Fast Disturbance Rejection in MIMO Process Based on Algorithms Switching

Ciprian Lupu
Department of Automatic Control and Systems Engineering
University Politehnica of Bucharest
Bucharest, Romania
ciprian.lupu@acse.pub.ro

Florin-Dan Secuianu
Department of Automatic Control and Systems Engineering
University Politehnica of Bucharest
Bucharest, Romania
dan.secuianu@gmail.com

Cosmin-Constantin Mihai
Department of Automatic Control and Systems Engineering
University Politehnica of Bucharest
Bucharest, Romania
mihaicosmin2007@hotmail.com

Catalin Petrescu
Department of Automatic Control and Systems Engineering
University Politehnica of Bucharest
Bucharest, Romania
catalin.petrescu@acse.pub.ro

***Abstract*— The restrictions imposed on real-time functioning are sometimes particularly severe and require firm action. At the same time, the design of the control algorithms for perturbations rejection includes a multitude of theoretical and practical solutions that exist in the literature. Here, we can note the methods based on robustification, on algorithms with specific degrees of freedom, etc. The paper presents a method based on the very rapid switching of the control algorithms, proposing at the same time a new switching method. The solutions proposed are tested on an MIMO process in a real time application in the vehicle control area.**

Keywords—real time, switching, disturbance rejection, control algorithm, automotive;

I. Introduction

Reference tracking and perturbations rejection are the most important functions of an control system. As a matter of fact, the literature and current practice contain a multitude of methods with outstanding results. Among these, we can highlight those based on the robustification of control loops [1-3], on the design of different performances in reference tracking and perturbations rejection [4] etc. Some of these are also found in real-time applications.

And yet, the emergence of new areas of applicability (eg smart cars, smart city, smart grid, etc.) or the advance of the current technology raise new and new challenges for automation control specialists. Moreover, real-time functioning/operation conditioning, having limited hardware and software architectures for minimal energy consumption, requires a fine balance between theory and practice.

This paper attempts to present the results of a practical research for the implementation of an assisted (autonomous) tracking system of the lanes. The subject falls within the broader scope of "smart cars" that can move on their own.

In the context of using electric vehicles, one of the current solutions is based on individual propulsion systems (engine ensemble) for each wheel or for each final power transmission element (propeller, caterpillar track, etc.) [2], [5]. This approach involves the control of a multivariable system, which is, most of the times, very complex.

In the current practice, the control solution of several multivariable systems often calls for the decentralization of the control loops, trying to reduce the multivariable system to the simplest one, possibly of SISO type, controlled by classical algorithms.

At the same time, the very high perturbations that can affect a system with a certain classical configuration cannot be rejected in a very short time. As it results from the practice, a large part of the solutions is based on the robustification of the control algorithms [3], [4], etc. Clearly, the cost of these approaches is high and sometimes extremely complicated for the real-time implementation.

The solution proposed for the rejection of large perturbations is based on a multi-controller structure that allows to switch the control algorithms. For the normal functioning/operation, a classical algorithm is used, e.g. PID, RST, etc. specifically designed for reference tracking and rejecting "normal" perturbations. If the perturbations exceed a certain threshold value, the algorithm switches to a specially designed one that has a firm, very fast action.

The switching method is very fast, assuming the "freezing" and "de-freezing" of the main loop, a procedure that uses a minimum of hardware - software resources.

The particularity of the axial (left-right) symmetry of the controlled process (an electric vehicle model) supports the proposed structure applicability.

II. Proposed Control Structure

A. *Existing Solutions*

The use of multi-model or multi-controller control structures provides outstanding results for complex, non-linear systems or with different operating regimes. Some examples can also be found in [6, 7] or [8].

Algorithms switching in these structures is a specific problem and has various solutions.

The most widely used switching procedure relies on the control algorithms "hot" state [6], [8]. The procedure deals with operating **all** the algorithms, as though they were connected to the real process and with applying the calculated command to the process, but just the most adequate one. We obtain efficient switching through decoupling (physical / informational) the active algorithm control and coupling an (other) command which is calculated by another algorithm.

For simple algorithms (PI, PID, or any other structure that hasn't got high numerical complexity), the outcome in practice is good and the bumps which are generated are, in general, small or acceptable.

Nevertheless, **all** the control algorithms continuous operation in the MM structure needs an outstanding hardware-software architecture, because its complexity grows together with the number of the necessary models / algorithms.

A variant of this solution proposes, similarly to fuzzy systems, to mix the algorithm commands in a proportion that depends on the adequacy of the respective algorithms relative to the current functioning point [9].

Another eye-catching solution, proposed in [10, 11] also known as the "backward computed reference", relies on switching the manual - automatic (M -> A) regimes with as few bumps as possible.

Still, this solution also requests a significant hardware-software effort, which can, at times, be higher than the classical one.

The real-time hardware architecture used in terrestrial or aerial electric vehicles has several restrictions in terms of energy consumption and complexity [12].

For this reason, we propose for the study an architecture that uses a minimum of resources.

B. Proposed Solution

The proposed structure is based on two algorithms that will be switched depending on the contextual situation of the controlled process. For the case in which the process is "normal," algorithm 1 - "classical" ensures the reference tracking and rejection of "normal" (small-medium) perturbations. If the value of the perturbations is high or very high, switching to algorithm 2, of multi-position type, will take place. This is a three-positional algorithm with several values of the "positive" and "negative" commands.

Figure 1 shows the general structure. The values are as follows: *sp* –system reference, *u* –calculated command of the whole system, *y* – adjusted value, u*1*, *u2* – calculated commands by the two algorithms, *switch, freeze* – commands given for algorithms switching, respectively for "freezing" algorithm 1.

Applying the second algorithm (No. 2) brings the process into the "normal" functioning band after a number of sampling periods. This situation leads to switching to the first control algorithm (No. 1 - the "classical" one). The functioning is explained in equation (1).

$$u(k)=\begin{cases} u_{classic}(k) & if\ \ -e_{\lim}\le e_{system}\le e_{\lim} \\ u_{multi-positional} & if\ \ -e_{\lim}> e_{system}> e_{\lim} \end{cases} \quad (1)$$

When switching / activating, the multi-positional algorithm 2, will apply a command equal to the last command calculated by the "classical" algorithm u_f – *u frozen*, to which a "positive" or "negative" value is added, corresponding to the error value which affects the process.

Typically, the greater the error, the more "firm" the corrective action, having the option of applying thresholds or steps. Clearly, these thresholds can be replaced by the value calculated in proportion to the error value.

Figure 2 shows this structure. The notations included are as follows: err = sp-y error (of control); +/- L_i_ON, +/- L_i_OFF limits of activation, deactivation of the commands steps; +/- u2_v1, +/- u2_v2 commands steps.

The steps variant is favored by how the system error is presented. In many systems, this information has a fuzzy representation (small - large - very large).

After rejecting the perturbations, before reactivating algorithm 1 ("classical"), the command applied to the process equals u_f.

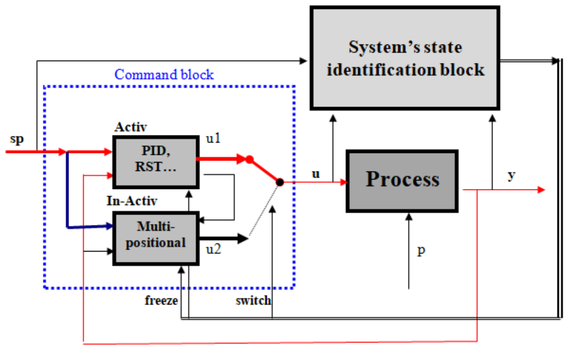


Fig. 1. Multi algorithms switching structure

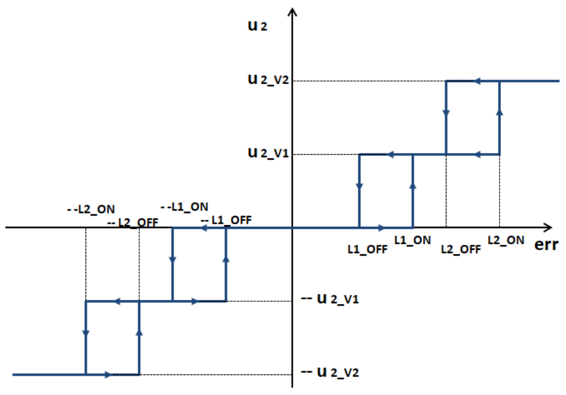


Fig. 2. Multi positional algorithm structure

C. *Switching Procedure Implementation*

One of the problematics specific to those types of multi-model or multi-algorithm structures [6-11], [13] is the command switching procedure. This should be done as quickly as possible with the commands shocks (determined by the procedure itself) as small as possible.

As presented in the introduction, the proposed solution is based on the functional "freezing" of the "classical" algorithm, of PID, RST type, etc.. This approach provides a minimum number of arithmetic operations and a non-shock passage of the command calculated at the reactivation ("de-freezing").

In order to clarify this procedure, we will make a few references to the real-time implementation of an RST algorithm [4]. For example, it can offer the possibility of imposing different performances in rejecting perturbations and reference tracking. The choice is justified by its general form, the implementation of a simpler numerical algorithm (PID) being obviously easier after understanding these aspects.

The form of this algorithm is the following:

$$S(q^{-1})u(k)+R(q^{-1})y(k)=T(q^{-1})y^{*}(k) \tag{2}$$

in which: u(k) – the output of the algorithm, y(k) – the output of the process, y^*(k) – the trajectory that is imposed or filtered set point. The polynomials which correspond to them are as follows:

$$\begin{aligned} R(q^{-1}) &= r_0+r_1q^{-1}+\ldots+r_{n_R}q^{-n_R}\\ S(q^{-1}) &= s_0+s_1q^{-1}+\ldots+s_{n_S}q^{-n_S}\\ T(q^{-1}) &= t_0+t_1q^{-1}+\ldots+t_{n_T}q^{-n_T}\end{aligned} \tag{3}$$

The representation of the closed-loop control is shown in [10].

The control algorithm in (3) can be rewritten as:

$$\begin{aligned} u(k)=\frac{1}{s_0}\Bigg[&\sum_{i=0}^{n_T}t_iy^{*}(k-i)-\\ &-\sum_{i=0}^{n_R}r_iy(k-i)-\sum_{i=1}^{n_S}s_iu(k-i)\Bigg]\end{aligned} \tag{4}$$

n_S , n_R , n_T are the degrees of the corresponding polynomials end entail the memory required for controller software algorithm implementation. For example, if n_R=2, then three memory locations must be kept for the output of the process: y(k), y(k-1), y(k-2). The same rule can be used for u(k) and y^*(k).

Inside the infinite loop [10] of the command calculation procedure *u(k)*, the code implementing equation (4) is executed after the acquisition of the controlled value *y(k)* has occurred.

Still in this infinite loop, these memory variables of the algorithm *(y(k), u(k)* and *y*(k))* go through an update process for a new cycle performed at the next period of the corresponding sampling:

$$\begin{aligned} &u(k-2)=u(k-1);\ u(k-1)=u(k);\\ &y(k-2)=y(k-1);\ y(k-1)=y(k);\\ &y^{*}(k-2)=y^{*}(k-1);\ y^{*}(k-1)=y^{*}(k);\end{aligned} \tag{5}$$

In this context, the "freezing" of the algorithm involves the suspension of the mathematical operations (implemented in the code) described by equations (4) and (5) for a number of *n* sampling periods. All the memory variables will retain their last values until the “de-freezing” time.

During the suspension of these operations, algorithm no. 2 - "multi-positional”, is executed.

III. Case Study and Experiments

The performances of the control structure and the rapidity of the switching method have been tested on the electric vehicle model. In addition to movement controlling the hardware -software structure has the role of implementing a "lane assist" functionality, which is also present in the case of a larger "smart car"-type system. The basic idea in the implementation was to find a real-time solution that uses as few resources as possible.

A. *Hardware and software structure*

Structurally, the controlled process is a MIMO type one. The vehicle has an axial symmetry (left-right, L-R), with electric motors for each of the four wheels. There are two main loops for the speed control for each side (L-R), each loop commanding the speed of two motors/engines. In order to change the direction of driving, we can impose an L-R speed difference, since the system does not have guiding wheels or other specific elements. This particularity renders the system applicable also to caterpillar track [14, 15] or propeller [5] vehicles.

Figure 3 shows this system. In the figure, the values are as follows: *sp_L, sp_R* the references of the two control loops; *y_L, y_R* the L, R speeds (coming from the encoders); *u_f_L, u_r_L, u_f_R, u_r_R* – the commands given to the motors front (f)/back (r) on the left side (L), respectively the right side (R).

Vehicle positioning to the demarcation line(s) of the traffic

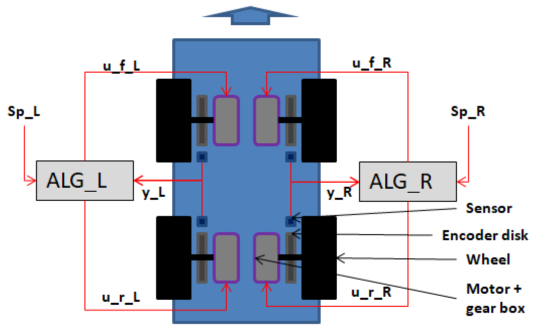


Fig. 3. Tested vehicle structure

lanes can be done by camera(s) or two L-R sensor lines (the solution used in this case study, as well). Each line has 3 sensors. Figure 4 shows this arrangement:

The information provided by this positioning system is discrete (correct positioning = normal error, large / very large deviation to the left / right = large / very large error).

In the case of a normal error, the control algorithms of the control loops are the "normal" ones (PID in RST variant). In the case of large or very large positioning errors, the algorithms are "frozen" and the multi-positional algorithm takes control.

In the figure, driving on lane 2 (L2), positions b) and c) is taken into account indicating a deviation of the vehicle to the left. Receiving the marking only by the middle sensors (case a) means a correct positioning (normal error). The simultaneous involvement of another side sensor (case b) signals a large error and the situation in which only one side sensor is active (case c) causes a very large error.

At the time of algorithm 2 activation, compared to the commands calculated by algorithm 1, "classical", a value corresponding to the type of error (large / very large) is applied additively. Depending on vehicle deviation (L-R), the values applied are positive or negative.

For reasons of efficiency, algorithm 2 is unique for both L-R parts because, for directional correction, both speed control loops are "frozen" simultaneously, the correction command being applied positively to one side and negatively for the other.

The four motors (integrated into the gearbox and driver structure) do not have the same behaviors, so that commands with different values (L-R) are required for their drivers. After leaving from the spot, which causes a slight L-R oscillation, the control loops "find" the values of the commands, which cause a rectilinear movement at a certain reference speed

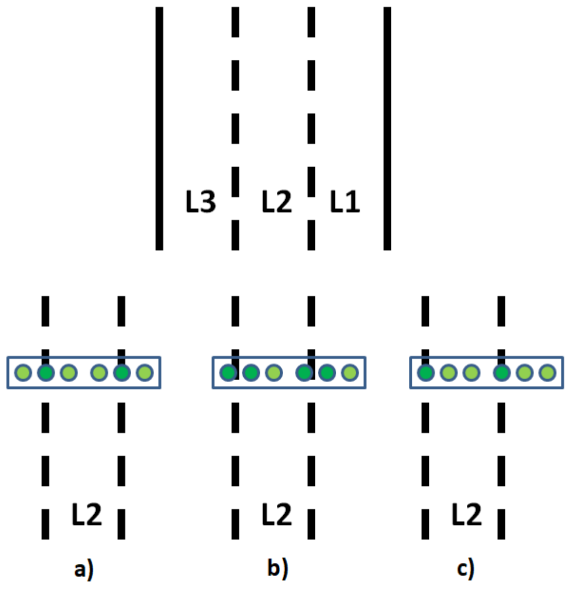


Fig. 4. Lane position detection system.

(imposed).

Another particularity of the speed control system (L-R) is the "poor" information, the encoder discs having a small number of slots.

B. Tests and results

The system testing was performed comparatively by means of a software version containing only the classical speed control algorithms (L-R) (Figure 5 (a)) and by means of one that implements the structure proposed in Figure 1 (Figure 5 (b)). For the first simple variant, in the case of placement error detection, the references of the speed control loops (L-R) were modified so that the system reinserted on the followed track.

This route had straight portions and turns (slight ones specific to highway-type roads), (see Fig. 5). The straight track could be followed by both variants, but the portion including turns could be followed only by the proposed multi-algorithm variant, (see Figure 5 b). In situation a) after the first turn, the vehicle did not get back on track and left the lane.

Despite changing the driving speed or the parameters of the speed control algorithm, for case a) no better practical results were obtained. One reason is that the system is, however, of the MIMO type, and the decentralized version with classical control loops (PID) has limitations.

Figure 6 shows the physical model of the electric vehicle and its evolution on a traffic lane.

Exemplifying how to apply the command, for a large error, deviation to the right, we have as follows:

uk1_apl = uk1 - d_com; // uk1_apl – command R

uk2_apl = uk2 + d_com; // uk1_apl – command L

respectively, for a very large error, deviation to left:

uk1_apl = uk1 + D_com;

uk2_apl = uk2 - D_com;

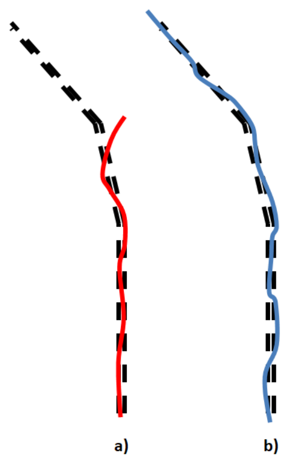
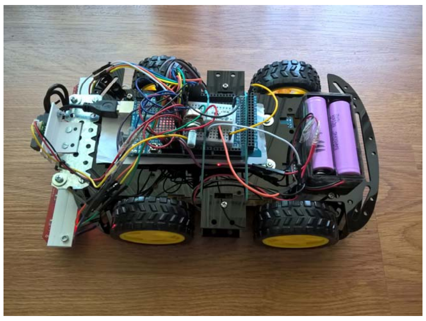


Fig. 5. Lane following results; a) only speed L-R loops, b) proposed control structure; (with black dash line – lane trajectory).



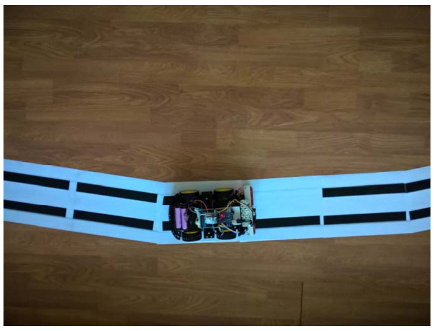


Fig. 6. Experimental electric vehicle and lane following evolution..

The PID L-R algorithms implemented, after an identification of the vehicle model dynamics are as follows:

float Kr2 = 0.25; // Kr – proportionality constant

float Ti2 = 15.0; // Ti – integration constant

float Td2 = 0.0; // Kr – derivation constant

float Te2 = 0.5; // Te – sampling period (in seconds)

The multi-positional algorithm has two (percent) values of the applied steps, determined experimentally in a first phase:

float d_com = 10; // large error step

float D_com = 20; // very large error step

The command steps applied to the vehicle do not manifest brutally due to the time constants and integrating effect of the electric motor systems that include besides the motor itself, the gearbox and the driver (mini-adjustable speed drive).

After returning to the classical algorithms, the possible driving speed differences (L-R) are treated as perturbations.

IV. Conclusions

The paper presents a useful solution for the rejection of large perturbations that cannot be rejected by means of the classical algorithms, without significant efforts in the hardware and software architecture on which we perform the implementation.

The switching of the main algorithm is based on the classical loop suspension (freezing) and the application of a command correction, which is proportional to the value of the perturbation.

The "corrective" command is applied for a certain amount of time (related to error) after which we return to the value from which we started. The control algorithm (classical) is immediately "de-frozen" and continues to operate.

The advantages of the solution include: simplicity and applicability in TR (speed); as disadvantages, we can first notice the lack of a very elaborate theoretical fundamentation.

Acknowledgment

This paper was financed together with the U. P. B. - A.C.P.C. Res. Center, and through the Faculty of Automatic Control and Computer and S.C. Control Engineering & Energy S.R.L. projects.

References

[1] Kang-Zhi Liu, Yu Yao, (2016), *Robust Control: Theory and Applications*, John Wiley & Sons, 2016, ISBN 9781118754375.

[2] Y. Li, B. Li, X. Xu and X. Sun, "A Nonlinear Decoupling Control Approach Using RBFNNI-Based Robust Pole Placement for a Permanent Magnet In-Wheel Motor," in *IEEE Access*, vol. 6, pp. 1844-1854, 2018.

[3] Pait, Felipe M., and Fuad Kassab. "Parallel algorithms for adaptive control: Robust stability." *Control using logic-based switching*. Springer, Berlin, Heidelberg, 1997. 262-276.

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

[5] T. Ryan and H. Jin Kim, "LMI-Based Gain Synthesis for Simple Robust Quadrotor Control," in *IEEE Transactions on Automation Science and Engineering*, vol. 10, no. 4, pp. 1173-1178, Oct. 2013.

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

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

[8] J. P. Hespanha, D. Liberzon and A. S. Morse, "Hysteresis-based switching algorithms for supervisory control of uncertain systems", Automatica, vol. 39, no. 2, Feb. 2003.

[9] M. Kuipers and P. Ioannou, "Multiple Model Adaptive Control With Mixing," in *IEEE Transactions on Automatic Control*, vol. 55, no. 8, pp. 1822-1836, Aug. 2010.

[10] C. Lupu, D. Popescu, B. Ciubotaru, C. Petrescu and G. Florea, "Switching Solution for Multiple-Models Control Systems," *2006 14th Mediterranean Conference on Control and Automation*, 2006, pp. 1-6.

[11] C. Lupu, A. Udrea, O. Pagès and M. Azzouzi, "Multi model control solution for some classes of hysteretic processes," *Control & Automation (MED), 2010 18th Mediterranean Conference on*, Marrakech, 2010, pp. 1103-1108.

[12] Bin Li, Avesta Goodarzi, Amir Khajepour, Shih-ken Chen & Baktiar Litkouhi (2015) An optimal torque distribution control strategy for four-independent wheel drive electric vehicles, Vehicle System Dynamics, 53:8, 1172-1189.

[13] V. Hassani, J. P. Hespanha, M. Athans, and A. Pascoal, "Stability analysis of robust multiple model adaptive control," in Proceedings of the 18th IFAC World Congress, August–September 2011.

[14] S. Kojima, K. Ohno, T. Suzuki, T. Westfechtel, Y. Okada and S. Tadokoro, "Motion control of tracked vehicle based on contact force model," *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, 2016, pp. 1177-1183.

[15] R. Stanciu and C. Sorandaru, "Low-cost visually servoed tracked vehicle," *IEEE EUROCON 2017 -17th International Conference on Smart Technologies*, Ohrid, 2017, pp. 680-685.