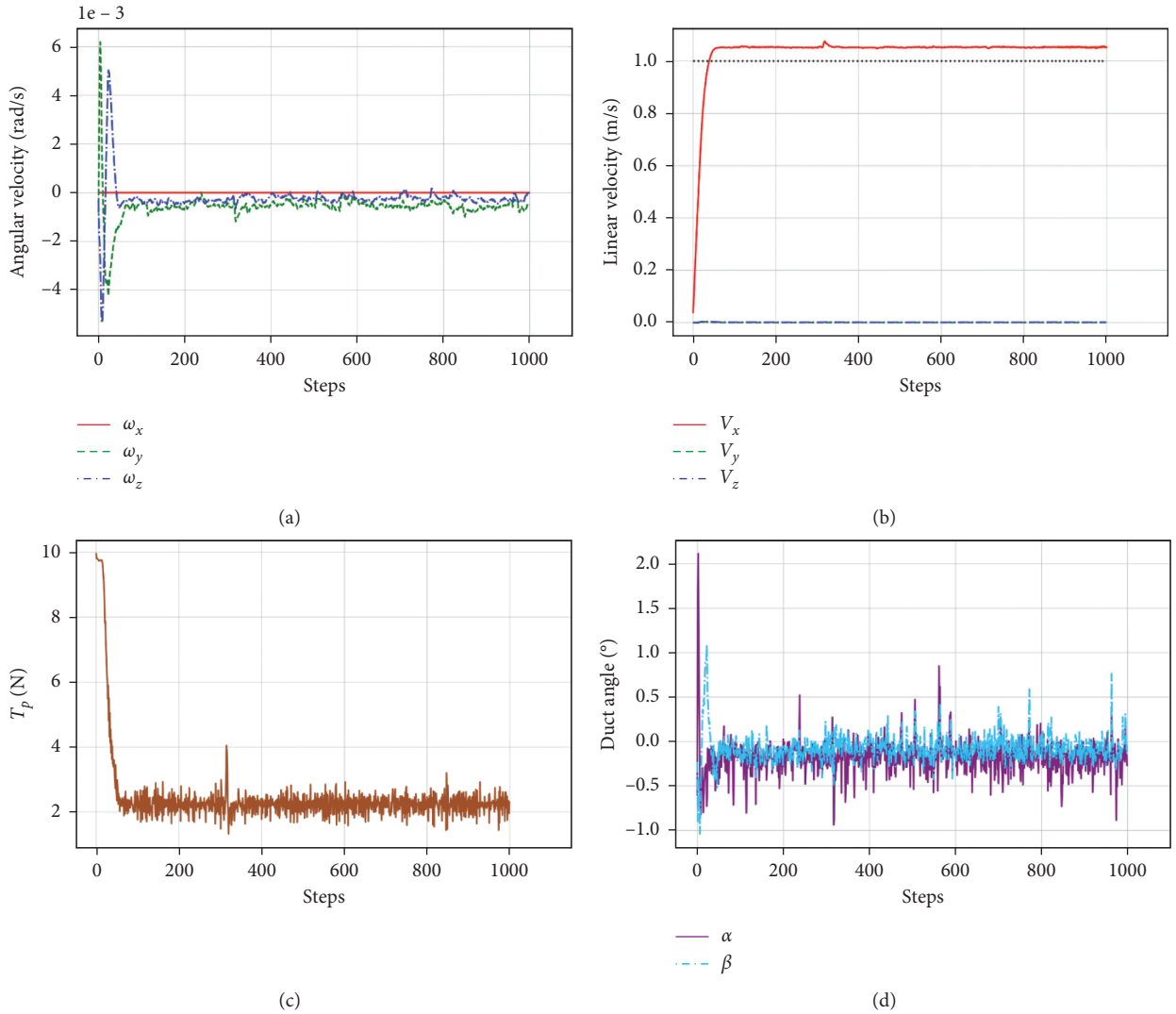

Figure 13: Simulation results with $\zeta = 1$, $\kappa = 0$, $\xi = 0.02$, and $\sigma = 0$. (a) Linear velocities. (b) Angular velocities. (c) Thrust for AUV. (d) Deflection angles of the duct.

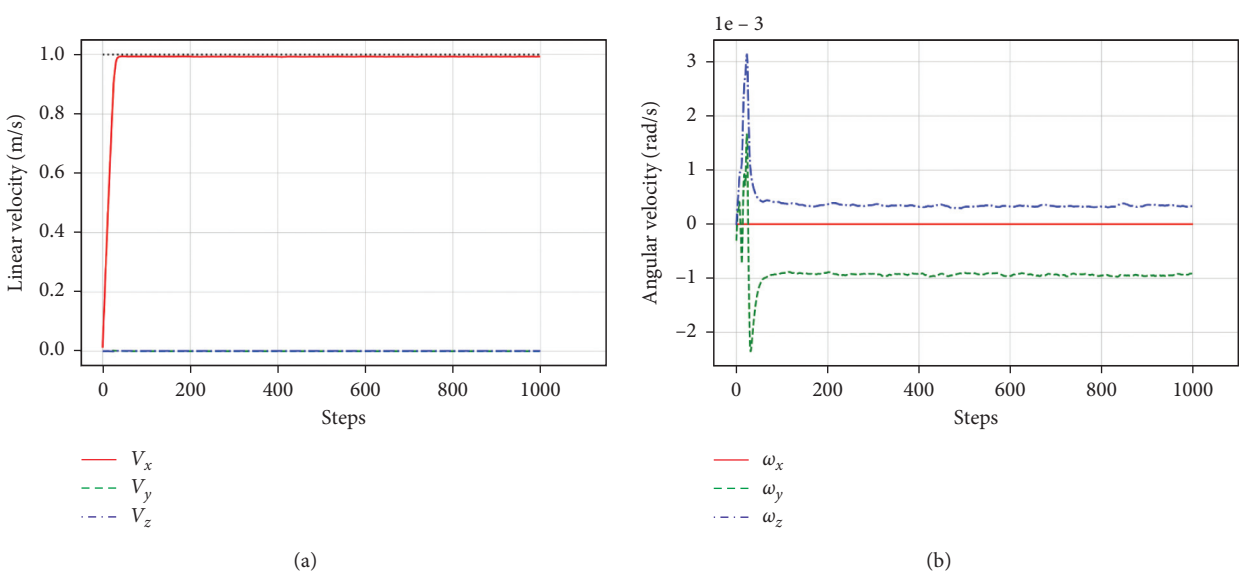

Figure 14: Continued.

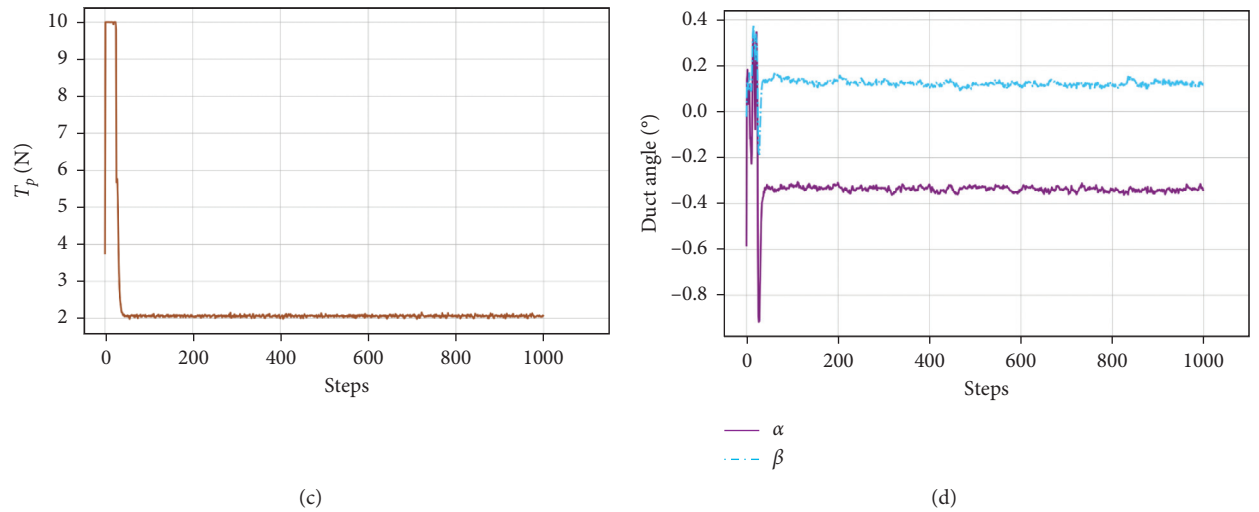

Figure 14: Simulation results with $\zeta = 1$, $\kappa = 0.001$, $\xi = 0.02$, and $\sigma = 0$. (a) Linear velocities. (b) Angular velocities. (c) Thrust for AUV. (d) Deflection angles of the duct.

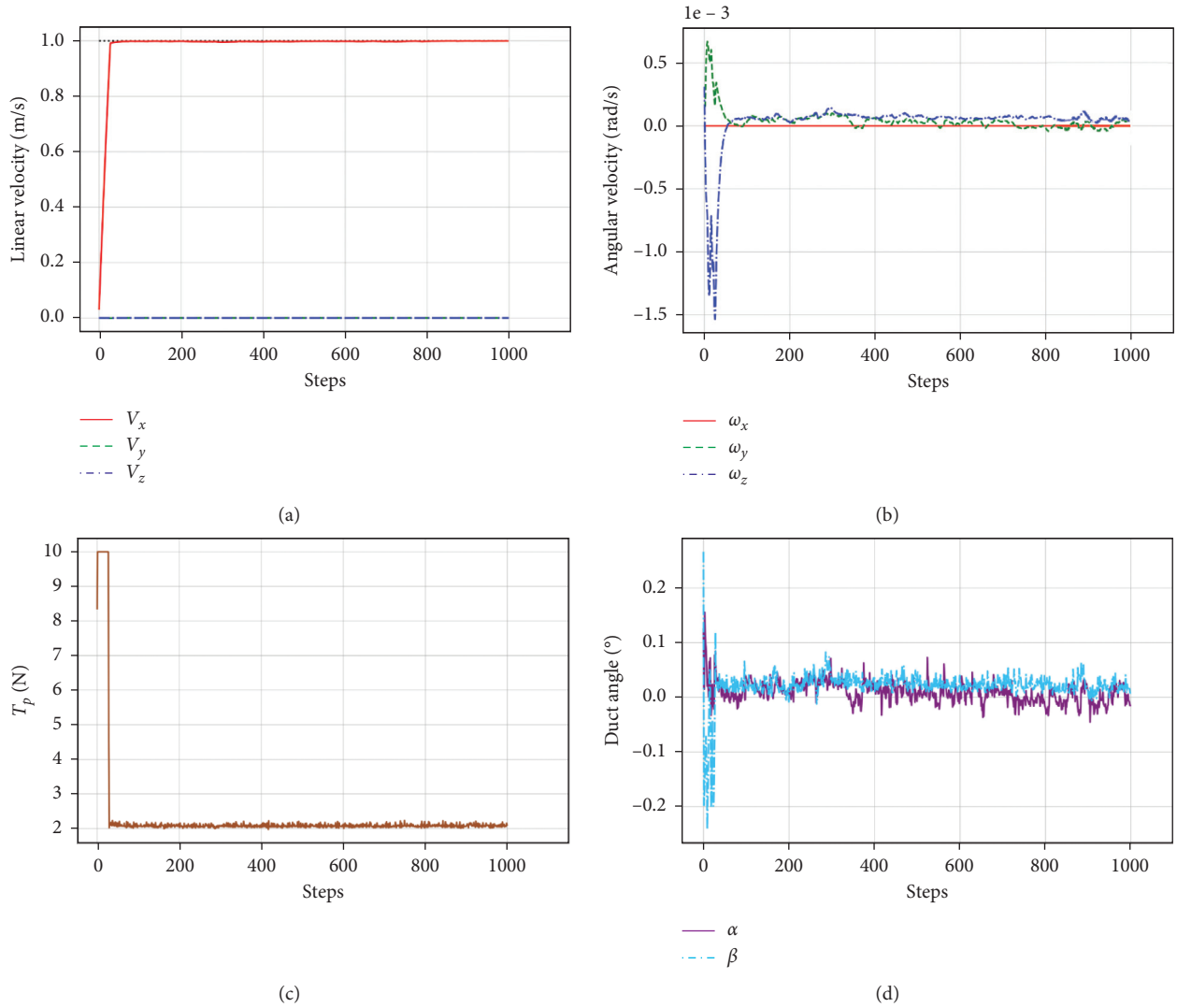

Figure 15: Continued.

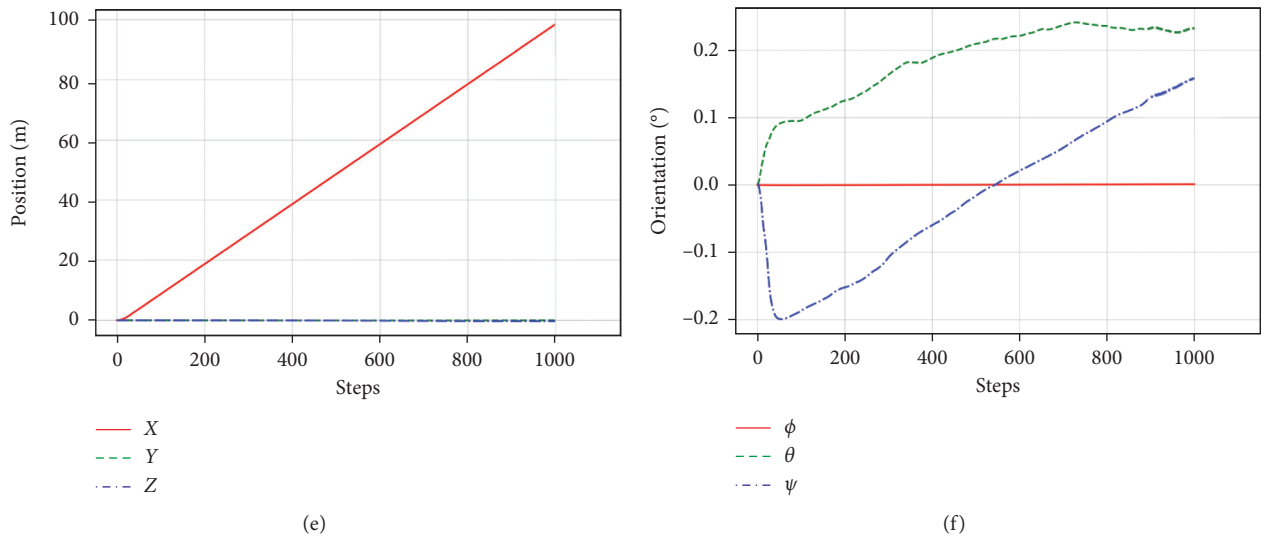

Figure 15: Simulation results with $\zeta = 1$, $\kappa = 0.001$, $\xi = 0.02$, $\sigma = 0$, and $\sigma = 1$. (a) Linear velocities. (b) Angular velocities. (c) Thrust for AUV. (d) Deflection angles of the duct. (e) Position of AUV. (f) Orientation of AUV.

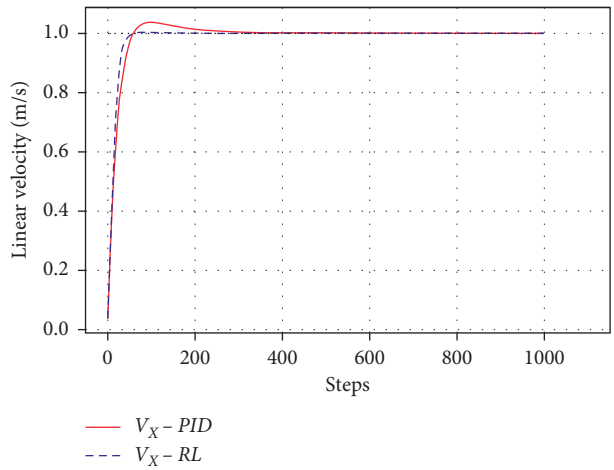

Figure 16: The simulation results comparing RL and PID.

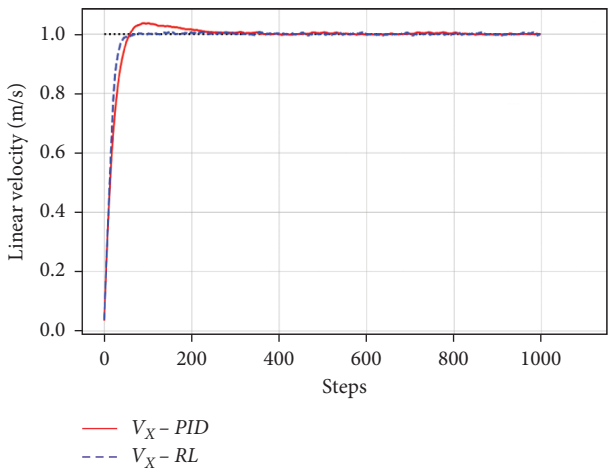

Figure 17: The simulation results comparing RL and PID with white noise.

and the PID controller could realize its function by the simulation. Based on the above research results, the results show that the designed control scheme based on DDPG has a good dynamic and static response and strong anti-interference ability. Simulation results from Figures 16 and 17 show that the proposed controller based on DDPG has better stability, fast convergence rate, and good tracking ability.

In order to test the capability of this algorithm, the other simulations with changed references are carried out. The new reference is set to $\omega_x^r = 0, \omega_y^r = 0.1$ (rad/s), $\omega_z^r = 0$, and the simulation results can be obtained after training the algorithm. The obtained results are shown in Figure 18.

As we can see in Figure 18, the angular velocity ω_y achieves the setting velocity requirements with good reliability. From Figures 18(c) and 18(d), the thrust output T_p is very stable within the limit of ultimate thrust, and the duct angle β is 15°, which is the limiting deflection angle of the

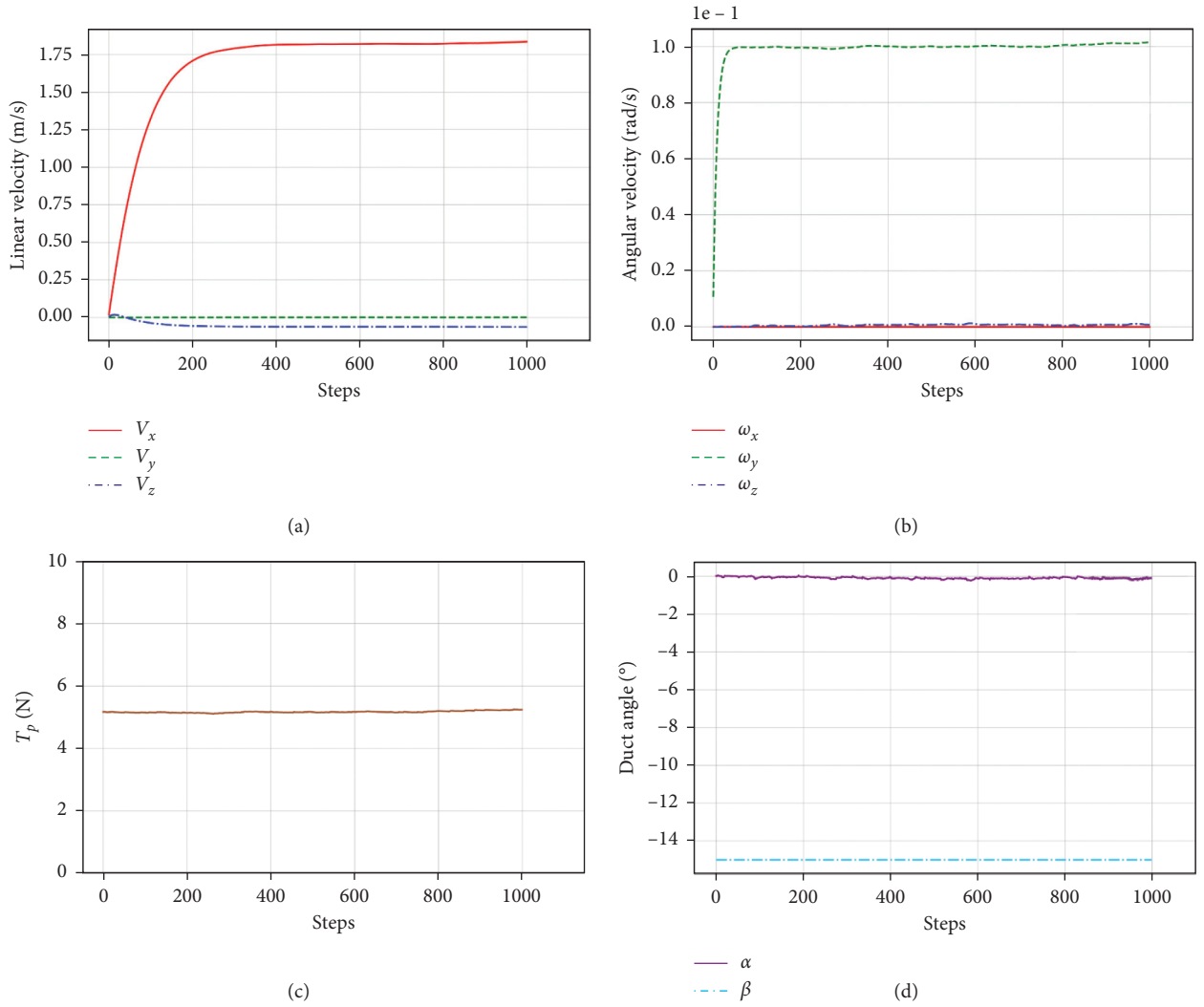

FIGURE 18: Simulation results with the reference velocities $\omega_x^r = 0, \omega_y^r = 0.1$ (rad/s), $\omega_z^r = 0$. (a) Linear velocities. (b) Angular velocities. (c) Thrust for AUV. (d) Deflection angles of the duct.

duct. By comparing the results between Figures 18 and 7, the designed controller and the reward function accomplish the functionality of controlling angular velocity for this vectored thruster AUV and the control performance is better than conventional PID controller.

We applied DDPG method on the proposed controller for the vectored thruster AUVs, and the training reward and the time consumption can be obtained as shown in Figures 19 and 20.

As we can see in Figure 19, the result showed that the value of the accumulated reward tended to monotonically increase until it reached about 1500 episodes. After that training episode, the accumulated reward tended to stabilize. According to this learning curve, we can discover the development tendency of the proposed controller based on DDPG method as the training proceeds. As we can see in Figure 20, the mean cost time of the episode is 9.24 seconds, and our method costs almost 7.7 hours in the whole 3000 episodes of simulated time, which corresponds to 3.5 days of computation in real time.

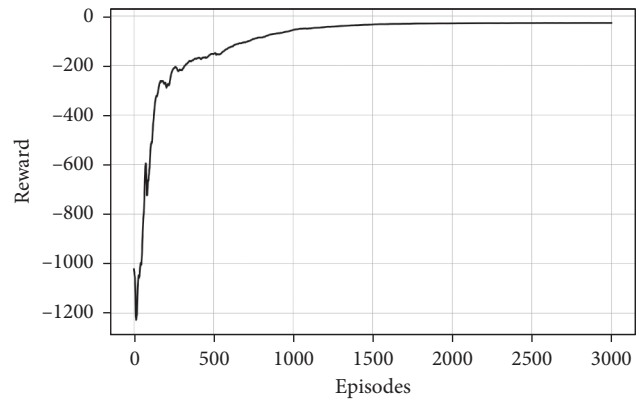

FIGURE 19: Episode reward while training under ideal conditions.

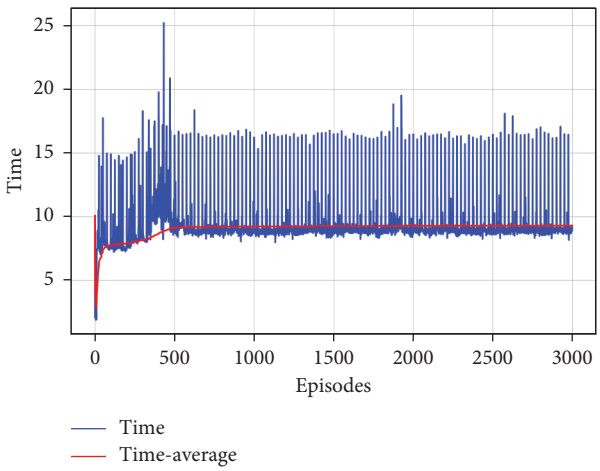

Figure 20: The cost time of episode.

6. Conclusion and Future Work

In this paper, an AUV controller based on the Deep Deterministic Policy Gradient (DDPG) was proposed for improving the control performance of the vectored thruster AUV. The proposed algorithm uses the information measured by internal sensors of AUV to provide the control commands for AUV to fulfill the task. There is no requirement to provide a model of the large complex nonlinear system about the vectored thruster AUV to the designed controller, which is essential to the classic control theory. It only needs some input parameters of the AUV, and our proposed algorithm is able to learn a control strategy for the AUV to meet exact implementation requirements. In the learning process, the reward function is fundamental to the DDPG controller to realize the system goal and related functions of the AUV. In this algorithm, a reward function is proposed by considering a series of control precision requirements and the influence of operational constraints. The designed reward function in this paper can effectively improve reliability and stability, reduce energy consumption, and restrain the vectored thruster sudden change. It should be particularly noted that the proposed control system based on DDPG algorithm was developed to realize the lower-layer motion control for the vectored thruster AUV, although some greater range of applications and more complex dynamic control systems can be solved by this method. Therefore, the controller based on DDPG algorithm has vast application and development prospects.

Furthermore, our proposed algorithm framework for AUV only uses some system states that can be measured by sensors directly as inputs, and it is different from a former method that uses images as input parameters. In this paper, it is proved that the motions of the AUV can be directly controlled by sending low-level control commands to the vectored thruster. In order to confirm the algorithm's effectiveness, a series of simulations are carried out in a simulation environment, which is established by the kinematic and dynamic analysis of the vectored thruster AUV. In this sense, we think the method using simulation environment to replace the real underwater application environment is proved to be cost saving and efficient improvement. In this sense, we think that our works have obtained certain improvements in expanding the application range of AUV control study using the deep reinforcement learning method. Furthermore, our proposed control algorithm provides an optional mentality for controlling underwater vehicles and other kinds of robotics.

Certainly, our present study has its limitations while achieving some achievement. In our proposed algorithm, the simulations are carried out under ideal conditions, so realistic experiments need to be completed to verify the correctness and feasibility of the proposed method. Moreover, more influence factors should be taken into account, such as time delay uncertainty among the sensors, actuators, and controllers. In addition, how to improve performance and achieve stability of the proposed controller is an important task for further research. Finally, control algorithms based on deep reinforcement learning have broad application background and important meanings in theory and practical engineering; therefore, the related research will become more important.

Data Availability

The data of hydrodynamic and thrust coefficients of the AUV used to support the findings of this study are included within the supplementary information file (Appendix B). The other data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Supplementary Materials

The supplementary file includes two parts: the element terms of dynamic equations of motion and the hydrodynamic and thrust coefficients of the AUV. The two supplementary files are important addition to model the complex dynamics of AUVs. (*Supplementary Materials*)

References

[1] Y.-C. Liu, S.-Y. Liu, and N. Wang, "Fully-tuned fuzzy neural network based robust adaptive tracking control of unmanned underwater vehicle with thruster dynamics," *Neurocomputing*, vol. 196, pp. 1–13, 2016.

[2] V. Kopman, N. Cavaliere, and M. Porfiri, "MASUV-1: a miniature underwater vehicle with multidirectional thrust vectoring for safe animal interactions," *IEEE/ASME Transactions on Mechatronics*, vol. 17, no. 3, pp. 563–571, 2011.

[3] T. K. Podder and N. Sarkar, "Fault-tolerant control of an autonomous underwater vehicle under thruster redundancy," *Robotics and Autonomous Systems*, vol. 34, no. 1, pp. 39–52, 2001.

[4] K. Tanakitkorn, P. A. Wilson, S. R. Turnock, and A. B. Phillips, "Depth control for an over-actuated, hover-capable autonomous underwater vehicle with experimental verification," *Mechatronics*, vol. 41, pp. 67–81, 2017.

[5] B. Xin, L. Xiaohui, S. Zhaocun, and Z. Yuquan, "A vectored water jet propulsion method for autonomous underwater vehicles," *Ocean Engineering*, vol. 74, pp. 133–140, 2013.

[6] R. Burcher and L. J. Rydill, *Concepts in Submarine design[M]*, Cambridge University Press, Cambridge, UK, 1995.

[7] T. Liu, Y. Hu, H. Xu, Q. Wang, and W. Du, "A novel vectored thruster based on 3-RPS parallel manipulator for autonomous underwater vehicles," *Mechanism and Machine Theory*, vol. 133, pp. 646–672, 2019.

[8] T. Liu, Y. Hu, H. Xu, Z. Zhang, and H. Li, "Investigation of the vectored thruster AUVs based on 3SPS-S parallel manipulator," *Applied Ocean Research*, vol. 85, pp. 151–161, 2019.

[9] R. Panish, "Dynamic control capabilities and developments of the bluefin robotics AUV fleet," in *Proceedings of the 16th International Symposium on Unmanned Untethered Submersible Technology*, Durham, UK, 2009.

[10] D. Caress, H. Thomas, W. Kirkwood, R. McEwen, R. Henthorn, and D. Clague, C. Paull, J. Paduan, K. Maier, J. Reynolds et al., "High-resolution multibeam, sidescan, and subbottom surveys using the MBARI AUV D. Allan B," *Marine Habitat Mapping Technology for Alaska*, pp. 47–70, 2008.

[11] R. Skjetne, T. I. Fossen, and P. V. Kokotović, "Adaptive maneuvering, with experiments, for a model ship in a marine control laboratory," *Automatica*, vol. 41, no. 2, pp. 289–298, 2005.

[12] K. D. Do and J. Pan, *Control of Ships and Underwater Vehicles: Design for Underactuated and Nonlinear Marine Systems*, Springer Science & Business Media, 2009.

[13] P. S. Londhe, M. Santhakumar, B. M. Patre, and L. M. Waghmare, "Task space control of an autonomous underwater vehicle manipulator system by robust single-input fuzzy logic control scheme," *IEEE Journal of Oceanic Engineering*, vol. 42, no. 1, pp. 13–28, 2016.

[14] B. M. Patre, P. S. Londhe, L. M. Waghmare, and S. Mohan, "Disturbance estimator based non-singular fast fuzzy terminal sliding mode control of an autonomous underwater vehicle," *Ocean Engineering*, vol. 159, pp. 372–387, 2018.

[15] B. Jalving, "The ndre-auv flight control system," *IEEE Journal of Oceanic Engineering*, vol. 19, no. 4, pp. 497–501, 1994.

[16] T. I. Fossen and O.-E. Fjellstad, "Robust adaptive control of underwater vehicles: a comparative study," *IFAC Proceedings Volumes*, vol. 28, no. 2, pp. 66–74, 1995.

[17] P. Herman, "Decoupled pd set-point controller for underwater vehicles," *Ocean Engineering*, vol. 36, no. 6-7, pp. 529–534, 2009.

[18] T. I. Fossen, *Guidance and Control of Ocean Vehicles*, Wiley, 1994.

[19] M. Takegaki and S. Arimoto, "A new feedback method for dynamic control of manipulators," *Journal of Dynamic Systems, Measurement, and Control*, vol. 103, no. 2, pp. 119–125, 1981.

[20] S. Miyamaoto, T. Aoki, T. Maeda et al., *Maneuvering Control System Design for Autonomous Underwater Vehicle*, vol. 1, pp. 482–489, 2001.

[21] P. Sarhadi, A. R. Noei, and A. Khosravi, "Model reference adaptive pid control with anti-windup compensator for an autonomous underwater vehicle," *Robotics and Autonomous Systems*, vol. 83, pp. 87–93, 2016.

[22] A. Malerba and G. Indiveri, "Complementary control of the depth of an underwater robot," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 8971–8976, 2014.

[23] F. Valentinis, A. Donaire, and T. Perez, "Energy-based motion control of a slender hull unmanned underwater vehicle," *Ocean Engineering*, vol. 104, pp. 604–616, 2015.

[24] G. Antonelli, S. Chiaverini, N. Sarkar, and M. West, "Adaptive control of an autonomous underwater vehicle: experimental results on odin," *IEEE Transactions on Control Systems Technology*, vol. 9, no. 5, pp. 756–765, 2001.

[25] J.-H. Li and P.-M. Lee, "Design of an adaptive nonlinear controller for depth control of an autonomous underwater vehicle," *Ocean Engineering*, vol. 32, no. 17-18, pp. 2165–2181, 2005.

[26] B. K. Sahu and B. Subudhi, "Adaptive tracking control of an autonomous underwater vehicle," *International Journal of Automation and Computing*, vol. 11, no. 3, pp. 299–307, 2014.

[27] G. Antonelli, "On the use of adaptive/integral actions for six-degrees-of-freedom control of autonomous underwater vehicles," *IEEE Journal of Oceanic Engineering*, vol. 32, no. 2, pp. 300–312, 2007.

[28] C. Barbalata, V. De Carolis, M. W. Dunnigan, Y. Petillot, and D. M. Lane, *An Adaptive Controller for Autonomous Underwater Vehicles*, pp. 1658–1663, 2015.

[29] R. Rout and B. Subudhi, "Inverse optimal self-tuning pid control design for an autonomous underwater vehicle," *International Journal of Systems Science*, vol. 48, no. 2, pp. 367–375, 2017.

[30] M. Santhakumar and T. Asokan, "A self-tuning proportional-integral-derivative controller for an autonomous underwater vehicle, based on taguchi method," *Journal of Computer Science*, vol. 6, no. 8, pp. 862–871, 2010.

[31] Y. Wang, B. Jiang, Z.-G. Wu, S. Xie, and Y. Peng, "Adaptive sliding mode fault-tolerant fuzzy tracking control with application to unmanned marine vehicles," *IEEE Transactions on Systems, Man, and Cybernetics*, 2015.

[32] Y. Wang, X. Yang, and H. Yan, "Reliable fuzzy tracking control of near-space hypersonic vehicle using aperiodic measurement information," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 12, pp. 9439–9447, 2019.

[33] Y. Wang, H. R. Karimi, H.-K. Lam, and H. Yan, "Fuzzy output tracking control and filtering for nonlinear discrete-time descriptor systems under unreliable communication links," *IEEE Transactions on Cybernetics*, vol. 50, no. 6, pp. 2369–2379, 2019.

[34] V. Utkin, "Variable structure systems with sliding modes," *IEEE Transactions on Automatic Control*, vol. 22, no. 2, pp. 212–222, 1977.

[35] K. D. Young, V. I. Utkin, and U. Ozguner, "A control engineer's guide to sliding mode control," *IEEE Transactions on Control Systems Technology*, vol. 7, no. 3, pp. 328–342, 1999.

[36] D. Yoerger and J. Slotine, "Robust trajectory control of underwater vehicles," *IEEE Journal of Oceanic Engineering*, vol. 10, no. 4, pp. 462–470, 1985.

[37] D. R. Yoerger and J.-J. Slotine, "Adaptive sliding control of an experimental underwater vehicle," in *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, pp. 2746–2751, IEEE, Sacramento, CA, USA, 1991.

[38] R. Cristi, F. A. Papoulias, and A. J. Healey, "Adaptive sliding mode control of autonomous underwater vehicles in the dive plane," *IEEE Journal of Oceanic Engineering*, vol. 15, no. 3, pp. 152–160, 1990.

[39] A. J. Healey and D. Lienard, "Multivariable sliding mode control for autonomous diving and steering of unmanned underwater vehicles," *IEEE Journal of Oceanic Engineering*, vol. 18, no. 3, pp. 327–339, 1993.

[40] T. Salgado-Jimenez and B. Jouvencel, "Using a high order sliding modes for diving control a torpedo autonomous underwater vehicle," *Teaming Toward the Future*, vol. 2, pp. 934–939, 2003.

[41] X. Liang, L. Wan, J. I. R. Blake, R. A. Shenoi, and N. Townsend, "Path following of an underactuated auv based on fuzzy backstepping sliding mode control," *International Journal of Advanced Robotic Systems*, vol. 13, no. 3, p. 122, 2016.

[42] Z. M. Zain and N. F. Harun, "Backstepping control strategy for an underactuated x4-auv," *Jurnal Teknologi*, vol. 74, no. 9, 2012.

[43] L. V. Steenson, A. B. Phillips, E. Rogers, M. E. Furlong, and S. R. Turnock, "Experimental verification of a depth controller using model predictive control with constraints onboard a thruster actuated auv," *IFAC Proceedings Volumes*, vol. 45, no. 5, pp. 275–280, 2012.

[44] C. Shen, Y. Shi, and B. Buckham, "Model predictive control for an AUV with dynamic path planning," in *Proceedings of the 2015 54th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), IEEE*, pp. 475–480, Hangzhou, China, July 2015.

[45] P. V. De Ven, C. Flanagan, and D. Toal, "Neural network control of underwater vehicles," *Engineering Applications of Artificial Intelligence*, vol. 18, no. 5, pp. 533–547, 2005.

[46] T. Fujii and T. Ura, "Development of motion control system for AUV using neural nets," in *Proceedings of the Symposium on Autonomous Underwater Vehicle Technology, IEEE*, pp. 81–86, Washington, DC, USA, June 1990.

[47] J. H. Li and P. M. Lee, "A neural network adaptive controller design for freepitch-angle diving behavior of an autonomous underwater vehicle," *Robotics 1025 and Autonomous Systems*, vol. 52, no. 2-3, pp. 132–147, 2005.

[48] K. Shojaei, "Neural network formation control of underactuated autonomous underwater vehicles with saturating actuators," *Neurocomputing*, vol. 194, pp. 372–384, 2016.

[49] F. Wang, Y. Xu, L. Wan, and Y. Li, "Real-time control of autonomous underwater vehicles based on fuzzy neural network," in *Proceedings of the 2009 International Workshop on Intelligent Systems and Applications, IEEE*, pp. 1–5, Wuhan, China, May 2009.

[50] Z. Yan, S. F. Chung, and J. Wang, "Model predictive control of autonomous underwater vehicles based on the simplified dual neural network," in *Proceedings of the 2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC), IEEE*, pp. 2551–2556, Seoul, Republic of Korea, October 2012.

[51] O. Elhaki and K. Shojaei, "Neural network-based target tracking control of underactuated autonomous underwater vehicles with a prescribed performance," *Ocean Engineering*, vol. 167, pp. 239–256, 2018.

[52] J. Zhang, X. Xiang, L. Lapierre, Q. Zhang, and W. Li, "Approach-angle-based three-dimensional indirect adaptive fuzzy path following of under-actuated AUV with input saturation," *Applied Ocean Research*, vol. 107, p. 102486, 2021.

[53] K. Shojaei, "Three-dimensional neural network tracking control of a moving target by underactuated autonomous underwater vehicles," *Neural Computing and Applications*, vol. 31, no. 2, pp. 509–521, 2019.

[54] Z. Zhong, Y. Zhu, and C. K. Ahn, "Reachable set estimation for takagi-sugeno fuzzy systems against unknown output delays with application to tracking control of auvs," *ISA Transactions*, vol. 78, pp. 31–38, 2018.

[55] J. Zhang, X. Xiang, Q. Zhang, and W. Li, "Neural network-based adaptive trajectory tracking control of underactuated auvs with unknown asymmetrical actuator saturation and unknown dynamics," *Ocean Engineering*, vol. 218, p. 1050, 2020.

[56] G. Che and Z. Yu, "Neural-network estimators based fault-tolerant tracking control for auv via adp with rudders faults and ocean current disturbance," *Neurocomputing*, vol. 411, pp. 442–454, 2020.

[57] O. Elhaki and K. Shojaei, "A robust neural network approximation-based prescribed performance output-feedback controller for autonomous underwater vehicles with actuators saturation," *Engineering Applications of Artificial Intelligence*, vol. 88, p. 103382, 2020.

[58] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, USA, 2018.

[59] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: a survey," *Journal of Artificial Intelligence Research*, vol. 4, no. 1, pp. 237–285, 1996.

[60] P. Dayan and K. C. Berridge, "Model-based and model-free pavlovian reward learning: revaluation, revision, and revelation," *Cognitive, Affective, & Behavioral Neuroscience*, vol. 14, no. 2, pp. 473–492, 2014.

[61] C. Gaskett and Z. A. Wettergreen, "Reinforcement learning applied to the control of an autonomous underwater vehicle," in *Proceedings of the Australian Conference on Robotics and Automation (AuCRA99)*, Brisbane, Australia, 1999.

[62] M. Carreras, J. Batlle, and P. Ridao, "Hybrid coordination of reinforcement learning-based behaviors for auv control," vol. 3, pp. 1410–1415, 2001.

[63] M. Carreras, J. Yuh, J. Batlle, and P. Ridao, "A behavior-based scheme using reinforcement learning for autonomous underwater vehicles," *IEEE Journal of Oceanic Engineering*, vol. 30, no. 2, pp. 416–427, 2005.

[64] M. C. El-Fakdi, "Policy gradient based reinforcement learning for real autonomous underwater cable tracking," in *Proceedings of the 2008 IEEE/RSJ Inter1075 National Conference on Intelligent Robots and Systems*, pp. 3635–3640, IEEE, St. Louis, MO, USA, 2008.

[65] S. A. Fjerdingen, E. Kyrkjebø, and A. A. Transeth, "Auv pipeline following using reinforcement learning," in *Proceedings of the ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics)*, pp. 1–8, VDE, Munich, Germany, 2010.

[66] G. Frost, F. Maurelli, and D. M. Lane, "Reinforcement learning in a behaviour-based control architecture for marine archaeology," in *Proceedings of the OCEANS 2015, IEEE*, pp. 1–5, Geneva, Switzerland, 2015.

[67] A. El-Fakdi and M. Carreras, "Two-step gradient-based reinforcement learning for underwater robotics behavior learning," *Robotics and Autonomous Systems*, vol. 61, no. 3, pp. 271–282, 2013.

[68] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[69] C. You, J.D. Filev, and P. Tsiotras, "Advanced planning for autonomous vehicles using reinforcement learning and deep inverse reinforcement learning," *Robotics and Autonomous Systems*, vol. 114, pp. 1–18, 2019.

[70] M. Zhu, X. Wang, and Y. Wang, "Human-like autonomous car-following model with deep reinforcement learning," *Transportation Research Part C: Emerging Technologies*, vol. 97, pp. 348–368, 2018.

[71] A. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep reinforcement learning framework for autonomous driving," *Electronic Imaging*, vol. 2017, no. 19, pp. 70–76, 2017.

[72] R. Yu, Z. Shi, C. Huang, T. Li, and Q. Ma, "Deep reinforcement learning based optimal trajectory tracking control of autonomous underwater vehicle," in *Proceedings of the 2017 36th Chinese control conference (CCC)," IEEE*, pp. 4958–4965, Dalian, China, July 2017.

[73] E. Anderlini, G. G. Parker, and G. Thomas, "Docking control of an autonomous underwater vehicle using reinforcement learning," *Applied Sciences*, vol. 9, no. 17, p. 3456, 2019.

[74] 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," *Robotics and Autonomous Systems*, vol. 107, pp. 71–86, 2018.

[75] T. Elmokadem, M. Zribi, and K. Youcef-Toumi, "Terminal sliding mode control for the trajectory tracking of underactuated autonomous underwater vehicles," *Ocean Engineering*, vol. 129, pp. 613–625, 2017.

[76] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT press, Cambridge, UK, 2018.

[77] J. C. van Rooijen, I. Grondman, and R. Babuška, "Learning rate free reinforcement learning for real-time motion control using a value-gradient based policy," *Mechatronics*, vol. 24, no. 8, pp. 966–974, 2014.

[78] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[79] M. Hessel, J. Modayil, H. Van Hasselt et al., "Rainbow: combining improvements in deep reinforcement learning," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, New Orleans, LA, USA, 2018.

[80] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proceedings of the Conference on Neural Information Processing Systems*, pp. 1057–1063, Long Beach, CA, USA, November 1999.

[81] G. L. Silver, N. Heess, and D. W. M. Degris, "Deterministic policy gradient algorithms," in *Proceedings of the International Conference on Machine Learning PMLR*, pp. 387–395, Beijing, China, June 2014.

[82] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 1008–1014, Denver, CO, USA, 2000.

[83] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: a brief survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, 2017.

[84] F. E. F. Junior and G. G. Yen, "Particle swarm optimization of deep neural networks architectures for image classification," *Swarm and Evolutionary Computation*.

[85] R. Liu and J. Zou, "The effects of memory replay in reinforcement learning," in *Proceedings of the 2018 56th Annual Allerton Conference on Communication, Control, and 1145 Computing (Allerton)*, pp. 478–485, IEEE, Monticello, IL, USA, 2018.