Industrial PC-based real-time controllers applied to second-order and first-order plus time delay processes.

R. BÁRCENA and A. ETXEBARRIA
Departamento de Electrónica y Telecomunicaciones
Universidad del País Vasco
E. U. Ingeniería Tca. Industrial. Plaza de La Casilla, 3. Aptdo. 48012 de Bilbao
SPAIN
rafa.barcena@ehu.es, ainhoa.etxebarria@ehu.es, http://www.ehu.es/apes

Abstract: - Nowadays, the industrial processes are controlled mainly by using Programmable Logic Controllers (PLC). Anyway, the more strict specifications imposed at present day are bringing about the gradual introduction of advanced control techniques (fuzzy, adaptive, …) in industry. In order to soundly implement such techniques, new hardware (DSP, Industrial PC, …) may be necessary. This *paper* describes a complete environment for a realistic experimentation on advanced controllers in order to facilitate its study and posterior implementation in productive processes. The experimentation environment is formed by an industrial PC that acts as real time controller and other PC or real actuators like process plants. Hardware-in-the-loop (HIL) experimentation provides a very useful tool for testing systems or subsystems at early stages of the development process. This can reduce the debugging process of the system, in addition to eliminate project risks, because hardware prototype is not necessary before carrying out testing and system integration. HIL experimentations work in real time and carry out input and output operations as a part of the real system in their operative environment. This enables the system or subsystem to be tested under work nominal conditions as well as at its intended operational boundaries.

Key-Words: - Real-time control, Rapid prototyping, Hardware-in-the-loop simulation, digital control, autotuning controllers, pole placement.

1 Introduction

Automation of manufacturing processes is growing up in the present days. The fast development of electronic hardware has allowed a rapid substitution of analogical controllers for digital ones. Distributed control, digital control and field buses are usual terms in the current industry. This change has been possible by introducing more sophisticated control equipments and with many possibilities of communication among them, for instance, PLC's (Programmable Logic Controllers). However, the actual trend is to change these devices for systems based on the open PC architecture.

At first, digital control techniques were applied to industry by means of PLC's. But these devices started to present some limitations as memory capacity, manufacturer dependency and so on. This is why that, in spite of its initial rejection, industrial PC's started to be used to carry on modern digital control theories. These systems, which present better characteristics such as more calculation capacity, less manufacturing dependency, an easy integration on communication systems and reduction of systems faults, in addition to an improvement on equipment robustness, are becoming more popular in the industrial environment –see, e.g. [1]-[9]-.

A similar development suffered by control hardware has happened to the control techniques. This is to say, the improvement of control hardware is involving the application of advanced control techniques (see, for example, [10]-[12] in the chemical and paper industrial context and [13]-[20] in the electromechanical context). On the other hand, controllers based on pole placement design method, which is not highly advanced, is in use on process control applications with, for instance, identifications methods for an automatic design of controller.

A gradual introduction of advanced control in industry has been produced because of the more strict specifications in the control of processes and development of microelectronic and hardware (DSP's, Industrial PC's, …). The use of advanced control methods, see e. g. [21]-[26], is necessary, because recent studies show that 80% of controllers in process control applications are poorly adjusted [27]. In order to find a solution to this problem, advanced control techniques have been developed like adaptive or autotuning techniques –see, for example, [28] and [29]-. These techniques adjust control loops automatically by only testing plant or process without external help.

This paper presents a Hardware-In-the-Loop experimentation environment for digital controllers applied to industrial processes using the pole location design method. This environment is formed by PC architecture equipment, multifunction I/O cards and the xPC-Target software tool of Mathworks. The main goal is to be capable of real-time experimenting, in its development stage, with controllers to be applied to industrial processes, see references [30]-[33].

Experimentation results described here are based on the using of the digital controller designed with the pole placement control theory. The pole placement design method simply attempts to find a controller that gives desired closed-loop poles, that is, the response of closed-loop can be determined. On the other hand, this experimentation environment is valid for any other type of advanced control.

2 System Description.

The experimentation laboratory is formed by an industrial PC, which is the controller, with a Pentium II processor to 700 Mhz and compact-PCI bus for connection with peripherals. A multifunction I/O board from *National Instruments* (PXI-6070E model) is allocated to acquire and send data, which enables to close the control loop on the plant. There is, as well, a process plant that can be a real plant or a model of the plant. In this case, A PC is used as emulator of the plant. Model emulator implemented is downloaded to the model PC so that emulates the plant behaviour. The PC used as model of the plant consists of a Pentium II MMX processor to 350 Mhz and a multifunction input-output board from National Instruments (PCI-6024E model) to receive control signal and send the output signal or process variable of the plant. Both, the industrial PC and the plant model PC, work as *targets* supervised by a desktop PC (*host*) where Matlab program, with toolboxes Simulink, Real Time Workshop and xPC-Target included, have been installed.

In fig.1 the experimentation environment of laboratory is depicted. In fig.2 the feedback control loop is shown, where the controller, *target* 1, sends the control signals or controlled variable through their I/O multifunction board. At the same time, the plant, *target* 2, collects the control signals and returns the output signal or process variable to the controller. The *host* is attended to the matter of downloading applications in each one of the *targets* and supervising its performance. The *host* also can change some parameters of experimentation environment, previously selected and prepared for this, for instance, reference signals or controller parameters.

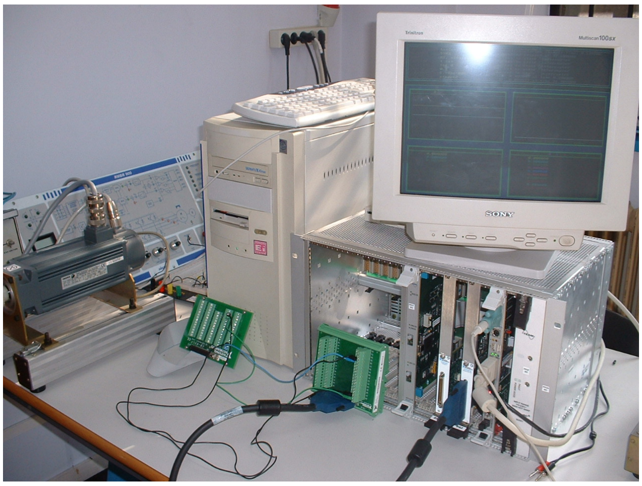

Figure 1. Image of HIL experimentation environment.

A *Matlab* tool, called *xPC-Target*, is the software used for implementation of HIL experimentation. xPC-Target is a solution for deploying, testing and prototyping real time systems using standard PC hardware [34]. It is an environment that uses a *target* PC, separate from a *host* PC, for running real time applications. xPC Target allows for adding I/O blocks to the model, and then use the *host* PC with Real-Time Workshop and a C/C++ compiler to create executable code.

In the *host*, with the help of *Simulink* tool of *Mathworks*, the plant model to be controlled is created. This model is compiled by a C compiler to use it in the Real-Time Kernel and then is downloaded to the *target* PC used for emulation.

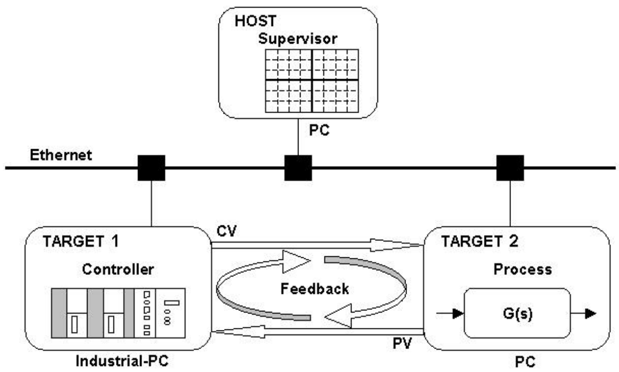

Figure 2. Description of system components for HIL experimentation.

Controller is created, in the same way, by the *host* in *Simulink*, but now using S-functions in language C and specific driver blocks for multifunction boards, that is allocated on the industrial PC. In fig.3 the *Simulink* model implemented for initial experimentations is shown. The model is formed by different *Simulink* blocks: controller block, named Pole-Placement Controller, is based on *Matlab* functions (S-functions) programmed in C language, I/O blocks of the National Instruments card, which are drivers to manage the card and signal display blocks, named Target Scope for *target* and Host Scope for *host*, present selected data for monitoring on graphic format.

The *Simulink* model used to experiment in real time is shown in fig. 3. The process of adjusting the controller, that want to be carried out, is based on three consecutive stages (see fig. 4). Process plant identification is realized to calculate an approximated model close to the real plant model. This process can be carried out in open or closed loop and as a result of that, identified model parameters are obtained. Afterwards, controller parameters are calculated from the identified model to obtain a desired dynamics. Finally, when the controller is adjusted, the feedback control is applied to the plant [35].

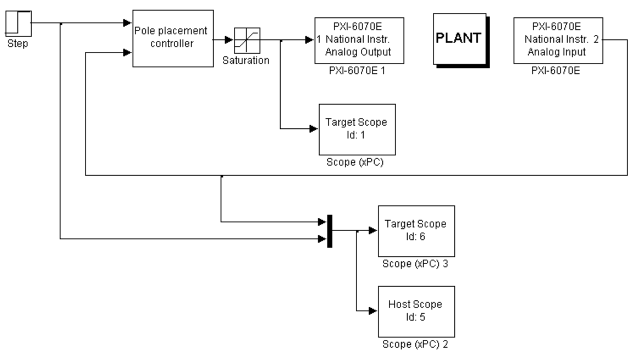

Figure 3. *Simulink* model of Pole placement controller.

Host PC is connected to *target* PC, where an operative Real-Time system kernel of *xPC-Target* is loaded. The kernel communicates the *host* and the *target* through TCP/IP, and executes in Real-Time code generated by *host*. In this way, real time plant or PC model control is carried out. At the same time, it is possible to display different signals (control, output signal, intermediate variables, reference signals, etc.) from *target* and monitor them from *host*.

3 Experimentation Examples.

The aim of the results presented here is verify the correct performance of the experimenting environment and show the complete process of create a digital controller. In this way, the pole placement design method has been used in the aforementioned toolbox, given the importance and generalization of such structure. On the other hand, any other design method can be easily implemented in the same manner.

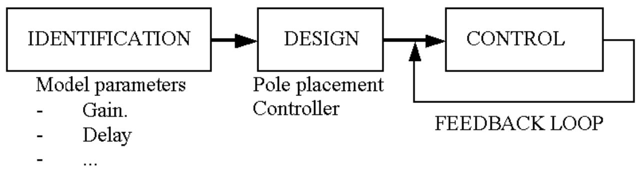

Figure 4. Stages of the autotuning process.

Moreover, the identification method to be used can be chosen given the type of the plant describing the industrial process to be handled and the controller to be implemented. Thus, the Areas method has been used in the first example, given that the process is described by a first-order plus time delay plant and we want to implement a self-tuning regulator (STR). On the other hand, the recursive least square (RLS) method will be utilized in the second example, given that the process may be described by a second order plant (to be used also when no prior knowledge is obtained about the process dynamics) and the goal is the implementation of an adaptive controller.

3.1 Example 1: STR applied to a first-order plus time delay process.

The first order plus dead time (FOPDT) plant (1) is selected for the experimentation given that the response of FOPDT describes many industry chemical processes [36], such as feed heat exchange, chemical reactors or product separation.

$$G(s) = \frac{1.2}{(10s+1)} e^{-0.5s} \qquad (1)$$

Supposing that the type of the process is previously known, the identification method may be selected accordingly. In this way, the very simple Areas Method will be used here.

Controller programming is carried out with two S-functions in C Language and *Simulink* blocks. The structure of the digital controller is implemented with direct programming (2) -see e.g. [2] and [4]-.

$$G_c(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} \tag{2}$$

where b_0, b_1 and b_2 are the coefficients of polynomial, that define the location of the zeros, and the parameters a_1 and a_2 are the coefficients of the controller characteristic equation. The controller is:

$$\begin{aligned} y(k) &= b_0 u(k) + b_1 u(k-1) + b_2 u(k-2) \\ &\quad - a_1 y(k-1) - a_2 y(k-2) \end{aligned} \tag{3}$$

This controller is programmed in a S-function according to the algorithmic description given by expression (3).

3.1.1 Identification method.

The identification method applied to recognize plant dynamics is the Areas method –see [37]-. By using process step response, the first order lag plus dead time model (4) can be obtained. Obtained model values are three: k (*gain*), T (*time constant*) and L (*dead time*). These values are utilized for calculate the discrete transfer function digital of the plant.

$$G(s) = \frac{K}{1 + sT} e^{-sL} \tag{4}$$

This method is based on the calculation of upper and lower areas (see fig. 5) and, from these values, calculate model parameters according to (5):

$$T = \frac{e\,A_1}{K} \qquad L = \frac{A_0 - e\,A_1}{K} \tag{5}$$

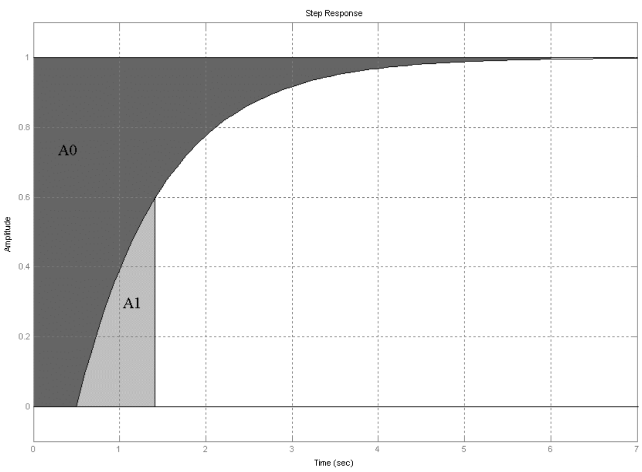

Figure 5. Areas method.

The plant, which this identification method is applied, needs to fulfil that its step response is monotonically increasing or asymptotically stable. The industrial process plant selected (1) verifies this condition; therefore, this identification method is valid to the process chosen for this experimentation.

3.1.2 Design method.

Pole placement design method has been selected to the experimentation process –see [2] and [4]-. This method calculates the transfer function coefficients of the digital controller to obtain the desired pole location of the system characteristic equation in closed-loop. The pole location is selected to determine the response of closed-loop system. The designer uses the continuous-time parameters of the characteristic polynomial in terms of factors of first and second order. From these parameters is a simple task to compute the discrete-time parameters.

The design procedure to obtain the pole placement desired uses the discrete-time transfer function of the plant model calculated and the characteristic equation desired. Solving the diophantine equation (6), the digital controller parameters are calculated.

$$A(z)S(z) + B(z)R(z) = A_{cl}(z) \tag{6}$$

where B/A is the discretized plant model, R/S is the controller, T is a precompensator and A_{cl} is the characteristic equation desired of the closed-loop system (fig. 6).

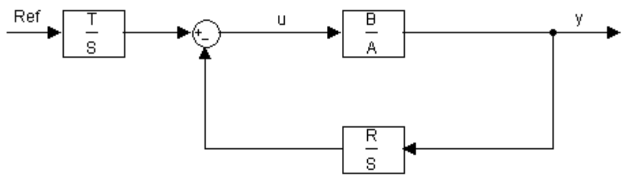

Figure 6. Pole-placement scheme.

3.2 Example 2: Adaptive control applied to a second-order process.

The availability of a digital computer permits the implementation of algorithms that automatically estimate the parameters of the discrete time models. It should be emphasized that the identification of the parametric discrete time models allows to obtain non-parametric models of the step-response or frequency-response type, with a far higher degree of accuracy with respect to a direct approach, an using extremely weak excitation signals. The identification of parametric sampled data models leads to a model of a very general use and offers several advantages over the other approaches. In addition, high performance identification algorithms, which have a recursive formulation tailored to real-time identification problems and to their implementation on micro-computer, have been developed. The fact that these identification methods can operate with extremely weak excitation signals is a very much appreciated quality in practical situations –see **¡Error! No se encuentra el origen de la referencia.** and [38]-.

On the other hand, very common industrial processes present oscillatory behaviour and may be described by second-order plants -see [1]-[7] and references therein-. For the test, the plant presented below is used.

$$G(s) = \frac{1.2}{s^2 + 3s + 2} \tag{7}$$

3.2.1 Identification method.

The Recursive Least Square (RLS) with constant forgetting factor algorithm is implemented for the identification of the plant –see e.g. [2] and [4]-.

Let describe the real process in discrete-time as

$$y_{k+1} = \sum_{i=1}^{n} a_i \, y_{k-i+1} + \sum_{i=0}^{m} b_i \, u_{k-i} + \varepsilon_k = x_k^T \, \theta_k + \varepsilon_k \tag{8}$$

and the estimated plant as

$$\hat{y}_{k+1} = \sum_{i=1}^{n} \hat{a}_i \, y_{k-i+1} + \sum_{i=0}^{m} \hat{b}_i \, u_{k-i} = x_k^T \, \hat{\theta}_k \tag{9}$$

where the estimated parameters and the input-output pairs are:

$$\hat{\theta}_k = \begin{bmatrix} \hat{a}_1 \\ \vdots \\ \hat{a}_n \\ \hat{b}_0 \\ \vdots \\ \hat{b}_m \end{bmatrix}_{1\times N=n+m+1} \qquad x_k = \begin{bmatrix} y_k \\ \vdots \\ y_{k-n+1} \\ u_k \\ \vdots \\ u_{k-m} \end{bmatrix}_{1\times N=n+m+1} \tag{10}$$

At each sampling time, the estimation error can be defined as

$$e_{k+1} = y_{k+1} - \hat{y}_{k+1} \tag{11}$$

In order to recursively minimize the expression (11), the algorithm presented below is executed at each sampling time

$$\hat{\theta}(t) = \hat{\theta}(t-1) + L(t)\left[y(t) - \varphi^T(t)\hat{\theta}(t-1) \right] \tag{12}$$

$$L(t) = \frac{P(t-1)\varphi(t)}{\lambda + \varphi^T(t)P(t-1)\varphi(t)} \tag{13}$$

$$P(t) = \left[P(t-1) - \frac{P(t-1)\varphi(t)\varphi^T(t)P(t-1)}{\lambda + \varphi^T(t)P(t-1)\varphi(t)} \right] \Big/ \lambda \tag{14}$$

where:

$$\varphi(t) = \begin{bmatrix} y(t-1) \\ \vdots \\ y(t-n) \\ u(t) \\ \vdots \\ u(t-m) \end{bmatrix} \tag{15}$$

and λ is the forgetting factor, normally between 0.9 and 1 in order to give the maximum weight to the most recent error. This type of profile is suited for the identification of slowly time varying systems.

3.2.2 Design method.

The design method is also based on the pole placement structure as described in section 3.1.2.

4 Results.

Next, the results of the experimentation on real-time over the two presented plants are presented sequentially.

4.1 Example 1: STR applied to a first-order plus time delay process.

Results of the experimentation example 1 presented are divided into three main parts: the first one is a step response, the second an identification of the model and, at last, control response already adjusted.

In the identification stage, the control signal is set to a step of 5 units of amplitude, then the supervisor program collects samples of the response (see fig. 7) until the signal arrives to the steady state. When all samples are stored in memory, the program calculates the identified model parameters, according to the Areas method.

Parameters of the identified model of first experimentation are shown in Table I. From these values, first order lag plus dead time model step response has been simulated through Simulink. Two responses (experimentation plant and identification model response) are shown in fig. 8. superimposed for a better comparative. Identified model response follows to the system and there is minimum errors in identifying.

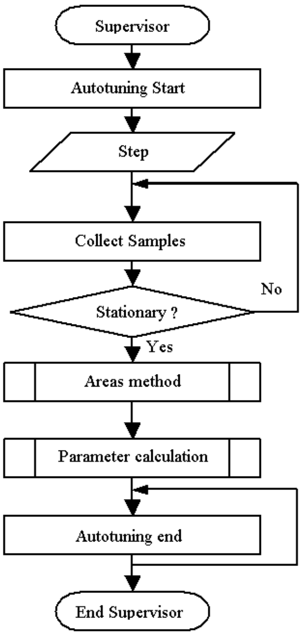

Figure 7. Functional flow diagram of the auto tuning supervisor program.

After identifying, plant is discretized to including it in the diophantine equation function. Solving the diophantine equation, the parameters of digital controller are obtained. In table II, calculated controller parameters by means of pole placement design method are shown. The characteristic equation desired for the closed loop is shown in expression (16).

$$A_{cl} = z^2 - 0.6z + 0.05 \qquad (16)$$

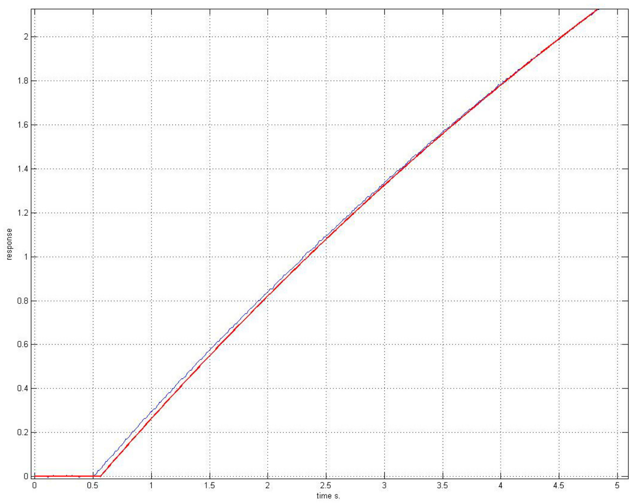

Figure 8. Open loop step response of real process and identification model comparative.

TABLE I

Gain	Time constant	Delay time
1,206055	9,835756 s.	0.561572 s.

TABLE II

R/S	T/S
$\dfrac{2.905631z}{z\ +0.150552}$	$\dfrac{3.859611z}{z+0.150552}$

The pole placement desired is selected to decrease rise time and no overshoot.

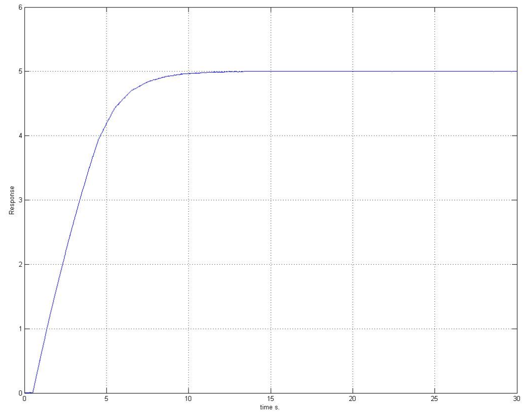

Figure 9. Closed loop step response of controlled process**.**

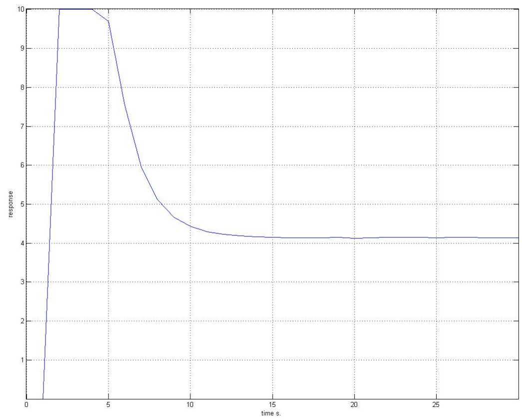

Figure 10. Signal control of controlled process.

Identification and calculation of digital controller stages are finished when calculated parameters have been loaded into the controller. Then, the supervision program block finishes its function and lets the digital controller block start to work. In this moment, plant is controlled in closed loop by digital controller. With the aim of verifying a correct performance, a step of 5 units on signal reference is introduced. In figure 9, controlled system response is shown before a change in the input signal. Control signal of the controlled plant before a step on input reference is shown in figure 10.

4.2 Example 2: Adaptive control applied to a second-order process.

The model of the process (7) is implemented in real-time by the Target 2 of the HIL system presented in section 2. On the other hand, the identification algorithm presented previously in section 3.2.1 –expressions (12)-(15)-, the design of the pole-placement controller and the real-time implementation of such controller is performed by the Target 1.

The results of the identification of the parameters of the plant are presented in Figure 11.

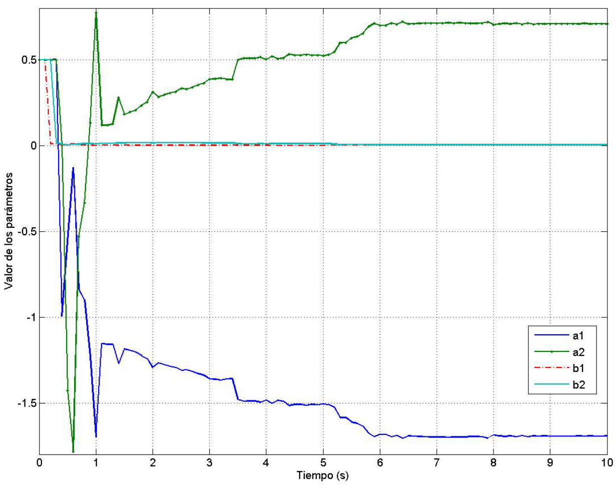

Figure 11. Estimation of the plant parameters.

As we can see, the estimated parameters stabilizes in 6 seconds and the final values are very close to the real values. Given that the sampling period is 0.1 seconds, the discrete-time plant obtained from the transfer function (7) is

$$G(z) = \frac{b_1 z + b_2}{z^2 + a_1 z + a_2} = \frac{0.005434\ z + 0.004916}{z^2\ -\ 1.724\ z + 0.7408} \qquad (17)$$

where the *real* parameters are, evidently, b_1=0.005434, b_2=0,004916, a_1=-1.724 and a_2=0.7408. By observing figure 11, the *estimated* parameters after 6 seconds are approximately b'_1=0.005542, b'_2=0,005107, a'_1=-1.7128 and a'_2=0.7303.

Next, the evolution of the parameters of the pole-placement controller applied to the plant is depicted in Figure 12. As we can observe, at the first sampling times the controller remains unchanged. It is due to a preventive measure that inhibit the recalculation of the controller when the estimation error is greater than 0.1.

Observing figures 11 and 12, we may say that the parameters of the controller require a longer period of evolution to stabilize that the estimation of the parameters of the plant. That is due to the sensibility of the equation (6) to little changes in the parameters of polynomials A(z) and B(z). Evidently, this effect must be greatly considered, given its practical consequences –see [39]-.

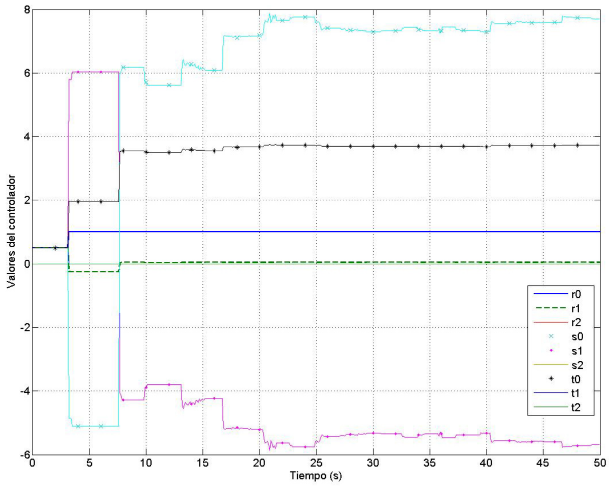

Figure 12. Adaptive controller parameters.

After 200 seconds, the obtained parameters for the controller are:

$$R(z) = z^2 + 0.03976\, z \qquad (18)$$

$$S(z) = 7.70315\, z^2 - 5.6935\, z \qquad (19)$$

$$T(z) = 3.72498\, z^2 \qquad (20)$$

By using expressions (18)-(20) and observing the controller structure given in Figure 6, an order reduction occurs and the controller is given in the table below.

TABLE III

R/S	T/S
$\frac{z+0.03976}{7.70315\,z-5.6935}$	$\frac{3.72498\,z}{7.70315\,z-5.6935}$

It is also important to check out the output of the plant, specially during the transient due to the estimation period, in order to detect possible out-of-range behaviors and dangerous dynamics. Such operation is presented in figure 13, where the reference signal and the corresponding plant output is graphed.

Observing Figure 13, it is clear that no out-of-range behavior occurs –probably thanks to the preventive inhibition of the controller recalculation during the estimation of the parameters of the plant-. Besides, the output closely follows the desired trajectory for the closed loop, defined previously by the characteristic equation (16).

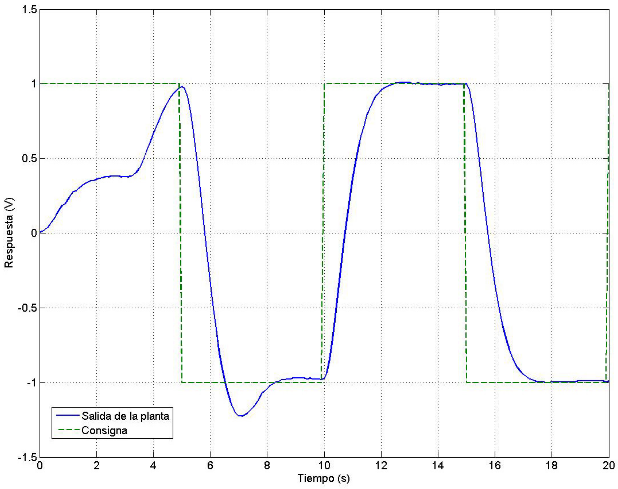

Figure 13. Reference signal and plant output .

Finally, it is very important to observe the control signal of the adaptive scheme, given that it is well known that saturation may occur, specially at the initial moments.

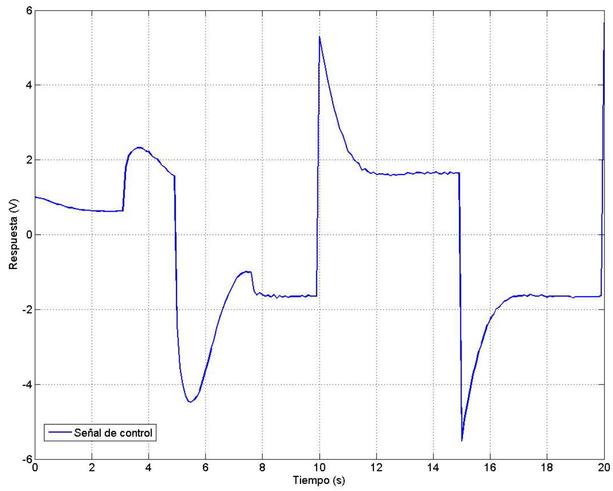

Figure 14. Control signal.

Well, observing the Figure 14 it is proven that no saturation happens and that the control signal is confined at every moment in a sensible range.

Evidently, the experiment can be easily extended by considering additional effects as different levels of measurement noise in the inputs, different codification, etc,… In addition, other values can be considered of the forgetting factor, the initialization of the parameters of the plant and the controller, the security level for the inhibition of the controller recalculation, etc,…

5 Conclusions.

A HIL experimentation environment for advanced controllers has been developed to experiment, in a realistic way, control loops in real time. The main objective of the environment is carrying out HIL experimentations in real-time to check a correct performance of controllers to be applied to productive processes, without using an expensive plant prototype.

In order to check out the performance of this experimentation environment, digital controllers has been applied to typical plants of the industrial environment. Specifically, two different types of controllers (self-tuning regulator and adaptive RLS controller) have been implemented and tested on two different types of plants (first-order plus time delay and second order plants). After observing the responses obtained in the experimentation, a correct performance of such a environment has been verified.

This system establishes a complete laboratory that allows experimentations with all types of digital real-time controllers. By means of this environment, the implementation of digital controllers is very easy by using *Simulink* blocks. Moreover, other types of advanced controllers that need other requirements not supplied by *Simulink* blocks can be developed. The designer may program blocks in a high level language like C language.

The experimentations can also be applied to real plants, with actuators of the real process, as well as to emulated plants, by means of the connection to one or several PC's that act like emulators. The system allows us to minimize the development time and the costs of creating prototypes. In the development stages, the experimentation can be carried out with real process, when possible, or with models, when is very expensive to get a real process into the lab. Nowadays, this HIL environment is in use by the research group in order to experiment with real positioning systems, included in machine-tools industrial processes.

ACKNOWLEDGMENT

This work has been partially supported by the University of the Basque Country through Project UPV05/118 and Basque Government (Project S-PE05UN09).

References:

[1] R. C. Dorf and R. H. Bishop, *Modern control systems*, Prentice Hall, 2008, 11th edition.

[2] I. D. Landau, G. Zito, *Digital Control Systems*, Springer-Verlag. 2006.

[3] G. Goodwin, S Graebe and M. Salgado, *Control System Design*, Prentice-Hall, 2001.

[4] K. J. Aström, B. Wittenmark, *Computer-controlled Systems. Theory and design*, Prentice-Hall, Inc. 1997.

[5] C. L. Phillips and T .Nagle., *Digital control systems. Analysis and design*, Prentice-Hall, Englewood Cliffs, New Jersey. 1995. 3rd edition.

[6] B. Kuo, *Digital control systems.* Oxford University Press. New York. 1992. 2nd edition.

[7] K. Ogata, *Modern Control Engineering*, Prentice-Hall, 2002, 4th Edition.

[8] J. K. Taylor, "North American and Western European market demand for ruggedized/indutrial computer systems," *Venture Development Corporation*, Available: www.vdc-copr.com

[9] J. E. Gulliksen, J. Taylor, M. Foley, "The worlwide market for integrated ruggedized/inudstrial grade computers," *Venture Development Corporation*, Available: www.vdc-corp.com.

[10] D. Cunninghan, "Adaptive control applications in pulp and paper," *Conference on Electrical and Computer Engineering*, 1993.

[11] B. Cough, J. Kay, G. Dumont, "Advanced model based control for continuous process industries," *American Control Conference*, ACC'97. 1997.

[12] M. Huzmezan, B. Gough, S. Kovac, "Advanced control of batch reactor temperature," *American Control Conference*, ACC'02. 2002.

[13] I. Hassanzadeh, M. Darabi, S. Hassanzadeh, S. Khanmohammadi, 'A Self Tuning Fuzzy Controller Design for a Robot Manipulator Using Hardware in the Loop (HIL) Method', *WSEAS Transactions on systems*, Vol. 5, No. 5, May 2006, pp. 1196-1203.

[14] A. Bagheri, M. E. Felezi, P. N. Mousavi, 'Adaptive Control and Simulation of a Seven-Link Biped Robot for the Combined Trajectory Motion and Stability Investigations', *WSEAS Transactions on systems*, Vol. 5, No. 5, May 2006, pp. 1214-1222.

[15] J. Ferreira, M. Crisostomo, A. Coimbra, 'Walk Control of a Biped Robot Using Neuro-Fuzzy', *WSEAS Transactions on systems*, Vol. 5, No. 12, Dec 2006, pp. 2892-2898.

[16] M. Camus, A. Cardon, 'An Adaptive System to Control Robots', *WSEAS Transactions on systems*, Vol. 5, No. 12, Dec 2006, pp. 2804-2812.

[17] J. Du, Y. Zhang, T. Lü, 'Unmanned Helicopter Flight Controller Design by Use of Model Predictive Control', *WSEAS Transactions on systems*, Vol. 7, No. 2, Feb 2008, pp. 81-87.

[18] A. Yang, J. Wu, W. Zhang, X. Kan, 'Research on Asynchronous Motor Vector Control System Based on Rotor Parameters Time-Varying', *WSEAS Transactions on systems*, Vol. 7, No. 4, April 2008, pp. 384-393.

[19] J. Jing, Y. Yingying, F. Yanxian, 'Optimal Sliding-mode Control Scheme for the Position Tracking Servo System', *WSEAS Transactions on systems*, Vol. 7, No. 5, May 2008, pp. 435-444.

[20] D. Popescu, D. Selisteanu, L. Popescu 'Neural and Adaptive Control of a Rigid Link Manipulator', *WSEAS Transactions on systems*, Vol. 7, No. 6, June 2008, pp. 632-641.

[21] R. Barcena, M. De La Sen, I. Sagastabeitia and J.M. Collantes, 'Discrete control for a computer hard disk by using a fractional order hold device', *IEE Proc. Control Theory Appl.*, 2001, 148 (2) pp. 117- 124.

[22] R. Barcena and M. De La Sen, 'On the sufficiently small sampling period for the convenient tuning of fractional-order hold circuits', *IEE Proc. Control Theory Appl.*, 2003, 150 (2) pp. 183- 188.

[23] R. Barcena and A. Etxebarria, 'Mitigating the effect of fractional time delays in hybrid control systems', *IET Control Theory Appl.*, 2008, vol. 2 (1) pp. 24- 31.

[24] Y. Chang, S. Chen, 'Static Output-Feedback Simultaneous Stabilization of Interval Time-Delay Systems', *WSEAS Transactions on systems*, Vol. 7, No. 3, March 2008, pp. 185-194.

[25] K. Shahriari, S. Tarasiewicz, O. Adrot, 'Linear Time-Varying Systems: Theory and Identification of Model Parameters', *WSEAS Transactions on systems*, Vol. 7, No. 5, May 2008, pp. 445-454.

[26] R. Bárcena, M. De La Sen and I. Sagastabeitia, 'Fractional Order Hold Tuning using Neural Networks', *Proceedings of the 2001 American Control Conference, ACC'2001*, Arlington, Virginia. EEUU, June 25-27. 2001.

[27] A. O'Dwyer, *Handbook of PI and PID controller tuning rules*, Imperial College Press, 2003.

[28] D. W. Clarke, 'Self-tuning control of nonminimum-phase systems', Automatica, 20 (5) pp. 501-517, 1984.
[29] M. M'saad, R. Ortega, I. D. Landau: 'Adaptive controllers for discrete-time systems with arbitrary zeros: an overview', *Automatica*, 21 (4) pp. 113-423, 1985.
[30] J. Ledin, *Embedded control systems in C/C++*, CMP Books, 2004.
[31] D. M. Auslander, J. .R. Ridgely, J. C. Jones, "*Real-Time software for implementation of feedback control*," The Control Handbook, CRC Press, Inc. 1996, pp. 323-343.
[32] S. Serhan, I. Salah, 'Performance Evaluation of a Multiprocessor System with Real Time Unit', *WSEAS Transactions on systems*, Vol. 6, No. 6, June 2007, pp. 1178-1183.
[33] K. Altaf, A. Akhtar, S. Rehman, J. Iqbal, 'Design, Implementation and Real-Time Digital Control of a Cart-Mounted Inverted Pendulum using Atmel AVR Microcontroller', *WSEAS Transactions on systems*, Vol. 6, No. 6, June 2007, pp. 1184-1190.
[34] *xPC Target User's Guide*, MathWorks, Inc. Ma, 2008.
[35] T. E. Marlin, *Process control: designing processes and control systems for dynamic performance*, McGraw Hill, New York 1995.
[36] W. L. Bialkowski, "*Control of the pulp and paper making process*," The Control Handbook, CRC Press, Inc. 1996.
[37] Y. Nishikawa, N. Sannomiya, T. Ohta, H. Tanaka, A method for auto-tuning of PID control parameters," *Automatica*, Volume 20, Issue 3, May 1984, pp. 321-332.
[38] K. J. Aström, B. Wittenmark, 'Self-tuning controllers based on pole-zero placement', *IEE Proc. D, Control Theory and Applications*, 1980, 127 (3) pp. 120-130
[39] R. H. Middleton, 'Trade-offs in linear control system design', *Automatica*, 27, (2), pp. 281-292, 1991.