International Journal of Innovative Computing, Information and Control
Volume **8**, Number **6**, June **2012**
 ISSN 1349-4198
pp. 4421–4430

GAIN OPTIMIZATION FOR INERTIA WHEEL PENDULUM STABILIZATION USING PARTICLE SWARM OPTIMIZATION AND GENETIC ALGORITHMS

Ricardo Martinez-Soto[1], Antonio Rodriguez[1], Oscar Castillo[2] and Luis T. Aguilar[3]

[1]School of Engineering
UABC University
Tijuana, México
mc.ricardo.martinez@hotmail.com; ardiaz@uabc.edu.mx

[2]Tijuana Institute of Technology
Tijuana, México
ocastillo@tectijuana.mx

[3]Instituto Politécnico Nacional
Ave. del Parque 1310, Mesa de Otay, Tijuana 22510, México
luis.aguilar@ieee.org

Received February 2011; revised August 2011

Abstract. *We describe in this paper the optimization of the gains of a PID controller to stabilize the inertia wheel pendulum (IWP) using bio-inspired and evolutionary methods. Particle swarm optimization and genetic algorithms are used to find the optimal gain values of the PID controller. Computer simulations and experiments are presented showing the control results using the optimal gain values to stabilize the inertia wheel pendulum. Both particle swarm optimization (PSO) and genetic algorithms (GAs) are shown to be effective tools for gain optimization of the inertia wheel.*
Keywords: Genetic algorithms, Particle swarm optimization, Optimization methods, Stabilization control

1. **Introduction.** Optimization is a term used to refer to a branch of computational science concerned with finding the "best" solution to a problem. Here, "best" refers to an acceptable (or satisfactory) solution, which may be the absolute best over a set of candidate solutions, or any of the candidate solutions. The characteristics and requirements of the problem determine whether the overall best solution can be found [1]. Optimization algorithms are search methods, where the goal is to find a solution to an optimization problem, such that a given quantity is optimized, possibly subject to a set of constraints [1]. Some optimization methods are based on populations of solutions. Unlike the classic methods of improvement for trajectory tracking control, in this case each iteration of the algorithm has a set of solutions. These methods are based on generating, selecting, combining, and replacing a set of solutions. Since they maintain and manipulate a set, instead of a unique solution throughout the entire search process, they require more computer time than other metaheuristic methods. This fact can be aggravated because the "convergence" of the population requires a great number of iterations. For this reason a concerted effort has been dedicated to obtaining methods that are more aggressive and manage to obtain solutions of quality in a nearer horizon.

Related works. Up to date there are several research papers using genetic algorithms and particle swarm optimization for different optimization problems, like vehicles routing problems [2], stabilization control [3], tuning of PID controllers [4-6] and algorithm modifications for better improvements [7-10]. However, in this paper we used the GA and PSO to help find optimal gain values to stabilize the inertia wheel pendulum. The main goal of this research is to learn how the GA and PSO perform when applied to the inertia wheel pendulum in simulation and physical experiments. The reason that we consider the inertia wheel pendulum is that it has been used by several researchers [11] as a benchmark to test the effectiveness of proposed control designs. The inertia wheel pendulum is a flat underactuated mechanical system with two degrees of freedom and a single actuator. The inertia wheel pendulum was first introduced by Spong et al. [11], and consists of a pendulum with a rotating uniform inertia wheel at its end. The pendulum is underactuated and the system has to be controlled via the rotating wheel. The task is to stabilize the pendulum in its upright equilibrium point while the wheel stops rotating. The specific angle of rotation of the wheel is not important in the present paper. The inertia wheel pendulum has been used by many researchers as a case of study, in different ways, such as in the control area [12-15]; like a model for other applications [16].

This paper is organized as follows. Section 2 presents the theoretical basis for this work. Section 3 presents the inertia wheel pendulum as a case of study. Section 4 presents the parameter configuration of the optimization methods to find the optimal gains. Section 5 presents the optimal gains obtained using a simulation and Section 6 presents the experimental results using the optimal gains obtained by the optimization methods, where a genetic algorithm (GA) and particle swarm optimization (PSO) are used to select the parameters for the stabilization of the inertia wheel pendulum described in Section 3. Finally, Section 7 presents the conclusions.

2. Theoretical Basis and Problem Statement.

In this section we present a brief overview of the basic concepts needed for this work.

2.1. Particle swarm optimization.

Particle swarm optimization is a population based stochastic optimization technique developed by Eberhart and Kennedy in 1995, inspired by social behavior of bird flocking or fish schooling [17]. PSO shares many similarities with evolutionary computation techniques such as the GA [18].

The particle swarm optimization concept consists of, at each time step, changing the velocity of (accelerating) each particle toward its *pbest* and *lbest* locations (local version of PSO). Acceleration is weighted by a random term, with separate random numbers being generated for acceleration toward the *pbest* and *lbest* locations [19,20,22]. In the past several years, PSO has been successfully applied in many research and application areas. It has been demonstrated that PSO gets better results in a faster, cheaper way when compared with other methods [21,22]. Another reason that PSO is attractive is that there are few parameters to adjust. One version, with slight variations, works well in a wide variety of applications. Particle swarm optimization has been considered for approaches that can be used across a wide range of applications, as well as for specific applications focused on a specific requirement [22,23].

The basic PSO algorithm has the following nomenclature:

x_k^i – Particle position, v_k^i – Particle velocity, w_{ij} – Inertia weight, p_k^i – Best "remembered" individual particle position, p_k^g – Best "remembered" swarm position, c_1, c_2 – Cognitive and Social parameters, r_1, r_2 – Random numbers between 0 and 1.

The equation to calculate the velocity is:

$$v_{k+1}^i = w_{ij}\, v_k^i + c_1\, r_1 \left(p_k^i - x_k^i\right) + c_2\, r_2 \left(p_k^g - x_k^i\right) \tag{1}$$

and the position of the individual particles is updated as follows:

$$x_{k+1}^i = x_k^i + v_{k+1}^i \tag{2}$$

The basic PSO algorithm is defined as follows:

1) Initialize

Set the constants $z_{\max}$, c_1, c_2

Randomly initialize particle positions $x_0^i \in D$ *in* R^n *for* $i = 1, \ldots, p$

Randomly initialize particle velocities $0 \le v_0^i \le v_0^{\max}$ *for* $i = 1, \ldots, p$

Set $z = 1$

2) Optimize

Evaluate function value f_k^i *using the design space coordinates* x_k^i

If $f_k^i \le f_{best}^i$ *then* $f_{best}^i = f_k^i$, $p_k^i = x_k^i$.

If $f_k^i \le f_{best}^g$ *then* $f_{best}^g = f_k^i$, $p_k^g = x_k^i$.

If stopping condition is satisfied then go to 3.

Update all particle velocities v_k^i *for* $i = 1, \ldots, p$

Update al particle positions x_k^i *for* $i = 1, \ldots, p$

Increment z.

Go to 2(a).

3) Terminate

2.2. **Genetic algorithms.** Genetic Algorithms (GAs) are adaptive heuristic search algorithms based on the evolutionary ideas of natural selection and genetic processes [24]. The basic principles of GAs were first proposed by John Holland in 1975, inspired by the mechanism of natural selection where stronger individuals are likely to be the winners in a competing environment [25-27]. The GA assumes that the potential solution to any problem is an individual and can be represented by a set of parameters. These parameters are regarded as the genes of a chromosome and can be structured by a string of values in binary form. A positive value, generally known as a fitness value, is used to reflect the degree of "goodness" of the chromosome for the problem, which would be highly related with its objective value. The GA works as follows:

- *Start with a randomly generated population of n chromosomes (candidate solutions to a problem).*
- *Calculate the fitness of each chromosome in the population.*
- *Repeat the following steps until n offspring have been created:*
  - *Select a pair of parent chromosomes from the current population, the probability of selection being an increasing function of fitness. Selection is done with replacement, meaning that the same chromosome can be selected more than once to become a parent.*
  - *With a certain probability (crossover rate), perform crossover to the pair at a randomly chosen point to a form two offspring.*
  - *Mutate the two offspring at each locus with probability (mutation rate), and place the resulting chromosomes in the new population.*
- *Replace the current population with the new population.*
- *Go to Step 2.*

The simple procedure just described above is the basis for most applications of GAs.

3. **Inertia Wheel Pendulum (IWP): A Case of Study.** In this paper, we use an inertial wheel pendulum to apply particle swarm optimization and genetic algorithms as methods to find the optimal gain values of the controller.

The dynamics of an inertia wheel pendulum can be described as follows [28]:

$$\begin{bmatrix} J_1 & J_2 \\ J_2 & J_2 \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} h \sin q_1 \\ f_s \, \dot{q}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ k \end{bmatrix} u \tag{3}$$

where $q_1(t) \in R$ is the absolute angle of the pendulum, counted clockwise from the vertical downward position; $q_2(t) \in R$ is the absolute angle of the disk; $f_s(\dot{q}_2)$ represents the viscous friction force affecting the actuator where $f_s > 0$ is the viscous friction coefficient; J_1, J_2 and h are positive physical parameters, that depend on the geometric dimensions and the inertia-mass distribution; $\tau = ku$ is the scalar bounded applied torque. The friction of the pendulum is quite small and for this reason will be ignored. It should be noted that system (1) is nonlinear and underactuated.

Let us start by defining the following control law [28]

$$u = -k_1 \, q_1 - k_2 \, \dot{q}_1 - k_3 \, \dot{q}_2 \tag{4}$$

where the k_1, k_2 and k_3 are the gains.

The gain values will be accepted only if k_1, k_2 and k_3 are positive values. The pendulum is influenced by accelerating the wheel. Because of the physics of the motor, it will saturate if the wheel velocity is too high. It is thus desirable to try to achieve the dual goals of stabilizing the pendulum and to keep velocity small. To achieve this we propose to use particle swarm optimization and genetic algorithms to find the best gain values k_i, $i = 1, 2, 3$ to obtain a fast stabilization time with a low steady state error state of the $x(q_1)$.

3.1. **Experimental test bed.** The experimental results are based on the laboratory inertia wheel pendulum from the Mechatronics Control Kit manufactured by Quanser Inc., where $J_1 = 4.572 \times 10^{-3}$, $J_2 = 2.495 \times 10^{-5}$, $h = 0.3544$, $k = 0.00494$ and $\beta = 10$ (see [28]). It consists of a physical pendulum (q_1) with a motor/flywheel assembly attached to the free end of the-pendulum (q_2). The wheel is actuated by a 24 Volt, permanent magnet DC-motor and the coupling torque between the wheel and the pendulum can be used to control the motion of the system. The system is a non-minimum phase at the top. The pendulum angle is measured by an encoder. The experimental setup includes a PC equipped with a C6713 DSK Quanser interface/PWM amplifier board. The algorithm was implemented using the C programming language. The sampling frequency for the algorithm implementation was set to 400 Hz. The viscous coefficient $f_s = 8.80 \times 10^{-5}$ was identified by applying the procedure from [29].

4. **Optimization Method Configuration.** PSO and GA are the optimization methods that are used to find the optimal gain values for the inertia wheel pendulum. The objective function of the optimization method is the average of the steady state error of the pendulum; the goal of the objective function is to minimize the steady state error.

4.1. **PSO configuration.** For the PSO configuration, we change the parameter values of the acceleration coefficients (c_1 cognitive, c_2 social parameters) situated between the range of 0.001 – 1.0 with random values; for the inertia weight (w_{ij}), we use a range of 0.005 – 1.0 with random values, and a maximum number of iterations ($z_{\max}$) to stop the algorithm.

4.2. **GA configuration.** For the GA configuration, we change the parameter values of the mutation rate and crossover rate with a simple point, using the roulette wheel method of selection; to stop the algorithm we use the maximum number of generations. We used a 3 gene chromosome with real values to represent the gains. Both methods are using the same search range values (minimum and maximum) for each of the gains (k_1, k_2 and k_3), and Table 1 shows these values that we used based on the physical conditions of the real system.

TABLE 1. Search range of the gains values

	Minimum Values	*Maximum Values*
k_1	50	250
k_2	5	50
k_3	0.001	0.1

5. **Optimal Gains Obtained by GA and PSO.** In this section, we present the optimal gains obtained by the optimization methods using the mathematical model of the inertia wheel pendulum in Matlab and Simulink.

5.1. **Optimal gain values using particle swarm optimization.** Table 2 shows the parameters of the PSO that we used to obtain the optimal gain values, the average error of the simulation model and the optimal k_1, k_2 and k_3. The best gain values are the ones shown in row 7.

TABLE 2. Optimization results for the gain values obtained by PSO

PSO Parameters							*Obtained Values*			
No.	*Swarm*	*Max Iter.*	c_1	c_2	*Inertia*	*Time Exec.*	*Sim. Average error*	k_1	k_2	k_3
1	40	70	0.8149	0.9059	0.1706	12 : 17 : 32	0.000340	999.9949	24.2560	0.0685
2	40	70	0.0557	0.6927	0.0668	13 : 39 : 31	0.000300	798.4747	12.8768	0.0037
3	50	90	0.7779	0.6065	0.3444	19 : 41 : 35	0.000340	999.9994	29.1371	0.0946
4	30	50	0.4526	0.4887	0.7602	5 : 20 : 54	0.000340	999.9640	29.4740	0.0973
5	60	80	0.1618	0.6980	0.479	22 : 56 : 21	0.000250	997.6851	15.4593	0.0010
6	30	50	0.8149	0.9059	0.1706	6 : 41 : 44	0.000340	999.9185	27.4600	0.0856
7	***55***	***70***	***0.2499***	***0.3757***	***0.4429***	***13:33:58***	***0.000470***	***499.9494***	***19.0849***	***0.0587***
8	40	50	0.6845	0.5153	0.7706	6 : 04 : 14	0.000360	498.4005	10.5642	0.0010
9	30	60	0.0389	0.5842	0.2618	5 : 57 : 40	0.000500	435.6690	25.6338	0.0962
10	20	80	0.0407	0.9331	0.1686	4 : 53 : 25	0.000530	377.3395	24.7157	0.0936
11	35	40	0.1017	0.5240	0.9704	3 : 57 : 43	0.000470	498.5026	26.9477	0.0969
12	45	70	0.2699	0.8169	0.4454	9 : 14 : 12	0.000470	499.9918	19.7955	0.0624
13	60	40	0.6496	0.7987	0.5154	8 : 25 : 19	0.000470	499.9900	22.4885	0.0766

5.2. **Optimal gain values using genetic algorithms.** Table 3 shows the parameters of the GA that we used to obtain the optimal gain values, the average error of the simulation model and the optimal k_1, k_2 and k_3.

6. **Experimental Results.** Once we found the optimal gain values using the mathematical model implemented in Matlab and Simulink, we tested these optimal gains in the physical experimental inertia wheel pendulum, and this produced different results according to the steady state error obtained in the simulations.

The initial conditions for the inertial wheel pendulum are as follows:

$$\begin{array}{l} q_1(0) = 3.1, \quad q_2(0) = 0 \\ \dot{q}_1(0) = \dot{q}_2(0) = 0 \end{array} \tag{5}$$

Table 3. Optimization results for the gain values obtained by GA

	GA Parameters						*Obtained Values*			
No.	*Popul.*	*Max Gener.*	*% Remp.*	*Cross.*	*Mut.*	*Time Exec.*	*Sim. Average error*	k_1	k_2	k_3
1	35	40	0.7	0.7	0.3	10 : 14 : 04	0.000340	1000	29.09	0.0949
2	35	40	0.7	0.7	0.3	10 : 11 : 45	0.000320	1000	22.75	0.0070
3	30	70	0.7	0.7	0.2	15 : 27 : 31	0.000250	1000	15.24	0.0010
4	30	70	0.7	0.7	0.2	18 : 50 : 42	0.000250	1000	15.11	0.0010
5	60	45	0.7	0.7	0.2	2 : 41 : 21	0.000250	1000	15.02	0.0010
6	30	50	0.7	0.7	0.2	8 : 57 : 06	0.000470	500	25.75	0.0937
7	***35***	***85***	***0.7***	***0.8***	***0.2***	***16:56:35***	***0.000470***	***500***	***26.45***	***0.0974***
8	35	85	0.7	0.8	0.2	10 : 45 : 58	0.000570	250	5.00	0.0010
9	40	65	0.7	0.8	0.2	10 : 38 : 09	0.000530	250	7.14	0.0010
10	25	100	0.7	0.8	0.1	10 : 12 : 16	0.000630	250	21.85	0.0841
11	30	45	0.7	0.5	0.2	6 : 02 : 57	0.000530	250	7.09	0.0010
12	25	60	0.7	0.8	0.3	4 : 40 : 01	0.000530	250	7.13	0.0010
13	35	55	0.7	0.8	0.3	5 : 22 : 01	0.000530	250	7.12	0.0010

Table 4. Experimental results of the IWP with the optimal gain values obtained by PSO

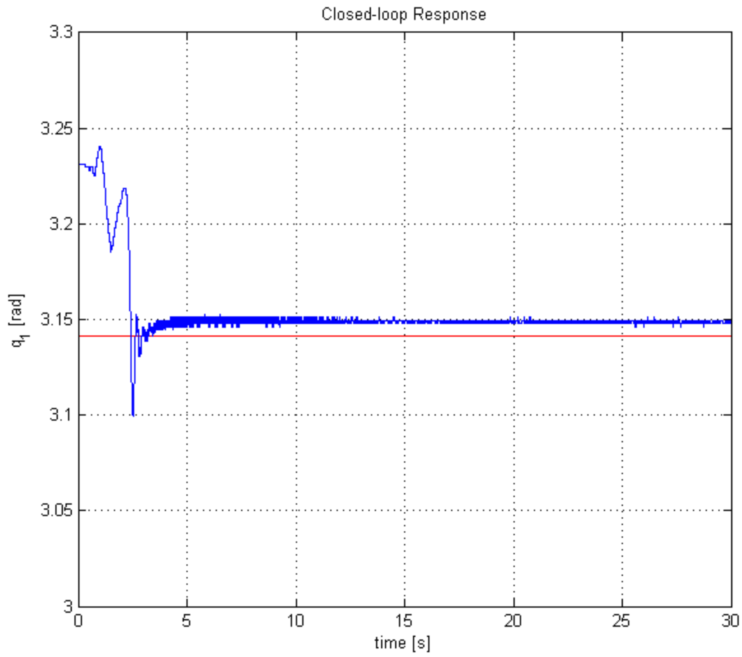
No.	*Optimal Gain Values*			*Experimental results*				
				Normal results			*Perturbed results*	
	k_1	k_2	k_3	*Aver. Error*	*max tau*	*Stabilization Time (S)*	*Aver. Error*	*max tau*
1	999.9949	24.2560	0.068500	0.01487	∞	∞	∞	∞
2	798.4747	12.8768	0.003700	0.0161	∞	∞	∞	∞
3	999.9994	29.1371	0.094600	0.00849	∞	∞	∞	∞
4	999.9640	29.4740	0.097300	0.01477	∞	∞	∞	∞
5	997.6851	15.4593	0.001000	∞	∞	∞	∞	∞
6	999.9185	27.4600	0.085600	∞	∞	∞	0.43641	46.39766
7	***499.9494***	***19.0849***	***0.058700***	***0.00723***	***6.89240***	***4.40***	***0.41301***	***50.33051***
8	498.4005	10.5642	0.001000	∞	∞	∞	∞	∞
9	435.6690	25.6338	0.096200	0.33395	5.44329	18.00	0.24084	46.85690
10	377.3395	24.7157	0.093600	0.00825	8.63359	2.79	0.21013	35.98965
11	498.5026	26.9477	0.096900	0.39999	13.46410	6.50	0.29181	64.97928
12	499.9918	19.7955	0.062400	0.33406	10.96320	4.25	0.30616	29.95476
13	499.9900	22.4885	0.076600	0.29829	12.84360	2.69	0.25259	39.12946

6.1. **Experimental results using the optimal gain values obtained by PSO.** Table 4 shows the experimental results using the optimal gains obtained by PSO showing in the seventh row the best stabilization time of the inertia wheel pendulum in the experiments.

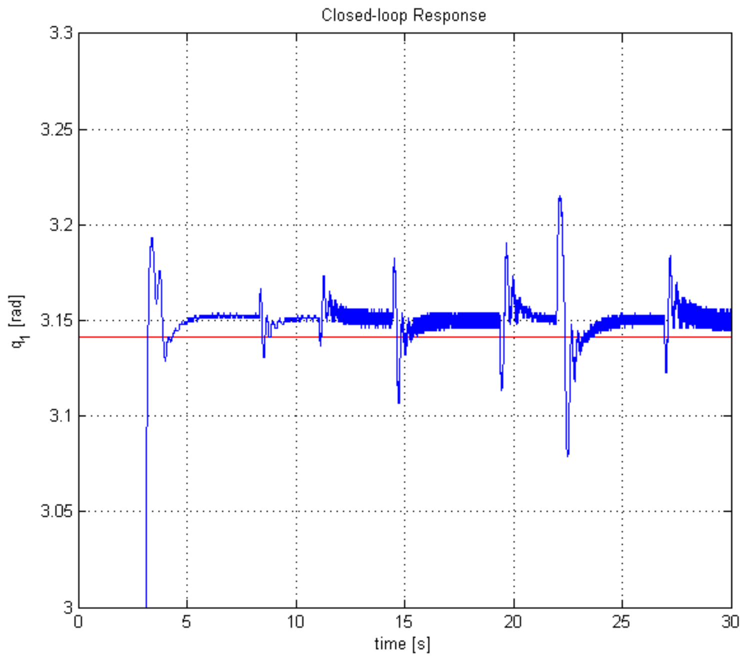
Figure 1(a) shows the experimental results of the inertia wheel pendulum response using the optimal gains: $k_1 = 499.94936$, $k_2 = 19.08493$ and $k_3 = 0.05867$, obtained by PSO resulting in an average error = 0.0072278 [rad], max $\tau = 6.8924$ [Nm] and tss = 4.4 seconds to stabilize and Figure 1(b) shows the experimental results with the same gains by randomly hitting the pendulum, in this case obtaining an average steady state error = 0.41301 [rad] and max $\tau = 50.33051$ [Nm].

6.2. **Experimental results using the optimal gains obtained by GA.** In Table 5, we show the experimental results using the optimal gains obtained with the GA, showing in the 10th row the best stabilization time of the inertia wheel pendulum in the experiments.

Figure 2(a) shows the experimental results of the inertia wheel pendulum response using the optimal gains: $k_1 = 250$, $k_2 = 21.849774$ and $k_3 = 0.0841414$, obtained with



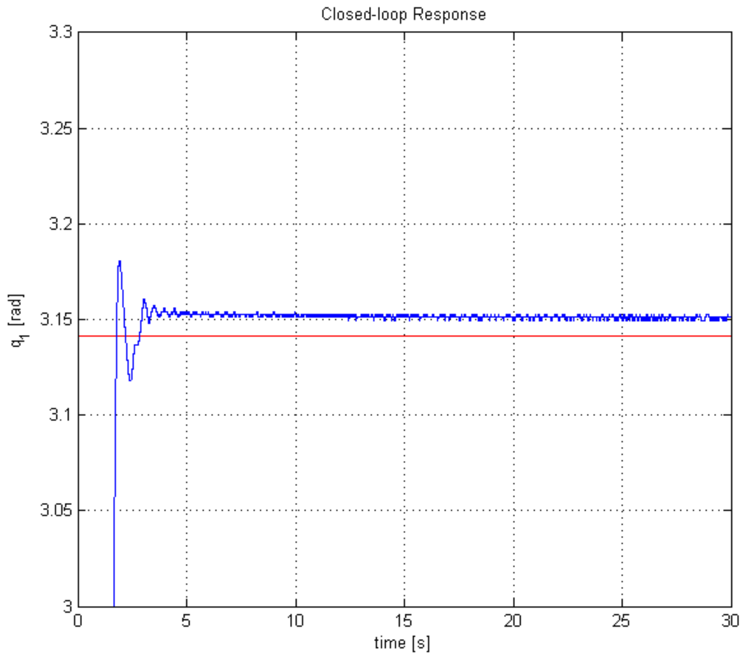
(a)



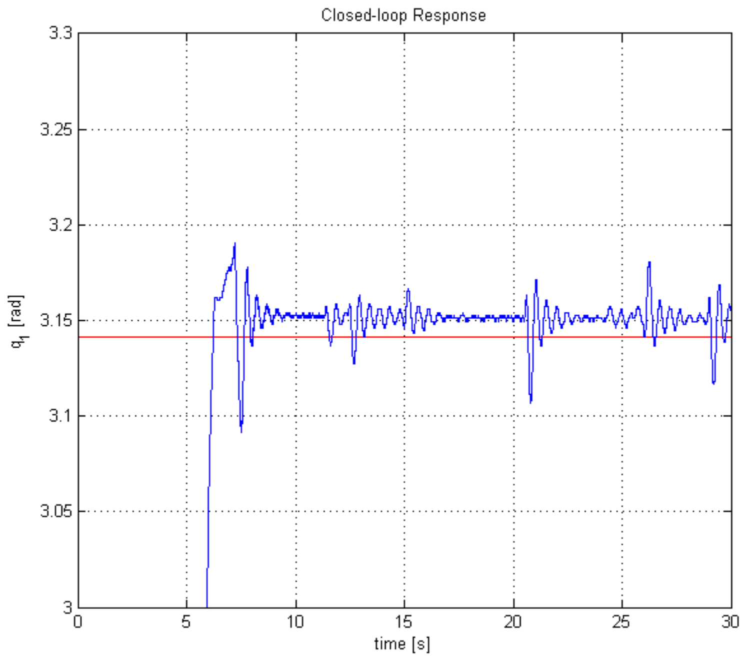
(b)

FIGURE 1. Experimental results of the IWP with PSO (a) normal and (b) perturbed control

the GA resulting on an average error = 0.22061 [rad], max τ = 3.7613548 [Nm] and tss = 4.5 seconds to stabilize and Figure 2(b) shows the experimental results with the same gains under perturbation, in this case obtaining an average steady state error = 0.27417 [rad] and max τ = 13.12699 [Nm].



(a)



(b)

FIGURE 2. Experimental results of the IWP with PSO (a) normal and (b) perturbed control

We made a comparison of the experimental results using the optimal gains obtained by particle swarm optimization and genetic algorithms. We can see that the experimental result using the optimal gains obtained by particle swarm optimization produces a better response in the stabilization than the experimental result with the optimal gains obtained with the genetic algorithm. However, under perturbation the experimental results using

TABLE 5. Experimental results of the GA for the gains of the IWP

No.	*Experimental results*				
	Normal results			*Perturbed results*	
	Average Error	*max tau*	*Stabilization Time (S)*	*Average Error*	*max tau*
1	∞	∞	∞	∞	∞
2	∞	∞	∞	∞	∞
3	∞	∞	∞	∞	∞
4	∞	∞	∞	∞	∞
5	∞	∞	∞	∞	∞
6	0.23737	17.83741	4	∞	∞
7	0.30960	13.77892	4	∞	∞
8	∞	∞	∞	∞	∞
9	∞	∞	∞	∞	∞
10	***0.22061***	***3.76135***	***4.5***	***0.27417***	***13.12699***
11	∞	∞	∞	∞	∞
12	∞	∞	∞	∞	∞
13	0.25126	3.96047	5	0.17291	23.48936

the optimal gains obtained by the genetic algorithm have a better stabilization response than the experimental results using the gains obtained by particle swarm optimization.

7. **Conclusions.** We described in this paper the use of optimization methods to find the optimal gains for the control of the inertia wheel pendulum. In particular we presented experimental results with the optimal gains obtained by PSO and GA. The results show that using the PSO algorithm we obtain the optimal gains for the inertia wheel pendulum and it is less time consuming than the GA results. Testing the optimal gains in the experimental system, we can verify which gain values stabilize the inertia wheel pendulum with less average error and stabilization time; finding our best results using particle swarm optimization.

We have achieved satisfactory results using the optimization methods to help us find the optimal gains to stabilize the inertia wheel pendulum. The next step is to solve more complex problems using fuzzy logic controllers and multiple objective optimizations, to obtain better results. Moreover, we will extend the results to nonlinear systems, like for autonomous mobile robots, and some other applications using different optimizations methods that we will be used to compare our proposed optimization method.

Acknowledgment. This work is partially supported by CONACYT under grant 127575.

REFERENCES

[1] A. P. Engelbretht, *Fundamentals of Computational Swarm Intelligence*, John Wiley & Sons, England, 2005.

[2] Y. Wu, P. Ji and T. Wang, An empirical study of a pure genetic algorithm to solve the capacitated vehicle routing problem, *ICIC Express Letters*, vol.2, no.1, pp.41-45, 2008.

[3] E. K. Boukas and N. K. M'Sirdi, Stabilization of linear systems via delayed state feedback controller, *ICIC Express Letters*, vol.2, no.1, pp.1-6, 2008.

[4] A. Kozáková, Tuning decentralized PID controllers for performance and robust stability, *ICIC Express Letters*, vol.2, no.2, pp.117-122, 2008.

[5] Š. Bucz, L. Harsányi and V. Veselý, A new approach to tuning robust PID controllers, *ICIC Express Letters*, vol.2, no.4, pp.317-322, 2008.

[6] W. Zu, H. Chang, G. Fan and J. Yi, A particle swarm optimization algorithm with equilibrious distribution parameter for global path planning of AUV, *ICIC Express Letters*, vol.4, no.6(A), pp.2207-2212, 2010.
[7] C. Wang, P. Sui and W. Liu, Improved particle swarm optimization algorithm based on double mutation, *ICIC Express Letters*, vol.3, no.4(B), pp.1417-1422, 2009.
[8] X. Cai and Z. Cui, Hungry particle swarm optimization, *ICIC Express Letters*, vol.4, no.3(B), pp.1071-1076, 2010.
[9] Z. Cui, L. Wang and Y. Tan, Particle swarm optimization with active congregation, *ICIC Express Letters*, vol.4, no.4, pp.1167-1172, 2010.
[10] S. Tsuda, L. Tao and M. Maeda, Solution search algorithm of particle swarm optimization with perturbation term, *ICIC Express Letters*, vol.5, no.5, pp.1515-1522, 2011.
[11] M. W. Spong, P. Corke and R. Lozano, Nonlinear control of the inertia wheel pendulum, *Automatica*, vol.37, pp.1845-1851, 2001.
[12] C. Aguilar, O. Gutierrez and M. Suarez, Controlling the strongly damping inertia wheel pendulum via nested saturation functions, *Computación y Sistemas*, vol.12, no.4, pp.436-449, 2009.
[13] D. Buitrago and V. Correa, Control local de un sistema no lineal: El péndulo con rueda de reacción, *Scientia Et Technica*, vol.8, no.34, pp.55-60, 2007.
[14] R. Olfati-Saber, Global stabilization of a flat underactuated system: The inertia wheel pendulum, *Proc. of the 40th Conference on Decision and Control*, Orlando, FL, USA, pp.3764-3765, 2001.
[15] L. T. Aguilar, I. Boiko, L. Fridman and L. Freidovich, Inducing oscillations in an inertia wheel pendulum via two-relays controller: Theory and experiments, *American Control Conference*, St. Louis, MO, USA, pp.65-70, 2009.
[16] S. H. Lee and A. Goswami, Reaction mass pendulum (RMP): An explicit model for centroidal angular momentum of humanoid robots, *IEEE International Conference on Robotics and Automation*, Rome, Italy, pp.4667-4672, 2007.
[17] R. C. Eberhart and J. Kennedy, A new optimizer using particle swarm theory, *Proc. of the 6th International Symposium on Micromachine and Human Science*, Nagoya, Japan, pp.39-43, 2002.
[18] V. Mukherjee and S. P. Ghoshal, Intelligent particle swarm optimized fuzzy PID controller for AVR system, *Electric Power Systems Research*, vol.77, pp.1689-1698, 2007.
[19] M.-R. Chen, X. Li, X. Zhang and Y.-Z. Lu, A novel particle swarm optimizer hybridized with extremal optimization, *Applied Soft Computing*, vol.10, pp.367-373, 2010.
[20] M. S. Arumugam and M. V. C. Rao, On the improved performances of the particle swarm optimization algorithms with adaptive parameters, cross-over operators and root mean square (RMS) variants for computing optimal control of a class of hybrid systems, *Applied Soft Computing*, vol.8, pp.324-336, 2008.
[21] D. H. Kim and K. Hirota, Vector control for loss minimization of induction motor using GA-PSO, *Applied Soft Computing*, vol.8, pp.1692-1702, 2008.
[22] J. Kennedy and R. Mendes, Population structure and particle swarm performance, *Proc. of IEEE Conference on Evolutionary Computation*, pp.1671-1676, 2002.
[23] P. J. Angeline, Using selection to improve particle swarm optimization, *Proc. of 1998 IEEE World Congress on Computational Intelligence*, Anchorage, AK, USA, pp.84-89, 1998.
[24] R. Alcalá, J. Alcalá-Fdez and F. Herrera, A proposal for the genetic lateral tuning of linguistic fuzzy systems and its interaction with rule selection, *IEEE Transactions on Fuzzy Systems*, vol.15, no.4, pp.616-635, 2007.
[25] R. Alcalá, M. J. Gacto, F. Herrera and J. Alcalá-Fdez, A multi-objective genetic algorithm for tuning and rule selection to obtain accurate and compact linguistic fuzzy rule-based systems, *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol.15, no.5, pp.539-557, 2007.
[26] J. Casillas, O. Cordon, M. J. del Jesús and F. Herrera, Genetic tuning of fuzzy rule deep structures preserving interpretability and its interaction with fuzzy rule set reduction, *IEEE Transactions on Fuzzy Systems*, vol.13, no.1, pp.13-29, 2005.
[27] O. Cordon, F. Gomide, F. Herrera, F. Hoffmann and L. Magdalena, Ten years of genetic fuzzy systems: Current framework and new trends, *Fuzzy Sets and Systems*, vol.141, no.1, pp.5-31, 2004.
[28] K. Astrom, D. Block and M. Spong, The reaction wheel pendulum, *Lecture Notes for the Reaction Wheel Pendulum (Part of the Mechatronics Control Kit)*, Urbana-Champaigne, IL, USA, 2001.
[29] R, Kelly, J. Llamas and R. Campa, A measurement procedure for viscous and Coulomb friction, *IEEE Trans. on Instrumentation and Measurements*, vol.49, no.4, pp.857-861, 2000.