Proceedings of the 7th WSEAS International Conference on Evolutionary Computing, Cavtat, Croatia, June 12-14, 2006 (pp28-32)

Evolvable Hardware a New Approach for Control Design

TATIANE JESUS DE CAMPOS JOSÉ RAIMUNDO DE OLIVEIRA MARIO JUNGBECK
School of Electrical and Computer Engineering
State University of Campinas
Av. Albert Einstein - 400 - 13083-852, Campinas
SP, BRASIL

Abstract: - Evolvable hardware (EHW) is a technique introduced to automatically design circuits where the circuit configuration is carried out by evolutionary algorithms. This paper shows that the evolutionary approach can discover the concept and learning behavior of the control systems. The system tested is a nonlinear pendulum. Experiments with different processes indicate that the gains obtained with evolvable hardware may provide better responses than those obtained by the classical PID controller whose gains are set by Genetic Algorithms. In this paper the Integral of the Square Error (ISE) criteria was used as performance index to be minimized through applying of Genetic Algorithms and Evolvable Hardware controllers. Analysis and simulation of nonlinear pendulum is provided to demonstrate that the proposed evolvable controller has excellent results.

Key-Words: - Evolvable Hardware, Control Systems, PID Controller, Evolvable Controller.

1 Introduction

Evolutionary algorithms are search and optimization procedures that find their origin and inspiration in the biological world, that one built on the key concept of Darwinian evolution. The Darwinian theory of evolution emphasizes the survival of the fittest in a dynamic environment, [1][2].

The possibilities for automatic design of electronic circuits using evolutionary algorithms have been explored in the analogue and digital domains [3]. After some successful applications of evolutionary algorithms to physical design (partitioning, placement and routing), now the same techniques are being considered also for structural design [4]. The application of evolution-inspired formalisms to hardware design and synthesis leads to the concept of Evolvable Hardware.

Evolvable Hardware is a new field at the confluence of Reconfigurable Hardware, Automatic Design, Artificial Intelligence and Autonomous Systems [5]. Evolvable Hardware is a reconfigurable hardware whose configuration is under the control of an evolutionary algorithm. Usually, genetic algorithm and genetic programming are used as evolutionary processes. The evolutionary approach to circuit synthesis is a powerful technique, which could provide innovative solutions to hard design problems [6].

Starting from a previous experience on the synthesis in hardware of a digital non-linear function and evolvable hardware systems [7], the work reports a comparison between traditional (linear PID controller) and evolvable methods (evolvable hardware) applied to the problem of control systems. The following solutions have been considered. Linear PID controller, which is the reference method. This is a commonly used method, with a well established theoretical background [8]. This is also a simple method which requires a very small silicon area. Evolvable hardware controller works differently. It works from the bottom up approach, adding components together to make partial solutions to the design problem, which are in turn combined and tinkered with, until the solution meets the design criteria [3]. This approach attempts to find correlations between sets of components that consistently improve the behavior of a circuit with respect to the problem at hand.

The paper presents a new method for control design using evolvable hardware and is organized as follows. Section 2 presents the description of the case study. Section 3 explains our method, presents a comparison between the approaches and experimental results. Section 4 gives the discussions and results are summarized.

2 Description of the Case Study

2.1 Nonlinear Pendulum Application

The nonlinear damped pendulum shown in Figure 1 was used as a test bench for the proposed controller. The equation of motion of the pendulum can be written in the form:

$$ml\ddot{\theta} + kl\dot{\theta} + mg\sin(\theta) = \tau \qquad (1)$$

where m = 0.210Kg is the mass, g = 9.82m/s2 is the acceleration of gravity, k = 0.3Nms/rad is the viscosity and l = 0.285m is the length of the pendulum. The Matlab/Simulink programs were used to build model, perform simulation and evaluate control performance of the controlled system, Figure 2.

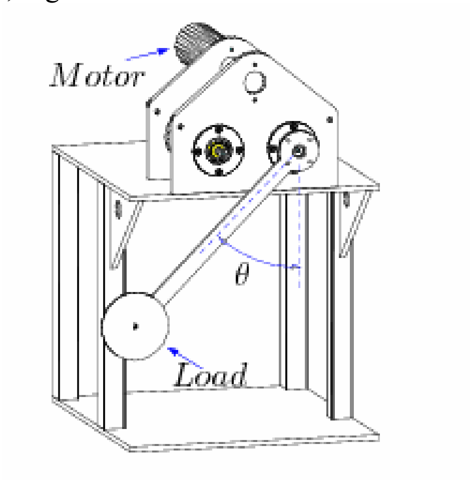

Fig. 1 - Nonlinear Simple Pendulum

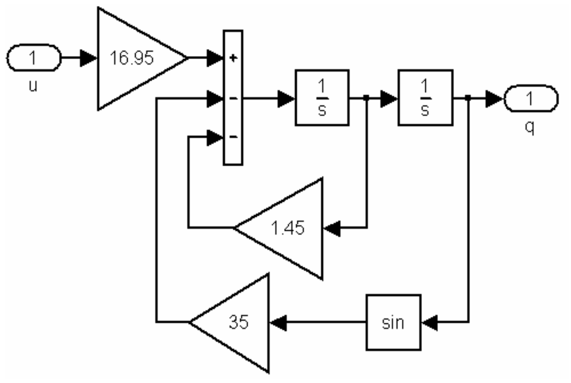

Fig. 2 - Matlab/Simulink model

2.2 Linear PID Controller

Proportional-Integral-Derivative (PID) controllers have been widely used in the field of the industrial process control due to its simplicity and robust performance in wide operation scenarios. Moreover, the PID controllers have simple structures and are relatively simple to maintain and tune. Therefore, it is still attractive to design discrete-time control systems with PID control structures. Discrete PID controller can be implemented in many different ways. Different structures correspond to different controllers [9]. The most common sampled data PID structures are described as:

Proportional Controller

$$u(n) = K_p e(n) \qquad (2)$$

Proportional-Derivative Controller

$$u(n) = \left[K_p + \frac{K_D}{T_S}\right]e(n) - \frac{K_D}{T_S}e(n-1) \qquad (3)$$

Proportional-Integral Controller

$$u(n) = u(n-1) + q_a e(n) + q_b e(n-1) \qquad (4)$$

$$q_a = K_p + \frac{K_I T_S}{2}$$

$$q_b = -K_p + \frac{K_I T_S}{2}$$

Proportional-Integral-Derivative Controller

$$q_0 = K_p + \frac{K_D}{T_S} + \frac{K_I T_S}{2} \qquad (5)$$

$$q_1 = -K_p - 2\frac{K_D}{T_S} + \frac{K_I T_S}{2}$$

$$q_2 = \frac{K_D}{T_S}$$

where e(t) denotes the control error signal given by

$$e(n) = r(n) - y(n) \qquad (6)$$

r(t) is the reference signal given by a piecewise constant component and y(t) is the system measured variable. KP , KD and KI are the proportional, integral and derivative gains, respectively. TS is the sample time.

The tuning of control parameters in PID control law (5) is important, since the performance in the control depend strongly on them. However, in some plants the tuning can be highly time consuming, in addition, aging and nonlinear effects can lead the controller parameters that are far from the optimal. Because of this, following the work of Ziegler and Nichols, a variety of PID tuning methods have been developed, which procedures are, however, in many cases nontrivial [9].

In the process industry there are three main classes of tuning methods: time response tuning, time domain optimization methods and frequency domain shaping. Today the framework for controller is better known and several control objectives can be incorporated to performance indices(e.g. reference tracking, rejection of supply disturbances, load disturbances and measurement noise), [10].

In this paper the Integral of the Square Error (ISE) criteria was used as performance index to be minimized through applying of Genetic Algorithms and Evolutionary Hardware controllers, as illustrated in the next section.

$$ISE = \int_0^T e(t)^2\, dt \qquad (7)$$

2.3 The Evolutionary Controller

The class of evolution algorithm most commonly used in evolvable hardware is the Genetic Algorithm (GA). The GA used operates on a fixed size population of fixed length binary strings called chromosomes. Each chromosome encodes a common set of parameters that describes a collection of electronic components and their interconnections, thus each set of parameter values represents an electronic circuit. The set of all possible combinations of parameters values defines the search space [3]. Every parameter set in the search space encodes a unique candidate circuit design which is associated with a genetic code or chromosome. The chromosome is called the genotype and the circuit is called the phenotype.

The first step of evolutionary synthesis is to generate a random population of chromosomes. In extrinsic evolution, the chromosomes are then converted into a model that gets simulation (by a circuit simulator such as SPICE) and produces responses that are compared against specifications. Preparation for a new iteration loop involves generation of a new population of individuals from the pool of the best individuals in previous generation. The fitness value of each genotype is evaluated. The genetic operators used in this work are extracted from standard GA procedure which includes selection using roulette wheel, with a strategy called elitism, crossover and mutation. In this work, the rates for crossover and mutation are chosen as 100% and 5% respectively. This was chosen through experimentation as it provides sufficient solution diversity. One point crossover recombines two chromosomes by choosing a position at random along the chromosome and swapping every bit beyond this point between the strings. Point mutation independently inverts each bit in the chromosome according to a probability.

The new population is now complete and the algorithm then iterates the steps of evaluation, selection and variation until a circuit that functions adequately is found or a pre-specified number of generations are completed.

2.3.1 Encoding

The genotype determined which of the boolean functions of Figure 3 was instantiated by each node and how the nodes were connected. The linear bit string genotype consisted of 100 segments, each of which directly coded for the function of a node and the sources of its inputs, as shown in Table 1.

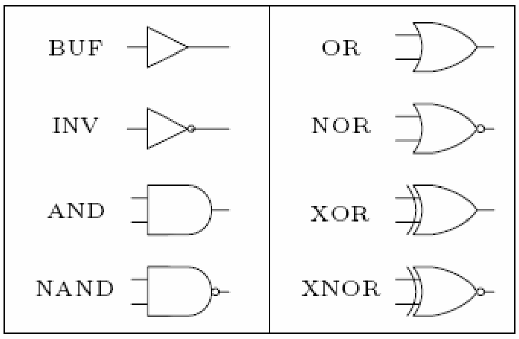

Fig. 3 - Node Functions

Bits	Meaning
0-2	Node Function
3-8	Length to first input
9-14	Length to second input

Table 1- Genotype segment for one node

The source of each input was specified by counting backwards along the genotypes a certain number of segments and four fixed inputs. The genotype is a symbolic netlist. Each row represents a gate, with its type and from where to connect its two inputs. Connections can only be made to lower numbered gates (or the circuit inputs). The straight forward encoding scheme enumerates all the gate types using three bits. An initial population of 12 individuals is generated randomly according to the coding rules.

The topology is specified by a list of component types together with their terminal nodes (PSPICE netlist). Unconnected components (gates) enable us to have variable number of components in a circuit, though the size of the chromosome is fixed.

2.3.2 Fitness Function

Every solution obtained by the GA was evaluated. The fitness function evaluates the evolved circuit in terms of their behavior. In our experiment a fitness function based on the ISE criteria was considered as performance index to be minimized.

2.4 Genetic PID Controller

The basic idea in genetic PID is the definition of a chromosome with the PID controller gains in the genetic information, Figure 4.

$$\mathbf{x} = (K, \tau_d, \tau_i) = \underbrace{010\cdots00}_{K}\underbrace{101\cdots01}_{\tau_d}\underbrace{111\cdots10}_{\tau_i}$$

Fig. 4 - Gain chromosome

Each chromosome is tested and evaluated. The set of gains that allow a better performance of the controller will have greater chances to inherit their genetic profile.

3 Evolving Controllers Circuits

In our investigation, the controller circuit is applied in a closed loop system given in Figure 5. This figure shows the continuous process given by the pendulum model, the zero order holding (Z.O.H) element and the saturation block.

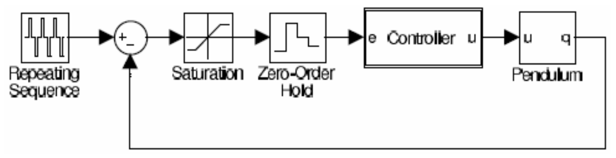

Fig. 5 - The block scheme of closed-loop system

Experiments with one different plant were performed for the evaluation of the Evolvable Hardware circuit controller. The results are compared with the Proportional controller obtained through the GA PID tuning. A MATLAB coded Analog-to-Digital Converter (ADC) converts input error signal, in the range of −1 to 1, into a discrete 4-bit digital representation, and the reverse operation is performed by the Digital-to-Analog Converter (DAC). The results presented are obtained by the simulation of the controlled systems implemented using Matlab/ Simulink for the GA-PD controller and Matlab/Simulink plus PSPICE for the EHW controller.

3.1 The 4-bits Controller System

The 4-bit control system experiment considered the digital representation of the error signal described by the MATLAB code presented in Eq. 8. This operation was performed by the ADC converter.

$$e_D(n) = dec2bin(round(15(e(n)+1)/2),4); \quad (8)$$

The reverse operation performed by the DAC was defined as:

$$u(n) = (bin2dec(u_D(n))/7.5-1); \quad (9)$$

The control error was defined as defined in Eq. 6, the sample time, TS, was set to 1ms and the reference signal, r(t), was defined as

$$r(t) = \frac{\pi}{2}[\mu(t) - \mu(t-8) - \mu(t-16) - \mu(t-24)] \quad (10)$$

where μ(t) is the step function.

3.1.1 Proportional Controller and Evolvable Hardware

The Proportional controller was performed by the MATLAB code described in Eq. 8, Eq. 9, Eq. 11 and Eq. 12.

$$u_{KP} = satlins(K_P(bin2dec(e_D(n))/7.5-1)); \quad (11)$$

$$u_D = dec2bin(round(15(u_{KP}+1)/2),4); \quad (12)$$

The Evolvable hardware controller evolved as a combinational circuit with 4 input and 4 output bits as shown in Figure6.

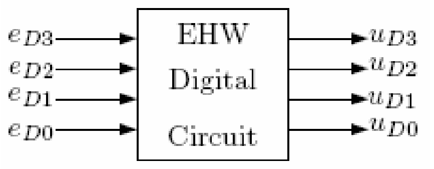

Fig. 6 - 4-bits EHW Controller

The evolution and simulation results are shown in Table 2, Figure 7, Figure 8 and Figure 9, respectively.

Controller	ISE
PD-GA	3.8074 ($K_P = 14.1$)
EHW	3.3043

Table 2 - Best individual of the GA for each controller

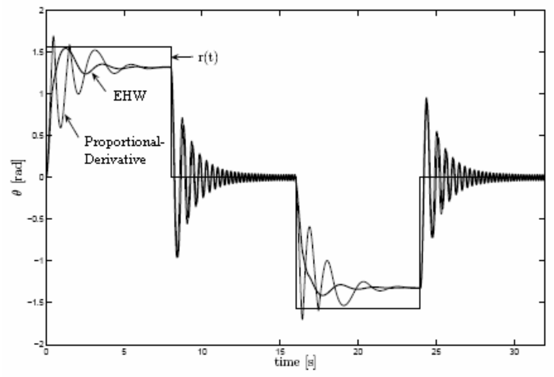

Fig. 7- Comparison of plant-outputs for the P-GA and EHW Controller

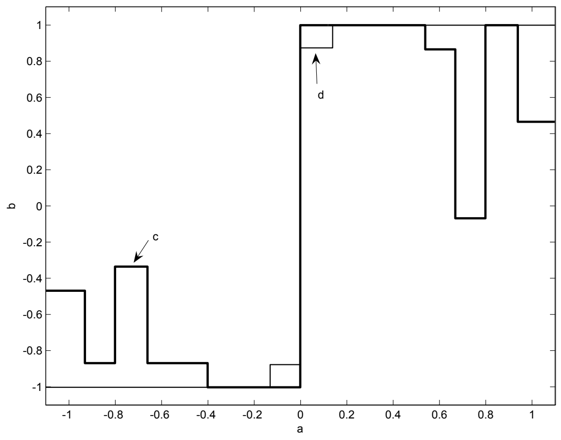

Fig. 8 - Comparison of Controls Functions

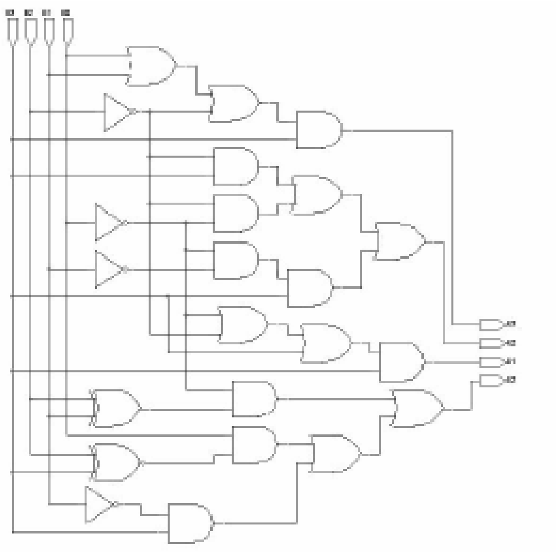
Fig. 9 - Evolution circuit for 4-bits Controller System

4 Conclusion

This work focused on design and implementation of an evolvable hardware for control system. Some relevant properties of the model have been investigated. The model was used to simulated position control of a pendulum with a PID controller, where tuning method is a genetic algorithm, and an Evolvable Hardware Controller. This approach is proved capable of automated design of logic circuits for controllers. As a result, it has potentials to provide innovative, efficient and better modules along with new ideas, principles and design formulas for human designers.

But in order to update such potentials of evolutionary design approaches, it is still an open problem how to automatically extract new rules and knowledge from an evolution process and its results (e.g. stability and robustness).

The simulation results obtained are promising and at the same time have exposed us to the complexities involved in the design process.

References:

[1] C. Darwin, *On the Origin of Species by Means of Natural Selection*. London, UK: John Murray, 1859.

[2] R. Dawkins, *The Blind Watchmaker*. New York, NY, USA: Norton, 1987.

[3] T. G. W. Gordon and P. J. Bentley, Handbook of Innovative Computational Paradigms - (Invited Book Chapter, to appear), A. Zomaya, Ed. Springer, 2005.

[4] E. Damiani and V. Liberali, "On-line evolution of fpga-based circuits: A case study on hash functions," in 1999 NASA/DoD Conference on Evolvable Hardware (EH'99). Pasadena, CA, USA: IEEE Computer Society, 1999, pp. 26–33.

[5]A. Stoica, D. Keymeulen, R. Zebulum, M.I.Ferguson, T. Daud, and T. Arslan, Evolvable Hardware for Autonomous Systems. Seattle, Washington, USA: 2004 NASA/DoD Conference on Evolvable Hardware (EH'04), 2004.

[6] A. Thompson, I. Harvey, and P. Husbands, "Unconstrained evolution and hard consequences." in Towards Evolvable Hardware. Lausanne, Switzerland: Lecture Notes in Computer Science, 1996, pp. 136–165.

[7] T. J. Campos, J. R. Oliveira, and M. Jungbeck, "Programação genética aplicada ao desenvolvimento de hardware evolutivo," in Anais do XV Congresso Brasileiro de Automática - CBA2004, Gramado, RS, Brasil, 2004.

[8] M. Chiaberge, J. J. Merelo, L. M.Reyneri, A. Prieto, and L. Zocca, "A comparison of neural networks, linear controllers, genetic algorithms ans simulated annealing for real time control," in Proc. ESANN 94, European Symposium on Artificial Neural Networks. Brussels, Belgium: Lecture Notes in Computer Science, 1994.

[9]G. C. Goodwin, S. F. Graebe, and M. E. Salgado, Control System Design. Prentice Hall, 2001.

[10] T. H. K. J. Aströn, Automatic Tuning of PID Controllers. Instrument Society of America, 1988.