Journal of Theoretical and Applied Information Technology
20th November 2014. Vol. 69 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

TRACKING PERFORMANCES OF A HOT AIR BLOWER SYSTEM USING DIFFERENT TYPES OF CONTROLLERS

[1]SITI FATIMAH SULAIMAN, [1,2]KHAIRUDDIN OSMAN, [3]MOHD FUA'AD RAHMAT,
[4]AMAR FAIZ ZAINAL ABIDIN, [5]ISMAIL MOHD KHAIRUDDIN
[1]Lecturer, Department of Industrial Electronics, Universiti Teknikal Malaysia Melaka, Melaka, MALAYSIA
[2]Student, Department of Control & Mechatronic Engineering, Universiti Teknologi Malaysia, Johor Bahru, MALAYSIA
[3]Professor, Department of Control & Mechatronic Engineering, Universiti Teknologi Malaysia, Johor Bahru, MALAYSIA
[4]Lecturer, Faculty of Electrical Engineering, Universiti Teknologi MARA, Masai, Johor, MALAYSIA
[5]Lecturer, Faculty of Manufacturing Engineering, Universiti Malaysia Pahang, Pekan, Pahang, MALAYSIA
E-mail: [1]sitifatimahsulaiman@utem.edu.my

ABSTRACT

System modeling is an important task to develop a mathematical model that describes the dynamics of a system. The scope for this work consists of modeling and controller design for a particular system. A heating and ventilation model is the system to be modeled and will be perturbed by pseudo random binary sequences (PRBS) signal. Parametric approach using AutoRegressive with Exogenous input (ARX) model structure will be used to estimate the mathematical model or approximated model plant. In this work, the approximated plant model is estimated using System Identification approach. Once the mathematical model is obtained, several controllers such as Self-Tuning Pole Assignment controller, Proportional-Integral-Derivative (PID) controller, and Generalized Minimum Variance (GMV) controller are designed and simulated in MATLAB. Finally, a comparative study based on simulation is analyzed and discussed in order to identify which controller deliver better performance in terms of the system's tracking performances. It is found from a simulation done that a Self-Tuning Pole Assignment Servo-Regulator controller with a small value of pole give a best performance in term of its ability to eliminate error ($\%e_{ss}$) and produce zero percentage of overshoot ($\%OS$), while GMV controller using PSO tuning method offers a fast rise-time (T_r), settling time (T_s), and also its ability in eliminating ($\%e_{ss}$).

Keywords: *Hot Air Blower System, System Modeling, System Identification, ARX Model, Controllers*

1. INTRODUCTION

In control system engineering, the ability to accurately control the system that involves the temperature of flowing air is vital to numerous design efforts [1]. This work was conducted due to this problem. From observation, the system to be controlled was non-linear and has significant time delay. In this work, the control objective is to maintain the process temperature at a given value. There are several steps to be considered while doing this work; identify a process, obtain the mathematical model of the system, analyze and estimate the parameters using System Identification approach, design appropriate controllers for controlling the system and implement it to the system by simulation, and lastly make analysis and justification based on the results obtained.

A mathematical modeling process was provided a very useful method in this work since it was used in identifying a process, representing the dynamic, and describing the behavior of a physical system. A mathematical model of a physical system can be obtained using two approaches; analytical approach (physics law) and experimental approach (System Identification) [2]. Study on [3] found that the main problem of applying a physical law is, if a physical law that governing the behavior of the system is not completely defined, then formulating a mathematical model may be impossible. Thus, an experimental approach using System Identification was considered in this work. The term

Identification was first introduced by Zadeh (1956) that refers to the problem of determining the input-output relationships of a black box or modeling based on observed experimental data. Lennart Ljung (2008) defined *System Identification* as the art and science of building mathematical models of dynamic systems from observed input-output data. In this work, a mathematical model of the temperature response for the system is developed based on the measured input and output data set obtained from Real Laboratory Process which can be obtained from MATLAB demos. System Identification Toolbox which is available in MATLAB is then used to estimate the parameters and approximate the system models according to the mathematical models obtained. Basically, System Identification approach offers two techniques in describing a mathematical model, which are parametric and non-parametric method. In this work, parametric approach using AutoRegressive with Exogenous input (ARX) model structure is chosen to estimate and validate the approximated system model. In order to ensure the validity of the ARX model, Model Validation Criterion was used to decide whether the ARX model obtained should be accepted or rejected. Once the model have been identified and validated, appropriate controllers were designed to improve the output performance of the system. Three types of controllers were proposed in this work; Self-Tuning Pole Assignment controller, Proportional-Integral-Derivative (PID) controller, and Generalized Minimum Variance (GMV) controller. The tracking performances of the system by simulation using different type of controllers designed in order to maintain the process temperature at a given value will then be carried out, analyzed, and justified.

1.1 PT326 Process Trainer

In this work, PT326 process trainer is employed as a hot air blower system to be modeled. The process of PT326 process trainer works as follows [5]: The air from atmosphere is fanned through a tube. It was then heated at the inlet as it passes over a heater grid before being releases into the atmosphere through a tube. Here, adjusting the electrical power supplied to the heater grid will affect the temperature of the air flowing in the tube. For instance, a voltage varying from 0 to +10 Volts produces an air temperature changes from 30˚C to 60˚C [5]. The flowing air temperature is measured by a thermistor at the outlet and the system generally introduces a significant time delay due to the spatial separation between the thermistor and the heater coil. Thus, the power over the heating device (Watt) is considered as the input to the system, while the outlet air temperature (˚C) as the output to the system.

1.2 Model Identification and Estimation

A mathematical model of the system is developed based on the measured input and output data set obtained from Real Laboratory Process which can be obtained from MATLAB demos. In this work, 1000 measurements of collected input and output data from Real Laboratory Process of PT326 was sampled at the sampling interval is 0.08 seconds. The input to the system was generated as Pseudo Random Binary Sequence (PRBS). PRBS is preferable to be used as an input signal to the system because of the advantage of easy to generate and introduce into a system. Besides, the signal intensity is low with energy spreading over a wide range of frequency makes PRBS as a good choice for force function [3]. Figure 1 shows a plot of measured input and output data of the system in time domain response.

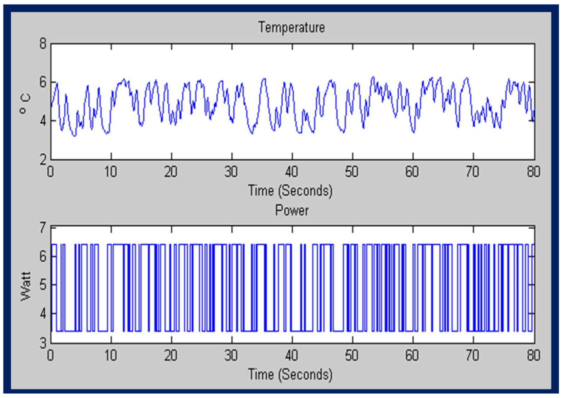

Figure 1: A Plot of Measured Input and Output Data of PT326

In System Identification, the measured input and output data obtained must be divided into two sets of data; the first data set for estimation, while the second data set for validation purpose. In this work, the first 1-500 samples of data were used for estimation and the remaining for validation purpose. To estimate a suitable model structure to approximate the model of the PT326 process trainer, System Identification Toolbox in MATLAB environment is employed. There are a few model structures which are commonly used in real world application and these structures also available in MATLAB System Identification Toolbox: AutoRegressive with Exogenous input (ARX), AutoRegressive Moving Average with Exogenous input (ARMAX), Output Error (OE), and Box Jenkins (BJ). In this work, the ARX model structure is chosen since it is the simplest model incorporating the stimulus signal. ARX with the order of $n_a = 2, n_b = 2,$ and $n_k = 3$ (ARX223) were selected in this work, and the discrete-time transfer function as obtained from MATLAB System Identification Toolbox can be represented as:

$$\frac{B_o(z^{-1})}{A_o(z^{-1})} = \frac{0.06518z^{-3}+0.04497z^{-4}}{1-1.278z^{-1}+0.3973z^{-2}} \tag{1}$$

The zero polynomials is,

$$B_c(z^{-1}) = 1 + 0.68994z^{-1} \tag{2}$$

The model validation is considered as a final stage of the System Identification approach. As described earlier in a beginning of Section 1, the second set of data (501-1000 samples) will be used for validation purpose. In this work, the model validation is to verify the identified model represents the process under consideration adequately; to check the validity between the measured and desired data under a validation requirement. Akaike's Model Validity Criterion is used since it is very popular method for validating a parametric model such as ARX and ARMAX model structure. The mathematical model obtained is validated based on its Best Fit, Loss Function, and Akaike's Final Prediction Error (FPE). A model is acceptable if the Best Fit is more than 80%. The term *fit* means the closeness between the measured and simulated model output, and it can be calculated using Eq. (3):

$$Fit = 100\left[1 - \frac{norm(\hat{y} - y)}{norm\ (y - \bar{y})}\right]\% \tag{3}$$

; y: true value,
$\hat{y}$: approximate value,
$\bar{y}$: mean value

A model is acceptable if the Loss Function and Akaike's FPE is as smallest as possible. The values of Loss Function and Akaike's FPE can be calculated using Eq. (4) and (5):

$$V = \frac{e^2(k)}{N} = \frac{e^T(k).e(k)}{N} \tag{4}$$

; $e(k)$: error vector

$$FPE = V\frac{\left(1 + \frac{d}{N}\right)}{\left(1 - \frac{d}{N}\right)} \tag{5}$$

; V: loss function
d: no. of approximated parameter
N: no. of sample

Using System Identification Toolbox, the best fit of the output model is 89.18% as depicted in Figure 2. From the plot, a measured value is indicated by a black curve and the simulated model output is indicated by a blue curve. The model plant is acceptable since the percentage of the best fit is greater than 80%. The Loss Function and Akaike's FPE of the ARX223 model is considered small with the value 0.00170053 and 0.00172774. The results are summarized in Table 1.

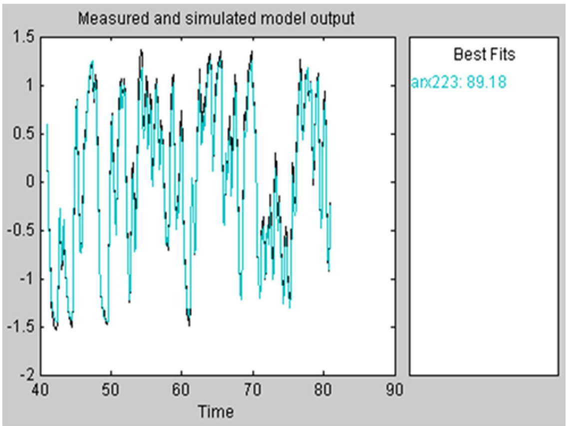

Figure 2: Measured and Simulated Model Output (Best Fit)

Table 1: Akaike's Model Validity Criterion Value Based on ARX223 Model Structure

ARX 223	
Best Fit	89.18%
Loss Function	0.00170053
Akaike's FPE	0.00172774

Thus, the approximated model of ARX223 is acceptable since all those three criteria of Model Validation Criterion are satisfied.

1.3 Controller Design

Several studies are currently kept on tackling this issue on designing a suitable controller for improving the output performance of the system considered. Eko Harsono (2009) designed a Proportional (P) and Proportional-Integral (PI) Controller, and has implemented both controllers to the simulation and real-time process. Mohd Fahmy

(2010) designed a Proportional-Integral-Derivative (PID) controller to control the system and he proposed Ziegler Nichols tuning method for tuning those PID parameters. An intelligent tuning method for PID controller using Radial Basis Function Neural Network (RBFNN) tuning method was presented by Ibrahim (2010). In this work, three different types of controllers, namely Self-Tuning Pole Assignment controller, Proportional-Integral-Derivative (PID) controller, and Generalized Minimum Variance (GMV) controller have been proposed. In Self-Tuning Pole Assignment controller, a method of Servo-Regulator control is chosen, in PID controller, a Ziegler Nichols (ZN) and Particle Swarm Optimization (PSO) tuning method were used, and in GMV controller, a Self-Tuning method will be compared with PSO tuning method.

A. Self-Tuning Pole Assignment Servo-Regulator Controller

A Self-Tuning Pole Assignment Controller using Servo-Regulator method is chosen as a first controller for this work. It was chosen since it is easy to implement, and it can also be used to improve the speed of the system response and make the system's output follow the reference signal at steady-state [7]. Study in [2] show that the performance of the model obtained using this method as a feedback controller is acceptable since all the poles of the closed-loop are placed at the desired location and it is also provided satisfactory and stable output performance. In this work, a Self-Tuning Pole Assignment Servo-Regulator controller is designed, where the control objective is to ensure the output of the closed-loop system, $y(t)$ to track the reference signal, $r(t)$, and at the same time rejects any external disturbances or noises in the system.

The closed-loop system equation of a Pole Assignment controller,

$$y(t) = \frac{z^{-k}BH}{AF + z^{-k}BG} r(t) + \frac{CF}{AF + z^{-k}BG} \xi(t) \qquad (6)$$

$$\begin{aligned} ;\ A &= 1 + a_1 z^{-1} + \cdots + a_{n_a} z^{-n_a} \\ B &= b_0 + b_1 z^{-1} + \cdots + b_{n_b} z^{-n_b} \\ C &= 1 + c_1 z^{-1} + \cdots + c_{n_c} z^{-n_c} \\ ;\ F &= 1 + f_1 z^{-1} + \cdots + f_{n_f} z^{-n_f} \\ G &= g_0 + g_1 z^{-1} + \cdots + g_{n_g} z^{-n_g} \\ H &= h_0 + h_1 z^{-1} + \cdots + h_{n_h} z^{-n_h} \end{aligned}$$

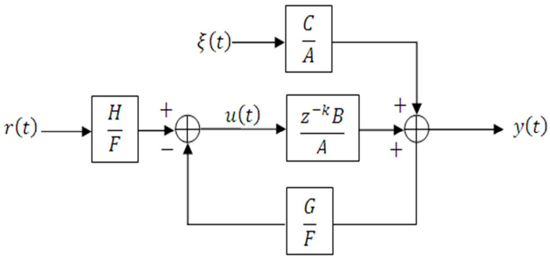

Figure 3: The Block Diagram of the Closed-Loop Pole Assignment Servo-Regulator Controller

Equipped with this information and using identity, *F*, *G*, and *H* values can be determined. Table 2 compares the values of *F*, *G*, and *H* for different values of t_1 assigned.

Table 2: Comparison between Small and Large Value of Pole Assigned

Pole at +0.2 (small)	**Pole at +0.8 (large)**
$T = 1 - 0.2z^{-1}$; $n_t = 1$	$T = 1 - 0.8z^{-1}$; $n_t = 1$
$n_f = 3$, $F = 1 + 1.078z^{-1} + 0.984z^{-2} + 0.376z^{-3}$	$n_f = 3$, $F = 1 + 0.478z^{-1} + 0.214z^{-2}\ 0.057z^{-3}$
$n_g = 1$ $G = 6.886 - 3.331z^{-1}$	$n_g = 1$ $G = 0.427 - 0.5z^{-1}$
$H = 7.263$	$H = 1.816$

B. Proportional-Integral-Derivative (PID) Controller

A PID controller is one of the feedback type controller normally used in process industries. C.C. Yu (1999) in [8] found that more than 90% of the control loops in industries are of this type. PID controller, on the other hand, has proved to be rather popular in many control system applications due to its flexibility, simple structure, performance is quite robust for a wide range of operating conditions, and also provides adequate performance in the vast majority of applications [9]. As the name suggested, a PID controller consists of three basic parameters, which are proportional, integral, and derivative. Each parameter has their own functionality [10] and the performance of a PID controller is mainly determined by these three parameters. Figure 4 illustrates the PID controller in a closed-loop system and a general PID equation is given by Eq. (7).

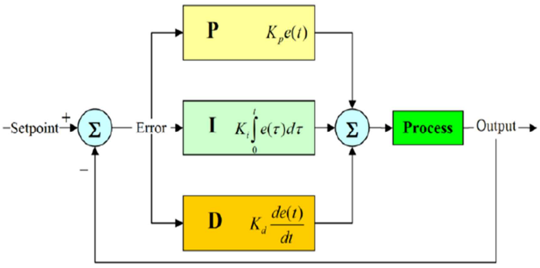

Figure 4: The PID Controller in a Closed-Loop System

$$PID = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{d}{dt} e(t) \tag{7}$$

In PID controller, it is necessary to decide which parameter to be used and specify its correct value. This is because; incorrect value of parameters may affect performances of the controller. Because of this, tuning these three PID parameters are crucial and many studies are kept tackling on this issue. As years passed by, "try and error" method (normally used to tune the PID parameters) is considered as a wasting time method. This is due to the performances of PID controller nowadays can be improved with automatic tuning, automatic generation of gain schedules, and continuous adaptation [10]. In this work, PID controller using Ziegler Nichols and PSO tuning method are discussed.

i) ZN-PID Controller

A popular PID tuning method, Ziegler Nichols is employed in this work in order to determine the appropriate value of PID parameters which are K_p, K_i, and K_d. To realized this, the PID parameters as shown in Table 3 are used.

Table 3: Ziegler Nichols Table

Controller	K_p	K_i	K_d
P	$0.5K_c$		
PI	$0.45K_c$	$\frac{1.2K_p}{P_c}$	
PID	$0.6K_c$	$\frac{2K_p}{P_c}$	$\frac{K_pP_c}{8}$

The updated values of PID parameter are indicated in Table 4.

Table 4: Updated Values Of K_p, K_i, and K_d Using Ziegler Nichols Tuning Method

Controller	K_p	K_i	K_d
PID	12.353	2.246	16.985

ii) PSO-PID Controller

The Particle Swarm Optimization (PSO) is an algorithm that used a concept of population-based search algorithm on the simulation of the social behavior of birds within a flock. Kennedy and Eberhart (1995) introduced the principle of PSO as a movement of the members of bird flocks and fish schools without colliding. Nowadays, PSO can be and has been used in a wide range of area of applications include: communication networks, control, design, biomedical, entertainment, and many more [11]. No matter what kind of applications, PSO was aimed to graphically simulate the graceful and unpredictable choreography of a bird flock [12]. Here, the ability of birds to fly synchronously and to suddenly change direction with a regrouping in an optimal formation was discovered. The principles of PSO algorithm are very simple, because it is actually based on the movement and information sharing of particles in a multi-dimensional search space. Each particle will always emulate the success of their neighboring individuals and compare it with their own *(pbest)*, and its position is adjusted according to their own experience and that of its neighbors *(gbest)*.

PSO used the concept of maintaining a swarm of particles each particle, thus each particle is able to discover their optimal regions of a high dimensional search space. Figure 5 shows the general flowchart of PSO.

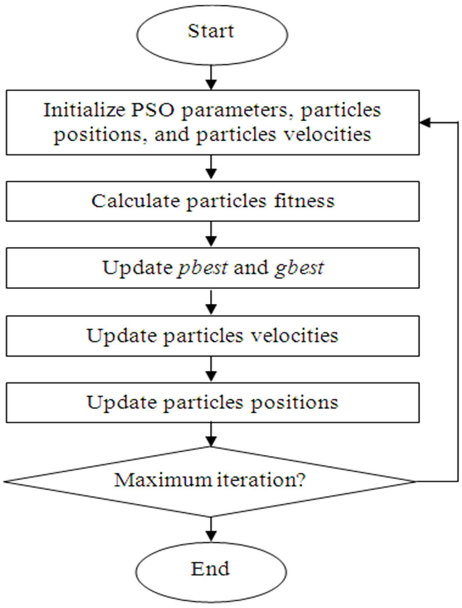

Figure 5: The General Flowchart of PSO

Since three parameters in PID which are K_p, K_i, and $\mathrm{K_d}$ are crucial to tune, this work will use PSO tuning method as an alternative to tune those parameters. The updated values of $\mathrm{K_p}$, $\mathrm{K_i}$, and $\mathrm{K_d}$ using PSO tuning method is shown in Table 5.

Table 5: Updated Values of K_p, K_i, and K_d using PSO Tuning Method

Controller	K_p	K_i	K_d
PID	2.334	3.095	5

C. Generalized Minimum Variance (GMV) Controller

GMVC is an extension of Minimum Variance Control (MVC). This method is introduced in order to accommodate servo control and to overcome disadvantages introduced by MVC, where in MVC there are some drawbacks that the designer must consider when applying it: the performance of MVC is affected by time delay, *k*, MVC ignores the amount of control effort required, and many more. The general block diagram of GMV controller is shown in Figure 6 below.

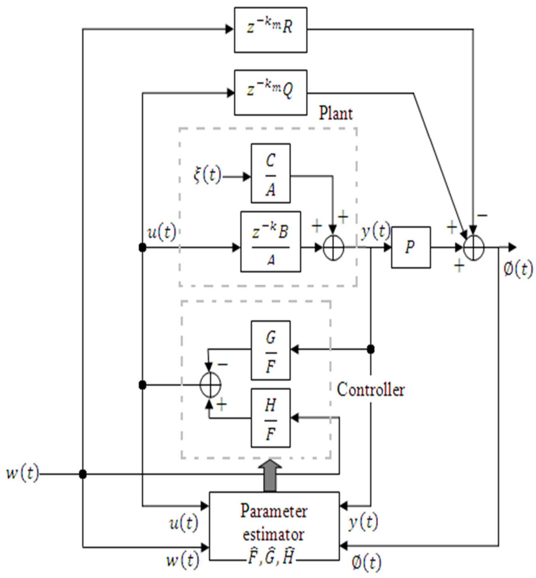

Figure 6: The General Structure of GMV Controller Block Diagram ($k_m \le k$)

Tthe steps in designing self-tuning GMVC in this work can be summarized as below:

<u>Step 1:</u>
The equation as below is determined:

$$\emptyset(t) = Py(t) - Rw(t-k) + Qu(t-k) \tag{8}$$

<u>Step 2:</u>
$\hat{F}, \hat{G}$, and $\hat{H}$ are estimated using RLS algorithm:

$$\emptyset_{RLS}(t) = \hat{G}y(t-k) + \hat{F}u(t-k) - \hat{H}w(t-k) + \xi(t) \tag{9}$$

Replacing $k = 3$, the regression form can be written as below:

$$\emptyset_{RLS}(t) = [y(t-3) \;\; y(t-4) \;\; u(t-3) \;\; u(t-4) \;\; u(t-5) \;\; u(t-6) \;\; -w(t-3)] \begin{bmatrix} \hat{g}_0 \\ \hat{g}_1 \\ \hat{f}_0 \\ \hat{f}_1 \\ \hat{f}_2 \\ \hat{f}_3 \\ \hat{h}_0 \end{bmatrix} + \xi(t) \tag{10}$$

<u>Step 3:</u>
The GMVC law is calculated and applied to the system:

$$u(t) = \frac{\hat{H}}{\hat{F}} w(t) - \frac{\hat{G}}{\hat{F}} y(t) \tag{11}$$

<u>Step 4:</u>
The algorithm is repeated for the next iteration or sampling time.

i) Self-Tuning GMV Controller

In this work, the value of *P* and *Q* is assumed to be 1, while *R* value to be 0 (set by the designer).

Table 6: Values of Weighting Factors, P, Q and R (Normally Used Values)

Weighting Factors	***P***	***Q***	***R***
	1	1	0

ii) PSO-GMV Controller

Compared to the second controller proposed, PSO tuning method is used in GMV controller to tune the value of weighting factors P, Q and R in the pseudo output.

Table 7: Updated Values of Weighting Factors, P, Q and R using PSO Tuning Method

Weighting Factors	***P***	***Q***	***R***
	0.705	−5	−4.72

2. RESULTS AND ANALYSIS

In this section, the proposed controllers are implemented by simulation using MATLAB software and the corresponding results are presented. A unit step input is considered as a desired temperature of the process, while the outputs from the controllers designed are the actual or measured values. A temperature of 40°C with a step change of 5 seconds is designed as a desired temperature of PT3276 process trainer. The aim of this work is to design appropriate controllers that can track or follow the setpoint (desired temperature) based on the approximated model plant that obtains using System Identification approach. Three types of controllers were developed and their performances are discussed and analyzed.

A. Self-Tuning Pole Assignment Servo-Regulator Controller

The simulation results for the output of the Self-Tuning Pole Assignment Servo-Regulator controller is shown in Figure 6. The performances of two different value of single pole assigned are compared and analyzed.

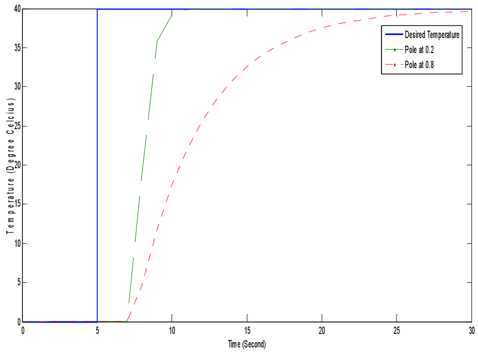

Figure 6: Output Responses of the Self-Tuning Pole Assignment Servo-Regulator Controller

Based on the results, both responses produce a time delay at the beginning of the simulation; it happen due to the approximated plant model is estimated using ARX223 model structure. The response that uses small value of pole is much more aggressive compared to the response that using a large value of pole. No overshoot occur for both responses and both responses are stables. In terms of percentage of steady-state error $(\%e_{ss})$, the response with a large value of pole give a percentage of steady-state error $(\%e_{ss})$ of 0.3% and clearly seen from graph that the response takes time to reach a desired temperature. By comparing both responses, the results clearly demonstrate that Self-Tuning Pole Assignment Servo-Regulator controller with a small value of pole give a fast response (T_r), no overshoot $(0\%OS)$, and met the design requirement which is produce no steady-state error $(0\%e_{ss})$.

B. Proportional-Integral-Derivative (PID) Controller

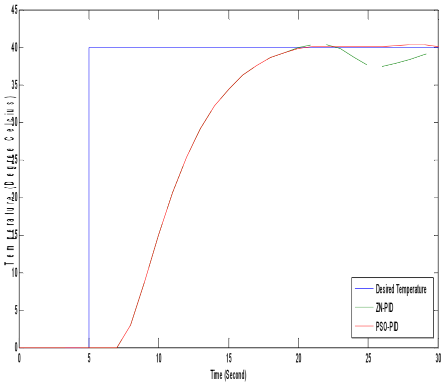

Figure 7: Output Responses of the PID Controller

Figure 7 shows the output performance of PID controller using two different tuning methods. At the beginning of the simulation, both responses show a similarity in term of rise time (T_r). However, after simulation about 20 seconds, clearly seen that PID controller that uses Ziegler Nichols tuning method has about 2.5% overshoot and the response tend to unstable as time increased. Compared to the response that using Ziegler Nichols, response that uses PSO tuning algorithm provides a good performance in term of overshoot, that is $0\%OS$.

C. Generalized Minimum Variance (GMV) Controller

Figure 8 shows the output performances of the GMV controller. From a simulation result obtained, responses that uses PSO tuning method gives the best stability results compared to the self-tuning responses that uses a default value to tune the parameters P, Q and R. This is because; GMV controller that uses PSO tuning method also has the fast response (T_r) and allowable value of percentage overshoot, which is only 2.6% of overshoot.

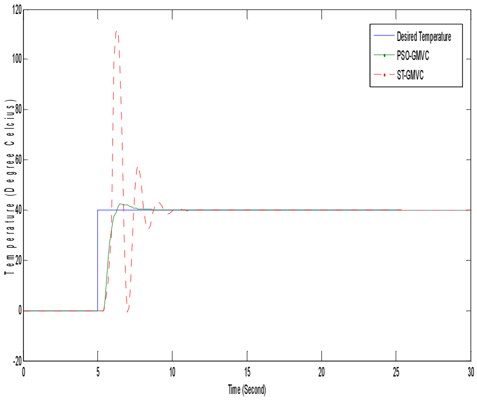

Figure 8: Output responses of the GMV controller

The summary of the performances of the controllers designed is shown in Table 7 -8 quantitatively:

Table 8: Performances of the Controllers Designed

Response Characteristic	**Controller**					
	Self-Tuning Pole Assignment Servo-Regulator Controller		**Proportional-Integral-Derivative (PID) Controller**		**Generalized Minimum Variance (GMV) Controller**	
	Pole at 0.2	**Pole at 0.8**	**ZN-PID**	**PSO-PID**	**ST-GMV**	**PSO-GMV**
Percent O/shoot ($\%OS$)	0%	0%	2.5%	0%	180%	2.6%
Peak Time (T_p)	0s	0s	23s	0s	6.6s	6.7s
Settling Time (T_s)	4.8s	17s	19.2s	12.5s	4.8s	3.8s
Rise Time (T_r)	2s	9.5s	8s	8s	0.4s	0.6s
Percent Steady-State Error ($\%e_{ss}$)	0%	0.3%	-	0.1%	0%	0%

Based on Table 8, clearly described that each controller has their own advantage and disadvantage. A Self-Tuning Pole Assignment Servo-Regulator controller with a small value of pole has the best performance in term of percentage of overshoot ($\%OS$) and steady state error ($\%e_{ss}$), while a GMV controller using PSO tuning method provide a good result in term of rising time (T_r) and settling time (T_s). However, both are good in stability, eliminating error, improve the speed of the system response and make the system's output follow the reference signal at steady-state.

CONCLUSIONS

In this work, several controllers are designed to control one system, namely hot air blower system (PT326 process trainer), which is non-linear and has a significant time delay. The control objective is; to accurately control the system that involves the temperature of flowing air, where the controllers designed must be able to maintain the process temperature of the system at a given value. Three types of controllers are designed and presented in this work, with two different tuning methods. From the simulation result obtained, it can be concluded that the Self-Tuning Pole Assignment Servo-Regulator controller with a small value of pole provide relatively high ability in controlling the system. This is due to its ability to reject noise and tracking the setpoint of the system. Besides, a GMV controller using PSO tuning method also obviously has improved the performance of the Self-Tuning GMV controller in term of rise time (T_r) and settling time (T_s). Hence, in terms of transient response of the controlled system, the decision on whether Self-Tuning Pole Assignment Servo-Regulator controller with a small value of pole or GMV controller using PSO tuning method offers a better result is actually related to the performance criteria required from the controlled response itself: for a zero percentage of overshoot ($0\%OS$), it is suggested Pole Assignment controller, while for a fast rise time (T_r) and settling time (T_s) points towards GMV controller.

Based on observation from this work, objective of this work is successfully achieved. For further work, effort can be devoted by implementing both controllers in real-time process, so that the results obtained from experiment can be validated with theoretical or simulation.

ACKNOWLEDGEMENT

This work was supported by a Short Term Research Grant Scheme of Universiti Teknikal Malaysia Melaka. The authors are indebted to Faculty of Electronics and Computer Engineering (UTeM), and Faculty of Electrical Engineering (UTM and UiTM), and Faculty of Manufacturing Engineering (UMP) for their help in conducting this work.

REFERENCES

[1] C.L. Philips and R.D. Harbor. *Feedback Control Systems.* Prentice Hall, 4th Ed. Pages 49-51.

[2] Norlela Ishak, Siti Karimah Harun, Hashimah Ismail, Noorfadzli Abdul Razak, and Ramli Adnan. Model Identification and Real-Time Linear Position Control of Servomotor. In *Proceedings of the IEEE: Control and System Graduate Research Colloquium*, pages 71-74, 2010.

[3] Mohd Fua'ad Rahmat, Yeoh Keat Hoe, Sahnius Usman, and Norhaliza Abdul Wahab. Modelling of PT326 Hot Air Blower Trainer Kit Using PRBS Signal and Cross-Correlation Technique. *Universiti Teknologi Malaysia: Jurnal Teknologi*, page 22, 2005.

[4] Mohd Fahmy Izwan Bin Azlan. *System Identification and Estimation of a Hot Air Blower System Using Parametric Method.* Universiti Teknologi Malaysia: Thesis, pages 45-48, 2010.

[5] *Process Trainer PT326 Manual Handbook*. Feedback Instruments Ltd.

[6] Ibrahim Mohd Ali Alsofyani. *A Model-Based Neural Network Controller for a Hot Air Blower System*. Universiti Teknologi Malaysia: Thesis, pages 42-43, 53-55, 2010.

[7] Hazlina Selamat. *Introduction to Adaptive & Self-Tuning Control (Chapter 5).* Universiti Teknologi Malaysia: Lecture Module (2010/2011-2), pages 128-158, 2011.

[8] C.C. Yu. *Auto-tuning of PID Controllers: Relay feedback Approach.* London: Springer-Verlag, 1999.

[9] K. Pirabakaran and V.M. Becerra. Automatic Tuning of PID Controllers Using Model Reference Adaptive Control Techniques. In *Proceedings of the IEEE Industrial Electronics Society*, pages 736-740, 2001.

[10] Abdul Rasyid Mohammad Ali. *Optimization of Controller Parameters for A Couple Tank System Using Metamodeling Technique*. Universiti Teknologi Malaysia: Thesis, pages 33, 2009.

[11] Riccardo Poli. An Analysis of Publications on Particle Swarm Optimisation Applications. Pages 1–10, 2007.

[12] J. Kennedy and R.C. Eberhart. Particle Swarm Optimization. In *Proceedings of the IEEE International Joint Conference on Neural Networks*, pages 1942-1948, 1995.