International Journal of Soft Computing 3 (2): 147-154, 2008
ISSN: 1816-9503


# A Computational Time Requirement Comparison Between Two Approaches in MBPC

[1]Ben Nasr Hichem and [2]Faouzi M. Sahli
[1]Research Unit on Numeric Control of the Industrial Processes (CONPRI),
Institut Supérieur des Etudes technologiques de Sfax Route de Mahdia km 2.5-Sfax (ISET),
B.P.88 A-3099 El Bustan, Tunisia
[2]Ecole Nationale d'Ingénieurs de Monastir (ENIM), Rue Ibn Eljazzar, 5019 Monastir, Tunisia

**Abstract:** In this study a comparison between two approaches in MBPC is made, the first one based on nonlinear programming NLP and the second based on a novel multi-agent controller for nonlinear models. Although, a nonlinear model predictive approach can achieve good performance and constraints fulfillment, its computational burden does not allow a real-time implementation and restricting the application to slow processes. In order to decrease the complexity of the controller, we propose a novel MPC scheme based on multi-agent controller approach. Simulation results show the effectiveness of the novel MPC scheme in reducing the computational burden, while achieving good results compared to NLMPC controller. Hence, the proposed approach has the potential to be applied to systems with faster time constants.

**Key words:** Nonlinear system, predictive control, constraints, multi-agent controller

## INTRODUCTION

Nonlinear Model Predictive Control (NMPC) is a powerful approach to deal with the complexity of the related nonlinear control problem, if an economic objective function is employed. One major element of any NMPC implementation takes difficulty into account to solve nonlinear dynamic optimization problem to obtain optimal control trajectories. During the past decade the implementation strategies of NMPC has been successfully applied to relatively slow plants and the most important problem discussed in the IFAC Workshop (2006) is how to implement this control strategy on relatively fast systems. In NMPC, the optimization problem of finding the sequence of actions can be of large size, in particular when the control horizon over which actions are computed becomes larger, the number of variables of which the controller has to find the optimal value increases quickly. Also, the resources needed for computation and memory may be high, increasing more when the prediction horizon increases.

Finally, the feasibility of the solution to the overall control problem of the system is not guaranteed. However, fast dynamic systems cause the need of short sampling time and the dynamic optimizer must solve the problem in a time span less than the sampling time, a necessary fast solution of the optimization problem is then so difficult to obtain within the sampling time. Most of the research has focused on computations carried out by one agent. In Negenborn *et al*. (2004) a survey how a distributed multi-agent MPC setting can reduce the computations of a single MPC agent. In Didier (2006) and Aswin (2006), a distributed model predictive control is considered and the proposed strategy allows dramatic reduction of the computational requirement for solving large-scale nonlinear MPC problem due to computation parallelism.

The goal of this research is to develop a method that minimize the computational time requirement in NMPC strategy. The reduction of computational time is approached by a novel structure of MPC scheme, in which the concept of multi-agent is applied. An algorithm for controller reconfiguration for non-linear systems based on a combination of a multiple model estimator and a generalized predictive controller is presented by Kanev and Vergaegen (2000), in which a set of models is constructed, each corresponding to a different operating condition of the system and interacting multiple model

**Corresponding Author:** Ben Nasr Hichem, Research Unit on Numeric Control of the Industrial Processes (CONPRI), Institut Supérieur des Etudes technologiques de Sfax Route de Mahdia km 2.5-Sfax (ISET), B.P.88 A-3099 El Bustan, Tunisia

estimator is utilized to yield a reconstruction of the state of the non-linear system. A parallel controller structure for system that shows multiple modes is presented by Kamalasadan (2007) and in the multi-agent model predictive control considered here, there are multiple agents, each of which uses a model of its subsystem to determine which action to take. The proposed novel multi-agent controller reposes on the fact that each system can be decomposed into a sub-system and a control problem is associated with its own goal. The action of each agent, can be performed on the system has to be chosen in such a way that the task of the system is achieved, keeping in mind the dynamics of the system and possible constraints on the actions. In this study, we show the reduction in computational time in the case where only constraints are input constraints.

## MATERIALS AND METHODS

This study is the part of research works in thesis carrying on predictive control of fast dynamics systems. The whole approach has been illustrated by simulation in the Matlab 6.1 environment.

**Novel multi-agent controller:** The main idea of the proposed concept model predictive control is to transform the nonlinear optimization procedure used in a standard way into sub-problems, in which the global task can be resolved. To reduce on-line computational requirement and improve control performance of the whole system, a block diagram of the novel concept of model predictive control is proposed in Fig. 1. The objective of this approach is to regulate the system output to the expected values and satisfying the above constraints. This can be done as follows. The global system can first be decomposed on sub-systems independent of one another, for each sub-system an MPC unit sub-system is made constituting the agent controller i. Based on analytical solution $u_i$ which correspond to the solution of the local receding horizon sub-problem, a logic unit switching tries to find the best sequence of actions given the desired trajectory. A sequence of actions that bring the global system in a desired trajectory are made and avoid any violated constraints on actions. A fuzzy controller is also made on , in objective to take handle the results of the actions on the global system and monitor the closed-loop system if necessary.

The multi-agent controller consists of synchronizing the outputs of the sub-models with the output of the true system at every decision step. In fact, at every decision step an MPC agent send its action, observes the true

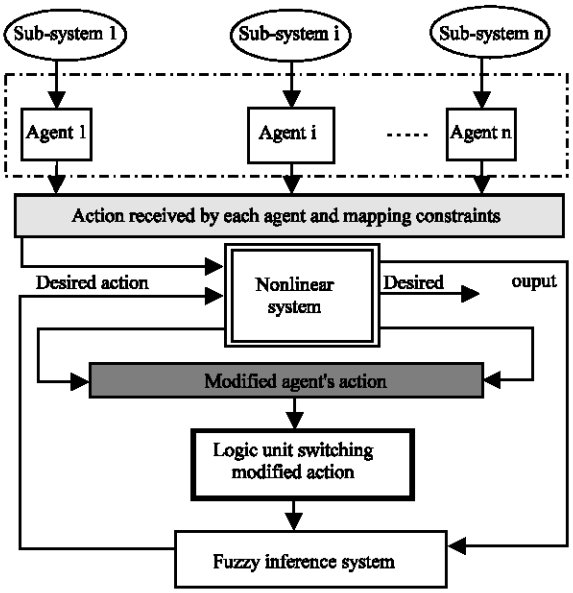


Fig. 1: Block diagram of multi-agent controller

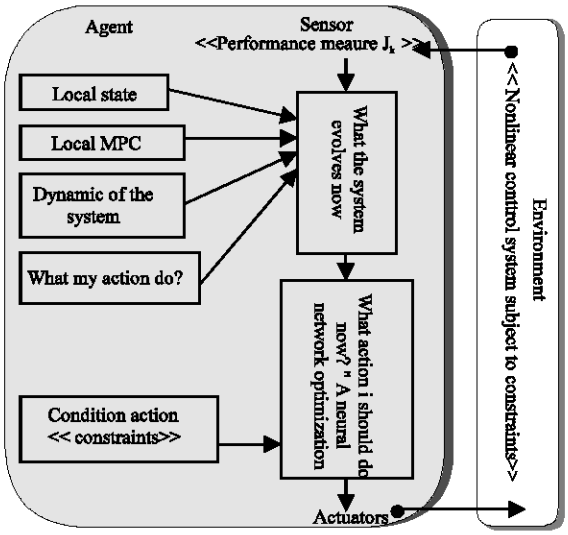


Fig. 2: Model based reflex agent

system and updates its control. Based on the model based reflex given in Fig. 2, each agent should strive to do the right thing, based on what it can perceive and the actions it can perform. The right action is the one that will cause the agent to be the most successful. In the multi-agent context, the controllers are the agents and the non linear plant is the environment.

A performance measure $J_k$ used as an objective criterion for success of an agent's actions, based on the

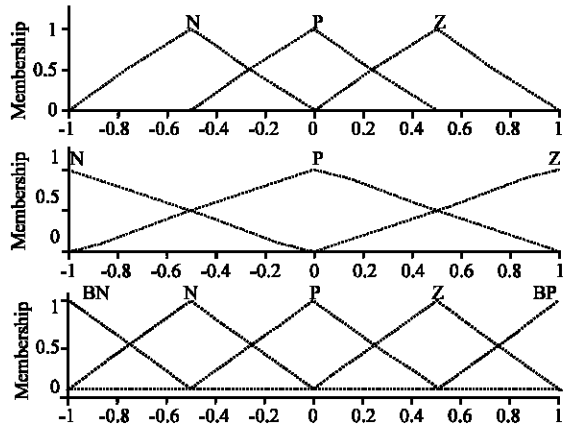


Fig. 3: Linguistic values and membership functions (a) MF for error, (b) MF for rate of change error, (c) MF for modified control action

Table 1: Fuzzy rules used in the infernce system

| If | And | Then |
|---|---|---|
| --- | --- | --- |
| e | Δe | Δ |
| Negative | Negative | Positive big |
| Negative | Zero | Positive |
| Negative | Positive | Zero |
| Zero | Negative | Positive |
| Zero | Zero | Zero |
| Zero | Positive | Negative |
| Positive | Negative | Negative Big |
| Positive | Zero | Negative |
| Positive | Positive | Negative Big |

output errors $\varepsilon_k$ for each agent's action and choosing that corresponds to the minimum. The performance measure used is given by: $J_k = \varepsilon_k - \varepsilon_{k-1} e^{-\lambda}$, $\lambda>0$.

Where: $\varepsilon_k = y_{Ref} - y_{ai}$, $y_{Ref}$: The desired set point and $y_{ai}$: The plant output after agent's action i.

A Mamdani-type FIS is used as fuzzy inference system (Sanjuan *et al*., 2006) it is designed as multi-input, single-output system where error (e) and rate of change of error (Δe) are the inputs, added control action (Δu) is the output. Triangular membership functions are used to relate variables with the degree of membership to linguistic values of their corresponding fuzzy variables. Three linguistic values are defined for both error and rate of change error: Negative (N), Positive (P), Zero (Z). Five linguistic values are defined for change control action: Big Negative (BN), Negative (N), Zero (Z), Positive (P) and Big Positive (BP). Figure 3, shows the membership functions relating input variables with degree of membership of the fuzzy variable to each linguistic value and Table 1 presents the nine rules in the rule base used to construct the FIS.

**Sub-system decomposition:** In the proposed concept, the global problem is typically broken up into a number of smaller problems. By decomposing the system into sub-systems and sub-problems, the computational burden can be lowered. A procedure of fuzzy modeling based on TS fuzzy model is then considered (Abonyi, 2003), in witch the rule has the following form:

$$R_j : \text{if } z_1(k) \text{ is } A_{j,1} \text{ and } ..... \text{ and } z_n(k) \text{ is } A_{j,n}$$

Then

$$y^j(k+1) = \sum_{i=1}^{n_y} a_{i,j} y(k-i+1) + \sum_{i=1}^{n_u} b_i^j u(k-i-n_d+1)$$

Where the element of z(k) are a subset of {y (k),..., y (k-n),... u (k-1),... u (k-$n_u$)} $A_{j,i}$ ($z_i$) is an antecedent fuzzy set for the i = 1… .nth input in the jth rule and $a_{i,j}$, $b_i$, are linear model parameters. The TS model is than linear combination of linear models A triangular membership function used and arranged by Ruspini-type partition keeping the sum of the membership degrees equal to 1. Therefore the global output can be expressed as:

$$y_p(k+1) = \sum_{j=1}^{k} y^j(k+1)$$

With fixed order equal to two, the local model has the following regressors:

$$y^j(k+1) = a_1 y(k-1) + a_2 y(k-2) + b_1 u(k-1) + b_2 u(k-2)$$

which can be rewritten in to state space form of:

$$x_i(k+1) = A_i x(k) + B_i u(k)$$
$$y_i(k) = C_i x_i(k)$$

Where:

$$A_i = \begin{bmatrix} a_1 & 1 \\ a_2 & 0 \end{bmatrix}; B_i = \begin{bmatrix} b_1 & b_2 \end{bmatrix}^T; C_i = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

**MPC unit sub-system:** A block diagram of MPC unit for each sub-system is shown in Fig. 4. This scheme constitutes the work of each agent in which the action is sent to the nonlinear system and modified with a feed-forward neural network in order to minimize the control error $\varepsilon_i$ by changing the regressor value inputs action using a Least Square Estimation.

In this research, the neural network is represented by feed-forward single-input single output neuron (Liu, 2001)

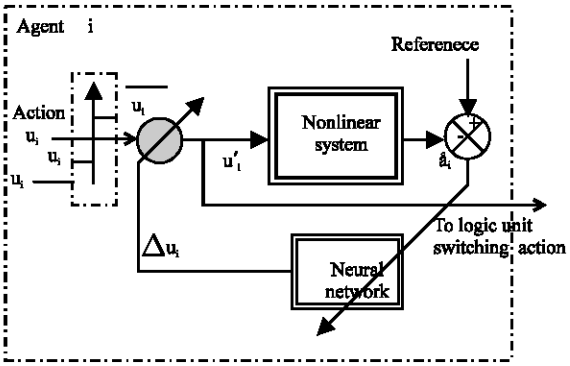


Fig. 4: Block diagram MPC unit

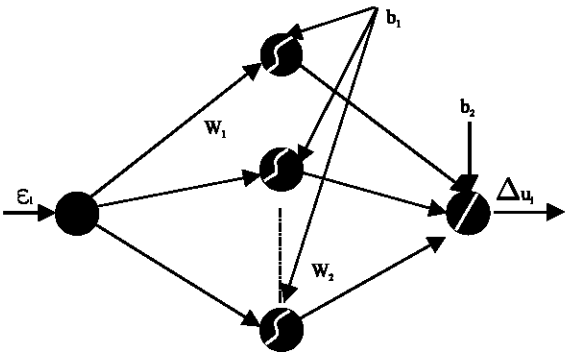


Fig. 5: Neural network architecture

and it is consisted of three hidden layer nodes with tangent sigmoid transfer function and one output layer with linear transfer function. The architecture of the neural network is given in Fig. 5 and described by the following function:

$$f_{NN}(\Delta u_i,\theta) = W_2 \tanh(W_1\varepsilon_i + b_1)$$
$$+b_2 = \sum_{j=1}^{N_h}(w_{2j} f(w_{1j}\varepsilon_i + b_1)) + b_2$$

Where:

$N_h$ : The number of hidden neuron.
$\varepsilon_i$ : The input of the network.
$W_1$ : The weight of hidden layer.
$B_1$ : The bias of hidden layer.
$W_2$ : The weight of output layer.
$b_2$ : The bias of output layer.
$W_{1f}$ : The jth element of $W_1$.
$W_{2f}$ : The jth element of $W_2$.
$b_{1f}$ : The jth element of $b_1$.

During the identification procedure these parameters are collected into the $\theta$ parameter vector, that is: $\theta = \{w_1, w_2, b_1, b_2\}$. In the optimization procedure, the first and second layers are selected randomly. They are uniformly randomly distributed between 0.5 and -0.5. The input and target are normalized with a mean value of zero and standard deviation of 1. The neural network tries to optimize the parameters $\theta$. The method of Levenberg Marquardt was designed for the optimization due to its properties of fast convergence and robustness. It leans on the techniques of the nonlinear root mean square answering to:

$$\underset{\theta_i}{\text{Min}} \sum((f_{NN}(\Delta u_i,\theta_i) - \varepsilon_i)^2)$$

The main incentive of the choice of the algorithm of Levenberg Marquardt rests on the fast guarantee of the convergence toward a minimum. The method of LM consists in determining once again $\theta_i$ according to:

$$\theta_{i+1} = \theta_i - (J^T J + \mu I)^{-1} J^T E$$

Where:

$$J = \frac{\partial f_{NN}}{\partial \theta}$$

is the Jacobian matrix that contains first derivatives of the network errors with respect to the weights and biases and E is a vector of network error. The Jacobian matrix is computed through a standard back propagation. The recursive least square method is used on line has end to express the tracking error in control in order to optimize the control law (Wenzhi *et al*., 2006). In order to find the action of each agent, the model of each sub-system i is represented by a discrete model of the form:

$$x_i(k+1) = A_i x_i(k) + B_i u_i(k)$$
$$y_i(k) = C_i x_i(k) \qquad (1)$$

The control action sent to the nonlinear system consists of its local optimal control taken after minimizing a local cost function given by:

$$V_i(k) = \sum_{i=Hw}^{Hp} \left\| \hat{y}_i(k+i/k - r(k+i) \right\|_{Q(i)}^2 + \sum_{i=0}^{Hu-1} \left\| \Delta \hat{u}_i(k+i/k) \right\|_{R(i)}^2$$

Where, $H_w$, $H_p$ are the minimum and maximum prediction horizon, $H_u$ is the control horizon and Q, R are a suitable weighting matrices. Consider the sub-system given by (1), the best prediction of x $(k + H_p|k)$ can be written as:

$$\begin{bmatrix} x_i(k+1|k) \\ . \\ . \\ . \\ . \\ x_i(k+H_u|k) \\ x_i(k+H_u+1|k) \\ . \\ . \\ . \\ . \\ x_i(k+H_p|k) \end{bmatrix} = \begin{bmatrix} A_i \\ . \\ . \\ . \\ . \\ A_i^{H_u} \\ A_i^{H_u+1} \\ . \\ . \\ . \\ . \\ A_i^{H_p} \end{bmatrix} x_i(k) + \begin{bmatrix} B_i \\ . \\ . \\ . \\ . \\ \sum_{j=0}^{H_u-1} A_i^j B_i \\ \sum_{j=0}^{H_u} A_i^j B_j \\ . \\ . \\ \sum_{j=0}^{H_p-1} A_i^j B_j \end{bmatrix} u_i(k-1) + \begin{bmatrix} \\ \\ B_i \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 0 \\ (A_i B_i + B_i) \ldots\ldots\ldots\ldots\ldots\ldots\ldots 0 \\ \ldots\ldots \\ \sum_{j=0}^{H_u-1} A_i^j B_i \ldots\ldots\ldots\ldots\ldots\ldots\ldots B_i \\ \sum_{j=0}^{H_u} A_i^j B_i \ldots\ldots\ldots\ldots\ldots (A_i B_i + B_i) \\ \ldots\ldots\ldots \\ \sum_{j=0}^{H_p-1} A_i^j B_i \ldots\ldots\ldots\ldots \sum_{j=0}^{H_p-H_u} A_i^j B_i \end{bmatrix} \begin{bmatrix} \Delta u_i(k|k) \\ . \\ . \\ . \\ \Delta u_i(k+H_u-1|k) \end{bmatrix}$$

And the prediction local predictive sub-system for future time instants as:

$$\begin{bmatrix} y_i(k+1|k) \\ . \\ . \\ . \\ . \\ y_i(k+H_u|k) \\ y_i(k+H_u+1|k) \\ . \\ . \\ . \\ . \\ y_i(k+H_p|k) \end{bmatrix} = \begin{bmatrix} C_i A_i \\ . \\ . \\ . \\ C_i A_i^{H_u} \\ C_i A_i^{H_u+1} \\ . \\ . \\ . \\ . \\ C_i A_i^{H_p} \end{bmatrix} x_i(k) + \begin{bmatrix} C_i B_i \\ . \\ . \\ C_i \sum_{j=0}^{H_u-1} A_i^j B_i \\ C_i \sum_{j=0}^{H_u} A_i^j B_j \\ . \\ . \\ C_i \sum_{j=0}^{H_p-1} A_i^j B_j \end{bmatrix} u_i(k-1) + \begin{bmatrix} \\ \\ C_i B_i \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 0 \\ C_i (A_i B_i + B_i) \ldots\ldots\ldots\ldots\ldots\ldots\ldots 0 \\ \ldots\ldots \\ C_i \sum_{j=0}^{H_u-1} A_i^j B_i \ldots\ldots\ldots\ldots\ldots\ldots\ldots C_i B_i \\ C_i \sum_{j=0}^{H_u} A_i^j B_i \ldots\ldots\ldots\ldots\ldots C_i (A_i B_i + B_i) \\ \ldots\ldots\ldots \\ C_i \sum_{j=0}^{H_p-1} A_i^j B_i \ldots\ldots\ldots\ldots\ldots C_i \sum_{j=0}^{H_p-H_u} A_i^j B_i \end{bmatrix} \begin{bmatrix} \Delta u_i(k|k) \\ . \\ . \\ . \\ \Delta u_i(k+H_u-1|k) \end{bmatrix}$$

Which can be written in the following matrix form as follows:

$$Y(k) = \Psi x(k) + \Gamma u(k-1) + \Theta \Delta U(k)$$

And the cost function can be expressed as:

$$V(k) = \left\| Y(k) - T(k) \right\|_Q^2 + \left\| \Delta U(k) \right\|_R^2$$

Where:

$$Y(k) = \left[ y(k+H_w|k) \ldots\ldots\ldots\ldots y(k+H_p|k) \right]^T;$$

$$T(k) = \left[ r(k+H_w|k) \ldots\ldots\ldots\ldots r(k+H_p|k) \right]^T;$$

$$\Delta u(k) = \left[ \Delta u(k|k) \ldots\ldots\ldots\ldots\ldots \Delta u(k+H_u-1|k) \right]^T$$

By defining the tracking error:

$$\varepsilon(k) = T(k) - \Psi x(k) - \Gamma u(k-1)$$

The cost function can be rewritten as:

$$V(k) = \left\| \Theta \Delta u(k) - \varepsilon(k) \right\|_Q^2 + \left\| \Delta u(k) \right\|_R^2$$

Which is minimal to the following optimal action given by Maciejowski (2002):

$$\Delta u(k) = (\Theta^T Q \Theta + R)^{-1} \Theta^T Q \varepsilon(k)$$

The constraints are considered as input saturation constraints of the type:

$$\underline{u}_i(k) < u_i(k) < \overline{u}_i(k)$$

Where $u_i$ constitutes the action of each agent tuned by a neural network and $\underline{u}_i$, $\overline{u}_i$ constitute respectively the lower and the upper limits. The control designer used in the proposed concept has to select the tuning parameters, $H_w$, $H_p$, $H_u$ and R to meet certain stability and performance objectives.

## RESULTS AND DISCUSSION

In this study, simulations are conducted to illustrate the validity of the proposed identification and predictive control concept. A nonlinear process with output dynamic nonlinearity (Wang, 2000) is considered and described by the following equation:

$$v(k) = 0.9v(k-1) - 0.4v(k-2) + 0.9u(k-1) + 0.1u(k-2)$$

$$y(k) = \frac{y(k-1)v(k-1) + v(k)}{1 + y(k-1)^2}$$

This model is used to generate output data for a random input distribution from -2 to 2 and a Least square estimation is applied to the identification of a Volterra model given in Table 2.

The procedure of identification and modeling have been applied to the whole measures input /output come out of the global system, driving to the different following sub-system models with an order equal to 2. The modeling performance was measured by the Variance of Accounted For (VAF) index, defined by:

$$VAF = 100\%[1 - \frac{var(y - \hat{y})}{var(y)}]$$

Where y is the true process output and $\hat{y}$ is the model output obtained by simulation of the overall model. In our case study it is evaluated for 99.9998 %.

$$A_1 = \begin{bmatrix} 0.7329 & 1 \\ -0.328 & 0 \end{bmatrix}; B_1 = \begin{bmatrix} 0 \\ 0.0328 \end{bmatrix}; A_2 = \begin{bmatrix} 0.2699 & 1 \\ 0.4386 & 0 \end{bmatrix}; B_2 = \begin{bmatrix} 0.3831 \\ 0.0122 \end{bmatrix};$$

$$A_3 = \begin{bmatrix} 1 & 1 \\ -1.0982 & 0 \end{bmatrix}; B_3 = \begin{bmatrix} 0 \\ 1.0982 \end{bmatrix}; A_4 = \begin{bmatrix} 0.0493 & 1 \\ 0.1791 & 0 \end{bmatrix}; B_4 = \begin{bmatrix} 0.9463 \\ -0.1774 \end{bmatrix};$$

$$C_{1-4} = [1\ 0];$$

The proposed concept is used to control the nonlinear system given by the regressor value in Table 1 subject to: $0 \le u(k) \le 0.35$. The tuning parameters are Hw = 1; Hp = 5; $H_u$ = 5, R = 0.1 for all agents. In Fig. 6 and 7 we present respectively the closed loop response in the unconstraint and constraint case.

Table 2: Volterra regressor value

| Regressor | Value |
|---|---|
| 1 | 0.2616 |
| y (k-1) | 0.2116 |
| y (k-2) | 0.1314 |
| u (k-1) | 0.2859 |
| u (k-2) | 0.4579 |
| u (k-1)$^2$ | 0.2616 |
| u (k-1) u (k-2) | 0.4822 |
| u (k-2)$^2$ | 0.2616 |

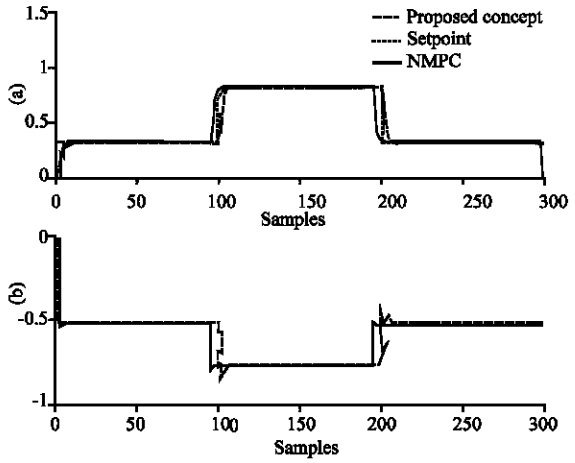


Fig. 6: Closed-loop response unconstraint case. (a) Outputs: Dotted line (setpoint), solid line (NMPC), Dashed line (proposed concept), (b) Control signals: Dashed line (proposed concept), Solid line (NMPC)

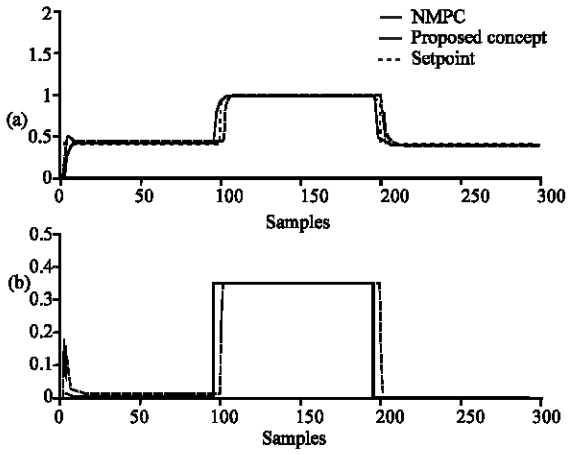


Fig. 7: Closed-loop response constraint case. (a) Outputs: Dotted line (setpoint), solid line (NMPC), Dashed line (proposed concept), (b) Control signals: Dashed line (proposed concept), Solid line (NMPC)

Based on the final error and rate of change error evolution, the action of the fuzzy supervisor is given by Fig. 8.

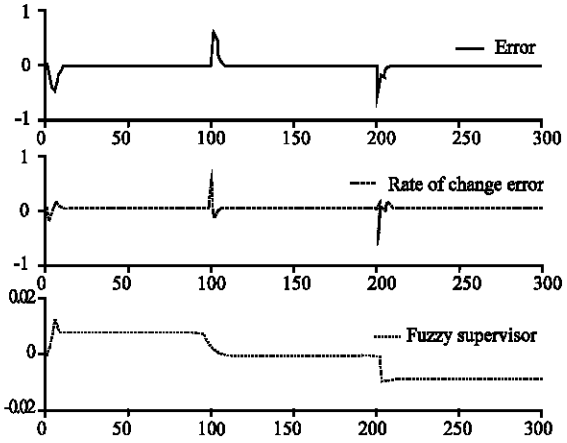


Fig. 8: Fuzzy supervisor action Solid line (control error), dashed line (rate of change error), dotted line (fuzzy action)

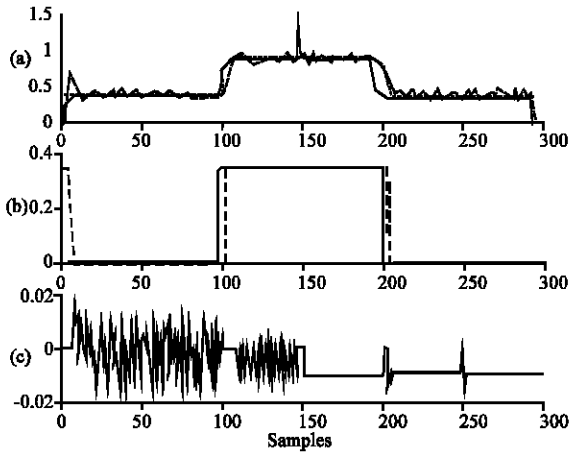


Fig. 9: Closed-loop response to disturbance and added noise. (a): Outputs: Dotted line (setpoint), solid line (NMPC), Dashed line (proposed concept), (b): Control signals: Dashed line (proposed concept), Solid line (NMPC), (c): Fuzzy action supervisor

In order to verify the proposed concept ability of rejecting disturbances, an output load disturbance of value 1.5 is introduced at the instant k = 150. An additive sequence of random noise with a zero mean value and a variance of 0.5 increase the nonlinear system under action. The results of simulation are reported in Fig. 9.

In the novel concept ,we can see the disturbance is eliminated, showing a satisfactory ability to reject disturbances better than in the standard way. The fuzzy supervisor presents with his intelligent action a good performance in control error for the global nonlinear system. In Fig. 10, the time required to compute the

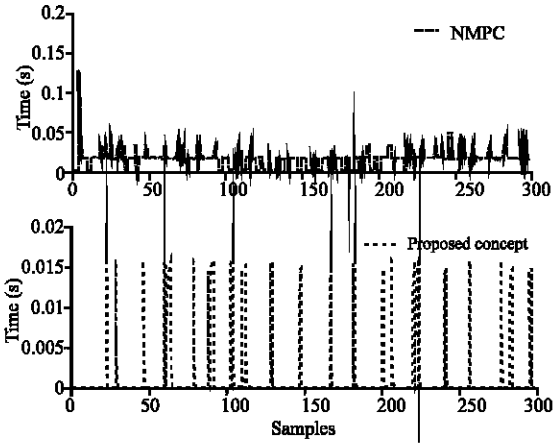


Fig. 10: Computation time comparison

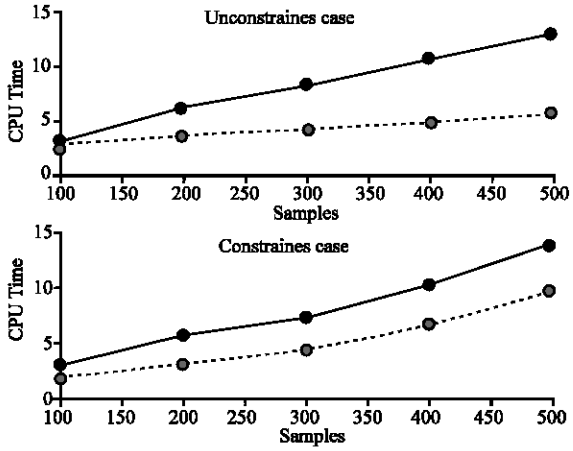


Fig. 11: CPU time comparison (Solid line: NMPC, Dashed line: proposed concept)

control input at each time step for the two approaches is plotted. On average, the NMPC method was about 10 times slower than the novel approach. The simulations results presented in the following have been obtained in the Matlab 6.1 environment running on a 2GHz Pentium IV Intel where the clock precision is 0.016 s.

The results of simulation show that the execution time varies from a meaningful way of a step to another step and these variations are owed to the change of the reference essentially and long prediction horizon. We have reported in Fig. 11 and 12, a comparison study of the CPU-time consuming for both approaches. We remark that the novel concept presents a good performance in time consuming for long prediction horizon and the nonlinear MPC controller is too CPU time consuming as well as a number of samples are increased which cannot be implemented in real time.

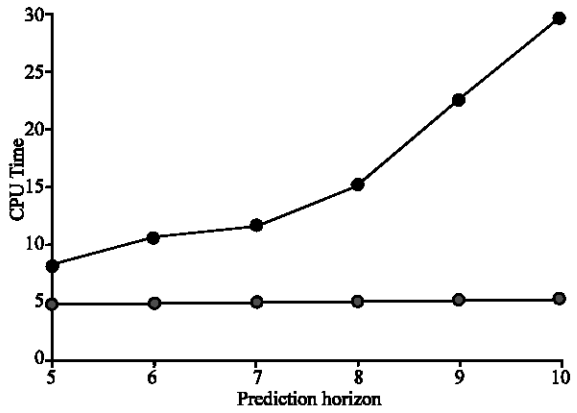


Fig. 12: CPU time comparison as a function of the prediction length (Solid line: NMPC, Dashed line: proposed concept)

However, in practice each MPC problem of the individual agent in multi-agent model predictive control can run in parallel, i.e, at the same time, instead of in serial, i.e., one agent after another. To estimate how much time we can gain by performing the computations in parallel instead of in serial, the following procedure is adopted. By timing the time it takes to perform the computations of each agent and then take the maximum over that as approximation of the time it would require for all agents to solve their problems in parallel, the maximum time have a value of 1.1507 s which is typically small than the serial computing and thus improving a strong points approach that can solve problem computation for fast dynamics systems. Let T (equal to 4.438s for 300 samples) be the time used for serial jobs and t (equal to1.1507s) be the time used to calculate in parallel in C (equal to 4) computers. The efficiency of the parallel computer is defined by T/(t×C). The performance is high as the efficiency approach 1 which is equal to 0.9642 in our study.

## CONCLUSION

This study has dealt with the problem of MPC for nonlinear discrete-time systems when a limitation on control taken into account. The design method proposed here is based on multi-agent model predictive control techniques. It has been shown that this proposed concept is effective for solving the global goal subject to constraints. The computation of the predictive control law is easy and does not need an on line optimization procedure. The contributions of this work show the performance of the multi-agent controller in time computing and enhance the disturbance rejection better than the classical approach. Numerical example have been considered to illustrate and discuss the advantages and limitations due to the complexity of the proposed method.

## REFERENCES

Abonyi, J.A., 2003. Fuzzy Model identification for control. Birkhauser Boston.

Abonyi, J.L., F. Nagy and Szeifert, 1997. Takagi Sugeno Fuzzy Control of Batch Polymerization Reactor. In Proc. 2nd On-line World Conf. Soft Comput. (WSC2), pp: 23-27.

Aswin N. Venkat, 2006. Distributed Model Predictive Control: Theory and applications. Theses. University of Wisconsin-Madison.

Didier, G., 2006. Distributed model predictive control via decomposition-coordination techniques and the use of an augmented lagrangian. IFAC. Workshop on NMPC, pp: 111-116.

IFAC, 2006. Workshop on Nonlinear Model Predictive Control for fast Systems NMPC-FS Grenoble, France.

Kamalasadan, S., 2007. A Novel Multi-Agent Controller for Dynamic Systems based on Supervisory Loop Approach. In Press, Special Issue on Soft Computing on Artificial Intelligence. J. Eng. Lett., 14: 2-EL_14_2_10.

Kanev, S. and M. Vergaegen, 2000. Controller Reconfiguration for Non-Linear Systems. Control Eng. Practice, 8: 1223-1235.

Liu, G., 2001. Nonlinear identification and control, a neural network approach. Advances in industrial control. Springer-London.

Maciejowski, J.M., 2002. Predictive Control with Constraints. Prentice Hall, Harlow, England.

Negenborn, R.R., B. De Schutter, M.A. Wiering and J. Hellendoorn, 2004. Experience-based model predictive control using reinforcement learning. Technical report 04-020. Delf Center for systems and control.

Sanjuan, M., A. Kandel and C.A. Smith, 2006. Design and implementation of a fuzzy supervisor for on-line compensation of nonlinearities: An instability avoidance module. Eng. Application of Artificial Intelligence, 19: 323-333.

Wang, F., 2000. Modeling and control pf processes with Output Dynamic Nonlinearity. Proc. Am. Control Conf. Chicago, pp: 240-243.

Wenzhi, G. and R. Rastko, 2006. Neural Network Control of a Class of Nonlinear Systems With Actuator Saturation. IEEE. Trans. Neural Networks, Vol. 17.