A SOFTWARE-BASED GAIN SCHEDULING OF PID CONTROLLER

Hussein Sarhan

Department of Mechatronics Engineering, Faculty of Engineering Technology, Amman, Jordan

ABSTRACT

In this paper, a scheduled-gain SG-PID controller using LabVIEW-based scheduling technique, which consists of a set of virtual instruments, has been designed and experimentally tested for heating process. Gain scheduling is realized by automatic setting of the controller parameters using three sets of programmed parameters, depending on the relative error between actual process temperature and setpoint (desired temperature). Experimental results show that the proposed controller, compared with conventional C-PID controller, responds faster to the changes in the setpoint temperature, reduces the overshoots in temperature during transient period and makes the system more stable. It was noticed that the dynamic and steady-state errors in the system have been reduced.

KEYWORDS

PID Controller, Gain Scheduling, LabVIEW Software, Heating Process Control

1. INTRODUCTION

A proportional-integral-derivative PID controller is a control loop feedback controller widely used in industrial control systems. A PID controller calculates an "error" value as the difference between a measured process variable and a desired set-point. The controller attempts to minimize the error in outputs by adjusting the process control inputs. The PID controller algorithm involves three separate constant parameters: the proportional, the integral and derivative values, denoted *P, I,* and *D.* These values can be interpreted in terms of time: *P* depends on the present error, *I* on the accumulation of past errors, and *D* is a prediction of future errors, based on current rate of change. The weighted sum of these three actions is used to adjust the process via a control element in the system. By tuning the three parameters in the PID controller algorithm, the controller can provide control action designed for specific process requirements. The response of the controller can be described in terms of the responsiveness of the controller to an error, the degree to which the controller overshoots the setpoint, and the degree of system oscillation [1, 2]. However, since the controller parameters are fixed during control after they have been chosen through a certain (optimal) method, the conventional C-PID controller can't always keep satisfying performances, so, the use of the PID algorithm for control does not guarantee optimal control of the system or system stability. To overcome these disadvantages, C-PID controller should be improved. The gains of PID controller should be adjusted dynamically during the control process. Many on-line tuning algorithms, such as fuzzy logic, neural network and genetic algorithm, adaptive PID controller, model predictive PID controller, analytical PID controller and GS-PID controller have been introduced into C-PID controller to achieve desired control

DOI : 10.5121/ijics.2014.4301

performances for the entire operating envelope of system [3-7]. Usually, PID tuning to find the proper values for PID controller gains involves trial and error. With the analytical PID libraries included in the LabVIEW Control Design and Simulation Module, the tools to find sets of PID gain values automatically for a given user model (process) are available, which ensures system closed-loop stability. Also, it is possible to input minimum gain and phase margin values to specify the optional performance constraints on the PID controller [8].

In this paper, gain scheduling technique, based on LabVIEW software is used to improve the behaviour of C-PID controller used to maintain the temperature of heating process constant.

2. GAIN SCHEDULING BASED ON LABVIEW SOFTWARE

LabVIEW- Laboratory Virtual Instrument NI Engineering Workbench- is a new graphical development software with built-in functionality for data acquisition, instrumentation and control, measurement analysis, and data presentation. LabVIEW delivers extensive acquisition, analysis, and presentation capabilities, so it is possible to create an application, simulation and presentation in a single environment without utilizing for other software.

Unlike general purpose programming languages, NI LabVIEW provides functionality specifically customized to the needs of measurement, control, and automation applications, accelerating the development process from built-in analysis capabilities to connectivity with a wide variety of input/ output I/O (digital I/O, analog I/O and counters). LabVIEW delivers what engineers and scientists need to quickly build, test and measure data acquisition, embedded control, scientific research, process monitoring systems, and a documentation facility that provides a full time and date details about the measuring values needed to be restored in a file. LabVIEW uses a graphical programming language called the G language. This language represents programs in a symbolic form called the block diagram, which reduces a lot of syntactical details, so the user will concentrate on the flow of information than the procedure. Also, the graphical form of the programs keeps users from making syntax errors as in text-based language and this saves time. In LabVIEW there is an ability to execute each sub-routine by itself without the need of making a main program, and also it is possible to run sub-routines and view the parameters in the main program. In addition to all of the facilities above, there is a very important ability of writing all the information needed to a text file with a full time and date details about the values generated by the program. Moreover, it is possible to create an error message or an alarm when there is a problem in the process, or if some specified values jumped over the permissible errors.

LabVIEW ties the creation of user interfaces, called front panels, into the development cycle. LabVIEW programs (subroutines) are called virtual instruments VIs. Each VI has three components: block diagram, front panel and connector pain. The last is used to represent the VI in the block diagrams of other VIs. The front panel is built using controls and indicators. Controls are inputs: they allow a user to supply information to the VI. Indicators are outputs: they indicate, or display, the results based on the inputs given to the VI. The back panel, which is a block diagram, contains the graphical source code. All of the objects placed on the front panel will appear on the block diagram as terminals. The block diagram also contains structures and functions which perform operations on controls and supply data to indicators. The structures and functions are found on the functions palette and can be placed on the block diagram. The LabVIEW programming environment makes it simple to create some applications. The PID Virtual Instruments VIs with National Instruments hardware can be used to develop different LabVIEW control applications based on PID controllers. In general, the block diagram of C-PID controller is presented in Figure 1, where $r(t)$ is the reference (setpoint sp) input of the system,

$y(t)$ is the system output signal (process variable pv), $e(t)$ is the error signal, which is the difference between $r(t)$ and $y(t)$, and $u(t)$ is the controller output signal.

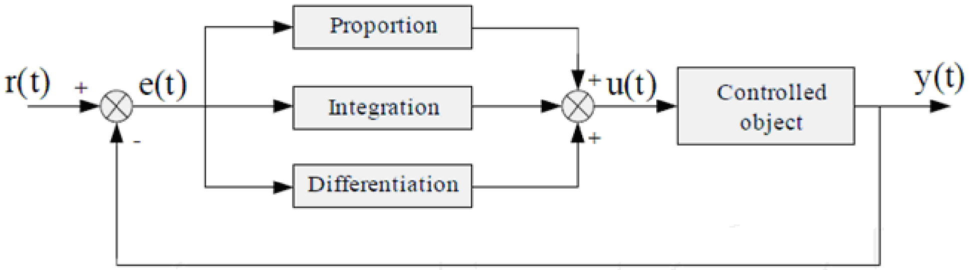

Figure 1 Block diagram of C-PID controller.

Sometimes, the relative error $\varepsilon(t)$ is used instead of error $e(t)$. The relative error is defined as:

$$\varepsilon(t) = \frac{r(t) - y(t)}{r(t)} = \frac{sp - pv}{sp} \qquad (1)$$

The algorithm of C-PID controller can be given as:

$$e(t) = r(t) - y(t) \qquad (2)$$

$$u(t) = K_p[e(t) + \frac{1}{T_i}\int e(t)dt + T_d \frac{de(t)}{dt}] \qquad (3)$$

where K_p is the proportional gain, T_i is the integral time constant and T_d is derivative time constant.

Equation (3) can be rewritten as:

$$u(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de(t)}{dt} \qquad (4)$$

where $K_i = K_p / T_i$ is integral gain and $K_d = K_p T_d$ is derivative gain [4, 5].

The LabVIEW PID Control Toolkit uses Ziegler and Nichol's heuristic methods for determining the parameters of PID controller, depending on the system dynamic performance: fast, normal and slow.

Since the proposed controller will be implemented and experimentally tested for heating process system, where the temperature change is considered to be slow, so the tuning formula for slow action will be used.

The PID controller parameters for slow action are listed in Table 1, where K_u and T_u are programmed values for proportional gain and integral time constant for critically damped process.

Table 1. PID controller parameters for slow action.

Controller	K_p	$T_i=1/K_i$	$T_d=K_d$
P	$0.13K_u$	-	-
PI	$0.13K_u$	$0.8T_u$	-
PID	$0.15K_u$	$0.5T_u$	$0.12T_u$

The block diagram of SG-PID controller is shown in Figure 2.

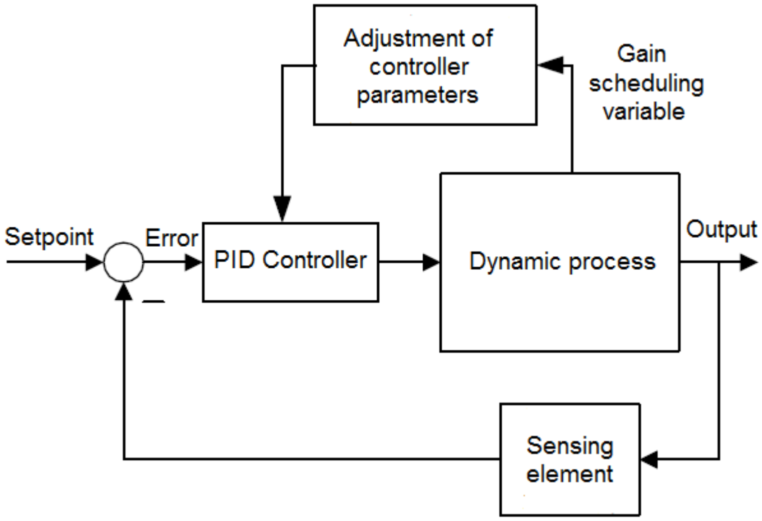

Figure 2. Block diagram of SG-PID controller.

Gain scheduling VI in LabVIEW software is the responsible for adjustment of controller parameters. This VI returns the set of PID gains corresponding to the error signal. It is a combination of different arithmetic and comparison functions that collectively are responsible of the controller behavior according to the change in process variable. The gain scheduling variable is the relative error, defined by equation (1), multiplied by minus unity. The relative error is examined to determine which gain set is the most suitable for the controller. Three ranges for the error are picked according to how large the relative error is. Each range has its conditions. The relationship between relative error and conditions is shown below:

$$\begin{aligned} &\textit{if } \varepsilon \geq 0.3, \text{then } K_p = K_{p1}, K_i = K_{i1}, K_d = K_{d1} \\ &\textit{if } 0.15 < \varepsilon < 0.3, \text{then } K_p = K_{p2}, K_i = K_{i2}, K_d = K_{d2} \\ &\textit{if } 0 \leq \varepsilon < 0.15, \text{then } K_p = K_{p3}, K_i = K_{i3}, K_d = K_{d3} \end{aligned} \tag{5}$$

The conditions were programmed with comparison functions. The three cases were programmed in a way, where only one case is activated, or they are all OFF ($pv < sp$). After the gains are loaded into the PID controller, in association with signals pv and sp, the controller calculates the error and returns an output signal C_{out}, which is expressed as:

$$C_{out} = \varepsilon K_p + K_p K_i \int \varepsilon dt + K_p K_d \frac{d\varepsilon}{dt} \qquad (6)$$

The controller output signal C_{out} controls the operation of the actuator in the system (fan motor in this paper).

3. PRACTICAL INVESTIGATION OF PROPOSED SG-PID CONTROLLER

The performance of SG PID controller based on LabVIEW software has been experimentally investigated for heating process control system. The block diagram of the heating system is shown in Figure 3. The system consists of a set of physical components (hardware) and virtual instruments VIs (LabVIEW software), which are interconnected to each other in order to realize the required control algorithm. The main hardware components are:

1. Plexi-glass box, which serves as a closed chamber, dimensions of 300*200*200 mm. The box can withstand high temperature up to $90^o C$. The controller should maintain the temperature in the box at the desired level.
2. Uncontrolled, 220V (AC), 750W electric heater, installed inside the box.

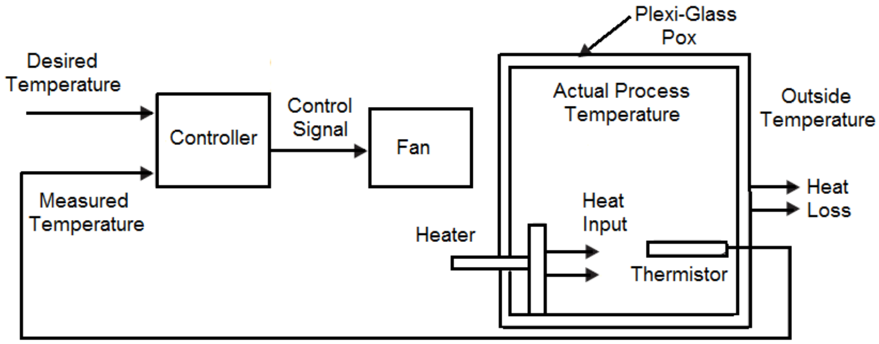

Figure 3. Block diagram of heating process control system.

3. Adjustable speed suction (centrifugal) fan (24V DC), which is used to cool the box. The suction fan speed is controlled according to the output of the controller by using a pulse-width modulation PWM and Darlington pair circuit.
4. Darlington pair circuit, based on a BJT transistor with MOSFET transistor, as shown in Figure 4.

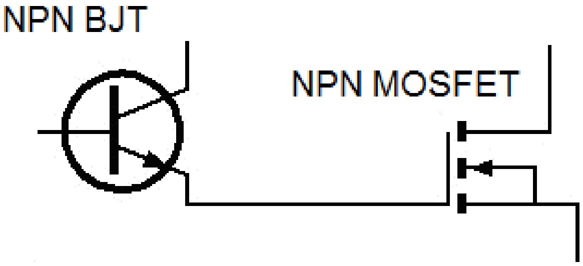

Figure 4. Darlington pair circuit.

5. Thermistor LM35 type, which is used as a temperature sensor. The thermistor has been installed inside the box to precisely measure the actual temperature.
6. Interfacing device NI myDAQ, is used for interfacing between the LabVIEW software, thermistor and fan motor. This device provides analog input AI, analog output AO, digital input and output DIO, power supplies and digital multimeter DMM functions. The internal structure of NI myDAQ is shown in Figure 5 [9].
The temperature controller in the system is a LabVIEW software (PID and gain scheduling) controller downloaded in a computer.

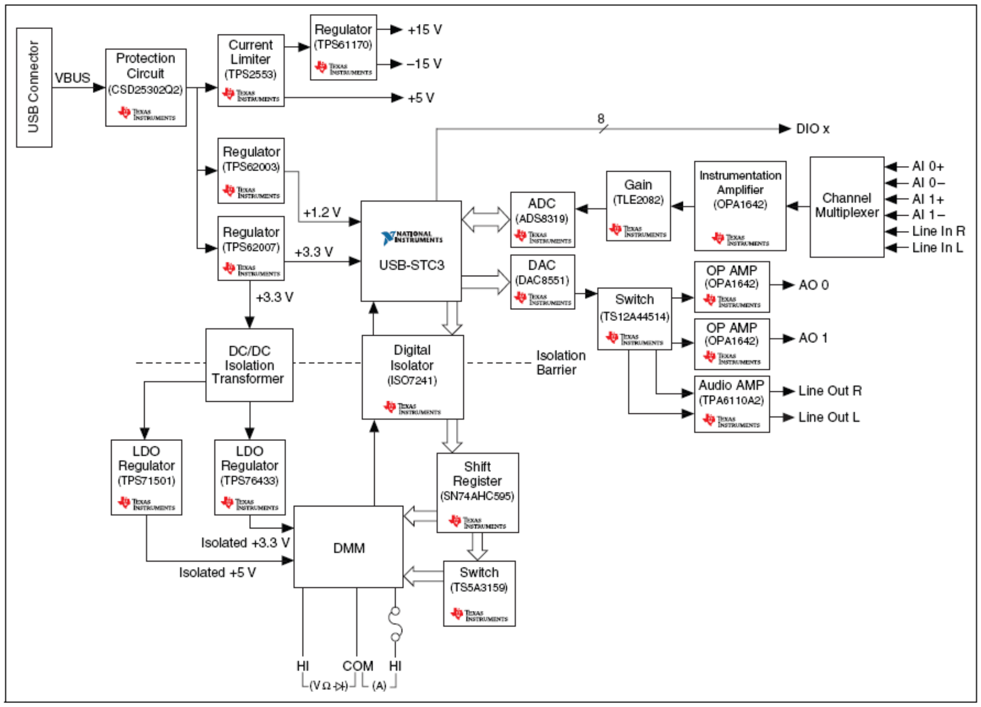

Figure 5. Internal structure of NI myDAQ

The schematic diagram and picture of experimental setup are shown in Figures (6, 7).

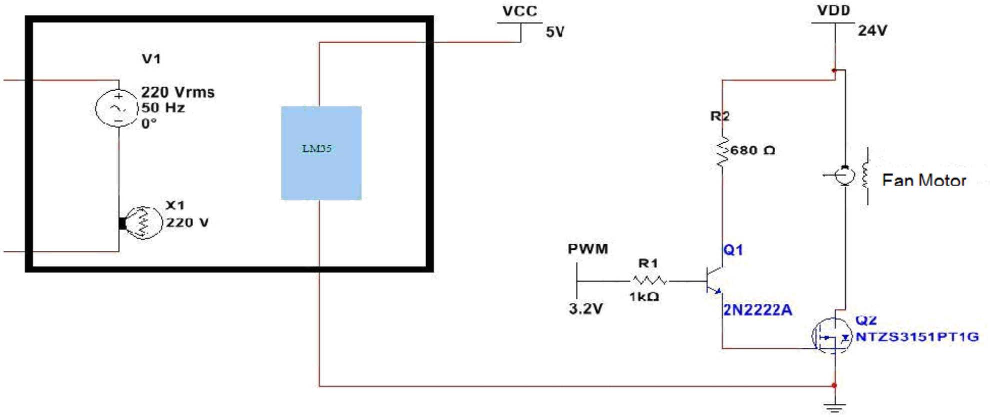

Figure 6. Schematic diagram of experimental setup.

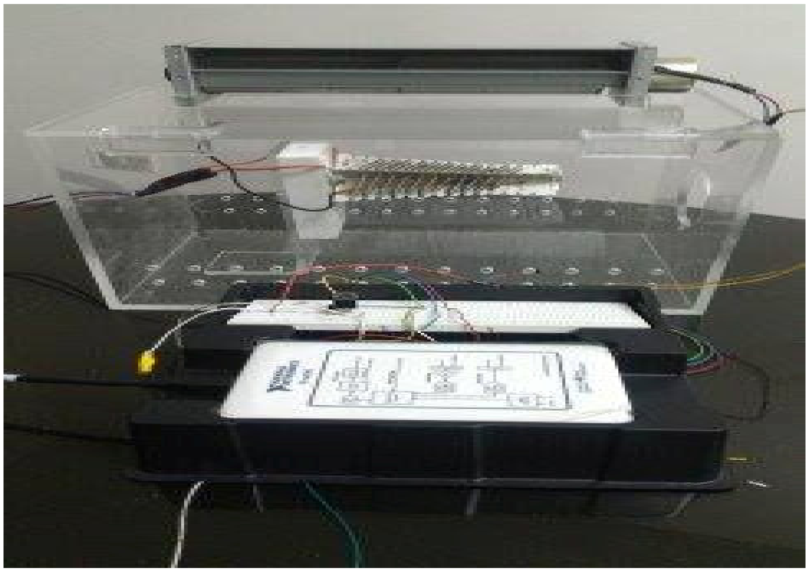

Figure 7. Picture of experimental setup.

In order to study the effect of using GS-PID controller on the system performance, two controllers were used: C-PID controller and GS-PID controller. The C-PID controller has only one set of gains, while the SG-PID controller was enhanced with the gain scheduling tool in LabVIEW software. The readings were taken from the waveform graph on the front panel window of LabVIEW, recorded and analyzed. The gains were calculated by using Nicholas Ziegler frequency method. These gains are represented in Table (1). These gains were loaded to controller and then tuned to get an optimal response. The two types of PID controller were experimentally tested under the same conditions. The outputs of the two controllers were computed by formulas shown in Tables (2, 3).

Table 2. Controller output without gain scheduling (C-PID controller).

Relative Error, ε	**Gains, (P, I, D)**	**C-PID controller output, C_{out}**
$\varepsilon > 0$	(-25, 0.02, 1x10^{-25})	$C_{out} = \varepsilon\text{x}(-25) + (-25)\text{x}0.02\int \varepsilon dt + (-25)\text{x}1\text{x}10^{-25}$
$\varepsilon < 0$	(0, 0, 0)	$C_{out} = \varepsilon\text{x}0 + 0\text{x}\int \varepsilon dt + 0\text{x}\dfrac{d\varepsilon}{dt}$, No output

Two tests were carried out. The initial conditions for the first test: setpoint temperature was $35^{o}C$ and process temperature $70^{o}C$. The responses of C-PID and GS-PID controllers are shown in Figures (8, 9). C-PID controller took 92 seconds to reach the setpoint temperature; and the overshoot was 4.5^{o}C. The system reached steady-state after 45 seconds.

The response of GS-PID controller, compared with that of C-PID controller has been improved. The time to reach the setpoint temperature has been reduced to 81 seconds and the overshoot was 1°*C*.

Table 3: Controller output with gain scheduling (GS-PID controller)

Relative Error, ε	**Gains, (P, I, D)**	**PID controller output, C_{out}**
$\varepsilon > 30\%$	(-25, 0.02, 1x10^{-25})	$C_{out} = \varepsilon\text{x}(-25) + (-25)\text{x}0.02\int \varepsilon dt + (-25)\text{x}1\text{x}10^{-25}$
$10\% < \varepsilon <$	(-25, 0.025, 1x10^{-28})	$C_{out} = \varepsilon\text{x}(-25) + (-25)\text{x}0.025\int \varepsilon dt + (-25)\text{x}1\text{x}10^{-2}$
$0 < \varepsilon < 10$	(-15, 0.03, 0)	$C_{out} = \varepsilon\text{x}(-15) + (-15)\text{x}0.03\int \varepsilon dt + (-15)\text{x}0\text{x}\frac{d\varepsilon}{dt}$
$\varepsilon < 0$	(0, 0, 0)	$C_{out} = \varepsilon\text{x}0 + 0\text{x}\int \varepsilon dt + 0\text{x}\frac{d\varepsilon}{dt}$, No output

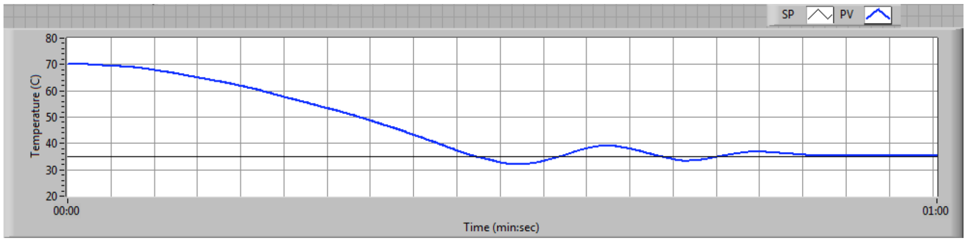

Figure 8. Response of C-PID controller.

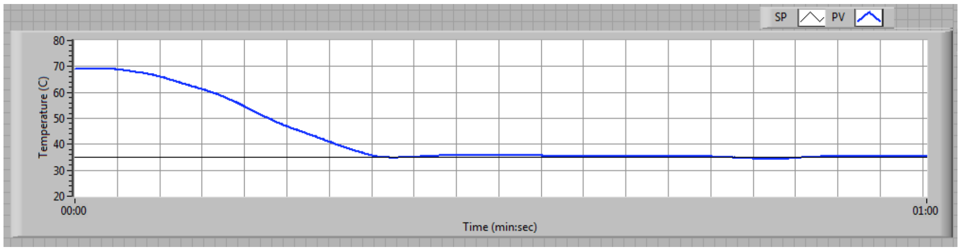

Figure 9. Response of SG-PID controller.

The second test illustrates the response of the two controllers to sudden change in setpoint temperature. Figure (10) shows the response of C-PID controller, while Figure (11) shows the response of SG-PID controller. It is clear from the Figures. (10,11) that the system with SG-PID controller has better response to the sudden change in setpoint temperature from 60°*C* to 40°*C*.

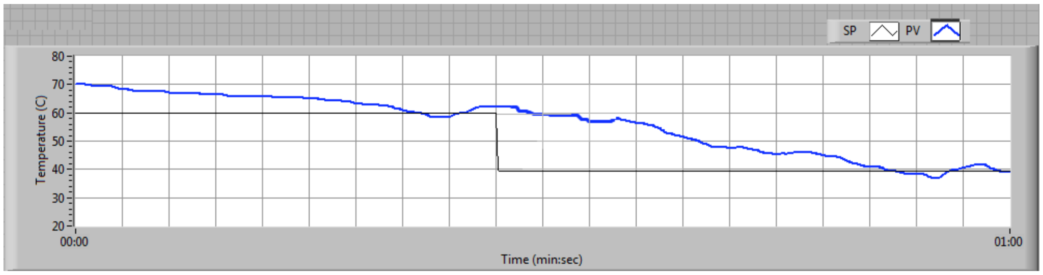

Figure 10. Response of C-PID controller to sudden change.

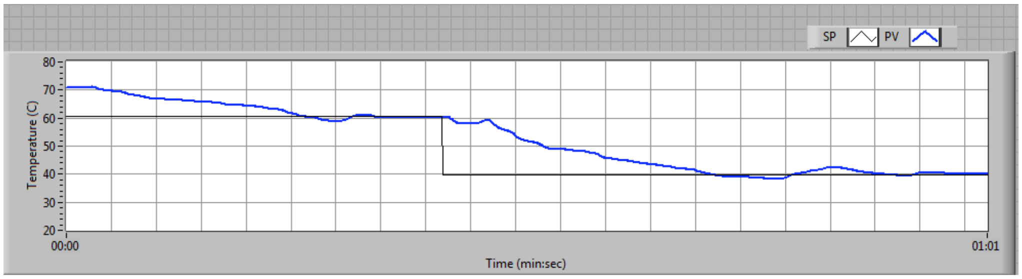

Figure 11. Response of SG-PID controller to sudden change.

3. CONCLUSIONS

Based on the experimental results and observations, the following conclusions can be done:

1. The feasibility of using LabVIEW-based SG-PID controller is demonstrated.

2. Gain scheduling makes the process variable reach the steady state faster, compared with conventional PID controller.
3. Gain scheduling makes the system more stable to sudden changes in the setpoint variable or process variable.
4. Gain scheduling reduces the dynamic and static errors in the system.

REFERENCES

[1] Daniel E. Rivera, Manfred Morari and Sigurd Skogestad, (1986) "Internal model control: PID controller design", Ind. Eng. Chem. Process Des. Dev, Vol. 25, No. 1, pp. 252-265.

[2] Leandro Dos Santos Coelho and Antonio Augusto Rodrigues Coelho, (1999) "Automatic tuning of PID and gain scheduling PID controllers by a derandomized evolution strategy", II EDAM, Vol. 13, pp. 341-349.

[3] Zhen-Yu Zhao, (1993) "Fuzzy gain scheduling of PID controllers", IEEE Transactions on Systems and Cybernetics, Vol. 23, No. 5, pp. 1392-1398.

[4] Bipin Krishna, Sagnik Gangopadhyay and Jim George, (2012) "Design and Simulation of Gain Scheduling PID Controller for Ball and Beam System", International Conference on Systems, Signal Processing and Electronics Engineering (ICSSEE'2012), December 26-27, 2012 Dubai (UAE), pp. 199-203.

[5] Chong Lin, Qing-Guo Wang and Tong Heng Lee, (2004) "An improvement on multivariable PID controller design via iterative LMI approach", Automatica, Vol. 40, pp. 519-526.

[6] P. Cominos and N. Munro, (2002) "PID controllers: recent tuning methods and design to specification", IEE Proceedings - Control Theory and Applications, Vol. 149, No. 1, , pp.46 – 53.

[7] O'Dwyer Aidan, (2006) "Performance improvement using simple PID controller tuning formulae", International The 3rd IET Conference on Power Electronics, Machines and Drives, April 4-6, 2006, Dublin, Ireland, pp. 276-280.

[8] National Instruments NI tutorial- Gain scheduling, 2005.

[9] National Instruments, “NI myDAQ User Guide and Specifications”

Author

Hussein Sarhan was born in Amman, Jordan, in 1952. He received the Master and PhD degrees in Electric Drive and Automation from Moscow Power Engineering Institute, USSR, in 1978 and 1981, respectively.His research areas are induction motor optimization techniques, energy efficient control of electric drives and NC positioning systems control.Dr. Sarhan is a faculty member (associate professor) in Faculty of Engineering Technology, Mechatronics Engineering Department.Dr. Sarhan is a member of Jordanian Engineering Association.