Multi-Objective Evolutionary Optimization of PID Controller by Chaotic Particle Swarm Optimization

Reza Gholipour, Jalil Addeh, Hamed Mojallali, and Alireza Khosravi

***Abstract*—In this paper, An Intelligent PID Controller has been tuned by minimizing the Integral of Time multiplied Absolute Error (ITAE) and squared control signal (ITAESCS) for a DC motor. The parameters of PID controller, automatically and intelligently are determined by Chaotic Particle Swarm Optimization (CPSO) Algorithm. The experimental results demonstrate that the performance of proposed Intelligent CPSO-PID controller is superior to the conventional Ziegler-Nichols method in terms of settling time, maximum overshoot and ITAESCS.**

***Index Terms*—PID controller, chaotic particle swarm optimization, DC motor.**

I. Introduction

Conventional Proportional-Integral-Derivative (PID) controllers are themost widely used in industry due to their simple control structure, ease of design, and low cost [1]-[3].Unfortunately, it has been quite difficult to tune properly the gains of PID controllers because many industrial plants are often burdened with problems such as high order, time delays, and nonlinearities [4]–[9]. Over the years, several heuristic methods have been proposed for the tuning of PID controllers. The first classical tuning rules proposed by Ziegler and Nichols. In general, it is often difficult to determine optimal or near optimal PID parameters with the Ziegler-Nichols formula in many industrial plants [4]–[6]. For these reasons, it is highly desirable to increase the capabilities of PID controllers by adding new features. Many artificial intelligence (AI) techniques have been employed to improve the controller performances for a wide range of plants while retaining their basic characteristics. AI techniques such as neural network, fuzzy system, and neural-fuzzy logic have been widely applied to proper tuning of PID controller parameters [4], [5].

Evolutionary algorithms with their heuristic and stochastic properties often suffer from getting stuck in local optima. This common characteristic led to the development of evolutionary computation as an increasingly important field. A GA is a stochastic search procedure based on the mechanics of natural selection, genetics and evolution [10]. Since this type of algorithm simultaneously evaluates many points in the search space, it is more likely to find a global solution to a given problem. PSO describes a solution process in which each particle moves through a multidimensional search space [11]. The particle velocity and position are constantly updated according to the best previous performance of the particle or of the particle's neighbors, as well as the best performance of all particles in the entire population. GAs have demonstrated the ability to reach near-optimal solutions for large problems; however, they may require a long processing time to reach a near-optimal solution. Similarly to GAs, Binary Particle Swarm Otimization (BPSO) is also a population-based optimizer. BPSO has a memory, so knowledge of good solutions is retained by all the particles and optimal solutions are found by the swarm particles if they follow the best particle. Unlike GAs, BPSO does not contain any crossover and mutation processes [12]. Hybridization of evolutionary algorithms with local search has been investigated in many studies [13], [14]. Such hybrids are often referred to as memetic algorithms (MA). An MA can be treated as a genetic algorithm coupled with a local search procedure [15]. The shuffled frog leaping algorithm (SFL algorithm) combines the benefits of an MA and the social PSO algorithm. Unlike in MAs and PSO, the population consists of a set of solutions (frogs), which is partitioned into subsets referred to as memeplexes. In the search space, each group performs a local search, and then exchanges information with other groups [16]. Ant-colony optimization algorithms (ACO) were developed by Dorigo et al. Similar to PSO, they evolve not based on genetics but on social behavior. Unlike PSO, the ACO uses ants to find the shortest route between their ant hill and a source of food; ants can deposit pheromone trails whenever they travel as a form of indirect communication [17].

Generating an ideal random sequence is of great importance in the fields of numerical analysis, sampling and heuristic optimization. Recently, a technique which employs chaotic sequences via the chaos approach (chaotic maps) has gained a lot of attention and been widely applied in different areas, such as the chaotic neural network (CNN) [18], chaotic optimization algorithms (COA) [19],[20], nonlinear circuits [21], DNA computing [22], and image processing [23]. All of the above-mentioned methods rely on the same pivotal operation, namely the adoption of a chaotic sequence instead of a random sequence, and thereby improve the results due to the unpredictability of the chaotic sequence [24].

Chaos can be described as a bounded nonlinear system with deterministic dynamic behavior that has ergodic and stochastic properties [25]. In what is called the ''butterfly effect'', small variations of an initial variable will result in huge differences in the solutions after some iteration. Mathematically, chaos is random and unpredictable, yet it also possesses an element of regularity.

PSO shows a promising performance on nonlinear function optimization and has thus received much attention [26].However, the performance of the traditional PSO greatly depends on its parameters, and it often suffers the problem of

Manuscript received October 12, 2012; revised November 14, 2012.

The authors are with the Department of Electrical and Computer Engineering, Babol (Noushirvani) University of Technology, Babol, Iran (e-mail: mojallali@guilan.ac.ir, mojallali@gmail.com)

being trapped in local optima [27], [28]. In order to avoid these disadvantages, the chaotic particle swarm optimization (CPSO) method based on the logistic equation has been proposed [28]. Such an algorithm which is known as Chaotic Particle Swarm Optimization (CPSO) is used in this paper in order to determine the optimal proportional-integral-derivative (PID) controller parameters.

The performance of the closed-loop system can defined in terms of overshoot, settling time and steady state error. In general, the system with fast settling time under no steady-state error and almost zero overshoot is desired. Hence, in this study to provide a desired performance, the Integral of Time multiplied Absolute Error (ITAE) and squared control signal (ITAESCS), is minimized by using CPSO. The merits of the proposed controller are illustrated by considering the Motor DC system.

The rest of the paper is organized as follows. Section 2 describes typical CPSO. Section 3 narrates The Ziegler-Nichols tuning method. In section 4, proposed CPSO-PID controller is described. The results obtained from simulations performed by considering the DC Motor system is discussed in Section 5. Finally, in section 6, the general conclusions are presented.

II. Method

A. Particle Swarm Optimization (PSO)

In original PSO [11], each particle is analogous to an individual ''fish'' in a school of fish. It is a population-based optimization technique, where a population is called a swarm. A swarm consists of N particles moving around in a D-dimensional search space. The position of the *i*th particle can be represented by $x_i = (x_{i1}, x_{i2}, ..., x_{iD})$. The velocity for the *i*th particle can be written as $v_i = (v_{i1}, v_{i2}, ..., v_{iD})$. Each particle coexists and evolves simultaneously based on knowledge shared with neighboring particles; it makes use of its own memory and knowledge gained by the swarm as a whole to find the best solution. The best previously encountered position of the *i*th particle is denoted its individual best position $p_i = (p_{i1}, p_{i2}, ..., p_{iD})$, a value called $pbest_i$. The best value of the all individual $pbest_i$ values is denoted the global best position $g_i = (g_1, g_2, ..., g_D)$ and called $gbest$. The PSO process is initialized with a population of random particles, and the algorithm then executes a search for optimal solutions by continuously updating generations. At each generation, the position and velocity of the *i*th particle are updated by $pbest_i$ and $gbest$ in the swarm. The update equations can be formulated as: r_1 and r_2 are random numbers between (0, 1), and c_1 and c_2 are acceleration constants, which control how far a particle will move in a single generation. Velocities v_{id}^{new} and v_{id}^{old} denote the velocities of the new and old particle, respectively. x_{id}^{old} is the current particle position, and x_{id}^{new} is the new updated particle position. The inertia weight w controls the impact of the previous velocity of a particle on its current one [29]. In general, the inertia weight is decreased linearly from 0.9 to 0.4 throughout the search process to effectively balance the local and global search abilities of the swarm [30]. The equation for the inertia weight w can be written as:

In Eq. (3), $w_{\max}$ is 0.9, $w_{\min}$ is 0.4 and $Iteration_{\max}$ is the maximum number of allowed iterations.

$$v_{id}^{new} = w \times v_{id}^{old} + c_1 \times r_1 \times (pbest_{id} - x_{id}^{old}) + c_2 \times r_2 \times (gbest_d - x_{id}^{old}) \quad (1)$$

$$x_{id}^{new} = x_{id}^{old} + v_{id}^{new} \quad (2)$$

$$w = (w_{\max} - w_{\min}) \times \frac{Iteration_{\max} - Iteration_i}{Iteration_{\max}} + w_{\min} \quad (3)$$

B. Chaotic Particle Swarm Optimization (CPSO)

In the field of engineering, it is well recognized that chaos theory can be applied as a very useful technique in practical application. The chaotic system can be described by a phenomenon, in which a small change in the initial condition will lead to nonlinear change in future behavior, besides that the system exhibits distinct behaviors under different phases, i.e. stable fixed points, periodic oscillations, bifurcations, and ergodicity [31]. Chaos [32] is also a common nonlinear phenomenon with much complexity and is similar to randomness. Chaos is typically highly sensitive to the initial values and thus provides great diversity based on the ergodic property of the chaos phase, which transits every state without repetition in certain ranges. It is generated through a deterministic iteration formula. Due to these characteristics, chaos theory can be applied in optimization.

In PSO, the parameters w, r_1 and r_2 are the key factors affecting the convergence behavior [33],[34]. The inertia weight controls the balance between the global exploration and the local search ability. A large inertia weight favors the global search, while a small inertia weight favors the local search. For this reason, an inertia weight that linearly decreases from 0.9 to 0.4 throughout the search process is usually used [30]. Since logistic maps are frequently used, chaotic behavior maps and chaotic sequences can be quickly generated and easily stored. There is no need for storage of long sequences [35]. In CPSO, sequences generated by the logistic map substitute the random parameters r_1 and r_2 in PSO. The parameters r_1 and r_2 are modified by the logistic map based on the following equation.

$$Cr_{(t+1)} = 4 \times Cr_{(t)} \times (1 - Cr_{(t)}) \quad (4)$$

In Eq.(4), $Cr_{(0)}$ is generated randomly for each independent run, with $Cr_{(0)}$ not being equal to {0, 0.25, 0.5, 0.75, 1}. The velocity update equation for CPSO can be formulated as:

$$v_{id}^{new} = w \times v_{id}^{old} + c_1 \times Cr \times (pbest_{id} - x_{id}^{old}) + c_2 \times (1 - Cr) \times (gbest_d - x_{id}^{old}) \quad (5)$$

In Eq. (5), Cr is a function based on the results of the logistic map with values between 0.0 and 1.0. Fig. 1 shows the chaotic Cr value using a logistic map for100 iterations where $Cr_{(0)} = 0.001$. The pseudo-code of CPSO is shown below [36].

CPSO pseudo-code

```
begin
  Randomly initialize particles swarm
  Randomly generate Cr(0)
  while (number of iterations, or the stopping criterion
  is not met)
    Evaluate fitness of particle swarm
    for n = 1 to number of particles
      Find pbest
      Find gbest
      for d = 1 to number of dimension of particle
        update the Chaotic Cr value by Eq. (4)
        update the position of particles by Eq. (5) and
        Eq. (2)
      next d
    next n
    update the inertia weight value by Eq. (3)
  next generation until stopping criterion
end
```

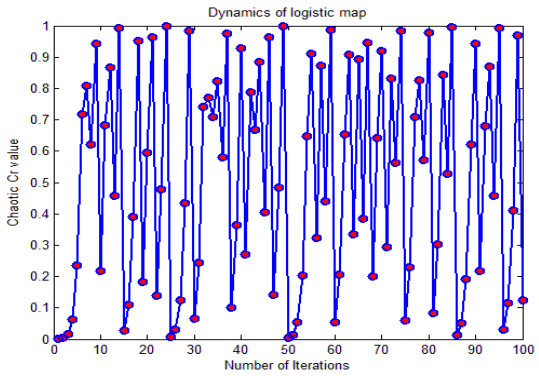

Fig. 1. Chaotic Cr value using a logistic map for 100 iterations; Cr(0)=0.001

In fact, In CPSO, a chaotic map was embedded to determine the PSO parameters r_1 and r_2. The PSO parameters r_1 and r_2 cannot ensure optimal ergodicity in the search space because they are absolutely random [37] i.e. the r_1 and r_2 are generated by a linear congruential generator (LCG) with a random seed. The generated sequence of LCG consists of pseudo-random numbers that have periodic characteristics [38]. Furthermore, the generated sequence of a logistic map also consists of pseudo-random numbers, but there are no fixed points, periodic orbits, or quasi-periodic orbits in the behavior of the chaos system [39]. As a result, the system can avoid being entrapment in local optima [28].

III. Ziegler-Nichols Closed-Loop Tuning Method

The closed-loop tuning method proposed by ZN requires the determination of the ultimate gain and ultimate period. The method can be interpreted as a technique of positioning one point on the Nyquist curve [40], [41]. This can be achieved by adjusting the controller gain (K_c) till the system undergoes sustained oscillations (at the ultimate gain or critical gain), whilst maintaining the integral time constant (T_i) at infinity and the derivative time constant (T_d) at zero.

A significant drawback of this closed-loop tuning method is that the ultimate gain has to be determined through trial and error and the system has to be driven to its stability limits. Another disadvantage is that when the process is unknown, the amplitudes of the undampened oscillations can become excessive when using trial and error to determine the ultimate gain of the system. This could lead to unsafe plant conditions. The closed loop tuning rules for P, PI and PID control are given in Table I.

TABLE I: Ziegler-Nichols Closed Loop Tuning Parameter

Controller	$K_c=K_p$	$T_i=K_p/K_i$	$T_d=K_d/K_p$
P	$0.5K_u$	∞	0
PI	$0.4K_u$	$0.8P_u$	0
PID	$0.6K_u$	$0.5P_u$	$0.125P_u$

IV. Proposed CPSO-PID Controller

The controller output of a conventional PID is a weighted sum of error, its derivative and integral values, i.e.

$$u(t) = k_p e(t) + k_I \int e(t)dt + k_D \frac{d(e(t))}{dt} \quad (6)$$

The simple error minimization criteria can be modified by introducing a suitable time domain performance index like ITAE or Integral of Time multiplied Squared Error (ITSE) to have a better control action. Also for a sudden change in set-point, ITSE based tuning produces a larger controller output than ITAE, hence in the present study only ITAE has been considered as a suitable time domain performance index [42] and not other performance indices having higher powers of error and time. The objective function, used for controller tuning has been taken as a weighted sum of the ITAE and squared control signal similar to that of [43], [44], i.e.

$$ITAESCS = \int_0^{t_f} \left[w_1 t \left| e(t) \right| + w_2 u^2(t) \right] dt \quad (7)$$

where t_f is the final time, in seconds and t is the time, in seconds.The proposed CPSO-PID controller structure is shown in Fig. 2. In CPSO-PID controller, CPSO algorithm is utilized to determine three optimal PID gains, i.e., k_p, k_I, and k_D. Obviously, PID gains optimization is in a 3-dimensional searching space.

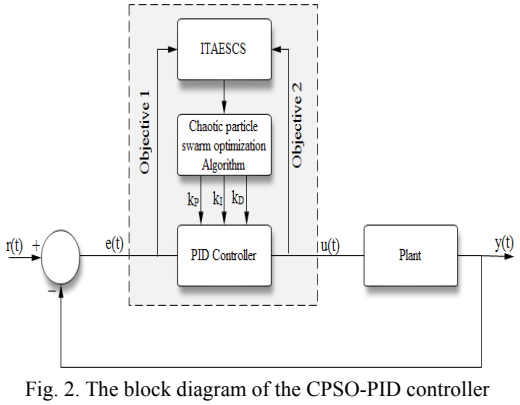

Fig. 2. The block diagram of the CPSO-PID controller

It is worth mentioning that the weights w_1 and w_2 have been introduced in the objective function (7) with a provision of balancing the impact of the error and control signal. The objective function ITAESCS in (7) is now minimized to find out the optimal set of controller parameters which simultaneously reduces the ITAE and control signal $u(t)$. The time multiplication term in error index ITAE minimizes the chance of oscillation at later stages, thus effectively reducing the settling time (t_s) of the closed loop system and the absolute value of error minimizes the percentage of overshoot ($\%M_p$). The minimization of the squared control signal reduces the chance of actuator saturation and also reduces the size of the actuator and thus the cost involved.

V. Simulation Results

In this study, a system simulation was carried out on the combination of the chaotic particle swarm optimization algorithm and the PID control system shown in Fig. 2. The input variable of the proposed CPSO-PID controller is the error $e(t)$ and the output variable is the control signal $u(t)$. On the one hand, to achieve the goal of minimizing the ITAESCS of the control system, we used the CPSO algorithm. The parameters of the CPSO Algorithm are set as shown in Table II. The sampling time in this simulation is 0.01 sec. In this section, in order to track the step input, the weights w_1 and w_2 of fitness function are chosen as 0.9998 and 0.0002, respectively.

In order to compare the performance of the proposed method with the Ziegler-Nichols technique, a DC Motor is considered with the plant model described by:

$$G(s)=\frac{1}{s^3+9s^2+23s+15} \tag{8}$$

The objective of this experiment is to compare the CPSO Algorithm tuning to that of the method of Ziegler-Nichols for DC Motor.For comparison, the following controllers are individually applied and simulated:(I) the PID controller with the parameters $k_p=115.2$, $k_I=177.2308$, and $k_D=18.72$, which are determined by means of the ZN(ZN-PID), and (II) the CPSO-PID controller with the parameters $k_p=57.3112$, $k_I=24.4258$, $k_D=15.8230$ which are determined by means of the CPSO Algorithm.The searching ranges for the PID parameters k_p, k_I, and k_D are limited to [0, 100].

TABLE II: Parameters Used in the CPSO

Population size	20
Acceleration constant c_1	2
Acceleration constant c_2	2
Inertia weight w	started from 0.9 and decreased linearly to 0.4
Number of iterations	30

A sample of the trajectory of the PID parameters and performance criterion during optimization is shown in Figs. 3-5 and 6, respectively. The PID parameters areobtained for 30 iterations. In this example t_f is equal to 10 seconds.

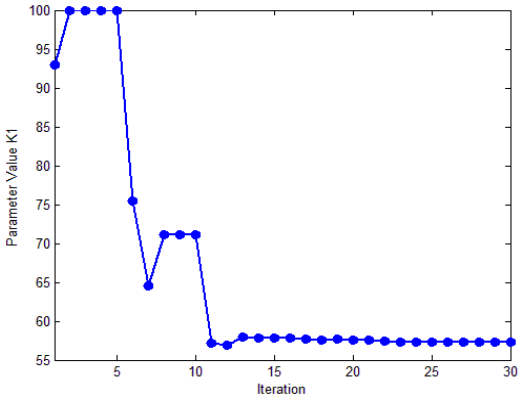

Fig. 3. The parameter value trajectory K1

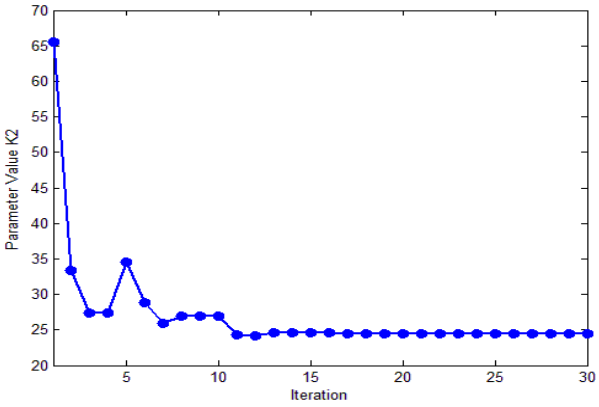

Fig. 4. The parameter value trajectory K2

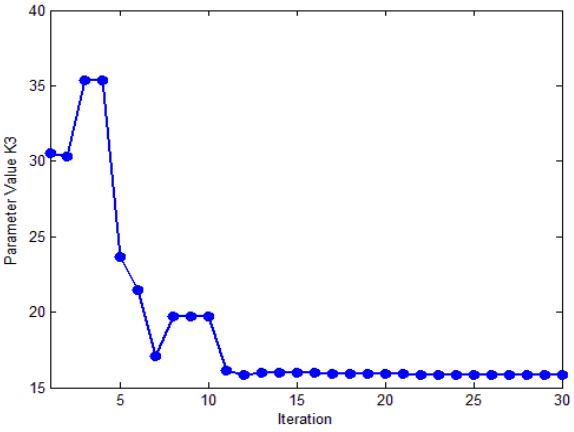

Fig. 5. The parameter value trajectory K3

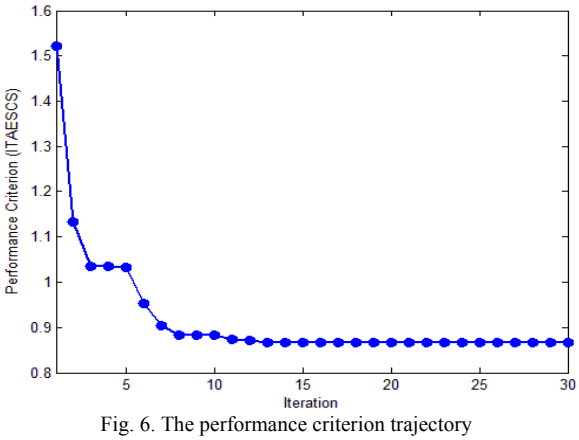

Fig. 6. The performance criterion trajectory

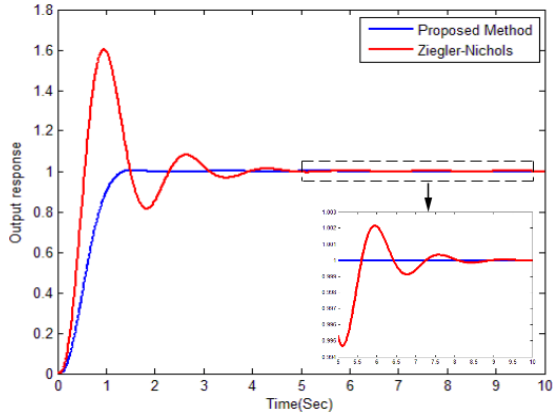

Fig. 7. The step responses of the DC Motor

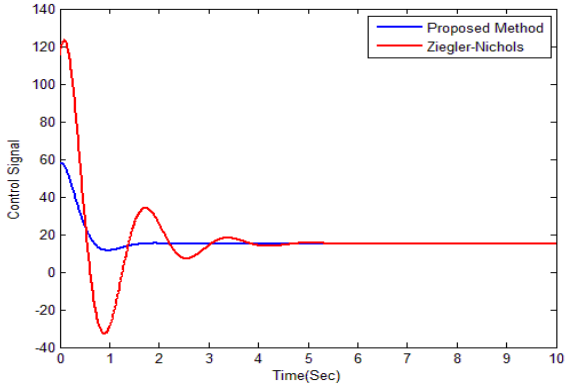

Fig. 8. The control signal

TABLE III: PERFORMANCE COMPARISON WITH STEP RESPONSE

Performance	Settling time (Sec) (5%)	Maximum overshoot (%)	ITAESCS
Ziegler-Nichols Method	2.89	60.3	1.5680
Proposed Method	**1.12**	**0.5**	**0.8657**

The step responses of the controllers mentioned above are simulated and the simulated responses are plotted in Fig. 7. The performance indices (settling time, maximum overshoot, and ITAESCS) of the simulated results are evaluated and these parameters are compared in Table III. It is shown that the proposed controller exhibits better performance as compared to Ziegler-Nichols method. In addition, according to Fig. 8, the proposed controller has created a limited control signal to set-point tracking in DC motor, because the control effort is appliedin the objective function (ITAESCS).

VI. CONCLUSIONS

This paper presents a novel approach to determining the parameters of an intelligent PID controller using the chaotic particle swarm optimization Algorithm. Without trial and error or experiences of designers, the PID gains are automatically optimized by using a chaotic particle swarm optimization Algorithm with a defined fitness function that is associated with the Integral of Time multiplied Absolute Error (ITAE) and squared control signal (ITAESCS) .Illustrative example has been presented to demonstrate that the performance of proposed controller is superior to the Ziegler-Nichols controller in terms of settling time, maximum overshoot, and ITAESCS.

REFERENCES

[1] K. Ogata, *Modern Control Engineering*, second ed., Prentice-Hall, India, 1992.
[2] A. Pollard, *Process Control, Heinemann Educational Books*, London, 1971.
[3] N. Saad and V. Kadirkamanathan, "A DES approach for the contextual load modelling of supply chain system for instability analysis," *Simulation Modelling Practice and Theory*, vol. 14, pp. 541–563, 2006.
[4] A. Visioli, "Tuning of PID controllers with fuzzy logic," in Proc. of *Inst. Elect. Eng. Contr. Theory Applicat*, vol. 148, no. 1, pp. 1–8, Jan. 2001.
[5] T. L. Seng, M. B. Khalid, and R. Yusof, "Tuning of a neuro-fuzzy controller by genetic algorithm," *IEEE Trans. Syst*, Man, Cybern. B, vol. 29, pp. 226–236, 1999.
[6] R. A. Krohling and J. P. Rey, "Design of optimal disturbance rejection PID controllers using genetic algorithm," *IEEE Trans. Evol. Comput*, vol. 5, pp. 78–82, 2001.
[7] Y. Mitsukura, T. Yamamoto, and M. Kaneda, "A design of self-tuning PID controllers using a genetic algorithm," in *Proc. of Amer. Contr. Conf*, San Diego, CA, June 1999, pp. 1361–1365.
[8] T. Kawabe and T. Tagami, "A real coded genetic algorithm for matrix inequality design approach of robust PID controller with two degrees of freedom," in *Proc. of 12th IEEE Int. Symp. Intell. Contr*, Istanbul, Turkey, July 1997, pp. 119–124.
[9] R. A. Krohling, H. Jaschek, and J. P. Rey, "Designing PI/PID controller for a motion control system based on genetic algorithm," in *Proc. of 12th IEEE Int. Symp. Intell. Contr*, Istanbul, Turkey, July 1997, pp. 125–130.
[10] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, MI, 1975.
[11] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proc. of IEEE International Conference on Neural Networks*, Perth, Australia,1995,pp. 1942–1948.
[12] E. Elbeltagi, T. Hegazy, and D. Grierson, "Comparison among five evolutionary-based optimization algorithms," Advanced Engineering Informatics,vol. 19, pp. 43–53, 2005.
[13] Y. T. Kao and E. Zahara, "A hybrid genetic algorithm and particle swarm optimization for multimodal functions," *Applied Soft Computing*, vol. 8, pp. 849–857, 2008.
[14] C. F. Juang, "A hybrid of genetic algorithm and particle swarm optimization for recurrent network design," *IEEE Transactions on Systems, Man, and Cybernetics*, Part B, vol. 34, pp. 997–1006, 2004.
[15] K. Sorensen and M. Sevaux, "MAPM: memetic algorithms with population management," *Computers and Operations Research*, vol. 33, pp. 1214–1225, 2006.
[16] M. M. Eusuff and K. E. Lansey, "Optimization of water distribution network design using the shuffled frog leaping algorithm," *Journal of Water Resources Plan Manage*, vol. 129, pp. 210–225, 2003.
[17] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 26, pp. 29–41, 2002.
[18] K. Aihara, T. Takabe, and M. Toyoda, "Chaotic neural networks," Physics Letters A, vol. 144, pp. 333–340, 1990.
[19] B. Li, and W.S. Jiang, "Optimizing complex functions by chaos search," *Cybernetics and Systems*, vol. 29, pp. 409–419, 1998.
[20] Z. Lu, L.S. Shieh, and G. R. Chen, "On robust control of uncertain chaotic systems: a sliding-mode synthesis via chaotic optimization," *Chaos, Solitons and Fractals*, vol. 18, pp. 819–827, 2003.
[21] P. Arena, R. Caponetto, L. Fortuna, A. Rizzo, and M. L. Rosa, "Self organization in non-recurrent complex system," *International Journal of Bifurcation and Chaos*, vol. 10, pp. 1115–1125, 2000.
[22] G. Manganaro and J. P. D. Gyvez, "DNA computing based on chaos," *Evolutionary Computation*, pp. 255–260, 2002.
[23] H. Gao, Y. Zhang, S. Liang, and D. Li, "A new chaotic algorithm for image encryption," *Chaos, Solitons and Fractals*, vol. 29, pp. 393–399, 2006.
[24] B. Alatas, E. Akin, and A. B. Ozer, "Chaos embedded particle swarm optimization algorithms," *Chaos, Solitons and Fractals*, vol. 40, pp. 1715–1734, 2009.
[25] H. G. Schuster, *Deterministic chaos an introduction*, Second revised ed., Physick-Verlag Gmn H, Weinheim, Federal Republic of Germany, 1988.
[26] Y. Liu, Z. Qin, Z. Shi, and J. Lu, "Center particle swarm optimization," *Neurocomputing*, vol. 70, pp. 672–679, 2007.
[27] P. J. Angeline, "Evolutionary optimization versus particle swarm optimization: philosophy and performance differences," *Evolutionary programming*, vol. VII, pp. 601–10, 1998.

[28] B. Liu, L. Wang, Y. H. Jin, F. Tang, and D. X. Huang, "Improved particle swarm optimization combined with chaos," *Chaos, Solitons and Fractals*, vol. 25, pp. 1261– 1271, 2005.
[29] Y. Shi and R. C. Eberhart, "A modified particle swarm optimizer," in *Proceedings of IEEE International Conference on Evolutionary Computation*, Anchorage, AK, 2002, pp. 69–73.
[30] Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," in *Proc. of Congress on Evolutionary Computation*, Washington, DC, 2002, pp. 1945-1949.
[31] L. D. S. Coelho and B. M. Herrera, "Fuzzy identification based on a chaotic particle swarm optimization approach applied to a nonlinear yo-yo motion system," *IEEE Transactions on Industrial Electronics*, vol. 54, pp. 3234–3245, 2007.
[32] H. Lu, H. M. Zhang, and L. H. Ma, "A new optimization algorithm based on chaos," *Journal of Zhejiang University Science A*, vol. 7, pp. 539–542, 2006.
[33] I. C. Trelea, "The particle swarm optimization algorithm: convergence analysis and parameter selection," *Information Processing Letters*, vol. 85, pp. 317– 325, 2003.
[34] S. Naka, T. Genji, T. Yura, and Y. Fukuyama, "A hybrid particle swarm optimization for distribution state estimation," *IEEE Transactions on Power Systems*, vol. 18, pp. 60–68, 2003.
[35] H. Gao, Y. Zhang, S. Liang, and D. Li, "A new chaotic algorithm for image encryption," *Chaos, Solitons and Fractals*, vol. 29, pp. 393–399, 2006.
[36] L. Y. Chuang, S. W. Tsai, and C. H. Yang, "Chaotic catfish particle swarm optimization for solving global numerical optimization problems," *Applied mathematics and computation*, vol. 217, pp. 6900-6916, 2011.
[37] J. Chuanwen and E. Bompard, "A self-adaptive chaotic particle swarm algorithm for short term hydroelectric system scheduling in deregulated environment," *Energy Conversion and Management*, vol. 46, pp. 2689–2696, 2005.
[38] D. E. Knuth, "The Art of Computer Programming, Seminumerical Algorithms," Third ed., vol. 2, Addison-Wesley, 1997, pp. 10–26.
[39] D. Kuo, "Chaos and its computing paradigm," *IEEE Potentials Magazine*, vol. 24, pp. 13–15, 2005.
[40] K. J. Astrom, and T. Hagglund, PID Controllers: Theory, Design and Tuning, ISA, Research Triangle, Par, NC, 1995.
[41] J. G. Ziegler and N. B. Nichols, "Optimum settings for automatic controllers," *Trans. ASME*, vol. 65, pp. 433-444, 1942.
[42] D. Maiti, A. Acharya, M. Chakraborty, A. Konar, and R. Janarthanan, "Tuning PID and PIλDμcontrollers using the integral time absolute error criteria," in *Proc. of 4th international conference on information and automation for sustainability* (ICIAFS 2008), 2008, pp. 457–62.
[43] J. Y. Cao and B. G. Cao, "Design of fractional order controllers based on particle swarm optimization," in *Proc.1st IEEE industrial electronics and applications*, 2006, pp. 1–6.
[44] J. Y. Cao, J. Liang, and B. G. Cao, "Optimization of fractional order PID controllers based on Genetic Algorithms," in *Proc. of the 2005 international conference on machine learning and cybernetics*, vol. 9, pp.5686–5689.