Neural Network-Based Model Predictive Control: Fault Tolerance and Stability

Krzysztof Patan

***Abstract*— This brief deals with nonlinear model predictive control designed for a tank unit. The predictive controller is realized by means of a recurrent neural network, which acts as a one-step ahead predictor. Then, based on the neural predictor, the control law is derived solving an optimization problem. An important issue in control theory is stability of the control system. In this brief, this problem is investigated by showing that a cost function is monotonically decreasing with respect to time. The derived stability conditions are then used to redefine a constrained optimization problem in order to calculate a control signal. As the automatic control system can prevent faults from being observed, the control system is equipped with a fault diagnosis block. It is realized by means of a multivalued diagnostic matrix, which is determined on the basis of residuals calculated using a set of partial models. Each partial model is designed in the form of a recurrent neural network. This brief proposes also a methodology of compensating sensor, actuator, as well as process faults. When a sensor fault is isolated, the system estimates its size and, based on this information, the controller is fed with a determined, close to real, tank level value. Actuator and process faults can be compensated due to application of an unmeasured disturbance model.**

***Index Terms*— Fault detection and isolation, fault tolerant control, nonlinear control systems, recurrent neural networks, stability.**

I. Introduction

MODEL predictive control (MPC) has been the subject of intensive research for the last three decades [1], [2]. This research effort has succeeded in many practical applications [3]–[5]. The attractiveness of predictive control algorithms comes from their natural ability to consider process and technological constraints imposed on input, output, or state variables. The second very important reason is that the operating principles are understandable and relatively easy to explain to practitioners, which seems to be a crucial aspect during implementation of a new control scheme in industry. Therefore, predictive control algorithms are widely used in petrochemical and related industries, where satisfaction of constraints is particularly important.

An important part of MPC strategies is a model of the controlled process used to predict future plant outputs based on past and current outputs as well as future control signals. These are calculated through minimization of the cost function taking into account the constraints. In many cases, constraints possess the form of inequalities imposed on process variables, e.g., input saturations. If such nonlinearities are not taken into account, it may result in degraded performance of closed-loop control and can lead to stability problems. Therefore, stability of the control system is a fundamental requirement. As pointed out in [6] as well as [7], predictive control theory has reached a high level of maturity, especially for linear systems. However, there are still a number of difficulties when dealing with nonlinear systems. Problems are still observed with the modeling of nonlinear processes, state estimation, fault diagnosis, or fault tolerant control (FTC) [6].

To date, many methods for nonlinear system identification have been derived, e.g., Volterra series, Group Method and Data Handling models, Wiener and Hammerstein models, nonlinear autoregressive models, and dynamic neural networks [8], [9]. Undoubtedly, neural networks are the most popular models used. They are very flexible, universal, and willingly used in cases when there is no accurate mathematical model of the process. That is the reason why recurrent neural networks have been used to design the one-step ahead predictor, which is then successively employed to obtain k-step ahead prediction of the plant output. Then, the control law is derived solving a constrained optimization problem using a modified Newton algorithm. Although the concept of neural network-based MPC is not new [10], results connected with stability of MPC based on input–output neural network models are rather scarce. Therefore, stability of the proposed predictive scheme is investigated by showing that a cost function is monotonically decreasing with respect to time. The derived stability conditions are then used to redefine a constrained optimization problem in order to calculate control.

In many situations, an automatic control system can prevent faults from being observed. Therefore, it is advisable to equip modern control systems with self-diagnostic mechanisms in order to manage abnormal situations in an automatic fashion. Fortunately, there is an exhaustive literature about model-based fault diagnosis [8], [11]. In [12], a fault detection and isolation subsystem derived by means of a binary diagnostic matrix was proposed. Unfortunately, such a prototype solution has one disadvantage. Some faults, due to the same fault signature, were indistinguishable. In this brief, the solution proposed in [12] is further developed in order to increase fault distinguishability. Improvements concern application of a multivalued diagnostic matrix (MDM) as well as fault size estimation to definitely distinguish as many faults as possible.

Manuscript received February 12, 2014; revised June 18, 2014; accepted August 10, 2014. Date of publication September 22, 2014; date of current version April 14, 2015. Manuscript received in final form September 2, 2014. This work was supported by the National Science Centre, Kraków, Poland, under Grant N-N514-6784-40 through the Project entitled Tolerant Nonlinear Predictive Control Systems. Recommended by Associate Editor A. Alessandri.

The author is with the Institute of Control and Computation Engineering, University of Zielona Góra, Zielona Góra 65-424, Poland (e-mail: k.patan@issi.uz.zgora.pl).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCST.2014.2354981

Nowadays, great effort is observed in designing control systems, which are able to maintain the current performance of the system as close as possible to the desirable one, and to preserve stability conditions in the presence of unexpected changes of system work caused by faults. Fault Tolerant Control (FTC) systems have received increased attention in the last decade [12]–[14]. Moreover, MPC seems to be suitable for FTC, as in MPC, the representation of both fault and control objectives is relatively simple. Except certain robust conservative solutions, MPC is not able to handle sensor or actuator faults. In this brief, it is shown that by applying the so-called unmeasured disturbance model, it possible to achieve a zero steady-state tracking error and compensation of actuator/process faults. The sensor fault is compensated using a reconstructed value of a process variable (output of the nominal model).

Compared with our previous works, contribution of this brief includes stability of neural network-based MPC, extension of MPC formulation to systems with time-delay, applying an unmeasured disturbance model to deal with perturbations, increasing fault distinguishability using a multivalued diagnostic matrix and fault size estimation, and experiments with process and actuator faults as well as multifault scenarios.

This brief is organized as follows. In Section II, nonlinear MPC based on a neural network predictor is portrayed. In Section III, the main result concerning stability of the proposed control scheme is presented and proved. Section III contains a description of the tank unit for which the control system was designed. The idea of a fault diagnosis system for the tank unit is presented in Section V. Experimental results are shown and discussed in Section VI. Section VII contains conclusions and final remarks.

II. Nonlinear MPC

Let us define the cost criterion in the form

$$J(k) = \sum_{i=N_1}^{N_2} e^2(k+i) + \rho \sum_{i=1}^{N_u} \Delta u^2(k+i-1) \tag{1}$$

where $e(k+i) = r(k) - \hat{y}(k+i)$ is the tracking error, $r(k)$ is the reference signal at time k, $\hat{y}(k+i)$ is the prediction of future outputs, $\Delta u(k+i-1) = u(k+i-1) - u(k+i-2)$, $u(k)$ is the control signal at time k, N_1 is the minimum prediction horizon, N_2 is the prediction horizon, N_u is the control horizon, and ρ represents a factor penalizing changes in the control signal. A fundamental issue in MPC is to design an accurate model of the controlled process. Let the process be represented by the following prediction form of a nonlinear difference equation:

$$\hat{y}(k+1) = f(y(k), \ldots, y(k-n_a+1), u(k-\tau), \ldots, u(k-\tau-n_b+1)) + d(k) \tag{2}$$

where f stands for a nonlinear function, $y(k)$ is the output of the process at time k, n_a and n_b are numbers of past outputs and inputs considered by the model, respectively, τ is the time-delay, and $d(k)$ stands for the disturbance model defined as

$$d(k) = k_c \varepsilon(k) - d(k-1) \tag{3}$$

where $\varepsilon(k) = y(k) - \hat{y}(k)$, k_c represents the gain of the disturbance model. As the disturbance model (3) includes an integrator, offset-free steady-state behavior of the control system can be achieved [1], [15]. Frequently, k_c is assumed to be equal to 1 and $d(k)$ is assumed to be constant within the prediction horizon [1]. In order to design a nonlinear model of the process, artificial neural networks can be successfully used. Neural networks proved their usefulness in the modeling of nonlinear dynamic processes [8], [10]. In the field of neural modeling, the simplest solution is to use feedforward networks with external dynamics

$$f(\boldsymbol{x}) = \sigma_o(\boldsymbol{W}_2 \sigma_h(\boldsymbol{W}_1 \boldsymbol{x} + \boldsymbol{b}_1) + \boldsymbol{b}_2) \tag{4}$$

where $\boldsymbol{x} = [y(k), \ldots, y(k-n_a+1), u(k), \ldots, u(k-n_b+1)]^T$, and $\boldsymbol{W}_1 \in \mathbb{R}^{n_a+n_b \times \upsilon}$ and $\boldsymbol{W}_2 \in \mathbb{R}^{\upsilon \times 1}$ are weight matrices of hidden and output layers, respectively, $\boldsymbol{b}_1 \in \mathbb{R}^{\upsilon}$ and $\boldsymbol{b}_2 \in \mathbb{R}^1$ are bias vectors of hidden and output units, respectively, $\sigma_h : \mathbb{R}^{\upsilon} \to \mathbb{R}^{\upsilon}$ is the activation function of the hidden layer, and $\sigma_o : \mathbb{R}^1 \to \mathbb{R}^1$ is the activation function of the output layer. The neural model (4) represents a neural network with one hidden layer with υ hidden units. Network parameters, i.e., weight matrices and bias vectors, are determined through the learning process carried out using the series–parallel identification scheme. Applying successive recursion of the one-step ahead predictor (2), one can calculate i-step ahead output predictions. Unfortunately, process output measurements are available up to time k, then one should substitute predictions for actual measurements $y(k+i) = \hat{y}(k+i)$, $\forall i > 1$. Consequently, one needs to feed the network with past predicted outputs, and the neural network becomes the recurrent one. Therefore, the feedforward network trained using the series–parallel identification model may be insufficient. The solution is to train the model using the parallel identification model, where the network input has the form $\boldsymbol{x} = [\hat{y}(k), \ldots, \hat{y}(k-n_a+1), u(k), \ldots, u(k-n_b+1)]^T$. In general, training of recurrent models is much more complicated than feedforward models, but they perform better in cases when process output is not available. Then, feedforward models are forced to use their own past outputs to determine the output at the current time instant. The comparison of neural models trained using series–parallel and parallel identification schemes was considered in [12].

Using the cost function (1) and neural predictor based on (2), the nonlinear MPC can be defined based on the following open-loop optimization problem:

$$\boldsymbol{u}(k) \triangleq \arg\min_{\boldsymbol{u}} \boldsymbol{J} \tag{5a}$$

$$\text{s.t.} \quad e(k+N_2+j) = 0, \quad \forall j \in [1, N_c] \tag{5b}$$

$$\underline{u} \leqslant u(k+j) \leqslant \overline{u}, \quad \forall j \in [0, N_u - 1] \tag{5c}$$

$$\Delta u(k+N_u+j) = 0, \quad \forall j \geqslant 0 \tag{5d}$$

where N_c is the constraints horizon, and $\underline{u}$ and $\overline{u}$ are lower and upper control bounds. The optimization problem (5) has to be solved at each sample time giving a sequence of future controls $\boldsymbol{u}(k)$, where the first element is taken to control the process. The distinct characteristic of MPC is that the control signal is derived in such a way to achieve the desired

behavior of the control system in the subsequent N_2 time steps. Another important property is the control horizon $N_u < N_2$. Only the first N_u future controls are determined. From that point, the control is assumed to be constant (5d). Using this assumption, constraints (5d) are not put into an optimization procedure. When a nonlinear neural network is applied as a process model, output predictions are nonlinear in the control inputs. This constitutes a complex nonlinear programming problem, which should be solved in the real-time while the optimization procedure should assure fast convergence and numerical robustness. Therefore, second-order optimization algorithms are a reasonable choice. In this brief, a combination of the Newton and Levenberg–Marquardt methods was used as proposed in [10], [12], and [16]

$$\boldsymbol{u}^{(i+1)} = \boldsymbol{u}^{(i)} - \left(\boldsymbol{H}^{(i)} + \lambda^{(i)}\boldsymbol{I}\right)^{-1}\boldsymbol{G}^{(i)} \tag{6}$$

where $\boldsymbol{u}^{(i)}$ is the current iteration of the future control inputs sequence, $\boldsymbol{H}^{(i)}$ is Hessian calculated at ith iteration, $\boldsymbol{G}^{(i)}$ is the gradient vector derived at ith iteration, λ is the parameter used to assure that the matrix to be inverted is positive definite. The used algorithm is characterized by the fast convergence and numerical robustness. Moreover, what is the most important, it allows relatively to consider constraints imposed on process variables, i.e., using a penalty cost as proposed in Section III-A.

III. Stability Analysis

In general, the stability of control systems is the problem of a crucial importance in the control theory. There is a rich literature about stability of MPC of dynamic systems, both linear and nonlinear. The interested reader can find in [7] the exhaustive review of solutions for achieving stability. The popular techniques reported in the literature can be divided into two main classes. The first one uses a cost function as a Lyapunov candidate function. Among this group of methods, we can distinguish: 1) terminal constraint [17]; 2) infinite output prediction horizon [17]; 3) terminal cost function [18]; and 4) terminal constraint set methods [19]. The second class requires that the state is decreasing in some norm, e.g., $\|\cdot\|_1$ or $\|\cdot\|_\infty$ [20].

Due to interesting properties of state-space approaches, the majority of papers is devoted to systems represented in the state-space. Consequently, the cost function is based on the state. In this brief, a methodology based on Generalized Predictive Control (GPC) is investigated and solutions derived for systems represented in the state-space cannot be applied. The stability of GPC was successfully addressed in a number of papers, e.g., infinite horizon GPC (GPC$^\infty$) [21], constrained receding-horizon predictive control [22], stable GPC [23], and min-max GPC [24]. These papers established the stability of linear input/output systems. Stability is achieved imposing terminal constraints on inputs and outputs over some constraint horizon. This brief proposes predictive control of a nonlinear process using nonlinear predictor based on the dynamic neural network. The stability is investigated checking the monotonicity of the cost, similar to [21], but in the considered case, the predictor is a nonlinear, prediction horizon is of a finite value, and control horizon is not greater than the prediction horizon.

Proposition 1: The nonlinear MPC system (5) using a predictor based on (2) is asymptotically stable if the following conditions are satisfied:

1) $\rho \neq 0$;
2) $N_c = \max[n_a + 1, n_b + \tau + 1 + N_u - N_2]$, regardless the choice of N_1, N_2, and N_u.

Proof: The cost function at time k has the form

$$J(k) = \sum_{i=N_1}^{N_2} e^2(k+i) + \rho \sum_{i=1}^{N_u} \Delta u^2(k+i-1); \tag{7}$$

Let us assume that $\boldsymbol{u}(k)$ is the optimal control at time k found by an optimization procedure. Now, let us introduce the suboptimal control $\boldsymbol{u}^*(k+1)$ postulated at time $k+1$

$$\boldsymbol{u}^*(k+1) = [u(k+1), \ldots, u(k+N_u-1), u(k+N_u-1)]^T.$$

The control sequence $\boldsymbol{u}^*(k+1)$ is formed based on the control derived at time k then, assuming that unmeasured disturbances $d(k)$ are constant within the prediction horizon, predictions $y(k+i)$ derived at time $k+1$ are the same as these derived at time k. Therefore, for the suboptimal control $\boldsymbol{u}^*(k+1)$, the cost function can be defined as follows:

$$J^*(k+1) = \sum_{i=N_1+1}^{N_2+1} e^2(k+i) + \rho \sum_{i=2}^{N_u} \Delta u^2(k+i-1). \tag{8}$$

The difference of cost functions $J(k)$ and $J^*(k+1)$ can be defined as

$$J^*(k+1) - J(k) = e^2(k+N_2+1) - e^2(k+N_1) - \rho\Delta u^2(k). \tag{9}$$

Taking into account a set of terminal equality constraints (5b), it is obvious that $e(k+N_2+1) = 0$, then

$$J^*(k+1) - J(k) = -e^2(k+N_1) - \rho\Delta u^2(k) \leqslant 0. \tag{10}$$

Analyzing the prediction equation beyond the prediction horizon

$$\begin{aligned}\hat{y}(k+N_2+j) &= f(y(k+N_2+j-1), \ldots, y(k+N_2+j-n_a),\\ &\quad u(k-\tau+N_2+j-1), \ldots, u(k-\tau+N_2+j-n_b))\\ &\quad + d(k)\end{aligned} \tag{11}$$

one can conclude that tracking error equality constraints hold for all $j \geqslant 1$ if:

1) $N_c \geqslant n_a + 1$, assuming that $n_a \geqslant n_b + \tau + 1 + N_u - N_2$;
2) $N_c \geqslant n_b + \tau + 1 + N_u - N_2$, assuming that $n_a < n_b + \tau + 1 + N_u - N_2$.

Using both results, setting the constraint horizon on the value

$$N_c = \max[n_a + 1, n_b + \tau + 1 + N_u - N_2]$$

guarantees that tracking error equality constraints hold not only for $j \in [1, N_c]$, but for all $j \geqslant 1$. Moreover, looking at the definition of $\boldsymbol{u}^*(k+1)$, inequality constraints (5c) at time $k+1$ are also satisfied. Using these considerations,

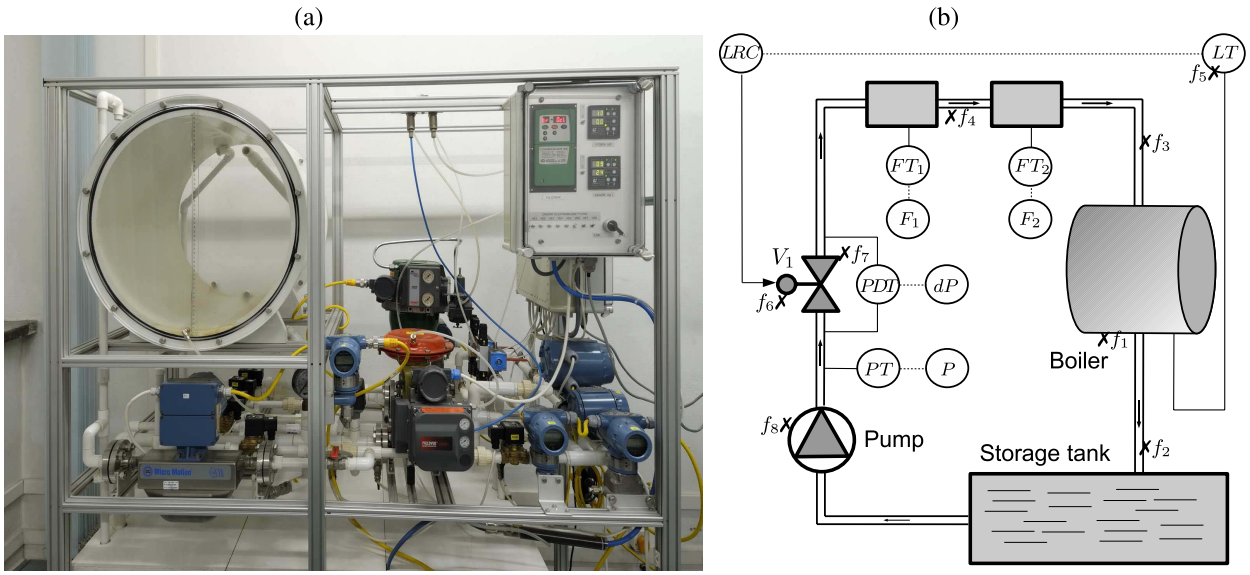

Fig. 1. (a) Real laboratory installation. (b) Block scheme of the tank unit and possible faults placement.

one can state that $\boldsymbol{u}^*(k+1)$, satisfies all constraints at time $k+1$, and subsequently the vector $\Delta\boldsymbol{u}^*(k+1)$ also satisfies constraints (5d). Furthermore, if $\boldsymbol{u}(k+1)$ is the optimal solution of the optimization problem time $k+1$, then $J(k+1) \leqslant J^*(k+1)$ as $\boldsymbol{u}^*(k+1)$, is the suboptimal one. Finally

$$\Delta J(k+1) \leqslant -e^2(k+N_1) - \rho\Delta u^2(k) \tag{12}$$

and it is clear that for $\rho \neq 0$, the cost is monotonically decreased with respect to time and control system is stable. ■

Remark 1: The presented results were achieved with the assumption that a set of feasible solutions of the optimization problem exists. As the considered MPC requires to solve a nonlinear optimization problem with constraints, the problem of feasibility is nontrivial. Feasibility considerations are not investigated in this brief, however, some numerical algorithms for solving the problem are proposed and discussed later on in this section.

A. Nonlinear Optimization With Constraints

Using a concept of Lagrangian, (5) can be solved using the first-order Karush–Kuhn–Tucker (KKT) conditions. Unfortunately, a direct analytical solution of the problem resulting from KKT conditions is extremely hard or even impossible to derive. The most popular methods for solving nonlinear programming problems are gradient projection and penalty function methods. In general, gradient projection methods possess the disadvantage that they need to maintain feasibility during the optimization. From numerical computation point of view, after a trial step, the solution have to be altered slightly to maintain feasibility. On one hand, derivation of the projection operator in the case of nonlinear constraints is a challenging problem. On the other hand, the penalty function methods make it possible to transform the original nonlinear programming problem into an unconstrained one, which can be solved using classical algorithms. A popular approach for considering constraints is to transform the original problem to its alternative unconstrained form using a penalty cost. Let us substitute $h_i(\boldsymbol{u}) = e(k+N_2+i)$ [equality constraint (5b)], and $\overline{g}_i(\boldsymbol{u}) = u(k+i) - \overline{u}$ and $\underline{g}_i(\boldsymbol{u}) = \underline{u} - u(k+i)$ [inequality constraints (5c)], then the penalty cost function can be defined as follows:

$$\bar{J}(k) = J(k) + \mu\sum_{i=1}^{N_c} h_i^2(\boldsymbol{u}) + \lambda\sum_{i=0}^{N_u-1} \overline{g}_i^2(\boldsymbol{u})S(\overline{g}_i(\boldsymbol{u})) + \lambda\sum_{i=0}^{N_u-1} \underline{g}_i^2(\boldsymbol{u})S(\underline{g}_i(\boldsymbol{u})) \tag{13}$$

where $S(x) = 1$, if $x > 0$ and $S(x) = 0$ otherwise. The function $S(x)$ makes it possible to consider a set of active inequality constraints $A(\boldsymbol{u})$ at the current iterate of the algorithm. Thus, equality constraints as well as all active inequality constraints are taken into account during the optimization. Now, the objective is to solve the following unconstrained problem:

$$\boldsymbol{u}(k) \triangleq \arg\min_{\boldsymbol{u}} \bar{J}(\boldsymbol{u}). \tag{14}$$

Now, one can use Newton-based algorithm presented in Section II to minimize the cost (13).

IV. Tank Unit

The technological process considered in this brief is the laboratory installation developed at the Institute of Automatic Control and Robotics of the Warsaw University of Technology [Fig. 1(a)]. The installation is dedicated to investigate various diagnostic methods for industrial actuators and sensors [12], [16]. The system includes a tank, a storage vessel, a control valve with a positioner, a pump, and transducers to measure process variables. The tank is realized in the form of a horizontally placed cylinder, which introduces a strong nonlinearity into the static characteristic of the system. The laboratory stand was implemented in MATLAB/Simulink software.

TABLE I
SPECIFICATION OF PROCESS VARIABLES

Variable	Specification	Range
CV	control value	0–100 %
dP	pressure difference on valve V_1	0–275 kPa
P	pressure before valve V_1	0–500 kPa
F_1	flow (electromagnetic flowmeter)	0–5 m^3/h
F_2	flow (Vortex flowmeter)	0–5 m^3/h
L	water level in boiler	0–0.5 m

TABLE II
SPECIFICATION OF FAULTY SCENARIOS

Fault	Description	Type	Size/Unit
f_1	leakage from boiler	additive	+0.083 m^3/h
f_2	outflow choking	multiplicative	0.5
f_3	loss of internal pipe diameter	multiplicative	0.5
f_4	leakage from pipe	additive	-1.67 m^3/h
f_5	level transducer failure	additive	-0.05 m
f_6	positioner fault	multiplicative	0.7
f_7	valve head or servo-motor fault	multiplicative	0.8
f_8	pump productivity reduction	multiplicative	0.8

The simulation model was successfully verified using real data acquired from the physical installation. Fig. 1(b) shows the scheme of the tank unit with measurably available process variables marked, whereas description of the process variables are presented in Table I. Acronyms LT, FT_1, FT_2, PDT, and dP stand for transducers measuring process variables, and the abbreviation LRC denotes level regulation controller. The advantage of the simulator over the real process is that the simulator makes it possible to check the behavior of the proposed predictive control in the presence of a wide range of possible faulty scenarios. The specification of faults is presented in Table II. The considered scenarios are different, including both additive and multiplicative faults. Faults in different parts of the installation are proposed, including sensor, actuator, and component faults. Faults placement is marked in Fig. 1(b). Thus, the proposed set of faults renders it possible to investigate fault tolerance properties of the proposed predictive controller in the widest range possible.

V. FAULT DIAGNOSIS

A. Multivalued Diagnostic Matrix

A popular method of fault diagnosis are diagnostic matrices [8], [25], which define relations between symptoms and faults. In this brief, it is proposed to use the MDM, which can improve fault distinguishability contrary to classical binary diagnostic matrices used in [16]. Symptoms can be determined as residuals of the so-called partial models of the diagnosed process. Then, the existence of a fault causes the appearance of a diagnostic signal $s(r)$ with a value selected as follows:

$$s(r_i) = \begin{cases} 0, & \text{if } r_i \in [T_l^i, T_u^i] \\ +1, & \text{if } r_i > T_u^i \\ -1, & \text{if } r_i < T_l^i \end{cases} \quad (15)$$

TABLE III
MULTIVALUED DIAGNOSTIC MATRIX

$r \setminus f$	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
r_1	0	0	-1	-1	0	0	0	-1
r_2	-1	+1	+1	-1	-1	+1	+1	+1
r_3	0	0	-1	-1	0	0	0	-1
r_4	0	0	-1	-1	-1	-1	-1	-1
r_5	-1	+1	+1	-1	-1	+1	+1	+1

where T_u^i and T_l^i represent the upper and lower thresholds, respectively, assigned to the residual r_i. A set of partial models should cover the whole diagnosed system. For the considered tank unit, five single-input single-output partial models are considered as proposed in [16]: the pump model $\hat{P} = f_{m1}(F_1)$, tank model $\hat{L}_1 = f_{m2}(F_1)$, valve model $\hat{F}_1 = f_{m3}(dP)$, positioner model $\hat{F}_2 = f_{m4}(\mathrm{CV})$, system model $\hat{L}_2 = f_{m5}(\mathrm{CV})$, f_{m1}, f_{m2}, f_{m3}, f_{m4}, and f_{m5} stand for nonlinear mappings between input and output variables for each single partial model, respectively, $\hat{L}_1$ and $\hat{L}_2$ are estimated water levels given by the tank and system models, $\hat{F}_1$ and $\hat{F}_2$ are estimated flows given by the valve and positioner models, and $\hat{P}$ is the pressure estimated by the pump model. Each residual is sensitive to a specific set of faults. Based on the expert knowledge about the considered process, MDM represented in Table III was formed. Analyzing MDM (Table III), it is observed that residuals r_2 and r_5 provide the same information into MDM. However, they can still give important data about faults especially taking into account dynamics of the residuals. Moreover, faults still exist that cannot be isolated separately (pairs $\{f_3, f_8\}$ and $\{f_6, f_7\}$). To definitely improve fault isolation quality, the dynamics of residuals can be taken into account. The possible solution is to investigate the order of occurring symptoms [8]. Unfortunately, in the considered case, the order of the symptoms is the same for each indistinguishable pair of faults. Fortunately, it was observed that considering the intensity of a fault may be a solution of the problem

$$S(f_k) = \frac{1}{N} \sum_{i:r_i \in R(f_k)} \sum_{j=1}^{N} \frac{r_i(j)}{T_i(j)}$$
$$T_i(j) = \begin{cases} T_u^i & \text{if } r_i(j) > T_u^i \\ T_l^i & \text{if } r_i(j) < T_l^i \end{cases} \quad (16)$$

where $S(f_k)$ is the size of the fault f_k, $R(f_k)$ represents the set of residuals sensitive to the fault f_k, and N is the window length determining the residual values taken to calculate the fault intensity. Based on the above deliberations, the process of fault isolation consists of two steps. The first step is to explore the diagnostic matrix (Table III). The second step is to calculate the fault intensity to definitely isolate the fault. The second step is carried out only if faults are indistinguishable.

B. Sensor Fault Size Estimation

Any automatic control system can hide faults from being observed, especially faults of a small size. As far as for

TABLE IV
QUALITY INDEXES OF PREDICTORS

Quality index	k-step ahead predictor					
	$k=1$	$k=3$	$k=5$	$k=7$	$k=10$	$k=15$
Q_{SSE}	10.84	11.44	12.68	14.64	18.61	26.46
Q_{MSE}	$5.42\cdot10^{-4}$	$5.72\cdot10^{-4}$	$6.34\cdot10^{-4}$	$7.32\cdot10^{-4}$	$9.3\cdot10^{-4}$	$1.3\cdot10^{-3}$

actuator or process faults, the predictive control may work in a required fashion, sensor faults should be treated in a different way. For example, if we consider a fault occurring in the sensor measuring the water level in the tank, the control system immediately reacts on the change in the system output, which is the direct consequence of the wrongly measured water level in the boiler. In that case, a control signal value should be kept the same as before the fault occurrence, because such a fault does not change the water level in the boiler. Thus, estimation of the size of a sensor fault is very important during the designing of fault tolerant predictive control.

In order to calculate the size of a sensor fault, the set of residuals sensitive to the fault f_k ($R(f_k)$) should be further reduced to the subset $\bar{R}(f_k)$ consisting of residuals, which are defined using process variable measured by the isolated sensor. Thus, the size of the fault can be directly calculated numerically as the value of a residual sensitive to the fault f_k, when only one residual is sensitive to this fault or as the mean value of all residuals belonging to $\bar{R}(f_k)$, if the set $\bar{R}(f_k)$ contains more than one residual.

VI. EXPERIMENTS

A. Modeling

To build a model of the process, a recurrent neural network presented in Section II was applied, where the model input was the control value (CV) and the model output was the level in the boiler (L). The modeling was carried out in the open-loop control feeding the system with the random steps with levels from the interval $(0, 100)$ contaminated by the white noise with the magnitude of 10. The problem of proper selection of the input signal was discussed in [12]. The model structure was selected taking into account a compromise between the model complexity and quality [26]. After some trials, the best neural model was found as $\upsilon = 7$, $n_a = 2$, $n_b = 2$, and $\tau = 0$. Hidden neurons consist of hyperbolic tangent activation function and the output neuron has the linear activation function. Sum of squared errors (Q_{SSE}) as well as the mean squared error (Q_{MSE}) for selected k-step ahead predictors are presented in Table IV. Although it is observed that the uncertainty of the one-step ahead predictor affects recursively k-step ahead predictors developed, the 15-step ahead predictor mimics the behavior of the process pretty well (see quality indexes in Table IV).

B. Control

The controller parameters are set as follows: 1) $N_1 = 1$; 2) $N_2 = 15$; 3) $N_u = 2$; and 4) $\rho = 2\cdot10^{-5}$. Such experiment settings were found using trial and error procedure and assure a pretty good performance of the control system. Second, the control sequence is constrained with the upper control bound $\overline{u} = 100$ (the maximal value generated by the actuator) and the lower control bound $\underline{u} = 20$ to cope with the problem of dead-zone of the boiler (when the control signal has the value <40, the outflow is greater than the inflow). In the view of Proposition 1, the constraint horizon $N_c = 3$. The penalty coefficients of the cost function (13) were set as follows: 1) $\mu = 1$ and 2) $\lambda = 1$. Additionally, the disturbance model (3) is used to consider unmeasured disturbances affecting the system. The disturbance model introduces some kind of robustness into the control system as shown later on in this brief. After some trials, the value of parameter k_c was set to 0.4. With this setting, the pretty good convergence of tracking error to zero can be achieved. The performance of the proposed stable predictive control in the presence of disturbance (white noise with zero mean and standard deviation $\sigma = 0.001$) is shown in Fig. 2. The plant output follows the reference almost immediately [Fig. 2(a)]. Introducing terminal equality constraints causes that the control signals has a smooth nature [Fig. 2(b)]. Tracking the reference is carried out with a pretty high quality ($Q_{\mathrm{SSE}} = 73.56$). Even more intensive disturbance (white noise with $\sigma = 0.01$) does not deteriorate the control quality significantly ($Q_{\mathrm{SSE}} = 75.30$). The proposed control scheme is also compared with the proportional-integral-differential (PID) controller. For the PID controller, the control quality was $\sim$7% is the difference between the control quality of stable MPC and the quality of the PID controller could not guarantee zero tracking error in steady-state when setpoint is changed. The important problem here is to select proper values of μ and λ. Using larger values for these penalty parameters assures an acceptable approximation to the solution of optimization problem (14). However, large values of μ and λ causes that penalty terms dominate the cost and the control quality has not such an important impact on the optimal solution. Too large values of μ and λ can give rise to some overshots in the plant response. Therefore, proper selection of these parameters is only a compromise between the control quality and optimality of the control sequence. In this brief, stability of the control system is also investigated. Due to application of terminal constraints, the stability of the loop and monotonicity of the cost function J is observed, which confirms that the proposed predictive control systems guarantees the stability [Fig. 2(c)]. The fast convergence of the cost to zero can be assured by relatively large values of penalty factors λ and μ. However, as pointed out previously, too large values of μ and λ can cause an overshot in the plant response and, consequently, some perturbations to the monotonicity of the cost.

C. Fault Diagnosis

Partial models were designed by means of the recurrent neural network (Section II). First, modeling was carried out in the open-loop control, however, the quality of achieved in this way models tested in the closed-loop control was not satisfactory. Therefore, modeling was performed in the closed-loop control. Training data was recorded during the normal work of the system using stable predictive controller with settings presented in Section VI-B. The reference signal

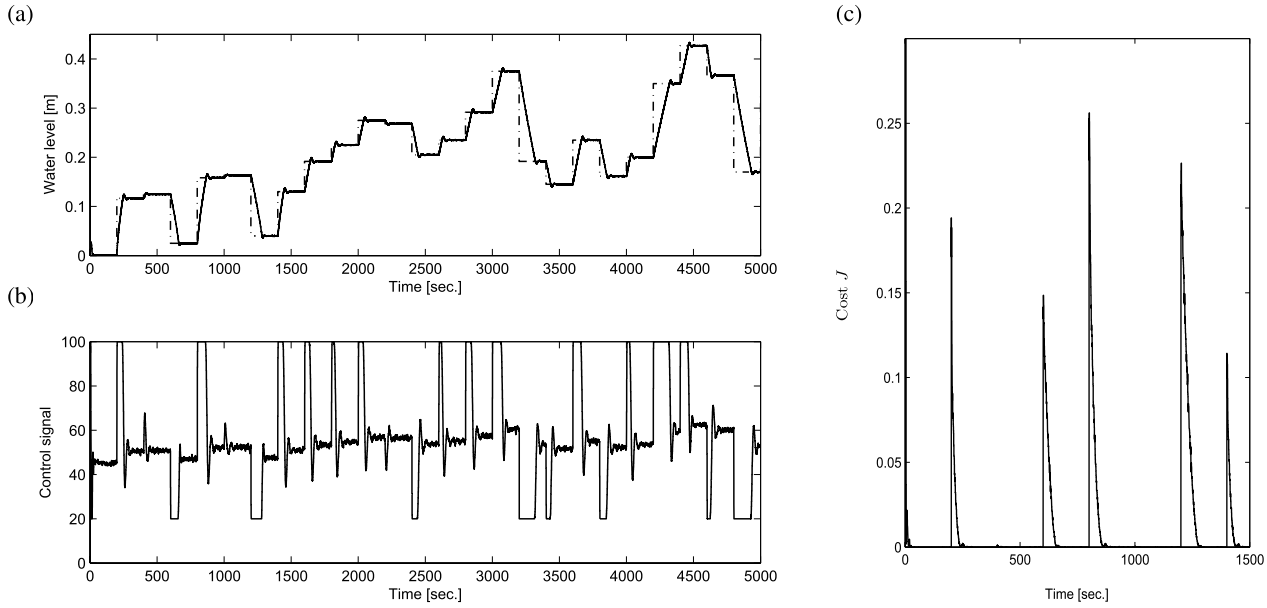

Fig. 2. (a) Process output (solid line) and reference signal (dashed line). (b) Control signal. (c) Evolution of the cost function J.

TABLE V
MODELS SPECIFICATION

Model	v	n_a	n_b	σ_h	σ_o	t_s[s]	T_l	T_u
pump model	3	1	1	tanh	linear	0.05	-0.0072	0.0178
boiler model	10	2	2	tanh	linear	0.05	-0.0023	-0.000195
valve model	5	2	1	tanh	linear	0.05	-0.0166	0.007
positioner model	5	1	1	tanh	linear	0.05	-0.0245	0.0093

was used in the form of the random steps sequence with levels from the interval $(0, 0.5)$; each step lasted 200 s. Due to large values, both the pressure before valve P and the pressure difference dP signals were scaled linearly to fall into the range $[0, 1]$. Structures of neural models were selected experimentally using the same criteria as presented in Section VI-A. All neural models had only one hidden layer. The best models specification is presented in Table V. For each model, the time-delay τ was set to zero.

Fault diagnosis is done analyzing residuals calculated using partial models. Thresholds values derived according to three-standard deviation rule, assigned to each partial model needed to decision making are presented in the last two columns of Table V. The number of false alarms caused by temporary fault diagnosis should be as small as possible. Thus, in order to determine the quality of fault detection, detection time t_d is used, defined as a period of time measured from the fault start-up time to a permanent, true decision about a fault. To be sure that a residual permanently exceeds thresholds, a time window with the length 1 s was used. If the residual exceeds a threshold for a period of time >1 s, then a fault is detected and signaled. Such a procedure is applied to each residual. Results of fault diagnosis for the considered set of faults are presented in Table VI. Each fault was introduced in the experiment at 500 s. Analyzing the results, it is obvious, that using the proposed approach, all faults were surely detected. The second

TABLE VI
FAULT DIAGNOSIS RESULTS

Index	**Fault scenario**							
	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
$t_d[s]$	11.45	2.3	1	1	1	1	1	1
$t_i[s]$	11.45	9.05	20.4	1	2.85	1	1	3.05
isolated as	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
$S(f_k)$	–	–	337.65	–	–	42	32.83	110.11

stage is fault isolation. If a fault is detected, the diagnostic system tries to localize it. This is done simply by searching the diagnostic table (Table III), in order to find a match for the fault signature. The fault signature is said to be permanent if it does not change within a time window. Then, similar to the case of fault detection to isolate faults, a time window with the length 1 s was applied. The fault isolation quality is determined by means of the so called isolation time t_i, which is defined as a period of time measured from the beginning of fault start-up time to the moment of fault isolation. Results of fault isolation are presented in the third row of Table VI. The isolation of some faults can be troublesome. Even if a fault was detected relatively fast, its isolation can take a long time, e.g., the fault f_3. Some faults are not distinguishable. In the case study considered, faults f_3 and f_8 as well as f_6 and f_7 possess the same fault signature, i.e., compare the third column with

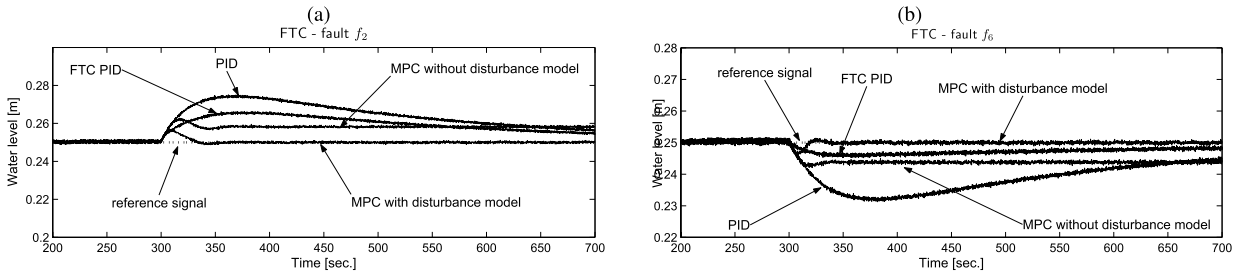

Fig. 3. Fault tolerance. (a) Fault f_2. (b) Fault f_6.

TABLE VII
FAULT TOLERANCE RESULTS

Method	**Fault scenario**									
	f_1	f_2	f_4	f_6	f_7	f_8	f_2+f_4	f_2+f_7	f_4+f_7	f_7+f_8
Q_{SSE} – MPC	0.00024	0.00865	0.00389	0.0499	0.0166	0.00127	–	–	–	–
Q_{SSE} – MPCD	0.00024	0.00459	0.00039	0.0013	0.00057	0.00026	0.00484	0.00476	0.00187	0.0016
Q_{SSE} – PID	0.00027	0.3002	0.05089	0.2218	0.0748	0.00605	–	–	–	–

the eighth one, and seventh and eighth columns in Table VI. This problem can be solved using a fault intensity, as discussed in Section V-A. The fault intensity is calculated using the same time window as in the case of both fault detection and fault isolation. Exemplary fault intensities are presented in the last row of Table VI. It is clear that using the fault intensity, one can definitely distinguish f_3 from f_8, and f_6 from f_7, and vice versa.

D. Fault Tolerance

Table VII includes fault tolerance results for MPC without disturbance model (the first row—MPC), MPC with disturbance model (the second row—MPCD), and classical PID (the third row—PID). Each considered fault was simulated at 300 s and the experiment lasted 700 s. For comparison, for the nominal work of the control system, $Q_{\mathrm{SSE}} = 0.0002$. Two examples of fault tolerant control are shown in Fig. 3(a) (fault f_2), and Fig. 3(b) (fault f_6). The only fault that was not compensated is f_3. In that case, it was observed that the control system tried to keep the required level of water but after ∼80 s from the fault start-up time, due to limited maximum inflow to the tank, which is less than the outflow, the fault effect cannot be compensated in any way. For MPC without the disturbance model, the fault effect changed the dynamic properties of the plant, whereas the predictive controller relies on the model developed for normal operating conditions. In such cases, prediction of the output can be inaccurate and some error in the steady-state can be observed (Fig. 3). Much better behavior is observed in the case of MPC, which uses the disturbance model. As can be seen in Table VII, almost all process/actuator faults (excluding f_3) are compensated with steady-state error near zero. The PID controller is able to somehow compensate faults but the compensation time is quite long (Fig. 3). The PID controller does not response for the changing conditions of the system as fast as predictive controller, therefore quality indexes for the PID controller presented in Table VII are not as good as for the MPCD controller.

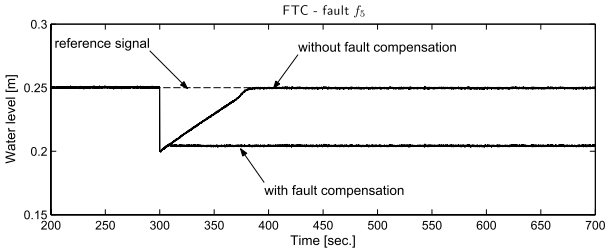

Fig. 4. Fault tolerance. Fault f_5.

The proposed control scheme is also compared with a different fault tolerant approach of the passive type [13], [26], where reconfiguration of the controller is simply done by switching between already designed PID controllers. Each controller is designed for one of the predefined faulty situation. The work of such a FTC system is illustrated in Fig. 3 (curves marked as FTC PID). FTC PID demonstrates better performance than the fixed PID controller, simultaneously works worser than the proposed MPCD controller.

The multiple fault case is also investigated. Two faults were simulated in a row, the first one at 300 s and the second one at 350 s. Results carried out only for the MPCD controller are presented in the last four columns of Table VII. Even in the case of multiple faults, the control system is able to operate properly, and faults are relatively and quickly compensated.

In the case of the sensor fault f_5, a control signal should be kept the same as before fault occurrence, because such a fault does not change the water level in the tank. Reconstruction of the water level (discussed in Section V-B) is done using the estimated level $\hat{L}$ at the moment of sensor fault isolation. In this brief, the estimated water level in the tank is derived using the tank model ($\hat{L}_1 = f_{m2}(F_1)$). Then, the difference (dL) between the measured boiler level and output of the tank model at the moment of fault isolation was equal to $dL = -0.0459$. System behavior in the case of sensor fault compensation is depicted in Fig. 4. It is observed that the

wrongly measured value of the tank level is changed to a proper value and transferred to the predictive controller, not affecting the control signal in a significant way. Consequently, the predictive controller does not try to increase the water level up to the reference value.

VII. Conclusion

This brief describes a stable nonlinear MPC scheme equiped with a fault diagnosis subsystem. The predictive controller was constructed using the model of the process based on the recurrent neural network. It should be pointed out that the neural model of the plant works pretty well. Stability of the proposed nonlinear predictive control scheme was proven checking the monotonicity of the cost function. Based on the derived stability conditions, an optimization method was proposed, which can relatively and simply consider the constraints imposed on process variables. Practical verification of the solutions confirms that the proposed algorithm works well. In spite of proper work of the developed control algorithms, selecting the control parameters is still a challenge. Moreover, searching for faster and more robust nonlinear optimization algorithms with inequality constraints handling is also expected.

This brief also proposes a methodology to design a fault diagnosis subsystem with the intention to build an automated fault tolerant control system. The proposed fault diagnosis subsystem was realized by means of a MDM, which was determined on the grounds of residuals calculated using a set of partial models. It is necessary to point out that, to make a decision about a fault, only models working at normal operating conditions are required. In addition, in this field, recurrent neural networks proved their usefulness. Thus, the proposed solution can be used in the engineering practice as data representing a fault is not needed. Furthermore, the problem of fault distinguishability is also solved using nonparametric fault size estimation. Fault diagnosis can be even improved by application of more sophisticated decision making techniques, e.g., adaptive thresholds or robust fault diagnosis methods.

Fault tolerance of a control system was also investigated. Because of the application of an unmeasured disturbances model, all component/actuator faults were reliable compensated. The only fault that was not compensated was f_3. However, f_3 is an example of potentially dangerous faults (called failures), which cannot be compensated in any way due to the limited maximum inflow to the tank, which is less than the outflow. In addition, because of the fault diagnosis subsystem, the sensor fault was reliably diagnosed, which results in estimating its size. Based on this information, the supervisory unit fed the controller with the determined, close to true, tank level.

References

[1] E. F. Camacho and C. B. Alba, *Model Predictive Control*. Berlin, Germany: Springer-Verlag, 2004.

[2] J. M. Maciejowski, *Predictive Control: With Constraints*. Essex, U.K.: Prentice-Hall, 2002.

[3] I. Prodan, S. Olaru, C. Stoica, and S.-I. Niculescu, "Predictive control for trajectory tracking and decentralized navigation of multi-agent formations," *Int. J. Appl. Math. Comput. Sci.*, vol. 23, no. 1, pp. 91–102, 2013.

[4] A. Koerber and R. King, "Combined feedback–feedforward control of wind turbines using state-constrained model predictive control," *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 4, pp. 1117–1128, Jul. 2013.

[5] A. Alessandri, C. Cervellera, and M. Gaggero, "Predictive control of container flows in maritime intermodal terminals," *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 4, pp. 1423–1431, Jul. 2013.

[6] M. Morari and J. H. Lee, "Model predictive control: Past, present and future," *Comput. Chem. Eng.*, vol. 23, no. 4, pp. 667–682, 1999.

[7] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, Jun. 2000.

[8] J. Korbicz, J. Kościelny, Z. Kowalczuk, and W. Cholewa, Eds., *Fault Diagnosis: Models, Artificial Intelligence, Applications*. Berlin, Germany: Springer-Verlag, 2004.

[9] K. Patan, "Approximation of state-space trajectories by locally recurrent globally feed-forward neural networks," *Neural Netw.*, vol. 21, no. 1, pp. 59–64, 2008.

[10] M. Nørgaard, O. Ravn, N. Poulsen, and L. Hansen, *Networks for Modelling and Control of Dynamic Systems*. London, U.K.: Springer-Verlag, 2000.

[11] R. Isermann, *Fault-Diagnosis Systems. An Introduction From Fault Detection to Fault Tolerance*. New York, NY, USA: Springer-Verlag, 2006.

[12] K. Patan and J. Korbicz, "Nonlinear model predictive control of a boiler unit: A fault tolerant control study," *Int. J. Appl. Math. Comput. Sci.*, vol. 22, no. 1, pp. 225–237, 2012.

[13] M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki, *Diagnosis and Fault-Tolerant Control*. Berlin, Germany: Springer-Verlag, 2006.

[14] M. Staroswiecki, H. Yang, and B. Jiang, "Progressive accommodation of parametric faults in linear quadratic control," *Automatica*, vol. 43, no. 12, pp. 2070–2076, 2007.

[15] P. Tatjewski, "Disturbance modeling and state estimation for offset-free predictive control with state-space process models," *Int. J. Appl. Math. Comput. Sci.*, vol. 24, no. 2, pp. 313–323, 2014.

[16] K. Patan and J. Korbicz, "Sensor fault estimation in the framework of model predictive control. Boiler case study," in *Proc. 8th Int. Symp. Fault Detection, Supervis. Safety Tech. Process. (SAFEPROCESS)*, 2012, pp. 403–408.

[17] S. S. Keerthi and E. G. Gilbert, "Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems: Stability and moving-horizon approximations," *J. Optim. Theory Appl.*, vol. 57, no. 2, pp. 265–293, 1988.

[18] J. B. Rawlings and K. R. Muske, "The stability of constrained receding horizon control," *IEEE Trans. Autom. Control*, vol. 38, no. 10, pp. 1512–1516, Oct. 1993.

[19] P. O. M. Scokaert, D. Q. Mayne, and J. B. Rawlings, "Suboptimal model predictive control (feasibility implies stability)," *IEEE Trans. Autom. Control*, vol. 44, no. 3, pp. 648–654, Mar. 1999.

[20] A. Bemporad, "A predictive controller with artificial Lyapunov function for linear systems with input/state constraints," *Automatica*, vol. 34, no. 10, pp. 1255–1260, Oct. 1998.

[21] P. Scokaert and D. W. Clarke, "Stabilizing properties of constrained predictive control," *IEE Proc. Control Theory Appl.*, vol. 141, no. 5, pp. 295–304, 1994.

[22] D. W. Clarke and R. Scattolini, "Constrained receding-horizon predictive control," *IEE Proc. D, Control Theory Appl.*, vol. 138, no. 4, pp. 347–354, 1991.

[23] J. R. Gossner, B. Kouvaritakis, and J. A. Rossiter, "Stable generalized predictive control with constraints and bounded disturbances," *Automatica*, vol. 33, no. 4, pp. 551–568, 1997.

[24] Y. H. Kim, W. H. Kwon, and Y. I. Lee, "Min–max generalized predictive control with stability," *Comput. Chem. Eng.*, vol. 22, no. 12, pp. 1851–1858, 1998.

[25] J. M. Kościelny, M. Bartyś, and M. Syfert, "Method of multiple fault isolation in large scale systems," *IEEE Trans. Control Syst. Technol.*, vol. 20, no. 5, pp. 1302–1310, Sep. 2012.

[26] A. Czajkowski, K. Patan, and M. Szymański, "Application of the state space neural network to the fault tolerant control system of the PLC-controlled laboratory stand," *Eng. Appl. Artif. Intell.*, vol. 30, pp. 168–178, Apr. 2014.