Optimization of PID Controllers Using Ant Colony and Genetic Algorithms

Book · January 2013

DOI: 10.1007/978-3-642-32900-5

4 authors:

Muhammet Ünal
Marmara University

Ayca Ak
Marmara University

Vedat Topuz
Marmara University

Hasan Erdal
Marmara University

Trajectory Tracking Performance Comparison Between Genetic Algorithm and Ant Colony Optimization for PID Controller Tuning on Pressure Process

MUHAMMET ÜNAL,[1] HASAN ERDAL,[1] VEDAT TOPUZ[2]

[1]*Technical Education Faculty, Department of Electronics & Computer Education, Marmara University, Goztepe Campus, Kadikoy, Istanbul, Turkey*

[2]*Marmara University, Vocational School of Technical Sciences, Goztepe Campus, Kadikoy, Istanbul, Turkey*

Received 15 September 2009; accepted 18 January 2010

ABSTRACT: The main goal of this study was to compare the performances of genetic algorithm (GA) and ant colony optimization (ACO) algorithm for PID controller tuning on a pressure control process. GA and ACO were used for tuning of the PID controller when predefined trajectory reference signal was applied. Offline learning approach was employed in both GA and ACO algorithms. Realized pressure process dynamic has nonlinear behavior, thus system was modeled by nonlinear auto regressive and exogenous input (NARX) type artificial neural network (ANN) approach. PID controller was also tuned by Ziegler–Nichols (Z–N) method to compare the results. A cost function was design to minimize the error along the defined cubic trajectory for the GA-PID and ACO-PID controller. Then PID controller parameters (K_p, K_i, K_d) were found by GA-PID, ACO-PID algorithms, which were adjusted with their optimal parameters. It was concluded that both ACO and GA algorithms could be used to tune the PID controllers in the pressure process with excellent performance. This material is suitable for an engineering course on neural networks, genetic algorithm, ant colony optimization and process control laboratory. © 2010 Wiley Periodicals, Inc. Comput Appl Eng Educ 20: 518–528, 2012; View this article online at wileyonlinelibrary.com/journal/cae; DOI 10.1002/cae.20420

Keywords: pressure process; ant colony optimization algorithm; genetic algorithm; PID controller; artificial neural network

INTRODUCTION

Pressured tank systems are commonly used in many industrial systems and applications such as HVAC systems' air supply units, water supplying, watering systems, water treatment systems, fire systems, industrial washing systems [1–4]. PID controllers are also widely used for those systems [5–7]. The PID controllers are the most common control algorithms because of their simplicity and robustness. Today, about 90–95% of all industrial control problems are solved by using this type of controllers in various forms [7]. Although conventional PID is popular, it is difficult to obtain satisfying controller parameters especially when the process has nonlinear dynamics [4,8]. Actually, much of the effort of researchers has been concentrated on the development of new tuning rules for PID controller. Although this is obviously a crucial issue, it is well known that a key role in the achievement of high performance in practical conditions is also played by those functionalities that have to (or can) be added to the basic PID control law.

There are different techniques that can be used in tuning PID controller parameters, which can be classified into the following:

Correspondence to M. Ünal (munal@marmara.edu.tr).

- analytical methods; where PID parameters are calculated from analytical or algebraic relations between a plant model and an objective [9,10];
- frequency response methods; where frequency characteristics of the controlled process are used to tune the PID controller [11];
- optimization methods; where these can be regarded as special types of optimal control, in which PID parameters are obtained ad hoc using offline numerical optimization methods [12];
- adaptive tuning methods; which are for automated online tuning [11];
- heuristic methods; which are evolved from practical experience in manual tuning and from artificial intelligence [5,7,13,14].

There are several reasons to develop better methods to design PID controllers. One is the significant impact it may give because of the widespread use of controllers. The other is the enhancing performance of controllers, which can be derived from improved design methods. It is possible to obtain controller parameters with analytical methods such as root-locus, bode analysis and pole replacement when system has linear behavior or it is possible to linearize the system [9]. Genetic algorithm (GA), ant colony optimization (ACO), artificial neural networks (ANNs), adaptive neuro-fuzzy inference system (ANFIS), swarm optimization (SO) and taboo search (TS) techniques could be used to model or control the system whether the process models have linear or nonlinear dynamics [5,15]. ACO and GA are general-purpose optimization techniques that have been developed and recognized as effective for combinatorial optimization problems such as the traveling salesman problem, quadratic assignment problem and controller tuning problems with successful results [7,13,16–24].

Realized pressure process system has nonlinear dynamics because of compressibility of air and nonlinear characteristic of valves. Therefore, it is difficult to obtain the mathematical model of the realized system. These difficulties forced us to use NARX type ANN for modeling approach [5]. Then GA and ACO were used for tuning of the PID controller when predefined trajectory reference signal was applied. We demonstrated the effectiveness of proposed approach by carrying out a series of experimental studies and the results were compared with Ziegler–Nichols (Z–N) tuning method. The demonstrated techniques could be used to teach any nonlinear system modeling and control approach in the real-time application.

This study is organized as follows: second section introduces the designed system and its ANN model. Proposed GA-PID and ACO-PID approach are explained in the third section. Fourth section discusses experimental results as illustrations. Conclusions are finally drawn in the last section.

SYSTEM DESCRIPTION AND ANN MODEL

The main aim of realized system was to experimentally teach the nonlinear modeling and control of process control system in our university. To achieve this goal, the pressure processes control system which is given in Figure 1 was designed. The purpose of our implemented system was to stabilize the pressure of the tank at the desired pressure level adjusting the input air flow despite the varying exhaust output. The tank input and output flows are controlled by pneumatic and manual valve respectively. The control signal, which is output of the DAC (4–20 mA) is converted to pneumatic signal via I/P converter to feed the pneumatic valve (Samson 241-7). This pneumatic valve is supplied with 6 bar air pressure continuously. Tank pressure is measured by P/I (Endress-Hauser) converter and it is used as a feedback signal to the ADC. The MATLAB based PID digital controller is used to control the pressure in the tank. The computer with Advantech PCI 1711 model data acquisition board (DAQ) [25] is used the hardware platform of the digital controller.

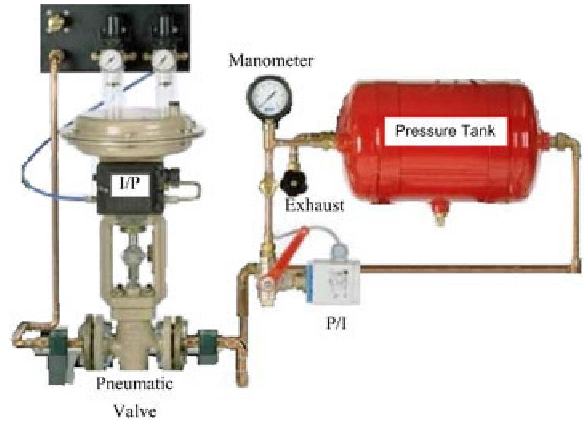


Figure 1 Realized pressure process system. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Because of compressibility of air and nonlinear characteristic of valves, realized system has nonlinear dynamics. To overcome this problem, some intelligent modeling and control techniques were used. In addition to these, third-order trajectory function was used as an input reference signal, to prevent the pressure fluctuations and large overshoot in tank, which it could be harmful in some process [26]. This trajectory is defined as follows;

$$
\begin{array}{ll}
p_Y = p_0 & t = t_0 \\
p_Y = \varphi_3 t^3 + \varphi_2 t^1 + \varphi_1 t + \varphi_0 t_0 < t < t_S \\
p_Y = p_5 & t \geq t_S
\end{array}
\tag{1}
$$

where p_0, p_s are initial and desired pressure values, φ_{0-3} are trajectory coefficients, t_0 is initial time, t_S is setting time defined as $3\alpha = [(p_Y - p_0)/(p_Y + p_0)]$ and α is trajectory slope.

The software architecture of designed system briefly explained as below:

MATLAB with Simulink environment and GA, ANN toolbox are used to perform modeling, optimization and real-time control phases of designed system [27]. MATLAB alone provides a platform for numeric calculation, analysis, and visualization. Simulink is an interactive environment for modeling and PC-based simulation with easy-to-use block diagrams. ANN toolbox is used to model the designed system with NARX type ANN architecture. Realized network was trained with serial-parallel architecture then converted the parallel architecture in the simulation phase [28]. GA toolbox is used to find optimal PID controller parameter values [29]. The cost function (fitness function) is defined to minimize the error along the trajectory within the toolbox, which is given in Equation (8). ACO algorithm is developed by the first author as a MATLAB procedures and outline of this procedure is given in PID tuning with GA and ACO section. Then optimal PID controller parameters are also found by ACO algorithm. Real-time PID controllers are designed as a Simulink block diagram. Real-time workshop integrated with Simulink generates C-code from this block diagram [30]. These C codes are basis for further processing with the DAQ. Main GUI of designed program is given in Figure 2.

Structure of NARX Type ANN

ANNs have been applied to a large number of problems because of their nonlinear system modeling capacity. Given a sample vector, ANNs are able to map the relationship between input and output; they "learn" this relationship, and store it into their parameters. As these two characteristics suggest, they should prove to be particularly useful when there is a little prior knowledge about the system. Most of the ANN applications use simple multi-layer perceptron (MLP) network training with back-propagation algorithm. A simple way to introduce dynamics into MLP network consists of using an input vector composed of past values of the system inputs and outputs. This the way by which the MLP can be interpreted as a NARX model of the system. This way of introducing dynamics into a static network has the advantage of being simple to implement [31]. To deduce the dynamic model of realized system, NARX type ANN model can be represented as follows:

$$\hat{y}(k) = f_{\mathrm{ANN}}[y(k-1), \ldots, y(k-n), u(k-1), \ldots, u, (k-m)] + \varepsilon(k) \tag{2}$$

where $\hat{y}(k)$ is model predicted output, f_{ANN} is a nonlinear function describing the system behavior, $u(k)$, $y(k)$, $\varepsilon(k)$ are input, output and approximation error vectors at the time instances k, n and m the orders of $y(k)$ and $u(k)$ respectively. Order of the process can be estimated from experience modeling by ANN relies on the consideration of an approximate function of f_{ANN}. Approximate dynamic model is constructed by adjusting a set of connection weight (**W**) and biases (**b**) via training function defined as follows [32,33]:

For train to MLP with back-propagation, the first step is propagating the inputs towards the forward layers through the network. For a three-layer feed-forward network, training process is initiated from the input layer [33]:

$$\begin{aligned} a^0 &= u \\ a^{m+1} &= f^{m+1}(W^{m+1}a^m + b^{m+1}), \quad m = 0, 1 \\ \hat{y} &= a^3 \end{aligned} \tag{3}$$

where $\hat{y}$ output vector, u is input vector, $f(.)$ is the activation function, W is weighting coefficients matrices, b is bias factor vector and m is the layer index. These matrices defined as:

$$W^1 = \begin{bmatrix} w_{1,1} & w_{1,2} & \ldots & w_{1,S0} \\ w_{2,1} & w_{2,2} & \ldots & \cdot \\ \cdot & \cdot & \ldots & w_{2,S0} \\ w_{S1,1} & w_{S1,2} & \ldots & w_{S1,S0} \end{bmatrix}$$

$$W^2 = [w_{1,1} w_{1,2} \ldots w_{1,S1}], \qquad b^1[b_1 b_2 \ldots b_{S1}]^{\mathrm{T}}$$

$$b^2 = [b_1], \qquad f^1 = \frac{(\exp^n - \exp^{-n})}{(\exp^n + \exp^{-n})}$$

$$f^2 = n.$$

where n is the total network output ; $n_i^m = \sum_{j=1}^{s^{m-1}} w_{i,j}^m a_j^{m-1} + b_i^m$; s_0 and s_1 are the size of network input and hidden layer.

Second step is propagating the sensibilities (d) from the last layer to the first layer through the network: d^3, d^2, d^1. The error (e) calculated for output neurons is propagated to the backward

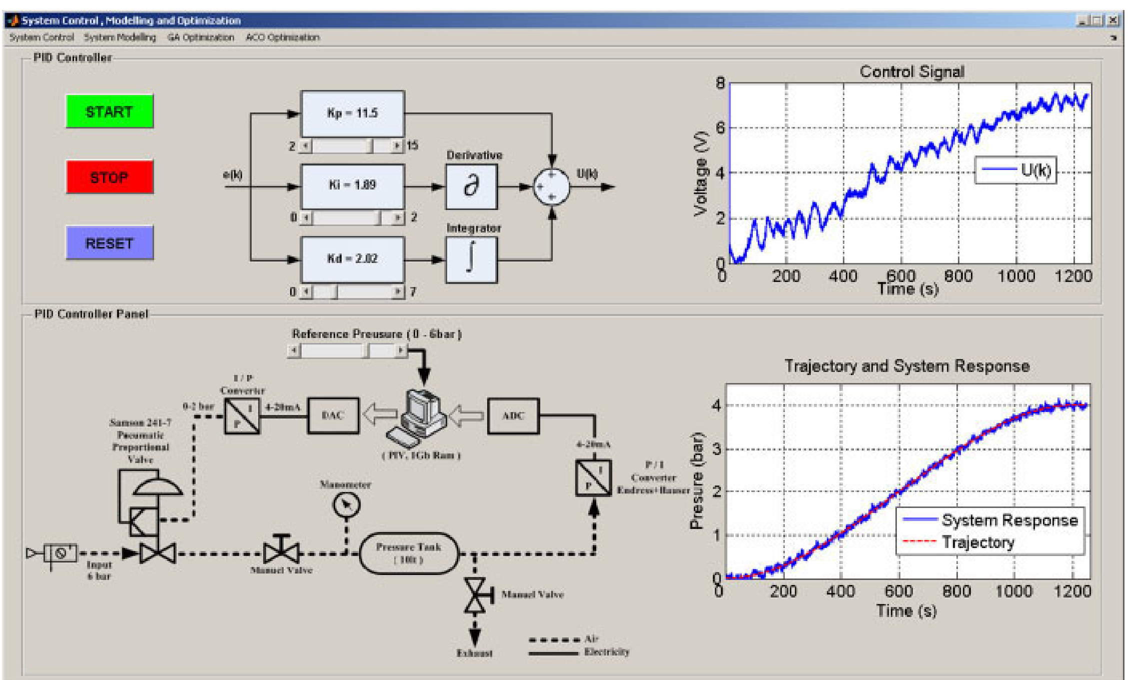


Figure 2 Main GUI of designed program. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

through the weighting factors of the network. It can be expressed in matrix form as follows:

$$\begin{aligned} d^3 &= -2\dot{F}3(n^3)(e) \\ d^m &= \dot{F}m(n^m)(W^{m+1})^T d^{m+1}, \quad \text{for } m = 2, 1 \end{aligned} \tag{4}$$

$\dot{F}^{\mathrm{m}}(n^m)$ is the Jacobian matrix;

$$\dot{F}m(n^m) = \begin{bmatrix} ((\partial f^m(n_1^m))/(\partial n_1^m)) & 0 & 0 \\ 0 & ((\partial f^m(n_2^m))/(\partial n_2^m)) & 0 \\ 0 & 0 & ((\partial f^m(n_{s^3}^m))/(\partial n_{s^3}^m)) \end{bmatrix}$$

e is mean square error,

$$e = \frac{1}{2}\sum_{\gamma=1}^{q}(y^{\gamma} - \hat{y}\gamma)^2 \tag{5}$$

where γ is the sample in dimension q.

The last step in back-propagation is updating the weighting coefficients. The state of the network always changes in such a way that the output follows the error curve of the network towards down.

$$\begin{aligned} W^m(k+1) &= W^m(k) - \alpha d^m (a^{m-1})^T \\ b^m(k+1) &= b^m(k) - \alpha d^m \end{aligned} \tag{6}$$

where α is the learning rate and k is the epoch number. By the algorithmic approach known as gradient descent algorithm using approximate steepest descent rule, the error is decreased repeatedly.

The NARX model of the plant plays a very crucial role in the proposed structure and designed NARX type ANN architecture was given in Figure 3.

The performance of ANN implementations depends on number of hidden layer and neurons in the hidden layer(s). Determining the optimal values of these numbers is still a question to deal with and the choice of these numbers related to the application. Although there is no theoretical basis for selecting these parameters, a few systematic approaches are also reported. But the most common way is still trial and error approach which we also used this method.

The performance of ANN with 5, 10, 15, 20, 30 and 50 neurons in the one hidden layer was evaluated according the mean square error (MSE) and correlation coefficient (R). The results were given at Table 1. Higher values of correlation coefficients and lower MSE values show that model and its predicted values are closed together. According the Table 1, ANN-30 network has satisfactory performance both learning and generalization phase.

Therefore, we choose NARX type ANN-30 network for modeling of dynamical behavior of realized pressure process system. To show the ANN performance clearly, system open-loop outputs and ANN outputs at the different pressure values were given in Figure 4. As we could infer from this figure, ANN model and real system behavior are consistent and quite similar.

PID TUNING WITH GA and ACO

Since its invention, the PID control scheme has found wide applications in industry. Owing to its simplicity and unexpectedly good performance provided by these three actions defined as a discrete positional form as follows:

$$u(k) = K_p e(k) + K_i \sum_{r=0}^{s} e(r) + K_d[e(k) - e(k-1)] \tag{7}$$

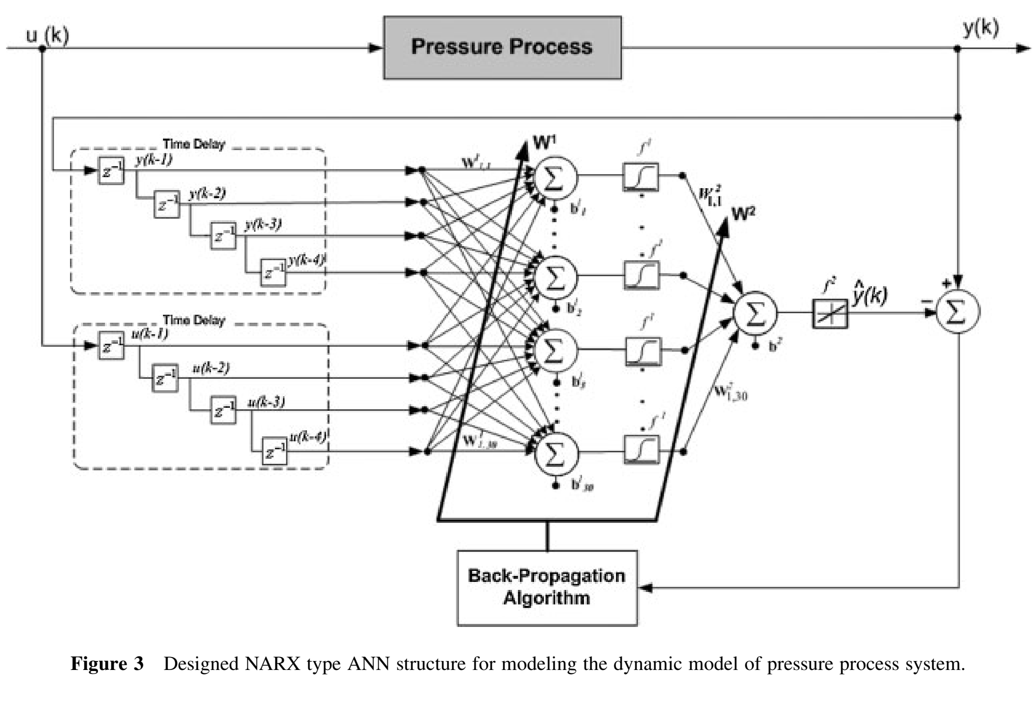


Figure 3 Designed NARX type ANN structure for modeling the dynamic model of pressure process system.

Table 1 MSE and Correlation Coefficent (R) Values at Different ANN Models

Model	Hidden layer neurons number	MSE generalization	Correlation coefficient (R)	
			Learning	Generalization
ANN-5	5	9.90E−06	0.975	0.778
ANN-10	10	8.20E−06	0.983	0.855
ANN-20	20	9.93E−07	0.987	0.892
ANN-30	30	3.70E−07	0.999	0.997
ANN-50	50	3.60E−07	0.999	0.998

where $e(k)$ is the error signal, K_p is the proportional gain, K_i is the integral gain, and K_d is the derivative gain of controller.

The tuning of PID controllers for industrial plants could frequently be difficult and time-consuming, even if expert systems are used to automate the tuning process. Indeed, the use of such expert systems for the tuning of PID controllers often creates more problems than it solves because of the difficulties inherent in the process of knowledge elicitation[7]. The easy-to-use Z−N tuning method fertilized the popularity of PID control in almost all sectors of industry. The widely accepted Z−N method does not, however, yield optimal closed-loop performance and its application range is rather limited. Therefore, the techniques of (GA-PID) and (ACO-PID) were proposed as an alternative means of tuning digital PID controllers in addition these algorithms provide a much simpler and robust approach than the classical methods.

Genetic algorithms are one of the evolutionary soft computing techniques [34]. A GA is a search algorithm inspired by Darwin's theory of biological evolution and was pioneered by Holland. GAs provide solutions using randomly generated bit strings (chromosomes) for different types of problems. GAs are the process of searching the most suitable one of the chromosomes that built the population in the potential solutions space. GAs start parallel searching from independent points of searching space in which the solution knowledge is poor or not available. The solution depends on interaction of the surroundings and genetic operators. For that reason, obtaining the suboptimal solutions of GAs are a small probability. It is particularly presented as an alternative for the traditional optimal search approaching in which it is hard to find the global optimum point in nonlinear and multimodal optimization problems.

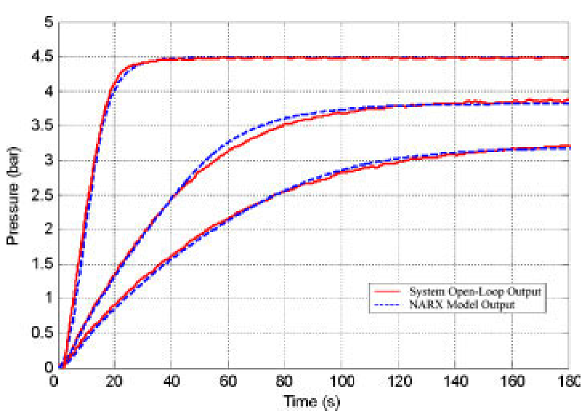


Figure 4 ANN model and system open-loop output at the different reference pressure. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Thus, GAs have been successful in solving many combinatorial problems [35] and recognized as a powerful tool in many control applications such as parameter identification and control structure design [36,37]. GAs have also found widespread use in controller optimization particularly in field of fuzzy logic and neural networks [38,39]. In the early 1990s, GAs were first investigated as an alternative means of tuning PID controllers. Then GAs have also been extensively applied to the off-line design of PID controllers, particularly as an alternative tuning technique for process which are otherwise difficult to tune. Oliveira et al. used a GA to determine initial estimates for the values of PID parameters [40]. They applied their methodology to a variety of classes of linear time-invariant (LTI) system, encompassing minimum-phase, nonminimum phase, and unstable systems. They improved the efficiency of their algorithm by identifying ancestral (already-assessed) chromosomes and avoiding re-evaluation of these. We implement GA for tuning PID controller in offline approach, which the structure was given in Figure 5.

The most important feature of GAs is how to transform the system output to the cost functions (or fitness value). Therefore, we defined the cost function to minimize RMS error through the all trajectory period. Defined trajectory has two different regions (transient and steady-state); therefore, cost function has also different weight factors ($\delta = 0.4$, $\beta = 0.6$) for each region. Designed cost function given as follows:

$$\phi_{\min} = \frac{1}{T}\left[\sum_{t_0}^{t_s} \delta\sqrt{(p_{y(k)} - y_{(k)})^2} + \sum_{t_s}^{t_{ss}} \beta\sqrt{(p_{y(k)} - y_{(k)})^2}\right] \quad (8)$$

where ϕ is the cost value, T is the number of sampled data through the period, $y_{(k)}$, $p_{y(k)}$ are system and desired trajectory output, t_s is the setting time and t_{ss} is the steady-state time, δ and β are cost gain factors.

GAs performance highly depends on its parameters values. Therefore realized GA optimal parameters values were found by making numerous experiments, which were given in Table 2.

Evolution of GA cost function on a typical run, which was set by optimal GA parameters was given in Figure 6. In addition, end of the 100 generation obtained PID controller parameters were given in Table 3.

ACO-PID Controller

ACO is an evolutionary meta-heuristic algorithm based on a graph representation that has been applied successfully to solve various hard combinatorial optimization problems. The main idea of ACO is to model the problem as the search for a minimum cost path in a graph. Artificial ants walk through this graph, looking for good paths. Each ant has a rather simple behavior so that it will typically only find rather poor-quality paths on its own. Better paths are found as the emergent result of the global cooperation among ants in the colony [13,14,18,19,41] . Many researchers have been working on PID tuning methods with ACO up to now and they used different kind of approach to improve the performance of ACO. Varol et al. [13] used the ACO to tune PID controller parameter for a second order system with different cost function. They found very satisfactory results over the classical tuning method. Ȟin et al. [41] also used improved ACO for

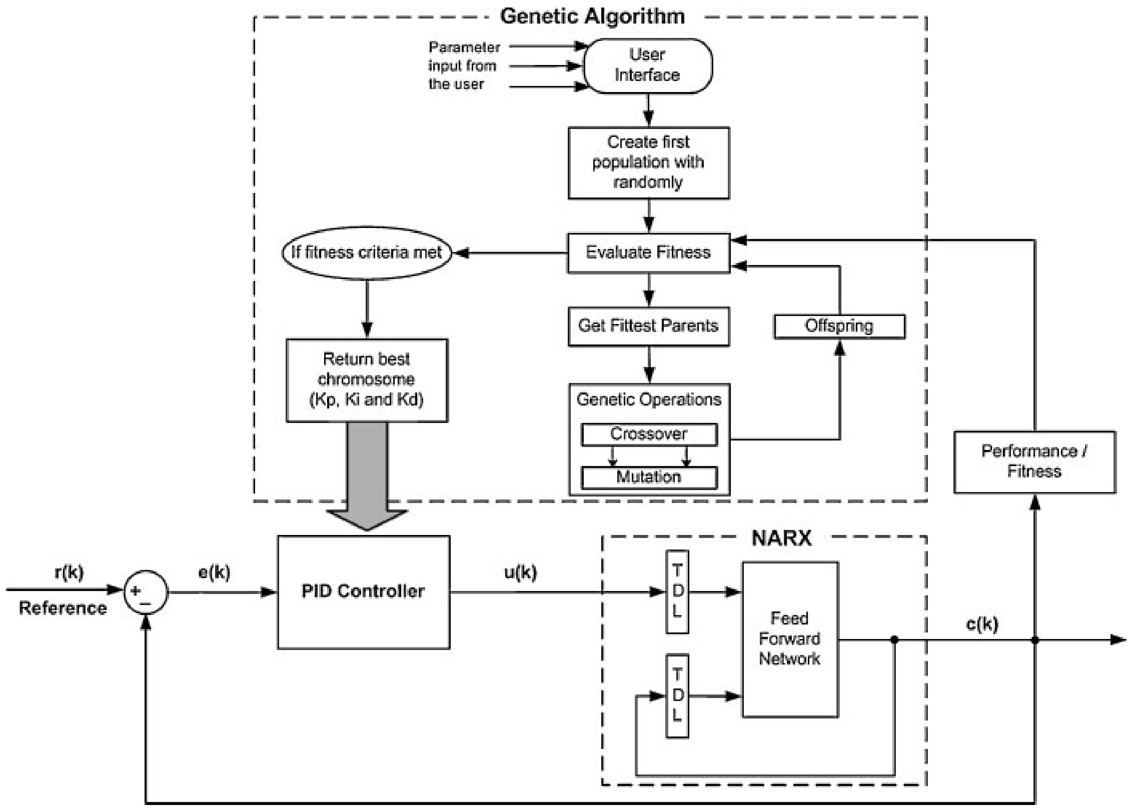


Figure 5 Implemented GA-PID controller structure.

nonlinear PID controller and they concluded that their improved approach very effective.

To solve PID controller design problem with ACO, this problem could be represented as a graph problem, which is given in Figure 7. All of the values for each parameter (K_p, K_i, K_d) are placed in three different vectors. To create a graph representation of the problem, these vectors can be considered as paths between the nests. In the tour, the ant must visit three nests by choosing path between start and end node. The objective of ACO was to find the best tour with the lowest cost function (given in Eq. 4) among the three nests. The ants deposit pheromone to the beginning of each path. Then the pheromones were updated in pheromone updating rule.

In proposed approach, each ant updates the pheromones deposited to the paths it followed after completing one tour defined as local pheromone updating rules as follows:

$$\zeta(k)_{ij} = \zeta(k-1)_{ij} + \frac{0.01\,\theta}{J} \tag{9}$$

Table 2 Realized GA Optimal Operator Values

GA parameters	Value
Coding type	Real code
Population size	20
Selection operator	Tournament (4 individuals)
Crossover operator	Arithmetic
Crossover probability	80%
Mutation operator	Gaussian ($c = 1$, $\sigma = 1$)
Elite percentage	20%
Scaling	Rank based

where $\xi(k)_{ij}$ is the pheromone value between nest (i) and (j) at the k. iteration, θ is the general pheromone updating coefficient, J is the cost function for the tour traveled by the ant.

In global pheromone updating rule, pheromones of the paths belonging to the best tour (10) and worst (11) tour of the ant colony are updated as given in the follows:

$$\zeta(k)_{ij}^{\text{best}} = \zeta(k)_{ij}^{\text{best}} + \frac{\theta}{J_{\text{best}}} \tag{10}$$

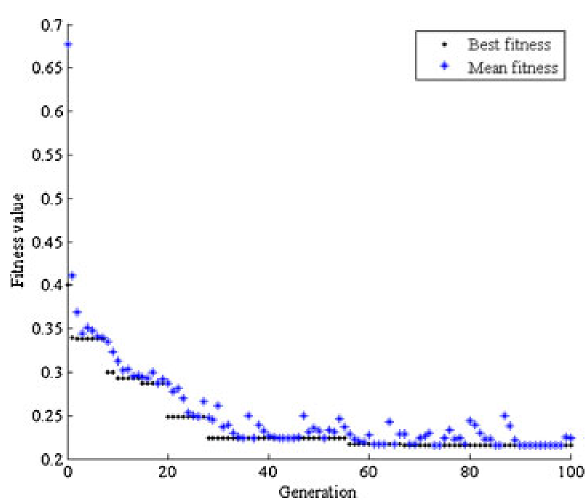


Figure 6 The evolution of GA cost function. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Table 3 Obtained PID Controller Parameters With GA, ACO and ZN Methods

Controller	K_p	K_i	K_d
GA-PID	11.34	1.89	5.43
ACO-PID	12.88	1.44	3.92
ZN-PID	8.16	0.13	2.53

$$\zeta(k)_{ij}^{\text{worst}} = \zeta(k)_{ij}^{\text{worst}} - \frac{0.3\theta}{J_{\text{worst}}} \tag{11}$$

where ζ^{best} and ζ^{worst} are the pheromones of the paths followed by the ant in the tour with the lowest cost value (J_{best}) and with the highest cost value (J_{worst}) in one iteration, respectively.

The pheromones of the paths belonging to the best tour of the colony are increased considerably, whereas those of the paths belonging to the worst tour of the iteration are decreased. After then pheromone evaporation (12) allows the ant algorithm to forget its past history, so that ACO can direct its search towards new directions without being trapped in some local minima.

$$\zeta(k)_{ij} = \zeta(k)_{ij}^{\lambda} + [\zeta(k)_{ij}^{\text{best}} + \zeta(k)_{ij}^{\text{worst}}] \tag{12}$$

where λ is the evaporation constant [13,21]

In this study, PID controller parameters were coded by 5000 nodes. In other words, one node represents a solution value of the parameters K_P, K_i and $K_{d.}$. Thus, the bigger the number of used nodes, the more accuracy trails are updated. Optimal ACO parameters ($\theta = 0.06$, $\lambda = 0.95$) were obtained end of numerous experiments. Evolution of ACO cost function on a typical run, which was set by optimal parameters was given in Figure 8. End of the 100 tour, obtained PID controller parameters were also given in Table 3.

Tuning PID Controller With Z−N Methods

The Z−N design methods are the most popular heuristic methods used in process control for determining the parameters of a PID controller. Although these methods date back to early 40 s for analog controllers, they are still appropriate for modern digital control systems. They proposed rules for determining values of the proportional gain, integral time and derivative time based on the transient response characteristics of a given plant. Such determination of the parameters of the PID controller or tuning controller can be made by engineers on site by experiments on the plant. The first method, which is known as the continuous cycling method, the controller gain is increased until a sustained oscillation takes place. In the second method, commonly known as the process reaction curve method, the open-loop unit step response of the plant is measured. Then the parameters of the PID controller can be calculated using table given by Ziegler and Nichols. In both methods, they aimed at obtaining 25% maximum overshoot in step response [42].

Because the realized system open-loop output matches the output defined by reaction curve method; process reaction curve method was used. After then required parameters were measured (delay time $L = 5$ s, time constant $\tau = 34$ s) and PID controller parameters were calculated using table given by Ziegler and Nichols which are given in Table 3.

EXPERIMENTS AND RESULTS

In this section, we compared the performance of GA-PID, ACO-PID and ZN-PID controller in the real time. Realized modeling, optimization and control applications have been developed using MATLAB software package. Initially trajectory slope effects then reference pressure effect were observed. Additionally disturbance rejection capabilities of each controller were examined. The performance evaluations were performed RMS error between trajectory and system output as follows:

$$E = \frac{1}{T}\sum_{t_0}^{t_{ss}} \sqrt{(p_y(k) - y(k))^2} \tag{13}$$

First, reference pressure was set to ($p_s = 2$ bar) and trajectory slope ($\alpha = 6$, 10, 15, 20) respectively. Then controllers' outputs were given in Figure 9a−c. To examine the each controller performance, obviously RMS errors were also given in Table 4. We can draw a conclusion both Table 4 and Figure 8 that, GA-PID and ACO-PID have similar and good performance (means lower RMS error) at the all slope values. On the other hand, ZN-PID controller can not have acceptably performance (nearly 10 times more RMS error then GA-PID and ACO-PID controller) especially in low slopes ($\alpha = 6$, 10).

Then, pressure reference was set to ($p_s = 4$ bar) and experiments were repeated. GA-PID, ACO-PID, ZN-PID controller

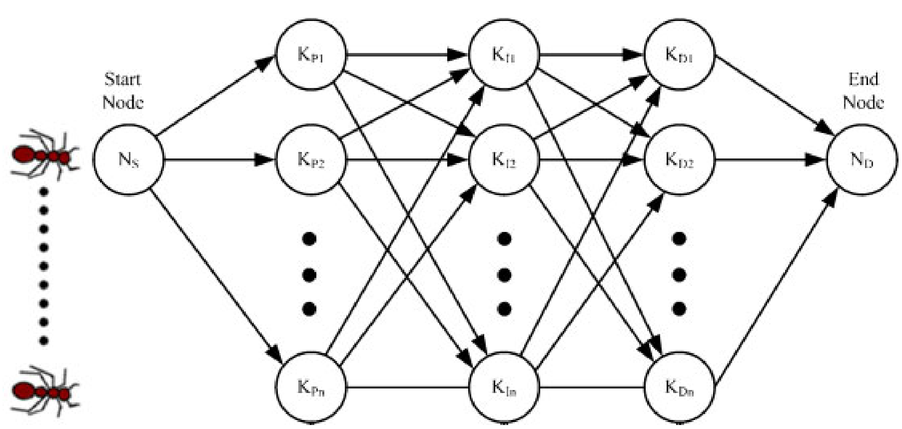


Figure 7 Graphical representation of ACO for PID tuning process.

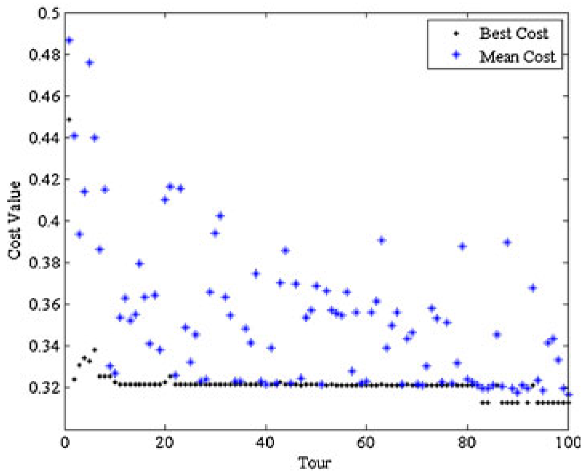


Figure 8 The evolution of ACO cost function. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

outputs were given in Figure 10a–c. Also RMS errors were listed in Table 3. As a result, we could make same conclusion as explained above. That means, GA-PID and ACO-PID controller were shown equal performances. Their performances were definitely superior to ZN-PID controller at the all trajectory values.

To test the disturbance rejection capability of designed PID controllers, the exhaust valve is opened 40% (t_1) time then closed to its initial position; (t_2) time then obtained outputs were given Figure 11. All designed controller were traced the defined trajectory within limited time. This period was taken 50 s on the GA-PID controller, 85 s on the ACO-PID controller and 100 s on the ZN-PID controller.

CONCLUSION

In this study, first dynamic model of pressure process system were obtained by NARX type ANN. Then validity of acquired model were proved by regression analysis ($R = 0.997$) and mean square

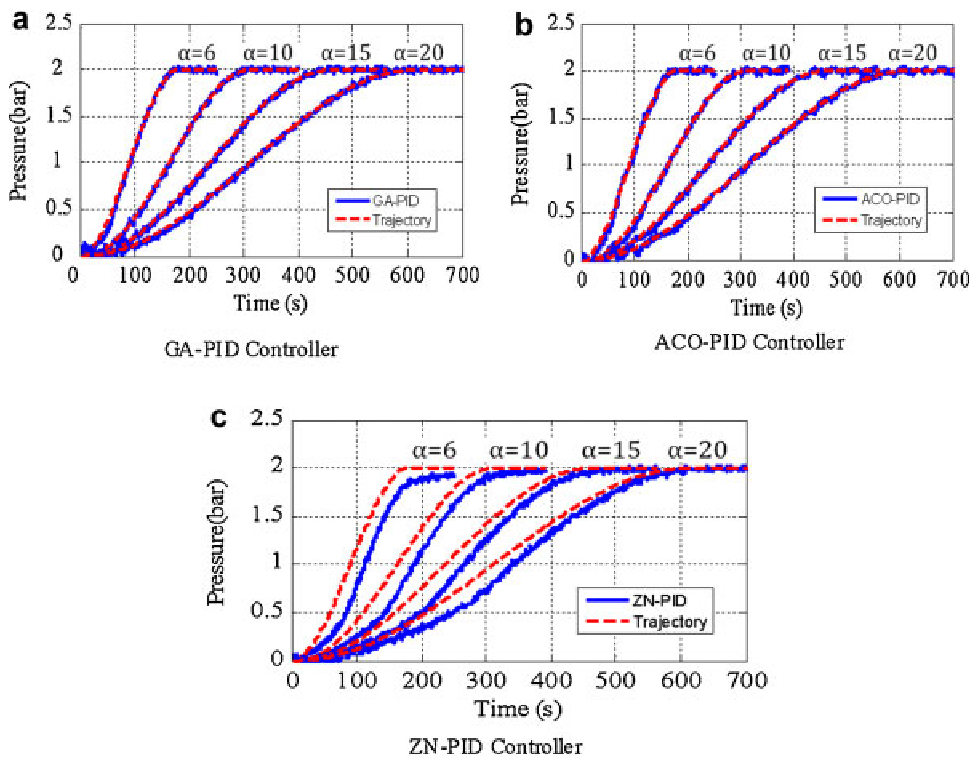


Figure 9 GA-PID, ACO-PID and ZN-PID controller outputs at the different trajectory slopes (reference pressure $p_s = 2$ bar). [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Table 4 GA-PID, ACO-PID and ZN-PID Controller RMS Errors

	RMS errors for $p_s = 2$ bar			RMS errors for $p_s = 4$ bar		
Trajectory slope, α	GA-PID	ACO-PID	ZN-PID	GA-PID	ACO-PID	ZN-PID
6	0.0294	0.0304	0.2018	0.0295	0.0360	0.2382
10	0.0222	0.0185	0.1362	0.0235	0.0289	0.1629
15	0.0191	0.0172	0.0986	0.0222	0.0225	0.1174
20	0.0169	0.0162	0.0762	0.0205	0.0223	0.0938

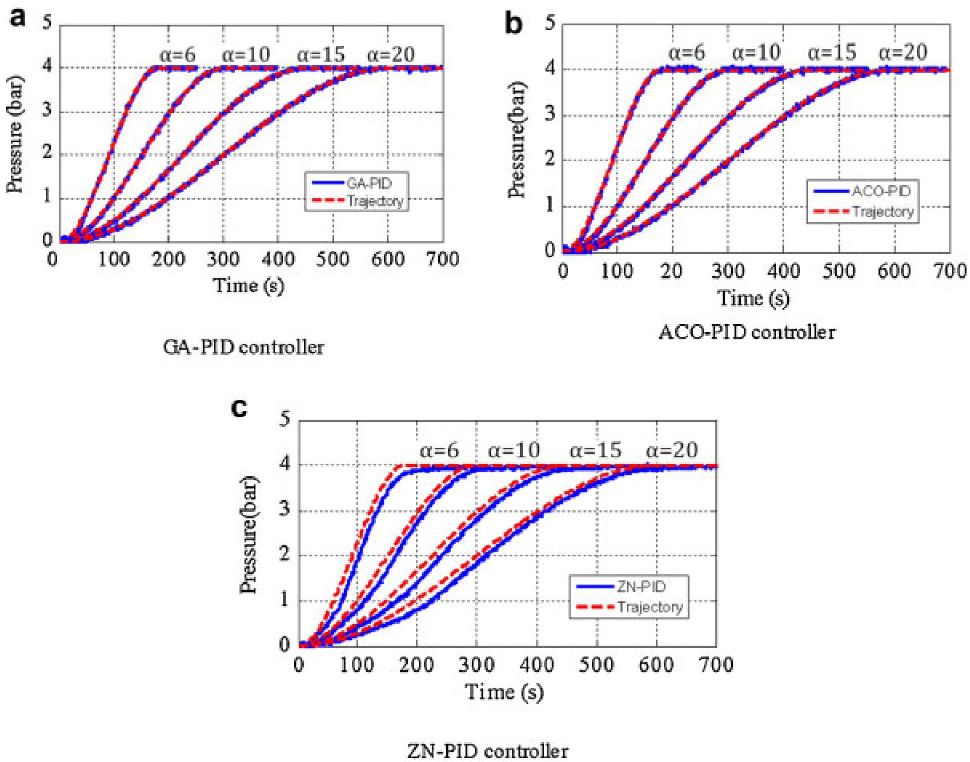


Figure 10 GA-PID, ACO-PID and ZN-PID controller outputs at the different trajectory slopes (reference pressure $p_s = 4$ bar). [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

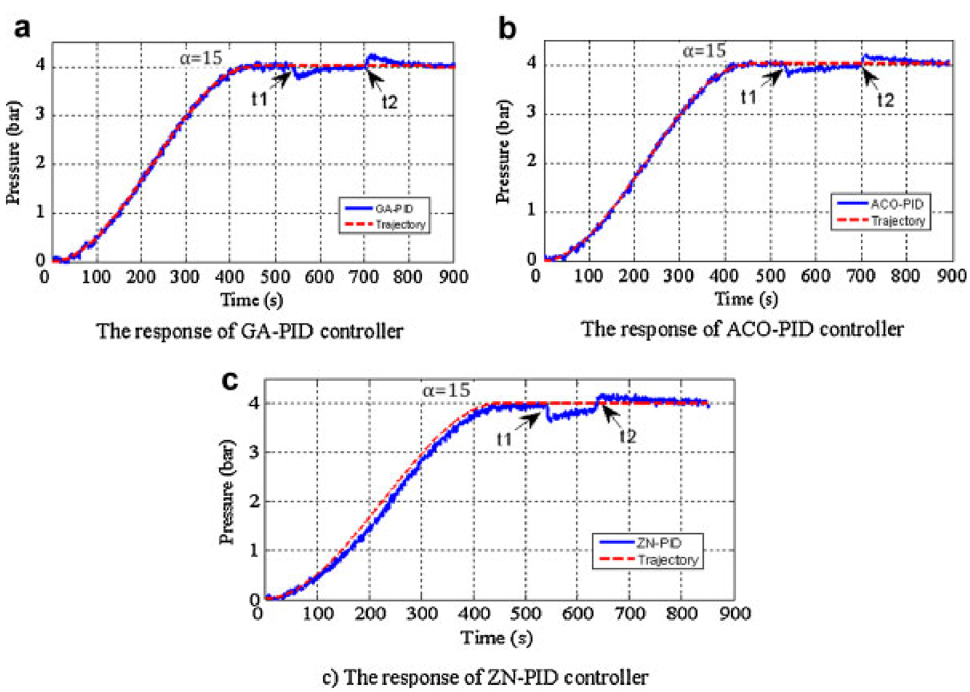


Figure 11 Disturbance rejection capalities of designed GA-PID, ACO-PID ve ZN-PID controller. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

error (MSE = 3.70E−07). At this step, different ANN parameters were tested ANN-30 model was found the good enough for modeling. Afterwards to prevent the fluctuations in the tank, a cubic trajectory function was used as an input reference signal. A cost function was design to minimize the error along the trajectory for the GA-PID and ACO-PID controller. To compare results with traditional approach PID controller was also tuned Z−N methods. Outputs were evaluated by means of RMS error the respect to change of trajectory slope and reference pressure and robustness of controllers were also observed.

At the end of the numerous real-time experiments, the following conclusions can be summarized:

(1) NARX type ANN architecture can be used the dynamic system modeling.
(2) Reference pressure has no effect on the controller performance.
(3) The RMS error was decreased when trajectory slope was increased.
(4) GA-PID and ACO-PID controllers show similar performance and theirs performances are better than ZN-PID controller at all reference pressure and trajectory values.
(5) All designed controller were traced the defined trajectory within limited time
(6) Finally, we concluded that both ACO and GA algorithms could be used to tune the PID controllers in the nonlinear system like pressure process easily with sufficient performance.

The example has been successfully integrated as the core of a process control course in the fourth year of the undergraduate program in computer and control education faculty in Marmara University.

REFERENCES

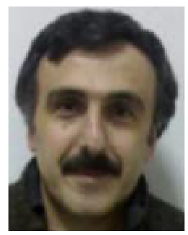
[1] J. Wu and W. Cai, Development of an adaptive neuro-fuzzy method for supply air pressure control in HVAC system, Systems, Man, and Cybernetics, IEEE International Conference, 3805 (2000), 3806−3809.
[2] H. S. Mok, G. T. Kim, M. H. Park, and H. W. Rhew, Pi controller gains tuning of the pressure control system by open-loop frequency response [thermal power plants], Industry Applications Society Annual Meeting, Conference Record of the IEEE, 551 (1988), 557−561.
[3] R. Namba, T. Yamamoto, and M. Kaneda, Robust PID controller and its application, IEEE (1997).
[4] G. Chengyi, S. Qing, and C. Wenjian, Real-time control of ahu based on a neural network assisted cascade control system, Conference on Robotics, Automation and Mechatronics, 2004.
[5] M. Ünal, Optimization of PID controller using ant colony/genetic algorithms and control of the gunt rt 532 pressure process, Institute for Graduate Studies in Pure and Applied Sciences, Master, Marmara University, İstanbul, Turkey, 2008.
[6] D. Jean-Sébastien and P. André, Process control through a case study: A mixing process. I. Siso case, Comput Appl Eng Educ 13 (2005), (4) 324−332.
[7] A. H. Jones and P. B. DeMoura Oliveira, Genetic auto-tuning of pid controllers, Genetic Algorithms in Engineering Systems: Innovations and Applications. GALESIA. First International Conference on (Conf. Publ. No. 414), 1995, 141−145.
[8] J. E. Seem, A new pattern recognition adaptive controller with application to hvac systems, Automatica 34 (1998), 969−982.
[9] K. J. Åström and T. Hägglund, PID controllers theory, design and tuning, Instrument Society of America, USA, 1995.
[10] Y. Lee, S. Park, M. Lee, and C. Brosilow, PID controller tuning for desired closed-loop responses for si/so systems, AIChE J 44 (1998), 106−115.
[11] P. Comino and N. Munro, Pid controllers: Recent tuning methods and design to specification, Control Theory and Applications, IEE Proceedings 149(1) (2002), 46−53.
[12] W. Feng and Y. Li, Performance indices in evolutionary cacsd automation with application to batch pid generation, Computer Aided Control System Design. Proceedings of the IEEE International Symposium (1999), 486−491.
[13] H. A. Varol and Z. Bingul, A new PID tuning technique using ant algorithm, American Control Conference, 2004.
[14] H. Ying-Tung, C. Cheng-Long, and C. Cheng-Chih, Ant colony optimization for designing of pid controllers, Computer Aided Control Systems Design. IEEE International Symposium (2004), 321−326.
[15] V. Topuz, Fuzzy genetic process control, Institute for Graduate Studies in Pure and Applied Sciences, PhD, Marmara University, Istanbul, Turkey, 2002.
[16] R. K. Ahuja, J. B. Orlin, and A. Tiwari, A greedy genetic algorithm for the quadratic assignment problem, Comp Oper Res 27 (2000), 917−934.
[17] E. Bonabeau, M. Dorigo, and G. Theraulaz, Inspiration for optimization from social insect behaviour, Nature 406 (2000).
[18] A. Colorni, M. Dorigo, and V. Maniezzo, Distributed optimization by ant colonies, The First European Conference on Artificial Life, Elsevier, 1992.
[19] M. Dorigo, Ant algorithms for discrete optimization, Artif Life 5 (1999), (no 2) 137.
[20] D. E. Goldberg and K. Sastry, A practical schema theorem for genetic algorithms design and tuning, University of Illinois, USA, 2001.
[21] E. A. Gonzalez and F. S. Caluyo, Normal Ziegler−Nichols-based PID retuning using sequential ant colony optimization (seqaco), 27th Annual Philippine-American Academy of Science and Engineering Meeting & Symposium (APAMS 2002), 2007.
[22] M. Jeffers, "A genetic algorithm based fuzzy logic controller," B.Eng. in Electronic Engineering, Dublin, 2001.
[23] M. Salami and G. Cain, An adaptive pid controller based on genetic algorithm processor, Genetic Algorithms in Engineering Systems: Innovations and Applications. GALESIA. First International Conference (Conf. Publ. No. 414) (1995), 88−93.
[24] G. Zhang, B. E. Patuwo, and M. Y. Hu, Forecasting with artificial neural networks: The state of the art, Int J Forecast 14 (1998), 35−62.
[25] "Advantech pci 1711 daq card", http://www.advantech.com/products/100-kS-s-12-bit-16-ch-SE-Input-PCI-Multifunction-Cards/mod_1-2MLGWA.aspx, December 30, 2009.
[26] B. Can, The optimal control of itu triga mark ii reactor, The Twelfth European Triga User's Conference, 1992.
[27] "Matlab & simulink", http://www.mathworks.com/, December 30, 2009.
[28] H. Demuth, M. Beale, and M. Hagan, Neural network toolbox 6 user's guide, Mathworks, September 2009.
[29] Matlab, Genetic algorithm and direct search toolbox 2 user's guide, Mathworks, September 2009.
[30] Matlab, Real-time windows target 3 user's guide, Mathworks, September 2009.
[31] E. Ronco and P. J. Gawthrop, Neural networks for modelling and control. Centre for System and Control, Department of Mechanical Engineering, University of Glasgow, Glasgow, Scotland, 1997.
[32] V. M. Ranković, Identification of nonlinear models with feedforward neural network and digital recurrent network, FME Trans 36 (2008), 87−92.
[33] M. T. Hagan, H. B. Demuth, and M. Beale, Neural network design. Thomson, Boston, 1997.

[34] Z. Michalewicz, Genetic algorithms + data structures = evolution programs. Springer, Berlin, 1996.
[35] D. E. Goldberg, Genetic algorithms in search, optimization and machine learning. Addison-Wesley, USA, 1989.
[36] L. Donghai, G. Furong, X. Yali, and L. Chongde, Optimization of decentralized PI/PID controllers based on genetic algorithm, Asian J Control 9 (2007), 306−316.
[37] T. Kanya, W. Yuji, A. Takuya, and O. Masato, Pi control adjusted by ga for ultrasonic motor, Electr Eng Jpn 169 (2009), (no 1) 59−65.
[38] H. Abdollah, B. Marwan, and G. Vijayarangan, Design using genetic algorithms of hierarchical hybrid fuzzy-PID controllers of two-link robotic arms, J Robot Syst 14 (1997), 449−463.
[39] W. K. L. Wilfred, K. Y. W. Allan, and S. L. W. Richard, Applying fuzzy logic and genetic algorithms to enhance the efficacy of the PID controller in buffer overflow elimination for better channel response timeliness over the internet, Concurr Comput Pract Exp 18 (2006), 725−747.
[40] P. Oliveira, J. Sequeira, and J. Sentieiro, Selection of controller parameters using genetic algorithms. Kluwer Academic Publishers, Dordrecht, Netherlands, 1991.
[41] H.-B. Duan, D.-B. Wang, and X.-F. Yu, Novel approach to nonlinear PID parameter optimization using ant colony optimization algorithm, J Bionic Eng 3 (2006), 73−78.
[42] K. Ogata, Modern control engineering. 3rd edition, Prentice Hall, New Jersey, 1997.

BIOGRAPHIES

Muhammet Ünal received the M.Sc. degree in Electronic and Computer Education from Marmara University, Turkey, in 2007. He has been research assistant in Technical Education Faculty of Marmara University since 2005. His research interests are system identification, parameter optimization and real time control.

Hasan Erdal received B.Sc. degrees from the Marmara University Technical Education Faculty, Department of Electrical Education, Istanbul, and the M.Sc. and the Ph.D. degrees in Electrical Education from the Marmara University, Institute for Graduate Studies in Pure and Applied Sciences, in 1990 and 1998 respectively. He has been Assistant Professor in the Electronics and Computer Education Department, Control Main Field since March 1999. His research interests focus on control education, automatic control applications, electrical machines, control algorithms and intelligent control.



Vedat Topuz received the MSc, and PhD degrees in Electronic and Computer Education Department from Marmara University, Istanbul, Turkey, in 1996, and 2002, respectively. Since 1992 he has been at the Vocational School of Technical Sciences at Marmara University, where he is currently assistant professor. His primary research interests in identification and control of nonlinear systems including artificial neural network, genetic algorithm, and fuzzy control applications.