PID Control based on Modified Particle Swarm Optimization for Nonlinear Process

Adel Taeib, Ltaeif Ali, Abdelkader Chaari
Research Unit C3S ESSTT
University of Tunis, TUNISIA
taeibadel@live.fr, {ltaief24, nabile.chaari}@yahoo.fr

ABSTRACT: *In this paper, a novel design method for determining the optimal proportional-integral-derivative (PID) controller parameters for Takagi-Sugeno (T-S) fuzzy model using a Modified Particle Swarm Optimization (MPSO) with random inertia weight algorithm is presented. In this scheme, an inertia weight of PSO is randomly adjusted during searching process to overcome the difficulties of the method of selecting inertia weight and to improve the performance of the standard PSO. The performance of the self-tuning PID controller based on MPSO has been tested on academic nonlinear system.*

Keywords: Nonlinear System, Takagi-Sugeno, PID Controller, Modified Particle Swarm Optimization

Received: 14 March 2013, Revised 18 April 2013, Accepted 23 April 2013

1. Introduction

During the past decades, the process control techniques in the industry have made great advances. Numerous control methods such as adaptive control,neural control, and fuzzy control have been studied [1] [2]. Among them, proportional- Integral-Derivative (PID) controllers have been widely used for speed and position control of various applications. Among the conventional PID tuning methods, the ZieglerNichols method [3] may be the most well known technique, but, In general, it is often hard to determine optimal or near optimal PID parameters with the Ziegler-Nichols formula in many industrial plants [4] [5]. For these reasons, People have made lots of research, and proposed some advanced PID control methods, such as expert PID control based on knowledge inference [6], self-learning PID control based on regulation, neural network PID control based on connection mechanism [7].

However, it is not easy for tuning PID gains and it is difficult for the fixed-gain PID controller to compensate for such characteristic changes and nonlinearity. to overcome these problems, the self tuning of PID gains using soft intelligent computation such as GA, NN and PSO has been proposed [8] [9]. Recently, PSO is attractively used for self tuning PID controller because it has superior properties such as easy implementation, simple algorithm, fast and stable convergence and efficient in time calculation.The PSO algorithm was an evolution computation technology based on population intelligent methods. it gives a better result than the previous methods in term of convergence speed and solution accuracy.

Although PSO has superior properties, they have some inherent problems such as premature convergence. possibility fall and

local optima. and inefficiency or low accuracy for multiple peak problems and dynamic environment. In order to overcome those problems. many researchers have attempted to improve the PSO algorithm [11] [12]. Each objective function is fundamentally the same except for the section of code that defines the specific error performance criterion being implemented to optimize the performance of a PID controlled system. In this work, a modified PSO which employs random inertia weight was applied to self tune the PID gains for nonlinear system. To show the effectiveness of our proposed method, the simulation in on academic nonlinear system was compared with that of the previous methods (PID-PSO).

The rest of the paper is organized as follows. In section 2, a brief review of the T-S fuzzy model formulation is given. In section 3, describes a PID controller design by the proposed MPSO algorithm is described . Some simulation results is shown in Section 4. Finally, some conclusions are made in section 5.

2. T-S Fuzzy Model of Nonlinear System

We consider a class of nonlinear systems defined by:

$$y(k+1) = f(x(k)) \tag{1}$$

With the regression vector represented by:

$$x(k) = [\, y(k), y(k-1),..., y(k-n), u(k), u(k-1)\, , ..., u(k-n)] \tag{2}$$

Here, k denotes the discrete time, and n define the number of delayed output. Through this contribution, the unknown function $f(x(k))$ is approximated by a T-S fuzzy model which is charities by rule consequents that are linear function of the input variables [13]. The rule base comprises r rules of the form:

$$\begin{aligned} \boldsymbol{R}^i &: if\ x_1 is\ A_1^i\ and\ if\ x_s\ is\ \boldsymbol{A}_s^i\ \ \textbf{then} \\ y(k+1) &= a_{i1}y(k) + ...+ a_{in}y(k-n) + b_{i1}u(k) + ...+ b_{in}u(k-n) \end{aligned} \tag{3}$$

Where $\boldsymbol{R}^i$ denotes the i^{th} fuzzy inference rule, r is the number of inference rules, $A_j^{\,i}(j = 1...s)$ are fuzzy sets, $u(k)$ is the system input variable, $y(k)$ is the system output, $a_{i1},...,a_{in}$, b_{i1}, ..., b_{in} are coefficient of the i^{th} subsystem and $x(k) = [x_i,..., x_s]$ are some measurable system variables.

Let $\mu_i(x(k))$ be the normalized membership function of the inferred fuzzy set $\boldsymbol{A}^i$, where $\boldsymbol{A}^i = \prod_{j=1}^{s} \boldsymbol{A}_j^{\,i}$ and $\sum_{j=1}^{r} \mu_i = 1$. The output of T-S fuzzy model is computed:

$$y(k) = \sum_{j=1}^{r} \mu_i [a_{i1}y(k) +...+a_{in}y(jk-n) + b_{i1}u(k) +...+b_{in}u(k-n)] \tag{4}$$

For nonlinear systems the online adaptation is necessary to obtain a model able to continue the system in its evolution. The system described by relation (4) can also be rewritten as:

$$y(k) = \theta^t \Phi(k-1) \tag{5}$$

This is a regression form, with θ being a system parameter vector and a regression vector. It should be noted that the system (5) is in general nonlinear but it is linear with respect to its unknown parameter vectors. Based on parameterization (5), the identification algorithm giving estimates $\widehat{\theta}(k)$ of $\theta(k)$ can be obtained using the normalized least-squares algorithm [14]. We define:

$$\begin{aligned} \varphi_i(k-1) = [&\mu_i\, y(k-1)...\mu_i y(k-n) \\ &\mu_i u(k-1)...\mu_i u(k-n)\ \mu_i] \end{aligned} \tag{6}$$

$$\theta_i = [a_{i1}... a_{in}\, b_{i1}... b_{in} c_i] \tag{7}$$

$$\varphi(k-1) = [\varphi_1^t(k-1)\ \varphi_2^t(k-1)...\varphi_r^t(k-1)]^t \tag{8}$$

The system described by (4) can also be rewritten as:

$$P_i(k) = P_i(k-1) - \frac{P_i(k-1) - \varphi_i \varphi_i^t(k)\, P_i(k-1)}{1 + \varphi_i^t(k)\, P_i(k-1)\, \varphi_i(k-1)} \tag{9}$$

3. PID Controller

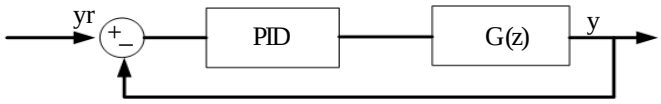

Figure 1. A common feedback control system

The feedback control system is illustrated in Figure 1 where *yr*, *y* are respectively the reference and controlled variables. The PID controller is used to improve the dynamic response as well as to reduce or eliminate the steady-state error. The derivative controller adds a finite zero to the open-loop plant transfer function and improves the transient response[15]. The integral controller adds a pole at the origin, thus increasing system type by one and reducing the steady-state error due to a step function to zero. The PID controller transfer function is

$$\boldsymbol{C}(z) = kp + ki\,\frac{z}{z-1} + kd\,\frac{z-1}{z} \tag{10}$$

where *kp*, *ki* and *kd* are respectively the proportional, integral and derivative gains parameters of the PID controllers.

2.1 Particle Swarm Optimization

Particle Swarm Optimization, first introduced by Kennedy and Eberhart, is one of optimization algorithms. It was developed through simulation of simplified social system, and has been found to be robust in solving continuous nonlinear optimization problems [16]. The PSO technique can generate a high quality solution within shorter calculation time and stable convergence characteristic than other stochastic methods. PSO is a population based search process where individuals, referred to as particles, are grouped into a swarm. Each particle in swarm represents a candidate solution to the optimization problem. In PSO technique, each particle is flown through the multidimensional search space, adjusting its position in search space according to its own experience and that of neighboring particles. A particle therefore makes use of best position encountered by itself and that of its neighbors to position itself toward an optimal solution. The effect is that particles fly toward a minimum, while still searching a wide area around the best solution. The performance of each particle (i.e., the closeness of a particle to a global optimum) is measured using a predefined fitness function, which encapsulates the characteristics of the optimization problem. As example,the ith particle is represented as $\boldsymbol{X}_i = (X_{i,1}, X_{i,2},..., X_{i,d})$ in the dimensional space. The best previous position of the j^{th} particle is recorded and represented as $\boldsymbol{Pbest}_i = (pbest_{i,1}, pbest_{i,2},...,pbest_{i,d})$. The index of best particle among all particles in the group is represented by the gbest. The rate of the position change (velocity) for particle j is represented as $V_i = (V_{i,1}, V_{i,2},..., V_{i,d})$. The modified velocity and position of each particle can be calculated using the current velocity and distance from $\boldsymbol{pbest}_{i,d}$ to $\boldsymbol{gbest}_{gd}$ as shown in the following formulas:

$$V_{id}(k+1) = w\boldsymbol{V}_{id}(k) + r_1 * c_1\,(\boldsymbol{pbest}_{id}(k) - \boldsymbol{X}_{id}(k)) + r_2 * c_2\,(\boldsymbol{gbest}_{gd}(k) - \boldsymbol{X}_{id}(k)) \tag{11}$$

$$X_{id}(k+1) = X_{id}(k) + V_{id}(k+1) \tag{12}$$

Where: $\boldsymbol{pbest}_i$ is *pbest* of particle *i*, $\boldsymbol{gbest}_g$ is *gbest* of the group, r_1, r_2 are two random numbers in the interval [0,1] and c_1, c_2 are positive constants.

w is the inertia weight, is a parameter used to control the impact of the previous velocities on the current velocity. It influences the tradeoff between the global and local exploitation abilities of the particles. Weight is updated as $\omega = \left(\frac{\omega_{max} - \omega_{min}}{iter_{max}}\right) iter$ where ω_{min}, ω_{max} t and $iter_{max}$ are minimum, maximum values of ω, the current iteration number, and pre-specified maximum number of iteration cycles, respectively.

2.2 Modified Particle Swarm Optimization (MPSO)

The experiments indicate that inertia weight is most important parameter to balance the global search ability (exploration) and local search ability (exploitation), when inertia weight is higher, the global search ability is strong, but the local search ability is low; whereas the local search ability is strong, but global search ability is low. This balancing is a key role to improve the performance of PSO. However, the method of selecting inertia weight is not easy and need to be further investigated. The linear and nonlinear decreasing inertia weight can make PSO adjust global search ability and local search ability, but it has shortcomings: firstly, due to the lack of local search ability at beginning of iteration and global search ability at the end of iteration, there is a possibility to get stuck in local optima; secondly, improper selected initial inertia weight (w_{max}), final inertia weight (w_{min}) and nonlinear index number can decrease the performance of PSO [10,11]. Moreover, major experiments show that particles can accumulate at point in searching area, but it is not a global optimum solution. The particles have stagnated and lost the ability to find the global optima solution. Especially, in dynamic environment, so the particles will often fall in local optima. To overcome those problems, the inertia weight employing a random number uniformly distributed in [0,1] was introduced to improve the performance of PSO. In this method, global search ability and local search ability can be processed in the same time. Thus, the proposed method is capable of escaping from the local optima. During iteration, the value of inertia weight is randomly varying according to the following equation:

$$w = \alpha_1 + \alpha_2.rand\,() \tag{13}$$

where *rand* () is a random number in [0, 1]; 1 is the lower limit of w; α_2 is the upper limit of rand ().

2.3 Implementation of a MPSO-PID Controller

A PID controller using the MPSO algorithm was developed to improve the step transient response of nonlinear system. The MPSO algorithm was mainly utilized to determine three optimal controller parameters, and such that the controlled system could obtain a good step response output. For our case of design, we had to tune all the three parameters of PID such that it gives the best output results or in other words we have to optimize all the parameters of the PID for best results. Here we define a three dimensional search space in which all the three dimensions represent three different parameters of the PID. Each particular point in the search space represent a particular combination of (*kp ki kd*) for which a particular response is obtained The performance of the point or the combination of PID parameters is determined by a fitness function or the cost function. This fitness function consists of several component functions which are the performance index of the design. The point in the search space is the best point for which the fitness function attains an optimal value. In this paper, We can also rewrite as controller design attempts to minimize the system error produced by certain anticipated inputs [17]. The system error is defined as the difference between the desired response of the system and its actual response. Performance criteria are mainly based on measures of the system error. The error signal *e* (*k*) will be entered for MPSO and subsequently evaluated in the fitness function to guide the particles during the optimization process. The fitness function for the proposed method is given as:

$$Fitness = \frac{1}{1 + e\,(k)} \tag{14}$$

Fitness shows the following-up of evaluation function for the object input. The purpose is to decrease the steady-state error by maximizing the function.

The MPSO-PID controller for searching the optimal controller parameters, kp, ki and kd with the MPSO algorithm. Each individual *K* contains three members *kp*, *ki* and *kd*. Its dimension is $n * 3$. The searching procedures of the proposed MPSO-PID controller were shown as below:

Step 1: Specify the lower and upper bounds of the three controller parameters and initialize randomly the individuals of the population including searching points, velocities, *pbest* and *gbest*.

Step 2: Evaluate the objective criterion in Equation (14).

Step 3: Compare the individual fitness of each particle to its previous *gbest*. If the fitness is better, update the fitness as *gbest*.

Step 4: Modify the velocity ***V*** of each individual *K* according to (11).

Step 5:

- if $v_{id}^{k} \succ V_d^{max}$, then $v_d^{k+1} = V_d^{max}$

• if $v_{id}^{k+1} + V_d^{min}$, then $v_d^{k+1} = V_d^{min}$

Step 6: Modify the member position of each individual K according to (12). Such that $K_d^{min} \preceq K_{id}^{k+1} \preceq K_d^{max}$.

Step 7: If the number of iterations reaches the maximum, then go to Step 8. Otherwise, go to Step 2.

Step 8: The individual that generates the latest is an optimal controller parameter.

The Figure 2 shows the basic strategy of a PID-MPSO algorithm.

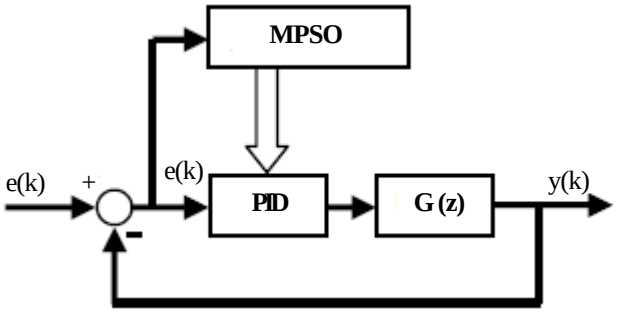

Figure 2. Block diagram of a PID-MPSO algorithm

4. Simulation Results

This section presents a simulation example to show an application of the proposed control algorithm and its satisfactory performance. The nonlinear system is written as the following recursive form:

$$y(k) = a_1'\sin(y(k-1)) + a_2' y(k-2) + a_3' u(k-3)\, y(k-1) + b_1' u(k-1) + b_2'(\tanh(0.7u(k-3)^2)) \qquad (15)$$

With $a_1' = 0.4$, $a_2' = 0.3$, $a_3' = 0.1$, $b_1' = 0.6$ and $b_2' = 1.8$. The input signal applied to plant (15) is a finite sequence of uniformly distributed random variables with range [–2, 2]. The consequent parameters of each rule T-S fuzzy model are computed from equation (5) and adapted by using *RLS* algorithm with for getting factor $\lambda = 0.9$, $\alpha_1 = 0.3$, and $\alpha_2 = 0.3$. The desired set-points $r(k)$ are switched between 2 and –2 every 200 iterations. The results of the comparison are shown in Figure 3, Figure 4 and Table 1. Figure 3, presented the system response for two method. In Figure 4, the blue solid denotes the fitness of the T-S fuzzy model based on MPSO and the red line denote the fitness of the T-S fuzzy models based on PSO. TABLE 1 shows the Rise Time, Setting time and OV of the two T-S fuzzy models. Figure 3, Figure 4 and Table 1 show that the performance of the T-S fuzzy model based on MPSO is better than those based on the PSO in terms of Rise time, Setting time and Overshoot.

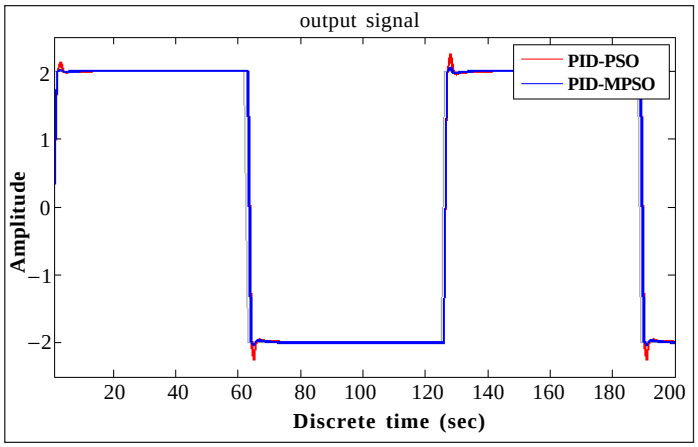

Figure 3. System Response

5. Conclusions

In this paper, a modified PSO with random inertia weight for self tuning PID for Takagi-Sugeno fuzzy model is proposed. This

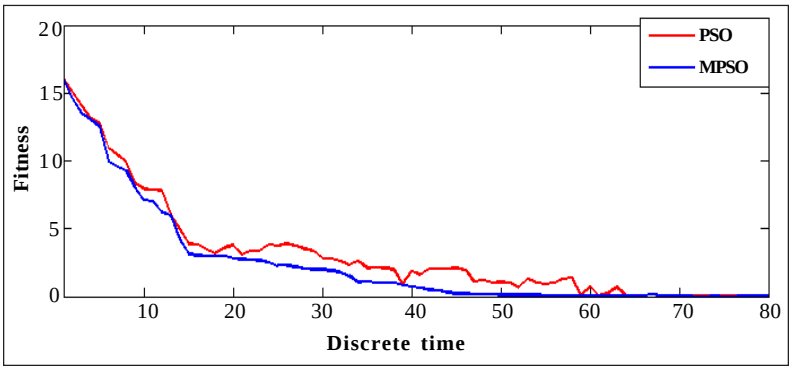

Figure 4. Objective function

Tuning Method	PSO	MPSO
Swarm size	100	100
kp	4.3	5.2
ki	5:4	3.2
kd	4:12	3.7
Overshoot	64:8174	56.5896
Rise time	0:1352	0.1318
Settling time	3:2941	1.7573

Table 1. Performance Estimation of Pid Controller

strategy is the simplest way to adjust the value of inertia weight. Through the simulation, the results show that the proposed controller can perform an efficient search for the optimal PID controller parameters.

References

[1] Visioli, A. (2008). Tuning of PID controllers with fuzzy logic, proc. Inst. Elec, proc. Inst. Elec. Eng. contr. theory application 148 (1) 1-8, Jan.

[2] Kawabe, T., Tagami, T. (1997). A real coded genetic algorithm for matrix inequality design approach of robust PID controller with two degrees of freedom, *In*: Proc. 12th IEEE Int. Symp. Intell. Contr, Istanbul, Turkey, July, p. 119-124.

[3] Ogata, K. (1987). Modern Control Systems, University of Minnesota, Prentice Hall.

[4] Visioli, A. (2001). Tuning of PID controllers with fuzzy logic,Proc. *Inst. Elect. Eng. Contr. Theory Application*, 148 (1) 1-8, Jan.

[5] Krohling, R. A., Rey, J. P. (2001). Design of optimal disturbance rejection PID controllers using genetic algorithm, *IEEE Trans. Evol, Comput*, 5, p. 78-82, Feb.

[6] Conradie, A., Miikkulainen, R., Aldrich, C. (2002). Adaptive Control Utilizing Neural Swarming, *In*: Proceedings of the Genetic and Evolutionary Computation Conferences, USA.

[7] Hossein Shayeghi, Heidar Ali Shayanfar and Aref Jalili, Multi Stage Fuzzy PID Load Frequency Controller in a Restructured Power System, *Journal of Electrical Engineering*, 58 (2) 61-70.

[8] Wong-Young, H., Wook Han, J., Chan-goolee. (1999). Development of a Self- Tuning PID controller based on Neural Network for Nonlinear system, *In*: Proceding of the 7th Mediterranean conference on control and automation (MED99), Haifa, Israel jeune p. 28-30.

[9] Kennedy, J., Eberhart, C. (1959). Particule Swarm Optimisation, *In*: Proceeding IEEE International Conference on Neural Networks, p. 1942-1945.

[10] Nickabadi, A., Ebadzadeh, M., Safabakhch, R. (2011). A novel Partivle Swarm Optimisation Algorithm with adaptive inertia weight, *Journal of applied Soft Computing*, 11, p. 3658-3670.

[11] Faridah, A., Rahman, Murata, Y., Tanaka, K., Nishimara, Y., Uchikado, S., Osa, Y. (2009). Variable gain type PID control using PSO for Ultrasonic Motor, 5th International Workshop on computational *Intelligence And application IEEE SMC Horishima Chapter*, Horishima University, November.

[12] Shi, Y., Eberhart, R. C. (1998). A modified Particle Swarm Optimisation, *IEEE INT. Conf. Evol. Comput, Anchorage*, AK, May, p. 4-9, .

[13] Lagrat, I., Ouakka, H., Boumhidi, I. (2007). Adaptive control of a class of nonlinear systems based on Takagi-Sugeno fuzzy model, ICTIS 07, Fez, Morocco, p. 3-5 April.

[14] Praly, L. (1983). Robustness of indirect adaptive control based on pole placement design. *In*: Proc, Of the 1st IFAC Workshop on adaptive systems in control and signal processing, San Francisco, USA.

[15] Iwan Solihin, M., Fook Tack, L., Leap Kean, M. (2011). Tuning of PID Controller Using Particle Swarm Optimization (PSO), School of Engineering, UCSI University, 56000, Jalan Menara Gading, UCSI Heights, (1) 14-15, January.

[16] Qin, S. J., Badgwell, T. A. (2003). A survey of industrial model predictive control technology, *Control Engineering Practic*e, (11) 733- 764.

[17] Morkos, S., Kamal, H. (2012). Optimal Tuning of PID Controller using Adaptive Hybrid Particle Swarm Optimization Algorithm, *Int. J. of Computers, Communications and Control*, ISSN p. 1841-9836.