Stabile Algorithms Switching for Multiple Models Control Systems

LUPU CIPRIAN, UDREA ANDREEA, POPESCU DUMITRU, FLUTUR CRISTIAN
Automatics and Computer Science
Politehnica University of Bucharest
313, Splaiul Independentei, Sector 6
ROMANIA
cip@indinf.pub.ro, udrea.andreea@yahoo.com, dpopescu@indinf.pub.ro, http://www.acs.pub.ro

Abstract: - Multiple models structure represents one of the successful solutions for the real-time control of the nonlinear or multi-regime processes. Best algorithm/model selection, switching between control algorithms etc. are the main problems of these structures. The switching issue is a theoretical and practical study subject for a lot of recently researches. The paper proposes some supplementary conditions that assure a correct real-time functioning in terms of stability for multiple models structures. The applicability of the method is proved using an RST control algorithm. In the end, its software implementation is also shown.

Key-Words: - real-time multi-model control systems, stabile switching algorithm

1 Introduction

The multi-model systems represent a relatively new approach on nonlinear control strategies. Since the 90's different studies for the multi-model control strategy have been developed. The Balakrishnan's and Narenda's first papers which proposed several stability and robustness methods using classical switching and tuning algorithms have to be mentioned [1].

Further research in this field determined the extension and improvement of the multi-model control concept. Magill and Lainiotis introduced the model representation through Kalman filters. In order to maintain the stability of minimum phase systems, Middelton improved the switching procedure using an algorithm with hysteresis. Petridis', Kehagias' and Toscano's work focused on nonlinear systems with time variable. Landau and Karimi have important contributions regarding the use of several particular parameter adaptation procedures, namely CLOE (Closed Loop Output Error) [5]. The multi-model control version proposed by Narenda is based on neural networks [9].

Finally, Dubois, Dieulot and Borne apply fuzzy procedures for switching and sliding mode control.

In terms of classical control, multi-model control needs considering specific supplementary aspects:

- Appropriate dimensioning of multiple-model configuration based on process particularities;
- Selection of the best algorithm for a specific state in the process dynamics;
- Control algorithms switching
- Solid consideration about structure stability while commuting between models.

One of the most general architectures for a MM control structure [1] is presented on Fig. 1.

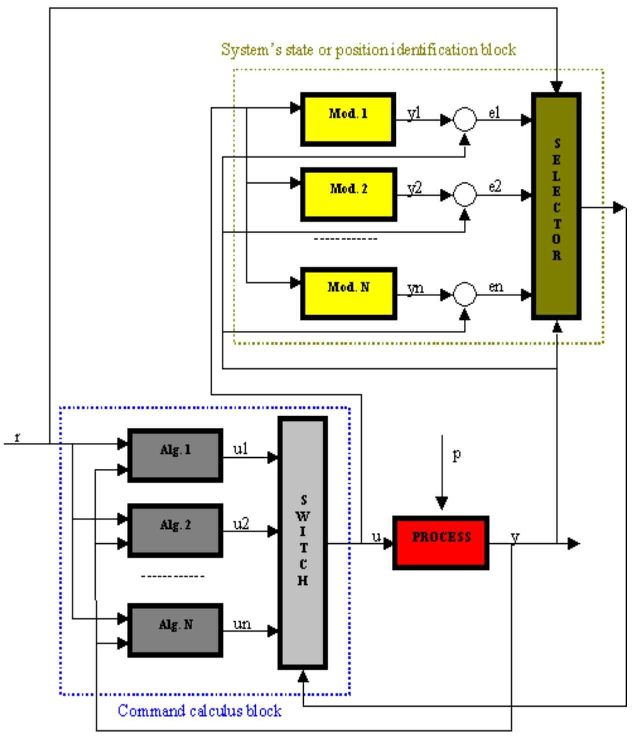


Fig. 1 General multi-model structure

In Fig. 1, the blocks and variables are as follows:

- Process – physical system to be controlled;
- Command calculus – unit that computes the process control law;
- System's state – component that provides information about the model–control algorithm "best" matching the actual system's state;
- Mod. 1, Mod. 2, …, Mod. N - previously identified models of different regimes or operating points;
- Alg. 1, Alg. 2, …, Alg. N – control algorithms designed for the N models mentioned above;
- SWITCH – mixing or switching between the control laws;
- SELECTOR – based on adequate criteria evaluations, provides information about the most appropriate model for the system's current state;
- y and y1, y2, …, yN – output of the process and outputs of the models, respectively;
- u and u1, u2, uN – output generate by Command calculus block;
- r – system's set point or reference trajectory;
- p – disturbances onto the physical process.

Depending on the process particularities and the approach used to solve the switching problem and/or "the best model choice" problem, this structure can be adapted by adding/eliminating some specific blocks.

From the multiple-models control systems viewpoint, two application oriented problems can be highlighted:

- Class of systems with nonlinear characteristic, which can not be controlled by a single algorithm;
- Class of systems with different operating regimes, where different functioning zones do not allow a unique algorithm being used or impose that a very complex method, usually presenting certain problems on implementation, must be developed.

This paper targets the multi-model control structure proposed in [8] by discussing some improved stability considerations.

In chapter 2 the problems occurring while switching between two control algorithms takes place is discussed. The conditions for stable commutation are enounced and two existing methods and there problems are shortly presented. As a solution for stable switching we propose a novel solution.

Chapter 3 is dedicated to the design of the real-time multi-model architecture by using RST controllers for each identified model and the implementation of the switching procedure.

Chapter 4 discuses the stability issues in detail. In chapter 5 an application on an experimental platform and the implemented real-time interface and multi-model structure are presented. The paper end with some conclusions on the proposed solution.

2 The problem of switching between control algorithms

Multi-model structure's design implies that after finding the best algorithm for the current process's functioning point, a switch between the active control algorithm and the best found control algorithm must be realized. Two essential conditions must be verified with respect to this operation:

- no stability issues arise at commutation (no bumps in the applications of the control law are encountered);
- the dynamics of the command law during switching must be fast.

The first condition refers at the fact that shocks determined by the switching operation cause non-efficient and/or dangerous behaviors possibly driving the system in instability.

The second condition interpretation is that slow switching determines boiling down the control algorithm's action zone, which involves alteration of the performances.

In order to meet these conditions different switching methods were proposed.

A solution [2], [5] is based on maintaining in active state all the control algorithms, also called "warm state", only the control law chosen by the switching block $u_i(k)$ being applied on the real process. No additionally considerations are made and, so, it gives the possibility of switching very fast between algorithms.

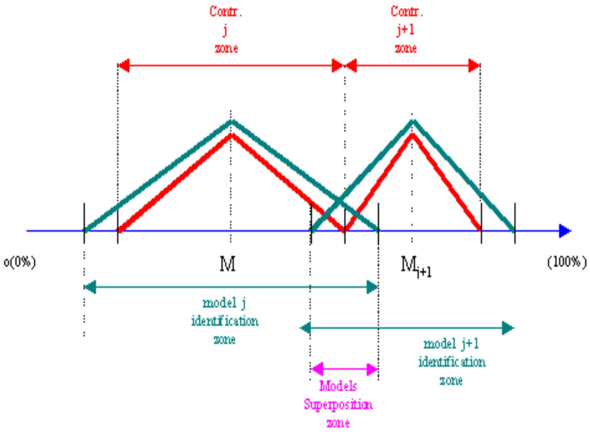


Fig.2 Superposition of identified regions for 2 neighbor-models; corresponding control actions

The only demand is matching the outputs of the control algorithm in the neighborhood switching zones.

The superposition of models identification zones (Fig. 2) accomplishes this aspect but leads to an increased number of models and also instability might arise due to possible big differences between two successive command values given by different controllers.

Another approach [3], [10] proposes mixing two or more algorithms' outputs. The "weighting" of each control law depends on the distance from the current process's operating point and the action zone of each algorithm (Fig. 3). This technique involves control gain problems, generated by the mixing of the algorithms' outputs.

The proposed solution based on previously implemented [8] provides very good results for the class of fast processes with nonlinear characteristics.

The main idea is that, during the current functioning of multiple-models control systems with N model-algorithm pairs, it is supposed that just one single algorithm is hold active (the one designed for the region where the process is functioning) and all the others N-1 algorithms rest inactive.

The output value of the active algorithm is used as fixed desired value for the command computed by all the other N-1 inactive algorithms such that during the transition period the value of the command will stay close to the previous values and after the commutation between controllers only a simple problem of set point tracking remains to be solved. Here the problem is posed in terms of small variations of the command, the process itself varying little at commutation between models. That is why, for stability improvement, a small superposition of the identified models is recommended.

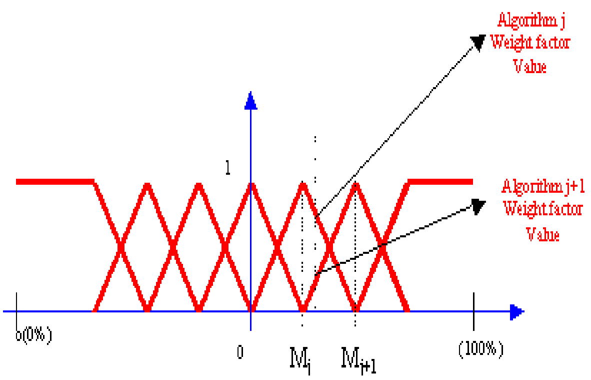


Fig.3. Algorithms weighting functions for a specified operating position

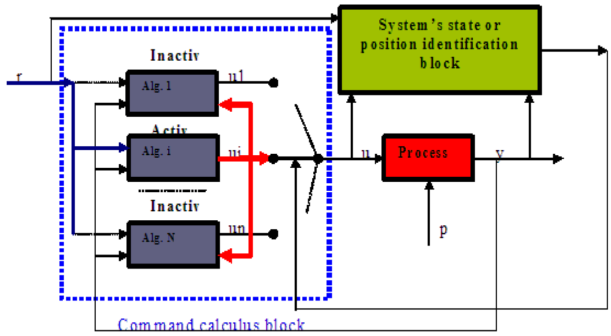


Fig. 3 Proposed solution for algorithms switching

For a stable (bumpless) commutation, the active-inactive shift strategy must be validated, and this it is done in section 3 and 4.

3 Solution for switching between active-inactive control algorithms

On practical exploitation, the process operation starts from an inactive control algorithm, this procedure being used as long as the process did not reach the nominal functioning zone. When commutation is done, it is recommended having a very good matching between the set point and process's output values. This strategy implies that not shocks are sent to the actuators in the system.

In the following, these facts will be illustrated using an RST control algorithm [4].

3.1 Practical considerations about the real-time algorithm implementation

Consider the process's discrete model:

$$A(q^{-1})y(k) = B(q^{-1})u(k) \qquad (1)$$

where $A(q^{-1})$ and $B(q^{-1})$ polynomials are:

$$\begin{aligned} A(q^{-1}) &= 1 + a_1 q^{-1} + \ldots + a_{n_A} q^{-n_A} \\ B(q^{-1}) &= b_0 + b_1 q^{-1} + \ldots + b_{n_B} q^{-n_B} \end{aligned} \qquad (2)$$

with $n_A \leq n_B$.

For this model, an RST control algorithm is used (Fig. 4):

$$S(q^{-1})u(k) + R(q^{-1})y(k) = T(q^{-1})y^*(k) \qquad (3)$$

where:

- u(k) - algorithm output;
- y(k) - process output;
- $y^*(k)$ - trajectory or filtered set point.

The corresponding polynomials are:

$$
\begin{aligned}
S(q^{-1}) &= s_0 + s_1 q^{-1} + \ldots + s_{n_S} q^{-n_S} \\
R(q^{-1}) &= r_0 + r_1 q^{-1} + \ldots + r_{n_R} q^{-n_R} \\
T(q^{-1}) &= t_0 + t_1 q^{-1} + \ldots + t_{n_T} q^{-n_T}
\end{aligned}
\quad (4)
$$

The decision for using this control algorithm instead of a PID control algorithm is due to the fact that has two degrees of freedom.

The closed-loop RST based control structure is given in Fig. 4.

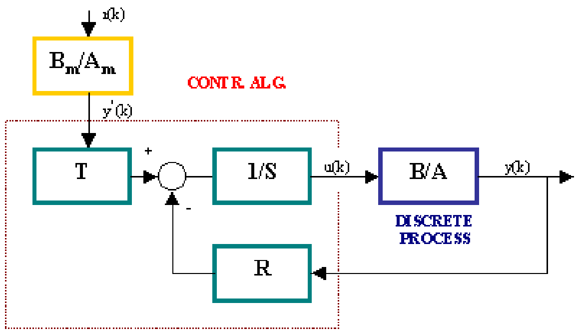


Fig. 4 RST algorithm, two freedom-degrees closed-loop canonical form

The control algorithm in (2) can be rewritten as follows:

$$u(k) = \frac{1}{s_0}\Big[-\sum_{i=1}^{n_S} s_i u(k-i) - \sum_{i=0}^{n_R} r_i y(k-i) + \sum_{i=0}^{n_T} t_i y^*(k-i)\Big] \quad (5)$$

where n_S , n_R , n_T polynomials degrees and also the memory dimension for the software implementation of the algorithm.

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

When necessary, an imposed trajectory can be generated using a trajectory model generator:

$$y^*(k+1) = \frac{B_m(q^{-1})}{A_m(q^{-1})} r(k) \quad (6)$$

with A_m and B_m like:

$$
\begin{aligned}
A_m(q^{-1}) &= 1 + a_{m1} q^{-1} + \ldots + a_{mn_{Am}} q^{-n_{Am}} \\
B_m(q^{-1}) &= b_{m0} + b_{m1} q^{-1} + \ldots + b_{mn_{Bm}} q^{-n_{Bm}}
\end{aligned}
\quad (7)
$$

An iteration of the monitoring program implies following steps:

- data acquisition from the process;
- trajectory computation;
- control law computation;
- sending the command to the actuators;
- graphical display of the process evolution;
- actualization of the algorithm's memory for the new iteration.

For example, the control law computation, when $n_R = n_S = n_T = 2$ and without trajectory generator ($y^*(k)=r(k)$), is the following:

$$u(k) = \frac{1}{s_0}(-s_1 u(k-1) - r_0 y(k) - r_1 y(k-1) + t_0 y^*(k) + t_1 y^*(k-1)) \quad (8)$$

and (9) gives the algorithm's memory actualization for the next iteration:

$$u(k-1) = u(k); y(k-1) = y(k); y^*(k-1) = y^*(k) \quad (9)$$

3.2 Inactive to active transfer

In the context of switching, since the inactive control algorithm's output is value of the active algorithm's command and the process's output depend on command, the set point remains the only "free" variable in the control algorithm's computation.

Therefore, the proposed solution consists in the modification of the set point value accordingly to the active control algorithm output, the selected inactive algorithm to become active and process's output. From (3), it results the expression for the recomputed set point's value:

$$y^*(k) = \frac{1}{t_0}\Big[\sum_{i=0}^{n_S} s_i u(k-i) + \sum_{i=0}^{n_R} r_i y(k-i) - \sum_{i=1}^{n_T} t_i y^*(k-i)\Big] \quad (10)$$

This recalculation of set point value will influence the value of the new control algorithm output just during the switching period (depending on *nT*).

When the set point (trajectory) generator (6) is used, keeping all the data in correct chronology must be with respect to the following relation:

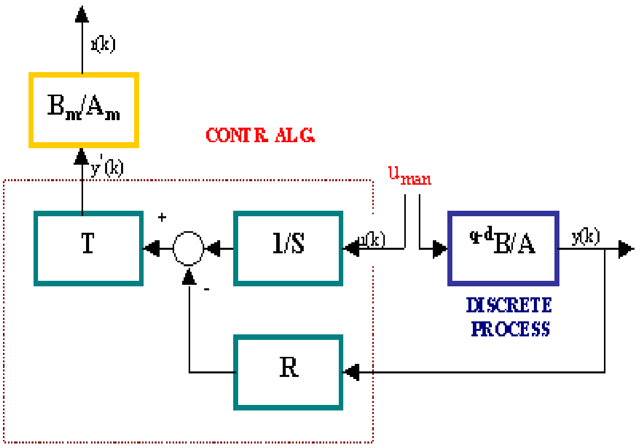


Fig. 5 Computation of the set point value for active algorithm given command- u_{man}

$$r(k) = \frac{A_m(q^{-1})}{B_m(q^{-1})} y^*(k) \qquad (11)$$

The switching system functioning scheme is presented on Fig. 5.

This solution proposes the computation of that set point value that determines, accordingly to the algorithm's history and process's output, a control value equal to the active algorithm command value in the switching moment.

Using this solution eliminates possible gaps in the control algorithm's memory that could determine stability problems.

An eventually mismatching between the set point and process's output is considered as a simple change of set point's value. Moreover, this solution can be successfully used in cases of command limitation.

The only inconvenient of this approach is represented by the necessity of big computation power when approaching high order systems, which is not, however, a problem nowadays.

4 Stability analysis

Multiple model stability for the proposed solution imposes satisfying the next important points:

- Stability of each control algorithm in active state – assured by using the classical poles placement procedure when designing the RST control law;
- Stability at switching moment – active-inactive controller switching [14];
- Stability of each control algorithm during the inactive states – discussed in the next paragraphs;

The proposed solution, in fact, consists in an inverse model of the command. This is calculated inside the inactive control algorithm before the switching moment. That is why a discussion on the stability of inverted model of the RST controllers is made.

Rewriting (2), the RST control law is:

$$u(k) = \frac{T(q^{-1})}{S(q^{-1})} y^*(k) - \frac{R(q^{-1})}{S(q^{-1})} y(k) \qquad (12)$$

and from (2) and (10) results that in inactive regime:

$$y^*(k) = \frac{S(q^{-1})}{T(q^{-1})} u(k) - \frac{R(q^{-1})}{T(q^{-1})} y(k) \qquad (13)$$

where $u(k) = u_{man}(k)$ (command value given by the active algorithm).

The complete algorithm structure includes the set point generator presented in (11) and used for the inverse data flow "transfer" situation.

From (13) and (11) the stability of each control algorithms in inactive state imposes stability of $\frac{S(q^{-1})}{T(q^{-1})}$, $\frac{R(q^{-1})}{T(q^{-1})}$ and $\frac{A_m(q^{-1})}{B_m(q^{-1})}$ transfer functions.

From [6], for pole placement algorithm design procedure, the $T(q^{-1})$ polynomial is:

$$T(q^{-1}) = \frac{P(q^{-1})}{B(1)} \qquad (14)$$

where $P(q^{-1})$ is the close loop characteristic polynomial.

This is usually designed as a discrete equivalent of stabile continuous second order system, which implies that:

$$\frac{R(q^{-1})}{T(q^{-1})} = \frac{R(q^{-1})B(1)}{P(q^{-1})} \qquad (15)$$

and

$$\frac{S(q^{-1})}{T(q^{-1})} = \frac{S(q^{-1})B(1)}{P(q^{-1})} \qquad (16)$$

Relation (15) represents one of the stability criteria for the RST structure [6] and, therefore, $\frac{R(q^{-1})}{T(q^{-1})}$ is stabile.

On the other hand, from the RST design procedure, $1/S(q^{-1})$ contains an integrator; this leads to a derivative comportment in the inverted form, assuring an increased degree of stability for the transfer function (16) and a slower dynamics.

The set point generator presented in (11) is designed as a discrete equivalent of a stabile continuous second order system.

Usually, for this multiple model control structure, this generator is designed to have a slow dynamic – that determines a stabile transfer between controllers. A compromise must be made because a very slow dynamic in "direct" data flow transfer implied very fast dynamics for inverse data flow "transfer" situation (11). Fast dynamics leads to important oscillations in the response.

For an adequate solution for the design of the discrete equivalent of a stabile continuous second order system:

$$H(s) = \frac{k\omega_0^2}{s^2 + 2\xi\omega_0 s + \omega_0^2} \qquad (17)$$

the values of ω_0 and ζ must be inside the intervals:

$$\omega_0 \in \left[0.25/T_e, 1.5/T_e\right],$$
$$\zeta \in [0.7, 1.0]\,. \qquad (18)$$

In proposed multiple models structure, the set point filter plays another important role: it does not allow important changes in set point evolution, eventually a pseudo continuous track. This protects the actuators of shocks.

A third role of the used filter is that it forces the system not to "jump" over one or more neighboring model – controller.

5 Experimental Results

We have evaluated the stability of the proposed multi-model control structure using a process interface and control software application, developed using National Instruments's LabWindows/CVI and a laboratory installation with a data acquisition device.

In Fig. 6 and Fig. 7 are presented the interface and the physical vertical positioning control system consisting in a ball located in a tube with small longitudinal opening.

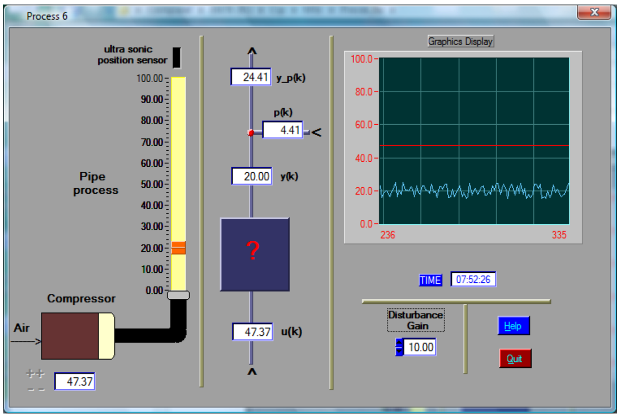


Fig. 6 Ball position process interface (the dynamics of the ball is shown in the left part of the window, while the numerical values are graphically displayed)

The goal is to control the ball's position by using our solution. The quality of the designed control solution consists in a smooth transition between successive algorithms.

The nonlinear evolution of the ball's position - Y(%) and the actuator command - U (%) is presented on Fig. 8.

The characteristic presented in figure 8 was obtained by meaning the process outputs while a set of tests consisting in increasing/decreasing the command values on the operating domain.

We have considered three operating points P_1, P_2, and P3 on the process nonlinear diagram (Fig. 8). Three different models are identified: M_1 (0-30%), M_2 (30-70%) and M_3 (70-100%). These will be the zones for corresponding algorithms.

Fig. 7 Laboratory experimental nonlinear installation for positioning a ball in a vertical tube with a longitudinal opening

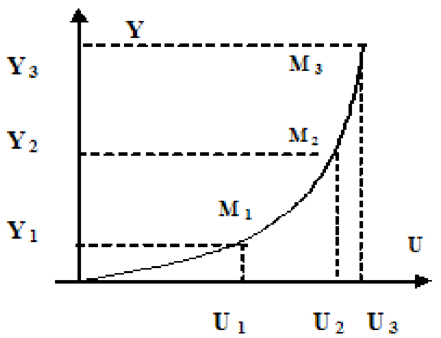


Fig. 8 Nonlinear diagram of the process and the identified different functioning regions providing models M1,M2,M3

Accordingly to the models-algorithms matching zones (Fig. 2), we have identified the models M_1, M_2 and M_3, as being appropriated to the following intervals (0-40%), (20-80%), (60-100%). For a sampling period T_e=0.2 sec, the least-squares identification method [15] from Adaptech/WinPIM platform identifies the next models:

$$M_1 = \frac{0.35620 - 0.05973q^{-1}}{1 - 0.454010q^{-1} - 0.09607q^{-2}}$$

$$M_2 = \frac{1.23779 - 0.33982q^{-1}}{1 - 0.98066q^{-1} - 0.17887q^{-2}}$$

$$M_3 = \frac{2.309530 - 0.089590q^{-1}}{1 - 0.827430q^{-1} - 0.006590q^{-2}}$$

In this case, we have computed three corresponding RST algorithms using a pole placement procedure [12] from Adaptech/WinREG platform.

The same nominal performances were imposed for all three systems, through a second order system, defined by the dynamics ω_0 = 3.0, ξ = 2.5 (tracking performances) and ω_0 = 7.5, ξ = 0.8 (disturbance rejection performances) respectively, keeping the same sampling period as for identification (T_e=0.2 sec). It can be observed that stability conditions are respected.

All of these algorithms control the process just in their corresponding zones:

$$R_1(q^{-1}) = 1.670380 - 0.407140q^{-1} - 0.208017q^{-2}$$
$$S_1(q^{-1}) = 1.000000 - 1.129331q^{-1} + 0.129331q^{-2}$$
$$T_1(q^{-1}) = 3.373023 - 3.333734q^{-1} + 1.015934q^{-2}$$

$$R_2(q^{-1}) = 0.434167 \;\; 0.153665q^{-1} - 0.239444q^{-2}$$

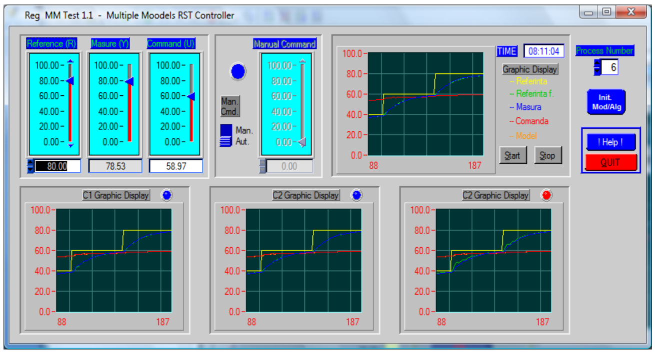


Fig. 9 Multi-model controller real-time software application

$$S_2(q^{-1}) = 1.000000 - 0.545100q^{-1} - 0.454900q^{-2}$$
$$T_2(q^{-1}) = 1.113623 - 1.100651q^{-1} + 0.335417q^{-2}$$

$$R_3(q^{-1}) = 0.231527 - 0.160386q^{-1} - 8.790E-04q^{-2}$$
$$S_3(q^{-1}) = 1.000000 - 0.988050q^{-1} - 0.011950q^{-2}$$
$$T_3(q^{-1}) = 0.416820 - 0.533847q^{-1} + 0.187289q^{-2}$$

To verify the proposed switching algorithm, a multi-model controller real-time software application was designed and implemented. It can be connected directly with the process. The user interface is presented in Fig. 8.

In the upper part of Fig. 9, there are respectively: the set point, the output and control values, active-inactive general switch, general active command and graphical system evolution display.

In the lower part of Fig. 9, one can see three graphical evolution displays corresponding to the three controllers (C_i, S_i, T_i, i=1...3). The colors are as follows:

- yellow – set point value;
- red – command value;
- blue – process output value
- green – filtered set point value.

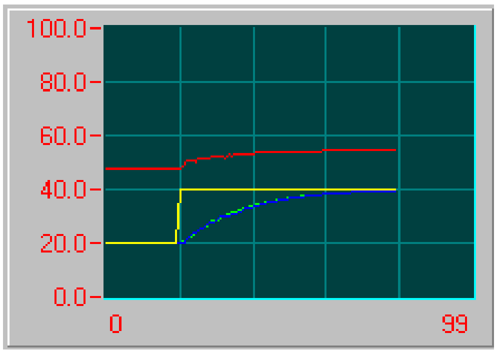
The values previously determined for the RST controllers are uploaded using this interface directly to the real-time application.

During runtime, the active algorithm is marked with by a red led in the upper corner of its specific sub- window.

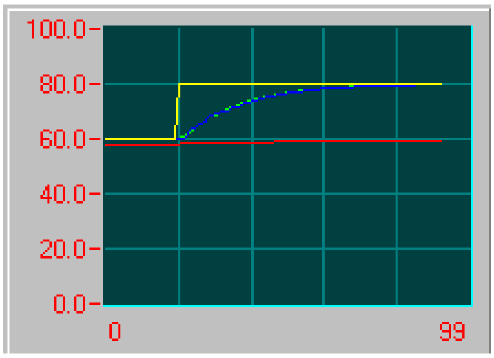
Using the obtained values and the application, tests were effectuated to verify the switching between the three algorithms.

The switching procedure is determinate by the change of the set point value.

In order to force the commutation between two consecutive models some tests were done:



a) switching between first and second algorithm



b) switching between second and third algorithm

Fig. 9. Switching between consecutive controllers tests

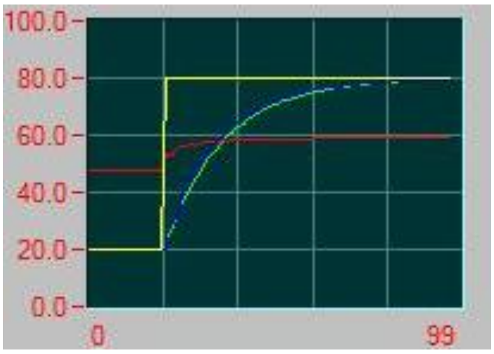
a) the set point is changed from 20% (where algorithm 1 is active) to 40% (where algorithm 2 is active). The effective switching operation is done when the filtered set point (and process output) becomes greater than 30%. Fig. 9(a) presents the evolutions of the command (red) and process output (blue) before, during and after the switching took place.

b) the set point is changed from 60% (where algorithm 2 is active) to 80% (where algorithm 3 is active). The effective switching operation takes place when the filtered set point (and process output) becomes greater then 70%. Fig. 9(b) presents the evolutions.

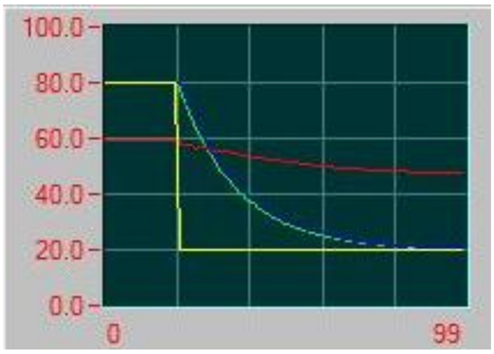
In all tests, one can see that there are no shocks in the command value and the reference tracking is almost perfect.

There are very small oscillations in the control evolution by applying this method.

In figure 10 there are presented two test in which the set point is changed in order to force the control algorithm to commute between controllers that are not consecutive:



a) switching between first and second algorithm



b) switching between third and first algorithm

Fig. 10. Switching between not consecutive controllers tests

a) the set point is changed from 20% (where algorithm 1 is active) to 80% (where algorithm 3 is active). The switching operation is done in two steps: when the filtered set point (and process output) becomes greater than 30% the second algorithm became active until the filtered set point reaches the value 70% and the third algorithm takes the lead. Fig. 10(a) presents the evolutions of the command and process output during all changes and after stabilization.

b) the set point is changed from 80% (where algorithm 3 is active) to 20% (where algorithm 1 is active). The switching operation between the third and second controller takes place when the filtered set point becomes smaller then 70% and another switch occurs when the filtered reference becomes smaller than 30% and the third algorithm becomes active. Fig. 10(b) presents the evolutions.

As in the tests shown in figure 9, in these cases we can see the same smooth transition during switching in terms of smooth command value and good reference tracking.

This comportment is assured by using the filter on the set point. Because the reference modifies slower, all the controllers are traversed until reaching the last one.

Without this smooth transition there is a risk of increasing the system instability if the set point is varied before a controller leads the process in stable regime.

The tests presented in figure 10 were done by applying a 10% disturbance on the process.

Comparing the results in figure 10 with the ones in figure 9, one can observe that, even if we obtained a good reference tracking in both cases, when the change in set point is large, the tracking error becomes larger.

Increasing the number of models-algorithms to 4 or 5 could eliminate the small oscillations and the tracking error can be reduced.

The multi model algorithm was tested for a higher number of models-controllers pairs, but we have considered that the 3 models-controllers solution is sufficient for the presented case.

A growth in the number of model - algorithm pairs for a multi-model control algorithm can be done only if the real-time architecture presents the necessary hardware and software resources.

An improvement effect can be obtained by increasing the precision of the operations from float to double. In this case, the available memory space must be consulted and the duration of all operations done in a single sample time must be calculated.

6 Conclusions

The paper provides a novel stable switching solution for the multi model algorithm. The method was successfully tested on a process with a software interface attached and a control application.

The experimental platform consists in a process that exhibits a nonlinear characteristic and the tests were made in the presence of disturbances.

A three multi-model/controller real-time numerical control software application was used for this particular case.

The control software allows that a higher number of algorithms may be used if necessarily.

The presented tests showed good performances of the control architecture with respect to the stability issues and also from the point of view of tracking and command variations.

With regards to the results obtained in the paper, the switching method can be successfully recommended in multi-model real-time control structures for fast processes.

The provided switching methodology satisfies the stability issues that are common for multi-model control in a new and improved manner.

The proposed method can be used in a series of industrial applications [11], [15].

A further study direction will be determining the optimum set point filter's parameters such that the commutation will be faster but the stability holds too.

Also, the dependence between the number of model-controllers pairs, respectively the sampling period used for real-time control and the filter form will be analyzed.

Acknowledgement

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

References:

[1] Balakrishnan., J., *Control System Design Using Multiple Models, Switching and Tuning*, Ph. D. Dissertation, University of Yale, USA, 1996.

[2] Dumitrache, I., *Ingineria Reglarii Automate*, Politehnica Press, Bucuresti, 2005, ISBN 973-8449-72-3.

[3] Dussud, M., S. Galichet and L. Foulloy, *Fuzzy supervision for continuous casting mold level control*, IFAC, 2000

[4] Foulloy L., Popescu D., Tanguy G.D., *Modelisation, Identification et Commande des Systemes*, Editura Academiei Romane, Bucuresti, 2004, ISBN 973-27-1086-1

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

[6] Landau, I. D., R. Lozano and M. M'Saad, *Adaptive Control*, Springer Verlag, London, 1997, ISBN 3-540-76187-X.

[7] Lupu, C., D. Popescu and S. Manza, "*Advanced Techniques for Non-Linear Control Systems*", 16^{th} IMACS World Congress 2000, Lausanne, August 21-25, 2000.

[8] Lupu C., D. Popescu, B. Ciubotaru, C. Petrescu, G. Florea , "*Switching Solution for Multiple Models Control Systems*", MED06, paper WLA2-2, 28-30 June, Ancona, Italy, 2006

[9] Narendra, K. S. and J. Balakrishnan, Adaptive Control using multiple models, *IEEE Transactions on Automatic Control*, vol. 42, no. 2, February, page. 171 – 187, 1997

[10] Pages, O., P. Mouille and B. Caron, *Multi-model control by applying a symbolic fuzzy switches*, IFAC, 2000

[11] Popa G. N., Dinis C. M., Iagar A., *Mathematical Modeling and Simulation in Matlab/Simulink of Processes from Iron Ore Sintering Plants*, WSEAS Transactions on Systems, Issue 1, Volume 8, January 2009, ISSN: 1109-2777

[12] Popescu D., Stefanoiu D., Lupu C., Petrescu C., Ciubotaru B., Dimon C., *Industrial Automatics (Automatica Industriala)* – Editura Agir, Bucuresti, 2006, ISBN 978-973-720-093-8

[13] Sadati N., Mazinan A. H., *Multiple Modeling and Fuzzy Predictive Control of A Tubular Heat Exchanger System* - WSEAS Transactions on Systems and Control, Issue 4, Volume 3, April 2008, ISSN: 1991-8763

[14] Sangtungtong, W., and Sujitjorn, S, *Adaptive Sliding-Mode Load-Torque Observer: Its Stability Aspects* - WSEAS Transactions on Systems, Issue 7, Volume 7, July 2008, ISSN: 1109-2777

[15] Stefanoiu D, Culita J., Stoica P., *Fundaments of systems modeling and identification*, (*Fundamentele modelarii si identificarii sistemelor)*, Editura Printech, Bucuresti, 2005, ISBN 973-718-368-1