Online Adaptive Control for Non Linear Processes Under Influence of External Disturbance

Nisha Jha *nishajha2010@gmail.com*
Department of Electronic Science
University of Delhi South Campus
New Delhi, 110021, India

Udaibir Singh *uday_mac2001@yahoo.co.in*
Department of Electronics
Acharya Narendra Dev College
University of Delhi
Govindpuri, Kalkaji, New Delhi, 110019, India

T.K. Saxena *tushyks@gmail.com*
National Physical Laboratory
Dr. K.S. Krishnan Road
New Delhi, 110 012, India

Avinashi Kapoor *avinashi_kapoor@yahoo.com*
Department of Electronic Science
University of Delhi South Campus
New Delhi, 110021, India

Abstract

In this paper a novel temperature controller, for non linear processes, under the influence of external disturbance, has been proposed. The control process has been carried out by Neural Network based Proportional, Integral and Derivative (NNPID). In this controller, two experiments have been conducted with respect to the setpoint changes and load disturbance. The first experiment considers the change in setpoint temperature in steps of 10oC from 50oC to 70oC for three different rates of flow of water. In the second experiment the load disturbance in terms of addition of 100ml/min of water at three different time intervals is introduced in the system. It has been shown that, in these situations, the proposed controller adjusts NN weights which are equivalent to PID parameters in both the cases to achieve better control than conventional PID. In the proposed controller, an error less than 0.08oC have been achieved under the effect of the load disturbance. Moreover, it is also seen that the present controller gives error less than 0.11oC, 0.12oC and 0.12oC, without overshoot for 50oC, 60oC and 70oC, respectively, for all three rate of flow of water.

Keywords: Neural Network Based PID (NNPID) Controller, Temperature Controller, Back-propagation Neural Network, Load Disturbance.

1. INTRODUCTION

Temperature control is an important factor in chemical, material and semiconductor manufacturing processes [1]-[3]. To design a general purpose temperature controller with good response time, smaller error and overshoot with load disturbance for the industrial implementation is still a challenge in the control research field. Over the past several years the on-off control and PID control schemes have been employed in commercial products with reasonable success.

A PID controller is the classical control algorithm in the field of process control. It still predominates in the process industries due to its robustness and effectiveness for a wide range of operating conditions and partly to its functional simplicity [4]. For the existing controllers, there

are three important parameters, namely, K_p, K_i and K_d which need to be evaluated [5]. The problem associated with the PID controller is to choose optimal value of these parameters so that the desired output is yielded for the appropriate process inputs. Usually, process engineers tune PID controller manually for an operation which, if done diligently, can take considerable time. Therefore, it is hard to establish an accurate dynamic model for a PID controller design. When the system has external disturbances, such as the variations of loads and changing process dynamics, then the transient response may go down. For this reason, free intelligent control schemes have gained the researcher attention.

In order to overcome the above disadvantages [4], [6], [7], researchers have proposed some adjusting rules for the self tuning controllers (STC) [8]-[19]. They have considerable potential for the process control problems since STCs provide a systematic and flexible approach for dealing with uncertainties, nonlinearities, and time varying parameters. A basic model structure for static nonlinearities is the back-propagation neural network (BPNN) [20]. The major advantages of BPNN over the traditional controller is that it can tune the three PID parameters on-line without requiring the prior knowledge of the mathematical model of different plants. Besides, the other advantages include its nonlinear mapping and self-learning abilities in various control processes, such as temperature control. It may be mentioned that the time varying and complex nonlinearity problems associated with PID controllers have been addressed by other researchers also using different algorithms [21], [22].

Neural Networks (NN) [23], which is the focus of the current work, is a better alternative to solve control engineering problems. It can be applied in two different ways: one is to use the NN to adjust the parameters of PID controller and the other is to use it as a direct controller. PID parameter values can also be adjusted by creating NN system based on the system output error signal [24]-[26], [27]-[30]. Prominent among them are the inverse model neuro-control approach by Widrow and Steams [29] and Psaltis, *et al*. [30] and further modified by other researchers [31]-[34].

In the present paper we have investigated two conditions viz the change in setpoint temperature and the load disturbance using Neural Network PID (NNPID) controller. In both the cases NN weights equivalent to PID parameters, are trained to achieve better control than existing conventional PID.

2. PROPOSED DESIGN APPROACH AND EXPERIMENTAL DETAILS

Fig.1 shows the block diagram of the proposed approach followed in the present work. According to this block diagram, the actuating error, T_{err}, can be expressed as

$$T_{err} = T_s - T_o \quad (1)$$

Where T_s and T_o are the setpoint temperature and observed temperature respectively and T_{err} is the error in terms of temperature.

The design of NNPID is shown in Fig. 2. It consists of three layers which are input layer, hidden layer and output layer. The input layer has two neurons represented by I_1 and I_2.The output layer

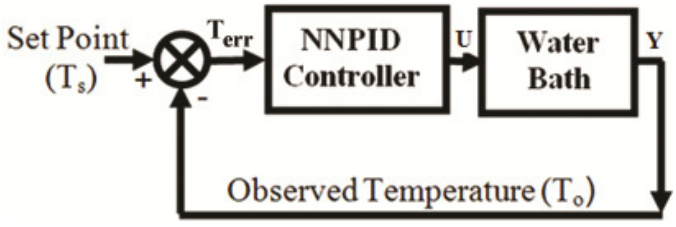

FIGURE 1: Block Diagram of the approach followed

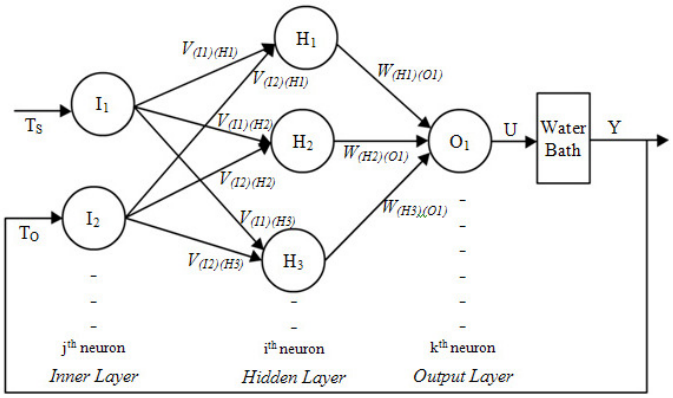

FIGURE 2: Neural Network tuning of PID Controller

has one neuron represented by O_1. The hidden layer has three neurons and they are symbolized as H1 (P-neuron), H2 (I-neuron) and H3 (D-neuron) respectively.

In the present case weights for the different layer combinations are taken as follows:
Weights between input layer and hidden layer are

$$V_{(I1)(H1)} = +1,\ V_{(I2)(H1)} = -1 \tag{2}$$

Weights between hidden layer and output layer are taken in terms of PID parameters as

$$W_{(H1)(O1)} = K_p,\ W_{(H2)(O1)} = K_i \text{ and } W_{(H3)(O1)} = K_d, \tag{3}$$

Then, input to hidden layer nodes are defined as

$$X_{(H1)} = V_{(I1)(H1)} I_1 + V_{(I2)(H1)} I_2 = T_S - T_O = T_{err} \tag{4}$$

$$X_{(H2)} = V_{(I1)(H2)} I_1 + V_{(I2)(H2)} I_2 = T_S - T_O = T_{err} \tag{5}$$

$$X_{(H3)} = V_{(I1)(H3)} I_1 + V_{(I2)(H3)} I_2 = T_S - T_O = T_{err} \tag{6}$$

where $X_{(H1)}$, $X_{(H2)}$ and $X_{(H3)}$ are the inputs of the hidden layer nodes.

The outputs of the hidden layer nodes are equal to their inputs, which can be expressed as function of proportional, integral and derivative as mentioned below:

$$Y_{(H1)} = T_{err} \tag{7}$$

$$Y_{(H2)} = \int T_{err}\, dt \tag{8}$$

$$Y_{(H3)} = \frac{dT_{err}}{dt} \tag{9}$$

Then, input to output layer becomes

$$X_{(O1)} = W_{(H1)(O1)} Y_{(H1)} + W_{(H2)(O1)} Y_{(H2)} + W_{(H3)(O1)} Y_{(H3)} \tag{10}$$

$$= K_p T_{err} + K_i \int T_{err}\, dt + K_d \frac{dT_{err}}{dt} \tag{11}$$

where $Y_{(H1)}$, $Y_{(H2)}$ and $Y_{(H3)}$ are output part of hidden layer nodes, and $X_{(O1)}$ is the input part of output layer.

Thus eq. (11) illustrates that PID parameters, which compared with weights as given in eq. (3), are tuned by using NNPID algorithm. It is well-known that most neural networks cannot be practically used in a controller because the initial connective weights of the neural networks are randomly selected. The randomized selection procedure imparts instability to the system. Therefore, it demands more experience to choose or tune PID parameters in order to ensure the stability. This can be achieved via training and learning capability of NNPID algorithm. The simple and prevalent algorithm which we have used in our work is BPNN algorithm [20] for weighting coefficients.

In the present controller, the main aim of the above algorithm is to minimize the error as given in eq. (12) in order to recover the system quickly from the effects of the external disturbance by tuning of PID parameters.

$$E_k = \frac{1}{2}[T_s - T_o]^2 \quad (12)$$

The weights of NNPID controller are adjusted by BPNN algorithm based on steepest descent on-line training process. It is done in terms of adjusted weights of hidden-to-output layer [$W_{(ik)}$]and input-to-hidden layer [$V_{(ij)}$] [35]. The increments of weight in hidden-to-output connection are updated by using the gradient descent method as

$$\Delta W_{ik}(n) = -\eta \frac{\partial E_k}{\partial W_{ik}} + \alpha \Delta W_{ik}(n-1) \quad (13)$$

$$= -\eta \left[\frac{\partial E_k}{\partial Y_{(ok)}}\right]\left[\frac{\partial Y_{(ok)}}{\partial X_{(ok)}}\right]\left[\frac{\partial X_{(ok)}}{\partial W_{(ik)}}\right] + \alpha \Delta W_{ik}(n-1) \quad (14)$$

where η and α are learning coefficient and momentum, respectively. Here the values of these terms are taken to be η = 0.005 and α = 0.5.

Further eq. (14) can be rewritten as [35]

$$\Delta W_{ik}(n) = -\eta\beta[T_k - Y_{(ok)}]Y_{(ok)}[1 - Y_{(ok)}]Y_{(hi)} + \alpha \Delta W_{ik}(n-1) \quad (15)$$

$$\Delta W_{ik}(n) = \eta \delta_k Y_{(hi)} + \alpha \Delta W_{ik}(n-1) \quad (16)$$

Where we have defined

$$\delta_k = \beta[T_k - Y_{(ok)}]Y_{(ok)}[1 - Y_{(ok)}] \quad (17)$$

Similarly, the incremental weights of input-to-hidden connection are updated as

$$\Delta V_{(ij)}(n) = -\eta \frac{\partial E_k}{\partial V_{(ij)}} + \alpha \Delta V_{(ij)}(n-1) \quad (18)$$

$$= -\eta \left[\frac{\partial E_k}{\partial Y_{(ok)}}\right]\left[\frac{\partial Y_{(ok)}}{\partial X_{(ok)}}\right]\left[\frac{\partial X_{(ok)}}{\partial Y_{(hi)}}\right]\left[\frac{\partial Y_{(hi)}}{\partial X_{(hi)}}\right]\left[\frac{\partial X_{(hi)}}{\partial V_{(ij)}}\right] + \alpha \Delta V_{(ij)}(n-1) \quad (19)$$

Now, eq. (19) can be rewritten as [35]

$$\Delta V_{(ij)}(n) = \eta \gamma_i \beta Y_{(hi)}[1 - Y_{(hi)}]I_{ij} + \alpha \Delta V_{(ij)}(n-1) \quad (20)$$

$$\Delta V_{(ij)}(n) = \eta \delta_k^* I_{ij} + \alpha \Delta V_{(ij)}(n-1) \quad (21)$$

Where we can define

$$\gamma_i = W_{ik}\delta_k \quad (22)$$

$$\delta_k^* = -\gamma_i \beta Y_{(hi)}[1 - Y_{(hi)}] \quad (23)$$

Now we use updated weights, $\Delta W_{(ik)}(n)$ and $\Delta V_{(ij)}(n)$ from eq. (16) and eq. (21) for finding new weights for hidden-to-output and input-to-hidden connections.

$$W_{ik}(n+1) = W_{ik}(n) + \eta \delta_k Y_{(hi)} + \alpha \Delta W_{ik}(n-1) \quad (24)$$

$$V_{(ij)}(n+1) = V_{(ij)}(n) + \eta \delta_k^* I_{ij} + \alpha \Delta V_{(ij)}(n-1) \quad (25)$$

The new weights are adjusted by updated weights as per eq. (24) and eq. (25) with iterations till we get the minimum mean square error in terms of temperature. Now these updated weights are employed for the experiment discussed below.

The schematic diagram of the experimental setup of the water bath temperature controller is shown in Fig. 3.

The hardware for controlling the temperature of the bath has been designed and fabricated around the Atmel microcontroller 89C51. The temperature of the bath is acquired with the help of platinum resistance thermometer (PRT). When the PRT is excited with a constant current source of 1mA current, it gives the output in voltage form. This voltage is then fed to the 4½ digit analog to digital converter (ADC). This digitized voltage is then sent to the personal computer (PC) by microcontroller 89C51through RS232C interface. The program in PC does the calculations using the NNPID algorithm. After doing the entire calculations microcontroller controls the TRIAC firing circuit and the firing angle for the required energy, through heater, to be given to the water bath. The NNPID program in PC continuously monitors the temperature and accordingly controls the same in the bath. In case it senses any change in the temperature, it automatically modifies the parameters of the temperature controller. The NNPID program in PC has been written in Visual

BASIC-5.0 language. The program stores the data in the user defined file as well as plots the online data in the form of graph on the screen. A specially designed varying environment is created by continuous flow of fresh water in such a way that the level of the water inside the bath remains constant even if the hot water is removed at random outflow rates. Uniform heat distribution is maintained using the circulator, and the isolated system is used to minimize external disturbance. The cooling is achieved at a constant rate using the refrigeration system of the bath.

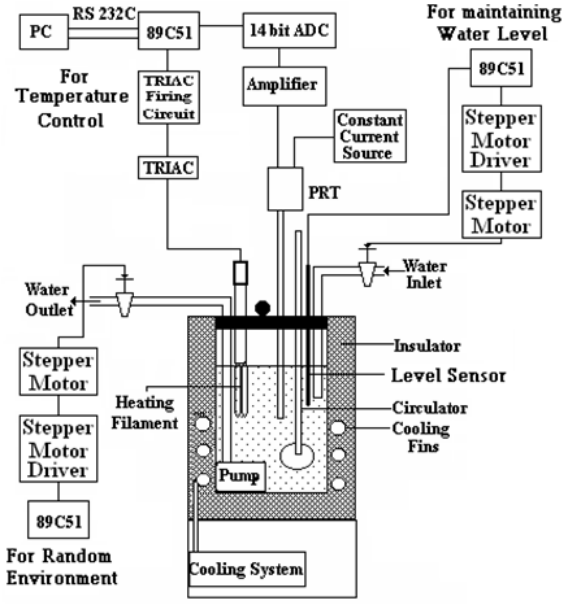

FIGURE 3: Block Diagram of the Experimental Setup

distribution is maintained using the circulator, and the isolated system is used to minimize external disturbance. The cooling is achieved at a constant rate using the refrigeration system of the bath.

3. EXPERIMENTAL AND SIMULATION RESULTS

In this paper two sets of experiments were conducted in the water bath. In the first set of experiments, the tracking performance of the two controllers i.e. NNPID controller and conventional PID controller with respect to setpoint changes are studied. In this system, further three set of experiments were conducted at three different flow of water i.e. at 100ml/min, 250ml/min and 500ml/min as shown in Figs. 4, 5 and 6 respectively. In these experiments the setpoint temperature of the water bath was increased in steps of $10^{o}C$ from $50^{o}C$ to $70^{o}C$ to investigate the effect of flow of water on temperature control at the different setpoint.

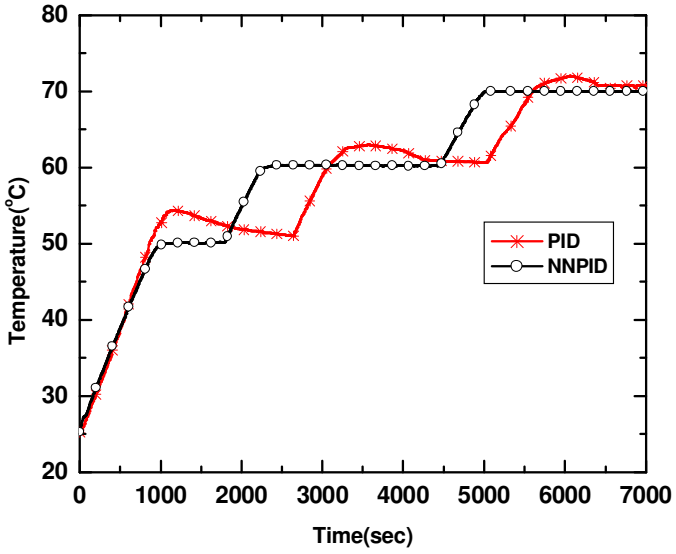

FIGURE 4: Showing the comparison of NNPID controller with the conventional PID controller of a water bath for 100 ml/min flow rate of water with respect to setpoint changes.

The simulation results subjected to the changes in setpoint for different flow rate of water are shown in Figs. (4-6). The three systems are categorized in terms of change in flow rate of water are shown in Table I. The settling time taken by NNPID and PID controllers to achieve target temperatures of 50°C, 60°C and 70°C for different flow rates of water are given in Table II. According to this table, when we refer Figs. (4-6), we infer that NNPID controller gives better performance in respect of less settling time as compared to the conventional PID controller in achieving change in setpoint temperature. Hence the experimental and simulation results of these systems show the simplicity, reliability and robustness of NNPID over conventional PID.

To compare the results of the NNPID controller with the results of the conventional PID controller, the parameters of the PID controller were tuned for initial gain setting of NNPID controller by its best fit values as proportional gain, K_p=2.5, integral gain, K_i=100 and derivative gain, K_d=10. The neural network fine tunes the system iteratively based on the performance of the closed loop

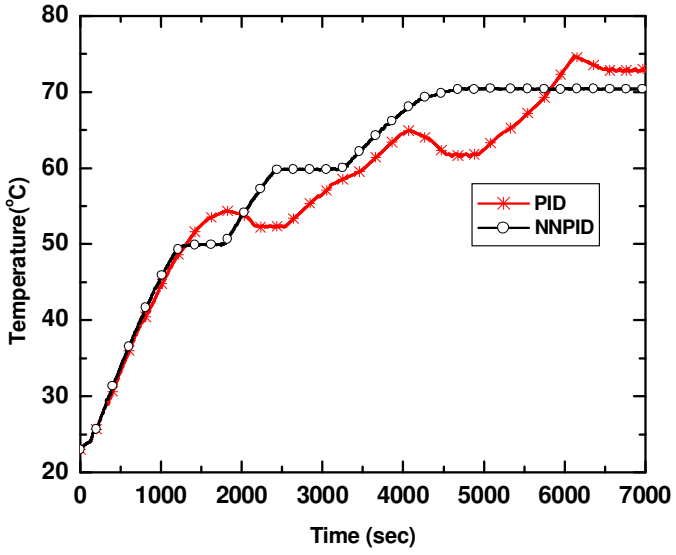

FIGURE 5: Showing the comparison of NNPID controller with the conventional PID controller of a water bath for 250 ml/min flow rate of water with respect to setpoint changes.

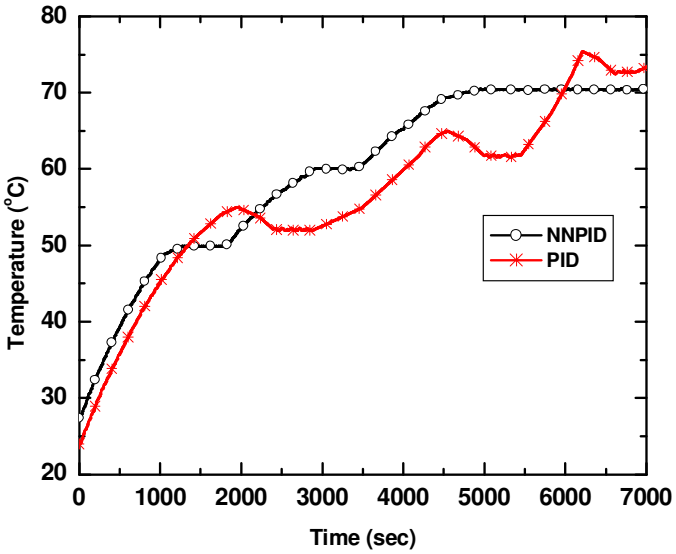

FIGURE 6: Showing the comparison of NNPID controller with the conventional PID controller of a water bath for 500 ml/min flow rate of water with respect to setpoint changes

K_p	2.5
K_i	100
K_d	10
Power of Heater	1500 Watt
Volume of water	15 liter
Voltage	5volts
Initial and Final Set point temperature	50°C and 70°C
Temperature change	+10°C
Flow rate of water	100ml/min, 250 ml/min, 500 ml/min
Load disturbance	100ml/min water

TABLE 1: Different Values of System Parameters

system. The temperature response of a water bath having 15 liter volume and heated with a power of 1.5KW for 100ml/min flow rate of water using NNPID and conventional PID are shown simultaneously for comparison in Fig.5. Similarly NNPID and conventional PID results for 250ml/min and 500ml/min flow rate of water are shown in Fig.5 and Fig.6 respectively. It is clear from these figures that there is always overshoot for conventional PID at initial settling time for each set temperature as 50°C, 60°C and 70°C of the system. This is shown in Table III. This table also indicates that NNPID controller gives error less than 0.11°C, 0.12°C and 0.12°C without overshoot for 50°C, 60°C and 70°C respectively for all the three flow rate of water. These errors are comparatively less than conventional PID controller. In addition, the neural network achieves setpoint fast as compared to the conventional PID controller as shown in Figs. (4-6). One can possibly say that the neural network controller tracked well all the three setpoint and has good generalization capability even with a small number of training patterns.

	NNPID Controller		PID Controller	
	Settling Time		Settling Time	
Temperature range	50°C-60°C	60°C-70°C	50°C-60°C	60°C-70°C
100 ml/min	7min	9min 30sec	23min	23min 30sec
250 ml/min	11min	18min 30sec	31min	31min
500 ml/min	17min	27min	35min	35min

TABLE 2: Settling Time of NNPID and PID Controllers For Three Flow Of Water

	NNPID Controller			Conventional PID Controller					
	Error without Overshoot			Error with Overshoot					
Set Temperature	50°C	60°C	70°C	50°C		60°C		70°C	
				Error	Over shoot	Error	Over shoot	Error	Over shoot
100 ml/min flow	0.09 °C	0.10 °C	0.10 °C	1.38 °C	4.49 °C	1.0 °C	3.03 °C	1.0 °C	2.01 °C
250 ml/min flow	0.10 °C	0.11 °C	0.12 °C	2.32 °C	4.35 °C	1.87 °C	4.9 °C	2.73 °C	4.47 °C
500 ml/min flow	0.11 °C	0.12 °C	0.11°C	2.54 °C	4.93 °C	1.90 °C	4.77 °C	2.88 °C	5.48 °C

TABLE 3: Error and Overshoot of NNPID and Conventional PID controller for three rate of flow of water

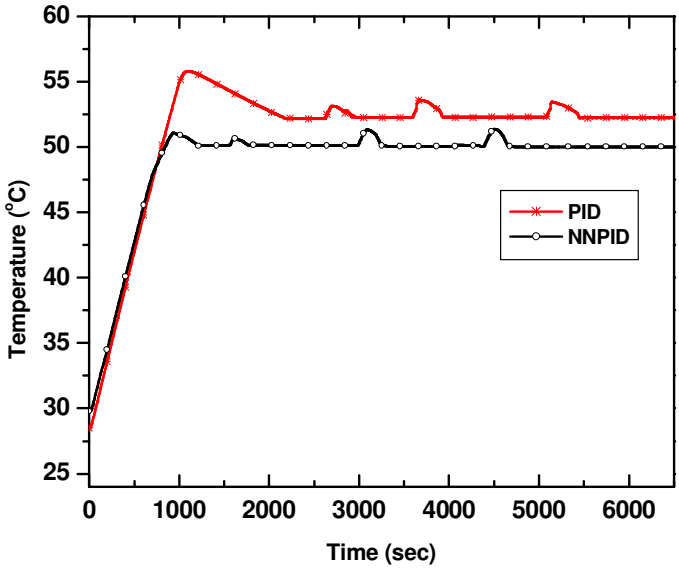

FIGURE 7: Showing the comparison of NNPID controller with the conventional PID controller of a water bath under the effect of load disturbances.

In second set of experiments, the load disturbances in terms of addition of 100ml/min water were introduced in the process of system for studying the ability of the two controllers when the

external disturbance was imposed. These external disturbances were made in three steps at different interval of time. These three disturbances were added to the output at 43min, 59min and 84min respectively for PID controller and for NNPID controller at 25min, 49min and 72min as shown in Fig. 7. It could be observed from this figure that when we introduce external disturbance of 100ml/min of water during three steps in the system for set temperature of 50^{o}C, the NNPID controller takes much less settling time and overshoot as compared to conventional PID controller. So it is appropriate to say that neural network controller recovered fast with error less than 0.08 ^{o}C with less overshoot under the effect of these load disturbances. So we are able to say that NNPID controller has ability to adapt quickly to changes at its input. On the other hand the conventional PID controller has poor rate of recovery which deteriorate the system. Additionally, it has error greater than 0.2^{o}C. Our experimental setup gives better settling time, less overshoot and minimum deviation in setpoint.

4. CONCLUSION

In conclusion, the present work shows the new approach of controlling the temperature of the dynamic system. This particular system designed and developed around Atmel's 89C51 microcontroller employed on a water bath. The temperature control of the system has been analyzed by conducting two experiments in respect of setpoint changes and load disturbances. The first experiment considers change in setpoint temperature in step of 10^{o}C from 50^{o}C to 70^{o}C for three different rate of flow of water. It is observed that NNPID controller gives error less than 0.11^{o}C, 0.12^{o}C and 0.12^{o}C without overshoot for 50^{o}C, 60^{o}C and 70^{o}C respectively for all three flow rate of water. In second experiment, the load disturbance in terms of addition of 100ml/min water at three different intervals of time is introduced. It gives error less than 0.08 ^{o}C with less overshoot under the effect of the load disturbance. In both the cases NN weights corresponding to PID parameters, are trained, to achieve better control than existing conventional PID. This paper has shown that inexpensive neural hardware may become an important technology for many modern industrial control applications.

5. REFERENCES

[1] M. Khalid, S. Omatu and R. Yusof, "MIMO furnace control with neural networks," *IEEE Trans. Contr. Syst. Technol.*, vol. 1, pp. 238–245, 1993.

[2] J. Tanomaru, S. Omatu, "Process Control by On-line Trained Neural Controllers," *IEEE Transactions on Industrial Electronics*, vol. 39,pp. 511-521, 1992.

[3] M. Khalid and S. Omatu, "A neural network controller for a temperature control system," *IEEE Contr. Syst.*, vol. 12, pp. 58–64, June 1992.

[4] W. Wu, J. Yuan and L. Cheng, "Self-tuning sub-optimal control of time-invariant systems with bounded disturbance," in *Proc. of the 2005 American Control Conference.*, vol. 2, 2005, pp. 876–882.

[5] C. Y. Guo, Q. Song, and W. J. Cai, "Supply Air Temperature Control of AHU with a Cascade Control Strategy and a SPSA Based Neural Controller," in *Proceedings of the 2005 International Joint Conference on Neural Networks*, vol. 4, 2005, pp. 2243-2248.

[6] S. Omatu, T. Iwasa, M. Yoshioka, "Skill-based PID Control by Using Neural Networks," in *Proceedings of the 1998 IEEE International Conference on System Man and Cybernetics*, vol. 2, 1998, pp. 1972-1977.

[7] Q. H. Hu, A. T. P. So, W. L. Tes and A. Dong, "Use of Adaline PID Control for a Real MVAC System," *Proceedings of the 2005 International Conference on Wireless Communications, Networking and Mobile Computing,* vol. 2, 2005, pp. 1374 – 1378.

[8] K. J. Astrom and T. Hagglund, "Automatic Tuning of Simple Regulators with Specifications on Phase and Amplitude Margins," *Automatica*, vol. 20, pp. 645-651, 1984.

[9] C. C. Hang, K.J. Astrom and W.K. Ho, "Refinements of the Ziegle-Nichols tuning formula," in 1991 *IEE proceedings Pt. D, Control theory & Applications*, vol.138, no. 2, 1991, pp. 111-118.

[10] W. K. Ho, C. C. Hang and L. S. Cao, "Tuning of PID Controllers Based on Gain and Phase Margin Specifications, *Automatica*, vol.31, no. 3, pp. 497-502, 1995.

[11] K. J. Astrom, C. C. Hang, P. Persson and W. K. Ho, "Toward Intelligent PID Control," *Automatica*, vol.28., no. 1, pp.1-9, 1992.

[12] W. K. Ho, O. P. Gan, E. B. Tay and E. L. Ang, "Performance and Gain and Phase Margins of Well Known PID Tuning Formulas," *IEEE Trans. On Control Systems Technology*, vol.4, pp.473-477, 1996.

[13] W. K. Ho, C. C. Hang and J. H. Zhou, "Performance and Gain and Phase Margins of Well-Known PI Tuning Formula," *IEEE Trans. On Control Systems Technology*, vol.3, no. 2, pp.245-248, 1995.

[14] F. Cameron and D.E. Seborg, "A self-tuning controller with a PID structure," *Int. J. Control* vol. 30, pp. 401-417, 1983.

[15] D.W. Clark and P.J. Gawthrop, "Self-tuning control," in *Proc. IEE, Pt-D*, vol. 126, 1979, pp. 633-640.

[16] R. Ortega and R. Kelly, "PID self-tuners: Some theoretical and practice aspects," *IEEE Trans. Ind. Electron*, vol. 31, pp. 312, 1984.

[17] C.G. Proudfoot, P.J. Gawthrop and O.L.R. Jacobs, "Self-tuning PI control of a pH neutralization process," in *Proc. IEE, Pt-D*, vol. 130, 1983, pp. 267-272.

[18] F. Radke and R. Isermann, "A parameter-adaptive PID controller with stepwise parameter optimization," *Automatic*, vol. 23, pp. 449-457, 1987.

[19] B. Wittenmark, "Self-tuning PID Controllers Based on Pole Placement," *Lund Institute Technical Report*, TFRT-7179, 1979.

[20] D. E. Rumelhart and J. L. McClelland, "Parallel Distributed Processing," vol. 1, MIT Press, Cambridge, MA, 1986.

[21] J. H. Taylor and K. J. Astrom, "A non-linear PID auto tuning algorithm", American Automatic control conference, Seattle, W.A., 1986, pp. 1-6.

[22] M. A. Unar, D. J. Murray-Smith and S. F. Ali Shah, "Design and tuning for fixed structure PID controllers—A survey", report CSC-96016, *Centre for systems and control & department of mechanical Engineering, university of Glaslow*, 1996.

[23] A. E. B. Ruano, P. J. Fleming and D. I. Jones, "Connectionist approach to PID autotuning," in *IEE proceedings-D*, vol. 139 (3), 1992, pp. 279-285.

[24] K. C. Chan, S. S. Leong and G. C. I. Lin, "A neural network PI controller tuner," *Artificial Intelligence in Engineering*, vol. 9, pp. 167-176, 1995.

[25] C. L. Chen and F. Y. Chang, "Design and analysis of neural/fuzzy variable structural PID control systems," in *IEE Proceedings Control Theory Application*, vol. 143 (2), 1996, pp. 200-208.

[26] V. VanDoren, "Model free adaptive control", *Control engineering, Europe*, pp. 25-31, 2001.

[27] M. Khalid, S. Omatu, "A neural network controller for a temperature control system," *IEEE Contr. Syst. Mag*., vol. 12, pp. 58-64, 1990.

[28] A. G. Barto, "Connectionist learning for control," in W. T. Miller, 111, R. S. Sutton, P. J. Werbos, eds., Neural Networks for Control. Cambridge, MA: MI, 1990.

[29] B. Widrow, S. D. Steams, "Adaptive Signal Processing," Englewood Cliffs, NJ: Prentice Hall, 1985.

[30] D. Psaltis, A. Sideris, A. Yamamura, "A multilayered neural network controller," *IEEE Control Syst. Mag*., vol. 10, pp. 44-48, 1988.

[31] K. S. Narendra, K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Trans.Neura1 Networks*, vol. 1, pp. 4-27, 1990.

[32] P. J. Werbos, "Backpropagation through time: What it does and how to do it?," in *Proc. IEEE. 78*, 1990, pp. 1550-1560.

[33] D. H. Nguyen and B. Widrow, "Neural networks for self-leaming control systems," *IEEE Control Syst. Mag*., vol. 10, pp. 18-23, 1990.

[34] M. Jordan and D. E. Rumelhart, "Forward models: Supervised learning with a distal teacher," *Cognitive Science*., vol. 16.pp. 307-354, 1992.

[35] A. N. Ponce, A. A. Behar, A. O. Hernandez and V. R. Sitar, "Neural Network for Self-tuning Control Systems", *Acta Polytechnica*, vol. 44, pp.49-52, 2004.