International Journal of Scientific Research in Computer Science and Engineering
Research Paper Vol-1, Issue-3 E-ISSN: 2320-7639

Emphasis on Genetic Algorithm (GA) Over Different PID Tuning Methods of Controlling Servo System Using MATLAB

M. Bandyopadhyay[1], S. Chattopadhyay[*2] and A. Das[3]

Department of Electrical Engineering
National Institute of Technical Teachers' Training and Research, Kolkata
[Under MHRD, Govt. of India]
Block – FC, Sector –III, Salt Lake City, Kolkata -700 106
Tel.:+91 33 23372312, Fax: +91 33 23376331
E-mail: subrata0507@sify.com; mom_saltlake@yahoo.co.in

Available online at: www.isroset.org

Received: 02 May 2013 Revised: 10 May 2013 Accepted: 23 June 2013 Published: 30 June 2013

Abstract: This paper describes Genetic Algorithm, which maintains the accuracy of the output of a system. Here we discuss about the control method of Genetic Algorithm tool on a Servo System. Our objective is to deduce the best tuning method among Genetic algorithm and other conventional tuning methods. The paper presents details on the algorithm and implementation, including the major components in our design: recombination, mutation, fitness function. The algorithm was implemented with Genetic Algorithm tool in MATLAB R2010 for performance evaluation. The simulation showed that the algorithm helped the output to be superior over all the other conventional methods of tuning. First of all the actual response of a "servo-system" is evaluated and the time domain specifications are noted. Thereafter, the specifications with variations of parameters are compared with original system values after the system is tuned by Zeigler-Nichols and Tyreus-Luben method. A greater improvement is observed with the tuning method of Genetic Algorithm.

Keywords: PID Tuning, PID Controller, Genetic Algorithm, MATLAB, Servo System

1. INTRODUCTION

A servo is a closed-loop system [10], [11] with negative feedback. To work the servo system properly, the feedback must always remain negative, otherwise the servo becomes unstable. In practice, it's not as clear-cut as this. The servo can almost become an oscillator, in which case it overshoots and rings following a rapid change at the input.

Tuning a servo system means to adjust the characteristics of the servo so that it follows the input signal as closely as possible. Any closed-loop servo system, whether analog or digital, will require some tuning. Tuning is necessary in a servo system to reduce the system error. A servo system is error-driven, in other words, there must be a difference between the input and the output before the servo will begin moving to reduce the error. The "gain" of the system determines how hard the servo tries to reduce the error. A high-gain system can produce large correcting torques when the error is very small. A high gain is required if the output is to follow the input faithfully with minimal error.

In this paper a servo system is tuned by PID tuning method and Genetic algorithm method and the main objective is to compare the effectiveness of those tuning methods. The basic continuous feedback controller is PID controller which posses good performance. However, Genetic Algorithm [1] is adaptive enough only with flexible tuning. Although many advanced control techniques such as self-tuning control, model reference adaptive control, sliding mode control and fuzzy control have been proposed to improve system performances, the conventional PI / PID controllers are still dominant in majority of real-world servo systems. PID Controller [7] is by far the most widely used control algorithm in the process industry and improvements in tuning of PID controllers will have a significant practical impact on its performance. The PID controller has three principal control effects. The proportional (P) action [1], [2], [3] and [13] gives a change in the controller output directly proportional to the control error. The integral (I) action gives a change in the controller output proportional to the integral error, and its main purpose is to eliminate offset. The less commonly used derivative (D) action is used in some cases to speed up the response or to stabilize the system, and it gives a change in the controller output proportional to the derivative error. The overall controller output V_0 is the sum of the contributions from these three terms. Equation (1) below provide the basic form of the PID filter equations in the continuous time domain and the discrete time domain respectively.

$$V_0 = K_p E(t) + K_i \int_0^t E(t) + \frac{K_d E(t)}{dt} \quad \text{..... (1)}$$

There was a lot of PID tuning techniques [10] developed since the popular [2], [12] Ziegler-Nichols method (Ziegler and Nichols (1942)) appeared. Some of these PID tuning methods were evaluated by Ho et al (1996). They are all together not very reliable (Schlegel (2002)) as they are only heuristic and based on one nominal process

Corresponding Author: *Chattopadhyay*

model. One of the most reputable tuning rules was authored by Astrom and Hagglund (2006) where a large benchmark set of processes was integrated into the design procedure. Though Engineers and Scientists have developed a number of servo compensation schemes over the years, yet the overwhelming favorite for motor positioning is PID loop. The PID position loop [8] requires us three values: the position loop gain (K_p), the integral gain (K_i) and the derivative gain (K_d). Even for the basic servo system, modern motion vendors provide a collection of additional options. The most common of these are an integrator limit, feed-forward gains, motor bias, and frequency-domain filtering such as notch filters or band-pass filters [1]. Cascaded velocity / position loops are both tuned inside and outside, and either four or five parameters are set by the user. The inner velocity loop (usually a PI controller) is tuned first, and then the outer position loop (generally either a PI or PID controller) is tuned.

In this paper, genetic algorithm is used to calculate these parameters. Genetic algorithm is a computational procedure that mimics the natural process of evolution [9]. It is a part of evolutionary computing, a rapidly growing area of artificial intelligence (AI). It is inspired by Darwin's theory of evolution [4] called "Survival of the fittest". It works by evolving population of solutions over a number of generations. For each generation, solutions are selected from the population based on the fitness value. These solutions by crossover (merging previous solutions) and by mutation (modifying the solutions) generate new population [6]. Since it searches many peaks in parallel, the trapping at local minima is avoided. Genetic Algorithm works on a collection of several alternative solutions called population. Each solution or individual in the population is called chromosome and individual character in this is called genes. To obtain better solutions (population) from existing one, a new generation is evolved in each iteration of the Genetic Algorithm [14]. The generation gap is the fraction of individuals in the population that are replaced from one generation to the next. Based on this, there are two basic Genetic Algorithm approaches, called simple Genetic Algorithm and steady state algorithm. Generation gap is equal to one in the simple Genetic Algorithm and is less than one for the other approach. Generation of a new population involves various steps. First evaluate each individual of the population by a user defined fitness function, which is opposite to the error function. Then highly fit individuals are selected from the population for reproduction. Selected individuals form pairs called parents. Different operations for reproduction are crossover and mutation. In the crossover operation, portions of two parents are combined to produce two new individuals, called offspring. This provides a mechanism for the chromosomes to mix and match their desirable qualities in forming offspring. For each pair of parents, crossover is performed with a crossover probability P_c. New features can be introduced into a population by mutation. It produces random changes in the offspring with a probability called mutation probability, P_M . Crossover is the main operation to search the solution space, but does not guarantee the reachability of the entire solution space with a finite population size. Mutation improves search space by introducing new genes into the population. With crossover and mutation there is a high risk that the optimum solution could be lost as there is no guarantee that these operators will preserve the fittest string. To counteract this, a mechanism is used in which, the best individual from a population is saved in the new population. Neighbors of the good solutions are also included in the new generation to improve the search process. When these neighbor solutions of the existing chromosome are evaluated by the algorithm, faster convergence with finer tuning could be achieved. Thus a considerable improvement in the solution quality could be obtained. In Genetic Algorithm the initial generation can be random or user specified. After the reproduction, new generation will replace the old one and evolve until some stopping criterion is met. The total process of evolution is shown by a flowchart, later in this paper.

2. METHOD OF APPROACH

The conventional method for tuning a servo system is done by PID Controller. Here we will be discussing briefly on the tuning methods of (1) PID Controller (2) Ziegler-Nichols method and (3) Tyreus –Luben method.

2.1. P CONTROLLER

The basic block diagram of PID controller is shown below. 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 by adjusting the process control inputs.

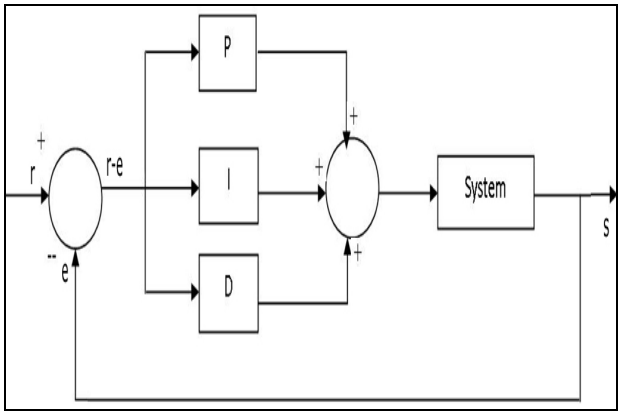

Fig.1. Block diagram of PID controller

2.1.1. EVALUATION

The use of the PID algorithm for control does not guarantee optimal control of the system or system stability. The choice of method will depend largely on whether or not the loop can be taken "offline" for tuning, and the response time of the system.

2.2. ZIEGLER-NICHOLS METHOD

Step1: Determine the sign of process gain.

Step2: Implement a proportional control and introducing a new set-point.

Step3: Increase proportional gain until sustained periodic oscillation.

Step4: Record ultimate gain and ultimate period: K_u and P_u

Step5: Evaluate control parameters as prescribed by Ziegler and Nichols.
It is performed by setting the "I" (integral) and "D" (derivative) gains to zero. The "P" (proportional) gain, K_p is then increased (from zero) until it reaches the ultimate gain K_u, at which the output of the control loop oscillates with a constant amplitude. K_u and the oscillation period T_u are used to set the P, I, D gains depending on the type of controller used.

Table 1. Controller value for Zeigler-Nichols method

Control Type	K_p	K_i	K_d
P	$0.5\,K_u$		
PI	$0.45\,K_u$	$1.2\,(K_p/P_u)$	
PID	$0.6\,K_u$	$2\,(K_p/P_u)$	(K_p/P_u)

2.2.1. EVALUATION

Z-N tuning creates "quarter wave decay". This is an acceptable result for some purposes, but not optimal for all applications. The Z-N tuning rule is meant to give PID loops best disturbance rejection performance. This setting typically does not give very good command tracking performance. Z-N yields an aggressive gain and overshoot – some application wish to instead minimize or eliminate overshoot, and for these Z-N is inappropriate.

2.3. TYREUS-LUBEN METHOD

The Tyeurs-Luben tuning method is another heuristic tuning approach for minimizing error and giving better output. We can see difference by applying the following steps.
Step1: Determine the sign of process gain.

Step2: Implement a proportional control and introducing a new set-point.

Step3: Increase proportional gain until sustained periodic oscillation.

Step4: Record ultimate gain and ultimate period K_u and P_u.

Step5: Evaluate control parameters as prescribed by Tyreus and Luben.

Table 2. Controller value for Tyreus-Luben method

Control Type	K_C	t_I	
PI control	$K_u/3.2$	$2.2P_u$	
PID control	$K_u/2.2$	$2.2P_u$	$P_u/6.3$

2.3.1 EVALUATION

As an alternative to the table above, another set of tuning values have been determined by Tyreus and Luben for PI and PID, often called the TLC tuning rules. These values tend to reduce oscillatory effects and improve robustness.

2.4. GENETIC ALGORITHM METHOD
The flowchart below shows the whole process of evaluation more clearly.

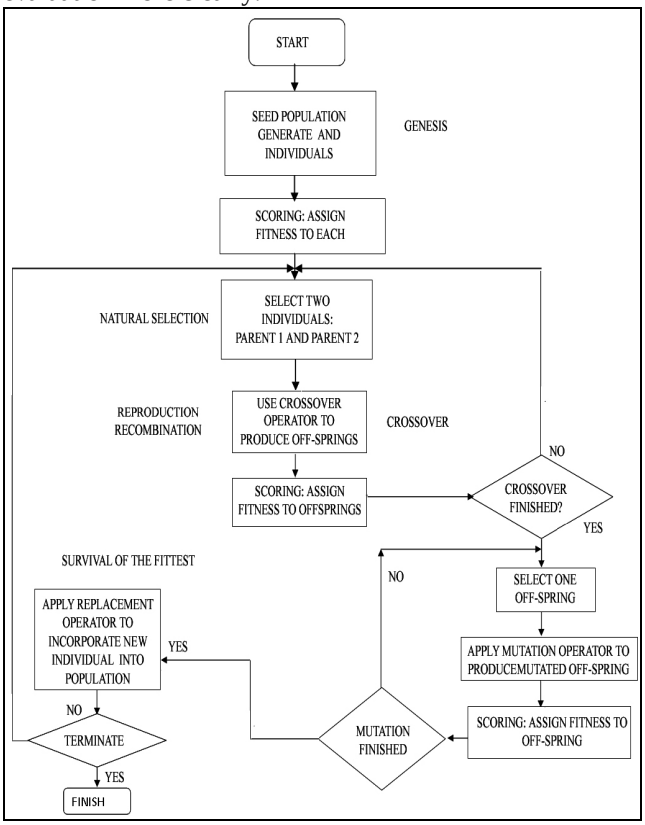

Fig.2. Flowchart of Genetic Algorithm

3. EXPERIMENTAL RESULTS

The experiment is performed by using the software MATLAB R2010. In MATLAB we can tune the system with or without *PID* controller. During tuning without *PID* controller there is a problem of overshoot that occurs in the system. Thereby, we prefer *PID* tuning. For *PID* tuning there are two methods such as: 1. Zeigler-Nichols method and 2. Tyreus-Luben method. But the best method for tuning is Genetic Algorithm method as it gives much improved result in comparison with the other methods. The experimental results are cited in figures 3, 4(a), 4(b) and 5 below based on the following third order closed loop transfer function,

$$G(s) = \frac{100}{0.1s^3 + 2s^2 + 12.5s + 25}$$

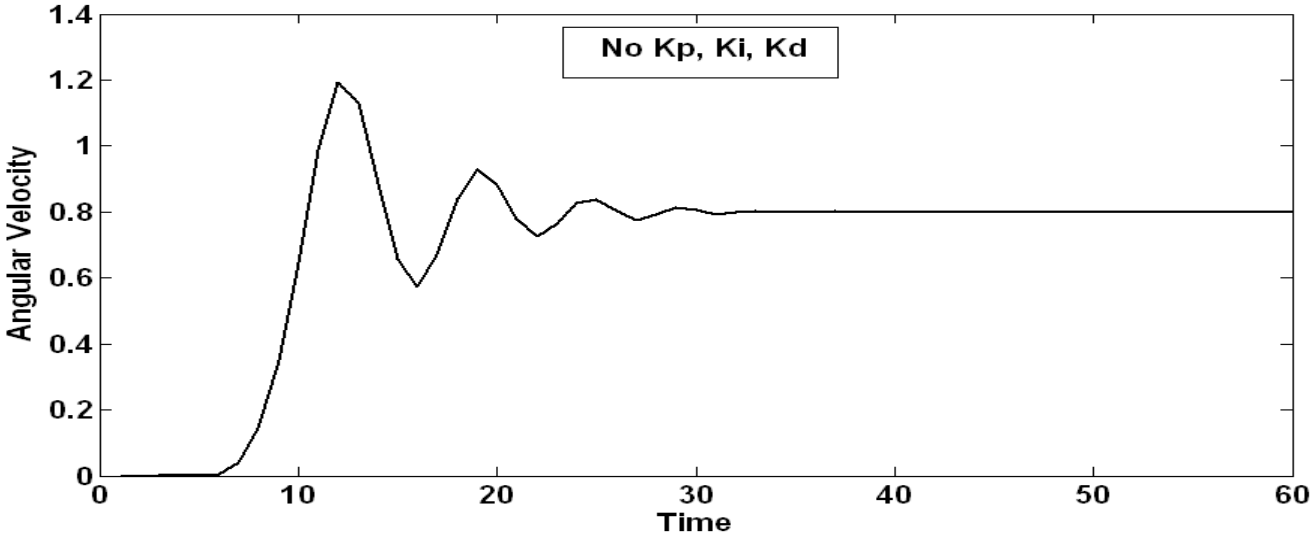

Fig.3. Manual tuning

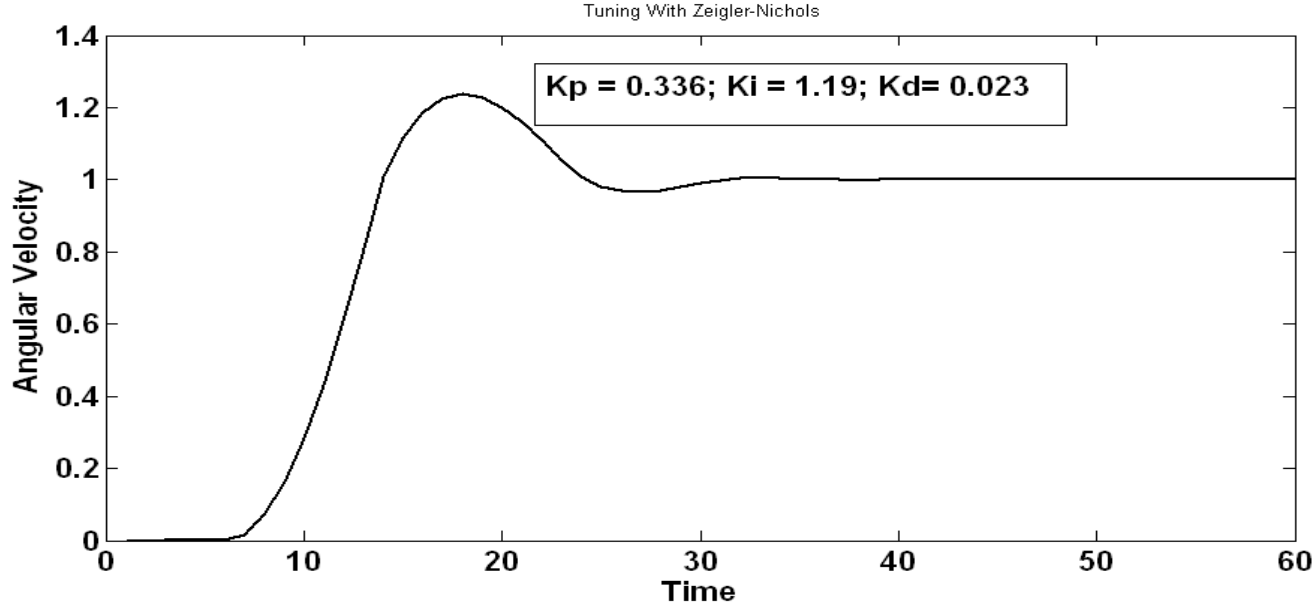

Fig.4(a) Tuning by Zeigler-Nichols

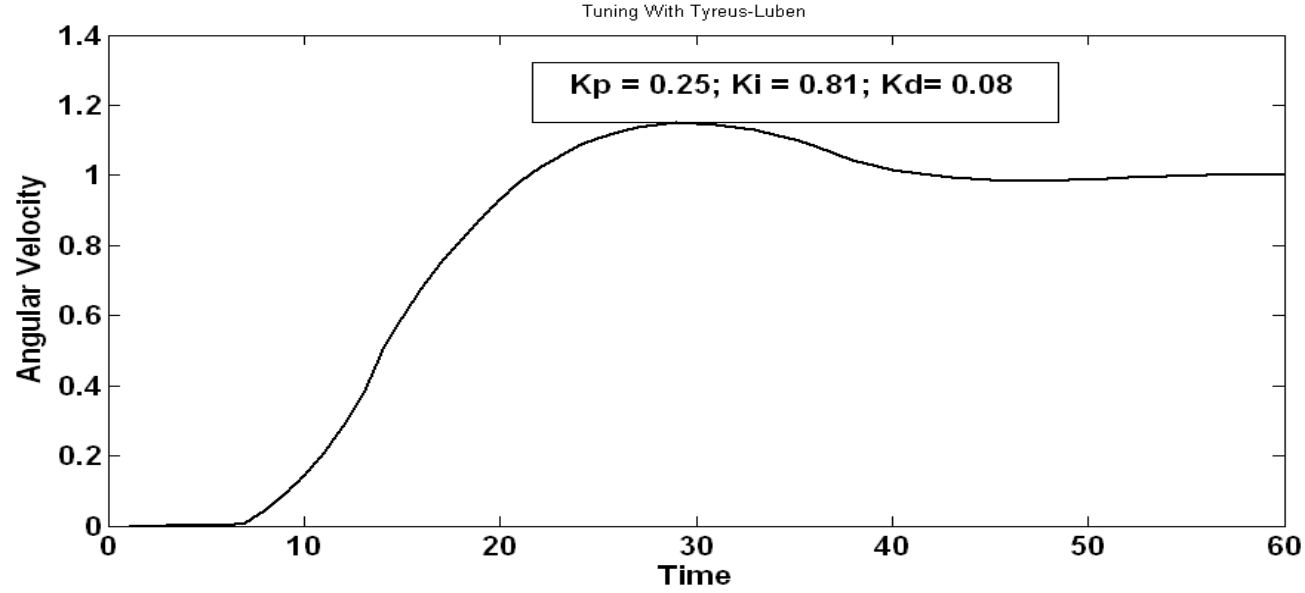

Fig.4(b) Tuning by Tyreus-Luben

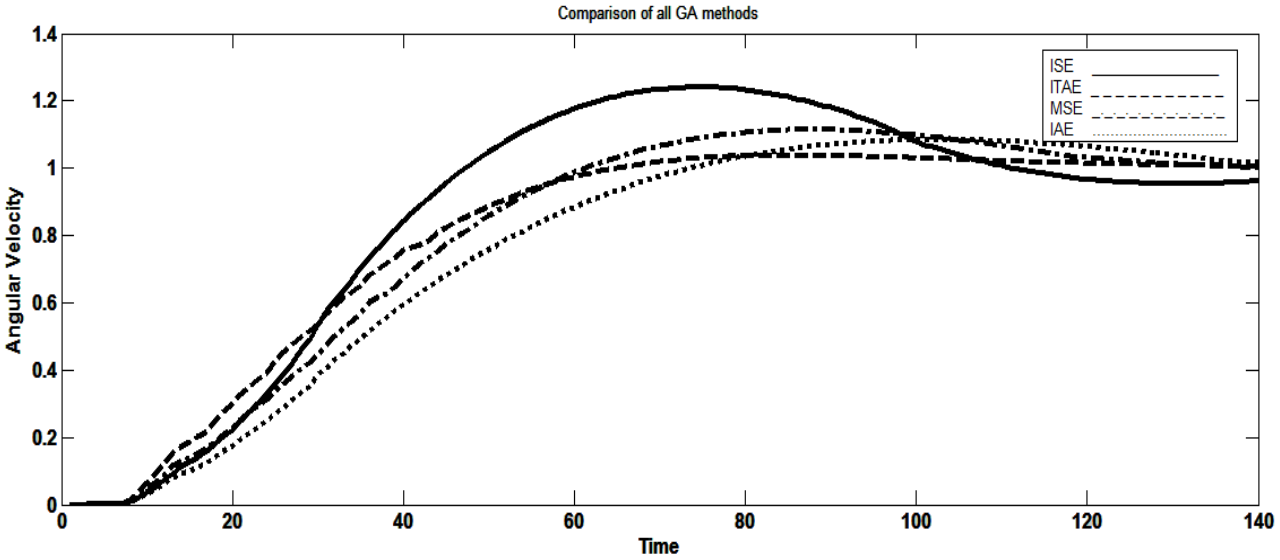

Fig.5. Comparison of all Genetic Algorithm methods

4. DISCUSSION

From the experimental results shown in fig. 3, 4(a), 4(b) and 5 it is observed that , tuning with *PID* controller method, which involves Zeigler- Nichols and Tyreus-Luben method, gives much better result than manual control method. These methods are efficient to decrease peak overshoot, rise time as well as settling time. However, by using *PID* controller , the steady state value of the system has been reached to 1 and so the steady state error has been decreased and as well as percentage error also decreased.

Table 3. Comparison among manual tuning, conventional tuning

PARA-METERS / TUNING METHODS	Rise time	Settling time	Peak overshoot	Steady state value	Steady state error	Percentage error
WITHOUT P CONTROLLER						
MANUAL TUNING	0.723	1.29	1.1926	0.8	0.3926	49
WITH P CONTROLLER						
ZEIGLER-NICHOLS	0.52	0.32	1.237	1	0.237	23.7
TYREUS-LUBEN	0.87	0.58	1.15	1	0.15	15

Table 4. Comparison among different Genetic Algorithm methods of tuning

PARA METERS / TUNING METHODS	Rise time	Settling time	Peak overshoot	Steady state value	Steady state error	Percentage error	Best value
WITH GENETIC ALGORITHM							
ISE	1.79	1.36	1.24	1	0.24	24	0.3816
ITAE	2.35	1.85	1.114	1	0.114	11.4	0.7427
MSE	3.5	2.35	1.085	1	0.08	8	114.87
IAE	2.7	1.95	1.04	1	0.04	4	0.2409

Comparing Tables 3 and 4 it reveals that, the tuning method with genetic algorithm is much better than the tuning method with *PID* controller. Less number of overshoots and decreased steady state error has been observed in Genetic Algorithm methods of tuning. Here percentage error has been reached to a nominal value.
Genetic Algorithm method is no doubt most efficient method in comparison with other conventional method. Still there is a comparison among the sub-methods of Genetic Algorithm, i.e. ISE (Integral Square Error), ITAE (Integral Time Absolute Error), MSE (Mean Square Error), IAE (Integral Absolute Error). Result shows that peak overshoot has been decreased gradually from ISE to IAE. So, the steady state value has also been decreased from 0.24 to 0.04. percentage error has been decreased from 24 to 4. So, IAE gives the best result among the above four sub-methods of Genetic Algorithm and with other conventional methods also.

REFERENCE

[1]. Bindu.R, Mini.K.Namboothiripad "Tuning Of *PID* Controller For DC Servo Motor Using Genetic Algorithm" International journal of emerging technology and advanced engineering, ISSN **2250-2459**, vol 2, issue 3, March **2012**.

[2]. R. Matousek, P. Minar, S. Lang and P. Pivonka, "Efficient Method In Optimal *PID* Tuning" Proceedings of the World Congress on Engineering and Computer Science, vol 1,WCECS, San Francisco,USA, October,**2011.**

[3]. Rahul Malhotra, Yaduvir Singh, Narinder Singh, "Genetic Algorithms:Concepts, Design for Optimazation of Process Controllers", Computer and Infornation science, vol.4, no.2, March **2011**.

[4]. R. C. Chakraborty, "Fundamentals of Genetic Algorithm" , AI course lecture **39-40**, notes, slides; e-mail: rcchak@gmail.com ; June **2010**

[5]. J.Roupec, J:, "Advanced Genetic Algorithms for Engineering Design Problems", Engineering Machines , vol.17, no **5/6, 2010**

[6]. Vladimir Bobal, Petr Chalupa, Marek Kubalcık Petr Dostal "Self Tuning Predictive Control On Non-Linear Servo Motor" , journal of Electrical Engineering, vol.61, no. 6, **365-372**, **2010**.

[7]. Neenu Thomas, Dr.P.Poongodi "Position Control Of DC Motor Using Genetic-Algorithm Based *PID* Controller" Proceedings of The Wrold Congress on engineering, vol.2,WCS 2009, July **2009**

[8]. Sigurd Skogestad "Probably the Best Simple *PID* Tuning Rules in the World" journal of process control, july,**2001**

[9]. MATLAB the mathematical toolbox version 7.6.0.324(R2010a), the product of math work. http://www.mathwork.com/product.

[10]. Dr. Sushil Dasgupta "Control System Theory", Khanna Publishers, Delhi 6, **1983**.

[11]. I. J. Nagrath and M. Gopal "Control System Engineering", New Age International Publishers, New Delhi, Fifth Edition, **2011**.

[12]. M. Cech and M. Schlegel " Computing *PID* Tuning Regions Based on Fractional-Order Model Set", IFAC conference on advances in *PID* control, March, **2012**

[13]. Chuck Lewin, CEO of Performance Motion Devices , 55, Old Bedford Road, Lincoln, MA 01773; e-mail: info@pmdcorp.com ; **2007**.

[14]. M. M. Kanai, J. N. Nderu, P. K. Hinga, "Adaptive *PID* Dc Motor Speed Controller With Parameters Optimized with Hybrid optimization Strategy", 2nd International conference on Advances in Engineering and Technology , **2011**

