Real-time Control Applications for Nonlinear Processes Based on Adaptive Control and the Static Characteristic

CIPRIAN LUPU, DUMITRU POPESCU, ANDREEA UDREA
Department of Automatic Control and Computers Science
University "POLITEHNICA" of Bucharest
313, Splaiul Independentei, sect.6, Bucharest
ROMANIA
cip@indinf.pub.ro, dpopescu@indinf.pub.ro, andreea@indinf.pub.ro, http://www.acs.pub.ro

Abstract: - Nonlinear systems control is a problem of great interest and some solutions are found in the area of adaptive control. In this paper we proposes a real-time control algorithm based on a model reference adaptive control mechanism for a class of nonlinear systems. In order to obtain the reference model, an off-line procedure is used, while the controller's parameters are determined on-line, using a static gain adaptation criterion. In order to demonstrate the relevance of this approach, several tests on nonlinear processes, using a real-time applications package developed in LabWindows/CVI were made. A solution, designed for a class of nonlinear systems, based on parametric adaptation of an RST controller is presented. Issues on the stability and robustness of the solution are also discussed.

Key-Words: model reference adaptive control, adaptive law, real-time application

1 Introduction

Nonlinear systems control is one of the greatest challenges of modern times. One of the solutions for this problem is found in the area of adaptive systems, which in the last 20 years, due to computational advancement, was largely investigated [1], [2], [5].

Adaptive control encountered many challenges in the field of real-time systems, which do not have precise classical models admissible to existing control designs [4]. It is an essential condition for the real-time control systems to preserve the closed-loop performances in case of non-linearity, structural disturbances or process uncertainties.

There are several adaptive design approaches that work for different types of nonlinear systems: model reference adaptive control, adaptive backstepping control etc [1], [4] and other approaches like multiple – model control [8].

This paper proposes a real-time control algorithm based on a reference model adaptive control mechanism for a class of nonlinear systems. This algorithm is used in order to implement a numerical control platform that will be described in the second half of the paper. A brief comparison to the multiple model numerical implementations will be performed in the last part of section 4.

The reference model is computed off-line, while the controller's parameters are determined on-line, via a static gain adaptation criterion.

For a real time implementation, the algorithm can be divided into two independent parts:

- *the identification algorithm*
  – is used in order to obtain the static model of the plant (Fig. 1)
  – this step gives the actual gain of the plant;
- *controller parameters' adaptive law* (Fig. 2)
  – which recalculates the controller's parameters.

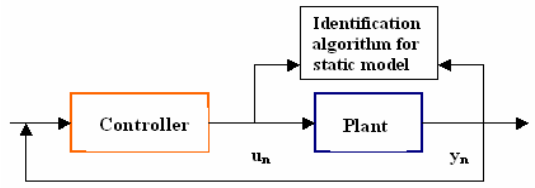

Fig. 1. Identification algorithm used to determine the static model of the plant

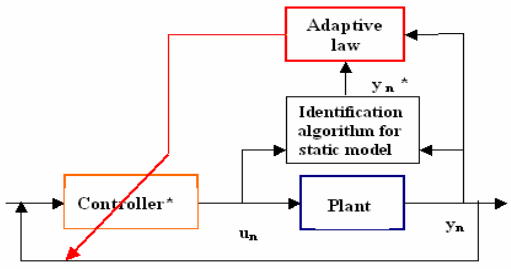

Fig. 2. Adaptive law influence on the controller's parameters

The proposed structure implies a series of steps that are made before the real-time running of the application:

- identification and storage of the static characteristic of the process – a series of measurements are performed in order to determine its mean value;
- selection of important inflexion points on the static characteristic - the specific points where the process changes visible its dynamics are marked and saved;
- identification of the process model (dynamical) in a randomly chosen functioning point;
- design of the regulation algorithm (PID, RST) - based on the dynamic mode; in this paper we will use a RST numerical controller.

In the next section, the most important steps are described.

2 Adaptive algorithm - off-line design

In this section, the actions that take place off-line (before the real-time running of the application), will be explained.

2.1 Plant's static model determination

This operation is based on several experiments. The discrete command - *u(k)* is gradually increased and decreased. The corresponding stabilized process output - *y(k)* is measured. The command *u(k)* covers all possible values: from 0 to 100% (in percentage representation).

Due to the fact that the process is disturbed by noise, usually, the static characteristics (measured at different times) are not identically.

A "global" static characteristic is calculated for using in the control algorithm. This characteristic is obtained by meaning of all correspondent positions of these experiments.

This operation is present in figure 3. The graphic between two "mean" points can be obtained using the extrapolation procedure.

According to system identification theory, the dispersion of the process trajectory can be found using expression (1):

$$\sigma^2[n] \cong \frac{1}{n-1}\sum_{i=1}^{n} y^2[i],\ \forall n \in N^* \setminus \{1\} \qquad (1)$$

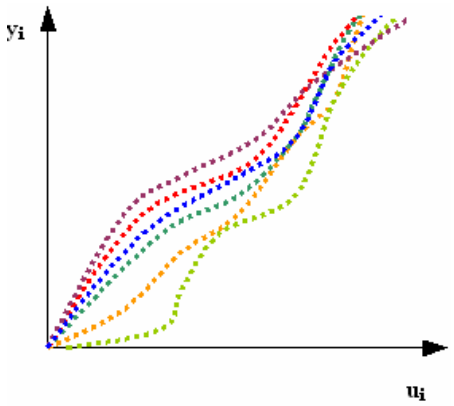

Fig. 3. Determination of static characteristic of the process (in light blue line - the final characteristic)

This can express a measure of superposing of the noise onto the process and process nonlinearities and is very important for the designed control algorithm robustness.

Another option is to find the position and the value ***mg*** of the maximal distance from the "mean" characteristic.

2.2 Control law design

The control algorithm's purpose is to reject the disturbances and to compensate for the differences between real process's gain and the estimated process gain.

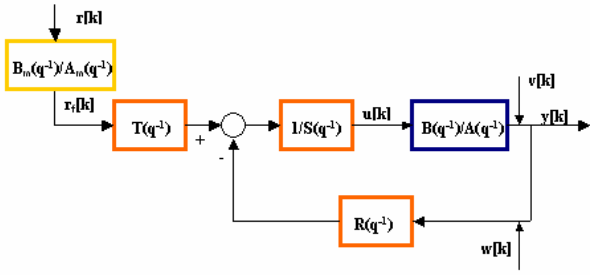

Fig. 4. RST control algorithm structure

A large variety of control algorithms have been proposed over the years and can be used here: PID, RST, fuzzy [2], [9], [10], [11].

For this study, we have selected a RST algorithm. This control algorithm's values are obtained by using a pole placement designed procedure [3]. In figure 4 the RST numerical algorithm structure is presented.

The R, S and T polynomials have the following form:

$$R\left(q^{-1}\right) = r_0 + r_1 q^{-1} + ... + r_{nr} q^{-nr}$$
$$S\left(q^{-1}\right) = s_0 + s_1 q^{-1} + ... + s_{ns} q^{-ns} \quad (2)$$
$$T\left(q^{-1}\right) = t_0 + t_1 q^{-1} + ... + t_{nt} q^{-nt}$$

where n_r, n_s and n_t are their degrees.

An imposed trajectory is sometimes useful. It can be generated using a trajectory model generator that can has the following form:

$$y^*(k+1) = \frac{B_m(q^{-1})}{A_m(q^{-1})} r(k) \quad (3)$$

where the A_m and B_m polynomials have these forms:

$$A_m(q^{-1}) = 1 + a_{m1} q^{-1} + ... + a_{mn_{Am}} q^{-n_{Am}}$$
$$B_m(q^{-1}) = b_{m0} + b_{m1} q^{-1} + ... + b_{mn_{Bm}} q^{-n_{Bm}} \quad (4)$$

The algorithm using pole placement design procedure is based on the identified process's model.

The plant model is obtained considering an ARX form:

$$y(k) = \frac{q^{-d} B(q^{-1})}{A(q^{-1})} u(k) \quad (5)$$

where:

$$B\left(q^{-1}\right) = b_1 q^{-1} + b_2 q^{-2} + ... + b_{nb} q^{-nb}$$
$$A\left(q^{-1}\right) = 1 + a_1 q^{-1} + ... + a_{na} q^{-na} \quad (6)$$

The identification is made in a process operating point. In order to obtain the coefficients of the *A* and *B* polynomials, recursive least squares algorithm exemplified in the next relations and fully presented in [5] has been used:

$$\hat{\theta}(k+1) = \hat{\theta}(k) + F(k+1)\phi(k)\varepsilon^0(k+1), \forall k \in N$$
$$F(k+1) = F(k) - \frac{F(k)\phi(k)\phi^T(k)F(k)}{1+\phi^T(k)F(k)\phi(k)}, \forall k \in N \quad (7)$$
$$\varepsilon^0(k+1) = y(k+1) - \hat{\theta}^T(k)\phi(t), \forall k \in N,$$

with the following initial conditions:

$$F(0) = \frac{1}{\delta} I = (GI) I, 0 < \delta < 1 \quad (8)$$

The estimated $\hat{\theta}(k)$ represents the parameters of the polynomial plant model and $\phi^T(k)$ represents the measures vector.

This approach allows the users to verify, and if is necessary, to calibrate the algorithm's robustness. The following expression presents the disturbance-output sensibility function.

$$S_{vy}(e^{j\omega}) \overset{def}{=} H_{vy}(e^{j\omega}) = \frac{A(e^{j\omega})S(e^{j\omega})}{A(e^{j\omega})S(e^{j\omega}) + B(e^{j\omega})R(e^{j\omega})}, \quad \forall \omega \in R \quad (9)$$

At the same time, the negative maximum value of sensibility function represents the module margin.

Another approach is to impose that the gain's limit to be greater or equal to the process static characteristic maximal distance:

$$\Delta G \geq mg \quad (10)$$

For our application, a controller that has sufficient robustness was designed.

3 Adaptive algorithm - on-line design

This section presents and arguments the adaptive algorithm which consists in following phases:

- establish the plant's gain; this is made considering the process position relatively to the inflexion points of the static characteristic. The gain is determined based on the inflexion point that is immediately inferior;
- gain filtering; is made using a very simple filter(first order). It is necessary in order to prevent the rough modification of the controller's parameters. A major change can lead to great discrepancies in the algorithm;
- RST controller adaptation; adaptation of the R and T polynomials coefficients in order to maintain the product between the plants gain and the controllers gain constant, so that the initial (off-line) closed-loop performances remain unchanged.

The adaptive structure based on a RST controller is presented in figure 5.

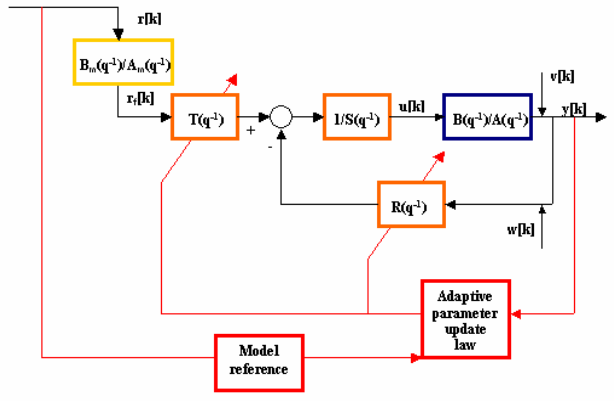

Fig. 5. Implemented real-time adaptive scheme based on a RST controller

3.1 Plant's gain

The plant's gain is determined based on the inflexion point that is immediately inferior – P_{i-1} considering the process position at moment k - P_p.

$$K_{plant} = \frac{y_p - y_{i-1}}{u_p - u_{i-1}} \tag{11}$$

This formula was chosen from a set of 3 versions. We have tried to determine the gain based on the rapport between the command u_k and the process output y_k at each sampling period:

$$K_{plant} = \frac{y_p}{u_p} \tag{12}$$

and also based on the rapport determined by the differences between two successive commands and output values:

$$K_{plant} = \frac{y_p - y_{p-1}}{u_p - u_{p-1}} \tag{13}$$

but, the process is affected by disturbances, the measurements contain noise and the command may oscillate, so this approach couldn't been used, although, in these cases, the model reference wouldn't have been necessary. This solution might have been useful for a linear process.

3.2 Gain filtering

In order to prevent the rough modification of the controller's parameters, the actual plant gain is filtered using a first order filter [12]:

$$C(q^{-1}) = \frac{C_1}{1 + C_2 q^{-1}} \tag{14}$$

The filtered gain is:

$$K_f = \frac{C_1}{1 + C_2 q^{-1}} K_{plant}. \tag{15}$$

The usage of this filter is augmented by the following fact. Because:

$$u(k) = \frac{1}{s_0}[-\sum_{i=1}^{n_S} s_i u(k-i) - \sum_{i=0}^{n_R} r_i y(k-i) + \sum_{i=0}^{n_T} t_i y^*(k-i)] \tag{16}$$

is an expression that links the controller's parameters to $u(k)$, $y(k)$ and $y^*(k)$, it is a demand not to abruptly modify these coefficients and so the command. The filter attenuates possible brusque changes.

3.3 Adaptive law

The adaptive law compares the output of the system to the output values of the precalculated model and modifies the controllers' parameters in order to maintain the product between the controllers' gain and the plants' gain constant:

$$K_{controller} \cdot K_{plant} = ct. \tag{17}$$

That implies that the controller's parameters are to be recalculated in order to satisfy relation (17).

The justification for the parameters adaptation procedure is the following: the transfer function for the closed loop system has the form:

$$H(q^{-1}) = \frac{\frac{T(q^{-1})B(q^{-1})}{S(q^{-1})A(q^{-1})}}{1 + \frac{R(q^{-1})B(q^{-1})}{S(q^{-1})A(q^{-1})}} \tag{18}$$

The model's gain is:

$$K_{plant} = \frac{\sum_{i=0}^{n_B} b_i}{1 + \sum_{j=1}^{n_A} a_j}. \tag{19}$$

If the static characteristic varies, creating an inflexion point, we have a new K_{plant}' and we need to recalculate the value for the controller's gain in order to satisfy the adaptive law.

The controller's gain would be:

$$K_{controller}' = K_{controller} \frac{K_{plant}'}{K_{plant}} \quad (20)$$

Making the notation:

$$F = \frac{K_{plant}'}{K_{plant}} \quad (21)$$

where F is called correction factor, we have:

$$K_{controller}' = K_{controller} F \quad (22)$$

From this and in order to maintain unchanged relation (17), the rapports:

$$rap_1 = \frac{\sum_{i=0}^{n_T} t_i}{1+\sum_{j=1}^{n_S} s_j}$$

and

$$rap_2 = \frac{\sum_{i=0}^{n_R} r_i}{1+\sum_{j=1}^{n_S} s_j}$$

must be multiplied by F.

If we multiply the parameters t_i and r_i of the controller's polynomials by F – the correction factor:

$$T_i , r_i \mid xF \quad (23)$$

we satisfy the adaptive law.

These parameters modification do not influence the stability of the system as it can be seen from the experimental results.

4 Control platform description

The theoretical solution was numerically implemented in order to demonstrate its performances. We have used LabWindows/CVI environment to develop an experimental control platform with a user-friendly interface.

Another needed feature is a real, effective process or a plant simulator. A series of these process simulators have been developed. The simulators are created so that they generate nonlinear characteristics. One of our applications will be used and explained in the experimental results section.

4.1 Control platform description

The interface for the control platform presents a series of objects for graphical display.

Figure 6 holds the interface for the real-time control platform.

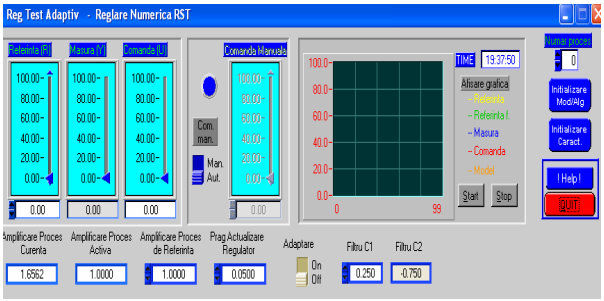

Fig. 6. Adaptive real-time control platform for nonlinear processes

On the left, there are three cursors that represent, in this order:

- the set point value – that can be modified by the user
- the process output – determined by the controller parameters and the controlled process
- the command value – calculated by the implemented numerical control algorithm.

Their values can vary between 0 and 100%. The exact value, at a moment of time, is written with two decimals bellow the corresponding cursor.

On the right, there is a dynamic graphic that shows:

- in yellow – the set point value
- in green – the filtered set point value
- in red – the command
- in blue – the output
- in orange – the model evolution.

The graphic is actualized at each sample period of time and one grid unit contains twenty sample periods.

As is has been earlier mentioned, a set of process simulators was developed, each describing a physical nonlinear process.

In order to select one of them to work with this control platform, its id value must be specified in the field Process Number.

For the control of the specified process, its model is realized and a set of parameters for the RST controller is calculated.

Using the interface sub – window presented in figure 7, one can upload the file that contains these data.

The dimension for the models and possible controllers is limited. The maximum dimension for the polynomial degrees for A and B, defining the identified model, is 5 and for the controller polynomials R, S and T, is 7.

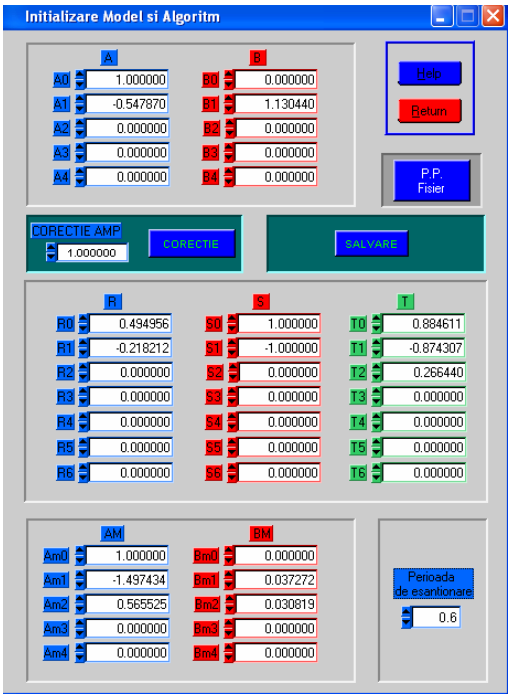

Fig 7. RST controller and model initial values

The reference can be filtered by using a maximum degree of 5 for A_m and B_m polynomials.

These limits are highly permissive for a good model – controller evolution.

Also, as it can be observed, here is the place where the value, in seconds, for the used sample period value must be specified (bottom right corner).

Another vital part of the application is shown in figure 8. In this sub – window, the user can specify the number of inflexion points that he considers important (maximum 10 special points) and select them directly on the static characteristic of the considered process using the mobile OX, OY axes (in blue).

The corresponding pair $(u(k), y(k))$, for the selected points, is kept in the algorithm memory and used during the real-time evolution.

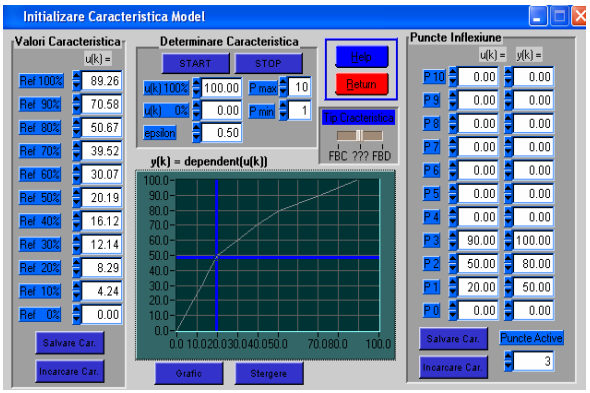

Fig. 8. Initialization of the models characteristics and of the inflexion points

4.2 Control platform implementation outlines

For a correct implementation of the numerical algorithm, two important aspects should be respected at all time:

- correct memory allocation
- memory actualization at each step consisting in a sampling period.

The command is calculated using the coefficients of the control polynomials and previous values of the command and process output.

It is absolutely necessary that, at all time, the controller parameters and all the previous command and output values are kept safe in algorithms the memory.

At each step, the actualization of the command value, with all the implied calculations must be waited for.

Otherwise, there will appear gaps in the command algorithm, which can lead to dangerous behaviors like entering in instability. The value for the command, for one sample period of time, is expressed in relation (16), where n_S, n_R , n_T give the corresponding polynomials degrees and, more important, from the implementation point of view, the memory dimension for the software implementation of the algorithm.

For example, if n_R=3, then there should be reserved four memory locations for the process's output: $y(k)$, $y(k-1)$, $y(k-2)$, $y(k-3)$. The same rule applies for $u(k)$ and $y^*(k)$.

So, in order to implement this solution, one must be interested in the control algorithm and, eventually, the trajectory's model generator.

In order to implement a real-time control algorithm, other than the realization of the user interface, an infinite loop with some specific steps should be considered.

A single iteration of the continuous monitoring program implies the following steps:

1. - data acquisition – the control platform searches for the process or simulator communication file, opens it and reads the actualized output of the process:

```
sprintf(cale_fisier,"%sP_YK.DAT",cale_director);
handlef_yk=fopen(cale_fisier,"r");
    if(handlef_yk==NULL){
        fseek(handlef_yk,0,0);
        fscanf(handlef_yk,"%f\n",&mas);
        fclose(handlef_yk);}
```

2. - computation of the filtered trajectory - (when this is used);

3. - control law computation –

3.1. – based on the acquired data, determination of the current plant's gain;

3.2. – if necessary, actualization of controller's gain, using the correction factor; relation (21)

3.3. – effective generation of the command from relation (16).

4. - sending the command value to the process- by writing the new value in the communication file.

```
sprintf(cale_fisier,"%sP_UK.DAT",cale_director);
handlef_uk=fopen(cale_fisier,"w");
fseek(handlef_uk,0,0);
fprintf(handlef_uk,"%f",val_com_dc);
fclose(handlef_uk);
```

5. - process evolution graphical display – all five graphics are actualized ;

6. - actualization of the algorithm's memory for the new iteration. For example, when $n_R = n_S = n_T = 2$ and (24) gives the algorithm's memory actualization for the next iteration:

$$\begin{aligned} u(k-1) &= u(k); \\ y(k-1) &= y(k); \\ y^*(k-1) &= y^*(k). \end{aligned} \qquad (24)$$

These steps are always performed during a sample period.

Comparing this implementation to the multiple – model solution for nonlinear fast processes proposed in [8], a series of advantages can be outlined: the number of used model-algorithms pairs is reduced to a single pair so the used memory is reduced. The algorithms switching problem from the multiple model control algorithm disappears, for the gain adaptation, the transition between two regions delimited by an inflexion point is easily controlled by using gain filtering (15). The number of operations is over all of the same order.

5 Experimental results

In order to demonstrate the applicability of this solution, a nonlinear process consisting in a tank with a filling point and multiple evacuation points was chosen as an example. The purpose of the control system is to maintain constant the liquid level in the tank.

A process simulator has been created using LabWindows/CVI:

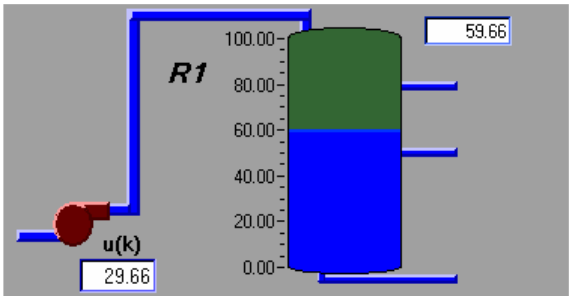

Fig. 9. Tank with single filling-multiple evacuation points simulator

The process simulator permits adding disturbances, in order to test the controller dynamics in real functioning.

Also, the evolution of the liquid in the tank is graphically represented.

The simulator communicates with the control platform using a dedicated communication file.

Using this process simulator, one can generate a nonlinear characteristic.

The input-output characteristic for this system can be approximated by a set of medium input and output values and has the form showed in the next figure 10.

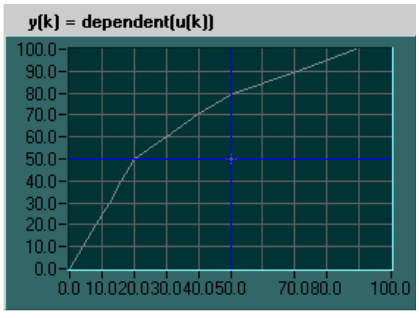

Fig. 10. Input-output characteristic for this system

In order to test the adaptive controller (figure 11), one must load this characteristic – model reference in the adaptive controller real-time software application and select a number of inflexion points, where the process dynamic changes.

On this characteristic, a set of inflexion points can be easily observed and learned, so that the new process's gain and controller's parameters will be calculated by using the adaptive law.

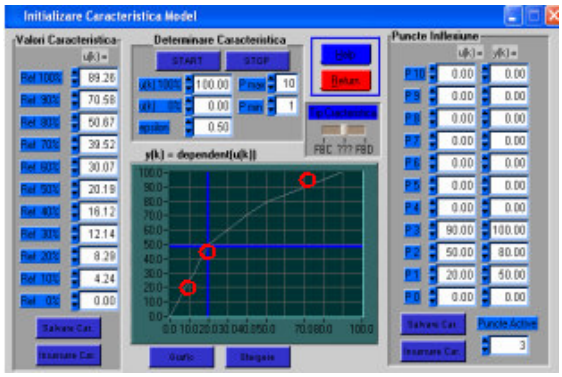

Fig. 11. Choosing the inflexion points on the static characteristic

Accordingly to figure above, we've identified the following four functioning intervals (0-20%), (20-50%), (50-90%), respectively (90-100%).

In order to identify the model for the first interval, a sampling period T_e=0.6 sec was used and least-squares identification method from Adaptech/WinPIM platform [13] was employed:

$$M_1(q^{-1}) = \frac{1.13044}{1-0.54787q^{-1}}$$

For this model, we have computed the corresponding RST control algorithm using a pole placement procedure and the Adaptech/WinREG platform.

For the poles placement procedure we've used a second order system, defined by the dynamics natural frequency ω_0 = 0.5 and the damping factor ξ=0.95 for tracking performances and ω_0 = 1.25, ξ = 0.8 for disturbance rejection performances, keeping the same sampling period as for identification T_e=0.6 sec.

The obtained parameters for the first functioning interval are as it follows:

$$R_1(q^{-1}) = 0.494956 - 0.218212\ q^{-1}$$
$$S_1(q^{-1}) = 1.000000 - 1.000000\ q^{-1}$$
$$T_1(q^{-1}) = 0.884611 - 0.874307\ q^{-1} + 0.266440\ q^{-2}$$

These initial values for the RST controller, are loaded into the simulator, see figure 7.

The pair model – controller can be identified, also, on another region and used in the same way.

Using this application, a few tests were made in order to demonstrate the fact that the adaptive control mechanism that was implemented can guarantee the closed loop stability under changes of reference and disturbances influence.

We first impose a disturbance of 1% and we test the performances for changes of reference:

- from 10% (where the first pre calculated RST algorithm is active) to 40% (where, normally, because the static gain value change the RST parameters were recalculated (by the adaptive law) – figure 12

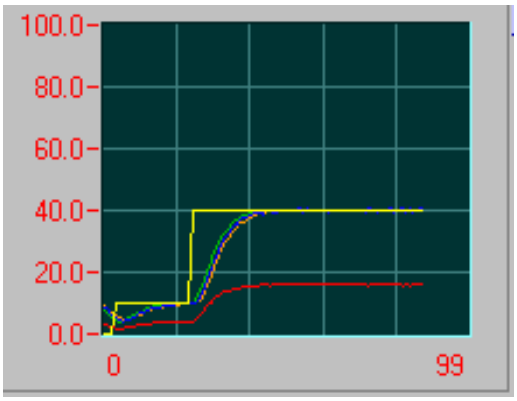

Fig. 12. Performances when changing the reference from 10% to 40%(reference - yellow, filtered reference – green, system output – blue, command – red, model - orange)

- from 40% to 60% - figure 13;
- from 60% to 90% - figure 14;
- from 90% to 30% - figure 15.

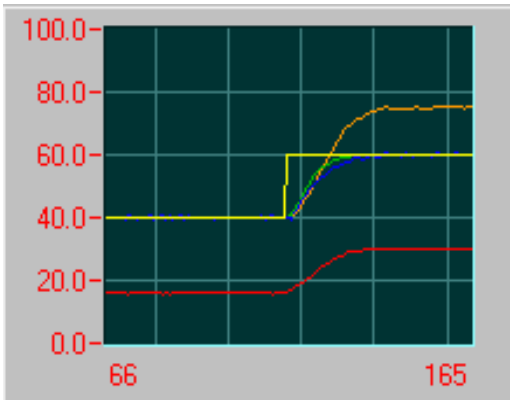

Fig. 13. Performances when changing the reference from 40% to 60%

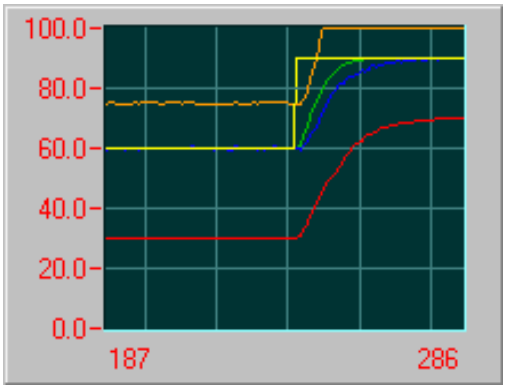

Fig. 14. Performances when changing the reference from 40% to 60%

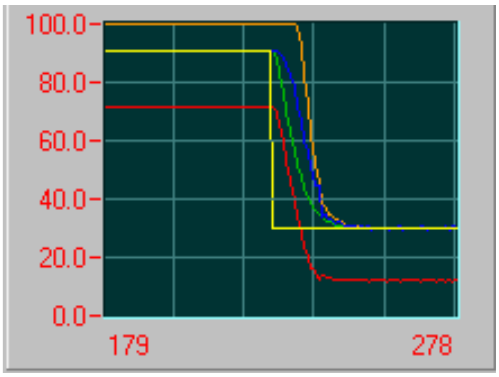

Fig. 15. Performances when changing the reference from 40% to 60%

As it can be observed, the tracking and rejection performances are quite good and the system remains stable.

The effective modification of parameters is done when the filtered process output becomes greater than 20%, 50% and 90%.

For the first 3 changes of reference, the difference between the filtered set point and the process output is quite insignificant, for the last example the difference is larger, but not compromising the quality of the control procedure.

In all tests, one can see that there are no shocks or oscillations in the control evolution by applying this approach neighed in the process output, nor in the command.

Increasing the number of selected inflection points improves the performances if the characteristic substantially changes its form in those points.

6 Conclusion

The method was successfully tested on several nonlinear processes from the same class, using a controller simulator software application implementing the proposed reference model identification and adaptive law. One of the process simulators results has been presented.

The solution that has been proposed is simple and easy to implement. It doesn't imply any kind of problems concerning the sample period value (satisfying tests were performed for a sample period of 10 ms).

The tracking performances are good, the disturbances rejection for this adaptive solution is well performed.

The command and process output values do not present brusque changes while crossing an inflexion point.

With regards to the results obtained in the paper, this adaptive method can be successfully recommended for real-time control structures for this class of nonlinear processes.

Acknowledgement

This work was partially supported by IDEI Program of Romanian Research, Development and Integration National Plan II, Grant no. 1044/2007.

References:

[1] Z. Bubnicki, *Modern Control Theory*, Springer-Verlag, 2005

[2] I. Dumitrache, *Ingineria Reglarii Automate*, Politehnica Press, Bucharest, 2005

[3] L. Foulloy, D. Popescu and G.D. Tanguy, *Modelisation, Identification et Commande des Systemes*, Editura Academiei Romane, Bucharest, 2004

[4] G. Tao, *Adaptive control design and analysis,* Wiley-Interscience, John Wiley&Sons Inc.,2003

[5] I.D. Landau, R. Lozano and M. M'Saad, *Adaptive Control*, Springer Verlag, London, 1997

[6] I.D. Landau, A. Karimi, *Recursive algorithm for identification in closed loop: a unified approach and evaluation*, Automatica, vol. 33, no. 8, 1997, pp. 1499-1523.

[7] Tain-Sou Tsay, *Automatic Gain Control for Unity Feedback Control Systems with Large Parameters Variations*, WSEAS TRANSACTIONS on SYSTEM and CONTROL, Issue 12, Vol.2, December 2007

[8] C. Lupu, D.Popescu, C. Petrescu, A.Ticlea, B. Irimia, C.Dimon, A.Udrea, *Multiple – model Design and Switching Solutions for Nonlinear Processes Control,* EUROSIS Publication, ISBN: 978-90-77381-4-03, Industrial Simulation Conference Proceedings, pag. 71-76, Lyon, France, 2008

[9] Kuan-Yu Chen, Mong-Tao Tsai, and Pi-Cheng Tung, *An Experimental Analysis of an Active Magnetic Bearing System Using PID-Type Fuzzy Controllers with Parameter Adaptive Methods,* 6th WSEAS International Conference on CIRCUITS, SYSTEMS, ELECTRONICS,CONTROL & SIGNAL PROCESSING, Cairo, Egypt, Dec 29-31, 2007

[10] M. A. Zanjani, GH. Shahgholian, S. Eshtehardiha, *Gain Tuning PID and IP Controller with an Adaptive Controller Based on the Genetic Algorithm for Improvement Operation of STATCOM*, 7th WSEAS International Conference on Electric Power Systems, High Voltages, Electric Machines, Venice, Italy, November 21-23, 2007

[11] M. Foltin, J. Murgaš, I. Sekaj, *A new Adaptive PID Control Approach Based on Closed-Loop Response Recognition*, Proceedings of the 7th WSEAS International Conference on Automation & Information, Cavtat, Croatia, June 13-15, 2006 (pp156-160)

[12] J. Richalet, *Practique de la Commande Predictiv,* Editura Herms, Paris, 1993

[13] I.D. Landau, *Identification et commande des systemes*, Ed. Hermes, Paris, 1993