manufacturing
industrial
engineering

Journal of Manufacturing and Industrial Engineering (MIE)

Scientific and professional journal of Faculty of Manufacturing Technology of Technical University of Košice with a seat in Prešov

Intelligent Positioning Plate Predictive Control and Concept of Diagnosis System Design

Matej Oravec[1]✉ - Anna Jadlovská[2]

[1]Department of Cybernetics and Artificial Intelligence, Faculty of Electrical Engineering and Informatics, TU Košice , Email: matej.oravec@tuke.sk
[2]Department of Cybernetics and Artificial Intelligence, Faculty of Electrical Engineering and Informatics, TU Košice , Email: anna.jadlovska@tuke.sk

Keywords

predictive control design, ARX model of the system, mechatronic system, diagnosis system

Abstract

This paper presents design of the predictive control algorithms, which are verified using the simulation and laboratory model of the Intelligent Positionig Plate. The results of the predictive control algorithm verification are presented also in this article. The created tool called the *IPPtools* is based on the designed predictive control algorithms and it is shortly presented. A part of the paper is dedicated to the concept of the diagnosis system, which is designed and implemented into the 5-level Distributed Control System of the Department of the Cybernetics and Artificial Intelligence. Possibility how to modify the predictive control algorithms using diagnosis system is also stated in the last section of this paper.

Article

History: Received 08 March 2017 / Revised 19 March 2017 / Accepted 05 May 2017

Category: Original Scientific Paper

Citation: Matej Oravec, Anna Jadlovská (2017) Intelligent Positioning Plate Predictive Control and Concept of Diagnosis System Design. Journal of Manufacturing and Industrial Engineering, 15(1-2):1-9, http://dx.doi.org/10.12776/mie.v15i1-2.895

Introduction

Predictive control methods are nowadays widely used for the control of the various dynamic systems. Methods of the predictive control provide highly efficient control of the dynamic systems, which is able to control during long periods of the time with hardly any intervention. Disadvantage of these predictive control methods is more complex derivation than that of the classical PID controllers, but the resulting control law is easy to implement. Different approaches to model predictive control design are presented in books [1],[2].

At present, several research groups deal with the predictive control of mechatronic systems. In the [3], [4] authors present the results of the design and verification of the predictive control algorithms for the *Ball and Beam* system. The papers [13], [14] present the predictive control of the *Ball and Plate* system using Internet-based control. The various methods of the *Ball and Plate* system control such as fuzzy or sliding mode control, which are used for the reference trajectory tracking goal, are presented in [15], [16].

In general, the model application called *Intelligent Positioning Plate* (shortly *IPP*) is very similar to the *Ball and Plate* system presented in [15], [16]. The *IPP* model application belongs to fund of the laboratory models of the Department of Cybernetics and Artificial Intelligence (DCAI), FEEI, TU and it is situated in the *Laboratory of Modern Control Techniques of Physical Systems* (http://kyb.fei.tuke.sk/laben/modely/gnk.php). The *IPP* model application is specific by its construction and consists of two servomotors, which communicate with the control PC via serial link with the single – microchip computer. The actual ball position is captured by the camera.

This paper is focused on the selected approaches to predictive control using linear predictor for prediction of the dynamic system future behavior. According to the selected predictive control approaches are designed predictive control algorithms, which are implemented into the Matlab environment and verified using the *IPP* model application.

The paper follows the *Center of Modern Control Techniques and Industrial Informatics* (http://kyb.fei.tuke.sk/laben/) members previous publications. The paper [10] presents results of the predictive control of the *Humusoft Helicopter CE150* laboratory model. Some obtained results of the LQ control of the *IPP* model application was listed in [11].

The one section of the paper is related to the tasks solved within the project *"Research and Development Operational Program for project: University Science Park TECHNICOM for innovative applications with knowledge technology support"* with subactivity *"Center for Nondestructive Diagnostics of Technological Processes"* and is oriented to the concept design of the diagnosis system within the infrastructure of *Distributed Control System* (*DCS*) within the DCAI, FEEI, TU.

Model application of the Intelligent positioning plate

The *Intelligent positioning plate* (*IPP*) represents an intelligent positioning mechatronic system based on the concept of position control of an object moving on an adjustable plate (Fig. 1). This application provides a platform for mathematical modeling and subsequent simulation of the mechatronic system in *Matlab*/*Simulink*, but also for the implementation of designed control algorithms with the fault diagnosis algorithms realized via the technological PC or the PLC [9].

The *IPP* model application (http://kyb.fei.tuke.sk/laben/modely/gnk.php) consists of the two servomotors and the plate with a ball whose position coordinates on the plate are determined using a fixed camera. The camera is connected directly to control PC where image processing algorithm is implemented (Fig. 2) for calculating of the ball position coordinates in programming language *C#* using *emguCV* libraries [11]. Communication between servomotors and the control PC is realized via a microcontroller through serial connection. The specific design solution of the model enables the implementation and verification of proposed control algorithms in various programming languages (C/C++/C#) or simulation tools (*Matlab/Simulink*) [9].

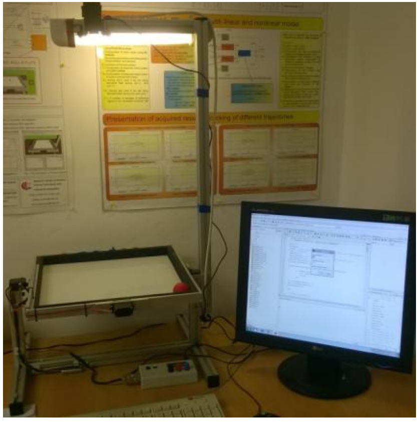

Figure 1 *Workplace of the Intelligent positioning plate model application*

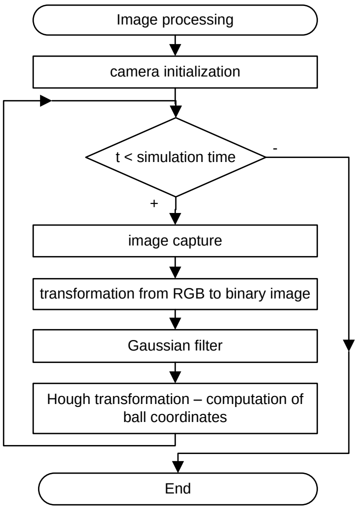
Image processing
camera initialization
t < simulation time
-
+
image capture
transformation from RGB to binary image
Gaussian filter
Hough transformation – computation of ball coordinates
End

Figure 2 *Flowchart of the image processing algorithm for computation of the ball position coordinates*

Inputs for the servomotors and for whole system are reference angles $\alpha_x(t)$, $\beta_y(t)$ of the plate tilt. Outputs of the model application are coordinates *x(t)*, *y(t)* of the ball position on plate. Actual tilt of the plate is represented by angles *α(t)* and *β(t)*. Physical variables and parameters of the *IPP* model application are listed in Tab. 1, Tab. 2.

Table 1 *Parameters of the IPP model application*

Description	Label	
servomotor gain – axis *x*	K_x	-
time constant of servomotor – axis *x*	T_x	[*s*]
servomotor gain – axis *y*	K_y	-
time constant of servomotor – axis *y*	T_y	[*s*]

Table 2 *Physical variables of the IPP model application*

Description	Label	
ball position – axis *x*	*x(t)*	[*m*]
ball position – axis *y*	*y(t)*	[*m*]
plate tilt – axis *x*	*α(t)*	[*rad*]
reference angle – axis *x*	$\alpha_x(t)$	[*rad*]
plate tilt – axis *y*	*β(t)*	[*rad*]
reference angle – axis *y*	$\beta_y(t)$	[*rad*]

The mathematical model of the *IPP* is derived with the assumption, that originally MIMO system is decomposed to the two SISO systems for the axis *x* and axis *y* (Fig. 3). Next, the SISO system consists of the subsystem *Ball on beam* and subsystem *Servomotor*. The subsystem *Ball on beam* is described by the nonlinear differential equation:

$$\ddot{x}(t) = \frac{5}{7} g \sin\alpha(t) \tag{1}$$

Actual tilt of the plate represented by angle $\alpha(t)$ is output of the subsystem *Servomotor*, which is described by transfer function:

$$F_s(s) = \frac{\alpha(s)}{\alpha_x(s)} = \frac{K_x}{T_x s + 1}, \tag{2}$$

where parameters K_x, T_x are identified from the step response of the subsystem *Servomotor* [11].

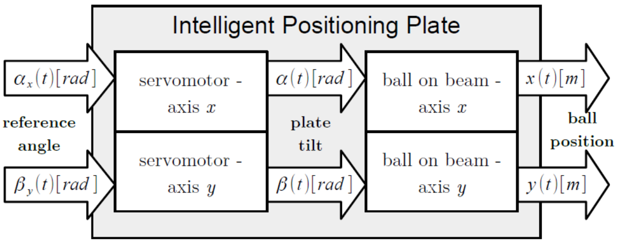

Figure 3 *System decomposition of the IPP model application*

Transfer function (2) can be expressed in form of the linear differential equation:

$$\dot{\alpha}(t) = \frac{1}{T_x} \cdot (K_x \alpha_x(t) - \alpha(t)) \tag{3}$$

Nonlinear differential equation (1) of the *Ball on beam* subsystem is adjusted into the canonical form using substitution $x(t) = x_1(t)$ and $\alpha\,(t) = x_3(t)$:

$$\begin{aligned} \dot{x}_1(t) &= x_2(t) \\ \dot{x}_2(t) &= \frac{5}{7} g \sin x_3(t) \end{aligned} \tag{4}$$

and linear differential equation (3) of the *Servomotor* subsystem is also modified into form:

$$\dot{x}_3(t) = \frac{1}{T_x}(K_x \alpha_x(t) - x_3(t)), \tag{5}$$

where $x(t) = x_1(t)$ represents coordinate of the ball position, $\dot{x}_1(t) = x_2(t)$ is velocity of the ball and $\alpha\,(t) = x_3(t)$ is angle of the plate tilt.

In general, the mathematical model of the *IPP* for axis *x* includes the *Ball on beam* subsystem and *Servomotor* subsystem represented by equation (25) and (26) can be expressed:

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{f}(\boldsymbol{x}(t), u(t), t) \\ \boldsymbol{y}(t) &= \boldsymbol{h}(\boldsymbol{x}(t), t) \end{aligned} \tag{6}$$

where $\boldsymbol{x}\,(t) = [x_1(t), x_2(t), x_3(t)]^T$ a $u(t) = [\alpha_x(t)]$.

For the reference trajectory tracking control target, the mathematical model (6) of the *IPP* is linearized around the equilibrium point $\boldsymbol{x}_{EP}=[x_1=0,\ x_2=0,\ x_3=0]$. The result of the linearization is the state space model of the system for the axis *x*:

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}u(t) \\ \boldsymbol{y}(t) &= \boldsymbol{C}\boldsymbol{x}(t) \end{aligned} \tag{7}$$

The matrix of dynamic $\boldsymbol{A}$, input matrix $\boldsymbol{B}$ and output matrix $\boldsymbol{C}$ are defined:

$$\boldsymbol{A}=\begin{bmatrix}\frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3}\\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3}\\ \frac{\partial f_3}{\partial x_1} & \frac{\partial f_3}{\partial x_2} & \frac{\partial f_3}{\partial x_3}\end{bmatrix}_{\boldsymbol{x}_{EP}}=\begin{bmatrix}0 & 1 & 0\\ 0 & 0 & \frac{5}{7}g\\ 0 & 0 & -\frac{1}{T_x}\end{bmatrix}_{\boldsymbol{x}_{EP}}, \tag{8}$$

$$\boldsymbol{B}=\begin{bmatrix}\frac{\partial f_1}{\partial u}\\ \frac{\partial f_2}{\partial u}\\ \frac{\partial f_3}{\partial u}\end{bmatrix}_{\boldsymbol{x}_{EP}}=\begin{bmatrix}0\\ 0\\ \frac{K_x}{T_x}\end{bmatrix}_{\boldsymbol{x}_{EP}}, \quad \boldsymbol{C}=[1 \quad 0 \quad 0]$$

Transfer function is derived from the state space model of the system by relation:

$$F(s)=\boldsymbol{C}(s\boldsymbol{I}-\boldsymbol{A})^{-1}\boldsymbol{B}, \tag{9}$$

where s is Laplace operator.

For the predictive control algorithm design purpose, it is necessary to discretize the state space model of the system (7) with selected sample period T_s. Discretization of the state space model (7) is done using the *Matlab* function *c2d* with sample period T_s=0,05 s.

Discrete transfer function $F(z^{-1})$ is obtained from the continuous transfer function (9) by discretization with selected sample period T_s in form:

$$F_Z(z^{-1})=\frac{B_Z(z^{-1})}{A_Z(z^{-1})}, \tag{10}$$

Following the same procedure as for the axis x, discrete linear state space model and transfer function of the *IPP* is obtained in the direction of the axis y.

Simulation model of the *IPP* is created according to mathematical model of the *IPP* model application in *Matlab/Simulink* environment. The time responses to the same step input signal for the nonlinear simulation and laboratory model *IPP* are compared in Fig. 4, Fig. 5.

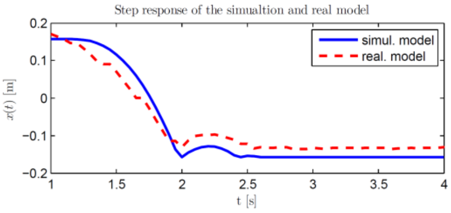

Figure 4 *Comparison of the* IPP *simulation and real model time responses to the same step input – direction of the axis x*

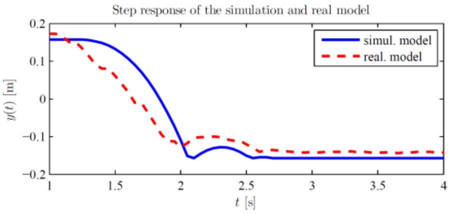

Figure 5 *Comparison of the* IPP *simulation and real model time responses to the same step input – direction of the axis y*

In the next sections of the article is simulation model of the *IPP* used for verification of the designed predictive control algorithms.

Simulation model of the *IPP* is also connected to the 3D virtual model (Fig. 6) in *Simulink* environment. The 3D virtual model, which is created in *Simulink 3D Animation toolbox,* is used for visualization of the *IPP* control simulation.

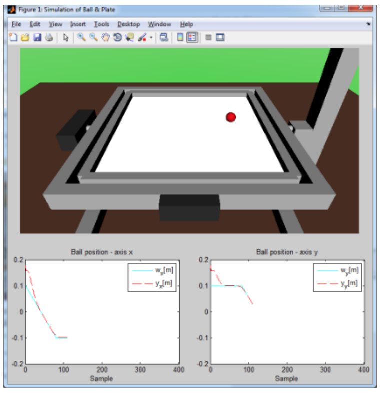

Figure 6 *Virtual model of the* IPP *created using Simulink 3D Animation toolbox*

Design of predictive control algorithm based on linear discrete state space model of the dynamic system

The various predictive control methods propose different cost functions for derivation of the control law. The presented predictive control algorithm is based on the solving optimization tasks and in general minimizes the cost function:

$$J=\sum_{i=1}^{N_p}\boldsymbol{Q}[\hat{\boldsymbol{y}}(k+i)-\boldsymbol{w}(k+i)]^2+ \\ +\sum_{i=1}^{N_u}\boldsymbol{R}[\boldsymbol{u}(k+i-1)]^2 \tag{11}$$

where N_p, N_u represents prediction horizon and control horizon, $\hat{\boldsymbol{y}}(k)$ is predicted value of the system output, $\boldsymbol{w}(k)$ denotes reference trajectory, $\boldsymbol{u}(k)$ is control input and $\boldsymbol{Q}$, $\boldsymbol{R}$ are weight matrices [1].

For the design of the predictive control algorithm is used the linear predictor, which is derived by iteration from the linear discrete state space model of the system:

$$\begin{aligned}\boldsymbol{x}(k+1)&=\boldsymbol{A}_d\boldsymbol{x}(k)+\boldsymbol{B}_d u(k)\\ \boldsymbol{y}(k)&=\boldsymbol{C}\boldsymbol{x}(k)\end{aligned} \tag{12}$$

and the linear predictor has form:

$$\hat{\boldsymbol{y}}(k)=\boldsymbol{V}\boldsymbol{x}(k)+\boldsymbol{G}\boldsymbol{u}(k), \tag{13}$$

where:

$\hat{\boldsymbol{y}}(k)$=[$\boldsymbol{y}(k+1)$, ..., $\boldsymbol{y}(k+N_p)$]$^{\mathrm{T}}$,
$\boldsymbol{u}(k)$=[$u(k)$, $u(k+1)$, ..., $u(k+N_p-1)$]$^{\mathrm{T}}$

In (13), the term $\boldsymbol{Vx}$(k) represents the free response and the term $\boldsymbol{Gu}$(k) is the forced response of the system. Matrices of the free response $\boldsymbol{V}$ and forced response $\boldsymbol{G}$, which are obtained by recursive computation, can be expressed in form:

$$\boldsymbol{V}=\begin{pmatrix}\boldsymbol{CA}_d\\ \vdots\\ \boldsymbol{CA}_d^{N_p}\end{pmatrix}, \boldsymbol{G}=\begin{pmatrix}\boldsymbol{CB}_d & 0 & \cdots & & 0\\ \boldsymbol{CA}_d\boldsymbol{B}_d & \boldsymbol{CB}_d & 0 & \cdots & 0\\ \vdots & & \ddots & & \vdots\\ \boldsymbol{CA}_d^{N_p-1}\boldsymbol{B}_d & \cdots & & \boldsymbol{CA}_d\boldsymbol{B}_d & \boldsymbol{CB}_d\end{pmatrix} \tag{14}$$

Selection of the control horizon N_u ($N_u \leq N_p$) affects the size of forced response matrix $\boldsymbol{G}$.

According to [1], the computation of the optimal control input $\boldsymbol{u}_{opt}(k)$ of the presented predictive control algorithm is based on the minimization of the cost function (11) with respect to the condition:

$$\frac{\partial J_{MPC}}{\partial \boldsymbol{u}(k)} \overset{!}{=} 0 \qquad (15)$$

Finally, the control law has form:

$$\boldsymbol{u}_{opt}(k) = -\boldsymbol{H}^{-1}\boldsymbol{g}, \qquad (16)$$

where the Hessian $\boldsymbol{H}$ and the gradient $\boldsymbol{g}$ are defined as follows:

$$\begin{aligned} \boldsymbol{H} &= (\boldsymbol{G}^T\boldsymbol{Q}\boldsymbol{G} + \boldsymbol{R}) \\ \boldsymbol{g}^T &= (\boldsymbol{V}\boldsymbol{x}(k) - \boldsymbol{w}(k))^T\boldsymbol{Q}\boldsymbol{G} \end{aligned} \qquad (17)$$

The optimal control sequence $\boldsymbol{u}_{opt}(k)$ can be computed in *Matlab* environment using *quadprog* function (from *Optimization toolbox*) with respect to constraints of physical variables of the controlled system. Function *quadprog* computes $\boldsymbol{u}_{opt}(k)$ by formula:

$$\min_{\boldsymbol{u}} \left(\frac{1}{2}\boldsymbol{u}\boldsymbol{H}\boldsymbol{u} + \boldsymbol{g}^T\boldsymbol{u}\right) \qquad (18)$$

with consideration of constraints of the controlled system, which are composed to inequality $\boldsymbol{U}_{con}\boldsymbol{u}_{opt} \leq \boldsymbol{v}_{con}$. For the constraints of the control input $\boldsymbol{u}_{opt}$ ϵ $<u_{min}, u_{max}>$ or system output $\boldsymbol{y}(k)$ϵ $<y_{min}, y_{max}>$ has matrix $\boldsymbol{U}_{con}$ and vector $\boldsymbol{v}_{con}$ form:

$$\boldsymbol{U}_{con} = \begin{pmatrix} \boldsymbol{I}_D \\ -\boldsymbol{I}_D \end{pmatrix}, \boldsymbol{v}_{con} = \begin{pmatrix} \boldsymbol{1}u_{\max} \\ -\boldsymbol{1}u_{\min} \end{pmatrix}, \qquad (19)$$

$$\boldsymbol{U}_{con} = \begin{pmatrix} \boldsymbol{G} \\ -\boldsymbol{G} \end{pmatrix}, \boldsymbol{v}_{con} = \begin{pmatrix} \boldsymbol{1}y_{\max} - \boldsymbol{y}_0 \\ -\boldsymbol{1}y_{\min} + \boldsymbol{y}_0 \end{pmatrix}, \qquad (20)$$

where **1** is a unit vector and $\boldsymbol{I}_D$ is an unit matrix [8].

Design of the state predictive control algorithm is shown in flowchart, Fig. 7.

This algorithm is implemented to the *Matlab* environment with respect to selected control structure illustrated in Fig. 8 as *m-file* called ***ssMPCcon***. For purpose of the predictive control algorithm is also created *paramMpcC* function:

[*H*,*G*,*V*,*Ucon*,*vcon*]=***paramMPCc***(*Ad*,*Bd*,*C*,*D*,*Q*,*R*,*Np*,*Nu*)

Function *paramMpcC* computes Hessian $\boldsymbol{H}$, matrix of forced response $\boldsymbol{G}$, matrix of free response $\boldsymbol{V}$, constraints matrix and vector $\boldsymbol{U}_{con}$, $\boldsymbol{v}_{con}$ for the predictive control algorithm.

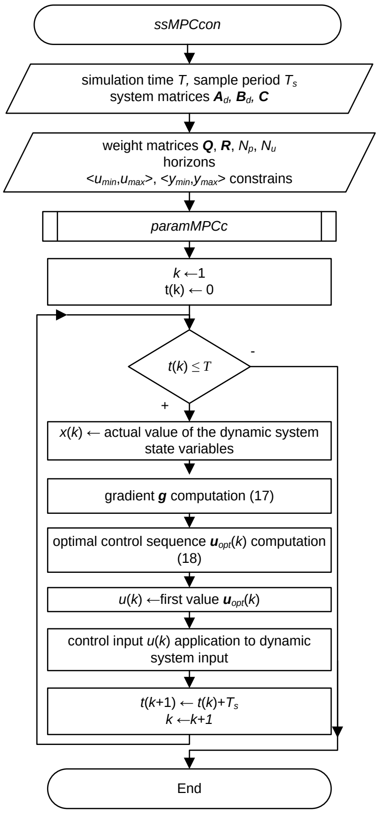
ssMPCcon
simulation time T, sample period Ts
system matrices Ad, Bd, C
weight matrices Q, R, Np, Nu
horizons
<umin,umax>, <ymin,ymax> constrains
paramMPCc
k ←1
t(k) ← 0
t(k) ≤ T
-
+
x(k) ← actual value of the dynamic system state variables
gradient g computation (17)
optimal control sequence uopt(k) computation (18)
u(k) ←first value uopt(k)
control input u(k) application to dynamic system input
t(k+1) ← t(k)+Ts
k ←k+1
End

Figure 7 *Flowchart of the designed predictive control algorithm based on the discrete state space model of the dynamic system*

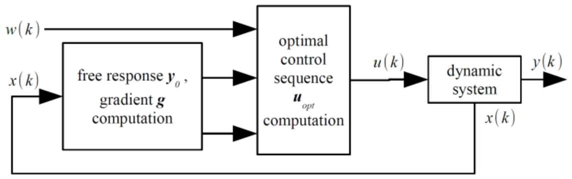
w(k)
x(k)
free response y0, gradient g computation
optimal control sequence uopt computation
u(k)
dynamic system
y(k)
x(k)

Figure 8 *The predictive control algorithm based on the discrete state space model of the dynamic system implemented to the control structure*

For verification of the designed predictive control algorithm, two experiments were done using the *IPP* model application. The experiments were realized for the circle reference trajectory tracking goal. Simulation and sampling time in experiments were set to T = 20s, T_s = 0,05s.

The first experiment of the *IPP* control was realized using the simulation model with parameters, which are listed in Tab. 4. Time responses of the ball position coordinates $x(t)$, $y(t)$ and control inputs $\alpha_x(t)$, $\beta_y(t)$ are shown in Fig. 9.

Table 3 *Parameters used for the predictive control of the IPP model application*

	Parameter	***Value***
axis *x*	N_p	20
	N_u	1
	$\boldsymbol{Q}$	1000 $\boldsymbol{I}$
	$\boldsymbol{R}$	0,01 $\boldsymbol{I}$
axis *y*	N_p	20
	N_u	1
	$\boldsymbol{Q}$	750 $\boldsymbol{I}$
	$\boldsymbol{R}$	0,01 $\boldsymbol{I}$

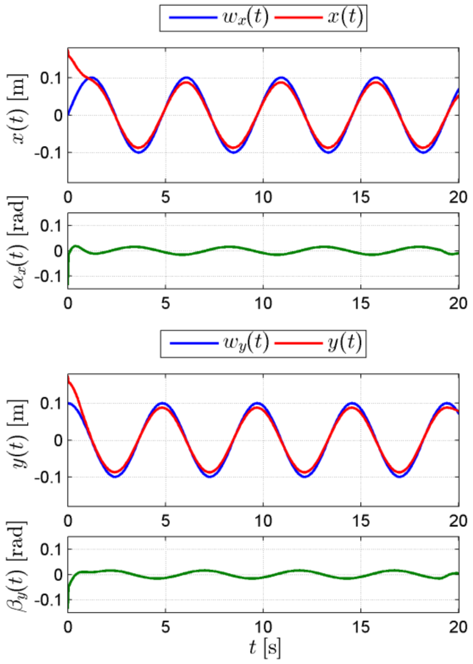

Figure 9 *State predictive control of the* IPP *simulation model – time responses of the ball position coordinates* $x(t)$, $y(t)$ *and control inputs* $\alpha_x(t)$, $\beta_y(t)$

The second experiment was realized with the real laboratory model of the *IPP* using the parameters presented in Tab. 3. Obtained results of the *IPP* laboratory model predictive control are represented by time responses of the ball position coordinates $x(t)$, $y(t)$ and control inputs $\alpha_x(t)$, $\beta_y(t)$ (Fig. 10).

According to the results, the designed predictive control algorithm is characterized by the small divergence of the ball position from desired reference trajectory using the optimal control input. In the case of the real laboratory model, the control input is represented by higher oscillations than the case of the simulation model. The lower control quality of the *IPP* laboratory model is influenced by:

- uncertainty in modeling of the *IPP* mechanical parts
- unmeasurable disturbances effect

Despite the influence of unmeasurable disturbances or modeling uncertainties, the designed predictive control algorithm fulfills the control requirements with sufficient quality.

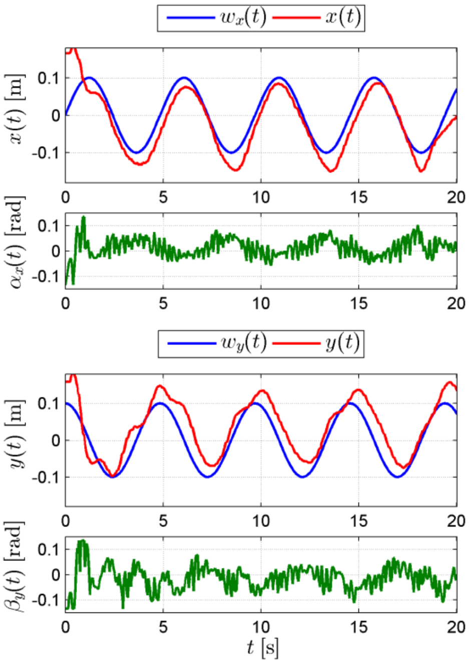

Figure 10 *State predictive control of the IPP laboratory model – time responses of the ball position coordinates x(t), y(t) and control inputs $\alpha_x(t)$, $\beta_y(t)$*

Design of predictive control algorithm based on regressive ARX model of the dynamic system

This section is dedicated to the predictive control algorithm design, which is based on the predictive control approach presented in [2]. In paper [3], predictive control based on the ARX model of the system is used for control of the Ball and Beam, which is similar to the *IPP* model application.

Presented algorithm of the predictive control assumes that the cost function (11) it is minimized, but the control law is derived using the regressive ARX model of the dynamic system:

$$y(k+1)=\sum_{i=1}^{m} b_i u(k-i+1)-\sum_{i=1}^{n} a_i y(k-i+1) \qquad (21)$$

Computation of the system output $y(k+1)$ in (21) can be arranged, according to [2], into the matrix form:

$$\overbrace{\begin{pmatrix} y(k-n+2) \\ \vdots \\ y(k) \\ y(k+1) \end{pmatrix}}^{\boldsymbol{X}(k+1)} =$$

$$= \overbrace{\begin{pmatrix} 0 & 1 & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & & \dots & 1 \\ -a_n & -a_{n-1} & \cdots & -a_1 \end{pmatrix}}^{\boldsymbol{A}_0} \overbrace{\begin{pmatrix} y(k-n+1) \\ \vdots \\ y(k-1) \\ y(k) \end{pmatrix}}^{\boldsymbol{X}(k)} + \qquad (22)$$

$$+ \underbrace{\begin{pmatrix} 0 & 0 & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & & \dots & 0 \\ b_n & b_{n-1} & \cdots & b_1 \end{pmatrix}}_{\boldsymbol{B}_0} \underbrace{\begin{pmatrix} u(k-m+1) \\ \vdots \\ u(k-1) \\ u(k) \end{pmatrix}}_{\boldsymbol{U}_0(k)}$$

$$y(k) = \underbrace{(0 \dots 0 \ 1)}_{\boldsymbol{C}_0} \underbrace{\begin{pmatrix} y(k-n+1) \\ \vdots \\ y(k) \end{pmatrix}}_{\boldsymbol{X}(k)}$$

In general, matrix form (22) can be written into pseudo-state space model:

$$\begin{aligned} \boldsymbol{X}(k+1) &= \boldsymbol{A}_0 \boldsymbol{X}(k) + \boldsymbol{B}_0 \boldsymbol{U}_0(k) \\ y(k) &= \boldsymbol{C}_0 \boldsymbol{X}(k) \end{aligned} \qquad (23)$$

The system output prediction $\hat{\boldsymbol{y}}(k) = [y(k+1), \dots, y(k+N_p)]$ using pseudo-state space model (22) can be expressed:

$$\begin{pmatrix} y(k+1) \\ \vdots \\ y(k+N_p) \end{pmatrix} = \begin{pmatrix} \boldsymbol{C}_0 \boldsymbol{A}_0 \\ \vdots \\ \boldsymbol{C}_0 \boldsymbol{A}_0^{N_p} \end{pmatrix} \begin{pmatrix} y(k-n+1) \\ \vdots \\ y(k) \end{pmatrix} + \overline{\boldsymbol{G}} \begin{pmatrix} u(k-m+1) \\ \vdots \\ u(k+N_p-1) \end{pmatrix}, \qquad (24)$$

where the matrix $\bar{\boldsymbol{G}}$ has form:

$$\overline{\boldsymbol{G}} = \begin{pmatrix} \boldsymbol{C}_0 \boldsymbol{B}_0 & \boldsymbol{0} \\ \vdots & \\ \boldsymbol{C}_0 \boldsymbol{B}_i & \boldsymbol{0} \\ \vdots & \\ \boldsymbol{C}_0 \boldsymbol{B}_{N_p-1} & \end{pmatrix}, \boldsymbol{B}_0 = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & & \dots & 0 \\ b_n & b_{n-1} & \cdots & b_1 \end{pmatrix} \qquad (25)$$

$$\boldsymbol{B}_i = (\boldsymbol{A}_0 \boldsymbol{B}_{i-1} \, 0 \dots 0) + (0 \dots 0 \ \boldsymbol{B}_0),$$
$$i = 1, \dots, N_p - 1$$

It is possible to express the predictor of the modified predictive control algorithm from the pseudo-state space model (22) according to [2] in form:

$$\hat{\boldsymbol{y}}(k) = \begin{pmatrix} y(k+1) \\ \vdots \\ y(k+N_p) \end{pmatrix}, \quad \boldsymbol{u}(k) = \begin{pmatrix} u(k) \\ \vdots \\ u(k+N_p-1) \end{pmatrix},$$

$$\boldsymbol{y_0}(k) = \begin{pmatrix} \boldsymbol{C}_0 \boldsymbol{A}_0 \\ \vdots \\ \boldsymbol{C}_0 \boldsymbol{A}_0^{N_p} \end{pmatrix} \begin{pmatrix} y(k-n+1) \\ \vdots \\ y(k) \end{pmatrix} + \qquad (26)$$

$$+ \overline{\boldsymbol{G}}_{(:,1:m-1)} \begin{pmatrix} u(k-m+1) \\ \vdots \\ u(k-1) \end{pmatrix},$$

$$\boldsymbol{G} = \overline{\boldsymbol{G}}_{(:,m:m+N_p-1)},$$

$$\hat{\boldsymbol{y}}(k) = \boldsymbol{y}_0(k) + \boldsymbol{G}\boldsymbol{u}(k),$$

where ":" convention is used for the selection of part of the matrix $\bar{\boldsymbol{G}}$ in predictor (26) and meaning of ":" convention is taken from *Matlab* environment.

With respect to constraints composed in form (19) or (20), the optimal control input sequence $\boldsymbol{u}_{opt}(k)$ is computed using function *quadprog* of *Matlab* environment based on formula (18) [8]. In case of the predictive control based on the regressive ARX model of the system, the Hessian $\boldsymbol{H}$ and the gradient $\boldsymbol{g}$ have form:

$$\begin{aligned} \boldsymbol{H} &= (\boldsymbol{G}^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{G} + \boldsymbol{R}^T\boldsymbol{R}) \\ \boldsymbol{g}^T &= (\boldsymbol{y}_0 - \boldsymbol{w}(k))^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{G} \end{aligned} \tag{27}$$

The designed modified predictive control algorithm based on the regressive ARX model of the system is shown in the flowchart, Fig. 11.

According to the flowchart (Fig. 11), designed predictive control algorithm is implemented as *m-file* called ***ioGPCcon*** to the *Matlab* environment with respect to control structure, which is shown in Fig. 12. For computation of the Hessian ***H***, matrices of the free response $\boldsymbol{y}_0$ and matrix of the forced response ***G*** is also created function:

[*H*,*y0_AC*,*y0_G*,*G*,*Ucon*,*vcon*]= ***paramGPCc***(*Bz*,*Az*,*Q*,*R*,*Np*,*Nu*)

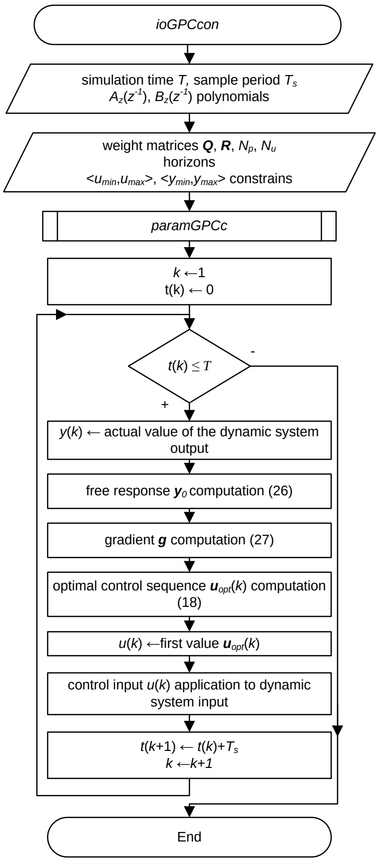
ioGPCcon
simulation time T, sample period Ts
Az(z-1), Bz(z-1) polynomials
weight matrices Q, R, Np, Nu
horizons
<umin,umax>, <ymin,ymax> constrains
paramGPCc
k ← 1
t(k) ← 0
t(k) ≤ T
-
+
y(k) ← actual value of the dynamic system output
free response y0 computation (26)
gradient g computation (27)
optimal control sequence uopt(k) computation (18)
u(k) ← first value uopt(k)
control input u(k) application to dynamic system input
t(k+1) ← t(k)+Ts
k ← k+1
End

Figure 11 *Flowchart of the designed predictive control algorithm based on the regressive ARX model of the dynamic system*

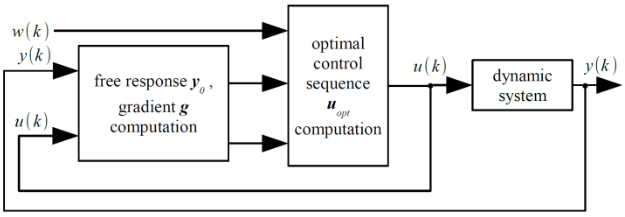
w(k)
y(k)
u(k)
free response y0, gradient g computation
optimal control sequence uopt computation
u(k)
dynamic system
y(k)

Figure 12 *The predictive control algorithm based on the regressive ARX model of the dynamic system implemented to the control structure*

The designed predictive control algorithm based on the ARX model of the dynamic system was also verified by experiments using the *IPP* model application. The goal of the predictive control experiments was the circle reference trajectory tracking.

In the first experiment, the simulation model of the *IPP* was used and the predictive control parameters are also listed in Tab. 3. Time responses of the ball position coordinates $x(t)$, $y(t)$ and control inputs $\alpha_x(t)$, $\beta_y(t)$ are shown in Fig. 13.

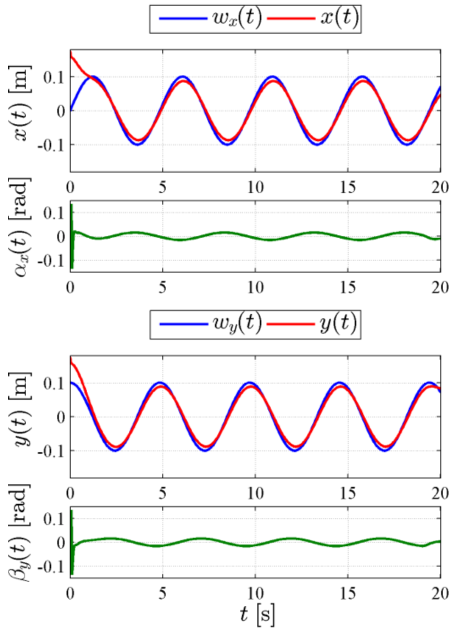

Figure 13 *Predictive control of the* IPP *simulation model – time responses of the ball position coordinates* $x(t)$, $y(t)$ *and control inputs* $\alpha_x(t)$, $\beta_y(t)$

The second experiment was realized using the real laboratory model with the same parameters of the predictive control algorithm like the first experiment. The results of the *IPP* laboratory model predictive control are characterized by the oscillations of the control input, but with the acceptable divergence of the ball position from the desired position. The time responses of the ball position coordinates $x(t)$, $y(t)$ and control inputs $\alpha_x(t)$, $\beta_y(t)$ are shown in Fig. 14.

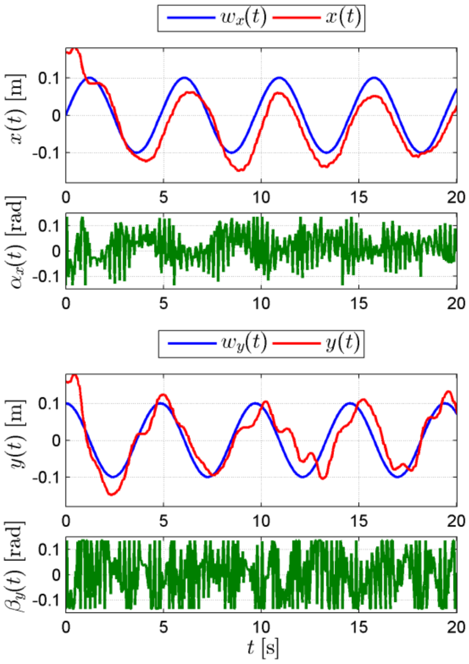

Figure 14 *Predictive control of the IPP laboratory model – time responses of the ball position coordinates* $x(t)$, $y(t)$ *and control inputs* $\alpha_x(t)$, $\beta_y(t)$

Both designed predictive control algorithms are a part of the tool for the control of the *IPP* model application, which is called *IPPtools*. The *IPPtools* also includes the generator of different reference trajectories like a circle, square or spiral.

Diagnosis system concept design of the dynamic system

Dynamic systems are often influenced by faults of their components. The task of the fault diagnosis research area is to design algorithms for detection of the fault, localization of the place of the fault occurrence (fault isolation) and determining its magnitude [5], [6]. Selected fault diagnosis methods and approaches can be implemented in the diagnosis system, which is used for the monitoring of the controlled system.

In general, the control design of the dynamic system is very important step to the correct functionality of the diagnosis system. Controlled dynamic system without fault occurrence is considered as nominal system, which is used for the fault detection algorithm design.

The actuators and sensors are the most sensitive parts of the dynamic system to the fault occurrence. Condition of these parts of the dynamic system can be monitored by the diagnosis system which consists of the algorithms for fault detection, isolation and estimation. Information from the diagnosis system can be used for the fault accommodation, what means to adapt the controller parameters to the properties of the faulty dynamic system (Fig. 15) [12].

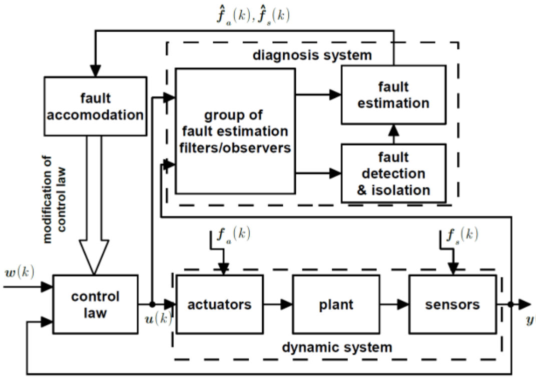

Figure 15 *Concept scheme of the diagnosis system*

The sensors fault diagnosis method presented in [6] uses the group of the fault estimation filters, which are based on the Kalman filtering principles. This method can be used for the actuators fault diagnosis after the modification of the faulty model of the dynamic system. Other method for actuators or sensors fault diagnosis is presented in [5] and its derivation is based on the SVD principle, but they are some similarity with the fault diagnosis method in [6]. Both methods can be useful for implementation to the diagnosis system structure shown in Fig. 15. The algorithms based on these methods can be implemented in the Matlab/Simulink enivornment, but also in the single-chip microcomputer or in the PLC.

The selected fault diagnosis methods use linear model of the system for the fault diagnosis. The presence of the actuators faults can be reflected in the linear model of the dynamic system:

$$\begin{aligned} \boldsymbol{x}(k+1) &= \boldsymbol{A}_d\boldsymbol{x}(k) + \boldsymbol{B}_d\boldsymbol{u}(k) + \boldsymbol{F}_a\boldsymbol{f}_a(k) \\ \boldsymbol{y}(k) &= \boldsymbol{C}\boldsymbol{x}(k) \end{aligned} \qquad (28)$$

The actuators faults matrix $\boldsymbol{F}_a = \boldsymbol{B}_d$ and vector of the actuators faults $\boldsymbol{f}_a$ can be expressed:

$$\boldsymbol{f}_a(k) = (\boldsymbol{\Gamma} - \boldsymbol{I})\boldsymbol{u}(k) + \boldsymbol{u}_f(k), \qquad (29)$$

where the vector $\boldsymbol{u}_f(k) = [u_{f1}, ..., u_{fp}]^{\mathrm{T}}$ corresponds to the effect of an additive actuators faults and $\boldsymbol{\Gamma u}(k)$ represents the effect of a multiplicative actuators faults.

The matrix of the multiplicative actuators faults $\boldsymbol{\Gamma}$ has form:

$$\boldsymbol{\Gamma} = \begin{bmatrix} \gamma_1 & 0 & \cdots & & 0 \\ 0 & \ddots & & & \vdots \\ \vdots & & \gamma_j & & \\ & & & \ddots & 0 \\ 0 & \cdots & & 0 & \gamma_p \end{bmatrix} \qquad (30)$$

If the j-th actuator is faulty then $0 \le \gamma_j < 1$ or $u_{fj} \ne 0$ [5]. Various types of the actuator faults are described in Tab. 5.

Table 3 *Different types of the actuator faults [5]*

	$u_{fj} = 0$	$u_{fj} \ne 0$
$\gamma_j = 1$	fault-free	bias
$0 < \gamma_j < 1$	loss of effectivness	loss of effectivness
$\gamma_j = 0$	loss of input	actuator blocked

Also the presence of the sensors faults can be reflected in the linear model of the dynamic system:

$$\begin{aligned} \boldsymbol{x}(k+1) &= \boldsymbol{A}_d\boldsymbol{x}(k) + \boldsymbol{B}_d\boldsymbol{u}(k) \\ \boldsymbol{y}(k) &= \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{F}_s\boldsymbol{f}_s(k) \end{aligned} \qquad (31)$$

where $\boldsymbol{F}_s = \boldsymbol{C}$ and the vector $\boldsymbol{f}_s$ represents the sensors faults.

The faulty models of the dynamic system are the basic assumption for the fault diagnosis algorithms design and they can be implemented as a part of the complex diagnosis system (Fig. 15) into the 5-level Distributed Control System (DCS) of the DCAI. Concept scheme of the diagnosis system implemented into the lowest three levels of the DCS of the DCAI is shown in Fig. 16.

The design of the diagnosis system is very important step to improve the presented predictive control algorithms to the form that expands capability of these algorithms to adapt to the changed properties of the controlled dynamic system.

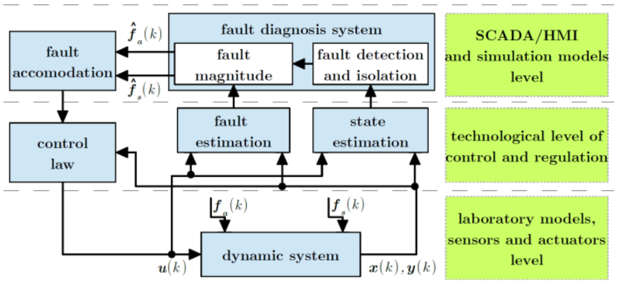

Figure 16 *Concept of the diagnosis system implemented in the 5 - level DCS of the DCAI*

In article [7] is presented possibility of changing the predictive control algorithm in case of the actuator fault what can be reflected in the algorithm by changing values u_{min}, u_{max} of the optimal control input $\boldsymbol{u}_{opt}$ constraints (19) according to the estimated magnitude of the actuator fault.

If the j-th actuator loss effectiveness, it affects constraints:

$$\boldsymbol{v}_{con} = \begin{pmatrix} \mathbf{1}(\gamma_j u_{\max} - u_{fj}) \\ -\mathbf{1}(\gamma_j u_{\min} + u_{fj}) \end{pmatrix} \qquad (32)$$

In case of the j-th actuator blocking, constraints have form:

$$\boldsymbol{v}_{con} = \begin{pmatrix} \mathbf{1}(u_{fj}) \\ -\mathbf{1}(u_{fj}) \end{pmatrix}, \tag{33}$$

where u_{fj} = const.

For loss of the j-th actuator input are constraints modified to form:

$$\boldsymbol{v}_{con} = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \end{pmatrix} \tag{34}$$

According to the presented diagnosis system concept design, it is planned the functionality of the *IPPtools* extend to the fault diagnosis algorithms, which should be used for the monitoring of the *IPP* actuators faults occurrence.

Conclusion

This article presents the design of the two predictive control algorithms, which are implemented in Matlab environment. For verification of the designed predictive control algorithms, series of the experiments were realized using the *Intelligent Positioning Plate* (*IPP*) model application.

For realizing the experiments, it was used simulation and real laboratory model of the *IPP* model application. The results of the experiments illustrate that designed predictive control algorithms are suitable for the reference trajectory tracking goal of the control. The best results of the *IPP* model application predictive control were obtained by the state predictive control algorithm. The tilt of the plate was more fluently using the state predictive control algorithm and the control inputs for servomotors were without strong oscillations. The best results of the predictive control of the laboratory model were achieved with the prediction horizon N_p = 20 and control horizon N_u = 1. The selection of the sample period is mainly limited by the camera, which is capable to capture maximal 30 images per second.

The *IPP* is similar to the *Humusoft CE151 Ball & Plate* laboratory model (http://kyb.fei.tuke.sk/laben/modely/gnp.php), which is also one of the laboratory models of the DCAI. There are some differences in the construction of the both laboratory models and more details of the construction differences are stated in [11]. The construction of the *IPP* model application provides better portfolio for our solution like the *Humusoft CE151 Ball & Plate* laboratory model. The main advantage of the *IPP* model application is possibility to communicate with the *Matlab* environment without using any toolbox. On the other side, it was necessary to create own image processing application in *C#* language.

The designed predictive control algorithms are the part of the *IPPtools*, which also includes generator for the circle, square or spiral reference trajectories. Created *m-files* (*ssMPCc*, *ioGPCc*) and functions (*paramMPCc*, *paramGPCc*) for the predictive control included in the *IPPtools* are customized to the *IPP* laboratory model, because *m-files* and functions of the *IPPtools* includes necessary transformations of the control inputs to form suitable for the single-chip microcomputer, which is used for control of the servomotors. For these reasons is computation of the control inputs of the predictive control algorithms for the servomotors more effective like using functions of the *Model Predictive Control toolbox* of Matlab environment.

Also this paper presents conceptual design of the diagnosis system, which is implemented in the lowest three levels of the Distributed Control System of the DCAI. According to this concept, the diagnosis system for the selected laboratory models of the DCAI should be created where the *IPP* model application is one of them. Information from the diagnosis system can be use for the elimination of the faults influence to the control of the dynamic system.

In respect to the goals of the project "USP TECHNICOM for Innovation Applications Supported by Knowledge Technology" with subactivity "Center for Nondestructive Diagnostics of Technological Processes," it is planned to design algorithms for the diagnosis system and implement them in the Matlab/Simulink environment. Also, one of the future research goals is modified the predictive control algorithms to the fault tolerant form, which can extend the *IPPtools* tool.

Acknowledgement

This publication is the result of the Project implementation: University Science Park TECHNICOM for Innovation Applications Supported by Knowledge Technology - II. phase, ITMS: 313011D232 supported by the Operational Programme Research & Development funded by the ERDF (50%), grant TUKE FEI-2015-33: Research Laboratory of Nonlinear Underactuated Systems (30%) and grant KEGA 001TUKE- 4/2015 (20%).

References

[1] E. F. Camacho, C. Bordons (2013) "Model predictive control", Springer Science & Business Media, ISBN 978-1-85233-694-3.

[2] A. W. Ordys, D. W. Clarke (1993) "A state-space description for GPC controllers", International Journal of Systems Science, Vol. 24, Issue 9, pp. 1727-1744. ISSN 1464 - 5319.

[3] K. Belda, J. Böhm (2006) "Adaptive Generalized Predictive Control for Mechatronic Systems", WSEAS Transactions on Systems, Vol. 5, Issue 8, August 2006, Athens, Greece , pp. 1830–1837.

[4] N. Hara, M. Takahashi, K. Konishi (2011) "Experimental evaluation of model predictive control of ball and beam systems", Proceedings of the 2011 American Control Conference, pp. 1130-1132. ISSN 0743-1619.

[5] H. Noura, et al., (2009) "Fault-tolerant control systems: Design and practical applications," Springer Science & Business Media, pp. 233. ISBN 978-1-84882-652-6.

[6] W. Zhen-Hua, et al., (2014) "Sensor fault estimation filter design for discrete-time linear time-varying systems," In: Acta Automatica Sinica, Vol. 40, No.10, pp. 2364-2369. ISSN 1874-1029.

[7] T. Miksch, A. Gambier, E. Badreddin (2008) "Real-time implementation of fault-tolerant control using model predictive control," In: IFAC Proceedings Volumes, Vol. 41, Issue 2, pp. 11136-11141. ISBN 978-1-1234-7890-2.

[8] Š. Jajčišin, A. Jadlovská (2013) "Design of predictive control algorithms using nonlinear models of physical systems" (in Slovak), Košice: elfa, pp. 139. ISBN 978-80-8086-229-9.

[9] J. Jadlovský, A. Jadlovská, S. Jadlovská, J. Čerkala, M. Kopčík, J. Čabala, M. Oravec, M. Varga, D. Voščeк (2016) "Research Activities of the Center of Modern Control Techniques and Industrial Informatics," In: SAMI 2016: IEEE 14th International Symposium on Applied Machine Intelligence and Informatics: proceedings, pp. 279–285. ISBN 978-1-4673-8739-2

[10] Š. Jajčišin, A. Jadlovská (2012) "Predictive Control Algorithms Verification on the Laboratory Helicopter Model", Acta Polytechnica Hungarica (Journal of Applied Sciences), Vol. 9, No. 4. ISSN 1785-8860. Online: <http://www.uni-obuda.hu/journal/Jadlovska_Jajcisin_36.pdf>.

[11] M. Oravec, A. Jadlovská (2014) "Optimal State Control of the Mechatronical Laboratory Model B&P_Kyb", In: Electrical Engineering and Informatics 5 : Proceedings of the Faculty of Electrical Engineering and Informatics of the Technical University of Košice, TU, pp. 9-16. ISBN 978-80-553-1704-5.

[12] M. Oravec (2016) "Contribution to Methods and Approaches of Nondestructive Diagnostics of Dynamic Systems," Dissertation prospectus, pp. 62.

[13] A. Chevalier, et al., (2015) "Development and student evaluation of an internet-based control engineering laboratory," IFAC-Papers OnLine, No. 48, Issue 29, pp. 1-6.

[14] G. Torres, et al., (2014) "Internet-based control of a ball-and-plate system: A case study of modeling and automatic code generation

for networked control systems," Industrial Electronics Society, IECON 2014-40th Annual Conference of the IEEE, pp. 4762-4767.

[15] X. Fan, N. Zhang, S. Teng (2004) "Trajectory planning and tracking of ball and plate system using hierarchical fuzzy control scheme," Fuzzy Sets and Systems, No. 144, Issue 2, pp. 297-312.

[16] H. Liu, Y. Liang (2010) "Trajectory tracking sliding mode control of ball and plate system," Informatics in Control, Automation and Robotics (CAR), pp. 142-145.