Hindawi Publishing Corporation
ISRN Artificial Intelligence
Volume 2013, Article ID 543607, 11 pages
http://dx.doi.org/10.1155/2013/543607

Research Article

PSO-Based PID Controller Design for a Class of Stable and Unstable Systems

K. Latha,[1] V. Rajinikanth,[2] and P. M. Surekha[2]

[1] *Department of Instrumentation Engineering, Anna University, M.I.T Campus, Chennai 600 044, India*
[2] *Department of Electronics and Instrumentation Engineering, St. Joseph's College of Engineering, Chennai 600 119, India*

Correspondence should be addressed to V. Rajinikanth; rajinisjceeie@gmail.com

Received 31 March 2013; Accepted 27 April 2013

Academic Editors: K. W. Chau and J. M. Molina López

Nonlinear processes are very common in process industries, and designing a stabilizing controller is always preferred to maximize the production rate. In this paper, tuning of PID controller for a class of time delayed stable and unstable process models using Particle Swarm Optimization (PSO) algorithm is discussed. The dimension of the search space is only three (K_p, K_i, and K_d); hence, a fixed weight is assigned for the inertia parameter. A comparative study is presented between various inertia weights such as 0.5, 0.75, and 1. From the result, it is evident that the proposed method helps to attain better controller settings with reduced iteration number. The efficacy of the proposed scheme has been validated through a comparative study with classical controller tuning methods and heuristic methods such as Genetic Algorithm (GA) and Ant Colony Optimization (ACO). Finally, a real-time implementation of the proposed method is carried on a nonlinear spherical tank system. From the simulation and real-time results, it is evident that the PSO algorithm performs well on the stable and unstable process models considered in this work. The PSO tuned controller offers enhanced process characteristics such as better time domain specifications, smooth reference tracking, supply disturbance rejection, and error minimization.

1. Introduction

In process industries, many important real-time processing units such as Continuous Stirred Tank Reactor (CSTR), biochemical reactor, and spherical tank system are highly nonlinear in nature. Tuning of controllers to stabilize these nonlinear chemical process loops and impart adequate disturbance rejection is critical because of their complex nature. Based on the operating regions, most of the chemical loops exhibit stable and/or unstable steady states.

Controller tuning is an essential preliminary procedure in almost all industrial process control systems. In control literature, a number of controller structures are available to stabilize stable, unstable, and nonlinear processes [1–5]. Designing controller for process with stable operating region is quite simple. For unstable systems, there exist minimum and maximum values of controller gain, and the average of this limiting value is considered to design the controller to stabilize the system. The increase in time delay in the process narrows down the limiting value and restricts the performance of closed loop system under control. In addition, these systems show unusual overshoot or inverse response due to the presence of positive zeros [4].

Despite the significant developments in advanced process control schemes such as predictive control, internal model control, and sliding mode control, PID controllers are still widely used in industrial control application because of their structural simplicity, reputation, robust performance, and easy implementation [2]. Many researchers proposed PID tuning rules to control various stable and unstable systems by different schemes to enhance closed loop performance [1]. For stable systems, PID controller offers a viable result for both reference tracking and disturbance rejection operations. However, for unstable systems, it can effectively work either for reference tracking or disturbance rejection. The proportional and derivative kick in the controller also results in large overshoot and large settling time [3].

Conventional controller tuning methods proposed by most of the researchers are model dependent, and they require a reduced order models such as first-order or

Table 1: Optimized PID parameters for Example 1.

W	Best values	Iteration number	K_p	K_i	K_d
0.5	1	58	0.6749	0.6525	0.0228
	2	35	0.6800	0.6415	0.0170
	3	41	0.6904	0.6965	0.0332
0.75	1	45	0.6976	0.6597	0.1136
	2	43	0.8579	0.6827	0.0781
	3	52	0.5779	0.6413	0.0812
1	1	44	0.4885	0.839	0.1013
	2	39	0.6781	0.9102	0.2954
	3	63	1.0119	0.7781	0.1377

Table 2: Quantitative analysis for Example 1.

W	Best values	M_p	t_r (s)	t_s (s)	ISE	IAE
0.5	1	0	5.5	5.5	9.395	3.065
	2	0	6.5	6.5	9.72	3.118
	3	0.004	3.9	8.5	8.245	2.871
0.75	1	0.0055	4.3	6.6	9.191	3.030
	2	0	8.1	8.1	8.582	2.929
	3	0.0185	4.0	9.8	9.726	3.119
1	1	0.13	2.6	9.5	5.682	2.384
	2	0.112	2.6	10	4.828	2.197
	3	0	6.5	6.5	6.606	2.570

second-order process model with time delay. Particularly for unstable systems, the tuning rule proposed for a particular reduced order model will not offer a fitting response for other models (higher order models, model with a positive or negative zero, model with a large delay time to process time constant ratio, etc.). Most of the classical PID tuning methods require numerical computations in order to get the best possible controller parameters. Due to these reasons, in recent years, heuristic algorithm-based controller tuning has greatly attracted the researchers.

From recent literature, it is observed that heuristic algorithm-based optimization procedures have emerged as a powerful tool for finding the solutions for variety of control engineering problems [6–12]. Heuristic algorithms are widely used in process control because of their structural simplicity, better optimization ability, and speed of response. Heuristic algorithms can effectively work for higher dimensional optimization problems compared to the existing classical optimization procedures. Due to their flexibility, they can easily adapt to the existing classical controller design procedures. They can be used as a vital tool to design classical and modified structured controllers for a class of unstable process models, irrespective of its model order. Most recent heuristic methods such as Genetic Algorithm (GA) [11], Ant Colony Optimization (ACO) [13], Bacterial Foraging Optimization (BFO) [14], and Particle Swarm Optimization (PSO) [12, 15] are extensively addressed by the researchers to tune controllers for a class of process models.

In this paper, PID controller parameter tuning is attempted using the PSO algorithm introduced by Kennedy and Eberhart [16]. The main advantage of PSO algorithm is that it is an autotuning method; it does not require detailed mathematical description of the process and finds the best possible K_p, K_i, and K_d based on the Objective Function (OF) provided to guide the algorithm. A detailed study is presented with a PSO algorithm using various inertia weights such as 0.5, 0.75, and 1. A comparative study is also carried out between GA, ACO, BFO, and classical controller tuning methods existing in the literature.

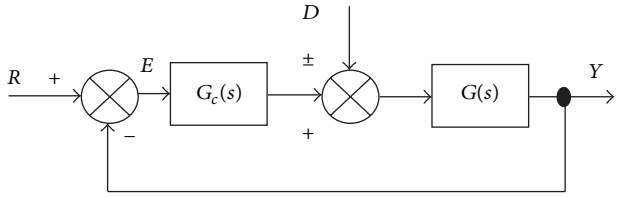

Figure 1: Block diagram of closed loop control system.

The remaining part of the paper is organized as follows. Section 2 presents the outline of classical PID and setpoint filter-based PID controller. A brief description of PSO- and OF-based controller tuning is provided in Section 3. PSO based PID controller design is discussed in Section 4. Section 5 presents the simulated results on different process models and a real-time implementation using a spherical tank system. Section 6 provides conclusion of the present research work.

2. Controller Structure

2.1. PID Controller. PID controller has a simple structure and is usually available as a packaged form. To perform well with the industrial process problems, the controller should have optimally tuned K_p, K_i, and K_d values. Figure 1 shows the basic block diagram of the closed loop systemm where $G_c(s)$ is responsible to maintain Y based on R, by eliminating the redundant input disturbance (D) [3].

The closed loop response of the above system can be expressed as

$$Y(s) = \left[\frac{G(s)\,G_c(s)}{1 + G(s)\,G_c(s)}\right] R(s) \pm \left[\frac{G(s)}{1 + G(s)\,G_c(s)}\right] D(s). \tag{1}$$

Let us assume that $G(s) = K \cdot (\text{num}(s)/\text{den}(s))$, $R(s) = A/s$, and

$$G_c(s) = K_p\left(1 + \frac{1}{\tau_i s} + \tau_d s\right) = \left[K_p + \frac{K_i}{s} + K_d s\right] = \left[\frac{K_i + K_p s + K_d s^2}{s}\right], \tag{2}$$

where $\tau_i = K_p/K_i$, $\tau_d = K_d/K_p$.

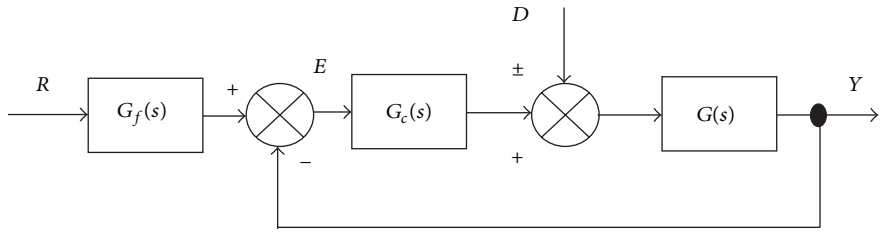

Figure 2: Block diagram of PID controller with prefilter.

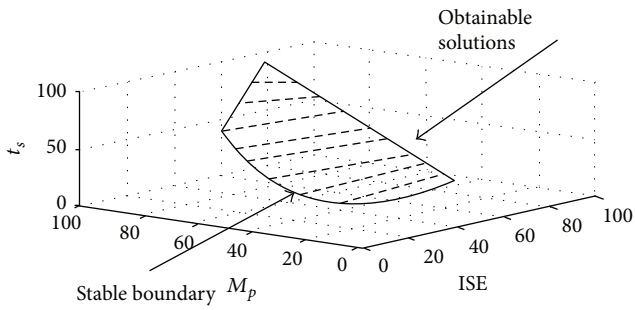

Figure 3: Obtainable solutions from search universe U.

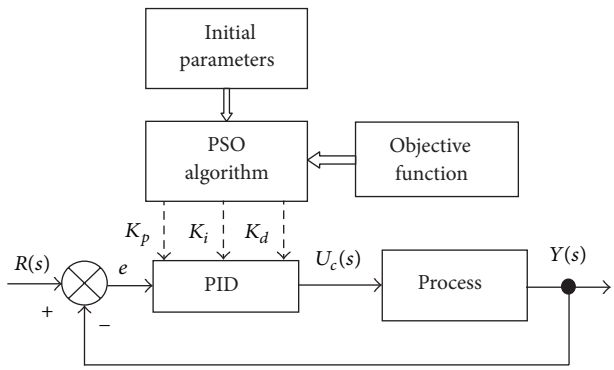

Figure 4: PSO-based PID controller design.

The final steady-state response $Y(\infty)$ of the system $G(s)$ for reference tracking and supply disturbance rejection is presented correspondingly as follows:

$$Y_R(\infty) = \lim_{t\to\infty} sY_R(s) = \lim_{t\to\infty} sx\left[\frac{G(s)G_c(s)}{1+G(s)G_c(s)}\right]\left(\frac{A}{s}\right) = A, \quad (3)$$

$$Y_D(\infty) = \lim_{t\to\infty} sx\left[\frac{G(s)}{1+G_p(s)G_c(s)}\right]\left(\frac{D}{s}\right) = 0, \quad (4)$$

where A is the amplitude of reference signal and D is the amplitude of disturbance.

2.2. PID Controller with Prefilter. Figure 2 depicts the structure of prefilter/setpoint filter-based PID controller discussed by Araki and Taguchi [17], where $G_f(s)$ is a first-order prefilter with the following structure $1/(1+\tau_f s)$.

Jung et al. reported that, when the filter time constant τ_f is set equal to integral time constant τ_i, the controller offers a smooth reference tracking performance [18]. Recently, Vijayan and Panda developed a detailed analytical expression

Table 3: PID parameters for Example 2.

Sl. Number	Method	K_p	K_i	K_d
1	ZN	2.808	1.712	1.151
2	GA	2.391	2.391	1.458
3	ACO	2.4911	0.8158	1.3540
4	MACO	2.808	1.021	1.668
5	PSO (0.75)	1.452	0.4259	0.5084

Table 4: Performance measure for Example 2.

Sl. Number	Method	% M_p	t_r (s)	t_s (s)	ISE
1	ZN	31.59%	0.664	4.78	0.854
2	GA	5.84%	0.676	3.63	0.797
3	ACO	4.95%	0.701	5.90	0.809
4	MACO	2.84%	0.700	3.00	0.772
5	PSO (0.75)	0.00%	2.713	9.836	4.108

Table 5: Controller values for various W.

Sl. Number	Method	K_p	K_i	K_d	ISE
1	ZN	−1.6722	−1.8580	−0.3762	1.782
2	GA	−1.2440	−1.3980	−0.2427	3.148
3	MOPSO	−0.5612	−2.0712	0.0133	1.434
4	PSO (0.75)	−0.6313	−1.7103	0.0159	1.603

Table 6: Controller settings for Example 4.

W	Best values	Iteration number	K_p	K_i	K_d
0.5	1	77	−0.5697	−0.0803	0.0199
	2	83	−0.6030	−0.0918	0.0099
	3	59	−0.5774	−0.1091	−0.0288
0.75	1	63	−0.7027	−0.1968	0.0443
	2	76	−0.6412	−0.1633	0.1990
	3	47	−0.7076	−0.2288	0.1669
1	1	62	−0.6962	−0.1432	−0.4182
	2	58	−0.6405	−0.1569	−0.4680
	3	83	−0.8869	−0.1671	−0.5309

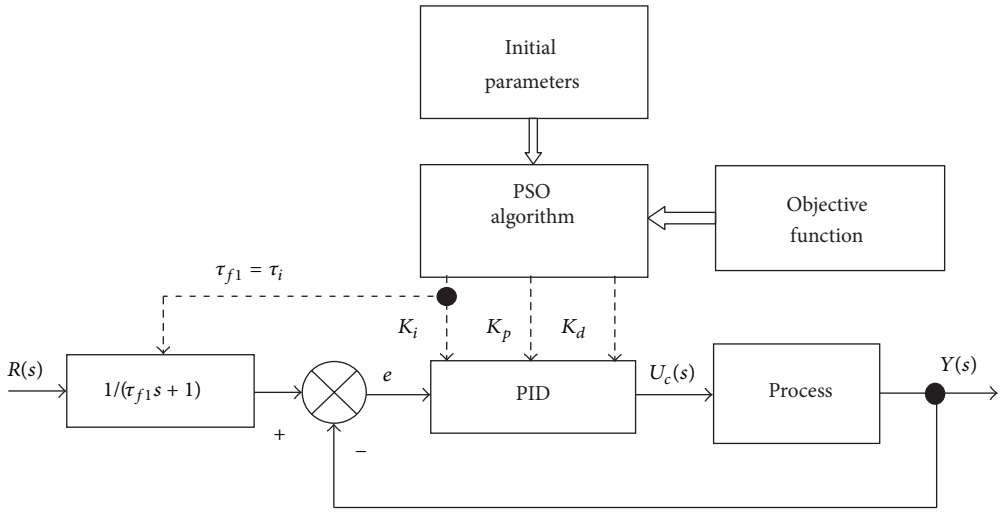

Figure 5: PSO-based PID controller with prefilter design.

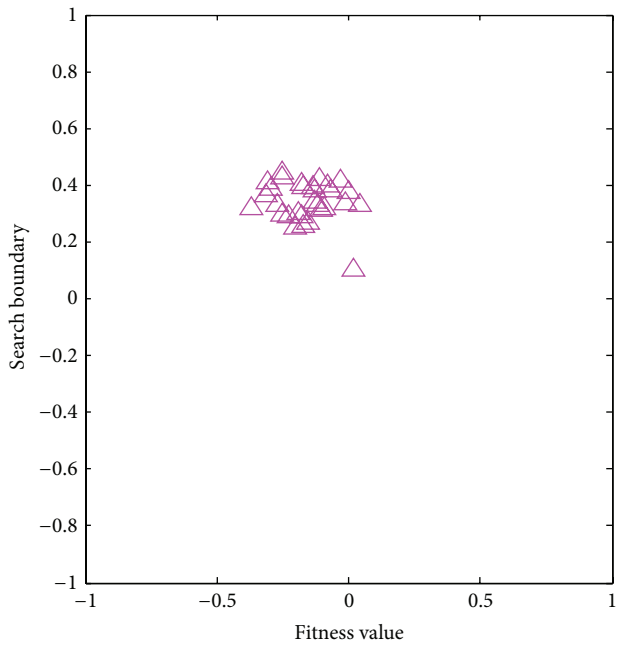

Figure 6: Convergence of particles with optimal solution.

to assign the above controller parameters and validated using a class of process models [19].

3. Methodology

3.1. PSO Algorithm. Particle Swarm Optimization (PSO) technique, proposed by Kennedy and Eberhart [16], is an evolutionary-type global optimization technique developed due to the inspiration of social activities in flock of birds and school of fish and is widely applied in various engineering problems due to its high computational efficiency [6–10, 20–23]. Compared with other population-based stochastic optimization methods, such as GA and ACO, PSO has comparable or even superior search performance for many hard optimization problems, with faster and more stable convergence rates. It has been proved to be an effective optimum tool in system identification and PID controller tuning for a class of processes [24].

In PSO algorithm, the number of parameters to be assigned is very few compared to other nature-inspired algorithms. In this, a group of artificial birds is initialized with arbitrary positions X_i and velocities V_i. At early searching stage, each bird in the swarm is scattered randomly throughout the D dimensional search space. With the supervision of the Objective Function (OF), own flying experience, and their companions flying experience, each particle in the swarm dynamically adjusts its flying position and velocity. During the optimization search, each particle remembers its best position attained so far (i.e., *pbest*—$(P^t_{i,D})$) and also obtains the global best position information achieved by any particle in the population (i.e., *gbest*—$(G^t_{i,D})$).

At iteration t, each particle i has its position defined by $X^t_{i,n} = [X_{i,1}, X_{i,2}, \ldots, X_{i,D}]$ and velocity defined as $V^t_{i,n} = [V_{i,1}, V_{i,2}, \ldots, V_{i,D}]$ in search space D. Velocity and position of each particle in the next iteration can be calculated as

$$V^{t+1}_{i,D} = W * V^t_{i,D} + C_1 * R_1 * \left(P^t_{i,D} - X^t_{i,D}\right) + C_2 * R_2 * \left(G^t_{i,D} - X^t_{i,D}\right), \quad (5)$$

$$X^{t+1}_{i,D} = X^t_{i,D} + V^{t+1}_{i,D}, \quad (6)$$

where $i = 1, 2, \ldots, N$; and acceleration constant C_1 called cognitive parameter pulls each particle towards local best position, and constant C_2 called social parameter pulls the particle towards global best position. R_1 and R_2 are known as random numbers in the range 0-1.

In (5), the inertia of weight W represented is an important factor for the PSO's convergence. It is used to control the impact of previous velocities on the current velocity at the current time step. The basic PSO, initially proposed by Kennedy and Eberhart, has no inertia weight, and this feature is first introduced by Shi and Eberhart [25]. The

work reports that a large inertia weight factor facilitates global exploration while small weight factor facilitates local exploration.

Further, for high-dimensional problems, dynamical adjusting of inertia weight was introduced by many researchers which can increase the search capabilities of PSO. A review of inertia weight strategies in PSO is given chronologically in subsequent paragraphs.

Eberhart and Shi [26] proposed a Random Inertia Weight strategy and experimentally found that this strategy increases the convergence of PSO in early iterations of the algorithm. The linearly decreasing strategy enhances the efficiency and performance of PSO. In spite of its ability to converge optimum, it gets into the local optimum solving the question of more apices function. In global-local best inertia weight, the inertia weight is based on the function of local best and global best of the particles in each generation. It neither takes a constant value nor a linearly decreasing time-varying value. To overcome the weakness of premature convergence to local minimum, adaptive inertia weight strategy [27] is proposed to improve its searching capability. Chen et al. [28] present two natural exponent inertia weight strategies which are based on the basic idea of decreasing inertia weight. Malik et al. [29] presented a Sigmoid Increasing Inertia Weight (SIIW). They found that sigmoid function has contributed in getting minimum fitness function while linearly increasing inertia weight gives contribution to quick convergence ability. So they combine sigmoid function and linear increasing inertia weight and provide an SIIW which has produced a great improvement in quick convergence ability and aggressive movement narrowing towards the solution region. Oscillating inertia weight provides periodically alternates between global and local search waves, and the conclusion was drawn that this strategy appears to be generally competitive and, in some cases, PSO outperforms particularly in terms of convergence speed. Gao et al. [30] proposed a new PSO algorithm which combined the logarithm decreasing inertia weight with chaos mutation operator. The logarithm decreasing inertia weight can improve the convergence speed, while the chaos mutation can enhance the ability to jump out of the local optima. In order to overcome the premature convergence and later period oscillatory occurrences of the standard PSO, an exponent decreasing inertia weight and stochastic mutation to produce an improved PSO has been proposed by Gao et al. [30], which uses the exponent decreasing inertia weight along with stochastic piecewise mutation for current global optimal particle during the running time, to support better optimization search.

In this paper, we considered constant inertia weight-based velocity updation. The implementation of PSO has the following steps.

Step 1 (initialization of swarm). For a population size N, the particles are randomly generated between the minimum and maximum limits of parameter values.

Step 2 (evaluation of objective function). Objective function values of particles are evaluated using the performance criteria for algorithm convergence.

Step 3 (initialization of *pbest* and *gbest*). The objective values obtained above for the initial particles of swarm are set as the initial *pbest* values of particles. The best value among all the *pbest* values is identified as *gbest.*

Step 4 (evaluation of velocity). The new velocity for each particle is computed using (5).

Step 5 (update the swarm). The particle position is updated using (6). The values of the objective function are calculated for updated positions of particles. If the new value is better than the previous *pbest*, the new value is set to *pbest*. Similarly, *gbest* value is also updated as the best *pbest*.

Step 6 (stopping criteria). If the stopping criteria are met, positions of particles represented by *pbest* are the optimal values. Otherwise, the above said procedure is repeated from Step 4 until the specified iteration is completed.

3.2. Objective Function. The overall performance (speed of convergence, efficiency, and optimization accuracy) of PSO algorithm depends on Objective Function (OF), which monitors the optimization search. The OF is chosen to maximize the domain constrains or to minimize the preference constrains. During the search, without loss of generality, the constrained optimization problem minimizes a scalar function *"J"* of some decision variable vector *"D"* in a universe *"U."* The objective function is to be framed by assuming, at least, that there exists one set of optimal parameters in "U" which satisfies all the constraints.

The minimization problem of preference constrains can be mathematically expressed as

$$\min_{D\in U} J(D). \tag{7}$$

The majority of controller design problems are multiobjective in nature. Multiobjective optimization always provides improved result compared to single-objective function. Without loss of generality, the multiobjective optimization problem simultaneously minimizes n objectives $\phi_i(p)$, $i = 1, \ldots, n$ of a variable vector D in a universe U:

$$\min_{D\in U} (\phi_1(p), \phi_2(p), \ldots, \phi_n(p)). \tag{8}$$

Weighted sum method is widely adopted by most of the researchers, and it converts the multiobjective problem of minimizing the objectives into a scalar form [31]. The general form of weighted sum of multiple objective functions is presented in (7). In this method, the constraints which are to be minimized are arranged as weighted sum:

$$\min_{D\in U} \sum_{k=1}^{n} W_k J_k(D), \tag{9}$$

where W_k is the weighting coefficient, n is the number of constrains to be solved, D is the dimension of the problem, and U is the search universe. Generally, the weight for W_k (between 0 and 1 or 0 and 100%) is assigned based on the preference constraints to be minimized.

In the control literature, there exist a number of weighted-sum-based objective functions [31]. In this paper, we considered three-parameter-based objective function by considering the error and important time domain constraints such as overshoot M_p and settling time t_s:

$$J(\theta) = w_1 \cdot \mathrm{ISE} + w_2 \cdot M_p + w_3 \cdot t_s,$$
$$\theta = \left[K_p K_i K_d\right], \tag{10}$$

where θ are parameters to be optimized, M_p is peak overshoot, t_s is settling time, and the weighting function $w_1 = w_2 = 1$ and $w_3 = 0.5$.

In many optimization cases, it is very difficult to satisfy all the considered constraints. Hence, there should be some negotiation between the preference constraint parameters without compromising the domain constraint [31]. In this work, we assigned more preference for ISE and M_p compared to t_s. Figure 3 depicts the obtainable optimal solution and stability boundary in the three-dimensional search space.

Search boundary for controller parameters is assigned as follows:

$$K_p : \min -60\% \text{ to } \max +60\%,$$
$$K_i : \min -30\% \text{ to } \max +30\%, \tag{11}$$
$$K_d : \min -50\% \text{ to } \max +50\%.$$

The algorithm continuously varies the values of controller parameters until the objective function J is minimised to $J_{\min}$.

4. Controller Tuning

The controller design process is to find the optimal values for controller parameters form the search space that minimizes the considered objective function. Figure 4 illustrates the basic block diagram of PSO algorithm-based PID controller tuning. Initially, PSO algorithm assigns arbitrary values for K_p, K_i, and K_d and computes the OF. This procedure continues until the J reaches $J_{\min}$ or the final iteration number is reached.

Figure 5 shows the block diagram of PSO tuned PID controller with a prefilter setup. For this procedure, dimension of the search is three, and the algorithm finds the best possible controller parameters along with filter time constant τ_f. This setup provides better result for both the stable and unstable process models.

PSO-based controller design procedure is developed with number of swarms $N = 20$, swarm step size = 20, $C_1 = C_2 = 2$, and maximum generation value of 200. Optimal tuning procedure is repeated 10 times independently, and the best value among the trials is considered to stabilize the process.

5. Results and Discussion

Example 1. The first-order stable process with the following transfer function model is considered [19]:

$$G(s) = \frac{1}{s+1} e^{-0.5s}. \tag{12}$$

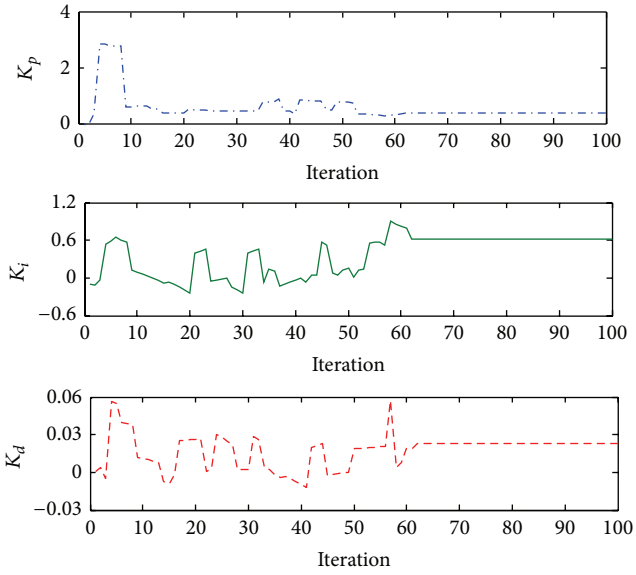

Figure 7: Optimal controller parameters.

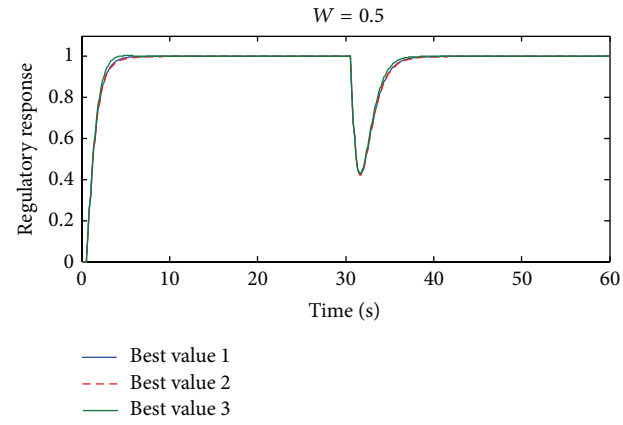

Figure 8: Regulatory response for $W = 0.5$.

PSO-based PID tuning is proposed with the method as shown in Figure 4. The final convergence of swarm is depicted in Figure 6. The multiple OF-based controller design procedure is converged at 58th iteration (for $W = 0.5$) as shown in Figure 7, and the final PID parameters with various W values and the corresponding iteration number are tabulated in Table 1.

Table 1 presents three best values among 10 trials. These values are individually evaluated using the process model (12). Figure 8 represents the reference tracking and input disturbance rejection performance of PID controller obtained when $W = 0.5$. Figure 9 shows the corresponding controller performance.

Similar responses are obtained for the above process model for $W = 0.75$ and $W = 1$, and the obtained performance measures are tabulated in Table 2.

From Table 2 and Figure 10, it is observed that, even though the total iteration taken by the best value 3 (for $W = 1$) is considerably large, it provides better overall performance compared to other values considered in this study.

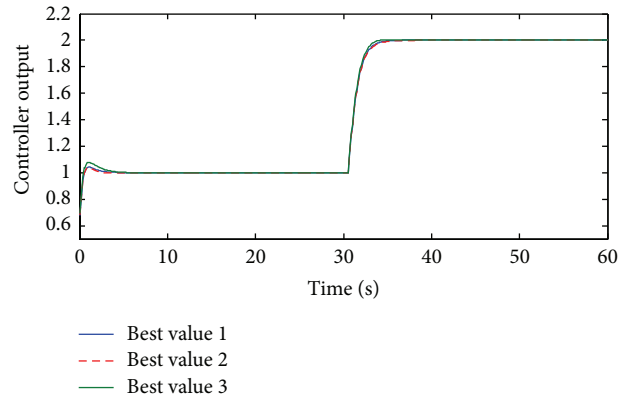

Figure 9: Controller output for $W = 0.5$.

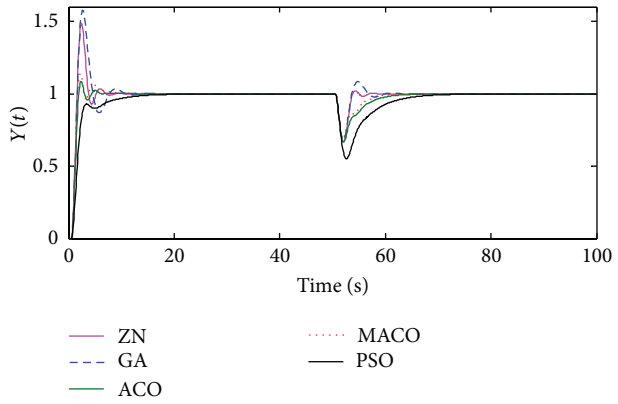

Figure 11: Regulatory response for $W = 0.5$.

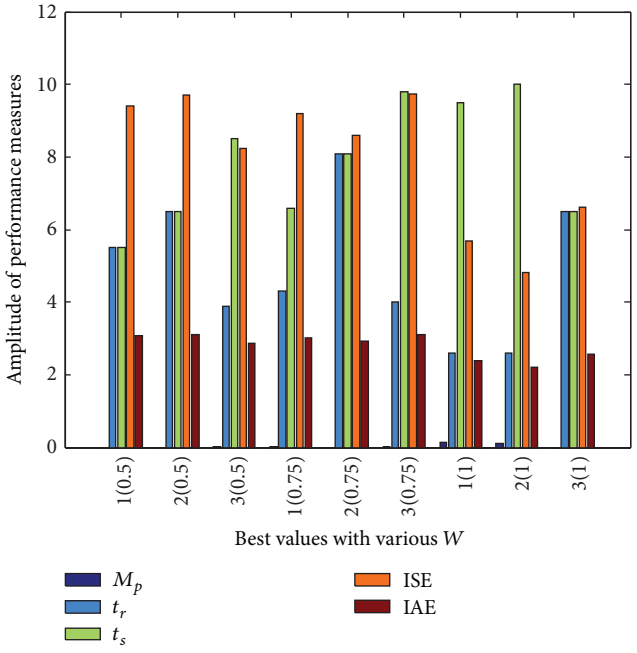

Figure 10: Process performance with various W.

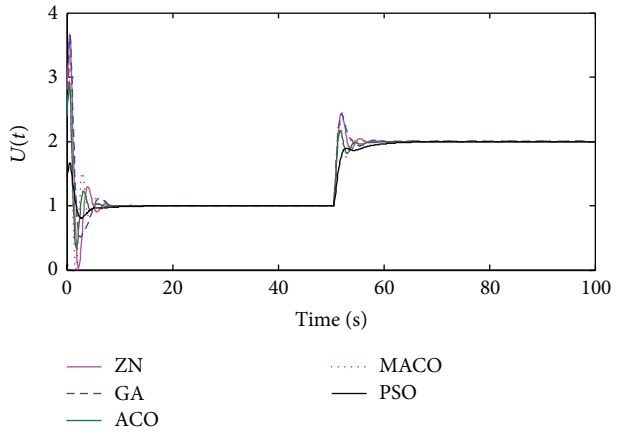

Figure 12: Controller output for $W = 0.5$.

Example 2. The stable second-order model discussed by Chiha et al. is considered [13];

$$G(s) = \frac{1}{(s+1)^2} e^{-0.1s}. \tag{13}$$

Chiha et al. proposed a Multiobjective Ant Colony Optimization (MACO) algorithm and validated the result with Ziegler-Nichols (ZN), GA, and ACO tuned PID controller. In this work, we considered the PSO tuned PID controller, and the controller values and its performance measure values are presented in Tables 3 and 4 respectively.

Figure 11 shows the process output for Example 2 for various controller settings. Initially a unity reference input signal is applied to the system. An unity input disturbance signal is applied at 50 s to study the disturbance rejection performance. The performance by the present method is smooth compared to other methods considered in this study. The ZN, GA, ACO, and MACO methods show oscillations during the reference tracking and the disturbance rejection cases compared to the PSO tuned PID. But the response by the present controller is sluggish than other methods. From Figure 12, it is observed that the controller performance is also very smooth. Even though the parameters such as t_r, t_s, and ISE are large, the present controller provides approximately null % overshoot (Table 4).

Example 3. The stable second-order steady-state model of the bioreactor is presented below [11]:

$$G(s) = \frac{-1.53s - 0.4588}{s^2 + 2.564s + 0.6792} e^{-0.1s}. \tag{14}$$

For this model, Kumar et al. [11] proposed a GA-based PID controller. Recently Rajinikanth and Latha proposed and validated the multiple-objective PSO-based PID tuning procedure using nonlinear process model [33].

In this work, PSO-based PID is proposed for the above process model with $W = 0.75$, and the final converged controller parameters are presented in Table 5. Figures 13 and 14 show the process response and controller output, respectively. The proposed methods provide improved reference tracking and disturbance rejection performance compared to the

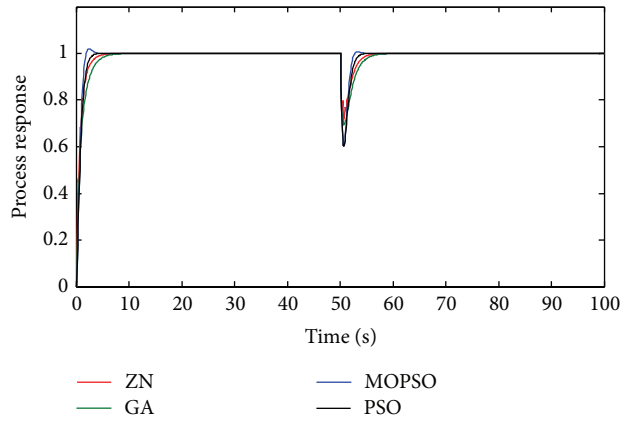

Figure 13: Process response for stable bioreactor model.

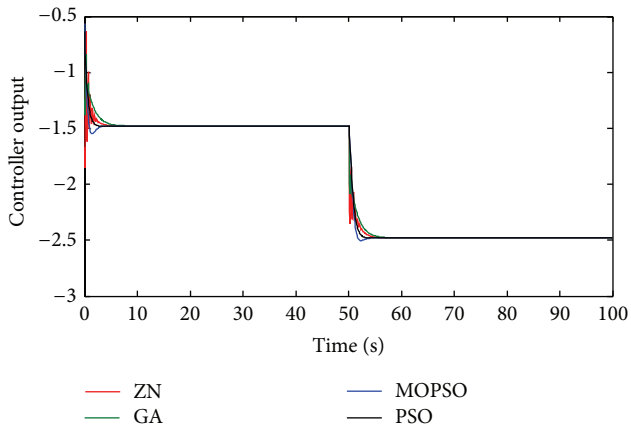

Figure 14: $U(t)$ for bioreactor model.

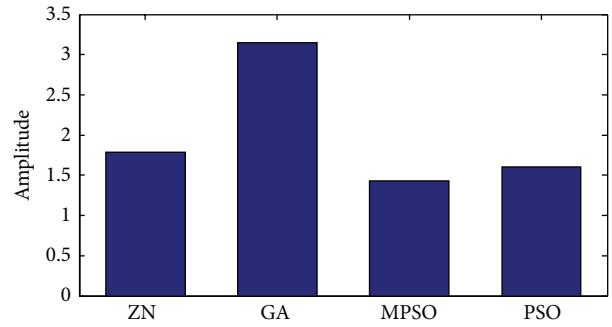

Figure 15: Graphical representation of ISE.

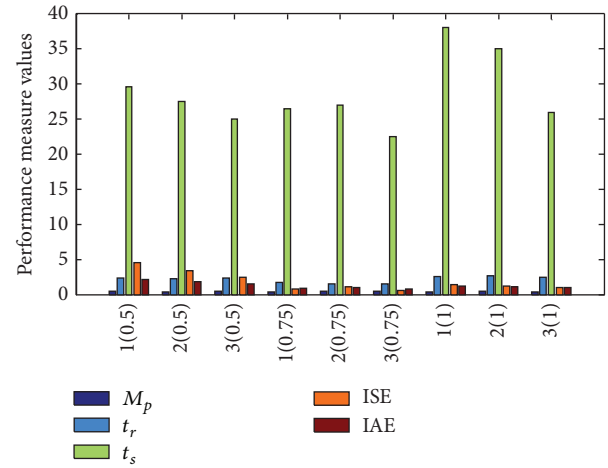

Figure 16: Graphical representation of ISE.

other methods considered. From Figure 15 the observation is that, ISE value by the proposed method is considerably smaller than ZN and GA.

Example 4. The unstable steady-state model of the bioreactor presented below is widely considered by the researchers [14, 15]:

$$G(s) = \frac{-0.9951s - 0.2985}{s^2 + 0.1302s - 0.0509} e^{-0.1s} = \frac{-5.8644}{5.89s - 1} e^{-0.1s}. \quad (15)$$

The above process model is a bench mark problem in the unstable controller design problem. In this paper, initially, the PSO controller is designed for the process later a prefilter-based PID structure is implemented as shown in Figure 5. The previous work by Rajinikanth and Latha provided the following controller parameters: $K_p = -0.5374$, $K_i = -0.0702$, and $K_d = -0.0537$ using the BFO algorithm [14].

The controller setting by the proposed method for various W values is presented in Table 6, and the corresponding performance values are tabulated in Table 7 (best three values among 10 trials). The performance measure values are graphically represented in Figure 16, and the PSO-based PID setting for $W = 0.75$ (third best value) provides improved performance compared to other values. Hence the above-mentioned value is considered for the comparative work with the BFO algorithm (with PID and prefilter-based PID controller).

Table 7: Performance measure values for Example 4.

W	Best values	M_p	t_r (s)	t_s (s)	ISE	IAE
0.5	1	0.434	2.3	29.6	4.497	2.121
	2	0415	2.25	27.5	3.446	1.856
	3	0.454	2.3	25	2.447	1.564
0.75	1	0.409	1.7	26.5	0.75	0.886
	2	0.435	1.5	27	1.09	1.044
	3	0.423	1.5	22.5	0.557	0.746
1	1	0.39	2.56	38.1	1.419	1.19
	2	0.44	2.7	35	1.178	1.05
	3	0.32	2.5	26	1.04	1.020

In this study, a disturbance of 50% (of setpoint value) is applied at 50 s to analyze the supply disturbance rejection performance. Figure 17 shows the regulatory response for the unstable bioreactor model, and Figure 18 shows the corresponding controller output. Even though the reference

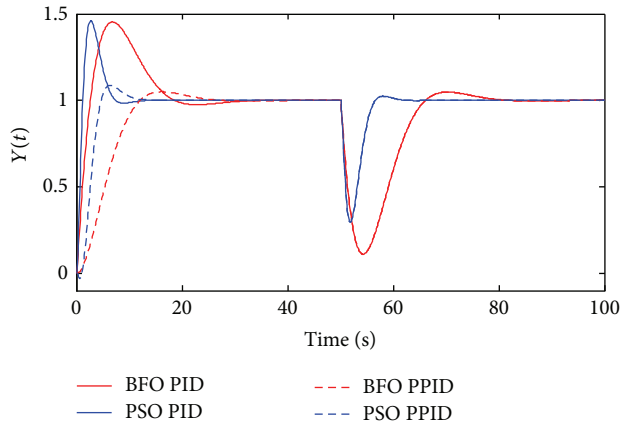

Figure 17: Output response with PID and PID with prefilter (PPID).

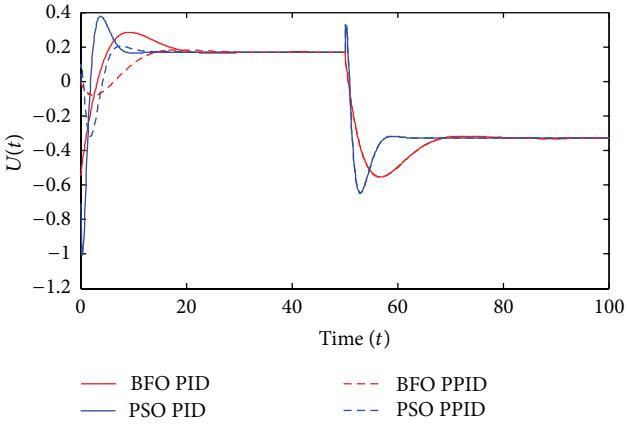

Figure 18: Controller output for PID and PID with prefilter (PPID).

tracking response of the PID and prefilter-based PID (PPID) is different, both show a similar response for the disturbance rejection problem, where the proposed PSO tuned PID and PPID show enhanced performance compared to the existing BFO tuned PID and PPID controller. From this, it is observed that the fixed inertia weight-based PID controller provides best optimized values when the dimension of the problem is small.

Example 5. The nonlinear spherical tank system recently discussed by Rajinikanth and Latha is considered to show the real-time implementation of the proposed method [34]. The transfer function model of the system for an operating range of 18 cm is

$$G(s) = \frac{3.6215}{330.46s + 1} e^{-11.7s}. \quad (16)$$

A simulation time of 500 s is considered in the PSO-based PID tuning procedure, and the algorithm is converged at 41th iteration for $W = 0.75$ with $K_p = 6.8115, K_i = 1.0042$, and $K_d = 1.9016$. The obtained PID values are then transferred to the real-time controller hardware installed in the process loop through the monitoring and control program developed in MATLAB Simulink with ODE 45 solver, which is directly

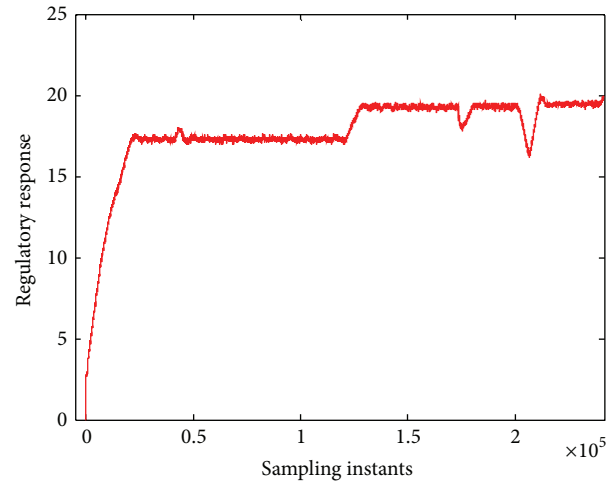

Figure 19: Real-time process response for nonlinear spherical tank system.

interfaced with the real-time process system through DAQ module. It is enabled with National Instruments VISA serial communication interface. The module supports ASCII data format with a sampling time of 0.1 s and a baud rate of 38400. With this, monitoring and control of the real-time process can be easily established with MATLAB software. In real-time implementation, the maximum controller output is set as 90% in order to reduce the thrust on the control valve which allows the flow rate to the tank.

Figure 19 shows the variation of level for the reference tracking, multiple reference tracking, supply disturbance, and load disturbance rejection problems.

Initially a reference input of 18 cm is assigned. When the process output reaches a final steady-state value, then the reference input is increased by 2 cm (i.e., 20 cm) at 1.3×10^5th sampling instant. Later a supply disturbance of 10% of the setpoint and a load disturbance of 25% of the setpoint are applied at 1.75×10^5th sampling instant and 2×10^5th sampling instants, respectively.

In the real-time study, the ISE and IAE values are obtained as 455.81 and 196.33, respectively. From this result, it can be noted that the PSO algorithm presents a smooth servo and regulatory response for the spherical tank level control problem and it can be easily implemented in real time using a MATLAB-supported real-time process loop.

6. Conclusion

This paper attempted a controller parameter optimization work with PSO algorithm with various inertia weights. The fixed inertia weight method helps to improve the speed of convergence and also maintains good accuracy in optimized parameters. Proposed weighted sum of multiple-objective function provides the necessary controller parameters and supports enhanced performance for both the reference tracking and disturbance rejection problems. The real-time implementation also confirms the adaptability of the proposed method on the MATLAB supportable real-time hardware.

The real-time result with PSO tuned PID offers better result for reference tracking, multiple reference tracking, and disturbance rejection problems.

References

[1] A. O'Dwyer, *Handbook of PI and PID Controller Tuning Rules*, Imperial College Press, London, UK, 3rd edition, 2009.

[2] R. C. Panda, "Synthesis of PID controller for unstable and integrating processes," *Chemical Engineering Science*, vol. 64, no. 12, pp. 2807–2816, 2009.

[3] M. A. Johnson and M. H. Moradi, *PID Control: New Identification and Design Methods*, chapter 2, Springer, London, UK, 2005.

[4] R. Padmasree and M. Chidambaram, *Control of Unstable Systems*, Narosa Publishing House, New Delhi, India, 2006.

[5] B. W. Bequette, *Process Control—Modeling, Design and Simulation*, Prentice Hall, New Delhi, India, 2003.

[6] U. S. Banu and G. Uma, "Fuzzy gain scheduled continuous stirred tank reactor with particle swarm optimization based PID control minimizing integral square error," *Instrumentation Science and Technology*, vol. 36, no. 4, pp. 394–409, 2008.

[7] M. S. Arumugam and M. V. C. Rao, "On the performance of the particle swarm optimization algorithm with various inertia weight variants for computing optimal control of a class of hybrid systems," *Discrete Dynamics in Nature and Society*, vol. 2006, Article ID 79295, 17 pages, 2006.

[8] M. S. Arumugam and M. V. C. Rao, "On the optimal control of single-stage hybrid manufacturing systems via novel and different variants of particle swarm optimization algorithm," *Discrete Dynamics in Nature and Society*, vol. 2005, no. 3, pp. 257–279, 2005.

[9] R. M. Chen and C. M. Wang, "Project scheduling heuristics-based standard PSO for task-resource assignment in heterogeneous grid," *Abstract and Applied Analysis*, vol. 2011, Article ID 589862, 20 pages, 2011.

[10] R. F. Abdel-Kader, "Particle swarm optimization for constrained instruction scheduling," *VLSI Design*, vol. 2008, Article ID 930610, 7 pages, 2008.

[11] S. M. G. Kumar, R. Jain, N. Anantharaman, V. Dharmalingam, and K. M. M. S. Begam, "Genetic algorithm based PID controller tuning for a model bioreactor," *Indian Institute of Chemical Engineers*, vol. 50, no. 3, pp. 214–226, 2008.

[12] M. Zamani, M. Karimi-Ghartemani, N. Sadati, and M. Parniani, "Design of a fractional order PID controller for an AVR using particle swarm optimization," *Control Engineering Practice*, vol. 17, no. 12, pp. 1380–1387, 2009.

[13] I. Chiha, N. Liouane, and P. Borne, "Tuning PID controller using multiobjective ant colony optimization," *Applied Computational Intelligence and Soft Computing*, vol. 2012, Article ID 536326, 7 pages, 2012.

[14] V. Rajinikanth and K. Latha, "Bacterial foraging optimization algorithm based pid controller tuning for time delayedunstable systems," *Mediterranean Journal of Measurement and Control*, vol. 7, no. 1, pp. 197–203, 2011.

[15] V. Rajinikanth and K. Latha, "Identification and control of unstable biochemical reactor," *International Journal of Chemical Engineering Applications*, vol. 1, no. 1, pp. 106–111, 2010.

[16] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, 1995.

[17] M. Araki and H. Taguchi, "Two-degree-of-freedom PID controllers," *International Journal of Control, Automation and Systems*, vol. 1, no. 4, pp. 401–411, 2003.

[18] C. S. Jung, H. K. Song, and J. C. Hyun, "Direct synthesis tuning method of unstable first-order-plus-time-delay processes," *Journal of Process Control*, vol. 9, no. 3, pp. 265–269, 1999.

[19] V. Vijayan and R. C. Panda, "Design of a simple setpoint filter for minimizing overshoot for low order processes," *ISA Transactions*, vol. 51, no. 2, pp. 271–276, 2012.

[20] C. H. Lin, J. L. Chen, and Z. L. Gaing, "Combining biometric fractal pattern and particle swarm optimization-based classifier for fingerprint recognition," *Mathematical Problems in Engineering*, vol. 2010, Article ID 328676, 14 pages, 2010.

[21] J. Zhang and K. W. Chau, "Multilayer ensemble pruning via novel multi-sub-swarm particle swarm optimization," *Journal of Universal Computer Science*, vol. 15, no. 4, pp. 840–858, 2009.

[22] K. W. Chau, "Application of a PSO-based neural network in analysis of outcomes of construction claims," *Automation in Construction*, vol. 16, no. 5, pp. 642–646, 2007.

[23] K. Chau, "Predicting construction litigation outcome using particle swarm optimization," in *Innovations in Applied Artificial Intelligence*, vol. 3533 of *Lecture Notes in Computer Science*, pp. 571–578, Springer, New York, NY, USA, 2005.

[24] A. Alfi and H. Modares, "System identification and control using adaptive particle swarm optimization," *Applied Mathematical Modelling*, vol. 35, no. 3, pp. 1210–1221, 2011.

[25] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proceedings of the IEEE World Congress on Computational Intelligence*, pp. 69–73, 1998.

[26] R. C. Eberhart and Y. Shi, "Tracking and optimizing dynamic systems with particle swarms," in *Proceedings of the Congress on Evolutionary Computation*, vol. 1, pp. 94–100, Seoul, Republic of Korea, 2001.

[27] A. Nikabadi and M. Ebadzadeh, "Particle swarm optimization algorithms with adaptive inertia weight: a survey of the state of the art and a novel method," *IEEE Journal of Evolutionary Computation*, 2008.

[28] G. Chen, X. Huang, J. Jia, and Z. Min, "Natural exponential inertia weight strategy in particle swarm optimization," in *Proceedings of the 6th World Congress on Intelligent Control and Automation (WCICA '06)*, pp. 3672–3675, Dalian, China, June 2006.

[29] R. F. Malik, T. A. Rahman, S. Z. M. Hashim, and R. Ngah, "New particle swarm optimizer with sigmoid increasing inertia weight," *International Journal of Computer Science and Security*, vol. 1, no. 2, pp. 35–44, 2007.

[30] Y. Gao, X. An, and J. Liu, "A particle swarm optimization algorithm with logarithm decreasing inertia weight and chaos mutation," in *Proceedings of the International Conference on Computational Intelligence and Security (CIS '08)*, vol. 1, pp. 61–65, 2008.

[31] G. P. Liu, J. B. Yang, and J. F. Whidborne, *Multiobjective Optimization and Control*, Printice Hall, New Delhi, India, 2008.

[32] J. G. Ziegler and N. B. Nichols, "Optimum settlings for automatic controllers," *Transactions of the ASME*, vol. 64, pp. 759–768, 1942.

[33] V. Rajinikanth and K. Latha, "Optimization of PID controller parameters for unstable chemical systems using soft computing technique," *International Review of Chemical Engineering*, vol. 3, no. 3, pp. 350–358, 2011.

[34] V. Rajinikanth and K. Latha, "Controller parameter optimization for nonlinear systems using enhanced bacteria foraging algorithm," *Applied Computational Intelligence and Soft Computing*, vol. 2012, Article ID 214264, 12 pages, 2012.