Automatic Approximation of the Parameters of a PID Controller using an Optimization Algorithm

Holman Montiel Ariza[1], Vicente Reyes Mozo, Henry Montaña Quintero
Universidad Distrital Francisco José de Caldas, FacultadTecnológica
Cll 68 D Bis A Sur No. 49F – 70, Bogotá D.C., Colombia
[1] hmontiela@udistrital.edu.co

***Abstract* -Automated processes require groups of controllers to manipulate the process variables, in order to reduce the margin of error between an established reference value and the value marked by the measurement instruments. One of these controllers is the PID (Proportional, Integral and Derivative) which reduces the error from a feedback given by a sensor. This control strategy depends on three constants, whose values are estimated in different ways depending on the particular characteristics of each process. Sometimes, this feature limits the effectiveness of the controller, since it depends on a mathematical approach and a simulation. An alternative solution to this limitation is proposed in this paper, which allows to estimate the constants of a PID controller automatically without needing to know the process or to perform a previous simulation. The value of the constants was estimated using a bio-inspired optimization algorithm, which is fed back from the process information by means of a data acquisition card. Said card is a physical controller that has the function of regulating an actuator from the information of a sensor.**

Keyword-Evolutionary algorithm, PID, controller, process variable, feedback.

I. INTRODUCTION

The feedback controllers have been widely studied in the area of automation and control, since they perform various operations to reduce the margin of error between an expected value and one measured by a sensor in an automated process [1]. There are several techniques to reduce the margin of error, ranging from the design of controller architectures to approximate mathematical models of the processes to be controlled. However, the effectiveness of these techniques decreases depending on the configuration of the controller, since, when trying to incorporate a controller in different processes, its configuration must be changed [2-3].

Despite this, techniques based on the designer's expertise or a simulation are still implemented, since at an industrial or academic level they are flexible enough to adapt to various types of processes. However, when implementing these techniques, environmental effects are not considered in the process, which causes the controller to be periodically readjusted. This limitation has been partially solved with industrial-type controllers, which have a self-configuration function [2-3].

A disadvantage of implementing this type of controllers is that they have a high cost and increase the number of components that must be used to control a process. This characteristic has allowed exploring other areas to contribute in the search for a solution to this problem, among which is machine learning (4).

Machine learning is studied in different branches of engineering, thanks to its versatility to be implemented in the solution of various types of problems. Normally supervised and unsupervised learning techniques are studied, which try to emulate the characteristics of human beings to reason and solve problems [5]. These techniques have been very popular in industrial applications, for example: the systems of classification of raw material by means of artificial vision, the prediction of faults by digital image processing systems or automated irrigation systems that are reconfigured depending on the season of the year [6-8].

Evolutionary algorithms have become one of the studied subjects of supervised learning, which emulate the behavior of Darwinian evolution through different computational algorithms. It can be said that this type of technique is inspired by natural evolution and allows finding several solutions to a problem from a set of initial solutions. This set undergoes modifications when applying to each individual an operator of selection, recombination or mutation, by means of which its aptitude value is improved so that it becomes the most suitable [9, 10].

Consequently, this paper describes a partial solution to the feedback controller limitation by means of a bio-inspired optimization strategy, which allows to automatically configure a PID-type controller. The advantage of this strategy is that it does not require prior knowledge of the system to be controlled, therefore, it is flexible enough to generalize to any type of programmable controller. The characteristics described above are organized

as follows: Sections 2 and 3 define the control parameters and their implementation respectively. In section 4 the results obtained are presented.

II. METHODOLOGY

PID controllers are widely used in industrial environments, to control processes that are connected in closed loop. Unlike them, evolutionary algorithms are usually run on computers and their goal is to find the best possible value in a given search space. The particular characteristics of these control and evolution methods are described below.

A. PID

The term PID controller (Proportional-Integral-Derivative) is used to designate a mechanism, which allows the control of a variable from the information of its state (normally called feedback). This control system estimates the error between a measured value and another expected value. The controller output $y(t)$can be represented by a mathematical expression as observed in equation 1 [11, 12].

$$y(t) = k_P e(t) + k_i \int_0^t e(\tau)\, d\tau + k_d \frac{de}{dt} \qquad (1)$$

As observed in the equation, the output of the controller depends on the error $e(t)$and three constants, which are: integral k_p, proportional k_i and derivativek_d. Where, the constant k_p depends on the current error, the constant k_i depends on the past errors and the constant k_d tries to predict future errors. The sum of the values obtained when replacing the error allows to adjust a process using a control element, as shown in Fig. 1 [11, 12].

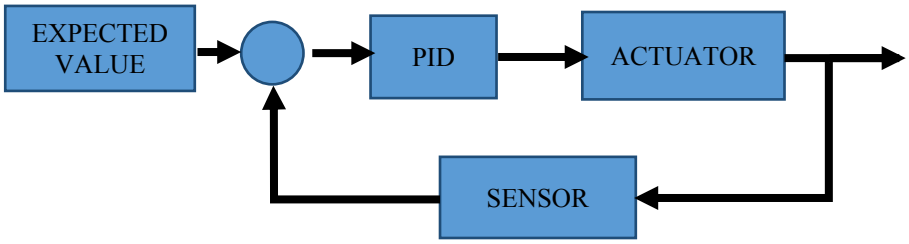


Fig. 1 Architecture of a PID controller [11, 12].

When a PID controller is implemented, the output of the system may have oscillations, which must be adjusted according to the three (3) constants mentioned above. Considering the behavior of the system, the design of the controller can be defined as a series of parameters that modify the amplitude of the oscillations and the operation of the actuator. Depending on the degree of influence of each variable, these determine the functioning as shown below [11, 12].

- Proportionality constantk_p: This constant determines the gain of the controller, that is, the amplitude that the control values can take to modify the state of the controller. Example: The speed of a fan to cool an enclosure with respect to a desired value. In this case the value of k_p increases or decreases the motor speed by multiplying the gain by the temperature difference.
- Integration constantk_i: This constant determines the speed with which the controller repeats the proportional action, in other words, increases or decreases the stability time of the process around the expected value.
- Derivation constantk_d: This constant allows the controller to multiply the value of the proportional constant, with the hope that the error in the process does not increase.

One of the advantages of this type of strategy is that it can be configured in combinations P, PI or PD eliminating control actions that are not necessary. Normally the derivative action is eliminated, since it amplifies the noise and if it is greater than the integral action it does not allow the system to reach the desired value. Each of the values can be estimated by analytical or experimental methods, among them the Zieger and Nichols tuning method is the most used and incorporated in industrial-type controllers [11, 12].

The Zieger and Nichols method states that the process must be online and the adjustment method says that initially the constants k_p, $k_i$and $k_d$must be equal to zero (0). Then the value of k_p is increased until the loop has oscillations around the setpoint value with a constant amplitude. Next, the constant $k_i$must be increased until the oscillations are reduced to zero (0) and the process value is maintained around the set point value. Finally, the constant k_d is increased if it is desired to reduce the establishment time and is carried out with very small steps until reaching the desired value. If this tuning protocol of the PID controller is respected, the system normally has a damped output and without oscillations when reaching its reference value [11, 12].

B. Evolutionary Algorithms

Evolutionary algorithms are based on the premise of generational change and natural selection, to determine which the best individual within a set of solutions. An individual is a possible solution to a problem and its aptitude value is the ability to solve it. There are several types of evolutionary algorithms, including heuristic techniques that mutate a single individual to find the best solution or population algorithms such as the genetic algorithm, which vary sets of solutions to determine the best one. Next, two of these evolutionary optimization techniques are described.

1. Ascent to the Hill

The hill climbing algorithm is based on the idea of a neighborhood, that is, an individual (one) is generated in a random way and their aptitude value is evaluated when crossing the neighborhood. This aptitude value is stored in a temporary memory and another individual (two) is generated using the same random number generator. In the same way that the individual one (1) evaluates his aptitude value and compares it with the stored value. If the aptitude value of the individual two (2) is better than that of the individual one (1), the individual is changed one (1) to the two (2) and repeated cyclically as shown in Algorithm 1 [15].

Algorithm 1. Ascent to the hill

```
Initiate Variables
I1=Generate Individual ()
I2= Generate Individual ()
F1=Evaluate Individual (I1)
F2= Evaluate Individual (I2)
While Termination Condition different from True do
        If F1>F2 then
          I1= I2
          F1=F2
        Else
          I2=Generate Individual ()
          F2=Evaluate Individual (I2)
        End if
End while
```

2. Genetic Algorithm

Unlike the hill climbing algorithm, a genetic algorithm is a population algorithm that is responsible for evaluating sets of solutions until you find one that ideally contains the best approach to the solution. Genetic algorithms mimic the process of natural selection through selection, recombination and mutation functions. This feature allows the algorithm to improve the search process by adjusting the aptitude values of individuals adaptively [13, 14].

A genetic algorithm is sufficiently versatile and can be modified in different ways to improve the search process, among them is the conventional genetic algorithm (See Algorithm 2). This algorithm begins with a selection of individuals based on the aptitude value of each of them. After the set of selected individuals, several of them are modified combining their characteristics. Finally, a mutation with a very small radius of action is carried out, that is, the individual is taken and part of its characteristics is modified to incorporate it back into the population [13, 14].

Algorithm 2. Conventional genetic algorithm

```
Initiate Variables;
P0=Generate initial population ();
/* a population is a set of individuals and is initially generated randomly */
While Termination Condition different from True do
        Aptitude=Evaluate Function (P0);
        P1=Selection (P0, Aptitude);
        P1=Crossing(P1);
        P1=Mutation (P1);
        P0=P1;
End while
```

A way of representing an individual was proposed by Holland and is called the Schema theory. Schema theory states that individuals in the population must be encoded in binary which represents a chromosome. These chromosomes can be crossed by dividing and combining their sections or mutate by changing one or several bits. In addition, the selection process is stochastic based on the assignment of a probability value to each individual depending on their aptitude value, that is, if the individual has a high aptitude value it has a high probability of being selected as shown in Fig. 2 [13,14].

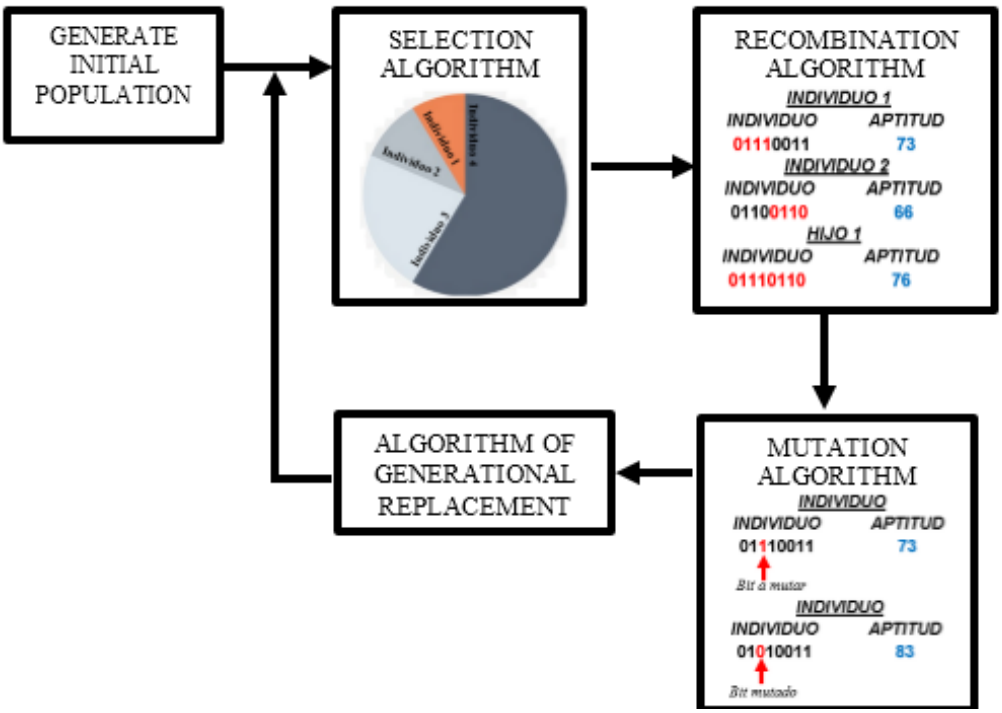


Fig. 2 Operation of the conventional genetic algorithm, based on [13].

III. DESIGN AND IMPLEMENTATION

The proposed algorithm looks for sets of parameters for the PID, which are varied depending on the response of the process. In this case, the process to be controlled is a closed loop of flow that is monitored with a vane sensor and the flow is modified with a proportional electro-pneumatic valve. The valve and the sensor have a communication protocol with the controller from 4 to 20 mA, that is, the controller reads and writes the process signals in terms of the current (See Fig. 3).

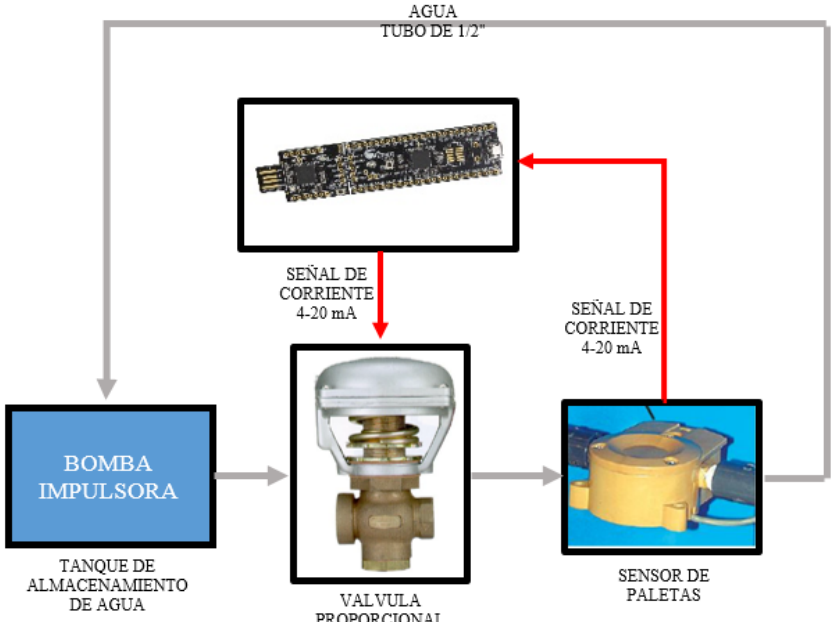
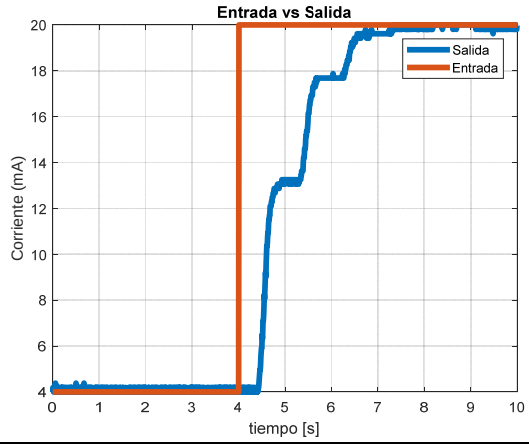
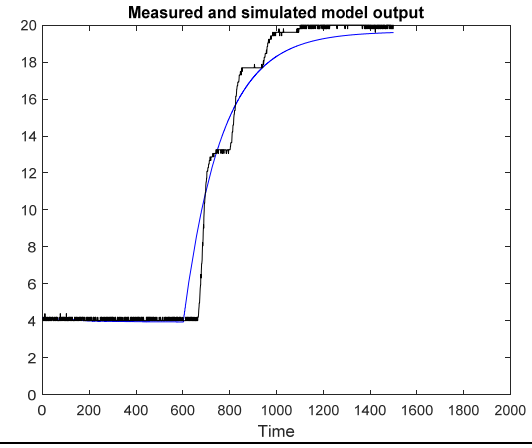


Fig. 3 Process.

The algorithm was implemented in a computer, which interprets the process data with the help of a PSLO 5LP. The PSOC works as an interface to send and receive information through current loops. This information was used to find an approximate mathematical model of the system (see Table I) using a simulator, in order to simulate four (4) techniques of tuning the controller to compare its performance.

The objective of each technique is to reduce the error between the expected value and the value measured by the sensor. In the evolutionary techniques each individual is composed of three parameters that refer to the constants $K_p, K_i and K_d$and their aptitude value is the steady-state error. The value of the error is the difference between the measured value and the expected value, which is stored in an accumulator to later estimate its average. The quantity measured during the execution of each individual is one hundred (100), since the execution time of each individual is 10 seconds and the computer sampling rate is one hundred milli-seconds (100 ms). However, the only tuning technique that does not evaluate individuals is the Zieger and Nichols, since it is not an evolutionary algorithm.

TABLE I. System Identification.

Applied variation	**Correlation of 86.93%**
	
Characteristic equation	
$$H(z) = \frac{0.01195\, z^{-1} + 0.01199\ z^{-2}}{1 + 0.992\, z^{-1} - 0.9877 z^{-2} - 0.9799\ z^{-3}}$$	

The evolutionary algorithm proposed is based on a genetic algorithm, which has the ability to adjust the rate of recombination and mutation. By means of a strategy that rewards the population that has an aptitude value of less than 0.5, in another case it is penalized and the probability of mutation increases. The reference aptitude value is taken close to zero (0) and the idea is that the steady state error reaches this value.

The developed algorithm incorporates the function of selection by the ranking method, which orders the individuals from highest to lowest aptitude value and selects them according to a random number generated that sets a threshold value to establish what individuals will be part of the population selected. Next, a recombination of two (2) individuals is carried out by exchanging two (2) of their components in a random manner. Then, the mutation is done by selecting a component of the individual and multiplying its value by a random number generated between -0.05 and 0.05. Finally, generational replacement is done by comparing the aptitude value of the individuals of the new population and the previous population to select those with the best aptitude value (see Algorithm 3).

Algorithm 3. Proposed evolutionary algorithm

```
Modified AE program ()
    Initiate variables
    P0=Initial population U~[0,1]
While Termination Condition different from True do
        A0=Evaluate population (P0)
        P1=Selection by ranking method (P0,A0)
If R~U[0,1]<Pc then
            Sons=recombine(P1)
Else
            Sons=P1
        End if
If R~U[0,1]<Pm then
            Sons=mutate(Sons)
        End if
        A1=Evaluate population (P1)
If mean (A1) < 0.5 then
            Pc=Pc+0.01
            Pm=Pm-0.01
Else
            Pc=Pc-0.01
            Pm=Pm+0.01
        End if
        P0=Generational replacement (P1)
End while
End program
```

IV. RESULTS AND DISCUSSION

There were 400 evaluations of the aptitude values of different individuals, which are equivalent to 40 generations of the genetic algorithm (see Table II). These evaluations were made to the approximate model and then the best individual found in generation zero (0), fifteen (15) and forty (40), were programmed in the real controller to verify their behavior (see Table III).

Unlike evolutionary techniques, tuning of the controller using the Zieger and Nichols strategy was simulated only once to verify if the system can be controlled with a PID, which is shown in Fig. 4.

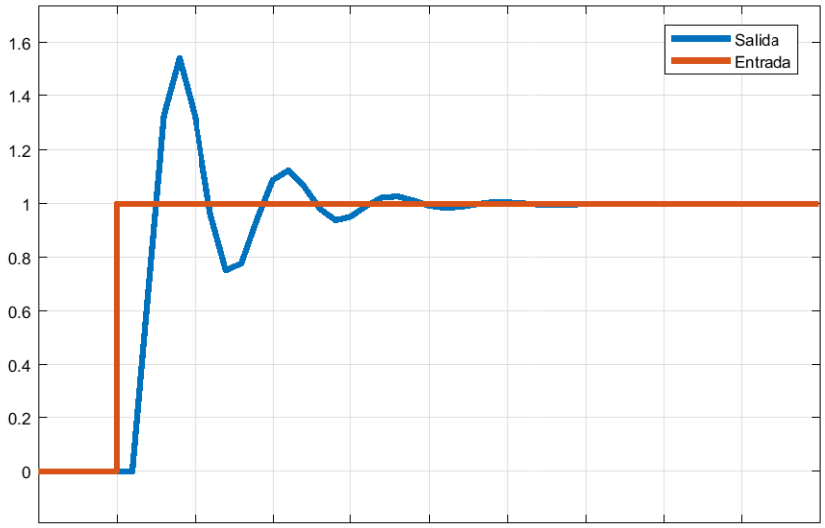


Fig. 4 Response implementing Zieger and Nichols

ISSN (Print) : 2319-8613
ISSN (Online) : 0975-4024

TABLE II. Results obtained in the simulation.

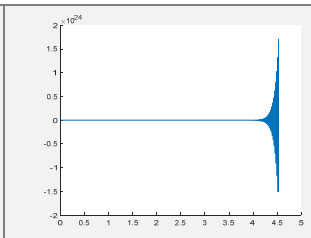
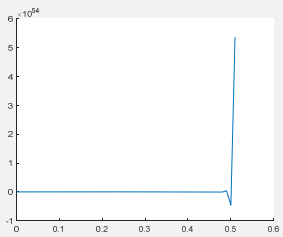
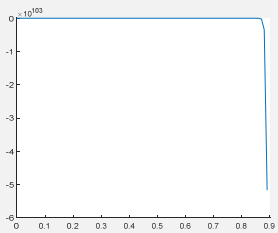
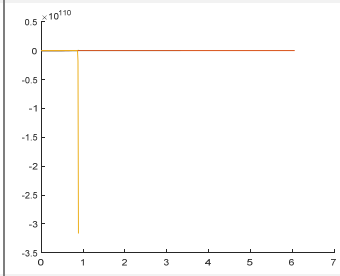
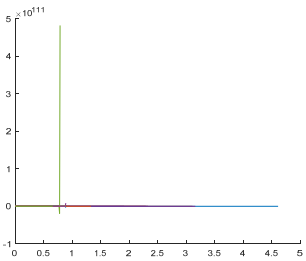
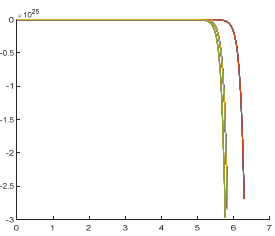
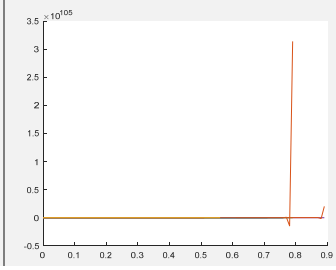
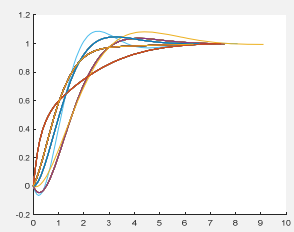
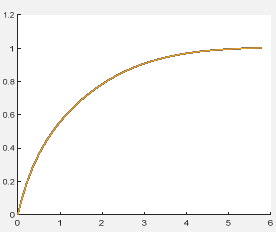
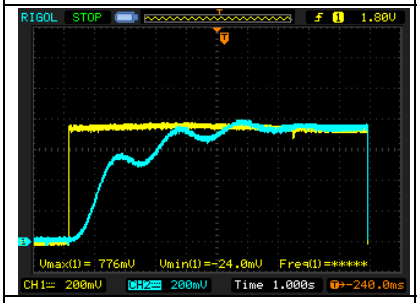
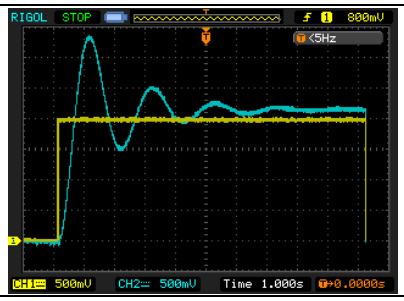
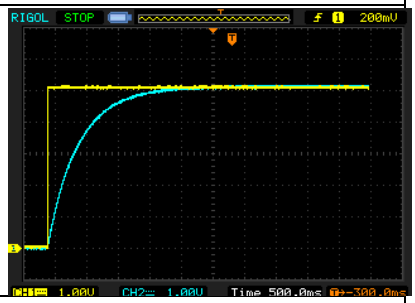
	10 EVALUATIONS	150 EVALUATIONS	400 EVALUATIONS
ASCENT TO THE HILL			
GENETIC ALGORITHM			
EVOLUTIONARY ALGORITHM			

TABLE III. Experiments with the real controller.

10 EVALUATIONS	150 EVALUATIONS	400 EVALUATIONS
		
$Kp = -0.6, K_i = 0.3, K_d = -0.01$	$Kp = 1.2, K_i = 2.88, K_d = 0.6$	$Kp = 3.8, K_i = 10.55, K_d = 3.24$

V. CONCLUSIONS

The proposed optimization strategy allows the controller to find sets of parameters to automatically tune a control loop with a steady state error less than 0.1, which is very useful since an operator could configure a PID controller without the need for a prior knowledge of the system to be controlled.

The developed strategy has a better performance than the conventional genetic algorithm and the hill climbing technique, because the automatic adaptation capacity of the crossing and mutation operators allow it to explore and exploit the areas where there is information of interest only. This characteristic is not observed in the other techniques tested in this paper, since its operators do not depend on any type of probability which causes a high variability during the optimization process reducing the effectiveness of the technique when trying to solve this type of problems.

The recombination and mutation operator worked properly in solving this problem, because the search space is relatively small (of three components). However, they are operators that induce a high variability to the optimization process and are not recommended for large search space, since the high variability can cause the algorithm to omit local optimum values because of its step size.

ACKNOWLEDGMENT

This work was supported by the Universidad Distrital Francisco José de Caldas Technological Faculty. The views expressed in this paper are not necessarily endorsed by the University. The authors thank the research group ARMOS for the evaluation carried out on prototypes of ideas and strategies.

REFERENCES

[1] P. J. Gawthrop, "Self-tuning PID control structures," IEE Colloquium on Getting the Best Our of PID in Machine Control, London, UK, 1996, pp. 4/1-4/4. doi: 10.1049/ic:19961463

[2] M. Tajjudin, M. H. F. Rahiman, N. Ishak, R. Adnan and H. Ismail, "Comparison between optimally-tuned PID with self-tuning PID for steam temperature regulation," 2012 4th International Conference on Intelligent and Advanced Systems (ICIAS2012), Kuala Lumpur, 2012, pp. 551-556. doi: 10.1109/ICIAS.2012.6306076

[3] W. Zhang and M. Yang, "Comparison of auto-tuning methods of PID controllers based on models and closed-loop data," Proceedings of the 33rd Chinese Control Conference, Nanjing, 2014, pp. 3661-3667. doi: 10.1109/ChiCC.2014.6895548

[4] R. R. Bambulkar, G. S. Phadke and S. Salunkhe, "Movement control of robot using fuzzy PID algorithm," 3rd International Conference on Electrical, Electronics, Engineering Trends, Communication, Optimization and Sciences (EEECOS 2016), Tadepalligudem, 2016, pp. 1-5. doi: 10.1049/cp.2016.1519

[5] M. Korkmaz, Ö. Aydoğdu and H. Doğan, "Design and performance comparison of variable parameter nonlinear PID controller and genetic algorithm based PID controller," 2012 International Symposium on Innovations in Intelligent Systems and Applications, Trabzon, 2012, pp. 1-5. doi: 10.1109/INISTA.2012.6246935

[6] T. Holtorf, F. J. Knoll and S. Hussmann, "Multimodal image stitching algorithm for weed control applications in organic farming," 2016 SAI Computing Conference (SAI), London, 2016, pp. 336-342. doi: 10.1109/SAI.2016.7556003

[7] E. Saenz, M. Jimenez and A. Ramirez, "Strawberries collecting robot prototype in greenhouse hydroponic systems," Symposium of Signals, Images and Artificial Vision - 2013: STSIVA - 2013, Bogota, 2013, pp. 1-4. doi: 10.1109/STSIVA.2013.6644933

[8] M. Halstead, C. McCool, S. Denman, T. Perez and C. Fookes, "Fruit Quantity and Ripeness Estimation Using a Robotic Vision System," in IEEE Robotics and Automation Letters, vol. 3, no. 4, pp. 2995-3002, Oct. 2018. doi: 10.1109/LRA.2018.2849514

[9] C. H. Chen and C. B. Liu, "Reinforcement Learning-Based Differential Evolution with Cooperative Coevolution for a Compensatory Neuro-Fuzzy Controller," in IEEE Transactions on Neural Networks and Learning Systems. doi: 10.1109/TNNLS.2017.2772870

[10] R. Janssen, S. Nolfi, P. Haselager and I. Sprinkhuizen-Kuyper, "Cyclic Incrementality in Competitive Coevolution: Evolvability through Pseudo-Baldwinian Switching-Genes," in Artificial Life, vol. 22, no. 3, pp. 319-352, Aug. 2016. doi: 10.1162/ARTL_a_00208

[11] Ogata, K., Ingeniería de control moderna (1980). Prentice Hall. Disponible en: https://books.google.com.co/books?id=aOUSNAAACAAJ

[12] Benjamin C. Kuo, Sistemas de control automático (1996). Disponible en: https://books.google.com.co/books/about/Sistemas_de_control_autom%C3%A1tico.html?hl=es&id=GyWr6cT8SEsC&redir_esc=y

[13] Cruz, P.P. Inteligencia artificial con aplicaciones a la ingeniería (2011). Marcombo. Disponible en: https://books.google.com.co/books?id=myyQpwAACAAJ

[14] T. Barman and N. Deb, "State of the art review of speech recognition using genetic algorithm," 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI), Chennai, India, 2017, pp. 2944-2946. doi: 10.1109/ICPCSI.2017.8392264

[15] S. K. N. A. Rahim, A. Bargiela and R. Qu, "The incorporation of late acceptance hill climbing strategy in the deterministic optimization of examination scheduling framework: A comparison with the traditional hill climbing," 2014 IEEE Conference on Systems, Process and Control (ICSPC 2014), Kuala Lumpur, 2014, pp. 147-152. doi: 10.1109/SPC.2014.7086247