Nonlinear Dyn (2018) 94:2715–2725
https://doi.org/10.1007/s11071-018-4520-7

ORIGINAL PAPER

An equilibrium-based learning approach with application to robotic fish

Xuefang Li · Qinyuan Ren · Jian-Xin Xu

Received: 21 July 2017 / Accepted: 10 August 2018 / Published online: 21 August 2018

Abstract In this work, we extend the concept of integral control to equilibrium-based learning control. As far as the plant reaches an equilibrium that deviates from the reference, a learning mechanism will update the control action. The new control action will drive the plant output to reach a new equilibrium that is closer the reference set-point. By applying fixed point theorem, we can prove the convergence of the controlled equilibrium to the reference set-point exponentially, where the plant dynamics can be generically nonlinear and non-affine. The only prior information required is a non-singular input–output gradient of the stabilized plant. As a real-time application, the proposed control method is applied to motion control of a tail-actuated robotic fish. To facilitate the controller design, the dynamical model of the robotic fish is established based on Newton's second law and Lighthill's small amplitude model. In the end, both simulations and experiments are conducted to illustrate the effectiveness of the proposed learning approach.

X. Li
Department of Electrical and Electronic Engineering, Imperial College London, London, UK
e-mail: xuefang.li@imperial.ac.uk

Q. Ren (✉)
College of Control Science and Engineering, Zhejiang University, Hangzhou, China
e-mail: latepat@gmail.com

J.-X. Xu
Department of Electrical and Computer Engineering, National University of Singapore, Singapore, Singapore
e-mail: elexujx@nus.edu.sg

Keywords Learning control · Robotic fish · Motion control · Turning control · Self-adaption

1 Introduction

Learning control is known to be efficient in dealing with repetitive control tasks. Compared to traditional control approaches, the basic idea of learning control is to utilize the information from previous trial or period so as to improve the control performance of the current trial or period. Generally, there are two main learning approaches, known as iterative learning control (ILC) [1–6] and repetitive control (RC) [7,8]. ILC aims at discontinuous operations and the learning updates are carried out from trial to trial. RC is applicable to systems operating continuously and the learning updates are carried out from period to period. Both ILC and RC aim at perfect tracking over the entire trial or period.

ILC is developed for systems that are able to complete some tasks over a fixed time interval and perform them repeatedly. The control objective of ILC is to achieve perfect tracking over the whole time interval, which relies on the strict resetting condition. In practical applications, however, most of the industrial processes operate continuously without resetting. In addition, majority of the control tasks are to investigate the step response of the control systems since knowing the step response of a dynamical system gives information on the stability of the system, and on its ability to reach one stationary state when starting from another. There-

fore, for systems operating continuously in time, ILC is no longer applicable.

In contrast to ILC, RC mainly focuses on the infinite time interval and is used specially in dealing with periodic signals, for example, tracking periodic reference or rejecting periodic disturbances. In RC, the command to be executed is a periodic function of time. There is no resetting of the system to the same initial condition before the start of the next period, and thus transients can propagate across periods. However, RC is actually a period-based pointwisely learning approach. For some of control processes, the control action is not able to update at every time instant but only updates once over an interval. One example is digital control in which the control action is updated once at the beginning of a sampling interval. Another example is the motion control of robotic fish. For a tail-actuated robotic fish prototype, the manipulable parameters are the undulatory frequency and amplitude of the caudal fin, which can only be updated at the beginning of each undulatory cycle. Within a undulatory cycle, the undulatory frequency and amplitude cannot be adjusted. Hence, from this point of view, RC is not applicable to systems whose control actions can only update once in an interval. This motivates us to explore new learning approaches.

In this work, our aim is to drive a plant output to reach a given reference set-point, where the plant dynamics is nonlinear. Without appropriate control, the plant will reach its equilibrium, but the equilibrium may deviate significantly from the reference set-point. From internal model theory, we need to incorporate an appropriate internal model which can alter the plant equilibrium and drive the equilibrium to reach the reference. Motivated by this idea, we extend the concept of integral control to equilibrium-based learning control. As far as the plant reaches an equilibrium that deviates from the reference, a learning mechanism will update the control action. The new control action will drive the plant to reach a new equilibrium that is getting closer the reference set-point. Thus, the learning controller plays the role as an internal model.

For a linear plant, it is well known that an integral controller can be incorporated in the control loop, which plays the role as an internal model. The integral action will ultimately eliminate the discrepancy between the actual plant output and the reference set-point, namely reach the equilibrium (steady state) that is consistent with the reference. For nonlinear plants, however, the analysis of the equilibrium status subject to integral control would be much more difficult and there lacks appropriate design methods. While for the proposed novel learning approach, we can prove the convergence of the controlled equilibrium to the reference set-point exponentially by applying the fixed point theorem, where the plant can be generically nonlinear and non-affine. The only prior information required for the stabilized plant is the non-singularity of input–output gradient.

As a real-time application, the proposed learning control algorithm will be applied to motion control of a tail-actuated robotic fish. Previous works reported on motion control of robotic fish mainly concern two aspects: (1) generate fish-like swimming gait in a robot and (2) drive a robotic fish to achieve a desired motion. The former explores producing coordinated movements of actuation components of a robotic fish, and the latter focuses on controlling the motion of whole fish body. In the past, there were several works reported on model-based feedback controllers for motion control of robotic fish or more general autonomous underwater vehicles (AUVs), such as [9–13]. However, the control performance of model-based controllers is highly dependent on the accuracy of the nonlinear dynamics that describes the interaction between robot and water. Actually, due to the complexity of hydrodynamics, precise modeling of a robotic fish dynamics is a challenging problem. To overcome the difficulty in accurate modeling, several partial model-based or model-free control design methods are investigated, such as fuzzy logic control [13–15], learning control [16,17], etc. It is worthwhile to note that the learning approaches in [16,17] are off-line learning, which is not desirable in practical applications.

In this paper, different from the previous learning approaches, the proposed equilibrium-based learning control is actually one kind of online learning approach. To facilitate the proposed equilibrium-based learning controller design, we first construct a mathematical model for the robotic fish by utilizing Newton's second law and Lighthill's small amplitude model, where the dynamical model is highly nonlinear and non-affine in control input. According to the constructed dynamical model, two learning schemes are proposed for turning control of the robotic fish. It is worthwhile to mention that only the system gradient information instead of the perfect model is required in the controller design. Furthermore, due to the complexity of the mathematical

model, a self-adaption rule is introduced for the determination of learning gains, which is able to expedite the convergence rate. The efficiency of the proposed learning controllers are illustrated by both simulations and real-time experiments, and excellent speed and turning control are achieved for the robotic fish.

The paper is organized as follows. Section 2 presents the problem formulation and control method. Section 3 details the hardware configuration and dynamical model of robotic fish. Section 4 presents the application of the proposed control method to the robotic fish. Finally, a brief conclusion is given in Sect. 5.

2 Problem formulation and method

Consider the following nonlinear non-affine system

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \tag{1a}$$

$$\mathbf{y}(t) = \mathbf{h}(\mathbf{x}(t)), \tag{1b}$$

where $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{u} \in \mathbb{R}^p$, and $\mathbf{y} \in \mathbb{R}^p$ are the system state, input and output vectors; $\mathbf{f}(\mathbf{x}, \mathbf{u})$ is continuously differentiable w.r.t $\mathbf{x}$ and $\mathbf{u}$.

To facilitate the controller design, we impose the following two assumptions to system 1.

Assumption 1 For a given control input, system 1 has a unique steady-state response.

Remark 1 Assumption 1 can be easily justified for majority of industrial control plants, including chemical processes and motion systems. If Assumption 1 does not hold, we can first design a stabilizing controller that is able to stabilize the closed-loop system, and then design the learning controller for the stable closed-loop system. The idea is demonstrated in the block diagram shown in Fig. 1, where the inner loop is the stabilizing controller, and the outer loop is the learning controller. For plants already stable or well damped, there is even no need to apply such a stabilizing controller.

Remark 2 For real-time applications, the stabilizing controller component in Fig. 1 can also be replaced by (or incorporated with) a fault-tolerant controller to detect and isolate faults, and thus take appropriate remedial actions to prevent critical failures in the control system. The details for design and analysis of the fault-tolerant controller will be discussed in future works.

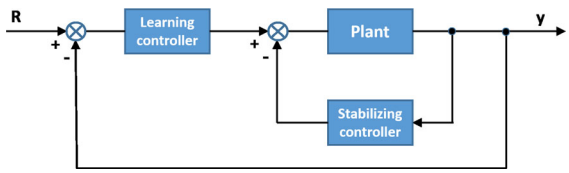


Fig. 1 Block diagram of a dual-loop control

Assumption 2 For system 1, both the matrices $\frac{\partial \mathbf{f}}{\partial \mathbf{x}}$ and $\frac{\partial \mathbf{h}}{\partial \mathbf{x}} \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)^{-1} \frac{\partial \mathbf{f}}{\partial \mathbf{u}}$ are full-rank.

Here, a simple linear system is illustrated to show the meaning of Assumption 2. For the following linear system

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t),$$
$$\mathbf{y}(t) = C\mathbf{x}(t),$$

Assumption 2 implies that both the matrices A and $CA^{-1}B$ have full rank.

Remark 3 When A is not full-rank, it is possible to apply a state feedback to make it full-rank. For instance, as far as (A, B) is controllable, there exists a state transformation such that A matrix can be transformed into canonical form, then an appropriate state feedback will make the closed-loop A full-rank. Similar to Remark 1, this state feedback can be a part of the stabilizing control. With a full-rank A, then the second condition in Assumption 2, i.e. $CA^{-1}B$ being full-rank, is equivalent to CB being full-rank. When output tracking is considered, full-rank of CB is necessary. In fact, in both ILC and RC, CB needs to be full-rank.

In this work, we define the ith learning interval as $[T_{i-1}, T_i]$ where the inner closed-loop system will reach steady state or equilibrium before T_i, $T_i > 0$, $i \in \mathbf{Z}^+$, and the corresponding control input signal is $\mathbf{u}_i$, namely,

$$\mathbf{u}(t) = \mathbf{u}_i, \quad T_{i-1} \le t \le T_i, \quad i \in \mathbf{Z}^+,$$

where $\mathbf{u}_i$ is a constant vector. Since it is assumed that the control action only updates as far as the plant reaches an equilibrium, it is obvious that $\dot{\mathbf{x}}(T_i) = 0$. Denote $\mathbf{x}_i \triangleq \mathbf{x}(T_i)$, then $(\mathbf{x}_i, \mathbf{u}_i)$ is the equilibrium of system (1), namely,

$$\begin{cases} 0 = \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i), \\ \mathbf{y}_i = \mathbf{h}(\mathbf{x}_i). \end{cases} \tag{2}$$

Let $\mathbf{y}_d$ be the target, which is generated by

$$\begin{cases} \dot{\mathbf{x}}_d = \mathbf{f}(\mathbf{x}_d, \mathbf{u}_d) = 0, \\ \mathbf{y}_d = \mathbf{h}(\mathbf{x}_d), \end{cases} \tag{3}$$

where $\mathbf{u}_d$ is the desired control input.

The control objective is to determine a sequence of control inputs $\mathbf{u}_i, i \in \mathbf{Z}^+$ such that the tracking error $\mathbf{e}_i \triangleq \mathbf{y}_d - \mathbf{y}_i$ converges to zero as the iteration number i increases.

The proposed equilibrium-based learning controller is presented as follows.

$$\mathbf{u}_{i+1} = \mathbf{u}_i + \Gamma \mathbf{e}_i \tag{4}$$

where Γ is the learning gain matrix to be determined.

Remark 4 The expression of the learning law (4) is similar to the *P*-type ILC law. However, different from classic ILC, the learning scheme (4) is continuous along the time axis without any resetting. The control action updates as far as the plant reach its equilibrium point that deviates from the reference set-point.

Remark 5 In practical applications, the learning interval differs from the system sampling interval. If the system response is slow, the learning interval needs to be sufficiently long so as to reach an equilibrium. In such circumstance, a learning interval may equal to multiple sample periods. On the contrary, if the system could respond fast, the learning interval can be short and the extreme is equal to one sampling interval.

The convergence property of the proposed learning controller is derived in the following theorem.

Theorem 1 *For the nonlinear system* (1)*, under the Assumptions* 1 *and* 2*, the learning scheme* (4) *guarantees that the tracking error* $\mathbf{e}_i(t)$ *will converge to zero asymptotically, provided that the learning gain matrix* Γ *is chosen such that for* $0 < \delta < 1$*,*

$$\sup_{\mathbf{x}\in\mathbb{R}^n, \mathbf{u}\in\mathbb{R}^p} \left\| I_p + \Gamma \frac{\partial \mathbf{h}}{\partial \mathbf{x}} \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right)^{-1} \frac{\partial \mathbf{f}}{\partial \mathbf{u}} \right\| \leq \delta. \tag{5}$$

Proof Let $\Delta \mathbf{u}_i \triangleq \mathbf{u}_d - \mathbf{u}_i$. From (4), we can obtain

$$\begin{aligned} \Delta \mathbf{u}_{i+1} &= \Delta \mathbf{u}_i - \Gamma \mathbf{e}_i \\ &= \Delta \mathbf{u}_i - \Gamma (\mathbf{y}_d - \mathbf{y}_i) \\ &= \Delta \mathbf{u}_i - \Gamma [\mathbf{h}(\mathbf{x}_d) - \mathbf{h}(\mathbf{x}_i)] \\ &= \Delta \mathbf{u}_i - \Gamma \frac{\partial \mathbf{h}}{\partial \mathbf{x}} (\mathbf{x}_d - \mathbf{x}_i). \end{aligned} \tag{6}$$

By applying the Mean Value Theorem, we have

$$\begin{aligned} 0 &= \mathbf{f}(\mathbf{x}_d, \mathbf{u}_d) - \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i) \\ &= \mathbf{f}(\mathbf{x}_d, \mathbf{u}_d) - \mathbf{f}(\mathbf{x}_d, \mathbf{u}_i) + \mathbf{f}(\mathbf{x}_d, \mathbf{u}_i) - \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i) \\ &= \frac{\partial \mathbf{f}}{\partial \mathbf{u}} (\mathbf{x}_d, \hat{\mathbf{u}}_i)(\mathbf{u}_d - \mathbf{u}_i) + \frac{\partial \mathbf{f}}{\partial \mathbf{x}} (\hat{\mathbf{x}}_i, \mathbf{u}_i)(\mathbf{x}_d - \mathbf{x}_i), \end{aligned} \tag{7}$$

where $\hat{\mathbf{u}}_i = \mathbf{u}_d + \theta_i^{\mathbf{u}}(\mathbf{u}_i - \mathbf{u}_d)$, $\hat{\mathbf{x}}_i = \mathbf{x}_d + \theta_i^{\mathbf{x}}(\mathbf{x}_i - \mathbf{x}_d)$, and $\theta_i^{\mathbf{u}}, \theta_i^{\mathbf{x}} \in (0, 1)$. Therefore,

$$\mathbf{x}_d - \mathbf{x}_i = -\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right)^{-1} \frac{\partial \mathbf{f}}{\partial \mathbf{u}} (\mathbf{u}_d - \mathbf{u}_i). \tag{8}$$

Substituting (8) into (6), we can obtain

$$\Delta \mathbf{u}_{i+1} = \left(I_p + \Gamma \frac{\partial \mathbf{h}}{\partial \mathbf{x}} \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right)^{-1} \frac{\partial \mathbf{f}}{\partial \mathbf{u}} \right) \Delta \mathbf{u}_i. \tag{9}$$

Taking norm on both sides of (9) yields

$$\| \Delta \mathbf{u}_{i+1} \| \leq \left\| I_p + \Gamma \frac{\partial \mathbf{h}}{\partial \mathbf{x}} \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right)^{-1} \frac{\partial \mathbf{f}}{\partial \mathbf{u}} \right\| \| \Delta \mathbf{u}_i \|, \tag{10}$$

which implies the convergence of $\mathbf{u}_i$ when the condition (5) holds. Therefore, we can obtain the convergence of the tracking error $\mathbf{e}_i$. □

3 Robotic fish prototype and dynamic modeling

Figure 2 shows the schematic structure of the robotic fish prototype. The length of the robotic fish is approximately 36 cm. The lunate tail is connected with the body by a high-torque servo motor (JR DS R8801). Due to mechanical restriction, the angle range for the servomotor is about $-60°$ to $60°$. The tail is made of perspex with chord length 12 cm and span 17 cm. The body of the robot has a sealed compartment composed of plastic side or top panel wrapped with waterproof tap. The compartment contains a micro controller (ATMEL ATSAM3X8E), an inertial measurement unit (VN-100), a Bluetooth wireless communication module and a lithium battery. The micro controller is responsible for controlling the servo motors, transferring diagnostic information via the wireless link, processing sensor data and making decisions. The wireless communication module is used to receive command from a host computer. The lithium battery is applied to provide power for the servomotors. The frequency of the control signal is fixed at 1 Hz in this work. In this work, the pectoral fins will not be used to propel the robotic fish, and the tail is the only actuator.

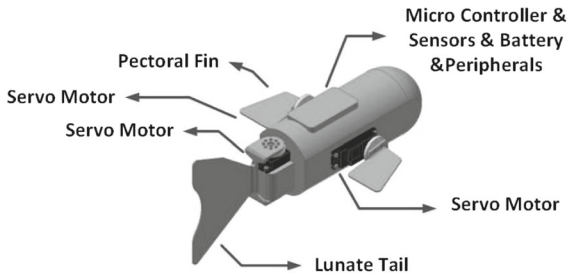


Fig. 2 Schematic structure of the robotic fish

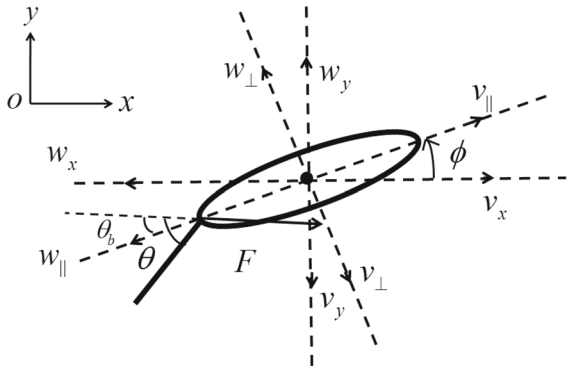


Fig. 3 External forces acting on the robotic fish

Figure 3 shows a simple diagram of the robotic fish with the principle acting forces. Without loss of generality, we assume that both the body center of mass and the central line locate at the center of the fish body. The body position is given relative to an inertial frame by $[x, y]$, where the positive direction is taken to the right. The body orientation ϕ is measured relative to the inertial $+x$-axis, and θ_b is the bias angle between the central line of the fish body and the fish tail. The length of the robotic fish is l and that of the tail is $l/3$.

3.1 Caudal fin thrust modeling

Figure 4 shows the top-view geometry of the robotic fish. The body of the robotic fish prototype is rigid, and it is actuated by a caudal fin periodically.

The displacement $h(x, t)$ at the position x can be calculated as follows

$$h(x, t) = x \cos(\theta_b) \tan(\theta(t)), \tag{11}$$

where $\theta(t)$ in radian is the angle between the tail and the central line of the robotic fish plus the angle bias θ_b. The angle $\theta(t)$ is directly proportional to the motor rotation, and it is used as the driving term as a function of time. If

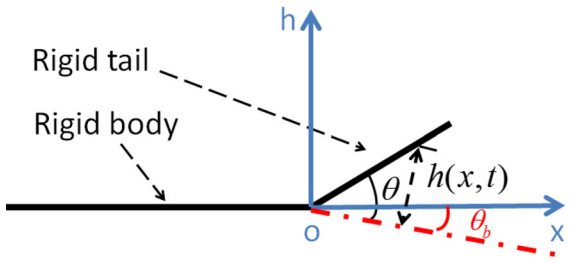


Fig. 4 The top-view geometry of the tail-actuated robotic fish

$\theta(t)$ is driven sinusoidally, i.e., $\theta(t) = \theta_a \sin(2\pi f t) + \theta_b$, the displacement $h(x, t)$ can be written as

$$h(x, t) = x \cos(\theta_b) \tan(\theta_a \sin(2\pi f t) + \theta_b), \tag{12}$$

where θ_a is the amplitude, f is the frequency of the sinusoidal motion, and $\theta_a \sin(2\pi f t)$ is the angle between the tail and the central line of the robotic fish. Now, taking the derivative with respect to time t on both side of (12), it follows that

$$\left(\frac{\partial h(x,t)}{\partial t}\right)_{x=\frac{l}{3}} = \frac{2}{3}\pi f \theta_a l \cos(\theta_b) \sec^2(\theta_a \sin(2\pi f t)+\theta_b) \times \cos(2\pi f t), \tag{13}$$

In addition, the spatial derivative of $h(x, t)$ at $x = l/3$ is

$$\left(\frac{\partial h(x,t)}{\partial x}\right)_{x=\frac{l}{3}} = \cos(\theta_b) \tan(\theta_a \sin(2\pi f t) + \theta_b). \tag{14}$$

According to Lighthill's small displacement model developed in [18,19], the average trust F generated by the fish is given as follows

$$F = \frac{\rho A(l)}{2}\left[\left(\frac{\partial h}{\partial t}\right)^2 - v^2\left(\frac{\partial h}{\partial x}\right)^2\right], \tag{15}$$

where ρ is the density of water, v is the velocity of fish, $A(l)$ is the area of a circle computed by using the overall dimension of the tail as a diameter, and the squares of the derivative values are averages over a typical cycle, namely,

$$\left(\frac{\partial h}{\partial t}\right)^2 \triangleq \frac{1}{T}\int_0^T \left(\frac{\partial h(x,t)}{\partial t}\right)^2_{x=\frac{l}{3}} \mathrm{d}t, \tag{16}$$

$$\left(\frac{\partial h}{\partial x}\right)^2 \triangleq \frac{1}{T}\int_0^T \left(\frac{\partial h(x,t)}{\partial x}\right)^2_{x=\frac{l}{3}} \mathrm{d}t, \tag{17}$$

where $T = 1/f$ is the period of sinusoidal motion. In this work, the undulatory frequency is fixed at

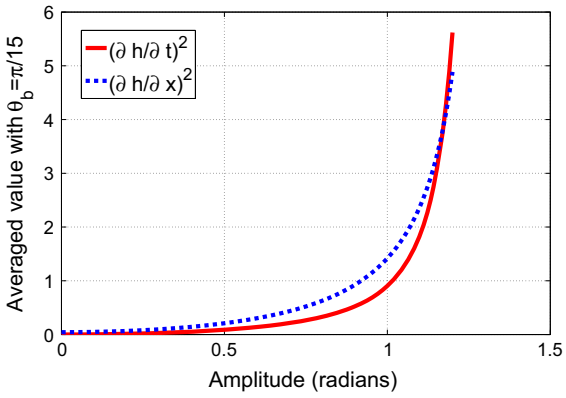


Fig. 5 The integrated square of the slope and velocity of the tail over one cycle versus amplitude

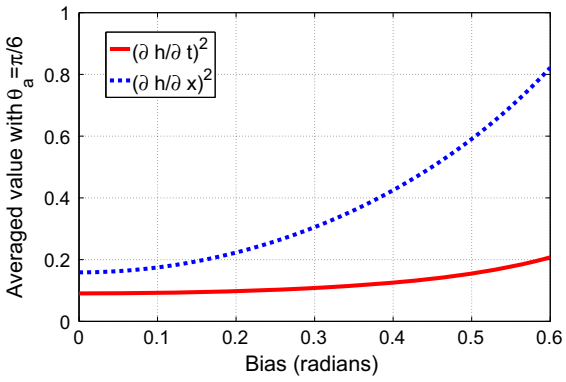


Fig. 6 The integrated square of the slope and velocity of the tail over one cycle versus bias

$f = 1$ Hz, namely, $T = 1$ s, and the amplitude of undulation and bias of fish tail will be used to manipulate the motion of the robotic fish. Further, the time-averaged values for these derivatives can be found numerically for various amplitudes and biases, and the results are shown in Figs. 5 and 6, respectively.

Denote

$$H_1(\theta_a, \theta_b, f) \triangleq \frac{\rho A(l)}{2}\left(\frac{\partial h}{\partial t}\right)^2 \tag{18}$$

and

$$H_2(\theta_a, \theta_b, f, v) \triangleq \frac{\rho A(l) v^2}{2}\left(\frac{\partial h}{\partial x}\right)^2. \tag{19}$$

The thrust F can be rewritten as

$$F(\theta_a, \theta_b, f, v) = H_1(\theta_a, \theta_b, f) - H_2(\theta_a, \theta_b, f, v) \tag{20}$$

Numerically, the variation of F with respect to the amplitude θ_a and the bias θ_b is shown in Fig. 7, where f and v are fixed as 1 Hz and 0.1 m/s.

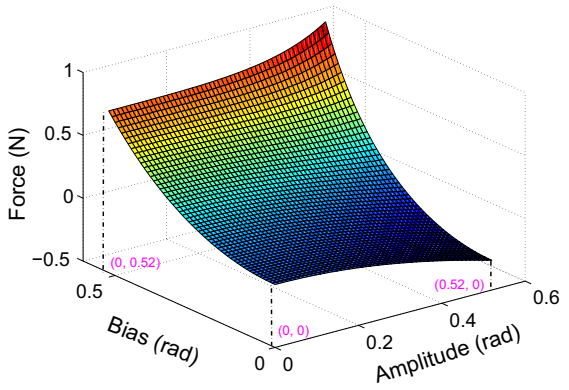


Fig. 7 The variation of thrust with respect to θ_a and θ_b

3.2 Drag force modeling

Besides the thrust generated by caudal fin undulations, the robot body also experiences drag force. The drag can be calculated through surface integrals of vector drag per area around the fish body. Since the drag is highly related with the geometry of the object immersed in water and relative velocity between the object and water, in principle, the exact force distribution can be obtained by solving the Navier–Stokes equation. However, the calculation is quite complicated and time consuming. Similar as [20–22], we assume the drag on the body of the robot is generated in steady or quasisteady flow, and for simplicity, the drag takes the form,

$$w_\perp = -f_\perp (v_\perp)^2 \text{sign}(v_\perp), \tag{21}$$

$$w_\parallel = -f_\parallel (v_\parallel)^2 \text{sign}(v_\parallel), \tag{22}$$

where $v_\perp$ and $v_\parallel$ are perpendicular component and parallel component of the velocity v, respectively, and $f_\perp$, $f_\parallel$ are the water resistance coefficients in the corresponding directions. Based on the geometric relationship (refer to Fig. 3), we have

$$\begin{aligned} v_\perp &= v_x \sin(\phi) + v_y \cos(\phi), \\ v_\parallel &= v_x \cos(\phi) - v_y \sin(\phi), \\ w_x &= w_\perp \sin(\phi) + w_\parallel \cos(\phi), \\ w_y &= w_\perp \cos(\phi) - w_\parallel \sin(\phi), \end{aligned}$$

where v_x, v_y are projection of the velocity v on x-axis and y-axis, and w_x, w_y are projection of the drag on x-axis and y-axis.

Furthermore, as shown in [23], the fluid adds a damping effect to rotations of the body about its principle axis, since the rotating body will induce separated flow about its bluff edges that will transfer rotational

kinetic energy from the body to the fluid. This effect is included in the model as a simple viscous moment that impedes body rotation

$$M_{\text{damp}} = -c_m\omega, \tag{23}$$

where ω is body fixed angular velocity and c_m is a simple damping coefficient

3.3 Dynamical model of robotic fish

By applying Newton's second law, the dynamical model of the tail-actuated robotic fish is derived as follows

$$\begin{cases} M\ddot{x} = F_x + w_x, \\ M\ddot{y} = F_y + w_y, \\ I\ddot{\phi} = F_x l\sin(\phi)/3 + F_y l\cos(\phi)/3 - c_m\dot{\phi}, \end{cases} \tag{24}$$

where M and I are the mass and moment of inertia of robotic fish, respectively, $F_x \triangleq F\cos(\theta_b - \phi)$, and $F_y \triangleq F\sin(\theta_b - \phi)$.

In turning control, the speed and angular velocity of robotic fish are our foremost concerns. Thus, the control output vector is

$$Y = [v, \omega]^T = \left[\sqrt{\dot{x}^2 + \dot{y}^2}, \dot{\phi}\right]^T. \tag{25}$$

Taking the time derivative of y, we can obtain that

$$\begin{aligned} \frac{\mathrm{d}Y}{\mathrm{d}t} &= \left[\frac{(\dot{x}\ddot{x} + \dot{y}\ddot{y})}{\sqrt{\dot{x}^2 + \dot{y}^2}}, \ddot{\phi}\right]^T \\ &= \begin{bmatrix} \frac{1}{v}\left(\frac{v_x(F_x+w_x)}{M} + \frac{v_y(F_y+w_y)}{M}\right) \\ \frac{F_x l\sin(\phi)+F_y l\cos(\phi)-3c_m\dot{\phi}}{3I} \end{bmatrix} \\ &\triangleq \begin{bmatrix} P_T^1(v, \theta_a, \theta_b) \\ P_T^2(v, \omega, \theta_a, \theta_b) \end{bmatrix} \end{aligned} \tag{26}$$

From the Eq. (26), it is clear that the relation between inputs and outputs is highly nonlinear. In the following subsection, we will show that for such a complex control system, two simple decoupled control algorithms are sufficient to drive the robot to the desired motion.

Before addressing the controller design problem, we first make the following assumption.

Assumption 3 In each undulation cycle, both the speed and angular velocity of robotic fish can reach their steady states. That is, for each pair of given control input (θ_a, θ_b), the robotic fish can approach to the equilibria $\dot{v} = 0$ and $\dot{\omega} = 0$ within the undulatory cycle $[iT, (i+1)T]$.

Remark 6 This is a reasonable assumption since the robotic fish model is a high damping system, and the assumption can be verified numerically in the following section.

4 Application to the robotic fish

In this section, the proposed equilibrium-based controller will be applied to the robotic fish prototype.

4.1 Controller design

Denote v_d and ω_d the desired speed and angular velocity of the robotic fish, respectively. θ_a^d and θ_b^d are the desired amplitude and bias. The actual speed and angular velocity at the ith motion are defined as $v_i \triangleq \sqrt{\dot{x}(iT)^2 + \dot{y}(iT)^2}$, and $\omega_i \triangleq \dot{\phi}(iT)$, respectively. Let $e_i^v \triangleq v_d - v_i$ and $e_i^\omega \triangleq \omega_d - \omega_i$ be tracking errors. Set the initial control inputs as θ_a^0 and θ_b^0.

In terms of the control inputs and the measured errors at the ith motion, take the following learning control schemes to generate the actual inputs at the $(i+1)$th motion:

$$\theta_a^{i+1} = \theta_a^i + \gamma_1 e_i^v, \tag{27}$$

$$\theta_b^{i+1} = \theta_b^i + \gamma_2 e_i^\omega, \tag{28}$$

where γ_1 and γ_2 are learning gains to be designed. Then the convergence of the tracking error can be summarized as follows.

Corollary 1 *Consider the robotic fish model (24) under the Assumption 3 and the learning schemes (27) and (28). Then e_i^v and e_i^ω will converge to zero as i goes to infinity if and only if*

$$\sup_{v,\omega,\theta_a,\theta_b} \left\| I + \Gamma A_i^{-1} B_i \right\| < 1, \tag{29}$$

where

$$\Gamma \triangleq \begin{bmatrix} \gamma_1 & 0 \\ 0 & \gamma_2 \end{bmatrix},$$

$$A_i \triangleq \begin{bmatrix} \frac{\partial P_T^1}{\partial v}\left(\hat{v}_i, \hat{\theta}_a^i, \hat{\theta}_b^i\right) & 0 \\ \frac{\partial P_T^2}{\partial v}\left(\hat{v}_i, \hat{\omega}_i, \hat{\theta}_a^i, \hat{\theta}_b^i\right) & \frac{\partial P_T^2}{\partial \omega}\left(\hat{v}_i, \hat{\omega}_i, , \hat{\theta}_a^i, \hat{\theta}_b^i\right) \end{bmatrix},$$

$$B_i \triangleq \begin{bmatrix} \frac{\partial P_T^1}{\partial \theta_a}\left(\hat{v}_i, \hat{\theta}_a^i, \hat{\theta}_b^i\right) & \frac{\partial P_T^1}{\partial \theta_b}\left(\hat{v}_i, \hat{\theta}_a^i, \hat{\theta}_b^i\right) \\ \frac{\partial P_T^2}{\partial \theta_a}\left(\hat{v}_i, \hat{\omega}_i, , \hat{\theta}_a^i, \hat{\theta}_b^i\right) & \frac{\partial P_T^2}{\partial \theta_b}\left(\hat{v}_i, \hat{\omega}_i, , \hat{\theta}_a^i, \hat{\theta}_b^i\right) \end{bmatrix},$$

$\hat{v}_i = v_d + \vartheta_i^v(v_i - v_d)$, $\hat{\omega}_i = \omega_d + \vartheta_i^\omega(\omega_i - \omega_d)$, $\hat{\theta}_a^i = \theta_a^d + \vartheta_i^a(\theta_a^i - \theta_a^d)$, $\hat{\theta}_b^i = \theta_b^d + \vartheta_i^b(\theta_b^i - \theta_b^d)$ *with* $\vartheta_i^v, \vartheta_i^\omega, \vartheta_i^a, \vartheta_i^b \in (0, 1)$.

Proof Define $\mathbf{x}_i = [v_i, \omega_i]^T$ and $\mathbf{u}_i = \left[\theta_a^i, \theta_b^i\right]^T$. Then system (26) can be represented in the form of system 1 with

$$\mathbf{f}(\mathbf{x}_i, \mathbf{u}_i) = \begin{pmatrix} P_T^1\left(v_i, \theta_a^i, \theta_b^i\right) \\ P_T^2\left(v_i, \omega_i, \theta_a^i, \theta_b^i\right) \end{pmatrix}, \ \mathbf{h}(\mathbf{x}_i) = \mathbf{x}_i.$$

Let

$$A_i \triangleq \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\left(\hat{v}_i, \hat{\omega}_i, , \hat{\theta}_a^i, \hat{\theta}_b^i\right), \quad B_i \triangleq \frac{\partial \mathbf{f}}{\partial \mathbf{u}}\left(\hat{v}_i, \hat{\omega}_i, , \hat{\theta}_a^i, \hat{\theta}_b^i\right).$$

Then following the similar procedures as the proof of Theorem 1, the convergence of the tracking errors e_i^v and e_i^ω can be obtained. □

Remark 7 Corollary 1 provides a sufficient convergence condition, namely, as long as the learning gain Γ is designed appropriately to satisfy (29), the convergence of e_i^v and e_i^ω can be guaranteed. In fact, the learning speed can be expedited by combining the learning scheme with other design techniques, e.g., the self-adaption rule that scales the learning gain up and down by a factor [24,25]. Mathematically, the self-adaption rule can be expressed as follows:

$$\gamma_i = \begin{cases} \eta\gamma_{i-1}, & \text{if } \operatorname{sign}(e_i) = \operatorname{sign}(e_{i-1}), \\ \frac{1}{\eta}\gamma_{i-1}, & \text{otherwise}, \end{cases} \tag{30}$$

before the tracking error converges to the prespecified neighborhood of zero, where $\eta > 1$ is a tuning parameter. Actually, the adaption rule (30) is based on the simple idea: On the one hand, if $\operatorname{sign}(e_i) = \operatorname{sign}(e_{i-1})$, it means that the increment of input value is not enough, thus it is desirable to increase the learning gain, hence increase the step size of learning. On the other hand, $\operatorname{sign}(e_i) \neq \operatorname{sign}(e_{i-1})$ means that the change of input value is too large, and that we shall tune it carefully with a smaller change, thus decreasing the learning gain.

Remark 8 Over the past few years, although several achievements have been made in linear speed control of robotic fish [13,16,17,26], works on turning control of robotic fish are substantially less in literature, and there are only some preliminary results. For example, in [27] an open-loop control method is developed to investigate the turning maneuvers of a multilink robotic fish. It is shown that by adding different deflections to symmetric swimming gaits, the robotic fish is able to perform turning motions. In [28], the authors adopt a fuzzy logic control method for orientation control of a multilink robotic fish. The drawback of this approach lies in the specificity of fuzzy rules design. If the system dynamics is unknown, it is difficult to tune the controller parameters. Even if the controller has been well-tuned, it is only applicable to the robotic fish in question, and difficult to be generalized to other robotic fish prototypes. From the following simulation and experiment results, it shows that the proposed control method performs very well for robotic fish turning control despite easy tuning and less requirement on accurate dynamical model.

4.2 Simulation results for turning control

In simulation, we set $\rho = 1000\ \text{kg/m}^3$ and $c_m = 0.5\ \text{kg m}^2$, and the following parameters are from the actual robotic fish: $M = 0.4\,\text{kg}$, $l = 0.36\,\text{m}$, and $A(l) = 0.165^2\,\pi/4\ \text{m}^2$. For our robotic fish, the water resistance coefficients $f_\perp$ and $f_\parallel$ are empirically determined to be 165.7056 kg/m according to the result in [17]. Let the desired speed be $\theta_a^0 = 0$, $\theta_b^0 = 0$. The target speed of the robotic fish is $v_d = 0.05$ (m/s), and the desired angular velocity is $\omega_d = 0.02$ (rad/s).

To expedite the learning speed, we adopt two self-adaption laws to determine learning gains. That is,

$$\gamma_{1,i} = \begin{cases} \eta_1\gamma_{1,i-1}, & \text{if } \operatorname{sign}\left(e_i^v\right) = \operatorname{sign}\left(e_{i-1}^v\right), \\ \frac{1}{\eta_1}\gamma_{1,i-1}, & \text{otherwise}, \end{cases} \tag{31}$$

and

$$\gamma_{2,i} = \begin{cases} \eta_2\gamma_{2,i-1}, & \text{if } \operatorname{sign}\left(e_i^\omega\right) = \operatorname{sign}\left(e_{i-1}^\omega\right), \\ \frac{1}{\eta_2}\gamma_{2,i-1}, & \text{otherwise}. \end{cases} \tag{32}$$

Set the initial learning gains $\gamma_{1,0} = 1$, $\gamma_{2,0} = 0.7$, and $\eta_1 = 1.1$, $\eta_2 = 1.2$. The simulation results are shown in Figs. 8, 9 and 10. Figure 8 shows the convergence of the tracking errors. The robotic fish is able to achieve both the desired speed and angular velocity simultaneously after about 15 sampling intervals. The speed and angular velocity profiles of the robotic fish along time axis are given in Fig. 9, where we can see that both the speed and angular velocity of the robotic fish can reach their steady states within one undulation cycle. Moreover, Fig. 10 presents the trajectory of the robotic fish. It shows that the robotic fish turns itself clockwise by using the "turning" motion pattern. In

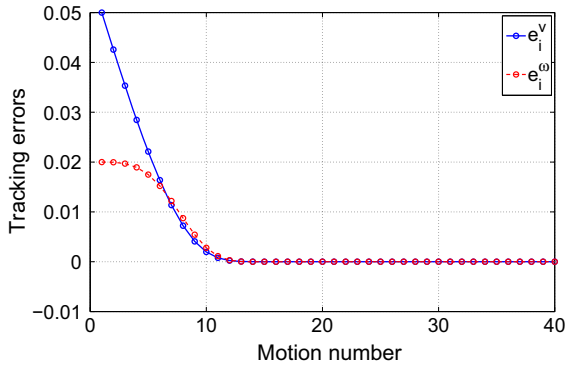


Fig. 8 The variation of tracking error for turning versus motion number

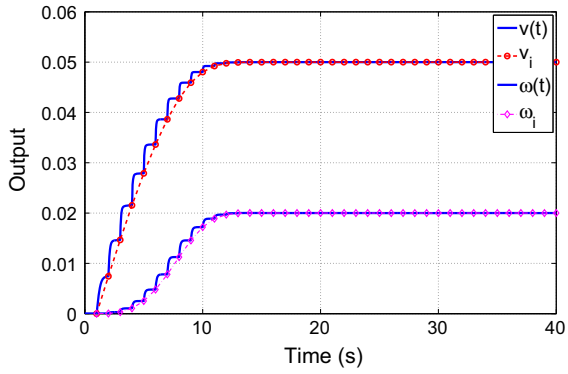
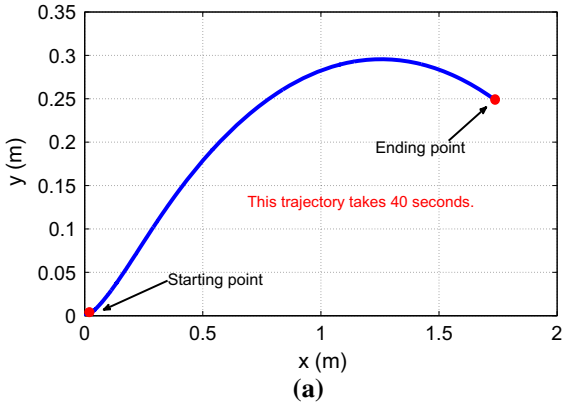


Fig. 9 The profile of speed and angular velocity of the robotic fish versus time

addition, Fig. 11 gives the variation of learning gains with the self-adaption.

To show the effectiveness of the self-adaption rules for learning gains, we calculate the average values of the learning gain profiles shown in Fig. 11, which are $\bar{\gamma}_{1,i} = 1.8056$ and $\bar{\gamma}_{2,i} = 15.0755$, then use them as fixed learning gains in learning rules (27) and (28), respectively. The convergence of tracking errors is presented in Fig. 12. Compared to Fig. 8, not only the convergence rates are slower, but also the transient performance in Fig. 12 is non-smooth. Such a phenomenon is not desirable in practical applications.

Remark 9 For highly nonlinear systems, how to select an optimal learning gain is a challenging task. The self-adaption law provides a simple way to design the learning gain, which only needs an initial learning gain and an appropriate tuning parameter.



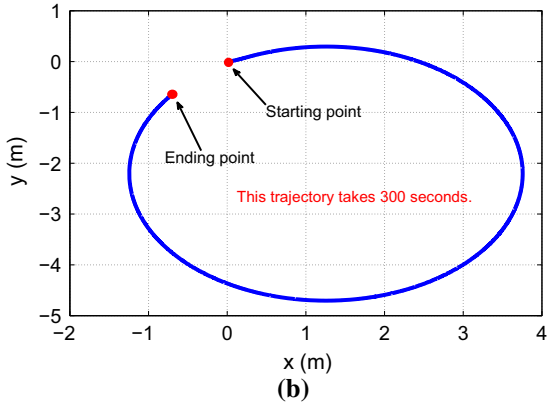


Fig. 10 The trajectory of the robotic fish in a clockwise circular maneuver: **a** trajectory within 40 s, **b** trajectory within 300 s

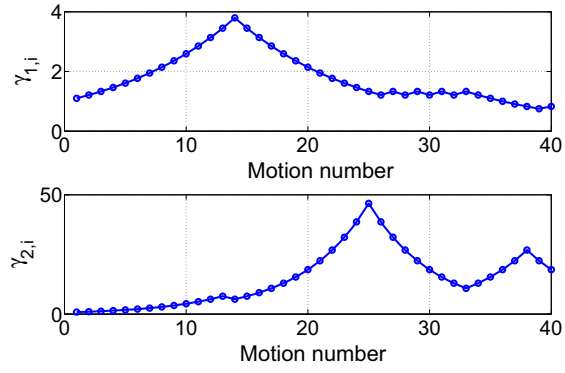


Fig. 11 The profiles of learning gains versus motion number

4.3 Experimental results for turning control

To verify the effectiveness of the proposed controllers (27) and (28), real-time experiments are conducted in a water tank of the size 3×1.8 m^2 with still water of 0.5 m in depth. In the experiment, an overhead cam-

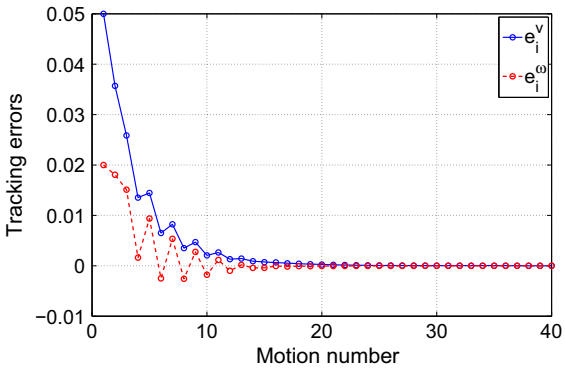


Fig. 12 The variation of tracking error for turning with fixed learning gains

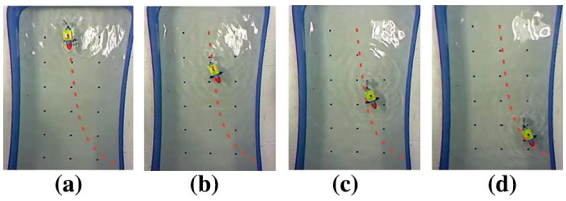


Fig. 13 The robotic fish swims at the experiment of the turning control

era is used to record the trajectories and positions of the robotic fish. We send the amplitude signals to the robotic fish from the host computer. After receiving the signals of amplitude through the wireless module, the processor transforms them to pulse width modulation (PWM) signals to drive the servomotor. Then, the motor starts to work and the corresponding swimming gait will be generated by the fish according to the received signals. The turning swimming mode of the robotic fish is presented in Fig. 13, where the swimming trajectory has been clearly indicated by an orange dashed curve.

Same as the simulations, the desired speed and angular velocity of the robotic fish are $v_d = 0.05$ (m/s) and $\omega_d = 0.02$ (rad/s), respectively. The convergence of the tracking errors is shown in Fig. 14, and the speed and angular velocity profiles are presented in Fig. 15. Compared to the simulation results, the convergence rates of speed and angular velocity in experiments are a little slower, which may be caused by the inaccuracy of dynamic model or external disturbances. The oscillations in the speed and angular velocity profiles shown in Fig. 15 are acceptable since the robotic fish is actually an open-loop control system within one undulation cycle.

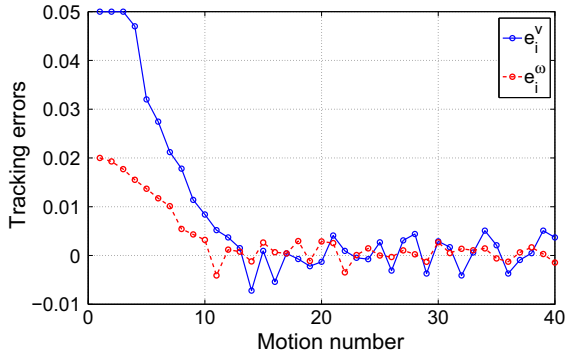


Fig. 14 The robotic fish swims at the experiment of the speed control

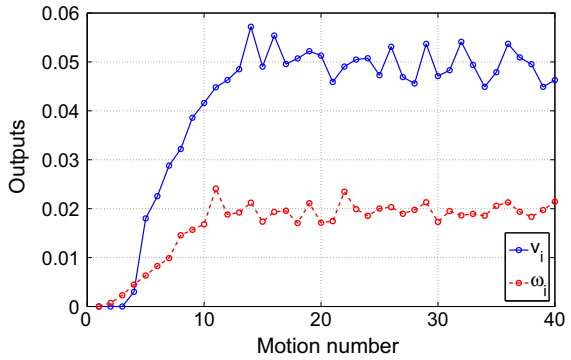


Fig. 15 The robotic fish swims at the experiment of the speed control

Remark 10 To remove the chattering errors in both speed and angular speed, a robust nonlinear controller that is equipped with uncertainty tolerance can be applied. Together with the proposed learning controller, a dual-loop control approach is able to reject the system uncertainties and drive the system plant to the desired control target simultaneously.

5 Conclusion

In this work, an equilibrium-based learning approach is developed. Different from the integral control, the proposed learning scheme only updates as far as the plant reaches an equilibrium that deviates from the reference set-point. For the integral control of nonlinear systems, it is well-known that there lacks appropriate analysis and design methods. While, for the proposed learning control, the convergence of tracking error can be proven analytically by applying the fixed point theo-

rem despite the high nonlinearities in the model. To validate the effectiveness, the developed control approach is applied to motion control of a tail-actuated robotic fish. In details, we first construct a mathematical model for the tail-actuated robotic fish according to Newton's second law and Lighthill's small amplitude model. By the virtue of the structure of the constructed dynamical model, two learning algorithms are developed for turning control of the robot. Furthermore, in order to expedite the convergence rate, a self-adaption rule is introduced to determine the learning gains. The efficiency of the proposed control approach is illustrated by both simulations and real-time experiments. In future work, the equilibrium-based learning control approach will be applied in 3D motion control of robotic fish.

Funding Open Research Project of the State Key Laboratory of Industrial Control Technology, Zhejiang University, China (Grant No. ICT170345).

References

1. Bristow, D.A., Tharayil, M., Alleyne, A.G.: A survey of iterative learning control. IEEE Control Syst. Mag. **26**, 96–114 (2006)
2. Chen, Y.Q., Moore, K.L., Yu, J., Zhang, T.: Iterative learning control and repetitive control in hard disk drive industry-a tutorial. Int. J. Adapt. Control Signal Process. **22**, 325–343 (2008)
3. Li, X., Xu, J.-X., Huang, D.: An iterative learning control approach for linear time-invariant systems with randomly varying trial lengths. IEEE Trans. Automat. Control **59**(7), 1954–1960 (2014)
4. Yu, M., Zhou, W., Liu, B.: On iterative learning control for MIMO nonlinear systems in the presence of time-iteration-varying parameters. Nonlinear Dyn. **89**, 2561–2571 (2017). https://doi.org/10.1007/s11071-017-3604-0
5. Kim, M., Kuc, T.Y., Kim, H., Wi, S., Lee, J.S.: An adaptive learning controller for MIMO uncertain feedback linearizable nonlinear systems. Nonlinear Dyn. **80**(1–2), 999–1016 (2015)
6. Xu, J.-X., Tan, Y.: Linear and nonlinear iterative learning control. In: Series of Lecture Notes in Control and Information Sciences. Springer, Berlin (2003)
7. Wang, Y., Gao, F., Doyle, F.J.: Survey on iterative learning control, repetitive control, and run-to-run control. J. Process Control **19**, 1589–1600 (2009)
8. Quan, Q., Cai, K.: A survey of repetitive control for nonlinear systems. Sci. Found. China **18**, 45–59 (2010)
9. McIsaac, K.A., Ostrowski, J.P.: Motion planning for anguilliform locomotion. IEEE Trans. Robot. Autom. **19**, 637–652 (2003)
10. Elmokadem, T., Zribi, M., Youcef-Toumi, K.: Trajectory tracking sliding mode control of underactuated AUVs. Nonlinear Dyn. **84**(2), 1079–1091 (2016)
11. Joe, H., Kim, M., Yu, S.C.: Second-order sliding-mode controller for autonomous underwater vehicle in the presence of unknown disturbances. Nonlinear Dyn. **78**(1), 183–196 (2014)
12. Ghavidel, H.F., Kalat, A.A.: Robust control for MIMO hybrid dynamical system of underwater vehicles by composite adaptive fuzzy estimation of uncertainties. Nonlinear Dyn. **89**, 2347–2365 (2017). https://doi.org/10.1007/s11071-017-3590-2
13. Wen, L., Wang, T., Wu, G.: Novel method for the modeling and control investigation of efficient swimming for robotic fish. IEEE Trans. Ind. Electron. **59**(8), 3176–3188 (2012)
14. Kato, N.: Control performance in the horizontal plane of a fish robot with mechanical pectoral fins. IEEE J. Ocean. Eng. **25**(1), 121–129 (2000)
15. Yu, J., Tan, M., Wang, S., Chen, E.: Development of a biomimetic robotic fish and its control algorithm. IEEE Trans. Syst. Man Cybern. **34**(4), 1798–1810 (2004)
16. Ren, Q., Xu, J.-X., Li, X.: A data-driven motion control approach for a robotic fish. J. Bionic Eng. **12**, 382–394 (2015)
17. Li, X., Ren, Q., Xu, J.-X.: Precise speed tracking control of a robotic fish via iterative learning control. IEEE Trans. Ind. Electron. **63**(4), 2221–2228 (2016)
18. Mcmasters, R.L., Grey, C.P., Sollock, J.M.: Comparing the mathematical models of lighthill to the performance of a biomimetic fish. Bioinspir. Biomim. **3**(1), 247–269 (2008)
19. Lighthill, M.J.: Note on the swimming of slender fish. J. Fluid Mech. **9**(1), 305–317 (1960)
20. Colgate, J.E., Lynch, K.M.: Mechanics and control of swimming: a review. IEEE J. Ocean. Eng. **29**(3), 660–673 (2004)
21. Wang, J., Tan, X.: A dynamic model for tail-actuated robotic fish with drag coefficient adaptation. Mechatronics **23**(6), 659–668 (2013)
22. Aureli, M., Kopman, V., Porfiri, M.: Free-locomotion of underwater vehicles actuated by ionic polymer metal composites. IEEE/ASME Trans. Mechatron. **15**(4), 603–614 (2010)
23. Morgansen, K.A., Triplett, B.I., Klein, D.J.: Geometric methods for modeling and control of free-swimming fin-actuated underwater vehicles. IEEE Trans. Robot. **23**(6), 1184–1199 (2007)
24. Salomon, R.: Evolutionary algorithms and gradient search: similarities and differences. IEEE Trans. Evol. Comput. **2**(2), 45–55 (1998)
25. Xu, J.-X., Huang, D.: Iterative learning in ballistic control: formulation of spatial learning processes for endpoint control. ASME J. Dyn. Syst. Meas. Control **135**, 024501–11 (2013)
26. Verma, S., Xu, J.-X.: Data assisted modelling and speed control of a robotic fish. IEEE Trans. Ind. Electron. **64**, 4150–4157 (2016). https://doi.org/10.1109/TIE.2016.2613500
27. Yu, J., Liu, L., Wang, L., Tan, M., Xu, D.: Turning control of a multilink biomimetic robotic fish. IEEE Trans. Robot. **24**(1), 201–206 (2008)
28. Ren, Q., Xu, J.-X., Guo,, Z., Ru, Y.: Motion control of a multi-joint robotic fish based on biomimetic learning. In: IEEE 23rd International Symposium on Industrial Electronics (ISIE), pp. 1566–1571 (2014)