Discrete-time neural predictive controller design

Chi-Huang Lu

Abstract

This paper presents a design methodology for generalized predictive control (GPC) using recurrent neural network (RNN). A discrete-time mathematical model using RNN is constructed and a learning algorithm adopting an adaptive learning rate (ALR) approach is employed to identify the unknown parameters in the recurrent neural network model (RNNM). The neural predictive controller (NPC) is obtained via a generalized predictive performance criterion, and the convergence of the NPC including the adaptive optimal rate (AOR) by the Lyapunov stability theorem is presented. The illustrative process system is used to demonstrate the effectiveness of the proposed strategy. Results from numerical simulations show that the proposed method is capable of controlling nonlinear system with satisfactory performance under setpoint and load changes.

Keywords: generalized predictive control, recurrent neural network, nonlinear system.

Chi-Huang Lu, Associate Professor, Department of Electrical Engineering, HIT.

離散時間類神經預估控制器設計

呂奇璜

摘 要

本論文提出遞迴式類神經網路(RNN)之廣義預估控制(GPC)其技術設計。離散數學模型使用 RNN 架構，學習演算法採用適應學習率(ARL)來辨別遞迴式類神經網路模型(RNNM)其未知參數。類神經預估控制器(NPC)係由預估性能準則推導獲得，而適應最佳率(AOR)藉由 Lyapunov 穩定定理使得 NPC 能夠收斂穩定。由程序系統範例來展示所提控制策略是具有效性，數值模擬結果顯示所提控制方法對非線性系統在設定點與負載變動下有滿意的性能。

關鍵詞：廣義預估控制、遞迴式類神經網路、非線性系統。

呂奇璜：修平技術學院電機工程系副教授
投稿日期：970218 接受刊登日期：970729

1. Introduction

Model predictive control (MPC) has been successfully and extensively used for a great deal of industrial plants in both academia and industry [1]-[6]. In recent years, researchers have proposed many theoretical and practical methods for solving nonlinear predictive control problems in both continuous-time and discrete-time settings [7]-[9].

Since neural networks can approximate any nonlinear functions with arbitrary accuracy, they have been applied to develop adaptive control of nonlinear systems [10]-[12]. In particular, the recurrent neural network is a dynamical mapping and demonstrates good control performance in the presence of unmodeled dynamics; each recurrent neuron has an internal feedback loop, and then captures the dynamic response of a system without external feedback through delays [13]. In the past decade, several researchers have extensively investigated RNN-based predictive control with its applications to nonlinear systems. For examples, Parlos *et al*. [14] presented an architecture for integrating neural networks with industrial controllers for use in predictive control of complex process systems, Li *et al*. [15] proposed a simple recurrent neural network-based adaptive predictive control for nonlinear systems with known one-step time-delay, and Yoo *et al.* [16] developed generalized predictive control based on self-recurrent wavelet neural network for stable path tracking of mobile robots.

There are two principal objectives in the paper. The first is to propose a controller for a class of nonlinear systems using NPC with RNN model. The RNNM is updated by the gradient descent method with the ALR, which are used to guarantee the convergences of RNNM. Moreover, the convergence of the NPC system including the AOR via the Lyapunov stability theorem is well studied. The second is to illustrate the effectiveness of this proposed method for control by computer simulations on the nonlinear process system.

The remainder of the paper is organized as follows. Section 2 presents the RNN model for a class of nonlinear systems. Section 3 derives the RNNM-based NPC strategy and the NPC algorithm with AOR. Section 4 details the capabilities of the proposed algorithm for controlling the nonlinear system utilizing computer simulations. Section 5 concludes this paper.

2. RNNM structure

The section is developed RNNM for the nonlinear systems discussed in [17]. This class of nonlinear system is described by the following nonlinear autoregressive moving averaging (NARMA) model

$$y(k) = f\big(y(k-1),\ \cdots,\ y(k-n_y),\ u(k-d),\ \cdots,\ u(k-n_u)\big) \tag{1}$$

where $y(\cdot): Z^+ \to \Re$ and $u(\cdot): Z^+ \to \Re$ are the process output and input, respectively; $f(\cdot): \Re^{(n_y+n_u-d+1)} \to \Re$ denotes the nonlinear function; $n_y \in Z^+$ and $n_u \in Z^+$ are the orders of $\{y(k)\}$ and $\{u(k)\}$ respectively; $d \in Z^+$ represents the known time-delay of the system.

In this paper, we use a three-layer recurrent neural network for RNNM, and the RNNM output $\hat{y}(k)$ is mathematically expressed by

$$\hat{y}(k) = \sum_{j=1}^{n_j} w_j^O s_j(k)$$

$$s_j(k) = \sigma\left(w_j^H s_j(k-1) + \sum_{i=1}^{n_y} w_{i,j}^I \hat{y}(k-i) + \sum_{i=1}^{n_u-d+1} w_{(n_y+i),j}^I u(k-d+1-i) \right) \tag{2}$$

where w_j^O is the connection weight between hidden layer and output layer, w_j^H is the self-feedback weight of hidden layer, $w_{i,j}^I$ is the connection weight between input layer and hidden layer, $s_j(k-1)$ is a node of the hidden layer for the discrete time $k-1$, and the activation function is given by $\sigma(\kappa_j) = (1-e^{-\kappa_j})/(1+e^{-\kappa_j})$.

To update the weights of the RNNM, the cost function (3) defines as follows;

$$\Psi(k) = \frac{1}{2}\big(y(k) - \hat{y}(k)\big)^2 . \tag{3}$$

The weights are recursively adjusted in order to reduce the cost function (3) to its minimum value by the gradient descent method, and we have

$$\begin{aligned}\mathbf{W}(k) &= \mathbf{W}(k-1) - \gamma \frac{\partial \Psi(k)}{\partial \mathbf{W}} \\ &= \mathbf{W}(k-1) + \gamma\big(y(k) - \hat{y}(k)\big)\frac{\partial \hat{y}(k)}{\partial \mathbf{W}}\end{aligned} \tag{4}$$

where

$$\mathbf{W} = \begin{bmatrix} w_1^O & \cdots & w_{n_j}^O & w_1^H & \cdots & w_{n_j}^H & w_{1,1}^I & \cdots & w_{(n_y+n_u-d+1),n_j}^I \end{bmatrix}^T$$

$$\frac{\partial \hat{y}(k)}{\partial \mathbf{W}} = \left[\frac{\partial \hat{y}(k)}{\partial w_1^O} \cdots \frac{\partial \hat{y}(k)}{\partial w_{n_j}^O}\ \frac{\partial \hat{y}(k)}{\partial w_1^H}\ \frac{\partial \hat{y}(k)}{\partial w_{n_j}^H}\ \frac{\partial \hat{y}(k)}{\partial w_{1,1}^I} \cdots \frac{\partial \hat{y}(k)}{\partial w_{(n_y+n_u-d+1),n_j}^I} \right]^T \tag{5}$$

The initial values of w_j^O, w_j^H, and $w_{i,j}^I$ are given randomly in rang of $[-1,\ 1]$. By using the chain rule, and then the $\partial\hat{y}(k)/\partial w_j^O$, $\partial\hat{y}(k)/\partial w_j^H$, and $\partial\hat{y}(k)/\partial w_{i,j}^I$ can be calculated as follows;

$$\frac{\partial \hat{y}(k)}{\partial w_j^O} = s_j(k) \tag{6}$$

$$\frac{\partial \hat{y}(k)}{\partial w_j^H} = \frac{\partial \hat{y}(k)}{\partial s_j(k)} \frac{\partial s_j(k)}{\partial w_j^H} = w_j^O s'_j(k) s_j(k-1) \tag{7}$$

$$\frac{\partial \hat{y}(k)}{\partial w_{i,j}^I} = \frac{\partial \hat{y}(k)}{\partial s_j(k)} \frac{\partial s_j(k)}{\partial w_{i,j}^I} = w_j^O s'_j(k) x_i(k) \tag{8}$$

where

$$s'_j(k) = \frac{2e^{-\left(w_j^H s_j(k-1) + \sum_{i=1}^{n_y} w_{i,j}^I \hat{y}(k-i) + \sum_{i=1}^{n_u-d+1} w_{(n_y+i),j}^I u(k-d+1-i)\right)}}{\left(1 + e^{-\left(w_j^H s_j(k-1) + \sum_{i=1}^{n_y} w_{i,j}^I \hat{y}(k-i) + \sum_{i=1}^{n_u-d+1} w_{(n_y+i),j}^I u(k-d+1-i)\right)}\right)^2}$$

$$\begin{aligned} &\left[x_1(k) \quad x_2(k) \quad \cdots \quad x_i(k) \quad \cdots \quad x_{n_y+n_u-d+1}(k)\right]^T \\ &= \left[y(k-1) \ \cdots \ y(k-n_y) \ u(k-d) \ \cdots \ u(k-n_u)\right]^T \end{aligned} \tag{9}$$

In general, if a small value is given for the positive learning rate γ, then the convergence of the RNNM will be guaranteed, but the convergence rate may be rather slow. Conversely, if a large value for γ is considered, then the RNNM may become unstable. The following theorem states a sufficient condition of the convergence of the RNNM for selecting an appropriate learning rate.

Theorem 1: Let $\gamma > 0$ be the learning rate for the weights of RNNM (2) and $\boldsymbol{\alpha}_{\max}$ be defined as $\boldsymbol{\alpha}_{\max} = \max\left(\partial \hat{y}(k)/\partial \mathbf{W}\right) \in \Re^{2n_j + (n_y+n_u-d+1)\times n_j}$. Then the convergence is guaranteed if γ is chosen as

$$\gamma < \frac{2}{\boldsymbol{\alpha}_{\max}^T \boldsymbol{\alpha}_{\max}}. \tag{10}$$

Proof: Let a Lyapunov function be selected as $L(k) = \left(y(k) - \hat{y}(k)\right)^2$, and the detailed proof procedure can be referred to [17]. The ALR of the RNNM can be use as half of the upper limits in (10) for guaranteeing the selected learning rate inside the stable region. *i.e.*,

$$\gamma^* = \frac{1}{\boldsymbol{\alpha}_{\max}^T \boldsymbol{\alpha}_{\max}}. \tag{11}$$

3. NPC algorithm

This section aims to present a NPC for the nonlinear controlled process. According to the MPC strategy for the proposed control method, the cost function is defined by the well-know generalized predictive performance criterion [18] as follows:

$$\begin{aligned} J(k) &= \frac{1}{2} \sum_{p=1}^{N_p} \left(r(k+p) - \hat{y}(k+p)\right)^2 + \frac{1}{2} \lambda \Delta u^2(k) \\ &= \frac{1}{2} \left(\mathbf{R}(k) - \mathbf{Y}(k)\right)^T \left(\mathbf{R}(k) - \mathbf{Y}(k)\right) + \frac{1}{2} \lambda \Delta u^2(k) \end{aligned} \tag{12}$$

where

$$\mathbf{R}(k) = \left[r(k+1) \quad r(k+2) \quad \cdots \quad r(k+N_p) \right]^T$$

$$\mathbf{Y}(k) = \left[\hat{y}(k+1) \quad \hat{y}(k+2) \quad \cdots \quad \hat{y}(k+N_p) \right]^T$$

The N_p is the predictive output horizon, $r(k+p)$ is an input reference signal for discrete time $k+p$, $\hat{y}(k+p)$ is the p-step-ahead of $\hat{y}(k)$, and $\lambda > 0$ is the selected weighting value.

The control $u(k)$ is obtained from the optimization of the cost function (12) based upon using the gradient descent method, that is

$$u(k) = u(k-1) + \eta\left(\mathbf{G}^T(k)\left(\mathbf{R}(k) - \mathbf{Y}(k)\right) - \lambda \Delta u(k)\right) \tag{13}$$

where

$$\mathbf{G}(k) = \left[\frac{\partial \hat{y}(k+1)}{\partial u(k)} \ \frac{\partial \hat{y}(k+2)}{\partial u(k)} \ \cdots \ \frac{\partial \hat{y}(k+N_p)}{\partial u(k)} \right]^T . \tag{14}$$

Note that the details of the prediction based on neural networks for $\hat{y}(k+p)$ can be referring to [12]. Letting $u(k+N_p) = \cdots = u(k+1) = u(k)$ to reduce the computational load, and $\hat{y}(k+p)/\partial u(k)$ is obtained as following;

$$\frac{\partial \hat{y}(k+p)}{\partial u(k)} = \frac{\partial \left(\sum_{j=1}^{n_j} w_j^O s_j(k+p) \right)}{\partial u(k)} = \sum_{j=1}^{n_j} w_j^O s_j'(k+p) w_{(n_y+1),j}^I \tag{15}$$

where

$$s_j'(k+p) = \frac{2e^{-\left(w_j^H s_j(k+p-1) + \sum_{i=1}^{n_y} w_{i,j}^I \hat{y}(k+p-i) + \sum_{i=1}^{n_u-d+1} w_{(n_y+i),j}^I u(k+p-d+1-i) \right)}}{\left(1 + e^{-\left(w_j^H s_j(k+p-1) + \sum_{i=1}^{n_y} w_{i,j}^I \hat{y}(k+p-i) + \sum_{i=1}^{n_u-d+1} w_{(n_y+i),j}^I u(k+p-d+1-i) \right)} \right)^2} \tag{16}$$

The control increment $\Delta u(k)$ is defined as follows:

$$\Delta u(k) = \frac{\eta}{1+\eta\lambda} \mathbf{G}^T(k)\left(\mathbf{R}(k) - \mathbf{Y}(k)\right) \tag{17}$$

To insure that the identification process will be successful, the persistently exciting (PE) signals [19] are used as the testing signals for accomplishing the PE conditions. The following theorem states that the resulting NPC is convergent

Theorem 2: Let $\eta > 0$ be a optimal rate for the NPC (17) and $\boldsymbol{\beta}_{\max}$ be defined as $\boldsymbol{\beta}_{\max} = \max \mathbf{G}(k) \in \Re^{N_p}$. Then, the convergence is guarantee if η is chosen to satisfy

$$\eta < \frac{2(1+\lambda)}{\boldsymbol{\beta}_{\max}^T \boldsymbol{\beta}_{\max}} . \tag{18}$$

Proof: Define a Lyapunov function candidate as

$$L(k) = \left(\mathbf{R}(k) - \mathbf{Y}(k)\right)^T \left(\mathbf{R}(k) - \mathbf{Y}(k)\right) = \mathbf{E}^T(k)\mathbf{E}(k) \tag{19}$$

where $\mathbf{E}(k) = \left[e(k+1) \ e(k+2) \ \cdots \ e(k+N_p) \right]^T$

with $e(k+p) = r(k+p) - \hat{y}(k+p)$. Then we obtain

$$\Delta L(k) = L(k+1) - L(k)$$

$$= 2\Delta \mathbf{E}^T(k)\mathbf{E}(k) + \Delta \mathbf{E}^T(k)\Delta \mathbf{E}(k) \,. \quad (20)$$

Since the $\Delta \mathbf{E}(k)$ can be represented, *i.e.*

$$\Delta \mathbf{E}(k) = \frac{\partial \mathbf{E}(k)}{\partial u(k)} \Delta u(k) = -\frac{\partial \mathbf{Y}(k)}{\partial u(k)} \Delta u(k) \,. \quad (21)$$

Using $\Delta u(k) = \frac{\eta}{1+\eta\lambda} \mathbf{G}^T(k)\mathbf{E}(k)$, and the (21) can be represent as follows;

$$\Delta \mathbf{E}(k) = -\frac{\eta}{1+\eta\lambda} \mathbf{G}(k)\mathbf{G}^T(k)\mathbf{E}(k) \quad (22)$$

Using (22), the (20) can be expressed as follows:

$$\Delta L(k) = -2\frac{\eta}{1+\eta\lambda}\left(\mathbf{G}(k)\mathbf{G}^T(k)\mathbf{E}(k)\right)^T \mathbf{E}(k)$$

$$+\left(\frac{\eta}{1+\eta\lambda}\right)^2 \left(\mathbf{G}(k)\mathbf{G}^T(k)\mathbf{E}(k)\right)^T \left(\mathbf{G}(k)\mathbf{G}^T(k)\mathbf{E}(k)\right)$$

$$= -\frac{\eta}{1+\eta\lambda} \mathbf{E}^T(k)\mathbf{G}(k)\mathbf{G}^T(k)\left(2I - \frac{\eta}{1+\eta\lambda}\mathbf{G}(k)\mathbf{G}^T(k)\right)\mathbf{E}(k)$$

$$= -\frac{\eta}{1+\eta\lambda} \mathbf{E}^T(k)\left[\left(\mathbf{G}(k)\mathbf{G}^T(k)\right)\left(2I - \frac{\eta}{1+\eta\lambda}\mathbf{G}(k)\mathbf{G}^T(k)\right)\right]\mathbf{E}(k)$$

(23)

In order to satisfy the $\Delta L(k) \le 0$, we should restrict η to

$$\eta < \frac{2}{\boldsymbol{\beta}_{\max}^T \boldsymbol{\beta}_{\max}} \le \frac{2}{\mathbf{G}^T(k)\mathbf{G}(k)} \,. \quad (24)$$

Therefore, if η choose as the condition (18), and then it can be shown that the convergence of the proposed NPC algorithm is guarantee. The AOR of the NPC is also used as half of the upper limits in (18) for guaranteeing the learning rate inside the stable region. *i.e.*,

$$\eta^* = \frac{1}{\boldsymbol{\beta}_{\max}^T \boldsymbol{\beta}_{\max}} \,. \quad (25)$$

The control algorithm is summarized in the following procedure.

Step 1) Set $r(k)$, n_j, λ, and N_p.

Step 2) Measure the system output $y(k)$.

Step 3) Update the weights with the ALR (11) and AOL (25).

Step 4) Compute the control increment $\Delta u(k)$ by (17).

Step 5) Output $u(k) = u(k-1) + \Delta u(k)$ to the controlled system.

Step 6) Repeat steps 2-5.

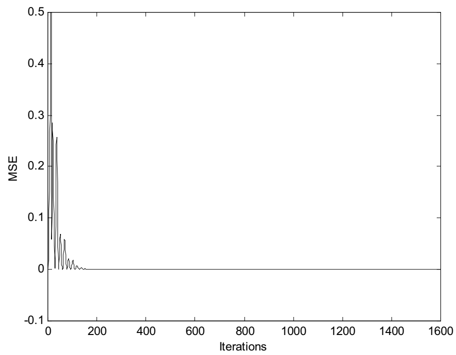

Fig. 1. Mean-square error of training.

4. Computer simulations

In this section, the illustrative example is provided to demonstrate the performance of the proposed neural predictive control. The example also shows the effect of setpoint changes and load disturbances on the control systems employing the proposed method.

A simulated laboratory scale liquid-level system of Sales and Billings [20] is considered. The simulated system is composed of a DC pump to feed water into a conical flask that, in turn, feeds square tank, giving the system second-order dynamics. The input is the voltage to the pump motor and the system output is the height of the water in the conical flask. The aim, under simulation conditions, is to follow some demanding trajectory for the water level. The process model was identified as follows

$$
\begin{aligned}
y(k) = {} & 0.9722\,y(k-1) + 0.3578\,u(k-1) - 0.1295\,u(k-2) \\
& - 0.3103\,y(k-1)u(k-1) - 0.04228\,y^2(k-2) \\
& + 0.1663\,y(k-2)u(k-2) - 0.03259\,y^2(k-1)y(k-2) \\
& - 0.3513\,y^2(k-1)u(k-2) + 0.3084\,y(k-1)y(k-2)u(k-2) \\
& + 0.1087\,y(k-2)u(k-1)u(k-2)
\end{aligned}
\tag{26}
$$

The simulation was performed for two sets of reference inputs $r(k)$ specified by

$$
r(k) = \begin{cases} 1, & 0 < k \le 400 \\ 0, & 400 < k \le 800 \end{cases}
$$

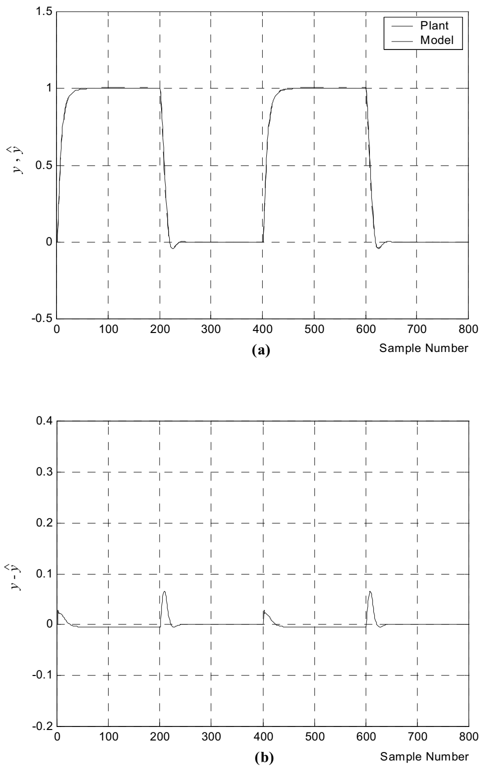

Fig. 2. System model validation.

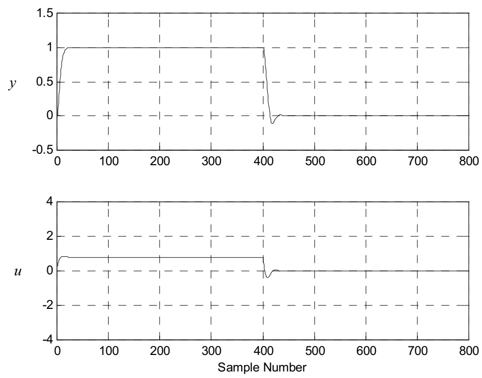

Fig. 3. Performance of the proposed controller.

(a) Setpoint tracking response. (b) Control signal.

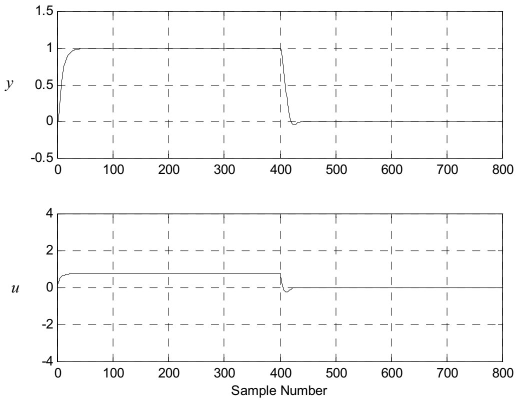

Fig. 4. Performance of the PID controller.

(a) Setpoint tracking response. (b) Control signal.

From the process model (26), it clearly indicates that the input vector of RNNM, the output order, the input order, and the time-delay can be easily specified by $X(k) = [y(k-1)\ y(k-2)\ u(k-1)\ u(k-2)]^T$, $n_y = 2$, $n_u = 2$, and $d = 1$. The key parameter of the RNNM is chosen as $n_j = 10$, which were shown effective under computer simulations.

Fig. 1 illustrates the mean-square error (MSE) curves of the training. Obviously, the ALR algorithm gives faster training to the neural network within 1600 iterations. The corresponding resulting step response of the model validation test is shown in Fig. 2.

The setpoint tracking responses from a proposed controller ($N_p = 40$, $\lambda = 50$) and a velocity-type PID controller ($K_p = 1$, $K_I = 0.2$ and $K_D = 0.01$) are given in Figs. 3-4. Figs. 3-4 shows that the NPC and PID control systems have good response in the absence of external loads.

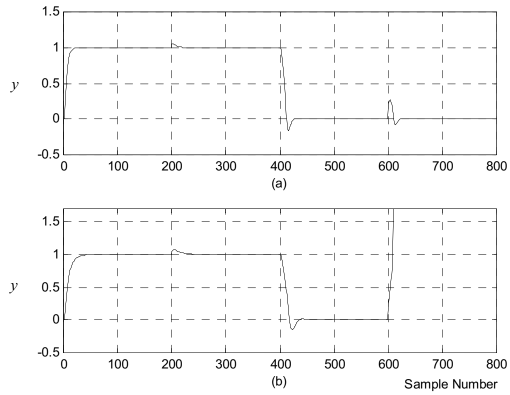

Fig. 5. Results for setpoint tracking.

(a) Performance of the proposed controller.

(b) Performance of the PID controller.

In order to investigate disturbance rejection capability of the proposed controller, the mathematical model (26) was perturbed by a disturbance $v(k)$, where $v(k) = 0.05$ for $200 \le k < 600$ and $v(k) = 0.2$ for $k \ge 600$. Fig. 4(a) shows the simulation results, which reveal that the proposed controller demonstrates a good disturbance rejection capability. Fig. 5(b) discloses that PID controller has an unstable tracking performance. The results in Fig. 4 reveal the usefulness of the proposed controller, especially, for this class of nonlinear process systems.

5. Conclusions

This paper has presented a systematic design methodology for developing a discrete-time neural predictive controller for a class of nonlinear systems. Both ALR for the RNNM and AOR for NPC are chosen based on the Lyapunov stability theory. The proposed real-time control algorithm has been successfully applied to achieve step tracking performance specifications for the nonlinear plant. Through the simulative results, the proposed control method has been proven effective in controlling a class of nonlinear systems.

References

[1]E. F. Camacho and C. Bordóns, *Model Predictive Control.* New York: Springer-Verlag, 1999.

[2]J. M. Maciejowski, *Predictive Control with Constraints.* London: Prentice Hall, 2002.

[3]D. W. Clarke, "Application of generalized predictive control to industrial process," *IEEE Contr. Syst. Mag.*, vol. 8, pp. 49-55, 1988.

[4]S. Roy, O. P. Malik, and S. G. Hope, "A *k*-step predictive scheme for speed control of diesel driven power plants," *IEEE Trans. Ind. Applica.*, vol. 29, no. 2, pp. 389-396, 1993.

[5]C. H. Lu and C. C. Tsai, "Adaptive predictive control with recurrent neural network for industrial processes: an application to temperature control of a variable-frequency oil-cooling machine," *IEEE Trans. Ind. Electron.*, vol. 55, no. 3, pp. 1-10, 2008.

[6]A. Casavola, M. Giannelli, and E. Mosca, "Global predictive regulation of null-controllable input-saturated linear system," *IEEE Trans. Automat. Contr.*, vol. 44, pp. 2226-2230, 1999.

[7]F. Allgower and A. Zheng, *Nonlinear model predictive control*. Boston: Birkhauser Verlag, 2000.

[8]B. Kouvaritakis and M. Cannon, *Nonlinear predictive control – theory and practice*. London: IEE, 2001.

[9]C. H. Lu and C. C. Tsai, "Generalized predictive control using recurrent fuzzy neural networks for industrial processes," *Journal of Process Control,* vol. 17, pp. 83-92, 2007.

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

[11]S. Jagannathan, “Control of a class of nonlinear discrete-time systems using multiplayer neural networks,” *IEEE Trans. Neural Networks*, vol. 12, no. 5, pp. 1113-1120, 2001.

[12]S. S. Ge, J. Zhang, and T. H. Lee, “Adaptive neural network control for a class of MIMO nonlinear systems with disturbances in discrete-time,” *IEEE Trans. Syst., Man, Cybern. B*, vol. 34, no. 4, pp. 1630-1645, 2004.

[13]R. J. Wai and F. J. Lin, “Adaptive recurrent-neural-network control for linear induction motor,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. 37, no. 4, pp. 1176-1191, 2001.

[14]A. G. Parlos, A. Parthasarathy, and A. F. Atiya, “Neural-predictive process control using on-line controller adaptation,” *IEEE Trans. Contr. Syst. Technol.*, vol. 9, no. 5, pp. 741-755, 2001.

[15]X. Li, Z. Chen and Z. Yuan, “Simple recurrent neural network-based adaptive predictive control for nonlinear systems,” *Asian Journal of Control*, vol. 4, no. 2, pp. 231-239, 2002.

[16]S. J. Yoo, Y. H. Choi, and J. B. Park, “generalized predictive control based on self-recurrent wavelet neural network for stable path tracking of mobile robots: adaptive learning rates approach,” *IEEE Trans. Circuits and Systems- I*, vol. 53, no. 6, pp. 1381-1394, 2006.

[17]C. C. Ku and K. Y. Lee, “Diagonal recurrent neural networks for dynamical system control,” *IEEE Trans. Neural Networks*, vol. 6, no. 1, pp. 144-156, 1995.

[18]D. W. Clarke and C. Mohtadi, “ Properties of generalized predictive control,” *Automatica*, vol. 25, no. 6, pp. 859-875, 1989.

[19]K. J. Åström and B. Wittenmark, *Adaptive Control.* New Jersey: Addison-Wesley, 1995.

[20]K. R. Sales and S. A. Billings, “Self-tuning control of nonlinear ARMAX model,” *Int. J. Control*, vol. 51, pp. 753-769, 1990.