

Designing efficient Model Predictive Control algorithm using easy-to-obtain fuzzy models

Piotr M. Marusak

Abstract—The paper describes the design process of an efficient model predictive control (MPC) algorithm based on fuzzy models. An interesting feature of the proposed approach is that it uses easy-to-obtain fuzzy Takagi–Sugeno (TS) models composed of a few step responses employed as local models; one of these models is used to derive the dynamic matrix, and the second one, being a skillful modification of the first one, to generate the free response. The designed MPC algorithm uses formulation as an efficient quadratic optimization task. Still, it offers control quality compared with the MPC algorithm formulated as a nonlinear optimization task, thanks to the skillful generation of the free response. The efficiency of the proposed approach is tested and demonstrated in the simulated control system of the nonlinear and non-minimum phase process of the chemical reactor with the van de Vusse reaction.

Keywords—prediction; process control; model predictive control; quadratic optimization; fuzzy systems; fuzzy control; nonlinear control, CSTR, van de Vusse reaction

I. INTRODUCTION

OPTIMIZATION is a handy tool for solving engineering problems. One of the tasks that can be done using optimization is the generation of control actions using advanced control algorithms. Applying a fuzzy model during this process can increase the control quality of a control algorithm. However, the resulting fuzzy optimization task can be hard to solve if such a model is used directly. Fortunately, in the MPC algorithms, the advantages of the fuzzy models can be exploited so that the optimization task solved in each iteration of the MPC algorithm can be formulated as the easy-to-solve quadratic optimization problem; this will be detailed in the latter part of the article.

In the MPC algorithms, control signal trajectories are obtained by solving the following optimization problem [1]–[3]:

$$\arg \min_{\Delta \mathbf{u}} \{ (\bar{\mathbf{y}} - \mathbf{y})^T \cdot \boldsymbol{\kappa} \cdot (\bar{\mathbf{y}} - \mathbf{y}) + \Delta \mathbf{u}^T \cdot \boldsymbol{\Lambda} \cdot \Delta \mathbf{u} \} \quad (1)$$

subject to:

$$\Delta \mathbf{u}_{\min} \leq \Delta \mathbf{u} \leq \Delta \mathbf{u}_{\max} \quad , \quad (2)$$

$$\mathbf{u}_{\min} \leq \mathbf{u} \leq \mathbf{u}_{\max} \quad , \quad (3)$$

$$\mathbf{y}_{\min} \leq \mathbf{y} \leq \mathbf{y}_{\max} \quad , \quad (4)$$

where

P.M. Marusak is with Faculty of Electrical Engineering and Communication, Warsaw University of Technology, Warszawa, Poland (e-mail: Piotr.Marusak@pw.edu.pl).

$$\mathbf{y} = \begin{bmatrix} \mathbf{y}^1 \\ \vdots \\ \mathbf{y}^{n_y} \end{bmatrix}, \quad \mathbf{y}^j = \begin{bmatrix} y_{k+1|k}^j \\ \vdots \\ y_{k+p|k}^j \end{bmatrix}, \quad (5)$$

$$\bar{\mathbf{y}} = \begin{bmatrix} \bar{\mathbf{y}}^1 \\ \vdots \\ \bar{\mathbf{y}}^{n_y} \end{bmatrix}, \quad \bar{\mathbf{y}}^j = \begin{bmatrix} \bar{y}_{k+1|k}^j \\ \vdots \\ \bar{y}_{k+p|k}^j \end{bmatrix}, \quad (6)$$

$$\Delta \mathbf{u} = \begin{bmatrix} \Delta \mathbf{u}^1 \\ \vdots \\ \Delta \mathbf{u}^{n_u} \end{bmatrix}, \quad \Delta \mathbf{u}^m = \begin{bmatrix} \Delta u_{k+1|k}^m \\ \vdots \\ \Delta u_{k+s-1|k}^m \end{bmatrix}, \quad (7)$$

$$\mathbf{u} = \begin{bmatrix} \mathbf{u}^1 \\ \vdots \\ \mathbf{u}^{n_u} \end{bmatrix}, \quad \mathbf{u}^m = \begin{bmatrix} u_{k+1|k}^m \\ \vdots \\ u_{k+s-1|k}^m \end{bmatrix}, \quad (8)$$

$$\boldsymbol{\kappa} = [\kappa_1, \dots, \kappa_{n_y}] \cdot \mathbf{I}, \quad \boldsymbol{\kappa}_j = [\kappa_j, \dots, \kappa_j], \quad (9)$$

$$\boldsymbol{\Lambda} = [\boldsymbol{\Lambda}_1, \dots, \boldsymbol{\Lambda}_{n_u}] \cdot \mathbf{I}, \quad \boldsymbol{\Lambda}_m = [\lambda_m, \dots, \lambda_m], \quad (10)$$

$y_{k+i|k}^j$ denotes a value, predicted using a process model the MPC algorithm is based on, of the j^{th} output for the $(k+i)^{\text{th}}$ sampling instant from the prediction horizon, derived at the k^{th} sampling instant, $\bar{y}_{k+i|k}^j$ is an element of the reference trajectory for the j^{th} output and for the $(k+i)^{\text{th}}$ sampling instant from the prediction horizon; if constant reference trajectories on the prediction horizon are used, then

$$\bar{y}_{k+i|k}^j = \bar{y}_k^j \quad (11)$$

where $\bar{y}_k^j$ denotes a setpoint value for the j^{th} output; $\Delta u_{k+i|k}^m$ are the decision variables of the optimization problem being future changes in the m^{th} manipulated variable; $\boldsymbol{\kappa}_j$ contains p elements, and $\kappa_j \geq 0$ denote weighting coefficients for the predicted control errors of the j^{th} output; $\boldsymbol{\Lambda}_m$ contains s elements, and $\lambda_m \geq 0$ denote weighting coefficients for the changes of the m^{th} manipulated variable; p is prediction horizon, s is control horizon, n_y is the number of output variables, n_u is the number of manipulated variables; $\Delta \mathbf{u}_{\min}$, $\Delta \mathbf{u}_{\max}$ denote vectors defining lower and upper bounds of changes of the control signals; $\mathbf{u}_{\min}$, $\mathbf{u}_{\max}$ denote vectors defining lower and upper bounds of values of the control signals; $\mathbf{y}_{\min}$, $\mathbf{y}_{\max}$ denote vectors defining lower and upper bounds of the values of output variables.

The optimization problem (1–4) is formulated and solved in each sampling instant. As a result, the optimal vector of future control action $\Delta \mathbf{u}$ is obtained, from which the $\Delta \mathbf{u}_{k|k}^m$ elements are applied in the control system and then, in the next time step, the optimization problem (1–4) is reformulated and solved.

To formulate the optimization task solved by the MPC algorithm, the prediction $\mathbf{y}$ is needed. The prediction is based on the process model. Assume first that a nonlinear process model is employed:

$$\hat{\mathbf{y}}_k = \mathbf{f}(\mathbf{y}_{k-1}, \mathbf{y}_{k-2}, \dots, \mathbf{y}_{k-n_a}, \mathbf{u}_{k-1}, \mathbf{u}_{k-2}, \dots, \mathbf{u}_{k-n_b}), \quad (12)$$

where $\mathbf{y}_{k-i} = [y_{k-i}^1, \dots, y_{k-i}^{n_y}]^T$ is a vector which contains output values measured at the $(k-i)^{\text{th}}$ sampling instant, $\mathbf{u}_{k-i} = [u_{k-i}^1, \dots, u_{k-i}^{n_u}]^T$ is a vector which contains control values applied at the $(k-i)^{\text{th}}$ sampling instant; denote output values generated by the model in the $(k+i)^{\text{th}}$ sampling instant as $\hat{\mathbf{y}}_{k+i} = [\hat{y}_{k+i}^1, \dots, \hat{y}_{k+i}^{n_y}]^T$, n_a, n_b determine how many past output and control values the model needs. To use the model (12) directly in the optimization problem (1–4), the following formulas, describing the prediction, should be passed to the optimization problem as a set of equality constraints:

$$\mathbf{y}_{k+i|k} = \mathbf{f}(\hat{\mathbf{y}}_{k+i-1}, \dots, \hat{\mathbf{y}}_{k+1}, \mathbf{y}_k, \dots, \mathbf{y}_{k-n_a+i-1}, \mathbf{u}_{k+i-1|k}, \dots, \mathbf{u}_{k|k}, \mathbf{u}_{k-1}, \dots, \mathbf{u}_{k-n_b+i-1}) + \mathbf{d}_k, \quad (13)$$

where $\mathbf{u}_{k+i|k} = [u_{k+i|k}^1, \dots, u_{k+i|k}^{n_u}]^T$ is a vector containing future control values, depending on decision variables from the vector $\Delta \mathbf{u}$ and

$$\mathbf{d}_k = \mathbf{y}_k - \hat{\mathbf{y}}_k, \quad (14)$$

where $\mathbf{y}_k = [y_k^1, \dots, y_k^{n_y}]^T$ is a vector of recently measured output values.

Unfortunately, the optimization problem (1–4) using the prediction (13), based on a nonlinear model, is, in general, non-convex nonlinear optimization problem; see, e.g., [4]–[6]. The MPC algorithm based on such an approach (NMPC algorithm) inherits all its drawbacks.

The nonlinear model (12) can also be a fuzzy model. However, if such a model is used directly to obtain the prediction (13), then the optimization problem (1–4) is still the nonlinear optimization problem with all its drawbacks. On the other hand, we can exploit the advantages of the fuzzy models in a skillful way, which will be detailed in the latter part of the article, but before that, let us recall the MPC algorithms based on linear models; it will be helpful later.

Assume a linear process model is used to obtain the prediction $\mathbf{y}$. In that case, the superposition principle holds, and the vector $\mathbf{y}$ can be decomposed into two parts: free response and forced response [1], [3]:

$$\mathbf{y} = \tilde{\mathbf{y}} + \mathbf{A} \cdot \Delta \mathbf{u}, \quad (15)$$

$$\tilde{\mathbf{y}} = \begin{bmatrix} \tilde{\mathbf{y}}^1 \\ \vdots \\ \tilde{\mathbf{y}}^{n_y} \end{bmatrix}, \quad \tilde{\mathbf{y}}^j = \begin{bmatrix} \tilde{y}_{k+1|k}^j \\ \vdots \\ \tilde{y}_{k+p|k}^j \end{bmatrix}, \quad (16)$$

where $\tilde{\mathbf{y}}$ is the free response of the control plant which describes the influence of the past values of control signals on the process, $\mathbf{A} \cdot \Delta \mathbf{u}$ is the forced response which depends on future changes of the control signals $\Delta \mathbf{u}$; the matrix $\mathbf{A}$ is called the dynamic matrix and has the following form:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}^{11} & \mathbf{A}^{12} & \dots & \mathbf{A}^{1n_u} \\ \mathbf{A}^{21} & \mathbf{A}^{22} & \dots & \mathbf{A}^{2n_u} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}^{n_y 1} & \mathbf{A}^{n_y 2} & \dots & \mathbf{A}^{n_y n_u} \end{bmatrix}, \quad (17)$$

$$\mathbf{A}^{jm} = \begin{bmatrix} a_{1,m}^{j,m} & 0 & \dots & 0 & 0 \\ a_{2,m}^{j,m} & a_{1,m}^{j,m} & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{p,m}^{j,m} & a_{p-1,m}^{j,m} & \dots & a_{p-s+2,m}^{j,m} & a_{p-s+1,m}^{j,m} \end{bmatrix}, \quad (18)$$

where $a_i^{j,m}$ denote step response coefficients of the process, describing the influence of the m^{th} control on the j^{th} output. The prediction (15) is linear with respect to decision variables $\Delta \mathbf{u}$, therefore applying it to the performance function (1) results in its transformation to a function quadratically depending on decision variables $\Delta \mathbf{u}$:

$$(\bar{\mathbf{y}} - \tilde{\mathbf{y}} - \mathbf{A} \cdot \Delta \mathbf{u})^T \cdot \boldsymbol{\kappa} \cdot (\bar{\mathbf{y}} - \tilde{\mathbf{y}} - \mathbf{A} \cdot \Delta \mathbf{u}) + \Delta \mathbf{u}^T \cdot \boldsymbol{\Lambda} \cdot \Delta \mathbf{u}. \quad (19)$$

Furthermore, after using the prediction (15) in the constraints on output values (4), the optimization problem (1–4) becomes an easy to solve, standard quadratic optimization problem, with linear constraints. The MPC algorithm based on such an approach (LMPC algorithm) can quickly generate the control action. However, the control performance offered by such an LMPC algorithm, applied to a nonlinear process, may be poor.

A few approaches were developed to overcome the drawbacks of NMPC and LMPC algorithms. In the so-called fast NMPC algorithms, see, e.g., [7], [8] suboptimal solution is generated faster than in the standard approach. The explicit approach in which most of the calculations are done off-line; see, e.g., [9] is also suboptimal in the nonlinear case, though it is optimal if the linear model is used [10]. However, unfortunately, in the explicit approach, the complexity of the controller grows significantly with the number of constraints taken into consideration.

Linearization of the process model the MPC algorithm is based on, obtained in each time step, results in the prediction linear relative to control changes (decision variables of the optimization problem); see, e.g., [3], [11]–[18]. Such an approach is popular because the optimization problem solved by the MPC algorithm in each time step is then formulated as quadratic (like in LMPC algorithms). The elements of the prediction in this approach (the free and the forced responses) can be obtained in different ways. When the fuzzy TS model is used, its properties can be exploited. In the algorithms based on the fuzzy TS model, described in [11], [15], the free response and the dynamic matrix are obtained using the model

obtained after the linearization. In [12], [16], [17], algorithms in which the free response is calculated using the nonlinear model (not the linearized one) are described. The algorithms in which the free response and the dynamic matrix are obtained using the nonlinear models are detailed in [18].

The approach proposed in the article, allows relatively easy design the MPC algorithm based on easy-to-obtain fuzzy models composed of a few step responses. It is sufficient to collect these responses and add the premises of the fuzzy rules, using, e.g., expert knowledge supported by the analysis of the steady-state characteristics; as a result, the fuzzy models are obtained. Note that in such a way, the existing control system, with the classical Dynamic Matrix Control (DMC) algorithm based on a linear model in the form of the step response, can be relatively easily updated to offer better control quality.

Then, the model with the normalized step responses is used to generate the dynamic matrix, and the model with the denormalized step responses is used to generate the free response. Thanks to such an approach, the obtained results can be much better than those generated by the algorithm proposed in [15], close to the ones obtained with the NMPC algorithm, like in the case of the algorithm from [16]. However, this time, the model used by the algorithm to obtain the free response can be obtained straightforwardly, and its usage is relatively simple. To sum up, the models and the prediction are obtained relatively easily and the prediction is still linear with respect to control changes (decision variables of the optimization problem). Therefore, the algorithm employing the proposed approach merges the LMPC algorithms' computational simplicity with the NMPC algorithms' control performance.

The next Section details the formulation of the fuzzy MPC (FMPC) algorithms based on easy-to-obtain fuzzy models. In Section 3, the operation of the FMPC algorithms is tested in a simulation example of the control system of a nonlinear chemical reactor with the van de Vusse reaction. Conclusions are presented in the last Section.

II. EFFICIENT FUZZY MPC ALGORITHM

A. Dynamic Matrix Derivation

The dynamic matrix needed to obtain the forced response can be calculated using an easy-to-obtain fuzzy TS model, employed also in, e.g., [15], [17], [18], which has local models in the form of normalized step responses:

Rule f :

$$\begin{aligned} &\text{if } y_{k-1}^{j_y} \text{ is } B_1^{f,j_y} \text{ and } \dots \text{ and } y_{k-n_a}^{j_y} \text{ is } B_{n_a}^{f,j_y} \text{ and} \\ &\quad u_{k-1}^{j_u} \text{ is } C_1^{f,j_u} \text{ and } \dots \text{ and } u_{k-n_b}^{j_u} \text{ is } C_{n_b}^{f,j_u} \\ &\text{then } \hat{y}_k^{j,f} = \sum_{m=1}^{n_u} \left[\sum_{n=1}^{p_d-1} a_n^{j,m,f} \cdot \Delta u_{k-n}^m + a_{p_d}^{j,m,f} \cdot u_{k-p_d}^m \right], \end{aligned} \quad (20)$$

where $y_k^{j_y}$ is the value of j_y^{th} output variable ($j_y = 1, \dots, n_y$) at the k^{th} sampling instant, $u_k^{j_u}$ is the value of j_u^{th} manipulated variable ($j_u = 1, \dots, n_u$) at the k^{th} sampling instant, $B_1^{f,j_y}, \dots, B_{n_a}^{f,j_y}, C_1^{f,j_u}, \dots, C_{n_b}^{f,j_u}$ are fuzzy sets, $a_n^{j,m,f}$ are the coefficients of step responses in the f^{th} local model,

$f = 1, \dots, l$, l is number of fuzzy rules. The output values of the model (20) can be calculated using the following formula:

$$\hat{y}_k^j = \sum_{m=1}^{n_u} \left[\sum_{n=1}^{p_d-1} \tilde{a}_n^{j,m} \cdot \Delta u_{k-n}^m + \tilde{a}_{p_d}^{j,m} \cdot u_{k-p_d}^m \right], \quad (21)$$

where $\tilde{a}_n^{j,m} = \sum_{f=1}^l \tilde{w}_f \cdot a_n^{j,m,f}$ and $\tilde{w}_f$ are the normalized activation levels of the fuzzy rules calculated using fuzzy reasoning, see e.g., [19], [20]. Note that one can consider the model (21) as the step response describing the behavior of the control plant near the current operating point.

The model (20) is then used at each sampling instant of the FMPC algorithm to obtain a new, updated dynamic matrix as described, e.g., in [17]:

$$A_k = \begin{bmatrix} A_k^{11} & A_k^{12} & \dots & A_k^{1n_u} \\ A_k^{21} & A_k^{22} & \dots & A_k^{2n_u} \\ \vdots & \vdots & \ddots & \vdots \\ A_k^{n_y1} & A_k^{n_y2} & \dots & A_k^{n_y n_u} \end{bmatrix}, \quad (22)$$

$$A_k^{j,m} = \begin{bmatrix} \tilde{a}_1^{j,m} & 0 & \dots & 0 & 0 \\ \tilde{a}_2^{j,m} & \tilde{a}_1^{j,m} & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \tilde{a}_p^{j,m} & \tilde{a}_{p-1}^{j,m} & \dots & \tilde{a}_{p-s+2}^{j,m} & \tilde{a}_{p-s+1}^{j,m} \end{bmatrix}. \quad (23)$$

B. Obtaining Free Response

In the proposed approach, the free response can be generated using the appropriately modified model (20). The local models in the model (20) were normalized; now, this normalization process should be reversed. Therefore, the model will now have the following form:

Rule f :

$$\begin{aligned} &\text{if } y_{k-1}^{j_y} \text{ is } B_1^{f,j_y} \text{ and } \dots \text{ and } y_{k-n_a}^{j_y} \text{ is } B_{n_a}^{f,j_y} \text{ and} \\ &\quad u_{k-1}^{j_u} \text{ is } C_1^{f,j_u} \text{ and } \dots \text{ and } u_{k-n_b}^{j_u} \text{ is } C_{n_b}^{f,j_u} \\ &\text{then } \hat{y}_k^{j,f} = y_0^{j,f} + \sum_{m=1}^{n_u} \left[\sum_{n=1}^{p_d-1} a_n^{j,m,f} \cdot \Delta u_{k-n}^m \right. \\ &\quad \left. + a_{p_d}^{j,m,f} \cdot (u_{k-p_d}^m - u_0^{m,f}) \right], \end{aligned} \quad (24)$$

where $y_0^{j,f}$ is the value of j^{th} output, in the f^{th} local model from which the step response generation started and $u_0^{m,f}$ is the value of m^{th} input, in the f^{th} local model from which the increase of the input was done to generate the step response. The outputs of the model (24) are thus as follows:

$$\hat{y}_k^j = \tilde{y}_0^j + \sum_{m=1}^{n_u} \left[\sum_{n=1}^{p_d-1} \tilde{a}_n^{j,m} \cdot \Delta u_{k-n}^m + \tilde{a}_{p_d}^{j,m} \cdot (u_{k-p_d}^m - \tilde{u}_0^m) \right], \quad (25)$$

where $\tilde{y}_0^j = \sum_{f=1}^l \tilde{w}_f \cdot y_0^{j,f}$ and $\tilde{u}_0^m = \sum_{f=1}^l \tilde{w}_f \cdot u_0^{m,f}$. The output values calculated using (25), are then used to obtain values of the d_k^j errors, including information about model uncertainty:

$$d_k^j = y_k - \hat{y}_k^j. \quad (26)$$

Once d_k^j errors are calculated, the free response can be derived iteratively. Namely, in the first step from the prediction

horizon, the following model (the model (24) with discrete time moved one step further) is used:

Rule f : (27)

$$\begin{aligned} &\text{if } y_k^{j_y} \text{ is } B_1^{f,j_y} \text{ and } \dots \text{ and } y_{k-n_a+1}^{j_y} \text{ is } B_{n_a}^{f,j_y} \text{ and} \\ &\quad u_k^{j_u} \text{ is } C_1^{f,j_u} \text{ and } \dots \text{ and } u_{k-n_b+1}^{j_u} \text{ is } C_{n_b}^{f,j_u} \\ &\text{then } \hat{y}_{k+1}^{j,f} = y_0^{j,f} + \sum_{m=1}^{n_u} \left[\sum_{n=1}^{p_d-1} a_n^{j,m,f} \cdot \Delta u_{k-n+1}^m \right. \\ &\quad \left. + a_{p_d}^{j,m,f} \cdot (u_{k-p_d+1}^m - u_0^{m,f}) \right]. \end{aligned}$$

However, $u_k^{j_u}$ are not known yet, but as the free response is generated here, these values are replaced by $u_{k-1}^{j_u}$. Therefore, the premises used to obtain the activation levels of the fuzzy rules will be as follows:

Rule f : (28)

$$\begin{aligned} &\text{if } y_k^{j_y} \text{ is } B_1^{f,j_y} \text{ and } \dots \text{ and } y_{k-n_a+1}^{j_y} \text{ is } B_{n_a}^{f,j_y} \text{ and} \\ &\quad u_{k-1}^{j_u} \text{ is } C_1^{f,j_u} \text{ and } \dots \text{ and } u_{k-n_b+1}^{j_u} \text{ is } C_{n_b}^{f,j_u}. \end{aligned}$$

As a result, the normalized activation levels of the fuzzy rules $\tilde{w}_{k+1|k}^f$ are obtained for the $(k+1)^{\text{st}}$ time step from the prediction horizon, at the k^{th} sampling instant and the outputs of the model are now calculated using the formula:

$$\begin{aligned} \hat{y}_{k+1|k}^j = \tilde{y}_{0,k+1}^j + \sum_{m=1}^{n_u} \left[\sum_{n=1}^{p_d-1} \tilde{a}_{n,k+1}^{j,m} \cdot \Delta u_{k-n}^m \right. \\ \left. + \tilde{a}_{p_d,k+1}^{j,m} \cdot (u_{k-p_d}^m - \tilde{u}_{0,k+1}^m) \right], \end{aligned} \quad (29)$$

where $\tilde{a}_{n,k+1}^{j,m} = \sum_{f=1}^l \tilde{w}_{k+1|k}^f \cdot a_n^{j,m,f}$, $\tilde{y}_{0,k+1}^j = \sum_{f=1}^l \tilde{w}_{k+1|k}^f \cdot y_0^{j,f}$, $\tilde{u}_{0,k+1}^m = \sum_{f=1}^l \tilde{w}_{k+1|k}^f \cdot u_0^{m,f}$.

In the second time step from the prediction horizon, the fuzzy model has the following form:

Rule f : (30)

$$\begin{aligned} &\text{if } y_{k+1}^{j_y} \text{ is } B_1^{f,j_y} \text{ and } \dots \text{ and } y_{k-n_a+2}^{j_y} \text{ is } B_{n_a}^{f,j_y} \text{ and} \\ &\quad u_{k+1}^{j_u} \text{ is } C_1^{f,j_u} \text{ and } \dots \text{ and } u_{k-n_b+2}^{j_u} \text{ is } C_{n_b}^{f,j_u} \\ &\text{then } \hat{y}_{k+2}^{j,f} = y_0^{j,f} + \sum_{m=1}^{n_u} \left[\sum_{n=1}^{p_d-1} a_n^{j,m,f} \cdot \Delta u_{k-n+2}^m \right. \\ &\quad \left. + a_{p_d}^{j,m,f} \cdot (u_{k-p_d+2}^m - u_0^{m,f}) \right]. \end{aligned}$$

This time, $u_k^{j_u}$, $u_{k+1}^{j_u}$ and $y_{k+1}^{j_y}$ are not known yet. Therefore, in order to obtain the free response, $u_k^{j_u}$ and $u_{k+1}^{j_u}$ are replaced with $u_{k-1}^{j_u}$. Whereas the $y_{k+1}^{j_y}$ values are replaced with already calculated outputs of the model, namely $\hat{y}_{k+1|k}^{j_y}$. Therefore, the premises used to obtain the activation levels in the $(k+2)^{\text{nd}}$ time step from the prediction horizon have the following form:

Rule f : (31)

$$\begin{aligned} &\text{if } \hat{y}_{k+1|k}^{j_y} \text{ is } B_2^{f,j_y} \text{ and } \dots \text{ and } y_{k-n_a+2}^{j_y} \text{ is } B_{n_a}^{f,j_y} \text{ and} \\ &\quad u_{k-1}^{j_u} \text{ is } C_1^{f,j_u} \text{ and } \dots \text{ and } u_{k-n_b+2}^{j_u} \text{ is } C_{n_b}^{f,j_u}. \end{aligned}$$

The normalized activation levels $\tilde{w}_{k+2|k}^f$ are obtained for the $(k+2)^{\text{nd}}$ time step from the prediction horizon and the outputs of the model are now calculated using the formula:

$$\begin{aligned} \hat{y}_{k+2|k}^j = \tilde{y}_{0,k+2}^j + \sum_{m=1}^{n_u} \left[\sum_{n=1}^{p_d-1} \tilde{a}_{n,k+2}^{j,m} \cdot \Delta u_{k-n}^m \right. \\ \left. + \tilde{a}_{p_d,k+2}^{j,m} \cdot (u_{k-p_d}^m - \tilde{u}_{0,k+2}^m) \right], \end{aligned} \quad (32)$$

where $\tilde{a}_{n,k+2}^{j,m} = \sum_{f=1}^l \tilde{w}_{k+2|k}^f \cdot a_n^{j,m,f}$, $\tilde{y}_{0,k+2}^j = \sum_{f=1}^l \tilde{w}_{k+2|k}^f \cdot y_0^{j,f}$, $\tilde{u}_{0,k+2}^m = \sum_{f=1}^l \tilde{w}_{k+2|k}^f \cdot u_0^{m,f}$.

The presented procedure is repeated iteratively in the next time steps from the prediction horizon (the model (24) is transformed appropriately for subsequent time steps in the prediction horizon), resulting in obtaining the normalized activation levels of the fuzzy rules $\tilde{w}_{k+i|k}^f$ in each, $(k+i)^{\text{th}}$ time step from the prediction horizon and calculating model outputs in the next time steps:

$$\begin{aligned} \hat{y}_{k+i|k}^j = \tilde{y}_{0,k+i}^j + \sum_{m=1}^{n_u} \left[\sum_{n=1}^{p_d-1} \tilde{a}_{n,k+i}^{j,m} \cdot \Delta u_{k-n}^m \right. \\ \left. + \tilde{a}_{p_d,k+i}^{j,m} \cdot (u_{k-p_d}^m - \tilde{u}_{0,k+i}^m) \right], \end{aligned} \quad (33)$$

where $\tilde{a}_{n,k+i}^{j,m} = \sum_{f=1}^l \tilde{w}_{k+i|k}^f \cdot a_n^{j,m,f}$, $\tilde{y}_{0,k+i}^j = \sum_{f=1}^l \tilde{w}_{k+i|k}^f \cdot y_0^{j,f}$, $\tilde{u}_{0,k+i}^m = \sum_{f=1}^l \tilde{w}_{k+i|k}^f \cdot u_0^{m,f}$.

Then, the free response elements are calculated using the formula:

$$\tilde{y}_{k+i|k}^j = \hat{y}_{k+i|k}^j + d_k^j. \quad (34)$$

C. Optimization Problem in the Numerical and Analytical Versions of the Algorithm

After using the free response obtained above and the dynamic matrix (22), one obtains the following prediction:

$$\mathbf{y} = \tilde{\mathbf{y}} + \mathbf{A}_k \cdot \Delta \mathbf{u}. \quad (35)$$

This prediction is then used in the optimization task (1–4), solved by the control algorithm in each time step:

$$\begin{aligned} \arg \min_{\Delta \mathbf{u}} \{ (\bar{\mathbf{y}} - \tilde{\mathbf{y}} - \mathbf{A}_k \cdot \Delta \mathbf{u})^T \cdot \boldsymbol{\kappa} \cdot (\bar{\mathbf{y}} - \tilde{\mathbf{y}} - \mathbf{A}_k \cdot \Delta \mathbf{u}) \\ + \Delta \mathbf{u}^T \cdot \boldsymbol{\Lambda} \cdot \Delta \mathbf{u} \} \end{aligned} \quad (36)$$

subject to:

$$\Delta \mathbf{u}_{\min} \leq \Delta \mathbf{u} \leq \Delta \mathbf{u}_{\max}, \quad (37)$$

$$\mathbf{u}_{\min} \leq \mathbf{u} \leq \mathbf{u}_{\max}, \quad (38)$$

$$\mathbf{y}_{\min} \leq \tilde{\mathbf{y}} + \mathbf{A}_k \cdot \Delta \mathbf{u} \leq \mathbf{y}_{\max}. \quad (39)$$

The performance function in (36) depends quadratically on decision variables $\Delta \mathbf{u}$, and all constraints depend linearly on decision variables. Thus, a standard, easy-to-solve linear-quadratic optimization problem is obtained.

Remark

If the optimization problem with performance function from (36), is solved without constraints, then it has the following solution given by the analytical formula:

$$\Delta \mathbf{u} = \left(\mathbf{A}_k^T \cdot \boldsymbol{\kappa} \cdot \mathbf{A}_k + \boldsymbol{\Lambda} \right)^{-1} \cdot \left(\mathbf{A}_k^T \cdot \boldsymbol{\kappa} \cdot (\bar{\mathbf{y}} - \tilde{\mathbf{y}}) \right). \quad (40)$$

D. Including Disturbance Measurement in the Prediction

It is assumed that the fuzzy TS model (24) is expanded with the step response model of disturbance influence on the process, i.e.:

Rule f : (41)

$$\begin{aligned} &\text{if } y_{k-1}^{j_y} \text{ is } B_1^{f,j_y} \text{ and } \dots \text{ and } y_{k-n_a}^{j_y} \text{ is } B_{n_a}^{f,j_y} \text{ and} \\ &\quad u_{k-1}^{j_u} \text{ is } C_1^{f,j_u} \text{ and } \dots \text{ and } u_{k-n_b}^{j_u} \text{ is } C_{n_b}^{f,j_u} \\ &\text{then } \hat{y}_k^{j,f} = y_0^{j,f} + \sum_{m=1}^{n_u} \left[\sum_{n=1}^{p_d-1} a_n^{j,m,f} \cdot \Delta u_{k-n}^m \right. \\ &\quad \left. + a_{p_d}^{j,m,f} \cdot (u_{k-p_d}^m - u_0^{m,f}) \right] \\ &\quad + \sum_{n=1}^{p_d-1} a_n^{j,d,f} \cdot \Delta u_{k-n}^d + a_{p_d}^{j,d,f} \cdot (u_{k-p_d}^d - u_0^{d,f}), \end{aligned}$$

where Δu_{k-n}^d is the increment of the disturbance variable at the $(k-n)^{\text{th}}$ time step, $u_{k-p_d}^d$ is the value of the disturbance variable at the $(k-p_d)^{\text{th}}$ time step, $a_n^{j,d,f}$ are the coefficients of disturbance step response in the f^{th} local model, $u_0^{d,f}$ is the value of disturbance input, in the f^{th} local model from which the increase of the disturbance input was done to generate the step response. The model (41) is then used to generate the free response, as described in Section II-B. Then the formula (33) becomes:

$$\begin{aligned} \hat{y}_{k+i|k}^j &= \hat{y}_{0,k+i}^j + \sum_{m=1}^{n_u} \left[\sum_{n=1}^{p_d-1} \tilde{a}_{n,k+i}^{j,m} \cdot \Delta u_{k-n}^m \right. \\ &\quad \left. + \tilde{a}_{p_d,k+i}^{j,m} \cdot (u_{k-p_d}^m - \tilde{u}_{0,k+i}^m) \right] \\ &\quad + \sum_{n=1}^{p_d-1} \tilde{a}_{n,k+i}^{j,d} \cdot \Delta u_{k-n}^d + \tilde{a}_{p_d,k+i}^{j,d} \cdot (u_{k-p_d}^d - \tilde{u}_{0,k+i}^d), \end{aligned} \quad (42)$$

where $\tilde{a}_{n,k+i}^{j,d} = \sum_{f=1}^l \tilde{w}_{k+i|k}^f \cdot a_n^{j,d,f}$ and $\tilde{u}_{0,k+i}^d = \sum_{f=1}^l \tilde{w}_{k+i|k}^f \cdot u_0^{d,f}$.

III. SIMULATION EXPERIMENTS

A. Control Plant

The simulation experiments were done in the control system of a nonlinear, isothermal CSTR in which the van de Vusse reaction takes place; it is a popular benchmark, with considerable nonlinearity and difficult dynamics, willingly used to test the control algorithms; see e.g. [17], [18], [21]–[23]. The process model of the reactor is as follows [21]:

$$\begin{aligned} \frac{dC_A}{dt} &= -k_1 \cdot C_A - k_3 \cdot C_A^2 + \frac{F}{V} (C_{Af} - C_A), \\ \frac{dC_B}{dt} &= k_1 \cdot C_A - k_2 \cdot C_B - \frac{F}{V} C_B, \end{aligned} \quad (43)$$

where C_A , C_B are the concentrations of components A and B, respectively, F is the flow rate through the reactor, $V = 1$ l is the constant volume in which the reaction takes place, C_{Af} is the main disturbance – the concentration of component A in the inlet flow stream ($C_{Af} = 10$ mol/l if not declared otherwise). The values of reaction parameters are: $k_1 = 50$ 1/h, $k_2 = 100$ 1/h, $k_3 = 10$ 1/(h · mol). The output variable is the concentration C_B of substance B, and the manipulated variable is the inlet flow rate F . The control plant is nonlinear and it

has the inverse response. Therefore, it is natural to use an MPC algorithm to control it.

The fuzzy models used in the researched algorithm are composed of three step responses obtained near the following operating points [17], [18]:

R1: $y_0^{1,1} = C_{B0,1} = 0.91$ mol/l, $u_0^{1,1} = F_{0,1} = 20$ l/h, $u_0^{d,1} = C_{Af0,1} = 10$ mol/l;

R2: $y_0^{1,2} = C_{B0,2} = 1.12$ mol/l, $u_0^{1,2} = F_{0,2} = 34.3$ l/h, $u_0^{d,2} = C_{Af0,2} = 10$ mol/l;

R3: $y_0^{1,3} = C_{B0,3} = 1.22$ mol/l, $u_0^{1,3} = F_{0,3} = 50$ l/h, $u_0^{d,3} = C_{Af0,3} = 10$ mol/l.

The assumed membership functions are as follows ($\tilde{C} = C_{B,k-1}$):

$$\mu_{R1}(\tilde{C}) = \begin{cases} 1 & \text{if } \tilde{C} \leq 0.91, \\ \frac{1.12-\tilde{C}}{0.21} & \text{if } \tilde{C} \in (0.91, 1.12), \\ 0 & \text{if } \tilde{C} \geq 1.12. \end{cases} \quad (44)$$

$$\mu_{R2}(\tilde{C}) = \begin{cases} 0 & \text{if } \tilde{C} \leq 0.91 \text{ or } \tilde{C} \geq 1.22, \\ \frac{\tilde{C}-0.91}{0.21} & \text{if } \tilde{C} \in (0.91, 1.12), \\ \frac{1.22-\tilde{C}}{0.1} & \text{if } \tilde{C} \in (1.12, 1.22). \end{cases} \quad (45)$$

$$\mu_{R3}(\tilde{C}) = \begin{cases} 0 & \text{if } \tilde{C} \leq 1.12, \\ \frac{\tilde{C}-1.12}{0.1} & \text{if } \tilde{C} \in (1.12, 1.22), \\ 1 & \text{if } \tilde{C} \geq 1.22. \end{cases} \quad (46)$$

B. Responses to Setpoint Changes

For the considered control plant, three MPC algorithms were designed:

- NMPC one based on the nonlinear model and nonlinear optimization,
- LMPC one based on a linear model in the form of the step response (of DMC type) obtained near the R2 point and the
- FMPC algorithm using two fuzzy models. The first one – with local models in the form of normalized step responses; the dynamic matrix is generated using this model. The second one is the fuzzy model with denormalized step responses; it is used to generate the free response.

During the experiments, the operation of the FMPC algorithm was compared with the operation of the NMPC and LMPC algorithms. The simulation experiments were done in Matlab. The sampling time equal to $T_s = 3.6$ s was assumed; the initial values of tuning parameters were as follows: prediction horizon $p = 70$, control horizon $s = 35$, and weighting coefficient $\lambda = 0.001$.

Responses of the control system to changes in the setpoint to $\bar{C}_{B1} = 1$ mol/l and to $\bar{C}_{B2} = 1.25$ mol/l are shown in Figure 1. In the case when the setpoint has been changed to $\bar{C}_{B1} = 1$ mol/l, responses obtained in the control system with the FMPC algorithm (blue lines in Figure 1) are very close to those obtained with NMPC algorithm (red lines in Figure 1). Both algorithms generate almost no overshoot, which occurs in the control system with the LMPC algorithm (green lines in Figure 1); as a result, the settling time is much shorter in the control systems with the NMPC and FMPC algorithms, both algorithms based on nonlinear models work almost the same. If the setpoint has been changed to $\bar{C}_{B2} = 1.25$ mol/l,

the response obtained with the FMPC algorithm is faster than the one generated with the NMPC algorithm; the LMPC algorithm is much slower than its nonlinear counterparts. All the algorithms generate no overshoot.

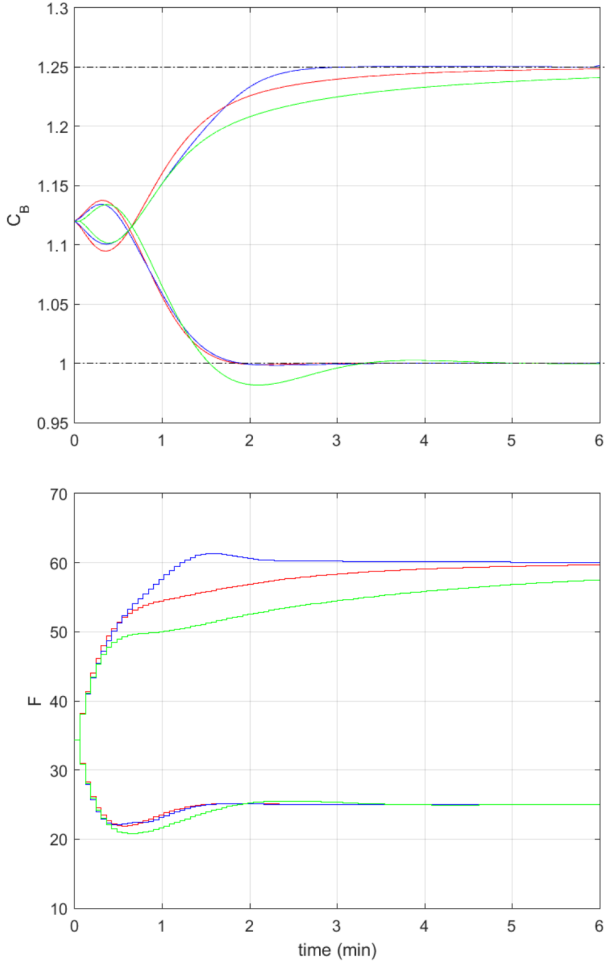


Fig. 1. Responses of the control system to changes in the setpoint to $\bar{C}_{B1} = 1$ mol/l and $\bar{C}_{B2} = 1.25$ mol/l; $\lambda = 0.001$; NMPC—red lines, LMPC—green lines, FMPC—blue lines

To check if the tested algorithms can work faster, experiments with $\lambda = 0.0001$ were also done; see Figure 2. The algorithms work faster, with more aggressive control action. However, the relations between the operation of the FMPC, NMPC, and LMPC algorithms remain the same as in the previous case ($\lambda = 0.001$; Figure 1). In the case of setpoint change to $\bar{C}_{B2} = 1$ mol/l, the FMPC algorithm offers practically the same control performance as the NMPC algorithm; for setpoint change to $\bar{C}_{B2} = 1.25$ mol/l, the FMPC algorithm works faster.

The control horizon decides on the number of decision variables in the optimization problem solved by the MPC algorithm to generate the control action; thus, shortening the control horizon causes the optimization problem to be solved faster. Therefore, in the next experiments, the influence of shortening of the control horizon on the control performance offered by the FMPC algorithm has been checked (Figure 3).

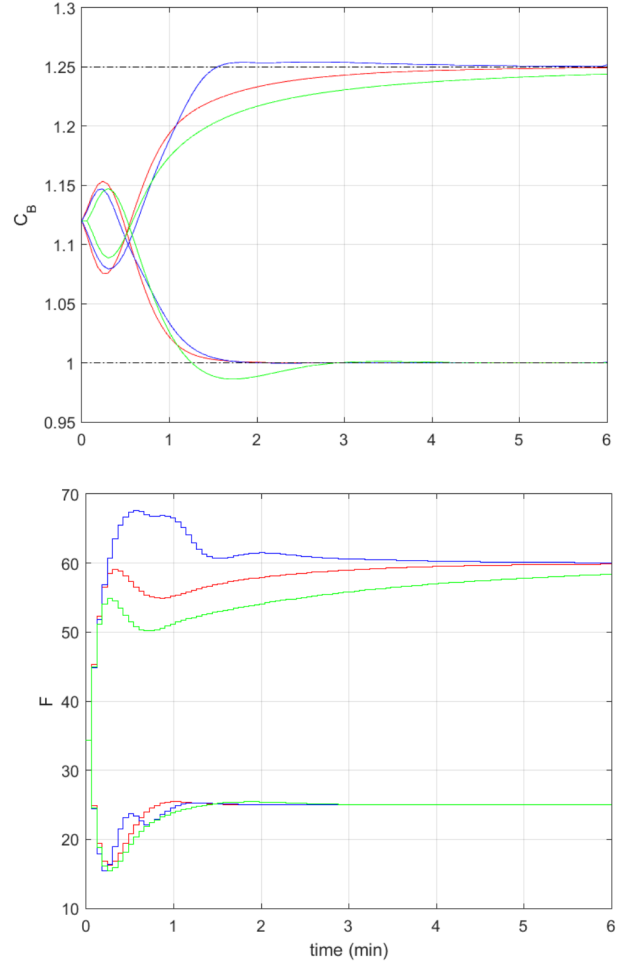


Fig. 2. Responses of the control system to changes in the setpoint to $\bar{C}_{B1} = 1$ mol/l and $\bar{C}_{B2} = 1.25$ mol/l; $\lambda = 0.0001$; NMPC—red lines, LMPC—green lines, FMPC—blue lines

For many values of the control horizon (between $s = 35$ (black lines in Figure 3) and for $s = 10$ (dashed green lines in Figure 3)), the obtained responses remain unchanged, and the control horizon can be shortened significantly. The changes in the responses start to occur for smaller values of the control horizon. For $s = 5$ (magenta lines in Figure 3), the response to the setpoint change to $\bar{C}_{B1} = 1.25$ mol/l remains unchanged. Unfortunately, the response to the setpoint change to $\bar{C}_{B1} = 1$ mol/l becomes slower. The observed tendencies remain the same when the control horizon is further decreased to $s = 3$ (orange lines in Figure 3), $s = 2$ (cyan lines in Figure 3) and $s = 1$ (blue lines in Figure 3). For the control horizon $s = 1$, the response to the setpoint change to $\bar{C}_{B1} = 1.25$ mol/l is the fastest one. Unfortunately, it is achieved at the cost of the slowest responses to the setpoint change to $\bar{C}_{B1} = 1$ mol/l.

To sum up, decreasing the control horizon can be useful. It may simplify the computations done by the algorithm at each sampling instant. However, decreasing the control horizon should be done carefully to achieve a demanded balance between numerical complexity and control performance.

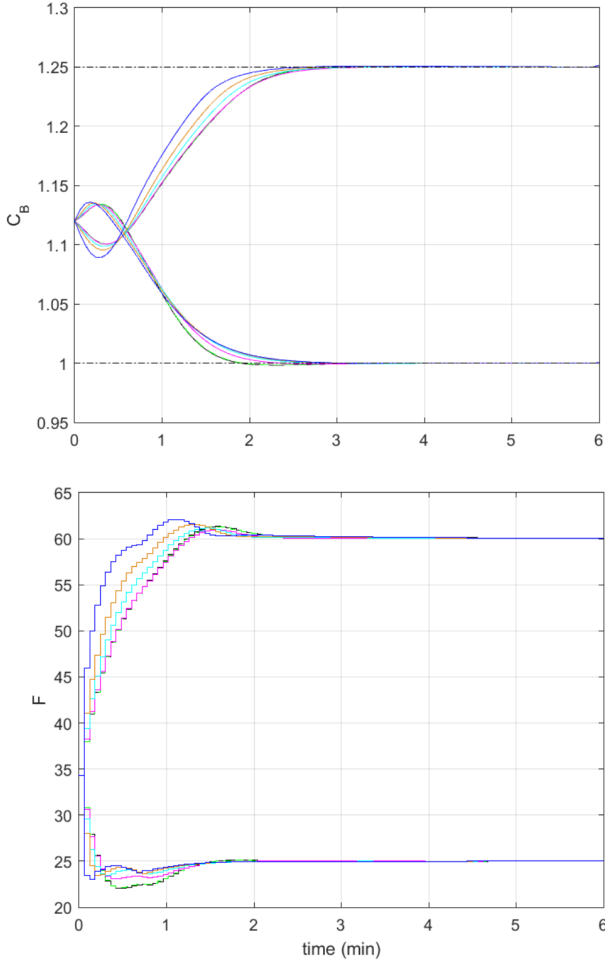


Fig. 3. Responses of the control system with FMPC algorithm and $\lambda = 0.001$ to changes in the setpoint to $\bar{C}_{B1} = 1$ mol/l and $\bar{C}_{B2} = 1.25$ mol/l; $s = 35$ —black lines, $s = 10$ —green dashed lines, $s = 5$ —magenta lines, $s = 3$ —cyan lines, $s = 2$ —orange lines, $s = 1$ —blue lines

C. Disturbance Compensation

Next, responses of the control system to the change of the disturbance by 10% in the 6th minute of the experiment has been obtained (Figure 4). The FMPC algorithm is slower than the NMPC one; both algorithms compensate the disturbance without overshoot. The LMPC algorithm generates the overshoot. The NMPC algorithm compensates the disturbance best. Fortunately, the proposed FMPC algorithm can be supplemented with the mechanism of including disturbance measurement in the prediction, as described in Section II-D; the appropriate experiment has been done and the responses depicted in Figure 5 has been obtained. The FMPC algorithm better compensates the disturbance after including the mechanism (red lines in Figure 5) than in the case when disturbance measurement is not exploited (blue lines in Figure 5). The maximal control error is significantly reduced (by around 70%).

IV. CONCLUSIONS

The proposed method of designing MPC controllers is based on the easy-to-obtain fuzzy models. The model with denor-

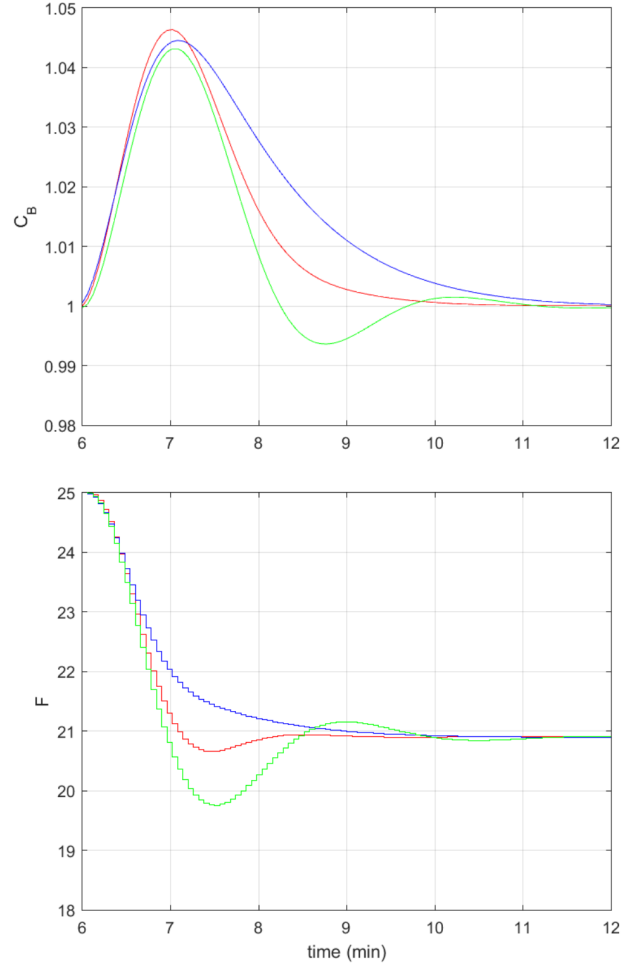


Fig. 4. Responses of the control system to the change of the disturbance by 10% in the 6th minute of the experiment ($\lambda = 0.001$); NMPC—red lines, LMPC—green lines, FMPC—blue lines

malized step responses allows us to obtain the free response, which well resembles the behavior of the nonlinear control plant. Thanks to such an approach, a very good prediction is obtained; however, it is linear with respect to control changes (decision variables in the optimization problem). Therefore, the optimization problem solved in each iteration by the FMPC algorithm is efficient, easy to solve linear-quadratic problem. Despite that, the FMPC algorithm offers a control performance that is very close to that of the NMPC algorithms.

The FMPC algorithm and the method of its design can be useful in the following cases:

- 1) It can be used to relatively easy update the existing control system with the classical DMC algorithm based on a linear model to offer better control quality. Namely, it is just sufficient to obtain a few step responses near a few operating points, extending the process model with them, formulating a fuzzy model, which then is used to design the FMPC algorithm offering better control performance than the classical MPC algorithm.
- 2) It can be used to design a controller for a highly nonlinear process with difficult dynamics.

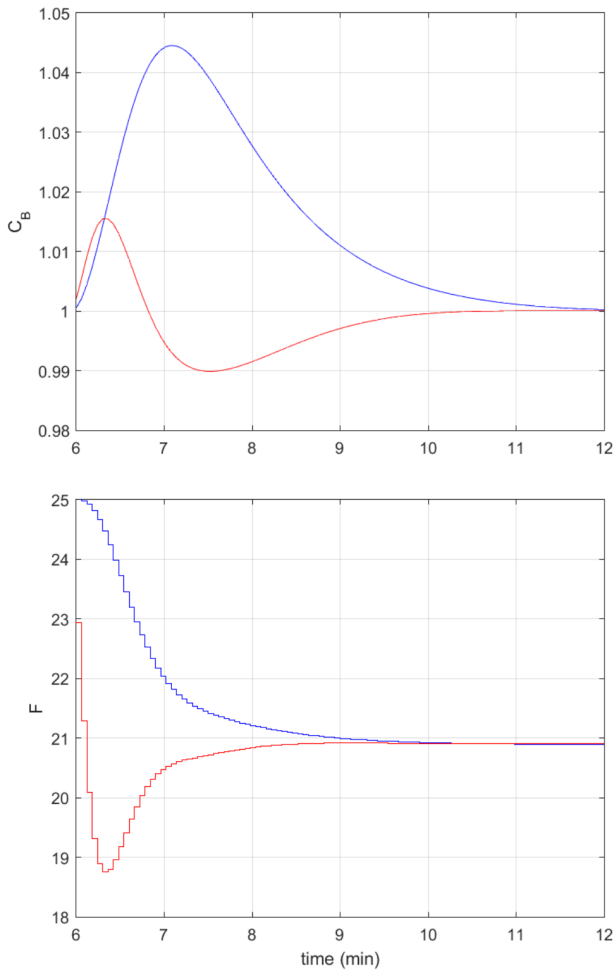


Fig. 5. Responses of the control system with FMPC algorithm and $\lambda = 0.001$ to the change of the disturbance by 10% in the 6th minute of the experiment; mechanism of disturbance measurement utilization: not employed—blue lines, employed—red lines

- 3) It can also be used as a support for the NMPC algorithm. It can generate the (suboptimal) solution relatively fast, which can be a good starting point for the NMPC algorithm. Now, the NMPC algorithm can improve the solution during the current time step, but if it does not succeed, one has a pretty good solution found by the FMPC algorithm.

REFERENCES

- [1] E.F. Camacho and C. Bordons, "Model Predictive Control", Springer, London, UK, 1999. <https://doi.org/10.1007/978-1-4471-3398-8>
- [2] J.A. Rossiter, "Model-Based Predictive Control: A Practical Approach", CRC Press, Boca Raton, USA, 2003. <https://doi.org/10.1201/9781315272610>
- [3] P. Tatjewski, "Advanced Control of Industrial Processes; Structures and Algorithms", Springer, London, UK, 2007. <https://doi.org/10.1007/978-1-84628-635-3>
- [4] M. Abdelaal; S. Schön, "Predictive Path Following and Collision Avoidance of Autonomous Connected Vehicles", *Algorithms* 13(3), 52 (2020). <https://doi.org/10.3390/a13030052>
- [5] H. Chen, F. Allgöwer, "A Quasi-Infinite Horizon Nonlinear Model Predictive Control Scheme with Guaranteed Stability", *Automatica* 34(10), 1205–1217 (1998). [https://doi.org/10.1016/S0005-1098\(98\)00073-9](https://doi.org/10.1016/S0005-1098(98)00073-9)
- [6] P. Tatjewski, "Offset-free nonlinear model predictive control with state-space process models", *Archives of Control Sciences* 27(4), 595–615 (2017). <https://doi.org/10.1515/acsc-2017-0035>
- [7] A. Schäfer, P. Kühn, M. Diehl, J. Schlöder, and H.G. Bock, "Fast reduced multiple shooting methods for nonlinear model predictive control", *Chem. Eng. Process.* 46(11), 1200–1214 (2007). <https://doi.org/10.1016/j.cep.2006.06.024>
- [8] V.M. Zavala, C.D. Laird, and L.T. Biegler, "A fast moving horizon estimation algorithm based on nonlinear programming sensitivity", *J. Process Control* 18(9), 876–884 (2008). <https://doi.org/10.1016/j.jprocont.2008.06.003>
- [9] T.A. Johansen, "Approximate explicit receding horizon control of constrained nonlinear systems", *Automatica* 40(2), 293–300 (2004). <https://doi.org/10.1016/j.automatica.2003.09.021>
- [10] A. Bemporad, F. Borrelli, and M. Morari, "Piecewise linear optimal controllers for hybrid systems", in *Proc. 2000 American Control Conference*, Chicago, 2000, vol. 2, pp. 1190–1194. <https://doi.org/10.1109/ACC.2000.876688>
- [11] I. Boulkaibet, K. Belarbi, S. Bououden, T. Marwala, and M. Chadli, "A new T-S fuzzy model predictive control for nonlinear processes", *Expert Systems With Applications* 88, 132–151 (2017). <https://doi.org/10.1016/j.eswa.2017.06.039>
- [12] E. Essien, H. Ibrahim, M. Mehrandehz, and R. Idem, "Adaptive neuro-fuzzy inference system (ANFIS) – based model predictive control (MPC) for carbon dioxide reforming of methane (CDRM) in a plug flow tubular reactor for hydrogen production", *Thermal Science and Engineering Progress* 9, 148–161 (2019). <https://doi.org/10.1016/j.tsep.2018.11.010>
- [13] M. Ławryńczuk, "Computationally Efficient Model Predictive Control Algorithms: A Neural Network Approach", Springer, Heidelberg, Germany, 2014. <https://doi.org/10.1007/978-3-319-04229-9>
- [14] M. Ławryńczuk, "Nonlinear state-space predictive control with on-line linearisation and state estimation", *International Journal of Applied Mathematics and Computer Science* 25(4), 833–847 (2015). <https://doi.org/10.1515/amcs-2015-0060>
- [15] P. Marusak, "Advantages of an easy to design fuzzy predictive algorithm in control systems of nonlinear chemical reactors", *Applied Soft Computing* 9(3), 1111–1125 (2009). <https://doi.org/10.1016/j.asoc.2009.02.013>
- [16] P. Marusak, "Efficient model predictive control algorithm with fuzzy approximations of nonlinear models", *Lecture Notes in Computer Science* 5495, 448–457 [Adaptive and Natural Computing Algorithms ICANNGA, 2009]. https://doi.org/10.1007/978-3-642-04921-7_46
- [17] P. Marusak, "Numerically Efficient Fuzzy MPC Algorithm with Advanced Generation of Prediction: Application to a Chemical Reactor", *Algorithms* 13(6), 143 (2020). <https://doi.org/10.3390/a13060143>
- [18] P. Marusak, "Advanced Construction of Dynamic Matrix in Numerically Efficient Fuzzy MPC Algorithms", *Algorithms* 14(1), 25 (2021). <https://doi.org/10.3390/a14010025>
- [19] A. Piegat, "Fuzzy Modeling and Control", Physica-Verlag, Heidelberg, Germany, 2001. <https://doi.org/10.1007/978-3-7908-1824-6>
- [20] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its application to modeling and control", *IEEE Trans. Systems, Man and Cybernetics* 15(1), 116–132 (1985). <https://doi.org/10.1109/TSMC.1985.6313399>
- [21] F. Doyle, B.A. Ogunnaike, and R.K. Pearson, "Nonlinear model-based control using second-order Volterra models", *Automatica* 31(5), 697–714 (1995). [https://doi.org/10.1016/0005-1098\(94\)00150-H](https://doi.org/10.1016/0005-1098(94)00150-H)
- [22] A. Jain and R. Taparia, "Laguerre function based model predictive control for van-de-Vusse reactor", in *Proc. 2nd IEEE Int. Conf. Power Electronics, Intelligent Control and Energy Systems, ICPEICES 2018, Dehli, 2018*, vol. 3, pp. 1010–1015. <https://doi.org/10.1109/ICPEICES.2018.8897438>
- [23] S. Mate, H. Kodamana, S. Bhartiya, and P.S.V. Nataraj, "A Stabilizing Sub-Optimal Model Predictive Control for Quasi-Linear Parameter Varying Systems", *IEEE Control Systems Letters* 4(2), 402–407 (2020). <https://doi.org/10.1109/LCSYS.2019.2937921>