Proceedings of the 36th
Conference on Decision & Control
San Diego, California USA • December 1997

Design of PID Controller for Precision Positioning Table Using Genetic Algorithms

Pai-Yi Huang and Yung-Yaw Chen
Department of Electric Engineering, National Taiwan University, Taipei, Taiwan, R.O.C.
E-mail: f81085@cctwin.ee.ntu.edu.tw

Abstract

In this paper, we demonstrate a PID controller design for high precision positioning table using a real-coded genetic algorithm. We choose the numerical internal presentation of the computer systems (IEEE P754 Standard) as the coding format of the genetic algorithm. By this approach, the time-consuming coding and encoding processes on traditional genetic algorithms is not necessary. In order to show the ability of this algorithm, a precision ballscrew table are used for evaluation. The experimental results indicate the success of this algorithm.

I. Introduction

In this article, we will try to design the PID controller for the precision positioning system. Generally speaking, the PID control is the most successful control scheme used in current industrial control processes. The Genetic algorithms (GAs) can be viewed as a general-purpose optimization method and have been successfully applied to search, optimization and machine learning tasks [1]. It has been already shown that GAs can learn to control a dynamic system without any prior knowledge about the system [2].

In this article, we used real-coded genetic algorithm as an optimizer for the PID controller design. The chromosome coding of this GA is based on IEEE P754 floating point standard that is used as the internal floating point representation in most computer systems [3]. With this approach, maximum resolution and accuracy of the numerical representation in computer can be achieved. The encoding and the decoding processes in common genetic algorithms become unnecessary due to the fact that the mapping operation from the real values to the encoded binary formats is intrinsically implemented in hardware. Besides, because the extreme wild coding ranges of the floating coding method, the searched parameters' range needs not to be known in advance. The experimental results on XY tables and precision table are shown to prove the success of this approach.

II. Real-coded Genetic Algorithm

The floating point representation covers a very wild range of the real number with great precision. The upper bounds and lower bounds of the parameters that have to be pre-specified in other real-coded GAs [4,5] are unnecessary now. From engineering point of view, the coding range $8.43 \cdot 10^{-37} \le |x| \le 3.37 \cdot 10^{38}$ with 6~7 decimal significant digits is rather enough for use.

Reproduction is a process based on the principle of survival of the fittest. The strings with a higher fitness value have a higher probability of contributing one or more offspring in the next generation. Crossover

Crossover provides a mechanism for individual strings to exchange information via a probabilistic process. In each generation, If there are n parameters needed to be determined, we define the parents $R_i \in R^n$ and the children $R_i' \in R^n, i = 1,2$.

$$\begin{cases} R_1' = R_1 + k_1 \times (R_2 - R_1) \\ R_2' = R_1 + k_1 \times (R_2 - R_1) \\ -2 \le k_1, k_2 \le 1 \end{cases}$$

The mutation operation is more like in a binary-coded GA. At each iteration, every gene is subject to a random change with probability of the pre-assigned mutation rate. In the binary-coded case, a mutation operator changes a bit from 0 to 1 or vice versa. There is only a little difference, that is, the mutation rate should be different in different fields. In the exponent part mutation rate should be lower, because the exponent part will have strongest effect on the coded values.

III. Experimental Method

The precision table (Figure 1) is made by NSK Inc. in Japan It is driven by DC motors with low noise linear amplifiers. The specifications of the table are listed in table1.

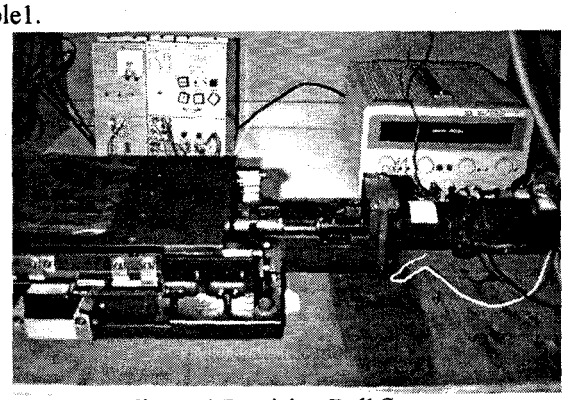

Figure 1 Precision Ball Screw

Table 1. Table Specifications

Stroke	200*mm*
Pitch	2*mm*
Repeatability	0.001*mm*
Backlash	0.001*mm*
Resolution	10*nm*

0-7803-3970-8/97 $10.00 © 1997 IEEE

The PID controller used in this study can be expressed by the following equation:

$$u(k) = K_p e(k) + K_i \sum_{m=1}^{k} e(m) + K_d (e(k) - e(k-1))$$

The goal of the controller design is to find the set of the gains that minimize the response time and control power. The fitness function is defined as follows:

$$F_t = \left[t\left(a\sum_{k=1}^{n} e^2(k) + b\sum_{k=1}^{n} u^2(k) \right) \right]^{-1} \quad a, b \in \text{constant}$$

The time weighted fitness function guarantees the response of the table to be as fast as possible. The control power will be minimized at the same time.

The genetic algorithm would evaluate these fitness values and generate the next generation of the PID pairs by selection, crossover and mutation. And these procedures repeated until the suitable PID gain be found.

The parameters of genetic algorithm are listed in table 2.

Table 2. The parameters of GA

Population Size	20
Crossover Rate	1.0
Mutation Rate (Exponent Part)	0.001
Mutation Rate (Mantissa Part)	0.01

The termination condition for the proposed real-coded genetic algorithms is defined as the best fitness function value remained steady for designated generations.

IV. Experimental Results

The best fitness values vs. time is in Figure 2. The best step response of the 133rd generation is in Figure 3. The control commands in Figure 4 has the same smooth characteristic as the results of the XY table. The steady state error is zero from Figure 5. The friction force is compensated successfully in the experiment shown.

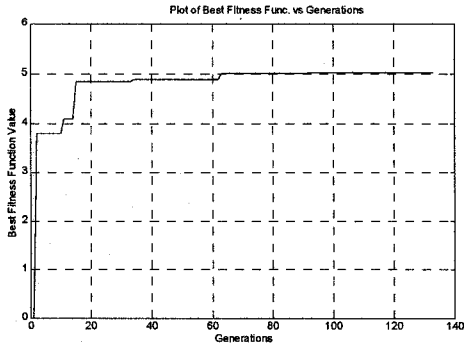

Figure 2 Best Fitness Func. Vs Generations

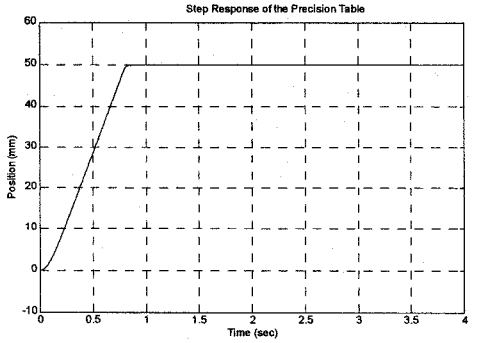

Figure 3 Step Response of The Precision Tables

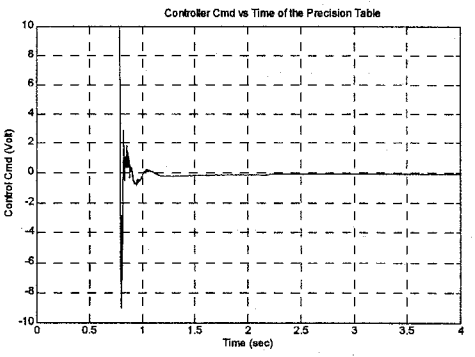

Figure 4 Controller Output Vs Time

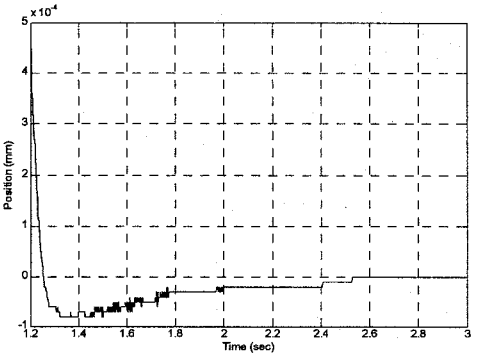

Figure 5 Position Errors Vs Time

V. Conclusion

In this paper, a real-coded genetic algorithm is proposed and tested by implementation on three different examples successfully. The PID gains found in these examples perform well in point-to-point (PTP) precision positioning. The friction effect is eliminated through the genetic algorithm and zero steady state error is achieved. The smooth control commands generate from the control gains found are very important to lengthen the life of the actuators.

VI. Reference

[1] D. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Reading, MA: Addison-Wesley, 1989.

[2] M.O. Odetayo and D. R. McGregor, "Genetic Algorithm for Inducing Control Rules for a Dynamic System" in Proceedings of the third International. Conference on Genetic Algorithms. Morgan Kaufmann, pp. 177-182, 1989.

[3] H. M"uhlenbein, M. Schomish and J. Born, "The Parallel Genetic Algorithm as Function Optimizer", Proceedings of the forth Conference on Genetic Algorithms. Morgan Kaufmann, pp. 271-278, 1991.

[4] Wright, "Genetic Algorithms for Real Parameter Optimization", in Foundations of Genetic Algorithms, G. J. E. Rawlins (editor), Morgan Kaufmann, San Mateo, CA, pp. 205-218, 1991.

[5] M. Caudill, "Evolutionary Neural Networks", AI Expert, pp. 28-32, March 1991