SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

February 2022, Vol. 65 122206:1–122206:15
https://doi.org/10.1007/s11432-020-3138-7

Neural learning control for discrete-time nonlinear systems in pure-feedback form

Min WANG[1*], Haotian SHI[1], Cong WANG[2] & Jun FU[3]

[1]*School of Automation Science and Engineering, South China University of Technology, Guangzhou* 510641*, China;*
[2]*School of Control Science and Engineering, Shandong University, Jinan* 250061*, China;*
[3]*State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang* 110819*, China*

Received 23 June 2020/Revised 17 September 2020/Accepted 30 October 2020/Published online 24 January 2022

Abstract This study focuses on state-feedback and output-feedback neural learning control problems for discrete-time nonlinear systems in the pure-feedback form. First, an extended result for the exponential stability for a class of discrete-time linear time-varying (LTV) systems with n-step delays is proposed to verify the exponential convergence of estimated weights. Subsequently, both state-feedback and output-feedback adaptive neural network (NN) controllers are constructed by combining the classical n-step and n-step input-output predictors. After ensuring convergence of the system output to a recurrent reference signal, the radial basis function subvector of NN is verified to satisfy the persistent exciting condition using the system state equation and the implicit function theorem. By combining the extended stability corollary of an LTV system, the estimated weights are verified to exponentially converge to their optimal values. By constructing "learning rules" and using a "mod" function, the estimated weights with a convergent sequence are synthetically represented and stored as the experience knowledge, which is reused to construct neural learning controllers. The proposed neural learning controllers not only accomplish similar control tasks but also reduce the burden of online computation compared with the conventional adaptive NN controllers. Finally, simulation results are presented to demonstrate the effectiveness of the presented schemes.

Keywords discrete-time nonlinear systems, pure-feedback systems, learning control, neural networks, persistent excitation

Citation Wang M, Shi H T, Wang C, et al. Neural learning control for discrete-time nonlinear systems in pure-feedback form. Sci China Inf Sci, 2022, 65(2): 122206, https://doi.org/10.1007/s11432-020-3138-7

1 Introduction

Neural networks (NNs) have been widely used for the controller design of uncertain nonlinear systems owing to their universal approximation ability and the ubiquitous unknown dynamics in practice [1]. For continuous-time nonlinear systems in a normal form, a large number of adaptive NN control schemes [2] have been proposed based on the Lyapunov stability theory and system identification. Using backstepping technology [3], these adaptive NN control schemes have been extended to nonlinear systems in the strict-feedback form [4–7]. A common assumption from these results is that the considered system is affine. Such an assumption is usually difficult to be practically satisfied owing to the inevitable nonaffine structure, such as the hypersonic flight vehicles, biochemical processes, and flexible robots. To discard this assumption, the implicit function theorem has been employed to develop adaptive NN control for nonaffine nonlinear systems in the normal form [3]. This result has been extended to special pure-feedback nonlinear systems (PFNSs) by combining backstepping and the mean-value theorem [8, 9], wherein the control input was required to be linear. The nonaffine form of the control input may result in the circular control design because the control input and its derivatives are involved in the input variables of NN. Such a difficulty has been tackled in [10] by the combination of the input-state-stability and the small-gain theorem, solving the control problem of more general PFNSs. Using the proposed framework, some interesting adaptive neural control schemes have been developed for PFNSs with different phenomena, including input nonlinearities [11–13], output/state constraints [14–16], and unmeasurable states [17,18].

* Corresponding author (email: auwangmin@scut.edu.cn)

© Science China Press and Springer-Verlag GmbH Germany, part of Springer Nature 2022

Compared with considerable studies on continuous-time systems, relatively few results for discrete-time nonlinear systems exist although digital computer technology has been widely used in practical engineering. The primary challenges include the following: (1) the difference of the Lyapunov function is nonlinear in discrete-time, making stability analysis become severer than the continuous-time case; (2) a noncausal problem may be encountered for the controller design of discrete-time lower-triangular systems, i.e., the future system states will be involved in the current control input. Nevertheless, some dedicated efforts have been put in for discrete-time nonlinear systems [19–28]. For discrete-time parametric strict-feedback systems, the robust backstepping adaptive control scheme has been developed to solve the system output tracking problem [19]. To overcome the noncausal problem of more general discrete-time nonlinear systems, an n-step predictor has been developed in [20] for discrete-time strict-feedback systems. Motivated by the concept of the n-step-ahead predictor, the n-step input-output predictor has been developed in [21] to achieve the tracking control for discrete-time PFNSs by combining the implicit function theorem. Subsequently, a novel adaptive NN control framework based on variable substitution has been proposed in [22, 29], which can also avoid the noncausal problem without n-step transformation of the original systems. With the aid of the three frameworks mentioned above, there are increasingly interesting results available on the control problem of discrete-time lower-triangular nonlinear systems subject to various conditions such as input nonlinearities [23, 24], limited communication bandwidth [25, 26], and performance optimization [27, 28].

The neural learning problem (i.e., knowledge acquirement, storage, and usage of the estimated weights) is not a major concern in the aforementioned adaptive NN control schemes. However, learning ability is a key factor for achieving high-level industrial automation. Moreover, the learning issue should be a primary concern of NN control, as the development of NN control is motivated by human learning. As we all know, the neural learning problem is a great challenge because the persistent exciting (PE) condition of NN weight convergence is too harsh to be verified in closed-loop control systems [30]. To handle such a challenge, a deterministic learning mechanism has been proposed in [31] for continuous-time nonlinear systems in the normal form, which verifies the convergence of partial estimated weights. It has been shown in [31] that for a radial basis function (RBF) NN constructed on regular lattices, the RBF NN satisfies a partial PE condition if NN inputs are recurrent. Motivated by the deterministic learning mechanism [31], a neural learning control scheme has been proposed for continuous-time strict-feedback systems [32], wherein the recurrent property of the NN inputs and the exponential convergence of estimated weights are verified by the system decomposition strategy and coordinate transformation technology. Next, a few interesting neural learning control schemes have been developed for continuous-time nonlinear systems with different structures, including the affine [33], nonaffine [34], output constrained [35], and the multiagent [36] forms. Furthermore, learning control schemes have been extended to many practical systems such as robot manipulators [37] and ocean surface ships [38, 39].

In contrast to the fruitful neural learning results for continuous-time systems, only a few results are available for discrete-time nonlinear systems [40–42] owing to the lack of exponential stability of linear time-varying (LTV) systems with time delays. Using the existing learning framework, the early studies [40–42] are suitable only for discrete-time nonlinear systems subject to the normal form and the unit control gain. To the best of our knowledge, the neural learning control for discrete-time nonaffine PFNSs is still an open problem although the PFNSs represent more practical systems than the normal form [40–42]. The primary difficulties stem from the following three parts: (1) The constructed neural weight adaptive laws usually contain n-step delays for discrete-time PFNSs. The delays lead to the weight estimate error system as a class of LTV systems with n-step delays, which makes it impossible to verify the convergence of estimated weights owing to the lack of relevant research results for exponential stability of discrete-time LTV time-delay systems. (2) For discrete-time PFNSs, all system state variables are used as NN inputs, which makes it difficult to verify the recurrent property using the existing system decomposition strategy. (3) Owing to the existence of time delays, NN estimated weights may converge to a constant sequence rather than a fixed value. The manner to express and store the knowledge of estimated weights is of vital significance for knowledge reuse to achieve a high-level automatous control.

Based on the above observation, it is by no means a trivial task to tackle such an open problem. Herein, these challenges are handled one at a time. First, both state-feedback and output-feedback adaptive NN control schemes are respectively proposed for discrete-time PFNSs with measurable and immeasurable system states to ensure that the system output converges to a small neighborhood of a given recurrent signal using the n-step predictor. Next, all the NN input variables are verified to be recurrent, thereby ensuring satisfaction with respect to the partial PE condition of RBF NN. According to the character-

istics of n-step delays of the weight updating law, the exponential convergence of the estimated weights is guaranteed by extending the existing exponential convergence of discrete-time LTV systems. The converged weights are represented and stored in a constant sequence through some specially constructed "learning rules". With the help of a "mod" function, the stored knowledge is reused to construct both state-feedback and output-feedback neural learning controllers to achieve improved tracking control performance, including better transient performance and less online computation. For clarity, the primary contributions of this study are summarized as follows:

(1) An extended result is developed to show the exponential stability condition for a class of discrete-time LTV systems with n-step delays. Such a result provides a powerful tool for convergence of the estimated weights under n-step delays.

(2) All the NN inputs are proven to be recurrent by combining the system state equation and implicit function theorem, guaranteeing that the PE condition of RBF NNs is satisfied.

(3) The estimated weights are systematically represented and stored as a constant sequence by the constructed "learning rules". The "mod" function is introduced to characterize the experience knowledge of the ideal control input based on the stored constant sequence. The stored knowledge is reused to construct neural learning controllers, improving the tracking performance to a great extent.

The rest of the study is organized as follows. Section 2 formulates the learning objective and provides some preliminaries. In Section 3, adaptive NN controllers are presented for PFNSs, and the convergence of the estimated weights is verified. The "learning rules" and the NN learning controllers are constructed in Section 4. Simulation results are presented in Section 5. Finally, a conclusion is included in Section 6.

Notation. Standard notations are employed herein. For a square matrix Γ, $\lambda_{\max}(\Gamma)$ denotes its maximum eigenvalue. The subscript $(\cdot)_\zeta$ stands for the region close to the trajectory $\phi(Z(k))$ and the subscript $(\cdot)_{\bar{\zeta}}$ stands for the region far away from the trajectory $\phi(Z(k))$, where $\phi(Z(k))$ represents the trajectory composed by the variables in the vector $Z(k)$.

2 Preliminaries and problem formulation

2.1 System description

Consider the following discrete-time PFNS with unknown nonaffine terms:

$$\begin{cases} x_i(k+1) = f_i(\bar{x}_i(k), x_{i+1}(k)), \quad i = 1, \ldots, n-1, \\ x_n(k+1) = f_n(\bar{x}_n(k), u(k)), \\ y(k) = x_1(k), \end{cases} \tag{1}$$

where $\bar{x}_i(k) = [x_1(k), x_2(k), \ldots, x_i(k)]^{\mathrm{T}}$ $(i = 1, 2, \ldots, n)$, $u(k)$, and $y(k)$ are the state vector, the control input, and the system output, respectively; $f_i(\bar{x}_i(k), x_{i+1}(k))$ and $f_n(\bar{x}_n(k), u(k))$ are unknown smooth nonlinear functions.

For the sake of controller design for the system (1), define the following smooth functions:

$$\begin{cases} g_i(\cdot) = \partial f_i(\bar{x}_i(k), x_{i+1}(k))/\partial x_{i+1}(k), \quad i = 1, \ldots, n-1, \\ g_n(\cdot) = \partial f_n\left(\overline{x}_n(k), u(k)\right)/\partial u(k). \end{cases} \tag{2}$$

Assumption 1 ([27])**.** System nonlinear functions $g_i(\cdot)$ $(i = 1, \ldots, n)$ satisfy $0 < \underline{g}_i < g_i(\cdot) < \overline{g}_i$, where $\underline{g}_i$ and $\overline{g}_i$ are two positive constants.

Assumption 2 ([41])**.** The desired system output $y_d(k)$ is a recurrent signal.

A recurrent trajectory represents a large set of periodic, quasi-periodic, almost-periodic, and even chaotic trajectories, which is characterized as: for a given constant $\xi > 0$, there exists a time interval k_ξ such that the trajectory $\phi(Z(k))$ returns to the ξ-neighborhood of any point on the trajectory within a time not greater than k_ξ [43].

Herein, our objective is to propose a dynamic learning control scheme for the system (1) to achieve the knowledge acquirement and storage of estimated weights during the steady-state control process, and then reuse the learned knowledge to construct the neural learning controller for the improved tracking control performance.

2.2 RBF neural networks

It has been proven in [30] that the RBF NN can approximate any continuous function $h(Z(k)) : \mathbb{R}^q \to \mathbb{R}$ over a compact set $\Omega_Z \subset \mathbb{R}^q$ as

$$h(Z(k)) = W^{*\mathrm{T}} S(Z(k)) + \epsilon(Z(k)), \quad \forall Z(k) \in \Omega_Z, \tag{3}$$

where $W^* \in \mathbb{R}^m$ is the ideal weights vector, m is the number of NN nodes, $Z(k) \subset \mathbb{R}^q$ is the NN input vector, $S(Z(k)) = [s_1(Z(k)), s_2(Z(k)), \ldots, s_m(Z(k))]^{\mathrm{T}} \in \mathbb{R}^m$ is the activation function vector, where $s_i(Z(k))$ is selected as $s_i(Z(k)) = \exp[-(Z(k) - \beta_i)^{\mathrm{T}}(Z(k) - \beta_i)/\eta_i^2]$ with $\beta_i = [\beta_{i1}, \beta_{i2}, \ldots, \beta_{iq}]^{\mathrm{T}}$ and η_i being the center and the width of NN, respectively, and $\epsilon(Z(k))$ stands for the approximation error, satisfying $|\epsilon(Z(k))| < \epsilon^*$, where ϵ^* is an arbitrarily small positive constant.

Furthermore, the RBF NN has the localized approximation ability [31]. In particular, for any recurrent trajectory $\phi(Z(k))$, the nonlinear function $h(Z(k))$ can be approximated by some neurons closing to $\phi(Z(k))$ as

$$h(Z(k)) = W_\zeta^{*\mathrm{T}} S_\zeta(Z(k)) + \epsilon_\zeta(Z(k)), \tag{4}$$

where $S_\zeta(Z(k)) = [s_{1\zeta}(Z(k)), \ldots, s_{j\zeta}(Z(k))]^{\mathrm{T}} \in \mathbb{R}^{m_\zeta}$ with $m_\zeta < m$, $W_\zeta^* \in \mathbb{R}^{m_\zeta}$, and $\epsilon_\zeta(Z(k))$ is the approximation error, with $|\epsilon_\zeta(Z(k))| - |\epsilon(Z(k))|$ being small.

As we know, the PE condition [31,41] is a key factor to achieve the exponential convergence of estimated weights, which is shown as follows.

Definition 1. A sequence $S(Z(k))$ is said to be PE if there exist positive constants α, k_0, and k_1 such that

$$\sum_{k=k_0}^{k_0+k_1-1} S(Z(k)) S^{\mathrm{T}}(Z(k)) \geqslant \alpha I, \quad \forall k_0 > 0. \tag{5}$$

Lemma 1 ([31])**.** Assume that the RBF NN input vector $Z(k)$ is recurrent. Then, it can be obtained that the subvector $S_\zeta(Z(k))$ composed by $S(Z(k))$ closing to the orbits $\phi(Z(k))$ satisfies the PE condition.

2.3 Stability of discrete-time linear time-varying systems

Consider a discrete-time LTV system as

$$x(k+1) = A(k)x(k) + d(k), \tag{6}$$

where $x(k) \in \mathbb{R}^m$ is the system state, $A(k) \in \mathbb{R}^{m \times m}$ is the system matrix, and $d(k)$ is a bounded disturbance. The exponential convergence of system (6) has been given in [41] as follows.

Lemma 2. Consider the discrete-time LTV system (6). Assume $A(k) = I_m - \Gamma\xi(k)\xi^{\mathrm{T}}(k)$, $\Gamma\xi(k)\xi^{\mathrm{T}}(k) < 2I_m$, and $\|d(k)\|$ is small, wherein $I_m \in \mathbb{R}^{m \times m}$ is an identity matrix, $\xi(k) \in \mathbb{R}^m$ is a time-varying matrix, and $\Gamma \in \mathbb{R}^{m \times m}$ is a positive definite symmetric constant matrix. If $\xi(k)$ satisfies the PE condition, then the system state $x(k)$ converges exponentially to a small neighbourhood of zero.

Lemma 2 is the core tool for discrete-time nonlinear systems in the normal form to verify the convergence of estimated weights. However, Lemma 2 is not applicable to the considered pure-feedback form (1). As such, this study proposes an extension of Lemma 2 as follows.

Corollary 1. Consider the discrete-time dynamic system

$$x(k+n) = A(k)x(k) + d(k). \tag{7}$$

Assume $A(k) = I_m - \Gamma\xi(k)\xi^{\mathrm{T}}(k)$, $\Gamma\xi(k)\xi^{\mathrm{T}}(k) < 2I_m$, and the value of $\|d(k)\|$ is small. If $\xi(k)$ is PE, then these states $x(k^1), x(k^2), \ldots, x(k^n)$ converge exponentially to a small neighborhood around zero, respectively, where $\{k^j |\ k^j = j, j+n, j+2n, \ldots\}$ stands for time sequence with $j = 1, 2, \ldots, n$.

Proof. Define $x^j(k) = x(kn+j)$, $A^j(k) = A(kn+j)$, $d^j(k) = d(kn+j)$, where $j = 1, \ldots, n$. Then, we have

$$x^j(k+1) = x((k+1)n + j) = x(kn + j + n). \tag{8}$$

According to (7), it is easy to obtain

$$x^j(k+1) = A(kn+j)x(kn+j) + d(kn+j) = A^j(k)x^j(k) + d^j(k). \tag{9}$$

Using Lemma 2, we can obtain that the state $x^j(k)$ converges exponentially to a small neighborhood around zero. According to the definition of $x^j(k)$, we obtain that these states $x(k^1), x(k^2), \ldots, x(k^n)$ converge exponentially to small neighborhoods around zero.

3 Dynamic learning from adaptive NN control: knowledge acquirement

This section focuses on the convergence of estimated weights during the stable closed-loop control process. To achieve such a goal, the state-feedback and output-feedback control are investigated for the discrete-time PFNSs (1) by the system transformation technology, respectively. Based on a combination of the developed Corollary 1, the estimated weights will be verified to be exponentially convergent for any recurrent reference $y_d(k)$.

3.1 State-feedback control

To avoid the causality contradiction, which may be encountered in the controller design of discrete-time PFNSs (1), the system transformation technology [21] is employed to transform the original system (1) into the following n-step-ahead predictor:

$$y(k+n) = x_1(k+n) = f_s(\bar{x}_n(k), u(k)), \tag{10}$$

where $f_s(\bar{x}_n(k), u(k))$ is a composite nonlinear function of original system (1). Moreover, it is easy from Assumption 1 to verify that $0 < \underline{g} < \partial f_s(\bar{x}_n(k), u(k))/\partial u(k) = \prod_{i=1}^{n} g_i(.) := g_s(k) < \bar{g}$, where $\underline{g} = \prod_{i=1}^{n} \underline{g}_i$, and $\bar{g} = \prod_{i=1}^{n} \bar{g}_i$.

In what follows, a state-feedback adaptive NN controller is constructed by using the implicit function theorem. Define the tracking error as $e_s(k) = x_1(k) - y_d(k)$. Then, it follows from (10) that

$$e_s(k+n) = f_s(\bar{x}_n(k), u(k)) - y_d(k+n). \tag{11}$$

According to the implicit function theorem [10], there exists a unique ideal control input $u_s^*(k)$ such that

$$f_s(\bar{x}_n(k), u_s^*(Z_s(k))) - y_d(k+n) = 0, \quad Z_s(k) = [\bar{x}_n^{\mathrm{T}}(k), y_d(k+n)]^{\mathrm{T}}. \tag{12}$$

Owing to the unknown system dynamics, the ideal controller $u_s^*(Z_s(k))$ cannot be solved directly. With the help of RBF NN, there exist ideal weights vector W_s^*, such that the ideal controller can be approximated by RBF NN (3) as

$$u_s^*(k) = W_s^{*\mathrm{T}} S_s(Z_s(k)) + \epsilon_s(k), \tag{13}$$

where $|\epsilon_s(k)| < \bar{\epsilon}_s$ is the NN approximation error. Then, the state-feedback adaptive NN controller is designed as

$$u_s(k) = \hat{W}_s^{\mathrm{T}}(k) S_s(Z_s(k)) \tag{14}$$

with the weight updating law

$$\hat{W}_s(k+1) = \hat{W}_s(k_1) - \Gamma_s S_s(Z_s(k_1)) e_s(k+1), \tag{15}$$

where $\hat{W}_s(k)$ is the estimate of W_s^*, $\Gamma_s = \Gamma_s^{\mathrm{T}} > 0$ with $\lambda_{\max}(\Gamma_s) = r_s$, and $k_1 = k - n + 1$. Define $\tilde{W}_s(k) = \hat{W}_s(k) - W_s^*$. Substituting (12)–(14) into (11) yields

$$e_s(k+n) = f_s(\bar{x}_n(k), u_s(k)) - f_s(\bar{x}_n(k), u_s^*(k)) = g_s(\bar{x}_n(k), u_s^c(k))(\tilde{W}_s^{\mathrm{T}}(k) S_s(Z_s(k)) - \epsilon_s(k)), \tag{16}$$

where $u_s^c(k) \in [\min\{u_s(k), u_s^*(k)\}, \max\{u_s(k), u_s^*(k)\}]$, according to the mean value theorem. For brevity, let $S_s(k) = S_s(Z_s(k))$, and $g_s(k) = g_s(\bar{x}_n(k), u_s^c(k))$.

Theorem 1. Consider the closed-loop system consisting of the plant (1) under Assumptions 1 and 2, the control input (14), the weight updating law (15), and the bounded initial conditions including $\bar{x}_n(1), \hat{W}_s(1), \ldots, \hat{W}_s(n)$. For the design parameter Γ_s appropriately chosen, the weights estimate vectors $\hat{W}_{s\zeta}(k^1), \hat{W}_{s\zeta}(k^2), \ldots, \hat{W}_{s\zeta}(k^n)$ converge exponentially to small neighborhoods of the ideal weights vector $W_{s\zeta}^*$, respectively.

Proof. Substituting (16) into (15) yields

$$\tilde{W}_s(k+1) = (I - \Gamma_s g_s(k_1) S_s(k_1) S_s^{\mathrm{T}}(k_1)) \tilde{W}_s(k_1) + \Gamma_s S_s^{\mathrm{T}}(k_1) g_s(k_1) \epsilon_s(k_1). \tag{17}$$

Further, one has

$$\tilde{W}_s(k+n) = A_s(k) \tilde{W}_s(k) + d_s(k), \tag{18}$$

where $A_s(k) = I - g_s\Gamma_s(k)S_s(k)S_s^{\mathrm{T}}(k)$ and $d_s(k) = \Gamma_s S_s(k)g_s(k)\epsilon_s(k)$. According to Corollary 1, the exponential convergence of $\tilde{W}_s(k)$ depends on the small disturbance $d_s(k)$ and the PE condition of $S_s(k)$. Therefore, we need to prove the following two conditions one by one:

(1) The tracking error $e_s(k)$ converges to a small neighborhood around zero in a finite time;

(2) The subvector $S_{s\zeta}(k)$ satisfies the PE condition, and then the weights estimate vectors $\hat{W}_{s\zeta}(k^1)$, $\hat{W}_{s\zeta}(k^2), \ldots, \hat{W}_{s\zeta}(k^n)$ can converge exponentially to small neighborhoods of the optimal value $W^*_{s\zeta}$, respectively.

Based on the above analysis, the proof of Theorem 1 is divided into two parts.

(1) To verify the convergence of $e_s(k)$, the Lyapunov function candidate is constructed as

$$V_s(k) = \frac{1}{\bar{g}}e_s^2(k) + \sum_{j=0}^{n-1}\tilde{W}_s^{\mathrm{T}}(k_1+j)\Gamma_s^{-1}\tilde{W}_s(k_1+j). \tag{19}$$

The first difference of $V_s(k)$ is

$$\Delta V_s = \frac{1}{\bar{g}}e_s^2(k+1) - \frac{1}{\bar{g}}e_s^2(k) + S_s^{\mathrm{T}}(k_1)\Gamma_s S_s(k_1)e_s^2(k+1) - 2\tilde{W}_s^{\mathrm{T}}(k_1)S_s(k_1)e_s(k+1). \tag{20}$$

By combining (16), the Young's inequality, $|\epsilon_s(k)| < \bar{\epsilon}_s$, and $\|S_s(k)\| \leqslant l_s$ [30], one has

$$\begin{aligned}
&\tilde{W}_s^{\mathrm{T}}(k_1)S_s(k_1)e_s(k+1) = \frac{1}{g_s(k_1)}e_s^2(k+1) + e_s(k+1)\epsilon_s(k_1),\\
&S_s^{\mathrm{T}}(k_1)\Gamma_s S_s(k_1) \leqslant r_s l_s^2,\\
&2e_s(k+1)\epsilon_s(k_1) < e_s^2(k+1) + \bar{\epsilon}_s^2.
\end{aligned} \tag{21}$$

Substituting (21) into (20) leads to

$$\Delta V_s \leqslant -\frac{1}{\bar{g}}e_s^2(k) - \frac{1 - 2r_s - r_s l_s^2\bar{g}}{\bar{g}}e_s^2(k+1) + \frac{\bar{g}}{r_s}\bar{\epsilon}_s^2. \tag{22}$$

Further, by choosing the design parameter $r_s < \frac{1}{1+l_s^2\bar{g}}$, Eq. (22) is rewritten as

$$\Delta V_s \leqslant -\frac{1}{\bar{g}}e_s^2(k) + \frac{\bar{g}}{r_s}\bar{\epsilon}_s^2. \tag{23}$$

From (23), there exist a small constant $\mu_s > \sqrt{\frac{\bar{g}^2\bar{\epsilon}_s^2}{r_s}}$ and a finite positive integer K_s, for all $k > K_s$, such that $e_s(k) < \mu_s$. Owing to the small approximation error $\bar{\epsilon}_s$, the tracking error $e_s(k)$ can converge to a small neighborhood around zero. Furthermore, according to the definition of $d_s(k)$ in (18), we can conclude that $d_s(k)$ can be made arbitrarily small with small $\bar{\epsilon}_s$ and r_s.

(2) In this part, we need to verify that $S_{s\zeta}(k)$ satisfies the PE condition. It can be shown from Lemma 1 that the key problem is to verify the recurrent property of all the NN input variables $Z_s(k) = [\bar{x}_n^{\mathrm{T}}(k), y_d(k+n)]^{\mathrm{T}}$. From (23), $e_s(k)$ can be made arbitrarily small after a finite time K_s. Since $e_s(k) = x_1(k) - y_d(k)$ and the reference signal $y_d(k)$ is recurrent, it is easy to obtain that the state $x_1(k)$ is also recurrent. From the first subsystem state equation of (1), one has

$$x_1(k+1) = f_1(x_1(k), x_2(k)). \tag{24}$$

According to implicit function theorem [10], there exists a unique $x_2(k)$ such that $f_1(x_1(k), x_2(k)) = x_1(k+1)$, i.e., there exists a unique nonlinear mapping F_1 that makes $x_2(k) = F_1(x_1(k), x_1(k+1))$ hold. Since $x_1(k)$ is recurrent and $x_2(k)$ is a nonlinear function of $x_1(k)$, we can infer that $x_2(k)$ is also recurrent. Similarly, we can infer that all the system states $x_i(k)$ $(i = 1, \ldots, n)$ are recurrent through the system state equation (1) and the implicit function theorem. Therefore, all the NN input variables $Z_s(k)$ are recurrent for $k > K_s$. According to Lemma 1, along the recurrent trajectory $Z_s(k)$, the subvector $S_{s\zeta}(Z_s(k))$ satisfies the PE condition.

Based on the localized approximation ability of RBF NNs, the dynamical equation (18) is rewritten along $\phi(Z_s(k))$ as

$$\tilde{W}_{s\zeta}(k+n) = A_{s\zeta}(k)\tilde{W}_{s\zeta}(k) + d_{s\zeta}(k), \tag{25}$$

where $A_{s\zeta}(k) = I_\zeta - g_s(k)\Gamma_{s\zeta}S_{s\zeta}(k)S_{s\zeta}^{\mathrm{T}}(k)$, and $d_{s\zeta}(k) = \Gamma_{s\zeta}S_{s\zeta}(k)g_s(k)\epsilon_{s\zeta}(k)$. Eq. (25) represents a class of discrete-time LTV systems shown in Corollary 1.

By choosing the appropriate parameter Γ_s, we can obtain that $g_s(k)\Gamma_{s\zeta}S_{s\zeta}(k)S_{s\zeta}^{\mathrm{T}}(k) < 2I_\zeta$. By combining the PE condition of $S_{s\zeta}(k)$ and the small $d_{s\zeta}(k)$, it can be verified from Corollary 1 that the weights estimate error vectors $\tilde{W}_{s\zeta}(k^1), \tilde{W}_{s\zeta}(k^2), \ldots, \tilde{W}_{s\zeta}(k^n)$ converge exponentially to small neighborhoods of zero with n as the time step, respectively. This implies that along the recurrent trajectory $\phi(Z_s(k))$, the weights estimate vectors $\hat{W}_{s\zeta}(k^1), \hat{W}_{s\zeta}(k^2), \ldots, \hat{W}_{s\zeta}(k^n)$ converge exponentially to small neighborhoods of the optimal vector W_s^*, respectively. On the other hand, for $\hat{W}_{s\bar{\zeta}}(k)$ far away from the recurrent trajectory, they almost keep unchanged because of the small $S_{s\bar{\zeta}}(k)$.

Remark 1. Lemma 1 and Corollary 1 show that the exponential convergence of estimated weights relies on the recurrent property of all the NN input variables. Unlike the existing methods on the continuous-time case [32,33], the n-step-ahead predictor is employed to obtain one weight estimate error system (25) with n-step delays. Owing to the existence of the delays and the different structure features of discrete-time nonlinear systems, these existing methods [32, 33] are invalid to solve the exponential convergence of estimated weights of discrete-time PFNSs (1). To tackle these problems, the following three dedicated actions are taken herein: (1) Corollary 1 is proposed and strictly proven to guarantee the exponential convergence of discrete-time LTV system with n-step delays; (2) the structural feature of discrete-time system (1) and the implicit function theorem are combined to verify that all NN input variables $Z_s(k)$ are recurrent; (3) the weights estimate vector $\hat{W}_s(k)$ is verified to converge to a sequence $\{\hat{W}_s(k)|\hat{W}_s(k_s^1), \hat{W}_s(k_s^2), \ldots, \hat{W}_s(k_s^n)\}$ rather than one fixed value.

3.2 Output-feedback control

Generally speaking, the system state variables are often immeasurable in practice. In this case, it is impossible to implement the state-feedback control scheme mentioned above. To solve such a problem, the transformation procedure [21] is employed to convert the original system (1) into the following n-step-ahead input-output predictor:

$$y(k+n) = x_1(k+n) = f_o(\bar{z}(k), u(k)), \tag{26}$$

where $\bar{z}(k) = [\underline{y}^{\mathrm{T}}(k), \underline{u}^{\mathrm{T}}(k-1)]^{\mathrm{T}}$, $\underline{y}(k) = [y(k-n+1), \ldots, y(k)]^{\mathrm{T}}$, $\underline{u}(k-1) = [u(k-n+1), \ldots, u(k-1)]^{\mathrm{T}}$, and $f_o(\bar{z}(k), u(k))$ is a composite nonlinear function of the original system (1). It has been shown from [21] that $0 < \underline{g} < \partial f_o(\bar{z}(k), u(k))/\partial u(k) := g_o(k) < \bar{g}$. Similar to the state-feedback process, The tracking error is defined as $e_o(k) = x_1(k) - y_d(k)$, and its nth difference is described by

$$e_o(k+n) = f_o(\bar{z}(k), u(k)) - y_d(k+n). \tag{27}$$

According to the implicit function theorem [10], there exists a unique ideal control input $u_o^*(k)$ such that

$$f_o(\bar{z}(k), u_o^*(Z_o(k))) - y_d(k+n) = 0, \quad Z_o(k) = [\bar{z}^{\mathrm{T}}(k), y_d(k+n)]^{\mathrm{T}}. \tag{28}$$

Then, the RBF NN (3) is employed to approximate the control input as: $u_o^*(k) = W_o^{*\mathrm{T}}S_o(Z_o(k)) + \epsilon_o(k)$, where W_o^* is the ideal weights vector, and $\epsilon_o(k) < \bar{\epsilon}_o$ is the NN approximation error. Afterwards, the output-feedback adaptive neural controller is designed as

$$u_o(k) = \hat{W}_o^{\mathrm{T}}(k)S_o(Z_o(k)) \tag{29}$$

with the weight updating law

$$\hat{W}_o(k+1) = \hat{W}_o(k_1) - \Gamma_o S_o(Z_o(k_1))e_o(k+1), \tag{30}$$

where $\hat{W}_o(k)$ is the estimate of W_o^*, $\Gamma_o = \Gamma_o^{\mathrm{T}} > 0$ with $\lambda_{\max}(\Gamma_o) = \gamma_o$, and $k_1 = k - n + 1$. Define $\tilde{W}_o(k) = \hat{W}_o(k) - W_o^*$. Substituting (29) into (27) yields

$$e_o(k+n) = g_o(\bar{z}(k), u_o^c(k))(\tilde{W}_o^{\mathrm{T}}(k)S_o(Z_o(k)) - \epsilon_o(k)), \tag{31}$$

where $u_o^c(k) \in [\min\{u_o(k), u_o^*(k)\}, \max\{u_o(k), u_o^*(k)\}]$.

Theorem 2. Consider the closed-loop system consisting of the plant (1) under Assumptions 1 and 2, the control input (29), the weight updating law (30), and the bounded initial conditions including $\bar{x}_n(1), \hat{W}_o(1), \ldots, \hat{W}_o(n)$. For the design parameter Γ_o appropriately chosen, the weights estimate vectors $\hat{W}_{o\zeta}(k^1), \hat{W}_{o\zeta}(k^2), \ldots, \hat{W}_{o\zeta}(k^n)$ converge exponentially to small neighborhoods of the optimal value $W^*_{o\zeta}$, respectively.

Proof. This proof is similar to Theorem 1. Owing to the space limit, this proof is omitted.

4 Neural learning control for discrete-time PFNSs

This section focuses on the knowledge representation and storage of neural estimated weights from the steady-state control process of discrete-time PFNSs, and the knowledge reuses for the higher control performance. To achieve such an objective, a learning rule is firstly summarized, and then the knowledge of the neural estimated weights is represented duly even if there are n-step delays. Based on the stored knowledge, static learning controllers are constructed for discrete-time PFNSs to achieve better control performance including the transient performance and the computational burden.

4.1 Neural learning control based on state-feedback case

In Section 3, we have shown that the estimated weights $\hat{W}_s(k)$ can exponentially converge to their optimal values. But the weights are verified to converge to a constant sequence $\{\hat{W}_s(k)|\hat{W}_s(k^1), \hat{W}_s(k^2), \ldots, \hat{W}_s(k^n)\}$, rather than a fixed value, owing to the n-step delays. This sequence presents a great challenge on the knowledge representation and storage of $\hat{W}_s(k)$. Such a challenge will be solved by constructing some dedicated "learning rules".

4.1.1 *Knowledge representation and storage*

It has been proven in Theorem 1 that the estimated weights $\hat{W}_s(k)$ converge exponentially to small neighborhoods of their optimal values. According to (15), one has

$$\hat{W}_s(k+n) = \hat{W}_s(k) - \Gamma_s S_s(Z_s(k)) e_s(k+1). \tag{32}$$

From (32), the estimated weights $\hat{W}_s(k+n)$ at moment $k+n$ is updated by $\hat{W}_s(k)$ at moment k.

Based on this analysis, we define the following "learning rules".

(1) $k_{s,a}$ stands for the beginning of the learning process, which is chosen to make the $k_{s,a}-1$ be a multiple of n.

(2) T_s represents the total number of learning steps, which is chosen as a multiple of n. Moreover, $k_{s,b} = T_s + k_{s,a} - 1$ stands for the end of the learning process.

(3) $T_{s1} = T_s/n$ represents the number of updating period in learning process.

(4) $\{k^j|\ k^j = j, j+n, j+2n, \ldots\}$ $(j = 1, \ldots, n)$ is used as time sequence for the jth weights vector updating.

(5) $\bar{W}_{sj}$ $(j = 1, \ldots, n)$ represents the jth constant weights vector, which is represented as

$$\bar{W}_{sj} = \frac{1}{T_{s1}} \sum_{k^j = k_{s,a}+j-1}^{k_{s,b}+j-n} \hat{W}_s(k^j). \tag{33}$$

Next, we employ the stored constant weights vector $\bar{W}_{sj}$ $(j = 1, 2, \ldots, n)$ in (33) to show the accurate approximation of unknown system ideal control input $u^*_s(Z_s(k))$. For the localization property of RBF NNs (4), the unknown system ideal control input $u^*_s(Z_s(k))$ along the recurrent trajectory $\phi(Z_s(k))$ can be accurately approximated by

$$\begin{aligned} &u^*_s(Z_s(k)) = \hat{W}^{\mathrm{T}}_s(k) S_s(Z_s(k)) + \varepsilon_{s1}(k) = \hat{W}^{\mathrm{T}}_{s\zeta}(k) S_{s\zeta}(Z_s(k)) + \varepsilon_{s2}(k) = \bar{W}^{\mathrm{T}}_{sl} S_s(Z_s(k)) + \varepsilon_{s3}(k), \\ &l = \begin{cases} k \bmod n, & \text{if } k \bmod n \neq 0, \\ n_i, & \text{if } k \bmod n = 0, \end{cases} \end{aligned} \tag{34}$$

where $\varepsilon_{sm}(k)$ $(m = 1, 2, 3)$ is the corresponding approximation error. The local accurate NN approximation region Ω_ϕ is described by

$$\text{dist}(\phi(Z_s(k)), Z_s(k)) < p \Rightarrow |u_s^*(Z_s(k)) - \bar{W}_{sl}^{\mathrm{T}} S_s(Z_s(k))| < \varepsilon_s, \tag{35}$$

where $p > 0$ is a positive constant and ε_s is the small approximation error.

Remark 2. Different from the existing neural learning control schemes [32,33] for the continuous-time systems, the weights estimate vector $\hat{W}_s(k)$ converges to n different values at the corresponding moments k^j, $(j = 1, 2, \ldots, n)$, respectively, rather than a fixed value. This convergence feature raises a significant challenge to the knowledge representation and storage of $\hat{W}_s(k)$. To tackle such a problem, we construct some dedicated "learning rules" to represent and store the constant weights, and then achieve an exact approximation of the unknown system ideal control input by combining the constant weights with a "mod" function (34).

Remark 3. The core idea of the "learning rules" is to help us acquire the convergent weights $\hat{W}_s(k^1)$, $\hat{W}_s(k^2), \ldots, \hat{W}_s(k^n)$ and store them as constant weights $\bar{W}_{s1}, \bar{W}_{s2}, \ldots, \bar{W}_{sn}$. Then the constant weights $\bar{W}_{s1}, \bar{W}_{s2}, \ldots, \bar{W}_{sn}$ can be reused at the corresponding time $k_1, k_2, \ldots, k_n$, respectively, with the help of a "mod" function. Therefore, $k_{s,a} - 1$ is selected as a multiple of n so that $k_{s,a}$ belongs to a moment of time sequence k^1; $k_{s,b} = T_s + k_{s,a} - 1$ and T_s are multiples of n so that $k_{s,b}$ belongs to a moment of time sequence k^n. Through the above efforts, the constant weights $\bar{W}_{sj}$ $(j = 1, 2, \ldots, n)$ can be obtained and stored by a simple mean function (33).

4.1.2 *Neural learning controller with knowledge reuse*

In this subsection, we will reuse $\bar{W}_{sl}^{\mathrm{T}} S_s(Z_s(k))$ to develop the neural learning controller for some same or similar tracking tasks. Consider the equivalent transformation form (10) of the original discrete-time SFNSs (1). Instead of the adaptive NN controller (14) with the weight updating law (15), we employ $\bar{W}_{sl}^{\mathrm{T}} S_s(Z_s(k))$ to design the following neural learning controller:

$$u_s(k) = \bar{W}_{sl}^{\mathrm{T}} S_s(Z_s(k)), \tag{36}$$

where $\bar{W}_{sl}$ $(l = 1, 2, \ldots, n)$ is given in (33), and l is defined in (34). Substituting (36) into (11), we can obtain the closed-loop error system

$$e_s(k + n) = g_s(k)\tilde{h}_s(k), \tag{37}$$

where $\tilde{h}_s(k) = u_s(k) - u_s^*(k)$, and $\tilde{h}_s(k) < \varepsilon_s$.

Theorem 3. Consider the closed-loop system consisting of the plant (1) and the learning controller (36) with stored constant weights vector given in (33). For the same or similar tracking control tasks as Theorem 1, we can obtain that the tracking error converges to a small neighborhood of zero, and all the closed-loop signals remain bounded.

Proof. Consider the following Lyapunov function candidate

$$V(k) = \frac{1}{\bar{g}^2} e_s^2(k). \tag{38}$$

Using (37), the difference of $V(k)$ is

$$\Delta V(k) < -\frac{1}{\bar{g}^2} e_s^2(k) + \varepsilon_s^2. \tag{39}$$

From (39), the tracking error satisfies $|e_s(k)| < \bar{\mu}_s$, where $\bar{\mu}_s > \sqrt{\bar{g}^2 \varepsilon_s^2}$. This means that the tracking error $e_s(k)$ converges to a small neighborhood around zero owing to the arbitrarily small ε_s. Noting $x_1(k) = y_d(k) + e_1(k)$, we have $x_1(k)$ is bounded. Based on the first equation of the system (1), $x_2(k)$ is bounded because of the boundedness of $x_1(k)$. Using a similar process, we can obtain that $x_i(k)$ $(i = 3, \ldots, n)$ is also bounded. From (36), $u_s(k)$ is bounded since $Z_s(k) = [\bar{x}_n^{\mathrm{T}}(k), y_d(k + n)]^{\mathrm{T}}$ is bounded and $\bar{W}_{sl}$ is the stored constant weights vector. As such, it can be concluded that the tracking error converges to a small neighborhood of zero and all the signals remain bounded.

Remark 4. It is worth pointing out that some adaptive NN control schemes [20–22] have been proposed for the discrete-time low-triangular nonlinear systems. In these schemes, the NN learning ability is not a major concern. Owing to the large NN approximation error, these schemes [20–22] usually encounter a poor transient process including the long transient time and the large transient error. Unlike the existing schemes [20–22], the proposed neural learning controller greatly improves the transient control performance for the same or similar tracking tasks owing to the high function approximation accuracy. Moreover, the proposed neural learning control scheme greatly alleviates the computation burden owing to the avoidance of updating estimated weights online.

4.2 Neural learning control based on output-feedback case

Similar to state-feedback case in the above section, the knowledge representation, storage, and reuse can be realized through the adaptive NN output feedback control process. The corresponding "learning rules" are defined as follows:

(1) $k_{o,a}$ stands for the beginning of the learning process, which is chosen to make the $k_{o,a}-1$ be a multiple of n.

(2) T_o represents the total number of learning steps, which is chosen as a multiple of n. Moreover, $k_{o,b}=T_o+k_{o,a}-1$ stands for the end of the learning process.

(3) $T_{o1}=T_o/n$ represents the number of updating period for the weights vector in learning process.

(4) $\{k^j|\ k^j=j,j+n,j+2n,\ldots\}$ $(j=1,\ldots,n)$ is used as time sequence for the jth weights vector updating.

(5) $\bar{W}_{oj}$ $(j=1,\ldots,n)$ represents the jth constant weights vector, which is expressed as

$$\bar{W}_{oj}=\frac{1}{T_{o1}}\sum_{k^j=k_{o,a}+j-1}^{k_{o,b}+j-n}\hat{W}_o(k^j). \tag{40}$$

Next, we employ the stored constant weights vector $\bar{W}_{oj}$ $(j=1,2,\ldots,n)$ to design the following output-feedback NN learnign controller:

$$u_o(k)=\bar{W}_{ol}^{\mathrm{T}}S_o(Z_o(k)),\quad l=\begin{cases}k \bmod n, & \text{if } k \bmod n\neq 0,\\ n, & \text{if } k \bmod n=0.\end{cases} \tag{41}$$

Theorem 4. Consider the closed-loop system consisting of the plant (1) and the learning controller (41) with the stored constant weights vector given in (40). For the same or similar tracking control tasks as Theorem 2, we can obtain that the tracking error converges to a small neighborhood of zero, and all the closed-loop signals remain bounded.

Proof. It is similar to the proof of Theorem 3 and is thus omitted.

5 Simulation studies

In this section, a second-order pure-feedback system is simulated to illustrate the effectiveness of the proposed state-feedback and output-feedback methods. Consider the following second-order pure-feedback system [27] described by

$$\begin{cases}x_1(k+1)=\dfrac{0.2x_1^2(k)x_2(k)}{1+x_1^2(k)}+0.5x_2(k),\\ x_2(k+1)=\dfrac{x_1}{1+x_1^2(k)+x_2^2(k)}+u(k)+0.2\sin(u(k)),\\ y(k)=x_1(k).\end{cases} \tag{42}$$

The desired reference signal is $y_d(k)=0.5\sin(k\pi/200)+0.5\sin(k\pi/100)$.

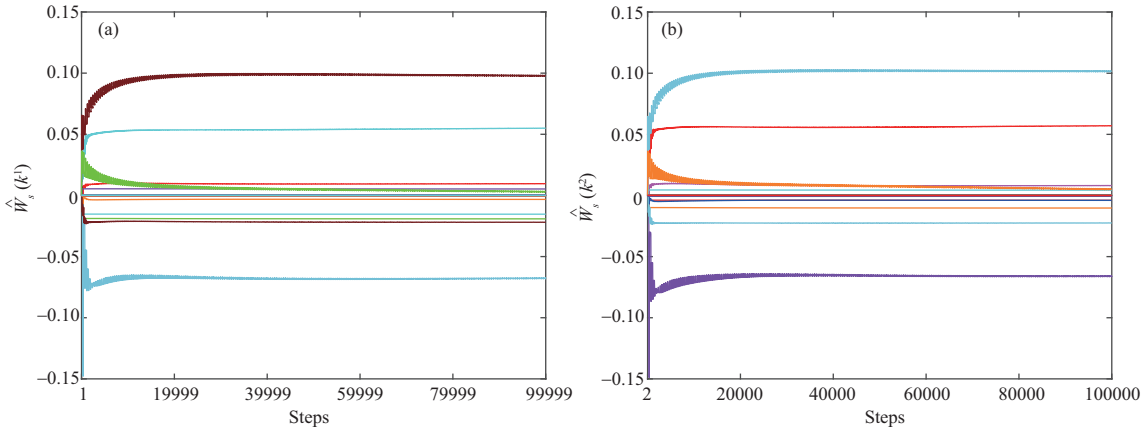

Figure 1 (Color online) Partial NN weights convergence of (a) $\hat{W}_s(k^1)$ and (b) $\hat{W}_s(k^2)$.

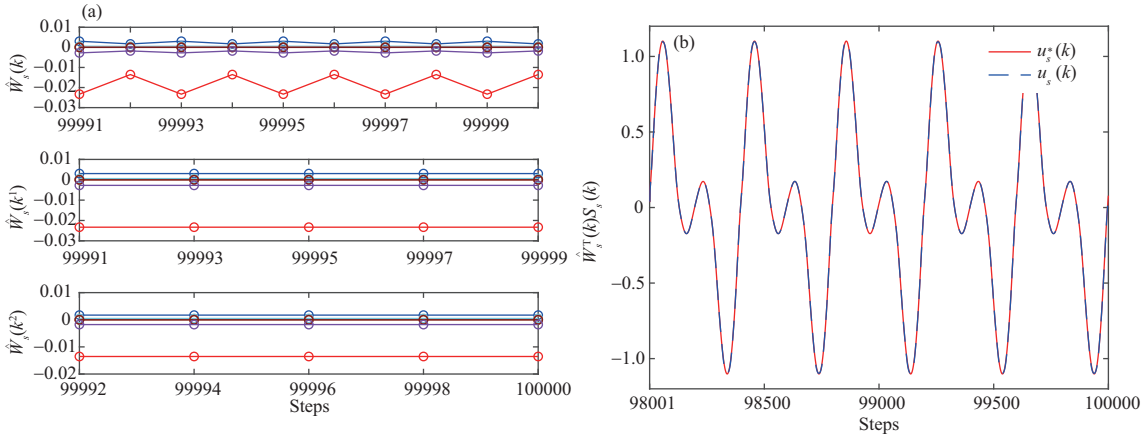

Figure 2 (Color online) (a) Partial NN weights comparison; (b) system ideal input $u_s^*(k)$ and $\hat{W}_s^{\mathrm{T}}(k)S_s(k)$.

5.1 State-feedback case

5.1.1 *Simulation results of knowledge acquirement*

According to Theorem 1, the control input is chosen as (14) and the weight updating law is chosen as (15). We choose RBF NN $\hat{W}_s^{\mathrm{T}}S_s(Z_s(k))$ using 1815 nodes, with the centers $Z_s(k)=[x_1(k),x_2(k),y_d(k+2)]^{\mathrm{T}}$ evenly spaced in $[-1.5,1.5]\times[-2.1,2.1]\times[-1.5,1.5]$ and the width $\eta=0.375$. The design parameter is $\Gamma_s=0.2I$. The initial NN weights and the initial states are $\hat{W}_s(1)=\hat{W}_s(2)=0$ and $[x_1(1),x_2(1)]=[0.1,0.1]$. Simulation results are displayed in Figures 1 and 2. From Figures 1(a) and (b), the estimated weights $\hat{W}_s(k)$ converge to $\hat{W}_s(k^1)$ and $\hat{W}_s(k^2)$ with the sequence $\{k^1|k^1=1,3,5,\ldots,2k-1,\ldots\}$ and $\{k^2|k^2=2,4,6,\ldots,2k,\ldots\}$, respectively. Figure 2(a) further verifies that $\hat{W}_s(k)$ is not exponentially convergent in the whole sampling moments $k=1,2,3,\ldots$. This phenomenon is consistent with Theorem 1, which is different from the continuous-time case in nature. Figure 2(b) shows that after sufficient learning, the NN control input $u_s(k)$ approaches the implicit desired control input $u^*(k)$ exactly within the steady-state time interval $[98001,100000]$.

5.1.2 *Simulation results of knowledge reuse*

In order to verify the proposed state-feedback neural learning control (NLC) method, we employ the same system and reference signal $y_d(k)$ in (42). According to the "learning rules" in Subsection 4.1.1, the constant weights vectors are represented and stored as

$$\bar{W}_{s1}=\frac{1}{1000}\sum_{k^1=98001}^{99999}\hat{W}_s(k^1),\quad \bar{W}_{s2}=\frac{1}{1000}\sum_{k^2=98002}^{100000}\hat{W}_s(k^2). \tag{43}$$

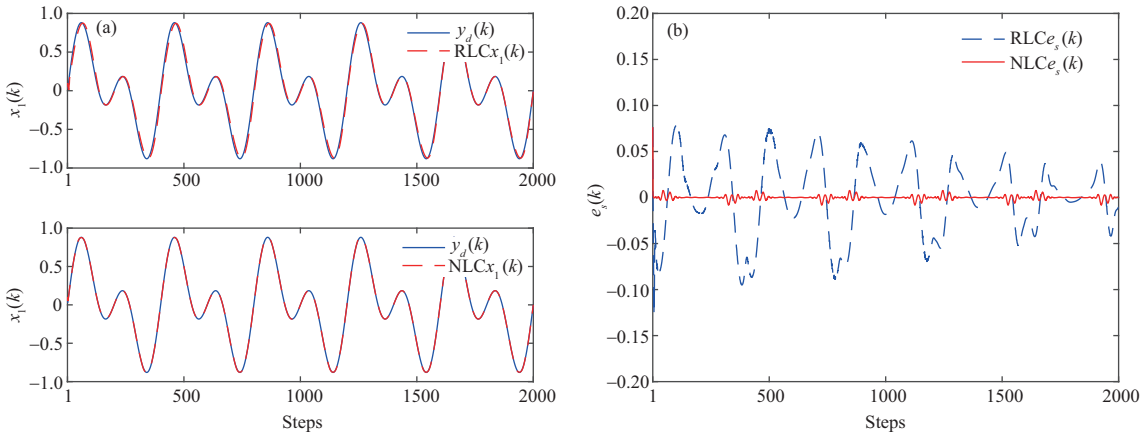

Figure 3 (Color online) (a) $y_d(k)$ and $x_1(k)$ by RLC [27] and NLC; (b) tracking error $e_s(k)$ by RLC [27] and NLC.

Table 1 Simulation comparison between RLC [27] and NLC for state-feedback case[a)]

Method	Overshoot	$\text{MAE}_{[1,2000]}$	Running time (s)
RLC [27]	0.1284	0.0331	15.0756
NLC	0.0082	0.0012	10.3931

a) "$\text{MAE}_{[1,2000]}$" denotes the mean absolute error during [1,2000].

Reusing (43), the neural learning controller is designed by "mod" function as

$$u(k)=\begin{cases}\bar{W}_{s1}^{\mathrm{T}}S_s(Z_s(k)), & \text{if}(k \bmod 2)=1,\\ \bar{W}_{s2}^{\mathrm{T}}S_s(Z_s(k)), & \text{if}(k \bmod 2)=0,\end{cases} \tag{44}$$

where $S_s(Z_s(k))$ is chosen to be the same as that used in Subsection 5.1.1.

To show the performance of the proposed NLC scheme, we compare the proposed NLC with the reinforcement learning control (RLC) method in [27]. Simulation results are displayed in Figure 3 and Table 1. Figure 3(a) shows the curves of $y(k)$ and $y_d(k)$ by RLC and NLC, respectively, and Figure 3(b) displays the evolution of the corresponding tracking error $e_s(k)$. Compared with the RLC, it is revealed from Figure 3 and Table 1 that the proposed NLC achieves better tracking control performance with a smaller overshoot and a smaller mean absolute error. Moreover, to compare the online calculation consumption of RLC and NLC, the same computer settings are used to run the same 10^5-steps simulation. It is easy to see from Table 1 that the time saving is nearly 1/3 using the proposed NLC method. The simulation results clearly show that the proposed NLC not only achieves the better tracking performance, but also eases the computation burden.

5.2 Output-feedback case

5.2.1 *Simulation results of knowledge acquirement*

According to Theorem 2, the output-feedback adaptive neural controller is chosen as (29) and the weight updating law is chosen as (30). We construct the RBF NN $\hat{W}_o^{\mathrm{T}}S_o(Z_o(k))$ using 5103 nodes, with the centers $Z_o(k)=[x_1(k),x_1(k-1),y_d(k+2),u_o(k-1)]^{\mathrm{T}}$ evenly spaced on $[-1.2,1.2]\times[-1.2,1.2]\times[-1.2,1.2]\times[-1.5,1.5]$ and the width $\eta=0.3$. The design parameter is $\Gamma_o=0.22I$. The initial NN weights and the initial states are $W_o(1)=W_o(2)=0$ and $[x_1(1),x_2(1)]=[0.1,0.1]$. Figures 4(a) and (b) display the convergence of the partial estimated weights. Figure 5(a) displays the accurate approximation ability of NNs in the steady-state time interval $[98001,100000]$.

5.2.2 *Simulation results of knowledge reuse*

In this part, the constant weights vectors $\bar{W}_{o1}$ and $\bar{W}_{o2}$ are stored based on (40) with the steady-state time interval $[98001,100000]$. $S_o(Z_o(k))$ is selected in the same way as that in Subsection 5.2.1, and the output-feedback NLC (41) is employed to complete the same tracking task. The tracking performance is displayed in Figure 5(b) and Table 2. Compared with the RLC [27], it is obviously seen from Figure 5(b) and Table 2 that the proposed NLC realizes the better tracking performance. Moreover, it is easy to see

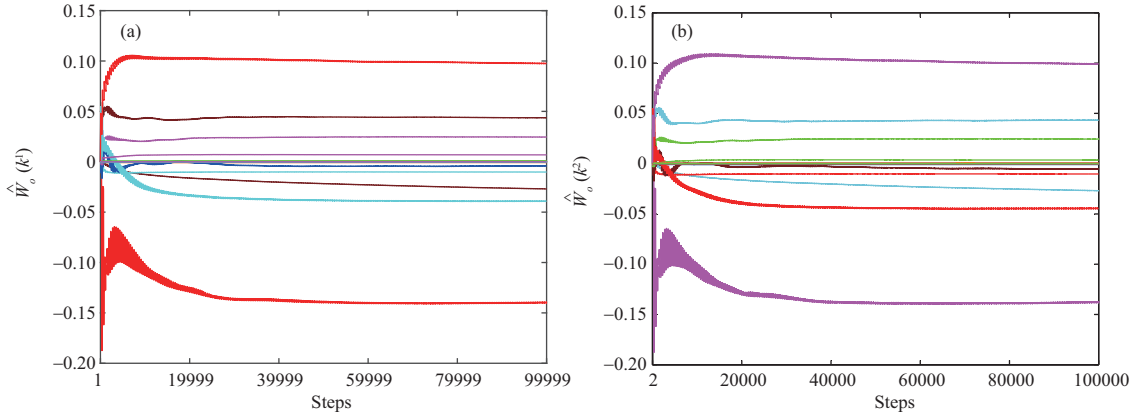

Figure 4 (Color online) Partial NN weights convergence of (a) $\hat{W}_o(k^1)$ and (b) $\hat{W}_o(k^2)$.

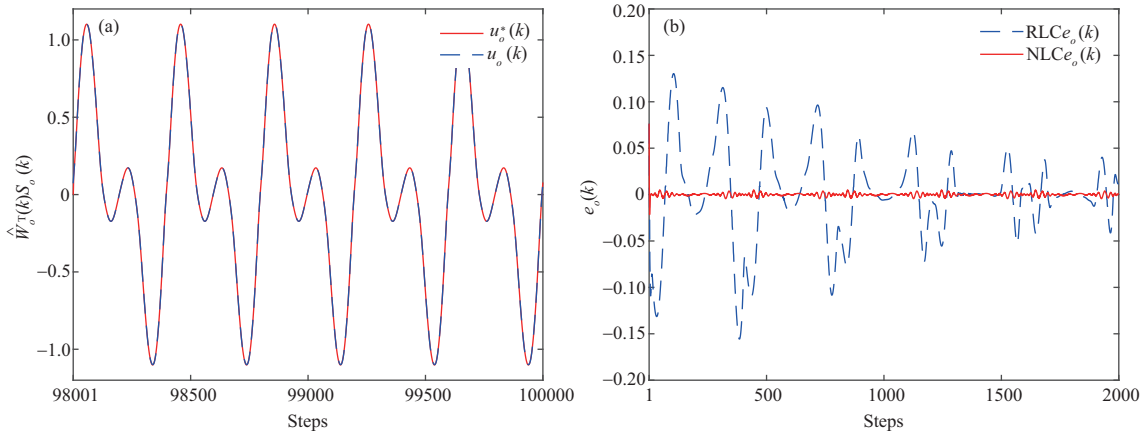

Figure 5 (Color online) (a) System ideal input $u_o^*(k)$ and $\hat{W}_o^{\mathrm{T}}(k)S_o(k)$; (b) tracking error $e_o(k)$ by RLC [27] and NLC.

Table 2 Simulation comparison between RLC [27] and NLC for output-feedback case

Method	Overshoot	$\mathrm{MAE}_{[1,2000]}$	Running time (s)
RLC [27]	0.1558	0.0343	38.4058
NLC	0.0215	0.0013	25.5094

from Table 2 that the time saving is nearly 1/3 owing to the constant weights without the online adaptive adjustment.

6 Conclusion

Herein, we have investigated neural learning control issues in discrete-time PFNSs with unknown non-affine terms. First, two effective adaptive NN controllers have been presented through state-feedback and output-feedback, respectively. By combining the extended stability result of the LTV system, the estimated weights have been verified to exponentially converge to optimal values. The estimated weights have been systematically represented and stored as a set of constant values by constructing the programmatic "learning rules". Next, the stored weights have been reused to structure neural learning controllers using the "mod" function. Compared with the conventional adaptive NN control, the proposed schemes can not only accomplish the same or similar tracking tasks but also greatly improve the transient control performance and alleviate online computation. Based on the schemes proposed herein, several promising directions for future investigations exist, including (i) neural learning control for other types of reference signals; (ii) neural learning control for more general nonlinear systems exhibiting different phenomena, including prescribed performances [15, 35] and constrained network resources [25, 26]; and (iii) the cooperative learning and control of multiagent systems [39, 41, 44, 45].

Acknowledgements This work was supported in part by Guangdong Natural Science Foundation (Grant No. 2019B151502058), National Natural Science Foundation of China (Grant Nos. 61773169, 61890922, 61973129), National Science Fund for Distinguished Young Scholars (Grant No. 61825301), National Key Research and Development Program of China (Grant No. 2018AAA0101603), Guangzhou Science and Technology Project (Grant No. 201904010295), Science and Technology Planning Project of Guangdong Province (Grant No. 2020B1111010002), and Guangdong Marine Economic Development Project (Grant No. 2020018).

References

1 Narendra K S, Parthasarathy K. Identification and control of dynamical systems using neural networks. IEEE Trans Neural Netw, 1990, 1: 4–27
2 Polycarpou M M. Stable adaptive neural control scheme for nonlinear systems. IEEE Trans Autom Control, 1996, 41: 447–451
3 Krstic M, Kanellakopoulos I, Kokotovic P V. Nonlinear design of adaptive controllers for linear systems. IEEE Trans Autom Control, 1994, 39: 738–752
4 Zhang T P, Ge S S, Hang C C. Adaptive neural network control for strict-feedback nonlinear systems using backstepping design. Automatica, 2000, 36: 1835–1846
5 Wang M, Zhang S Y, Chen B, et al. Direct adaptive neural control for stabilization of nonlinear systems with time-varying delays. Sci China Inf Sci, 2010, 53: 800–812
6 Tong S C, Li Y M. Robust adaptive fuzzy backstepping output feedback tracking control for nonlinear system with dynamic uncertainties. Sci China Inf Sci, 2010, 53: 307–324
7 Zhou Q, Zhao S Y, Li H Y, et al. Adaptive neural network tracking control for robotic manipulators with dead zone. IEEE Trans Neural Netw Learn Syst, 2019, 30: 3611–3620
8 Ge S S, Wang C. Adaptive NN control of uncertain nonlinear pure-feedback systems. Automatica, 2002, 38: 671–682
9 Wang D, Huang J. Adaptive neural network control for a class of uncertain nonlinear systems in pure-feedback form. Automatica, 2002, 38: 1365–1372
10 Wang C, Hill D J, Ge S S, et al. An ISS-modular approach for adaptive neural control of pure-feedback systems. Automatica, 2006, 42: 723–731
11 He W, David A O, Yin Z, et al. Neural network control of a robotic manipulator with input deadzone and output constraint. IEEE Trans Syst Man Cybern Syst, 2016, 46: 759–770
12 Chen M, Ren B B, Wu Q X, et al. Anti-disturbance control of hypersonic flight vehicles with input saturation using disturbance observer. Sci China Inf Sci, 2015, 58: 070202
13 Xu B, Yuan Y. Two performance enhanced control of flexible-link manipulator with system uncertainty and disturbances. Sci China Inf Sci, 2017, 60: 050202
14 Zhang T P, Xia M Z, Yi Y, et al. Adaptive neural dynamic surface control of pure-feedback nonlinear systems with full state constraints and dynamic uncertainties. IEEE Trans Syst Man Cybern Syst, 2017, 47: 2378–2387
15 Liu Y J, Tong S. Barrier Lyapunov functions for Nussbaum gain adaptive control of full state constrained nonlinear systems. Automatica, 2017, 76: 143–152
16 Dai S L, He S D, Wang M, et al. Adaptive neural control of underactuated surface vessels with prescribed performance guarantees. IEEE Trans Neural Netw Learn Syst, 2019, 30: 3686–3698
17 Chen M, Ge S S. Adaptive neural output feedback control of uncertain nonlinear systems with unknown hysteresis using disturbance observer. IEEE Trans Ind Electron, 2015, 62: 7706–7716
18 Tong S C, Li Y M. Observer-based adaptive fuzzy backstepping control of uncertain nonlinear pure-feedback systems. Sci China Inf Sci, 2014, 57: 012204
19 Yeh P C, Kokotović P V. Adaptive control of a class of nonlinear discrete-time systems. Int J Control, 1995, 62: 303–324
20 Ge S S, Li G Y, Lee T H. Adaptive NN control for a class of strict-feedback discrete-time nonlinear systems. Automatica, 2003, 39: 807–819
21 Ge S S, Yang C G, Lee T H. Adaptive predictive control using neural network for a class of pure-feedback systems in discrete time. IEEE Trans Neural Netw, 2008, 19: 1599–1614
22 Wang M, Wang Z D, Dong H L, et al. A novel framework for backstepping-based control of discrete-time strict-feedback nonlinear systems with multiplicative noises. IEEE Trans Autom Control, 2021, 66: 1484–1496
23 Shao S Y, Chen M. Sliding-mode-disturbance-observer-based adaptive neural control of uncertain discrete-time systems. Sci China Inf Sci, 2020, 63: 149204
24 Liu Y J, Li S, Tong S C, et al. Adaptive reinforcement learning control based on neural approximation for nonlinear discrete-time systems with unknown nonaffine dead-zone input. IEEE Trans Neural Netw Learn Syst, 2019, 30: 295–305
25 Wang M, Wang Z D, Chen Y, et al. Adaptive neural event-triggered control for discrete-time strict-feedback nonlinear systems. IEEE Trans Cybern, 2020, 50: 2946–2958
26 Sahoo A, Xu H, Jagannathan S. Adaptive neural network-based event-triggered control of single-input single-output nonlinear discrete-time systems. IEEE Trans Neural Netw Learn Syst, 2016, 27: 151–164
27 Xu B, Yang C G, Shi Z K. Reinforcement learning output feedback NN control using deterministic learning technique. IEEE Trans Neural Netw Learn Syst, 2014, 25: 635–641
28 Wang Z S, Liu L, Wu Y M, et al. Optimal fault-tolerant control for discrete-time nonlinear strict-feedback systems based on adaptive critic design. IEEE Trans Neural Netw Learn Syst, 2018, 29: 2179–2191
29 Wang M, Wang Z D, Chen Y, et al. Event-based adaptive neural tracking control for discrete-time stochastic nonlinear systems: a triggering-threshold compensation strategy. IEEE Trans Neural Netw Learn Syst, 2020, 31: 1968–1981
30 Kurdila A J, Narcowich F J, Ward J D. Persistency of excitation in identification using radial basis function approximants. SIAM J Control Optim, 1995, 33: 625–642
31 Wang C, Hill D J. Learning from neural control. IEEE Trans Neural Netw, 2006, 17: 130–146
32 Wang C, Wang M, Liu T F, et al. Learning from ISS-modular adaptive NN control of nonlinear strict-feedback systems. IEEE Trans Neural Netw Learn Syst, 2012, 23: 1539–1550

33 Wang M, Wang C. Learning from adaptive neural dynamic surface control of strict-feedback systems. IEEE Trans Neural Netw Learn Syst, 2015, 26: 1247–1259

34 Dai S L, Wang C, Wang M. Dynamic learning from adaptive neural network control of a class of nonaffine nonlinear systems. IEEE Trans Neural Netw Learn Syst, 2014, 25: 111–123

35 Wang M, Wang C, Shi P, et al. Dynamic learning from neural control for strict-feedback systems with guaranteed predefined performance. IEEE Trans Neural Netw Learn Syst, 2016, 27: 2564–2576

36 Abdelatti M, Yuan C Z, Zeng W, et al. Cooperative deterministic learning control for a group of homogeneous nonlinear uncertain robot manipulators. Sci China Inf Sci, 2018, 61: 112201

37 Wang M, Yang A L. Dynamic learning from adaptive neural control of robot manipulators with prescribed performance. IEEE Trans Syst Man Cybern Syst, 2017, 47: 2244–2255

38 Dai S L, Wang M, Wang C. Neural learning control of marine surface vessels with guaranteed transient tracking performance. IEEE Trans Ind Electron, 2016, 63: 1717–1727

39 He S D, Wang M, Dai S L, et al. Leader-follower formation control of USVs with prescribed performance and collision avoidance. IEEE Trans Ind Inf, 2019, 15: 572–581

40 Wang C, Chen T R, Liu T F. Deterministic learning and data-based modeling and control. Acta Autom Sin, 2009, 35: 693–706

41 Chen W S, Hua S Y, Ge S S. Consensus-based distributed cooperative learning control for a group of discrete-time nonlinear multi-agent systems using neural networks. Automatica, 2014, 50: 2254–2268

42 Zhang J T, Yuan C Z, Wang C, et al. Composite adaptive NN learning and control for discrete-time nonlinear uncertain systems in normal form. Neurocomputing, 2020, 390: 168–184

43 Fradkov A L, Evans R J. Control of chaos: methods and applications in engineering. Annu Rev Control, 2005, 29: 33–56

44 Dai S L, He S, Ma Y, et al. Distributed cooperative learning control of uncertain multiagent systems with prescribed performance and preserved connectivity. IEEE Trans Neural Netw Learn Syst, 2021, 32: 3217–3229

45 Dai S L, He S D, Lin H, et al. Platoon formation control with prescribed performance guarantees for USVs. IEEE Trans Ind Electron, 2018, 65: 4237–4246