SWITCHING IN MULTIPLE MODELS CONTROL SYSTEMS

Ciprian Lupu, Catalin Petrescu, Dumitru Popescu

Politehnica University of Bucharest, Department of Automatics and Computers Science Bucharest, 313, Splaiul Independentei, cip@router.indinf.pub.ro

Abstract: The systems with multiple models or multicontroller structure represent one of success solutions for real time control of nonlinear or multi regime process. Utilization of these structures imposes solving of some specific problems like best algorithm selection or switching of control algorithm. The paper proposes a method for switching of the algorithms of multimodel structure based on principles of manual to automate bumpless transfer. The method is presented for a real time structure with RST control algorithm.

Keywords: control systems, switching algorithm, manual - automatic bumpless transfer, real time systems

1. INTRODUCTION

The essential condition for the real time control system function is conservation of performances in condition of non-linearity, structural disturbances or process incertitude. A valuable way to solve these problems is the multimodels or multicontroller structure command. Historically, the first papers contain the notion of "multimodel" structure or system appeared in '90 years. From first authors must be remembered Balakrisnan and Narenda who present in their papers problems of stability, robust, switching and designing for this structures (Balakrishnan, 1996), (Narenda and Balakrishnan, 1997).

In time, researcher's accumulations from this field bring the extension and refinance of multimodel control concept. So, Landau and Karimi (1997) have frequent contributions about parametric adaptation procedures - Close Loop Output Error, Magill and Lainiotis are extended model representation possibilities starting from Kalman filters. Another important step is multimodel command version subsequent proposed by Nareda based on Neural Network. Last but not least, there are distinguished command switching procedures proposed by Dubois, Dieulant and Borne and abided by fuzzy systems.

Relative to classical control loops, multimodel systems need solving of some supplementary specific problems:

- Dimension of multimodel configuration;
- Selection of the best algorithm;
- Command switching.

From multimodels systems application viewpoint there are two important classes:

- Class of systems with nonlinear characteristic – which can not be controlled by a single algorithm;
- Class of systems with different functioning regime– where different function regime doesn't allow used of a unique algorithm or imposes usage of very complex one whit special problems on implementation.

On dependence of solving these problems and process particularity there are proposed a lot of structures for multimodel system architecture. One of the most general structures is presented in Fig. 1.

Legend for figure 1:

- PROCESS –physical process that must be controlled;
- Command calculus block – is the component that must calculate the command for the process;

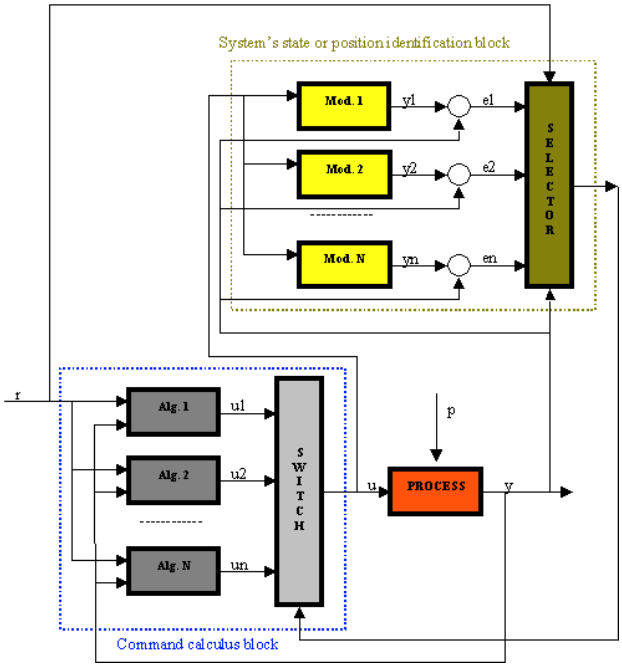

Fig. 1. General scheme for multimodel structure

- System's state or position identification block – component that provide about best model – algorithm for actual system's state or position;
- Mod. 1, Mod. 2, Mod. N - modeless of different regimes or functioning points previously identified;
- Alg. 1,Alg. 2, Alg. N – control algorithms designed for each N different functioning points;
- SWITCH – switching or mixing block for control algorithms output;
- SELECTOR –block for identify system's state using adequate criterion and algorithms that compute information provided by references, models output;
- y, y1, y2, yN – outputs of the process and models;
- u – output generate by Command calculus block;
- u1, u2, uN – outputs of the N control algorithms;
- r – represent the system's set point or trajectory;
- p – disturbances of physical process.

How we note before, dependent on process particularities and way to solve "switching algorithms" and "best model chose" problems, the scheme can be particularized by adding or eliminating of some specific blocks. On next capitols the paper will be focused on algorithms switching problem.

2. CONTROL ALGORITHMS SWITCHING

Corresponding to multimodel structure's function logic, after finding the best algorithm for the current process's functioning point, the next step consists on switching of control algorithm. This operation must respect two essential aspects:

- To be realized so that to not determine any shocks of the command;
- To be very faster.

Shocks determined by switching operation cause uneconomical and dangerous behaviors and slow speed switching cause "moving" of control algorithms action zone that give on happiest cases just system's performances alteration.

These are the main problems to solve on designing of algorithms switching block. On first hand, structurally point of view, this block must contains all algorithms implementation or, at least algorithm's coefficients. The switching operation is done based on information provided by system's state or position identification block. This information consists on a signal to start switching operation and the number of the algorithm that will become active.

2.1. Classic solutions

Present solutions solve more or less this problem and are based on maintaining in functional state all control algorithms. This state is also named "warm state" and suppose that every algorithms receive information about process output y(k) and set point value (eventually filtered $r_f(k)$) and function as an active one with difference that corresponding output $u_i(k)$ is not applied on real process. This solution not imposes supplementary function logic for system's architecture and for these reasons gives the possibilities to switch very fast the algorithms. The main disadvantage is that on multimodel's structure designing step there are supplementary precautions.

These imposes control algorithms command coincidence in neighborhood zones where is made the switching. To realize this aspect is needed a superposition of models identification zone. Fig. 2 present graphical this aspect.

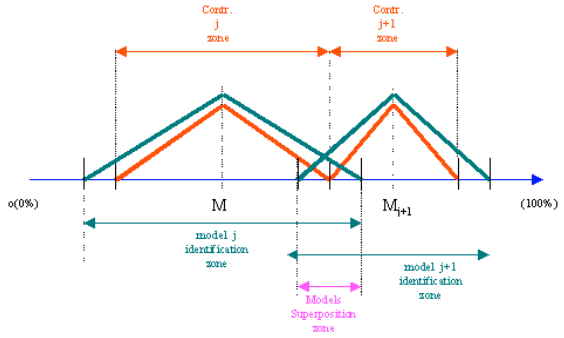

Fig. 2. Superposition of identification zones for two neighborhood models and disposing action of correspondent algorithms

As a result of this superposition, multimodel structure will have an increase number of the models or very complex models that impose complex algorithms.

Another approaches, (Dussud, et al., 2000), (Pages, et al., 2000) propose the mixing of two or more algorithm's outputs. The "weight" of each command depends on distance from current process's function point and the zone of action of each algorithm. Base on this, the pass from an algorithm to another is done using whiting functions with a continuous evolution in 0 – 1 intervals. This approach can be easy implemented using fuzzy systems. An example of this is presented in Fig. 3:

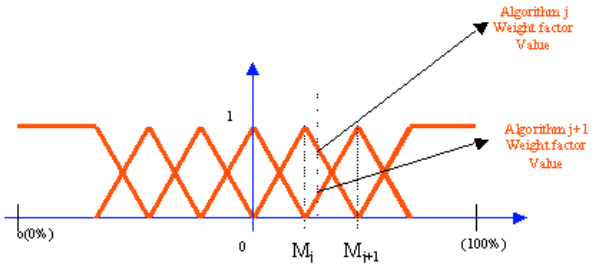

Fig. 3. Algorithms weight functions for a specified functioning position

This solution supposes the solving of command gain problems, determinate by the superposition of algorithm output in process input.

2.2. Proposed solution

In this approach is presented a solution that provides very good results on process with nonlinear characteristic. This proposes maintaining of all inactive algorithms on manual command and commuting that in automate regime in switching moment. To solve manual –automate transfer problems it can be used eventually; the method exposed in capitol 3. The value of active algorithm output represents manual command for all other algorithms.

The system can be implemented in two variants – first - with all inactive algorithms hold on manual regime, or – second - just a single functioning algorithm (the active) and activation of the "new" by calculus of current corresponding manual regime and switching on automate regime. Both variants have advantages and disadvantages. For variant choosing is necessary to know hardware system's performances. At first sight the first variant look to be more reasonable but this fact cannot be generalized.

In all situations, algorithm's manual value is construct considering that active algorithm's output values represent manual commands for "new" selected algorithm.

3. MANUAL–AUTOMATE BUMPLESS TRANSFER

Important problems in algorithm implementation practice are given by manual to automate, automate to manual regime commutations and access, respectively turning out in/from command saturation states. These problems exist of course, in analogical systems and have specific procedures that in most cases are no applicable on numeric systems.

Starting of a process is made on "manual" regime, this command being used as long as the process is not on nominal parameters zone. In this zone is recommended a very good superposing of set point and process's output values. In this state is made manual to automate transfer. This strategy releases the system by shock sends to actuators. Sometimes these shocks can change process's functioning zone.

We will illustrate these facts on RST control algorithm. On beginning few practice considerations.

3.1. Practical consideration on real time algorithm implementation

On consider the next process's discrete model:

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

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

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

with $n_A \le n_B$. For this model is use a RST algorithm:

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

In this, we note u(k) - algorithm output, y(k) - process output, $y^*(k)$ - trajectory or filtered set point and correspondent polynoms are:

$$S(q^{-1}) = s_0 + s_1 q^{-1} + \ldots + s_{n_S} q^{-n_S} \tag{4}$$

$$R(q^{-1}) = r_0 + r_1 q^{-1} + \ldots + r_{n_R} q^{-n_R} \tag{5}$$

$$T(q^{-1}) = t_0 + t_1 q^{-1} + \ldots + t_{n_T} q^{-n_T} \tag{6}$$

Close loop control representation is give in Fig. 4:

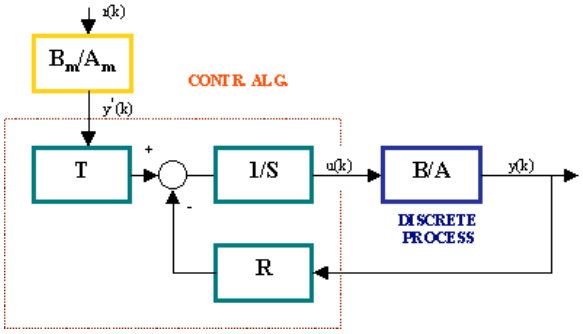

Fig. 4. Two-liberty degree controller's canonical form

The control algorithm presented in (3) can be redefined and writing u(k) command (in) function of his past values, imposed set point and process's output:

$$u(k)=\frac{1}{s_0}[-\sum_{i=1}^{n_S}s_i u(k-i)- \\ -\sum_{i=0}^{n_R}r_i y(k-i)+\sum_{i=0}^{n_T}t_i y^*(k-i)] \quad (7)$$

n_S , n_R , n_T parameters express corresponding polynoms's degrees and implementation algorithm's memory. For example, for $R(q^{-1})$ polynomial, if $n_R=2$ it should be reserved for y(k) – process's output - three-memory location: y(k), y(k-1), y(k-2). For the other variables, respective u(k) and $y^*(k)$ the same condition must be respect.

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

$$y^*(k+1)=\frac{B_m(q^{-1})}{A_m(q^{-1})}r(k) \quad (8)$$

with A_m and B_m like:

$$A_m(q^{-1})= 1+a_{m1}q^{-1}+\ldots+a_{mn_{Am}}q^{-n_{Am}} \quad (9)$$
$$B_m(q^{-1})=b_{m0}+b_{m1}q^{-1}+\ldots+b_{mn_{Bm}}q^{-n_{Bm}}$$

In practical implementation, we are interested about control algorithm and eventually trajectory's model generator because the real process is represented by acquired measure and sent command values, received/send with corresponding sampling rate. The algorithm for a single iteration contained in infinite program's loop is:

- Process's dates acquisition;
- Trajectory calculus (if it is necessary);
- Command calculus;
- Sending to the command into the process;
- Graphical display;
- Algorithm's memory actualization for a new iteration.

Particularizing for a control algorithm without trajectory generator ($y^*(k)=r(k)$), where $n_R = n_S = n_T = 2$, command calculus is made using the next expression:

$$u(k)=\frac{1}{s_0}(-s_1u(k-1)-r_0y(k)- \\ -r_1y(k-1)+t_0y^*(k)+t_1y^*(k-1)) \quad (10)$$

and relation (11) gives algorithm's memory actualization for next iteration:

$$u(k-1)=u(k);\ y(k-1)=y(k);\ y^*(k-1)=y^*(k); \quad (11)$$

3.2. Manual/automate transfer

In real functioning M→A transfer is preceded by "driving" of the process in nominal action zone. To avoid command's "bumps" on transfer moment two conditions must be respected:

- Process's output must be perfect superpose on set point value;
- In accordance with algorithm complexity, a number of sampling period (equal to maximum number of control algorithm's reserved locations) must be waiting.

Neglect of this conditions lead to "bumps" in transfer because the control algorithm's output value is calculate according to actual and past values of command, process and set point values.

In the same time by reason of process's dynamics perfect "superposition" between process's output and set point value, some time is very difficult to be obtained and need very long time. This procedure becomes impossible in presence of important disturbance.

In this context, because algorithm's output is manual command – set by operator, process's output – depend on command, "free" in algorithm relation rest just the set point, that can be modified so that to be according with control algorithm.

Accordingly solution consists in modification of set point value, in accordance with existent control algorithm, manual command and process's output.

Algorithm's memory actualization is normal like in automate regime. For practically implementation it is necessary a supplementary memory location for the set point value (unmodified by algorithm) that will be necessary in automate regime. From (8) result the express for set point's value:

$$y^*(k)=\frac{1}{t_0}[\sum_{i=0}^{n_S}s_i u(k-i)+ \\ +\sum_{i=0}^{n_R}r_i y(k-i)-\sum_{i=1}^{n_T}t_i y^*(k-i)] \quad (13)$$

When set point (trajectory) generator (9) exist, to keep all the data in correct chronology, we can use the same method. Practically this transfer function is:

$$r(k)=\frac{A_m(q^{-1})}{B_m(q^{-1})}y^*(k) \quad (14)$$

System's functioning scheme is presented in Fig. 5:

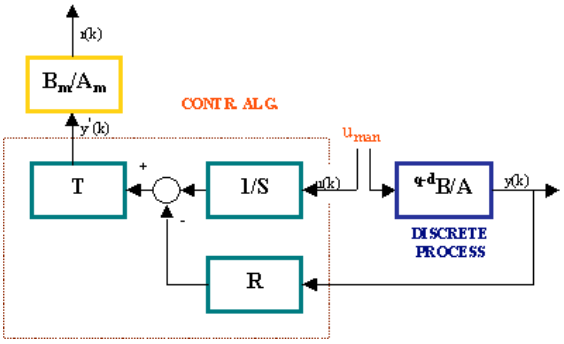

Fig. 5. Calculus of set point value for imposed manual command

Concluding, this solution propose calculus of that set point value that cause, according to algorithm's history and process's output a command equal to manual command applied by operator. On the moment of M→A transfer any "fracture" is present on control algorithm's memory that determine a bumpless activity. An eventually missing of superposing of set point and process's output is considerate as a simple change of set point's.

The inconvenience of this solution consists in necessary of powerful implement hardware, but this problem is not very important today because actual equipment use powerful processors.

This solution can be successfully used in cases of command limitation.

4. EXPERIMENTAL RESULTS

We have evaluated the achieved performances of the multimodel control structure using a process simulator software application, developed in National Instruments's LabWindows/CVI as in Fig. 6, witch represent a position system functioning in medium with variable viscosity. The main goal is to control in closed loop the position of the piston.

The nonlinear relation between the position Y (in %) and actuator command U (%) is presented in Fig. 7.

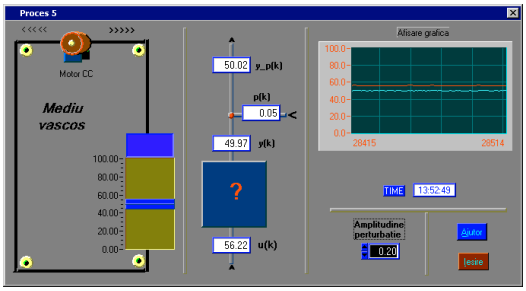

Fig. 6. Process simulator software application

On consider three operating points P_1, P_2, and P3 on plant's nonlinear diagram (Fig. 7). We have identified three different models: M_1 for 0 to 30%, M_2 for 30 to 70% and M_3 for 70 to 100%. These will be the zones for corresponding algorithms.

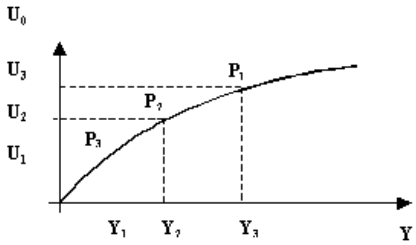

Fig. 7. Nonlinear diagram of the process

According to models/algorithms superposing zone (see Fig. 2) we have identify M_1 on 0 to 40%, M_2 on 20 to 80% and M_3 on 60 to 100% intervals. Choosing sampling time T_e=0.2 s and Least Square identification method from Adaptech/WinPIM software was obtained and validated 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 correspondent R-S-T algorithms using a pole placement procedure from Adaptech/WinREG software. The same nominal performances are given for all systems, by a second order standard dynamic system described by $\omega_0 = 3.0$, $\xi = 2.5$ (tracking performances) $\omega_0 = 7.5$, $\xi = 0.8$ (disturbance rejection performances) respectively, sampling time Te = 0.1s.

All of these algorithms control the process in correspondent zone but can't do it in others.

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

$$R_2(q^{-1}) = 0.434167 \;\; 0.153665q^{-1} \;\; - 0.239444q^{-2}$$
$$S_2(q^{-1}) = 1.000000 \; - 0.545100q^{-1} \; - 0.454900q^{-2}$$
$$T_2(q^{-1}) = 1.113623 \; - 1.100651q^{-1} \; + \; 0.335417q^{-2}$$
$$R_3(q^{-1}) = 0.231527 \; - 0.160386q^{-1} \; - 8.790E-04q^{-2}$$
$$S_3(q^{-1}) = 1.000000 \; - 0.988050q^{-1} \; - 0.011950q^{-2}$$
$$T_3(q^{-1}) = 0.416820 \; - 0.533847q^{-1} \; + \; 0.187289q^{-2}$$

To verify the proposed switching algorithm it was designed and implemented a multimodel controller real time software application. This can be connected with process simulator. The user interface is presented in Fig. 8.

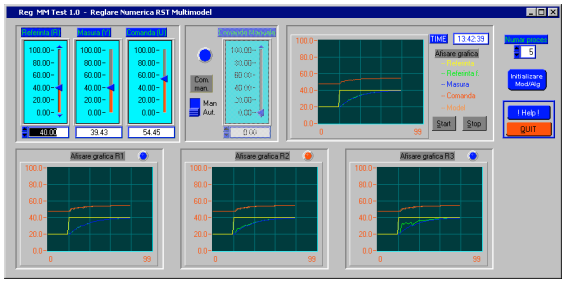

Fig. 8. Multimodel controller real time software application

On top there are the set point, output and command values, manual-automate general switch, general manual command and graphical system evolution display. On bottom side there are three graphical evolution displays corresponding to the tree controllers. The color legend for graphical evolution is: yellow – set point value, red – command value, blue – process – value and green – filtered set point value.

With this application was effectuated few test to verify the switching between two algorithms. The switching procedure is determinate by changing of set point value. The tests are:

a) changing from 20% where algorithm 1 is active to 40% where algorithm 2 is active. The effective switching operation is made when the filtered set point (and process output) becomes greater then 30%. Figure 9 a) present the evaluations.
b) changing from 60% where algorithm 2 is active to 80% where algorithm 3 is active. The effective switching operation is made when the filtered set point (and process output) becomes greater then 70%. Fig. 9 b) present the evaluations.

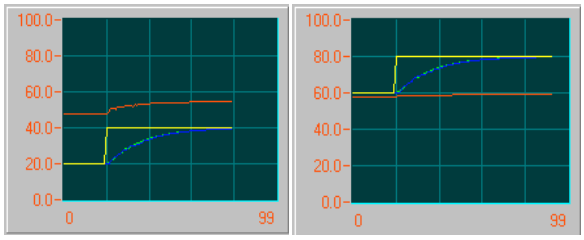

Fig. 9. a) switching test b) switching test

In both test a), b) where the set point has an important step there are no shocks or there are very small oscillations in command evolution. Increasing of algorithms superposing zones or increasing or models number to 4 or 5 can eliminate the small oscillations.

5. CONCLUSIONS

Because this method suppose the calculus of set point value for corresponding values of manual command and process output, as if the algorithms is "traverse" in opposite sense, it was used the name "feedback references method".

The method was successfully tested on process simulator software application with nonlinear characteristic, using a multimodel controller real time software application that contains 3 control algorithms. Because the computer used to run software applications provide enough calculus resources it was chosen the first variant of implementation – with all algorithms active – ensuring the possibilities of very fast switching (one step).

Supported by achieved performances, proposed switching method can be successfully use in multimodel structures for fast process real time control.

REFERENCES

Balakrishnan., J., (1996). Control System Design Using Multiple Models, Switching and Tuning, Ph. D. Dissertation, University of Yale, USA.

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

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

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

Lupu, C., D. Popescu and S. Manza, (2000). "*Advanced Techniques for Non-Linear Control Systems*", 16th IMACS World Congress 2000, Lausanne, August 21-25.

Narenda, K. S. and J. Balakrishnan, (1997). Adaptive Control using multiple models, *IEEE Transactions on Automatic Control*, vol. 42, no. 2, February, page. 171 – 187

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