Automatic Control of Ball and Beam System Using Particle Swarm Optimization

Muhammad Asif Rana, Zubair Usman, Zeeshan Shareef
Department of Electrical Engineering
Lahore University of Management Sciences
Lahore, Pakistan
asif1253@gmail.com, zubairusman7@hotmail.com, zeeshan_iiee@yahoo.com

***Abstract*— Over the last few decades, many evolutionary algorithms have emerged. One such algorithm is Particle Swarm Optimization which emulates social and cognitive behavior of bird-flocks. In this paper, particle swarm optimization algorithm is presented as a robust and highly useful optimization technique to tune the gains of the PID controllers in the two feedback loops of the classic Ball and Beam Control System. As the name implies, Ball and Beam control system tends to balance a ball on a particular position on the beam as defined by the user. Various trials of the algorithm with varying parameters is implemented on the control system and the fitness of the response of the system to a unit step input is used as a criterion for judging the most optimal solution. A slight variation in the fitness value function is also introduced, rather than using conventional performance integrals as a criterion. Furthermore, the results from PSO tuning are quantitatively compared to ITAE equations method of PID tuning and Fuzzy-logic controller to reach a conclusion on the most efficient controlling technique.**

Keywords- Particle Swarm optimization; Ball and Beam system; fitness function; ITAE equations; Fuzzy-logic controller

I. INTRODUCTION

The ball and beam control system has always remained one of the favorite introductory control system problems for control engineers. As the name implies, in this system, a ball is placed on a beam and the control system should tend to balance the ball at particular position on the beam as prescribed the user. Since the ball position is unstable in open-loop, there is a need for a feedback loop with a suitable controller to minimize the error. The work of a robust controller should be to provide least possible transients in the step-response, while minimizing the steady-state error. PID (Proportional, Integral, and Derivative) controllers have always remained one of the highly-demanded choices for such applications [1]. However, tuning a PID controller to acquire the most optimum results is a tricky task and many tuning techniques have been used in the past to do so. The traditional PID tuning methods include: Ziegler-Nichol's method [2], Gain & phase margin tuning method [3], ITAE equations method [4], SISO tool and so on [5]. Furthermore, many unconventional techniques have been developed over the past few decades which have proven to be very fruitful in control systems. Fuzzy-logic and Evolutionary algorithms [6] are examples of such unconventional methods.

Particle Swarm Optimization algorithm is one such evolutionary algorithm which can be used in tuning PID controllers. The algorithm simulates the movement of birds in flocks. It was in 1995, that particle swarm optimization was first presented as a tool for optimization of functions by Kennedy and Eberhart [7].

The algorithm works on the scenario of birds randomly searching for food. It begins with a swarm of birds/particles being initialized at random positions in the problem-space. Typically, the numbers of birds in the swarm (n) is kept in the range 20-40. During the journey, each bird keeps track of its coordinates in the problem space. A record is also kept of the least distance (fitness value) from the food achieved by each individual bird so far, called *pbest* (personal best). The minimum value of *pbest* in the swarm is called *gbest* (global best), that is, the minimum distance achieved by any of the birds in the swarm so far.

On each iteration, the velocity of each bird is updated and added to the current coordinates of the respective birds.

$$\text{position}_{new}[\,] = \text{position}_{old}[\,] + \text{velocity}_{new}[\,] \quad (1)$$

The velocity is governed by three factors: (1) inertia ($velocity_{old}$); (2) cognitive influence(*pbest*); and (3) social influence(*gbest*). The directions of cognitive and social influences depend on the direction vector of the bird's position from *pbest* and *gbest* respectively. Both these influences are updated on each iteration, and added to inertia to get the new velocity. Furthermore, randomness is maintained in the system by weighing the influences with random numbers between 0 and 1, called learning factors (c_1 and c_2) and the inertia with another random number in the same range, called inertia factor (w) [8].

$$\begin{aligned}\text{velocity}_{new}[\,] = {} & w\times\text{velocity}_{old}[\,] \\ & + c_1\times\text{rand}\times(\text{pbest}[\,] - \text{position}[\,]) \\ & + c_2\times\text{rand}\times(\text{pbest}[\text{gbest}] - \text{position}[\,])\end{aligned} \quad (2)$$

Innovations have been continually introduced into the algorithm, with almost all having the objective of decreasing the time required to achieve the optimum solution. These include use of linearly decreasing inertia factor(w) [8], introduction of constriction factor(χ) [9], development of Binary PSO[10], Bare Bone PSO[11], and Fully informed PSO (FIPS) [12].

Experiments have also been carried out to find out the most optimum values for the parameters in the PSO algorithm and

978-1-4577-0045-3/11/$26.00 ©2011 IEEE

the most optimum bounds to be imposed on the parameters to be tuned[8][13]. The parameters must be such that there is a balance between the social and cognitive behavior, side-by-side maintaining equilibrium between stability and randomness in the swarm. However, there is no proper method as to how the parameters should be calculated. We have considered the work by Shi and Eberhart as our starting point in selecting the PSO parameters [8]. An analysis of the 30 trials performed by them makes it evident that decreasing the inertia weight factor, decreases the number of iterations required to achieve the global optimum. However, having a smaller inertia weight increases the probability of failure in achieving the global optimum. On the other hand, a very large inertia weight also leads to a higher failure rate. The PSO parameters are further varied by hit and trial unless a feasible solution is achievable within the given number of iterations.

The rest of the paper deals with the implementation of Particle Swarm Optimizer to ball and beam control system. A quantitative comparison is then made with ITAE equations method and Fuzzy-logic control method, to decide the most efficient method.

II. Dynamics of Ball and Beam System

The practical setup is the same as used by Amjad et al [14] and the system is shown in figure 1. Hence the system dynamics remain intact. The transfer functions are mentioned in this section again briefly.

To understand the complete dynamics of the setup accurately, the transfer functions of the motor and the ball on beam system need to be analyzed separately. The results can later be used to obtain the overall response of the system.

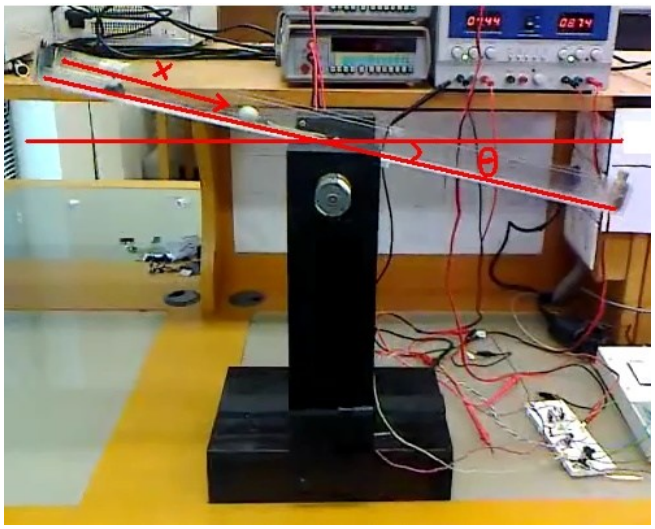

Figure 1. A cross sectional view of the system

A. Model for Motor

The position of a DC motor is observed to have a transfer function [15]:

$$G_m(s) = \frac{\theta(s)}{V(s)} = \frac{K}{s(\tau s + 1)} \tag{3a}$$

Based on experimental deduction of the motor parameters, K= O.7/rev/sec/volts, $\tau = 0.014$ sec,

$$G_m(s) = \frac{0.7}{s(0.014s + 1)} \tag{3b}$$

B. Model for Ball and Beam system

According to Rosales [16] and others [17][18][14], the linearized model of the ball and beam setup within small angles around $\theta = 0^o$ is:

$$G_{bb}(s) = \frac{X(s)}{\theta(s)} = \frac{g}{s^2[1 + \frac{2}{5}(\frac{a}{a'})^2]} \tag{4a}$$

Where, a is the radius of the ball, a' is the distance between the axis of rotation of the ball and point of contact of the ball with the beam, and g is the acceleration due to gravity.

Taking g to be 9.81 m/s^2 and substituting in the mathematical expression will yield:

$$G_{bb}(s) = \frac{1}{1.713s^2} \tag{4b}$$

C. Model for the whole system

Combining the results from the previous subsections, the open-loop transfer function of the ball and beam setup as well as the motor becomes,

$$G_{total}(s) = G_m(s) \times G_{bb}(s) = \frac{0.7}{s^3(0.02398s + 1.713)} \tag{5}$$

III. Controller Tuning and Implementation

The block diagram of the whole system is shown in Figure 2. The system consists of two loops: (1) Inner motor control loop, (2) outer ball and beam control loop. The design strategy is to first stabilize the inner loop followed by the outer loop.

This section presents three methods for controlling the loops: (1) PID control with Particle Swarm Optimization, (2) PD and PI control with standard ITAE equations, (3) PD and fuzzy-logic control

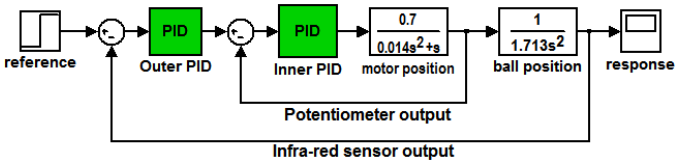

Figure 2. Block-diagram of the ball and beam system. PID controllers highlighted in green are to be tuned

A. PID tuning with Particle Swarm Optimization

Before implementing the particle swarm optimization technique on a problem, one has to consider first the number of variables to be tuned in order to optimize the solution to the problem. The number of dimensions of the matrices in the algorithm must equal the number of variables in the system.

1) Defining the PSO parameters and bounds on PID gains

In our case, the food is analogous to reference position of the ball, defined by the user. The problem-space at hand is 6 dimensional, since we are required to tune 6 PID gains.

Since each iteration takes a moderately large time, we wish to achieve the global optimum within a comparatively small number of iterations, hence we keep $w=0.4$ so that there is a higher probability of attaining global optimum within the given number of iterations. On the other hand, we have not kept $w \approx 0$ since this would greatly increase the failure rate [8]. The number of birds is kept at $n=70$ which is more than enough for our control problem. c_1 is kept at 1.2 and c_2 is set to 1.6. This gives a higher weightage to the social influence.

$$\begin{aligned} velocity_{new}[\,] = {} & 0.4\times velocity_{old}[\,] \\ & + 1.2\times rand\times (pbest[\,] - position[\,]) \\ & + 1.6\times rand\times (pbest[gbest] - position[\,]) \end{aligned} \quad (6)$$

Table I show the upper and lower limits set for the 6 PID controller gains. The birds/particles are initialized at random positions within these limits and are not allowed to trespass during the algorithm.

TABLE I. THE BOUNDS IMPOSED ON PID PARAMETERS

	Outer-loop PID			**Inner-loop PID**		
	K_P	K_D	K_I	K_P	K_D	K_I
Lower bound	5	2	0	5	0	0
Upper bound	15	8	1	15	1	2

2) Fitness Value

The fitness value needs to be selected according to the design requirements. The following fitness criterion is proposed and applied. In this function, the weighing factors can be varied according to the design requirements. For example, a control designer concerned with minimizing overshoot alone would set $a_1=1$ and other factors to zero.

$$fitness = a_1\times M_p + a_2\times t_r + a_3\times t_s + a_4\times|e_{ss}| \quad (7)$$

Where a_1, a_2, a_3, a_4 are weighing factors and M_P, t_r, t_s and e_{ss} are peak-overshoot, rise-time, settling-time, and steady-state error respectively. We have set $a_1=a_2=a_3=a_4=1$.

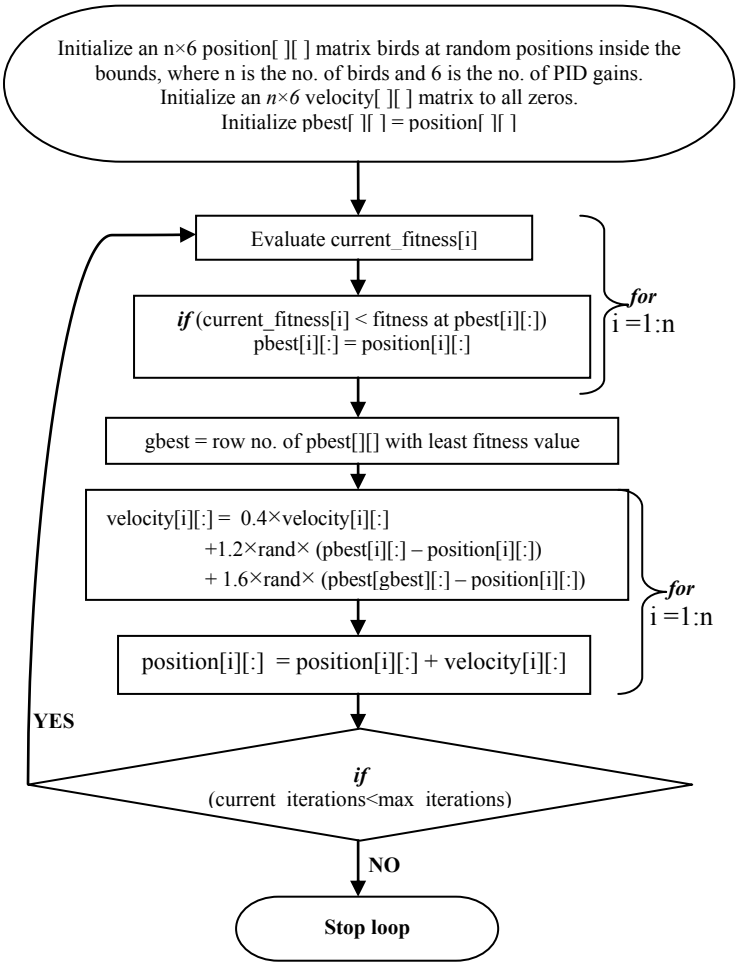

Figure 3. Flowchart of the PSO algorithm

Alternatively, any performance integral can also be used as fitness criterion e.g. IAE (Integral Absolute Error), ITAE (Integral Time Absolute Error), ISE (Integral Squared Error) etc.

3) Finding most optimum solution

A unit step for ball-position is applied as a reference to the system in figure 1. The algorithm, as explained in figure 3, is run around 50 times with varying number of iterations from 20 to 300. During the whole process, it also becomes apparent that a local or global optimum is usually achievable within the given number of iterations if they are kept more than 50.

TABLE II. PID GAINS OBTAINED FOR THE 5 BEST PSO TRIALS

	Outer-loop PID			Inner-loop PID		
	K_P	K_D	K_I	K_P	K_D	K_I
trial 1	14.9998	7.9094	1.3186×10^{-5}	11.9813	0.0017	0.1258
trial 2	14.9915	7.6728	1.2423×10^{-7}	14.9023	1.2511×10^{-8}	1.2701
trial 3	13.7210	7.4379	0.0097	14.9699	5.3954×10^{-8}	1.9487
trial 4	14.9986	8.0000	0.0398	13.9506	1.7297×10^{-13}	0.1424
trial 5	14.8617	7.9298	7.9953×10^{-15}	14.9999	0.2138	1.9365

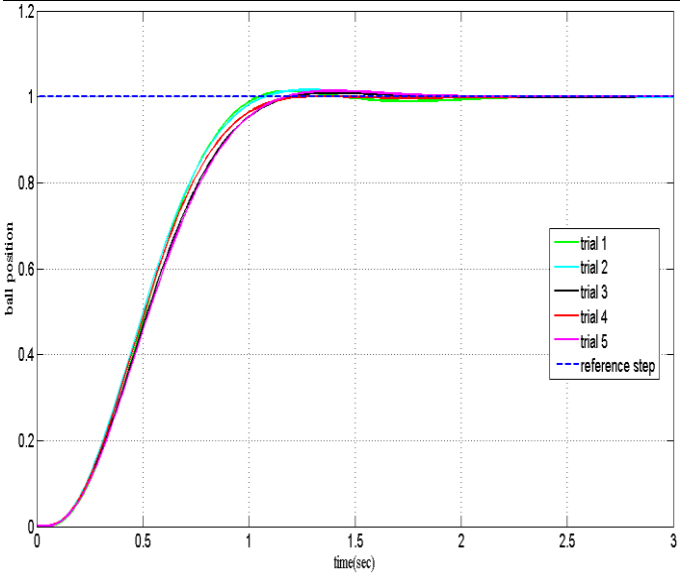

Figure 4. Step-responses of the 5 best PSO trials

Figures 5 and 6 show the zoomed-in versions of the step responses in figure 4.

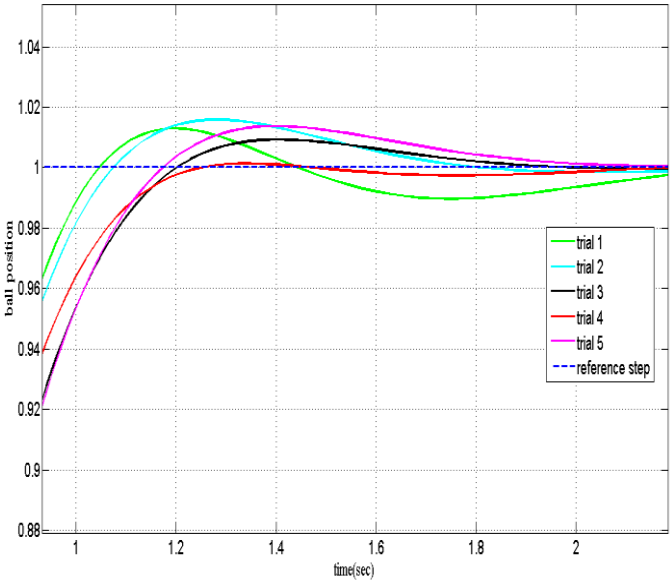

Figure 5. Over-shoots and under-shoots of the 5 best PSO trials

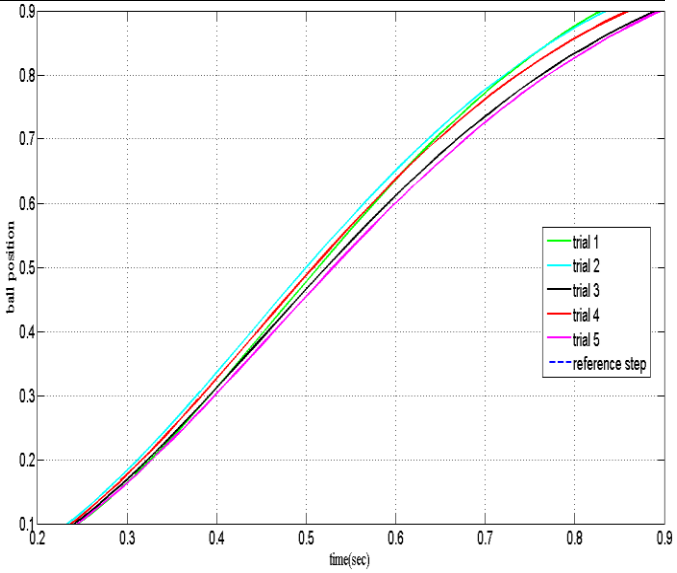

Figure 6. Rise times of the 5 best PSO trials

Based on the comparison in TABLE III, trial 4 seems the most optimum solution to our ball and beam control system with a negligibly small peak overshoot and peak undershoot and a pretty fair rise time and settling-time. The steady-state error in trial 4, although larger than other trials, is still very small.

TABLE III. COMPARISON OF PSO TRIALS

	trial 1	trial 2	trial 3	trial 4	trial 5
Rise time	0.5880	0.6060	0.6540	0.6290	0.6520
Peak overshoot (%)	1.700	2.070	1.357	0.496	1.264
Decay ratio	0.4529	0.0580	0.0442	0.4028	0.0395
Steady-state error	1.7949×10^{-6}	3.1566×10^{-6}	3.8728×10^{-4}	1.400×10^{-3}	5.382×10^{-4}
IAE value	0.5382	0.5312	0.5584	0.5462	0.5642
Settling-time (±5%)	0.9120	0.9240	0.9940	0.9660	0.9930

B. PD and PID controller designing using ITAE equations

There are number of classic techniques to design the PID controller which involves Ziegler and Nichols, SISO Tool, ITAE Equations, root locus etc. In this design, we have used the ITAE equations for the outer-loop PD and inner-loop PID gains computation

The main advantage for using the ITAE equation method over other conventional tools is that there is no requirement for any graph or chart like the root locus and the bode plot. These equations can easily be used for the gains calculation of up to 6th order characteristics equation. After calculating the gains from the ITAE equations we usually require a little bit tuning of the gains.

The ITAE performance integral is defined as:

$$\text{ITAE} = \int_0^T t|e(t)|\,dt \qquad (8)$$

The procedure for calculating the PID gains using the ITAE equations is as follows. These steps are carried out for first for inner-loop and then the outer-loop:

Step-1: We multiply the general form of the PID controllers with the open loop transfer function.
Step-2: The characteristics equation is compared with the standard ITAE equations. The value of damping ratio (ζ) is constant that is 0.6.

Using standard equations of ITAE, the PID controllers designed for the inner and outer loop respectively can be written as [14], [4]:

Inner loop PD controller: $K_P = 11$; $K_D = 2$
Outer loop PID controller: $K_P = 6$; $K_I = 0.2$; $K_D = 5$

C. *Fuzzy-logic controller and PD controller implementation*

Fuzzy-logic controller has been of high interest to control engineers who deal with complex industrial control problems which cannot be tuned by conventional methods. Unlike PSO, fuzzy-logic controller makes use of human thinking and logic (fuzzy-logic) rather than behavior of birds in flocks. [19]

Implementation of a fuzzy-logic controller (FLC) on this system has been presented by Amjad et al [20]; the same FLC has been used here. A FLC for the outer loop and a PD controller using ITAE equations (found in previous section) for the inner loop, of this ball and beam system were implemented. In this controller for the fuzzification process the triangular member functions are used for input as well as for output. The defuzzification technique in this case that gave the least integral square error was the Center of Gravity approach. The response of the system with fuzzy logic controller is shown in Figure-7(green curve) for comparison.

IV. Comparison of PSO with other techniques

The input step response of the control system tuned by PSO has negligible transients as compared to that tuned by ITAE equations. The steady-state error is however minimum for FLC controller, but the trade-off between steady-state error and the rise-time is large enough that the steady-state error can be ignored. Hence, it can be concluded that PSO tuning is a much better option for tuning PID than the conventional ITAE method of tuning.

V. Conclusion

In this paper, Particle swarm optimization algorithm is successfully presented as a useful tool for optimizing the step-response of a ball on beam control system. An analytical approach has been used to choose the best trial among the various trials of the algorithm on the system model. Finally, the step-response is compared with a step-response optimized through ITAE equations and Fuzzy-logic Controller. The Particle Swarm Optimizer out-classes the other two techniques. This is mainly because this algorithm efficiently searches the entire 6-dimensional problem space for the optimum solution. Unlike conventional methods, it provides a general and easy-to-use method for tuning any control problem, without the need of complex analysis of the system.

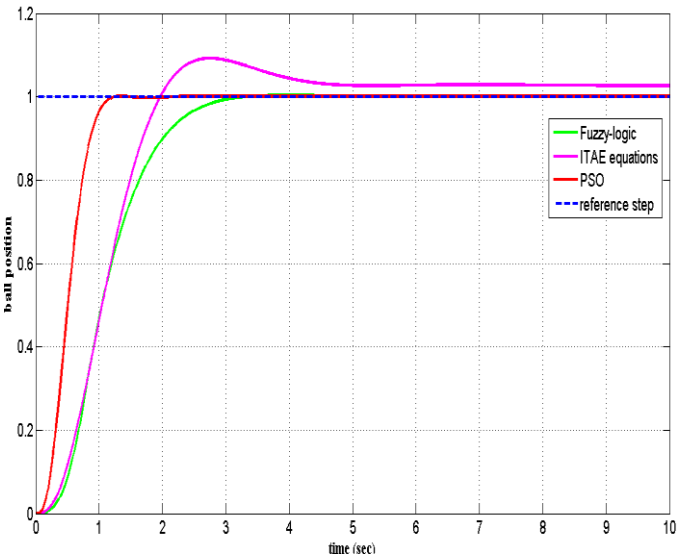

Figure 7. Step responses of ball and beam system with various controllers

TABLE IV: COMPARISON OF STEP RESPONSE CHARACTERISTICS OF THE CONTROLLERS

	PSO	**ITAE**	**Fuzzy-logic**
Rise time	0.6290	1.2380	1.4800
Peak overshoot	0.4958%	4.9501%	0.3217%
Steady-state error	$1.400\text{x}10^{-3}$	$2.5900\text{x}10^{-2}$	$1.1256\text{x}10^{-6}$
IAE value	0.5462	1.3255	1.1742
Settling time (±5%)	0.9660	2.7420	2.3400

References

[1] Araki M., "PID control", *Control systems, robotics, and automation*, vol. 2, Encyclopedia of Life Support Systems (EOLSS).

[2] Ziegler J. G. et al,"Optimum settings for automatic controllers", *Transactions of ACME*, 1942, vol.64, pp. 759-768.

[3] Astrom K. J. and Hagglund T., "Automatic tuning of simple regulators with specifications on phase and amplitude margins", *Automatica*, 1984, vol. 20, pp. 645-651.

[4] D'Azzo J. J. and Houpis C.H., "Linear control system analysis and design: conventional and modern", *McGraw-Hill Series in Electrical and Computer Engineering*, New York 1995, 4th edition.

[5] Astrom K. J. and Hagglund T., "PID Controllers: Theory, Design, and Tuning", *Instrument Society of America*, 1995, 2nd ed., pp. 134-229.

[6] Zang H., Zhang S. and Hapeshi K., "A review of nature-inspired algorithms", *Journal of Bionic Engineering*, September 2010, vol. 7, Supplement 1, pp. S232-S237.

[7] Kennedy J. and Eberhart R., "Particle Swarm Optimization", *IEEE International Conference on Neural Networks*, Perth – Australia, 1995, pp. 1942-1948.

[8] Shi, Y. and Eberhart R., "A modified particle swarm optimizer", *Evolutionary Computation Proceedings*, *IEEE World Congress on Computational Intelligence*, Anchorage– USA, 4-9 May 1998, pp.69-73.

[9] Clerc M. and Kennedy J., "The particle swarm - explosion, stability, and convergence in a multidimensional complex space", *IEEE Transactions on Evolutionary Computation*, Feb 2002, vol.6, no.1, pp. 58-73.

[10] Kennedy J. and Eberhart R., "A discrete binary version of the particle swarm algorithm", *Systems, Man, and Cybernetics*, *IEEE International Conference on Computational Cybernetics and Simulation*, 12-15 Oct 1997, vol. 5, pp. 4104-4108.

[11] Kennedy J., "Bare bones particle swarms", *Proceedings of the 2003 IEEE Swarm Intelligence Symposium*, 24-26 April 2003, pp. 80- 87.

[12] Mendes R., Kennedy J. and Neves J., "The fully informed particle swarm: simpler, maybe better", *IEEE Transactions on Evolutionary Computation*, June 2004, vol.8, no. 3, pp. 204- 210.

[13] Chuan L. and Quanyuan F., "The Standard Particle Swarm Optimization Algorithm Convergence Analysis and Parameter Selection", *Third International Conference on Natural Computation*, 24-27Aug. 2007, vol. 3, pp. 823-826.

[14] Amjad M. et al, "A simplified intelligent controller for ball and beam system", *2nd International Conference on Education Technology and Computer*, 22-24 June 2010, vol.3, pp.V3-494-V3-498.

[15] Cockburn J. C. and Savakis A., "DC Motor Transfer Function", *http://www.ce.rit.edu/~cockburn/courses/ce553_su02/labs/lab3_r1.pdf*

[16] Rosales E. A., "A Ball-on-Beam Project Kit", *Department of Mechanical Engineering, Massachusetts Institute of Technology*, June 2004

[17] Hirsch R., "Mechatronic Instructional Systems Ball on Beam System", *Shandor Motion Systems*, 1999. *http://www.ro.feri.uni-mb.si/predmeti/skup_sem/projekt1/shandor.pdf*.

[18] Wellstead P., "Ball and Beam I: Basics", *http://www.controlsystemsprinciples.co.uk/whitepaperslball-and-beam*.

[19] Lee C. C., "Fuzzy logic in control systems: fuzzy logic controller -Part I and II", *IEEE Transactions on System, Man and Cybernetics*, Mar-Apr 1990,vol. 20, no. 2, pp. 404-435.

[20] Amjad M. et al, "Fuzzy Logic Control of Ball and Beam",*2nd International conference on Electronic Computer Technology (ICECT 2010)*, Kuala Lumpur, Malaysia, 22-24 June 2010, vol. 2, pp. V3-489 - V3-493.