Modern Optimization Techniques for PID Parameters of Electrohydraulic Servo Control System

Mohammed Alhanjouri
Computer EngineeringDepartment
Islamic University of Gaza
Gaza, Palestine
mhanjouri@iugaza.edu.ps

***Abstract*—**Electrohydraulic servo system has been used in industry in a wide number of applications. Its dynamics are highly nonlinear and also have large extent of model uncertainties and external disturbances. In order to in-crease the reliability, controllability and utilizing the superior speed of response achievable from electrohydraulic systems, further research is required to develop a control software has the ability of overcoming the problems of system nonlinearities. In This paper, a Proportional Integral Derivative (PID) controller is designed and attached to electrohydraulic servo actuator system to control its stability. The PID parameters are optimized by using four techniques: Particle Swarm Optimization (PSO), Bacteria Foraging Algorithm (BFA), Genetic Algorithm (GA), and Ant colony optimization (ACO). The simulation results show that the steady-state error of system is eliminated; the rapidity is enhanced by PSO applied on Proportional Integral Derivative (PPID), Bacteria Foraging Algorithm applied on Proportional Integral Derivative (BPID), GA applied on Proportional Integral Derivative (GPID), and ACO Algorithm applied on Proportional Integral Derivative (ACO-PID) controllers when the system parameter variation was happened, and has good performances using in real applications. A comparative study between used modern optimization techniques are described in the paper and the tradeoff between them.

Keywords - *Electrohydraulic, PSO, BFA, ACO, GA, PID.*

*****

I. INTRODUCTION

Electrohydraulic servo-systems are widely used in many industrial applications because of their high power-to- weight ratio, high stiffness, and high payload capability, and at the same time, achieve fast responses and high degree of both accuracy and performance [1, 2]. However, the dynamic behavior of these systems is highly nonlinear due to phenomena such as nonlinear servo-valve flow-pressure characteristics, variations in trapped fluid volumes and associated stiffness, which, in turn, cause difficulties in the control of such systems.

Control techniques used to compensate the nonlinear behavior of hydraulic systems include adaptive control, sliding mode control and feedback linearization. Adaptive control techniques have been proposed by researchers assuming linearized system models. These controllers have the ability to cope with small changes in system parameters such as valve flow coefficients, the fluid bulk modulus, and variable loading. However, there is no guarantee that the linear adaptive controllers will remain globally stable in the presence of large changes in the system parameters, as was demonstrated experimentally by Bobrow and Lum [3]. These controllers are robust to large parameter variations, but the nearly discontinuous control signal excites unmolded system dynamics and degrades system performance. This can be reduced by smoothing the control discontinuity in a small boundary layer bordering the sliding manifold as introduced in simulations [4, 5]. The nonlinear nature of the system behavior resulting from valve flow characteristics and actuator nonlinearities has been taken into account in application of the feedback linearization technique [6]. The main drawback of the resulting linearizable control law is that it relies on exact cancellation of the nonlinear terms.

In nowadays industry field, the PID control that has the characters of simple arithmetic, small static error, good dynamic and steady performance, is widely used. But there are some control objects with non-linearity, time lags, strong coupling and high-order in modern industry. For these systems, traditional PID control can't provide content efforts. [1]

Over the past a few years, many different techniques have been developed to acquire the optimum control parameters for PID controllers. The academic control community has developed many new techniques for tuning PID controllers such as Genetic Algorithm (GA), Ant Colony Optimization (ACO), Bacteria Foraging Algorithm (BFA) and Particle Swarm Optimization (PSO). A BFA is one such direct search optimization techniques which are based on the mechanics of natural bacteria and A PSO is one such direct search optimization techniques which are based on the behaviour of a colony or a swarm of insects, such as ants, termites, bees and wasps. [2]

Advantages of the GA, ACO, BFA and PSO for auto tuning are that they do not need gradient information and therefore can operate to minimize naturally defined cost functions without complex mathematical operations. [3][4]

This paper describes the application of GA, ACO, BFA and PSO techniques based on the transfer function was determined in [5] to optimal tuning the three terms of the classical PID controller to Electrohydraulic Servo Control System.

II. SYSTEM STATE SPACE DYNAMIC MODEL

The mathematical model can be deduced based on hydraulic control theory which can provide evidence for model identification. In this paper, the mathematical model of each

part in the system and the simplified mathematical model of the system are given directly.

1) Servo Magnifying: This link can be simplified into proportional component, the gain is K,

2) The Electro-hydraulic Servo Valve: The transfer function is represented by oscillational element, that is.

$$G_v(s) = \frac{Q_0}{\Delta I} = \frac{K_v}{s^2/\omega_v{}^2 + 2\gamma_v s/\omega_v + 1} \quad (1)$$

Where, G_v is the transfer function, Q_0 is the unloading flow, ΔI is the increment of input current, G_v is the flow gain of the electro-hydraulic servo valve, ω_v is the nature frequency, ε_v is the damping ratio, dimensionless.

3) The symmetrical Hydraulic Cylinder: The transfer function of hydraulic cylinder x_p relative to the hydraulic cylinder x_v

$$G_0(s) = \frac{K_q/A_p}{s(s^2/\omega_h{}^2 + 2\gamma_h s/\omega_h + 1)} \quad (2)$$

Where, K_p is the flow gain, ω_h is the hydraulic natural frequency, ε_h is the damping ratio

4) The Position Sensor: The place of the feedback can be represented by the K proportional component f

5) Simplified Transfer Function of the System: The simplified mathematical model can be obtained by the mechanism analysis of electro-hydraulic servo system mathematical model:

$$G_0(s) = \frac{K}{s(s^2/\omega_h{}^2 + 2\gamma_h s/\omega_h + 1)(s^2/\omega_v{}^2 + 2\gamma_v s/\omega_v + 1)} \quad (3)$$

The simplified mathematical model controlled by the electrohydraulic position servo control system can be seen as a fifth order system. The different models obtained by on-line identification can be compared with the identification toolbox in MATLAB. Choose the best mode as the identification model. After on-line identification and proper correction of the model gain, the closed-loop transfer function of the identification model is that

$$\varphi(s) = \frac{0.0062s^2 - 4.07s + 2925}{s^3 + 9.43s^2 + 13.68s + 2925} \quad (4)$$

The model obtained by identification is a closed loop model of cylinder controlled by the servo valve. After analysis, the open loop transfer function of position servo system is: [5]

$$G(s) = \frac{0.0062s^2 - 4.07s + 2925}{s^3 + 9.43s^2 + 13.11s} \quad (5)$$

III. PID CONTROLLER TUNING

The popularity of PID controllers in industry stems from their applicability and due to their functional simplicity and reliability performance in a wide variety of operating scenarios. Moreover, there is a wide conceptual understanding of the effect of the three terms involved amongst non-specialist plant operators. In general, the synthesis of PID can be described by,

$$u(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt} \quad (6)$$

Where e(t) is the error, u(t) the controller output, and KP, KI, and KD are the proportional, Integral and derivative gains.

There is a wealth of literature on PID tuning for scalar systems, [5-7]. Good reviews of tuning PID methods are given in Tan et al. [8] and Cominos and Munro [9]. Among these methods are the well-known Ziegler and Nichols [10], Cohen and Coon [11]. Many researchers have attempted to use advanced control techniques such as optimal control to restrict the structure of these controllers to PID type.

For instance, in some systems with fast output, this method can't get perfect result or realize the real-time and automatic control, so some other controller is needed in these fields. The effect of PID controller is showed in Figure 1. [5]

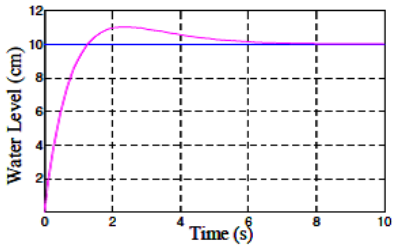


Figure 1. Control effect of PID controller

IV. Bacterial Foraging Algorithm (BFA)

Recently, bacterial foraging algorithm (BFA) has emerged as a powerful technique for the solving optimization problems. BFA mimics the foraging strategy of E. coli bacteria which try to maximize the energy intake per unit time. From the very early days it has drawn attention of researchers due to its effectiveness in the optimization domain. So as to improve its performance, a large number of modifications have already been undertaken. The bacterial foraging system consists of four principal mechanisms, namely chemotaxis, swarming, reproduction and elimination-dispersal. A brief description of each of these processes along with the pseudo-code of the complete algorithm is described below.

Chemotaxis: This process simulates the movement of an E.coli cell through swimming and tumbling via flagella. Biologically an E.coli bacterium can move in two different ways. It can swim for a period of time in the same direction or it may tumble, and alternate between these two modes of operation for the entire lifetime. Suppose $\theta^i(j,k,l)$ represents ith bacterium at jth chemotactic, kth reproductive and lth elimination-dispersal step. C(i) is the size of the step taken in the random direction specified by the tumble (run length unit). Then in computational chemotaxis the movement of the bacterium may be represented by.

$$\theta^j(j+1,k,l) = \theta^i(j,k,l) + c(i)\frac{\Delta(i)}{\sqrt{\Delta^t(i)\Delta(i)}} \quad (7)$$

Where Δ indicates a vector in the random direction whose elements lie in [-1, 1].

Swarming: An interesting group behavior has been observed where a group of E.coli cells arrange themselves in a traveling ring by moving up the nutrient gradient when placed amidst a semisolid matrix with a single nutrient chemoeffecter. The cells, when stimulated by a high level of succinate, release an attractant aspertate, which helps them to aggregate into groups and thus move as concentric patterns of swarms with high bacterial density. The cell-to-cell signaling in E. coli swarm may be represented by the following function.

$$J_{cc}(\theta, P(j,k,l) = \sum_{}^{S} J_{cc}(\theta, \theta^{i}(j,k,l) \qquad (8)$$

$$J_{cc} = \sum_{i=1}^{S} \left[-d_{attrac} \tan t^{e^{(-w_{attrac} \tan t \sum_{m=1}^{P} (\theta_m - \theta_m{}^{i})^2}} \right] + \sum_{i=1}^{S} \left[-h_{repellant} {}^{e^{(-w_{repellant} \sum_{m=1}^{P} (\theta_m - \theta_m{}^{i})^2}} \right]$$

where $J_{cc}(\theta, P(j,k,l)$ is the objective function value to be added to the actual objective function (to be minimized) to present a time varying objective function, S is the total number of bacteria, p is the number of variables to be optimized, which are present in each bacterium and $\theta = [\theta_1, \theta_2, \ldots, \theta_p$ is a point in the p dimensional search domain.

Reproduction: The least healthy bacteria eventually die while each of the healthier bacteria (those yielding lower value of the objective function) asexually split into two bacteria, which are then placed in the same location. This keeps the swarm size constant.

Elimination and Dispersal: Gradual or sudden changes in the local environment where a bacterium population lives may occur due to various reasons e.g. a significant local rise of temperature may kill a group of bacteria that are currently in a region with a high concentration of nutrient gradients. Events can take place in such a fashion that all the bacteria in a region are killed or a group is dispersed into a new location.

Size of population 'S': Increasing S can significantly increase the computational complexity of the algorithm. However, for larger values of S, it is more likely at least some bacteria near an optimum point should be started, and over time, it is then more likely that many bacterium will be in that region, due to either chemotaxis or reproduction.

Length of chemotactic step 'C(i)': If C(i) are too large, then if the optimum value lies in a valley with steep edges, the search will tend to jump out of the valley, or it may simply miss possible local minima by swimming through them without stopping. On the other hand, if C(i) are too small, convergence can be slow, but if the search finds a local minimum it will typically not deviate too far from it. c(i) is a sort of a "step size" for the algorithm.

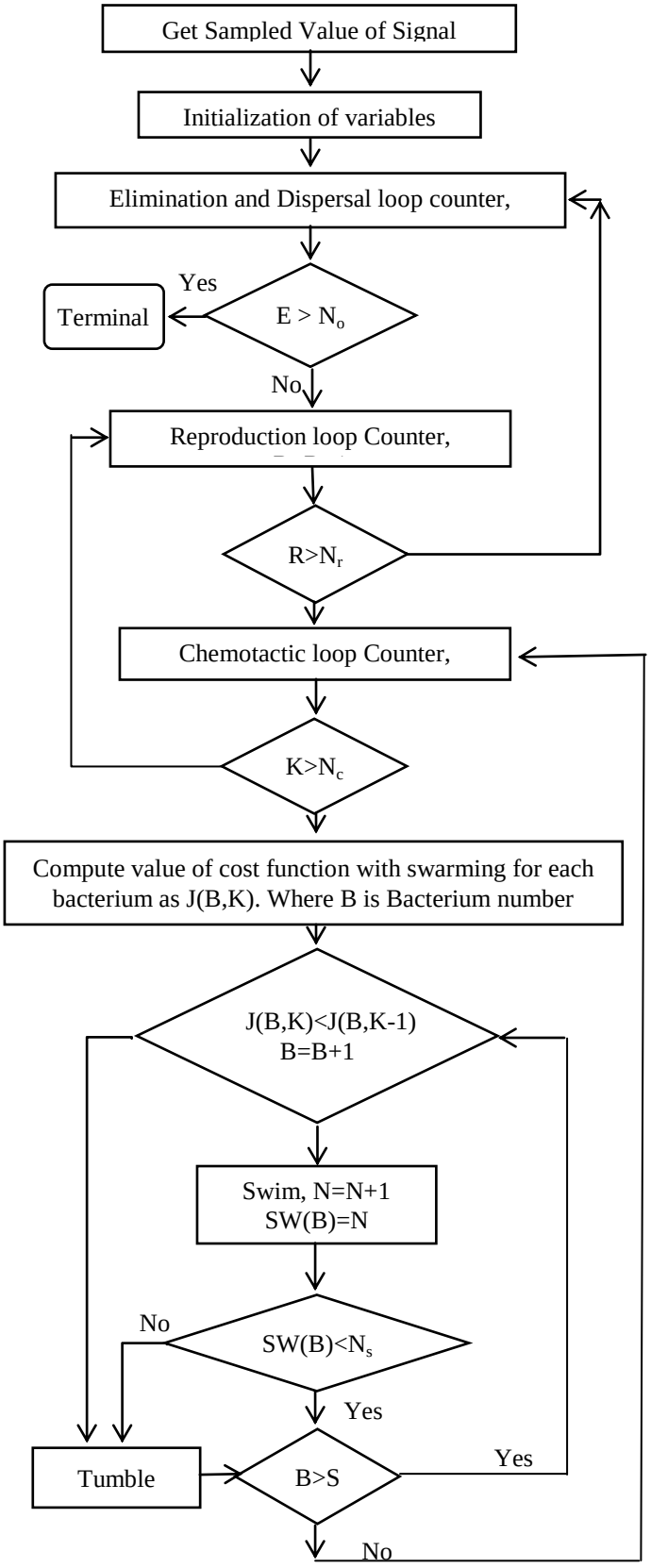


Figure 2: BFA flow chart

Chemotactic step 'Nc': If the size of Nc is chosen to be too short, the algorithm will generally rely more on luck and reproduction, and in some cases, it could more easily get

trapped in a local minimum (premature convergence). Ns creates a bias in the random walk (which would not occur if Ns = 0), with large values tending to bias the walk more in the direction of climbing down the hill.

Reproduction number 'Nre': If Nre is too small, the algorithm may converge prematurely; however, larger values of Nre clearly increase computational complexity.

Elimination and dispersal number 'Ned': A low value for Ned dictates that the algorithm will not rely on random elimination-dispersal events to try to find favorable regions. A high value increases computational complexity but allows the bacteria to look in more regions to find good nutrient concentrations. Clearly, if ped is large, the algorithm can degrade to random exhaustive search. If, however, it is chosen appropriately, it can help the algorithm jump out of local optima and into a global optimum.

Parameters defining cell-to-cell attractant functions 'Jcc': If the attractant width is high and very deep, the cells will have a strong tendency to swarm (they may even avoid going after nutrients and favor swarming). On the other hand, if the attractant width is small and the depth shallow, there will be little tendency to swarm and each cell will search on its own. Social versus independent foraging is then dictated by the balance between the strengths of the cell-to-cell attractant signals and nutrient concentrations. [12]

BPID CONTROLLER: the specification of the designed BFA technique is shown in Table1.

TABLE 1. SPECIFICATION OF THE BFA

The number of bacteria	10
Number of chemotactic steps	5
Limits the length of a swim	4
The number of reproduction steps	4
The number of elimination dispersal events	2

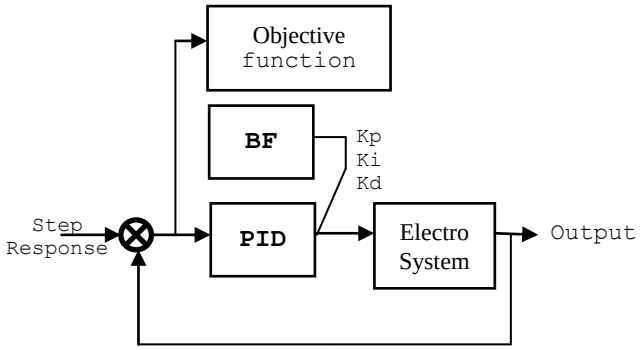
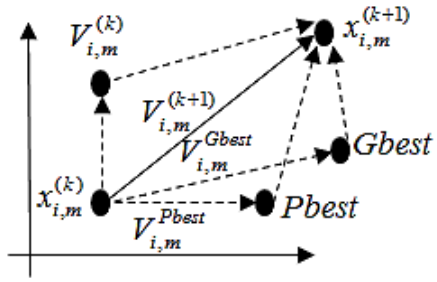


Figure 3: Block diagram of BPID to the Electro system

V. OVERVIEW PARTICLE SWARM OPTIMIZATION

PSO is a population based optimization method first proposed by Eberhart and Colleagues [13-15]. Some of the attractive features of PSO include the ease of implementation and the fact that no gradient information is required. It can be used to solve a wide array of different optimization problems. Like evolutionary algorithms, PSO technique conducts search using a population of particles, corresponding to individuals. Each particle represents a candidate solution to the problem at hand. In a PSO system, particles change their positions by flying around in a multidimensional search space until computational limitations are exceeded. Concept of modification of a searching point by PSO is shown in Figure 4.



$x_{i,m}^{(k)}$ Current position $x_{i,m}^{(k+1)}$ Modified position

$V_{i,m}^{(k)}$ Current velocity $V_{i,m}^{(k+1)}$ Modified velocity

$V_{i,m}^{Pbest}$ Velocity based on Pbesti

$V_{i,m}^{Gbest}$ Velocity based on Gbesti

Figure 4: Concept of modification of a searching point by PSO

Where:

N = Number of particles in the group, d = dimension,

t = Pointer of iterations (generations),

$V_{i,m}^{(t)}$=Velocity of particle I at iteration t,

w = Inertia weight factor,c1,c2 = Acceleration constant,

rand() Random number between 0 and 1.

$X_{i,m}^{(t)}$ = Current position of particle i at iterations,

Pbesti = Best previous position of the ith particle,

gbest = Best particle among all the particles in the population.

The PSO technique is an evolutionary computation technique, but it differs from other well-known evolutionary computation algorithms such as the genetic algorithms. Although a population is used for searching the search space, there are no operators inspired by the human DNA procedures applied on the population. Instead, in PSO, the population dynamics simulates a 'bird flock's' behaviour, where social sharing of information takes place and individuals can profit from the discoveries and previous experience of all the other companions during the search for food. Thus, each companion, called particle, in the population, which is called swarm, is assumed to 'fly' over the search space in order to find promising regions of the landscape. For example, in the minimization case, such regions possess lower function values than other, visited previously. In this context, each particle is treated as a point in a d-dimensional space, which adjusts its own 'flying' according to its flying experience as well as the flying experience of other particles (companions). In PSO, a particle is defined as a moving point in hyperspace. For each particle, at the current time step, a record is kept of the position, velocity, and the best position found in the search space so far.

The assumption is a basic concept of PSO [12]. In the PSO algorithm, instead of using evolutionary operators such as mutation and crossover, to manipulate algorithms, for a variable optimization problem, a flock of particles are put into the d-dimensional search space with randomly chosen

velocities and positions knowing their best values so far (Pbest) and the position in the d-dimensional space.

The velocity of each particle, adjusted according to its own flying experience and the other particle's flying experience. For example, the i-th particle is represented as xi = (xi,1 ,xi,2 ,…, xi,d) in the d-dimensional space. The best previous position of the i-th particle is recorded and represented as:

$$\text{Pbesti} = (\text{Pbesti},1, \text{Pbesti},2 ,..., \text{Pbest i},d) \qquad (9)$$

The index of best particle among all of the particles in the group is gbestd. The velocity for particle i is represented as vi = (vi,1 ,vi,2 ,…, vi,d). The modified velocity and position of each particle can be calculated using the current velocity and the distance from Pbesti, d to gbestd as shown in the following formulas [16-18]:

$$V_{i,m}{}^{(i+1)} = W.V_{i,m}{}^{(i)} + C_1 * \text{rand}(\ ) * \left(\text{Pbest}_{i,m} - X_{i,m}{}^{(i)}\right) + C_2 * \text{rand}(\ ) * \left(\text{Gbest}_{i,m} - X_{i,m}{}^{(i)}\right) (10)$$

$$X_{i,m}{}^{(i+1)} = X_{i,m}{}^{(i)} + V_{i,m}{}^{(i+1)} \quad i = 1,2,\dots n \quad m = 1,2,\dots d \ (11)$$

PPID Controller

The specification of the designed BFA technique is shown in Table2.

TABLE 2. SPECIFICATION OF THE PSO

Size of the swarm " no of birds "	50
Maximum number of "birds steps"	50
pso momentum or inertia	0.9

Figure 5 shows the block diagram for adjusting the PID parameters via PSO on line with the SIMULINK model. To begin with, the PSO should be provided with a population of PID sets.

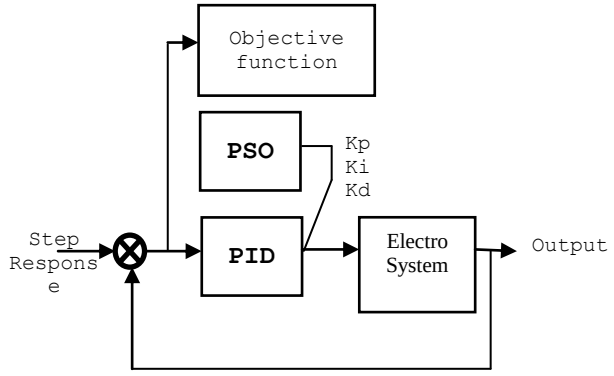


Figure 5: The block diagram of proposed PID Controller with PSO algorithms

VI. GENETIC ALGORITHM (GA)

Genetic programming [16-18] is an automated method for solving problems. Specifically, genetic programming progressively breeds a population of computer programs over a series of generations. Genetic programming is a probabilistic algorithm that searches the space of compositions of the available functions and terminals under the guidance of a fitness measure. Genetic programming starts with a primordial ooze of thousands of randomly created computer programs and uses the Darwinian principle of natural selection, recombination (crossover), mutation, gene duplication, gene deletion, and certain mechanisms of developmental biology to breed an improved population over a series of many generations.

Genetic programming breeds computer programs to solve problems by executing the following three steps:

1) Generate an initial population of compositions of the functions and terminals of the problem.

2) Iteratively perform the following substeps (referred to herein as a generation) on the population of programs until the termination criterion has been satisfied:

a) Execute each program in the population and assign a fitness value using the fitness measure.

b) Create a new population of programs by applying the following operations. The operations are applied to program selected from the population with a probability based on fitness (with reselection allowed).

Reproduction: Copy the selected program to the new population. The reproduction process can be subdi-vided into two subprocesses: Fitness Evaluation and Selection. The fitness function is what drives the evolutionary process and its purpose is to determine how well a string (individual) solves the problem, al-lowing for the assessment of the relative performance of each population member.

Crossover: Create a new offspring program for the new population by recombining randomly chosen parts of two selected programs. Reproduction may proceed in three steps as follows: 1) two newly re-produced strings are randomly selected from a Mating Pool; 2) a number of crossover positions along each string are uniformly selected at random and 3) two new strings are created and copied to the next generation by swapping string characters between the crossover positions defined before.

Mutation: Create one new offspring program for the new population by randomly mutating a randomly chosen part of the selected program.

Architecture-altering operations: Select an architecture-altering operation from the available repertoire of such operations and create one new offspring pro-gram for the new population by applying the selected architecture-altering operation to the selected program.

3) Designate the individual program that is identified by result designation (e.g., the best-so far individual) as the result of the run of genetic programming. This result may be a solution (or an approximate solution) to the problem. The specification of the designed GA technique is shown in Table3.

TABLE 3. SPECIFICATION OF THE GA.

Population Size	20
Crossover Rate	0.7
Mutation Rate	0.05
Chromosome Length	12
Precision of Variables	3
Generation Gap	1

Figure 6 shows the flowchart of the parameter optimizing procedure using GA. For details of genetic operators and each block in the flowchart, one may consult literature [19].

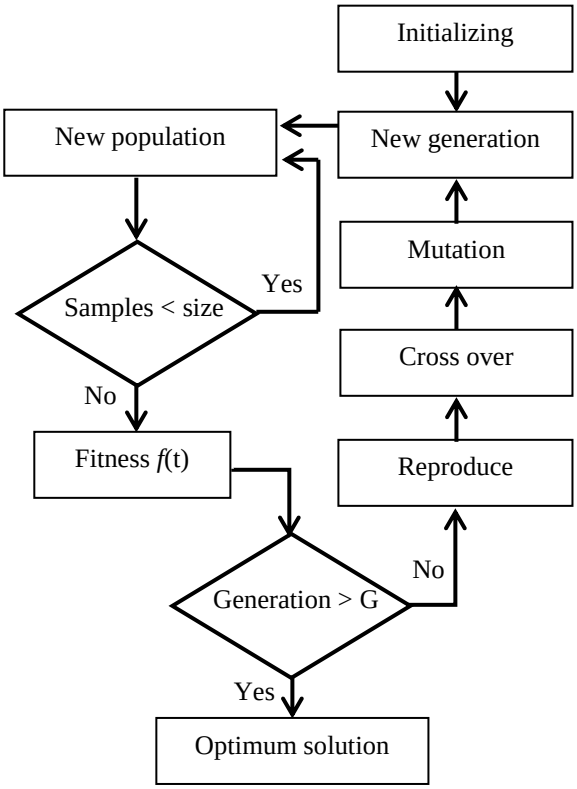


Figure 6. The optimization flowchart of GA technique.

The fitness measure is a mathematical implementation of the problem's high level requirements. That is, our fit-ness measure attempts to optimize for the integral of the time absolute error (ITAE) for a step input and also to optimize for maximum sensitivity.

The initial population for choosing PID parameters are selected as: KP = 1.2560, KI = 0.0062 and KD = 0.0275 by trial and error.

A fitness evaluation function is needed to calculate the overall responses for each of the sets of PID values and from the responses generates a fitness value for each set of individuals expressed by:

$$f(t) = \int_0^t t|e(t)|dt \qquad (12)$$

Here the goal is to find a set of PID parameters that will give a minimum fitness value over the period [0,t].

VII. ACO-PID IMPLEMENTATION

ACO is an evolutionary meta-heuristic algorithm based on the collective behavior emerging from the interaction of the different search threads that has proved effective in solving combinatorial optimization problems [20].

The Conventional fixed gain PID controller is well known technique for industrial control process. The design of this controller requires the three main parameters, Proportional gain (Kp), Integral time constant (Ki) and derivative time constant (Kd). The gains of the controller are tuned by trial and error method based on the experience and plant behavior. In proposed ACO-PID controller, ACO algorithm is used to optimize the gains and the values are applied into the controller of the plant as shown in Fig.7.

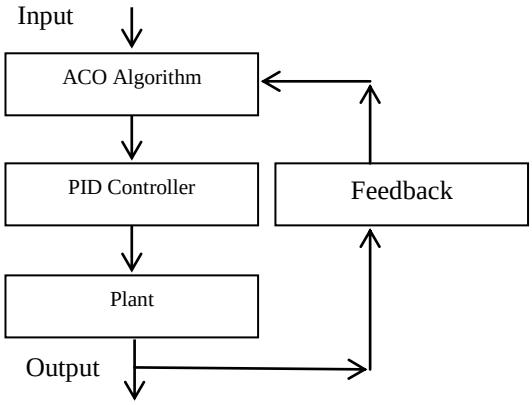


Figure 7. ACO-PID Controller

The objective of this algorithm is to optimize the gains of the PID controller for the given plant. The proportional gain makes the controller respond to the error while the integral derivative gain helps to eliminate steady state error and prevent overshoot respectively. The plant is replaced by the electrohydraulic model developed using simulink in MATLAB. With the optimum gains generated by the proposed ACO algorithm the models are simulated to validate the performance. The flowchart for ACO based PID controller is shown in Fig.8.

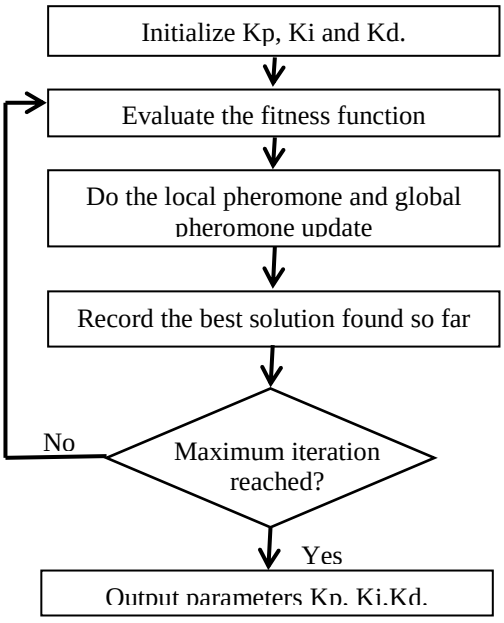


Figure 8. The optimization flowchart of ACO

VIII. SIMULATION OF THE SYSTEM

A. Simulation with BPID Controller

The closed loop control system was solved using numerical integration technique with varying step size that type ode5. The simulation method combines SIMULINK module and M functions where, the main program is realized in M function

and the optimized PID controller is predicted using SIMULINK. The model of the system is shown in Figure 6.
The results of tuning PID controller using BFA is shown in Table 4; Figure 10 shows the step responses obtained by using the optimized feedback. The optimal gains of PID controller are calculated to minimize the fitness function which was described in section III.

TABLE 4. BPID PARAMETERS AND RESULTS

	Kp	*Ki*	*Kd*	*Overshoot*	*Rising Time*
PI	0.8637	-0.0506	0	0.1	0.8
PD	0.8411	0	0.4085	0	0.65
PID	0.3846	-0.0725	-0.0890	0	2.2

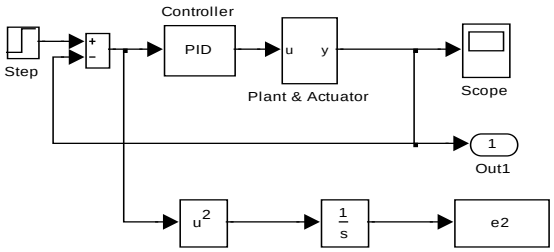


Figure 9: The Simulink model of BPID and PPID on the Electro system

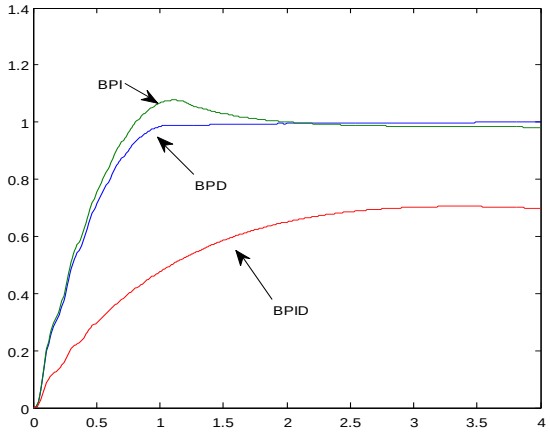


Figure 10: The step response after applying BPID on the Electro system

B. Simulation with PPID Controller

The closed loop control system was solved using numerical integration technique with varying step size that type ode5. The simulation method combines SIMULINK module and M functions where, the main program is realized in M function and the optimized PID controller is predicted using SIMULINK. The model of the system is shown in Figure 9.
The tuning PID controller using PSO is founded in [21]. The results of the model are shown in Table 5; Figure 11 shows the step responses obtained by using the optimized feedback. The optimal gains of PID controller are calculated to minimize the fitness function which was described in section V.

TABLE 5. PPID PARAMETERS AND RESULTS

	Kp	*Ki*	*Kd*	*Overshoot*	*Rising Time*
PI	0.7863	-0.1265	0	0	0.9
PD	0.7582	0	2.4700	0	0.7
PID	0.7352	-0.2040	4.2379	0	0.6

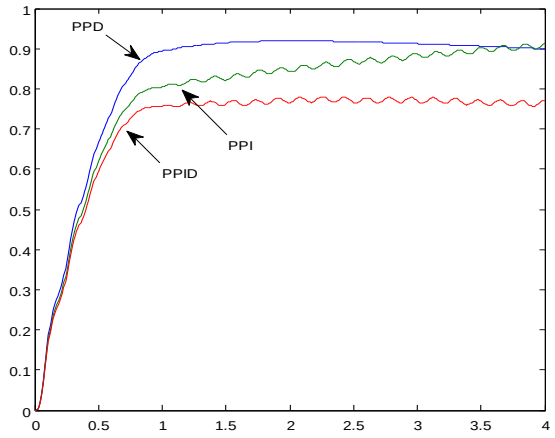


Figure 11: The step response after applying PPID on the Electro system

C. 3- Simulation with GPID Controller

The closed loop control system was solved using numerical integration technique of Runge-Kutta method with sampling time of 0.001 s. The simulation method combines SIMULINK module and M functions where, the main program is realized in SIMULINK and the optimized PID controller is predicted using M function.

TABLE 6. GPID PARAMETERS AND RESULTS

	Kp	*Ki*	*Kd*	*Overshoot*	*Rising Time*
PI	1.357	0.0623	0	0.05	0.7
PD	1.342	0	0.463	0	0.6
PID	1.138	0.053	0.537	0	0.4

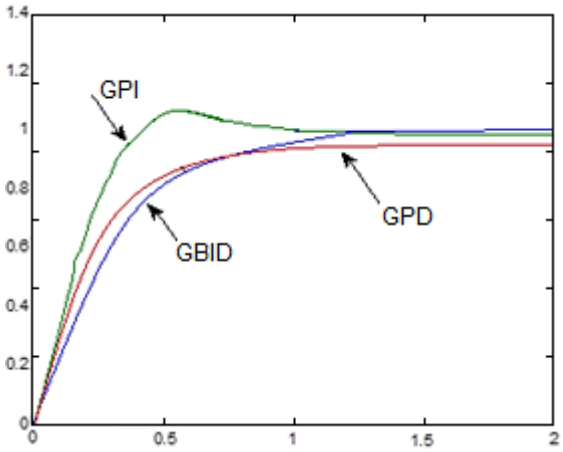


Figure 12: The step response after applying GPID on the Electro system

D. Simulation with ACO-PID Controller

The ACO algorithm was simulated and testes by tuning the various parameters like number of ants =500, number of nodes = 150, number of generations = 30, to get the optimum values as in table 7, and fig. 13

TABLE 7. ACO-PID PARAMETERS AND RESULTS

	Kp	*Ki*	*Kd*	*Overshoot*	*Rising Time*
PI	0.954	-0.054	0	0.1	0.9
PD	0.940	0	1. 618	0	0.8
PID	0.898	-0.063	-0.231	0	0.6

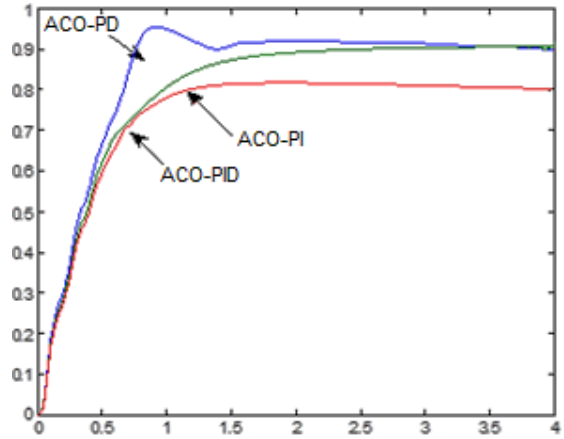


Figure 13: The step response after applying ACO-PID on the Electro system

IX. Discussions

We show that the response of the electrohydraulic servo system by applied several optimization techniques: BPID, PPID, GPID, and ACO-PID is more stable than the system without controller, and the controller make's faster in rising time and degrade the overshoot.

In BPID we show the BPD response is the best response comparing with BPI and BPID because the system was not stable and has the overshoot and delay in the rising time, so the parameters Kp and Kd makes decrease the rising time. In other hand, Kd decrease the overshoot so the response in PD and PID has no overshoot.

In PPID, the analysis of the results is the same in BPID but in the PPID there is no overshoot in PPI compare with BPI because the Kp in PPI is smaller than BPI where when we increase Kp it will increase the overshoot. But in other hand, the gain in PPID is less than 1 so the steady state error is high when the system trying to eliminate the overshot, so this is disadvantage for applying PPID and BPID.

In GPID, there is overshoot for high value of Kp. GA can solve this problem if it has more time and space, all experiments are applied on Core i2, 0.5G ram.

But in ACO-PID, we show overshoot in PI, that solved in PID, to give good relative rising time as 0.6s as in PPID.

X. Conclusions

This paper presents a design method for determining the PID controller parameters by using BFA, PSO,GA, and ACO that applied on the Electrohydraulic Servo Control System to make it in stable condition. The stability was found by minimizing the error in step response. To study the comparison between four optimization techniques for the Electrohydraulic Servo Control System, we show the results that the proposed BFA method can avoid the shortcoming of premature convergence of PSO method and can obtain higher quality solution with better computation efficiency.also GA can find higher quality solution with better response if use modern personal PC or workstation PC with high specifications. while the results of proposed PPID and ACO-PID methodsgave better results for electro-hydraulic servo control system in this works.

Therefore, the proposed methods has more robust stability and efficiency, and can solve the searching and tuning problems of PID controller parameters more easily and quickly than the traditional method.

References

[1] Ayman A. Aly, "PID Parameters Optimization Using Genetic Algorithm Technique for Electrohydraulic Servo Control System", Intelligent Control and Automation, 2011, 2, 69-7

[2] Andreas Antoniou and Wu-Sheng Lu, Practical Optimization Algorithms and Engineering Applications, Springer, October 2007.

[3] Matthew Robert Mackenzie, "PID Controller Optimisation Using Genetic Algorithms", Research Project towards the degree of Bachelor of Engineering (Computer Systems) ,October 2007.

[4] Boumediène ALLAOUA, Brahim GASBAOUI and Brahim MEBARKI "Setting Up PID DC Motor Speed Control Alteration Parameters Using Particle Swarm Optimization Strategy", Leonardo Electronic Journal of Practices and Technologies, Issue 14, January-June 2009 p. 19-32

[5] ZhongwenWang,JunpengShao,Jianying Lin and Guihua Han, "Research on Controller Design and Simulation of Electro-hydraulic Servo System", International Conference on Mechatronics and Automation. IEEE Proceedings of the 2009.

[6] M. H. Moradi, "New Techniques for PID Controller Design," IEEE Conference on Control Applications, Vol. 2, 2003, pp. 903-908.

[7] A. A. Aly, "A Non Linear Optimal PID Control of a Hydraulic Crane," Journal of Engineering Science, Vol. 33, No. 1, 2007, pp. 199-209.

[8] K. K. Tan, S. N. Huang and T. H. Lee, "Development of a GPC-based PID Controller for Unstable Systems with Dead Time," ISA Transactions, Vol. 39, 2000, pp. 57-70.

[9] P. Cominos and N. Munro, "PID Controllers: Recent Tuning Methods and Design to Specification," IEE Proceedings Control Theory & Applications, Vol. 149, No. I, January 2002.

[10] J. G. Ziegler and N. B. Nichols, "Optimum Settings for Automatic Controllers," Transactions of the ASME, Vol. 64, 1942, pp. 759-768.

[11] C. Coon, "Theoretical Consideration of Retarded Control," Transactions of the ASME, Vol. 75, 1952, pp. 827-834.

[12] Ahmed Bensenouci, "PID Controllers Design for a Power Plant Using Bacteria Foraging Algorithm", IEEE 2011.

[13] Kennedy J., Eberhart R., Particle swarm optimization, Proc. IEEE Int. Conf. on Neural Network, Vol. 4, p. 1942 – 1948, 1995.

[14] Shi Y., Eberhart R., A modified particle swarm optimizer, Proc. 1998 Int. Conf. on Evolutionary Computation, The IEEE World Congress on Computational Intelligence, Anchorage, May 1998, p. 69 – 73.

[15] Clerc M., Kennedy J., The particle swarm-explosion, stability, and convergence in a multidimensional complex space, IEEE Trans. Evolutionary Computation, 2002, Vol. 6, p. 58 –73.

[16] L. Hao, C. L. Ma and F. Li, "Study of Adaptive PID Controller Based on Single Neuron and Genetic Optimization," The 8th International Conference on Electronic Measurement and Instruments ICEMI, Vol. 1, 2007, pp. 240-243.

[17] J. R. Koza, M. A. Keane, M. J. Streeter, W. Mydlowec, J. Yu and G. Lanza, "Genetic Programming IV. Routine Human-Competitive Machine Intelligence," Kluwer Academic Publishers, Dordrecht, 2003.

[18] C. R. Reeves, “Using Genetic Algorithms with Small Populations,” Proceedings of the 5th International Conference on Genetic Algorithms, 1993.

[19] Housam Binous, "Dynamic and control of tank’s height using genetic algorithm toolbox and fminsearch", Updated 24 September 2007, this toolbox is available at: (http://www.mathworks.com/matlabcentral/fileexchange/16523-dynamic-and-control-of-tanks-height-using-genetic-algorithm-toolbox-and-fminsearch).

[20] Colorni.A, “Distributed optimization by ant colonies”,Proc. of ECAl’91, European Conference on Artificial Life,Elsevier Publishing, 1991

[21] wael korani, "Tunning of PID controller using Particle Swarm Optimization", Updated in 12 June 2008, this material is available at: (http://www.mathworks.com/matlabcentral/fileexchange/20252-tunning-of-pid-controller-using-particle-swarm-optimization)

[22] Kim T. H., Maruta I., Sugie T., Robust PID controller tuning based on the constrained particle swarm optimization, Automatica, Vol. 44, Issue 4, Apr. 2008, p. 1104 – 1110

[23] Gaing Z. L., A particle swarm optimization approach for optimum design of PID controller in AVR system, IEEE Trans. Energy Conversion, vol. 19, June 2004, p. 384 –391.

[24] Mukherjee V., Ghoshal S. P., Intelligent particle swarm optimized fuzzy PID controller for AVR system, Electric Power Systems Research, V. 77, Issue 12, Oct. 2007, p. 1689 –1698.

[25] Housam Binous, "Dynamic and control of tank’s height using genetic algorithm toolbox and fminsearch", Updated 24 September 2007, this toolbox is available at: (http://www.mathworks.com/matlabcentral/fileexchange/16523-dynamic-and-control-of-tanks-height-using-genetic-algorithm-toolbox-and-fminsearch).