The Fourteenth International Symposium on Artificial Life and Robotics 2009 (AROB 14th '09),
B-Con Plaza, Beppu, Oita, Japan, February 5 - 7, 2009

A PSO Method with Nonlinear Time-Varying Evolution for Optimal Design of PID Controllers in a Pendubot System

Pi-Yun Chen[1], Chia-Ju Wu[2], Yu-Yi Fu[3], Chia-Nan Ko[4], and Li-Chun Lai[5]

[1]Graduate School of Engineering Science and Technology, National Yunlin University of Science and Technology
Douliou, Yunlin 640, Taiwan
[2]Department of Electrical Engineering, National Yunlin University of Science and Technology
Douliou, Yunlin 640, Taiwan
[3,4]Department of Automation Engineering, Nan-Kai University of Technology
Tasotun, Nantou 542, Taiwan
[5]Department of Electrical Engineering, Chungchou Institute of Technology
Yuanlin, Changhua 510, Taiwan
(E-mail: [1]g9310820@yuntech.edu.tw; [2] wucj@yuntech.edu.tw ; [3] t098@nkut.edu.tw ; [4] t105@nkut.edu.tw;
[5]lclai@dragon.ccut.edu.tw)

Abstract: A particle swarm optimization method with nonlinear time-varying evolution (PSO-NTVE) is employed in designing an optimal PID controller for asymptotical stabilization of a pendubot system. In the PSO-NTVE method, parameters are determined by using matrix experiments with an orthogonal array, in which a minimal number of experiments would have an effect that approximates the full factorial experiments. The PSO-NTVE method and other PSO methods are then applied to design an optimal PID controller in a pendubot system. Comparing the simulation results, the feasibility and the superiority of the PSO-NTVE are verified.

Keywords: Particle swarm optimization, orthogonal array, nonlinear time-varying evolution, PID controllers, pendubot system.

I. INTRODUCTION

A pendubot is a two-link (two-degree-of-freedom) underactuated system and it includes two links rotating on two joints, in which the first link (shoulder) is actuated and the second link (elbow) is not actuated. This system is a simple underactuated mechanical system to permit complete dynamic analyses, but complex enough for investigating many advanced nonlinear control methods. Therefore, it is widely used as a benchmark in the study of underactuated systems [1-4].

In order to stabilize the pendubot to the unstably inverted equilibrium position, a two-stage control strategy is always used. In the first stage, swing-up control is used to move the pendubot close to the equilibrium manifold. Then in the second stage, the swing-up controller is replaced by a balance controller for position control. Many methods such as the partial feedback linearization technique[1,2], the energy based controller[3], and the bang-bang controller[4] have been conducted for swing-up control of the pendubot. Once both links are swung up, the problem of balancing the pendubot about the unstable equilibrium was investigated. A linear quadratic regulator was proposed for the balance control[1,4]. Furthermore, the hybrid controller and the energy based controller[2,3] were presented.

Particle swarm optimization (PSO) has evolved recently as an important branch of stochastic techniques to explore the search space for optimization[5]. Nowadays, PSO has been developed to be real competitors with other well-established techniques for evolutionary-based optimization methods[6-10]. In this paper, a PSO-NTVE method is employed in the designing of an optimal PID controller in a pendubot system since it can effectively deal with continuous nonlinear programming problems and generate high quality solutions. From the simulation results of the illustrative examples, the feasibility and the validity of the PSO-NTVE are verified.

II. PSO-NTVE-BASED PID CONTROLLERS

1. Review of some PSO methods

In PSO algorithm, each particle keeps track of its own position and velocity in the problem space. The initial position and velocity of a particle are generated randomly. At each iteration, the new positions and velocities of the particles are updated using the following two equations:

$$P_i(k+1) = P_i(k) + V_i(k+1) \quad \text{for } i = 1, 2, \cdots, m \tag{1}$$

$$V_i(k+1) = V_i(k) + c_1 \cdot r_1 \cdot (P_i^l(k) - P_i(k)) + c_2 \cdot r_2 \cdot (P^g - P_i(k)) \tag{2}$$

where m is the number of particles in a population, k is the number of current iteration, c_1 and c_2 are acceleration coefficients, r_1 and r_2 are random numbers between 0 and 1, $P_i(k)$, $P_i^l(k)$, and $V_i(k)$ are the position, the local best, and the velocity of ith particle at iteration k, P^g is the global best of all particles.

Since the introduction of the PSO method in 1995, researchers have put much effort to improve the original version of PSO. Shi and Eberhart[11] used a linearly varying inertia weight over iterations. The mathematical representations of this PSO method are given as shown in (1) and

$$V_i(k+1) = \omega(k)\cdot V_i(k) + c_1\cdot r_1\cdot (P_i^l(k) - P_i(k)) + c_2\cdot r_2\cdot (P^g - P_i(k)) \text{ for } i=1,\ 2,\cdots,\ m \quad (3)$$

where the acceleration coefficients c_1 and c_2 are fixed, r_1 and r_2 are two random numbers. The inertia weight starts with a high value $\omega_{\max}$ and linearly decreases to $\omega_{\min}$ at the maximal number of iterations. From hereafter, this PSO algorithm will be referred to as the time-varying inertia weight factor method (PSO-TVIW).

Eberhart and Shi[12] found that the PSO-TVIW method is not very effective in tracking dynamic systems. Considering the dynamic nature of real-world applications, they proposed a random inertia weight factor to track dynamic systems. In their method, the representations are the same as those in the PSO-TVIW method except that the inertia weight factor changes randomly. In the rest of this paper, this algorithm will be referred to as the PSO-RANDW method.

An automation strategy for the PSO with time-varying acceleration coefficients was proposed[13]. The objective is to enhance the global search in the early part of the optimization and to encourage the particles to converge toward the global optimum at the end of the search. In their method, the representations are the same as those in the PSO-TVIW method except that the acceleration coefficients change according to linear time-varying evolution. From hereafter, this algorithm will be referred to as the PSO-TVAC method.

A time-varying nonlinear function modulated inertia weight adaptation was proposed by Chatterjee and Siarry[14]. In this method, the acceleration coefficients are also fixed. However, the inertia weight starts with a high value $\omega_{\max}$ and nonlinearly decreases to $\omega_{\min}$ at the maximal number of iterations. This means that the representations are the same as those in the PSO-TVIW method except that the inertia weight factor changes according to

$$\omega(k) = \omega_{\min} + \left(\frac{iter_{\max} - iter}{iter_{\max}}\right)^{\alpha}\cdot(\omega_{\max} - \omega_{\min}) \quad (4)$$

where $iter_{\max}$ is the maximal number of iterations and $iter$ is the current number of iterations.

2. PSO-NTVE method based on orthogonal arrays

In this section, based on the concept presented[13,14], a PSO-NTVE method is proposed. In the proposed PSO method, the inertia weight is given as described in (4). The gnitive parameter c_1 starts with a high value $c_{1\max}$ and nonlinearly decreases to $c_{1\min}$. Meanwhile, the social parameter c_2 starts with a low value $c_{2\min}$ and nonlinearly increases to $c_{2\max}$. This means that the mathematical expressions are given as shown in (1), (4), and

$$V_i(k+1) = \omega(k)\cdot V_i(k) + c_1(k)\cdot r_1\cdot (P_i^l(k) - P_i(k)) + c_2(k)\cdot r_2\cdot (P^g - P_i(k)) \text{ for } i=1,\ 2,\cdots,\ m \quad (5)$$

$$c_1(k) = c_{1\min} + \left(\frac{iter_{\max} - iter}{iter_{\max}}\right)^{\beta}\cdot(c_{1\max} - c_{1\min}) \quad (6)$$

$$c_2(k) = c_{2\max} + \left(\frac{iter_{\max} - iter}{iter_{\max}}\right)^{\gamma}\cdot(c_{2\min} - c_{2\max}) \quad (7)$$

where α, β, and γ are constant coefficients.

The proposed PSO method will encourage particles to wander through the entire search space, instead of clustering around a local optimum, during early iterations of the optimization. On the other hand, the algorithm will expedite convergence toward the global optimum during latter iterations. In this manner, the optimal solution should be obtained in a computation-efficient way.

To determine the optimal combination of α, β, and γ, all combinations must be tested. For example, if it is assumed that α, β, and γ are all within the set $\{0.5, 1, 1.5, 2, 2.5\}$. Then there are 5^3 possible combinations for the values of α, β, and γ. However, if α, β, and γ have many possible values, then it may not be possible to perform the experiments of all combinations. An $L_{25}(5^6)$ is an orthogonal array that can deal with at most six variables in five possible values with 25 experiments[15,16]. Instead of 5^3 possible combinations, one only needs to perform 25 experiments to determine the optimal combination of α, β, and γ.

3. PSO-NTVE tuning PID controllers

In a PID control system, the time-domain form of a PID controller is usually expressed as

$$u(t) = K_{P1}e_1(t) + K_{I1}\int e_1(t)dt + K_{D1}\dot{e}_1(t) + K_{P2}e_2(t) + K_{I2}\int e_2(t)dt + K_{D2}\dot{e}_2(t) \quad (8)$$

where $u(t)$ is the control signal, $e_1(t)$, $e_2(t)$ and $\dot{e}_1(t)$, $\dot{e}_2(t)$ are the error signals and their derivatives, and $K_{P1}, K_{I1},\cdots, K_{D2}$ denote the proportional gain, the integral gain, and the derivative gain, respectively. In the PSO, a particle contains these gains. The optimal values of these gains are obtained by the PSO-NTVE method according to a defined fitness.

III. A SIMULATION EXAMPLE

1. Pendubot system

The general dynamic model of underactuated mechanisms with m actuated joints from a total of n joint can be expressed as follows[17]:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\ \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) = \boldsymbol{\tau} \quad (9)$$

where $\boldsymbol{q} \in R^n$ is the position vector indicating link angles, $\boldsymbol{M}(\boldsymbol{q})$ denotes the $n\times n$ inertia matrix, $\boldsymbol{C}(\boldsymbol{q},\ \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \in R^n$ is the vector of damping, coriolis, and centrifugal torques, $\boldsymbol{G}(\boldsymbol{q}) \in R^n$ represents the gravitational term and $\boldsymbol{\tau} \in R^n$ is the vector of control torque which has $(n-m)$ zero components. For the pendubot system in Fig. 1, let m_1 and m_2 denote the distributed mass of the actuated link (link 1) and the unactuated link (link 2), respectively. Mean- while, let q_1 and q_2, l_1 and l_2, l_{1c} and l_{2c}, and I_1 and I_2 denote the angles, the lengths, the distances to the center of mass, the moments of inertia about their centroids of link 1 and link 2, respectively.

Since the inertia matrix $\boldsymbol{M}(\boldsymbol{q})$ is positive definite for all $\boldsymbol{q}$, the dynamics of the Pendubot in (10) can be written as

$$\ddot{\boldsymbol{q}} = \boldsymbol{M}^{-1}(\boldsymbol{q})[\boldsymbol{\tau} - \boldsymbol{C}(\boldsymbol{q},\ \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{G}(\boldsymbol{q})] \tag{10}$$

$$\boldsymbol{M}(\boldsymbol{q}) = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix},\quad \boldsymbol{C}(\boldsymbol{q},\ \dot{\boldsymbol{q}}) = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix},$$

$$\boldsymbol{G}(\boldsymbol{q}) = \begin{bmatrix} G_1 \\ G_2 \end{bmatrix},\quad \boldsymbol{q} = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix},\quad \boldsymbol{\tau} = \begin{bmatrix} \tau_1 \\ 0 \end{bmatrix} \tag{11}$$

where

$$M_{11} = m_1 l_{c1}^2 + m_2 (l_1^2 + l_{c2}^2 + 2 l_1 l_{c2} \cos q_2) + I_1 + I_2$$

$$M_{12} = m_2 (l_{c2}^2 + l_1 l_{c2} \cos q_2) + I_2$$

$$M_{21} = M_{12},\quad M_{22} = m_2 l_{c2}^2 + I_2$$

$$C_{11} = -m_2 l_1 l_{c2} \dot{q}_2 \sin q_2$$

$$C_{12} = -m_2 l_1 l_{c2} (\dot{q}_1 + \dot{q}_2) \sin q_2$$

$$C_{21} = m_2 l_1 l_{c2} \dot{q}_1 \sin q_2 + K_d,\quad C_{22} = K_d$$

$$G_1 = (m_1 g l_{c1} + m_2 g l_1) \cos q_1 + m_2 g l_{c2} \cos(q_1 + q_2)$$

$$G_2 = m_2 g l_{c2} \cos(q_1 + q_2)$$

where K_d denotes the damping coefficient.

When the configuration is at equilibrium state; that is, pendubot balances at a state $\dot{q} = 0$ and $\ddot{q} = 0$, the following can be derived from (9).

$$(m_1 g l_{c1} + m_2 g l_1) \cos q_1 + m_2 g l_{c2} \cos(q_1 + q_2) = \tau_1 \tag{12}$$

$$m_2 g l_{c2} \cos(q_1 + q_2) = 0 \tag{13}$$

In the natural equilibriums of the pendubot, which means $\tau_1 = 0$, the solutions of (12) and (13) can be $q_1 = \pi/2$, $q_2 = 0$. In this manner, both link 1 and link 2 are in their upper positions. From the analysis, one can realize that the control of the pendubot system is not an easy task. Therefore, in the following sections, it will be shown how to design a PID controller to asymptotically drive the pendubot to the equilibrium state.

2. Fitness

In the time domain, the fitness function of a PID controller can include performance criteria such as the overshoot, the rise time, the settling time, and the steady-state error[18,19]. In general, the PID controller design method using the integrated absolute error (IAE), or the integral of squared-error (ISE), or the integrated of time-squared-error (ITSE) is often employed in control system designs. In this paper, the ITSE performance criterion is adopted to evaluate the PID controller. The performance criteria can be included in the same fitness as follows:

$$f = \frac{1}{\int t\,[e_1^2(t) + e_2^2(t)]\,dt} \tag{14}$$

From the definition (16), the fitness value can be calculated to evaluate the performance of the PID controller and a higher fitness value denotes a better performance.

IV. SIMULATION RESULTS

The parameters of the pendubot system shown in Fig. 1 are chosen as $m_1 = 2.0$ kg, $m_2 = 1.5$ kg, $l_1 = 0.3$ m, $l_2 = 0.5$ m, $l_{1c} = 0.15$ m, $l_{2c} = 0.25$ m, $g = 9.8$ m/s^2. The initial state and the desired final state of the pendubot system are $[q_1, \dot{q}_1, q_2, \dot{q}_2] = [-\pi/2,\ 0,\ 0,\ 0]$ and $[q_1, \dot{q}_1, q_2, \dot{q}_2] = [\pi/2,\ 0,\ 0,\ 0]$. Meanwhile, the input torque $\tau(t)$ of the motor is assumed to be within the range $[-8\ \text{Nm}, 8\ \text{Nm}]$.

In the proposed algorithm, the population size and the maximal iteration number are chosen to be 40, 10000, respectively. Moreover, the particles in PSO methods are all chosen as real numbers in the range $[-10, 10]$. In the proposed PSO-NTVE method, the values of α, β, and γ in (4), (6), and (7) are 0.5, 1.0, and 2.5 determined by experiments of orthogonal arrays. The average values of the optima PID gains and fitness for 20 trials are shown in Table 1.

From the results, it is clear that most considered in this paper are competitive in finding the optimal solution. Hoever, the performance of PSO-NTVE was found to be relatively better than other PSO methods in finding the optimal PID gains.

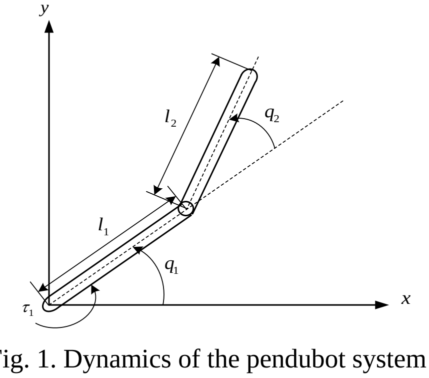

Fig. 1. Dynamics of the pendubot system

V. CONCLUSION

A PSO-NTVE method is presented in the designing of an optimal PID controller for a pendubot system. An orthogonal array is adopted to determine the parameters of the proposed PSO method. Then, this PSO-NTVE method is applied to design an optimal PID controller for asymptotically stabilizing the pendubot system. The simulation results verify the feasibility and the validity of the proposed PSO-NTVE method in the design of an optimal PID controller of a pendubot system.

Table 1 Average values of the optimal PID gains and fitness (14) for 20 trials.

PSO method	PID gains						Fitness(x10^{-2})
	K_{P1}	K_{I1}	K_{D1}	K_{P2}	K_{I2}	K_{D2}	
PSO-TVIW	1.373742	-0.521886	-1.959868	-9.999187	-0.207341	-2.865347	**8.638798**
PSO-RANDW	0.151508	0.063432	-8.694078	-9.991733	0.014415	-9.999336	**8.588553**
PSO-TVAC	0.912552	-0.760333	-2.479824	-9.999038	-0.307233	-3.409042	**8.638881**
PSO-NTVE	1.751383	-0.103256	-3.467748	-9.227502	-0.045068	-4.446959	**8.765612**

ACKNOWLEDGMENT

This work was supported in part by the National Science Council, Taiwan, R.O.C., under grant NSC96-2628-E-224-007-MY2.

REFERENCES

[1] Spong MW, Block DJ (1995), The Pendubot: a mechatronic system for control research and education. Proceedings of the 34th IEEE conference on decision and control, pp. 555-556
[2] Zhang M, Tarn TJ (2002), Hybrid control of the Pendubot. IEEE/ASME Trans on Mechatronics (1):79-86
[3] Fantoni I, Lozano R, Spong MW (2000), Energy based control of Pendubot. IEEE Trans on Automatic Control 45(4):725-729
[4] Zhao D, Yi J (2003), Swing up Pendubot with a GA-tuned bang-bang controller. Proceedings of the IEEE International Conference on Robotics, Intelligent Systems and Signal Processing, pp. 752-757
[5] Kennedy J, Eberhart R (1995), Particle swarm optimization. Proceedings of the IEEE International Conference on Neural Networks, pp. 1942-1948
[6] Gaing ZL (2004), A particle swarm optimization approach for optimum design of PID controller in AVR system. IEEE Trans on Energy Conversion 19(2):384-391
[7] Eberhart RC, Shi Y (1998), Comparison between genetic algorithms and particle swarm optimization. Proceedings of the IEEE International Congress on Evolutionary Computation, pp. 611-616
[8] Boeringer DW, Werner DH (2004), Particle swarm optimization versus genetic algorithms for phased array synthesis. IEEE Trans on Antennas and Propagation 52(3):771-779
[9] Mendes R, Kennedy J, Neves J (2004), The fully informed particle swarm: Simpler, maybe better. IEEE Trans on Evolutionary Computation 8(3):204-210
[10] Elbeltagi E, Hegazy T, Grierson D (2005), Comparison among five evolutionary-based optimization algorithms. Advanced Engineering Informatics 19(1):43-53
[11] Shi Y, Eberhart RC (1998), A modified particle swarm optimizer. Proceeding of the IEEE International Conference on Evolutionary Computation, pp. 69-73
[12] Eberhart RC, Shi Y (2001), Tracking and optimizing dynamic systems with particle swarms. Proceeding of the IEEE International Congress on Evolutionary Computation, pp. 94-100
[13] Ratnaweera A, Halgamuge SK, Watson HC (2004), Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Trans on Evolutionary Comput 8(3):240-255
[14] Chatterjee A, Siarry P (2006), Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization. Computers and Operations Research 33:859-871
[15] Montgomery DC (1991), Design and Analysis of Experiments, third edition. Wiley, New York
[16] Hicks CR (1993), Fundamental Concepts in the Design of Experiments, fourth edition. Saunders College, Texas
[17] Spong MW, Vidyasagar M (1989), Robot dynamics and control. Wiley, New York
[18] Krohling RA, Jaschek H, Rey JP (1997), Designing PI/PID controllers for a motion control system based on genetic algorithms. Proceedings of the IEEE International Symmetric Intelligent Control, pp. 125-130
[19] Ogata K (2002), Modern control engineering, fourth edition. Prentice-Hall, New Jersey