Predictive Control Algorithms Verification on the Laboratory Helicopter Model

Anna Jadlovská, Štefan Jajčišin

Department of Cybernetics and Artificial Intelligence, Faculty of Electrotechnics and Informatics, Technical University in Košice
Letná 9, 042 00 Košice, Slovak Republic
anna.jadlovska@tuke.sk, stefan.jajcisin@tuke.sk

Abstract: The main goal of this paper is to present the suitability of predictive control application on a mechatronic system. A theoretical approach to predictive control and verification on a laboratory Helicopter model is considered. Firstly, the optimization of predictive control algorithms based on a state space, linear regression ARX and CARIMA model of dynamic systems are theoretically derived. A basic principle of predictive control, predictor deducing and computing the optimal control action sequence are briefly presented for the particular algorithm. A method with or without complying with required constraints is introduced within the frame of computing the optimal control action sequence. An algorithmic design manner of the chosen control algorithms as well as the particular control structures appertaining to the algorithms, which are based on the state space or the input-output description of dynamic systems, are presented in this paper. Also, the multivariable description of the educational laboratory model of the helicopter and a control scheme, in which it was used as a system to be controlled, is mentioned. In the end of the paper, the results of the real laboratory helicopter model control with the chosen predictive control algorithms are shown in the form of time responses of particular control closed loop's quantities.

Keywords: ARX; CARIMA model; generalized predictive control; state model-based predictive control; optimization; quadratic programming; educational Helicopter model

1 Introduction

The predictive control based on the dynamic systems' models is very popular at present. In the framework of this area, several different approaches or basic principle modifications exist. They can be divided in terms of many different attributes, but mainly in terms of the used model of the dynamic system. Predictive control based on the transfer functions is called generalized predictive control (GPC) [1]. Also predictive control based on the state space dynamic systems models exists [2]. Both approaches use linear models. Until now some modifications of basic predictive control principle have been created. Some of them, and other important issues like stability are mentioned in [3] or [4].

In this paper, we are engaged in a theoretical derivation of some predictive control methods based on the linear model of controlled system, and in preparing them for subsequent algorithmic design and verification on a real laboratory helicopter model from Humusoft [5], which serves as an educational model for identification and control algorithms verification at the Department of Cybernetics and Artificial Intelligence at the Faculty of Electrotechnics and Informatics at the Technical University in Košice. Particularly, we are concerned with the predictive control algorithm that is based on the state space description of the MIMO (**M**ulti **I**nput **M**ulti **O**utput) system [6], [7] and with generalized predictive algorithm, which is started from linear regression ARX (**A**uto**R**egressive e**X**ogenous) [9] model of SISO (**S**ingle **I**nput **S**ingle **O**utput) systems. Thus it is based on the input-output description of dynamic systems. Moreover, we apply it to the GPC algorithm based on the CARIMA (**C**ontrolled **A**utoRegressive **I**ntegrated **M**oving **A**verage) [8] model of the SISO system. All of mentioned control methods differ, whether in the derivation manner of predictor based on the system's linear model or in computing the optimal control action sequence. This implies that we have to take an individual approach to programming them. We programmed the mentioned predictive control algorithms as Matlab functions, which compute the value of the control action on the basis of particular input parameters. This allowed us to use a modular approach in control. We used these functions in specific control structures, which we programmed as scripts in simulation language Matlab. We carried out communication with a laboratory card connected to helicopter model through Real-Time Toolbox functions [13]. The acquired results will be presented as the time responses of the optimal control action and reference trajectory tracking by output of the Helicopter model.

2 Theoretical Base of Predictive Control

We introduce some typical properties of predictive control in this section. Next we deal with mathematical fundamentals of predictive control algorithm design and their programming as a Matlab functions.

Predictive control algorithms constitute optimization tasks and in general they minimize a criterion

$$J = \sum_{i=N_1}^{N_p} Q_e(i)\left[\hat{y}(k+i) - w(k+i)\right]^2 + \sum_{i=1}^{N_u} R_u(i)\left[u(k+i-1)\right]^2 , \tag{1}$$

where $u(k)$ is a control action, $\hat{y}(k)$ is a predicted value of controlled output and $w(k)$ denotes a reference trajectory. Values N_1 and N_p represent a prediction horizon. According to [8], the value N_1 should be at least $d+1$, where d is a system transport delay, in our case we suppose $N_1 = 1$. The positive value N_u denotes a control horizon, on which the optimal control action $u(k)$ is computed, whereby $N_u \leq N_p$. If the degrees of freedom of control action reduction is used in the predictive control algorithms, it is valid that $N_u < N_p$ [6]. Values $Q_e(i)$ and $R_u(i)$ constitute weighing coefficients of a deviation between system output and reference trajectory on the prediction horizon and control action on the control horizon. Next we will assume that $Q_e(i)$ and $R_u(i)$ are constant on the entire length of the prediction and control horizon, and thus they do not depend on variable i. In terms of weighing coefficients, their single value is not important, but mainly ratio $\lambda = R_u / Q_e$.

In some cases, the rate of control action $\Delta u(k)$ is used instead its direct value $u(k)$ in the criterion (1), whereby the control obtains an integration character, which results in the elimination of the steady state control deviation in the control process [6].

It is necessary to know the reference trajectory $w(k)$ on the prediction horizon in each sample instant in predictive control algorithms. The simplest reference trajectory example is a constant function with desired value w_0. According to [8], it is the more preferred form of smooth reference trajectory, whose initial value equals to the current system output and comes near to the desired value w_0 through a first order filter. This approach is carried out by equations

$$w(k) = y(k); \quad w(k+i) = \alpha w(k+i-1) + (1-\alpha) w_0, \quad \text{for } i = 1, 2, \dots, \tag{2}$$

where the parameter $\alpha \in \langle 0;1 \rangle$ expresses the smoothness of reference trajectory. If $\alpha \to 0$ then the reference trajectory has the fastest slope, and, on the contrary, if $\alpha \to 1$ the slowest. In the case when the reference trajectory is unknown, it is customary to use the so-called *random walk* [6], where $w(k+1) = w(k) + \xi(k)$, whereby $\xi(k)$ is a white noise.

Predictive control algorithm design can be divided in two phases:

1) predictor derivation (dynamic system behavior prediction),
2) computing the optimal control by criterion minimization.

The advantage of predictive control consists in the possibility to compose different constraints (of control action, its rate or output) in computing the optimal control action sequence. Most commonly, it is carried out by quadratic programming. In our case we used a *quadprog* function, which is one of functions in the *Optimization Toolbox* in Matlab and computes a vector of optimal values $\boldsymbol{u}$ by formula

$$\min_{\boldsymbol{u}} \frac{1}{2}\boldsymbol{u}^T \boldsymbol{H}\boldsymbol{u} + \boldsymbol{g}^T \boldsymbol{u}\text{ , subject to } \boldsymbol{A}_{con}\boldsymbol{u} \le \boldsymbol{b}_{con}\,. \tag{3}$$

The basic syntax for using the *quadprog* function to compute the vector $\boldsymbol{u}$ is

$$\boldsymbol{u} = quadprog(\boldsymbol{H}, \boldsymbol{g}, \boldsymbol{A}_{con}, \boldsymbol{b}_{con})\,, \tag{4}$$

whereby a form of matrix $\boldsymbol{H}$ (Hessian) and row vector $\boldsymbol{g}^T$ (gradient) depends on the predictive control algorithm used. It is necessary to compose the matrix $\boldsymbol{A}_{con}$ and the vector $\boldsymbol{b}_{con}$ in compliance with required constraints. The detail specification of their structures will be presented next, particularly with each algorithm. If the combination of more constraints is needed, the matrix $\boldsymbol{A}_{con}$ and the vector $\boldsymbol{b}_{con}$ are created by matrices and vectors for concrete constraint, which are organized one after another. The *quadprog* function also permits entering the function's output constraints as the function's input parameters, which abbreviates entering required constraints in matrix $\boldsymbol{A}_{con}$ and vector $\boldsymbol{b}_{con}$.

In the framework of the control process using predictive control algorithms, the so-called *receding horizon computing* is performed [6]. The point is that the sequence of optimal control action $\boldsymbol{u}_{opt} = \left[u_{opt}\left(k\right) \;\; \cdots \;\; u_{opt}\left(k+N_u-1\right)\right]$ is computed on the entire length of control horizon at each sample instant *k*, but only the first unit $u_{opt}(k)$ is used as the system input $u(k)$.

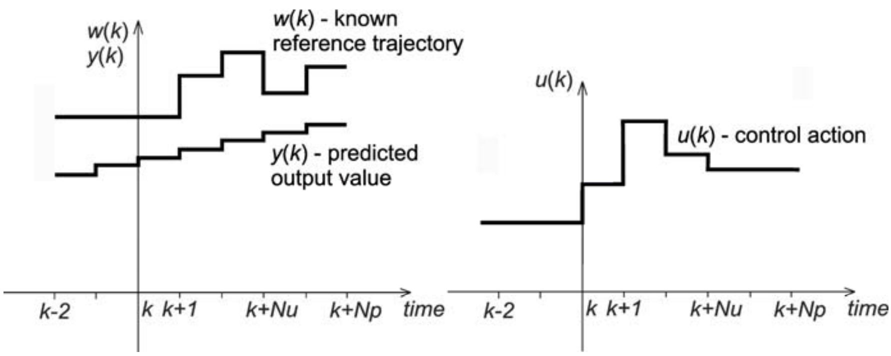


Figure 1

Predictive control principle

As we used the receding horizon principle in control algorithms, computing the optimal control action sequence $\boldsymbol{u}_{opt}$ is evaluated in conformity with Fig. 1. The authors of this paper designed the next procedure, which is carried out within the frame of every control process step *k*:

step 1: the assigning of the reference trajectory $\boldsymbol{w}$ on the prediction horizon,

step 2: the detection of the actual state $\boldsymbol{x}(k)$ or output $\boldsymbol{y}(k)$ of system in specific sample instant, step 3: the prediction of system response on the prediction horizon based on the actual values of optimal control action $u_{opt}(k)$ and state $\boldsymbol{x}(k)$ or input $\boldsymbol{u}(k)$ and output $\boldsymbol{y}(k)$ in previous sample instants without influence of next control action, the so-called system free response,

step 4: computing the sequence of optimal control action $\boldsymbol{u}_{opt}$ by the criterion J minimization with known parameters N_1, N_p, N_u, $Q(i)$ and $R(i)$,

step 5: using $u_{opt}(k)$ as a system input.

We implemented the introduced five steps into every type of predictive control algorithm with which we have been concerned, and which are introduced in the next particular parts of this paper.

3 State Space Model-based Predictive Control Algorithm Design

The State-space Model based Predictive Control (SMPC) algorithm predicts a system free response on the basis of its current state. The control structure using the SMPC algorithm is depicted in Fig. 2, where $\boldsymbol{w}$ is a vector of the reference trajectory on the prediction horizon, $\boldsymbol{y}_0$ is a system free response prediction on the prediction horizon, $\boldsymbol{x}(k)$ denotes a vector of current values of state quantities, $\boldsymbol{u}(k)$ represents a vector of control action, $\boldsymbol{d}(k)$ is a disturbance vector and $\boldsymbol{y}(k)$ is a vector of the system outputs, i.e the controlled quantities.

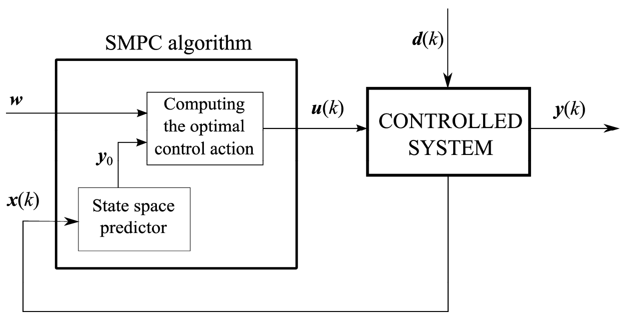


Figure 2
Control structure with SMPC algorithm

The SMPC algorithm belongs to the predictive control algorithms family, which use a state space description of MIMO dynamic systems for system output prediction (provided that there is no direct dependence between the system input and output)

$$\begin{aligned} \boldsymbol{x}(k+1) &= \boldsymbol{A}_d\boldsymbol{x}(k) + \boldsymbol{B}_d\boldsymbol{u}(k) \\ \boldsymbol{y}(k) &= \boldsymbol{C}\boldsymbol{x}(k) \end{aligned}, \tag{5}$$

($\boldsymbol{A_d}$ is a matrix of dynamics with dimension $nx \times nx$, $\boldsymbol{B_d}$ is an input matrix of dimension $nx \times nu$, $\boldsymbol{C}$ is an output matrix of dimension $ny \times nx$, $\boldsymbol{x}(k)$ is a vector of state quantities with length nx, $\boldsymbol{u}(k)$ is a vector of inputs with length nu, $\boldsymbol{y}(k)$ is a vector of outputs with length ny, where variables nx, nu and ny constitute the number of state quantities, inputs and outputs of dynamic system) and compute the sequence of optimal control action $\boldsymbol{u}(k)$ by the minimization of criterion

$$J_{\text{MPC}} = \sum_{i=N_1}^{N_p} Q_e \left[\hat{\boldsymbol{y}}(k+i) - \boldsymbol{w}(k+i) \right]^2 + \sum_{i=1}^{N_u} R_u \left[\boldsymbol{u}(k+i-1) \right]^2 . \tag{6}$$

The coefficients in the criterion (6) have the same meaning as in the criterion (1); however they denote vectors and matrices for multivariable system (5). We also assume equal weighing coefficients for each output/input of the system.

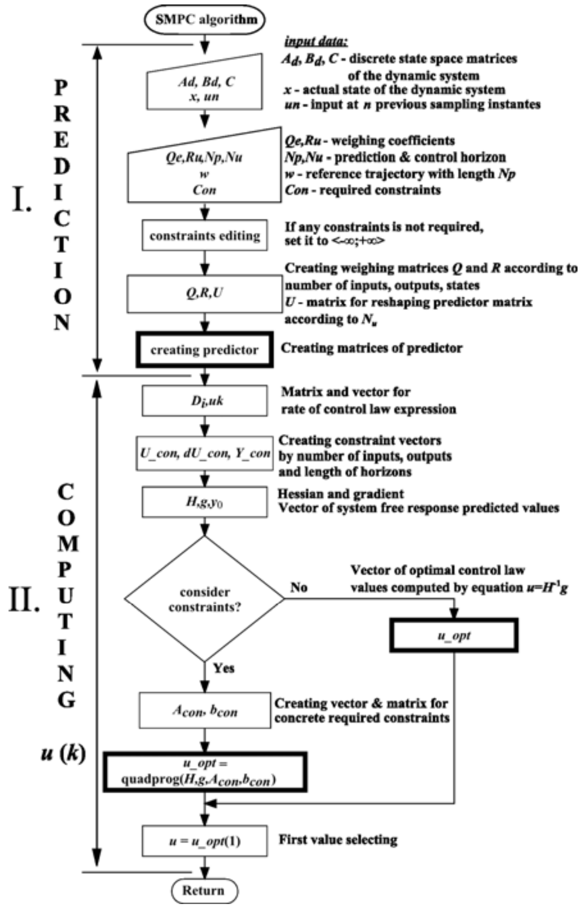


Figure 3

Flow chart of SMPC algorithm function

We programmed the SMPC algorithm as a Matlab function on the basis of the designed flow chart diagram, depicted in Fig. 3. The SMPC algorithm design is divided into two phases in compliance with the procedure mentioned in Section 2.

It is clear from the flow chart that control algorithms offer the optimal control action, computing with and without respect to required constraints of control action value, its rate or output of dynamic system. The mathematical description of two phases of SMPC algorithm design is described in next subsections 3.1 and 3.2.

3.1 Predictor Derivation in SMPC

The state prediction over the horizon N_p can be written according to [6] in form

$$\begin{array}{l}\hat{\boldsymbol{x}}(k+1)=\boldsymbol{A}_d\hat{\boldsymbol{x}}(k)+\boldsymbol{B}_d\boldsymbol{u}(k)\\ \hat{\boldsymbol{x}}(k+2)=\boldsymbol{A}_d\hat{\boldsymbol{x}}(k+1)+\boldsymbol{B}_d\boldsymbol{u}(k+1)=\boldsymbol{A}_d^2\ \boldsymbol{x}(k)+\boldsymbol{A}_d\boldsymbol{B}_d\boldsymbol{u}(k)+\boldsymbol{B}_d\boldsymbol{u}(k+1)\\ \hat{\boldsymbol{x}}(k+3)=\boldsymbol{A}_d\hat{\boldsymbol{x}}(k+2)+\boldsymbol{B}_d\boldsymbol{u}(k+2)=\boldsymbol{A}_d^3\ \boldsymbol{x}(k)+\boldsymbol{A}_d^2\boldsymbol{B}_d\boldsymbol{u}(k)+\boldsymbol{A}_d\boldsymbol{B}_d\boldsymbol{u}(k+1)+\boldsymbol{B}_d\boldsymbol{u}(k+2)\\ \quad\vdots \qquad\qquad \vdots \qquad\qquad \ddots\\ \hat{\boldsymbol{x}}(k+N_p)=\boldsymbol{A}_d^{Np}\ \boldsymbol{x}(k)+\boldsymbol{A}_d^{Np-1}\boldsymbol{B}_d\boldsymbol{u}(k)+\ \cdots\ +\ \boldsymbol{B}_d\boldsymbol{u}(k+N_p-1)\end{array} . \tag{7}$$

Then the system output prediction is

$$\begin{array}{ll}\hat{\boldsymbol{y}}(k) & =\boldsymbol{C}\boldsymbol{x}(k)\\ \hat{\boldsymbol{y}}(k+1) & =\boldsymbol{C}\boldsymbol{x}(k+1)=\boldsymbol{C}\boldsymbol{A}_d\boldsymbol{x}(k)+\boldsymbol{C}\boldsymbol{B}_d\boldsymbol{u}(k)\\ \hat{\boldsymbol{y}}(k+2) & =\boldsymbol{C}\boldsymbol{x}(k+2)=\boldsymbol{C}\boldsymbol{A}_d^2\boldsymbol{x}(k)+\boldsymbol{C}\boldsymbol{A}_d\boldsymbol{B}_d\boldsymbol{u}(k)+\boldsymbol{C}\boldsymbol{B}_d\boldsymbol{u}(k+1)\\ \quad\vdots & \qquad\vdots \qquad\qquad \ddots\\ \hat{\boldsymbol{y}}(k+N_p) & =\boldsymbol{C}\boldsymbol{A}_d^{Np}\ \boldsymbol{x}(k)\ +\ \boldsymbol{C}\boldsymbol{A}_d^{Np-1}\boldsymbol{B}_d\boldsymbol{u}(k)+\ \cdots\ +\boldsymbol{C}\boldsymbol{B}_d\boldsymbol{u}(k+N_p-1)\end{array} , \tag{8}$$

that can be written in a matrix form

$$\hat{\boldsymbol{y}}=\boldsymbol{V}\boldsymbol{x}(k)+\boldsymbol{G}\boldsymbol{u}\ , \tag{9}$$

where

$$\hat{\boldsymbol{y}}=\left[\hat{\boldsymbol{y}}(k+1)\quad \hat{\boldsymbol{y}}(k+2)\quad \cdots\quad \hat{\boldsymbol{y}}(k+N_p)\right]^T,\ \boldsymbol{u}=\left[\boldsymbol{u}(k)\quad \boldsymbol{u}(k+1)\quad \cdots\quad \boldsymbol{u}(k+N_p-1)\right]^T,$$

$$\boldsymbol{V}=\begin{pmatrix}\boldsymbol{C}\boldsymbol{A}_d\\ \vdots\\ \boldsymbol{C}\boldsymbol{A}_d^{Np}\end{pmatrix},\ \boldsymbol{G}=\begin{pmatrix}\boldsymbol{C}\boldsymbol{B}_d & \boldsymbol{0} & \\ \vdots & \ddots & \ddots\\ \boldsymbol{C}\boldsymbol{A}_d^{Np-1}\boldsymbol{B}_d & \cdots & \boldsymbol{C}\boldsymbol{B}_d\end{pmatrix}.$$

In equation (9) the term $\boldsymbol{V}\boldsymbol{x}(k)$ represents the *free response* $\boldsymbol{y}_0$ and the term $\boldsymbol{G}\boldsymbol{u}$ the *forced response* of system. In the case that the control horizon N_u is considered during computing the optimal control action sequence, it is necessary to multiply the matrix $\boldsymbol{G}$ by matrix $\boldsymbol{U}$ from right: $\boldsymbol{G}\leftarrow\boldsymbol{G}\boldsymbol{U}$, where matrix $\boldsymbol{U}$ has form

$$\boldsymbol{U} = \begin{pmatrix} \boldsymbol{I} & & \\ & \ddots & \\ & & \boldsymbol{I} \\ & & \vdots \\ & & \boldsymbol{I} \end{pmatrix} \text{ with dimension } \left[nu \cdot N_p\right] \times \left[nu \cdot N_u\right]. \tag{10}$$

The basic mathematical fundamental for the first part of the flow chart diagram depicted in Fig. 3 has been shown in this section.

3.2 Computation of the Optimal Control Action in SMPC

In this section, the mathematical fundamental for the second part of the flow chart diagram depicted in Fig. 3 is derived.

The matrix form of criterion J_{MPC} (6) is

$$J_{\mathrm{MPC}} = \left(\hat{\boldsymbol{y}} - \boldsymbol{w}\right)^T \boldsymbol{Q} \left(\hat{\boldsymbol{y}} - \boldsymbol{w}\right) + \boldsymbol{u}^T \boldsymbol{R} \boldsymbol{u}, \tag{11}$$

where matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ are diagonal with particular dimension and created from weighing coefficients Q_e and R_u ($\boldsymbol{Q} = Q_e \boldsymbol{I}$, $\boldsymbol{R} = R_u \boldsymbol{I}$).

After the predictor (9) substitution into the criterion (11) and multiplication we can obtain the equation

$$J_{\mathrm{MPC}} = \boldsymbol{u}^T \left(\boldsymbol{G}^T \boldsymbol{Q} \boldsymbol{G} + \boldsymbol{R}\right) \boldsymbol{u} + \left[\left(\boldsymbol{V}\boldsymbol{x}(k) - \boldsymbol{w}\right)^T \boldsymbol{Q}\boldsymbol{G}\right] \boldsymbol{u} + \boldsymbol{u}^T \left[\boldsymbol{G}^T \boldsymbol{Q} \left(\boldsymbol{V}\boldsymbol{x}(k) - \boldsymbol{w}\right)\right] + c, \tag{12}$$

from which on the basis of condition of minimum $\frac{\partial J_{\mathrm{MPC}}}{\partial \boldsymbol{u}} \overset{!}{=} \boldsymbol{0}$ and with using equations for vector derivation [6]

$$\frac{\partial}{\partial \boldsymbol{u}}\left(\boldsymbol{u}^T \boldsymbol{H} \boldsymbol{y}\right) = \boldsymbol{H}\boldsymbol{y}, \quad \frac{\partial}{\partial \boldsymbol{u}}\left(\boldsymbol{y}^T \boldsymbol{H} \boldsymbol{u}\right) = \boldsymbol{H}^T \boldsymbol{y}, \quad \frac{\partial}{\partial \boldsymbol{u}}\left(\boldsymbol{u}^T \boldsymbol{H} \boldsymbol{u}\right) = \boldsymbol{H}\boldsymbol{u} + \boldsymbol{H}^T \boldsymbol{u}, \tag{13}$$

it is possible to derive an equation for the sequence of optimal control action

$$\boldsymbol{u} = -\boldsymbol{H}^{-1} \boldsymbol{g}, \tag{14}$$

where $\boldsymbol{H} = \boldsymbol{G}^T \boldsymbol{Q} \boldsymbol{G} + \boldsymbol{R}$ and $\boldsymbol{g}^T = \left(\boldsymbol{V}\boldsymbol{x}(k) - \boldsymbol{w}\right)^T \boldsymbol{Q}\boldsymbol{G}$.

It is also possible to ensure the rate of control action $\Delta\boldsymbol{u}$ weighting in the criterion (11) by $\Delta\boldsymbol{u}$ expression with formula $\Delta\boldsymbol{u} = \boldsymbol{D}_i \boldsymbol{u} - \boldsymbol{u}_k$, where

$$\boldsymbol{D}_i = \begin{pmatrix} \boldsymbol{I} & \boldsymbol{0} & \cdots & \cdots & \boldsymbol{0} \\ -\boldsymbol{I} & \boldsymbol{I} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & -\boldsymbol{I} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \boldsymbol{0} \\ \boldsymbol{0} & \cdots & \boldsymbol{0} & -\boldsymbol{I} & \boldsymbol{I} \end{pmatrix} \text{ and } \boldsymbol{u}_k = \begin{pmatrix} \boldsymbol{u}(k-1) \\ \boldsymbol{0} \\ \vdots \\ \boldsymbol{0} \end{pmatrix}. \tag{15}$$

Subsequently $\boldsymbol{H} = \boldsymbol{G}^T\boldsymbol{Q}\boldsymbol{G} + \boldsymbol{D}_i^T\boldsymbol{R}\boldsymbol{D}_i$ and $\boldsymbol{g}^T = \left(\boldsymbol{V}\boldsymbol{x}(k) - \boldsymbol{w}\right)^T \boldsymbol{Q}\boldsymbol{G} - \boldsymbol{u}_k^T\boldsymbol{R}\boldsymbol{D}_i$.

It is necessary to use the *quadprog* function (4) for computing the optimal control action sequence, which should be limited by the given constraints, whereby the particular values of matrix $\boldsymbol{H}$ and vector $\boldsymbol{g}$ depend on the control action weighting manner in the criterion (11). It is necessary to compose matrix $\boldsymbol{A}_{con}$ and vector $\boldsymbol{b}_{con}$ in compliance with required constraints:

- for the rate of control action constraints $\Delta u_{\min} \le \Delta u \le \Delta u_{\max}$

$$\boldsymbol{A}_{con} = \begin{pmatrix} \boldsymbol{D}_i \\ -\boldsymbol{D}_i \end{pmatrix},\ \boldsymbol{b}_{con} = \begin{pmatrix} \boldsymbol{1}\Delta u_{\max} + \boldsymbol{u}_k \\ -\boldsymbol{1}\Delta u_{\min} - \boldsymbol{u}_k \end{pmatrix}, \tag{16}$$

- for the value of control action constraints $u_{\min} \le u \le u_{\max}$

$$\boldsymbol{A}_{con} = \begin{pmatrix} \boldsymbol{I} \\ -\boldsymbol{I} \end{pmatrix},\ \boldsymbol{b}_{con} = \begin{pmatrix} \boldsymbol{1}u_{\max} \\ -\boldsymbol{1}u_{\min} \end{pmatrix}, \tag{17}$$

- for system output constraints $y_{\min} \le y \le y_{\max}$

$$\boldsymbol{A}_{con} = \begin{pmatrix} \boldsymbol{G} \\ -\boldsymbol{G} \end{pmatrix},\ \boldsymbol{b}_{con} = \begin{pmatrix} \boldsymbol{1}y_{\max} - \boldsymbol{V}\boldsymbol{x}(k) \\ -\boldsymbol{1}y_{\min} + \boldsymbol{V}\boldsymbol{x}(k) \end{pmatrix}, \tag{18}$$

where $\boldsymbol{I}$ is an unit matrix, $\boldsymbol{1}$ denotes an unit vector, $\boldsymbol{D}_i$ and $\boldsymbol{u}_k$ are the matrix and vector from equation (15), $\boldsymbol{G}$ and $\boldsymbol{V}\boldsymbol{x}(k)$ are from predictor equation (9).

4 Predictive Control Algorithm Based on the ARX Model Design

This algorithm belongs to set of generalized predictive control (GPC) algorithms, i.e. it is based on the input-output description of dynamic systems.

Particularly, this algorithm is based on the regression ARX model

$$A_z(z^{-1})y(k) = B_z(z^{-1})u(k) + \xi(k)\ , \tag{19}$$

where $B_z(z^{-1})$ is m ordered polynomial numerator with coefficients b_i, $A_z(z^{-1})$ is n ordered polynomial denominator with coefficients a_i, $u(k)$ is an input, $y(k)$ is an output of dynamic system and $\xi(k)$ is a system output error or a noise of output measurement [9].

The control structure with GPC algorithm is depicted in Fig. 4, whereby the meaning of particular parameters is the same as in Fig. 2. Additionally, $\boldsymbol{u}_n$ and $\boldsymbol{y}_n$ are vectors of control action values and system output values in n previous samples; n is system's order.

The flow chart, which served as the basis for programming the function of SMPC algorithm (depicted in Fig. 3) is very similar to the flow chart for algorithmic design of GPC algorithm considered in this paper. However, they differ each other in some blocks (steps), which treat the specific data of GPC algorithm.

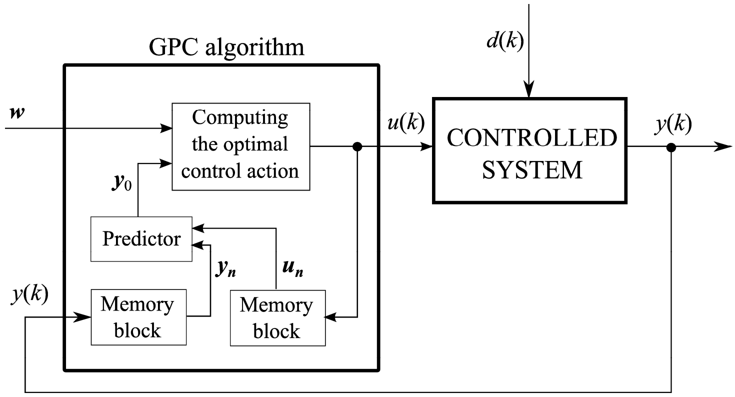


Figure 4

Control structure with GPC algorithm

The next subsections contain the mathematical description of two phases of the GPC algorithm based on the ARX model design.

4.1 Predictor Derivation in GPC Based on the ARX Model

According to [9], provided that $b_0 = 0$ along with $\xi(k) = 0$, we can express the output of dynamic system in sample $k + 1$ from the ARX model (19) by equation

$$y(k+1) = \sum_{i=1}^{n} b_i u(k-i+1) - \sum_{i=1}^{n} a_i y(k-i+1)\,. \tag{20}$$

We are able to arrange (20) into a matrix form:

$$\begin{pmatrix} y(k-n+2) \\ \vdots \\ y(k) \\ y(k+1) \end{pmatrix} = \begin{pmatrix} 0 & 1 & & \\ \vdots & & \ddots & \\ 0 & & & 1 \\ -a_n & & \cdots & -a_1 \end{pmatrix} \begin{pmatrix} y(k-n+1) \\ \vdots \\ y(k-1) \\ y(k) \end{pmatrix} + \begin{pmatrix} 0 & \cdots & 0 \\ \vdots & & \vdots \\ 0 & \cdots & 0 \\ b_n & \cdots & b_1 \end{pmatrix} \begin{pmatrix} u(k-n+1) \\ \vdots \\ u(k-1) \\ u(k) \end{pmatrix}, \quad (21)$$

$$\boldsymbol{X}(k+1) = \boldsymbol{A} \quad \boldsymbol{X}(k) + \boldsymbol{B}_0 \quad \boldsymbol{u}(k)$$

$$y(k) = \begin{pmatrix} 0 & \cdots & 0 & 1 \end{pmatrix} \boldsymbol{X}(k)$$

$$y(k) = \boldsymbol{C} \quad \boldsymbol{X}(k)$$

which represents a “pseudostate“ space model of a dynamic system [10].

By the derivation mentioned in [10] or [12], it is possible to express the predictor from the pseudostate space model as

$$\begin{pmatrix} \hat{y}(k+1) \\ \vdots \\ \hat{y}(k+N_p) \end{pmatrix} = \begin{pmatrix} \boldsymbol{CA} \\ \vdots \\ \boldsymbol{CA}^{Np} \end{pmatrix} \begin{pmatrix} y(k-n+1) \\ \vdots \\ y(k) \end{pmatrix} + \begin{pmatrix} \boldsymbol{CB}_0 & \cdots & 0 \\ \ddots & & \vdots \\ & \boldsymbol{CB}_{Np-1} & \end{pmatrix} \begin{pmatrix} u(k-n+1) \\ \vdots \\ u(k+N_p-1) \end{pmatrix}$$

$$\hat{\boldsymbol{y}} = \bar{\boldsymbol{y}}_0 + \bar{\boldsymbol{G}} \quad \bar{\boldsymbol{u}}$$

$$\hat{\boldsymbol{y}} = \bar{\boldsymbol{y}}_0 + \bar{\boldsymbol{G}}_{(:,1:n-1)} \begin{pmatrix} u(k-n+1) \\ \vdots \\ u(k-1) \end{pmatrix} + \bar{\boldsymbol{G}}_{(:,n:n+Np-1)} \begin{pmatrix} u(k) \\ \vdots \\ u(k+N_p-1) \end{pmatrix}, \quad (22)$$

$$\hat{\boldsymbol{y}} = \boldsymbol{y}_0 + \boldsymbol{G}_{Np} \quad \boldsymbol{u}$$

where $\boldsymbol{y}_0$ introduces the *free response* and $\boldsymbol{G}_{Np}\boldsymbol{u}$ represents the *forced response* of the system. Following the control horizon length, it is necessary to create a $N_p \times N_u$ matrix $\boldsymbol{U}$. It is recommended to right multiply $\boldsymbol{U}$ with $\boldsymbol{G}_{Np}$. Thus, we obtain a matrix $\boldsymbol{G} = \boldsymbol{G}_{Np}\boldsymbol{U}$, which ensures that the optimal control action will be considered over the control horizon in computing the optimal control action sequence. The final matrix form of predictor thus will be

$$\hat{\boldsymbol{y}} = \boldsymbol{y}_0 + \boldsymbol{G}\boldsymbol{u}. \quad (23)$$

4.2 Computation of the Optimal Control Action in GPC Based on the ARX Model

The algorithm for SISO system minimizes the criterion

$$J_{\text{ARX}} = \sum_{i=N_1}^{N_p} \left\{ Q_e \left[\hat{y}(k+i) - w(k+i) \right] \right\}^2 + \sum_{i=1}^{N_u} \left\{ R_u \left[u(k+i-1) \right] \right\}^2, \quad (24)$$

which in contrast to the SMPC algorithm powers also weighing coefficients.

The criterion J_{ARX} (24) has the matrix form

$$J_{\mathrm{ARX}} = \left((\hat{\boldsymbol{y}} - \boldsymbol{w})^T \quad \boldsymbol{u}^T \right) \begin{pmatrix} \boldsymbol{Q} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R} \end{pmatrix}^T \begin{pmatrix} \boldsymbol{Q} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R} \end{pmatrix} \begin{pmatrix} \hat{\boldsymbol{y}} - \boldsymbol{w} \\ \boldsymbol{u} \end{pmatrix}, \tag{25}$$

where the matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ are created from weighing coefficients Q_e and R_u with dimensions $N_p \times N_p$ and $N_u \times N_u$.

According to [10], it is sufficient to minimize only one part:

$$J_{\mathrm{k}} = \begin{pmatrix} \boldsymbol{Q} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R} \end{pmatrix} \begin{pmatrix} \hat{\boldsymbol{y}} - \boldsymbol{w} \\ \boldsymbol{u} \end{pmatrix} = \begin{pmatrix} \boldsymbol{QG} \\ \boldsymbol{R} \end{pmatrix} \boldsymbol{u} - \begin{pmatrix} \boldsymbol{Q}(\boldsymbol{w} - \boldsymbol{y}_0) \\ \boldsymbol{0} \end{pmatrix}. \tag{26}$$

According to [10], the minimization of J_{k} (26) is based on solving the algebraic equations, which are written in a matrix form, when the value of control action $\boldsymbol{u}$ or its rate $\Delta\boldsymbol{u}$ is weighted in the criterion J_{ARX} (25):

$$\underbrace{\begin{pmatrix} \boldsymbol{QG} \\ \boldsymbol{R} \end{pmatrix}}_{\boldsymbol{S}} \boldsymbol{u} - \underbrace{\begin{pmatrix} \boldsymbol{Q}(\boldsymbol{w} - \boldsymbol{y}_0) \\ \boldsymbol{0} \end{pmatrix}}_{\boldsymbol{T}} = \boldsymbol{0} \quad \text{or} \quad \underbrace{\begin{pmatrix} \boldsymbol{QG} \\ \boldsymbol{RD}_i \end{pmatrix}}_{\boldsymbol{S}} \boldsymbol{u} - \underbrace{\begin{pmatrix} \boldsymbol{Q}(\boldsymbol{w} - \boldsymbol{y}_0) \\ \boldsymbol{Ru}_k \end{pmatrix}}_{\boldsymbol{T}} = \boldsymbol{0}. \tag{27}$$

Since the matrix $\boldsymbol{S}$ is not squared, it is possible to use pseudo-inversion for the optimal control $\boldsymbol{u}$ computing, which is the solution of the system of equations (27)

$$\boldsymbol{u} = (\boldsymbol{S}^T \boldsymbol{S})^{-1} \boldsymbol{S}^T \boldsymbol{T} \tag{28}$$

or according to [11], by the QR-decomposition of matrix $\boldsymbol{S}$, where a transformational matrix $\boldsymbol{Q}_t$ transforms the matrix $\boldsymbol{S}$ to upper triangular matrix $\boldsymbol{S}_t$ as shown in Fig. 4:

$$\begin{aligned} \boldsymbol{Su} &= \boldsymbol{T} \qquad / \times \boldsymbol{Q}_t^T \\ \boldsymbol{Q}_t^T \boldsymbol{Su} &= \boldsymbol{Q}_t^T \boldsymbol{T} \\ \begin{pmatrix} \boldsymbol{S}_t \\ \boldsymbol{0} \end{pmatrix} \boldsymbol{u} &= \begin{pmatrix} \boldsymbol{T}_t \\ \boldsymbol{T}_z \end{pmatrix} \end{aligned} \tag{29}$$

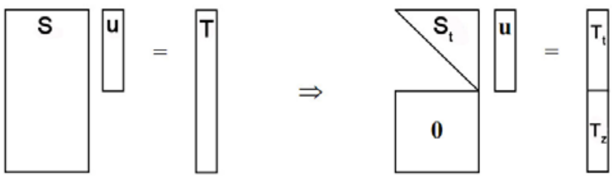


Figure 5
Matrix transformation with QR decomposition

It results from above-mentioned that the optimal control $\boldsymbol{u}$ can also be computed by formula

$$\boldsymbol{u} = \boldsymbol{S}_t^{-1} \boldsymbol{T}_t. \tag{30}$$

The matrix $\boldsymbol{H}$ and vector $\boldsymbol{g}$, which are the input parameters of *quadprog* function (if the optimal control computing with constraints is carried out), have the following form with $\boldsymbol{u}$ or $\Delta\boldsymbol{u}$ weighted in the criterion J_{ARX} (25)

$$\begin{aligned} \boldsymbol{H} &= \boldsymbol{G}^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{G} + \boldsymbol{R}^T\boldsymbol{R} \\ \boldsymbol{g}^T &= (\boldsymbol{y}_0 - \boldsymbol{w})^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{G} \end{aligned} \quad \text{or} \quad \begin{aligned} \boldsymbol{H} &= \boldsymbol{G}^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{G} + \boldsymbol{D}_i^T\boldsymbol{R}^T\boldsymbol{R}\boldsymbol{D}_i \\ \boldsymbol{g}^T &= (\boldsymbol{y}_0 - \boldsymbol{w})^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{G} - \boldsymbol{u}_k^T\boldsymbol{R}^T\boldsymbol{R}\boldsymbol{D}_i \end{aligned} \tag{31}$$

and the matrix $\boldsymbol{A}_{con}$ and vector $\boldsymbol{b}_{con}$ are as well as in previous algorithm given by equations (16), (17), (18), but the system free response is constituted by vector $\boldsymbol{y}_0$ instead of the term $\boldsymbol{Vx}(k)$.

5 Predictive Control Algorithm Based on the CARIMA Model Design

Similarly to generalized predictive control (GPC) algorithm based on the ARX model, this algorithm also belongs to the GPC algorithms family, but it is based on the CARIMA model of dynamic systems

$$A_z(z^{-1})y(k) = B_z(z^{-1})u(k-1) + \frac{C_z(z^{-1})}{\Delta}\xi(k), \tag{32}$$

Where, in contrast to the ARX model, $C_z(z^{-1})$ is multi-nominal and $\Delta = 1 - z^{-1}$ introduces an integrator [8].

According to [8], the criterion that is minimized in this GPC algorithm has the form

$$J_{\text{CARIMA}} = \sum_{i=N_1}^{N_p}\left[P(z^{-1})\hat{y}(k+i) - w(k+i)\right]^2 + \lambda\sum_{i=1}^{N_u}\left[\Delta u(k+i-1)\right]^2, \tag{33}$$

where in contrast to the previous algorithm, λ is a relative weighing coefficients that expresses a weight ratio between the deviation $y(k) - w(k)$ and the control action $u(k)$. $P(z^{-1})$ provides the same effect as equation (2) in the first part of paper. According to [8], the corresponding first order filter for constant reference trajectory is $P(z^{-1}) = \dfrac{1-\alpha z^{-1}}{1-\alpha}$, where $\alpha \in \langle 0;1 \rangle$.

Next, we will restrict ourselves to $\alpha = 0$, i.e. $P(z^{-1}) = 1$.

The next subsections contain the mathematical description of two phases of the GPC algorithm based on the CARIMA model design.

5.1 The Predictor Derivation in GPC Algorithm Based on the the CARIMA Model

According to [8], the output of the dynamic system that is defined by equation (32) in sample instant $k + 1$ is given by the following equation (for notation convenience without z^{-1}):

$$y(k+j) = \frac{B_z}{A_z} u(k+j-1) + \frac{C_z}{\Delta A_z} \xi(k+j) \ . \tag{34}$$

On the basis of the derivation mentioned in [8] and with polynomial dividing

$$\frac{C_z(z^{-1})}{\Delta A_z(z^{-1})} = E_j(z^{-1}) + z^{-1} \frac{F_j(z^{-1})}{\Delta A_z(z^{-1})} \quad \text{and} \quad \frac{B_z(z^{-1})E_j(z^{-1})}{C_z(z^{-1})} = G_j(z^{-1}) + z^{-j} \frac{\Gamma(z^{-1})}{C_z(z^{-1})}$$

or alternatively by solving diophantine equations

$$C_z = E_j \Delta A_z + z^{-j} F_j \quad \text{a} \quad B_z E_j = G_j C_z + z^{-j} \Gamma_j$$

we are able to express the j steps predictor in the matrix form

$$\hat{\boldsymbol{y}} = \boldsymbol{G}\Delta\boldsymbol{u} + \boldsymbol{y}_0 , \tag{35}$$

in which the rate of control action $\Delta\boldsymbol{u}$ is present directly.

The form of the matrix $\boldsymbol{G}$ is $\boldsymbol{G} = \begin{pmatrix} g_0 & 0 & \cdots & \cdots & 0 \\ g_1 & g_0 & 0 & \cdots & 0 \\ \vdots & & \ddots & & \\ \vdots & & & g_0 & 0 \\ g_{N_p-1} & & \cdots & & g_0 \end{pmatrix}$,

where coefficients g_i can be obtained by division $B / \Delta A$ and $\boldsymbol{y}_0$ is the *free response* of system. If we take value N_1 into consideration, we will able to remove first $N_1 - 1$ rows of matrix $\boldsymbol{G}$. Moreover, regarding the control horizon, only the first N_u columns of matrix G are necessary for the next calculations. Thus, the reduced matrix $\boldsymbol{G}$ will have dimension $(N_p - N_1 + 1) \times N_u$ [8].

5.2 Computation of the Optimal Control Action in the GPC Algorithm Based on the CARIMA Model

It is possible to express the criterion J_{CARIMA} (33) in the matrix form

$$\begin{aligned} J_{\text{CARIMA}} &= (\hat{\boldsymbol{y}} - \boldsymbol{w})^T (\hat{\boldsymbol{y}} - \boldsymbol{w}) + \lambda \Delta\boldsymbol{u}^T \Delta\boldsymbol{u} = \\ &= (\boldsymbol{G}\Delta\boldsymbol{u} + \boldsymbol{y_0} - \boldsymbol{w})^T \ (\boldsymbol{G}\Delta\boldsymbol{u} + \boldsymbol{y_0} - \boldsymbol{w}) + \lambda \Delta\boldsymbol{u}^T \Delta\boldsymbol{u} = \\ &= c + 2\boldsymbol{g}^T \Delta\boldsymbol{u} + \Delta\boldsymbol{u}^T \boldsymbol{H} \Delta\boldsymbol{u}, \end{aligned} \tag{36}$$

where $\boldsymbol{H} = \boldsymbol{G}^T\boldsymbol{G} + \lambda\boldsymbol{I}$, $\boldsymbol{g}^T = (\boldsymbol{y_0} - \boldsymbol{w})^T \boldsymbol{G}$, c is a constant and $\boldsymbol{I}$ is a unit matrix.

If it is necessary to weigh the value of control action $\boldsymbol{u}$ in the criterion (36); it is possible to obtain $\boldsymbol{H} = \boldsymbol{G}^T\boldsymbol{G} + \lambda \boldsymbol{D}_i^{-T}\boldsymbol{D}_i^{-1}$, $\boldsymbol{g}^T = (\boldsymbol{y_0} - \boldsymbol{w})^T\boldsymbol{G} + \lambda \boldsymbol{u}_k^T \boldsymbol{D}_i^{-T}\boldsymbol{D}_i^{-1}$ on the basis of formula (15) in the first paper.

Following the condition of minimum $\frac{\partial J_{\text{CARIMA}}}{\partial \Delta \boldsymbol{u}} \overset{!}{=} \boldsymbol{0}$, it is easy to express the equation for the optimal control action in disregard of required constraints

$$\Delta \boldsymbol{u} = -\boldsymbol{H}^{-1}\boldsymbol{g}\ . \tag{37}$$

The optimal control computing with regard to the required constraints can be carried out by the *quadprog* function, whereby the matrix $\boldsymbol{A}_{con}$ and vector $\boldsymbol{b}_{con}$ are:

- for the rate of control action constraints $\Delta u_{\min} \le \Delta u \le \Delta u_{\max}$

$$\boldsymbol{A}_{obm} = \begin{pmatrix} \boldsymbol{I} \\ -\boldsymbol{I} \end{pmatrix},\ \boldsymbol{b}_{obm} = \begin{pmatrix} \boldsymbol{1}\Delta u_{\max} \\ -\boldsymbol{1}\Delta u_{\min} \end{pmatrix}, \tag{38}$$

- for the value of control action constraints $u_{\min} \le u \le u_{\max}$

$$\boldsymbol{A}_{obm} = \begin{pmatrix} \boldsymbol{L} \\ -\boldsymbol{L} \end{pmatrix},\ \boldsymbol{b}_{obm} = \begin{pmatrix} \boldsymbol{1}u_{\max} - \boldsymbol{1}u(k-1) \\ -\boldsymbol{1}u_{\min} + \boldsymbol{1}u(k-1) \end{pmatrix}, \tag{39}$$

- for system output constraints $y_{\min} \le y \le y_{\max}$

$$\boldsymbol{A}_{obm} = \begin{pmatrix} \boldsymbol{G} \\ -\boldsymbol{G} \end{pmatrix},\ \boldsymbol{b}_{obm} = \begin{pmatrix} \boldsymbol{1}y_{\max} - \boldsymbol{y}_0 \\ -\boldsymbol{1}y_{\min} + \boldsymbol{y}_0 \end{pmatrix}, \tag{40}$$

where $\boldsymbol{I}$ is an unit matrix, $\boldsymbol{1}$ is an unit vector and $\boldsymbol{L}$ is a lower triangular matrix inclusive of ones. It is necessary to realize that the vector of control action rate $\Delta\boldsymbol{u}$ is the result of optimal control computing in this case.

Equally, as in the previous GPC algorithm, we programmed the GPC algorithm based on the CARIMA model as a Matlab function on the basis of the designed flow chart diagram in Fig. 3, with particular modifications, which are clear from theoretical background of the algorithm.

6 Predictive Control Algorithms Verification on the Helicopter Model

We introduced the basic principle, theoretical background and the manner of forming the algorithms of three different predictive control algorithms in the previous sections. Next we are concerned with using them in a laboratory helicopter model control by Matlab with implemented Real-Time Toolbox functions.

6.1 The Real Laboratory Helicopter Model

The educational helicopter model constitutes a multivariable nonlinear dynamic system with three inputs and two measured outputs, as depicted in Fig. 6.

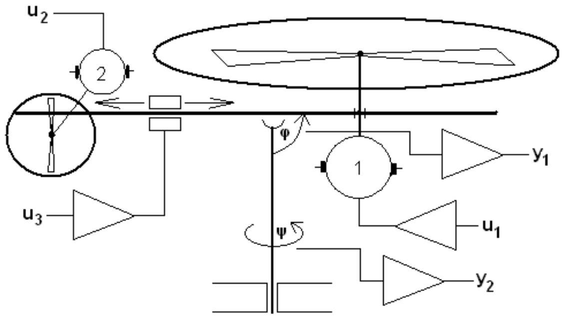


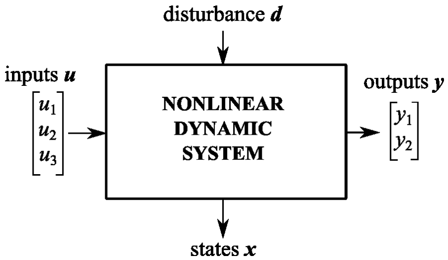
Figure 6
Mechanical system of real laboratory Helicopter model

The model is composed of a body with two propellers, which have their axes perpendicular and are driven by small DC motors; i.e. the helicopter model constitutes a system with two degrees of freedom [5]. The movement in the direction of axis y (elevation = output y_1) presents the first degree of freedom, and the second degree of freedom is presented by the movement in the direction of axis x (azimuth = output y_2). The values of both the helicopter's angular displacements are influenced by the propellers' rotation. The angular displacements (φ – angle for elevation, ψ – angle for azimuth) are measured by incremental encoders.

The DC motors are driven by power amplifiers using pulse width modulation, whereby a voltage introduced to motors (u_1 and u_2) is directly proportional to the computer output. The volatge u_3 serves for controlling the center of gravity, which constitutes a system's disturbance. It is necessary to note that we did not consider this during the design of the control algorithms. The model is connected to the computer by a multifunction card MF614, which communicates with the computer by functions of Real Time Toolbox [13].

The system approach of the real laboratory helicopter model and constraints of the inputs and outputs are shown in Fig. 7.

On the basis of helicopter's mathematic-physical description, mentioned in the manual [5], we can redraw Fig. 7 to Fig. 8, where M_{ep} is the main propeller torque performing in the propeller direction, M_{etp} is torque performing in the turnplate direction, M_{ap} is the auxilliary propeller torque performing in the propeller direction and M_{etp} is torque performing in the turnplate direction.



$u_1 \in \langle -1,1 \rangle$ V – main motor's volatge,

$u_2 \in \langle -1,1 \rangle$ V – side motor's volatge,

$u_3 \in \langle -1,1 \rangle$ V – center of gravity input,

$y_1 \in \langle -45^\circ, 45^\circ \rangle$ – position in elevation,

Figure 7

System approach with technical parameters (constraints) of the real laboratory Helicopter model

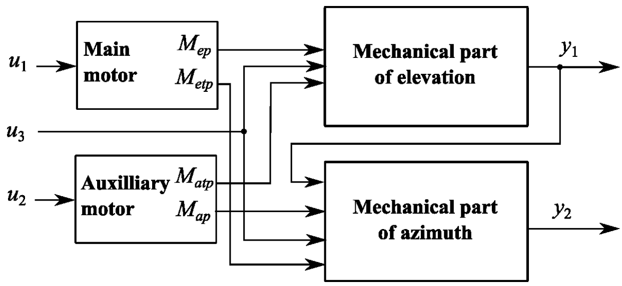


Figure 8

Subsystems of the real laboratory Helicopter model

According to mathematical description and block structure in [5], [7] or [16], it is possible to express the considered system's dynamics by nonlinear differential equations (for simplicity we omitted *(t)* in inputs, states and outputs expression):

$$
\begin{aligned}
&\dot{x}_1 = -\frac{1}{T_m}\cdot x_1 + \frac{1}{T_m}\cdot u_1 \qquad\qquad \dot{x}_2 = -\frac{1}{T_s}\cdot x_2 + \frac{1}{T_s}\cdot u_2 \\
&\dot{x}_3 = \alpha_1 \cdot x_1 \cdot |x_1| + \beta_1 \cdot x_1 + \left(\gamma_2 \cdot x_2 \cdot |x_2| + \delta_2 \cdot x_2\right)\cdot \cos\eta - J_{el}\left(u_3\right)\cdot \delta_{el}\cdot x_3 - M_g\left(u_3\right)\cdot \cos x_4 \\
&\dot{x}_4 = \frac{1}{J_{el}\left(u_3\right)}\cdot x_3 \\
&\dot{x}_5 = \left(\alpha_2 \cdot x_2 \cdot |x_2| + \beta_2 \cdot x_2\right)\cdot \cos\left(x_4 + \eta\right) + \left(\gamma_1 \cdot x_1 \cdot |x_1| + \delta_1 \cdot x_1\right)\cdot \cos x_4 - J_{az}(u_3).\delta_{az}\cdot x_5 \\
&\dot{x}_6 = \frac{1}{J_{az}\left(u_3\right)\cdot \cos x_4}\cdot x_5 \\
&y_1 = \frac{1}{\pi}\cdot x_4 \qquad\qquad y_2 = \frac{1}{\pi}\cdot x_6
\end{aligned}
\tag{41}
$$

where x_1 is the rotation speed of main motor, x_2 is the rotation speed of the auxiliary motor, x_3 is the rotation speed of the model in elevation, x_4 is the position of the model in elevation, x_5 is the rotation speed of the model in azimuth, x_6 is the position of the model in azimuth, T_m and T_s are the time constants of the main and auxilliary motors, δ_{el} and δ_{az} is the friction constant in elevation and azimuth. The moment of gravity M_g, the moment of inertia in elevetaion J_{el}, and the azimuth J_{az} depend on u_3. These dependencies M_g, J_{el}, J_{az} and parameters α_1, α_2, β_1, β_2, γ_1, γ_2, δ_1, δ_2, η are introduced in detail in [5] or [7].

Note that it is also possible to express the model dynamics of the helicopter by nonlinear mathematical description introduced in [7] or [16].

As this paper has considered predictive control algorithms based on the flow chart in Fig. 3 and the utilization of the linear model of dynamic system, it is necessary to carry out the Taylor linearization of equations (41) in an operating point $P \equiv [\boldsymbol{x}_E, \boldsymbol{u}_E]$:

$$\begin{aligned}\dot{\boldsymbol{x}}(t) &= \boldsymbol{A}_C \boldsymbol{x}(t) + \boldsymbol{B}_C \boldsymbol{u}(t) \\ y(t) &= \boldsymbol{C}_C \boldsymbol{x}(t)\end{aligned}, \quad \text{where} \quad \boldsymbol{A}_C = \left[\frac{\partial \boldsymbol{f}_i}{\partial x_j}\right]_{\substack{\boldsymbol{x}=\boldsymbol{x}_E \\ \boldsymbol{u}=\boldsymbol{u}_E}} \quad \boldsymbol{B}_C = \left[\frac{\partial \boldsymbol{f}_i}{\partial u_j}\right]_{\substack{\boldsymbol{x}=\boldsymbol{x}_E \\ \boldsymbol{u}=\boldsymbol{u}_E}}. \tag{42}$$

For the purpose of linearization, we considered the operating point $P \equiv [\boldsymbol{x}_E, \boldsymbol{u}_E]$, in which angular displacements in elevation and azimuth were zero: $y_1 = 0$, $y_2 = 0$. The forms of matrices $\boldsymbol{A}_C$, $\boldsymbol{B}_C$, $\boldsymbol{C}_C$, describing the helicopter's continuous state space model in operating point P are

$$\boldsymbol{A}_C = \begin{bmatrix} A_{11} & 0 & 0 & 0 & 0 & 0 \\ 0 & A_{22} & 0 & 0 & 0 & 0 \\ A_{31} & A_{32} & A_{33} & A_{34} & 0 & 0 \\ 0 & 0 & A_{43} & 0 & 0 & 0 \\ A_{51} & A_{52} & 0 & A_{54} & A_{55} & 0 \\ 0 & 0 & 0 & A_{64} & A_{65} & 0 \end{bmatrix} \boldsymbol{B}_C = \begin{bmatrix} B_{11} & 0 \\ 0 & B_{22} \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \boldsymbol{C}_C = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ C_{14} & 0 \\ 0 & 0 \\ 0 & C_{26} \end{bmatrix}^T. \tag{43}$$

Note that the experimental identification of the laboratory helicopter model, which resulted in linear models in state-space and input-output description, was solved in [17]. In our case, we used only linear models obtained from the linearization process. The numerical values of particular elements in matrices (43) can be obtained on the basis of numerical values of model parameters in (41), which are supplied with the model from the manufacturer.

Subsequently, it is possible to create a discrete linear model from the continuous with specific sample period T_s. Then we can use the discrete linear model of helicopter dynamic system in the introduced control algorithms.

3.2 Control Structures Programming for Predictive Control Verification on the Helicopter Model

We carried out the control of the real laboratory helicopter model in accordance with the control structure for particular predictive control algorithm, which have been mentioned in this paper.

Unfortunately it is also necessary to note that the steady state deviation between reference trajectory and system output appeared in cases when the SMPC algorithm and the GPC algorithm based on the ARX model were used. Therefore, in order to eliminate this, we inserted a feedforward branch into the control structure, as seen in Fig. 9. A control component in the feedforward branch performed a function which generated steady state values of the main propeller's motor voltage for particular angular displacement. The transient characteristic between the particular angular displacement and the voltage steady state value was obtained experimentally in [15].

Thus, in cases where SMPC and GPC based on the ARX model algorithms were used, it is possible to express the entire control action by equation:

$$\boldsymbol{u}(k) = \boldsymbol{u}_{opt}(k) + \boldsymbol{u}_0(k)\,, \tag{44}$$

where $\boldsymbol{u}_{opt}(k)$ is the optimal control action computed in the function of the predictive control algorithm and $\boldsymbol{u}_0(k)$ is the control action generated by the feedforward controller.

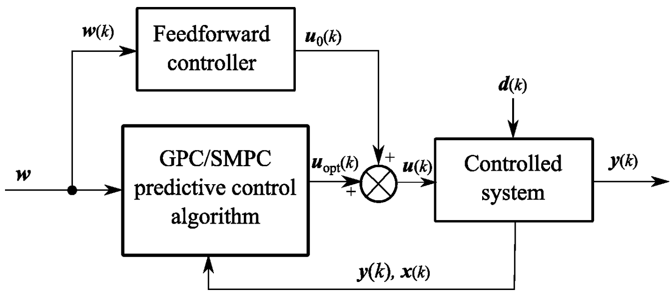


Figure 9
Control structure with feedforward branch

As has already been mentioned, we programmed each used control structures in the form of functions/scripts in Matlab, where the current helicopter's state is obtained by communication with the laboratory card using Real Time Toolbox functions [13] in the control closed loop. The data obtained are utilized by the control algorithm, which results in the particular value of control action. This value is sent back to the laboratory card, thus to the real helicopter model by Real Time Toolbox functions again. Note that we did not use Simulnik functional blocks, but only Real Time Toolbox functions for communication between the laboratory card and the helicopter model (*rtrd* for reading and *rtwr* for writing data to laboratory card).

It is necessary to note that control closed loop, which we considered, is based on the execution of the optimization problem in one sample period. Although it is possible to compute some matrices and vectors in advance, computing the system free response and optimal control action by quadratic programming must be carried out at each sample instant. In general, optimization tasks are very time-consuming and they require powerful computers. To comply with the defined sample period, we checked the calculation time spent in computing the control action by predictive control algorithm function at every control step. In the case when the calculation time exceeded the sample period, the control algorithms were interrupted. The multifunction card MF614 allowed to use the minimal sample period 1 ms. Unfortunately, in our case, with the given computer we were only able to use 30 ms.

3.3 Results of Algorithm Verification on the Helicopter Model

In this part we present the results of the real laboratory Hhlicopter model control as the time responses of control action and controlled model's outputs. The next table illustrates the settings of variable parameters' values of predictive control algorithms used. If the settings for elevation and azimuth differ from each other, they are written in two rows for particular algorithm in the Tab. 1.

Table 1

Settings of predictive control algorithms' parameters

Algorithms	T_s	N_p	N_u	Q_v	R_v	Constraint	Weighting	Fig.
SMPC	0.03s	40	1	4 400	30 4	$u_e \in \langle 0.4;0.8 \rangle$ $u_a \in \langle -1;1 \rangle$	$\Delta u(k)$	5
2x GPC SISO ARX	0.05s	25 20	1	1 10	2 1	$u_e \in \langle 0.4;0.8 \rangle$ $u_a \in \langle -1;1 \rangle$	$\Delta u(k)$	6
GPC SISO ARX	0.05s	18	1	1	1	$u_e \in \langle 0.4;0.8 \rangle$ $u_a \in \langle -1;1 \rangle$	$\Delta u(k)$	7
GPC SISO CARIMA	0.05s	10	1	3	1	$u_e \in \langle 0.4;0.8 \rangle$ $u_a \in \langle -1;1 \rangle$	$\Delta u(k)$	7

At this point, we wish to note that the real laboratory helicopter model control fulfilled the aim of control with above presented settings of algorithms. However, the results were markedly influenced by small changes in horizons and the weighing coefficients' values. On the other hand, this did not happen in simulation control, which we used as a primary test of the designed algorithms. We carried out the simulation control of the nonlinear model (41) by numerical solving with Runge-Kutta fourth order method in its own Matlab function.

In Fig. 10 are the results of the real laboratory helicopter model with two degrees of freedom control using the SMPC algorithm. As the model's states are not measured, we used the state values estimation by Kalman's predictor, which we designed on the basis of duality principle with LQ control design according to [14]. We used weighing coefficients Q_{est} = 10000 and R_{est} = 0.001 for the estimator's parameters design. Also, it is necessary to note that the feedforward branch was incorporated in the control structure in compliance with Fig. 9.

The results of MIMO system control are depicted in Fig. 11, too. However, two independent GPC algorithms for SISO systems were used as controllers, instead of one algorithm for the MIMO system. We designed GPC algorithms, which were based on the ARX model especially for elevation and for azimuth control, whereby we neglected mutual interactions and used only relevant states of the system (41). Also, the feedforward branch was incorporated in the control structure.

Fig. 12 illustrates the time responses of the real laboratory helicopter model control only in elevation direction. The model was latched; thus it was impossible to move it in the azimuth direction. The results of the control with the GPC algorithm designed for SISO systems are depicted in the figure, with the GPC algorithm based on the ARX model on left and the GPC algorithm based on the CARIMA model on the ride. It can be seen from figure that control with the GPC algorithm based on the ARX model gets better results than control with the GPC algorithm based on the CARIMA model. However, it must be stated that the feedforward branch was incorporated in the control structure together with GPC algorithm based on the ARX model.

The displayed results were obtained with the rate of control action $\Delta u(k)$ weighted in the criterion. If only the value of control action $u(k)$ was weighted, the time responses were similar to their counterparts, where $\Delta u(k)$ was weighted, but the deviation between system output and reference trajectory appeared.

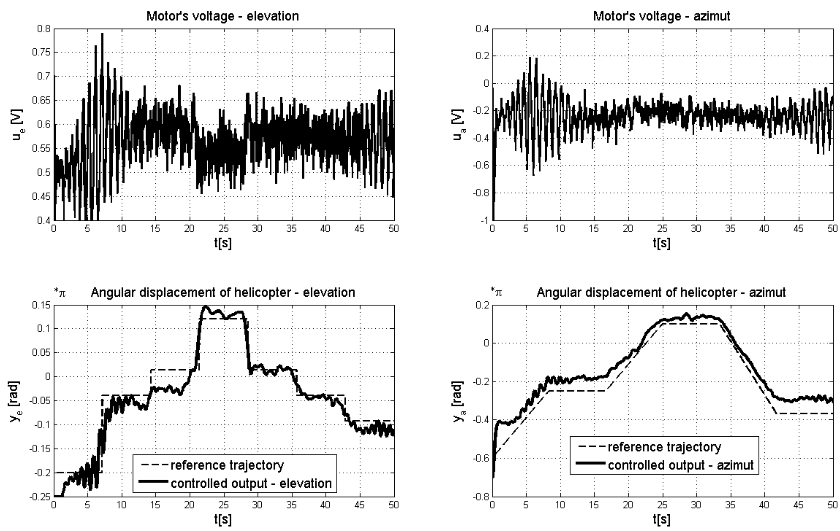


Figure 10

Time responses of Helicopter control with SMPC algorithm

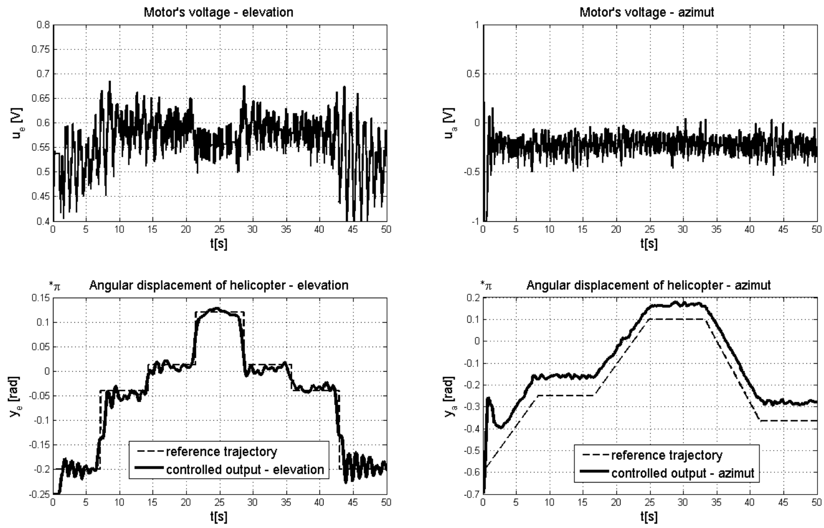


Figure 11

Time responses of Helicopter control with two GPC (ARX) algorithms

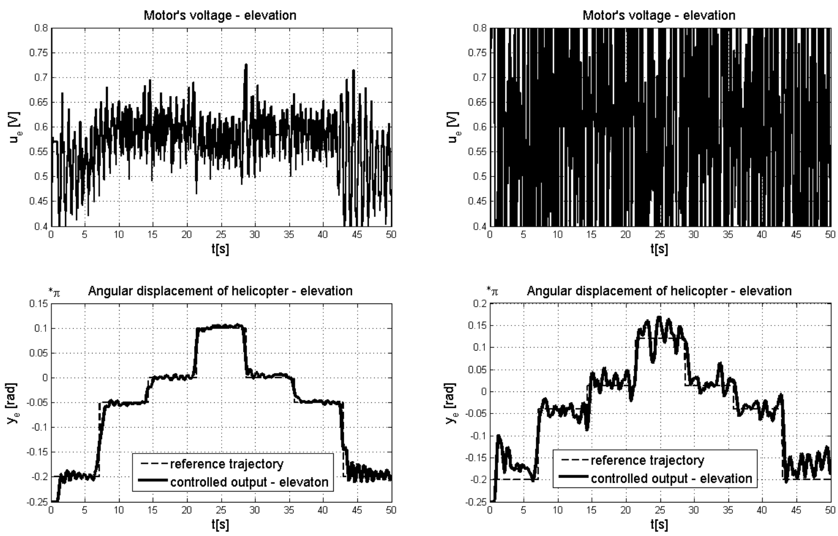


Figure 12

Time responses of Helicopter control in elevation with GPC (ARX) algorithm on left and GPC (CARIMA) algorithm on right

Although constraints of the controlled system input are given by range <-1; +1>, it is necessary to note that we reduced it to <0.4; 0.8> V for the main motor in order to obtain better performance of the control process.

Similar results were obtained by classical PID control in or optimal LQ control of the laboratory helicopter model, which were published in [15].

To accept the applicability of introduced algorithms, the comparison of the obtained results in Fig. 10 – Fig. 12 and the results published in [7] and [16] were useful, too. The time responses of the controlled system output presented in this article are quite similar to the results in [16], where a fuzzy logic controller was used. So it is possible to conclude that predictive control is a particular variant of intelligent control methods.

Conclusions

We have mentioned the theoretical basis of predictive control algorithms and the manner of their implementation to programming language Matlab in this paper. We were engaged in mathematical derivation of state space model based on predictive control algorithms and generalized predictive control algorithms based on the ARX and CARIMA models and their implementation in a control structure.

We have implemented all the mentioned algorithms in Matlab to verify them in a real laboratory helicopter model control in the Laboratory of Cybernetics at the Department of Cybernetics and Artificial Intelligence at Faculty of Electrotechnics and Informatics at the Technical University in Košice.

We have concluded from the obtained results that using GPC algorithms, which are based on the input-output description, seems to be preferable to algorithms based on the state space description of dynamic systems, mainly because it is not possible to measure the states of the helicopter model. Unfortunately, such algorithms comparing in the control of systems, whose states cannot be measured, depend very much on the type and settings of used parameters of the state estimator.

We supported the fact that it is possible to eliminate the deviation between system output and reference trajectory by control with an integration character, in our case by weighting the rate of control action $\Delta\boldsymbol{u}$ in the criterion. Unfortunately, it was valid in the real laboratory model control only when the GPC algorithm based on the CARIMA model was used and the rate of control action $\Delta\boldsymbol{u}$ appeared in the predictor expression. In other cases, when the remaining two mentioned algorithms were used and the rate of control action $\Delta\boldsymbol{u}$ did not appear in the predictor expression, but only its direct value $\boldsymbol{u}$ did, we modified the control structure by including the feedforward branch to control process. In this way it was possible to eliminate the deviation between the system output and reference trajectory.

Also we believe that a reduction in the sample period or an increase in the prediction horizon would improve control results obtained by control with the GPC algorithm based on the CARIMA model. Unfortunately, due to the predictive control algorithms computational demands, especially if the sequence of optimal control action with respect to required constraints was carried out, it was impossible to verify this assumption in our case.

Although we programmed the mentioned GPC algorithms in a broad range for MIMO dynamic systems as well, we must point out that in the laboratory helicopter model control, it was preferable to neglect any mutual interactions between system inputs and outputs and use GPC algorithms for SISO systems extra for each degree of freedom.

For future solutions to the problem of dynamic system control by predictive control algorithms, we suggest verifying the algorithms' extension by a summator of the deviation between the system output and reference trajectory, which will be particularly weighed in the criterion. This solution should include the integration character into the control process.

It is also necessary to note that predictive control insufficiency, which relates to the relatively long calculation time, is a substantial issue in fast mechatronic system control. We suppose it is possible to handle by explicit predictive control, which we also want to verify on the helicopter model. For that purpose we would like to use multi-parametric programming.

On the other hand, we can accept that the mentioned nonlinear mathematical description of the system is not precise enough. So we also plan to use a neural network that would be trained from data measured on a real laboratory model as a nonlinear predictor in nonlinear predictive control. However, it will be hard to handle computational phase, where the iteration optimization task for the minimization of the nonlinear functions must be executed at each sample instant.

We think the best solution can be found in some kind of combination of using a neural network and multi-parametric programming, as this would make it possible to generate nonlinear prediction by neural network and concurrently to compute the corresponding optimal control in advance, thus offline. We assume this way would permit controlling systems with a shorter sample period.

Acknowledgement

This work has been supported by the Scientific Grant Agency of Slovak Republic under project Vega No.1/0286/11 Dynamic Hybrid Architectures of the Multiagent Network Control Systems (50%) and by the project Development of the Center of Information and Communication Technologies for Knowledge Systems (project number: 26220120030) supported by the Research & Development Operational Program funded by the ERDF (50%).

References

[1] Clarke, D. W., Mohdani, C., Tuffs, P. S.: Generalized Predictive Control. Part 1 and 2. *Automatica*, Vol. 23, No. 2, pp. 137-160, 1987

[2] Grimble, M. J., Ordys, A. W.: Predictive Control for Industrial Applications. *Annual Reviews in Control*, 25, pp. 13-24, 2001, ISSN 1367-5788

[3] Camacho, E. F., Bordons, C.: Model Predictive Control. *Springer*, 1999

[4] Rossiter, J. A.: Model-based Predictive Control: A Practical Approach. *CRC Press*, 2004, ISBN 0-8493-1291-4

[5] Humusoft: CE150 Helicopter model, Educational Manual, 1996-2004

[6] Roubal, J, Havlena, V.: Range Control MPC Approach For Two-Dimensional System, Proceedings of the 16th IFAC World Congress, Volume 16, Part 1, 2005

[7] Dutka, Arkadiusz S., Ordys, Andrzej W., Grimble, Michael J.: Non-linear Predictive Control of 2 dof helicopter model. Proceeding of the 42nd IEEE Conference on Decision and Control, pp. 3954-3959, Hawai USA, 2003

[8] Fikar, M.: Predictive Control – An Introduction. Slovak Technical University - FCHPT, Bratislava 1999

[9] Belda, K., Böhm, J.: Adaptive Predictive Control for Simple Mechatronic Systems. *Proceedings of the WSEAS CSCC & EE International Conferences*. WSEAS Press, Athens, Greece 2006, pp. 307-312

[10] Belda, K., Böhm, J.: Adaptive Generalized Predictive Control for Mechatronic Systems. *WSEAS Transactions on Systems*. Volume 5, Issue 8, August 2006, Athens, Greece 2006, pp. 1830-1837

[11] Belda, K.: Control of Parallel Robotic Structures Driven by Electromotors. (Research Report). Czech Technical University, FEE, Prague 2004, 32 pp.

[12] Jajčišin, Š.: Application Modern Methods in Control of Non-linear Educational Models (in Slovak). Diploma thesis (Supervisor: doc. Ing. Anna Jadlovská, PhD) TU-FEI, Košice 2010

[13] Humusoft: Real-Time Toolbox, User’s manual, 1996-2002

[14] Krokavec, Dušan, Filasová, Anna: Diskrétne systémy. Košice: Technical University-FEI, 2006, ISBN 80-8086-028-9

[15] Jadlovská, A., Lonščák, R.: Design and Experimental Verification of Optimal Control Algorithm for Educational Model of Mechatronic System (in Slovak) In: Electroscope – online journal for Electrotechnics, Vol. 2008, No. I, ISSN 1802-4564

[16] Velagic, Jasmin, Osmic, Nedim: Identification and Control of 2DOF Nonlinear Helicopter Model Using Intelligent Methods. *Systems Man and Cybernetics* (*SMC*), 2010 IEEE International Conference, pp. 2267-2275, 2010

[17] Dolinský, Kamil, Jadlovská, Anna: Application of Results of experimental Identification in Control of Laboratory Helicopter Model. Advances in Electrical and Electronic Engineering, scientific reviewed Journal published in Czech Republic, Vol. 9, Issue 4, 2011, pp. 157-166, ISSN 1804 3119