Direct Adaptive Iterative Learning Control of Nonlinear Systems Using an Output-Recurrent Fuzzy Neural Network

Ying-Chung Wang, Chiang-Ju Chien, and Ching-Cheng Teng

***Abstract*—In this paper, a direct adaptive iterative learning control (DAILC) based on a new output-recurrent fuzzy neural network (ORFNN) is presented for a class of repeatable nonlinear systems with unknown nonlinearities and variable initial resetting errors. In order to overcome the design difficulty due to initial state errors at the beginning of each iteration, a concept of time-varying boundary layer is employed to construct an error equation. The learning controller is then designed by using the given ORFNN to approximate an optimal equivalent controller. Some auxiliary control components are applied to eliminate approximation error and ensure learning convergence. Since the optimal ORFNN parameters for a best approximation are generally unavailable, an adaptive algorithm with projection mechanism is derived to update all the consequent, premise, and recurrent parameters during iteration processes. Only one network is required to design the ORFNN-based DAILC and the plant nonlinearities, especially the nonlinear input gain, are allowed to be totally unknown. Based on a Lyapunov-like analysis, we show that all adjustable parameters and internal signals remain bounded for all iterations. Furthermore, the norm of state tracking error vector will asymptotically converge to a tunable residual set as iteration goes to infinity. Finally, iterative learning control of two nonlinear systems, inverted pendulum system and Chua's chaotic circuit, are performed to verify the tracking performance of the proposed learning scheme.**

***Index Terms*—Direct adaptive control, iterative learning control, nonlinear systems, output-recurrent fuzzy neural network.**

I. INTRODUCTION

DURING the past two decades, iterative learning control (ILC) has been known to be one of the most effective control strategies in dealing with repeated tracking control or periodic disturbance rejection for nonlinear dynamic systems. For instance, the backing up control of a vehicle is a very difficult exercise for most of the beginners because of its high degree of nonlinear uncertainty. In order to become a skilled vehicle driver, the beginner learns how to successfully control the vehicle through a learning process over and over again. This simple example explains the feature and objective of iterative learning control. In general, the ILC system improves its control performance by a self-tuning process without using an accurate system model and can be applied to practical applications such as the control of robotics, servo motors, etc.. To begin with, the D-type, P-type, or PID-type iterative learning controllers [1]–[7] were developed for nonlinear plants with nonlinearities satisfying global Lipschitz continuous condition. Basically, the control input is directly updated by a learning mechanism using the information of error and input in the previous iteration. Due to the difficulty to apply these ILCs for non-Lipschitz nonlinear plants and design of the learning gain depends on the input–output coupling matrix, other new types of ILC algorithms have been widely studied in recent years. One of the most interesting and important developments is the so-called adaptive iterative learning control (AILC) [8]–[11]. The control parameters, instead of the control input itself, are updated between successive iterations in the design of AILC. This concept is similar to typical adaptive control problem. The main difference is that control parameters are tuned along iteration axis, but not time axis. Substantial efforts of AILC have been reported for broader applications to non-Lipschitz nonlinear plants, high relative degree plants, etc. However, as most of the traditional time-domain adaptive controls of nonlinear systems [12], the plant unknown parameters must be linear with respective to some known nonlinear functions in those AILC schemes. The dependence on the structure of plant model unfortunately losses the most important feature of iterative learning system.

To solve the ILC problem for nonlinear systems whose nonlinearities are not Lipschitz continuous or not linearly parameterizable, a powerful strategy is to apply fuzzy system [13] or neural network [14] as a nonlinear approximator when designing the iterative learning controller. Actually, the fuzzy system [15]–[17] or neural-network-based [18]–[20] control strategy has been well known and studied extensively in the time-domain adaptive control of nonlinear systems for many years. Very few results were found in the literature of ILC. Recently, the learning controller based on adaptive fuzzy or neural compensation force were proposed by [21], [22]. As the other existing AILC schemes [8]–[11], the parameters of fuzzy system or neural network are tuned along iteration domain. The adaptive learning controller must guarantee both time domain boundedness and iteration domain convergence due to the special repeated property of ILC. So the controller structure, adaptive law, and stability analysis of fuzzy system or neural-network-based AILC are different from those in traditional adaptive control. Basically, the AILC systems proposed in [21] and [22] can be classified into an indirect scheme [16] since the fuzzy system or neural network is used to model the

Manuscript received December 24, 2002; revised July 22, 2003. This work was supported by the National Science Council of Taiwan, R.O.C., under Grant NSC91-2213-E-211-001. This paper was recommended by Associate Editor H. Qiao.

Y.-C. Wang and C.-C. Teng are with the Department of Electrical and Control Engineering, National Chiao-Tung University, Hsinchu, Taiwan, R.O.C.

C.-J. Chien is with the Department of Electronic Engineering, Huafan University, Shihtin, Taipei County, Taiwan, R.O.C. (e-mail: cjc@huafan.hfu.edu.tw).

Digital Object Identifier 10.1109/TSMCB.2004.824525

1083-4419/04$20.00 © 2004 IEEE

plant nonlinearities. In this paper, we aim to present a direct AILC (DAILC) based on a new output-recurrent fuzzy neural network (ORFNN). The ORFNN-based DAILC scheme is different from [21] and [22] as the ORFNN is used to approximate the certainty equivalent controller but not to model the plant nonlinearities. The main features of this learning controller and its contributions relative to the related works (especially compared with the works in [21] and [22]) are summarized as follows.

1) The robustness against initial state error at the beginning of each iteration is a very special and important issue in ILC. It has been extensively studied and solved for traditional PID-type iterative learning controllers [2]–[7]. For example, in [4] the learned tracking error can be estimated in terms of initial state error and parameters of PD-type ILC algorithm, and in [7] the learning controller can be robust to variable initial shifting by using a finite initial rectifying action. For all the existing AILC systems [8]–[11], [21], [22], initial state error is still a challenging work since it must be exactly zero for stability analysis. It is hard to directly apply those techniques in [2]–[7] to AILC design. In this paper, the proposed DAILC scheme will relax this critical requirement. By introducing a concept of time-varying boundary layer, the initial state errors can be allowed to be nonzero. Moreover, they could be varying at each iteration and not necessarily small. It is shown that the norm of state tracking errors will asymptotically converge to a residual set whose size depends on the width of boundary layer.
2) Consider a simple first-order nonlinear system in the form of $\dot{x}(t) = f(x(t)) + b(x(t))u(t)$ where $x(t)$ is the state and $b(x(t))$ is the input (control) gain function. For the fuzzy system or neural-network-based AILC in [21] and [22] (or even the fuzzy system or neural network-based adaptive control in [15]–[18]), the input gain $b(x(t))$ is always a critical issue for controller design. In addition to the well-known necessary condition $b(x(t)) > 0$ for a controllable plant, some special constrains are usually required on $b(x(t))$ for stability and convergence analysis in the literature. For example, a constant lower bound of $b(x(t))$ should be known in [21] and [22], the lower and upper bounding functions of $b(x(t))$ should exist and be known in [16], or the time derivative of $b^{-1}(x(t))$ is bounded by a known positive function in [17] and [21], and so on. Furthermore, two fuzzy systems or neural networks are often used to estimate both $f(x(t))$ and $b(x(t))$ when designing the adaptive iterative learning controller or adaptive controller. This leads to a more complicated control structure since two sets of parameters are needed to be tuned. In our ORFNN-based DAILC scheme, only one network is required for nonlinear compensation and $b(x(t))$ is allowed to be totally unknown. The only mild requirement on $b(x(t))$ is that it is bounded away from zero.
3) As we know, both the fuzzy system and neural network are to mimic human-like knowledge processing capability. Therefore, the fuzzy neural network (FNN) [23], [24], a combination of the above two approaches, has become a popular research topic in a variety of applications. However, the application domain of FNN is limited to static problems, and it usually processes dynamic problems inefficiently. Hence, the concept of recurrent FNN (RFNN) [27]–[30] was then proposed in the literature. The RFNN, which in general has the internal feedback connections, captures the dynamic response of system without using external feedback through delays so that it has the superior dynamic mapping capability than FNN. In this paper, besides the main contributions to the field of ILC, we also present a new ORFNN to design our learning controller. This ORFNN is a dynamic mapping function with global feedback structure and has a smaller number of network parameters when compared with some existing RFNN [27], [29], [30] if the network input, output, and rule numbers are the same. The ORFNN will reduce to be a static FNN if the recurrent weight is set to be zero. By using the technique of Taylor series expansion, the approximation error between the estimated ORFNN and an optimal ORFNN can be expressed in a linearly parameterized form modulo a residual term linear with some known functions. This fact enables us to derive adaptive laws such that all the consequent, premise, and recurrent parameters can be tuned during the iteration processes.

This paper is organized as follows. In Section II, the plant description, control objective, and design steps of the proposed ORFNN-based DAILC are presented. Analysis of closed-loop stability and learning performance will be studied extensively in Section III. To demonstrate the learning effects of this DAILC, two examples, including a repetitive tracking control of an inverted pendulum system and Chua's chaotic circuit, are used for computer simulation in Section IV. Finally, a conclusion is made in Section V. The detailed description of the proposed ORFNN is given in the Appendix.

II. ORFNN-Based Direct Adaptive Iterative Learning Controller

In this paper, we consider a nonlinear system which can perform a given task repeatedly over a finite time interval $[0, T]$ as follows:

$$\begin{aligned} \dot{x}_1^j(t) &= x_2^j(t) \\ \dot{x}_2^j(t) &= x_3^j(t) \\ &\vdots \\ \dot{x}_n^j(t) &= -f(X^j(t)) + b(X^j(t))u^j(t) \end{aligned} \quad (1)$$

where $X^j(t) = [x_1^j(t), \ldots, x_n^j(t)]^\top \in I\!R^{n\times 1} \times [0, T]$ is the state vector of the system, $u^j(t)$ is the control input, and $f(X^j(t))$ and $b(X^j(t))$ are unknown real continuous nonlinear functions of state, respectively. Here, j denotes the index of iteration and $t \in [0, T]$. The iterative learning control objective for a repeatable system is different from a typical control problem since the control domain is only a finite time interval. Given a specified desired trajectory $X_d(t) = [x_d(t), \dot{x}_d(t), \ldots, x_d^{(n-1)}(t)]^\top$, $t \in [0, T]$ and a possible initial resetting error $X_d(0) \neq X^j(0)$ for all $j \geq 1$, the control objective is to force the state vector $X^j(t)$ to follow $X_d(t)$ such that $\lim_{j\to\infty}\int_0^T \|X^j(t)-X_d(t)\|\,dt \le \epsilon$ for some small positive error tolerance bound ϵ. In order to achieve this control objective, some assumptions on the nonlinear system and desired trajectory are given as follows.

A1) The input gain function satisfies $b(X^j(t)) > 0$ for all $X^j(t) \in I\!R^{n\times 1}\times[0,T]$ and $j \ge 1$.

A2) Define state errors as $e_1^j(t) = x_1^j(t) - x_d(t), e_2^j(t) = \dot{x}_1^j(t) - \dot{x}_d(t), \ldots, e_n^j(t) = x_1^{(n-1),j}(t) - x_d^{(n-1)}(t)$. The initial state errors at each iteration are not necessarily zero, small, and fixed, but are assumed to satisfy $|e_i^j(0)| \le \varepsilon_i$ for some known positive constants $\varepsilon_i, i = 1,\ldots n$.

A3) The desired state trajectory vector $X_d(t) = [x_d(t), \dot{x}_d(t), \ldots, x_d^{(n-1)}(t)]^\top$ is bounded.

In order for the system (1) to be controllable, we require $b(X^j(t)) \ne 0$ for $X^j(t) \in I\!R^{n\times 1}\times[0,T], j \ge 1$, and without loss of generality we assume that $b(X^j(t)) > 0$ in assumption (A1). This is almost a necessary condition for most of the researches dealing with the similar control problem in iteration domain [21], [22] or in time domain [15], [17], [18], [26], [29], [31], [32]. One of the main contributions in this paper will be to show that the estimation of $b(X^j(t))$ or the bounding information on $b(X^j(t))$ is not required in our learning scheme. The condition on initial state errors is given in assumption (A2). In general, they can be varying and large. The only requirement is their upper bounds which are used for controller design. Now, in order to illustrate the idea of the learning controller, we use the following three steps to explain the design approach.

Step 1: Using the well-known design approach in most of the adaptive control for this class of nonlinear affine system (1), we first design a control function $s^j(t)$ as a linear combination of all the state tracking errors, i.e.,

$$s^j(t) = c_1e_1^j(t) + c_2e_2^j(t) + \cdots + c_{n-1}e_{n-1}^j(t) + e_n^j(t) \tag{2}$$

where $c_1,\ldots,c_{n-1}$ are the coefficients of a Hurwitz polynomial $\Delta(D) = D^{n-1} + c_{n-1}D^{n-2} + \cdots + c_1$. It is clear that if the learning controller can drive $s^j(t)$ to zero for all $t \in [0,T]$, then the state tracking errors will also asymptotically converge to zero for all $t \in [0,T]$. However, it is impossible since $s^j(0) \ne 0$. To overcome the uncertainty from initial state errors, let ε^* be the upper bound on $|s^j(0)|$, i.e., $|s^j(0)| \le c_1\varepsilon_1 + c_2\varepsilon_2 + \cdots + \varepsilon_n \equiv \varepsilon^*$, and define a new error function $s_\phi^j(t)$ as follows:

$$\begin{aligned} s_\phi^j(t) &= s^j(t) - \phi(t)\,\mathbf{sat}\left(\frac{s^j(t)}{\phi(t)}\right)\\ \phi(t) &= \varepsilon^* e^{-kt}, \quad k > 0. \end{aligned} \tag{3}$$

In (3), **sat** is the saturation function defined as

$$\mathbf{sat}\left(\frac{s^j(t)}{\phi(t)}\right) = \begin{cases} 1, & \text{if } s^j(t) > \phi(t)\\ \frac{s^j(t)}{\phi(t)}, & \text{if } |s^j(t)| \le \phi(t)\\ -1, & \text{if } s^j(t) < -\phi(t)\end{cases}$$

and $\phi(t)$ is the width of the boundary layer which is time-varying depending on time t, but not related to the iteration number j. Now $s_\phi^j(t)$ will play the main role in our controller design since it can be easily shown that $s_\phi^j(0) = 0, \forall j \ge 1$. If $\lim_{j\to\infty} s_\phi^j(t) = 0, \forall t \in [0,T]$ and $\phi(t)$ is small, then the learning performance will be satisfied since $\lim_{j\to\infty}|s^j(t)| \le \phi(t)$. To find the approach for the controller design later, we first derive the time derivative of $(s_\phi^j(t))^2$ as follows:

$$\begin{aligned} &\frac{d}{dt}\left(s_\phi^j(t)\right)^2\\ &\quad= 2s_\phi^j(t)\left(\dot{s}^j(t) - \mathbf{sgn}\left(s_\phi^j(t)\right)\dot{\phi}(t)\right)\\ &\quad= 2s_\phi^j(t)\left\{\sum_{i=1}^{n-1} c_ie_{i+1}^j(t) - x_d^{(n)}(t) - f(X^j(t))\right.\\ &\qquad\left. + b(X^j(t))u^j(t) - \mathbf{sgn}\left(s_\phi^j(t)\right)\dot{\phi}(t)\right\} \end{aligned} \tag{4}$$

where **sgn** is the notation for the sign function. If the nonlinear functions $f(X^j(t))$ and $b(X^j(t))$ are completely known, we can define the certainty equivalent controller as

$$u_\star^j(t) = \frac{f(X^j(t)) + x_d^{(n)}(t) - \sum_{i=1}^{n-1} c_ie_{i+1}^j(t) - ks^j(t)}{b(X^j(t))} \tag{5}$$

with the positive constant k the same as that in (3). Then substituting (5) into (4), it yields

$$\begin{aligned} &\frac{d}{dt}\left(s_\phi^j(t)\right)^2\\ &\quad= 2s_\phi^j(t)\left(-ks^j(t) - \mathbf{sgn}\left(s_\phi^j(t)\right)\dot{\phi}(t)\right)\\ &\quad= 2s_\phi^j(t)\left\{-ks_\phi^j(t) - k\phi(t)\mathbf{sat}\left(\frac{s^j(t)}{\phi(t)}\right)\right.\\ &\qquad\left. - \mathbf{sgn}\left(s_\phi^j(t)\right)\dot{\phi}(t)\right\}\\ &\quad= -2k\left(s_\phi^j(t)\right)^2 - 2\left|s_\phi^j(t)\right|\left(\dot{\phi}(t) + k\phi(t)\right)\\ &\quad= -2k\left(s_\phi^j(t)\right)^2 \end{aligned} \tag{6}$$

since $s_\phi^j(t)\mathbf{sat}((s^j(t))/(\phi(t))) = s_\phi^j(t)\mathbf{sgn}(s_\phi^j(t)) = |s_\phi^j(t)|$. This implies $s_\phi^j(t) = 0$ for all $t \in [0,T]$ and $j \ge 1$ since $s_\phi^j(0) = 0$. However, $f(X^j(t))$ and $b(X^j(t))$ are in general unknown or only partially known. Hence, the result of (6) can not be achieved. According to the certainty equivalent controller (5), (4) can only be rewritten as

$$\begin{aligned} &\frac{d}{dt}\left(s_\phi^j(t)\right)^2\\ &\quad= 2s_\phi^j(t)\left\{\sum_{i=1}^{n-1} c_ie_{i+1}^j(t) - x_d^{(n)}(t) - f(X^j(t))\right.\\ &\qquad\left. + b(X^j(t))u^j(t) - \mathbf{sgn}\left(s_\phi^j(t)\right)\dot{\phi}(t)\right\}\\ &\quad= -2k\left(s_\phi^j(t)\right)^2 + 2s_\phi^j(t)b(X^j(t))\left(u^j(t) - u_\star^j(t)\right). \end{aligned} \tag{7}$$

Remark 1: Equation (7) will be considered as the error equation for our controller design later. The main control objective will be achieved if we can drive $s_\phi^j(t)$ to zero for all $t \in [0,T]$ as learning iterations are large enough. It is emphasized that the careful choice of boundary layer with $s_\phi^j(0) = 0$ is quite different from traditional approaches for variable structure control and will be very important to ILC analysis. Also different from typical adaptive control design, the control purpose of ILC is to ensure $\lim_{j\to\infty} s_\phi^j(t) = 0, \forall t \in [0,T]$, but not $\lim_{t\to\infty} s_\phi^j(t) = 0$. In the next step, an ORFNN-based DAILC will be proposed to compensate for both of the unknown certainty equivalent controller $u_\star^j(t)$ and the ORFNN approximation error in order to achieve the control purpose.

Step 2: Consider the ORFNN given in Appendix A and Appendix B, the proposed ORFNN-based DAILC is designed as

$$u^j(t) = u_{L_1}^j(t) + u_{L_2}^j(t) \tag{8}$$

with

$$\begin{aligned} u_{L_1}^j(t) &= O^{(4)}\left(O^{(4)^{j-1}}(t), X^j(t), W^j(t), m^j(t), \sigma^j(t), \gamma^j(t)\right) \\ &\quad - \mathbf{sat}\left(\frac{s^j(t)}{\phi(t)}\right)\theta^{j\top}(t)Y^j(t) \end{aligned} \tag{9}$$

$$\begin{aligned} u_{L_2}^j(t) &= -\delta_w s_\phi^j(t)\Big(O^{(3)^j}(t) - O_m^{(3)'^j}(t)m^j(t) \\ &\quad - O_\sigma^{(3)'^j}(t)\sigma^j(t) - O_\gamma^{(3)'^j}(t)\gamma^j(t)\Big)^\top \\ &\quad \times \Big(O^{(3)^j}(t) - O_m^{(3)'^j}(t)m^j(t) \\ &\quad - O_\sigma^{(3)'^j}(t)\sigma^j(t) - O_\gamma^{(3)'^j}(t)\gamma^j(t)\Big) \\ &\quad - \delta_m s_\phi^j(t)\left(W^{j\top}(t)O_m^{(3)'^j}(t)\right)\left(W^{j\top}(t)O_m^{(3)'^j}(t)\right)^\top \\ &\quad - \delta_\sigma s_\phi^j(t)\left(W^{j\top}(t)O_\sigma^{(3)'^j}(t)\right)\left(W^{j\top}(t)O_\sigma^{(3)'^j}(t)\right)^\top \\ &\quad - \delta_\gamma s_\phi^j(t)\left(W^{j\top}(t)O_\gamma^{(3)'^j}(t)\right)\left(W^{j\top}(t)O_\gamma^{(3)'^j}(t)\right)^\top \\ &\quad - \delta_\theta s_\phi^j(t)Y^{j\top}(t)Y^j(t) \end{aligned} \tag{10}$$

where $\delta_w, \delta_m, \delta_\sigma, \delta_\gamma, \delta_\theta > 0$ are the learning gains, $W^j(t) \in I\!R^{M\times 1} \times [0,T]$, $m^j(t) \in I\!R^{N\times 1} \times [0,T]$, $\sigma^j(t) \in I\!R^{N\times 1} \times [0,T]$, $\gamma^j(t) \in I\!R \times [0,T]$ are the network parameter vectors and $\theta^j(t) \in I\!R^{5\times 1} \times [0,T]$ is the robust control parameter vector. In this controller, $u_{L_1}^j(t)$ is designed to compensate for the unknown certainty equivalent controller $u_\star^j(t)$ and approximation error $\eta^j(t)$ given in (35). $u_{L_2}^j(t)$ is a feedback stabilization component which will be clear in later convergence and stability analysis. Now if we do not consider the effect of $u_{L_2}^j(t)$ in this moment and substitute (9) into (7), we have

$$\begin{aligned} &\frac{1}{b(X^j(t))}\frac{d}{dt}\left(s_\phi^j(t)\right)^2 \\ &= -\frac{2k}{b(X^j(t))}\left(s_\phi^j(t)\right)^2 + 2s_\phi^j(t)\Big\{\tilde{W}^{j\top}(t)\Big(O^{(3)^j}(t) \\ &\quad - O_m^{(3)'^j}(t)m^j(t) - O_\sigma^{(3)'^j}(t)\sigma^j(t) - O_\gamma^{(3)'^j}(t)\gamma^j(t)\Big) \\ &\quad + W^{j\top}(t)\Big(O_m^{(3)'^j}(t)\tilde{m}^j(t) + O_\sigma^{(3)'^j}(t)\tilde{\sigma}^j(t) \\ &\quad + O_\gamma^{(3)'^j}(t)\tilde{\gamma}^j(t)\Big) - \mathbf{sat}\left(\frac{s^j(t)}{\phi(t)}\right)\theta^{j\top}(t)Y^j(t) \\ &\quad + \eta^j(t) + u_{L_2}^j(t)\Big\} \\ &\le -\frac{2k}{b(X^j(t))}\left(s_\phi^j(t)\right)^2 + 2s_\phi^j(t)\tilde{W}^{j\top}(t)\Big(O^{(3)^j}(t) \\ &\quad - O_m^{(3)'^j}(t)m^j(t) - O_\sigma^{(3)'^j}(t)\sigma^j(t) - O_\gamma^{(3)'^j}(t)\gamma^j(t)\Big) \\ &\quad + 2s_\phi^j(t)W^{jT}(t)\Big(O_m^{(3)'^j}(t)\tilde{m}^j(t) + O_\sigma^{(3)'^j}(t)\tilde{\sigma}^j(t) \\ &\quad + O_\gamma^{(3)'^j}(t)\tilde{\gamma}^j(t)\Big) - 2\left|s_\phi^j(t)\right|\tilde{\theta}^{jT}(t)Y^j(t) \\ &\quad + 2s_\phi^j(t)u_{L_2}^j(t) \end{aligned} \tag{11}$$

by using the results of (35) and (36) given in Appendix B.

Remark 2: Since the optimal parameters $W^*, m^*, \sigma^*, \gamma^*$, and θ^* for an optimal approximation are generally unknown, the weights of ORFNN, $W^j(t), m^j(t)$, $\sigma^j(t)\gamma^j(t)$ and parameters $\theta^j(t)$ at time t of the jth iteration will be tuned via some suitable adaptive laws between successive iteration. A set of stable adaptation algorithms is necessary to update the parameters such that closed loop stability is guaranteed and learning performance is improved.

Step 3: The adaptation algorithms for control parameters $W^{j+1}(t), m^{j+1}(t), \sigma^{j+1}(t), \gamma^{j+1}(t)$, and $\theta^{j+1}(t)$ at the (next) $j+1$th iteration are given as

$$\begin{aligned} W_p^{j+1}(t) = W^j(t) - \delta_w s_\phi^j(t)\Big(&O^{(3)^j}(t) - O_m^{(3)'^j}(t)m^j(t) \\ &- O_\sigma^{(3)'^j}(t)\sigma^j(t) - O_\gamma^{(3)'^j}(t)\gamma^j(t)\Big) \end{aligned} \tag{12}$$

$$m_p^{j+1}(t) = m^j(t) - \delta_m s_\phi^j(t)\left(W^{j\top}(t)O_m^{(3)'^j}(t)\right)^\top \tag{13}$$

$$\sigma_p^{j+1}(t) = \sigma^j(t) - \delta_\sigma s_\phi^j(t)\left(W^{j\top}(t)O_\sigma^{(3)'^j}(t)\right)^\top \tag{14}$$

$$\gamma_p^{j+1}(t) = \gamma^j(t) - \delta_\gamma s_\phi^j(t)\left(W^{j\top}(t)O_\gamma^{(3)'^j}(t)\right)^\top \tag{15}$$

$$\theta_p^{j+1}(t) = \theta^j(t) + \delta_\theta\left|s_\phi^j(t)\right|Y^j(t) \tag{16}$$

and

$$W^{j+1}(t) = \text{proj}\left(W_p^{j+1}(t)\right) \tag{17}$$

$$m^{j+1}(t) = \text{proj}\left(m_p^{j+1}(t)\right) \tag{18}$$

$$\sigma^{j+1}(t) = \text{proj}\left(\sigma_p^{j+1}(t)\right) \tag{19}$$

$$\gamma^{j+1}(t) = \text{proj}\left(\gamma_p^{j+1}(t)\right) \tag{20}$$
$$\theta^{j+1}(t) = \text{proj}\left(\theta_p^{j+1}(t)\right) \tag{21}$$

where proj denotes the projection mechanism

$$\text{proj}\big(z_p^{j+1}(t)\big) = \begin{cases} \bar{z}, & \text{if } z_p^{j+1}(t) \geq \bar{z} \\ -\bar{z}, & \text{if } z_p^{j+1}(t) \leq -\bar{z} \\ z_p^{j+1}(t), & \text{otherwise} \end{cases}$$

with $\bar{z}$ being the upper bound of $|z^*|$ (z^* belongs to an element of $\{W^*, m^*, \sigma^*, \gamma^*, \theta^*\}$).

Remark 3: According to the projection algorithm, it is noted that the parameter errors will be bounded for all iterations and for all $t \in [0, T]$. The adaptation process is realized between successive iteration so that the convergence could be guaranteed along the iteration domain. This is the main reason to develop an ORFNN with iteration domain delay for our control task. Due to this ORFNN, the network parameters at the $(j+1)$th iteration will not only depend on the parameters and system information at the jth iteration, but also depend on those at the $(j-1)$th, $(j-2)$th $\cdots$ iterations.

Before analyzing stability and convergence of the ORFNN-based DAILC system, we summarize the design procedures as follows.

D1) Construct the control function $s^j(t)$ as in (2) and error function $s_\phi^j(t)$ as in (3).

D2) Design the controller $u^j(t)$ as in (8) with the iterative learning components as in (9) and (10). The fuzzy rule base of the ORFNN in $u_{L_1}^j(t)$ is given by

$$\text{IF } x_1^j(t) \text{ is } A_{(1,k_1)} \text{ and } \cdots \text{ and } x_n^j(t) \text{ is } A_{(n,k_n)},$$
$$\text{THEN } O^{(4)^j}(t) \text{ is } B_{(k_1,\ldots,k_n)}.$$

D3) Update the control parameters $W^j(t), m^j(t), \sigma^j(t), \gamma^j(t)$ and $\theta^j(t)$ for the next iteration by using the adaptation algorithms as in (12)–(21).

III. Analysis of Stability and Convergence

Define the projected parameter error as $\tilde{z}^j(t) = z^j(t) - z^*$ and unprojected parameter error as $\tilde{z}_p^j(t) = z_p^j(t) - z^*$ where $z \in \{W, m, \sigma, \gamma, \theta\}$. Then we have $\tilde{W}_p^{j\top}(t)\tilde{W}_p^j(t) \geq \tilde{W}^{j\top}(t)\tilde{W}^j(t)$, $\tilde{m}_p^{j\top}(t)\tilde{m}_p^j(t) \geq \tilde{m}^{j\top}(t)\tilde{m}^j(t)$, $\tilde{\sigma}_p^{j\top}(t)\tilde{\sigma}_p^j(t) \geq \tilde{\sigma}^{j\top}(t)\tilde{\sigma}^j(t)$, $(\tilde{\gamma}_p^j(t))^2 \geq (\tilde{\gamma}^j(t))^2$, and $\tilde{\theta}_p^{j\top}(t)\tilde{\theta}_p^j(t) \geq \tilde{\theta}^{j\top}(t)\tilde{\theta}^j(t)$. Furthermore, it is easy to show by subtracting the optimal control gains on both sides of (12)–(16), such that

$$\tilde{W}_p^{j+1}(t) = \tilde{W}^j(t) - \delta_w s_\phi^j(t)\left(O^{(3)^j}(t) - O_m^{(3)'^j}(t)m^j(t) - O_\sigma^{(3)'^j}(t)\sigma^j(t) - O_\gamma^{(3)'^j}(t)\gamma^j(t)\right)$$
$$\tilde{m}_p^{j+1}(t) = \tilde{m}^j(t) - \delta_m s_\phi^j(t)\left(W^{j\top}(t)O_m^{(3)'^j}(t)\right)^\top$$
$$\tilde{\sigma}_p^{j+1}(t) = \tilde{\sigma}^j(t) - \delta_\sigma s_\phi^j(t)\left(W^{jT}(t)O_\sigma^{(3)'^j}(t)\right)^\top$$
$$\tilde{\gamma}_p^{j+1}(t) = \tilde{\gamma}^j(t) - \delta_\gamma s_\phi^j(t)\left(W^{j\top}(t)O_\gamma^{(3)'^j}(t)\right)^\top$$
$$\tilde{\theta}_p^{j+1}(t) = \tilde{\theta}^j(t) + \delta_\theta \left|s_\phi^j(t)\right| Y^j(t).$$

The main results about the closed-loop stability and learning convergence for our proposed ORFNN-based DAILC are now shown in the following theorem.

Main Theorem: Consider the nonlinear system (1) satisfying the assumptions A1)–A3). If we design the ORFNN-based DAILC following the design steps D1)–D3), and define $E^j(t) = [e_1^j(t), e_2^j(t), \ldots, e_{n-1}^j(t)]^\top$, then the following facts will hold.

t1) $\lim_{j\to\infty} s_\phi^j(t) = s_\phi^\infty(t) = 0, \forall t \in [0, T]$.

t2) $\lim_{j\to\infty} |s^j(t)| = |s^\infty(t)| \leq \phi(t) = e^{-kt}\varepsilon^*, \forall t \in [0, T]$.

t3) All adjustable control parameters $W^j(t)$, $m^j(t)$, $\sigma^j(t)$, $\gamma^j(t), \theta^j(t)$ and internal signals $s^j(t)$, $s_\phi^j(t), u^j(t), e_1^j(t), \ldots, e_n^j(t)$ are bounded $\forall t \in [0, T]$ and $\forall j \geq 1$.

t4) Let λ be the positive constant such that $\Delta(D - \lambda)$ is still a Hurwitz polynomial. Then there exists a constant $m_1 > 0$ such that for all $t \in [0, T]$

$$\lim_{j\to\infty} \|E^j(t)\| \leq m_1 e^{-\lambda t}\|E^\infty(0)\| + m_1\varepsilon^* \frac{e^{-kt} - e^{-\lambda t}}{\lambda - k} \tag{22}$$
$$\lim_{j\to\infty} \left|e_n^j(t)\right| \leq \sum_{i=1}^{n-1} c_i \left|e_i^\infty(t)\right| + e^{-kt}\varepsilon^*. \tag{23}$$

Proof: In this proof, the time argument inside the integration will be omitted for the notation brevity unless otherwise specified.

t1) Define the cost functions of performance as

$$V^j(t) = \int_0^t \left[\frac{1}{\delta_w}\tilde{W}^{j\top}\tilde{W}^j + \frac{1}{\delta_m}\tilde{m}^{j\top}\tilde{m}^j + \frac{1}{\delta_\sigma}\tilde{\sigma}^{j\top}\tilde{\sigma}^j + \frac{1}{\delta_\gamma}(\tilde{\gamma}^j)^2 + \frac{1}{\delta_\theta}\tilde{\theta}^{j\top}\tilde{\theta}^j\right] d\tau$$
$$V_p^j(t) = \int_0^t \left[\frac{1}{\delta_w}\tilde{W}_p^{j\top}\tilde{W}_p^j + \frac{1}{\delta_m}\tilde{m}_p^{j\top}\tilde{m}_p^j + \frac{1}{\delta_\sigma}\tilde{\sigma}_p^{j\top}\tilde{\sigma}_p^j + \frac{1}{\delta_\gamma}\left(\tilde{\gamma}_p^j\right)^2 + \frac{1}{\delta_\theta}\tilde{\theta}_p^{j\top}\tilde{\theta}_p^j\right] d\tau$$

then we can derive

$$\begin{aligned} V^{j+1}(t) - V^j(t) &\leq V_p^{j+1}(t) - V^j(t) \\ &= \int_0^t \Bigg[-2s_\phi^j\tilde{W}^{j\top}\left(O^{(3)^j} - O_m^{(3)'^j}m^j - O_\sigma^{(3)'^j}\sigma^j - O_\gamma^{(3)'^j}\gamma^j\right) - 2s_\phi^j\tilde{m}^{j\top}\left(W^{j\top}O_m^{(3)'^j}\right)^\top \\ &\quad - 2s_\phi^j\tilde{\sigma}^{j\top}\left(W^{j\top}O_\sigma^{(3)'^j}\right)^\top \\ &\quad - 2s_\phi^j\tilde{\gamma}^j\left(W^{j\top}O_\gamma^{(3)'^j}\right)^\top + 2\left|s_\phi^j\right|\tilde{\theta}^{j\top}Y^j + \delta_w\left(s_\phi^j\right)^2 \\ &\quad \times \left(O^{(3)^j} - O_m^{(3)'^j}m^j - O_\sigma^{(3)'^j}\sigma^j - O_\gamma^{(3)'^j}\gamma^j\right)^\top \\ &\quad \times \left(O^{(3)^j} - O_m^{(3)'^j}m^j - O_\sigma^{(3)'^j}\sigma^j - O_\gamma^{(3)'^j}\gamma^j\right) \end{aligned}$$

$$+\delta_m\left(s_\phi^j\right)^2\left(W^{j\top}O_m^{(3)'j}\right)\left(W^{j\top}O_m^{(3)'j}\right)^\top$$
$$+\delta_\sigma\left(s_\phi^j\right)^2\left(W^{j\top}O_\sigma^{(3)'j}\right)\left(W^{j\top}O_\sigma^{(3)'j}\right)^\top$$
$$+\delta_\gamma\left(s_\phi^j\right)^2\left(W^{j\top}O_\gamma^{(3)'j}\right)\left(W^{j\top}O_\gamma^{(3)'j}\right)^\top$$
$$\left.+\delta_\theta\left(s_\phi^j\right)^2 Y^{j\top}Y^j\right]d\tau. \tag{24}$$

In order to further analyze (24), we integrate (11) over time interval $[0,t], t\in(0,T]$, as follows:

$$\int_0^t \frac{1}{b(X^j(\tau))}\frac{d}{d\tau}\left(s_\phi^j(\tau)\right)^2 d\tau$$
$$=\int_{(s_\phi^j(0))^2}^{(s_\phi^j(t))^2}\frac{1}{b(X^j(\tau))}d\left(s_\phi^j(\tau)\right)^2$$
$$\le\int_0^t\left[-\frac{2k}{b(X^j)}\left(s_\phi^j\right)^2+2s_\phi^j\tilde{W}^{j\top}\right.$$
$$\times\left(O^{(3)j}-O_m^{(3)'j}m^j-O_\sigma^{(3)'j}\sigma^j-O_\gamma^{(3)'j}\gamma^j\right)$$
$$+2s_\phi^jW^{j\top}\left(O_m^{(3)'j}\tilde{m}^j+O_\sigma^{(3)'j}\tilde{\sigma}^j+O_\gamma^{(3)'j}\tilde{\gamma}^j\right)$$
$$\left.-2\left|s_\phi^j\right|\tilde{\theta}^{j\top}Y^j+2s_\phi^ju_{L_2}^j\right]d\tau. \tag{25}$$

After some direct manipulations on (25) and using $s_\phi^j(0)=0$, we can find

$$\int_0^t\left[-2s_\phi^j\tilde{W}^{j\top}\left(O^{(3)j}-O_m^{(3)'j}m^j-O_\sigma^{(3)'j}\sigma^j-O_\gamma^{(3)'j}\gamma^j\right)\right.$$
$$-2s_\phi^jW^{j\top}O_m^{(3)'j}\tilde{m}^j-2s_\phi^jW^{j\top}O_\sigma^{(3)'j}\tilde{\sigma}^j$$
$$\left.-2s_\phi^jW^{j\top}O_\gamma^{(3)'j}\tilde{\gamma}^j+2\left|s_\phi^j\right|\tilde{\theta}^{j\top}Y^j\right]d\tau$$
$$\le\int_0^t\left[-\frac{2k}{b(X^j)}\left(s_\phi^j\right)^2+2s_\phi^ju_{L_2}^j\right]d\tau$$
$$-\int_0^{(s_\phi^j(t))^2}\frac{1}{b(X^j(\tau))}d\left(s_\phi^j(\tau)\right)^2. \tag{26}$$

The feedback stabilization component $u_{L_2}^j(t)$ in step 2 is now clear if we substitute (10) and (26) into (24), and show that

$$V^{j+1}(t)-V^j(t)$$
$$\le\int_0^t\left[-\frac{2k}{b(X^j)}\left(s_\phi^j\right)^2+2s_\phi^ju_{L_2}^j+\delta_w\left(s_\phi^j\right)^2\right.$$
$$\times\left(O^{(3)j}-O_m^{(3)'j}m^j-O_\sigma^{(3)'j}\sigma^j-O_\gamma^{(3)'j}\gamma^j\right)^\top$$
$$\times\left(O^{(3)j}-O_m^{(3)'j}m^j-O_\sigma^{(3)'j}\sigma^j-O_\gamma^{(3)'j}\gamma^j\right)$$
$$+\delta_m\left(s_\phi^j\right)^2\left(W^{j\top}O_m^{(3)'j}\right)\left(W^{j\top}O_m^{(3)'j}\right)^\top$$
$$+\delta_\sigma\left(s_\phi^j\right)^2\left(W^{j\top}O_\sigma^{(3)'j}\right)\left(W^{j\top}O_\sigma^{(3)'j}\right)^\top$$
$$+\delta_\gamma\left(s_\phi^j\right)^2\left(W^{j\top}O_\gamma^{(3)'j}\right)\left(W^{j\top}O_\gamma^{(3)'j}\right)^\top$$
$$\left.+\delta_\theta\left(s_\phi^j\right)^2 Y^{j\top}Y^j\right]d\tau$$
$$-\int_0^{(s_\phi^j(t))^2}\frac{1}{b(X^j(\tau))}d\left(s_\phi^j(\tau)\right)^2$$
$$=\int_0^t\left[-\frac{2k}{b(X^j)}\left(s_\phi^j\right)^2-\delta_w\left(s_\phi^j\right)^2\right.$$
$$\times\left(O^{(3)j}-O_m^{(3)'j}m^j-O_\sigma^{(3)'j}\sigma^j-O_\gamma^{(3)'j}\gamma^j\right)^\top$$
$$\times\left(O^{(3)j}-O_m^{(3)'j}m^j-O_\sigma^{(3)'j}\sigma^j-O_\gamma^{(3)'j}\gamma^j\right)$$
$$-\delta_m\left(s_\phi^j\right)^2\left(W^{j\top}O_m^{(3)'j}\right)\left(W^{j\top}O_m^{(3)'j}\right)^\top$$
$$-\delta_\sigma\left(s_\phi^j\right)^2\left(W^{j\top}O_\sigma^{(3)'j}\right)\left(W^{j\top}O_\sigma^{(3)'j}\right)^\top$$
$$-\delta_\gamma\left(s_\phi^j\right)^2\left(W^{j\top}O_\gamma^{(3)'j}\right)\left(W^{j\top}O_\gamma^{(3)'j}\right)^\top$$
$$\left.-\delta_\theta\left(s_\phi^j\right)^2 Y^{j\top}Y^j\right]d\tau-\int_0^{(s_\phi^j(t))^2}\frac{1}{b(X^j(\tau))}d\left(s_\phi^j(\tau)\right)^2$$
$$\le-\int_0^{(s_\phi^j(t))^2}\frac{1}{b(X^j(\tau))}d\left(s_\phi^j(\tau)\right)^2. \tag{27}$$

Thus, we have

$$\int_0^{(s_\phi^j(t))^2}\frac{1}{b(X^j(\tau))}d\left(s_\phi^j(\tau)\right)^2\le V^j(t)-V^{j+1}(t)\le V^1(t)$$

for any iteration $j\ge 1$. Note that $V^1(t)$ is bounded $\forall t\in[0,T]$ due to projection algorithms (17)–(21) and $V^j(t)$ converges to some positive function since $V^j(t)$ is positive definite and monotonically decreasing by the fact of (27). Hence, $V^{j+1}(t)-V^j(t)$ converges to zero and

$$\lim_{j\to\infty}\int_0^{(s_\phi^j(t))^2}\frac{1}{b(X^j(\tau))}d\left(s_\phi^j(\tau)\right)^2=0,\quad\forall\, t\in[0,T].$$

Therefore, we have $\lim_{j\to\infty}s_\phi^j(t)=s_\phi^\infty(t)=0,\forall\, t\in[0,T]$, according to assumption A1) that $b(X^j(t))>0$ for all $t\in[0,T]$ and $j\ge 1$. This proves t1) of the main theorem.

t2) The boundedness of $s^j(t)$ at each iteration over $[0,T]$ can be concluded from (3) because $\phi(t)$ is always bounded and the bound of $s^\infty(t)$ will satisfy

$$\lim_{j\to\infty}|s^j(t)|=|s^\infty(t)|\le\phi(t)=e^{-kt}\varepsilon^*,\quad\forall\, t\in[0,T]$$

This proves t2) of the main theorem.

t3) Boundedness of $s^j(t)$ implies boundedness of $e_1^j(t),\ldots,e_n^j(t)$. Together with the fact that all the adjustable parameters are bounded, t3) is guaranteed.

t4) To find the learning performance of each state tracking error at the final iteration, we consider the following state–space equation:

$$\dot{E}^\infty(t)=A_cE^\infty(t)+B_cs^\infty(t) \tag{28}$$

where

$$A_c = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -c_1 & -c_2 & -c_3 & \cdots & -c_{n-1} \end{bmatrix}$$

$$B_c = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}.$$

by using assumption (A2) and the definition of control function $s^j(t)$ in (2). Solution of (28) in the time domain is given by

$$E^{\infty}(t) = e^{A_c t} E^{\infty}(0) + \int_0^t e^{A_c(t-\tau)} B_c s^{\infty}(\tau)\, d\tau \tag{29}$$

where the state transition matrix $e^{A_c t}$ satisfies $\|e^{A_c t}\| \le m_1 e^{-\lambda t}$ for some suitable positive constant m_1. Taking norms on (29), it yields

$$\begin{aligned} \|E^{\infty}(t)\| &\le m_1 e^{-\lambda t}\|E^{\infty}(0)\| + m_1 \int_0^t e^{-\lambda(t-\tau)} \|B_c\| |s^{\infty}(\tau)|\, d\tau \\ &\le m_1 e^{-\lambda t}\|E^{\infty}(0)\| + m_1 \int_0^t e^{-\lambda(t-\tau)} e^{-k\tau} \varepsilon^*\, d\tau \\ &= m_1 e^{-\lambda t}\|E^{\infty}(0)\| + m_1 \varepsilon^* \frac{e^{-kt} - e^{-\lambda t}}{\lambda - k} \end{aligned}$$

which shows (22). Finally, tracking performance of $e_n^{\infty}(t)$ given in (23) can be easily found by using (2). This concludes t4) of the main theorem. Q.E.D.

Remark 4: In the main theorem, we show that $s_{\phi}^j(t)$ converges to zero as $j \to \infty$, and hence, $\lim_{j\to\infty} |s^j(t)| = |s^{\infty}(t)| \le \phi(t) = e^{-kt}\varepsilon^*, \forall t \in [0, T]$. Since the initial state errors (or equivalently, ε^*) may be large and the requirement of $\phi(0) = \varepsilon^*$ is necessary, it is not practical to set the boundary layer as a constant, i.e., $\phi(t) = \phi(0) = \varepsilon^*, \forall t \in [0, T]$, when ε^* is large. Actually, $\phi(t)$ should be as small as possible because $|s^{\infty}(t)| \le \phi(t)$. If $|s^{\infty}(t)|$ is small enough, the learning performance of each $e_i^{\infty}(t), i = 1, \ldots, n$, which is directly related to $s^{\infty}(t)$, can be guaranteed. This is the main reason for choosing a time varying boundary layer $\phi(t) = e^{-kt}\varepsilon^*$ which will decrease along the time axis. In this design, the parameter k plays an important role of reducing tracking error at the final iteration. In general, $\int_0^T |s^{\infty}(t)|\, dt$ and hence, $\int_0^T |e_i^{\infty}(t)|\, dt, i = 1, \ldots, n$ can be small if k is large.

Remark 5: In addition to the parameter k, the design parameters $\delta_w, \delta_m, \delta_\sigma, \delta_\gamma$ and δ_θ, defined as the learning gains, play another important roles in this ORFNN-based DAILC system. In general, it is only required to set these gains as positive constants. Due to the result shown in (27), the convergent speed of $s_{\phi}^j(t)$ and $s^j(t)$ will increase if $\delta_w, \delta_m, \delta_\sigma, \delta_\gamma$, and δ_θ are large. The effects of design parameters k and learning gains on the improvement of learning performances will be clearly shown in the following simulation examples.

IV. Simulation Examples

Example 1: Consider the inverted pendulum system [16] with a state equation of

$$\begin{aligned} \dot{x}_1^j(t) &= x_2^j(t) \\ \dot{x}_2^j(t) &= \frac{g \sin x_1^j(t) - \frac{m\ell (x_2^j(t))^2 \cos x_1^j(t) \sin x_1^j(t)}{m_c + m}}{\ell\left(\frac{4}{3} - \frac{m \cos^2 x_1^j(t)}{m_c + m}\right)} \\ &\quad + \frac{\frac{\cos x_1^j(t)}{m_c + m}}{\ell\left(\frac{4}{3} - \frac{m \cos^2 x_1^j(t)}{m_c + m}\right)} u^j(t) \end{aligned}$$

where $x_1^j(t)$ and $x_2^j(t)$ denote the angular displacement (rad) and velocity (rad/s) of the pole, respectively. In this simulation, the plant parameters are set as $g = 9.8$ m/s^2, $m_c = 1$ kg, $m = 0.1$ kg, and $\ell = 0.5$ m. It is assumed that the angular displacement is limited to $x_1^j(t) < 1.483$ rad (about 85°) so that the input gain $b(X^j(t))$ satisfied $0 < b(X^j(t))) < \infty$, i.e., assumption A1) in Section II is valid. The control objective is to control the state $X^j(t) = [x_1^j(t), x_2^j(t)]^\top$ to track the desired trajectory $X_d(t) = [x_d(t), \dot{x}_d(t)]^\top = [(\pi/30)\sin(t) + (\pi/30)\sin(2t), (\pi/30)\cos(t) + (\pi/15)\cos(2t)]^\top$ for $t \in [0, 10]$. The design steps are given in the following.

D1) The control and error functions are set to be $s^j(t) = c_1 e_1^j(t) + e_2^j(t)$ and $s_{\phi}^j(t) = s^j(t) - \phi(t)\, \mathbf{sat}(s^j(t)/\phi(t))$ with $\phi(t) = \varepsilon^* e^{-kt}$.

D2) Design the controller $u^j(t)$ as in (8) with the two iterative learning control components $u_{L_1}^j(t)$ and $u_{L_2}^j(t)$ as in (9) and (10), respectively. There are four fuzzy rules for the ORFNN in $u_{L_1}^j(t)$ with the initial parameters are chosen as

$$\begin{aligned} m^1(t) &= \left[m_{(1,1)}^1(t), m_{(1,2)}^1(t), m_{(2,1)}^1(t), m_{(2,2)}^1(t)\right]^\top \\ &= [-0.5, 0.5, -0.5, 0.5]^\top \\ \sigma^1(t) &= \left[\sigma_{(1,1)}^1(t), \sigma_{(1,2)}^1(t), \sigma_{(2,1)}^1(t), \sigma_{(2,2)}^1(t)\right]^\top \\ &= [1, 1, 1, 1]^\top \\ \gamma^1(t) &= 0.1 \\ W^1(t) &= \left[w_{(1,1)}^1(t), w_{(1,2)}^1(t), w_{(2,1)}^1(t), w_{(2,2)}^1(t)\right]^\top \\ &= [-1.5, 1.5, -1.5, 1.5]^\top \end{aligned}$$

for all $t \in [0, 10]$. Besides, the initial value of $\theta^1(t)$ in the robust learning component (9) is chosen as

$$\begin{aligned} \theta^1(t) &= \left[\theta_1^1(t), \theta_2^1(t), \theta_3^1(t), \theta_4^1(t), \theta_5^1(t)\right]^\top \\ &= [0.1, 0.1, 0.1, 0.1, 0.1]^\top \end{aligned}$$

for all $t \in [0, 10]$.

D3) Finally, the parameters $m^j(t), \sigma^j(t), W^j(t), \gamma^j(t)$ and $\theta^j(t)$ are updated for the next iteration by using the projection type adaptation algorithms (12)–(21). In general, the upper bounds on the optimal parameters for an arbitrary certainty equivalent control function

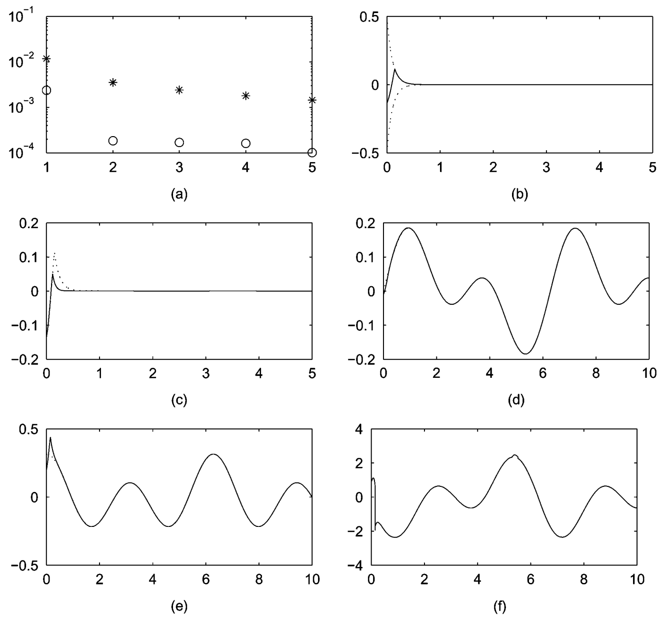

Fig. 1. (a) $\sup_{t\in[0,10]}|s_\phi^j(t)|$ versus iteration j; $*$ for $\text{learning gains} = 10$ and $\circ$ for $\text{learning gains} = 50$. (b) $s^5(t)$ (solid line) and $\phi(t), -\phi(t)$ (dotted lines) versus time $t; k = 10;\text{learning gains} = 50$. (c) $s^5(t)$ with $k = 10$ (dotted line) and $s^5(t)$ with $k = 20$ (solid line) versus time t. (d) $x_1^5(t)$ (solid line) and $x_d(t)$ (dotted line) versus time $t; k = 10$ and $\text{learning gains} = 50$. (e) $x_2^5(t)$ (solid line) and $\dot{x}_d(t)$ (dotted line) versus time $t; k = 10$ and $\text{learning gains} = 50$. (f) $u^5(t)$ versus time $t; k = 10$ and $\text{learning gains} = 50$.

are not easy to estimate. For real implementation of this DAILC, suitable values of the upper bounds are usually selected as large as possible. In this simulation, these upper bounds are all set to be 10.

To begin this simulation, the parameters of k and c_1 are chosen as $k = 10$ and $c_1 = 2$. We assume there always exist fixed initial displacement and velocity errors at the beginning of each iteration. That is, $x_1^j(0) = -0.01, x_2^j(0) = 0.2$ for all $j \geq 1$ so that $e_1^j(0) = x_1^j(0) - x_d(0) = -0.01$ and $e_2^j(0) = x_2^j(0) - \dot{x}_d(0) = 0.2 - 0.3141 = -0.1141$. Since $s^j(0) = c_1 e_1^j(0) + e_2^j(0) = 2 \times (-0.01) + (-0.1141) = -0.1341$, the initial value of $\phi(t)$ is chosen as $\phi(0) = \varepsilon^* = 0.5 > |s^j(0)|$. Fig. 1(a) shows the supremum value of $|s_\phi^j(t)|$ versus iteration j with two different learning gains $\delta_m = \delta_\sigma = \delta_\gamma = \delta_w = \delta_\theta = 10$ and $\delta_m = \delta_\sigma = \delta_\gamma = \delta_w = \delta_\theta = 50$. The asymptotic convergence of $\sup_{t\in[0,10]}|s_\phi^j(t)|$ clearly proves the technical result t1) of the main theorem. In addition, it is found that faster convergent speed is achieved by larger learning gains as commented in Remark 5. In order to demonstrate t2) of the main theorem, we show the trajectory of $s^5(t)$ for the fifth iteration in Fig. 1(b), where the trajectory of $s^5(t)$ is confined between $\phi(t)$ and $-\phi(t)$. This fact not only satisfies t2), but also implies that the transient response of $s^5(t)$ in time domain can be improved by increasing k since $|s^5(t)| \leq \phi(t) = 0.5e^{-kt}$. We show the results of $s^5(t)$ for $k = 10$ and $k = 20$ in Fig. 1(c), respectively. The nice learning performance of $s^j(t)$ directly gives the contribution to the tracking of each state as shown in t4) of the main theorem. The actual tracking behaviors of both states at the fifth iteration for $k = 10$ and $\delta_m = \delta_\sigma = \delta_\gamma = \delta_w = \delta_\theta = 50$ are shown in Figs. 1(d) and (e), respectively. Finally, the bounded control input $u^5(t)$ is demonstrated in Fig. 1(f).

Remark 6: Theoretically, it is more interesting and attractive if all the free parameters of ORFNN can be tuned during the learning process. The price we pay is the complexity of the controller and adaptation law. Actually, as with most of the works, it is easy to simplify our learning controller such that only consequent parameters are updated during learning processes. That is, the basis functions of the ORFNN are fixed. This will drastically reduce the complexity of the controller. We have simulated example 1 again by using this approach. It is interesting to find that the learning system still works well, except the learning speed is a little slower than our proposed scheme. This observation leads to a suggestion for practical implementation. A smart realization is to use an ORFNN with only consequent parameter adaptation; then choose a larger adaptation gain such that a reasonable convergent speed can be guaranteed.

Example 2: The typical Chua's chaotic circuit is a nonlinear oscillator circuit, which displays very rich bifurcation and chaotic phenomena. In this example, we adopt the transformed state equation of Chua's circuit in [32]

$$\begin{aligned}
\dot{x}_1^j(t) &= x_2^j(t)\\
\dot{x}_2^j(t) &= x_3^j(t)\\
\dot{x}_3^j(t) &= \frac{14}{1805}x_1^j(t) - \frac{168}{9025}x_2^j(t) + \frac{1}{38}x_3^j(t)\\
&\quad - \frac{2}{45}\left(\frac{28}{361}x_1^j(t) + \frac{7}{95}x_2^j(t) + x_3^j(t)\right)^3 + u^j(t).
\end{aligned}$$

As Chua's chaotic circuit is sensitive to the initial condition, it is a challenge for an iterative learning control problem. The control objective is to control the state vector $X^j(t) = [x_1^j(t), x_2^j(t), x_3^j(t)]^\top$ to track the desired trajectory $X_d(t) = [x_d(t), \dot{x}_d(t), \ddot{x}_d(t)]^\top = [1.5\sin(t), 1.5\cos(t), -1.5\sin(t)]^\top$ for $t \in [0, 10]$. The design steps of the ORFNN-based DAILC are roughly described as follows.

D1) Define $s^j(t) = c_1 e_1^j(t) + c_2 e_2^j(t) + e_3^j(t)$ where $e_1^j(t) = x_1^j(t) - 1.5\sin t, e_2^j(t) = x_2^j(t) - 1.5\cos t, e_3^j(t) = sx_3^j(t) + 1.5\sin t$, and $s_\phi^j(t) = s^j(t) - \phi(t)\mathbf{sat}(s^j(t)/s\phi(t))$ with $\phi(t) = \varepsilon^* e^{-kt}$.

D2) We choose eight fuzzy rules for the ORFNN in $u_{L_1}^j(t)$ in this case. The initial parameters of the controller are chosen as

$$\begin{aligned}
m^1(t) &= [-5, 5, -5, 5, -5, 5]^\top\\
\sigma^1(t) &= [1, 1, 1, 1, 1, 1]^\top\\
\gamma^1(t) &= 0.1\\
W^1(t) &= [-5, 5, -5, 5, -5, 5, -5, 5]^\top\\
\theta^1(t) &= [0.1, 0.1, 0.1, 0.1, 0.1]^\top
\end{aligned}$$

for all $t \in [0, 10]$.

D3) The projection-type adaptation algorithms are adopted to update the parameters with upper bounds on the optimal parameters being 10.

Due to the chaotic feature of Chua's circuit, we especially study the effect of varying initial state errors on the learning performance in this example. Also, the magnitude

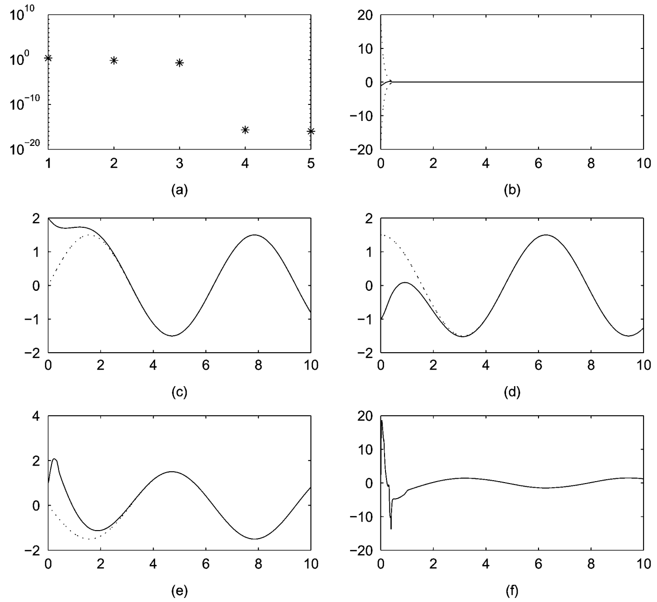

Fig. 2. (a) $\sup_{t\in[0,10]}|s_\phi^j(t)|$ versus iteration j. (b) $s^5(t)$ (solid line) and $\phi(t), -\phi(t)$ (dotted lines) versus time t. (c) $x_1^5(t)$ (solid line) and $x_d(t)$ (dotted line) versus time t. (d) $x_2^5(t)$ (solid line) and $\dot{x}_d(t)$ (dotted line) versus time t. (e) $x_3^5(t)$ (solid line) and $\ddot{x}_d(t)$ (dotted line) versus time t. (f) $u^5(t)$ versus time t.

of initial errors are chosen to be large to demonstrate the robustness of the learning system. All the other design parameters of this DAILC are fixed. For example, we let $k = 10, c_1 = 4, c_2 = 4$, and $\delta_m = \delta_\sigma = \delta_\gamma = \delta_w = \delta_\theta = 10$. Five different initial state values are given at each iteration as $X^1(0) = [0,0,0]^\top, X^2(0) = [-2,-1,-1]^\top, X^3(0) = [1,-1,0]^\top, X^4(0) = [2,-0.5,2]^\top$ and $X^5(0) = [2,-1,1]^\top$. Hence, the initial value of $\phi(t)$ is selected as $\phi(0) = \varepsilon^* = 20 \geq \max_j(|c_1 e_1^j(0)| + |c_2 e_2^j(0)| + |e_3^j(0)|)$. As in example 1, we first show the supremum value of $|s_\phi^j(t)|$ with respective to iteration j in Fig. 2(a) and observe the asymptotic convergence given in (t1) of the main theorem. The learning process is almost completed after the fourth iteration since the value is less than 10^{-16}. We demonstrate $s^5(t)$ in Fig. 2(b) to prove that the trajectory of $s^5(t)$ satisfies $-20e^{-10t} \leq s^5(t) \leq 20e^{-10t}$. Fig. 2(c)–(e) are the comparisons between system states $x_1^j(t), x_2^j(t), x_3^j(t)$ and desired states $x_d(t), \dot{x}_d(t), \ddot{x}_d(t)$. It is emphasized that the proposed ORFNN-based DAILC can achieve a successfully iterative learning control objective even the variable and large initial state errors exist. The bounded control input at fifth iteration is finally shown in Fig. 2(f).

V. Conclusion

For adaptive fuzzy control [16] of nonlinear systems, it is usually classified into two categories: indirect adaptive control and direct adaptive control. The fuzzy system is used to describe the plant knowledge for indirect scheme or control knowledge for direct scheme, respectively. This kind of control strategy was recently applied to an iterative learning control problem where the control domain contains both typical time domain and learning iteration domain. Such a control problem is quite different from the traditional control task along time axis. For example, the initial state errors become a critical robustness issue for the ILC design. Up to now, the fuzzy system or neural-network-based AILC are all indirect schemes [21], [22]. In this paper, a direct scheme is first adopted. In addition to a simpler control structure, there are three major contributions in this paper compared with the related works [21], [22]. First, the robustness issue of variable initial state errors is solved by using a technique of time-varying boundary layer. Second, the nonlinear input gain $b(X^j(t))$ can be allowed to be totally unknown before controller design. Third, a new ORFNN is proposed to compensate for the unknown certainty equivalent controller. The ORFNN approximation error is also derived such that we can design an adaptation law to update all the ORFNN parameters. In summary, this paper not only extends the study and application of fuzzy adaptive control to iterative learning control problems, but also provides some new design techniques. Rigorous analysis of the learning system is given and all the technical results are demonstrated by computer simulations for an inverted pendulum system and Chua's chaotic circuit. For practical applications, this approach can be applied to repeatable tasks of control systems such as typical rotational servo motors and linear synchronous motors [29].

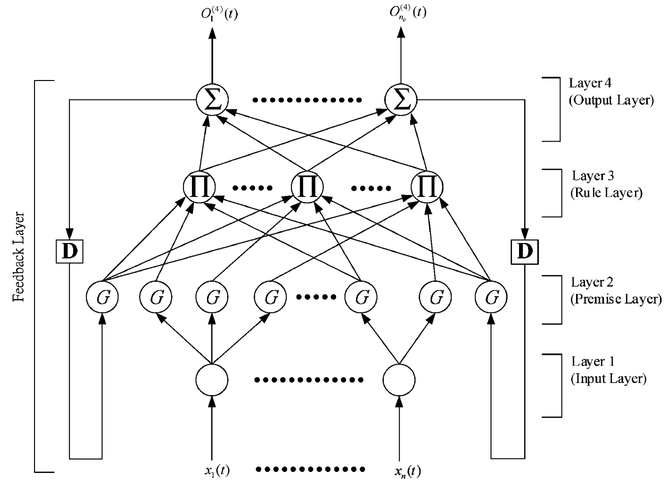

Fig. 3. Configuration of ORFNN.

Appendix A
Structure of the ORFNN

The configuration of the proposed ORFNN is shown in Fig. 3, which consists of n input linguistic variables, N input term nodes, M rule nodes, n_o recurrent nodes and n_o output nodes. Herein, we assume that N_i is the number of fuzzy sets of the ith input variable x_i. Therefore, $N = \sum_{i=1}^n N_i, M = \prod_{i=1}^n N_i$ and there are $n+N+M+2n_o$ nodes in this proposed ORFNN. It is noted that dependency on the iteration index "j" and time index "t" will be omitted unless emphasis on the temporal or iterative relationships is required. The layered operation of the ORFNN is briefly described below.

Layer 1 (Input Layer): Each node in this layer represents an input linguistic variable, which only transmits input value to the next layer directly. For the ith input node, $i = 1,\ldots,n$

$$\mathrm{net}_i^{(1)} = x_i^{(1)}; \quad O_i^{(1)} = f_i^{(1)}\left(\mathrm{net}_i^{(1)}\right) = \mathrm{net}_i^{(1)}$$

where $x_i^{(1)}$ represents the input signal to the ith node of layer 1.

Layer 2 (Premise Layer): Each node in this layer represents a membership function. In this layer, there are two kinds of nodes whose inputs come from input layer and output layer, respectively. For the first kind of nodes, the Gaussian membership function is adopted as a membership function such that the k_ith term node of the ith input linguistic variable $i = 1, \ldots, n, k_i = 1, \ldots, N_i$ will be represented as

$$\begin{aligned}\text{net}_{(i,k_i)}^{(2)} &= -\left(\frac{x_i^{(2)} - m_{(i,k_i)}}{\sigma_{(i,k_i)}}\right)^2\\ O_{(i,k_i)}^{(2)} &= f_{(i,k_i)}^{(2)}\left(\text{net}_{(i,k_i)}^{(2)}\right)\\ &= \exp\left(\text{net}_{(i,k_i)}^{(2)}\right)\end{aligned}$$

where $x_i^{(2)} = O_i^{(1)}, m_{(i,k_i)}$ and $\sigma_{(i,k_i)}$ are the mean and variance, respectively. For the second kind of nodes, another Gaussian membership function is used for the oth recurrent node as follows:

$$\begin{aligned}\text{net}_o^{(2)} &= -\left(\gamma_o \cdot \mathbf{D}\left[O_o^{(4)}\right]\right)^2\\ O_o^{(2)} &= f_o^{(2)}\left(\text{net}_o^{(2)}\right) = \exp\left(\text{net}_o^{(2)}\right)\end{aligned}$$

where γ_o is the recurrent weight and $\mathbf{D}[O_o^{(4)}]$ denotes the delay of $O_o^{(4)}, o = 1, \ldots, n_o$. We will discuss $\mathbf{D}[O_o^{(4)}]$ later.

Layer 3 (Rule Layer): Each node in this layer represents the firing strength of a fuzzy rule. The dynamical fuzzy reasoning is performed by product of the input signals of the $(k_1, \ldots, k_n)$th rule node, which is represented as

$$\begin{aligned}\text{net}_{(k_1,\ldots,k_n)}^{(3)} &= \prod_{i=1}^{n+n_o} x_i^{(3)};\\ O_{(k_1,\ldots,k_n)}^{(3)} &= f_{(k_1,\ldots,k_n)}^{(3)}\left(\text{net}_{(k_1,\ldots,k_n)}^{(3)}\right) = \text{net}_{(k_1,\ldots,k_n)}^{(3)}\end{aligned}$$

where $x_i^{(3)}$ is the ith input to the $(k_1, \ldots, k_n)$th rule node of layer 3.

Layer 4 (Output Layer): Each node in this layer represents an output node, which is the consequence with respect to the oth output variable from each rule. For the oth output node, $o = 1, \ldots, n_o$

$$\begin{aligned}\text{net}_o^{(4)} &= \sum_{k_1=1}^{N_1} \cdots \sum_{k_n=1}^{N_n} w_{o,(k_1,\ldots,k_n)} \cdot x_{(k_1,\ldots,k_n)}^{(4)}\\ O_o^{(4)} &= f_o^{(4)}\left(\text{net}_o^{(4)}\right) = \text{net}_o^{(4)}\end{aligned}$$

where $x_{(k_1,\ldots,k_n)}^{(4)} = O_{(k_1,\ldots,k_n)}^{(3)}$ and the link weight $w_{o,(k_1,\ldots,k_n)}$ is the action strength of the oth output associated with the $(k_1, \ldots, k_n)$th rule.

Hence, this ORFNN performs a dynamic fuzzy reasoning in the traditional fuzzy rule form of

$$\begin{aligned}&\text{IF } x_1 \text{ is } A_{(1,k_1)} \text{ and } \cdots \text{ and } x_n \text{is } A_{(n,k_n)}\\ &\text{THEN } O_1^{(4)} \text{ is } B_{(1,(k_1,\ldots,k_n))} \text{ and } \cdots\\ &\quad \text{and } O_{n_o}^{(4)} \text{ is } B_{(n_o,(k_1,\ldots,k_n))}\end{aligned}$$

For the special feature of iterative learning control, we design the delay mechanism as $\mathbf{D}[O_o^{(4)}] = {O^{(4)}}_o^{j-1}(t)$ where t and j denote time index and iteration index, respectively. Thus, the overall representation from input $x_i^{(1)}$ in layer 1 to the oth output $O_o^{(4)}$ in layer 4 is then given by

$$\begin{aligned}{O_o^{(4)}}^j(t) &= \sum_{k_1=1}^{N_1} \cdots \sum_{k_n=1}^{N_n} w_{o,(k_1,\ldots,k_n)}^j\\ &\quad \times \prod_{o=1}^{n_o} \exp\left\{-\left(\gamma_o^j \cdot {O^{(4)}}_o^{j-1}(t)\right)^2\right\}\\ &\quad \times \prod_{i=1}^{n} \exp\left\{-\left(\frac{x_i^j(t) - m_{(i,k_i)}^j}{\sigma_{(i,k_i)}^j}\right)^2\right\}. \end{aligned} \tag{30}$$

By using the similar techniques in [33], it is easily shown that each output of the ORFNN will depend not only on past network inputs, but also on past network outputs. Therefore, the ORFNN is a dynamic mapping function, which has the global feedback structure. Besides, this ORFNN has a smaller number of network parameters than the RFNN in [27] or TRFN in [30]. For example, if $n = 4, M = 16$, and $n_o = 4$, then the network parameters for FNN in [25], RFNN in [27] and TRFN in [30] are 192, 256, and 448, respectively. In addition, if we assume $N_i = 2, i = 1, 2, 3, 4$ for our proposed ORFNN such that same condition $M = 16$ can be satisfied, then we have only 84 parameters in this ORFNN.

APPENDIX B
THE ORFNN AS A UNIVERSAL APPROXIMATOR

From the representation (30), a multi-input single-output ORFNN can be further described in a matrix form as follows:

$$\begin{aligned}O^{(4)}(X) &= O^{(4)}\left(\mathbf{D}\left[O^{(4)}\right], X, W, m, \sigma, \gamma\right)\\ &= W^\top \cdot O^{(3)}\left(\mathbf{D}\left[O^{(4)}\right], X, m, \sigma, \gamma\right)\end{aligned} \tag{31}$$

with

$$\begin{aligned}W &= \left[w_{(1,\ldots,1)}, \ldots, w_{(N_1,\ldots,N_n)}\right]^\top \in I\!R^{M\times 1}\\ m &= \left[m_{(1,1)}, \ldots, m_{(n,N_n)}\right]^\top \in I\!R^{N\times 1}\\ \sigma &= \left[\sigma_{(1,1)}, \ldots, \sigma_{(n,N_n)}\right]^\top \in I\!R^{N\times 1}\\ \gamma &\in I\!R^{1\times 1}\\ O^{(3)} &= \left[O_1^{(3)}\left(\mathbf{D}\left[O^{(4)}\right], X, m, \sigma, \gamma\right), \ldots,\right.\\ &\qquad \left. O_M^{(3)}\left(\mathbf{D}\left[O^{(4)}\right], X, m, \sigma, \gamma\right)\right]^\top \in I\!R^{M\times 1}.\end{aligned}$$

In fact, the ORFNN (30) or (31) can be easily shown that it is an universal approximator. That is, for any real continuous nonlinear function $f(X)$ on a compact set $\mathcal{A} \subset I\!R^{n\times 1}$, and arbitrary $\epsilon > 0$, there exists optimal constants W^*, m^*, σ^* and γ^* such that the functional approximation error $\epsilon_{\text{opt}}(X)$ between the optimal ORFNN $O^{(4)}(\mathbf{D}[O^{(4)}], X, W^*, m^*, \sigma^*, \gamma^*)$ and $f(X)$ will satisfy $|O^{(4)}(\mathbf{D}[O^{(4)}], X, W^*, m^*, \sigma^*, \gamma^*) - f(X)| = |\epsilon_{\text{opt}}(X)| < \epsilon$, $\forall X \in \mathcal{A}$. The detailed proof of universal approximation theorem for this ORFNN can be easily derived by using the similar techniques in [27]. However, the weights of ORFNN are often tuned via suitable adaptation laws in the adaptive schemes since it is not easy to get the optimal ones for the controller design. In the following we will show how the approximation error $\epsilon_f(X) = O^{(4)}(\mathbf{D}[O^{(4)}], X, W, m, \sigma, \gamma) - f(X)$ can be expressed in a linearly parameterized form modulo a residual term. This enables us to tune all the network parameters via a uitable adaptive law. First, define the estimation errors of consequent, premise and recurrent parameter vectors as $\tilde{W} \equiv W - W^*, \tilde{m} \equiv m - m^*, \tilde{\sigma} \equiv \sigma - \sigma^*, \tilde{\gamma} \equiv \gamma - \gamma^*$. For simplicity, define $O^{(3)} = O^{(3)}(\mathbf{D}[O^{(4)}], X, m, \sigma, \gamma)$, $O^{(3)^*} = O^{(3)}(\mathbf{D}[O^{(4)}], X, m^*, \sigma^*, \gamma^*)$ and $\tilde{O}^{(3)} = O^{(3)} - O^{(3)^*}$. Then the functional approximation error $\epsilon_f(X)$ will satisfy

$$\begin{aligned}\epsilon_f(X) &= O^{(4)}\left(\mathbf{D}\left[O^{(4)}\right], X, W, m, \sigma, \gamma\right) - f(X)\\ &= W^\top \cdot O^{(3)} - W^{*\top} \cdot O^{(3)^*} + \epsilon_{\text{opt}}(X)\\ &= W^\top \cdot O^{(3)} - W^{*\top} \cdot O^{(3)} + W^{*\top} \cdot O^{(3)}\\ &\quad - W^{*\top} \cdot O^{(3)^*} + \epsilon_{\text{opt}}(X)\\ &= \tilde{W}^\top \cdot O^{(3)} + W^{*\top} \cdot \tilde{O}^{(3)} + \epsilon_{\text{opt}}(X). \end{aligned} \tag{32}$$

In order to deal with $\tilde{O}^{(3)}$, we use the Taylor series expansion of $O^{(3)^*}$ at $m^* = m, \sigma^* = \sigma, \gamma^* = \gamma$ as follows:

$$\begin{aligned}&O^{(3)}\left(\mathbf{D}\left[O^{(4)}\right], X, m^*, \sigma^*, \gamma^*\right)\\ &= O^{(3)}\left(\mathbf{D}\left[O^{(4)}\right], X, m, \sigma, \gamma\right)\\ &\quad + O_m^{(3)'}\left(\mathbf{D}\left[O^{(4)}\right], X, m, \sigma, \gamma\right) \cdot (m^* - m)\\ &\quad + O_\sigma^{(3)'}\left(\mathbf{D}\left[O^{(4)}\right], X, m, \sigma, \gamma\right) \cdot (\sigma^* - \sigma)\\ &\quad + O_\gamma^{(3)'}\left(\mathbf{D}\left[O^{(4)}\right], X, m, \sigma, \gamma\right) \cdot (\gamma^* - \gamma)\\ &\quad + o(X, \tilde{m}, \tilde{\sigma}, \tilde{\gamma}) \end{aligned} \tag{33}$$

where $o(\cdot)$ denotes a sum of the high-order terms of the argument in a Taylor series expansion and $O_m^{(3)'} \in \mathbb{R}^{M \times N}, O_\sigma^{(3)'} \in \mathbb{R}^{M \times N}$ and $O_\gamma^{(3)'} \in \mathbb{R}^{M \times 1}$ are derivatives of $O^{(3)}(\mathbf{D}[O^{(4)}], X, m^*, \sigma^*, \gamma^*)$ with respective to m^*, σ^* and γ^* at $(m^*, \sigma^*, \gamma^*) = (m, \sigma, \gamma)$. In other words

$$O_m^{(3)'} = \left.\frac{\partial O^{(3)}\left(\mathbf{D}\left[O^{(4)}\right], X, m^*, \sigma^*, \gamma^*\right)}{\partial m^*}\right|_{m^*=m, \sigma^*=\sigma, \gamma^*=\gamma}$$

$$O_\sigma^{(3)'} = \left.\frac{\partial O^{(3)}\left(\mathbf{D}\left[O^{(4)}\right], X, m^*, \sigma^*, \gamma^*\right)}{\partial \sigma^*}\right|_{m^*=m, \sigma^*=\sigma, \gamma^*=\gamma}$$

$$O_\gamma^{(3)'} = \left.\frac{\partial O^{(3)}\left(\mathbf{D}\left[O^{(4)}\right], X, m^*, \sigma^*, \gamma^*\right)}{\partial \gamma^*}\right|_{m^*=m, \sigma^*=\sigma, \gamma^*=\gamma}.$$

This implies that

$$\begin{aligned}\tilde{O}^{(3)} = O_m^{(3)'} \cdot (m - m^*) + O_\sigma^{(3)'} \cdot (\sigma - \sigma^*)\\ + O_\gamma^{(3)'}(\gamma - \gamma^*) - o(X, \tilde{m}, \tilde{\sigma}, \tilde{\gamma}). \end{aligned} \tag{34}$$

Substituting (34) into (32), we have

$$\begin{aligned}&\epsilon_f(X)\\ &= \tilde{W}^\top \cdot O^{(3)} + W^{*\top} \cdot \Big(O_m^{(3)'} \cdot (m - m^*)\\ &\quad + O_\sigma^{(3)'} \cdot (\sigma - \sigma^*) + O_\gamma^{(3)'} \cdot (\gamma - \gamma^*)\\ &\quad - o(X, \tilde{m}, \tilde{\sigma}, \tilde{\gamma})\Big) + \epsilon_{\text{opt}}(X)\\ &= \tilde{W}^\top \cdot O^{(3)} + (W - \tilde{W})^\top \cdot \Big(O_m^{(3)'} \cdot (m - m^*)\\ &\quad + O_\sigma^{(3)'} \cdot (\sigma - \sigma^*) + O_\gamma^{(3)'} \cdot (\gamma - \gamma^*)\Big)\\ &\quad - W^{*\top} \cdot o(X, \tilde{m}, \tilde{\sigma}, \tilde{\gamma}) + \epsilon_{\text{opt}}(X)\\ &= \tilde{W}^\top \cdot \Big(O^{(3)} - O_m^{(3)'} \cdot m - O_\sigma^{(3)'} \cdot \sigma - O_\gamma^{(3)'} \cdot \gamma\Big)\\ &\quad + W^\top \cdot \Big(O_m^{(3)'} \cdot \tilde{m} + O_\sigma^{(3)'} \cdot \tilde{\sigma} + O_\gamma^{(3)'} \cdot \tilde{\gamma}\Big)\\ &\quad + \tilde{W}^\top \cdot \Big(O_m^{(3)'} \cdot m^* + O_\sigma^{(3)'} \cdot \sigma^* + O_\gamma^{(3)'} \cdot \gamma^*\Big)\\ &\quad - W^{*\top} \cdot o(X, \tilde{m}, \tilde{\sigma}, \tilde{\gamma}) + \epsilon_{\text{opt}}(X)\\ &\equiv \tilde{W}^\top \cdot \Big(O^{(3)} - O_m^{(3)'} \cdot m - O_\sigma^{(3)'} \cdot \sigma - O_\gamma^{(3)'} \cdot \gamma\Big)\\ &\quad + W^\top \cdot \Big(O_m^{(3)'} \cdot \tilde{m} + O_\sigma^{(3)'} \cdot \tilde{\sigma} + O_\gamma^{(3)'} \cdot \tilde{\gamma}\Big) + \eta. \end{aligned} \tag{35}$$

Since $\tilde{O}^{(3)} = O^{(3)} - O^{(3)^*}$ and $O_m^{(3)'}, O_\sigma^{(3)'}, O_\gamma^{(3)'}$ are bounded due to the bounded properties of Gaussian membership function and its derivative, it is easy to show that $o(X, \tilde{m}, \tilde{\sigma}, \tilde{\gamma})$ in (34) will satisfy

$$\begin{aligned}&\|o(X, \tilde{m}, \tilde{\sigma}, \tilde{\gamma})\|\\ &\quad = \left\|O_m^{(3)'} \cdot \tilde{m} + O_\sigma^{(3)'} \cdot \tilde{\sigma} + O_\gamma^{(3)'} \cdot \tilde{\gamma} - \tilde{O}^{(3)}\right\|\\ &\quad \le k_1\|\tilde{m}\| + k_2\|\tilde{\sigma}\| + k_3|\tilde{\gamma}| + k_4 \end{aligned}$$

where k_1, k_2, k_3, k_4 are some bounded constants. On the other hand, by the fact of $\|\tilde{W}\| \le \|W^*\| + \|W\|$, $\|\tilde{m}\| \le \|m^*\| + \|m\|$, $\|\tilde{\sigma}\| \le \|\sigma^*\| + \|\sigma\|$ and $|\tilde{\gamma}| \le |\gamma^*| + |\gamma|$, we can find that

$$\begin{aligned}|\eta| &= \Big|\tilde{W}^\top \cdot \Big(O_m^{(3)'} \cdot m^* + O_\sigma^{(3)'} \cdot \sigma^* + O_\gamma^{(3)'} \cdot \gamma^*\Big)\\ &\qquad - W^{*\top} \cdot o(X, \tilde{m}, \tilde{\sigma}, \tilde{\gamma}) + \epsilon_{\text{opt}}(X)\Big|\\ &\le (\|W\| + \|W^*\|)\Big(\|O_m^{(3)'}\|\|m^*\| + \|O_\sigma^{(3)'}\|\|\sigma^*\|\\ &\qquad + \|O_\gamma^{(3)'}\||\gamma^*|\Big) + \|W^*\|(k_1(\|m^*\| + \|m\|)\\ &\qquad + k_2(\|\sigma^*\| + \|\sigma\|) + k_3(|\gamma^*| + |\gamma|) + k_4) + \epsilon\\ &= \Big(\|W^*\|\Big(\|O_m^{(3)'}\|\|m^*\| + \|O_\sigma^{(3)'}\|\|\sigma^*\| + \|O_\gamma^{(3)'}\||\gamma^*|\\ &\qquad + k_1\|m^*\| + k_2\|\sigma^*\| + k_3|\gamma^*| + k_4\Big) + \epsilon\Big)\\ &\qquad + \Big(\|O_m^{(3)'}\|\|m^*\| + \|O_\sigma^{(3)'}\|\|\sigma^*\| + \|O_\gamma^{(3)'}\||\gamma^*|\Big)\|W\|\\ &\qquad + (k_1\|W^*\|)\|m\| + (k_2\|W^*\|)\|\sigma\| + (k_3\|W^*\|)|\gamma|\\ &= [\theta_1^*, \theta_2^*, \theta_3^*, \theta_4^*, \theta_5^*]\begin{bmatrix}1\\ \|W\|\\ \|m\|\\ \|\sigma\|\\ |\gamma|\end{bmatrix}\\ &\equiv \theta^{*\top} Y \end{aligned} \tag{36}$$

REFERENCES

[1] S. Arimoto, S. Kawamura, and F. Miyazaki, "Bettering operation of robots by learning," *J. Robot. Syst.*, vol. 1, no. 2, pp. 123–140, 1984.

[2] G. Heinzinger, D. Fenwick, B. Paden, and F. Miyazaki, "Stability of learning control with disturbances and uncertain initial conditions," *IEEE Trans. Automat. Contr.*, vol. 37, pp. 110–114, Jan. 1992.

[3] C. J. Chien and J. S. Liu, "A P-type iterative learning controller for robust output tracking of nonlinear time-varying systems," *Int. J. Contr.*, vol. 64, no. 2, pp. 319–334, 1996.

[4] H. S. Lee and Z. Bien, "Study on robustness of iterative control with nonzero initial error," *Int. J. Contr.*, vol. 64, no. 3, pp. 345–359, 1996.

[5] K. H. Park, Z. Bien, and D. H. Hwang, "A study on robustness of PID-type iterative learning controller aganist initial state error," *Int. J. Syst. Sci.*, vol. 30, no. 1, pp. 49–59, 1999.

[6] Y. Chen, C. Wen, Z. Gong, and M. Sun, "An iterative learning controller with initial state learning," *IEEE Trans. Automat. Contr.*, vol. 44, pp. 371–376, Feb. 1999.

[7] M. Sun and D. Wang, "Iterative learning control with initial rectifying action," *Automatica*, vol. 38, pp. 1177–1182, 2002.

[8] B. H. Park, T. Y. Kuc, and J. S. Lee, "Adaptive learning of uncertain robotic systems," *Int. J. Contr.*, vol. 65, pp. 725–744, May 1996.

[9] J. Y. Choi and J. S. Lee, "Adaptive iterative learning control of uncertain robotic systems," in *Proc. Inst. Elect. Eng. D, Control Theory Application*, vol. 147, 2000, pp. 217–223.

[10] T. Y. Kuc and W. G. Han, "An adaptive PID learning control of robot manipulators," *Automatica*, vol. 36, pp. 717–725, 2000.

[11] J. X. Xu and B. Wiswanathan, "Adaptive robust iterative learning control with dead zone scheme," *Automatica*, vol. 36, pp. 91–99, 2000.

[12] S. S. Sastry and A. Isidori, "Adaptive control of lineariable systems," *IEEE Trans. Automat. Contr.*, vol. 34, pp. 1123–1131, Oct. 1989.

[13] C. C. Lee, "Fuzzy logic in control systems: Fuzzy logic controller-part I and II," *IEEE Trans. Syst., Man, Cybern.*, vol. 20, pp. 404–435, Apr. 1990.

[14] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical system using neural networks," *IEEE Trans. Neural Networks*, vol. 1, pp. 4–27, Mar. 1990.

[15] C. Y. Su and Y. Stepanenko, "Adaptive control of a class of nonlinear systems with fuzzy logic," *IEEE Trans. Fuzzy Syst.*, vol. 2, pp. 285–294, Nov. 1994.

[16] L. X. Wang, *A Course in Fuzzy Systems and Control*. Englewood Cliffs, NJ: Prentice-Hall, 1997.

[17] H. Han, C. Y. Su, and Y. Stepanenko, "Adaptive control of a class of nonlinear systems with nonlinearly parameterized fuzzy approximators," *IEEE Trans. Fuzzy Syst.*, vol. 9, pp. 315–323, Apr. 2001.

[18] R. M. Sanner and J. J. E. Slotine, "Gaussin networks for direct adaptive control," *IEEE Trans. Neural Networks*, vol. 3, pp. 837–863, Nov. 1992.

[19] F. C. Chen and H. K. Khalil, "Adaptive control of a class of nonlinear discrete-time systems using neural networks," *IEEE Trans. Automat. Contr.*, vol. 40, pp. 791–801, May 1995.

[20] L. A. Lewis, A. Yesildirek, and K. Liu, "Neural net robot controller with guaranteed tracking performance," *IEEE Trans. Neural Networks*, vol. 6, pp. 703–715, May 1995.

[21] W. G. Seo, B. H. Park, and J. S. Lee, "Adaptive fuzzy learning control for a class of nonlinear dynamic systems," *Int. J. Intell. Syst.*, vol. 15, pp. 1157–1175, 2000.

[22] C. J. Chien and L. C. Fu, "An iterative learning control of nonlinear systems using neural network design," *Asian J. Contr.*, vol. 4, no. 1, pp. 21–29, 2002.

[23] C. T. Lin and C. S. G. Lee, "Neural-network-based fuzzy logic control and decision system," *IEEE Trans. Comput.*, vol. 40, pp. 1320–1336, Dec. 1991.

[24] J. S. Jang, "ANFIS: Adaptive-network-based fuzzy inference system," *IEEE Trans. Syst., Man, Cybern. B*, vol. 23, pp. 665–684, June 1993.

[25] Y. C. Chen and C. C. Teng, "A model reference control structure using a fuzzy neural network," *Fuzzy Sets Syst.*, vol. 73, pp. 291–312, 1995.

[26] J. T. Sponner and K. M. Passino, "Stable adaptive control using fuzzy systems and neural networks," *IEEE Trans. Fuzzy Syst.*, vol. 4, pp. 339–359, Aug. 1996.

[27] C. H. Lee and C. C. Teng, "Identification and control of dynamic systems using recurrent fuzzy neural network," *IEEE Trans. Fuzzy Syst.*, vol. 8, pp. 349–366, Aug. 2000.

[28] Y. C. Wang, C. J. Chien, and C. C. Teng, "Takagi-Sugeno recurrent fuzzy neural networks for identification and control of dynamic systems," in *Proc. IEEE Int. Conf. Fuzzy Systems*, Melbourne, Australia, 2001, pp. 537–540.

[29] F. J. Lin, R. J. Wai, and C. M. Hong, "Hybrid control using recurrent fuzzy neural network for tracking periodic inputs," *IEEE Trans. Neural Networks*, vol. 12, pp. 68–90, Jan. 2001.

[30] C. F. Juang, "A TSK-type recurrent network for dynamic systems processing by neural network and genetic algorithms," *IEEE Trans. Fuzzy Syst.*, vol. 10, pp. 155–170, Apr. 2002.

[31] C. H. Wang, H. L. Liu, and T. C. Lin, "Direct adaptive fuzzy-neural control with state observer and supervisory controller for unknown nonlinear dynamical systems," *IEEE Trans. Fuzzy Syst.*, vol. 10, pp. 39–49, Feb. 2002.

[32] C. H. Wang, T. C. Lin, T. T. Lee, and H. L. Liu, "Adaptive hybrid intelligent control for uncertain nonlinear dynamical systems," *IEEE Trans. Syst., Man, Cybern. B*, vol. 32, pp. 583–597, Oct. 2002.

[33] C. C. Ku and K. Y. Lee, "Diagonal recurrent neural networks for dynamic systems control," *IEEE Trans. Neural Networks*, vol. 6, pp. 144–156, Jan. 1995.

Ying-Chung Wang was born in Taipei, Taiwan, R.O.C., in 1972. He received the B.S. degree in electronic engineering and the M.S. degree in mechatronic engineering from Huafan University, Taipei, in 1996 and 1998, respectively, and the Ph.D. degree in electrical and control engineering from National Chiao Tung University, Hsinchu, Taiwan, in 2003.

He is currently a Postdoctoral Fellow at the Institute of Information Science, Academia Sinica, Taiwan. His main research interests are fuzzy systems, neural networks, fuzzy neural systems, iterative learning control, nonlinear control, adaptive control, and robot control.

Chiang-Ju Chien was born in Taiwan, R.O.C., in 1963. He received the B.S. degree in nuclear engineering from National Tsing Hua University, Hsinchu, Taiwan, in 1985 and Ph.D. degree in electrical engineering from National Taiwan University, Taipei, in 1992.

Since 1993, he has been with the Department of Electronic Engineering, Huafan University, Taipei, where he is currently an Associate Professor and the Chairman of the Department Electronic Engineering. His present research interests include iterative learning control, adaptive control, fuzzy-neuro systems, and control circuit design.

Dr. Chien is a member of the Chinese Automatic Control Society.

Ching-Cheng Teng was born in Taiwan, R.O.C., in 1938. He received the B.S. degree in electrical engineering from National Cheng Kung University, Taiwan, in 1961.

From 1991 to 1998, he was the Chairman of the Department of Electrical and Control Engineering, National Chiao Tung University, Hsinchu, Taiwan. He is currently a Professor of the Department of Electrical and Control Engineering, National Chiao Tung University. His research interests include H^∞ optimal control, signal processing, and fuzzy neural systems.