Direct Adaptive Control of Process Control Benchmark Using Dynamic Recurrent Neural Network

Mohamed A. Hussien(✉), Tarek A. Mahmoud, and Mohamed I. Mahmoud

Faculty of Electronic Engineering,
Industrial Electronics and Control Engineering Department,
Menoufia University, Shibin Al Kawm, Egypt
mohamed.hussien@el-eng.menofia.edu.eg

Abstract. In this article, we develop a direct adaptive control scheme based on Dynamic Recurrent Neural Network (DRNN) for a process control benchmark. The DRNN is represented in a general nonlinear state space form for producing the control action that force the system output to a desired trajectory. The control algorithm can be implemented without a priori knowledge of the controlled system. Indeed, the weights of the DRNN controller are adjusted on-line using the truncated Back Propagation Through Time (BPTT) method. Unlike the approaches in the literature, the learning signal of the network weights is generated by a control error estimator stage in the developed controller. Finally, the developed controller is applied to a laboratory flow control system with two experimental scenarios.

Keywords: Direct adaptive control · Dynamic recurrent neural network · State space neural network · Flow control system

1 Introduction

During the past decades, Neural Networks (NNs) have been become attractive paradigms in the modeling and control of nonlinear processes. In literature, the structure of NNs is classified as feed forward and recurrent. The feed forward structure uses static discrete-time models that capture the dynamics of the real process through the use of tapped-delay lines in the model inputs and outputs. The feed forward neural networks have been developed in the direct inverse control [1,2], the internal model-based control(IMC) [3–5] and predictive control [6–11]. They have introduced improved modeling performance than linear models and as a result they are able to achieve better control performance of nonlinear systems. However, the feed forward neural networks suffer from number of drawbacks. In the identification of complex dynamic systems, the feed forward networks are unable to identify time dependent nonlinear dynamics with

A.E. Hassanien et al. (eds.), *Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2016*, Advances in Intelligent Systems and Computing 533, DOI 10.1007/978-3-319-48308-5_27

high accuracy [12]. Besides, these networks suffer from large number of neurons, number of required delays and the weight updates do not use the internal neural network information.

To address these issues, the second structure of NNs, the dynamic recurrent NNs (DRNNs) have been introduced for the identification and control of nonlinear systems such as [13–17]. The advantage of the dynamic recurrent neural networks (DNN) is threefold with respect to static networks [18]. First, the DRNN has self-loops and backward connections to memorize past information, hence it can capture the dynamic of the system. Second, the number of the network parameters is considerably lower. It is only necessary to identify the dimension of the state space, since the number of inputs and outputs is specified by their counterparts in the real process. Third, they can be represented in the state-space form, which is more suitable to most control schemes [19].

Motivated by the aforementioned review, this work develops a direct adaptive control scheme using the dynamic recurrent neural network for a practical system. In this scheme, DRNN is designed to represent the control action the make the system output to track a desired action. The network structure is implemented without any information about the controlled system. In the online stage, the network parameters are adjusted by using the truncated BPTT algorithm. Likewise to [20], a control error estimator is introduced to the control algorithm to estimate the learning error needed for the DRNN weights adaptation. The bounded input bounded output stability of the DRNN controller is discussed. Finally, the control algorithm is applied in the control of a process control laboratory system which is a real-life installation. In the real process, the problem of control is much more harder to achieve in the presence of noise and disturbances. The obtained results of the process control system depicted that the DRNN controller can be apply to the real life system with acceptable performance despite the external disturbance. Moreover, as the process works in the real-time, the computation complexity is also an important issue, because there are hard time constraints imposed on the control algorithm. The work shows that software implementation of the developed control method is possible, and the methods can be practically used in real industrial process with the sampling time equal to 0.22 s.

The rest of the paper is structured as follows. In Sect. 2, the description of the DRNN in the state space form is briefly provided. Section 3 develops the proposed direct adaptive control scheme based on the DRNN model. The benchmark system description and the control results are given in Sect. 4. Finally, Sect. 5 provides a concluding summary of this work.

2 Dynamic Recurrent Neural Network

In this section, we describe the structure of the DRNN which is investigated as a direct controller in this article. Let's consider a fully recurrent neural network with n neurons and m inputs as depicted in Fig. 1. Define $y(k) \in \Re$, $x(k) = [x_1(k), x_2(k), .., x_n(k)]^T \in \Re^n$ and $u(k) = [u_1(k), u_2(k), ..., u_m(k)]^T \in \Re^m$ the

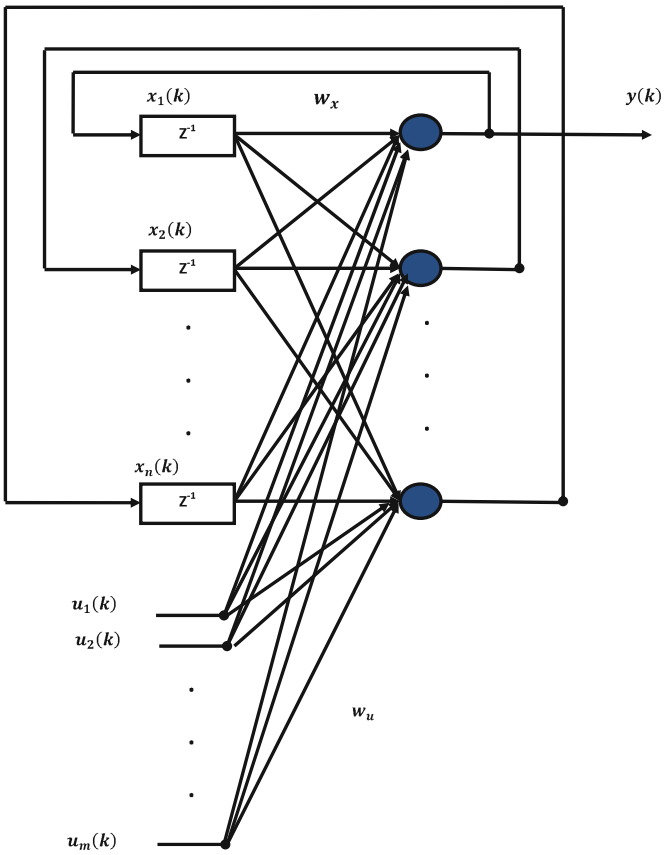

Fig. 1. The DRNN structure (where Z^{-1} denotes to the back shift operator)

network output, the hidden neuron outputs, and the external inputs at time step k respectively; $w_x \in \Re^{n \times n}$ is the matrix of the weights of the feedback connection of the hidden neurons; $w_u \in \Re^{n \times m}$ is the matrix of the weight connections between the external input and the hidden neurons. The equation of the DRNN model at time step k in the form of the state space equations is described by:

$$\begin{aligned} x(k) &= F(w_x x(k-1) + w_u u(k)) \\ y(k) &= C^T x(k) \end{aligned} \tag{1}$$

where $F(.)$ stands for the vector valued activation function of the hidden neurons and $C \in \Re^n$ is the vector of the output connection weights. Typically, the hyperbolic tangent activation function is selected giving well modeling results. In this work, the output weight vector is given as $C = [1, 0, ..., 0]^T$.

In control system applications, the NNs can be used in the controllers in either indirect or direct control schemes. In the former scheme, NN is designed to model the system dynamics. In such a way, the neural network model can

be implicitly used to compute the control action that satisfies the controller's design specifications. In the direct scheme, NN is employed to produce the control action that force the system states to its target states. In the following section, a proposed direct adaptive control scheme based on the DRNN will be given.

3 Direct Adaptive Control Scheme Based on DRNN

Consider a single-input single output discrete system defined by:

$$y_p(k) = f(y_p(k-1), y_p(k-2), ..., y_p(k-n), u(k), u(k-1), ..., u(k-m)) \quad (2)$$

where u, y_p denote to the input and the output, respectively, k is the discrete time index, $n, m > 0$, and $f :\in \Re^{n+m} \rightarrow \Re$. The main objective of this work is to design a direct adaptive DRNN controller such that the plant output described in Eq. (2) tracks a specified reference output y_d such that the function $f(.)$ is unknown. Consider a DRNN can be designed to produce the control action $u(k)$ such that the system output y_p match a desired output y_d. The internal network states $x(k)$ and the output are:

$$\begin{aligned} x(k) &= F_u(w_x x(k-1) + w_e E(k)) \\ u(k) &= x_1(k) \end{aligned} \quad (3)$$

where $e(k) = y_d(k) - y_p(k)$ is the tracking error, $E(k) = [e(k), e(k-1), ..., e(k-m)]^T$, $x(k) = [x_1(k), x_2(k), ..., x_n(k)]^T$, and F_u is the vector of a symmetric tanch activation function of the DRNN controller.

3.1 Weights Update

In this subsection, we present the adaptation law of the DRNN controller weights in the proposed scheme. The learning law of the DRNN network weights w_x and w_e is derived through the minimization of a suitable cost function. Despite the training of the static neural networks, the cost function for training the DRNN is a time varying error. The back-propagation through time (BPTT) algorithm is the basic method for the dynamic recurrent neural networks learning [18,21]. In fact, the stander algorithm of the BPTT is more theoretical than practical interest because it makes use of potentially unbounded history storage. Therefore, more practical extension methods to the BPTT have been developed such as the truncated BPTT methods for the on-line training phase [18]. Define the following error for the DRNN controller:

$$e_u(k) = u_d(k) - u(k) \quad (4)$$

where $u_d(k)$ is the desired control action the make the system output $y_p(k)$ to track a desired signal $y_d(k)$. Then, the cost function of the DRNN can be given by:

$$E_u(k) = \frac{1}{2} e_u^2(k) \quad (5)$$

In order to describe the details of the on-line training of the DNNN controller using the truncated BPTT method, merge $E(k)$ and $x(k)$ to form a vector $Z(k) \in \Re^{n+m}$ as $Z(k) = [e(k), e(k-1), ..., e(k-m), x_1(k), x_2(k), ..., x_n(k)]^T$. Also, all the weights of the network w_e and w_x can be collected into a single matrix $W \in \Re^{n\times(n+m)}$. Thus, for $l = 1, 2, ..., n$ and $i = 1, 2, ..., n+m$, the network equations becomes:

$$\begin{aligned} S_l(k) &= \sum_{i=1}^{n+m} w_{li} z_i(k) \\ x_l(k) &= f_u(S_l(k)) \\ u(k) &= x_1(k) \end{aligned} \tag{6}$$

where w_{li} represents the connection weight to the l^{th} state unit (i.e., $x_l(k)$) from the i^{th} unit of of the input (i.e., $z_i(k)$), and $f_u(.)$ is the tanch function. The learning law of the network weights W can be derived through the minimization of the cost function defined in (5) using the truncated BPTT method. Then, the on-line updating rule of any particular weight w_{li} of the DRNN controller can be calculated by

$$\Delta w_{li} = -\eta \sum_{k-h+1}^{t} \frac{\partial E_u(k)}{\partial w_{li}(t)} \tag{7}$$

where η is the learning rate, h is the past history considered in the calculation and must be fixed and chosen longer for better. We can calculate the term $\frac{\partial E_u(k)}{\partial w_{li}(t)}$ as:

$$\frac{\partial E_u(k)}{\partial w_{li}(t)} = \frac{\partial E_u(k)}{\partial S_l(t)} \frac{\partial S_l(k)}{\partial w_{li}(t)} = \frac{\partial E_u(k)}{\partial S_l(t)} z_i(t-1) \tag{8}$$

where

$$\frac{\partial E_u(k)}{\partial S_l(t)} = \frac{\partial E_u(k)}{\partial x_l(t)} \acute{f}_u(S_l(t)) \tag{9}$$

and

$$\frac{\partial E_u(k)}{\partial x_l(t)} = \begin{cases} -e_u(t) & \text{if } t = k, \\ \sum_{j=1}^{n} \frac{\partial E_u(k)}{\partial S_l(t)} w_{li} & \text{if } t < k. \end{cases} \tag{10}$$

Define $\delta_l(t)$ as

$$\delta_l(t) = \begin{cases} -e_u(t) \acute{f}_u(S_l(t)) & \text{if } t = k, \\ \acute{f}_u(S_l(t)) \sum_{i=1}^{n} \delta_l(t+1) w_{li} & \text{if } t < k. \end{cases} \tag{11}$$

Finally, the on line adaptation rule for the DRNN controller is

$$w_{li}(new) = w_{li}(old) + \eta \sum_{k-h+1}^{t} \delta_l(t) z_i(t) \tag{12}$$

Due to the desired control action $u_d(k)$ is not available, the learning error defined in (4) cannot be obtained. Then, an estimated control error expressed by $\hat{e}_u$ can be introduced in the adaptive law of the DRNN controller (11) and (12) as in the following subsection.

3.2 Learning Error Estimator

In [20], a simple method was proposed to estimate the learning error $\hat{e}_u$ such that it can be estimated directly as:

$$\hat{e}_u(k) = k_e e(k) + k_c \Delta e(k) \tag{13}$$

where $e(k) = y_d(k) - y_p(k)$ is the tracking error, $\Delta e(k)$ is the change of this error, k_e and k_c are positive small values. Finally we can write:

$$\hat{e}_u(k) = G e_u(k) \tag{14}$$

where G is a positive scaling factor that will be included in the learning rate η. Consequently, the adaptive laws (11) and (12) of the controller's weights can be rewritten as:

$$\delta_l(t) = \begin{cases} -\hat{e}_u(t) \acute{f}_u(S_l(t)) & \text{if } t = k, \\ \acute{f}_u(S_l(t)) \sum_{i=1}^{n} \delta_l(t+1) w_{li} & \text{if } t < k. \end{cases} \tag{15}$$

$$w_{li}(new) = w_{li}(old) + \eta \sum_{k-h+1}^{t} \delta_l(t) z_i(t) \tag{16}$$

3.3 Convergence Analysis

Here, we discuss the bounded input bounded output (BIBO) stability of the DRNN controller. To investigate the BIBO stability of the DRNN controller, consider the network states has the following linear form:

$$x(k) = w_x x(k-1) + w_e E(k) \tag{17}$$

Let λ_{max} as the largest absolute eigenvalue of w_x and according to the linear system theory, if $|\lambda_{max}| < 1$, the linear system defined in (17) will be bounded. For different eigen values of w_x, the linear system (17) has different transient properties. Thus, for the DRNN defined in (3) with the tanch activation function, we have

$$\|tanch(w_x x(k-1) + w_e E(k))\| \leq \|w_x x(k-1) + w_e E(k)\| \tag{18}$$

Therefore, the DRNN defined in (17) with the tanch activation function and the condition that the largest absolute eigenvalue of w_x is smaller than 1, we can say that the DRNN output is bounded for all inputs.

To summarize, Fig. 2 describes the overall diagram of the proposed direct adaptive control scheme based on the DRNN.

4 Experimental Results

4.1 Benchmark System

The considered benchmark system is the Process Control System (PCS) developed at Research Laboratory in Industrial Electronics and Control Engineering Department, Faculty of Electronic Engineering, Menoufia University, Egypt. The system consists of two water tanks and one pump as illustrated in Fig. 3. In turn, the PCS configuration diagram is shown in Fig. 4. The pump allows to move the water from the lower tank to the upper one through the valve V1 or from the lower tank to itself through the valve V2. Under the influence of gravity, the water is automatically transfered from the upper tank to the lower one through V3 to ensure a water supply. In the upper tank, an ultrasonic sensor is installed to measure the water level. A pressure sensor is used to measure the pressure in the pipes. There is also a flow sensor to measure the water flow after pump. Different water loops can be implemented by a set of different types of valves (solenoid and proportional ones). Moreover, a heater installed in the lower tank and a temperature sensor to measure the temperature when water circulates through the pipes. The specifications of the process parts are shown in Table 1.

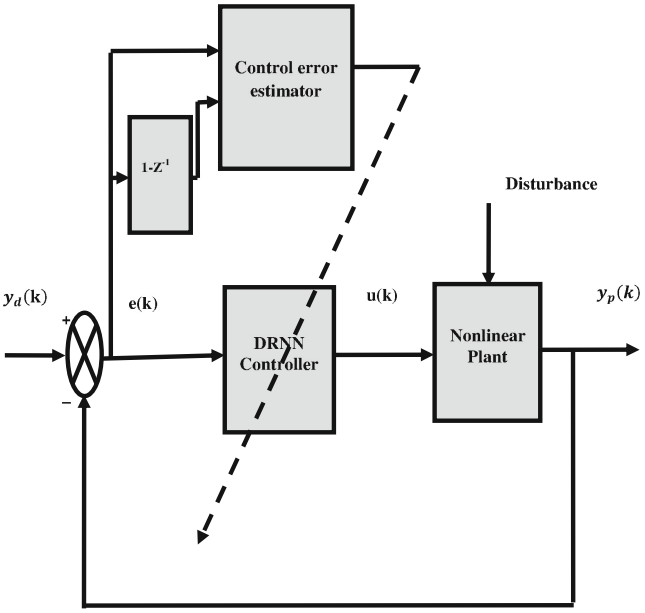

Fig. 2. The proposed control structure

Fig. 3. The laboratory installation actual view

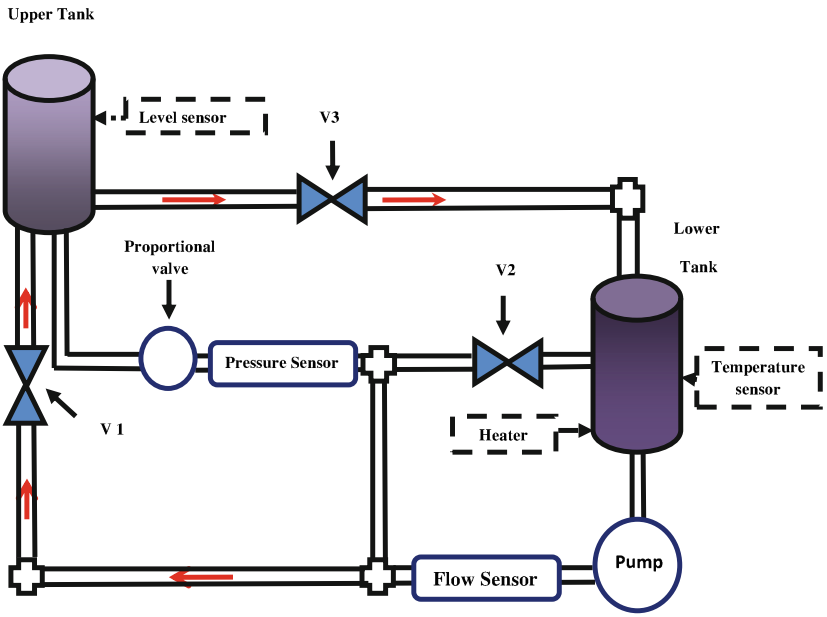

Fig. 4. PCS configuration diagram

4.2 Control Results

The benchmark system allows for many different configurations. It can be configured as different simple of SISO (Single Input Single Output) systems based on the variable to be controlled. In this work, the flow control is selected where

Table 1. The specifications of the process components

The component	Specifications
Pump	Pressure 2.8 bar : Flow 12 L/min
Flow sensor	Flow range : [2 16] L/min
Pressure sensor	Pressure range : [0 10] bar
Ultrasonic sensor	Sensing range : [60 300] mm
Solenoid valves (V1,V2,V3)	Operating pressure range is 7 bar
Proportional valve	Operating : [0 8] bar

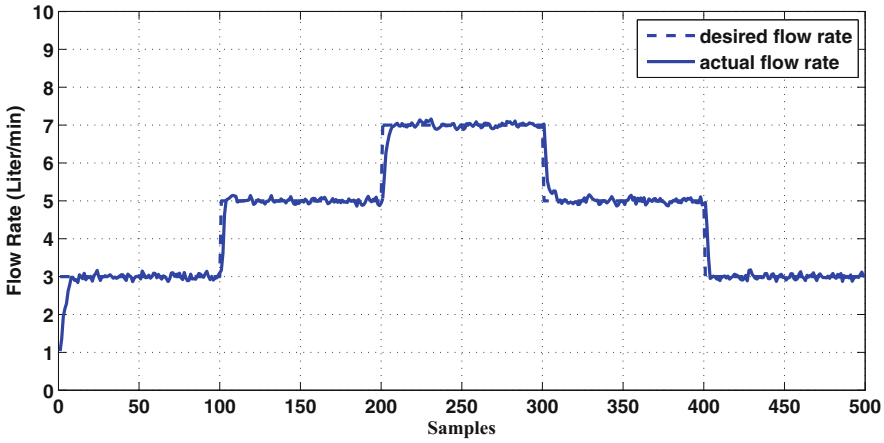

Fig. 5. The flow rate of the PCS using the proposed direct adaptive control scheme (experimental task 1)

the flow sensor signal is the input of the control system and the control effort is applied to the motor pump. When the pump is running, it realizes two tasks, speeding up the flow of water to the upper tank and also makes the water circulation in the outer loop. Herein, the controller aims to maintain a constant flow of water from the lower tank to the upper tank regardless of the water amount in the tanks. In the closed loop system, the process control system is interfaced with a personal computer (PC) through a data acquisition card (NI-CDAQ-9171) which receives the input values (0 to 9V) from the computer and transmits it to the system with sampling period of 0.22 s. An amplifier is used to convert the sampled-input signal from the data acquisition card to variable DC voltage from 0 to 24 V and fed it to the motor pump. Thus, PC (Processor Intel core i3, CPU 3.06 GHz, Ram 2.00 GB, operating System 32 bit Windows 7) that runs Lab view program code is adopted as the controller. During the experiments, the valves V1 and V3 were permanently opened to continuously create connection between the two tanks, and the valve V2 was permanently closed.

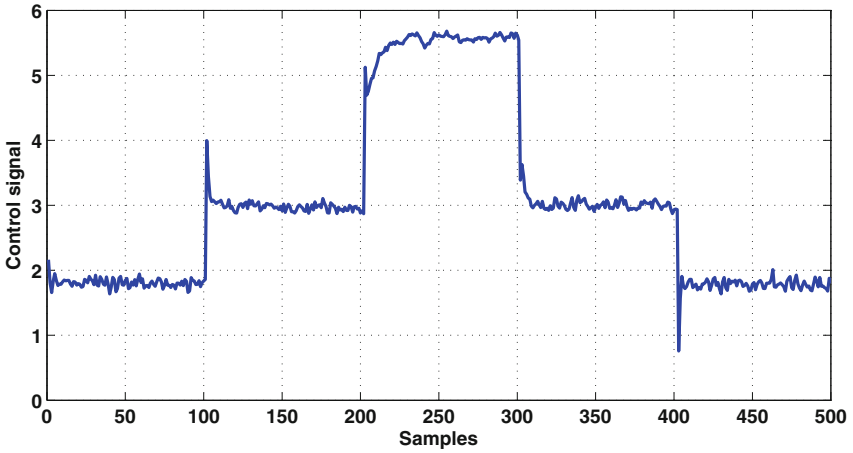

Fig. 6. Change of the control signal applied to the amplifier of the motor pump (experimental task 1)

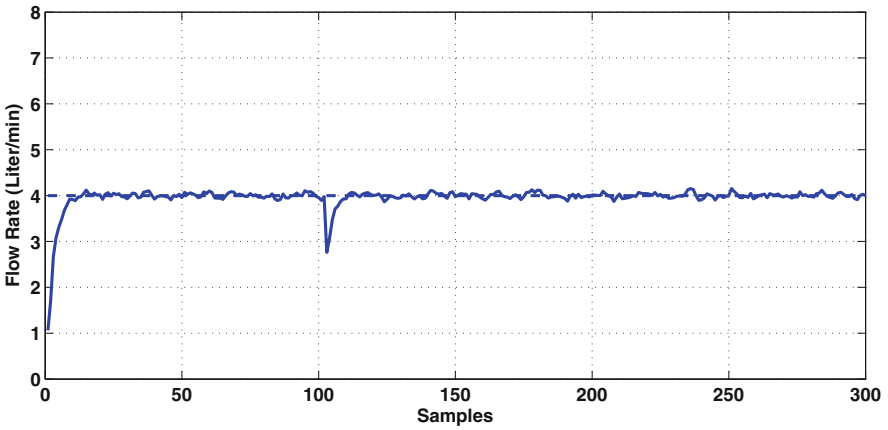

Fig. 7. The flow rate of the PCS using the proposed direct adaptive control scheme (experimental task 2)

Table 2. Comparison of RMSE index for the proposed controller, conventional PI and fuzzy PI controllers

Controller	Experimental task (1)	Experimental task (2)
The proposed adaptive DRNN controller	0.2649	0.2652
PI controller	0.6288	0.5490
Fuzzy PI controller	0.3085	0.2738

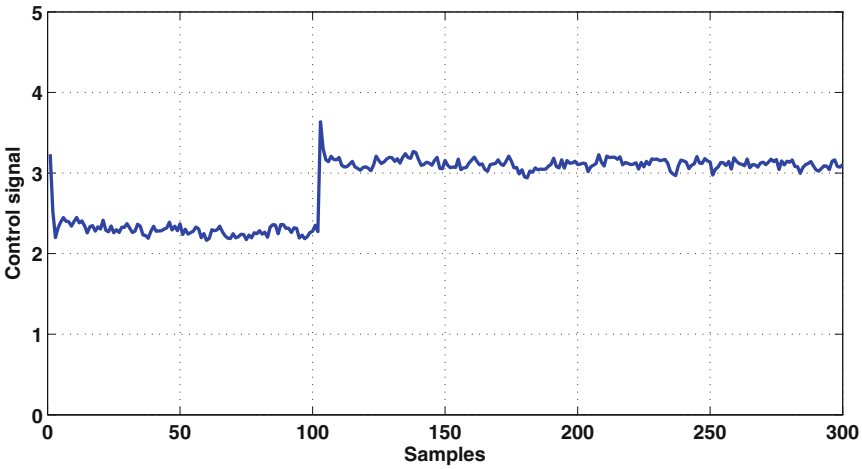

Fig. 8. Change of the control signal applied to the amplifier of the motor pump (experimental task 2)

In order to apply the proposed direct control scheme to the flow control system, several parameters should be selected priori such as the network parameter structure (i.e., n and m), the initial values of the network weights (i.e., w_e and w_x), and the parameters of the control error estimator parameters (i.e., K_e and K_c). In this study, we have employed DRNN controller with $n = 3$, and $m = 1$. When we have increased both n and m values, no further improvements in the control performance have been observed. Actually, the initial values of DRNN controller weights are of no importance as they are tuned on line and they eventually converge to their suitable values. Thus, w_e and w_x were initialized at small random values. Also, the two parameters k_e and k_c were selected as small positive values to avoid the output oscillation and overshoots (i.e., $k_e = 0.5$ and $k_c = 0.35$).

Two experimental tasks were achieved to investigate the performance of the developed controller. In the first one, we have investigated the controller performance with the set point changes which was assumed as:

$$y_d(k) = \begin{cases} 3 & \text{if } k < 100, \\ 5 & \text{if } 100 \leq < k < 200, \\ 6 & \text{if } 200 \leq < k < 300, \\ 5 & \text{if } 300 \leq < k < 400, \\ 3 & \text{if } k \geq < 400. \end{cases} \tag{19}$$

where k is the sampling instant. Figure 5 illustrates the change of flow rate according to the desired flow defined in (19). The obtained result of the proposed controller showed that it can achieve acceptable tracking performance with

set-point changes. Moreover, the control effort applied to the amplifier of the motor pump is smooth as depicted in Fig. 6.

During the last experiment, the controller performance was tested when the flow process control system was corrupted by external disturbance. An external disturbance, 30% of the set-point, was added to the system and the set point was set to 4 liter/min. Figures 7 and 8 show the obtained flow rate and the control signal for the second experimental task, respectively. Despite the presence of the external disturbance, the results of this task ensure that the controller algorithm can follow the set-point accurately with a very small steady state error and smooth control effort.

Moreover, the performance of the proposed DRNN adaptive controller can be verified using the following Root Mean Squared Error (RMSE) performance index:

$$RMSE \sqrt{\frac{1}{N} \sum_{k=1}^{k=N} (y_d(k) - y_p(k))^2} \tag{20}$$

where N is the number of samples, y_d and y_p are the desired and the actual flow rate, respectively. For the comparison reason, a conventional PI and a fuzzy PI controllers are tested in the same platform for the two experimental tasks. Table 2 provides comparative results of the proposed controller and both PI and fuzzy PI controllers. These comparative results reavel that the overall performance of the proposed DRNN controller for the two experimental scenarios is better to the classical PI and the fuzzy PI controllers. The aforementioned experimental results show that the efficancy of the direct adaptive DRNN controller method in controlling a real time application.

5 Conclusion

In this paper, a direct adaptive control using a dynamic recurrent neural network for a real time application is developed. The developed control scheme design has the following features: (1) The configuration parameters of the DRNN controller used in the controller method is considerably lower. Thus, the proposed scheme is more suitable for real time implementation. (2) The DRNN controller is implemented without a priori knowledge of the controlled and the off line training of the network is not required. The experimental results illustrated that the developed DRNN controller method can perform good tracking. Furthermore, a successful control and a desired performance can be obtained in the presence of external disturbances.

References

1. Cabrera, J., Narendra, K.: Issues in the application of neural networks for tracking based on inverse control. IEEE Trans. Autom. Control **40**(11), 2007–2027 (1999)
2. Hussain, M.A., Kershenbau, L.: Implementation of neural network-basedinverse-model control strategies on an exothermic reactor. Sci. Asia **27**, 41–50 (2001)

3. Awais, M.: Application of internal model control methods to industrial combustion. Appl. Soft Comput. **5**(2), 223–233 (2005)
4. Chidrawar, S., Patre, B.: Implementation of neural network for internalmodel control and adaptive control. In: Proceedings of the International Conference on Computer and Communication Engineering, Malaysia (2008)
5. Deng, H., Xu, Z., Han-Xiong, L.: A novel neural internal model control formulti-input multi-output nonlinear discrete-time processes. J. Process Control **19**, 1392–1400 (2009)
6. Yu, D.L., Yu, D.W., Gomm, J.B.: Neural model adaptation and predictive control of a chemical process rig. IEEE Trans. Control Syst. Technol. **14**(5), 828–840 (2006)
7. Kittisupakorn, P., Thitiyasook, P., Hussain, M.A., Daosud, W.: Neural network based model predictive control for a steel pickling process. J. Process Control **19**(4), 579–580 (2009)
8. Peng, H., Wu, J., Inoussa, G., Deng, Q., Nakano, K.: Nonlinear system modelingand predictive control using the RBF nets-based quasi-linear ARX model. Control Eng. Pract. **17**(1), 59–66 (2009)
9. Tiwari, S., Naresh, R., Jha, R.: Neural network predictive control of UPFC for improving transient stability performance of power system. Appl. Soft Comput. **11**(8), 4581–4590 (2011)
10. Nikdel, N., Nikdel, P., Badamchizadeh, M.A., Hassanzadeh, I.: Using neuralnetwork model predictive control for controlling shape memory alloybasedmanipulator. IEEE Trans. Ind. Electron. **61**(3), 1394–1401 (2014)
11. Yan, Z., Wang, J.: Robust model predictive control of nonlinear systems withunmodeled dynamics and bounded uncertainties based on neural networks. IEEE Trans. Neural Netw. Learn. Syst. **25**(3), 457–469 (2014)
12. Chun-Fei, H., Chin-Min, L., Ang-Bung, T., Chao-Ming, C.: Adaptive control for MIMO uncertain nonlinear systems using recurrent wavelet neural network. Int. J. Neural Syst. **22**(1), 37–50 (2012)
13. Ku, C.C., Lee, K.Y.: Diagonal recurrent development neural networks for dynamic systems control. IEEE Trans. Neural Netw. **6**(1), 144–156 (1995)
14. Pan, Y., Wang, J.: Model predictive control of unknown nonlinear dynamical systems based on recurrent neural networks. IEEE Trans. Ind. Electron. **58**(8), 3089–3101 (2011)
15. Fairbank, M., Li, S., Fub, X., Alonso, E., Wunsch, D.: An adaptive recurrentneural-network controller using a stabilization matrix and predictive inputs to solve a tracking problem under disturbances. Neural Netw. **49**, 74–86 (2014)
16. Xiao, L., Chao, J., De-Xin, L., Da-Wei, D.: Nonlinear adaptive control using multiple models and dynamic neural networks. Neurocomputing **136**, 190–200 (2014)
17. Hong, G., Lu, Z., Ying, H., June-Fei, Q.: Nonlinear model predictive control based on a self-organizing recurrent neural network. IEEE Trans. Neural Netw. Learn. Syst. **27**(2), 402–415 (2016)
18. Williams, R.J., Zisper, D.: An efficient gradient-based algorithm for on-line training of recurrent networks. Neural Comput. **2**(4), 490–501 (1990)
19. Zamarreno, J.M., Vega, P.: State-space neural network, properties and application. Neural Netw. **11**, 1099–1112 (1998)
20. Mahmoud, T.A., Elshenawy, L.M.: Echo state neural network based statefeedback control for SISO affine nonlinear systems. In: Proceedingsof 1st IFAC Conference on Modeling, Identification and Control of NonlinearSystems (MICNON-2015), Saint-Petersburg, Russia, 24-26 June 2015 (2015)
21. Pearlmutter, B.A.: Gradient calculations for dynamic recurrent neural networks: a survey. IEEE Trans. Neural Netw. **6**(5), 1212–1227 (1995)