Advance in Electronic and Electric Engineering.
ISSN 2231-1297, Volume 3, Number 5 (2013), pp. 553-560

http://www.ripublication.com/aeee.htm

Optimal Fuzzy Supervised PID Controller using Ant Colony Optimization Algorithm

R. Mitra and Samarth Singh

Electronics and Communication Department, IIT Roorkee
Roorkee, Uttarakhand, INDIA.

Abstract

PID controllers are the well known and most widely used controllers in the industries. The reason behind this is because of its simple structure and reliability. Nonetheless when the plant to be controlled is highly non linear or is subjected to disturbances or we have less knowledge about it, under these conditions poor performance is obtained when we are using fixed parameter PID as controller. Thus an expert supervisor is required to online tune its controller parameters. But because of the known shortcomings of a human supervisor such as chances of poor tuning or damage to the plant caused due to carelessness of supervisor, automatic tuning is preferred. Using a fuzzy block for the role of supervisor is a good option because of its known qualities of mimicking human operator and robustness. However an expert is still necessary to determine the parameters of membership functions, fuzzy rules and scaling factors of the fuzzy block. Also a good degree of knowledge about system is also required for tuning parameters of fuzzy block. So we can use of an optimization technique which can tune the parameters of the fuzzy block which in turn gives us a superior performance. Several techniques are available, here we have made use of ACO (Ant Colony Optimization) algorithm which follows the food gathering ability of one of the most successful species on earth i.e. ants. Simulation on MATLAB and SIMULINK show better performance obtained by using above mentioned tuning scheme as compared to well known Zieglar-Nicolas scheme for a second order system.

Keywords: PID, Fuzzy, Fuzzy supervisor, ACO (Ant Colony Optimization) algorithm.

1. Introduction

PID controller are commonly used controllers in process industries. This is because of their various merits such as it can be easily be understood by plant operators, near optimal performance ,wide applicability etc. Drawbacks of using PID are that it requires frequent online tuning ,gives poor performance for systems which are highly non linear and subjected to disturbances. So in order to make use of advantages of PID controller a modified PID controller is an attractive scheme. In this paper we have Integrated a fuzzy block as a supervisor with the PID which facilitates automatic online tuning. Further optimizing the fuzzy block parameters by using Ant Colony Algorithm minimizes the involvement of any human expert in adjusting these parameters, thereby reducing time and effort.

2. Fuzzy Supervisor

2.1 Effectiveness of fuzzy block as supervisor

We know a fuzzy block can implement human knowledge by making use of linguistic variables, linguistic rules and fuzzy membership functions. Thus it can mimic an expert human operator and can efficiently tune the control parameters of PID. The Fuzzy supervisor scheme utilizes fuzzy logic and reasoning to determine controller parameters, PID controller generates the control signal. Incorporating fuzzy block also makes the overall controller non linear and thus can handle non-linear systems. Also it brings desirable quality of robustness into the overall controller. Thus better performance can be expected from this method as compared to fixed parameter PID controller.

2.2 Fuzzy gain scheduling

Here the fuzzy supervisor make use of gain scheduling technique to tune the PID parameters. Let us first see the control strategy implemented by the PID controller.

$$(t) = K_p e(t) + K_i \int e(t) + K_d \frac{de(t)}{dt} \mathrm{U} \qquad (1)$$

Where K_p ,K_i and K_d are PID controller parameters that are to be tuned.

In the following figure a block diagram for fuzzy gain scheduling is given, the figure shows how fuzzy block tunes the PID parameters

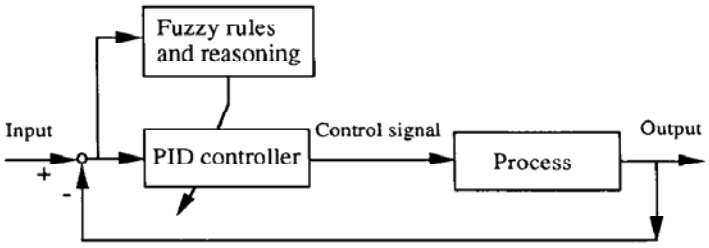

Figure 1: Fuzzy gain scheduling.

Kp and Kd are in range [Kpmax and Kpmin] and [Kdmax and Kdmin].The values of Kpmax, Kpmin, Kdmax and Kdmin has to be found from following:

$K_{pmin}=0.32*K_u$,$K_{pmax}=0.6*K_u$, $K_{dmin}=0.08*K_u*T_u$, $K_{dmax}=0.15*K_u*T_u$ (2)

Eq.(2) can be used as a thumb rule for finding above. Here K_u is the critical gain and T_u is the period of oscillation when only P control is used for the system. After this K_p and K_d is normalized as K_p' and K_d' as follows:

$$K_p^{'} = \frac{(K_p - K_{pmin})}{(K_{pmax} - K_{pmin})} \tag{3}$$

$$K_d^{'} = \frac{(K_d - K_{dmin})}{(K_{dmax} - K_{dmin})} \tag{4}$$

K_i can be found out from the following:

$$K_i = \frac{K_p^2}{\alpha K_d} \tag{5}$$

Value of α can be between values 2 to 4 ,a value less than 4 generally generates strong integral action.

Membership functions used are given in the following figure 2a,2b(In next page).

Rules can be formulated experimentally based upon step response of the process. Like suppose at point a_1(In Figure 2c next page) ,to have a big control signal, a large proportional gain ,a large integral gain and a small derivative gain is required. At b_1, we need to have a small control signal to avoid large overshoot,thus a small integral gain, small proportional gain is required and a large derivative gain is required.

Thus subsequently from rest of the points we can formulate all the rules. The rule structure is as follows:

if e(k) is W_i and Δe(k) is X_i then K_p' is Y_i ,K_d' is Z_i and $\alpha=\alpha_i$

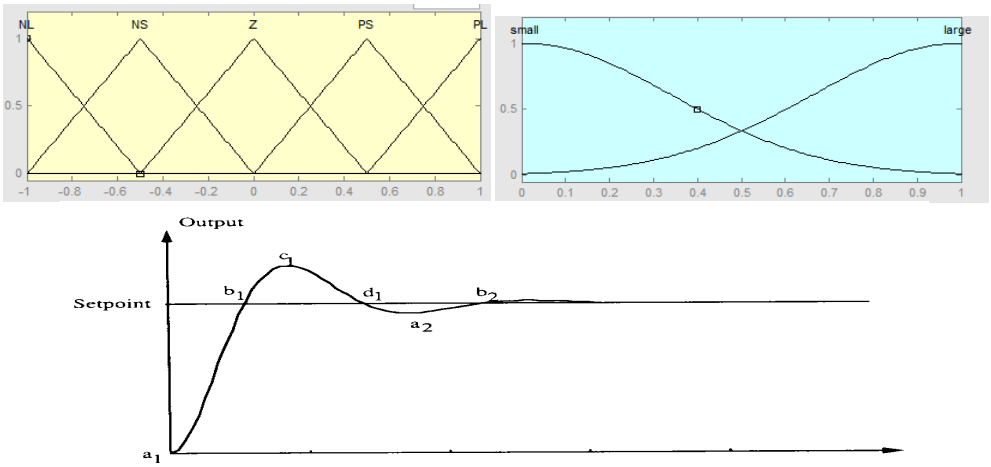

Figure 2: (From left) a.Membership function for e(k) and Δe(k) b.Membership function for output of Kp' and Kd' c.Step response of a system Where Wi, Xi, Yi and Zi are membership functions. For defuzzification bisector method has been use as from simulation results it was seen to give better response. Scaling factors have also been added.

3. Ant Colony Optimization Algorithm

3.1 Basic concept

Ant colony algorithm was inspired from the food gathering behavior of ants. It is a probabilistic technique used for finding shortest path in graphs. . The ACO is an algorithmic implementation that adapts the behavior of real ants to solutions of minimum cost path problems on graphs . 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 systems of the real ants .

3.2Implementation of algorithm

Following are the steps for implementation:

1. First we will define the objective function that is to be optimized, here we will make use of ITSE(Integral Time multiplied by Square of Error) in the objective function. ITSE is given as:
2.

$$ITSE=\sum k*e(k) \qquad (6)$$

where k is the sampling instant and e(k) is the error at that instant. Objective function given as:

$$Obj=Q/ITSE \qquad (7)$$

Where Q is some positive constant ,we have taken objective function in such fashion because ant colony algorithm would maximize the objective function ,whereas the ITSE needs to be minimized.

3. Discritize all the parameters that are to be optimized. Here we are optimizing the scaling factors of the fuzzy block. Decide their range and minimum division. Consider each division as a node on a graph , for example suppose range of a certain scaling factor is 0 to 5 and its minimum division is taken as 0.1 ,the nodes on the graph for that parameter are(0.1,0.2,0.3.....,4.9,5).
4. Now decide the number of ants required as per your convenience, for each ant form a path table which stores the path of that ant for that particular iteration, here path comprises of nodes of the parameters ,each node stored is of distinct parameter. There also should be a pheromone table storing pheromone content at each node. An optimal table should also be there to store the most optimal path.
5. Specify the number iterations required, set a counter having value equal to zero. Then begin the algorithm.
6. For each ant find the transitional probability to each node as it travel along a particular path. We can find transitional probability from the following:

$$P_{sd}^{k} = \frac{\tau_{sd}^{\alpha}(t)}{\sum_{j=1}^{q} \tau_{sj}^{\alpha}(t)} \tag{8}$$

where P_{sd}^{k} represent probability of any k[th] ant from a particular source node s to some destination node d. Here τ_{sd} is the pheromone content of node d which is in neighborhood of s, α is certain constant that give us certain degree of control over the pheromone deposition and τ_{sj} represent pheromone content of any neighboring node of s. All these neighboring nodes are associated with a single parameter.

1. After the probabilities have been found out allow the ants to travel, the ants will choose a path depending upon these probabilities. Let them store their path in their respective path table. Set the path table values as parameter values of the fuzzy block, run the simulation and obtain Obj(see Eqn(6)) for each ant, also store this value of Obj for that particular ant.
2. At the end of iteration update the pheromone deposition at each node from the following:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t) \tag{9}$$

Where i is representing a particular parameter and j a particular node associated with that parameter, ρ is the decay constant which is necessary so that pheromone deposition on nodes associated wih non optimal paths decrease significantly as compared to nodes on optimal path after each iteration, also

$$\Delta\square_{ij} = \sum_{k=1}^{m} \Delta\square_{ij}^{k} \tag{10}$$

where $\Delta\square_{ij}^{k}$ represent pheromone of any k[th] ant on node j of parameter i,

$$\Delta\square_{ij}^{k} = Obj \tag{11}$$

Obj can be found out from Eqn(7) and it value has already been found out for every ant. Store the most optimal path giving the maximum Obj in this as well as all the previous iterations in the optimal path table. Increment the counter by 1.

1. End the algorithm if either all the ants start to follow a single path or the counter achieves maximum value of iterations.Else go to step 5.
2. Set the optimal parameters so obtained for the fuzzy block , this gives us the best possible performance.

4. Simulation and Results

Here figure 3a shows the responses for a simple second order plant system $\frac{2}{s^2+4s-1}$ by using Fixed parameter PID controller and Fuzzy supervised PID controller. The ACO algorithm was run to optimize the scaling factors for the fuzzy block in the fuzzy supervisor ,as we can see the Fuzzy supervised PID absolutely shows no overshoot and settles to the final value in less than 0.5 second ,whereas we can see the fixed parameter PID(It was tuned using Zieglar-Nicholas tuning followed by further manual tuning which involved more time and effort) shows noticeable overshoots and settles around in 1 second.

Now to see which controller offers more robustness, we change the parameters of the plant system, let the plant system's transfer function now be $\frac{1}{s^2+2s-1}$,responses are given in Figure 3b. we see that in case of fuzzy supervised PID a slight overshoot can be seen and the settling time is still less than 0.5 second, whereas for fixed parameter PID Overshoot has increased as well as the settling time is around 2 second. Thus we can clearly see that not only fuzzy supervised PID offers better response but it also is more robust as compared to fixed parameter PID.

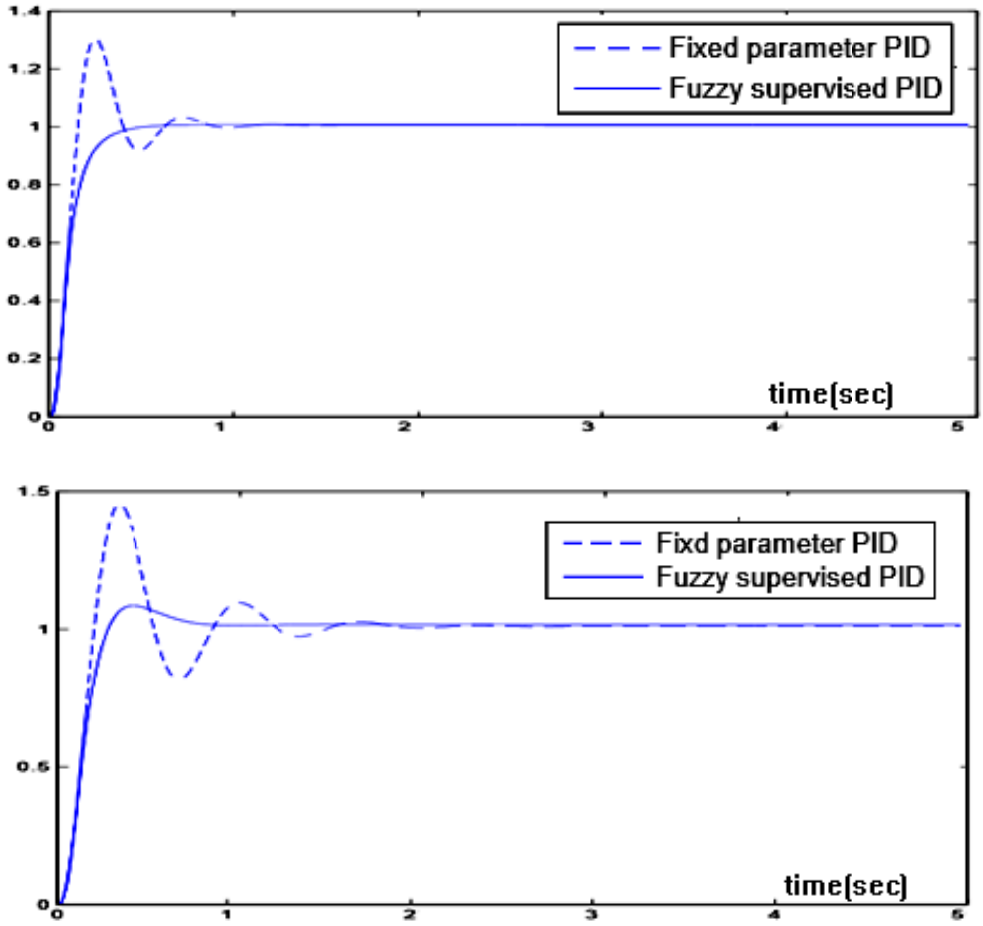

Figure 3: (From top)a. Response 1st b. Response 2nd

5. Conclusion

From the simulation results we can conclude that the addition of a fuzzy supervisor to the PID has not only significantly improved the performance but has also contributed to the automatic online tuning of the PID. While the use of ACO in optimizing the scaling factors of the fuzzy block not only saves time and effort but has also resulted in a superior performance. Desirable quality like robustness has also been achieved.

References

[1] Marco Dorigo,Thomas Stutzle(2004) ,Ant Colony Optimization, The MIT Press Cambridge, London, England

[2] Oscar Castillo,Witold Pedrycz, and Janusz Kacprzyk(2009), Evolutionary Design of Intelligent Systems in Modeling, Simulation and Control, Springer-Verlag Berlin Heidelberg, PP 4-5.

[3] Yongsheng Zhao, Baoying Li(2007), A New Method for Optimizing Fuzzy Membership Function, *International Conference on Mechatronics and Automation* Harbin, China. PP 674-676.

[4] Zhen-yu Zhao, Masayoshi Tomizuka and Satoru Isaka(1993), Fuzzy gain scheduling of PID controllers, *IEEE transactions on System, Man. and Cybernetics, Vol 23,No 5,*PP 1392-1394.