IJCSC International Journal of Computer Science & Communication Vol. 1, No. 1, January-June 2010, pp. 273-277

Performance Comparison of on Line Auto Tune PID Controller with Conventional PID Controller

[1]S. S. Gade, [2]S. B. Shendage & [3]M. D. Uplane

[1]Lecturer, E&TC Dept. ADCET, Ashta, Maharastra, India
[2]Sr. Lecturer & head, Computer Dept., PVPIT, Budhgaon, Maharastra, India
[3]Prof. & head, Electronics Dept., Shivaji University, Kolhapur, Maharastra India
[1]Sachin_gade123@yahoo.co.in, [2]Sangam333@gmail.com, [3]Uplane_suk@hotmail.com

ABSTRACT

This paper presents the performance comparison of an on-line version of a new auto-tuning algorithm for proportional-integral – derivative (PID) controllers based on the successive approximation method and conventional PID controller [3]. The new auto tuner causes only minor perturbation on the normal operation of the process, needs little a prior information, and is robust to noise. The performance and design for automatic selection of the PID constants are also discussed. The accuracy and performance of this new auto-tuning method have been substantiated by extensive lab works.

Keywords: *Auto Tuning, PID Controller, Successive Approximation Method.*

1. INTRODUCTION

The proportional integral and derivative (PID) controller has been used in process industries to control the plant (system) for the desired set point. The PID control method is most flexible and simple method. This method is more popular among all control methods. The determination of proportional (KP), derivative (KD) and integral (KI) constants are known as tuning of PID controller. Ziegler and Nichols proposed the manual tuning of PID controller [13]. This is off line practical method of PID constants determination. In this method there is chance of system become unstable. Only Experts can do the tuning for the PID controllers. The online determination of PID constants without manual interference is called as auto tuning of PID controller. The proceeding work is carried out of PID controller's on line auto tuning that is based on successive approximation method (SAM).The advantage of SAM-PID controller is that the auto tuning is also carry out for higher order systems. The results of auto tune PID is compared with conventional PID. The performance comparison is done by using practical results and MATLAB simulation results [14, 15].

2. SOFTWARE MODEL OF PID EQUATION

The general form of PID controller is [1]

$$Y = K_P e + K_d \frac{de}{dt} + \int K_I e dt \qquad \ldots(1)$$

The PID equation has many different forms. The software model of PID equation is derived as follows.

Differentiate the equation 1 w.r.t. time.

$$\frac{dY}{dt} = K_P \frac{de}{dt} + K_d \frac{\partial^2 e}{\partial t^2} + K_I e$$

Multiply with dt

$$dY = K_P de + K_d \frac{\partial^2 e}{\partial t} + K_I edt$$

Replace the time varying quantity with discrete elements as [1][2]

$$\Delta Y = K_P \Delta e + K_d \frac{\Delta^2 e}{\Delta t} + K_I e \Delta t$$

$$Y_1 = Y_0 + K_P \Delta e + K_d \frac{\Delta^2 e}{T} + K_I eT \qquad \ldots(2)$$

Equation number 2 describes software model of PID equation.

3. AUTO TUNE BY SUCCESSIVE APPROXIMATION METHOD [3][4][5]

The successive approximation method (SAM) is versatile method to find out a value of parameter if it is lies between some known limit. Successively the value is determined and taking more number of iteration for accurate value. This method is very simple and need only computer to do some basic calculations. Consider Y is output of PID equation and e is error. Let for the initial condition assumes that derivative and integral constant is close to zero or very small. Now the output is given by the equation

$$Y_0 = K_p e_0$$

$$K_p = \frac{Y_0}{e_0}$$

Where K_p is proportional constant.

For the next sampling output is Y_1 but by considering previousvalue of K_p the second approximation is

$$Y_1 = K_p e_1$$

$$Y_1 = \left(\frac{Y_0}{e_0}\right) e_1$$

Similarly, this can approximate up to n

$$Y_n = \left(\frac{Y_{n-1}}{e_{n-1}}\right) e_n$$

Integral Constant [3][4][5]

Similarly, the integral constant can be determined as

$$Y_n = \frac{Y_{n-1}}{\int_0^{t_{n-1}} e_{n-1} dt_{(n-1)} - \int_0^{t_{n-1}} \lim_{e_{n-1} \to 0} e_{n-1} dt_{(n-1)}} \times \left(\int_0^{t_n} e_n dt_n - \int_0^{t_n} \lim_{e_n \to 0} e_n dt_n \right)$$

Derivative Constant [3][4][5]

Similarly, the derivative constant can be determined as

$$Y_n = \frac{Yn - 1}{\frac{de_{n-1}}{dt} - \lim_{e_{n-1} \to 0} \frac{de_{n-1}}{dt}} \times \left(\frac{de_n}{dt} - \lim_{e_n \to 0} \frac{de_n}{dt} \right)$$

Thus, using computer program one by one PID constants are obtained.

4. SOFTWARE AND HARDWARE APPROACH

The SAM-PID controller comprises two major components 1. Hardware 2. Application program running on windows machine [6] [7]. The figure 1 shows the hardware structure of SAM-PID controller. The plant output is sampled at sampling frequency Fs and Analog to digital converter is used to send this data to the microcontroller. The microcontroller acts as interface between plant and application program running on windows platform. The microcontroller sends this data to the application program using RS-232 serial communication protocol. The application program has been developed on windows platform using VS2005 toolset. [6][7].The figure 3 shows the data flow in the application program. The embedded firmware has been written using assembly level programming. [8][9]

The application program has event based structure (refer figure 2). When an event occurs the control is transferred to service routine. The data is being transferred to and from using a serial port. The MsComm built in visual module can handle the data. The input buffer of MsComm consists of data sent by microcontroller. Thus the process value is obtained by using MsComm read property. The control detects for the plant and if the plant is new plant then a separate database file is created to note down the details of plant for further detection [12]. The application program running in auto tune mode in this mode the PID constants are calculated using successive approximation method. The output Y is calculated using PID constants. Thus the PID constants are iterated after each sampling to near about its accurate value. The control enters into running mode after tuning is completed. [10][11]

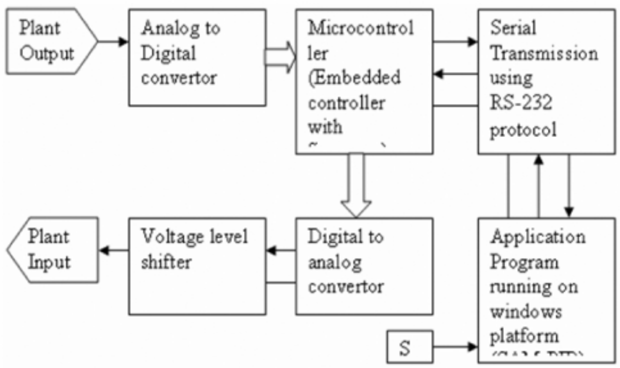


Fig 1: Block Diagram of SAM-PID Controller

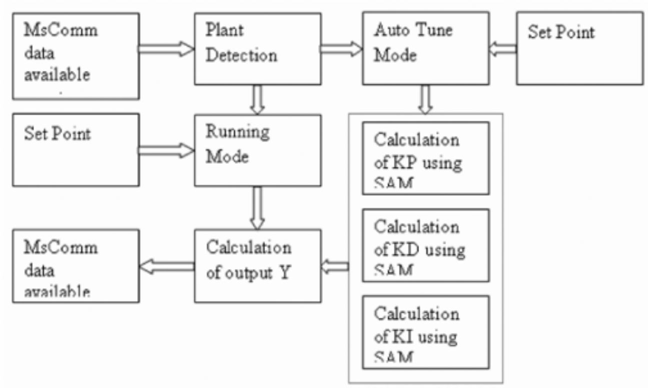


Fig 2: Data Flow and Control in Software

5. THEORETICAL CALCULATION OF KP, KI AND KD [13]

Consider the second order system represented by a transfer function of

$$G(s) = \frac{1}{(0.1s + 1)(0.2s + 1)}$$

The PID controller can be tuned by using following relationship

$$K_P = 0.6K_{CR}$$

$$\tau_i = 0.5T_{CR}$$

$$\tau_d = 0.125T_{CR}$$

The critical gain K_{CR} is determined by Routh array for the characteristic equation

$$0.02s^2 + 0.3s + 1 + K_P = 0$$

It is seen from Routh array that the stability of plant would be given by following equation like

$$0.02s^2 + 1 + K_P = 0$$

To find the frequency of oscillations T_{CR}

Let critical gain K_{CR} is 1.5

$$0.02s^2 + 1 + 1.5 = 0$$

$$0.02s^2 + 2.5 = 0$$

$$s = \sqrt{\frac{2.5}{0.02}}$$

$$s = \pm j11.18$$

$$T_{CR} = \frac{2\pi}{11.18} = 0.5619$$

$$\tau_i = 0.281$$

$$\tau_d = 0.07024$$

$$K_P = 0.9$$

$$K_I = \frac{K_P}{\tau_i} = 3.2$$

$$K_d = \tau_d K_P = 0.06$$

6. MATLAB SIMULATION

The simulation result of proceeding plant is obtained by using MATLAB model. The model is constructed as shown in figure [14] [15]. The calculated value is used for KP, KI and KD.

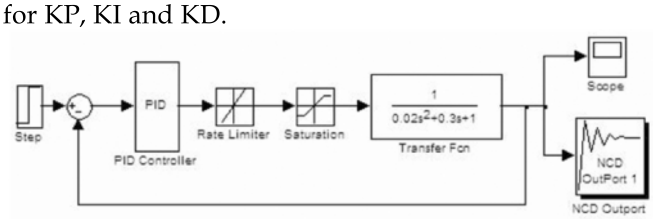
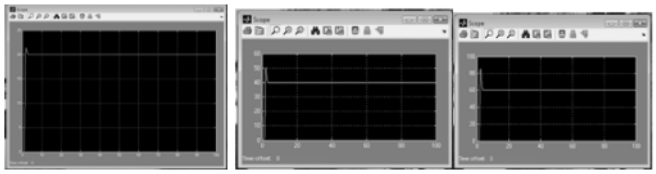


Fig 3: MATLAB Model

The simulation results are obtained for the set point of 20, 40 and 60 are



Graph1: Theoretical Results at different Set Point

The simulation results shows the performance of PID controller with the ideal or theoretical value of KP,KI and KD. The various parameters are calculated by referring above simulation results.

The results are entered in the tabular form as

Table 1
For Calculated PID Constants

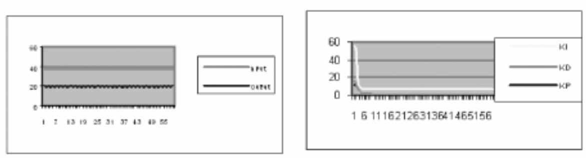
Set Point	*Peak Value*	*Settling Time*
20	21.425	3.2 min
40	50.42	3.85 min
60	85.5	4.2 min

The next step is to draw the graph of on line auto tune PID controller. This controller on line determines the value of PID constants and simultaneously controls the plant. The successive approximation method is used to determine PID constants.

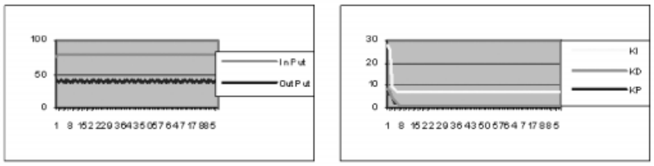
7. RESULTS OF AUTO TUNING BY SUCCESSIVE APPROXIMATION METHOD

The successive approximation method type PID controller (SAM-PID Controller) is used to obtain value of PID constants. This is on line determination method. The front end of application program is shown in the figure

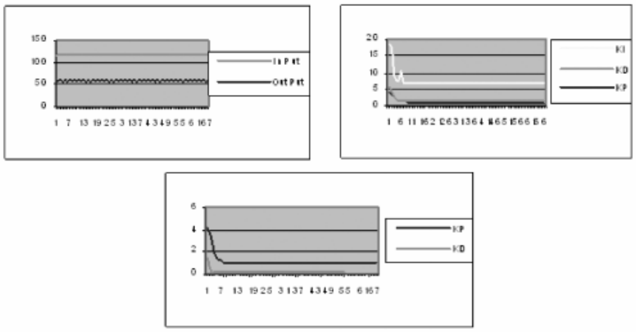
For the three different set point results of SAM-PID controller are



Graph 2: Practical Results at Set Point of 20



Graph 3: Practical Results at Set Point of 40



Graph 4: Practical Results at Set Point of 60

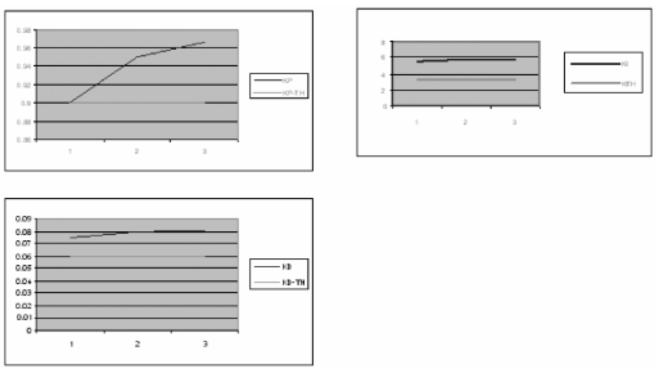
Results obtained using SAM-PID controller

Table 2
Auto Tune Results using SAM-PID Controller

OutPut	*InPut*	*SetPoint*	*KP*	*KD*	*KI*	*Error*
21	17.1	20	0.9	0.074999	5.4	–1
38	39.9	40	0.949999	0.079166	5.7	2
63	55.1	60	0.966666	0.080555	5.8	–3

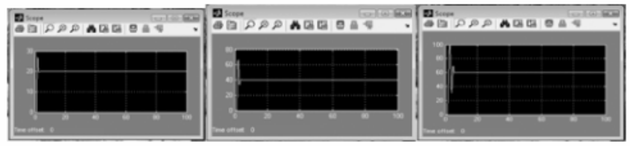
Table 3
For Constants by Auto Tune PID

Set Point	*Peak Value*	*Settling Time*
20	27	4.5 min
40	67	5 min
60	97	7 min



Graph 5: Comparison of Theoretical PID Constants and Auto Tune Results using SAM

MATLAB results are obtained for the PID constants which are getting from on line auto tune SAM-PID controller. The simulation results are taking for the set point and respective value of KP,KI and KD from using the data in table 2



Graph 6: MATLAB Simulation at PID Constants Obtained by SAM for different Set Points

Table 4
Simulation Results for PID Constants by Auto Tune PID

Set Point	*Peak Value*	*Settling Time*
20	26.83	3.39 min
40	66.91	4.45 min
60	98.96	6.34 min

The results obtained are integrated as summary in the table number 5.

Table 5
Auto Tune Value of PID Constants

Set point	*Calculated Value*			*Auto tune value*			*Simulation of auto tune value*		
	KP	KI	KD	KP	KI	KD	KP	KI	KD
20				0.9	5.4	0.074	Same as		
40	0.9	3.2	0.06	0.94	5.7	0.079	obtained by		
60				0.96	5.8	0.080	using auto		
							tune PID		

Table 6
Performance Comparison

Set point	*Calculated Value*		*Auto tune value Value*		*Simulation of auto tunevalue*	
	Peak Value	*Settling Time*	*Peak Value*	*Settling Time*	*Peak Value*	*Settling Time*
20	21.425	3.2	27	4.5	26.83	3.39
40	50.42	3.85	67	5	66.91	4.45
60	85.5	4.2	97	7	98.96	6.34

8. CONCLUSION

The proceeding SAM-PID controller shows the satisfactory performance. It is proved from the theoretical analysis and results obtained by simulation. The SAM-PID controller can auto tunes the controller keeping good performance. From table 5 it is clear that the auto tune PID controller has been fulfill the requirements.

REFERENCES

[1] "Handbook of PI and PID Controller Tuning Rules" BY AIDAN O'DWYER.

[2] AVR221: Discrete PID Controller ATMEL Data Sheet.

[3] "On Line Auto Tuning of PID Controller using Successive Approximation Method", by Gade S S , Shendage S B and Dr. Uplane M D Published in International Conference ITC2010 Date 12-13 March 2010 "In Press".

[4] Vertikar "Engineering Mathematics" **1**.

[5] Vertikar "Engineering Mathematics" **2**.

[6] "Visual Basic 6.0 Programming" Black Book by Holzner.

[7] "Visual Basic .net Programming" Black Book by Holzner.

[8] Programming and Customizing the 8051 Microcontroller – Myke Predko.

[9] Programming and Customizing the PIC Microcontroller – Myke Predko.

[10] Microcontroller Design Idea Book– Myke Predko.

[11] Embedded Software Architecture Primer – David.

[12] Embedded Systems – Raj Kamal.

[13] I. J. Nagrath, M. Gopal "Control Systems Engineering" 5th edition, New Age International Publication ISBN : 81-224-2008-7.

[14] Help Documents of MATLAB 7.1 "Neural Network Tool Box".

[15] Help Documents of MATLAB 7.1 "Control System Tool Box".