FPGA-Based Embedded System Architecture for Micro-Genetic Algorithms Applied to Parameters Optimization in Motion Control

Arturo Yosimar JAEN-CUELLAR[1], Luis MORALES-VELAZQUEZ[1], Rene de Jesus ROMERO-TRONCOSO[2], Roque Alfredo OSORNIO-RIOS[1]
[1]*HSPdigital-CA Mecatronica, Facultad de Ingenieria, Universidad Autonoma de Queretaro, Campus San Juan el Rio, Rio Moctezuma 249, 76807, Queretaro, Mexico*
[2]*HSPdigital-CA Telematica, DICIS, Universidad de Guanajuato, Carr. Salamanca-Valle km 3.5+1.8, Palo Blanco, 36700 Salamanca, Guanajuato, Mexico*
raosornio@hspdigital.org

***Abstract*—Meta-heuristic techniques are powerful tools used to find an optimal solution for complex problems to which classical techniques find difficult to solve. The features among all the meta-heuristic techniques are the high amount of computational resources spent on their implementation and the computing effort generated on their execution. For this reason, many works have proposed their use on the base of software methodologies without achieving online or real-time performance. In the present work, two strategies that implement the Genetic Algorithms are presented by using the micro-population concept with the objective of reducing computational resources, increasing the heuristic search speed, and providing simplicity in its design. Both strategies are implemented in hardware architecture; the first, as a software strategy in a proprietary embedded processor, the second, as a hardware co-processor unit. In order to validate the proposed approaches, several tests to optimize a motion controller in a servo system are presented and compared with a classical tuning technique.**

***Index Terms*—Control design, Genetic algorithms, Field programmable gate arrays, Microprocessors, Servo systems.**

I. INTRODUCTION

During the last decades, the study of modern techniques based on heuristic search, called meta-heuristics, has captured more attention than the study of classical techniques based on gradients to solve a wide variety of problems in many research fields [1-2]. Some classical techniques provide optimal results to problems whose design space and constraints can be described by a convex function; whereas modern optimization techniques can handle functions with convexity, non-convexity, linearity, and non-linearity features, meaning that the gradient information is not required. Hence, through these techniques, concepts and algorithms are applied so as to develop a searching method to provide optimal solutions to a problem that would be a challenging task to be solved if classical techniques were used. Moreover, meta-heuristic techniques provide solutions even for problems which require multi-objective optimization [3]. These techniques can be designed and applied where there is poor knowledge of a particular problem; for instance, in problems in which the dynamic model is not available or is not known. The meta-heuristic techniques explore promising regions of the design space, containing high-quality solutions, and escape from the local optimal solutions [4]. Considering all the advantages of the meta-heuristic techniques to solve complex problems, it is easy to see why they are widely used in optimization problems. Although the importance of implementing meta-heuristic techniques to perform parameters optimization in online applications should be highlighted, they required many computational resources.

Many works have presented studies of the meta-heuristic techniques such as the Genetic Algorithms (GA), the Particle Swarm Optimization (PSO), the Ant Colony Optimization (ACO), and the Simulated Annealing (SA), among others [5-8]. Other investigations report comparative studies of their implementation, explaining their features, benefits, advantages and disadvantages [1], [4], [9]. Among all the different heuristic techniques, the GA is the most popular due to its resemblance with nature behavior based on natural genetics and selection [10-12]. They have been intensively studied and applied in many optimization problems demonstrating to have obtained better results than classical techniques [13-16]. However, the Standard GA (SGA) has the drawback of pre-maturity and stagnation, while an optimal solution is being looked for, not to mention the huge quantity of computational resources required to its implementation [17-19]. Nevertheless, there is a necessity of online implementations that can show the technique effectiveness [20-21]. Although alternative solutions based on hardware implementations are being considered, they still need improvement in order to overcome the aforementioned drawbacks [22-27]. Therefore, it would be desirable to develop an architecture that allows the implementation of the GA in embedded systems for the online optimization. Recently, the micro-population concept coupled to the GA, known as the Micro-Genetic Algorithms (MGA), has drawn the attention of scientists due to its advantages over the SGA. This methodology reduces the computational resources required for a digital implementation, and provides appropriate solutions to many optimization problems [28-29]. These particularities make the MGA technique an excellent candidate for an online

This work was partially supported by CONACyT scholarship 331141 and by project SEP-CONACyT 222453-2013.

Digital Object Identifier 10.4316/AECE.2015.01004

implementation through hardware-software strategies.

This paper presents the implementation of the MGA scheme for the online parameter optimization of motion system controllers. This scheme compares the implementation of two strategies of the MGA algorithm; one based on software (SW), and the other based on hardware-software co-design (HS). The SW approach is a description of the heuristic technique, mainly coding the genetic operators into a proprietary embedded processor. Meanwhile, in the HS approach the genetic operations are implemented in a hardware architecture described as a co-processor unit which executes genetic actions and makes use of the embedded processor as a data-flow control unit for the optimization process. The embedded processor and its peripherals are described under a hardware description language (HDL) and synthesized in a low-cost field programmable gate array (FPGA) implemented in a proprietary board, conforming a complete embedded system. The MGA scheme was used instead of other heuristic techniques due to its simplicity and compatibility with digital systems. Besides, the micro-population concept allows increasing the convergence rate and reduces considerably the computational resources required [30]. The above mentioned schemes were used in this investigation for the online optimization of the PID controller parameters used in an industrial servo system. Several experimental tests were performed to validate the proposed schemes and they were compared with the parameters obtained by using a classical tuning technique to show the improvement reached.

II. GA Overview and the Micro-population Concept

The GA is a powerful technique used for heuristic search in optimization problems; this technique is based on the Darwin's theory of survival of the fittest [11], [31-32]. Yet, the SGA scheme could not be the most appropriate in solving complex problems because it requires many function evaluations implying a high computational effort and deriving in a slow convergence [22-28]. To overcome these drawbacks the MGA scheme was defined to use a very small population [29]. It is applied to prevent both the premature convergence and the problem of stagnation in the SGA [28]. In this work, the MGA scheme is proposed as a process that possesses simplicity and minimizes the computational cost allowing the computational speed to be increased. Despite the benefits mentioned, global optimal solutions are not guaranteed, but for the purposes of this work, this scheme provides appropriate near optimal solutions in a short time.

In the SGA, the design variables corresponding to the genomes in the natural genetics are represented as binary strings and they are concatenated to form an individual, corresponding analogically to a chromosome in natural genetics; then, a number of individuals conforms the population [11]. In the present paper, the parameters of the motion controller are proposed as the design variables. The specified population size, *Ps*, consists of only four individuals; which are initially generated taking equidistant values in the design range to avoid the generation of individuals closely located when the design range is extensive. The genetic operators consist of selection, crossover and mutation [10]. In this paper, the "tournament selection" is applied considering that the micro-population concept is adopted, as consequence tournaments between individuals require less evaluation time. Then, the selected individuals are classified according to their fitness value, *f=1/J*, typically provided by an objective function, *J*. For this work, the objective function is the sum of the integral of absolute error (*IAE*) and the integral of square error (*ISE*) defined in (1) to (3). Equations (4) and (5) are used to decode an individual from the population. This means that every design variable is converted from its binary string representation to a decimal value, v_{real}, which in this work corresponds to a PID gain. First, the design range is defined by the difference between the maximum, v_{max}, and the minimum, v_{min}, values that the gain can reach. Then, this range is multiplied by the normalization factor K_{ind}. Finally, the v_{min} value is added to the obtained result in order to get a v_{real} value (PID gain) into the design range. The K_{ind} term is a normalization factor, from 0 to 1, obtained by dividing the magnitude from the binary string of the individual, d_i, and the maximum magnitude considering the binary string length, S_l. The *i*=1,2,3, and 4 are the individuals to be decoded.

$$IAE = \int_0^{\infty} abs[e(t)]dt \tag{1}$$

$$ISE = \int_0^{\infty} [e(t)]^2 dt \tag{2}$$

$$J = ISE + IAE \tag{3}$$

$$v_{real} = v_{\min} + (v_{\max} - v_{\min}) \cdot K_{ind} \tag{4}$$

$$K_{ind} = \frac{d_i}{2^{S_l} - 1} \tag{5}$$

The dynamic for the generation of the next population in the iterative process is defined as: the fittest individual is directly copied to the next population; the worst individual is randomly replaced, and by using the remaining individuals in the actual population, through crossover and mutation, the rest of the individuals are generated. The uniform crossover and one-point mutation operations are employed in this work with the purpose of achieving simplicity in the application, diversity of the population, and fast convergence of the methodology. Since the population size is small, the crossover probability, *Cp*, is fixed as a value of one, while the mutation operation is executed according to an adaptive mutation probability, *aMp*, whose value is initialized at 0.4 based on repeated observations of previous experiments. The *aMp* value is decreased linearly to a zero value during the iterative process and its objective is to raise the searching speed at initial iterations. It must be remarked that the mutation operation could be applied to all the individuals with the exception of the fittest one to avoid losing possible potential solutions. A maximum number of iterations, *G*, is defined as the end criterion for the proposed scheme. Fig. 1 shows the general flow chart of the proposed MGA scheme and the implementation of the two strategies developed in this work (SW and HS) are based on it. Additionally, Table I summarizes the parameters specified for the optimization process.

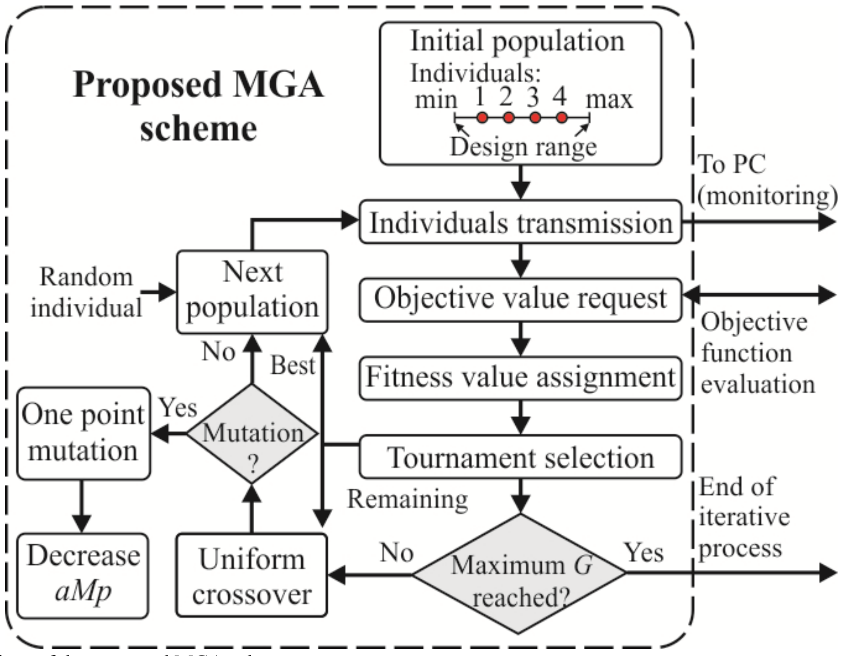

Figure 1. General flow chart of the proposed MGA scheme

TABLE I. SPECIFIED PARAMETERS OF THE METHODOLOGY

Parameter	Value
Maximum number of iterations, *G*	64
Population size, *Ps*	4
Variable string length, S_l (bits)	16
Crossover probability, *Cp*	1
Adaptive mutation probability, *aMp*, (initial value)	0.4
Objective value, *J*	*IAE + ISE*
Fitness value, *f*	*1/J*

III. Hardware and Software Strategies of the Proposed Scheme

In this section, the description of an embedded system for the online parameters optimization by using the proposed MGA scheme is made. The goal is to implement the heuristic technique through the SW and HS strategies, and compare them with a classical tuning technique as it is shown in the following sections.

A. General system architecture

A mechatronic application was selected in order to test the proposed scheme, consisting of the online parameter optimization of a PID controller used in an industrial servo system. Fig. 2 depicts the general block diagram of the proposed methodology based on a hardware-software platform. It may be noticed that three main parts can be identified: the software implementation, the hardware implementation, and the physical system. The implementation of the MGA scheme of Fig. 1 into the proprietary FPGA board is performed through the SW approach, depicted in Fig. 3a, and through the HS approach, shown in Fig. 3b.

The software part is constituted by the user-PC interface, which drives and manages the actions of the optimization process; the data visualization is performed here, as well. The proprietary embedded processor and all the peripherals required for the optimization are described under HDL and are synthetized in a proprietary low-cost FPGA board, which makes up the hardware part. The FPGA usage is justified due to its characteristics such as parallel data processing, versatility, configurability and high performance [33]. For example, motion commands to the PID controller in the FPGA are sent to the motor driver making the system move while the encoder data is passed to the FPGA to be coded and processed. Additionally, the implementation of the MGA schemes (SW or HS) searches for an optimal solution at the same time.

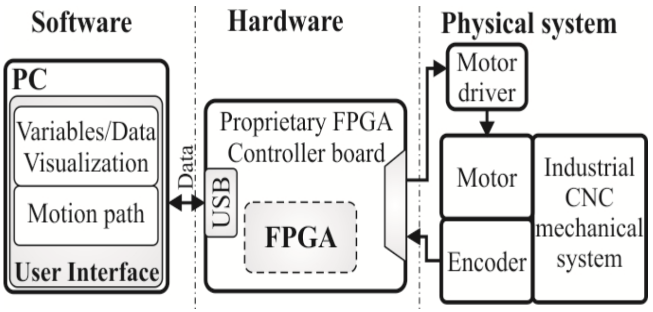
Software
Hardware
Physical system
PC
Variables/Data Visualization
Motion path
User Interface
Data
USB
Proprietary FPGA Controller board
FPGA
Motor driver
Motor
Encoder
Industrial CNC mechanical system

Figure 2. Block diagram of the embedded system for online parameter optimization

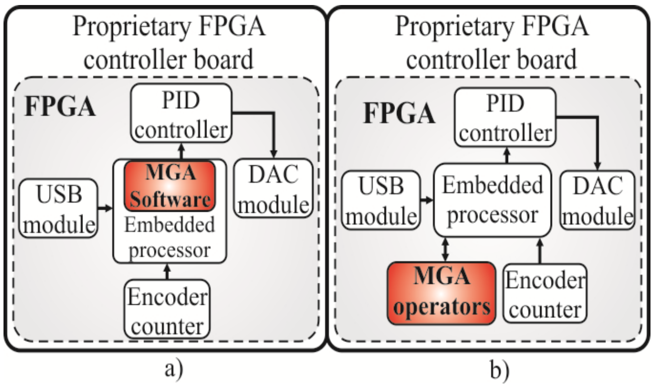

Figure 3. Implementation of the MGA scheme through a) the SW strategy and b) the HS strategy

The sequence of the actions managed by the user-PC interface and executed in the embedded processor is presented in Fig. 4, this sequence is followed in both the SW and the HS approaches.

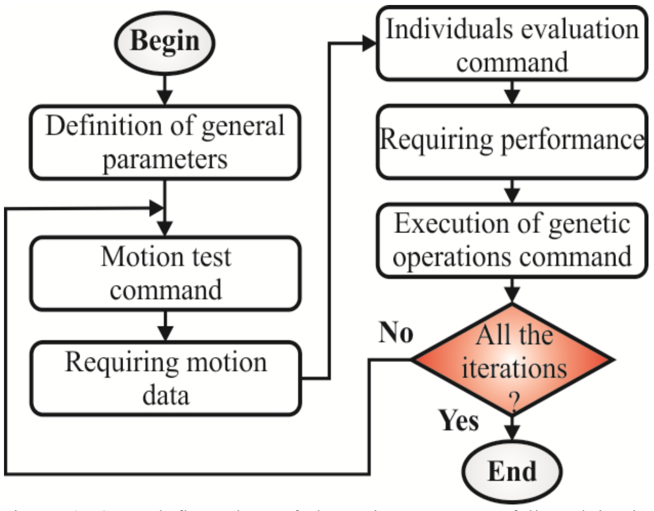
Begin
Definition of general parameters
Motion test command
Requiring motion data
Individuals evaluation command
Requiring performance
Execution of genetic operations command
All the iterations ?
No
Yes
End

Figure 4. General flow chart of the actions sequence followed in the processor

The proprietary FPGA-embedded processor is a reduced instruction set computer (RISC) with partially-decoded instructions. This processor has a set of software development tools in order to be programmed and configured.

B. SW strategy implementation

The implementation of the SW strategy into the embedded processor is according to the flow chart shown in Fig. 5, and the pseudo-code of every event in the genetic process is presented in Table II.

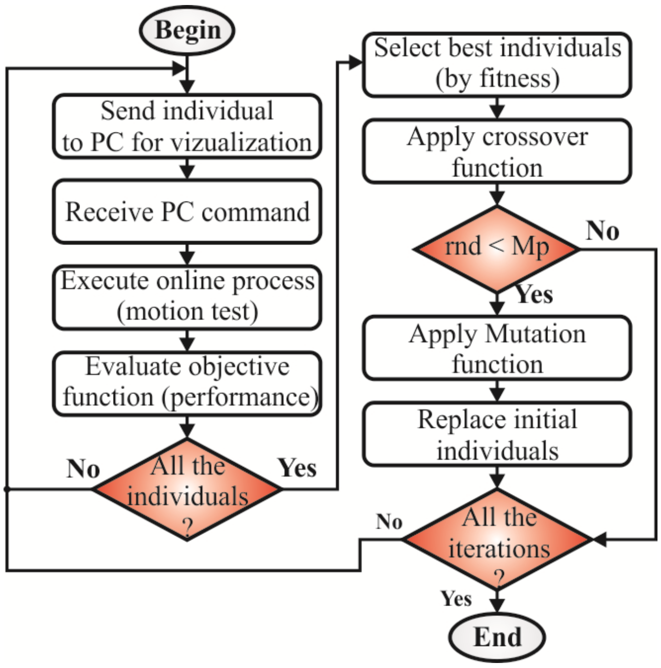
Begin
Send individual to PC for vizualization
Receive PC command
Execute online process (motion test)
Evaluate objective function (performance)
All the individuals ?
No
Yes
Select best individuals (by fitness)
Apply crossover function
rnd < Mp
No
Yes
Apply Mutation function
Replace initial individuals
All the iterations ?
No
Yes
End

Figure 5. Flow chart of the implementation of the MGA scheme through the SW strategy

C. HS strategy implementation

The general block diagram shown in Fig. 6a depicts the implementation of the MGA scheme through the HS strategy, as a co-processor unit that executes the genetic operators giving support to the embedded processor; which is in turn the data-flow control unit of the optimization process, in accordance with what is displayed in Fig. 3b.

TABLE II. PSEUDO-CODE OF THE GENETIC EVENTS

Crossover	**Individuals_selection**

function Crossover{
c_point = randnumber;
head_mask = 0xFFFF << c_point;
tail_mask = 0xFFFF << 16-c_point;
Head1 = Individual1 & head_mask;
Tail1 = Individual1 & tail_mask;
Head2 = Individual2 & head_mask;
Tail2 = Individual2 & tail_mask;
NewIndividual1 = Head1 | Tail2;
NewIndividual2 = Head2 | Tail1; }
	Function Select_individuals{
	for(i=0; i<3;i++){
	for(j=0; j<2;j++){
	if Objective_value[j]>
	Objective_value[j+1]{
	auxiliar = Objective_value[j];
	Objective_value[j]=
	Objective_value[j+1];
	Objective_value[j+1]=auxiliar
	;
	auxiliar2=Individual[j];
	Individual[j]=Individual[j+1];
	Individual[j+1]=auxiliar2;
	}}}
	if Objective_value[0] <
	ObjectiveBest {
	ObjectiveBest =
	Objective_value[0];
	BestIndividual =
	Individual[0];}

Adaptive_mutation	

function Adaptive_mutation{
value=
MaxIterations/mutation_probability;
if ActualIteration=value
decrement mutation_probability;}	

Mutation	

function Mutation{
m_point = randnumber;
m_mask = 1 << m_point;
if randnumber < mutation_probability
NewIndividual = NewIndividual ^
m_mask;}	

The hardware module receives the individuals to be operated (I1 and I2), a random number (RND) which defines the operation points, the value of the mutation probability (MP) to specify whether the operations will be executed or not, and a configuration value (CNF) that defines if the genetic operation is either the crossover or the mutation. Finally, the module provides the operated individual offspring (OFFS), and a flag that indicates the end of the operation (RDY). Since all the genetic processes depend on the generation of random numbers. A random number generator is also developed in hardware so as to provide this function. The module of the random number generator is shown in Fig. 6b, remarking that this module is also used to give support in the SW strategy.

IV. EXPERIMENTAL TESTS AND DISCUSSION

In this section, the case studies for the implementation of the SW and HS strategies are described. Both experiments are described in detail with the purpose of demonstrating the robustness and versatility of the proposed architecture, compared with the classical tuning technique.

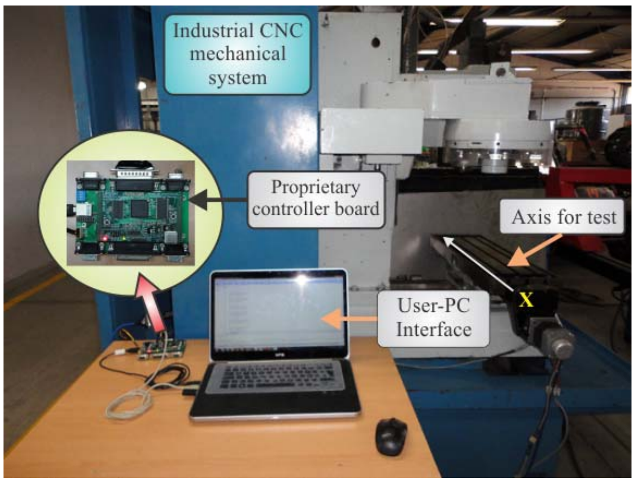
Industrial CNC mechanical system
Proprietary controller board
Axis for test
X
User-PC Interface

Figure 7. Experimental setup and physical system

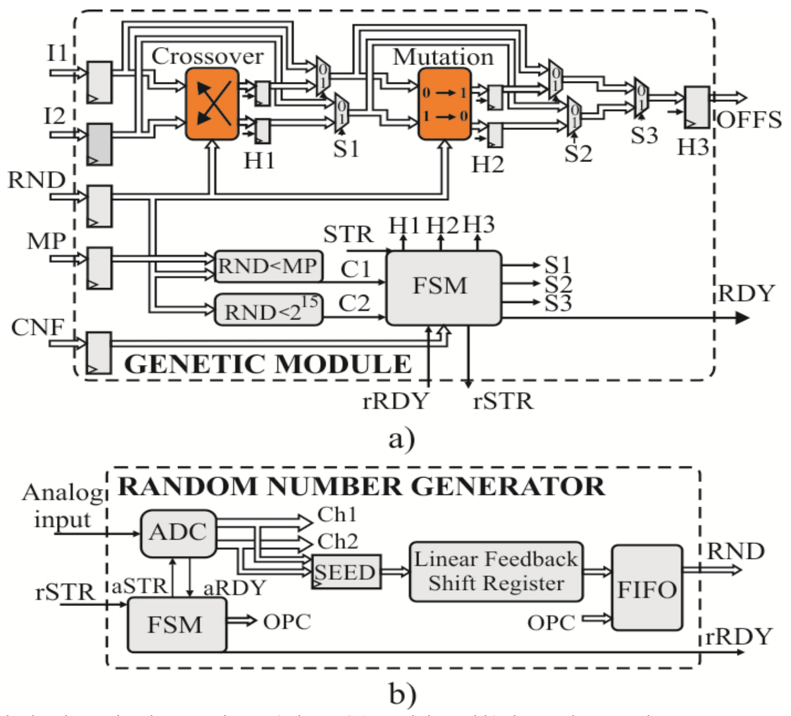

Figure 6. Block diagram of the hardware implementations: a) the MGA module and b) the random number generator

A. *Experimental setup*

The experimental tests are carried out online by using a proprietary board based on a high-performance FPGA technology. The board is a proprietary motion controller of four axes designed to be used in CNC machinery or industrial robots, integrated in two main parts: the system interface media (SIM) and the signal processing unit module (SPU). The SIM consists of the physical connectors, digital inputs/outputs, supplying voltage, analog-to-digital (ADC) converter and digital-to-analog (DAC) converter. Another characteristic is that the SPU includes the external memories and a Spartan 3E-1600 FPGA operating at 48MHz. The proprietary embedded processor is implemented into the FPGA, operating at 24MHz, together with the necessary complementary modules for the embedded system. The user-PC interface is developed using standard C++ language and free libraries in a PC-laptop with an operating system of 64-bits, 16 GB of RAM, an Intel Core i7-6312QM CPU operating at 2.1 GHz. Additionally, the classical tuning method used in the comparison with the proposed optimization is the Gain Phase Margin Method (GPM) presented in [34] and programed in the PC. The servo-system consists of a CNC milling machine of three axes. The x-axis is selected to perform the tests. In this axis a brushless servo motor 3485-ME8137 of MCG with a 4000 counts/rev incremental encoder is used. The servo motor is driven by a brushless PWM servo-amplifier from Advanced Motion Control. The motion test used for the optimization process consists of a step motion reference of 200 counts for a total of 300 samples acquired at 1 kHz. On the other hand, the motion test used to evaluate the performance of the optimized PID gains finally obtained uses a trapezoidal profile for a motion command of 1000 counts forward and a motion command of 1000 counts backward. The specified values of the profile are a maximum velocity of 25600counts/s and ±256000counts/s^2 for the acceleration and deceleration, respectively. Finally, the test bench of each strategy and the GPM tuning consists of 40 tests and the results presented in this work are the average values obtained. The Fig. 7 shows the experimental setup described previously.

B. *Case studies*

In both strategies the general parameters are defined in Table I, and the criterion of the obtained behavior uses the performance indexes specified in (1) and (2). The decoding of equations (4) and (5) to obtain the PID gains for the optimization process considers the design ranges summarized in Table III. These ranges were proposed aiming to make the heuristic search into a wide design space.

TABLE III. PROPOSED DESIGN VARIABLE AND DESIGN RANGE

Design variable	**Design range** $[v_{\min}, v_{\max}]$
Proportional gain, K_p	[1, 5000]
Integral gain, K_i	[0.1, 50000]
Derivative gain, K_d	[0.1,50]

C. *Results and discussion*

The computing times required for each implemented strategy are presented in Table IV. They only consider the events related to the genetic process. Referring to this table, it can be observed that the HS approach executes the genetic process faster than the SW approach since this strategy implements the architecture shown in Fig. 6a as a hardware module; in the meantime, the SW approach implements the genetic process according to the pseudo-code shown in Table II in the embedded processor. The evaluation of the individuals and their selection take the same computing time because these events are performed in the processor. The total time required per generation considers the performance indexes, selection, crossover process, mutation process, random individuals and adaptive mutation probability events. The computing time of the crossover operation and mutation operation in Table IV specifies the time required to apply these operations only between two design variables.

TABLE IV. COMPUTING TIME REQUIRED BY THE STRATEGIES

Computing time	MGA-SW	MGA-HS
Performance indexes	2.86 ms	2.86 ms
Selection	≈ 47.763 μs	≈ 47.763 μs
Crossover operation	168.0 μs	160.0 ns
Crossover process *	1.0 ms	960.0 ns
Mutation operation	453.3 μs	480.0 ns
Mutation process *	2.72 ms	2.88 μs
Random individual	300.0 ns	160.0 ns
Random individuals [a]	900.0 μs	480.0 ns
Adaptive mutation probability	12.7 μs	12.7 μs
Total time per generation	≈ 6.6413 ms	≈ 2.9247 ms

* Time consumed in executing the genetic operations for all the individuals in one iteration of the process. [a]Time spent in the generation of all the individuals randomly replaced.

In the case of the GPM tuning method implemented in the PC offline, it takes approximately 2 seconds to provide the PID gains for the system model. It must be emphasized that due to the inaccuracy of the model, it is necessary to make a manual adjustment to the gains in order to obtain a good performance. The servo system model obtained through the least square method is presented in equation (6).

$$G(s) = \frac{-1.08e-005s^2 - 0.2032s + 568.8}{s^2 + 0.01s - 10} \qquad (6)$$

Table V presents the summary of the resources used to implement the MGA trough the SW and HS strategies into the FPGA. From this table, it can be seen that the complete embedded system was implemented into the device. Therefore, if additional cores and functions were needed, they can still be added.

TABLE V. RESOURCES USED BY THE EMBEDDED SYSTEM

Logic Utilization	Used	Available	Utilization
Number of Slice Flip Flops	6,340	29,504	21%
Number of 4 input LUTs	10,322	29,504	34%
Number of occupied Slices	5,453	14,752	36%
Total Number of 4 input LUTs	10,583	29,504	35%
Number of bonded IOBs	50	250	20%
Number of RAMB16s	1	36	2%
Number of BUFGMUXs	2	24	8%
Number of MULT18X18SIOs	6	36	16%

Fig. 8 to Fig. 10 show the plots of some relevant steps in the optimization process of the MGA implementation. As mentioned before, four individuals are initially distributed, being named the initial seed, and for illustrative purposes only two of them are selected to show the evolution of the variables in the optimization process, specified in Fig. 8 and Fig. 9 as “Individual 1” and “Individual 2.” Fig. 8 shows the performance of the individuals from the first to the final iteration in the SW strategy. Fig. 8a depicts the dynamic response of individuals in the first iteration (notice that the response is far from the reference). After the optimization process, the dynamic response of the individuals at the last iteration is remarkably improved as can be seen in Fig. 8b. The behavior of the mean error along the 64 iterations during the optimization process is shown in Fig. 8c, where this error is significantly reduced against the first iterations. For the time being, Fig. 9 depicts the same plots corresponding to the HS strategy.

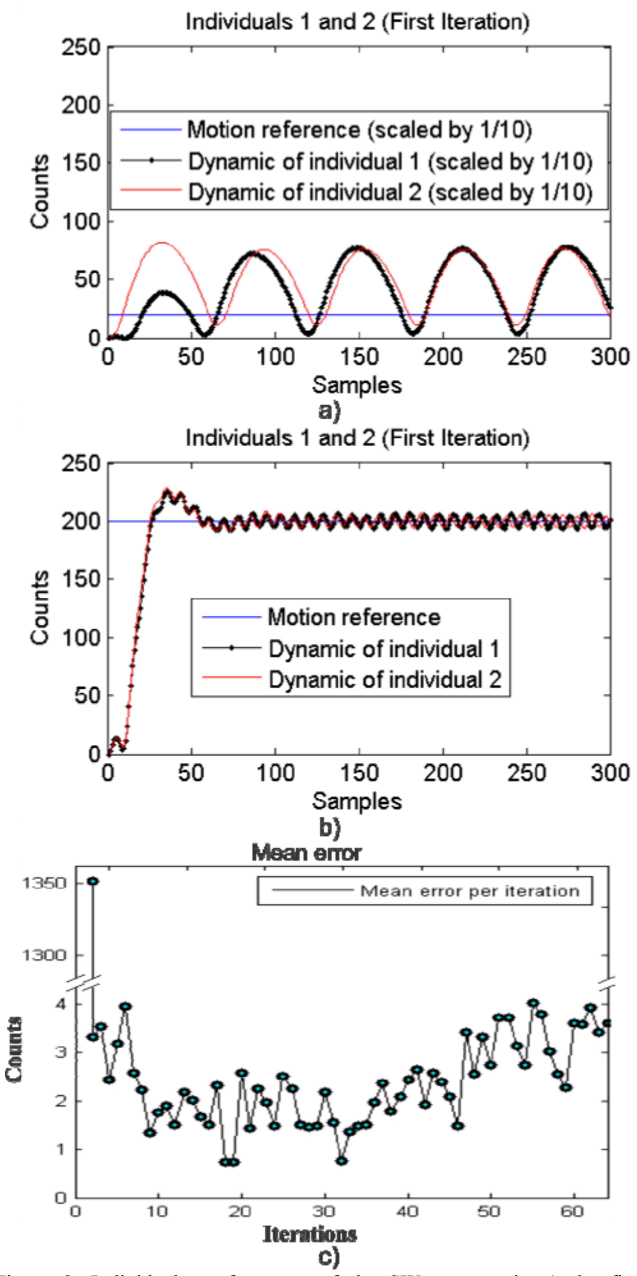

Figure 8. Individuals performance of the SW strategy in a) the first iteration, in b) the final iteration and in c) the mean error along the 64 iterations.

The plots of Fig. 10 show the evolution of the design variables along the 64 iterations of the optimization process. Fig. 10a shows the convergence of the controller gains to their final values in the SW strategy, and Fig. 10b displays the convergence of the gains to their final values corresponding to the HS approach. In both cases, the initial values of the design variables are equally distributed, but the optimal local values of the gains are reached in the iteration 20 for the case of the HS approach and in the iteration 50 for the case of the SW description. The gains obtained for the proportional, integral and derivative parts of the PID are from the SW strategy: 292.82, 30196.09, and 2.1814; and from the HS strategy: 386.39, 31677.05, and 2.65, respectively. The corresponding gains obtained from the GPM tuning are: 250, 29000, and 1.78. The graphical response of the motion test using the motion profile, described in the Experimental setup section, and the gains recently presented are shown in Figs. 11a to 11c. The corresponding tracking errors of every approach are depicted in Fig 12.

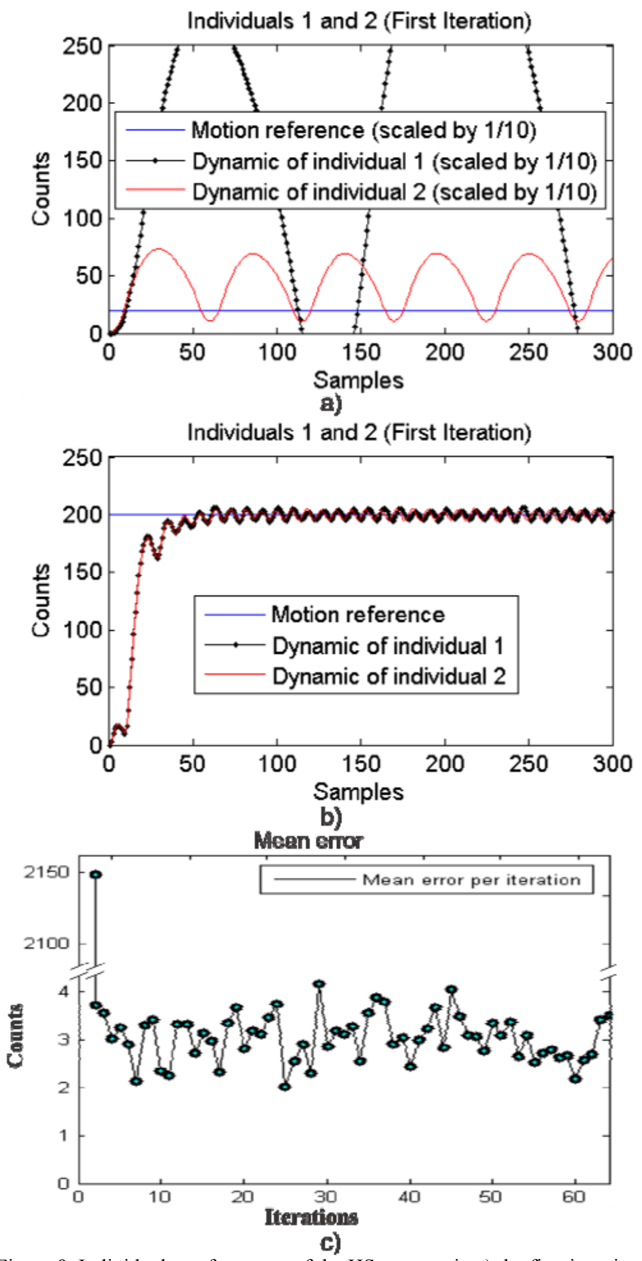

Figure 9. Individuals performance of the HS strategy in a) the first iteration, in b) the final iteration and in c) the mean error along the 64 iterations

Table VI summarizes the values of the *IAE* and the *ISE* of the motion tests. In this table, it can be noticed that the HS approach slightly improves the performance over the SW approach by minimizing the errors, since they are based on the same MGA scheme. Nevertheless, both strategies present a better performance than the GPM tuning.

TABLE VI. IAE AND ISE VALUES OF EACH MOTION TEST WITH PROFILE

Implementation of the MGA	*IAE* (counts)	*ISE* (counts)
SW strategy	2185	18769
HS strategy	2095	18167
GPM tuning	30746	4167484

The positioning control systems used in the manufacturing industry require that the overshoot value remains below 25% and the steady state error falls into the 2% of the reference input, in accordance with the control theory, since those values can affect the quality of the final product [35]. Considering the above mentioned, in the proposed optimization process, every individual is evaluated iteration by iteration for a step response, described in the Experimental setup section. Then, in Fig. 8a the step response of the gains obtained by using equation (4) of individuals 1 and 2 in the first iteration of the SW strategy can be appreciated. A deficient behavior in the response can be noticed in both individuals; the overshoot is over 100% and there is no steady state (oscillatory response), due to the initial distribution of the individuals which can be located far from the optimal local individuals. In contrast, Fig. 8b presents the graphical response in the final iteration, when the individuals have converged to the optimal local solution. The overshoot is reduced to a value of 12% for both individuals and the steady state error is minimized into the required 2%. A similar analysis could be made to Fig. 9 for the case of the HS strategy where there is no overshoot visualized in Fig. 9b and the steady state error is also into the required 2% (final iteration of the process). Therefore, it can be concluded that the gains acquired from the individuals after the optimization process are local optimal gains.

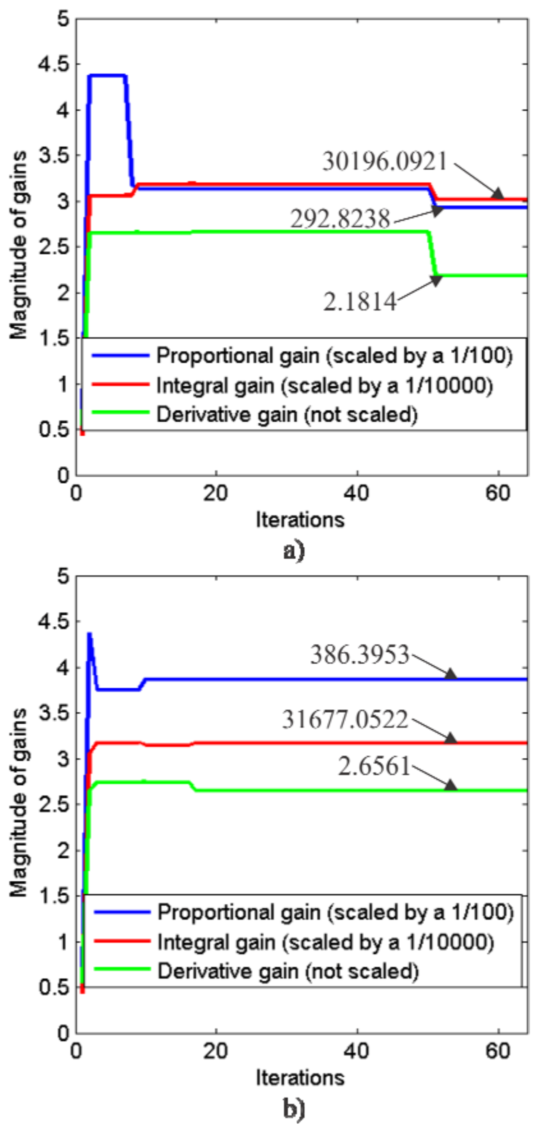

Figure 10. Evolution of the design variables, PID controller gains, using a) the SW and b) the HS approaches

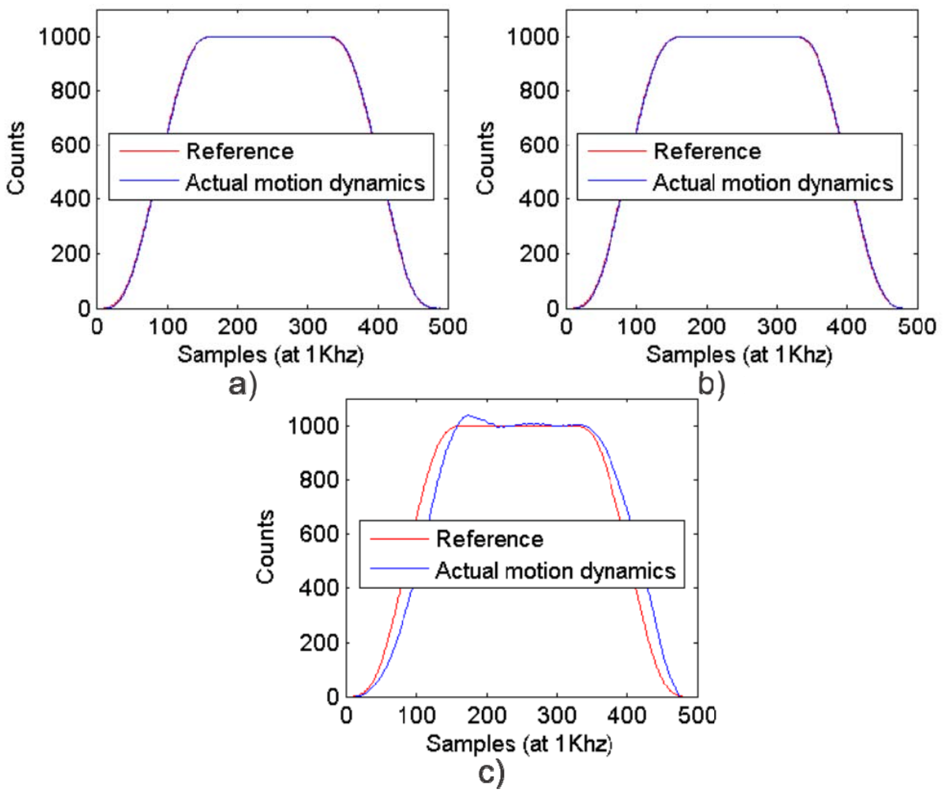

Figure 11. Motion test with profile, reference and actual motion dynamics of a) SW strategy and b) HS strategy, and c) GPM tuning.

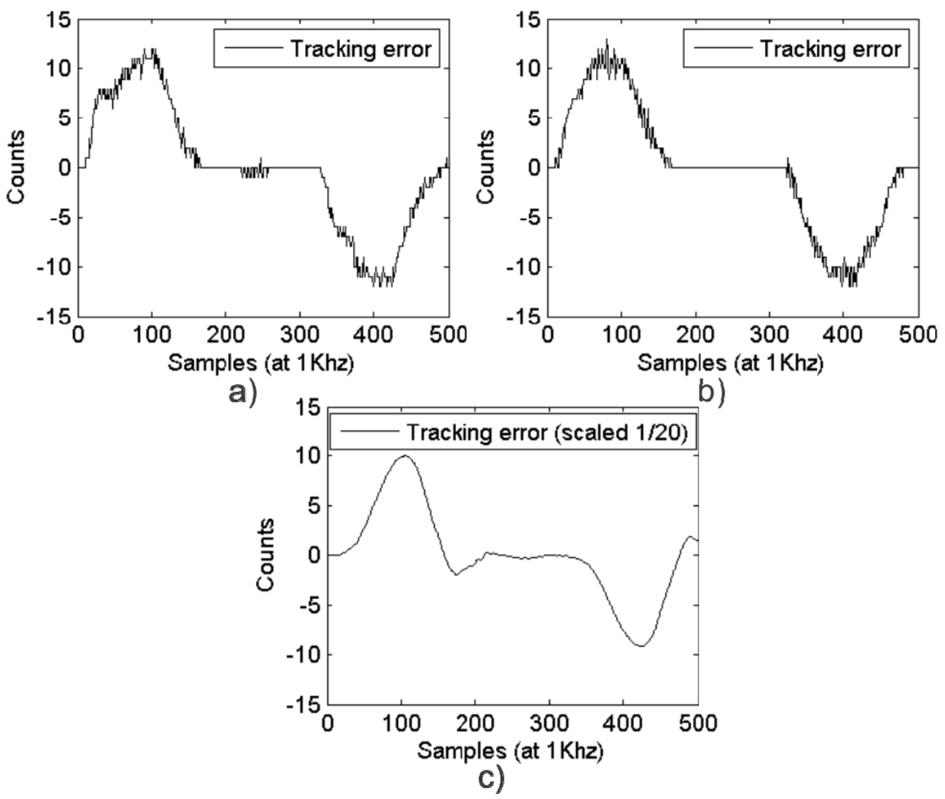

Figure 12. Tracking error of the motion trajectory of a) SW strategy, b) HS strategy, and c) GPM tuning.

The gains evolution of the PID controller, Fig. 10, along the optimization process demonstrates the functionality of the embedded system and the effectiveness in the online implementation of the heuristic technique, since both strategies reach similar values. These graphics show how the initial seeds evolve iteration by iteration, converging to a near optimal solution.

It must be highlighted that the kinematics of the motion profile, as specified in the Experimental setup section, demands a fast response of the controller due to the inertia presented in the physical system for the command reference given. For this reason, the parameters of the controller must be effectively tuned. Then, the obtained parameters through the optimization process in the embedded system demonstrate the effectiveness of the MGA scheme, as shown in Figs. 11a and 11b. From Fig. 11c it is observed that the GPM tuning provides good controller parameters since the performance satisfies the specifications of control theory; in this case, the overshoot value is 3.9%. However, the proposed SW and HS strategies perform better than the GPM tuning since they do not present overshoot value.

Additionally, Fig. 11 indicates that the system reach the reference around the sample 150. Therefore, from Fig. 12a the SW strategy provides controller gains that reach the reference around the specified sample with steady state error of ±2counts from sample 200 to 250. Also, from Fig. 12b the HS strategy provides controller gains that reach the reference around the specified sample without steady state error once the system reaches the reference. Finally, Fig. 12c indicates that GPM tuning provides controller gains with steady state error around the reference into ±2%.

V. Conclusions

This work proposes a new architecture that implements an online parameter optimization of a PID controller in a real system through two strategies, the SW and HS. These strategies reduce the amount of resources demanded by the heuristic process and they attain the computing time required to achieve an online optimization. The online implementation is achieved through the usage of the micro-population concept. Even when the MGA scheme does not provide optimal global solutions, the obtained results demonstrated that local solutions were excellent options to the mechatronic problem defined in this research, when it is compared with a classical tuning method such as the GPM. It can be seen from the obtained results that the mean error is decreased along the iterative process in magnitude enough to consider the MGA scheme as a good architecture, which achieves its goal in tuning optimization of the PID controller in the motion system. Moreover, the convergence time indicates that the proposed architecture is suitable for the online parameters optimization. Finally, the MGA scheme through the SW approach provides flexibility if some modifications are needed to be done quickly; for the time being, the HS approach increases the speed of convergence of the optimization process, and the HS implementation allows having a modular system in which only the required cores can be selected in order to use them in a specific optimization problem, reducing both computing effort and resources.

Acknowledgment

The authors are grateful to the Mechatronics Laboratory and to the Department of Foreign Languages of the *Universidad Autonoma de Queretaro* for their support in the development of this work.

References

[1] S. Panda, and N. P. Padhy, "Comparison of Particle Swarm Optimization and Genetic Algorithm for FACTS-Based Controller Design." Applied Soft Computing, vol. 8, no. 4, pp. 1418–1427, 2008. [Online]. Available: http://dx.doi.org/10.1016/j.asoc.2007.10.009

[2] M. El Semelawy, A. O. Nassef, and A. A. El Damatty, "Design of Prestressed Concrete Flat Slab Using Modern Heuristic Optimization Techniques." Expert Systems with Applications, vol. 39, no. 5, pp. 5758–5766, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2011.11.093

[3] M. W. Bloomfield, J. E. Herencia, and P. M. Weaver, "Analysis and Benchmarking of Meta-Heuristic Techniques for Lay-up Optimization." Computers & Structures, vol. 88, no. 5–6, pp. 272–282, 2010. [Online]. Available: http://dx.doi.org/10.1016/j.compstruc.2009.10.007

[4] K. Hammouche, M. Diaf, and P. Siarry, "A Comparative Study of Various Meta-Heuristic Techniques Applied to the Multilevel Thresholding Problem." Engineering Applications of Artificial Intelligence, vol. 23, no. 5, pp. 676–688, 2010. [Online]. Available: http://dx.doi.org/10.1016/j.engappai.2009.09.011

[5] D. E. Goldberg, "Genetic Algorithms in Search, Optimization, and Machine Learning" Addison-Wesley Longman Publishing Co, Inc., 1989.

[6] J. Kennedy, and R. Eberhart., "Particle Swarm Optimization", pp. 1942–1948. Proc. ICNN, 1995.

[7] M. Dorigo, and C. Blum, "Ant Colony Optimization Theory: A Survey." Theoretical Computer Science, vol. 344, no. 2–3, pp. 243–278, 2005. [Online]. Available: http://dx.doi.org/10.1016/j.tcs.2005.05.020

[8] L. Ingber, "Simulated Annealing: Practice versus Theory." Mathematical and Computer Modelling, vol. 18, no. 11, pp. 29–57, 1993. [Online]. Available: http://dx.doi.org/10.1016/0895-7177(93)90204-C

[9] B. Nagaraj, and N. Murugananth, "A Comparative Study of PID Controller Tuning Using GA, EP, PSO and ACO." IEEE International Conference on Communication Control and Computing Technologies (ICCCCT), pp. 305–313, 2010. [Online]. Available: http://dx.doi.org/10.1109/ICCCCT.2010.5670571

[10] I. J. Graham, K. Case, and R. L. Wood, "Genetic Algorithms in Computer-Aided Design." Journal of Materials Processing Technology, vol. 117, no. 1–2, pp. 216–221, 2001. [Online]. Available: http://dx.doi.org/10.1016/S0924-0136(01)01144-X

[11] S. S. Rao, "Engineering Optimization Theory and Practice", pp. 693–730, John Wiley & Sons Inc, 2009.

[12] L. Di, Z. Lei, and L. Xiang, "PID Parameter Optimization of Shunting and Winch Control System in Coal Transportation Based Online Adaptive Genetic Algorithm." International Conference on E-Product E-Service and E-Entertainment (ICEEE), pp. 1–4, 2010. [Online]. Available: http://dx.doi.org/10.1109/ICEEE.2010.5661509

[13] A. J. A. Nazir, Gautham, R. Surajan, and L. S. Binu, "A Simplified Genetic Algorithm for Online Tuning of PID Controller in LabView." World Congress on Nature & Biologically Inspired Computing (NaBIC), pp. 1516–1519, 2009. [Online]. Available: http://dx.doi.org/10.1109/NABIC.2009.5393665

[14] Z. Weiping, D. Yu, and Z. Hu. "Parameters Optimization for Small Helicopter Highly Controller Based on Genetic Algorithm." In World Automation Congress (WAC), 2012, pp. 1–4, 2012.

[15] K. A. Mohideen, G. Saravanakumar, K. Valarmathi, D. Devaraj, and T. K. Radhakrishnan, "Real-Coded Genetic Algorithm for System Identification and Tuning of a Modified Model Reference Adaptive Controller for a Hybrid Tank System." Applied Mathematical Modelling, vol. 37, no. 6, pp. 3829–3847, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.apm.2012.08.019

[16] H. V. Hultmann-Ayala, and L. dos Santos-Coelho, "Tuning of PID Controller Based on a Multiobjective Genetic Algorithm Applied to a Robotic Manipulator." Expert Systems with Applications, vol. 39, no. 10, pp. 8968–8974, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2012.02.027

[17] L. Hong-Yan, "The Adaptive Niche Genetic Algorithm for Optimum Design of PID Controller." International Conference on Machine Learning and Cybernetics, pp. 487-491, 2007. [Online]. Available: http://dx.doi.org/10.1109/ICMLC.2007.4370194

[18] T. Wu, Y. Cheng, J. Tan, and T. Zhou, "The Application of Chaos Genetic Algorithm in the PID Parameter Optimization." 3rd International Conference on Intelligent System and Knowledge Engineering (ISKE), vol. 1, pp. 230–234, 2008. [Online]. Available: http://dx.doi.org/10.1109/ISKE.2008.4730932

[19] Z. S. Abo-Hammour, O. M. K. Alsmadi, S. I. Bataineh, M. A. Al-Omari, and N. Affach, "Continuous Genetic Algorithms for Collision-

free Cartesian Path Planning of Robot Manipulators", International Journal of Advanced Robotic Systems, vol. 8, no. 6, pp. 14–36, 2011. [Online]. Available: http://dx.doi.org/10.5772/50902

[20] A. Ghanbari, and SMRS. Noorani, "Optimal Trajectory Planning for Design of a Crawling Gait in a Robot Using Genetic Algorithm." International Journal of Advanced Robotic Systems, vol. 8, no. 1, pp. 29–36, 2011. [Online]. Available: http://dx.doi.org/10.5772/10526

[21] X. Wang, T. Lu, and P. Zhang, "State Generation Method for Humanoid Motion Planning Based on Genetic Algorithm." International Journal of Advanced Robotic Systems, vol. 9, no. 23, pp. 1–8, 2012, . [Online]. Available: http://dx.doi.org/10.5772/50918

[22] W. Tang, and L. Yip, "Hardware Implementation of Genetic Algorithms Using FPGA." The 47th Midwest Symposium on Circuits and Systems (MWSCAS), vol. 1, pp. 549–552, 2004. [Online]. Available: http://dx.doi.org/10.1109/MWSCAS.2004.1354049

[23] Y. Chen, and Q. Wu, "Design and Implementation of PID Controller Based on FPGA and Genetic Algorithm." International Conference on Electronics and Optoelectronics (ICEOE), vol. 4, pp. 308–311, 2011. [Online]. Available: http://dx.doi.org/10.1109/ICEOE.2011.6013491

[24] Z. Yan-Cong, G. Jun-Hua, D. Yong-Feng, and H. Huan-Ping, "Implementation of Genetic Algorithm for TSP Based on FPGA." In Control and Decision Conference, pp. 2226–2231, 2011. [Online]. Available: http://dx.doi.org/10.1109/CCDC.2011.5968577

[25] B. Vasumathi, and S. Moorthi, "Implementation of Hybrid ANN–PSO Algorithm on FPGA for Harmonic Estimation." Engineering Applications of Artificial Intelligence, vol. 25, no. 3, pp. 476–483, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.engappai.2011.12.005

[26] G. Mamdoohi, A. F. Abas, K. Samsudin, N. H. Ibrahim, A. Hidayat, and M. A. Mahdi, "Implementation of Genetic Algorithm in an Embedded Microcontroller-Based Polarization Control System." Engineering Applications of Artificial Intelligence, vol. 25, no. 4, pp. 869–873, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.engappai.2012.01.018

[27] H. R. Mahdiani, A. Banaiyan, M. H. S. Javadi, S. M. Fakhraie, and C. Lucas, "Defuzzification Block: New Algorithms, and Efficient Hardware and Software Implementation Issues." Engineering Applications of Artificial Intelligence, vol. 26, no. 1, pp. 162–172, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.engappai.2012.07.001

[28] B. H. Dennis, and G. S. Dulikravich, "Optimization of Magneto-Hydrodynamic Control of Diffuser Flows Using Micro-Genetic Algorithms and Least-Squares Finite Elements." Finite Elements in Analysis and Design, vol. 37, no. 5, pp. 349–363, 2001. [Online]. Available: http://dx.doi.org/10.1016/S0168-874X(00)00052-4

[29] A. C. Coello-Coello, and G. Pulido-Toscano., "A micro-Genetic Algorithm for multi-objective optimization," Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2001(1993), pp. 126–140.

[30] P. C. Ribas, L. Yamamoto, H. L. Polli, L. V. R. Arruda, and F. Neves-Jr, "A Micro-Genetic Algorithm for Multi-Objective Scheduling of a Real World Pipeline Network." Engineering Applications of Artificial Intelligence, vol. 26, no. 1, pp. 302–313, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.engappai.2012.09.020

[31] A. Patelli, and L. Ferariu, "Elite Based Multiobjective Genetic Programing in Nonlinear Systems Identification." Advances in Electrical and Computer Engineering, vol. 10, no. 1, pp. 94–99, 2010. [Online]. Available: http://dx.doi.org/ 10.4316/AECE.2010.01017

[32] A. Rezazadeh, "Genetic Algorithm Based Servo System Parameter Estimation During Transients." Advances in Electrical and Computer Engineering, vol. 10, no. 2, pp. 77–81, 2010. [Online]. Available: http://dx.doi.org/10.4316/AECE.2010.02013

[33] A. Melnyk, and V. Melnyk, "Self-Configurable FPGA-Based Computer Systems." Advances in Electrical and Computer Engineering, vol. 13, no. 2, pp. 33–38, 2013. [Online]. Available: http://dx.doi.org/10.4316/AECE.2013.02005

[34] J. Tal, "Step by Step Design of Motion Control Systems." Galil Motion Control Inc, 1994.

[35] R. A. Osornio-Rios, R. J. Romero-Troncoso, G. Herrera-Ruiz, and R. Castañeda-Miranda, "The Application of Reconfigurable Logic to High Speed CNC Milling Machines Controllers." Control Engineering Practice, vol. 16, no. 6, pp. 674–684, 2008. [Online]. Available: http://dx.doi.org/ 10.1016/j.conengprac.2007.08.004