Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2013, Article ID 679039, 10 pages
http://dx.doi.org/10.1155/2013/679039

Research Article

Robust Adaptive Control for Nonlinear Discrete-Time Systems by Using Multiple Models

Xiao-Li Li, De-Xin Liu, Jiang-Yun Li, and Da-Wei Ding

School of Automation and Electrical Engineering and the Key Laboratory of Advanced Control of Iron and Steel Process (Ministry of Education), University of Science and Technology Beijing, Beijing 100083, China

Correspondence should be addressed to Xiao-Li Li; lixiaoli@hotmail.com

Received 23 August 2013; Accepted 28 September 2013

Academic Editor: Zhiguang Feng

Back propagation (BP) neural network is used to approximate the dynamic character of nonlinear discrete-time system. Considering the unmodeling dynamics of the system, the weights of neural network are updated by using a dead-zone algorithm and a robust adaptive controller based on the BP neural network is proposed. For the situation that jumping change parameters exist, multiple neural networks with multiple weights are built to cover the uncertainty of parameters, and multiple controllers based on these models are set up. At every sample time, a performance index function based on the identification error will be used to choose the optimal model and the corresponding controller. Different kinds of combinations of fixed model and adaptive model will be used for robust multiple models adaptive control (MMAC). The proof of stability and convergence of MMAC are given, and the significant efficacy of the proposed methods is tested by simulation.

1. Introduction

Due to the strong ability of approximation, neural network has been widely used in the identification of nonlinear system. It is also a very useful tool for prediction, pattern recognition, and control [1]. The network structure comprises the interconnected group of nodes and the weight. There are many kinds of neural networks such as back propagation (BP), radial basis function (RBF), cerebellar model articulation controller (CMAC). As the most effective learning algorithm for feedforward networks [2], BP neural network has been the research focus for many years [3–6].

Adaptive control of nonlinear systems using neural network has been an active research area for over two decades [7–9]. The controller will be set up by adjusting the weights of the neural network [10, 11]. But adaptive control using neural network still has the same shortcomings as conventional adaptive control; it is extensively studied in time-invariant system with unknown parameters or time-variant system with slow drifting parameters [12, 13]. While the system has abrupt changes in parameters, the algorithm cannot find the exact identification model and will respond slowly to system parameter variations. To solve this kind of problem, MMAC has been a very useful tool in recent years.

Since MMAC was presented in 1970s, it has attracted a lot of attention of experts [14–17]. MMAC is an effective approach to solve problems such as time variations and uncertainties. It has the ability to improve the transient responses and the control performance. According to the dynamic character of controlled plant, multiple models are set up to cover the uncertainty of parameter. Much research has been done on continuous-time and discrete-time linear systems [18, 19]. For nonlinear system, only a few results have been given. In recent years, the MMAC based on neural network has been considered by some researchers [20, 21]. But in these papers, the nonlinear system has been modeled by the combination of linear model (the main part) and neural network model (the unmodeled dynamics). The multiple models are still multiple linear models with different parameters, and neural network is used only to compensate for the modeling error of linear model. In this case, the nonlinear system should not be very complex, and too big modeling error between the system and linear model is forbidden. The parameter and structure uncertainty of a relatively complex nonlinear system cannot be modeled by this method. This kind of MMAC with neural network still follows the main ideas of linear MMAC.

In this paper, a kind of robust MMAC is proposed for nonlinear system. Multiple BP neural networks with different weights will be used to cover the uncertainty of the parameters of the system. A performance index function based on the identification errors will be used to choose the best model and the corresponding controller. Considering the unmodeling error of neural network, a dead-zone recursive algorithm will be used, and the proof of robust property and stability of the MMAC are given. Different combinations of adaptive models and fixed models will be used for MMAC, and the effectiveness of the proposed method has been tested in simulations.

2. Robust Adaptive Control Using Neural Network

The single-input/single-output nonlinear discrete-time system can be represented as follows:

$$y(k+1) = f_0(\cdot) + g_0(\cdot)\, u(k-d+1), \tag{1}$$

where f_0 and g_0 are infinitely differentiable functions of

$$y(k-n+1), \dots, y(k), u(k-d-m+1), \dots, u(k-d), \tag{2}$$

where y is the output, u is the input, $m \le n$, d is the relative degree of the system, and g_0 is bounded away from zero. The arguments of f_0 and g_0 are real variables.

Due to the existence of noncausal problem, normally state transformation should be made first [11], and a causal system as follows can be given:

$$y(k+d) = f_{d-1}[\mathbf{x}(k)] + g_{d-1}[\mathbf{x}(k)]\, u(k). \tag{3}$$

Assumptions 1 and 2 in [11] about $g_0(\mathbf{x}), \dots, g_{d-1}(\mathbf{x})$ and minimum phase assumption should still be satisfied.

As Assumption 3 in [11], there exist the weights $\mathbf{w}$, $\mathbf{v}$ of neural network; the functions $\hat{f}_{d-1}[\mathbf{x}(k), \mathbf{w}]$ and $\hat{g}_{d-1}[\mathbf{x}(k), \mathbf{v}]$ can approximate the functions f_{d-1} and g_{d-1} with any accuracy ϵ.

Plant (3) can be modeled by the neural network.

Consider

$$\hat{y}(k+d) = \hat{f}_{d-1}[\mathbf{x}(k), \mathbf{w}] + \hat{g}_{d-1}[\mathbf{x}(k), \mathbf{v}]\, u(k). \tag{4}$$

The functions $\hat{f}_{d-1}[\cdot,\cdot]$ and $\hat{g}_{d-1}[\cdot,\cdot]$ depend on the structure of the neural network and the number of neurons. For example, if $\hat{f}_{d-1}[\cdot,\cdot]$ and $\hat{g}_{d-1}[\cdot,\cdot]$ are three-layer neural networks with p and q hidden neurons, respectively, then they can be expressed as

$$\begin{aligned} \hat{f}_{d-1}[\mathbf{x}(k), \mathbf{w}] &= \sum_{i=1}^{p} w_i H\left(\sum_{j=1}^{m+n+d-1} w_{ij} x_j(k) + \hat{w}_i \right), \\ \hat{g}_{d-1}[\mathbf{x}(k), \mathbf{v}] &= \sum_{i=1}^{q} v_i H\left(\sum_{j=1}^{m+n+d-1} v_{ij} x_j(k) + \hat{v}_i \right), \end{aligned} \tag{5}$$

where H is a hyperbolic tangent function.

Let $\mathbf{w}(k)$ and $\mathbf{v}(k)$ denote the estimates of $\mathbf{w}$ and $\mathbf{v}$ at time k. Rewrite (3) and (4) as follows:

$$\begin{aligned} y(k+1) &= f_{d-1}[\mathbf{x}(k-d+1)] \\ &\quad + g_{d-1}[\mathbf{x}(k-d+1)]\, u(k-d+1), \\ \hat{y}(k+1) &= \hat{f}_{d-1}[\mathbf{x}(k-d+1), \mathbf{w}] \\ &\quad + \hat{g}_{d-1}[\mathbf{x}(k-d+1), \mathbf{v}]\, u(k-d+1). \end{aligned} \tag{6}$$

We have the estimated plant output as:

$$\begin{aligned} y^*(k+1) &= \hat{f}_{d-1}[\mathbf{x}(k-d+1), \mathbf{w}(k)] \\ &\quad + \hat{g}_{d-1}[\mathbf{x}(k-d+1), \mathbf{v}(k)]\, u(k-d+1). \end{aligned} \tag{7}$$

Define $e^*(k+1)$ as

$$e^*(k+1) = y^*(k+1) - y(k+1). \tag{8}$$

If the neural network could approximate the nonlinear system with zero error; that is, $\epsilon = 0$, the following weight $\boldsymbol{\Theta} = [\begin{smallmatrix}\mathbf{w}\\ \mathbf{v}\end{smallmatrix}]$ updating rule can be used:

$$\boldsymbol{\Theta}(k+1) = \boldsymbol{\Theta}(k) - \frac{1}{r(k)} e^*(k+1)\, \mathbf{J}(k-d+1), \tag{9}$$

where

$$\begin{aligned} &\mathbf{J}(k-d+1) \\ &\quad = \left[\left. \frac{\partial y(k+1)^*}{\partial \boldsymbol{\Theta}} \right|_{\boldsymbol{\Theta}(k)} \right]' \\ &\quad = \begin{bmatrix} \left(\left. \dfrac{\partial \hat{f}_{d-1}[\mathbf{x}(k-d+1), \mathbf{w}]}{\partial \mathbf{w}} \right|_{\mathbf{w}(k)} \right)' \\ \left(\left. \dfrac{\partial \hat{g}_{d-1}[\mathbf{x}(k-d+1), \mathbf{v}]}{\partial \mathbf{v}} \right|_{\mathbf{v}(k)} \right)' u(k-d+1) \end{bmatrix} \end{aligned} \tag{10}$$

and $r(k)$ is the reference command. One has

$$r(k) = 1 + \mathbf{J}'(k-d+1)\, \mathbf{J}(k-d+1). \tag{11}$$

Due to the existence of unmodeling dynamics $\epsilon > 0$, the design of robust adaptive controller should be considered. A dead-zone algorithm will be used instead of (9) for updating the weights. Therefor,

$$D(e) = \begin{cases} 0, & \text{if } |e| \le d_0, \\ \mathrm{e} - d_0, & \text{if } e > d_0, \\ e + d_0, & \text{if } e < -d_0, \end{cases} \tag{12}$$

where d_0 will be the function of ϵ which can be seen in proof procedure of the Theorem 1.

The output of the dead-zone function is used in the following updating rule:

$$\boldsymbol{\Theta}(k+1) = \boldsymbol{\Theta}(k) - \frac{1}{r(k)} D\left(e^*(k+1)\right) \mathbf{J}(k-d+1). \tag{13}$$

Define the parameter error as

$$\widetilde{\boldsymbol{\Theta}}(k) = \boldsymbol{\Theta}(k) - \boldsymbol{\Theta} \tag{14}$$

and give the control $u(k)$ as follows:

$$u(k) = \frac{-\hat{f}_{d-1}[\mathbf{x}(k), \mathbf{w}(k)] + r(k)}{\hat{g}_{d-1}[\mathbf{x}(k), \mathbf{v}(k)]}. \tag{15}$$

We have Theorem 1 for the feedback control system.

Theorem 1 (see [11]). *Suppose $|r(k)| \le d_1$ for all $k \ge 0$. Given any constant $\rho > 0$ and any small constant $d_0 > 0$, there exist positive constants $\rho_1 = \rho_1(\rho, d_1)$, $\rho_2 = \rho_2(\rho, d_1)$, $\epsilon^* = \epsilon^*(\rho, d_0, d_1)$, and $\delta^* = \delta^*(\rho, d_0, d_1)$ such that if Assumptions 1 and 3 are satisfied on $S \supset B_{\rho 1}$, with $\epsilon < \epsilon^*$, Assumption 2 is satisfied on $B_{\rho 2}$, $|\mathbf{x}(0)| \le \rho$, and $|\widetilde{\boldsymbol{\Theta}}(0)| \le \delta < \delta^*$, then*

(1) *the $\mathbf{x}(k)$ and $u(k)$ are bounded for all k,*

(2) *$|\widetilde{\boldsymbol{\Theta}}(k)|$ will be monotonically nonincreasing, and $\boldsymbol{\Theta}(k+1) - \boldsymbol{\Theta}(k)$ will converge to zero,*

(3) $\sum_{k=1}^{\infty}([D(e^*(k+1))]^2/(1+\mathbf{J}'(k-d+1)\mathbf{J}(k-d+1))) \le 0$,

(4) *the tracking error between the plant output and the reference command will converge to a ball of radius d_0 centered at the origin.*

3. Robust Multiple Model Adaptive Control

The conventional adaptive control systems are usually based on a fixed or slowly adaptive model. It cannot react quickly to abrupt changes and will result in large transient errors before convergence. For this kind of problem, MMAC algorithm is presented as a useful tool. The rationale for using MMAC is to ensure that there is at least one model with parameters sufficiently close to those of the unknown plant. By the switching rule, the control strategy is to determine the best model for the current environment at every instant and activate the corresponding controller. The structure of the multiple model adaptive control is shown in Figure 1.

3.1. Architecture of the Control System. Multiple adaptive models can be regarded as an extension of conventional indirect adaptive control. The objective is to make the control error $e^c = r - y$ tend to zero, where r is the desired output. The control system contains N identification models, denoted by $I^{(l)}$, $l \in \{1, 2, \ldots, N\}$ according to (7), operating in parallel. Consider the following:

$$\begin{aligned} I^{(l)} : y^{*(l)}(k+1) = {} & \hat{f}_{d-1}^{(l)}\left[\mathbf{x}(k-d+1), \mathbf{w}^{(l)}(k)\right] \\ & + \hat{g}_{d-1}^{(l)}\left[\mathbf{x}(k-d+1), \mathbf{v}^{(l)}(k)\right] \\ & \times u^{(l)}(k-d+1), \end{aligned} \tag{16}$$

where

$$\hat{f}_{d-1}^{(l)}\left[\mathbf{x}(k), \mathbf{w}^{(l)}(k)\right] = \sum_{i=1}^{p} w_i^{(l)}(k) H\left(\sum_{j=1}^{m+n+d-1} w_{ij}^{(l)}(k) x_j - \hat{w}_i^{(l)}(k)\right),$$

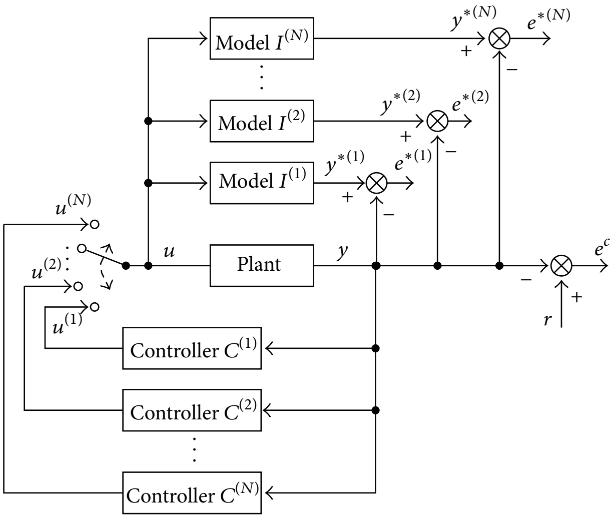


Figure 1: Structure of multiple model adaptive control.

$$\hat{g}_{d-1}^{(l)}\left[\mathbf{x}(k), \mathbf{v}^{(l)}(k)\right] = \sum_{i=1}^{q} v_i^{(l)}(k) H\left(\sum_{j=1}^{m+n+d-1} v_{ij}^{(l)}(k) x_j - \hat{v}_i^{(l)}(k)\right). \tag{17}$$

The identification error between the output $y^{*(l)}$ of $I^{(l)}$ and y of the plant is denoted as $e^{*(l)} = y^{*(l)} - y$. Corresponding to each $I^{(l)}$ is a parameterized controller $C^{(l)}$ which achieves the control objective for $I^{(l)}$. The output of $C^{(l)}$ is denoted by $u^{(l)}$.

One has

$$u^{(l)}(k) = \frac{-\hat{f}_{d-1}^{(l)}\left[\mathbf{x}(k), \mathbf{w}^{(l)}(k)\right] + r(k)}{\hat{g}_{d-1}^{(l)}\left[\mathbf{x}(k), \mathbf{v}^{(l)}(k)\right]}, \tag{18}$$

where $l \in \{1, 2, \ldots, N\}$, and $r(k)$ is the reference command.

At every instant, one of the models $I^{(l)}$ is selected by a switching rule, and the corresponding control input is used to control the plant.

Given prior knowledge of the different possible environments, the control problem is to determine suitable rules for switching and tuning these parameters to yield the best performance for the given objective while assuring stability.

3.2. Choice of Multiple Models. The following three different combinations have been considered [22].

3.2.1. N Adaptive Models. N adaptive models with different initial parameter values can be viewed as an extension of conventional indirect adaptive control. When the parameters of the plant change abruptly, the change can be detected by identification error. Then, the parameters of models are reset to initial values, and the model with the smallest error is selected. Hence, we can construct multiple adaptive models with different initial parameters which adjust dynamically in any instant. The method was considered in detail in [8, 19].

3.2.2. $N-1$ Fixed Models and One Adaptive Model. The previous method reveals that massive calculation may be produced because each adaptive model needs to adjust dynamically. Hence, if the models are fixed, the same strategy can be used in stationary and time-varying environments. However, fixed models can represent exactly only a finite number of environments. Thus, $N-1$ parallel fixed model and one adaptive model are combined, and the efficiency can be improved by the multiple fixed models and the accuracy can be increased by the adaptive model.

3.2.3. $N-2$ Fixed Models, One Free Running Adaptive Model, and One Reinitialized Adaptive Model. It is commonly accepted that the convergence time of an adaptive model will be large for large initial parametric errors. Hence, in the configuration described above, a large number of fixed models may be needed to keep the transient response under control until the adaptive model has converged. If the fixed model, which is the closest to the given plant, is assumed to be known, faster convergence can be obtained by initiating a new adaptive model from the location of the former. The same objective can be achieved on-line by starting adaptation from the location of each different fixed model that is successively chosen by the switching scheme.

The reinitialized adaptive model I^r included is introduced, and its parameters are determined as follows: if a fixed model $I^{(l)}$, $l \in \{1, 2, \ldots, N-2\}$, is activated by the switching rule at any instant k, then the parameters of I^r are reinitialized to the value of $I^{(l)}$. Thereafter, this adaptive model will be left to adapt until the next reinitialization.

3.3. Choice of the Switching Rule. A natural way to decide when and to which controller one should switch is to determine performance cost indexes for each controller $C^{(l)}$, $l \in \{1, 2, \ldots, N\}$, and switch to the one with the minimum index at every instant. However, since only one control input can be used at any instant, the performance of any candidate controller can be evaluated only after it has been used. On the other hand, the performance of all the identification models can be evaluated in parallel at every instant. Hence, the indexes must be based on the performance of the models rather than the controllers, that is, using identification errors $e^{(l)}$ rather than the control error e^c. From an adaptive control point of view, this rationale extends the principle of certainty equivalence from tuning to switching.

Considering the unmodeling error of neural network and robustness of the adaptive controller, the specific performance index proposed has the form

$$J_m(k) = \sum_{k=1}^{\infty} \frac{\left[D\left(e^*(k+1)\right)\right]^2}{1 + \mathbf{J}'(k-d+1)\,\mathbf{J}(k-d+1)}, \tag{19}$$

where $e^*(k+1) = y^*(k+1) - y(k+1)$.

The switching scheme consists of monitoring the performance indexes $J_m(k)$ at every instant. After every switching, the controller corresponding to the model with the minimum index is chosen (switched) to control the plant.

Theorem 2. *Suppose $|r(k)| < d_1$ for all $k \geq 0$. Given any constant $\rho > 0$, for all the model (adaptive model or fixed model), if $|x(0)| \leq \rho$, $|\widetilde{\boldsymbol{\Theta}}^{(l)}(0)| \leq \delta < \delta^*$, $l = 1, 2, \ldots, N$, and the conditions in Theorem 1 are satisfied, then when index switching* (19) *is used,*

(1) *all the signals in the system are bounded,*

(2) *the tracking error between $r(k)$ and $y(k+d)$*

$$\lim_{k\to\infty} \left|r(k) - y(k+d)\right| < d_0. \tag{20}$$

3.4. Proof of the Multiple Models Stability

3.4.1. N Adaptive Model. At time k, lth adaptive model will be selected, $l \in \{1, 2, \ldots, N\}$.

The control input

$$r(k) = \hat{f}_{d-1}\left[\mathbf{x}(k), \mathbf{w}^{(l)}(k)\right] + \hat{g}_{d-1}\left[\mathbf{x}(k), \mathbf{v}^{(l)}(k)\right] u(k). \tag{21}$$

From Theorem 1, if $|\mathbf{x}(0)| \leq \rho$, $|\widetilde{\boldsymbol{\Theta}}^{(l)}(0)| < \delta < \delta^*$, with control input (21), we can have that $\mathbf{x}(k)$ and $u(k)$ are bounded, for all k, and

$$\begin{aligned} \lim_{k\to\infty}\left|e^{*(l)}(k)\right| &= \lim_{k\to\infty}\left|y^{*(l)}(k) - y(k)\right| < d_0, \\ \lim_{k\to\infty}\left|\widetilde{\boldsymbol{\Theta}}^{(l)}(k)\right| &= \lim_{k\to\infty}\left|\boldsymbol{\Theta}^{(l)}(k+1) - \boldsymbol{\Theta}^{(l)}(k)\right| \longrightarrow 0. \end{aligned} \tag{22}$$

Then, the control error at time k is given by

$$\begin{aligned} \left|y^{*(l)}(k+d) - r(k)\right| \leq{} & \Big|\hat{f}_{d-1}\left[\mathbf{x}(k), \mathbf{w}^{(l)}(k+d-1)\right] \\ & - \hat{f}_{d-1}\left[\mathbf{x}(k), \mathbf{w}^{(l)}(k)\right]\Big| \\ & + \Big|\Big\{\hat{g}_{d-1}\left[\mathbf{x}(k), \mathbf{v}^{(l)}(k+d-1)\right] \\ & - \hat{g}_{d-1}\left[\mathbf{x}(k), \mathbf{v}^{(l)}(k)\right]\Big\} u(k)\Big| \\ \leq{} & k\left|\boldsymbol{\Theta}^{(l)}(k-d+1) - \boldsymbol{\Theta}^{(l)}(k)\right| \\ & \longrightarrow 0 \quad \text{as } k \longrightarrow \infty, \end{aligned} \tag{23}$$

$$\begin{aligned} \left|r(k) - y(k+d)\right| ={} & \Big|r(k) - y^{*(l)}(k+d) \\ & + y^{*(l)}(k+d) - y(k+d)\Big| \\ \leq{} & \left|r(k) - y^{*(l)}(k+d)\right| \\ & + \left|y^{*(l)}(k+d) - y(k+d)\right| \\ \leq{} & \left|y^{*(l)}(k+d) - y(k+d)\right| \\ <{} & d_0 \quad \text{as } k \longrightarrow \infty. \end{aligned} \tag{24}$$

Let one of N controller be chosen at random, for any instant of time k and any model chosen; then (24) holds.

So we have all the signals in the system bounded, and

(1) $\lim_{k\to\infty}|e^{*(l)}(k)| < d_0$,

(2) $\lim_{k\to\infty}|r(k) - y(k+d)| < d_0$.

3.4.2. N−1 Fixed Model and One Adaptive Model. Consider the following index function:

$$J_m^{(l)}(k) = \sum_{k=1}^{\infty} \frac{\left[D^{(l)}\left(e^*(k+1)\right)\right]^2}{1 + \left(\mathbf{J}^{(l)}(k-d+1)\right)'\mathbf{J}^{(l)}(k-d+1)}. \quad (25)$$

At every instant k, model $I^{(l)}$,

$$l = \arg\min_{1\le l\le N} J_m^{(l)}(k), \quad (26)$$

will be selected.

Proof. From Theorem 1, we have that, for adaptive model,

$$\begin{aligned}\lim_{k\to\infty} J_m^{(a)}(k) &= \lim_{k\to\infty} \sum_{k=1}^{\infty} \frac{\left[D^{(a)}\left(e^*(k+1)\right)\right]^2}{1 + \left(\mathbf{J}^{(a)}(k-d+1)\right)'\mathbf{J}^{(a)}(k-d+1)} \\ &< \infty.\end{aligned} \quad (27)$$

For the fixed models l, $\lim_{k\to\infty} J_m^{(l)}(k)$ is either bounded or ∞, if the performance index $J_m^{(l)}(k)$ tends to ∞; there exist a time t, $J_m^{(a)}(k) < J_m^{(l)}(k)$, $k \ge t$, which implies that the adaptive model will be selected finally.

If $\lim_{k\to\infty} J_{\mathrm{m}}^{(l)}(k)$ is bounded, then

$$\lim_{k\to\infty} \sum_{k=1}^{\infty} \frac{\left[D^{(l)}(e^*(k+1))\right]^2}{1 + \left(\mathbf{J}^{(l)}(k-d+1)\right)'\mathbf{J}^{(l)}(k-d+1)} < \infty, \quad (28)$$

$$\lim_{k\to\infty} \frac{\left[D^{(l)}(e^*(k+1))\right]^2}{1 + \left(\mathbf{J}^{(l)}(k-d+1)\right)'\mathbf{J}^{(l)}(k-d+1)} \longrightarrow 0. \quad (29)$$

If the fixed model is selected, the relationship of ρ, ϵ, δ in Theorem 1 is satisfied; the proof procedure will be similar to that of multiple adaptive controller. We also have the following:

(1) all the signals in the system are bounded,

(2) $\lim_{k\to\infty}|e^{*(l)}(k)| < d_0$,

(3) $\lim_{k\to\infty}|r(k) - y(k+d)| < d_0$.

□

3.4.3. N−2 Fixed Model, One Free Running Adaptive Model, and One Reinitialized Adaptive Model. The introduction of the reinitialized adaptive model will not affect the stability of the whole system, and the proof of the stability will be similar to the case of $N-1$ fixed model and one adaptive model.

4. Simulation

4.1. The Problem. PH neutralization is a very important procedure in the chemical industry. Usually, we use the logarithmic behavior to present pH characteristic; the existing nonlinearity always makes the identification and control of pH neutralization more difficult. A strong acid flows into a tank and is thoroughly mixed with a strong base whose inward rate of flow is controlled in such a way to produce a neutral outward flow from the tank. Because the acid and the base are strong, they are completely dissociated, and also the dissociation of the water can be disregarded [23]. The equation describing this model is

$$V\frac{dy}{dk} = F(k)\left(a - y(k)\right) - u(k)\left(b + y(k)\right), \quad (30)$$

where $y(k) = [\mathrm{H}^+] - [\mathrm{pH}^-]$ is the distance from neutrality,

V = volume of the tank,

$F(k)$ = rate of flow of the acid,

a = concentration of the acid,

$u(k)$ = rate of flow of the base,

b = concentration of the base.

Note that pH value can be determined from the $y(k)$, pH(k) by the following nonlinear transformation:

$$\mathrm{pH}(k) = -\log_{10}\left(\frac{y(k) + \sqrt{\left(y(k)\right)^2 + 4K_w}}{2}\right), \quad (31)$$

where K_w = water equilibrium constant $\simeq 10^{14}$.

We suppose that $\overline{b}$ and a are fixed and known, that $\overline{F}(k)$ can be measured online, and that $\overline{u}(k)$ can be given assigned values within certain limits.

An approximate discrete-time model can be developed, incorporating measurement and input actuator errors, as follows:

$$y_{k+1} \simeq y_k + \frac{T}{V}\left[F(k)\left(a - y_k\right) - u_k\left(b + y_k\right)\right]. \quad (32)$$

The following values were adopted for the various quantities of interest:

$$\begin{gathered} 0.1 \le F(k) \le 0.125\,\mathrm{l/min}, \qquad 0 \le u(k) \le 0.2\,\mathrm{l/min}, \\ a = 10^{-3}\,\mathrm{mol/l}, \qquad V = 2\,\mathrm{l}, \\ b = 10^{-3}\,\mathrm{mol/l}, \qquad T = 2\,\mathrm{min}. \end{gathered} \quad (33)$$

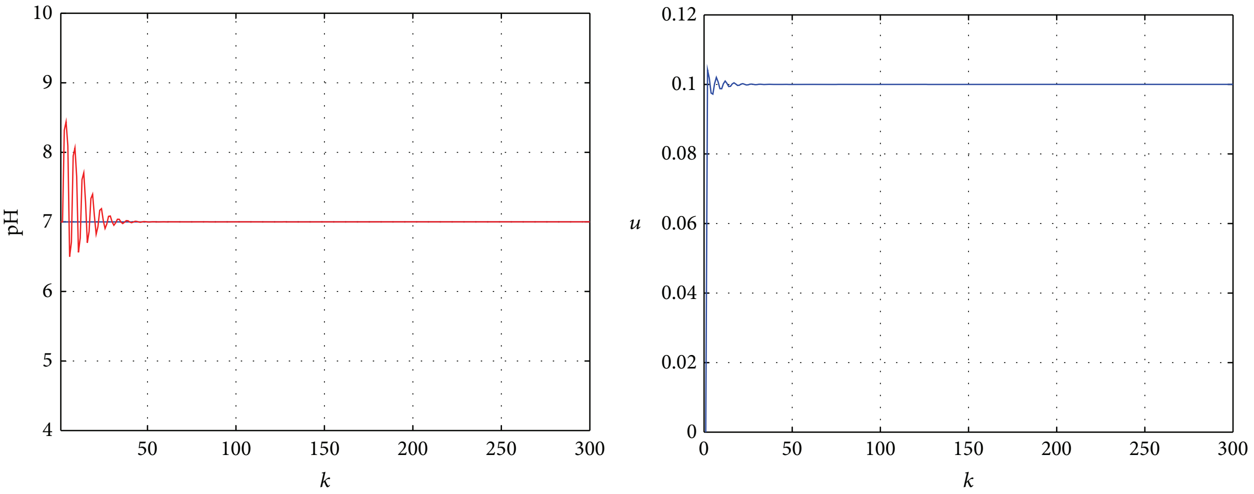
Then our goal is to control the plant as follows:

$$\begin{aligned} y(k+1) = {} & y(k)\left[1 - 0.5F(k)\right] + 0.5\times 10^{-3}F(k) \\ & - 0.5\left[10^{-3} + y(k)\right]u(k), \end{aligned} \quad (34)$$

where $F(x)$ is the variable parameter. And the single-input/single-output nonlinear discrete-time system represented as (1) can be modeled by

$$y^*(k+1) = \hat{f}\left[y(k), \mathbf{w}(k)\right] + \hat{g}\left[y(k), \mathbf{v}(k)\right]u(k), \quad (35)$$

where $\hat{f}$ and $\hat{g}$ are the output of neural network.



(a) System output pH(k)

(b) Control input $u(k)$

Figure 2: One adaptive mode with fixed parameter.

The neural networks $\hat{f}$ and $\hat{g}$ are 3 layered with 4 neurons in each hidden layer. Based on the error between the plant output $y(k+1)$ and the model output $y^*(k+1)$, the network parameters $\mathbf{w}(k)$ and $\mathbf{v}(k)$ are updated to $\mathbf{w}(k+1)$ and $\mathbf{v}(k+1)$ using the standard back propagation algorithm.

4.2. Experiments

Adaptive Model Only. In the process of parameter identification, let $F(k) = 0.1$, $d_0 = 0.001$, $y(1) = 0$, $y(2) = 0$, the reference command $r(k) = 0$ (according to (31), $\mathrm{pH}(k) = 7$), and $k = 1, 2, \ldots, 300$. The initial weights $\mathbf{w}$ and $\mathbf{v}$ are given a random number in the range $[-1, 1]$. The identification results are shown in Figure 2.

After 300 sample times, the weights will converge to the following values:

$$\begin{aligned}\mathbf{w} = {} & [0.865, 0.799, -0.823, 1.478, 0.232, 0.484, \\ & -0.251, 0.794, 1.052, -0.742, 0.591, -0.019]^T, \\ \mathbf{v} = {} & [-0.483, -0.665, -0.288, -0.767, -0.519, -0.187, \\ & 0.506, -0.236, -0.268, 0.965, 1.201, 0.891]^T.\end{aligned} \quad (36)$$

Now, we consider a worse case of the plant where abruptly changing parameters appear:

$$F(k) = \begin{cases} 0.1, & 1 \le k < 100, \\ 0.113, & 100 \le k < 200, \\ 0.125, & 200 \le k < 300. \end{cases} \quad (37)$$

Given the initial weights as the convergent weights of $\mathbf{w}$ and $\mathbf{v}$ in (36).

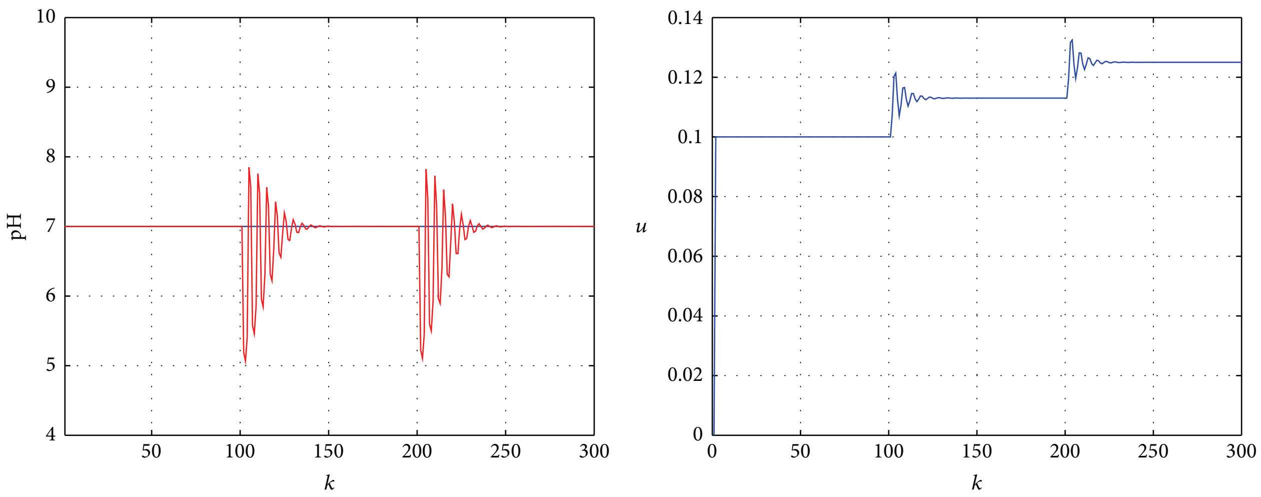
As the parameters change at $k = 100$ and $k = 200$, the overshoot of the system is big and the settling time is long. The nonlinear system cannot track the reference trajectory in time (Figure 3). When MMAC is used, the following simulation results can be obtained.

4.2.1. Three Adaptive Models. Three adaptive models $I^{(a1)}$, $I^{(a2)}$, and $I^{(a3)}$ are established. According to the three different values of $F(k)$ obtained ($k = 1, 100, 300$), each group of weights can be got using the same method as that in (36). One has

$$\begin{aligned} \mathbf{w}^{(a1)} &= \mathbf{w}, \qquad \mathbf{v}^{(a1)} = \mathbf{w} \\ \mathbf{w}^{(a2)} &= [-0.423, 0.482, -0.054, 1.023, 0.487, 0.248, \\ & \quad 0.072, 0.825, 0.230, -0.214, -0.678, 0.032]^T, \\ \mathbf{v}^{(a2)} &= [-1.064, -0.103, 0.563, 0.578, -0.439, -0.487, \\ & \quad 0.258, -0.557, -0.820, 0.385, 0.842, 0.831]^T, \\ \mathbf{w}^{(a3)} &= [-0.584, 0.341, 0.376, 0.733, 0.394, 0.476, \\ & \quad 1.067, 1.089, -0.144, -0.577, 0.051, -0.058]^T, \\ \mathbf{v}^{(a3)} &= [0.811, 0.076, -0.532, 0.399, 0.412, -0.268, \\ & \quad 0.897, -0.384, 0.933, -0.131, -0.313, -0.758]^T. \end{aligned} \quad (38)$$

The multiple models based on neural networks are chosen as in (16). Figures 4(a) and 4(b) present the responses of the plant. Switching sequence of controllers is shown in Figure 4(c). Obviously, this method can track the reference trajectory fast and improve the transient response. According to the index function, the system can choose an approximate model to identify the unknown plant. Once the parameters change, the weights and the index functions of neural network models will be initialized and the system will choose the optimal model again to conduct identification. In this way, the overshoot of the system can be decreased and the reference trajectory can be tracked fast at the same time.

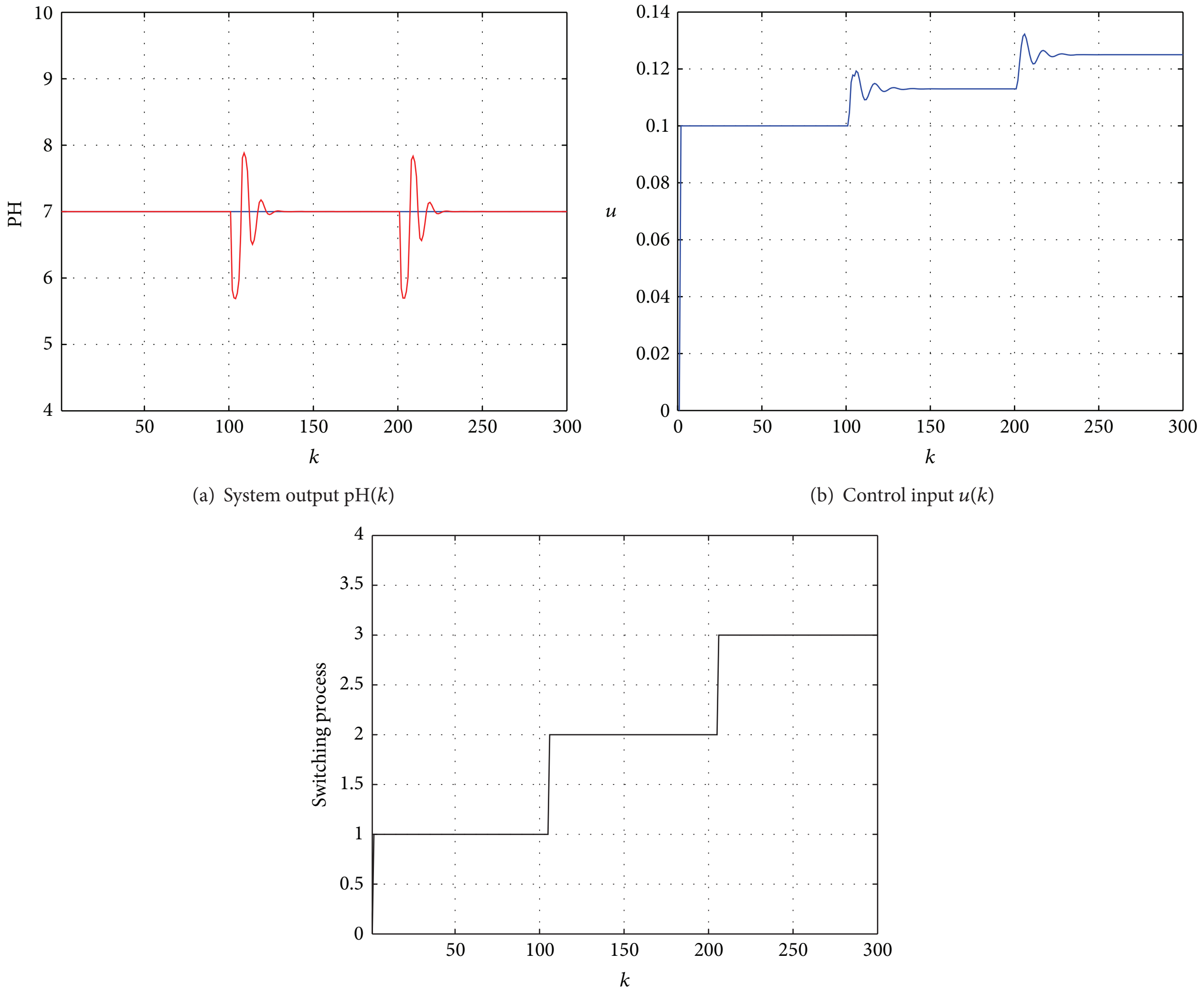
4.2.2. Three Fixed Models and One Adaptive Model. In this case, three fixed models $I^{(f1)}$, $I^{(f2)}$, and $I^{(f3)}$ are used to



(a) System output pH(k)

(b) Control input $u(k)$

Figure 3: One adaptive mode with variable parameter.

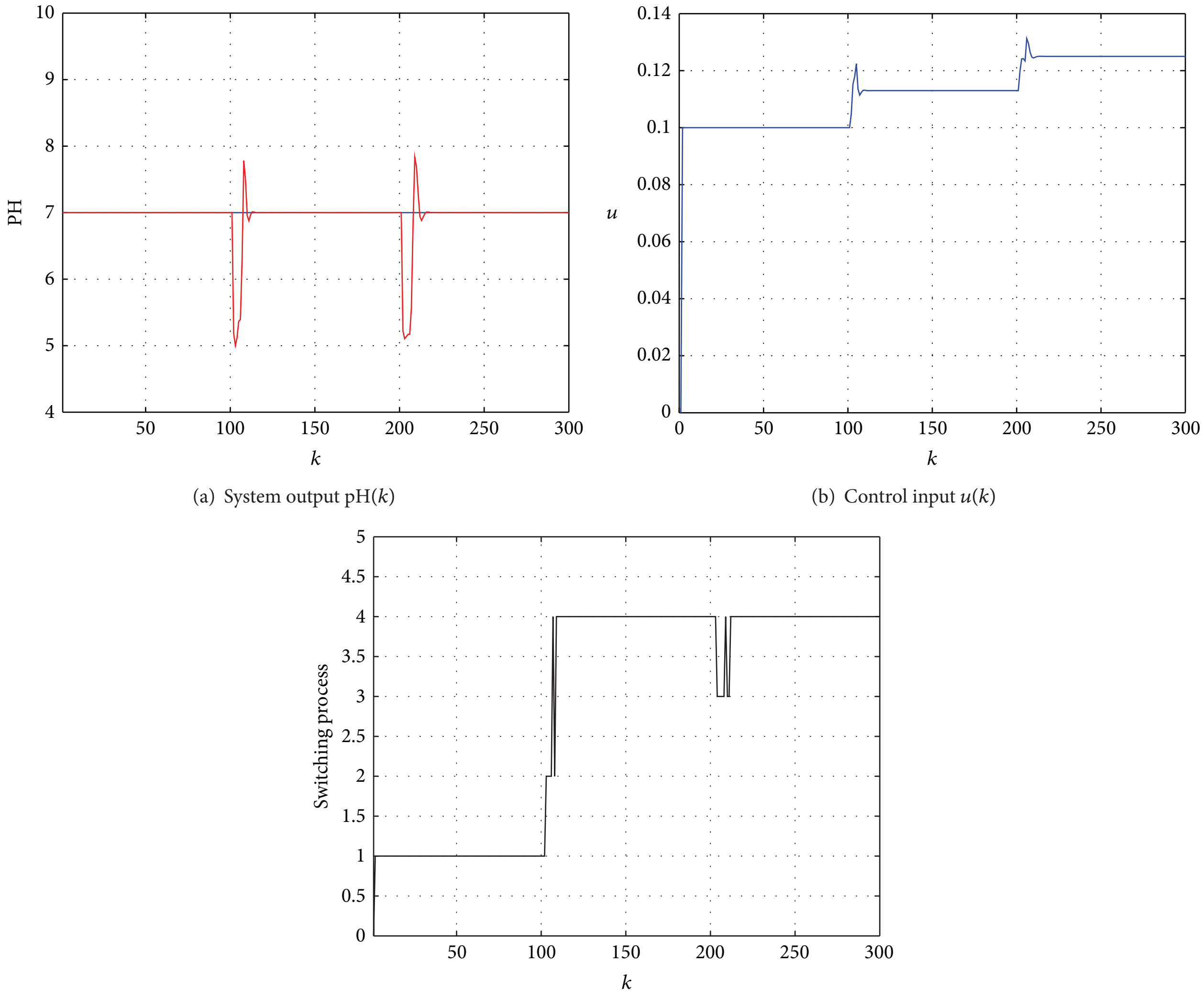


(a) System output pH(k)

(b) Control input $u(k)$

(c) Switching scheme

Figure 4: Three adaptive models.



(a) System output pH(k)

(b) Control input $u(k)$

(c) Switching scheme

Figure 5: Three fixed models and one adaptive model.

improve the transient response, and one adaptive model $I^{(a4)}$ is used to guarantee the stability. The initial weights of the three fixed models are the same as those of the three adaptive models, and the weights of one adaptive model $I^{(a4)}$ are equal to those of $I^{(f1)}$.

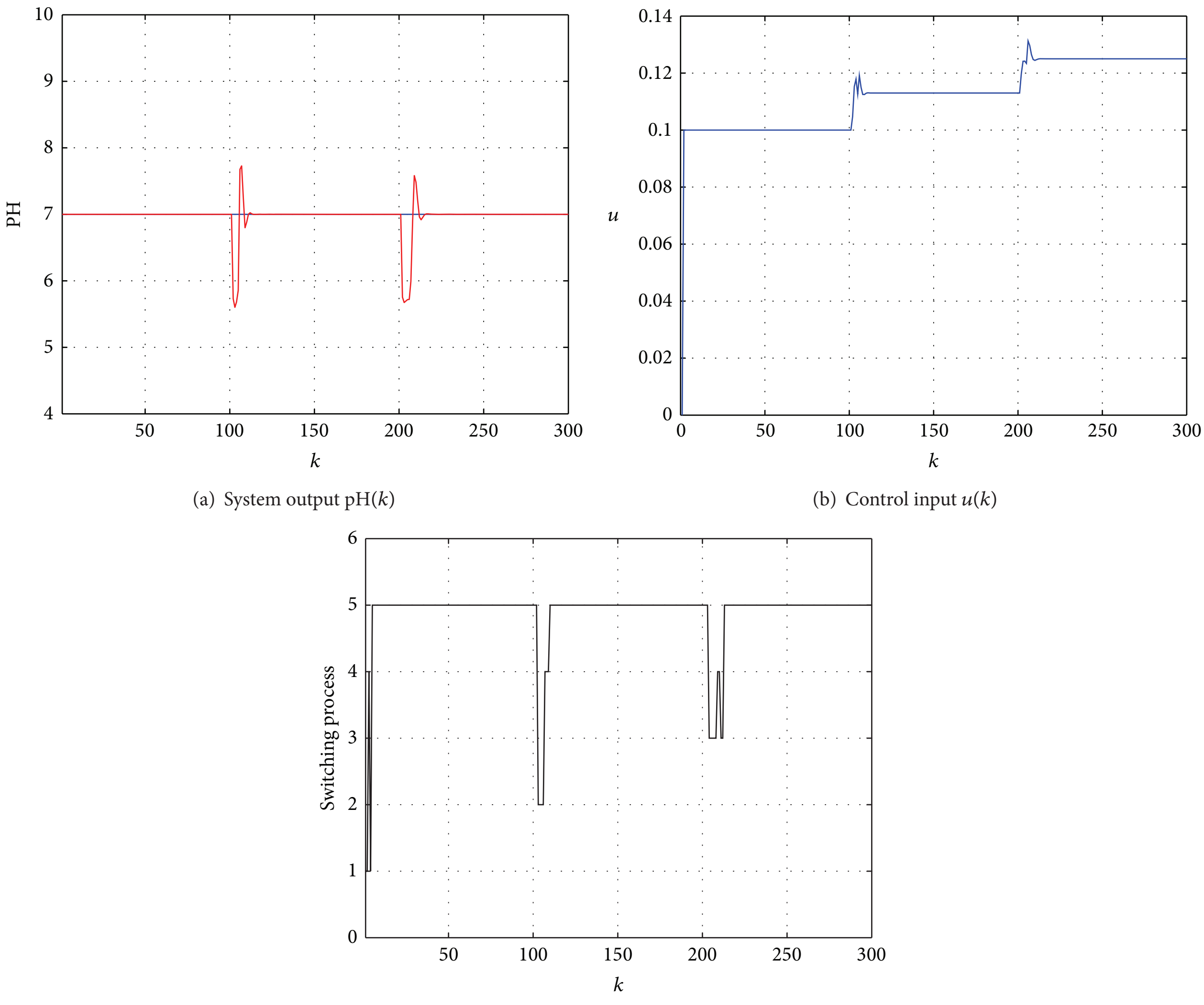
In the process of parameter identification, this method could improve the transient response compared with the conventional adaptive control (Figures 5(a) and 5(b)). Switching sequence of controllers is shown in Figure 5(c). Once the parameter changes abruptly at 100 or 200, the controller will switch to the nearest fixed model $I^{(f2)}$, $I^{(f3)}$ to reduce the error. When the adaptive model gradually converges to the true value, the system will switch to the adaptive model $I^{(a4)}$. Multiple fixed models play a transitional role in the process of identification. This method can reduce massive calculation compared with the case of three adaptive models, but it produces a larger overshoot compared with Figures 4(a) and 4(b).

4.2.3. Three Fixed Models, One Free Running Adaptive Model, and One Reinitialized Adaptive Model. In this case, we establish three fixed models $I^{(f1)}$, $I^{(f2)}$, and $I^{(f3)}$ with different initial weights; $I^{(a4)}$ is the free adaptive model and $I^{(r5)}$ is the reinitialized adaptive model. The reinitialized adaptive model can achieve the initial weights by choosing a set of fixed models based on the past performance of the plant. If at any instant one of them is determined to be the best, the reinitialized adaptive model can be adapted from this model.

From the simulation, we can see that this method can improve the control quality dramatically (Figures 6(a) and 4(b)). Switching sequence of controllers is shown in Figure 6(c). Compared with the other algorithm proposed before, this method show perfect performance in reducing the overshoot and tracking the reference trajectory, and computation time is reduced greatly.

5. Conclusion

In this paper, multiple models are used to establish robust multiple models adaptive controller for a class of nonlinear discrete-time systems by using neural networks. Three kinds of combinations of adaptive model and fixed model are used



(a) System output pH(k)

(b) Control input $u(k)$

(c) Switching scheme

FIGURE 6: Three fixed models, one free running adaptive model and one reinitialized adaptive model.

to make the multiple model set, and a switching law is suitably defined to make the decision of the best model. The principal contribution of this paper is the proof of stability of robust MMAC by using neural networks. Multiple neural network models with different weights represent different dynamical characters of the plant when it operates in different environments, which can be described by a mount of input and output data. So the design of the model set can also be regarded as a kind of data driven problem [24, 25]. How to divide the region of data into suitable numbers of subregions which can be represented by multiple neural network models will decide the accuracy of MMAC. A moving or dynamically optimal model set will be an important problem that needs to be solved in the future.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported by the Fundamental Research Funds for the Central Universities under Grant FRF-TP-12-005B, the Program for New Century Excellent Talents in Universities under Grant NCET-11-0578, and the National Natural Science Foundation of China under Grant 61074055.

References

[1] D. Wang and J. Huang, "Adaptive neural network control for a class of uncertain nonlinear systems in pure-feedback form," *Automatica*, vol. 38, no. 8, pp. 1365–1372, 2002.

[2] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[3] X. J. Jing and L. Cheng, "An optimal PID control algorithm for training feedforward neural networks," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 6, pp. 2273–2283, 2013.

[4] B. William and L. Stanislao, "Robot navigation control based on monocular images: an image processing algorithm for obstacle avoidance decisions," *Mathematical Problems in Engineering*, vol. 2012, Article ID 240476, 14 pages, 2012.

[5] M. Arvaneh, C. Guan, K. K. Ang, and C. Quek, "Optimizing spatial filters by minimizing within-class dissimilarities in electroencephalogram-based brain—computer interface," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 4, pp. 610–619, 2013.

[6] H. A. Talebi and K. Khorasani, "A neural network-based multiplicative actuator fault detection and isolation of nonlinear systems," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 3, pp. 842–851, 2012.

[7] K. J. Astrom and B. Wittenmark, *Adaptive Control*, Addison-Wesley, Longman Publishing, Boston, Mass, USA, 1995.

[8] K. S. Narendra and A. Annaswamy, *Stable Adaptive Systems*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1989.

[9] P. A. Ioannou and J. Sun, *Robust Adaptive Control*, Prentice-Hall, Upper Saddle River, NJ, USA, 1996.

[10] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 4–27, 1990.

[11] F. C. Chen and H. K. Khalil, "Adaptive control of a class of nonlinear discrete-time systems using neural networks," *IEEE Transactions on Automatic Control*, vol. 40, no. 5, pp. 791–801, 1995.

[12] F. C. Chen and H. K. Khalil, "Adaptive control of nonlinear systems using neural networks," *International Journal of Control*, vol. 55, no. 6, pp. 1299–1317, 1992.

[13] G. Kreisselmeier and B. D. O. Anderson, "Robust model reference adaptive control," *IEEE Transactions on Automatic Control*, vol. 31, no. 2, pp. 127–133, 1986.

[14] L. Giovanini, "Robust adaptive control using multiple models, switching and tuning," *IET Control Theory & Applications*, vol. 5, no. 18, pp. 2168–2178, 2011.

[15] W. Wang and X. L. Li, *Multiple Model Adaptive Control*, Science Press, Beijing, China, 2001.

[16] D. T. Magill, "Optimal adaptive estimation of sampled stochastic processes," *IEEE Transactions on Automatic Control*, vol. 10, pp. 434–439, 1965.

[17] D. G. Lainiotis, "Partitioning: a unifying framework for adaptive systems. I. Estimation," vol. 64, no. 8, pp. 1126–1143, 1976.

[18] M. Athans, D. Castanon, K.-P. Dunn et al., "The stochastic control of the F-8C aircraft using a multiple model adaptive control (MMAC) method—part I: equilibrium flight," *IEEE Transactions on Automatic Control*, vol. 22, no. 5, pp. 768–780, 1977.

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

[20] L. Chen and K. S. Narendra, "Nonlinear adaptive control using neural networks and multiple models," *Automatica*, vol. 37, no. 8, pp. 1245–1255, 2001.

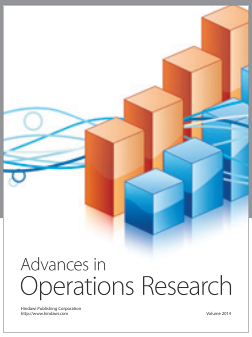
[21] Y. Fu and T. Chai, "Nonlinear multivariable adaptive control using multiple models and neural networks," *Automatica*, vol. 43, no. 6, pp. 1101–1110, 2007.

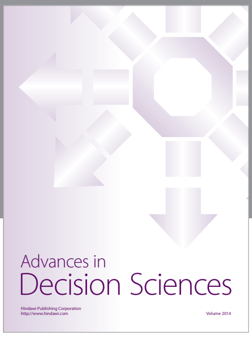
[22] K. S. Narendra and C. Xiang, "Adaptive control of discrete-time systems using multiple models," *IEEE Transactions on Automatic Control*, vol. 45, no. 9, pp. 1669–1686, 2000.

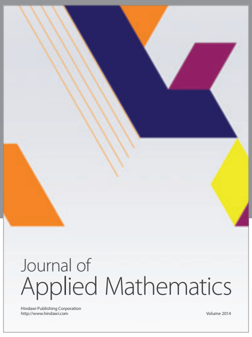
[23] G. C. Goodwin and K. S. Sin, *Adaptive Filtering Prediction and Control*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1984.

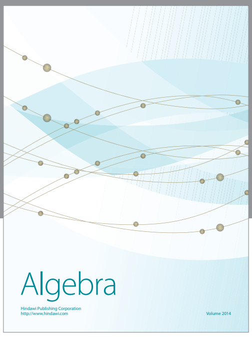
[24] S. Yin, S. X. Ding, A. Haghani, H. Y. Hao, and P. Zhang, "A comparison study of basic data-driven fault diagnosis and process monitoring methods on the benchmark Tennessee Eastman process," *Journal of Process Control*, vol. 22, no. 9, pp. 1567–1581, 2012.

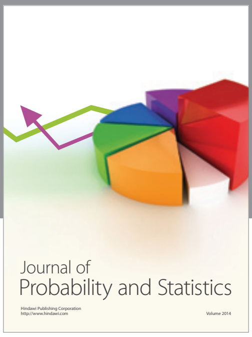
[25] S. Yin, H. Luo, and S. X. Ding, "Real-time implementation of fault-tolerant control systems with performance optimization," *IEEE Transactions on Industrial Electronics*, no. 99, p. 1, 2013.


Advances in Operations Research


Advances in Decision Sciences


Journal of Applied Mathematics

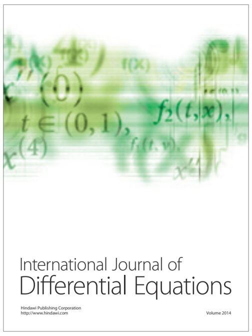

Algebra


Journal of Probability and Statistics

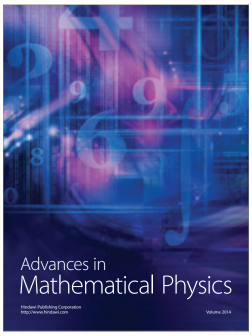
The Scientific World Journal

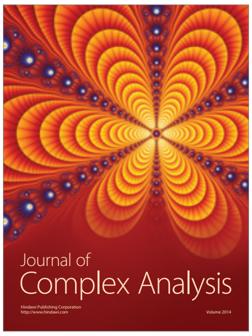
Hindawi

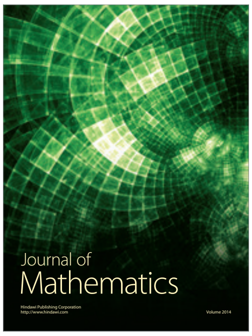
Submit your manuscripts at
http://www.hindawi.com

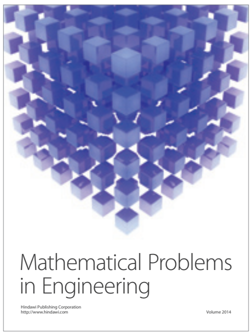

International Journal of Differential Equations

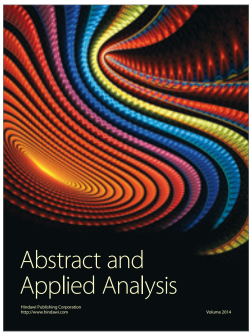
International Journal of Combinatorics

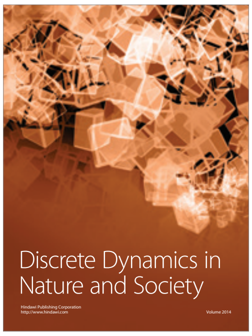

Advances in Mathematical Physics

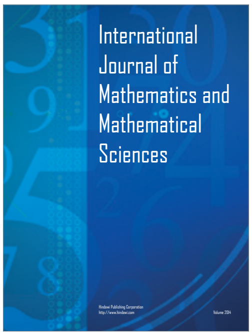

Journal of Complex Analysis


Journal of Mathematics


Mathematical Problems in Engineering

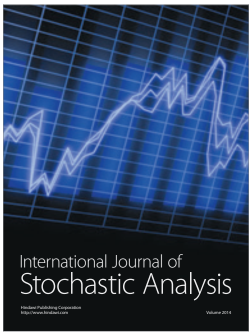

Abstract and Applied Analysis


Discrete Dynamics in Nature and Society


International Journal of Mathematics and Mathematical Sciences

Journal of Discrete Mathematics

Journal of Function Spaces


International Journal of Stochastic Analysis

Journal of Optimization