Target Tracking Control of a Biomimetic Underwater Vehicle Through Deep Reinforcement Learning

Yu Wang, *Member, IEEE*, Chong Tang, Shuo Wang, *Member, IEEE*, Long Cheng, *Senior Member, IEEE*, Rui Wang, Min Tan, and Zengguang Hou, *Fellow, IEEE*

***Abstract*— In this article, the underwater target tracking control problem of a biomimetic underwater vehicle (BUV) is addressed. Since it is difficult to build an effective mathematic model of a BUV due to the uncertainty of hydrodynamics, target tracking control is converted into the Markov decision process and is further achieved via deep reinforcement learning. The system state and reward function of underwater target tracking control are described. Based on the actor–critic reinforcement learning framework, the deep deterministic policy gradient actor–critic algorithm with supervision controller is proposed. The training tricks, including prioritized experience replay, actor network indirect supervision training, target network updating with different periods, and expansion of exploration space by applying random noise, are presented. Indirect supervision training is designed to address the issues of low stability and slow convergence of reinforcement learning in the continuous state and action space. Comparative simulations are performed to show the effectiveness of the training tricks. Finally, the proposed actor–critic reinforcement learning algorithm with supervision controller is applied to the physical BUV. Swimming pool experiments of underwater object tracking of the BUV are conducted in multiple scenarios to verify the effectiveness and robustness of the proposed method.**

***Index Terms*— Biomimetic underwater vehicle (BUV), reinforcement learning, target tracking control.**

I. Introduction

THE motion of fishes exhibits numerous advantages in terms of speed, efficiency, maneuverability, adaptability, and stealth [1]–[3]. Therefore, many fish-inspired underwater vehicles have been researched and developed [4], [5]. The early fish-like underwater robot was developed in the 1990s at Duke University, Durham, NC, USA, and the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA [6]. Since then, many studies related to fish-inspired vehicles have been done.

Fish-inspired vehicles, whose motions were achieved by bending their bodies, were developed. Zhou and Low [7] designed a robotic manta ray with biomimetic mechanisms, and its locomotion control was realized. Wang and Xie [8] developed a boxfish-like robot. The online high-precision probabilistic localization approach for this robot equipped with limited computational capacities and low-cost sensing devices was presented. Verma and Xu [9] developed a small two-link (one-joint) robotic fish, and a data-assisted dynamic modeling and control approach was proposed for robotic fish speed tracking. Zhang *et al*. [10] designed an agile robotic fish, where two caudal fins were equipped at the tail of the robotic fish in parallel as the main propulsion mechanism, and the opposite flapping of the two caudal fins produced mutually opposing lateral forces during cruising. Thus, stable and high-performance swimming was achieved. In addition, fish-inspired vehicles, whose motions were achieved by modulating median and/or paired fins, were developed. These vehicles, inspired by the swimming motions of knifefish, cuttlefish, and stingrays [11], [12], demonstrated potential in swimming stability and environment adaptation [13]. Wang *et al*. [14] developed a bioinspired robot with undulatory fins. Swimming motions comprising basic motion control, depth/course control, and waypoint tracking were achieved. Zhang *et al*. [15] developed a propulsion device with undulating fins. The unsteady flow field of undulating fins in stationary water was calculated, and the thrust generated by undulating fins was measured. Liu and Curet [16] designed a free-swimming robot with a single undulating fin, and several maneuvers, including forward swimming, reverse motion, diving, station-keeping, and vertical swimming, were replicated. Lin *et al*. [17] realized the motion control of an underwater robot with undulating fins.

Manuscript received September 3, 2019; revised February 23, 2020, July 8, 2020, and November 5, 2020; accepted January 20, 2021. This work was supported in part by the Youth Innovation Promotion Association CAS under Grant 2018162; in part by the National Natural Science Foundation of China under Grant U1713222, Grant 62025307, Grant 62073316, Grant U1806204, and Grant 62033013; and in part by the Beijing Municipal Natural Science Foundation under Grant JQ19020. *(Corresponding author: Long Cheng.)*

Yu Wang, Long Cheng, Rui Wang, and Zengguang Hou are with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: long.cheng@ia.ac.cn).

Chong Tang is with NUCTECH Company Ltd., Beijing 100084, China, and also with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China.

Shuo Wang is with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, also with University of Chinese Academy of Sciences, Beijing 100049, China, and also with the CAS Center for Excellence in Brain Science and Intelligence Technology, Shanghai 200031, China.

Min Tan is with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and also with the University of Chinese Academy of Sciences, Beijing 100049, China.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2021.3054402.

Digital Object Identifier 10.1109/TNNLS.2021.3054402

Tracking control of robots is an important study and has been dedicated by many researchers [18]–[20]. Huang [21] proposed a conventional Lyapunov-based motion controller for tracking a moving target by using a wheeled mobile robot with velocities. Chwa [22] presented a distance-based tracking controller for mobile robots in the presence of kinematic disturbances. A backstepping-like feedback linearization method was used to compensate for the unknown velocities and kinematic disturbances. Moreover, Yang *et al.* [23] devised a nonlinear controller for tracking and obstacle avoidance of a wheeled mobile robot with nonholonomic constraints. The above methods usually require an accurate model of robots. However, it is difficult to establish theoretical models due to the complicated and uncertain hydrodynamics of undulating fins. Some optimal control methods can be applied to realize robotic control without an accurate model [24], [25]. The iterative learning-based control methods that do not rely on the accurate model can be used to realize the tracking control of robots [26], [27]. Furthermore, reinforcement learning is also a model-free control method, which is a dynamic programming process to solve the Markov decision process (MDP) [28]–[30]. It can be used for underwater vehicle control. As it is difficult to build an effective dynamic model of an autonomous underwater vehicle (AUV), Wu *et al.* [31] presented a model-free reinforcement learning method that learned a state-feedback controller from the sampled trajectories of the AUV, and curved depth control was achieved. In addition, Shi *et al.* [32] devised a pseudo-Q-learning method for trajectory tracking control of underactuated AUVs with unknown dynamics and constrained inputs. Cui *et al.* [33] proposed an adaptive trajectory tracking control law using NN approximation for a fully actuated AUV and the NN-based reinforcement learning algorithm to address unknown disturbances, parameter uncertainties, and control input nonlinearities. Simulations were conducted to show the effectiveness of the above reinforcement learning methods.

Lillicrap *et al.* [34] proved the robustness of deep deterministic policy gradient (PG) provided guidance for the application to robot control. This article mainly focuses on the actual underwater robotic application of deep deterministic PGs (DPGs). The contributions of this article are given as follows.

1) Target tracking control of a biomimetic underwater vehicle (BUV) with undulatory fins based on deep reinforcement learning is studied. Deep reinforcement learning with a supervision controller is proposed. The supervised controller is presented to address the issues of low stability and slow convergence of reinforcement learning in the continuous state and action space, which can supply helpful control experience for the actor network.
2) Underwater object tracking experiments of the BUV in the swimming pool are conducted in multiple scenarios using the proposed deep reinforcement learning.

The remainder of this article is organized as follows. Section II describes the target system and the control problems. Section III describes the designing process of the state and reward function. In Section IV, the evaluation functions and policy functions are designed. Section V presents the training policy of reinforcement learning. A model is trained and simulated in Section VI. Section VII gives the experiment results, and the proposed method is validated. This article is concluded in Section VIII.

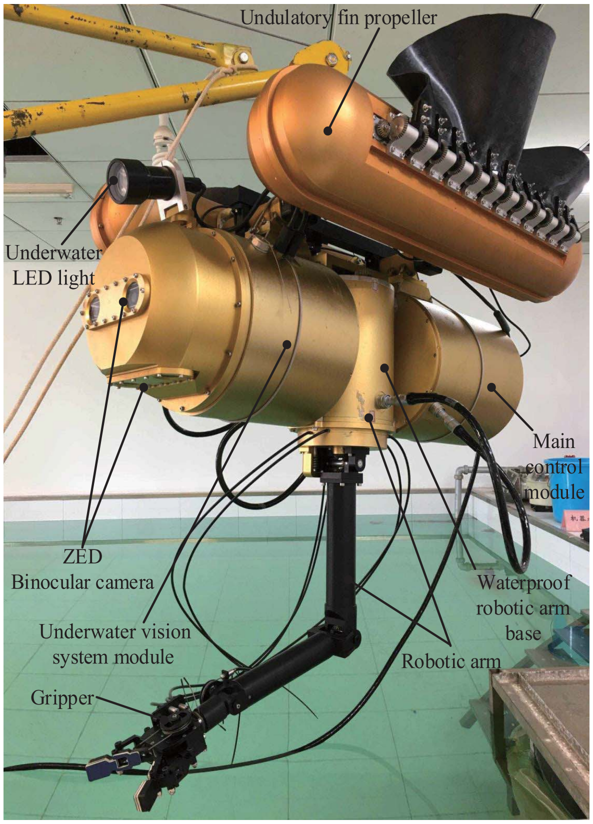

Fig. 1. BUV prototype.

II. System and Problems Description

A. System Description

A BUV propelled by undulatory fins is designed, as shown in Fig. 1. It consists of the main control module, a visual system module, an underwater manipulator, and two propulsors with undulatory fins. The main control module, visual system module, and robotic arm form an inverted triangular layout. The propulsors with undulatory fins are located symmetrically on both sides of the main body. We make the following approximations and assumptions for BUV. The BUV is a rigid body with good stability. This structure increases the good balance of the system in pitch and row directions. Therefore, the roll angle is $\phi = 0$, and the pitch angle is $\theta = 0$. Moreover, we assume that the depth of BUV is constant, and only the horizontal displacement is considered. An upper PC/104 unit and sensors for specific needs are installed inside the main control module. The visual system is mainly composed of the waterproof housing, Nvidia Jetson TX2, two pairs of ZED binocular cameras, floodlights, and a microcontroller board. Except for the floodlights, all other devices are mounted inside the waterproof housing. The Nvidia Jetson TX2 is served to process underwater images and videos because it is a

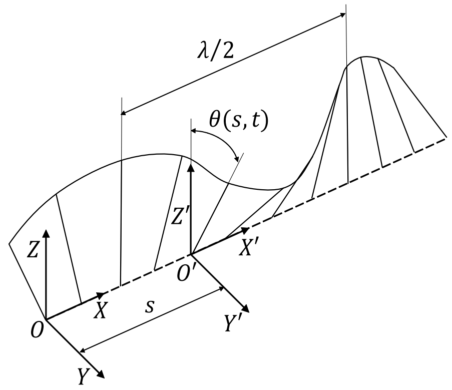

Fig. 2. Sinusoidal wave pattern of the biomimetic propeller.

relatively powerful and energy-efficient embedded device with a graphics processing unit (GPU).

The biomimetic propeller has flexible undulating fins that consist of 12 short rays connected by a black silicone sheet. By distributing all rays in interval phases of a sinusoid, thrust can be generated by the propeller. As shown in Fig. 2, the motion model of a single fin ray is

$$\theta(s,t) = \Theta \sin 2\pi \left(\frac{s}{\lambda} + f_i t + \phi\right) \quad (1)$$

where $\theta(s,t)$ defines the angular deflection of the ray at distance s along the axis X of coordinate system O-XYZ at time t. Θ is the maximum angular deflection of the sinusoidal waves. λ is the wavelength. $2\pi\phi$ is the initial phase of the waves. f_i is the frequency of the waves. Here, $i = 1, 2$. f_1 and f_2 denote the wave frequencies of the left fin and the right fin, respectively. Note that f_i can be positive or negative, and the propagating direction of the wave is determined by the plus–minus sign of f_i.

B. Problem Description

Reinforcement learning is an effective tool for solving the dynamic programming of MDP. A practical issue is first converted into a Markov decision problem, and the problem is then addressed via reinforcement learning. Since the BUV has inherent stabilization in rolling and pitching directions, the motion state of the BUV is defined as $\chi = [x, y, \psi]^T$, where (x, y) represents the coordinates of the BUV and ψ represents the yaw angle. The target object is detected and located by the onboard binocular vision. This visual scope is limited, and the tetrahedron $O_{c1}ABCD$ in Fig. 3 is approximate to the effective detection range of the binocular vision. The purpose is to keep the target within this effective detection range. Therefore, target object approaching and tracking control problem are described by

$$|g(x_o, y_o, \psi_o) - g(\chi)| < \varepsilon \quad (2)$$

where (x_o, y_o) denotes the target position, ψ_o denotes the target azimuth, and ε denotes the tolerance error.

MDP in reinforcement learning includes: state space S, action space A, reward function $S \times A \to R : r(s, a)$, and state transition probability distribution p. As the states in the state space S are bounded, the action space A covers the entire manipulation range. When the Markov property is satisfied, the next step system state is only determined by the current state and action, which is described as $p(x_t | x_{t-1}, a_{t-1})$. Because the stability of reinforcement learning is still an open question, the role of the rewards (equivalently, negative costs) is to represent stability requirements [41]. Therefore, the cumulative reward function is defined as the reward summation weighted by the future reward discount factor, which is given as

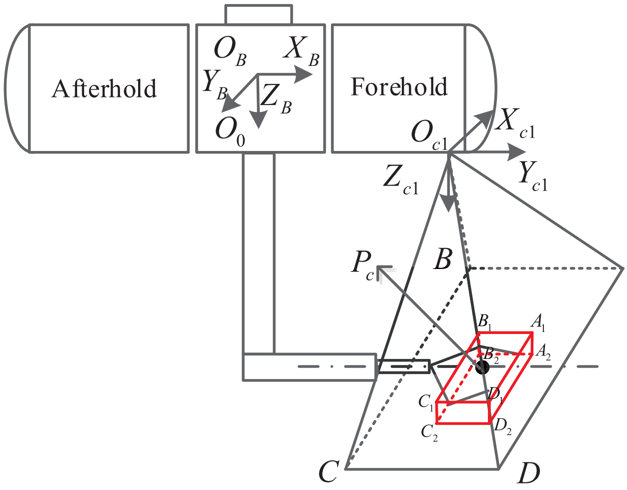

Fig. 3. Description of the target tracking control.

$$R_t = \sum_{i=t}^{T} \gamma^{(i-t)} r(s_i, a_i) \quad (3)$$

where $\gamma \in (0, 1]$ is the discount factor. When reinforcement learning control is applied to the BUV, the action a_t at t moment is the input control value of undulatory fins. During the learning process, the intelligent agent is trained by reinforcement learning interacts with the BUV. $a_t \in A$ is selected from the action space. System state transits from $x_t \in X$ to $x_{t+1} \in X$. Meanwhile, the reward r_t at t moment is used to evaluate a_t adopted under state x_t. The reward function is used to evaluate whether the action is optimal. Therefore, the task is to acquire the optimal policy π^* and maximum reward J^*. The maximum reward is calculated by

$$J^* = \max_{\pi \in P} J(\pi) = \max_{\pi \in P} E_\pi \left[\sum_{i=0}^{T} \gamma^i r_i | \pi \right] \quad (4)$$

where P denotes the policy space, and E_π denotes the average expectation of the policy. The final control value of the system is $u = a$. Therefore, the key to the above problem is to define the state of the MDP and the single-step reward function.

III. Design of System State and Reward Function

In order to solve the above problem, first, the system state and reward function is designed in this section. The pose of the tracking target in the BUV body-fixed frame is represented by $P_0 = [x_0, y_0, \psi_0]$. The pose of the workspace center is represented by $P_c = [x_c, y_c, \psi_c]$. System state also needs to include u, r_z, D. Here, u represents the forward speed, r_z represents the rotating speed. D is used to determine whether the BUV approaches the target or not. The MDP state is defined as

$$s = \left[D, x_o - x_c, y_o - y_c, \psi_o - \psi_c, u, r_z\right]^T. \tag{5}$$

The above equation is simplified to

$$s = \left[D, \Delta x, \Delta y, \Delta\psi, u, r_z\right] = \left[g, \Delta P_o^T, \Delta\psi, \nu^T\right]^T. \tag{6}$$

The reward function needs to consider both the position error and the yaw angle error. Thus, the position control of the BUV can be achieved through reinforcement learning. The BUV orientation is expected to be adjusted before the BUV starts to approach the target. The energy consumption of the BUV also needs to be considered. Consequently, the reward function is defined as

$$r = r_0 - \rho_1||\Delta\psi||_2 - \rho_2||\Delta P_0||_2 - \nu^T\rho_3\nu \tag{7}$$

where r_0 denotes a constant reward, $r_0 = 1$, indicating that the mission is accomplished; otherwise, $r_0 = 0$. $||\Delta\psi||_2$ represents the relative orientation error. $||\Delta P_0||_2$ represents the relative position error. $\nu^T\rho_3\nu$ represents the regular terms, which is used to reduce system energy consumption. ρ_1, ρ_2, and ρ_3 denote the weight coefficients.

IV. Evaluation Function and Policy Function

In order to solve the MDP problem in the previous chapter, the action-value function (Q-function) is given as

$$\begin{aligned} Q^\pi(s, a) &= E_\pi[R_t|s_t = s, a_t = a] \\ &= E_\pi\left[\sum_{k=0}^{K}\gamma^k r_{t+k+1}|s_t = s, a_t = a\right]. \end{aligned} \tag{8}$$

In (8), $\gamma \in (0, 1]$ denotes the discounting factor for future reward weight, and R_t denotes the sum of discounted future rewards. The system starts from state s, and the optimal policy π^* is executed. Thus, the optimal action-value function $Q^*(s\ a)$ satisfies the Bellman optimality principle, which is given as

$$Q^\pi(s_t, a_t) = \arg\max_a\left\{r_t + \gamma E_{s_{t+1}}\left[Q^*(s_{t+1}, a_{a+1})|x_t, a_t\right]\right\} \tag{9}$$

where $a_t = \pi^*(s_t)$. After the optimal Q^* is obtained by iterative computation, the optimal policy is given as

$$\pi^*(s) = \arg\max_\pi Q^*(s, a). \tag{10}$$

The PG method [35] is commonly used for reinforcement learning in continuous spaces. In order to improve the performance of the control policy, the policy function is updated toward the maximum reward function. However, the independent approximation function used in the PG method is a stochastic policy with low calculating efficiency. To overcome this disadvantage, Silver *et al.* [36] designed the DPG algorithm. A determined state-action mapping function $a = \mu_\theta(s)$ is used, where θ is the parameter of the policy function.

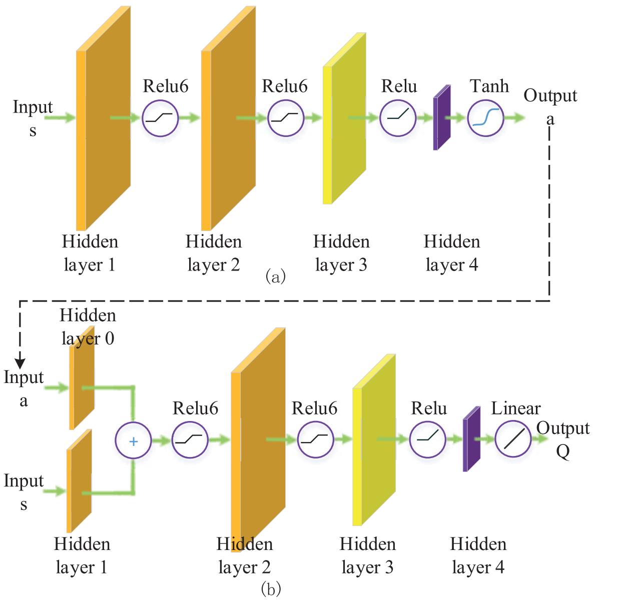

Fig. 4. Neural network of the evaluation function and the policy function. (a) Actor network. (b) Critic network.

The parameter θ is updated by the positive gradient of the cumulative reward, and the updating function is given as

$$\theta_{t+1} = \theta_t + \alpha_\theta\widehat{\nabla_\theta J(\theta)} \tag{11}$$

where $\widehat{\nabla_\theta J(\theta)}$ in the DPG algorithm is represented by [36]

$$\widehat{\nabla_\theta J(\theta)} = \frac{1}{M}\sum_{i=1}^{M}\nabla_\theta\mu_\theta(s_i)\nabla_{a_i}Q^\mu(s_i, a_i) \tag{12}$$

where Q^μ denotes the Q function associated with the policy function $\mu(a|s)$. By further derivation, we can get

$$\widehat{\nabla_\theta J(\theta)} = \frac{1}{M}\sum_{i=1}^{M}\nabla_\theta\mu_\theta(s_i)\nabla_{a_i}Q^\mu(s_i, \mu_\theta(a_i|s_i)). \tag{13}$$

The deep neural network is a very effective method to fit nonlinear functions such as the policy function and the action-value function. Here, an actor network and a critic network are constructed to approximate $\mu_\theta(a|s)$ and $Q_W(s, a)$. Their parameters are represented by θ and W, respectively. The parameter updating function for the Q function is [36]

$$\delta_t = r_t + \gamma Q_W(s_{t+1}, a_{t+1}) - Q_W(s_t, a_t) \tag{14}$$

$$W_{t+1} = W_t + \alpha_W\delta_t\nabla_W Q_W * (s_t, a_t). \tag{15}$$

The structures of the actor network and critic network are shown in Fig. 4. The input of the actor network is $s \in R^6$. The output of the actor network is $a = [f_1, f_2]^T \in R^2$, where f_1 and f_2 are the wave frequencies of the left propulsor and the right one, respectively.

In the actor network, the number of neuron cells in both hidden layers 1 and 2 is 200, and the activation function is Relu6. The number of neuron cells in hidden layer 3 is 10, and the activation function is Relu. The number of neuron cells in the output layer is 2, and its activation function is tanh. The output value is normalized into $[-1, 1]$. As depicted

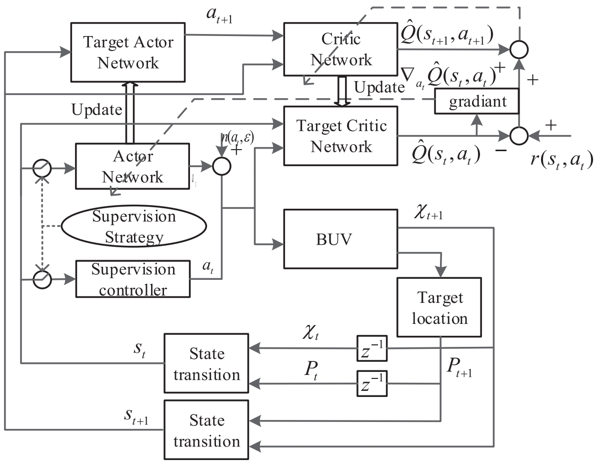

Fig. 5. Schematic of DPG actor–critic algorithm with the supervision controller.

in Fig. 4(b), the inputs of the critic network include state s and action a. The number of neuron cells in both hidden layers 0 and 1 is 200. State s and action a go through the convolution layer 0 and convolution layer 1, respectively. They are integrated together by summation and activated by the Relu6. The result is served as the input of the hidden layer 2. The number of neuron cells in hidden layer 2 is 200, and the activation function is Relu6. The number of neuron cells in hidden layer 3 is 10, and the activation function is Relu. The output layer has one nerve cell, and the linear activation function is used.

V. Training Tricks

Functions and networks have been designed according to previous chapters. However, the training strategy affects the result of training and control directly. In order to solve the problems of convergence and stability of reinforcement learning training, the following training tricks are adopted.

A. Prioritized Experience Replay

In order to eliminate the correlation of training data and to improve the training effect, an experience database needs to be built to develop an experience replay [37]. The action a_t is taken in the state s_t. Hereafter, the next state s_{t+1} is observed, and the reward r_t in t moment is obtained. A training experience is represented by $\{s_t, a_t, s_{t+1}, r_t\}$. In this section, the prioritized experience in [31] is used, and learning becomes more necessary as the error becomes bigger. The current experience deviation is obtained as

$$e_t = |r_t + \gamma\, Q_W(s_{t+1}, a_{t+1}) - Q_W(s_t, a_t)|. \tag{16}$$

The dimension of experiences is expanded, which is denoted as $\{s_t, a_t, s_{t+1}, r_t, e_t\}$.

B. Actor Network Indirect Supervision Training

Reinforcement learning training is experiential learning based on data, so successful experiences are critical to the network training. In the initial training phase of reinforcement learning, insufficient successful experiences in the database will lead to the low speed of convergence and low learning efficiency. Hence, a supervisory controller is designed to address this problem, which plays an important role in the initial phase of reinforcement learning. The training process then gradually transits to the self-learning phase. During the self-learning process, the supervisory controller still intervenes in reinforcement learning in a low probability. The action a generated by the policy network is disturbed to optimize the policy network until it surpasses the supervisory controller. Hence, the supervision achieved by a conventional controller is presented to guarantee the stable convergence of the control actor network. It implements indirect supervised training of actor networks through evaluation functions and repeated training processes. Similar effects can be achieved with any traditional controllers, including fuzzy control and adaptive control [38]. Here, the PID controller is applied as the supervisory controller

$$\begin{cases} f_{r1} = P_r\Delta\psi + I_r\int\Delta\psi + D_r\Delta\dot{\psi} \\ f_{r2} = -f_{r1} \\ f_{x1} = P_x\Delta x + I_x\int\Delta x + D_x\Delta\dot{x} \\ f_{x2} = f_{x1} \\ f_1 = \rho_{r1} f_{r1} + \rho_{x1} f_{x1} \\ f_2 = \rho_{r2} f_{r2} + \rho_{x2} f_{x2} \end{cases} \tag{17}$$

where $P_i, I_i, D_i, i \in \{r, x\}$ are the parameters of PID. f_{r1} and f_{r2} denote the wave frequency components of the left fin and the right fin, which are used to control yaw angle, respectively. f_{x1} and f_{x2} denote the wave frequency components of the left fin and the right fin, which are used to control forward movement, respectively. ρ_{r1}, ρ_{r2}, ρ_{x1}, and ρ_{x2} denote the weighting coefficients of wave frequency rotating and forward components. Thus, the supervision strategy is designed as

$$\begin{cases} \mathrm{PRO}_t = \mathrm{PRO}_0 * 0.999^t \\ \mathrm{PRO}_r = \max(\mathrm{rand}(1), 0.01) \\ a = \begin{cases} \mathrm{Actor}(s_t), & \text{if } \mathrm{PRO}_r > \mathrm{PRO}_t \\ \mathrm{PID}(s_t), & \text{if } \mathrm{PRO}_r \le \mathrm{PRO}_t \end{cases} \end{cases} \tag{18}$$

where PRO_0 represents the initial supervision strategy training probability. PRO_t represents the supervision strategy training probability at moment t. Its value decreases exponentially with the increase in the training steps. PRO_r represents the random supervision training probability. In case of $\mathrm{PRO}_r > \mathrm{PRO}_t$, the actor network output action a is adopted. Otherwise, the supervision PID controller's output action a is adopted.

C. Target Network Updating With Different Periods

Accurate evaluations are conducive to the actor network training. Hence, the critic network updating speed needs to be faster than that of the actor network during the training process [39]. The actor network can be trained and updated by better evaluations. The updating strategies of the actor network and the critic network are given as

$$\begin{cases} \theta' = \tau\theta + (1-\tau)\theta', & \text{if } \operatorname{mod}(t, \mathrm{FA}) = 0 \\ W' = \tau W + (1-\tau)W', & \text{if } \operatorname{mod}(t, \mathrm{FQ}) = 0 \end{cases} \tag{19}$$

where θ' denotes the target parameter of the actor network. W' denotes the target parameter of the critic network. τ denotes the updating coefficient. FA and FQ denote the update cycles of the target actor network and target critic network, respectively. Generally, FQ $<$ FA. mod(,) is the remainder function.

Algorithm 1 Actor–Critic Algorithm With Supervision Controller

```
Initialization
Initialize the critic network $Q_w(s,a)$ and the actor network $\mu_\theta(s)$ randomly;
Initialize the target critic network $Q'_{W'} = Q_W$ and the target actor network $\mu'_{\theta'} = \mu_\theta$;
Initialize the experience database $D$;
Initialize the epoch $M$, training step $N$, batch size $T$, learning rate $\alpha_\theta, \alpha_W$, discount factor $\gamma$ and updating coefficient $\tau$.
for Epoch number = 1:M do
  Initialize the initial state $s_t$ randomly;
  for Training step = 1:N do
    if ($\mathrm{PRO}_r \le \mathrm{PRO}_t$) then
      PID supervision controller is used to generate control output $a_t$;
    else
      Execute the actor network, $a_t = \mu(s_t) + \Delta\mu_t$, where $\Delta\mu_t$ is a random variable;
    end if
    Execute $a_t$, observe the new state $s_{t+1}$, calculate the reward $r(s_t, a_t)$ and error $e_t$;
    $\{s_t, a_t, s_{t+1}, r_t, e_t\}$ is stored into experience data base D;
    Select $T$ set of training experiences from database $D$, which constitute the training data for this time;
    for i = 1:T do
      Calculate the error $\delta_t = r_i + \gamma Q_W(S_{i+1}, a_{i+1}) - Q_W(s_i, a_i)$;
      Calculate the gradient $\nabla_{a_i} Q^\mu(s_i, a_i)$;
      Update $\{s_t, a_t, s_{t+1}, r_t, e_t\}$;
    end for
    Batch update the parameters of Critic network;
    $W_{t+1} = W_t + \alpha_W (1/T) \sum_{i=1}^{T} \delta_i \nabla_W Q_W(s_t, a_t)$;
    Batch update the parameters of Actor network;
    $\theta_{t+1} = \theta_t + \alpha_\theta (1/T) \sum_{i=1}^{T} \nabla_\theta \mu_\theta(s_i) \nabla_{a_i} Q^\mu(s_i, \mu_\theta(s_i))$;
  end for
  Update the target network;
  $\theta' = \tau\theta + (1-\tau)\theta'$;
  $W' = \tau W + (1-\tau)W'$;
end for
```

D. Expansion of Exploration Space by Applying Random Noise

The tradeoff between exploration and exploitation is important in reinforcement learning. Sufficient generalization capability of the actor network benefits from sufficient data. To improve the actor network generalization ability, the random noise is added to the actor network output [40], which is computed as

$$\begin{cases} c_t = \max\big(c_0 * 0.999^t, 0.01\big) \\ a_t = a_t + \mu(t) = a_t + e^{-(\omega - a_t)/\left(2c_t^2\right)} \end{cases} \tag{20}$$

where c_0 represents the initial noise standard deviation. c_t represents the noise standard deviation at t moment, and its value reduces exponentially with an increase in the training steps. The mean value of $\mu(t)$ is denoted by a_t. c_t represents the random Gaussian noise of standard deviation.

Based on the training strategies and actor–critic reinforcement learning algorithm framework, Fig. 5 shows the schematic of the DDPG actor–critic algorithm with supervision controller. The details of this algorithm are given in Algorithm 1.

VI. Model Training and Simulation

The network is trained based on the above model structure and training methods. Reinforcement learning with indirect supervision is a model-free method. Simulations are conducted to verify the effectiveness of the proposed method. The controlled object can be a high order system. The construction and training of reinforcement learning algorithms are based on TensorFlow. The programming language is python. The simulation schemes are uploaded to GitHub (see the link: https://github.com/liuheng92/tensorflow_PSENet/issues/73).

The procedure of the simulation is given as follows. First, the neural networks described in this article are built and linked on the TensorFlow. Then, according to the law of state transition, the weight of the neural networks is continuously updated by the reinforcement learning algorithms based on the collected training data. When the algorithms converge, the required control strategy is obtained. The training results are represented by the weights of these neural networks. The number of training epochs for this network is 2000. The number of training steps in each epoch is 600. The size of the experiences database is 10 000. The batch size is 32. The learning rate of the actor network and the critic network is 0.001. The value of the weight discount factor is $\gamma = 0.9$. Five different training methods are organized and tested; their configurations are shown in Table I. The initial position of the target in the body coordination is $P = [-82, 220]$ mm. The results are shown in Table II and Fig. 6. The results indicate that the network trained by method 5 has the highest cumulative rewards and the least number of control steps. Random noise that is added to the training process should be able to improve the generalization ability of the model. However, the performance of the network trained by method 4 is slightly lower than the one of the network trained by method 5. In the middle and late periods of model training, the supervision controller turns to the random noise of the

TABLE I

TRAINING METHOD CONFIGURATION

Training methods	Prioritized experience replay	Random noise	Target network asynchronous update	Actor network supervised training
Method 1	√			
Method 2	√	√		
Method 3	√		√	
Method 4	√	√	√	√
Method 5	√		√	√

TABLE II

COMPARISON OF DIFFERENT TRAINING POLICIES

Training methods	Method 1	Method 2	Method 3	Method 4	Method 5
Cumulative rewards	13.01	13.37	13.43	15.02	15.67
Control steps	66	63	64	64	63

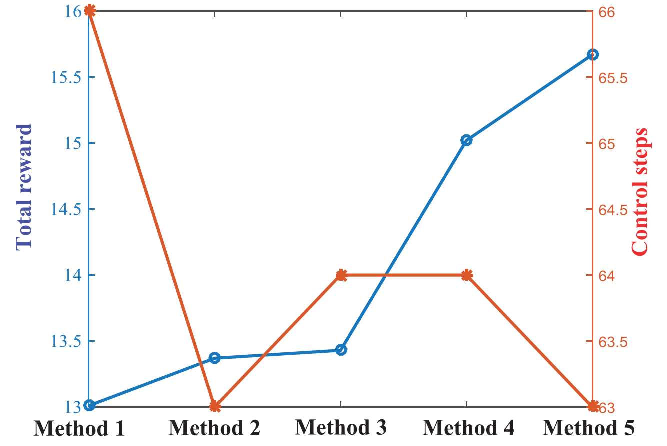

Fig. 6. Comparison of reinforcement learning with different methods. The left vertical coordinate (blue) is the value of the cumulative rewards. The right vertical coordinate (orange) is the number of the required control steps to achieve the control objective.

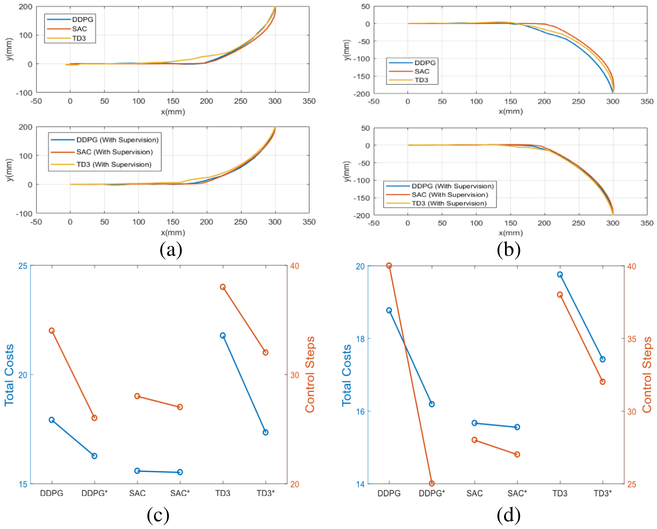

Fig. 7. Comparative results between DDPG, SAC, and TD3 and these methods with indirect supervision. (a) BUV path when the BUV initial position is $(300, 200)$. (b) BUV path when the BUV initial position is $(300, -200)$. (c) Results of total costs and control steps when the BUV initial position is $(300, 200)$. (d) Results of total costs and control steps when the BUV initial position is $(300, -200)$.

TABLE III

SIMULATION RESULTS OF TARGET TRACKING CONTROL UNDER DISTURBANCE

Experience No.	Initial position (mm)	R_{PID}	R_A	σ_{PID}	σ_A
Results under state disturbance a	(800,200)mm	-94.08	24.31	416	56
Results under state disturbance b	(800,-200)mm	-94.11	24.86	416	57
Results under control disturbance a	(800,200)mm	-31.38	24.35	289	58
Results under control disturbance b	(800,-200)mm	-33.58	24.31	293	58

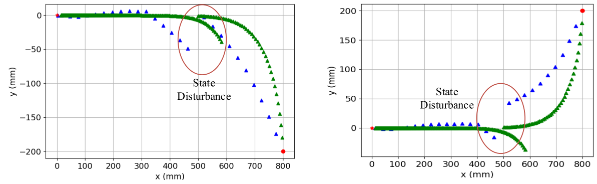

Fig. 8. Results of target tracking under state disturbance. (a) Results in the case of the initial position of the BUV at (800, 200) mm. (b) Results in the case of the initial position of the BUV at (800, −200) mm.

actor network. Therefore, the supervision controller provides sufficient generalization ability for the model. The added random noise becomes an uncertain factor, which may reduce the accuracy of the model.

In order to further verify the effectiveness of the proposed indirect supervision, the comparisons between DDPG, SAC, and TD3 and these methods with indirect supervision are given. Fig. 7 shows the results of target tracking when the target position is $(0, 0)$. As shown in Fig. 7(a) and (b), the DDPG, TD3, and SAC control algorithms are used to control the BUV to approach the target rapidly. However, as shown in Fig. 7(c) and (d), the target tracking control achieved by these algorithms with indirect supervision has fewer total costs and control steps. Indirect supervision is introduced to make the control algorithms converge faster.

In order to test the robustness of the designed policy network, state disturbance or control disturbance is added randomly during the simulation experiment. The control results are shown in Figs. 8 and 9. Table III shows the simulation results under state disturbance or control disturbance. The state disturbance is the Gaussian noise with a 50-mm standard deviation. The control disturbance is the Gaussian noise with a 0.5-mm standard deviation. The red circle indicates the position where the disturbance is added, and the apparent discontinuity occurs. The proposed policy network is barely affected by those disturbances. The cumulative rewards and control steps change a little compared to those when without disturbance. However, the PID controller is greatly affected by the disturbance. The cumulative rewards are greatly reduced. The control steps are greatly increased as well. The results show that the policy network has greater robustness against the disturbance.

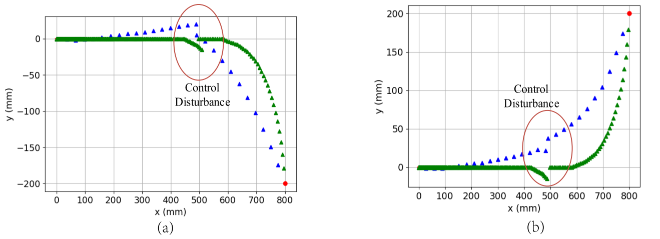

Fig. 9. Results of target tracking under control disturbance. (a) Results in the case of the initial position of the BUV at (800, 200) mm, (b) Results in the case of the initial position of the BUV at (800, −200) mm.

VII. Experiment and Analysis

Multiple sets of target tracking experiments are carried out to verify the effectiveness of the proposed method in spite of lacking theoretical convergence proof of the deep reinforcement learning. To the best of our knowledge, this is the first time show successful object tracing experiments on the real physical autonomous underwater robots by deep reinforcement learning. Reinforcement learning is guided by the optimal action-value function. It moves toward the gradient direction of the optimal action-value function. Thus, the outputs of the policy network have a high controlling effect. Moreover, the BUV inertia and flow turbulence always exist. As a result, the target is very likely to escape from the limited vision scope of the BUV. To address this, the control output of the policy network is optimized by the Gompertz growth curve. The effective precision of wave frequency of the propeller with undulatory fins is 0.1. The range of wave frequency is $[-1, 1]$. The positive value of frequency means that the propeller provides the forward force. The negative value of frequency means that the propeller provides the backward force. The wave frequency of the propeller is further optimized by

$$\begin{cases} \hat{f} = \text{round}\big(10ka^{bf}\big)/10, & \text{if } 0.1 \le f \le 1 \\ \hat{f} = \text{round}\big(-10ka^{b-f}\big)/10, & \text{if } -1 \le f \le -0.1 \\ \hat{f} = \text{round}(10f)/10, & \text{if } -0.1 \le f \le 0.1 \end{cases} \quad (21)$$

where k, a, and b are the parameters of the Gompertz growth curve. Based on practical engineering experience, the parameters are set as follows: $k = 0.0577, a = 1.5974$, and $b = 5$. round() represents the rounding off integral function, f represents the wave frequency of the biomimetic propeller generated by the policy network, and $\hat{f}$ represents the actual wave frequency applied to the propeller.

A. Static-Target Tracking Experiment

The BUV is controlled to track a static target and maintain the relative position between the BUV and the target stably. The target is detected and located through the binocular vision. Then, the position information is converted as the input of the policy network, and the wave frequency is generated. Figs. 10 and 11 show the snapshot sequences of the static-target tracking experiment. The target position in the BUV body-fixed frame is shown in Fig. 12. The objective is to keep the target in front of the BUV at 400 mm, i.e., $[x, y] = [400, 0]$ mm. The policy network generates wave frequencies and the Gompertz growth curve optimized wave frequencies, as shown in Figs. 13 and 14. The wave frequencies in Fig. 14 are applied to the propeller. Fig. 15 depicts the reward during the static-target tracking experiment. The experiment results indicate that the position error of the BUV is reduced under the control of the policy network. BUV enters a stable state after 25 s; then, it keeps a position relative to the target stably.

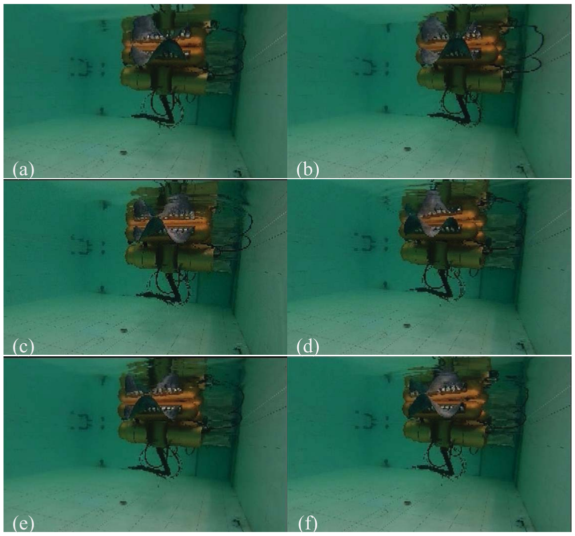

Fig. 10. Snapshot sequence of the static target tracking experiment by an underwater camera. (a)–(f) Time series subgraphs.

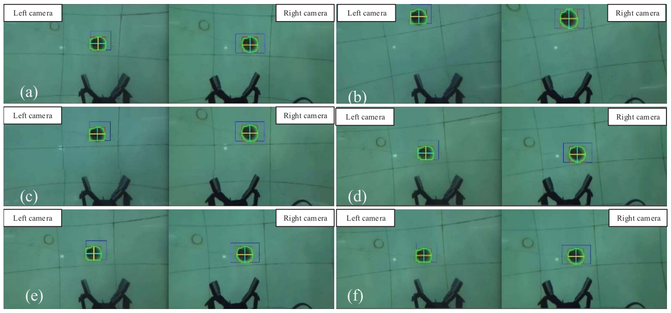

Fig. 11. Snapshot sequence of the static target tracking experiment by ZED binocular cameras of the BUV. (a)–(f) Time series subgraphs.

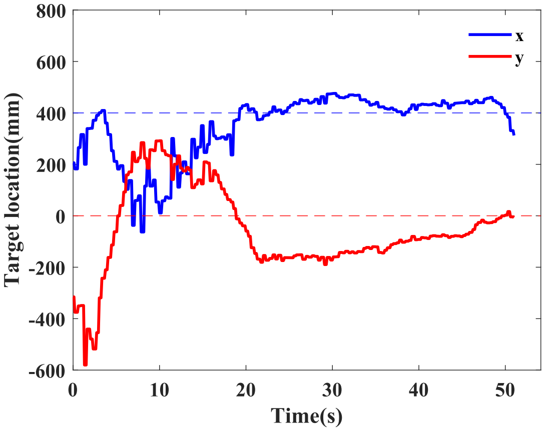

Fig. 12. Target position in the BUV body-fixed frame during static-target tracking experiment.

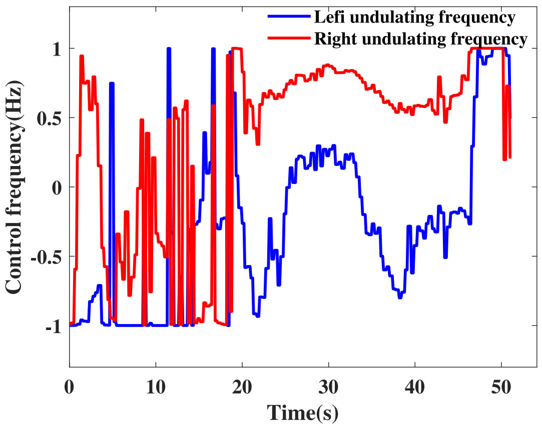

Fig. 13. Wave frequency generated by the policy network.

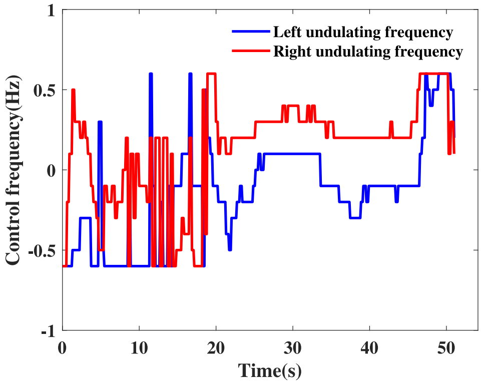

Fig. 14. Optimized wave frequency applied to the propeller with undulatory fins.

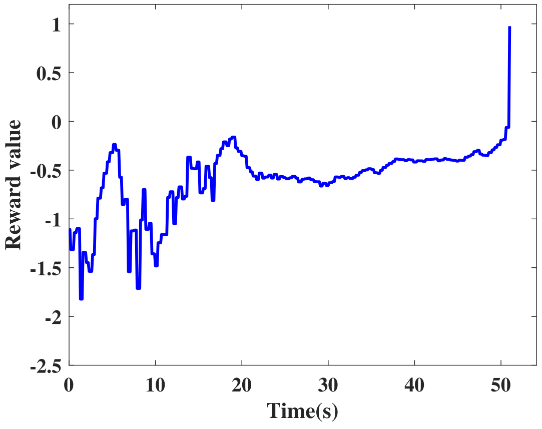

Fig. 15. Reward during static-target tracking experiment.

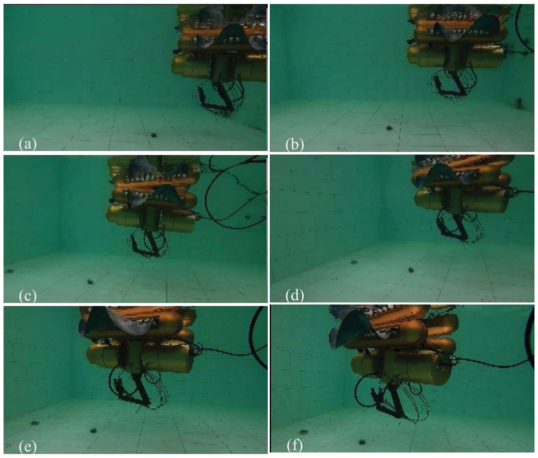

Fig. 16. Snapshot sequence of moving-target tracking experiment captured by an underwater camera. (a)–(f) Time series subgraphs.

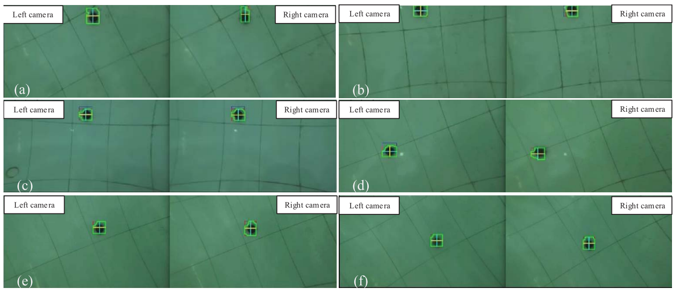

Fig. 17. Snapshot sequence of moving-target tracking experiment captured by ZED binocular cameras of the BUV. (a)–(f) Time series subgraphs.

When the reward turns to 1, as shown in Fig. 15, it means that the target position error is reduced in the range of 1.5 cm. This experiment proved the validation of the proposed method. It also proves the significance of the Gompertz growth curve in improving control stability and reducing fluctuation.

B. Moving-Target Tracking Experiment

The BUV is controlled by the proposed algorithms to track a moving target along a straight line. Moving-target tracking is realized. The experiment process is shown in Figs. 16 and 17. Fig. 16(a)–(d) shows the phase of the target tracking of the BUV. Fig. 16(e) and (f) shows the phase of position maintaining after the target stops moving. Fig. 18 shows the position of the moving target in the BUV body-fixed frame. The desired relative position is $[x, y] = [400, 0]$ mm. The paths indicate that the relative position in the x-direction is mostly over 400 mm during the tracking process. The robot needs to move forward to track the target. Therefore, the wave frequencies for both left and right fins are mostly positive, as shown in Fig. 19. When the BUV surpasses the target (the troughs of the x curve occur near 0, 65, and 90 s in Fig. 18), the wave frequencies need to be reversed to reduce the speed of the BUV. When the relatively position shows a large deviation in the left and right directions (the troughs and peaks of the y curve are observed near 50 and 75 s in Fig. 18), the yaw angle of the BUV needs to be adjusted to reduce the error. Hence, different wave frequencies are, respectively, applied to the left and right propellers, as shown in Fig. 18. When the target stops moving, the BUV tracks the target and maintains relative stability around the desired position with slight fluctuation. Fig. 20 shows the reward during the moving-target tracking experiment. The experimental results show that the proposed control method can achieve moving-target tracking effectively.

A stochastic moving target is tracked by the BUV to test the proposed algorithms. Fig. 21 shows the target position in

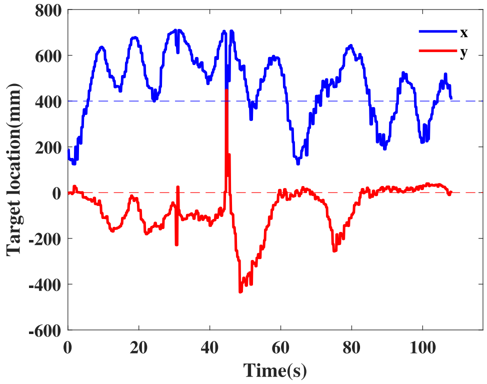

Fig. 18. Target position in the BUV body-fixed frame during moving-target tracking experiment.

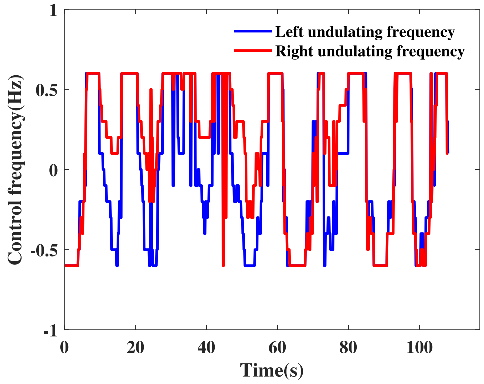

Fig. 19. Optimized control frequency of undulatory fins.

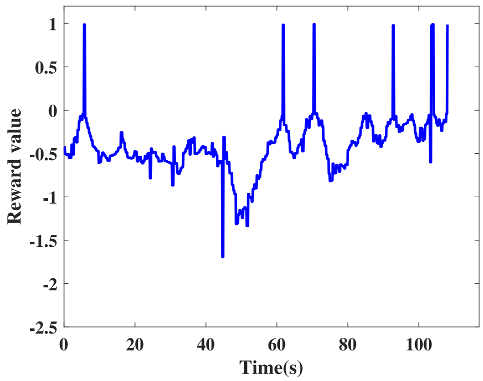

Fig. 20. Reward during moving-target tracking experiment.

the BUV body-fixed frame during this tracking experiment. As shown in Fig. 21, the target position has great fluctuation. The experiment results show that the BUV is still able to track the target continuously even if the target motion is stochastic.

C. Moving-Target Tracking Experiment Under Man-Made Disturbance

In order to further test the robustness of the proposed algorithms, when the BUV is tracking moving-target, man-made disturbances that are generated by randomly pushing the BUV with wooden sticks in different directions are exerted on the BUV during the tracking process. Fig. 22 shows the target position in the BUV body-fixed frame when BUV is subjected to man-made disturbance. This curve indicates that the target position relative to the robot has large fluctuation. Fig. 23 gives the reward during moving-target tracking under the influence of man-made disturbances. The experiment result shows that the trained policy network has a good control effect even when the BUV is subjected to man-made disturbance. The moving target can still be tracked continuously. Thus, the robustness of the proposed method is verified.

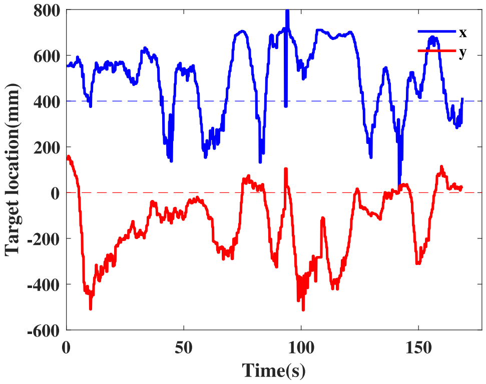

Fig. 21. Target position in robot body-fixed frame (during the randomly moving target tracing experiment).

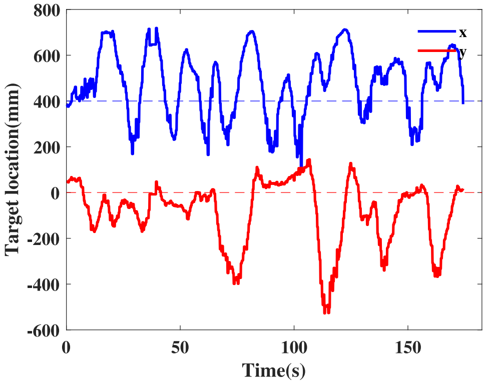

Fig. 22. Target position during moving-target tracking experiment under man-made disturbance.

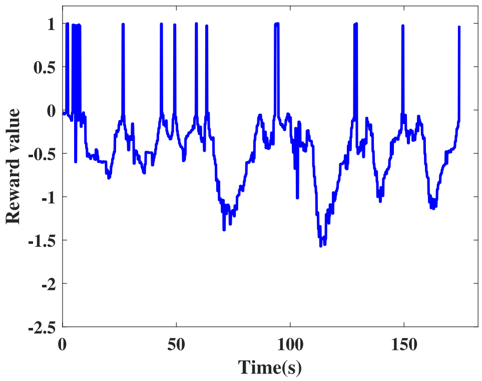

Fig. 23. Reward during moving-target tracking under man-made disturbance.

VIII. Conclusion

In this article, the deep DPG actor–critic reinforcement learning with supervision controller is proposed to address the target tracking control problem of the BUV. The system state and reward function of underwater target tracking control are provided. Training tricks are presented to improve the performance of reinforcement learning. Prioritized experience replay is utilized to eliminate the correlation in training data and improve the training effect. Actor network indirect supervision training is used to address the issues of low stability and slow convergence of reinforcement learning. Target network updating with different periods is used to update the actor-network with better evaluation from the critic-network. Random noise is added to the actor network to expand the exploration space. Finally, four groups of underwater object tracking experiments of the BUV using the actor–critic reinforcement learning with supervision controller have been conducted. It has been validated that the developed reinforcement learning is effective and practical.

References

[1] B. Kwak and J. Bae, "Toward fast and efficient mobility in aquatic environment: A robot with compliant swimming appendages inspired by a water beetle," *J. Bionic Eng.*, vol. 14, no. 2, pp. 260–271, Jun. 2017.

[2] F. E. Fish, "Advantages of natural propulsive systems," *Mar. Technol. Soc. J.*, vol. 47, no. 5, pp. 37–44, Sep. 2013.

[3] R. Du, Z. Li, K. Youcef-Toumi, and P. V. y Alvarado, *Robot Fish: Bioinspired Fishlike Underwater Robots*. Berlin, Germany: Springer, 2015.

[4] J. Yu, M. Tan, J. Chen, and J. Zhang, "A survey on CPG-inspired control models and system implementation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 3, pp. 441–456, Mar. 2014.

[5] J. Yu, Z. Wu, M. Wang, and M. Tan, "CPG network optimization for a biomimetic robotic fish via PSO," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 9, pp. 1962–1968, Sep. 2016.

[6] M. S. Triantafyllou and G. S. Triantafyllou, "An efficient swimming machine," *Sci. Amer.*, vol. 272, no. 3, pp. 64–70, Mar. 1995.

[7] C. Zhou and K. H. Low, "Design and locomotion control of a biomimetic underwater vehicle with fin propulsion," *IEEE/ASME Trans. Mechatronics*, vol. 17, no. 1, pp. 25–35, Feb. 2012.

[8] W. Wang and G. Xie, "Online high-precision probabilistic localization of robotic fish using visual and inertial cues," *IEEE Trans. Ind. Electron.*, vol. 62, no. 2, pp. 1113–1124, Feb. 2015.

[9] S. Verma and J.-X. Xu, "Data-assisted modeling and speed control of a robotic fish," *IEEE Trans. Ind. Electron.*, vol. 64, no. 5, pp. 4150–4157, May 2017.

[10] S. Zhang, Y. Qian, P. Liao, F. Qin, and J. Yang, "Design and control of an agile robotic fish with integrative biomimetic mechanisms," *IEEE/ASME Trans. Mechatronics*, vol. 21, no. 4, pp. 1846–1857, Aug. 2016.

[11] R. W. Blake, "Swimming in the electric eels and knifefishes," *Can. J. Zool.*, vol. 61, no. 6, pp. 1432–1441, Jun. 1983.

[12] M. Sfakiotakis, D. M. Lane, and J. B. C. Davies, "Review of fish swimming modes for aquatic locomotion," *IEEE J. Ocean. Eng.*, vol. 24, no. 2, pp. 237–252, Apr. 1999.

[13] R. Wang, S. Wang, Y. Wang, C. Tang, and M. Tan, "Three-dimensional helical path following of an underwater biomimetic vehicle-manipulator system," *IEEE J. Ocean. Eng.*, vol. 43, no. 2 pp. 391–401, Apr. 2018.

[14] S. Wang, Y. Wang, Q. Wei, M. Tan, and J. Yu, "A bio-inspired robot with undulatory fins and its control methods," *IEEE/ASME Trans. Mechatronics*, vol. 22, no. 1, pp. 206–216, Feb. 2017.

[15] J. Zhang, Y. Bai, S. Zhai, and D. Gao, "Numerical study on vortex structure of undulating fins in stationary water," *Ocean Eng.*, vol. 187, Sep. 2019, Art. no. 106166.

[16] H. Liu and O. Curet, "Swimming performance of a bio-inspired robotic vessel with undulating fin propulsion," *Bioinspiration Biomimetics*, vol. 13, no. 5, Jul. 2018, Art. no. 056006.

[17] L. Lin, H. Xie, D. Zhang, and L. Shen, "Supervised neural Q-learning based motion control for bionic underwater robots," *J. Bionic Eng.*, vol. 7, pp. 177–184, Sep. 2010.

[18] M. Zhang and H. H. T. Liu, "Game-theoretical persistent tracking of a moving target using a unicycle-type mobile vehicle," *IEEE Trans. Ind. Electron.*, vol. 61, no. 11, pp. 6222–6233, Nov. 2014.

[19] R.-J. Wai and Y.-W. Lin, "Adaptive moving-target tracking control of a vision-based mobile robot via a dynamic Petri recurrent fuzzy neural network," *IEEE Trans. Fuzzy Syst.*, vol. 21, no. 4, pp. 688–701, Aug. 2013.

[20] L. Zhou and P. Tokekar, "Active target tracking with self-triggered communications in multi-robot teams," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 3, pp. 1085–1096, Jul. 2019.

[21] L. Huang, "Control approach for tracking a moving target by a wheeled mobile robot with limited velocities," *IET Control Theory Appl.*, vol. 3, no. 12, pp. 1565–1577, Dec. 2009.

[22] D. Chwa, "Robust distance-based tracking control of wheeled mobile robots using vision sensors in the presence of kinematic disturbances," *IEEE Trans. Ind. Electron.*, vol. 63, no. 10, pp. 6172–6183, Oct. 2016.

[23] H. Yang, X. Fan, P. Shi, and C. Hua, "Nonlinear control for tracking and obstacle avoidance of a wheeled mobile robot with nonholonomic constraint," *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 2, pp. 741–746, Mar. 2016.

[24] D. Yuan, D. W. C. Ho, and G.-P. Jiang, "An adaptive primal-dual subgradient algorithm for online distributed constrained optimization," *IEEE Trans. Cybern.*, vol. 48, no. 11, pp. 3045–3055, Nov. 2018.

[25] R. P. A. Gil, Z. C. Johanyak, and T. Kovacs, "Surrogate model based optimization of traffic lights cycles and green period ratios using microscopic simulation and fuzzy rule interpolation," *Int. J. Artif. Intell.*, vol. 16, no. 1, pp. 20–40, Mar. 2018.

[26] S. Preitl, R.-E. Precup, Z. Preitl, S. Vaivoda, S. Kilyeni, and J. K. Tar, "Iterative feedback and learning control. Servo systems applications," *IFAC Proc. Volumes*, vol. 40, no. 8, pp. 16–27, 2007.

[27] K. K. Tan, S. Zhao, and J. X. Xu, "Online automatic tuning of a proportional integral derivative controller based on an iterative learning control approach," *IET Control Theory Appl.*, vol. 1, no. 1, pp. 90–96, Jan. 2007.

[28] B. Kiumarsi, K. G. Vamvoudakis, H. Modares, and F. L. Lewis, "Optimal and autonomous control using reinforcement learning: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 6, pp. 2042–2062, Jun. 2018.

[29] Y.-J. Liu, S. Li, S. Tong, and C. L. P. Chen, "Adaptive reinforcement learning control based on neural approximation for nonlinear discrete-time systems with unknown nonaffine dead-zone input," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 1, pp. 295–305, Jan. 2019.

[30] J. Qin, M. Li, Y. Shi, Q. Ma, and W. X. Zheng, "Optimal synchronization control of multiagent systems with input saturation via off-policy reinforcement learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 1, pp. 85–96, Jan. 2019.

[31] H. Wu, S. Song, K. You, and C. Wu, "Depth control of model-free AUVs via reinforcement learning," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 49, no. 12, pp. 2499–2510, Dec. 2019, doi: 10.1109/TSMC.2017.2785794.

[32] W. Shi, S. Song, C. Wu, and C. L. P. Chen, "Multi pseudo Q-learning-based deterministic policy gradient for tracking control of autonomous underwater vehicles," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 12, pp. 3534–3546, Dec. 2019, doi: 10.1109/TNNLS.2018.2884797.

[33] R. Cui, C. Yang, Y. Li, and S. Sharma, "Adaptive neural network control of AUVs with control input nonlinearities using reinforcement learning," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 6, pp. 1019–1029, Jun. 2017.

[34] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*. [Online]. Available: http://arxiv.org/abs/1509.02971

[35] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2000, pp. 1057–1063.

[36] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 387–395.

[37] L. J. Lin, "Self-improving reactive agents based on reinforcement learning, planning and teaching," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 293–321, 1992.

[38] F. Fathinezhad, V. Derhami, and M. Rezaeian, "Supervised fuzzy reinforcement learning for robot navigation," *Appl. Soft Comput.*, vol. 40, pp. 33–41, Mar. 2016.

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

[40] S. Raschka and V. Mirjalili, *Python Machine Learning: Machine Learning and Deep Learning With Python, Scikit-Learn, and TensorFlow 2*. Birmingham, U.K.: Packt, 2019.

[41] S. Raschka and V. Mirjalili, "Reinforcement learning for control: Performance, stability, and deep approximators," *Annu. Rev. Control*, vol. 46, pp. 8–28, Oct. 2018.

Yu Wang (Member, IEEE) received the B.E. degree in automation from the Beijing Institute of Technology, Beijing, China, in July 2011, and the Ph.D. degree in control theory and control engineering from the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, in July 2016.

He is currently an Associate Professor with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences. His research interests include intelligent control, robotics, and biomimetic robots.

Chong Tang received the B.E. degree from Northwestern Polytechnical University, Xi'an, China, in July 2014, and the Ph.D. degree in control theory and control engineering from the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China, in July 2019.

He currently holds a post-doctoral position with the NUCTECH Company Ltd., Beijing, and the Department of Computer Science and Technology, Tsinghua University, Beijing. His research interests include robotics, object and text detection, reinforcement learning.

Shuo Wang (Member, IEEE) received the B.E. degree in electrical engineering from the Shenyang Architectural and Civil Engineering Institute, Shenyang, China, in 1995, the M.E. degree in industrial automation from Northeastern University, Shenyang, in 1998, and the Ph.D. degree in control theory and control engineering from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2001.

He is currently a Professor with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences and the Center for Excellence in Brain Science and Intelligence Technology, Chinese Academy of Sciences, and also serves with the University of Chinese Academy of Sciences, Beijing. His research interests include biomimetic robots, underwater robots, and multirobot systems.

Long Cheng (Senior Member, IEEE) received the B.S. degree in control engineering from Nankai University, Tianjin, China, in July 2004, and the Ph.D. degree in control theory and control engineering from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in July 2009.

He is currently a Full Professor with the Institute of Automation, Chinese Academy of Sciences. He is also an Adjunct Professor with the University of Chinese Academy of Sciences, Beijing. His current research interests include rehabilitation robots, intelligent control, and neural networks.

Prof. Cheng was a recipient of the IEEE Transactions on Neural Networks Outstanding Paper Award from the IEEE Computational Intelligence Society, the Aharon Katzir Young Investigator Award from the International Neural Networks Society, and the Young Researcher Award from the Asian Pacific Neural Networks Society. He is also serving as an Associate Editor/Editorial Board Member for the IEEE TRANSACTIONS ON CYBERNETICS, *Neural Processing Letters*, *Neurocomputing*, *International Journal of Systems Science*, and *Acta Automatica Sinica*.

Rui Wang received the B.E. degree in automation from the Beijing Institute of Technology, Beijing, China, in 2013, and the Ph.D. degree in control theory and control engineering from the Institute of Automation, Chinese Academy of Sciences, Beijing, in 2018.

He is currently an Assistant Professor with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences. His research interests include intelligent control, robotics, underwater robots, and biomimetic robots.

Min Tan received the B.E. degree from Tsinghua University, Beijing, China, in 1986, and the Ph.D. degree in control theory and control engineering from the Institute of Automation, Chinese Academy of Sciences, Beijing, in 1990.

He is currently a Professor with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences. His research interests include advanced robot control, biomimetic robot, and multirobot systems.

Zengguang Hou (Fellow, IEEE) received the B.E. and M.E. degrees in electrical engineering from Yanshan University (formerly North-East Heavy Machinery Institute), Qinhuangdao, China, in 1991 and 1993, respectively, and the Ph.D. degree in electrical engineering from the Beijing Institute of Technology, Beijing, China, in 1997.

From May 1997 to June 1999, he was a Post-Doctoral Research Fellow with the Key Laboratory of Systems and Control, Institute of Systems Science, Chinese Academy of Sciences, Beijing. He was a Research Assistant with The Hong Kong Polytechnic University, Hong Kong, from May 2000 to January 2001. From July 1999 to May 2004, he was an Associate Professor with the Institute of Automation, Chinese Academy of Sciences, where he has been a Full Professor since June 2004. From September 2003 to October 2004, he was a Visiting Professor with the Intelligent Systems Research Laboratory, College of Engineering, University of Saskatchewan, Saskatoon, SK, Canada. He is currently a Professor and the Deputy Director of the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences. His research interests include neural networks, robotics, and intelligent systems.

Dr. Hou is an Editorial Board Member of *Neural Networks*. He is also the Chair of the Adaptive Dynamic Programming and Reinforcement Learning Technical Committee and the Neural Network Technical Committee of the Computational Intelligence Society (CIS). He was an Associate Editor of the *IEEE Computational Intelligence Magazine* and the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS. He is also an Associate Editor of the IEEE TRANSACTIONS ON CYBERNETICS, *ACTA Automatica Sinica*, and so on.