International Journal of Sciences and Techniques of Automatic control & computer engineering
IJ-STA, Volume 5, N° 2, December 2011, pp. 1648–1659.

# Using Ant Colony Optimization Algorithm for Enhancing an Optimal PI Controller

Sofiane Bououden[1], Salim Filali[2], Mohammed Chadli[3], Fouad Allouani[1]

[1] Institut des Sciences et Technologies, Centre Universitaire Abbes Laghrour Khenchela, Algérie
ss_bououden@yahoo.fr

[2] Laboratoire d'automatique et de robotique Faculté des Sciences Route d'Ain El Bey, Université de Constantine, Algérie
sfilali@hotmail.com

[3] Laboratoire Modélisation Information et Systèmes, Université de Picardie Jules Verne,7 rue Moulin Neuf, 80000 Amiens, France
mohammed.chadli@u-picardie.fr

**Abstract.** *In this paper we describe the application of an Ant Colony Optimization (ACO) algorithm to optimize the parameters in the design of PI controller and to find the best optimal intelligent controller. The ACO algorithm is a bio-inspired optimization method that has proven its success through various combinatorial optimization problems. The parameters of the PI Controller are evaluated by an ant colony optimization by using an objective function based on position tracing error was constructed. The results obtained by the simulations are compared with previous work results obtained by a PI controller.*

**Keywords.** *PI controller, Bio-inspired algorithms, Ant Colony Optimization (ACO), parameters optimization.*

## 1. Introduction

The PI controller is a well known method for industrial control processes because of its simple structure and robust performance in a wide range of operating conditions [1]. Although conventional PI is popular, it is difficult to obtain satisfying control results to use the controller for systems which are nonlinear, time dependent and coupled. In the tuning process of a PI controller, tow parameters must b selected in such a way that the closed loop system has to give the desired response. There are many methods for tuning controller parameters [2-4]. Recently, intelligent systems such as fuzzy logic controller algorithm and evolutionary algorithms; genetic algorithm [5], particle swarm optimization [6], and ant colony system [7] have been applied to optimization problems, by introducing available information into the control design.

This paper was recommended for publication in revised form by the editor Staff.
Edition: CPU of Tunis, Tunisia, ISSN: 1737-7749

In the past decades, fuzzy control algorithms have become an alternative to address the difficulties of conventional PI controllers [8, 9]. They can deal with complex processes and combine the advantage of classical controllers and human operator experience. Suitable choice of control variables is important in fuzzy control design. The Ant Colony Optimization (ACO) algorithm is a meta-heuristic algorithm for the approximate solution of optimization problems that has been inspired by the foraging behavior of real ant colonies [10-13].

This work, describes the application of ACO as optimization methods to find the best possible intelligent fuzzy PI controllers where the desired response should have minimal settling time with a small or non overshoot in the step response of the closed system. In this study the ant colony optimization algorithm is used to obtain the excellent and optimal PI parameters by synthesizing them [14, 15].

This paper is organized as follows: Section 2 presents the modeling of nonlinear system using Takagi-Sugeno fuzzy model. Section 3 shows the basic concept of ACO algorithm and in Section 4 the development of the optimization methods is described. Section 5 includes a discussion of the results obtained and finally the conclusion is offered in the last section.

## 2. T-S fuzzy modeling

Consider a single-input single-output (SISO) nonlinear system. The system is decomposed into '$r$' subsystems such that each subsystem demonstrates a linear or nearly linear behavior. Using the Takagi_Sugeno's modeling methodology [16, 17], a fuzzy quasi-linear model, $R^i$ or FI, is developed for each subsystem. In such a model, the cause-effect relationship between control $u$ and output $y$ at sampling time $k$ is established in a discrete time representation. The subsystems are defined in the fuzzy regions, $R^i$, $i = 1, 2, \ldots, r$.

A SISO discrete-time nonlinear system can be described as follows:

$$\begin{aligned} x(k+1) &= F(x(k), u(k)) \\ y(k) &= H(x(k)) \end{aligned} \tag{1}$$

Where $u(k) \in R$ and $y(k) \in R$ are the system input and output at time $k$ respectively, $x(k) \in R^n$ is the state vector of the system, and $F(x(k), u(k)) \in R^n$ and $H(x(k)) \in R$ are nonlinear functions.

It is assumed that $F[0, 0] = 0$ and $H(0) = 0.$ for both a controllable and observable system, $x(k)$ can be expressed as function of $y(k), \ldots, y(k-n+1), u(k), \ldots, u(k-n+1)$, and $n$ represents the order of the system. Therefore, when the nonlinear system (1) is investigated around the origin, its equivalent system can be expressed as follows [18]:

$$\begin{aligned} y(k+d) &= a_1 y(k) + \ldots + a_{na} y(k-na-1) + b_0 u(k) + \ldots + b_{nb} u(k-nb) + l \\ &= \xi(k)^T \theta + l \end{aligned} \tag{2}$$

Where $1 \le d < n$ correspond to the time delay of the system, and $\xi(k) = [y(k), y(k-1), \ldots, y(k-n_a-1), u(k), \ldots, u(k-n_b)]^T$ is referred to as the regression vector $\theta = [a_1, \ldots, a_{na}, b_0, \ldots, b_{nb}]^T$. When the time delay is $d=1$, system (2) can be rewritten as Controlled Auto-Regressive Integrated Moving Average model (CARIMA) [19]:

$$A(z^{-1}) y(k) = B(z^{-1}) u(k-1) + \frac{\zeta(t)}{\Delta(z^{-1})} \tag{3}$$

Where $A(z^{-1})$ and $B(z^{-1})$ are polynomials in the backward shift operator $z^{-1}$

$$\begin{cases} A(z^{-1}) = 1 + a_1 z^{-1} + \ldots + a_{na} z^{-na} \\ B(z^{-1}) = b_0 + b_1 z^{-1} + \ldots + b_{nb} z^{-nb} \end{cases}$$

$\zeta(t)$ is an uncorrelated random sequence and the use of the operator $\Delta(z^{-1}) = 1 - z^{-1}$ ensures an integral control law or a closed loop type I system.

FI consists of a set of symbolic antecedents in the IF part (premise) and a linear numerical expression in the THEN part (consequence). Using a CARIMA model structure, the fuzzy implication of the (3) can be written as follows [22]:

$R^i$ : IF $y(k)$ is $M_1^i$, $y(k-1)$ is $M_2^i, \ldots, y(k-na-1)$ is $M_{na}^i$ and $u(k)$ is $L_0^i$, $u(k-1)$ is $L_1^i, \ldots, u(k-nb)$ is $L_{nb}^i$ THEN $y^i(k+1) = \dfrac{B^i(z^{-1})}{A^i(z^{-1})} u(k)$

$$i = 1, \ldots, r \tag{4}$$

Where

$M_j^i$ fuzzy set corresponding to output $y(k-j)$ in the *i*th FI;

$L_p^i$ fuzzy set corresponding to output $u(k-p)$ in the *i*th FI;

The system output $y(k+1)$ is computed as the weighted average of the individual rules consequents

$$y(k+1) = \frac{\sum_{i=1}^{r} w^i\, u(k) B^i(z^{-1}) / A^i(z^{-1})}{\sum_{i=1}^{r} w^i} \tag{5}$$

The degree of fulfillment of the *i*th rule, $w^i$ is obtained as the product of the membership degrees of the antecedent variables in that rule

$$w^i = \prod_{j=1}^{na} \mu_{M_j^i}(y(k-j-1)) \prod_{p=0}^{nb} \mu_{M_j^i}(u(k-p)) \tag{6}$$

## 3. Based Ant Colony Algorithm

Ant colony algorithm is the simulation of cooperation course of real ant group. Each ant searches the solution independently in the candidate solution space, meanwhile leaves some information on the found solution. Ants leave more information on the solution with better performance, which has more possibility to be selected. All the solutions have the same information on it in the elementary period of the algorithm. With the progress of the computing, the better solution gets more and more information. Therefore, the algorithm reaches the optimization or approximately optimization solution.

We illustrate is an algorithmic implementation that adapts the behavior of real ants to solutions of minimum cost problems on graphs [21]. A number of artificial ants build solutions for a certain optimization problem and exchange information about the quality of these solutions making allusion to the communication system of real ants [22].

Let us define the graph $G$=($V$, $E$), where $V$ is the set of nodes and $E$ is the matrix of the links between nodes. $G$ has n=|$V$| nodes. Let us define $L^k$ as the number of hops in the path built by the ant $k$ from the origin node to the destiny node. Therefore, it is necessary to find:

$$Q = \{q_a, \dots, q_f \,|\, q_1 \in C\} \tag{7}$$

Where $Q$ is the set nodes representing a continuous path with no obstacles; $q_a,\dots, q_f$ are former nodes of the path and $C$ is the set of possible configurations of the free space. If $x^k(t)$ denotes a $Q$ solution in time $t$, $f\,(x^k(t))$ expresses the quality of the solution. The ACO algorithm is based on Eqs (8)-(10):

$$p_{ij}^k(t) = \begin{cases} \dfrac{\tau_{ij}^k}{\sum_{j \in N_{ij}^k} \tau_{ij}^k(t)} & \text{if } \ j \in N_i^k, \\ 0 & \text{if } \ j \notin N_i^k, \end{cases} \tag{8}$$

$$\tau_{ij}(t) \leftarrow (1-\rho)\tau_{ij}(t), \tag{9}$$

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \rho.\Delta\tau_{ij}^k \tag{10}$$

Where $\Delta\tau_{ij}^k = \sum_{k=1}^{n} \tau_{ij}(t).$

Eq. (8) represents the probability for an ant $k$ located on a node $i$ selects the next node denoted by $j$, where, $N_i^k$ is the set of feasible nodes (in a neighborhood) connected to

node *i* with respect to ant *k*, $\tau_{ij}$ is the total pheromone concentration of link (*i, j*), and $\alpha$ is a positive constant used as a gain for the pheromone influence.
Eq. (9) represents the evaporation pheromone update, where $\rho \in [0, 1]$ is the evaporation rate value of the pheromone trail. The evaporation is added to the algorithm in order to force the exploration of the ants, and avoid premature convergence to sub-optimal solution [21]. For $\rho$=1 the search becomes completely random [21].
Eq. (10) represents the concentration pheromone update, where $\Delta\tau_{ij}^{k}$ is the amount of pheromone that an ant *k* deposits in a link (*i, j*) in a time *t*.
The general steps of ACO are the following:

1. Set a pheromone concentration $\tau_{ij}$ to each link (*i, j*).
2. Place a number *k*=1, 2, …, *n* in the nest.
3. iteratively build a path to the food source, using Eq. (8) for every ant.
- Remove cycles and compute each route weight $f\,(x^k(t))$. A cycle could be generated when there are no feasible candidates nodes, that is for any *i* and any *k*, $N_i^k = \Phi$; then the predecessor of that node is included as a former node of the path.
4. Apply evaporation using Eq.9.
5. Update of the pheromone concentration using Eq. (10).
6. Finally, finish the algorithm in any of the three different ways:
- When a maximum number of epochs has been reached.
- When it has been found an acceptable solution, with $f\,(x^k(t))<\varepsilon$.
- When all ants follow the same path.

## 4. Optimization of the linear fuzzy PI controller

In this section we consider the problem of tuning the linear fuzzy PI controller. To each parameter we associate several competing candidates. Fro each parameter we need the minimum and the maximum possible values. To simplify, the proposed values are equally distributed between these bounds. Lets $M^i_{\min}$ and $M^i_{\max}$ be the bounds of *i*th element of the parameter vector *M*. Therefore we can take:

$$M_{i1} = M^i_{\min},\; M_{i2} = M_{i1} + \frac{M^i_{\max} - M^i_{\min}}{j-1}, \ldots, M_{iN} = M^i_{\max} \tag{11}$$

Thus:

$$M_i \in \{M_{i1}, M_{i2}, \ldots, M_{iN}\} \tag{12}$$

The problem is to find the best parameter combination that minimizes the cost function.

The tour of an ant consists of a combination of the fuzzy controller parameters. Staring from its nest, an ant moves through the KP and KI. Finally, the ant reaches the food source F which is added here just to match the real world.

### 4.1. The rule of selecting parameters by ants

For each set of parameters, the node visited by the ant is selected as the value of the parameter. Selection of a parameter value is based on pheromone trails between parameter vectors. The size of pheromone matrix $\tau_{ij}$ is 2 *N, *i*= 1, 2 and *j*=1, 2, …, *N*. As when the ant arrives at vector $M_r$, selection of the next parameter value $M_{r+1j}$ among the candidate list $M_{r+1}$ is done using the state transition rule (14) which depends on pheromone trails $\tau_{r+1j}$ , the cost function and some heuristic.

$$s = \begin{cases} \arg\max_{y\in J}\{\tau_{iy}(t)\} & if\ q \le q_0 \\ S & else\ according\ to\,(8) \end{cases} \tag{14}$$

Where $q$ is an equally distributed random number in region [0, 1], $q_0$ is a parameter $(0 \le q \le 1)$, $S$ is a random variable selected according to the possibility form (8).

The ant finishes one cycle after time. The pheromone intensity change on the route can be local updated according (9), and can be rewrite as

$$\tau_{ij}(t) \leftarrow (1-\rho)\tau_{ij}(t) + \rho\tau_0 \tag{15}$$

Where $\tau_0 = (ITAE_0)^{-1}$

$$ITAE = T_e\int_0^t |e(t)|dt \tag{16}$$

Where $T_e$ is the sampling time and *ITAE* is the fitness value of the system when the gains KP, and KI of the PI controller are tuned by the Ziegler-Nichols criterion.

When all ants have completed a tour, the best tour needs to be updated using the global pheromone updating formula according (4), and can be rewrite also as

$$\tau_{ij}(t) \leftarrow (1-\rho)\tau_{ij}(t) + \rho\Delta\tau_{ij}(t) \tag{17}$$

$$\Delta\tau_{ij}(t) = 1/\,ITAE^{*} \tag{18}$$

Where ITAE* is the value of ITAE corresponding to the best path.

However the parameters are fixed based on the available knowledge or by a first tuning pass of the linear fuzzy controller by the ACO algorithm up presented or by any other optimization method. The tuning parameters are the conclusion of fuzzy rules.

## 5. Simulations results

The objective of this section to test the efficient of the the PI-ACO controller and PI controller and show the performances of these controllers.

*Example:* control of surge tank model

Consider the surge tank model given in Fig. 1. It is fed by a pump driven by a current $i(t)$. That can be represented by the following differential equations:

Model of the valve

$$\frac{dQ(t)}{dt} + k_0 Q(t) = k_1\, i(t) \tag{19}$$

The change in water level in the tank is given by:

$$\frac{dV(t)}{dt} = A\frac{dH(t)}{dt} = Q(t) - Q_s(t) \tag{20}$$

where: $Q_s(t) = 0.6a\sqrt{2g\left(H(t) - H_s\right)}$

| Process variable | Normal operation condition |
|---|---|
| Initial value of tank level ($H_0$) | 0.15m |
| Initial value of the output channel level Hs | 0.015m |
| Section of the channel output (a) | 0.0001m² |
| Section of the tank (A) | 0.04m² |
| the initial flow $q_0$ | 0.0001 $m^3/s$ |
| Constant ($k_0$) | 1 |
| Constant ($k_1$) | 0.1 |

Table 1: specification of the surge tank

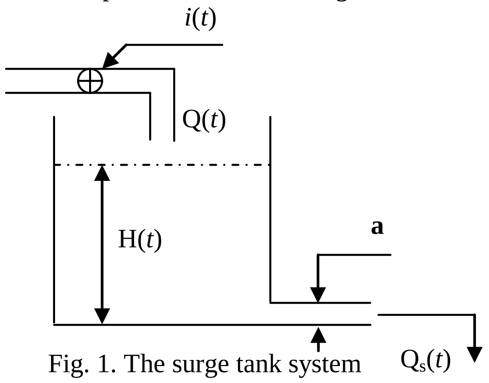

Fig. 1. The surge tank system

where Q($t$) the feed rate, $i(t)$ is the supply current of the pump, H($t$) the liquid level in the tank, $Q_s(t)$ the output Flow, **a** is the section of the output channel, **A** is the section of the tank and $H_S$ the water level in the output channel.

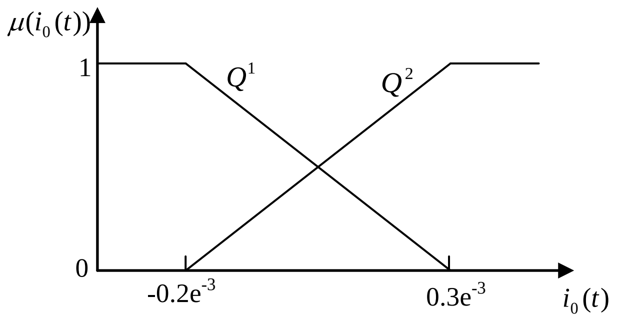

Fig. 2. Definition of fuzzy sets $Q^1$ and $Q^2$ for FIs $R^1$ and $R^2$, respectively

***Fuzzy modeling***: The fuzzy model was identified using data from the real process, sampled with the period $T_s$=0.1s. For a good approximation of the plant, we suppose that the subsystems are in the third order.

The model consists of two rules of the form

$R^1$: IF $i_0$ is $Q^1$
THEN $H^1(k+1)=a_{11}H(k-1)+\ldots+ a_{1na} H(k-n_a)+b_{11}i_0(k-1)+\ldots+b_{1nb}i_0(k-n_b)$ (21)
$R^2$: IF $i_0$ is $Q^2$
THEN $H^2(k+1)=a_{21}H(k-1)+\ldots+ a_{2na} H(k-n_a)+b_{21}i_0(k-1)+\ldots+b_{2nb} i_0 (k-n_b)$ (22)

The fuzzy model is structurally very simple and requires only two FIs, Fig. 9 corresponds to the fuzzy sets $Q_1$ and $Q_2$. The vector of parameters of $i^{th}$ rule is obtained by using the recursive least squares (RLS) [23].

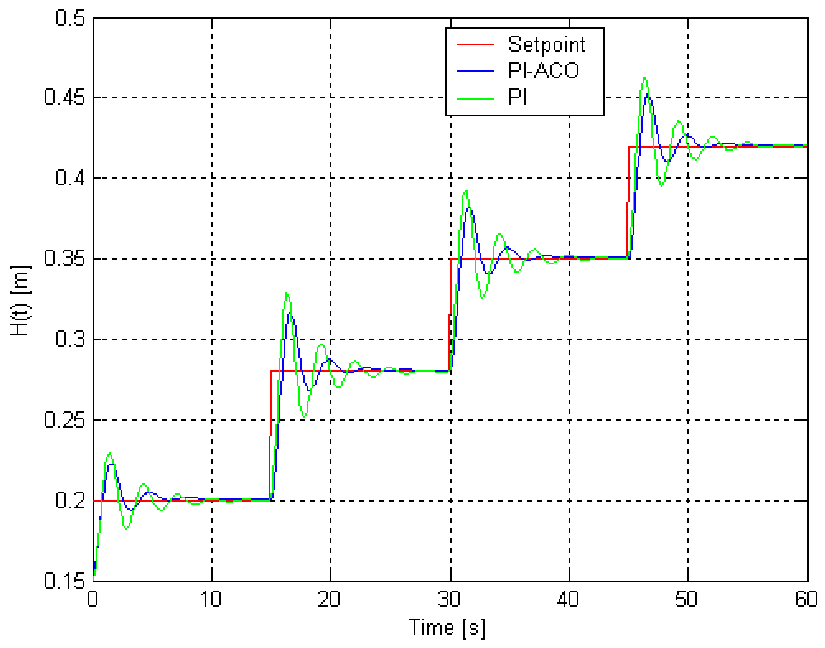

Fig.3 . Comparison of the closed-loop dynamic responses by the PI-ACO, and PI controller.

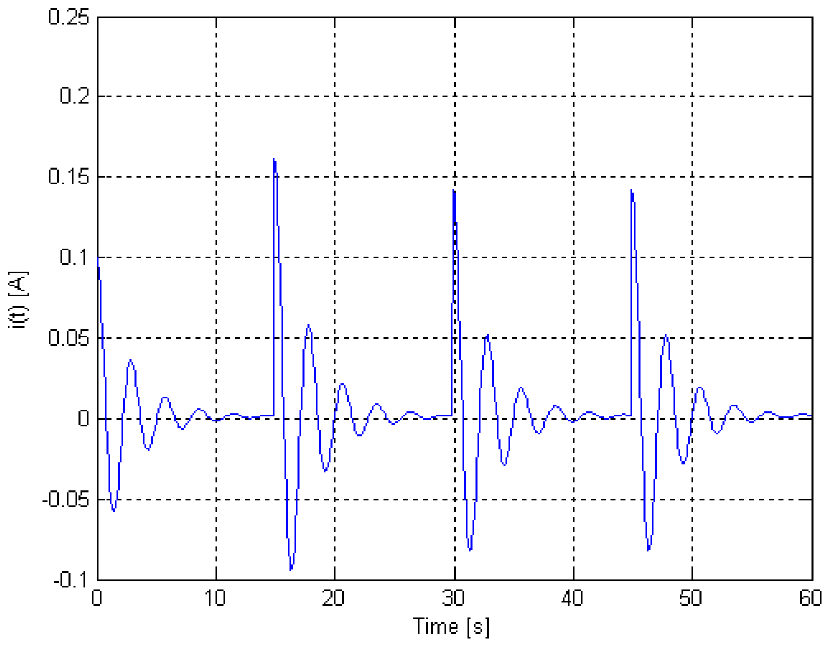

Fig. 4. Control performance of the PI controller

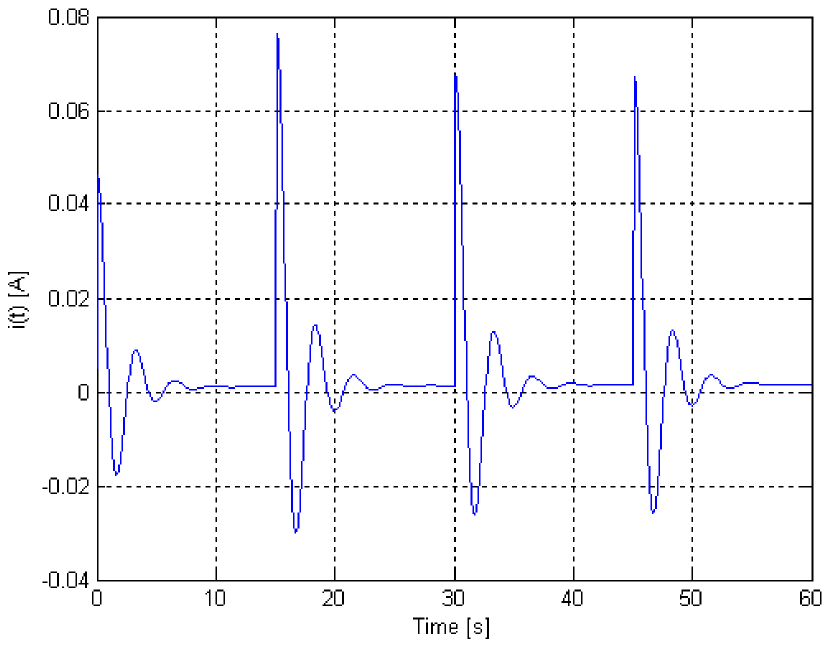

Fig. 5. Control performance of the PI-ACO controller

| | PI | | | | PI-PSO | | | |
|---|---|---|---|---|---|---|---|---|
| Time(s) | 0-15 | 15-30 | 30-45 | 45-60 | 0-15 | 15-30 | 30-45 | 45-60 |
| Maximum overshoot (%) | 14.65 | 17.14 | 12.17 | 10.11 | 11.27 | 13.03 | 9.14 | 7.61 |
| Rise time (s) | 0.8 | 0.7 | 0.7 | 0.7 | 0.87 | 0.85 | 0.85 | 0.85 |
| Settling time(s) | 6.18 | 7.3 | 5.85 | 4.7 | 5.25 | 5.31 | 3.65 | 3.3 |
| Steady state error | 0.002 | 0.0 | 0.002 | 0.003 | 0.0 | 0.0 | 0.0 | 0.0 |

Table 2: Comparison of performance of the PI-ACO and AFMPC

This table shows the performances obtained by each method. In each time interval, we have changed the set point for evaluated each method to control a valve of the tank system.

The comparison shows some interesting results. The PI controller is not capable of keeping the level at the desired value. It is important to observe that with PI-ACO the settling time has been reduced comparing to that obtained from the PI controller without any increase in overshoot. Between 45-60s we can see from the overshot, settling time and steady state error that obtained by the PI controller are degraded. With PI-ACO the overshoot is reduced to one third compared to that obtained from PI controller. The same observation can be made for the settling time, where in the PI-ACO case we notice a reduction of nearly to one quarter compared to that obtained from PI. So, the PI-ACO is able to keep better stability with less control effort applied.

## 6. Conclusion

This paper introduce an ACO based tuning method for PI controller to control a given water level, this ACO algorithmic approach is used for optimizing the parameters of a PI controller. The ACO algorithm shows great advantages in solving combination optimization problems, because it is realized easily, and it is adaptable with the change of the plant parameters. The PI-ACO controller is equivalent to the classical PI controller, but this controller combines the advantage of classical PI and ACO algorithm, namely; the rapidly attaining the optimal solution, the simplicity and interpretably which make the parameters tuning task easier. The simulation results show that the proposed method can improve the performances and efficiently of ACO algorithm to solve difficult optimization problems.

## References

1. Astrom, K.J., Hagglund, T., Theory, Design and Tuning, Instrument Society of America, Research Triangle Park, North Carolina (1995).
2. Ziegler, J.G., Nichols, N.B.: Optimum Setting for Automatic Controllers. Trans. ASME, (1942) 759–768.
3. Garcia, C.E., Morari, M.: Interanl Model Control. 1.A Unifying Review and Some New Results. Ind. Eng. Chem. Process. Dev. 21, 308–323 (1982)
4. Astrom, K.J., Hagglud, T.: PID Controller: Theory, Design and Tuning. Research Triangle Park, NC (1995)
5. Man, K., Tang, F.K.S., Kwong, S.: Genetic Algorithms. Springer, Heidelberg (1999)
6. Zheng, Y.L., Ma, L., Zhang, L., Qian, J.: On the Convergence Analysis and Parameter Selection in Particle Swarm Optimization. In: 2nd IEEE International Conference on Machine Learning and Cybernetics, pp. 1802–1807 (2003)
7. Dorigo, M., Maniezz, V., Colorni, A.: Ant System: An Autocatalytic Optimizing Process. Technical Report No. 91-016 Revised, Politecnico Di Milano, Italy (1991)

8. Jantzen, J., Foundations of fuzzy Control. West Sussex, John Wiley & Sons Ltd, 2001.

9. Yager, R.R., Filev, D.P., Essential of fuzzy modeling and control. John Wiley & Sons Inc, 1995.

10. Dorigo, M., Gambardella, Ant colony system: Acooperative learning approach to the travelling salesman problem, IEEE Transaction on Evolutionary Computation, 1 (1) (1997) 53–66.

11. Maier, H.R., Simpson, A.R., Zecchin, A.C.,Foong, W.K., Phang, K.Y., Seah, H.Y.,Tan, C.L., Ant Colony Optimization for the design of water distribution systems, Journal of Water Resources Planning and Management, ASCE 129 (3) (2003) 200–209.

12. Zecchin, A.C., Simpson, A.R., .Maier, H.R., Nixon, J.B., Parametric study for an ant algorithm applied to water distribution system optimization, IEEE Transactionson Evolutionary Computation 9 (2) (2005) 175–191.

13. Dorigo, M., Birattari, M., Stutzle, T., Ant colony optimization, IEEE Computational Intelligence Magazine (2006) 28–39.

14. Duan, H.-B., Wang, D.-B., Yu, X.-F., Novel approach to nonlinear PID parameter optimization using ant colony optimization algorithm, Journal of Bionic Engineering, 3 (2006) 73–78.

15. Varol, H.A., Bingul, Z., A new PID Tuning Technique Using Ant Algorithm, Proc. American Control Conf., Boston, Massachustts, USA, 2004 2154–2159.

16. Takagi, T., Sugeno, M., Fuzzy identification of systems and its application to modeling and control, IEEE Transactions on Systems. Man and Cybernetics. 15 (1) (1985) 116–132.

17. Espinosa, J., Vandewalle, J., Wertz, V., Fuzzy logic, Identification and Predictive Control. London, UK: Springer, 2005.

18. Bououden, S., Filali, S., Kemih, K., Adaptive Fuzzy Tracking Control for unknown nonlinear systems, International Journal of Innovative Computing, Information and Control, 6 (2) (2010) 541–549.

19. Bingül, Z., Karahan , O., A Fuzzy Logic Controller tuned with PSO for 2 DOF robot trajectory control, Expert Systems with Applications 38 (2011) 1017–1031.

20. Flores, A., Saez, D., Araya, J., Berenguel, M., Cipriano, A., Fuzzy predictive control of a solar power plant, IEEE Transactions on Fuzzy Systems, 13 (1) (2005) 58–68

21. Garcia, M.P., Montiel, O., Sepulveda, R., An ACO path planner using a FIS for a path selection adjusted with a simple tuning algorithm, JAMRIS 2 (1) (2008) 5–11.

22. Dorigo, M., Stutzle, T., Ant Colony Optimization, Bardford, Cambridge, Massachusetts, 2004.

23. Lightbody, G., Irwin, G., Nonlinear control structures based on embedded neural system models, IEEE Transactions on Neural Networks 8 (1997) 553–567.