7

Fuzzy–neural Model Predictive Control of Multivariable Processes

Michail Petrov, Sevil Ahmed, Alexander Ichtev and Albena Taneva
Technical University Sofia, Branch Plovdiv/Control Systems Department
Bulgaria

1. Introduction

Predictive control is a model-based strategy used to calculate the optimal control action, by solving an optimization problem at each sampling interval, in order to maintain the output of the controlled plant close to the desired reference. Model predictive control (MPC) based on linear models is an advanced control technique with many applications in the process industry (Rossiter, 2003). The next natural step is to extend the MPC concept to work with nonlinear models. The use of controllers that take into account the nonlinearities of the plant implies an improvement in the performance of the plant by reducing the impact of the disturbances and improving the tracking capabilities of the control system.
In this chapter, Nonlinear Model Predictive Control (NMPC) is studied as a more applicable approach for optimal control of multivariable processes. In general, a wide range of industrial processes are inherently nonlinear. For such nonlinear systems it is necessary to apply NMPC. Recently, several researchers have developed NMPC algorithms (Martinsen et al., 2004) that work with different types of nonlinear models. Some of these models use empirical data, such as artificial neural networks and fuzzy logic models. The model accuracy is very important in order to provide an efficient and adequate control action. Accurate nonlinear models based on soft computing (fuzzy and neural) techniques, are increasingly being used in model-based control (Mollov et al., 2004).
On the other hand, the mathematical model type, which the modelling algorithm relies on, should be selected. State-space models are usually preferred to transfer functions, because the number of coefficients is substantially reduced, which simplifies the computation; systems instability can be handled; there is no truncation error. Multi-input multi-output (MIMO) systems are modelled easily (Camacho et al., 2004) and numerical conditioning is less important.
A state-space representation of a Takagi-Sugeno type fuzzy-neural model (Ahmed et al., 2010; Petrov et al., 2008) is proposed in the Section 2. This type of models ensures easier description and direct computation of the gradient control vector during the optimization procedure. Identification procedure of the proposed model relies on a training algorithm, which is well-known in the field of artificial neural networks.
Obtaining an accurate model is the first stage of the of the NMPC predictive control strategy. The second stage involves the computation of a future control actions sequence. In order to obtain the control actions, a previously defined optimization problem has to be solved. Different types of objective and optimization algorithms (Fletcher, 2000) can be used

in the optimization procedure. Two different approaches for NMPC are proposed in Section 3. They consider the unconstrained and constrained model predictive control problem. Both of the approaches use the proposed Takagi-Sugeno fuzzy-neural predictive model.
The proposed techniques of fuzzy-neural MPC are studied in Section 4, by experimental simulations in Matlab® environment in order to control the levels in a multi tank system (Inteco, 2009). The case study is capable to show how the proposed NMPC algorithms handle multivariable processes control problem.

2. Multivariable fuzzy-neural predictive model

The Takagi-Sugeno fuzzy-neural models are powerful modelling tools for a wide class of nonlinear systems. Fuzzy reasoning is capable of handling uncertain and imprecise information while neural networks can learn from samples. Fuzzy-neural networks combine the advantages of both artificial intelligent techniques and incorporate them in adaptive features. Those futures, based on a real time learning algorithm are the main advantage of the fuzzy-neural models.
The importance of the used in MPC strategy models and their adaptive characteristics is obvious. The accuracy of the model determines the accuracy of the control action. The proposed fuzzy-neural model is implemented in a classical NMPC scheme (Fig. 1) as a predictor (Camacho et al., 2004).

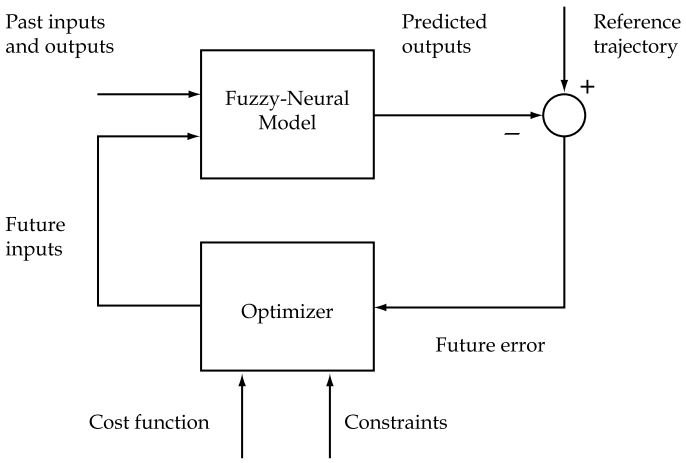

Fig. 1. Basic structure of the proposed Fuzzy-Neural NMPC

In this chapter a nonlinear discrete time state-space implementation is considered to represent the system dynamic:

$$\begin{aligned} x(k+1) &= f_x(x(k), u(k)) \\ y(k) &= f_y(x(k), u(k)) \end{aligned} \tag{1}$$

where $x(k) \in \Re^n$, $u(k) \in \Re^m$ and $y(k) \in \Re^q$ are state, control and output variables of the system, respectively. The unknown nonlinear functions f_x and f_y can be approximated by Takagi-Sugeno type fuzzy rules in the next form:

$$R_l: \; if \quad z_1(k) \; is \; M_{l1} \quad and \quad \dots \quad and \quad z_i(k) \; is \; M_{li} \quad and \quad \dots \quad z_p(k) \; is \; M_{lp}$$
$$then \quad \begin{cases} x_l(k+1) = A_l x(k) + B_l u(k) \\ y_l(k) = C_l x(k) + D_l u(k) \end{cases} \tag{2}$$

where R_l is the l-th rule of the rule base. Each rule is represented by an *if-then* conception. The antecedent part of the rules has the following form "$z_i(k)$ *is* M_{li}" where $z_i(k)$ is an i-th linguistic variable (i-th model input) and M_{li} is a membership function defined by a fuzzy set of the universe of discourse of the input z_i. Note that the input regression vector $\mathbf{z}(k) \in \Re^p$ in this chapter contains the system states and inputs $\mathbf{z}(k)=[x(k)\ u(k)]^T$. The consequent part of the rules is a mathematical function of the model inputs and states. A state-space implementation is used in the consequent part of R_l, where $A_l \in \Re^{n\times n}$, $B_l \in \Re^{n\times m}$, $C_l \in \Re^{q\times n}$ and $D_l \in \Re^{q\times m}$ are the state-space matrices of the model (Ahmed et al., 2009).
The states in the next sampling time $\hat{x}(k+1)$ and the system output $\hat{y}(k)$ can be obtained by taking the weighted sum of the activated fuzzy rules, using

$$\begin{aligned} \hat{x}(k+1) &= \sum_{l=1}^{L} \bar{\mu}_{yl}(k)(A_l x(k) + B_l u(k)) \\ \hat{y}(k) &= \sum_{l=1}^{L} \bar{\mu}_{yl}(k)(C_l \hat{x}(k) + D_l u(k)) \end{aligned} \tag{3}$$

On the other hand the state-space matrices A, B, C, and D for the global state-space plant model could be calculated as a weighted sum of the local matrices A_l, B_l, C_l, and D_l from the activated fuzzy rules (2):

$$\begin{aligned} A(k) &= \sum_{l=1}^{L} A_l \bar{\mu}_{yl}(k) \qquad \mathrm{B}(k) = \sum_{l=1}^{L} B_l \bar{\mu}_{yl}(k) \\ C(k) &= \sum_{l=1}^{L} C_l \bar{\mu}_{yl}(k) \qquad \mathrm{D}(k) = \sum_{l=1}^{L} D_l \bar{\mu}_{yl}(k) \end{aligned} \tag{4}$$

where $\bar{\mu}_{yl} = \mu_{yl} \Big/ \sum_{l=1}^{L} \mu_{yl}$ is the normalized value of the membership function degree μ_{yl} upon the l-th activated fuzzy rule and L is the number of the activated rules at the moment k.

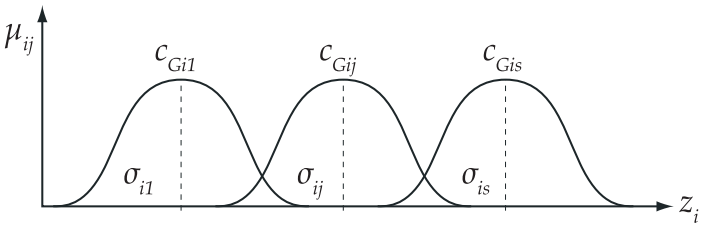

Fig. 2. Gaussian membership functions of the i-th input

Fuzzy implication in the l-th rule (2) can be realized by means of a product composition

$$\mu_{yl} = \prod_{i=1}^{p} \mu_{ij} \tag{5}$$

where μ_{ij} specifies the membership degree (Fig. 2) upon the activated j^{-th} fuzzy set of the corresponded i^{-th} input signal and it is calculated according to the chosen here Gaussian membership function (6) for the l^{-th} activated rule:

$$\mu_{ij}(z_i) = \exp - \frac{(z_i - c_{Gij})^2}{2\sigma_{ij}^2} \tag{6}$$

where z_i is the current input value of the i^{-th} model input, c_{Gij} is the centre (position) and σ_{ij} is the standard deviation (wide) of the j^{-th} membership function ($j=1, 2, .., s$) (Fig.2).

2.1 Identification procedure for the fuzzy–neural model

The proposed identification procedure determines the unknown parameters in the Takagi-Sugeno fuzzy model, i.e. the parameters of membership functions, according to their shape and the parameters of the functions f_x and f_y in the consequent part of the rules (2). It is realised by a five-layer fuzzy-neural network (Fig. 3). Each of the layers performs typical fuzzy logic strategy operations:

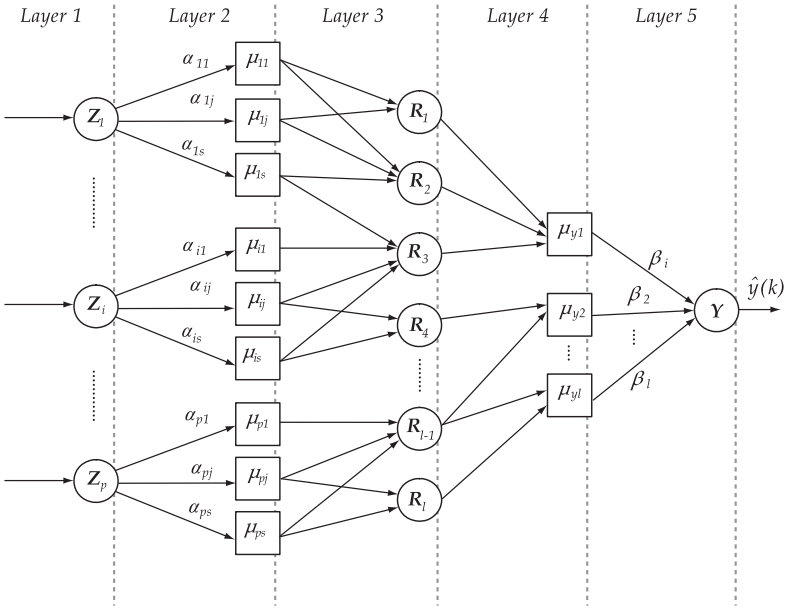

Fig. 3. The structure of the proposed fuzzy - neural model

Layer 1. The first layer represents the model inputs through its own input nodes Z_1, Z_2, …, Z_p. The network synaptic weights are set to one, so the model inputs are directly passed through the nodes to the next layer. Neurons here are represented by the elements of the regression vector $\mathbf{z}(k)$.

Layer 2. The fuzzification procedure of the input variables is performed in the second layer. The weights in this layer are the parameters of the chosen membership functions. Their number depends on the type and the number of the applied functions. All these parameters α_{ij} are adjustable and take part in the premise term of the Takagi-Sugeno type fuzzy rule base (2). In that section the membership functions for each model input variable are represented by Gaussian functions (Fig. 2). Hence, the adjustable parameters α_{ij} are the centres c_{Gij} and standard deviations σ_{ij} of the Gaussian functions (6). The nodes in the second layer of the fuzzy-neural architecture represent the membership degrees $\mu_{ij}(z_i)$ of the activated membership functions for each model input $z_i(k)$ according to (6). The number of the neurons depends on the number of the model inputs p and the number of the membership functions s in corresponding fuzzy sets. It is calculated as $p \times s$.

Layer 3. The third layer of the network interprets the fuzzy rule base (2). Each neuron in the third layer has as many inputs as the input regression vector size p. They are the corresponding membership degrees for the activated membership functions calculated in the previous layer. Therefore, each node in the third layer represents a fuzzy rule R_l, defined by Takagi-Sugeno fuzzy model. The outputs of the neurons are the results of the applied fuzzy rule base.

Layer 4. The fourth layer implements the fuzzy implication (5). Weights in this layer are set to one, in case the rule R_l from the third layer is activated, otherwise weights are zeros.

Layer 5. The last layer (one node layer) represents the defuzzyfication procedure and forms the output of the fuzzy-neural network (3). This layer also contains a set of adjustable parameters – β_l. These are the parameters in the consequent part of Takagi-Sugeno fuzzy model (2). The single node in this layer computes the overall model output signal as the summation of all signals coming from the previous layer.

$$I^5 = \sum_{l=1}^{L} f_{yl}\mu_{yl} \text{ or } I^5 = \sum_{l=1}^{L} f_{xl}\mu_{yl} \quad O^5 = \frac{\sum_{l=1}^{L} f_{yl}\mu_{yl}}{\sum_{l=1}^{L} \mu_{yl}} \text{ or } O^5 = \frac{\sum_{l=1}^{L} f_{xl}\mu_{yl}}{\sum_{l=1}^{L} \mu_{yl}} \tag{7}$$

where $f_{xl} = A_l x(k) + B_l u(k)$ and $f_{yl} = C_l x(k) + D_l u(k)$.

2.2 Learning algorithm of the fuzzy–neural model

Two-step gradient learning procedure is used as a learning algorithm of the internal fuzzy-neural model. It is based on minimization of an instant error function E_{FNN}. At time k the function is obtained from the following equation

$$E_{FNN}(k) = \varepsilon^2(k) / 2 \tag{8}$$

where the error $\varepsilon(k)$ is calculated as a difference between the controlled process output $y(k)$ and the fuzzy-neural model output $\hat{y}(k)$:

$$\varepsilon(k) = y(k) - \hat{y}(k) \tag{9}$$

During step one of the procedure, the consequent parameters of Takagi-Sugeno fuzzy rules are calculated according to summary expression (10) (Petrov et al., 2002).

$$\beta_l(k+1) = \beta_l(k) + \eta\left(-\frac{\partial E_{FNN}}{\partial \beta_l}\right) \tag{10}$$

where η is a learning rate and β_l represents an adjustable coefficient a_{ij}, b_{ij}, c_{ij}, d_{ij} (11) for the activated fuzzy rule R_l (2). The coefficients take part in the state matrix A_l, control matrix B_l and output matrices C_l and D_l of the l-th activated rule (Ahmed et al., 2009). The matrices approximate the unknown nonlinear functions f_x and f_y according to defined fuzzy rule model (2). The matrix dimensions are specified by the system parameters – numbers of inputs m, outputs q and states n of the system.

$$A_l = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \quad B_l = \begin{bmatrix} b_{11} & \cdots & b_{1m} \\ \vdots & & \vdots \\ b_{n1} & \cdots & b_{nm} \end{bmatrix} \quad C_l = \begin{bmatrix} c_{11} & \cdots & c_{1n} \\ \vdots & & \vdots \\ c_{q1} & \cdots & c_{qn} \end{bmatrix} \quad D_l = \begin{bmatrix} d_{11} & \cdots & d_{1m} \\ \vdots & & \vdots \\ d_{q1} & \cdots & d_{qm} \end{bmatrix} \tag{11}$$

In order to find a weight correction for the parameters in the last layer of the proposed fuzzy-neural network the derivative $\frac{\partial E_{FNN}}{\partial \beta_l}$ of the instant error should be determined. Following the chain rule, the derivative is calculated considering the expressions (7) and (8)

$$\frac{\partial E_{FNN}}{\partial \beta_l} = \frac{\partial E_{FNN}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial I^5} \cdot \frac{\partial I^5}{\partial \beta_l} \tag{12}$$

After the calculation of the partial derivatives, the matrix elements for each matrix of the state-space equations corresponding to the l-th activated rule (2) are obtained according to the summary expression (12) (Petrov et al., 2002; Ahmed et al., 2010):

$$\begin{aligned} a_{ij}(k+1) &= a_{ij}(k) + \eta \varepsilon(k) \bar{\mu}_{yl}(k) x_i(k) && i = j = 1 \div n \\ b_{ij}(k+1) &= b_{ij}(k) + \eta \varepsilon(k) \bar{\mu}_{yl}(k) u_j(k) && i = 1 \div n, j = 1 \div m \\ c_{ij}(k+1) &= c_{ij}(k) + \eta \varepsilon(k) \bar{\mu}_{yl}(k) x_j(k) && i = 1 \div q, j = 1 \div n \\ d_{ij}(k+1) &= d_{ij}(k) + \eta \varepsilon(k) \bar{\mu}_{yl}(k) u_j(k) && i = 1 \div q, j = 1 \div m \end{aligned} \tag{13}$$

The proposed fuzzy-neural architecture allows the use of the previously calculated output error (8) in the next step of the parameters update procedure. The output error E_{FNN} is propagated back directly to the second layer, where the second group of adjustable parameters are situated (Fig. 3). Depending on network architecture, the membership degrees calculated in the fourth and the second network layer are related as $\mu_{yl} \rightarrow \mu_{ij}$. Therefore, the learning rule for the second group adjustable parameters can be done in similar expression as (10):

$$\alpha_{ij}(k+1) = \alpha_{ij}(k) + \eta \left(-\frac{\partial E_{FNN}}{\partial \alpha_{ij}} \right) \tag{14}$$

where the derivative of the output error E_{FNN} is calculated by the separate partial derivatives:

$$\frac{\partial E_{FNN}}{\partial \alpha_{ij}} = \frac{\partial E_{FNN}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial \mu_{ij}} \cdot \frac{\partial \mu_{ij}}{\partial \alpha_{ij}} \tag{15}$$

The adjustable premise parameters of the fuzzy-neural model are the centre c_{Gij} and the deviation σ_{ij} of the Gaussian membership function (6). They are combined in the representative parameter α_{ij}, which corresponds to the i-th model input and its j-th activated fuzzy set. Following the expressions (14) and (15) the parameters are calculated as follows (Petrov et al., 2002; Ahmed et al., 2010):

$$c_{Gij}(k+1) = c_{Gij}(k) + \eta\varepsilon(k)\bar{\mu}_{yl}(k)[f_{yl} - \hat{y}(k)]\frac{[z_i(k) - c_{Gij}(k)]}{\sigma_{ij}^2(k)} \tag{16}$$

$$\sigma_{ij}(k+1) = \sigma_{ij}(k) + \eta\varepsilon(k)\bar{\mu}_{yl}(k)[f_{yl} - \hat{y}(k)]\frac{[z_i(k) - c_{Gij}(k)]^2}{\sigma_{ij}^3(k)} \tag{17}$$

The proposed identification procedure for the fuzzy-neural model could be summarized in the following steps (Table 1).

Step 1.	Initialize the membership functions – number, shape, parameters;
Step 2.	Assign initial values for the network inputs;
Step 3.	Start the algorithm at the current moment *k*;
Step 4.	Fuzzify the network inputs and calculate the membership degrees upon the activated fuzzy set of the membership functions according to (6);
Step 5.	Perform fuzzy implication according to (5);
Step 6.	Calculate the fuzzy-neural network output, which is represented by state-space description of the modelled system – (3) and (4);
Step 7.	Calculate the instant error according to (8) and (9);
Step 8.	Start training procedure for fuzzy-neural network;
Step 9.	Adjust the consequent parameters according to (13);
Step 10.	Adjust the premise parameters according to (16) and (17).
Repeat the algorithm from Step 3 for each sampling time.	

Table 1. Fuzzy-neural model identification procedure

3. Optimization algorithm of multivariable model predictive control strategy

The model provided by the Takagi-Sugeno type fuzzy-neural network is used to formulate the objective function for the optimization algorithm and to calculate the future control actions. The second stage of the predictive control strategy includes an optimization procedure. It utilizes the obtained results during the first (modelling) stage predictive model of the system. Using the Takagi-Sugeno fuzzy-neural model (3), the optimization algorithm computes the future control actions at each sampling period, by minimizing the typical for MPC strategy (Generalized Predictive Control – GPC) cost function (Akesson, 2006):

$$J(k) = \sum_{i=H_w}^{H_p+H_w-1} \left\|\hat{y}(k+i) - r(k+i)\right\|_{\bar{Q}}^2 + \sum_{i=0}^{H_u-1} \left\|\Delta u(k+i)\right\|_{\bar{R}}^2 \tag{18}$$

where $\hat{y}(k)$, $r(k)$ and $\Delta u(k)$ are the predicted outputs, the reference trajectories, and the predicted control increments at time k, respectively. The length of the prediction horizon is

H_p, and the first sample to be included in the horizon is H_w. The control horizon is given by H_u. $\tilde{Q} \geq 0$ and $\tilde{R} > 0$ are weighting matrices representing the relative importance of each controlled and manipulated variable and they are assumed to be constant over the H_p.
The cost function (18) may be rewritten in a matrix form as follows

$$J(k) = \left\| Y(k) - T(k) \right\|_Q^2 + \left\| \Delta U(k) \right\|_R^2 \tag{19}$$

where $Y(k)$, $T(k)$, $\Delta U(k)$, Q and R are predicted output, system reference, control variable increment and weighting matrices, respectively,

$$Y(k) = \begin{bmatrix} \hat{y}(k \mid k) \\ \vdots \\ \hat{y}(k + H_p - 1 \mid k) \end{bmatrix}, \; T(k) = \begin{bmatrix} r(k \mid k) \\ \vdots \\ r(k + H_p - 1 \mid k) \end{bmatrix}, \; \Delta U(k) = \begin{bmatrix} \Delta u(k \mid k) \\ \vdots \\ \Delta u(k + H_u - 1 \mid k) \end{bmatrix}$$

$$Q = \begin{bmatrix} \tilde{Q}(1) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \tilde{Q}(H_p) \end{bmatrix} \qquad R = \begin{bmatrix} \tilde{R}(1) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \tilde{R}(H_u) \end{bmatrix}$$

The linear state-space model used for Takagi-Sugeno fuzzy rules (2) could be represented in the following form:

$$\begin{aligned} \hat{x}(k+1) &= Ax(k) + Bu(k-1) + B\Delta u(k) \\ \hat{y}(k) &= C\hat{x}(k) + Du(k-1) + D\Delta u(k) \end{aligned} \tag{20}$$

Based on the state-space matrices A, B, C and D (4), the future state variables are calculated sequentially using the set of future control parameters:

$$\hat{x}(k+1) = Ax(k) + Bu(k-1) + B\Delta u(k)$$
$$\hat{x}(k+2) = A^2 x(k) + (AB+B)u(k-1) + (AB+B)\Delta u(k) + B\Delta u(k+1)$$
$$\hat{x}(k+3) = A^3 x(k) + (A^2B + AB + B)u(k-1) + (A^2B + AB + B)\Delta u(k) + (AB+B)\Delta u(k+1) + B\Delta u(k+2)$$
.....
$$\hat{x}(k+j) = A^j x(k) + \sum_{i=0}^{j-1} A^i B u(k-1) + \sum_{i=0}^{j-1} A^i B \sum_{m=0}^{j-1-i} \Delta u(k+m)$$
.....
$$\hat{x}(k+H_p) = A^{H_p} x(k) + \sum_{i=0}^{H_p-1} A^i B u(k-1) + \sum_{i=0}^{H_p-1} A^i B \Delta u(k) + \sum_{i=0}^{H_p-2} A^i B \Delta u(k+1) + \cdots + A^{H_p-H_u} B \Delta u(k+H_u-1)$$

The predictions of the output $\hat{y}$ for j steps ahead could be calculated as follows

$$\hat{y}(k+1) = Cx(k+1) + Du(k+1) = CAx(k) + (CB+D)u(k-1) + (CB+D)\Delta u(k) + D\Delta u(k+1)$$
$$\hat{y}(k+2) = CA^2 x(k) + (CAB + CB + D)u(k-1) + (CAB + CB + D)\Delta u(k) + (CB+D)\Delta u(k+1) + D\Delta u(k+2)$$
.....

$$\hat{y}(k+j)=CA^{j}x(k)+\left(C\sum_{i=0}^{j-1}A^{i}B+D\right)u(k-1)+\left(C\sum_{i=0}^{j-1}A^{i}B+D\right)\Delta u(k)+(CB+D)\Delta u(k+j-1)+D\Delta u(k+j)$$

.....

$$\hat{y}(k+H_p-1)=CA^{H_p-1}x(k)+\left(C\sum_{i=0}^{H_p-2}A^{i}B+D\right)u(k-1)+\left(C\sum_{i=0}^{H_p-2}A^{i}B+D\right)\Delta u(k)+$$
$$+\left(C\sum_{i=0}^{H_p-3}A^{i}B+D\right)\Delta u(k+1)+\cdots+\left(C\sum_{i=0}^{H_p-H_u-1}A^{i}B+D\right)\Delta u(k+H_u-1)$$

The recurrent equation for the output predictions $\hat{y}(k+j_p)$, where j_p= *1, 2,..., H_p –1,* is in the next form:

$$\cdot\,\hat{y}(k+j_p)=CA^{j_p}x(k)+\left(C\sum_{i=0}^{j_p-1}A^{i}B+D\right)u(k-1)+\begin{cases}\sum_{i=0}^{j_p}\left(C\sum_{j=0}^{j_p-1}A^{j}B+D\right)\Delta u(k+i), j_p<H_u\\ \sum_{i=0}^{H_u-1}\left(C\sum_{j=0}^{j_p-i-1}A^{j}B+D\right)\Delta u(k+i), j_p>H_u\end{cases}. \quad (21)$$

The prediction model defined in (21) can be generalized by the following matrix equality

$$Y(k) = \Psi x(k) + \Gamma u(k-1) + \Theta\Delta U(k) \quad (22)$$

where

$$\Psi=\begin{bmatrix}C\\ CA\\ CA^{2}\\ \vdots\\ CA^{Hp-1}\end{bmatrix}\quad \Gamma=\begin{bmatrix}D\\ CB+D\\ CAB+CB+D\\ \vdots\\ C\sum_{i=0}^{H_p-2}A^{i}B+D\end{bmatrix}\quad \Theta=\begin{bmatrix}D & 0 & \cdots & 0\\ CB+D & D & \cdots & \vdots\\ CAB+CB+D & CB+D & \ddots & \vdots\\ \vdots & & \ddots & 0\\ C\sum_{i=0}^{H_u-2}A^{i}B+D & & \cdots & D\\ \vdots & & \ddots & \vdots\\ C\sum_{i=0}^{H_p-2}A^{i}B+D & & \cdots & C\sum_{i=0}^{H_p-H_u-1}A^{i}B+D\end{bmatrix}$$

All matrices, which take part in the equations above, are derived by the Takagi-Sugeno fuzzy-neural predictive model (4).
It is also possible to define the vector

$$E(k) = T(k) - \Gamma u(k-1) - \Theta\Delta U(k) \quad (23)$$

This vector can be thought as a *tracking error*, in the sense that it is the difference between the future target trajectory and the *free response* of the system, namely the response that would occur over the prediction horizon if no input changes were made, i.e. $\Delta U(k)=0$. Hence, the quantity of the so called *free response F(k)* is defined as follows

$$F(k) = \Psi x(k) + \Gamma u(k-1) \quad (24)$$

3.1 Unconstrained model predictive control

In this section, the study is focused on the optimization problem of the unconstrained nonlinear predictive control with the quadratic cost function (18). The section presents an approximate solution of the problem where the information given by the obtained fuzzy-neural model is used to solve the problem.

The unconstrained optimization problem can be formulated in a matrix form. First, the predictor can be constructed as follows

$$Y(k) = \Theta \Delta U(k) + F(k) \tag{25}$$

Second, the cost function (19) can be rewritten as

$$J(\Delta U) = (T - Y)^T Q(T - Y) + \Delta U^T R \Delta U \tag{26}$$

Hence, substituting the predictive model (25) into expression (26), the cost function of the model predictive optimization problem can be specified as follows:

$$J(\Delta U) = \Delta U^T (\Theta^T Q \Theta + R) \Delta U + 2(F - T)^T Q \Theta \Delta U + (T - F)^T Q (T - F) \tag{27}$$

The minimum of the function $J(\Delta U)$ can be obtained by calculating the input sequence ΔU so that $\partial J / \partial \Delta U = 0$:

$$\frac{\partial}{\partial \Delta U} J(\Delta U) = \frac{\partial}{\partial \Delta U} \left[\Delta U^T (\Theta^T Q \Theta + R) \Delta U + 2(F - T)^T Q \Theta \Delta U + (F - T)^T Q (F - T) \right] = 0 \tag{28}$$

Then the optimal sequence ΔU^* is

$$\Delta U^* = (\Theta^T Q \Theta + R)^{-1} \Theta^T Q (T - F) \tag{29}$$

The input applied to the controlled plant at time k is computed according to the receding horizon principle, i.e. the first element from the control sequence $\Delta u^*(k)$ of the vector ΔU^* is taken. Then, control signal is calculated from:

$$u(k) = u(k - 1) + \Delta u^*(k) \tag{30}$$

It is evident that the expression given by the matrix equation (29) is the same as expression obtained for the generalized predictive control. However, in the GPC formulation the components involved in the calculation of the formula (29) are obtained from a linear model. In the present case the components introduced in this expression are generated by the designed nonlinear fuzzy-neural model. A more rigorous formulation of (29) will be representation of the components as time-variant matrices, as they are shown in the expression (22). In this case the matrix $\Theta(k)$ and the vectors $\Psi(k)$, $T(k)$ are being reconstructed at each sampling time. The vector $\Psi(k)$ is obtained by simulating the fuzzy model with the current input $u(k)$; the matrix $\Theta(k)$ is also rebuilt using a method described below.

$$\Delta U^*(k) = \left[\Theta^T(k) Q \Theta(k) + R \right]^{-1} \Theta^T(k) Q \left[T(k) - F(k) \right] \tag{31}$$

The proposed method solves the problem of unconstrained MPC. A system of equations is solved at each sampling time k. The proposed approach decreases computational burden avoiding the necessity to inverse the gain matrix in (31) at each sampling time k.

Applying this method, minimization of the GPC criterion (18) is based on a calculation of the gradient vector of the criterion cost function J at the moment k subject to the predicted control actions:

$$\nabla J(k)=\left[\frac{\partial J(k)}{\partial \Delta u(k)},\frac{\partial J(k)}{\partial \Delta u(k+1)},\ldots,\frac{\partial J(k)}{\partial \Delta u(k+H_u-1)}\right]^T \tag{32}$$

Each element of this gradient vector (32) can be calculated using the following derivative matrix equation:

$$\frac{\partial J(k)}{\partial \Delta U(k)}=\left[2\left[T(k)-Y(k)\right]^T Q\frac{\partial Y(k)}{\partial \Delta U(k)}+2\Delta U(k)^T R\frac{\partial \Delta U(k)}{\partial \Delta U(k)}\right] \tag{33}$$

From the above expression (33) it can be seen that it is necessary to obtain two groups of partial derivatives. The first one is $\left[\frac{\partial Y(k)}{\partial \Delta U(k)}\right]$, and the second one is $\left[\frac{\partial \Delta U(k)}{\partial \Delta U(k)}\right]$. The first partial derivatives in (33) have the following matrix form:

$$\frac{\partial Y(k)}{\partial \Delta U(k)}=\begin{bmatrix}\frac{\partial \hat{y}(k+H_w)}{\partial \Delta u(k)} & \cdots & \frac{\partial \hat{y}(k+H_w)}{\partial \Delta u(k+H_u-1)}\\ \vdots & & \vdots\\ \frac{\partial \hat{y}(k+H_p+H_w-1)}{\partial \Delta u(k)} & \cdots & \frac{\partial \hat{y}(k+H_p+H_w-1)}{\partial \Delta u(k+H_u-1)}\end{bmatrix} \tag{34}$$

For computational simplicity assume that H_w=0 (18). Then each element of the matrix (34) is calculated by the expressed equations according to the Takagi-Sugeno rules consequents (2). For example the derivatives from first column of the matrix (34) have the following form:

$$\frac{\partial \hat{y}(k)}{\partial \Delta u(k)}=\sum_{l=1}^{L} D_l\bar{\mu}_{yl}(k) \tag{35}$$

$$\frac{\partial \hat{y}(k+1)}{\partial \Delta u(k)}=\sum_{l=1}^{L}\left(C_lB_l+D_l\right)\bar{\mu}_{yl}(k+1) \tag{36}$$

$$\frac{\partial \hat{y}(k+2)}{\partial \Delta u(k)}=\sum_{l=1}^{L}\left(C_lA_lB_l+C_lB_l+D_l\right)\bar{\mu}_{yl}(k+2) \tag{37}$$

……..

$$\frac{\partial \hat{y}(k+H_p-1)}{\partial \Delta u(k)}=\sum_{l=1}^{L}\left(C_l\sum_{j=0}^{H_p-2}A_l^jB_l+D_l\right)\bar{\mu}_{yl}(k+H_p-1) \tag{38}$$

The second group partial derivatives in (33) has the following matrix form:

$$\frac{\partial \Delta U(k)}{\partial \Delta U(k)}=\begin{bmatrix} \frac{\partial \Delta u(k)}{\partial \Delta u(k)} & \cdots & \frac{\partial \Delta u(k)}{\partial \Delta u(k+H_u-1)} \\ \vdots & & \vdots \\ \frac{\partial \Delta u(k+H_u-1)}{\partial \Delta u(k)} & \cdots & \frac{\partial \Delta u(k+H_u-1)}{\partial \Delta u(k+H_u-1)} \end{bmatrix} \tag{39}$$

Since $\Delta u(k)=u(k)-u(k-1)$, the matrix (39) has the following form:

$$\frac{\partial \Delta U(k)}{\partial \Delta U(k)}=\begin{bmatrix} 1 & 0 & \cdots & 0 \\ -1 & 1 & & \\ \vdots & -1 & \ddots & \vdots \\ 1 & \vdots & \ddots & 1 \\ -1 & 1 & \cdots & -1 \end{bmatrix} \tag{40}$$

Following this procedure it is possible to calculate the rest column elements of the matrix (34) which belongs to the next gradient vector elements (32). Finally, each element of the gradient-vector (32) could be obtained by the following system of equations:

$$\frac{\partial J(k)}{\partial \Delta u(k+1)}=2\hat{e}(k+1)\tilde{Q}(1)\frac{\partial \hat{y}(k+1)}{\partial \Delta u(k)}+\ldots+2\hat{e}(k+H_p)\tilde{Q}(H_p)\frac{\partial \hat{y}(k+H_p)}{\partial \Delta u(k)}+ \\ +2\tilde{R}(1)\Delta u(k)-2\tilde{R}(2)\Delta u(k+1)+\ldots-2\tilde{R}(H_u)\Delta u(k+H_u-1)=0 \tag{41}$$

$$\frac{\partial J(k)}{\partial \Delta u(k+2)}=2\hat{e}(k+1)\tilde{Q}(1)\frac{\partial \hat{y}(k+1)}{\partial \Delta u(k+1)}+\ldots+2\hat{e}(k+H_p)\tilde{Q}(H_p)\frac{\partial \hat{y}(k+H_p)}{\partial \Delta u(k+1)}+ \\ +2\tilde{R}(2)\Delta u(k+1)-2\tilde{R}(3)\Delta u(k+2)+\ldots-2\tilde{R}(H_u)\Delta u(k+H_u-1)=0 \tag{42}$$

$$\frac{\partial J(k)}{\partial \Delta u(k+H_u-2)}=2\hat{e}(k+1)\tilde{Q}(1)\frac{\partial \hat{y}(k+1)}{\partial \Delta u(k+H_u-2)}+\ldots+2\hat{e}(k+H_p)\tilde{Q}(H_p)\frac{\partial \hat{y}(k+H_p)}{\partial \Delta u(k+H_u-2)}+ \\ +2\tilde{R}(H_u-1)\Delta u(k+H_u-2)-2\tilde{R}(H_u)\Delta u(k+H_u-1)=0 \tag{43}$$

$$\frac{\partial J(k)}{\partial \Delta u(k+H_u-1)}=2\hat{e}(k+1)\tilde{Q}(1)\frac{\partial \hat{y}(k+1)}{\partial \Delta u(k+H_u-1)}+\ldots+2\hat{e}(k+H_p)\tilde{Q}(H_p)\frac{\partial \hat{y}(k+H_p)}{\partial \Delta u(k+H_u-1)}+ \\ +2\tilde{R}(H_u)\Delta u(k+H_u-1)=0 \tag{44}$$

where $\hat{e}(k+j)=r(k+j)-\hat{y}(k+j),\quad j=1,2,\ldots,H_p$ is the predicted system error.

The obtained system of equations (41)-(44) can be solved very easily, starting from the last equation (44) and calculating the last control action *u(k+H$_u$-1)* first. Then, the procedure can continue with finding the previous control action *u(k+H$_u$-2)* from (43). The calculations continue until the whole number of the control actions over the horizon H_u is obtained. The

calculation order of the control actions is very important, since the calculations should contain only known quantities. After that, only the first control action *u(k)* (30) will be used at the moment *k* as an input to the controlled process. The software implementation of the proposed algorithm is realized easily according to the following equations:

$$\Delta u(k+H_u-1)=\tilde{R}(H_u)^{-1}\left[\hat{e}(k+1)\tilde{Q}(1)\frac{\partial\hat{y}(k+1)}{\partial\Delta u(k+H_u-1)}+\cdots+\hat{e}(k+H_p)\tilde{Q}(H_p)\frac{\partial\hat{y}(k+H_p)}{\partial\Delta u(k+H_u-1)}\right] \quad (45)$$

$$\Delta u(k+H_u-2)=\Delta u(k+H_u-1)+ \\ +\tilde{R}(H_u-1)^{-1}\left[\hat{e}(k+1)\tilde{Q}(1)\frac{\partial\hat{y}(k+1)}{\partial\Delta u(k+H_u-2)}+\cdots+\hat{e}(k+H_p)\tilde{Q}(H_p)\frac{\partial\hat{y}(k+H_p)}{\partial\Delta u(k+H_u-2)}\right] \quad (46)$$

$$\Delta u(k+1)=\Delta u(k+2)+\tilde{R}(2)^{-1}\left[\hat{e}(k+1)\tilde{Q}(1)\frac{\partial\hat{y}(k+1)}{\partial\Delta u(k+1)}+\cdots+\hat{e}(k+H_p)\tilde{Q}(H_p)\frac{\partial\hat{y}(k+H_p)}{\partial\Delta u(k+1)}\right] \quad (47)$$

$$\Delta u(k)=\Delta u(k+1)+\tilde{R}(1)^{-1}\left[\hat{e}(k+1)\tilde{Q}(1)\frac{\partial\hat{y}(k+1)}{\partial\Delta u(k)}+\cdots+\hat{e}(k+H_p)\tilde{Q}(H_p)\frac{\partial\hat{y}(k+H_p)}{\partial\Delta u(k)}\right] \quad (48)$$

The proposed unconstrained predictive control algorithm could be summarized in the following steps (Table 2).

Step 1.	Initial identification of the Takagi-Sugeno fuzzy-neural predictive model;
Step 2.	Start the algorithm at the sample *k* with the initial parameters;
Step 3.	Calculate the predicted model output *ŷ(k+j)* using the tuned fuzzy-neural model (2);
Step 4.	Calculate the derivatives for the matrix (34) according to the equations (35)-(38);
Step 5.	Calculate predicted control actions according to (45)-(48) and update the sequence;
Step 6.	Apply the first optimal control action *u(k);*
Step 7.	Modify the model parameters into the rule (3) and update them for the next step 3 for the next sample *k*

Table 2. Basic fuzzy-neural model unconstrained predictive control algorithm

3.2 Constrained model predictive control

The constrained nonlinear predictive control problem can be described as a problem of finding the "optimal" input sequence to move a dynamic system to a desired state, taking into account the constraints on the inputs and the outputs of the control systems. This section reveals the formulation of the constrained control problem for MPC uses. Essentially, the problem becomes a quadratic programming problem with linear inequality constraints (LICQP). It follows by the nature of the operational constraints, which are usually described by linear inequalities of the control and plant variables.

The problem of nonlinear constrained predictive control is formulated as a nonlinear quadratic optimization problem. By means of local linearization (20) the problem can be solved using QP. That way the solution to the linear constrained predictive control problem is obtained. At each sampling time the LICQP is solved with new parameters, which are obtained by the Takagi-Sugeno fuzzy-neural model. An active set method is used for solving the constructed quadratic programming problem.

3.2.1 Constraint types in model predictive control

The operational constraints may be classified in three major types according to the type of the system variables, which they are imposed on. The first two types of constraints deal with the control variable incremental variation $\Delta u(k)$ and control variable $u(k)$. The third type is concerned with output $y(k)$ or state variable $x(k)$ constraints.

Related to the origin model predictive control problem, the constraints are expressed in a set of linear equations. All types of constraints are taken into consideration for each moving horizon window.

$$\begin{aligned} U_{min}(k) &\le U(k) \le U_{max}(k) \\ \Delta U_{min}(k) &\le \Delta U(k) \le \Delta U_{max}(k) \\ Y_{min}(k) &\le Y(k) \le Y_{max}(k) \end{aligned} \tag{49}$$

Where

$$U_{\max}(k)=\begin{bmatrix} u_{\max}(k) \\ u_{\max}(k+1) \\ \vdots \\ u_{\max}(k+N_u-1) \end{bmatrix} \quad \Delta U_{\max}(k)=\begin{bmatrix} \Delta u_{\max}(k) \\ \Delta u_{\max}(k+1) \\ \vdots \\ \Delta u_{\max}(k+N_u-1) \end{bmatrix}$$

$$U_{\min}(k)=\begin{bmatrix} u_{\min}(k) \\ u_{\min}(k+1) \\ \vdots \\ u_{\min}(k+N_u-1) \end{bmatrix} \quad \Delta U_{\min}(k)=\begin{bmatrix} \Delta u_{\min}(k) \\ \Delta u_{\min}(k+1) \\ \vdots \\ \Delta u_{\min}(k+N_u-1) \end{bmatrix}$$

$$Y_{\max}(k)=\begin{bmatrix} y_{\max}(k) \\ y_{\max}(k+1) \\ \vdots \\ y_{\max}(k+N_p-1) \end{bmatrix}$$

$$Y_{\min}(k)=\begin{bmatrix} y_{\min}(k) \\ y_{\min}(k+1) \\ \vdots \\ y_{\min}(k+N_p-1) \end{bmatrix}$$

Therefore, for multi-input case the number of the constraints for the change of the control variable $\Delta u(k)$ is $m \times N_u$. Similarly, the number of the constraints for the control variable amplitude is also $m \times N_u$ and for the output constraints it is $q \times N_{\mathrm{p}}$.

3.2.2 Quadratic programming in use of constrained MPC

Since the cost function $J(k)$ (19) is quadratic and the constraints are linear inequalities, the problem of finding an optimal predictive control becomes one of finding an optimal solution to a standard quadratic programming problem with linear inequality constraints

$$\begin{aligned} &minJ(x) = \frac{1}{2}x^T Hx + f^T x \\ &subject\ to\ Ax \le b \end{aligned} \tag{50}$$

where H and f are the Hessian and the gradient of the Lagrange function, x is the decision variable. Constraints on the QP problem (50) are specified by $Ax \le b$ according to (49).
The Lagrange function is defined as follows

$$L(x,\lambda) = J(x) + \sum_{i=1}^{N} \lambda_i a_i,\ \ i = 1,2,\ldots N\ , \tag{51}$$

where λ_i are the Lagrange multipliers, a_i are the constraints on the decision variable x, N is the number of the constraints considered in the optimization problem.
Several algorithms for constrained optimization are described in (Fletcher, 2000). In this chapter a primal active set method is used. The idea of active set method is to define a set S of constraints at each step of algorithm. The constraints in this active set are regarded as equalities whilst the rest are temporarily disregarded and the method adjusts the set in order to identify the correct active constraints on the solution to (52)

$$\begin{aligned} &\min J(x) = \frac{1}{2}x^T Hx + f^T x \\ &subject\ to\ a_i x = b_i \\ &\qquad\qquad a_i x \le b_i \end{aligned} \tag{52}$$

At iteration k a feasible point $x(k)$ is known which satisfies the active constraints as equalities. Each iteration attempts to locate a solution to an equality problem (EP) in which only the active constraints occur. This is most conveniently performed by shifting the origin to $x(k)$ and looking for a correction $\delta(k)$ which solves

$$\begin{aligned} &\min_{\delta}\left\{\frac{1}{2}\delta^T Hx + f^T \delta\right\} \\ &subject\ to\ a_i\delta = 0 \quad a_i \in S \end{aligned} \tag{53}$$

where $f(k)$ is defined by $f(k) = f + Hx(k)$ and is $\nabla J(x(k))$ for the function defined by (52). If $\delta(k)$ is feasible with regard to the constraints not included in S, then the feasible point in next iteration is taken as $x(k+1) = x(k) + \delta(k)$. If not, a line search is made in the direction of $\delta(k)$ to find the best feasible point. A constraint is active if the Lagrange multipliers $\lambda_i \ge 0$, i.e. it is at the boundary of the feasible region defined by the constraints. On the other hand, if there exist $\lambda_i < 0$, the constraint is not active. In this case the constraint is relaxed from the active constraints set S and the algorithm continues as before by solving the resulting equality constraint problem (53). If there is more than one constraint with corresponding $\lambda_i < 0$, then the $\min_{i \in S} \lambda_i(k)$ is selected (Fletcher, 2000).

The QP, described in that way, is used to provide numerical solutions in constrained MPC problem.

3.2.3 Design the constrained model predictive problem

The fuzzy-neural identification procedure from the Section 2 provides the state-space matrices, which are needed to construct the constrained model predictive control optimization problem. Similarly to the unconstrained model predictive control approach, the cost function (18) can be specified by the prediction expressions (22) and (23).

$$\begin{aligned} J(k) &= \left[\Psi x(k) + \Gamma u(k-1) + \Theta \Delta U(k) - T(k)\right]^T Q\left[\Psi x(k) + \Gamma u(k-1) + \Theta \Delta U(k) - T(k)\right] + \Delta U^T(k) R \Delta U(k) = \\ &= \left[\Theta \Delta U(k) - E(k)\right]^T Q\left[\Theta \Delta U(k) - E(k)\right] + \Delta U^T(k) R \Delta U(k) = \\ &= \Delta U^T(k)\left[\Theta^T Q \Theta + R\right]\Delta U(k) + E^T(k) Q E(k) - 2\Delta U^T(k) \Theta^T Q E(k) \end{aligned}$$

Assuming that

$$H = \Theta^T Q \Theta + R \text{ and } \Phi = 2\Theta^T Q E(k), \tag{54}$$

the cost function for the model predictive optimization problem can be specified as follow

$$J(k) = \Delta U^T(k) \mathrm{H}\, \Delta U(k) - \Delta U^T(k) \Phi + E^T(k) Q E(k) \tag{55}$$

The problem of minimizing the cost function (55) is a quadratic programming problem. If the Hessian matrix H is positive definite, the problem is convex (Fletcher, 2000). Then the solution is given by the closed form

$$\Delta U = \frac{1}{2} H^{-1} \Phi \tag{56}$$

The constraints (49) on the cost function may be rewritten in terms of $\Delta U(k)$.

$$\begin{gathered} U_{\min}(k) \le I_u u(k-1) + I_{\Delta u} \Delta U(k) \le U_{\max}(k) \\ \Delta U_{\min}(k) \le \Delta U(k) \le \Delta U_{\max}(k) \\ Y_{\min}(k) \le \Psi x(k) + \Gamma u(k-1) + \Theta \Delta U(k) \le Y_{\max}(k) \end{gathered} \tag{57}$$

where $I_m \in \Re^{m \times m}$ is an identity matrix, $I_u = \begin{bmatrix} I_m \\ I_m \\ \vdots \\ I_m \end{bmatrix} \in \Re^{mN_u \times m}$, $I_{\Delta u} = \begin{bmatrix} I_m & 0 & \cdots & 0 \\ I_m & I_m & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ I_m & I_m & \cdots & I_m \end{bmatrix} \in \Re^{mN_u \times mN_u}$.

All types of constraints are combined in one expression as follows

$$\begin{bmatrix} -I_{\Delta u} \\ I_{\Delta u} \\ -I \\ I \\ -\Theta \\ \Theta \end{bmatrix} \Delta U \le \begin{bmatrix} -U_{\min} + I_u u(k-1) \\ U_{\max} - I_u u(k-1) \\ \Delta U_{\min} \\ \Delta U_{\max} \\ -Y_{\min} + (\Psi x(k) + \Gamma u(k-1)) \\ Y_{\max} - (\Psi x(k) + \Gamma u(k-1)) \end{bmatrix} \tag{58}$$

where $I \in \Re^{mN_u \times mN_u}$ is an identity matrix.
Finally, following the definition of the LIQP (50), the model predictive control in presence of constraints is proposed as finding the parameter vector ΔU that minimizes (55) subject to the inequality constraints (58).

$$\begin{aligned} &\min\ J(k) = \Delta U^T H\, \Delta U - \Delta U^T \Phi + E^T Q E \\ &subject\ to\ \Omega \Delta U \le \omega \end{aligned} \tag{59}$$

In (59) the constraints expression (58) has been denoted by $\Omega \Delta U \le \omega$, where Ω is a matrix with number of rows equal to the dimension of ω and number of columns equal to the dimension of ΔU. In case that the constraints are fully imposed, the dimension of ω is equal to $4 \times m \times N_u + 2 \times q \times N_p$, where m is the number of system inputs and q is the number of outputs. In general, the total number of constraints is greater than the dimension of the ΔU. The dimension of ω represents the number of constraints.
The proposed model predictive control algorithm can be summarized in the following steps (Table 3).

At each sampling time:
Step 1. Read the current states, inputs and outputs of the system;
Step 2. Start identification of the fuzzy-neural predictive model following Algorithm 1;
Step 3. With $A(k)$, $B(k)$, $C(k)$, $D(k)$ from Step 2 calculate the predicted output $Y(k)$ according to (17);
Step 4. Obtain the prediction error $E(k)$ according to (23);
Step 5. Construct the cost function (55) and the constraints (58) of the QP problem;
Step 6. Solve the QP problem according to (59);
Step 7. Apply only the first control action $u(k)$.

Table 3. State-space implementation of fuzzy-neural model predictive control strategy

At each sampling time, LIQP (59) is solved with new parameters. The Hessian and the Lagrangian are constructed by the state-space matrices $A(k)$, $B(k)$, $C(k)$ and $D(k)$ (4) obtained during the identification procedure (Table 1). The problem of nonlinear constrained predictive control is formulated as a nonlinear quadratic optimization problem. By means of local linearization a relaxation can be obtained and the problem can be solved using quadratic programming. This is the solution of the linear constrained predictive control problem (Espinosa et al., 2005).

4. Fuzzy-neural model predictive control of a multi tank system. Case study

The case study is implemented in MATLAB/Simulink® environment with Inteco® Multi tank system. The Inteco® Multi tank System (Fig. 4) comprises from three separate tanks fitted with drain valves (Inteco, 2009). The additional tank mounted in the base of the set-up acts as a water reservoir for the system. The top (first) tank has a constant cross section, while others are conical or spherical, so they are with variable cross sections. This causes the main nonlinearities in the system. A variable speed pump is used to fill the upper tank. The liquid outflows the tanks by the gravity. The tank valves act as flow resistors C_1, C_2, C_3. The area ratio of the valves is controlled and can be used to vary the outflow characteristic. Each tank is equipped with a level sensor PS_1, PS_2, PS_3 based on hydraulic pressure measurement.

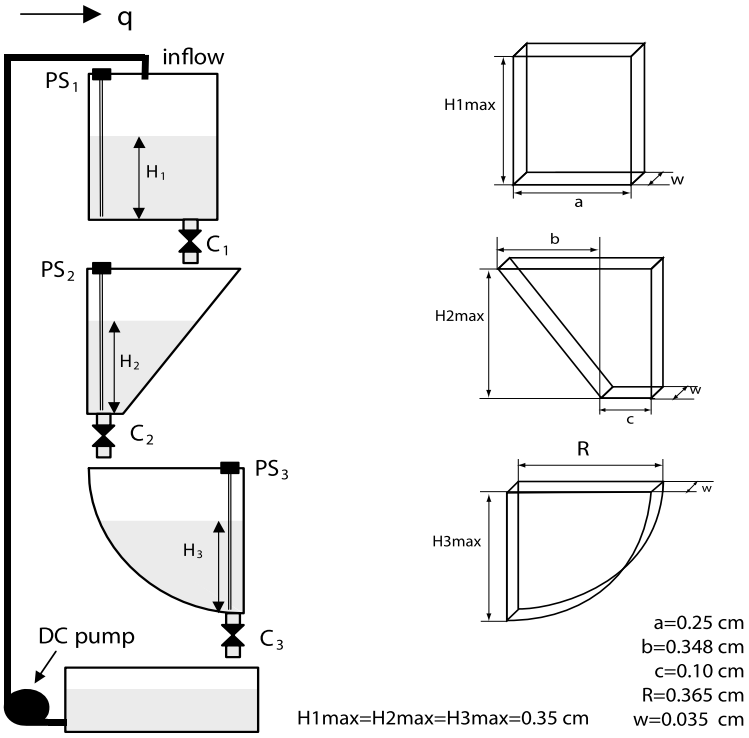

Fig. 4. Controlled laboratory multi tank system

The linearized dynamical model of the triple tank system could be described by the linear state-space equations (2) where the matrices A, B, C and D are as follow (Petrov et al., 2009):

$$A=\begin{bmatrix} \dfrac{-\alpha_1}{awH_1^{1-\alpha_1}} & 0 & 0 \\ \dfrac{\alpha_1}{w\left(c+bH_1/H_{1\max}\right)H_1^{1-\alpha_1}} & \dfrac{-\alpha_2}{w\left(c+bH_2/H_{2\max}\right)H_2^{1-\alpha_2}} & 0 \\ 0 & \dfrac{\alpha_2}{w\sqrt{R^2-(H_{3\max}-H_3)^2}\,H_2^{1-\alpha_2}} & \dfrac{-\alpha_2}{w\sqrt{R^2-(H_{3\max}-H_3)^2}\,H_3^{1-\alpha_3}} \end{bmatrix}$$

$$B=\begin{bmatrix} \dfrac{1}{aw} & \dfrac{-1}{awH_1^{1-\alpha_1}} & 0 & 0 \\ 0 & 0 & \dfrac{-1}{w\left(c+bH_2/H_{2\max}\right)H_2^{1-\alpha_2}} & 0 \\ 0 & 0 & 0 & \dfrac{-1}{w\sqrt{R^2-(H_{3\max}-H_3)^2}\,H_3^{1-\alpha_3}} \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{60}$$

The parameters α_1, α_2 and α_3 are flow coefficients for each tank of the model. The described linearized state-space model is used as an initial model for the training process of the fuzzy-neural model during the experiments.

4.1 Description of the multi tank system as a multivariable controlled process

Liquid levels H_1, H_2, H_3 in the tanks are the state variables of the system (Fig. 4). The Inteco Multi Tank system has four controlled inputs: liquid inflow q and valves settings C_1, C_2, C_3. Therefore, several models of the tanks system can be analyzed (Fig. 5), classified as pump-controlled system, valve-controlled system and pump/valve controlled system (Inteco, 2009).

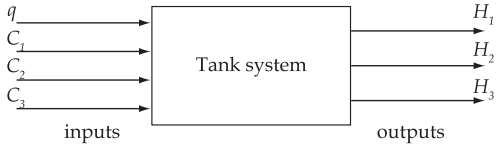

Fig. 5. Model of the Multi Tank system as a pump and valve-controlled system

In this case study a multi-input multi-output (MIMO) configuration of the Inteco Multi Tank system is used (Fig. 5). This corresponds to the linearized state-space model (60). Several issues have been recognized as causes of additional nonlinearities in plant dynamics:

- nonlinearities (smooth and nonsmooth) caused by shapes of tanks;
- saturation-type nonlinearities, introduced by maximum or minimum level allowed in tanks;
- nonlinearities introduced by valve geometry and flow dynamics;
- nonlinearities introduced by pump and valves input/output characteristic curve.

The simulation results have been obtained with random generated set points and following initial conditions (Table 4):

Model predictive controller parameters	Prediction horizon H_p=*10* First included sample of the prediction horizon H_w=*1* Control horizon H_u=*3*
Inteco Multi tank system parameters	Flow coefficients for each tank α1=0.29; α2=0.2256; α3=0.2487
Operational constraints on the system	Constraints on valve cross section ratio $0 \le C_i \le 2e\text{-}04$, $i=1,2,3$ Constraint on liquid inflow $0 \le q \le 1e\text{-}04$ m^3/s Constraints on liquid level in each tank $0 \le H_i \le 0.35$ *m*, $i=1,2,3$
Simulation parameters	Time of simulation *600 s* Sample time T_s=*1 s*

Table 4. Simulation parameters for unconstrained and constrained fuzzy-neural MPC

Figures below show typical results for level control problem. The reference value for each tank is changed consequently in different time. The proposed fuzzy-neural identification procedure ensures the matrices for the optimization problem of model predictive control at each sampling time T_s. The plant modelling process during the unconstrained and constrained MPC experiments are shown in Fig. 6 and Fig. 9, respectively.

4.2 Experimental results with unconstrained model predictive control

The proposed unconstrained model predictive control algorithm (Table 2) with the Takagi-Sugeno fuzzy-neural model as a predictor has been applied to the level control problem. The experiments have been implemented with the parameters in Table 4. The weighting matrices are specified as follow: $\tilde{Q} = 0.01 * diag(1,\ 1,\ 1)$ and $\tilde{R} = 10e4 * diag(1,\ 1,\ 1,\ 1)$. Note that the weighting matrix $\tilde{R}$ is constant over all prediction horizon, which allows to avoid matrix inversion at each sampling time with one calculation of $\tilde{R}^{-1}$ at time $k=0$.

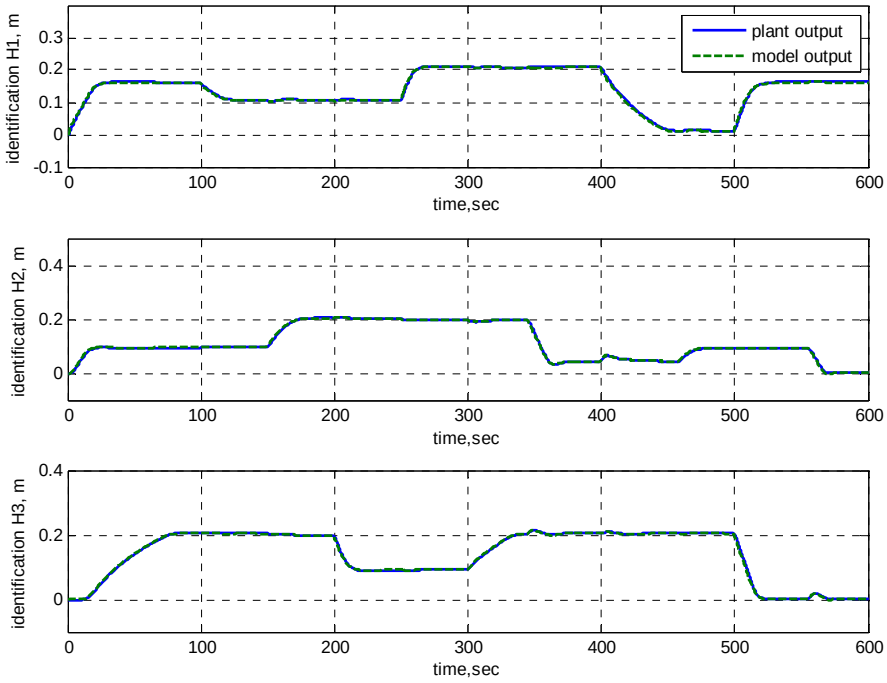

Fig. 6. Fuzzy-neural model identification procedure of the multi tank system – unconstrained NMPC

The next two figures - Fig. 7 and Fig. 8, show typical results regarding level control, where the references for H_1, H_2 and H_3 are changed consequently in different time. The change of every level reference behaves as a system disturbance for the other system outputs (levels). It is evident that the applied model predictive controller is capable to compensate these disturbances.

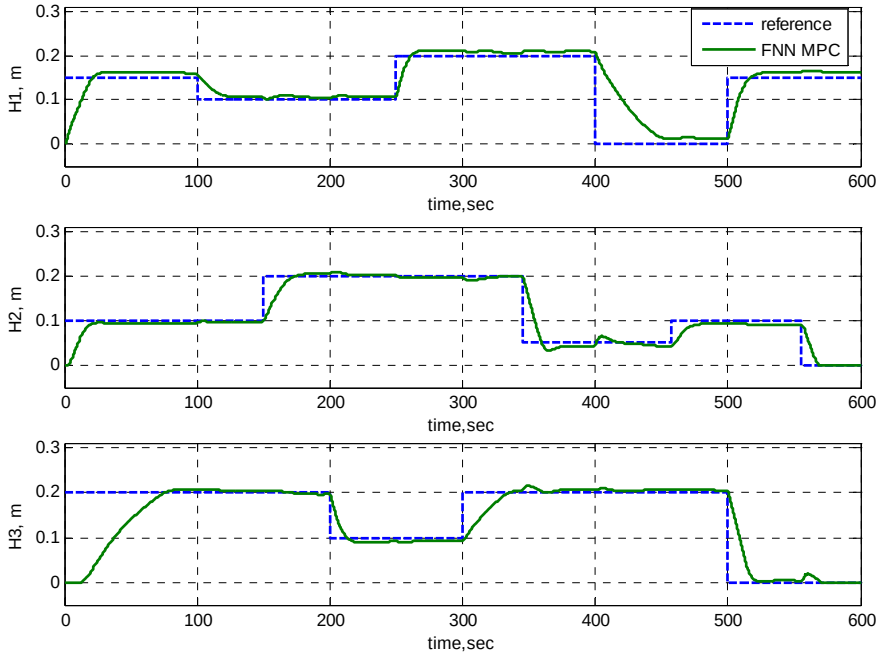

Fig. 7. Transient responses of multi tank system outputs – unconstrained NMPC

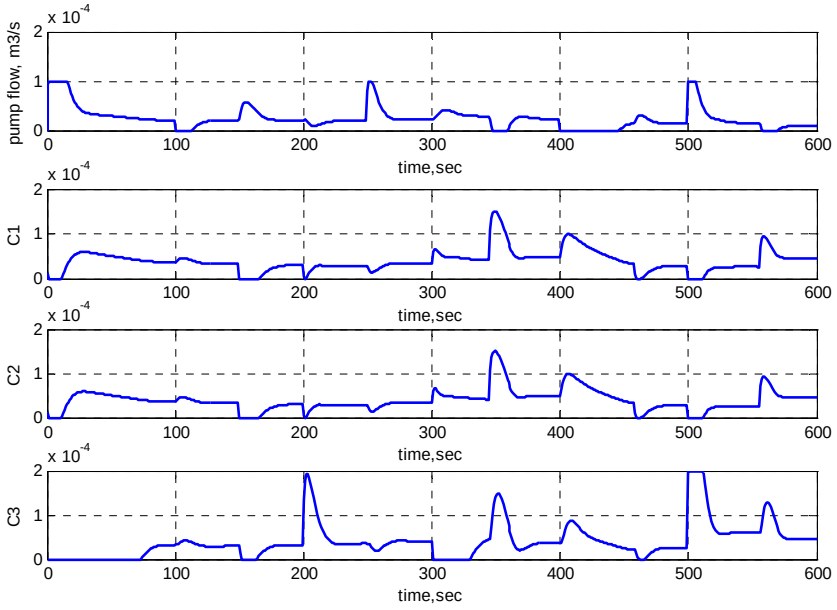

Fig. 8. Transient responses of multi tank system inputs – unconstrained NMPC

4.3 Experimental results with fuzzy-neural constrained predictive control

The experiments with the proposed constrained model predictive control algorithm (Table 3) have been made with level references close to the system outputs constraints. The weighting matrices in GPC cost function (19) are specified as $\tilde{Q} = diag(1,\ 1,\ 1)$ and $\tilde{R} = 15e4 * diag(1,\ 1,\ 1,\ 1)$. System identification during the experiment is shown on Fig. 9. The proposed identification procedure uses the linearized model (60) of the Multi tank system as an initial condition.

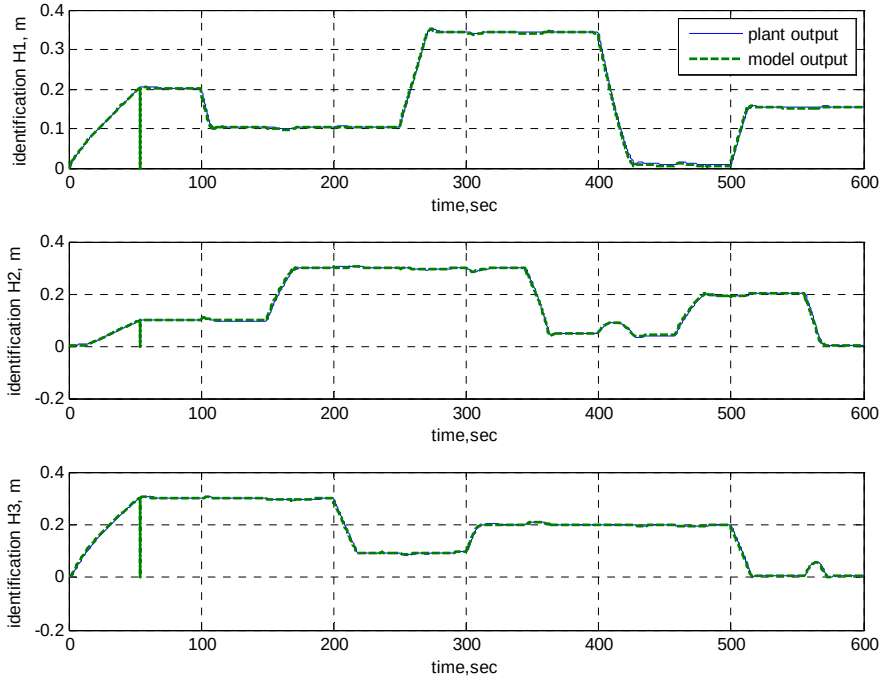

Fig. 9. Fuzzy-neural model identification procedure of the multi tank system – constrained NMPC

The proposed constrained fuzzy-neural model predictive control algorithm provides an adequate system response as it can be seen on Fig. 10 and Fig. 11. The references are achieved

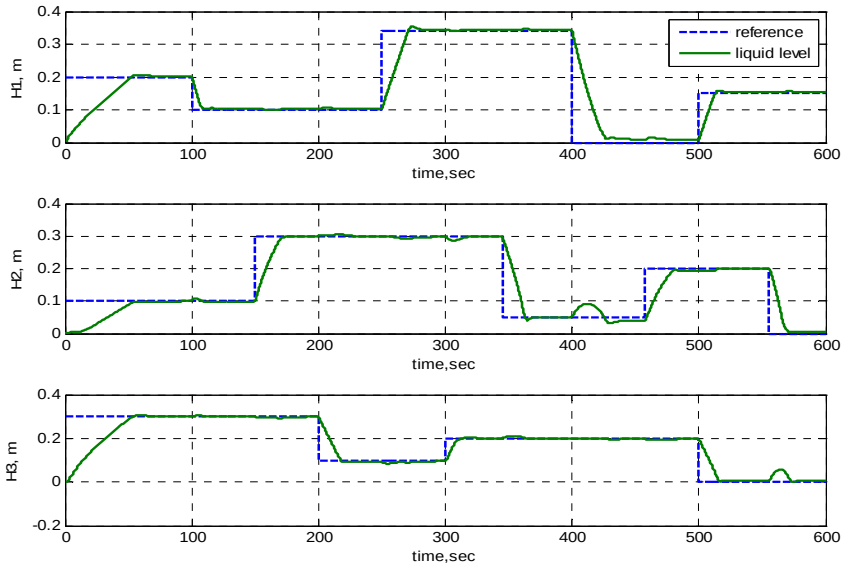

Fig. 10. Transient responses of the multi tank system outputs – constrained NMPC

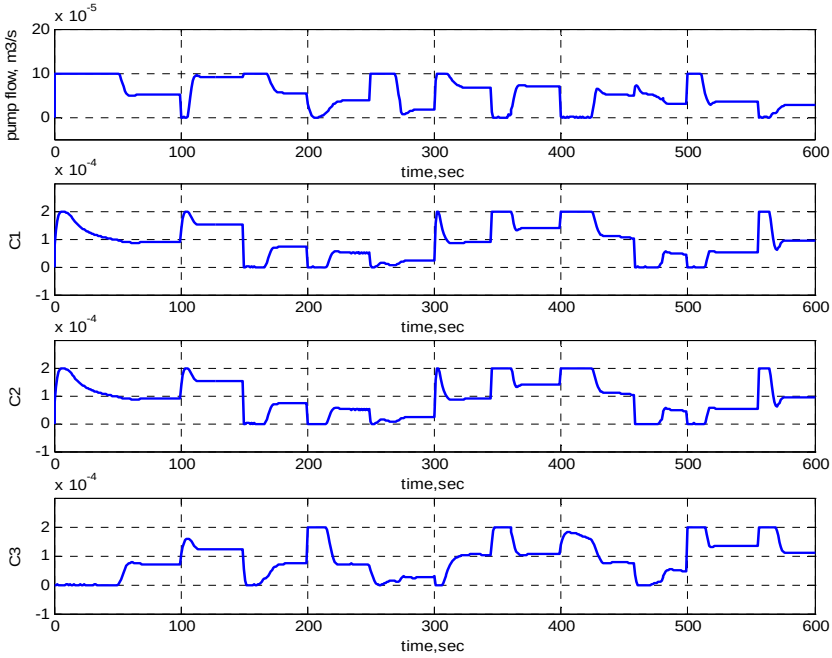

Fig. 11. Transient responses of the multi tank system inputs – constrained NMPC

without violating the operational constraints specified in Table 4. Similarly to the unconstrained case, the Takagi-Sugeno type fuzzy-neural model provides the state-space matrices *A, B* and *C* (the system is strictly proper, i.e. *D*=0) for the optimization procedure of the model predictive control approach. Therefore, the LIQP problem is constructed with "fresh" parameters at each sampling time and improves the adaptive features of the applied model predictive controller. It can be seen on the next figures that the disturbances, which are consequences of a sudden change of the level references, are compensated in short time without violating the proper system work.

5. Conclusions

This chapter has presented an effective approach to fuzzy model-based control. The effective modelling and identification techniques, based on fuzzy structures, combined with model predictive control strategy result in effective control for nonlinear MIMO plants. The goal was to design a new control strategy - simple in realization for designer and simple in implementation for the end user of the control systems.

The idea of using fuzzy-neural models for nonlinear system identification is not new, although more applications are necessary to demonstrate its capabilities in nonlinear identification and prediction. By implementing this idea to state-space representation of control systems, it is possible to achieve a powerful model of nonlinear plants or processes. Such models can be embedded into a predictive control scheme. State-space model of the system allows constructing the optimization problem, as a quadratic programming problem. It is important to note that the model predictive control approach has one major advantage – the ability to solve the control problem taking into consideration the operational constraints on the system.

This chapter includes two simple control algorithms with their respective derivations. They represent control strategies, based on the estimated fuzzy-neural predictive model. The two-stage learning gradient procedure is the main advantage of the proposed identification procedure. It is capable to model nonlinearities in real-time and provides an accurate model for MPC optimization procedure at each sampling time.

The proposed consequent solution for unconstrained MPC problem is the main contribution for the predictive optimization task. On the other hand, extraction of a "local" linear model, obtained from the inference process of a Takagi–Sugeno fuzzy model allows treating the nonlinear optimization problem in presence of constraints as an LIQP.

The model predictive control scheme is employed to reduce structural response of the laboratory system - multi tank system. The inherent instability of the system makes it difficult for modelling and control. Model predictive control is successfully applied to the studied multi tank system, which represents a multivariable controlled process. Adaptation of the applied fuzzy-neural internal model is the most common way of dealing with plant's nonlinearities. The results show that the controlled levels have a good performance, following closely the references and compensating the disturbances.

The contribution of the proposed approach using Takagi–Sugeno fuzzy model is the capacity to exploit the information given directly by the Takagi–Sugeno fuzzy model. This approach is very attractive for systems from high order, as no simulation is needed to obtain the parameters for solving the optimization task. The model's state-space matrices can be

generated directly from the inference of the fuzzy system. The use of this approach is very attractive to the industry for practical reasons related with the capacity of this model structure to combine local models identified in experiments around the different operating points.

6. Acknowledgment

The authors would like to acknowledge the Ministry of Education and Science of Bulgaria, Research Fund project BY-TH-108/2005.

7. References

Ahmed S., M. Petrov, A. Ichtev (July 2010). Fuzzy Model-Based Predictive Control Applied to Multivariable Level Control of Multi Tank System. *Proceedings of 2010 IEEE International Conference on Intelligent Systems (IS 2010)*, London, UK. pp. 456 - 461.

Ahmed S., M. Petrov, A. Ichtev, "Model predictive control of a laboratory model – coupled water tanks," in *Proceedings of International Conference Automatics and Informatics'09*, October 1–4, 2009, Sofia, Bulgaria. pp. VI-33 - VI-35.

Åkesson Johan. *MPCtools 1.0 – Reference Manual*. Technical report ISRN LUTFD2/TFRT--7613--SE, Department of Automatic Control, Lund Institute of Technology, Sweden, January 2006.

Camacho E. F., C. Bordons (2004). Model Predictive Control (Advanced Textbooks in Control and Signal Processing). Springer-Verlag London, 2004.

Espinosa J., J. Vandewalle and V. Wertz. *Fuzzy Logic, Identification and Predictive Control. (Advances in industrial control)*. © Springer-Verlag London Limited, 2005.

Fletcher R. (2000). *Practical Methods of Optimization*. 2nd.ed., Wiley, 2000.

Inteco Ltd. (2009). *Multitank System - User's Manual*. Inteco Ltd., http://www.inteco.com.pl.

Lee, J.H.; Morari, M. & Garcia, C.E. (1994). State-space interpretation of model predictive control, *Automatica*, 30(4), pp. 707-717.

Maciejowski J. M. (2002). *Predictive Control with Constraints*. Prentice Hall Inc., NY, USA, 2002.

Martinsen F., Lorenz T. Biegler, Bjarne A. Foss (2004). A new optimization algorithm with application to nonlinear MPC, *Journal of Process Control,* vol.14, pp 853–865, 2004.

Mendonça L.F., J.M. Sousa J.M.G. Sá da Costa (2004). Optimization Problems in Multivariable Fuzzy Predictive Control, *International Journal of Approximate Reasoning,* vol. 36, pp. 199–221, 2004 .

Mollov S, R. Babuska, J. Abonyi, and H. Verbruggen (October 2004). Effective Optimization for Fuzzy Model Predictive Control. *IEEE Transactions on Fuzzy Systems*, Vol. 12, No. 5, pp. 661 – 675.

Petrov M., A. Taneva, T. Puleva, S. Ahmed (September, 2008). Parallel Distributed Neuro-Fuzzy Model Predictive Controller Applied to a Hydro Turbine Generator. *Proceedings of the Forth International IEEE Conference on "Intelligent Systems"*,

Golden Sands resort, Varna, Bulgaria. ISBN 978-1-4244-1740-7, Vol. I, pp. 9-20 - 9-25.

Petrov M., I. Ganchev, A. Taneva (November 2002). Fuzzy model predictive control of nonlinear processes. *Preprints of the International Conference on "Automation and Informatics 2002"*, Sofia, Bulgaria, 2002. ISBN 954-9641-30-9, pp. 77-80.

Rossiter J.A. (2003). *Model based predictive control – A practical Approach*. CRC Press, 2003.