Control Applications Using Computational Intelligence Methodologies

P. Burbano[1], Member, IEEE, O. Cerón[1], Member, IEEE, A. Prado[1], Member, IEEE

[1]Dept. of Automation and Industrial Electronics, Escuela Politécnica Nacional,
Isabel La Católica S/N Quito, Ecuador,
email: oceron@uio.satnet.net

Abstract – **The ambiguity, robustness and performance of systems becomes increasingly evident as the complexity and size increase in industrial processes. Under these circumstances, conventional control techniques such as PID control, nonlinear feedback control, LQG control and H∞ control have several disadvantages because they are hard or inflexible and cannot handle qualitative knowledge. This may explain the dominant role of emerging** "intelligent control, using soft computing" **as an interesting alternative. This means that increasing complexity requires an increase in intelligence in order to provide robustness and adaptability. Intelligent techniques such as** fuzzy systems (FS), neural networks (NN) and genetic algorithms (GA) **can be combined using the resources of such techniques complementary. This paper is a report of a research work in intelligent control of the GISIA group (Grupo de Investigación en Sistemas Inteligentes y Aplicaciones) at Escuela Politécnica Nacional.**

I. INTRODUCTION

When system dynamics are represented by input–output measurements, conventional techniques employ identification methods for modeling the plant and adjust the control parameters.

Neural networks can be trained to system identification and control. Fuzzy logic systems use heuristic knowledge for modeling and control. Genetic algorithms are employed to improve controller performance by adjusting the control parameters according to some optimization criterion. Genetic algorithms can also be applied to fuzzy systems, thus enhancing fuzzy logic technique.

If we combine the neural network learning capacity with the approximate reasoning of fuzzy systems, we reach a powerful neuro–fuzzy methodology, which also uses input–output data and is not limited to heuristic knowledge only.

The MATLAB computational intelligence toolbox is used to simulate results over different models with the aid of SIMULINK.

The neural network toolbox gives a NARMA-L2 control, which is implemented with a NARMA model (Nonlinear Auto Regressive Moving Average) to identify systems to be controlled by feedback linearization (L2 control).

The fuzzy control toolbox can be used in three different ways, as a FIS system (Fuzzy Inference System), Mamdani type or Sugeno type and as an ANFIS system (Adaptive Neural Fuzzy Inference System) for adapting membership functions to input–output data.

The genetic toolbox requires a fitness function or a shape constraint with SIMULINK response optimization toolbox, which is simpler to use.

II. COMPUTATIONAL INTELLIGENCE METHOLOGIES

The principal constituents of soft computing are fuzzy logic, neuro-computing, genetic programming and probabilistic reasoning, which are not excluding but rather complement each other and can be coordinated through an intelligent agent. This paper presents several simulation studies in the use of neural networks (NN), fuzzy logic (FL) and genetic algorithms (GA) applied to the design and optimization of controllers.

A. Neural networks

A neural network is a structure that fits a model for a given data, which means that it can be trained to represent a model. For example, it can be trained to represent an industrial process taking into account the non linearity that it may have because the network approach to modeling a plant uses a generic nonlinearity and allows all the parameters to be adjusted.

All neural networks learn from a given data set by adjusting their weights and bias in a manner to fit this data set; meaning that the network response is almost the same as the data entered to train the network.

For to control engineers, neural networks are commonly used for modeling a plant or to create intelligent controllers, as they can learn and adapt; they can approximate nonlinear functions, they are suited for parallel and distributed processing, and they naturally model multivariable systems. When a neural network is used to model a plant, it is placed in parallel to the system. During the training phase the neural network tries to minimize the error between the plant output and the network output (the prediction error). This is to represent the forward dynamics of the plant, as shown in Figure 1.

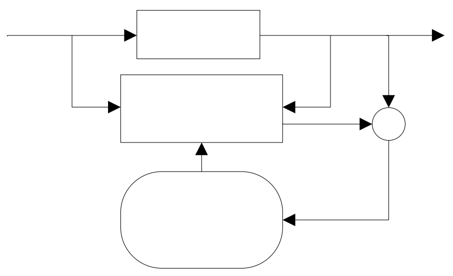

Fig.1 Representation of forward dynamics of the plant

There are many tools for building and using neural networks, one of them is MATLAB which has a toolbox with many different controllers and learning techniques which will be discussed in this section.

There are many ways to connect the neurons, and depending on the objective of the network, it is possible to add as many neurons as required. There are also many ways to train a network, it can follow a supervised or unsupervised training scheme, but once again, it depends on the application.

MATLAB has a simulation environment called SIMULINK where the neural network toolbox can be used to create controllers or models of a process. For example, the NARMA-L2 (Nonlinear Autoregressive-Moving Average) control block, which transforms the nonlinear system dynamics into linear dynamics by canceling the nonlinearities, this model employs Equation (1) to control the plant.

$$\begin{aligned} y(t+d) = \; & f[y(t),\ldots,y(t-n+1),u(t),\ldots,u(t-n+1)] + \\ & + g[y(t),\ldots,y(t-n+1),u(t),\ldots,u(t-n+1)]\cdot u(t+1) \end{aligned} \quad (1)$$

Where $d \geq 2$ and g is the function that will minimizes the mean square error.

Equation (1) can be represented as a neural network model as shown in Figure 2 [5].

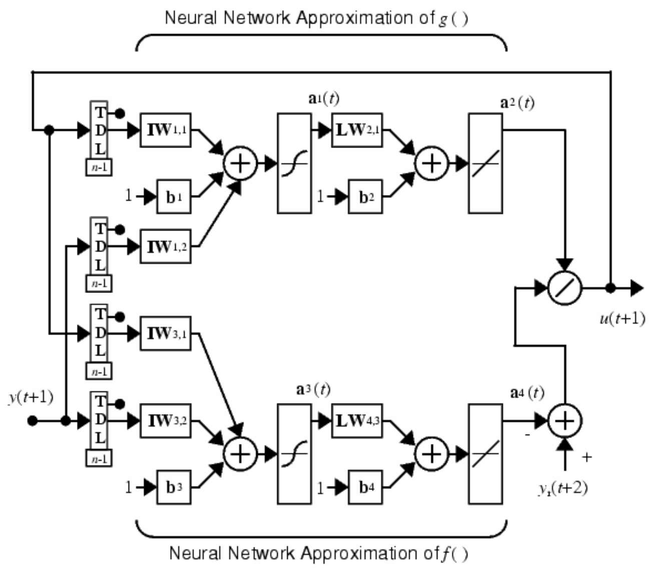

Fig.2 Approximation of a NARMA-L2 controller

MATLAB contains other neural network controllers, all based on plant identification.

B. Fuzzy logic

Sometimes to deal with some processes it is necessary to use more than just signals of true or false signals. Often it is needed to embed the human reasoning to the plant control; so fuzzy logic takes place to simulate the way people think, therefore fuzzy logic deals with imprecise data.

There are times where a model of a plant is not readily available because it is too expensive or the process is very complex, but there is an expert who has a vast knowledge of the process; the challenge is to transmit this knowledge to an automatic controller to deal with the plant even if there is nobody to control it. A fuzzy logic controller employs membership functions to describe a degree of membership of the input by linguistic variables.

A fuzzy logic controller uses fuzzification in the inputs and defuzzification on the output to transform values to a linguistic form. In between a rule base system and an inference engine are used to match with an adequate output. A diagram of this process is presented in Figure 3 [3].

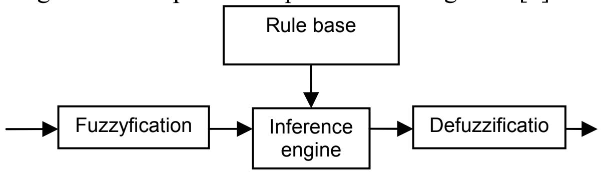

Fig.3 Fuzzy controller block diagram

The rules contain knowledge of human experts in a particular process. The inference engine takes decisions over which rules are more relevant to the actual situation and executes the correspondent actions.

There are several types of membership functions, like triangular, trapezoidal, gaussian, singleton, and others. When the membership functions of the output are of a singleton type or a linear combination of the inputs, or even a function of the inputs, the fuzzy logic is called a Sugeno-Takagi rule structure. A Sugeno model is used in an Adaptive Neuro-Fuzzy Inference System (ANFIS), which employs a neural network to represent a fuzzy system. The advantage of using ANFIS is that the fuzzy system learns from input-output data.

C. Genetic algorithms

Genetic algorithms try to imitate the evolution of species, meaning that they find out which individual of the population is best adapted to an environment. Genetic algorithms (GA) begin by generating a large number of possible solutions for a given problem; then it evaluates each solution by a fitness function and decides which individuals are better adapted. The fitness function gives a score for each individual of the population depending on how close it is to solve the problem.

This means that the GA tries to find an optimal solution for a given problem. The main advantage against other optimization methods is that GA will not stay in a local minimum of the fitness function because it doesn't use the gradient of the function to find the solution [4].

The GA can be used in control problems to find an optimum controller. GA's can be used to tune a PID controller or even in a fuzzy controller to find out which membership functions and rules are best suited for an optimal performance of a process.

In order to tune a PID controller using GA's it's necessary to optimize three parameters that will lead to an optimal controller. To do this, first GA's create a great number of PID's, in a random mode, and then GA will run until these three parameters are obtained.

In order to tune a fuzzy controller using GA's it is necessary to define which parameters are to be optimized, for example the membership functions, the rule set or both. This means that there will be many parameters to optimize and the algorithm will be slower than the tune of a PID controller.

The fitness function that evaluates the controllers has to measure the output of the plant and be capable of giving its score in function of the settling time, peak response, rise time, etc.; meaning that the score is a function of the typical measures for control systems.

III. CONTROL APPLICATIONS AND SIMULATION RESULTS

The above methodologies have been tested widely; in this paper results are obtained by employing different representative techniques over interesting models as case studies.

A. ANFIS controller

This simulation was made over an F-14 model control, shown in Figure 4. The model of the F-14 plane was obtained from MATLAB and makes a longitudinal control of the Grumman Aerospace F-14 [2].

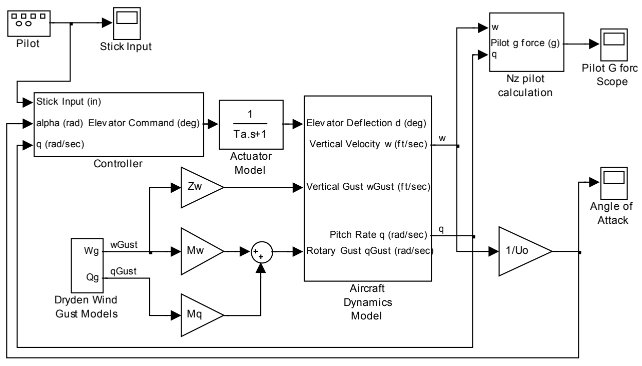

Fig.4 Grumman aerospace F-14 model

The original controller is a PID, but it will be replaced by an ANFIS controller, so the controller block will be as shown in Figure 5.

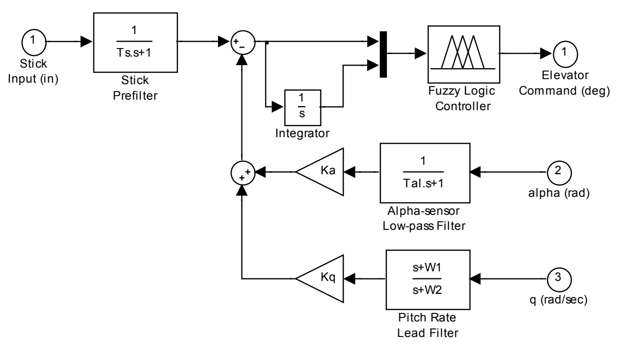

Fig.5 Controller F-14 model

The input-output data is obtained from Figure 6.

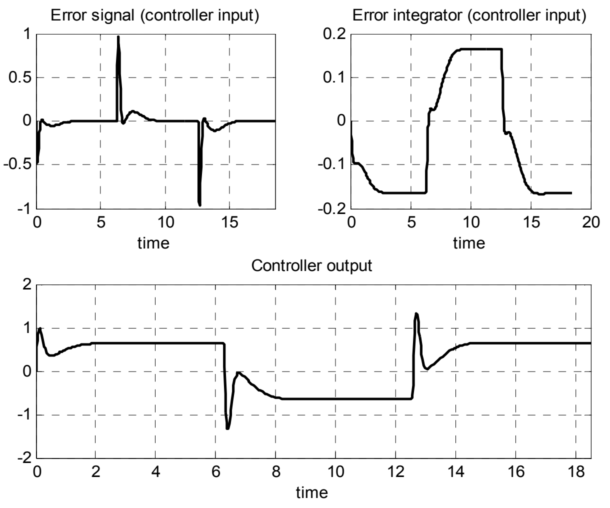

Fig.6 Training data for the fuzzy controller

Once the data is selected and the fuzzy controller is trained, the response, which is as good as the response using a PID controller, is shown in Figure 7.

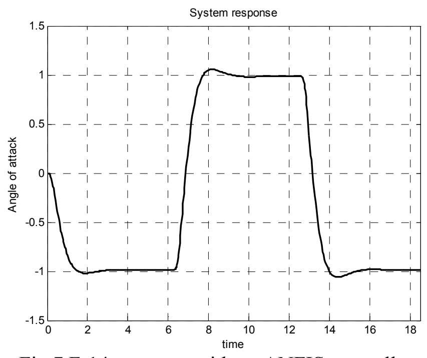

Fig.7 F-14 response with an ANFIS controller

The fuzzy system obtained is shown in Figure 8.

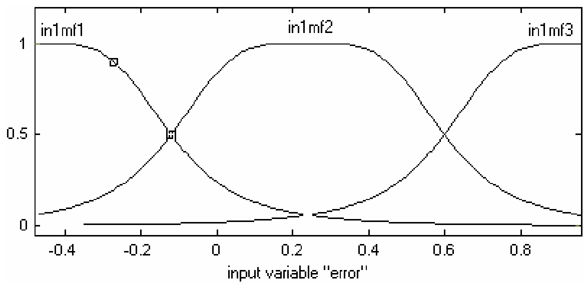

Fig.8 Membership functions for the F-14 ANFIS controller

The Sugeno type outputs, which are linear functions of the inputs, are [6]:

- Outmf1: [-1.854 -3.424 0.04774] Outmf2: [-1.764 -3.266 0.02481]
- Outmf3: [-1.027 -1.767 -0.3363] Outmf4: [-1.773 -3.979 -0.01334]
- Outmf5: [-1.729 -4.026 0.005672] Outmf6: [-1.853 -4.131 0.01955]
- Outmf7: [-1.867 -2.958 0.2613] Outmf8: [-1.192 -0.6372 -0.3679]
- Outmf9: [-0.08004 -0.03529 0.2524]

B. Optimization method of a PID

There are many ways to control a DC motor; one of the possible solutions is to use a thyristor half bridge, which is the solution adopted for this application [6].

The circuit used for this purpose can be separated in three steps, the control circuit, the power circuit and the comparator circuit. The control circuit is the PI and the input is the error between the set point and a tachometer attached for the motor shaft. The power circuit is built with the half bridge and the optocouplers that carry the fire signals to the thyristors. The comparator circuit generates a cosenoidal signal which is compared with the PI output. This gives a fire pulse for each thyristor. This circuit is implemented in order to have a velocity control over a DC motor.

The GA's are used to obtain the PI parameters. To do this requires a process model, so first a set of input-output data is collected using the model shown in Figure 9. The input and output signals are given in volts and the input signal is corrupted with noise.

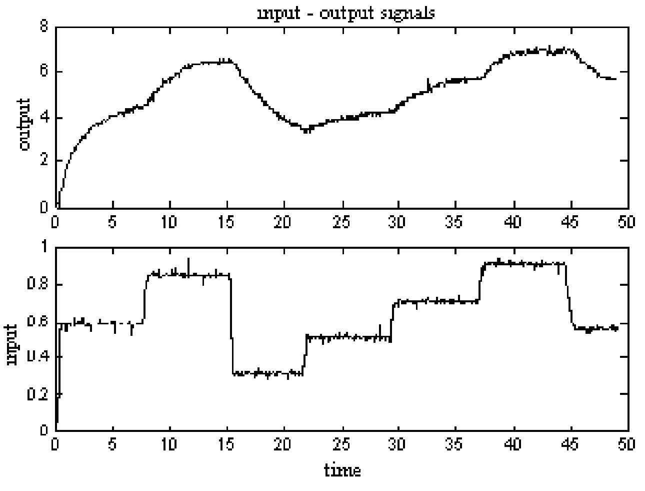

Fig.9 Collection of input-output data for the plant equation

Using the MATLAB identification toolbox, process models option, the system transfer function is obtained. This is shown in Equation (2).

$$G_{(s)} = \frac{8.264}{0.002295 \cdot s^2 + 2.299 \cdot s + 4.517\,1} \tag{2}$$

Then the PI controller is adjusted using classical control obtaining the transfer function of Equation (3). The system responses are shown in Figures 10 and 11.

$$PI = 3.8 + \frac{0.07}{S} \tag{3}$$

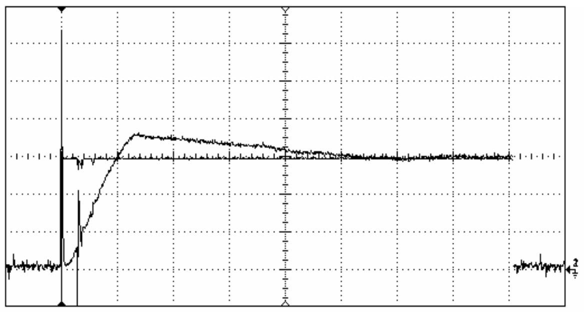

Fig.10 DC motor step response with a conventional PI

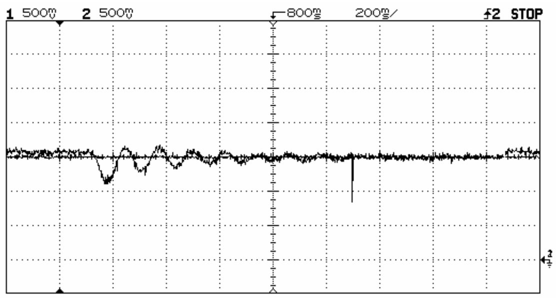

Fig.11 DC motor response when adding load at 400 rpm

Once the transfer function is created the next step is to apply the GA. When applying GA's a fitness function is needed, and because this is a control system the response of the plant will be optimized. A fitness function requires constraints to see if the current plant response is close enough to the desired response. To do this the settling time, the error and maximum overshoot are necessary measures. First, each one of these has to be multiplied by a weight factor, to indicate which one is the most important and to have every measure in the same rank. Because the algorithm tries to obtain the minimum score of the fitness function, a sum of the new values (after applying these weight factors) could be used to obtain the score.

The PI transfer function of Equation (4) and motor responses in Figures 12 and 13 were obtained with GA's.

$$PI = 4.9 + \frac{0.066}{s} \tag{4}$$

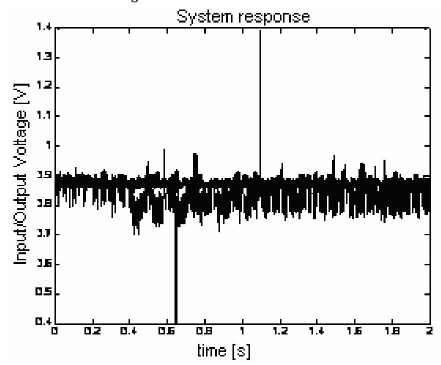

Fig.12 DC motor response when adding load at 270 rpm

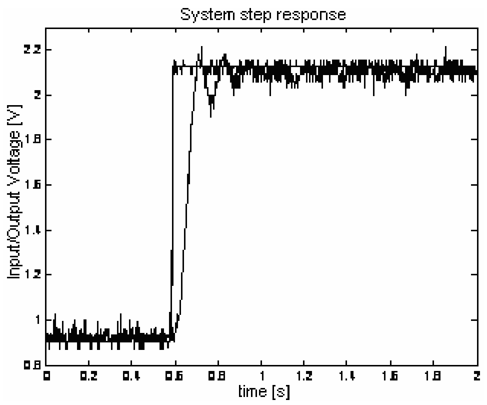

Fig.13 DC motor step response using GA's for PI parameters

Observing these figures is obvious that the PI obtained using GA's is better because the motor responds faster and when adding a load the motor recovers faster and with a minimum of oscillations.

C. Optimal PID control using response optimization toolbox

The magnetic levitation model is a non linear problem and that is why it is used to prove many controllers. In this example the levitation model is going to be controlled by a PID optimized with GA's.

In order to optimize the response of the model the *response optimization toolbox* in SIMULINK is used with some signal constraints set to search for an appropriate solution. The constraints can be introduced in the form of bars allowing a simple handling for searching an optimal solution. This is shown in Figure 14 [6].

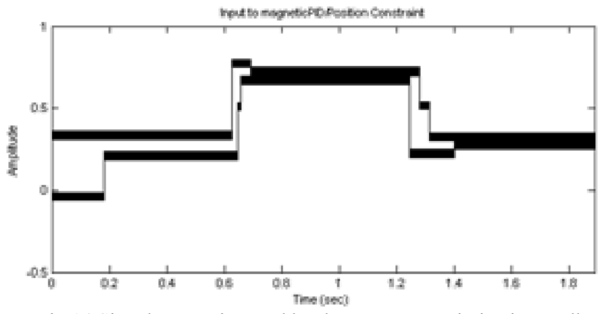

Fig.14 Signal constraint used by the response optimization toolbox

The search can be done by many methods including *pattern search* with GA's, which is the one used. Once the search algorithm, the constraints and the parameters to be optimized are selected the toolbox is ready to start and it begins searching for the solution that best fit the constraints.

For this model a PID controller is optimized using this method and the advantage is that no fitness function is needed, the fitness function is created by the signal constraints and the toolbox. When the algorithm finishes the following parameters for a discrete PID controller were calculated.

- Kp = 0.8299
- Ki = 8.2769
- Kd = 0.0357

With these parameters the solution can be found in Figure 15.

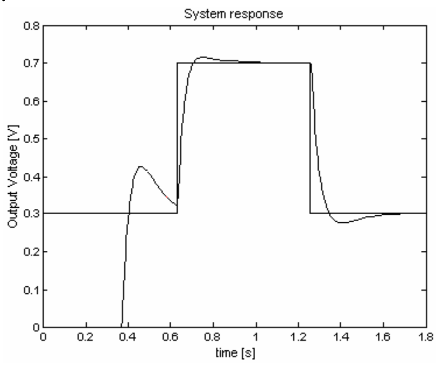

Fig.15 Response of the magnetic levitation using a PID controller

Using this toolbox saves a lot of time because it doesn't need the creation of a fitness function.

D. Water level control using GA's for a fuzzy control

There are many times when a level control is needed and one of the best solutions is to use a fuzzy controller because it's simple to use and doesn't need a lot of knowledge in engineering; if these are the requirements it can be helpful to use GA's in order to obtain an optimal fuzzy controller. This will be the approach of this section.

A fuzzy controller will be created using the following inputs: the water level and the changing rate of its output. Since there are only a few rules needed it will be very simple to create a controller. The design of the controller is not discussed but the difference with the optimized controller using GA's will be highlighted.

The creation of the fitness function uses the error, maximum overshoot and settling time as the key measures to give a score to each controller.

Many references set up that the fuzzy controllers can be optimized by changing the rules of the inference system, but changing the membership functions of its inputs are another approach and this is used to create a better controller.

Figures 16 and 17 show the membership functions for each input used to create the first controller and the output of the controller changes the opening of the valve that controls the input flow to the tank. These membership functions were created using common sense.

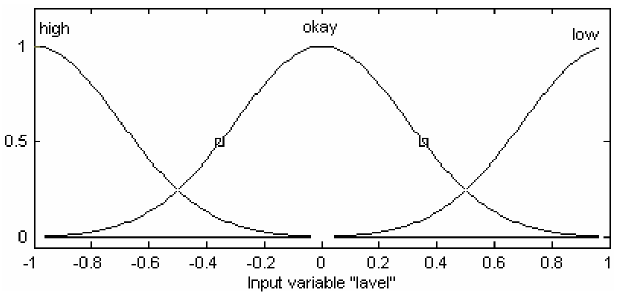

Fig.16 Membership functions for the input level

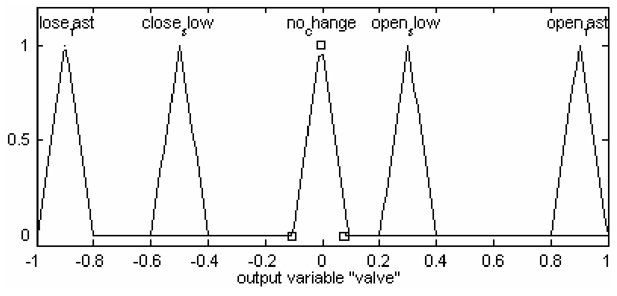

Fig.17 Membership functions for the output

These membership functions are a good solution to the problem. After applying GA's to optimize the membership functions it resulted in a better solution.

The membership functions for the inputs remain almost the same, meaning that only the output needs to be optimized.

Figure 18 shows that the undershoot has disappeared and the settling time is less than in the original system, but both controllers are suitable for the system.

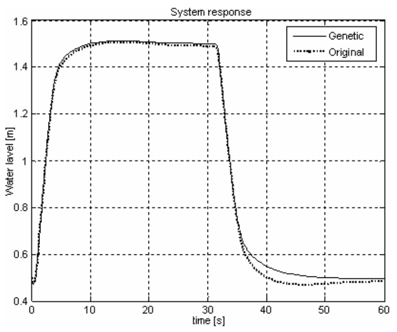

Fig.18 Water level system response

E. Control simulation using NARMA-L2

As seen in previous examples, the F-14 model is very complex and the controllers tend to fail, so this model is used to prove that neural controllers work well. First of all the NARMA-L2 controller needs to have a neural plant model to work with it and obtain a network that controls a plant in a reasonable way [5].

First a set of input/output data is generated using the F-14 model [6]. Then a neural controller is trained using this data. The obtained neural network is applied to the plant. In this case of study the response of the plant is compared to a very good PI controller to see the differences. The outputs obtained using these controllers are shown below.

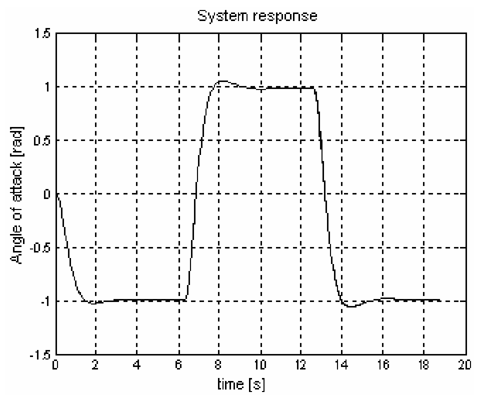

Fig. 19 F-14 angle of attack response

In Figure 19 it's really hard to see the difference between the two controllers because both give excellent responses. To watch the real difference between them a zoom is needed and it's shown in Figure 20.

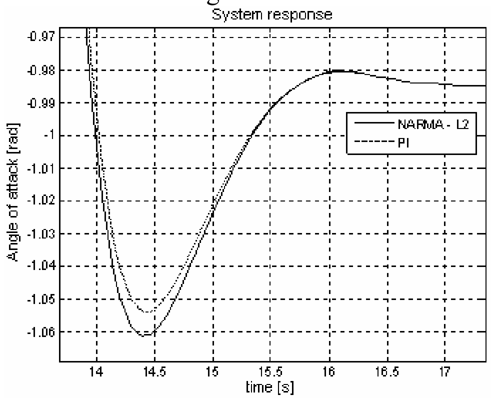

Fig. 20 F-14 angle of attack response

This figure show that both controllers act in the same manner so either one of them can be used.

V. Conclusions

Looking at the results we conclude that intelligence control techniques are suitable to implement complex control systems. Genetic algorithms enhanced the tuning of the popular PID industrial control and it allows to improve the performance of fuzzy control, especially when we deal with input-output data to identify the fuzzy model through a suitable member functions set.

A wide research work can be made using neural networks for identification and control. Here we apply a specific algorithm easy to implement with good results for a complex model such as the F-14 airplane model. In this case the results are similar to those reach in the reference. Also these methodologies combined with fuzzy systems enhance the flexibility and reliability of fuzzy control when input-output data is used in an ANFIS context.

VI. Acknowlegements

Research sponsored by the Automation and Industrial Electronics Department of the Escuela Politécnica Nacional and GISIA group.

VII. References

[1] J. M. Mendel, Fellow, IEEE, Fuzzy Logic Systems for Engineering: A Tutorial, Department of Electrical Engineering Systems, University of Southern California, Los Angeles CA 989-2564 USA.

[2] A. Zilouchian, M. Jamshidi, Intelligence Control Systems Using Soft Computing Methologies, CRC Press, USA, 2001, 291-299.

[3] C. W. de Silva, Intelligence Control, CRC Press, Canada, 1995, 69-81.

[4] K. F. Man, M. S. Tang and S. Kwong, Genetic Algorithms, Springer Verlag, Hong Kong, 1998, 23-42.

[5] M. A. Abramson, Genetic Algorithm and Direct Search Toolbox, The MathWorks, Inc., USA, 2004, 6.14 -6.19.

[6] A. Prado, Control con Inteligencia Computacional, Final year project, Escuela Politécnica Nacional, Ecuador, 2005, 73-77, 109-112, 121-128, 144-146.

[7] G. Nakamiti, R. Freitas, J. Prado and Gomide F., "Fuzzy Distributed Artificial Intelligence Systems" in *Proceedings of the IEEE International Conference on Fuzzy Systems,* 1994, 462-467.

[8] J. Jantzen, A Robustness Study of Fuzzy Control Rules, Technical University of Denmark Dept. of Automation, Bldg 326, DK-2800 Lyngby, Denmark.

[9] V. B. Rao, C++ Neural Networks and Fuzzy Logic, M&T Books, IDG Books Worldwide, Inc., ISBN: 1558515526, Pub Date: 06/01/95.

[10] J. Jantzen, Tutorial on Fuzzy Logic, Technical University of Denmark Dept. of Automation, Bldg 326, DK-2800 Lyngby, Denmark, Tech. report, no 98-E868, 19 Aug 1998 (logic).