Proceedings of the 40th Chinese Control Conference
July 26-28, 2021, Shanghai, China

Online Optimization of Normalized CPGs for a Multi-Joint Robotic Fish

Ru Tong[1,2], Zhengxing Wu[2], Jian Wang[1,2], Min Tan[2], Junzhi Yu[2,3]*

1. School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China

2. State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China

3. Dept. Adv. Manuf. Robot., BIC-ESAT, College of Engineering, Peking University, Beijing 100871, China
E-mail: tongru2019@ia.ac.cn, zhengxing.wu@ia.ac.cn, wangjian2016@ia.ac.cn, min.tan@ia.ac.cn, junzhi.yu@ia.ac.cn

Abstract: As a popular control rhythm of the multi-joint robotic fish, Center Pattern Generators (CPGs) plays an important role for motion performance. However, its optimal parameters are tough to seek through traditional methods. In order to address this problem, we propose an online optimization method for CPG parameters, including a novel normalized CPGs (N-CPGs) and a learning-based optimization algorithm. Via N-CPGs, the network parameters can be fully decoupled, which provides a great convenience for model parameter optimization. In particular, by applying the established dynamic model of the robotic fish, we use the deep Q network (DQN) to optimize the N-CPGs, aiming at improving the speed performance. Finally, extensive simulation results verify the effectiveness of proposed method, laying a solid foundation for real-time online control optimization of versatile motion modes in complex application scenarios.

Key Words: Biomimetic, robotic fish, motion optimization, Central Pattern Generator (CPG), online optimization.

1 Introduction

Inspired by the natural underwater creatures, biomimetic robotic fish has attracted extensive attention of researchers due to the low noise and high maneuverability, thereby shows great potential in the field of future marine operations. From the point of the driving mechanism, the robotic fish propelled by Body and/or Caudal Fin (BCF) mode can be divided into three categories [1]: the single joint design, the multi-joint design, and special mechanical structure (such as smart material-based robot fish and soft-body robot fish). On one hand, the mechanical design of robotic fish with single joint is relatively simple, yet its propulsion efficiency is low. On the other hand, there are many difficulties in controlling the special robotic fish since the efficient and mature prototypes are hard to design. Unlike the previous two kinds, multi-joint robotic fish not only can more closely resemble the fish's flapping compared with single joint ones, but also can be well designed and controlled. From the structural design to advanced algorithms such as path tracking, the multi-joint robotic fish has been extensively studied [2–4].

Biological research indicates that fish swing rhythms are produced by central pattern generators (CPGs) in neuronal tissues, and the corresponding oscillation network can embody this abstract rhythm [5]. In order to stimulate the potential of the multi-joint mechanism and make the robotic fish swim more elegantly, CPG is widely used in the control of the bionic robotic fish [6–8]. Although the CPG greatly simplifies the complexity of robotic fish control, its parameters are hard to determined due to the multiple joints. In order to obtain satisfactory CPG parameters, extensive trial and error may be required by experienced researchers. This means that not only a lot of time will be consumed, but also there is no guarantee that manually adjusted parameters can work in the changing environment. In view of this problem, we focus on the CPG parameter optimization problem, and propose a reinforcement learning algorithm that trains robotic fish to "intelligently" learn to online optimize CPG parameters in changing environment.

In previous researches for the CPG parameter optimization of multi-joint robotic fish, Particle Swarm Optimization (PSO) was adopted [9], the notion of which lies in the collaboration and information sharing between individuals of a group to find the optimal solution. By integrating the PSO algorithm into the simulated dynamics model of the robotic fish, the results can be iteratively obtained. Further, there are four steps for PSO algorithm to optimize CPG parameters [10]. First, a kinetic model should be established. Second, through identifying the kinetics model parameters with experimental data. The model can be sufficient to simulate the actual swimming of the robotic fish. Next, the PSO algorithm will be executed to obtain the optimal CPG parameters with the goal of optimizing a certain performance. Finally, the optimized parameters are verified on the biomimetic robotic fish platform.

Although the PSO algorithm has the advantages of easy implementation and good global optimization capabilities, there are some serious shortcomings when the PSO algorithm is used for the biomimetic robot parameters optimizing problem. First, the optimization process is performed offline, which greatly hinders the practical applications of the robotic fish. Second, due to the reliance on the simulation environment, PSO cannot be implemented in the robot entity. Finally, after obtaining the optimal parameters, the robot cannot adaptively adjust the CPG parameters in the changing ocean currents through the PSO algorithm.

Inspired by machine learning, this paper designs a Reinforcement Learning (RL) algorithm to make the robotic fish obtain the intelligence of improving CPG parameters online. The reward in the algorithm is set according to the expected swimming performance, and some prior knowledge about CPG parameters is used to constrain the action space, which

This work was supported by National Natural Science Foundation of China (Grant Nos. 61725305, U1909206, 62033013, 62022090) and S&T Program of Hebei (Grant No. F2020203037).

speeds up the training of RL algorithm. In this work, the swimming performances we expect are high forward swimming speed and small head swing amplitude of robotic fish. More importantly, we particularly design a novel normalized CPGs (N-CPGs), which makes parameter modification more robust and greatly contributes to model parameter optimization. The learning results include a set of optimized CPG parameters and a RL-based intelligent Agent (RL-Agent) capable of optimizing CPG parameters through feedback sensor data. This set of optimized parameters is used as the startup parameters of the robot platform, and the agent is loaded as the initial intelligent algorithm of the robot. Based on this, the RL-Agent can continue to learn in the aquatic experiments of the robot, while the CPG parameters can be adaptively optimized online.

The remainder of this paper is structured as follows. The prototype and hydrodynamic model of robotic pike is detailed in Section II. The proposed normalized CPGs (N-CPGs) is presented in Section III. Optimization algorithm based on RL is illustrated in Section IV. Simulation results and analysis are provided in Section V. Finally, concluding remarks are offered in Section VI.

2 Robotic Prototype and Hydrodynamic Model

2.1 Overview of Robotic Prototype

The biomimetic robotic pike was developed in our previous work [11]. The robotic pike consists of a four-joint tail, a hard-shelled head and two pectoral fins with 2 degrees of rotational freedom. Its mechanical structure and prototype are illustrated in Fig. 1. The tail joints are driven by four servomotors. The control signal of the servomotors comes from the STM32 microcontroller located at its head. The microcontroller obtains the motion information of the robotic pike from the IMU module and the depth sensor, and communicates with the experimental console through the radio frequency module. This robotic fish can not only obtain comprehensive sensor information, but also can swim flexibly with the cooperation of the tail joints. Its streamlined body shape and flexible tail fin give it great swimming potential. The real-time communication and control capabilities provide conditions for online CPG parameter optimization. In the following parts, taking this kind of multi-joint robotic fish as the research object, we will illustrate the design process of online CPG parameter optimization algorithm.

2.2 Hydrodynamic Model

In this part, we will establish a dynamic model for the robotic pike. The simulation environment based on the dynamic model is set to train the RL-Agent. It is worth noting that proposed algorithms are not limited to the simulation environment and can be exported to the robotic pike environment, which will be further verified later.

With the focus on the tail's motion and force, we can consider the multi-joint robotic fish as a multi-link mechanism. From the point of kinematic analysis, the head speed can be calculated by the Newton–Euler equation in the simulation environment. Then the kinematic data of each tail joint are calculated by the converting matrix of the coordinate system of the multi-link. The recursive relationship of the velocity

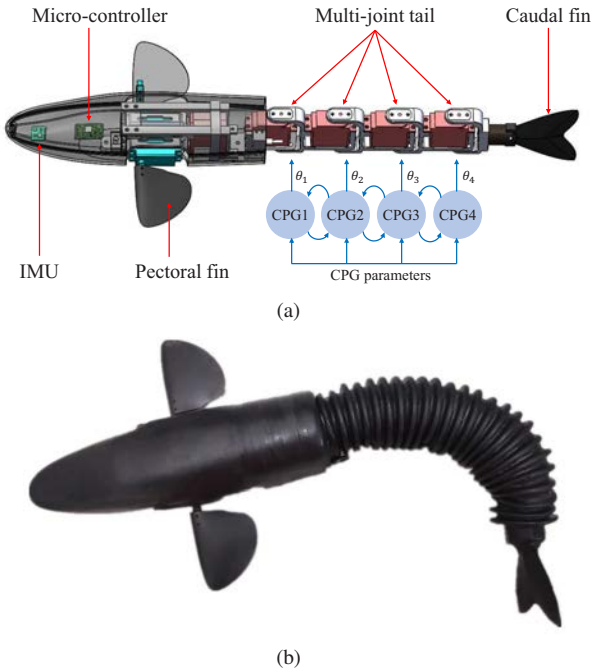

Fig. 1: Overview of the robotic prototype. (a) Schematic design. (b) Robotic prototype.

vector is as formula (1) shown.

$$\begin{aligned} \boldsymbol{V}_i &= \begin{bmatrix} \boldsymbol{U}_i \\ \boldsymbol{\Omega}_i \end{bmatrix} = \boldsymbol{H}_{i-1}^{i} \boldsymbol{V}_{i-1} + \dot{\theta}_i \boldsymbol{K}_i, \\ \boldsymbol{H}_{i-1}^{i} &= \begin{bmatrix} \boldsymbol{R}_{i-1}^{i} & \boldsymbol{R}_{i-1}^{i} \widehat{\boldsymbol{P}}_{i-1}^{i} \\ 0_{3\times 3} & \boldsymbol{R}_{i-1}^{i} \end{bmatrix}, \boldsymbol{K}_i = \begin{bmatrix} \boldsymbol{0}_{3\times 1} \\ \boldsymbol{k}_i \end{bmatrix} \\ \boldsymbol{R}_i^{i-1} &= \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 \\ \sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{P}_i^{i-1} = \begin{bmatrix} l_{i-1} \\ 0 \\ 0 \end{bmatrix} \end{aligned} \tag{1}$$

where $\boldsymbol{H}_{i-1}^{i}$, $\boldsymbol{R}_i^{i-1}$ and $\boldsymbol{P}_i^{i-1}$ are the converting matrix, rotation matrix and position vector, respectively; $\boldsymbol{V}_i$ is the i-th joint's velocity vector; θ_i is the i-th joint angle; $\boldsymbol{k}_i$ is the vertical upward unit vector at each joint.

With regard to dynamic analysis, the thrust of the robotic fish is mainly generated at the last joint of the tail. The force can be converted forward in turn through the coordinate system converting matrix of the multi-link, and finally the force of the head can be obtained. Thus, the kinematic data such as speed and attitude can be computed by Newton–Euler law. The recurring relationship of the interaction forces between adjacent joints is as follows:

$$\boldsymbol{G}_{i+1,i}^{i} = \boldsymbol{H}_{i+1}^{i} \boldsymbol{G}_{i+1,i}^{i+1} = \boldsymbol{H}_{i+1}^{i} \boldsymbol{G}_{i,i+1}^{i+1} \tag{2}$$

where the $\boldsymbol{G}_{i,j}^{k}$ is the representation of the force of the i-th joint on the j-th joint in the k-th joint coordinate system.

The force analysis of each joint is obtained by hydrodynamic theory analysis [12]. Each joint is mainly subject to lift $\boldsymbol{L}_i$ and resistance $\boldsymbol{D}_i$ of the tail fin, additional mass forces $\boldsymbol{f}_{ad,i}$, resistance $\boldsymbol{f}_{dr,i}$ from water, Corioli forces and conceding forces $\boldsymbol{\gamma}_i$. According to Newton–Euler equation, the relationship between the force and acceleration of the joints is as follows:

$$\boldsymbol{G}_{i-1,i}^{i} - \boldsymbol{G}_{i+1,i}^{i} + \boldsymbol{L}_i + \boldsymbol{D}_i + \boldsymbol{f}_{ad,i} + \boldsymbol{f}_{dr,i} = \boldsymbol{M}_i \boldsymbol{V}_i + \boldsymbol{\gamma}_i \tag{3}$$

Fig2

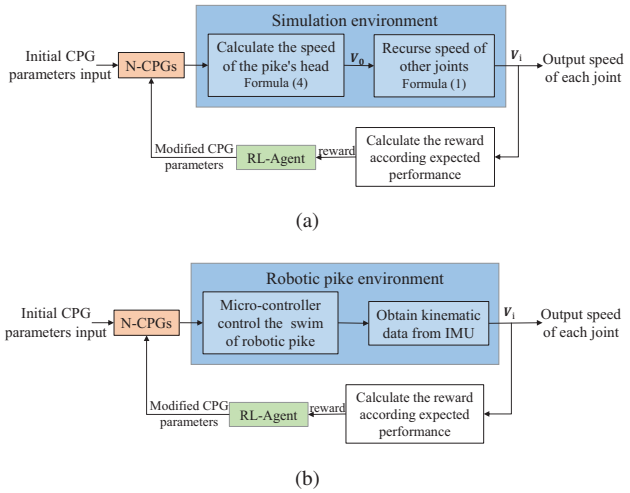

Fig. 2: Schematic diagram of data flow in robotic pike environment. (a) Hydrodynamic model, i.e., simulation environment. (b) Robotic pike environment.

where $\boldsymbol{f}_{ad,i} = -\boldsymbol{M}_{ad,i}\boldsymbol{V}_i - \boldsymbol{\gamma}_{ad,i}$; $\boldsymbol{f}_{ad,i}$ is the additional mass of each joint; $\boldsymbol{\gamma}_{ad,i}$ is the Corioli forces and conceding forces caused by additional mass.

The speed is recursively from the head to the tail, and the force is recursively from the tail to the head. In the head, the speed and the force are connected by formula (1–3) so that the explicit dynamic equation is obtained as shown below.

$$\begin{aligned}\dot{\mathbf{V}}_0 = &-\left[\sum_{i=0}^{n}\left(\mathbf{H}_i^0\left(\mathbf{M_i}+\mathbf{M}_{ad,i}\right)\mathbf{H}_0^i\right)\right]^{-1}\\ &\left[\sum_{i=1}^{n}\mathbf{H}_i^0\left(\dot{\mathbf{H}}_{i-1}^i\mathbf{V}_{i-1}+\ddot{\theta}_i\mathbf{K}_i\right)\sum_{j=i}^{n}\mathbf{H}_j^i\left(\mathbf{M}_j+\mathbf{M}_{ad,j}\right)\mathbf{H}_i^j\right.\\ &\left.-\sum_{i=0}^{n}\mathbf{H}_i^0\left(\mathbf{L}_i+\mathbf{D}_i-\gamma_{ad,i}-\gamma_i+\mathbf{f}_{dr,i}\right)\right].\end{aligned} \tag{4}$$

Note that the hydrodynamic parameters in the model have been identified through aquatic experiments, and more details on dynamic model can be found in [7].

In order to clearly demonstrate how to use the dynamic model to build a simulation environment and how the RL algorithm adjusts the CPG parameters to train the RL-Agent, the calculation process and data flow are summarized into a block diagram, as shown in Fig. 2(a). To better illustrate that the proposed algorithm is not limited to the simulation environment, we present a data flow block diagram in the robotic pike environment (Fig. 2(b)). For robotic pike environment, the head speed is obtained from the IMU sensor directly.

Until now, the multi-joint robotic pike environment has been established. Before training the RL-Agent in the created environment, we propose a normalized CPGs (N-CPGs), which makes it easier and more stable to modify the CPG parameters during training.

3 Normalized CPGs (N-CPGs)

The construction of the CPG network depends on the oscillator. In addition to the complex biological-neurons-based CPG model, the two common models based on nonlinear oscillators are the Kuramoto phase oscillator and the Hopf oscillator. The limit-cycle-based CPG model, built with the Hopf oscillator, can quickly converge to the limit cycle under the appropriate learning rate α [13]. This CPG model greatly reduces the parameters number and the complexity of the oscillator by building linear coupling between the nearest adjacent units through a simple chain topology. The CPGs network used for robotic pike control connects adjacent oscillators directly, as shown in Fig. 1(a). Each oscillator corresponds to a tail joint.

The proposed N-CPGs is based on the Hopf CPGs, and the topology of the N-CPGs controlling the robotic pike is the same as that of the CPGs. Firstly, we give the original mathematical model of Hopf CPGs, as shown in (5-7).

$$\begin{aligned}\begin{bmatrix}\dot{x}_i\\ \dot{y}_i\end{bmatrix} = &\begin{bmatrix}\alpha\left(A_i-r_i^2\right) & -\omega_i\\ \omega_i & \alpha\left(A_i-r_i^2\right)\end{bmatrix}\begin{bmatrix}x_i\\ y_i\end{bmatrix}\\ &+\sum_{j=i\pm1}h\begin{bmatrix}\cos\left(\Delta\varphi_{ij}\right) & -\sin\left(\Delta\varphi_{ij}\right)\\ \sin\left(\Delta\varphi_{ij}\right) & \cos\left(\Delta\varphi_{ij}\right)\end{bmatrix}\begin{bmatrix}x_j\\ y_j\end{bmatrix}\end{aligned} \tag{5}$$

$$\begin{bmatrix}x_i\\ y_i\end{bmatrix} = \begin{bmatrix}x_i\\ y_i\end{bmatrix} + \begin{bmatrix}\dot{x}_i\\ \dot{y}_i\end{bmatrix}\mathrm{d}t \tag{6}$$

$$\theta_i = y_i \tag{7}$$

where i refers to the each joint of robotic pike ($i=1,\ldots,4.$); x_i and y_i is CPGs' output; α is the learning rate; $r_i^2 = x_i^2+y_i^2$; A_i is the swing amplitude of each joint; θ_i is the joint angle; $\Delta\varphi_{ij}$ is the phase difference between joint i and joint j; ω_i is the angular frequency; $\mathrm{d}t$ is the control period; h is the coupling coefficient.

In the original CPGs model, the optimal learning rate α is different under different amplitudes. If both the chosen learning rate and the CPG parameter-amplitude are large, the output curve of the oscillator will oscillate strongly (Fig. 3(a)). On the contrary, if they are small, the output curve of the oscillator will converge slowly (Fig. 3(b)). It can be seen that the optimal learning rate of the CPGs network should not be fixed. Therefore, in the training process, the algorithm adjusts the amplitude while correspondingly modifying the learning rate, which is very cumbersome.

In addition, when the parameters (frequency or amplitude) of the CPGs network are drastically changed, the oscillator may produce a large $\dot{x}_i$ and $\dot{y}_i$ for quickly converging, so that the output joint angle curve may have a spike (Fig. 3(c)). During the long training process, CPGs may diverge due to drastic changes of CPG parameters at a certain uncontrollable moment (Fig. 3(d)). This divergence is undesirable for the RL-Agent training process.

In order to solve the problems in the CPGs network, we propose the N-CPGs. To put it simply, formula (5) is divided by the amplitude A to remove the amplitude gain, and the output amplitude of the oscillator is normalized to unit 1, so as to obtain the normalized formula (8-11). Note that the gain of the amplitude will be compensated in the calculation of the joint angles. In addition, there are three improvements in N-CPGs model.

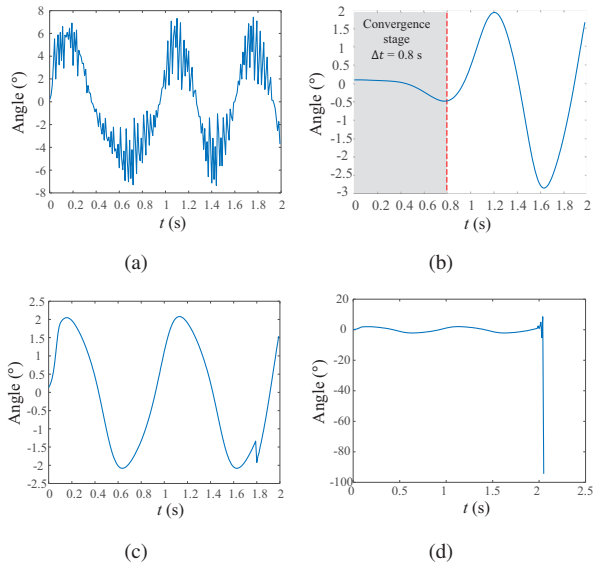

Fig. 3: Unexpected conditions of original Hopf CPGs. (a) Too Large learning rate. (b) Too small learning rate. (c) Spikes caused by drastic changes in CPG parameters. (d) Divergence caused by drastic changes in CPG parameters.

1) A constraint function CF (w_i) is added to constrain $\dot{x}_i$ and $\dot{y}_i$, as illustrated in (9). The function CF (w_i) is fitted by the maximum value of $\ddot{x}_i$ and $\ddot{y}_i$ at different angular frequencies
2) The output of the oscillator should be compensated for the amplitude gain before it outputs as the control quantity of the joint angles, as shown in (11).
3) When the parameter-amplitude is modified, it should be calculated according to (12), and other parameters can be modified directly.

$$\begin{bmatrix}\dot{x}_i\\ \dot{y}_i\end{bmatrix}=\begin{bmatrix}\alpha'\left(1-r_i^2\right) & -\omega_i\\ \omega_i & \alpha'\left(1-r_i^2\right)\end{bmatrix}\begin{bmatrix}x_i\\ y_i\end{bmatrix} + \sum_{j=i\pm1} h'\begin{bmatrix}\cos\left(\Delta\varphi_{ij}\right) & -\sin\left(\Delta\varphi_{ij}\right)\\ \sin\left(\Delta\varphi_{ij}\right) & \cos\left(\Delta\varphi_{ij}\right)\end{bmatrix}\begin{bmatrix}x_j\\ y_j\end{bmatrix} \tag{8}$$

$$\begin{bmatrix}\dot{x}_i\\ \dot{y}_i\end{bmatrix}\in\left[\begin{bmatrix}\dot{x}_{i,\text{old}}\\ \dot{y}_{i,\text{old}}\end{bmatrix}-\text{CF}\left(w_i\right),\begin{bmatrix}\dot{x}_{i,\text{old}}\\ \dot{y}_{i,\text{old}}\end{bmatrix}+\text{CF}\left(w_i\right)\right] \tag{9}$$

$$\begin{bmatrix}x_i\\ y_i\end{bmatrix}=\begin{bmatrix}x_i\\ y_i\end{bmatrix}+\begin{bmatrix}\dot{x}_i\\ \dot{y}_i\end{bmatrix}\mathrm{d}t \tag{10}$$

$$\theta_i = A_i \times y_i \tag{11}$$

where $\dot{x}_i$ and $\dot{y}_i$ is N-CPGs' output, whose amplitude is 1; α' is the learning rate; h' is the coupling coefficient.

$$\begin{bmatrix}x_i\\ y_i\end{bmatrix}=\begin{bmatrix}x_i\\ y_i\end{bmatrix}\div A_{i,\text{old}} \qquad A_i = A_{i,\text{new}} \tag{12}$$

The advantages of N-CPGs are reflected in the following four points.

1) The amplitude of the oscillator is normalized to 1, and the optimal learning rate that matches unit amplitude can be chosen to ensure that CPGs can transition to the next convergence state as quickly as possible when the CPG parameters are modified.
2) Depending on the chosen learning rate, the error between the actual amplitude and the set amplitude can be kept within a fixed range. By adjusting the learning rate according to control requirements, the magnitude of error can be controlled. In the case of α=20, the amplitude error is $(1.053-1)\div1\approx5.3\ \%$, as shown in Fig. 4.
3) The added constraint function enables CPGs to transition smoothly even when the parameters change drastically.
4) The effective amplitude modification method is proposed for the normalized model.

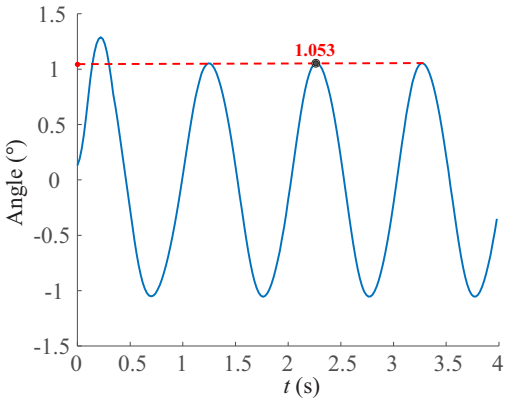

Fig. 4: Fixed amplitude error at learning rate α=20.

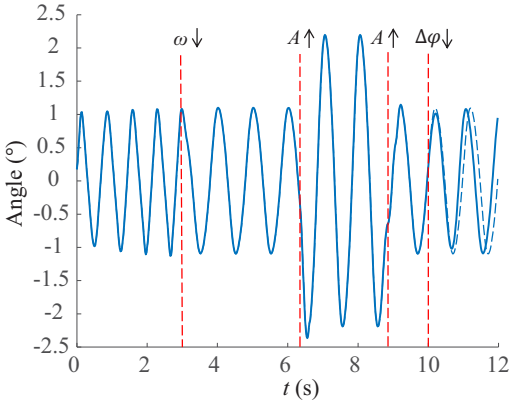

Fig. 5: Comprehensive performance of N-CPGs.

Fig. 5 comprehensively shows the ability of N-CPGs to adapt to changing parameters. Within 12 s of decreasing the frequency, increasing the amplitude, decreasing the amplitude, and reducing the phase difference sequentially, the curve can all be smoothly and quickly transitioned.

4 Optimization Algorithm Based on RL

The purpose of the optimization algorithm is to build an original RL-Agent, and obtain a set of optimized CPG parameters at the same time. The classic deep Q network (DQN) method is used as the framework of the reinforcement learning algorithm. DQN algorithm approximates the

action value function $Q(action, state)$ through a deep network. Thanks to the deep network's ability to fit complex functions, the learning problem of multi-dimensional actions space can be well solved. In this paper, a six-layer neural network serves as an approximator of the action value Q function. For the multi-joint robotic fish environment, the construction of the DQN algorithm has some special features. On the one hand, the action space can be constrained by the prior knowledge of CPG parameters. On the other hand, the globally optimal CPG parameters can be recorded throughout the training process and used to reset the CPG parameters in the next episode. Prior knowledge and global data can improve the convergence speed of training to a certain extent.

The learning algorithm will be introduced from five aspects.

- Action space: It is composed of 15 discrete actions. In detail, 8 actions control the increase or decrease of the amplitude of four joint angles, and 6 actions control the increase or decrease of the phase difference between the four joints, and the last action does not change any CPG parameters. The execution of the action is based on the prior knowledge of the robotic fish system on CPG parameters. The prior knowledge is reflected in that the amplitude of each joint increases from front to back, and that the phase differences between the joints are positive, that is, the phase of the next joint is behind the previous joint. Due to mechanical limit, the effective range of amplitude is $[0,\ 60°]$, and according the experience the range of phase difference is also $[0,\ 60°]$. A non-zero exploration rate ε is configured to accelerate the exploration of the action space.
- State space: It is an 8×1 vector, as shown in (13). $\overline{V}_x$ is the average speed in one period after the speed stabilizes.

$$State = \left[\boldsymbol{A}_{4\times1}, \boldsymbol{\Delta\varphi}_{3\times1}, \overline{V}_x\right] \tag{13}$$

- Reward function: Its design starts with the expected swimming performance. The desired performance in this paper is high swimming speed and small head swing. The small head swing is reflected in the small fluctuation range of the lateral velocity. As a result, the reward function is designed using variables forward speed V_x, and the speed interval V_{range} in one period. After several attempts, a feasible reward function design is shown in the (14). The meanings of the two parts of the reward function are "kinetic energy reward" and "fluctuation range penalty" respectively. Among (14), a and b are the weights of the two parts.

$$Reward = a \times \overline{V}_x^2 - b \times V_{range} \tag{14}$$

- Termination condition: It is mainly determined by two factors. On the one hand, it is expected that the trained agent can obtain optimized parameters within 30 executions of the action. On the other hand, it is never expected that the forward speed is negative. Therefore, when the speed reaches a negative value, a large penalty is given, and then the episode ends. When the parameter modification action is executed more than 30 times, the episode ends.

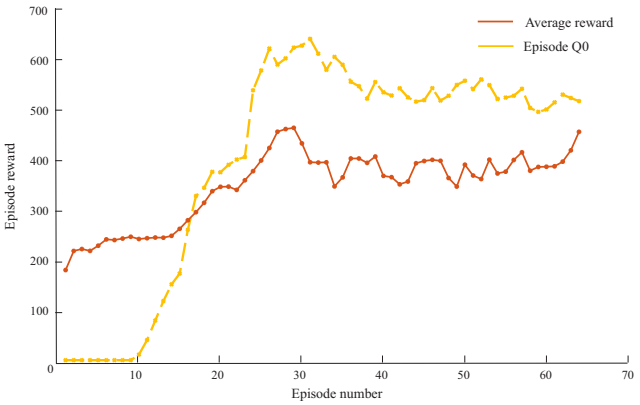

Fig. 6: Plot of training convergence curve.

- Environment reset: When the environment is reset, the CPG parameters are reset with the recorded global optimal parameters. The global optimal parameters are used as the initial state of the next episode, which can effectively speed up the training.

5 Training and Result

Within the algorithm framework presented in Section IV, RL-Agent is trained. According to experience, the initial CPG parameters are set to $\boldsymbol{A} = [10, 15, 20, 25], \boldsymbol{\Delta\varphi} = [30, 15, 15]$. The frequency of CPG is a fixed value 2π rad/s, that is, the tail flaps once per second. The weight a and b are set to 200 and 100, respectively. The exploration rate ε is set to 0.1.

After training for 60 episodes, the training curve is converged, as shown in Fig. 6. From the convergence curve, it can be seen that after training 30 episodes, the original RL-Agent and optimized CPG parameters are obtained.

In order highlight the effect of the optimized CPG parameters, we use the initial experience parameters before training and the optimized parameters after training to control the robotic pike. The curves of forward speed under two sets of parameters are illustrated in Fig. 7. Notice that the blue curve is the speed under the initial experience parameters, while the red curve is the speed under the optimized parameters. As can be seen, at the frequency of 1 HZ, the forward speed is increased from 0.30 m/s to 0.32 m/s and the head swing amplitude is reduced from 0.24 m/s to 0.14 m/s, which verifies the effectiveness of the optimization algorithm.

After obtaining the RL-Agent, we can load the RL-Agent on the robotic fish. The RL-Agent no longer needs the exploration of random actions, so the ε is set to zero. The optimized CPG parameters are used as startup parameters of robotic pike. The RL-Agent can online optimize the parameters through the sensor data and learn online to improve its intelligence. Through online learning during swimming, robotic fish gradually learns to adjust tail flapping parameters more robust and stable. As a result, no matter what environment the robotic fish swims in, it can be started up with the optimized optimal parameters.

It should be remarked that, the focus in this paper is the swimming speed performance, but the optimization algorithm is not limited to this indicator. By designing appropriate reward functions according to the expected performance, the optimization algorithm can be easily transplanted to other

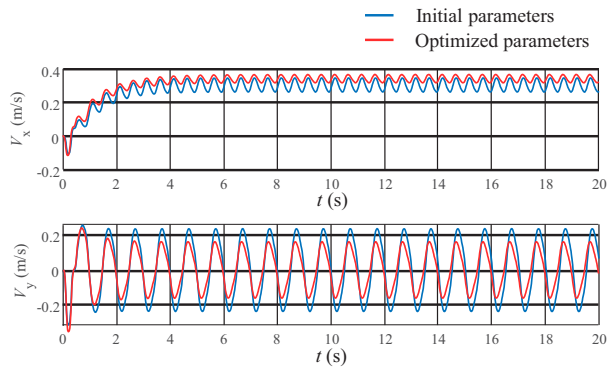

Fig. 7: Speed performance comparison under the CPG parameters before and after training.

tasks, such as high acceleration, quick turning, high energy efficiency, to name a few.

6 Conclusion and Future work

In this paper, we proposed a novel N-CPGs model for the robotic fish, based on which an online RL-based algorithm for optimizing the control parameters is presented. Firstly, a complete dynamic model of the multi-joint robotic fish has been established as the training environment. Further, by applying the N-CPGs, the parameter optimization is more convenient and robust than original Hopf CPGs model, which lays the foundation for the stability of training. Finally, the optimization algorithm based on DQN is designed for the built training environment. With the goal of improving speed performance, an RL-Agent and a set of optimized parameters are successfully obtained. A comparative simulation verifies the effectiveness and superiority of the proposed methods. This work sheds light on the intelligence of multi-joint robotic fish to adjust control parameters autonomously with the goal of improving motion performance.

In the future, we will improve the optimization performance of the algorithm through extensive experiments in real-world aquatic scenarios. Furthermore, the agent will learn the ability to optimize more swimming performance by perceiving more environmental information.

References

[1] R. Du, Z. Li, K. Youcef-Toumi, and P. V. Y Alvarado, *Robot Fish: Bio-Inspired Fishlike Underwater Robots*. Springer, 2015.

[2] K. Xu, R. Wang, and Y. Lu, "Structure design of multi-joint bionic robot based on Solidworks," *IOP Conference Series: Earth and Environmental Science*, vol. 526, no. 1. IOP Publishing, 012165, 2020.

[3] S. Howe, A. Duff, and H. Astley, "Comparing the turn performance of different motor control schemes in multi-link fish-inspired robots," *Bioinspiration & Biomimetics*, doi: 10.1088/1748-3190/abe7cc, 2021.

[4] S. Verma and J. Xu, "Analytic modeling for precise speed tracking of multilink robotic fish," *IEEE Transactions on Industrial Electronics*, 65(7): 5665–5672, 2017.

[5] A. J. Ijspeert, "Central pattern generators for locomotion control in animals and robots: A review," *Neural networks*, 21(4): 642–653, 2008.

[6] X. Niu, J. Xu, Q. Ren, and Q. Wang, "Locomotion learning for an anguilliform robotic fish using central pattern generator approach," *IEEE Transactions on Industrial Electronics*, 61(9): 4780–4787, 2013.

[7] J. Yu, M. Wang, Z. Su, M. Tan, and J. Zhang, "Dynamic modeling of a CPG-governed multijoint robotic fish," *Advanced Robotics*, 27(4): 275–285, 2013.

[8] D. Korkmaz, G. Ozmen Koca, G. Li, C. Bal, M. Ay, and Z. H. Akpolat, "Locomotion control of a biomimetic robotic fish based on closed loop sensory feedback CPG model," *Journal of Marine Engineering & Technology*, 2019: 1–13.

[9] S. Sengupta, S. Basak, and R. A. Peters, "Particle swarm optimization: A survey of historical and recent developments with hybridization perspectives," *Machine Learning and Knowledge Extraction*, 1(1): 157–191, 2019.

[10] M. Wang, H. Dong, X. Li, Y. Zhang, and J. Yu, "Control and optimization of a bionic robotic fish through a combination of CPG model and PSO," *Neurocomputing*, 337: 144–152, 2019.

[11] J. Yu and M. Tan, "Design and control of a multi-joint robotic fish," *Robot Fish*. Springer, 2015: 93–117.

[12] F. El-Hawary, *The Ocean Engineering Handbook*. CRC Press, 2000.

[13] X. Li, H. Liu, X. Wu, R. Li, and X. Wang, "Improved CPG model based on hopf oscillator for gait design of a new type of hexapod robot," in *Proceedings of International Conference on Intelligent Robotics and Applications*. Springer, 2019: 72–83.