Intelligent Adaptive Learning and Control for Discrete-Time Nonlinear Uncertain Systems In Multiple Environments

Jingting Zhang[a], Chengzhi Yuan*[a], Cong Wang[b], Wei Zeng[c], Shi-Lu Dai[d]

[a]*Department of Mechanical, Industrial and Systems Engineering*
University of Rhode Island, Kingston, RI 02881, USA
Email: jingting_zhang@uri.edu; cyuan@uri.edu
[b]*School of Control Science and Engineering*
Shandong University, JINAN 250061, China
Email: wangcong@sdu.edu.cn
[c]*School of Mechanical and Electrical Engineering*
Longyan University, Longyan 364012, China
Email: zw0597@126.com
[d]*School of Automation Science and Engineering*
South China University of Technology, Guangzhou 510641, China
Email: audaisl@scut.edu.cn

Abstract

This paper focuses on an adaptive learning and control problem for a class of discrete-time nonlinear uncertain systems operating under multiple environments. A novel intelligent learning control framework is proposed by using a combination of offline and online learning methods. Specifically, in the offline learning mode, a deterministic learning (DL) based adaptive dynamics learning approach is first proposed to achieve locally-accurate identification of associated nonlinear uncertain system dynamics under each anticipated individual environment, and the learned knowledge is obtained and stored in a set of constant radial basis function neural network models. Then, with the learned knowledge, an online adaptive learning control scheme is further developed, which consists of: (i) an online adaptive learning control mechanism composed by multiple experience-based controllers and a DL-based adaptive learning controller, aiming to provide desired control performance for the plant operating under each individual environment; and (ii) a learning-based recognition mechanism composed by multiple recognition estimators and a DL-based identifier, aiming to recognize the active environment and schedule

appropriate control strategies in real time. To guarantee the system stability during environment transition, a robust quasi-sliding-mode controller is further developed and embedded in the overall controller architecture. With this new intelligent adaptive learning control framework, the overall system is capable of adapting not only to any anticipated (pre-defined) environment by re-utilizing the knowledge obtained from both offline and online learning, but also to unanticipated (new) environments by actively acquiring new knowledge online. Simulation studies are conducted to verify the effectiveness and advantages of this new framework.

Keywords: Offline and online learning, deterministic learning, adaptive learning control, dynamic recognition, discrete-time nonlinear uncertain system, neural networks.

---

1. Introduction

Many aspects of modern life involve the use of intelligent machines, such as servant robots that assist humans in their daily activities and maintenance machines that replace human workforce to operate in hazardous environments [1, 2, 3, 4]. Such machines are expected to successfully perform various tasks that could involve complex system dynamics under varying operational environments. Over the past decades, considerable efforts have been devoted to developing advanced intelligent control schemes to enable autonomy of such machines. For instance, [6] developed an optimal critic learning based control scheme for enabling intelligent robots working under time-varying environments. In [7], another important problem of pattern classification for intelligent machines operating under nonstationary environments was investigated, and a new class of probabilistic neural networks was proposed for classifying patterns of time-varying probability distributions. Moreover, [5] discussed the application of intelligent control techniques for the humanoid robotic systems operating in unstructured environments. However, these existing methods have their own limitations, for example, they largely require the operation/control environments to vary slowly with time. For those cases when the environments are changing rapidly, the associated intelligent control design problem becomes rather challenging and has obtained few success in current literature.

When operating in rapidly time-varying environments, the system could encounter large uncertainties, e.g., system faults, external disturbances, and

changes in system parameters [8]. Adaptive control methods (e.g., [9, 10, 11, 12]), which are capable of accommodating various system uncertainties caused by the environment changes [9], provide promising techniques to overcome such issues. However, for those cases when the environment is switched from one context to another, traditional adaptive controllers may react too slowly to such abrupt changes, resulting in slow convergence and large transient tracking errors, or even instability of the overall system [8]. To overcome this issue, an advanced adaptive control approach, i.e., multiple model adaptive control (MMAC), was proposed in [8, 13, 15]. The non-stationary environments considered therein were represented by switching among a finite number of different stationary environments. Multiple adaptive estimators were developed to identify the active environment, and the identification result was further used to guide the switching among multiple adaptive controllers. However, such MMAC scheme did not well-explore the learning capability in both processes of adaptive identification and control, thereby lacking the real intelligent capabilities of knowledge acquisition and re-utilization [16]. As a result, even when a same/similar environment recurs, the processes of identification and control still need to repeat the online parameter tuning or adaptation, which are usually computationally expensive and time-consuming.

The learning capability, i.e., learning from the dynamical environment and using learned knowledge to improve control performance, is essential for modern intelligent control systems [17]. Recently, the deterministic learning (DL) theory proposed in [16, 18, 19] offers a new paradigm for the development of advanced intelligent adaptive learning control schemes that possess the real learning capability defined above. This theory has been demonstrated successful in solving not only adaptive learning control problems, but also many other intelligence-related problems, for example, pattern identification, representation, and recognition [14, 20, 21]. In particular, based on the DL theory, a so-called pattern based neural network (NN) learning control scheme was developed in [22, 23] for intelligent control of a class of continuous-time nonlinear uncertain systems operating under multiple environments. This approach requires all possible environments to be predefined for offline NN training, so as to obtain the related knowledge for the subsequent development of online recognition and control. However, in those cases when the system operates in an unanticipated/new environment that cannot be predefined for offline training purpose, the schemes of [22, 23] may not be applicable. Hence, to further enhance the system capability, it is necessary

to incorporate online learning approaches that could actively acquire new knowledge when encountering an unanticipated environment.

In this paper, we focus on the intelligent learning control problem for a class of discrete-time nonlinear uncertain systems operating in multiple environments, including anticipated/predefined environments and unanticipated/new environments. We seek to address several interrelated intelligent control issues, including: (i) how to provide desired tracking control performance for the system operating in each individual environment; (ii) how to guarantee the system stability when the operation environment is abruptly changed from one to another; and (iii) how to rapidly recognize in real time the active environment and thereby to schedule appropriate real-time control strategies. To this end, a novel intelligent learning control framework is proposed. One important feature of this new framework lies in the combinational use of offline and online learning. The former is used to acquire knowledge of associated nonlinear uncertain system dynamics under each anticipated environment, such that improved system performance can be achieved by using the resulting experience-based controllers/estimators; the latter is operating in real-time to acquire new knowledge of an unanticipated/new environment, so as to expand/enrich the knowledge library online. The proposed framework develops the human-like intelligence capabilities of knowledge acquisition and knowledge re-utilization, enabling that: (i) when the controlled plant is operating in an anticipated environment, desired performance of recognition and control can be achieved through learned knowledge re-utilization; and (ii) when the controlled plant is operating in an unanticipated environment, associated new knowledge can be acquired through active online learning. We stress that this current work is developed by significantly advancing the previous work [24], in which the multiple environments were all considered as anticipated environments, while our present work considers a more practical and challenging scenario with both anticipated and unanticipated environments.

The rest of this paper is organized as follows. Section 2 presents the problem statement and some preliminary results. Section 3 presents the adaptive learning control scheme for the system in each fixed individual environment. In Section 4, a quasi-sliding-mode control (QSMC) scheme is proposed for stabilizing the system during the environment transition. Section 5 addresses learning-based recognition of active environments. Section 6 summarizes the overall learning, recognition, and control architecture. Extensive simulation results are given in Section 7. Conclusions are drawn in Section 8.

Notation. $\mathbb{R}$, $\mathbb{R}_+$ and $\mathbb{N}_+$ denote, respectively, the set of real numbers, the set of positive real numbers and the set of positive integers; $\mathbb{R}^{m\times n}$ denotes the set of $m\times n$ real matrices; $\mathbb{R}^n$ denotes the set of $n\times 1$ real column vectors; $|\cdot|$ is the absolute value of a real number; $\|\cdot\|$ is the 2-norm of a vector or a matrix, i.e. $\|x\| = (x^Tx)^{\frac{1}{2}}$.

2. Preliminaries and Problem Formulation

2.1. Radial Basis Function Neural Networks (RBF NNs)

The RBF networks can be described by $f_{nn}(Z) = \sum_{i=1}^{N_n} w_i s_i(Z) = W^T S(Z)$ [25], where $Z \in \Omega_Z \subset \mathbb{R}^q$ is the input vector, $W = [w_1, \cdots, w_{N_n}]^T \in \mathbb{R}^{N_n}$ is the weight vector, N_n is the NN node number, and $S(Z) = [s_1(\|Z - \varsigma_1\|), \cdots, s_{N_n}(\|Z - \varsigma_{N_n}\|)]^T$, with $s_i(\cdot)$ being a radial basis function, and ς_i $(i = 1, 2, \cdots, N_n)$ being distinct points in state space. The Gaussian function $s_i(\|Z - \varsigma_i\|) = \exp[\frac{-(Z-\varsigma_i)^T(Z-\varsigma_i)}{\eta_i^2}]$ is one of the most commonly used radial basis functions, where $\varsigma_i = [\varsigma_{i1}, \varsigma_{i2}, \cdots, \varsigma_{iq}]^T$ is the center of the receptive field and η_i is the width of the receptive field. The Gaussian function belongs to the class of localized RBFs in the sense that $s_i(\|Z - \varsigma_i\|) \to 0$ as $\|Z\| \to \infty$. It is easily seen that $S(Z)$ is bounded and there exists a real constant $S_M \in \mathbb{R}_+$ such that $\|S(Z)\| \leqslant S_M$ [16]. As shown in [25, 26], for any continuous function $f(Z) : \Omega_Z \to \mathbb{R}$ where $\Omega_Z \subset \mathbb{R}^q$ is a compact set, and for the NN approximator, where the node number N_n is sufficiently large, there exists an ideal constant weight vector W^*, such that for any $\epsilon^* > 0$, $f(Z) = W^{*T}S(Z) + \epsilon$, $\forall Z \in \Omega_Z$, where $|\epsilon| < \epsilon^*$ is the ideal approximation error. The ideal weight vector W^* is an "artificial" quantity required for analysis, and is defined as the value of W that minimizes $|\epsilon|$ for all $Z \in \Omega_Z \subset \mathbb{R}^q$, i.e. $W^* \triangleq arg\min_{W\in\mathbb{R}^{N_n}}\{\sup_{Z\in\Omega_Z}|f(Z) - W^TS(Z)|\}$. Moreover, based on the localization property of RBF NNs [16], for any bounded trajectory $Z(k)$ within the compact set Ω_Z, $f(Z)$ can be approximated by using a finite number of neurons located in a local region along the trajectory: $f(Z) = W_\zeta^{*T}S_\zeta(Z) + \epsilon_\zeta$, where ϵ_ζ is the approximation error, with $\epsilon_\zeta = O(\epsilon) = O(\epsilon^*)$, $S_\zeta(Z) = [s_{j1}(Z), \cdots, s_{j\zeta}(Z)]^T \in \mathbb{R}^{N_\zeta}$, $W_\zeta^* = [w_{j1}^*, \cdots, w_{j\zeta}^*]^T \in \mathbb{R}^{N_\zeta}$, $N_\zeta < N_n$, and the integers $j_i = j_1, \cdots, j_\zeta$ are defined such that $|s_{j_i}(Z_p)| > \theta$ ($\theta > 0$ is a small positive constant) for some $Z_p \in Z(k)$.

Lemma 1 ([16]). *Consider any recurrent trajectory*[1] *$Z(k)$ that remains in a bounded compact set $\Omega_Z \subset \mathbb{R}^q$. For RBF network $W^T S(Z)$ with centers placed on a regular lattice (large enough to cover compact set Ω_Z), the regressor subvector $S_\zeta(Z)$ consisting of RBFs with centers located in a small neighborhood of $Z(k)$ satisfies the persistently exciting (PE) condition.*

2.2. Problem Formulation

Consider the following discrete-time nonlinear uncertain system:

$$\begin{cases} x_i(k+1) = x_{i+1}(k), \quad i = 1, 2, \cdots, n-1, \\ x_n(k+1) = f^j(x(k)) + u(k), \end{cases} \tag{1}$$

where $x = [x_1, \cdots, x_n]^T \in \mathbb{R}^n$ is the system state, $u \in \mathbb{R}$ is the system input, $f^j(x)$ is unknown nonlinear function with the superscript j ($j \in J = \{1, \cdots, N\}$, $N \in \mathbb{N}_+$) indicating different system dynamics varying under different environments. J is a finite set containing N number of different operation environments. We assume that J is composed by an anticipated environment set $J_1 \subseteq J$, i.e., those environments that can be pre-defined and the associated nonlinear uncertain system dynamics can be learned through offline training; and an unanticipated environment set $J_2 = J - J_1$, i.e., those environments that are not known in *a priori* and have to be dealt with through online learning. Only one environment will be active at each time instant, which needs to be recognized in real time for controller implementation.

Assumption 1. *The nominal model of system (1), denoted by $\bar{f}(x)$, is known and satisfies $\left|\bar{f}(x) - f^j(x)\right| < H$ for some known constant $H \in \mathbb{R}_+$ and for all $j \in J$.*

The system state x of (1) will be required to track over the reference signal x_d generated from the following model:

$$\begin{cases} x_{d_i}(k+1) = x_{d_{i+1}}(k), \quad i = 1, 2, \cdots, n-1, \\ x_{d_n}(k+1) = f_d(x_d(k)), \end{cases} \tag{2}$$

[1] A recurrent trajectory represents a large set of periodic and periodic-like trajectories generated from linear/nonlinear dynamics systems. A detailed characterization of recurrent trajectories can be found in [16].

where $x_d = [x_{d_1}, \cdots, x_{d_n}]^T \in \mathbb{R}^n$ is the reference state, $f_d(x_d)$ is a known nonlinear function.

Assumption 2. *All state signals of the reference model (2) are bounded and recurrent.*

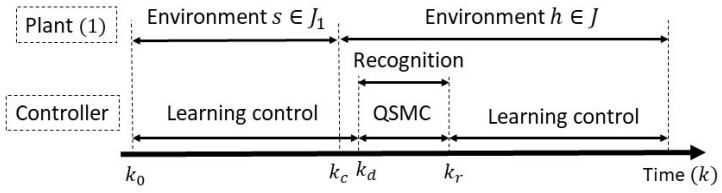


Figure 1: Illustration of the real-time control and recognition process under multiple environments $s \in J_1$, $h \in J$ $(s \neq h)$. k_0 is the initial time; k_c is the occurrence time of environment switching; k_d is the detection time of environment switching; k_r is the recognition time of active environment h.

In this paper, our objective is to develop a systematic and holistic framework for intelligent adaptive learning control of system (1) such that its state will track over the desired reference states of (2). We seek to answer three specific questions: (i) how to deal with the system uncertainty $f^j(x)$ of (1); (ii) how to ensure overall stability when the environment is abruptly changed from one to another; and (iii) how to rapidly recognize active environment to facilitate scheduling of proper control strategies online. The overall process of real-time system operation and control under consideration is illustrated in Fig. 1. Specifically, as shown in the upper part of Fig. 1, we assume that the controlled plant (1) is initially operating under a known anticipated environment, say $s \in J_1$, and at an unknown time k_c, the active environment is abruptly switched to an unknown environment, say $h \in J$ $(h \neq s)$. The new active environment could be anticipated (i.e., $h \in J_1$) or unanticipated (i.e., $h \in J_2$). The proposed intelligent adaptive learning controller (as shown in the lower part of Fig. 1) is expected to be capable of: (i) ensuring stable tracking control under any fixed individual environment (e.g., environment s during $k_0 \leq k \leq k_c$ and environment h during $k > k_r$ in Fig. 1), which will be addressed in Section 3 using a combination of offline and online adaptive NN learning approaches; (ii) stabilizing the overall system during the transition period when the operation environment is switching from environment s to h (e.g., the time interval $k_d < k \leq k_r$ in Fig. 1), which will be addressed in

Section 4 using a robust QSMC approach; and (iii) rapidly recognizing active environment h (e.g., during the time interval $k_d < k \leq k_r$ in Fig. 1) for efficient scheduling of appropriate control strategies, which will be addressed in Section 5 using a dynamic recognition approach combined with an online adaptive learning based estimation method.

3. Learning Control Under Individual Environment

In this section, to fulfill the first objective mentioned above, a novel adaptive learning control scheme will be proposed, which consists of an offline learning mode and an online learning control mode.

3.1. Offline Learning Mode

In the offline learning/training mode, we assume that the controlled plant (1) is operating under any fixed individual anticipated environment. A DL-based dynamics learning control approach will be developed to achieve stable tracking control and accurate learning of the associated nonlinear uncertain system dynamics, such that the learned knowledge can be obtained and stored in constant RBF NN models for later use.

Specifically, consider the plant (1) operating in any fixed individual anticipated environment $j \in J_1$, for the unknown dynamics $f^j(x)$, from Section 2.1, we know that there exists an ideal constant NN weight $W^{j*} \in \mathbb{R}^{N_n}$ (with N_n denoting the number of NN nodes) such that:

$$f^j(x) = W^{j*T} S(x) + \epsilon^j, \tag{3}$$

where $S(x) \in \mathbb{R}^{N_n}$ is a smooth RBF vector; ϵ^j is the ideal approximation error satisfying $|\epsilon^j| < \epsilon^*$, with ϵ^* being a positive constant that can be made arbitrarily small by constructing sufficiently large number of neurons. Based on this, consider the plant (1) and the reference model (2), we propose to design an adaptive dynamics learning controller using RBF NNs as follows:

$$\begin{aligned} \hat{u}^j(k) &= f_d(x_d(k)) - \hat{W}^{jT}(k) S(x(k)) + a_c^j r(k) - \lambda_1 z_n(k) - \cdots - \lambda_{n-1} z_2(k), \\ \hat{W}^j(k+1) &= \hat{W}^j(k) + c_c^j r(k+1) S(x(k)), \end{aligned} \tag{4}$$

where $\hat{u}^j$ is the control signal, $\hat{W}^j$ is the estimate of the constant weight W^{j*} in (3), $f_d(x_d)$ is from (2), a_c^j, c_c^j are design parameters, $\lambda_1, \cdots, \lambda_{n-1}$ are design constants such that $z^{n-1} + \lambda_1 z^{n-2} + \cdots + \lambda_{n-1}$ is a Schur polynomial,

$z_i = x_i - x_{d_i}$ $(i = 1, \cdots, n)$ are state tracking errors, r is filtered tracking error defined as:

$$r = z_n + \lambda_1 z_{n-1} + \cdots + \lambda_{n-1} z_1. \tag{5}$$

Theorem 1. *Consider the closed-loop system consisting of the plant (1) with any fixed $j \in J_1$, the reference model (2), and the controller (4). Under Assumption 2, given any initial conditions $x(0) \in \Omega_0$ (where Ω_0 is a compact set) and $\hat{W}^j(0) = 0$, if the controller coefficients satisfy $0 < c_c^j \leq \frac{1}{3}$, $0 < a_c^j < \sqrt{(c_c^j)^2 + 1} - c_c^j$, then, it is guaranteed that: (i) all signals in the closed-loop system remain uniformly ultimately bounded (UUB); (ii) there exists a finite time K_c such that for all $k > K_c$, the state tracking error $x(k) - x_d(k)$ converges to a small neighborhood around the origin; and (iii) a locally accurate approximation of $f^j(x)$ in (1) is obtained by $\hat{W}^{jT} S(x)$ as well as $\bar{W}^{jT} S(x)$ along the NN input orbit Z, where $\bar{W}^j := \frac{1}{K_2 - K_1 + 1} \sum_{k=K_1}^{K_2} \hat{W}^j(k)$ with $K_2 > K_1 > K_c$ being a time segment after the transient process.*

The proof of Theorem 1 can be readily completed by following a similar line of that of [29, Th. 1], which will be omitted here.

Through the above training process, the knowledge of the dynamics in $f^j(x)$ of (1) can be obtained and stored in the constant RBF NN model $\bar{W}^{jT} S(x)$ for each $j \in J_1$, i.e.,

$$f^j(x) = \bar{W}^{jT} S(x) + \epsilon_1^j, \tag{6}$$

where $\epsilon_1^j = O(\epsilon^*)$ is the approximation error. Moreover, according to [19, 22, 18], the accurate approximation in (6) can be achieved in a local region Ω_c^j $(j \in J_1)$ along the trajectory x_d, with

$$\Omega_c^j := \{x \,|\, \text{dist}(x, x_d) < d_c^* \Rightarrow |\bar{W}^{jT} S(x) - f^j(x)| < \varepsilon_c^{j*}\}, \tag{7}$$

where the constant d_c^* characterizes the size of NN approximation region, and ε_c^{j*} is the maximum NN approximation error within Ω_c^j, which can be made arbitrarily small by constructing a sufficiently large number of neurons in the training process.

3.2. Online Learning Control Mode

In the online learning control mode, the active environment, say $h \in J$, could be either anticipated, i.e., $h \in J_1$, or unanticipated, i.e., $h \in J_2$. Our objective in this section is to develop an efficient learning control scheme

such that (i) when the active environment h is anticipated, the desired stable tracking control performance can be achieved by re-utilizing the related knowledge learned from the previous subsection; and (ii) when the active environment h is unanticipated, in addition to achieving stable tracking control, the associated new knowledge can also be learned accurately in real-time.

To this end, we first consider the case that the plant (1) is operating in an anticipated environment with $h \in J_1$. By utilizing the learned knowledge from the offline training mode in Section 3.1, a family of the so-called experience-based controllers (EBCs) are designed as follows:

$$u^j = f_d(x_d) - \bar{W}^{jT} S(x) + b_c r - \lambda_1 z_n - \cdots - \lambda_{n-1} z_2, \tag{8}$$

for all $j \in J_1$, where j denotes the corresponding environment index, $\bar{W}^{jT} S(x)$ is the constant NN model given in (6), which is used as an accurate approximation of $f^j(x)$ in (1), $0 \leq b_c < 1$ is a design parameter. When the plant (1) is operating under the $h \in J_1$ environment, the associated h-th controller from the above EBC family will be activated accordingly for stable tracking control.

Theorem 2. *Consider the closed-loop system consisting of the plant (1) with a fixed $h \in J_1$, the reference model (2), the h-th EBC of (8). Under Assumption 2, given the same recurrent reference orbits x_d that are experienced in the offline learning mode of Section 3.1, and with initial condition $x(0)$ in a close vicinity of $x_d(0)$, we have that: (i) all signals in the closed-loop system remain UUB; (ii) the state tracking error $x - x_d$ converges to a small neighborhood around the origin.*

The proof of Theorem 1 can be readily completed by following a similar line of that of [29, Th. 2], which will be omitted here.

We then proceed to consider the case that the plant (1) is operating in an unanticipated environment with $h \in J_2$. In this case, since no corresponding EBC is available from the offline training mode, an online adaptive learning controller is needed to address the uncertainty issues for stable tracking control. To this end, an adaptive learning controller (ALC) is designed as follows:

$$\begin{aligned} \hat{u}(k) &= f_d(x_d(k)) - \hat{W}^T(k) S(x(k)) + a_c r(k) - \lambda_1 z_n(k) - \cdots - \lambda_{n-1} z_2(k), \\ \hat{W}(k+1) &= \hat{W}(k) + c_c r(k+1) S(x(k)), \end{aligned} \tag{9}$$

where $\hat{u}$ is the control signal, $S(x)$ is an RBF network, $\hat{W}$ is an associated NN weight vector, r is the filtered tracking error defined in (5), and a_c, c_c are two design parameters satisfying $0 < c_c \leq \frac{1}{3}$, $0 < a_c < \sqrt{c_c^2 + 1} - c_c$, respectively. With such a controller, the resulting closed-loop stability and state tracking performance can be verified by following a similar line of Theorem 1. Specifically, for the plant (1), the controller (9) will render not only stable tracking control, but also locally accurate learning of associated uncertain dynamics $f^h(x)$ of (1). As a result, the related knowledge will be obtained and stored in a constant RBF model $\bar{W}^{hT}S(x)$ in the form of (6). Then, a new EBC can be further constructed as:

$$u^h = f_d(x_d) - \bar{W}^{hT}S(x) + b_c r - \lambda_1 z_n - \cdots - \lambda_{n-1} z_2, \tag{10}$$

which will be added as a new member to the family of EBCs as described in (8).

4. Quasi-Sliding-Mode Control for Stable Environment Transition

In the previous section, the resulting family of EBCs have one-to-one correspondence with the anticipated environments, i.e., each individual EBC is guaranteed stabilizing only when the plant is steadily operating under the matched environment. This means that the EBCs might not guarantee the system's overall stability when the active environment is abruptly transitioning from one to another. To address this issue, we propose to develop a robust quasi-sliding-mode controller (QSMC) to stabilize the system during such a transition process. Specifically, corresponding to Fig. 1, we formulate the plant dynamics during the transition process (i.e., for $k_c < k \leq k_r$ as shown in Fig. 1) in the following form:

$$\begin{cases} x_i(k+1) = x_{i+1}(k), \quad i = 1, 2, \cdots, n-1 \\ x_n(k+1) = \begin{cases} f^s(x(k)) + u^s(k), & k_0 \leq k \leq k_c \\ f^h(x(k)) + u^s(k), & k_c < k \leq k_d, \\ f^h(x(k)) + u^0(k), & k_d < k \leq k_r \end{cases} \end{cases} \tag{11}$$

where k_0, k_c, k_d and k_r are all defined in Fig. 1. In particular, k_d denotes also the activation time of QSMC. $s \in J_1$ and $h \in J$ $(s \neq h)$ denote respectively the pre-switching and post-switching active environment indexes, u^s represents the s-th EBC from (8), and u^0 represents the QSMC to be developed.

Since the environment switching time k_c is unknown, prior to the design of the QSMC, we will need to first design a detection mechanism to rapidly detect occurrence of environment switching and thereby to determine the activation time k_d of the QSMC. For this purpose, we first consider the time interval $k_0 \leq k \leq k_c$ in (11), during which the operating controller, i.e., the s-th EBC, matches the active environment s. From (2), (7), and (8), the filtered tracking error r of (5) can be readily proved to satisfy:

$$\begin{aligned} |r(k)| &= \left|b_c r(k-1) + f^s(x(k-1)) - \bar{W}^{sT} S(x(k-1))\right| \\ &< b_c \left|r(k-1)\right| + \varepsilon_c^{s*} < b_c^{k-k_0} \left|r(k_0)\right| + \frac{1-b_c^{k-k_0}}{1-b_c}\varepsilon_c^{s*}. \end{aligned} \tag{12}$$

Note that $0 < b_c < 1$ and $r(k_0)$ has a known small value, we use $\bar{\mu}_c$ to denote such a constant number that $\bar{\mu}_c \geq b_c^{k-k_0} r(k_0)$. Then, we have:

$$|r(k)| < \bar{\mu}_c + \frac{1}{1-b_c}\varepsilon_c^{s*}, \tag{13}$$

Based on this, we can design a threshold as:

$$\bar{e}_c := \bar{\mu}_c + \frac{1}{1-b_c}\varepsilon_{c_m}^{*}, \tag{14}$$

where $\varepsilon_{c_m}^{*} := \max_{j \in J_1} \{\varepsilon_c^{j*}\}$, in which ε_c^{j*} is the NN approximation error given in (7). The detection time k_d can be deduced by evaluating the filtered tracking error $|r|$ of (5) against the threshold $\bar{e}_c$ of (14). The rationale behind this is as follows. The threshold $\bar{e}_c$ of (14) can be considered as a maximum tolerable tracking error under the proposed EBCs of (8). Specifically, for the system (11), as long as the real-time filtered tracking error $|r|$ of (5) maintains smaller than the designed threshold $\bar{e}_c$ of (14), the active EBC is considered to be able to guarantee desired tracking control performance under the active environment. Whenever $|r|$ becomes larger than $\bar{e}_c$, it indicates that the active environment is changed and the operating EBC is no longer competent/matched.

Remark 1. *Note that the threshold of (14) is designed with the parameters ε_c^{j*} ($\forall j \in J_1$), which are related to the EBCs of (8). It requires that whenever constructing a new EBC (e.g., the h-th EBC of (10) from the online learning mode in Section 3.2), the corresponding parameter (i.e., ε_c^{h*}) should be obtained simultaneously to update the design of the threshold (14). According*

to (7), this parameter ε_c^{h*} can be computed online as the upper bound of the absolute steady approximation error between $\bar{W}^{hT}S(x(k-1))$ and the system signal $f^h(x(k-1)) = x_n(k) - u^h(k-1)$, according to (7).

Once the switching of the active environment in (11) is detected (at time k_d), a robust QSMC will be activated immediately to stabilize the system (11). The design of such a QSMC is as follows. We first design an s function:

$$s(k) = s(k-2) + \alpha_0 z_n(k) + \cdots + \alpha_{n-1} z_1(k), \tag{15}$$

where $z_i = x_i - x_{d_i}$ $(i = 1, \cdots, n)$ are state tracking errors, the parameters $\alpha_0, \cdots, \alpha_{n-1}$ are selected such that $\alpha_0 z_n + \cdots + \alpha_{n-1} z_1$ is a Schur polynomial. The QSMC is constructed as:

$$\begin{aligned} u^0(k) = f_d(x_d(k)) - \bar{f}(x(k)) - \frac{1}{\alpha_0}(\alpha_1 z_n(k) + \cdots + \alpha_{n-1} z_2(k) \\ + s(k-1) + \rho \operatorname{sgn}(s(k))), \end{aligned} \tag{16}$$

where $\bar{f}(x)$ is known from Assumption 1, $f_d(x_d)$ is from (2), ρ is a design constant.

Theorem 3. *Consider the plant (11), the reference model (2), and the QSMC (16). Under Assumption 1, if the controller coefficient ρ satisfies $\rho > \alpha_0 H$ with H given in Assumption 1, then, it is guaranteed that: (i) the closed-loop system is stable; and (ii) the state tracking error $x - x_d$ converges to a close vicinity of the origin.*

Proof. With the closed-loop system consisting of (2), (11) and (16), the function $s(k)$ of (15) will be governed by the following dynamics:

$$s(k+1) = \alpha_0(f^h(x(k)) - \bar{f}(x(k))) - \rho \operatorname{sgn}(s(k)). \tag{17}$$

Note that $\left|f^h(x) - \bar{f}(x)\right| < H$ under Assumption 1, following a similar line of the analysis in [28, Sec. III], if the design parameter ρ satisfies $\rho > \alpha_0 H$, it can be guaranteed that:

$$\begin{aligned} & s(k)(s(k+1) - s(k)) < 0, \\ & \operatorname{sgn}(s(k+1)) = -\operatorname{sgn}(s(k)) \Rightarrow \begin{cases} \operatorname{sgn}(s(k+2)) = \operatorname{sgn}(s(k)), \\ |s(k+1)| \leq \alpha_0 H + \rho. \end{cases} \end{aligned}$$

According to [28, Sec. II.B.], the closed-loop system can be guaranteed to be stable and the state tracking error $x - x_d$ converges to a close vicinity of the origin. □

Remark 2. *The QSMC of (16) is capable of guaranteeing global stability of the overall control system. Note that the EBC of (10) and ALC of (9) proposed in the previous section can provide stable control only when the associated system trajectory lies in the local region Ω_c^j of (7). When the system trajectory is not belonging to the region Ω_c^j, e.g., during the transient process of environments switching, activating the QSMC can guarantee the system's overall stability, and will drive the system trajectory back into the region Ω_c^j.*

5. Learning-Based Recognition of Active Environment

As illustrated in Fig. 1, once the QSMC of (16) is activated at time k_d, an online recognition mechanism will be activated at the same time to identify the index of the new active environment, such that the corresponding EBC/ALC from Section 3.2 can be subsequently activated to replace the QSMC for stabilizing the overall tracking control system. In this section, we will specify how such an online recognition mechanism can be designed, which involves an offline training mode and an online recognition mode.

5.1. Offline Training Mode

In the offline training mode, we will first assume that the plant is operating under any fixed anticipated environment with the QSMC, whose dynamics is governed by:

$$\begin{cases} x_i(k+1) = x_{i+1}(k), \quad i = 1, 2, \cdots, n-1 \\ x_n(k+1) = f^j(x(k)) + u^0(k), \end{cases} \tag{18}$$

for any fixed $j \in J_1$, where u^0 is the control signal generated from the QSMC in (16). Our objective is to develop a DL-based dynamics learning approach to achieve locally-accurate identification of the unknown nonlinear dynamics $f^j(x) + u^0$ in (18), such that the obtained knowledge can be stored and re-utilized for online recognition as will be discussed in the next subsection.

To this end, according to Section 2.1, there exists an ideal constant NN weight $W_r^{j*} \in \mathbb{R}^{N_n}$ (with N_n denoting the number of NN nodes) such that:

$$f^j(x) + u^0 = W_r^{j*T} S_r(x, u^0) + \epsilon_r^j \tag{19}$$

with $S_r(x, u^0)$ being a smooth RBF NN vector, (x, u^0) being the system trajectory of (18), ϵ_r^j being an ideal approximation error satisfying $|\epsilon_r^j| <$

ϵ_r^*, and ϵ_r^* being a positive constant that can be made arbitrarily small by constructing sufficiently large number of neurons. Based on this, we propose to design an adaptive identifier as:

$$\begin{aligned} \hat{x}_r^j(k+1) &= a_r^j(\hat{x}_r^j(k) - x_n(k)) + \hat{W}_r^{jT}(k)S_r(x(k), u^0(k)), \\ \hat{W}_r^j(k+1) &= \hat{W}_r^j(k) - c_r^j\tilde{x}_r^j(k+1)S_r(x(k), u^0(k)), \end{aligned} \tag{20}$$

where $\tilde{x}_r^j = \hat{x}_r^j - x_n$, x_n is the n-th state of (18), a_r^j, c_r^j are design parameters, and $\hat{W}_r^j$ is the estimate of W_r^{j*} in (19).

Theorem 4. *Consider the adaptive learning system consisting of the plant (18) with a fixed $j \in J_1$, and the identifier (20). Under Assumption 2, with initial conditions $(x(0), u^0(0)) \in \Omega_{r_0}$ (where Ω_{r_0} is a compact set) and $\hat{W}_r^j(0) = 0$, if the associated coefficients in (20) satisfy $0 < a_r^j < \frac{\sqrt{5}-1}{2}$ and $0 < c_r^j < \frac{1}{S_{r_M}^2(2+a_r^j)}$ with S_{r_M} being the upper bound of $\|S_r(x, u^0)\|$, then, we have: (i) all signals in the system remain UUB; (ii) there exists a finite time K_r such that for all $k > K_r$, the estimation error $\hat{x}_r^j(k) - x_n(k)$ converges to a small neighborhood around the origin; and (iii) a locally accurate approximation of the nonlinear uncertain system dynamics $f^j(x) + u^0$ can be obtained by $\hat{W}_r^{jT}S_r(x, u^0)$ as well as $\bar{W}_r^{jT}S_r(x, u^0)$ along the recurrent NN input orbit (x, u^0), where $\bar{W}_r^j = \frac{1}{K_2 - K_1 + 1}\sum_{k=K_1}^{K_2}\hat{W}_r^j(k)$ and $K_2 > K_1 > K_r$ represents a time segment after the transient process.*

Detailed proof can be completed by following the same process of the proof of [20, Th. 1], which is omitted here.

Through the offline training process, the nonlinear uncertain system dynamics $f^j(x) + u^0$ in (18) can ultimately be approximated and represented by the constant RBF NN model $\bar{W}_r^{jT}S_r(x, u^0)$ for each $j \in J_1$, i.e.,

$$f^j(x) + u^0 = \bar{W}_r^{jT}S_r(x, u^0) + \epsilon_{r_1}^j, \tag{21}$$

where $\epsilon_{r_1}^j = O(\epsilon_r^*)$ is the approximation error. Furthermore, according to [20], accurate approximation of $f^j(x) + u^0$ in (21) remains valid in a local region Ω_r^j along the experienced recurrent trajectory (denoted by φ_r^j) generated from system (18), where Ω_r^j can be characterized as:

$$\Omega_r^j := \{(x, u) \,|\, \mathrm{dist}((x, u), \varphi_r^j) < d_r^* \Rightarrow |f^j(x) + u^0 - \bar{W}_r^{jT}S_r(x, u)| < \varepsilon_r^{j*}\}, \tag{22}$$

where d_r^* characterizes the size of NN approximation region; ε_r^{j*} is the NN estimation error within Ω_r^j, which can be made arbitrarily small by constructing a sufficiently large number of neurons in the training process.

5.2. Online Recognition Mode

In the online recognition mode, a learning-based recognition scheme will be developed to achieve rapid recognition of the active environment in (11), and to determine the time k_r for activating the corresponding EBC/ALC. Such a scheme will be operating for time $k_d < k \leq k_r$ as shown in Fig. 1, i.e., after the active controller is switched to the QSMC of (16) and before switching back to a matched EBC/ALC. Note that the active environment to be recognized could be either anticipated or unanticipated. To ensure successful recognition under the proposed scheme, the following assumption is needed, which imposes sufficient differences among the system dynamics under different environments.

Assumption 3. *The difference between the system dynamics induced by different environments for each pair of* $h, \bar{h} \in J$ $(h \neq \bar{h})$*, denoted as* $\varrho^{h,\bar{h}}(x, u^0) := (f^{\bar{h}}(x) + u^0) - (f^h(x) + u^0)$*, satisfies* $|\varrho^{h,\bar{h}}(x, u^0)| > \bar{\varrho}$*, where* $\bar{\varrho} > 2\varepsilon_r^*$ *is a known constant with* $\varepsilon_r^* := \max_{j \in J_1}\{\varepsilon_r^{j*}\}$ *and* ε_r^{j*} *being the NN approximation error of (22).*

We first consider the case when system (11) is operating in an anticipated environment (with index $h \in J_1$) for $k_d < k \leq k_r$. Resorting to the learned knowledge from the offline training mode in Section 5.1, we can design a bank of recognition estimators as follows:

$$e_r^j(k) = b_r e_r^j(k-1) + \left|\bar{W}_r^{jT} S(x(k-1), u^0(k-1)) - x_n(k)\right|, \qquad (23)$$

where $e_r^j(k)$ $(\forall j \in J_1)$ is called a recognition residual signal with $e_r^j(k_d) = 0$; $0 < b_r < 1$ is a design parameter; (x, u^0) is the system signals of (11); x_n is the n-th state of system (11); $\bar{W}_r^{jT} S_r(x, u^0)$ is the constant RBF NN model obtained from the offline training mode (as specified in (21)) to locally accurately approximate $f^j(x) + u^0$ of (18). The residual signal e_r^j can be used for real-time recognition decision making, the details are given below.

With the recognition estimators (23), motivated by the methodology from [27], we propose to develop an adaptive threshold based online recognition scheme. Specifically, an adaptive threshold (denoted as $\bar{e}_r$) is designed as follows, such that it will upper bound the residual signal e_r^j of (23) in real time whenever the corresponding estimator index $j \in J_1$ matches the active environment index h of (11):

$$\bar{e}_r(k) = \frac{(1 - b_r^{k-k_d}) q_r \varepsilon_r^*}{1 - b_r}, \qquad (24)$$

where b_r is the design parameter from (23), $1 < q_r \leq \frac{\bar{\varrho}-\varepsilon_r^*}{\varepsilon_r^*}$ is a tunable auxiliary parameter, $\varepsilon_r^* = \max_{j\in J_1}\{\varepsilon_r^{j*}\}$, and $\bar{\varrho}$ is a known constant defined in Assumption 3.

The key idea of achieving rapid active environment recognition based on (23) and (24) is that: under Assumption 3, the residual signal generated from one and only one of the recognition estimators of (23), say the h-th one, will ultimately become smaller than the threshold $\bar{e}_r$ of (24), while all other residual signals remain larger than the threshold, then the active environment undergone by system (11) can be readily recognized as the h-th environment.

Theorem 5. *Consider the system (11), the online recognition estimators (23) and the adaptive threshold (24). Under Assumption 3, we have: (i) there exists a unique $h \in J_1$ and a finite time $k_r > k_d$ such that $e_r^h(k) < \bar{e}_r(k)$ holds for $k \geq k_r$, and it can be recognized at time k_r that the system (11) is operating in the h-th anticipated environment; and (ii) if $\forall j \in J_1$, $e_r^j(k) \geq \bar{e}_r(k)$ holds for all $k > k_d$, then it can be deduced that the system (11) is operating in an unanticipated environment.*

Proof. When the active environment h of (11) is anticipated, i.e., $h \in J_1$, we first consider the matched recognition estimator, i.e., the h-th estimator of (23). In the recognition process, according to (22), its embedded constant RBF NN model $\bar{W}_r^{hT}S(x,u^0)$ will quickly recall the stored knowledge to provide accurate approximation of the uncertain dynamics $x_n(k) = f^h(x(k-1))+u^0(k-1)$ of (11), such that: $|\bar{W}_r^{hT}S(x(k-1),u^0(k-1))-x_n(k)| < \varepsilon_r^{h*} < \varepsilon_r^*$. Based on this, following a similar line of the proof of [27, Th. 1], it can be deduced that there exists a finite time $k_r > k_d$ such that the residual signal generated from the h-th estimator satisfies $e_r^h(k) < \frac{(1-b_r^{k-k_d})q_r\varepsilon_r^*}{1-b_r} = \bar{e}_r(k)$ for all $k > k_r$. On the other hand, we consider all the other mismatched recognition estimators of (23), e.g., the $\bar{h}$-th estimator with $\forall \bar{h} \in J_1$ $(\bar{h} \neq h)$. Under Assumption 3, we have: $\sum_{i=k_d}^{k-1} b_r^{k-1-k_d}|\varrho^{h,\bar{h}}(x(i),u^0(i))| > \frac{(1-b_r^{k-k_d})\bar{\varrho}}{1-b_r} \geq \frac{(1-b_r^{k-k_d})(q_r+1)\varepsilon_r^*}{1-b_r}$, $\forall k > k_d$, where $\bar{\varrho} \geq (q_r+1)\varepsilon_r^*$ is guaranteed by the constraint on the design parameter q_r, i.e., $1 < q_r \leq \frac{\bar{\varrho}-\varepsilon_r^*}{\varepsilon_r^*}$. By following the similar method of the proof of [27, Th. 1], the associated residual signal $e_r^{\bar{h}}(k)$ can be proved to satisfy $e_r^{\bar{h}}(k) \geq \frac{(1-b_r^{k-k_d})q_r\varepsilon_r^*}{1-b_r} = \bar{e}_r(k)$ for all $k > k_d$. Consequently, we obtain: $e_r^h(k) < \bar{e}_r(k)$ and $e_r^{\bar{h}}(k) \geq \bar{e}_r(k)$ for all $k \geq k_r$, indicating that the active environment of (11) can be identified as the h-th anticipated environment at time k_r.

When the active environment is unanticipated, i.e., $h \in J_2$, according to the above analysis, no corresponding recognition residual signal of (23) will become smaller than the threshold $\bar{e}_r$ of (24). In other words, all residual signals $e_r^j(k)$ $(\forall j \in J_1)$ will satisfy $e_r^j(k) \geq \bar{e}_r(k)$ for all $k > k_d$. In such cases, the active environment of system (11) will be identified as an unanticipated environment. This ends the proof. □

From Theorem 5, an unanticipated environment can be recognized using the online recognition scheme proposed above when all residuals of (23) maintain no smaller than the threshold of (24). In this case, we will further develop an online adaptive identifier to enable accurate learning of the associated nonlinear uncertain system dynamics under an unanticipated environment, such that the learned knowledge can be reused to construct a new recognition estimator in the form of (23), which will be added as a new member to the estimator family for later online recognition use. To this end, an online adaptive identifier is designed as follows:

$$\begin{aligned} \hat{x}_r(k+1) &= a_r(\hat{x}_r(k) - x_n(k)) + \hat{W}_r^T(k) S_r(x(k), u^0(k)), \\ \hat{W}_r(k+1) &= \hat{W}_r(k) - c_r \tilde{x}_r(k+1) S_r(x(k), u^0(k)), \end{aligned} \tag{25}$$

where $\tilde{x}_r = \hat{x}_r - x_n$, x_n is the n-th state of system (11), (x, u^0) is the system signal generated from (11), a_r and c_r are two design parameters satisfying $0 < a_r < \frac{\sqrt{5}-1}{2}$ and $0 < c_r < \frac{1}{S_{r_M}^2(2+a_r)}$, respectively, with S_{r_M} being the upper bound of $\|S_r(x,u)\|$. The identifier (25), which operates in parallel with the recognition systems (23)–(24), is used to online learn the dynamics $f^h(x) + u^0$ of system (11) during the recognition process for $k_d < k \leq k_r$ (as shown in Fig. 1). According to Theorem 4, with (25), the learned knowledge will finally be obtained and stored in a new constant RBF NN model $\bar{W}_r^{hT} S(x, u^0)$. As a result, a new recognition estimator can be constructed as:

$$e_r^h(k) = b_r e_r^h(k-1) + |\bar{W}_r^{hT} S(x(k-1), u^0(k-1)) - x_n(k)|, \tag{26}$$

which is added as a new member into (23) to enrich the recognition estimator family.

Remark 3. *When the new recognition estimator of (26) is constructed and added into (23), the recognition threshold of (24) needs to be re-designed. This is because the threshold of (24) is designed with the parameter* $\varepsilon_r^* = \max_{j \in J_1}\{\varepsilon_r^{j*}\}$, *in which the set* $\{\varepsilon_r^{j*}, \forall j \in J_1\}$ *is related to the recognition*

estimators of (23). Thus, when a new estimator of (26) is constructed, a new parameter ε_r^{h*} needs to be obtained and added correspondingly into such a parameter set. Note that ε_r^{h*} can be computed online as the upper bound of absolute steady error between $\bar{W}_r^{hT} S(x(k-1), u^0(k-1))$ of (26) and the real-time system signal $x_n(k) = f^h(x(k-1)) + u^0(k-1)$ of (11), according to (22).

Remark 4. *In online recognition mode, it is impossible to have the identifier of (25) operate only after the active environment h of (11) is recognized unanticipated. It requires us to keep activating the identifier of (25) whenever the recognition process is ongoing. This will result in the following two situations: (i) when the active environment is finally recognized anticipated at time k_r, the identifier of (25) will be deactivated and reset immediately; and (ii) when the active environment is recognized unanticipated, the identifier will keep operating until the learning process is completed, which also yields the recognition time k_r. An example will be provided in the simulation section for illustration.*

Remark 5. *Rapid recognition for anticipated environments $j \in J_1$ can be achieved with our recognition scheme. Specifically, note that the recognition estimators of (23) are designed with the constant models $\bar{W}_r^{jT} S(x, u^0)$, which represent the associated knowledge of each anticipated environment $j \in J_1$ that is obtained through the offline learning mode of Section 5.1. The operations of such estimators do not need any parameter adaptation, which help significantly shorten the recognition time, such that the recognition for anticipated environments can be achieved in a rapid manner. This would facilitate our control scheme to schedule proper control actions under rapidly-varying environments.*

Remark 6. *Achieving accurate and rapid recognition for an unanticipated environment $j \in J_2$ is a quite challenging problem. One critical difficulty lies in that unanticipated environments cannot be pre-defined and their knowledge cannot be obtained through the offline learning process. To overcome this issue, our recognition scheme is developed with a combination of the adaptive-threshold-based recognition system of (23)–(24) and an adaptive identifier of (25). Specifically, the recognition system of (23)–(24) is capable of distinguishing the unanticipated environment from other anticipated environments; and the identifier (25) can online actively acquire the knowledge of the active unanticipated environment. With the learned knowledge, whenever a*

same/similar unanticipated environment recurs in the future, accurate and rapid recognition can be achieved.

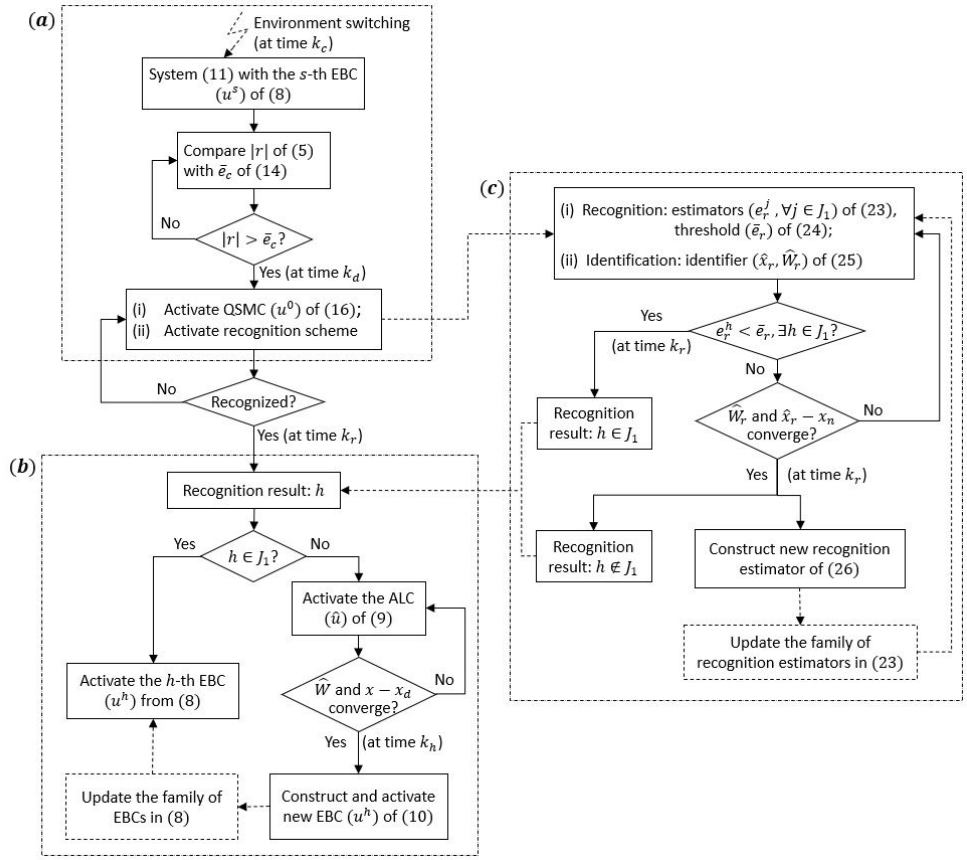


Figure 2: Schematic of the overall intelligent learning, recognition, and control architecture. (a) Quasi-sliding-mode control during environments transition (Section 4); (b) online learning control under environment $h \in J$ (Section 3.2); and (c) online recognition of active environment $h \in J$ (Section 5.2).

6. The Overall Architecture

In this section, we will present the overall architecture of the proposed intelligent adaptive learning control framework by summarizing all the components developed from the previous sections. This summary will consider two cases: (i) when the active environment is switched to an anticipated

environment; and (ii) when the active environment is switched to an unanticipated environment.

6.1. Anticipated Environment Case

In the following, we consider the case that the new active environment is an anticipated environment, i.e., $h \in J_1$ in Fig. 1. The overall processes of control and recognition are illustrated in Fig. 2. Specifically, when the environment switching occurs (at time k_c), the real-time filtered tracking error $|r|$ of (5) will start increasing; when it becomes larger than the detection threshold $\bar{e}_c$ of (14) (at time k_d), detection of environment switching can be achieved. Meanwhile, the QSMC of (16) will be activated for stabilizing the overall tracking control system, and the online recognition scheme of (23)–(25) is activated at the same time k_d for recognizing the active environment index h. Accurate recognition (see Fig. 2c) will be achieved (at time k_r) when the matched residual signal e_r^h of (23) becomes smaller than the recognition threshold $\bar{e}_r$ of (24). Based on this, the corresponding h-th EBC from (8) will be immediately activated at time k_r to replace the QSMC for stable tracking control. The overall control system dynamics can be summarized as follows:

$$\begin{cases} x_i(k+1) = x_{i+1}(k), \quad i = 1, 2, \cdots, n-1 \\ x_n(k+1) = \begin{cases} f^s(x(k)) + u^s(k), & k_0 \le k \le k_c \\ f^h(x(k)) + u^s(k), & k_c < k \le k_d \\ f^h(x(k)) + u^0(k), & k_d < k \le k_r \\ f^h(x(k)) + u^h(k), & k_r < k, \end{cases} \end{cases} \tag{27}$$

where $s, h \in J_1$ ($h \neq s$); u^s and u^h are the control signals generated from the s-th and the h-th EBCs of (8), respectively; u^0 is the control signal generated from the QSMC of (16); k_0, k_c, k_d and k_r are defined in Fig. 1.

Theorem 6. *Consider the system (27). Under Assumptions 1–3, given the recurrent reference orbit x_d, and with initial condition $x(k_0)$ in a close vicinity of $x_d(k_0)$, we have: (i) all signals in the closed-loop system remain UUB; and (ii) the state tracking error $x - x_d$ converges to a small neighborhood around zero.*

Proof. For the time interval $k_0 \le k \le k_c$, system (27) operates in the environment s with the matched s-th EBC u^s. Then, according to Theorem 2, with the given initial conditions, we have that all signals in the closed-loop system remain UUB.

At time k_c, the active environment is switched to $h \in J_1$ ($h \neq s$), which is detected at time k_d, at which the QSMC u^0 of (16) is activated. According to the results of Theorem 3, during the time interval $k_d < k \leq k_r$, it is guaranteed that the overall system of (27) is stable and the system state x stays close to the reference state x_d of (2). At time k_r when rapid recognition of the active environment h is achieved and the h-th EBC of (8) is activated, according to Theorem 2, all signals in the closed-loop system (27) remain UUB, and the state tracking error will converge to a small neighborhood around zero. □

Remark 7. *Note that when the system (27) is operating in an anticipated environment $h \in J_1$, both online recognition and control are realized with knowledge/experience that was obtained and pre-stored through the offline training processes in Section 5.2 for recognition and Section 3.2 for control, which do not involve any real-time parameter adaptations. This could significantly reduce the computational complexity and time consumption while still ensuring satisfactory recognition and control performance.*

6.2. Unanticipated Environment Case

We further consider another case that the new active environment in Fig. 1 is an unanticipated environment, i.e., $h \in J_2$. As illustrated in Fig. 2, when the environment switching occurs (at time k_c), it will be detected (at time k_d) when the filtered tracking error $|r|$ of (5) becomes larger than the detection threshold of (14). At the same time k_d, both QSMC of (16) and online recognition scheme of (23)–(25) will be activated. In the online recognition process (see Fig. 2c), the active environment index h will be recognized (at time k_r) when the identifier of (25) completes the dynamics learning, such that a new recognition estimator of (26) can be constructed and added into the family of recognition estimators in (23). At the same time k_r, the ALC of (9) will be immediately activated to achieve stable tracking control and accurate learning of the associated nonlinear uncertain system dynamics. Once the learning process is completed (say, at time k_h), a new EBC of (10) can be constructed and activated immediately to replace the ALC for stable tracking control. Such a new EBC will be added as a new member into the family of EBCs in (8), and the unanticipated environment will thus become an anticipated environment. The overall control system

dynamics is summarized as follows:

$$\begin{cases} x_i(k+1) = x_{i+1}(k), \quad i = 1, 2, \cdots, n-1 \\ x_n(k+1) = \begin{cases} f^s(x(k)) + u^s(k), & k_0 \le k \le k_c \\ f^h(x(k)) + u^s(k), & k_c < k \le k_d \\ f^h(x(k)) + u^0(k), & k_d < k \le k_r \\ f^h(x(k)) + \hat{u}(k), & k_r < k \le k_h \\ f^h(x(k)) + u^h(k), & k_h < k, \end{cases} \end{cases} \tag{28}$$

where $s \in J_1$, $h \in J_2$, u^s, u^0, $\hat{u}$, u^h are control signals generated from the s-th EBC of (8), the QSMC of (16), the ALC of (9), and the new EBC of (10), respectively, k_0, k_c, k_d and k_r are defined in Fig. 1, k_h is the time when the learning process of ALC is completed.

Theorem 7. *Consider the system (28). Under Assumptions 1–3, given the recurrent reference orbit x_d, and with initial condition $x(k_0)$ in a close vicinity of $x_d(k_0)$, we have: (i) all signals in the closed-loop system remain UUB; (ii) the state tracking error $x - x_d$ converges to a small neighborhood around the origin; and (iii) through the processes of control and recognition, a new EBC of (10) and a new recognition estimator of (26) are obtained and added into (8) and (23), respectively, such that the corresponding environment h becomes a new anticipated environment, leading to $J_1 = \{h\} \cup J_1$, $\forall k > k_h$.*

Detailed proofs can be conducted by following a similar line of the proof of Theorem 6.

Remark 8. *The recognition time k_r in (28) can be determined when the learning process of identifier (25) is completed. It can be estimated by observing when the NN weight $\hat{W}_r$ of (25) converges to constant steady-state values, and the associated identification error $\hat{x}_r - x_n$ converges to a close vicinity of the origin, as illustrated in Fig. 2c. Similarly, k_h in (28) can be obtained by observing when the learning process of ALC in (9) is completed, i.e., the NN weight $\hat{W}$ of (9) converges to constant steady-state values, and the associated tracking error $x - x_d$ converges to a close vicinity of the origin, as illustrated in Fig. 2b. An example will be provided in the next section for further illustration.*

Remark 9. *Existing pattern-based NN learning control methods of [22, 23] are not applicable to the unanticipated environment case as discussed in Section 6.2. The main reason lies in that these methods are developed with an*

offline learning mechanism, which does not possess online learning capability to deal with an unanticipated environment. This issue is overcome with our method by incorporating the active online learning approaches (i.e., the identifier ([25]) and the ALC ([9])) in the real-time recognition and control processes, which can online acquire the associated knowledge of the operating unanticipated environment.

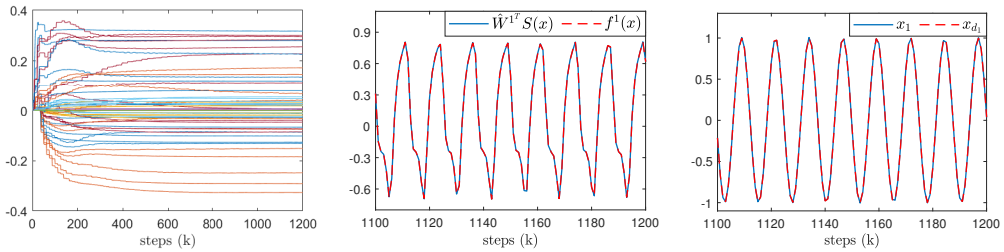
Remark 10. *Existing MMAC methods of [13, 15] cannot guarantee satisfaction of the PE condition, thus lacking the capability of accurate learning of system uncertainties. As a result, (i) these methods cannot realize accurate learning of the uncertain system dynamics under each individual environment, thus multiple adaptive models are required to operate in parallel; (ii) their implementation needs to repeat parameter adaptation even for a recurred anticipated environment, which suffers high computational burdens and is time consuming; (iii) they cannot achieve accurate and rapid recognition of the operating environments, which could deteriorate tracking control performance. In contrast to [13, 15], our approach can guarantee satisfaction of the PE condition with the utilization of the DL-based learning approach. For the system dynamics under each individual environment, accurate identification can be achieved and the associated knowledge can be obtained and represented by a constant model (instead of multiple adaptive models). In particular, for anticipated environments, the real-time recognition and control processes do not involve any parameter adaptation, while still ensuring satisfactory recognition and control performance, as discussed in Remark 7. These advantages distinguish our method from those existing ones of [13, 15].*

7. Simulation Studies

In this section, we consider a discrete-time nonlinear uncertain system (1) operating under multiple environments with $n = 2$, $j \in J = \{1, 2, 3\}$, and $f^1(x) = \frac{\sin x_1}{0.8+x_2^2} + 0.2x_1x_2$, $f^2(x) = \frac{0.8\sin x_1 + x_1}{1+\cos x_1+x_2^2} + 0.2\cos x_2$, $f^3(x) = \frac{x_1}{1+2x_2^2} + 0.15(1-x_1^2)x_2$. The nominal model of $f^j(x)$ $(\forall j \in J)$ is $\bar{f}(x) = \frac{x_1}{1+x_2^2}$. For simulation purpose, it is assumed that the environments $j = 1, 2$ are anticipated, i.e., $J_1 = \{1, 2\}$, and the environment $j = 3$ is unanticipated, i.e., $J_2 = \{3\}$. The reference model in the form of (2) is constructed with $n = 2$ and $f_d(x_d(k)) = \sin(0.5(k+1))$.

7.1. Performance of Learning and Control Under Individual Environment

We first implement and evaluate the adaptive learning control scheme developed from Section 3 for (1) under each fixed individual environment $j \in J$. In the offline learning mode, uncertain nonlinear dynamics of $f^j(x)$ for each $j \in J_1$ will be learned with the learning controller (4). The RBF network $\hat{W}^{jT}S(x)$ $(j = 1, 2)$ is constructed in a regular lattice with nodes $N_n = 169$, the centers evenly spaced on $[-1.2, 1.2] \times [-1.2, 1.2]$ and the width $\eta_t = 0.2$. The learning controller (4) is implemented with parameters $a_c^j = 0.6$, $\lambda_1 = 0.2$, and $c_c^j = 0.1$. The initial conditions are set as $\hat{W}^j(0) = 0$, $x(0) = [0.7, 0.5]^T$, and $x_d(0) = [0, 0]^T$. The simulation results corresponding to the environment $j = 1$ are plotted in Fig. 3. Fig. 3a shows the convergence of NN weights $\hat{W}^1$, Fig. 3b shows the accurate learning performance for unknown dynamics $f^1(x)$, and Fig. 3c illustrates the stable tracking control performance. Through the offline training process, the knowledge associated with $f^1(x)$ can be obtained and stored by an constant RBF NN model $\bar{W}^{1T}S(x)$ with $\bar{W}^1 = \frac{1}{100}\sum_{k=1101}^{1200}\hat{W}^1(k)$. For the case of the environment $j = 2$, the associated simulation results are similar and thus not included. Based on these learning results, according to Remark 1, the parameter that is needed for the design of (14) is obtained as $\varepsilon_{cm}^* = \max_{i=1,2}\{\varepsilon_c^{j*}\} = 0.0023$.

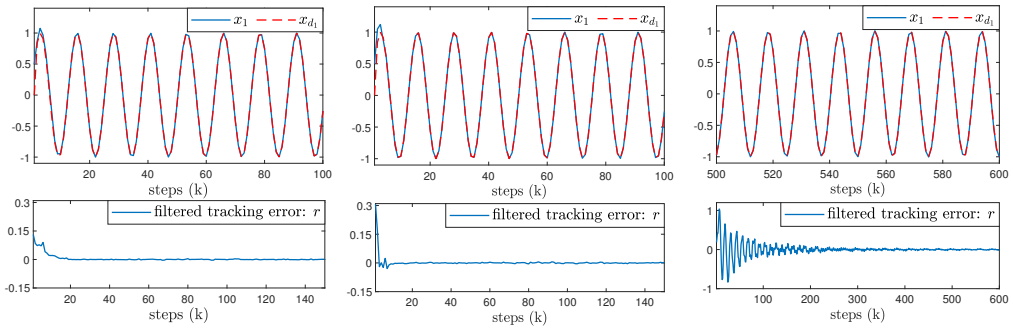


(a) (b) (c)

Figure 3: Learning control performance under fixed environment $j = 1$ using controller (4). (a) Convergence of NN weight $\hat{W}^1$. (b) Function approximation: $\hat{W}^{1T}S(x)$ and $f^1(x)$. (c) State tracking control performance: x_1 and x_{d_1}.

In the online learning control mode, based on the learned knowledge, we implement the EBCs of (8) and the ALC of (9) with $b_c = 0.6$, $\lambda_1 = 0.2$, $a_c = 0.6$ and $c_c = 0.1$. The tracking control performance and the corresponding filtered tracking error $r = (x_2 - x_{d_2}) + \lambda_1(x_1 - x_{d_1})$ are plotted in Fig. 4. It is seen that compared to the ALC in Fig. 4c, the EBCs in Figs. 4a and 4b achieve better control performance in terms of smaller transient

tracking errors and faster convergence rate. One important reason for such a performance gain is that the EBCs leverage the knowledge learned from the training process to provide stable tracking control without repeating online adaptation of NN weights.



(a) (b) (c)

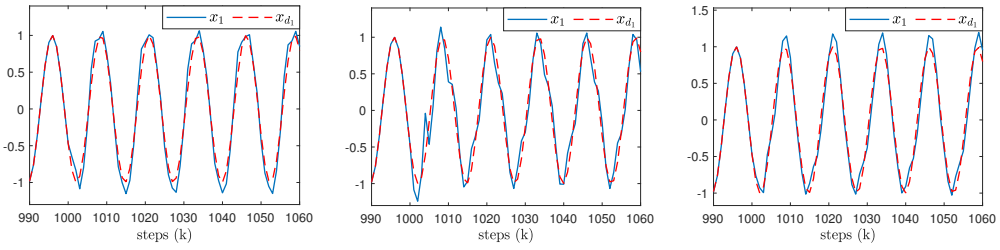
Figure 4: Control performance under different environments. (a) The 1-st EBC under environment $j = 1$; (b) The 2-nd EBC under environment $j = 2$; (c) The ALC under environment $j = 3$.

7.2. Performance of the QSMC for Stable Environment Transition

We further examine the control performance of the QSMC (16) during environment transitions. The associated parameters are set as $\alpha_0 = 1$, $\alpha_1 = 0.2$, and $\rho = 5$. The initial conditions are set as $x(0) = [1, 1]^T$, $x_d(0) = [0, 0]^T$, $s(0) = 4.9+\alpha_0 z_2(0)+\alpha_1 z_1(0)$ and $s(1) = -4.89+\alpha_0 z_2(1)+\alpha_1 z_1(1)$. With the QSMC, we examine three cases when the active environment of (1) is abruptly switched (i) from $j = 1$ to $j = 2$, (ii) from $j = 2$ to $j = 3$, and (iii) from $j = 3$ to $j = 1$, all at time $k = 1000$, respectively. The corresponding simulation results are plotted in Fig. 5, showing that the QSMC can guarantee the system stability during the environment transition, and the system trajectory x of (1) can be guaranteed to converge to a close vicinity of the reference orbit x_d of (2).

7.3. Performance of Adaptive Learning for Recognition of Active Environment

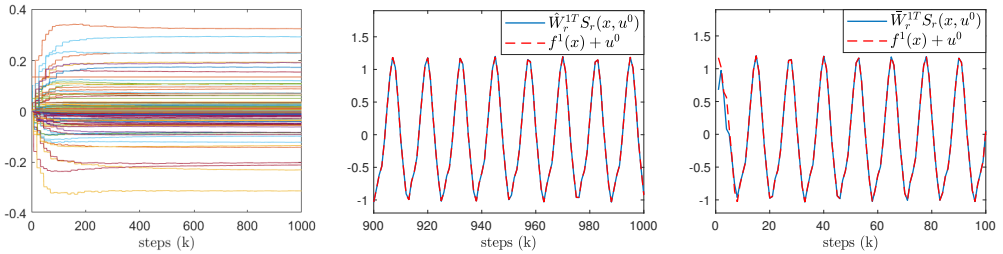
In the following, we continue to examine the recognition scheme proposed in Section 5. Offline training is first implemented for each fixed anticipated environment $j \in J_1$. The associated nonlinear uncertain system dynamics



(a) (b) (c)

Figure 5: Performance of the QSMC (16) during environment transition. Active environment is switched at time $k = 1000$ from (a) $j = 1$ to $j = 2$; (b) $j = 2$ to $j = 3$; (c) $j = 3$ to $j = 1$.

$f^j(x)+u^0$ is learned by using the identifier (20). The RBF NN $\hat{W}_r^{jT} S_r(x, u^0)$ $(j = 1, 2)$ is constructed in a regular lattice with nodes $N_n = 2366$, the centers evenly spaced on $[-1.2, 1.2] \times [-1.2, 1.2] \times [-1.4, 1.2]$ and the width $\eta_t = 0.2$. The NN weights $\hat{W}_r^j$ are updated according to (20) with $a_r^j = 0.2$, $c_r^j = 0.2$ and the initial conditions $\hat{W}_r^j(0) = 0$, $\hat{x}_r^j(0) = 0$. The simulation results corresponding to the environment $j = 1$ are shown in Fig. 6. Fig. 6a shows the convergence of NN weights $\hat{W}_r^1$, and Fig. 6b shows the accurate learning of $f^1(x)+u^0$ by $\hat{W}_r^{1T} S(x, u^0)$. The knowledge is obtained and stored in $\bar{W}_r^{1T} S_r(x, u)$ with $\bar{W}_r^1 = \frac{1}{100} \sum_{k=901}^{1000} \hat{W}_r^1(k)$, which also accurately approximates $f^1(x) + u^0$, as seen in Fig. 6c. For the case of the environment $j = 2$, the associated simulation results are similar and thus not included. Based on the identification results, from Remark 3, the learning accuracy level that is needed for the design of (24) is obtained as $\varepsilon_r^{1*} = 0.0042$, $\varepsilon_r^{2*} = 0.0100$.



(a) (b) (c)

Figure 6: Learning performance of (20). (a) Convergence of NN weight $\hat{W}_r^1$. (b) Function approximation: $\hat{W}_r^{1T} S_r(x, u^0)$ and $f^1(x)+u^0$. (c) Function approximation: $\bar{W}_r^{1T} S_r(x, u^0)$ and $f^1(x) + u^0$.

In the online recognition mode, the recognition estimators of (23) will be constructed by utilizing $\bar{W}_r^{jT} S_r(x, u^0)$ for all $j \in J_1$, and the adaptive threshold of (24) and the identifier of (25) will be implemented. The associated simulation results by following a specific environment switching sequence will be given and discussed in the next subsection.

7.4. Real-Time Control and Recognition Performance of the Overall Architecture

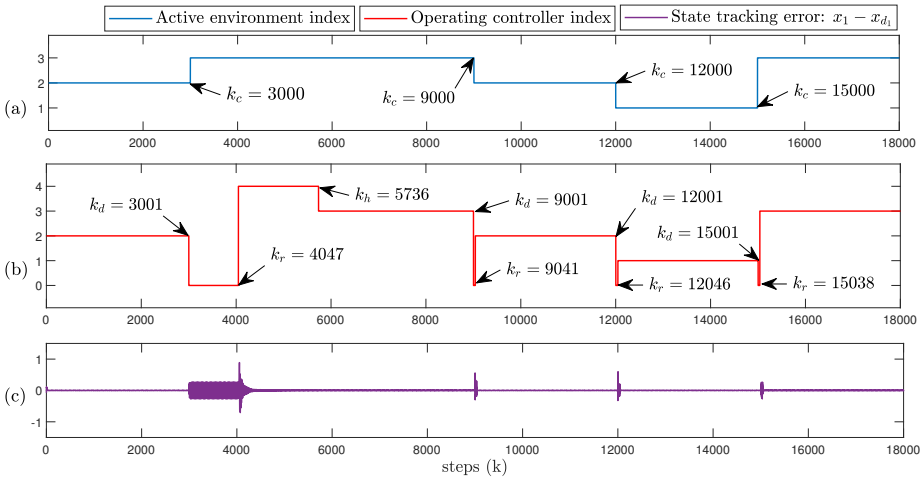


Figure 7: Real-time control performance under multiple environments. (a) Active environments with indexes $j = 1, 2, 3$. (b) Operating controllers with indexes 0: QSMC of (16); 1, 2: EBCs of (8); 3: new EBC of (10); 4: ALC of (9). (c) State tracking error: $x_1 - x_{d_1}$.

To examine the effectiveness of the overall control and recognition architecture, we assume that the active environment of (1) will follow the switching sequence of: $2\,(0 < k \leq 3000) \rightarrow 3\,(3000 < k \leq 9000) \rightarrow 2\,(9000 < k \leq 12000) \rightarrow 1\,(12000 < k \leq 15000) \rightarrow 3\,(15000 < k)$, as illustrated in Fig. 7a. We first examine the process of control and recognition when the plant (1) operates in the unanticipated environment $j = 3$ for time $3000 < k \leq 9000$. Specifically, when the active environment is switched from $j = 2$ to $j = 3$ at time $k_c = 3000$, which is detected rapidly in one step at time $k_c = 3001$ when the filtered tracking error $|r|$ becomes larger than the threshold $\bar{e}_c$ of (14) (with $\bar{\mu}_c = 0.05$, $b_c = 0.6$, $\varepsilon_{cm}^* = 0.0023$), as shown in Fig. 8a. Then, the QSMC of (16) and the recognition scheme of (23)–(25) (with $a_r = 0.2$, $c_r = 0.2$, $q_r = 5$, $b_r = 0.95$ and $\varepsilon_r^* = 0.01$) are both activated at time

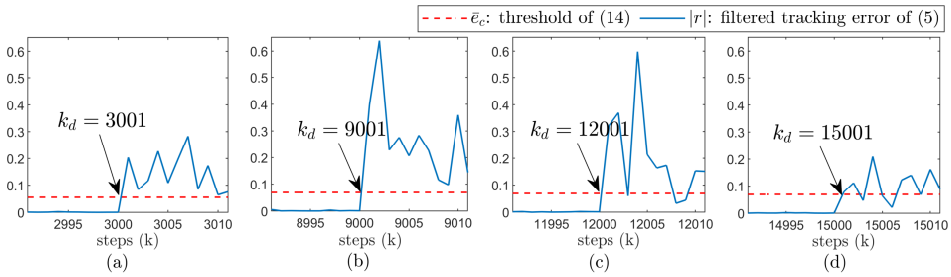


Figure 8: Detection of environment transition. The active environment switching: (a) $2 \to 3$ occurred at time $k_c = 3000$, detected at time $k_d = 3001$; (b) $3 \to 2$ occurred at time $k_c = 9000$, detected at time $k_d = 9001$; (c) $2 \to 1$ occurred at time $k_c = 12000$, detected at time $k_d = 12001$; and (d) $1 \to 3$ occurred at time $k_c = 15000$, detected at time $k_d = 15001$.

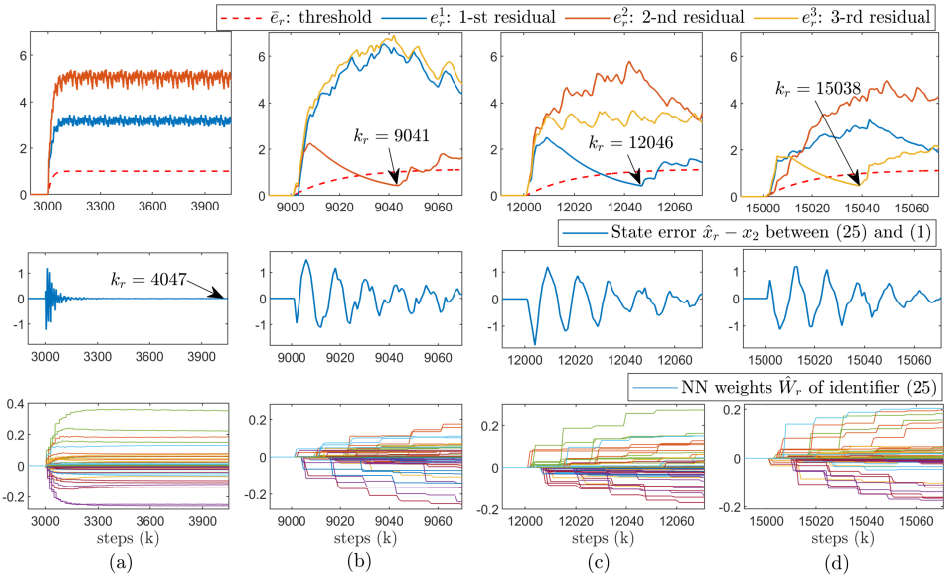


Figure 9: Recognition of active environment: (a) environment $j = 3$ for time $3000 < k \leq 9000$, (b) environment $j = 2$ for time $9000 < k \leq 12000$, (c) environment $j = 1$ for time $12000 < k \leq 15000$, and (d) environment $j = 3$ for time $k > 15000$.

$k_d = 3001$, as indicated in Fig. 7b. The real-time recognition performance is illustrated in Fig. 9a. It is seen from first row of Fig. 9a that all residuals e_r^j $(j = 1, 2)$ of (23) remain no smaller than the threshold $\bar{e}_r$ of (24), according to Theorem 5, the active environment can thus be identified as an unanticipated environment. In this case, the online adaptive learning

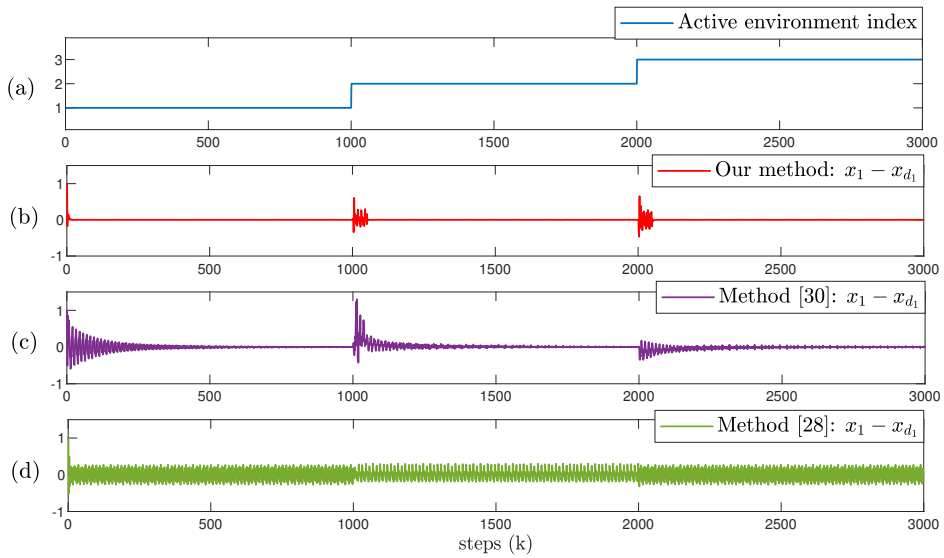


Figure 10: Tracking control performance with different methods. (a) Active environment switching sequence: $1\,(0 < k \leq 1000) \rightarrow 2\,(1000 < k \leq 2000) \rightarrow 3\,(2000 < k)$; (b) State tracking error $x_1 - x_{d_1}$ with our method; (c) State tracking error $x_1 - x_{d_1}$ with the method of [30]; (d) State tracking error $x_1 - x_{d_1}$ with the method of [28].

mechanism of (25), which is activated also at time $k_d = 3001$, completes the associated online learning mission at time $k_r = 4047$ when the identification state error $\hat{x}_r - x_2$ of (25) converges close to zero and the NN weights $\hat{W}_r$ of (25) converge to steady constant values, as shown in the second and third rows of Fig. 9a, respectively. According to Section 5.2, whenever the online adaptive learning process of the unanticipated environment is completed, the corresponding recognition process is completed, i.e., at time $k_r = 4047$. As a result, a new recognition estimator (i.e., the 3-rd estimator in the form of (26)) can be constructed for later online recognition use; and the recognition threshold of (24) can be updated with a new parameter $\varepsilon_r^{3*} = 0.0115$, according to Remark 3. Based on this, the ALC of (9) (with $a_c = 0.6$, $c_c = 0.1$) is activated at time $k_r = 4047$ to replace the QSMC for active online learning control over the time interval $4047 < k \leq 5736$, as indicated in Fig. 7b. At time $k_h = 5736$ when the learning process of ALC is completed (i.e., when the NN weights $\hat{W}$ of (9) converge to steady constant values and the associated tracking error $x - x_d$ converges into a close vicinity of the origin, the associated simulation results are similar to those in Figs. 3a and 4c and

thus omitted here), a new EBC (i.e., the 3-rd EBC in the form of (10)) can be constructed and activated immediately to replace the ALC for stabilizing tracking control without repeating parameter adaptations over the time interval $5736 < k \leq 9001$, as illustrated in Fig. 7b. When such a new EBC is constructed, from Remark 1, the detection threshold of (14) is updated with parameter $\varepsilon^*_{c_m} = \max_{j=1,2,3}\{\varepsilon_c^{j*}\} = 0.0077$. The overall control performance under the environment $j = 3$ (during time interval $3000 < k < 9000$) is plotted in Fig. 7c. It should be noted that once the learning process is completed (at time $k_h = 5736$), the environment $j = 3$ will become a new "anticipated environment", leading to $J_1 = \{1, 2, 3\}$. As a result, when the environment $j = 3$ recurs (for example, $k > 15000$ as shown in Fig. 7a), its associated recognition estimator of (26) and EBC of (10) can be used readily to achieve rapid recognition and desired tracking control performance, respectively, without resorting to any online adaptive learning any more. The associated simulation results are summarized in Figs. 9d and 7c (for $k > 15000$).

Next, we examine the case that the plant (1) is operating under the anticipated environment $j = 2$ for time $9000 < k \leq 12000$. Detection of the occurrence of environment switching (i.e., from environment $j = 3$ to environment $j = 2$ at $k_c = 9000$) is achieved at time $k_d = 9001$ when the filtered tracking error $|r|$ becomes larger than the threshold $\bar{e}_c$ of (14) (with $\bar{\mu}_c = 0.05$, $b_c = 0.6$, $\varepsilon^*_{c_m} = 0.0077$), as shown in Fig. 8b. Recognition of the new active environment $j = 2$ is achieved at time $k_r = 9041$ when the matched residual e_r^2 of (23) remains smaller than the threshold $\bar{e}_r$ of (24) (with $b_r = 0.95$, $q_r = 5$, $\varepsilon_r^* = 0.0115$) for 15 time steps, as shown in Fig. 9b. At time $k_r = 9041$, the active controller is switched to the corresponding 2-nd EBC, as indicated in Fig. 7b. The overall tracking control performance under environment $j = 2$ (for time $9000 < k \leq 12000$) is plotted in Fig. 7c. For another anticipated environment $j = 1$ (operating during $12000 < k \leq 15000$), the corresponding simulation results are summarized in Figs. 7, 8c and 9c. These results demonstrate the effectiveness of our scheme.

To verify the superiority of our framework, in the following, we will compare the control performance with those obtained by using the existing methods of [30, 28]. For fair comparison, our method is developed by assuming that all the environments $j = 1, 2, 3$ are anticipated, and the associated offline training (corresponding to Sections 3.1 and 5.1) have been completed. With the method of [30], the adaptive controller is constructed specifically accord-

ing to [30, Eqs. (8)-(10)] with the associated design parameters $\lambda_1 = 0.2$, $A = 0.2$, $\alpha = 0.06$. With the method of [28], the robust sliding mode controller can be constructed in the form of (16). With each of these three different types of methods, simulation studies are carried out by considering the plant (1) operating in switching environments with the same switching sequence of $1\,(0 < k \leq 1000) \rightarrow 2\,(1000 < k \leq 2000) \rightarrow 3\,(2000 < k)$, as shown in Fig. 10a. All simulation results are given in Fig. 10. It is seen from Fig. 10c that when the active environment is switched from one to another (e.g., at time instants $k = 1000$, $k = 2000$, respectively), the adaptive control of [30] will render slow convergence and large tracking errors. In Fig. 10d, the tracking control performance under the method of [28] is fairly satisfactory; while significantly improved performance can be witnessed in Fig. 10b with our method.

8. Conclusions

In this paper, we have proposed a novel, systematic and holistic intelligent adaptive learning control framework for a class of discrete-time nonlinear uncertain systems operating in multiple environments, which could be anticipated/predefined or unanticipated/new. This framework consists of three important components: (i) learning-based control when the system operates under each fixed individual environment; (ii) robust QSMC when the system operates in environment transition; and (iii) learning-based recognition of active environments. Offline and online learning approaches are incorporated in the algorithmic design of each component, equipping the proposed framework two important intelligence capabilities of (i) adapting to any anticipated environment through knowledge re-utilization; and (ii) adapting to unanticipated environments through active real-time acquisition of new knowledge. Extensive simulation studies have been conducted to verify the effectiveness and advantages of the proposed results.

Acknowledgements

This work was supported in part by the Natoinal Science Foundation under Grant CMMI-1929729.

References

[1] V. Panwar, N. Kumar, N. Sukavanam, and J.-H. Borm, "Adaptive neural controller for cooperative multiple robot manipulator system manipulating a single rigid object," *Applied Soft Computing*, vol. 12, no. 1, pp. 216–227, 2012.

[2] W. He, Z. Li, and C. P. Chen, "A survey of human-centered intelligent robots: issues and challenges," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 4, pp. 602–609, 2017.

[3] F. Shi, Q. Cao, C. Leng, and H. Tan, "Based on force sensing-controlled human-machine interaction system for walking assistant robot," in *2010 8th World Congress on Intelligent Control and Automation*. IEEE, 2010, pp. 6528–6533.

[4] B. Brumitt, B. Meyers, J. Krumm, A. Kern, and S. Shafer, "Easyliving: Technologies for intelligent environments," in *International Symposium on Handheld and Ubiquitous Computing*. Springer, 2000, pp. 12–29.

[5] D. Katić and M. Vukobratović, "Survey of intelligent control techniques for humanoid robots," *Journal of Intelligent and Robotic Systems*, vol. 37, no. 2, pp. 117–141, 2003.

[6] C. Wang, Y. Li, S. S. Ge, and T. H. Lee, "Optimal critic learning for robot control in time-varying environments," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 10, pp. 2301–2310, 2015.

[7] L. Rutkowski, "Adaptive probabilistic neural networks for pattern classification in time-varying environment," *IEEE transactions on neural networks*, vol. 15, no. 4, pp. 811–827, 2004.

[8] K. S. Narendra and J. Balakrishnan, "Adaptive control using multiple models," *IEEE transactions on automatic control*, vol. 42, no. 2, pp. 171–187, 1997.

[9] G. Tao, "Multivariable adaptive control: A survey," *Automatica*, vol. 50, no. 11, pp. 2737–2764, 2014.

[10] W. He, Z. Li, Y. Dong, and T. Zhao, "Design and adaptive control for an upper limb robotic exoskeleton in presence of input saturation," *IEEE*

Transactions on Neural Networks and Learning Systems, vol. 30, no. 1, pp. 97–108, 2019.

[11] G. Tao, *Adaptive control design and analysis*. New York, NY: John Wiley & Sons, 2003, vol. 37.

[12] W. He and Y. Dong, "Adaptive fuzzy neural network control for a constrained robot using impedance learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 4, pp. 1174–1186, 2018.

[13] K. S. Narendra and C. Xiang, "Adaptive control of discrete-time systems using multiple models," *IEEE Transactions on Automatic Control*, vol. 45, no. 9, pp. 1669–1686, 2000.

[14] J. Zhang, C. Yuan, P. Stegagno, W. Zeng, and C. Wang "Small fault detection from discrete-time closed-loop control using fault dynamics residuals," *Neurocomputing*, vol. 365, pp. 239–248, 2019.

[15] K. S. Narendra, O. A. Driollet, M. Feiler, and K. George, "Adaptive control using multiple models, switching and tuning," *International journal of adaptive control and signal processing*, vol. 17, no. 2, pp. 87–102, 2003.

[16] C. Wang and D. J. Hill, *Deterministic learning theory for identification, recognition, and control*. Boca Raton, FL: CRC Press, 2009.

[17] K.-S. Fu, "Learning control systems-review and outlook," *IEEE transactions on pattern analysis and machine intelligence*, no. 3, pp. 327–342, 1986.

[18] C. Yuan and C. Wang, "Persistency of excitation and performance of deterministic learning," *Systems & Control Letters*, vol. 60, no. 12, pp. 952–959, 2011.

[19] J. Zhang, C. Yuan, W. Cong, P. Stegagno, and W. Zeng, "Composite adaptive nn learning and control for discrete-time nonlinear uncertain systems in normal form," *Neurocomputing*, vol. 390, pp. 168–184, 2020.

[20] J. Zhang, C. Yuan, P. Stegagno, H. He, and C. Wang, "Small fault detection of discrete-time nonlinear uncertain systems," *IEEE Transactions on Cybernetics*, 2019, doi: 10.1109/TCYB.2019.2945629.

[21] T. Chen, C. Wang, and D. J. Hill, "Rapid oscillation fault detection and isolation for distributed systems via deterministic learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 6, pp. 1187–1199, 2013.

[22] F. Yang and C. Wang, "Pattern-based nn control of a class of uncertain nonlinear systems," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 4, pp. 1108–1119, 2017.

[23] F. Zhang, C. Wang, and F. Yang, "Pattern-based nn control for uncertain pure-feedback nonlinear systems," *Journal of the Franklin Institute*, vol. 356, no. 5, pp. 2530–2558, 2019.

[24] J. Zhang, C. Yuan, and P. Stegagno, "A novel intelligent learning control scheme for discrete-time nonlinear uncertain systems in multiple environments," *Proceedings of the ASME Dynamic Systems and Control Conference*, Pittsburgh, Pennsylvania, 2020, Oct. 4-7.

[25] J. Park and I. W. Sandberg, "Universal approximation using radial-basis-function networks," *Neural computation*, vol. 3, no. 2, pp. 246–257, 1991.

[26] M. POEWELL, *The Theory of Radial Basis Function Approximation.* Oxford, U.K.: Clarendon Press, 1992.

[27] J. Zhang, Q. Gao, C. Yuan, W. Zeng, S.-L. Dai, and C. Wang, "Similar fault isolation of discrete-time nonlinear uncertain systems: An adaptive threshold based approach," *IEEE Access*, vol. 8, pp. 80 755–80 770, 2020.

[28] D. Munoz and D. Sbarbaro, "An adaptive sliding-mode controller for discrete nonlinear systems," *IEEE transactions on industrial electronics*, vol. 47, no. 3, pp. 574–581, 2000.

[29] W. Chen, S. Hua and S.S. Ge, "Consensus-based distributed cooperative learning control for a group of discrete-time nonlinear multi-agent systems using neural networks," *Automatica*, vol. 50, no. 9, pp. 2254–2268, 2013.

[30] T. Chen and C. Wang, "Learning from neural control for a class of discrete-time nonlinear systems," *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pp. 6732–6737, 2009.