Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online
CONFERENCE PAPER ARCHIVE

IFAC-PapersOnLine 49-6 (2016) 034–039

Teaching and Practicing Model Predictive Control

Daniel Honc*, Rahul Sharma K.*, Anuj Abraham,**
František Dušek*, Natarajan Pappa**

* Department of Process Control, Faculty of Electrical Engineering and Informatics,
University of Pardubice, Czech Republic
rahul.sharma@student.upce.cz,{daniel.honc,frantisek.dusek}@upce.cz
**Department of Instrumentation Engineering, Madras Institute of Technology Campus,
Anna University, Chennai, India
anuj1986aei@gmail.com, npappa@rediffmail.com

Abstract: How to explain Model Predictive Control (MPC) to students? How to practise it? The paper deals with chain of actions involving teaching, practicing and laboratory application of MPC at University of Pardubice in Czech Republic and at Anna University in India. Individual steps are presented and discussed with examples from educational experience – e.g. modelling and identification, derivation of MPC controller, simulations and laboratory applications. Every phase has a key and weak point as well. Desired results is that students understand better the theoretical concepts and they are able to apply predictive controllers at least for laboratory processes. Derivations and MATLAB scripts are available online.

Keywords: Model Predictive Control, MPC, modelling, system identification, optimization, teaching, experiments.

1. INTRODUCTION

Model predictive control is very popular and frequently used in the industry for optimal control of multivariable systems with constrains. The method is suitable for unstable or non-minimum phase systems, systems with dead-times, with different numbers of controlled and manipulated variables even for non-linear processes. The key feature is explicit use of a dynamical process model for controlled variable prediction at a future time horizon and calculation of a control actions to minimize a cost function. Future set-points or disturbances can be handled as well if available. The receding strategy concept means that at every sample time instant, only first the control action from the optimal vector is used and the horizon is shifted towards the future and the procedure is repeated again for updated system state. The various predictive algorithms differ amongst themselves in the model used to represent the process, the cost function to be minimized and optimization method.

The whole concept of MPC is straightforward and easy to understand but still enough general so special control objectives can be defined and fulfilled. Controller is designed in time domain, dynamical model of controlled process must be known and also some optimization method to get the solution is required. From this point of view, teaching and practicing of MPC is an important part of university education of process control engineers. Books about MPC are great but for master students this is quite expensive and maybe too much detailed source of information (Camacho and Bordons, 2007), (Rossiter, 2003), (Maciejowski, 2002), (Kouvaritakis and Cannon, 2016), (Rawlings and Mayne, 2009), (Wang, 2009), (Mareš and Hrnčiřík, 2012), (Mikleš and Fikar, 2004). Online books and tutorials can be good alternative for most of the students (Borrelli et al., 2015), (Rossiter 2014), (Bemporad, 2009), (Boyd, 2008), (Jay, 2005), (Pekař, 2010). The interesting task is how to explain MPC to students, how to practise their theoretical knowledge and what tools to use. Educational framework based on the Lego Mindstorms NXT robotic platform with two-wheeled inverted pendulum experiments was published in (Canale and Casale-Brunet, 2014). Adaptive cruise control with LabVIEW, National Intruments Robotics Starter Kit robot and code deployed on FPGA was presented in (Shakouri et al., 2013). (Richmond and Chen, 2012) created software package for teaching chemical engineering undergraduates similar to existing industrial MPC packages. MATLAB graphical user interface with MPC educational application was presented in (Yilmazlar and Kaplanoğlu, 2012). Our subject Automatic Control III aims to provide a MPC guidance to students and practice their theoretical knowledge. Students are getting not only new information but they are also practicing topics like modelling, identification, optimization, simulation, data acquisition and programming. Final year master students are learning MPC theory, they program controller functions in MATLAB, simulate control experiments first and at the end of the semester they apply their controllers to different laboratory systems. We are using GUNT level control and speed control training system because the systems are not too fast but fast enough, they are first and second order systems with low nonlinearity and we

10.1016/j.ifacol.2016.07.149

are able to use them from different environments like MATLAB, Simulink, LabView etc. (Honc, et al., 2014).

The outline of the paper is as follows. Basics of MPC theory is discussed in section 2. Laboratory system is presented in section 3. Modelling and identification is described in section 4, Simulation and experimental results are shown in section 5. Conclusions are given in section 6.

2. MPC BASIC THEORY

We want that the students understand MPC concept so we are not using tools like MATLAB's Model Predictive Control Toolbox (MATHWORKS, 2016), (jMPC Toolbox, 2016), (MPT Toolbox, 2016) or similar products. We are explaining step-by step predictive concept and deriving all necessary equations – complete documentation can be downloaded from MPC derivation. After MPC history overview and introduction we start with SISO system cost function formulation,

$$J = \sum_{j=1}^{N_2} r_j \left(y(k+j) - w(k+j)\right)^2 + \sum_{j=1}^{N_u} q_j \Delta u(k+j-1)^2 \quad (1)$$

where y is the controlled variable, w is the set-point, Δu is the manipulated variable increment, r_j are the penalization parameters for control errors and q_j are the penalization parameters for control incremets, N_2 is the length of horizon for following the set-point and N_u is the length of horizon for control actions (after N_u control changes the control action is kept constant).

Cost function expressed in matrix representation is

$$J = (\mathbf{Y} - \mathbf{W})^T \mathbf{R} (\mathbf{Y} - \mathbf{W}) + \mathbf{U}^T \mathbf{Q} \mathbf{U} \quad (2)$$

The next step is how to get predictions based on state-space and transfer function process model. The state-space algorithm is easier for the students (Honc and Dušek, 2013b). State-space model in discrete-time form

$$\begin{aligned} \mathbf{x}(k+1) &= \mathbf{A}\,\mathbf{x}(k) + \mathbf{B}\,u(k) \\ y(k) &= \mathbf{C}\,\mathbf{x}(k) \end{aligned} \quad (3)$$

is converted to incremental form as described below,

$$\begin{aligned} \underbrace{\begin{bmatrix} \mathbf{x}(k+1) \\ u(k) \end{bmatrix}}_{\mathbf{x}_p(k+1)} &= \underbrace{\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{0}_{1\times nx} & 1 \end{bmatrix}}_{\mathbf{M}} \underbrace{\begin{bmatrix} \mathbf{x}(k) \\ u(k-1) \end{bmatrix}}_{\mathbf{x}_p(k)} + \underbrace{\begin{bmatrix} \mathbf{B} \\ 1 \end{bmatrix}}_{\mathbf{N}} \Delta u(k) \\ y(k) &= \underbrace{\begin{bmatrix} \mathbf{C} & 0 \end{bmatrix}}_{\mathbf{O}} \underbrace{\begin{bmatrix} \mathbf{x}(k) \\ u(k-1) \end{bmatrix}}_{\mathbf{x}_p(k)} \end{aligned} \quad (4)$$

and the predictions in matrix form can be written as

$$\begin{aligned} \underbrace{\begin{bmatrix} \mathbf{y}(k+1) \\ \mathbf{y}(k+2) \\ \mathbf{y}(k+3) \\ \vdots \\ \mathbf{y}(k+N_2) \end{bmatrix}}_{\mathbf{Y}} &= \underbrace{\begin{bmatrix} \mathbf{ON} & 0 & \cdots & 0 \\ \mathbf{OMN} & \mathbf{ON} & \cdots & 0 \\ \mathbf{OM^2N} & \mathbf{OMN} & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ \mathbf{OM}^{N_2-1}\mathbf{N} & \mathbf{OM}^{N_2-2}\mathbf{N} & \cdots & \mathbf{OM}^{N_2-N_u}\mathbf{N} \end{bmatrix}}_{G} \cdot \\ \underbrace{\begin{bmatrix} \Delta\mathbf{u}(k) \\ \Delta\mathbf{u}(k+1) \\ \Delta\mathbf{u}(k+2) \\ \vdots \\ \Delta\mathbf{u}(k+N_u-1) \end{bmatrix}}_{\mathbf{U}} &+ \underbrace{\begin{bmatrix} \mathbf{OM} \\ \mathbf{OM^2} \\ \mathbf{OM^3} \\ \vdots \\ \mathbf{OM}^{N_2} \end{bmatrix}}_{\mathbf{F}_p} \mathbf{x}_p(k) \\ &\underbrace{\qquad\qquad}_{\mathbf{f}} \end{aligned} \quad (5)$$

In the past we were using different methods for transfer function model like Diophantine equations and state-space transfer function equivalent. Following derivation seems to be easiest for the students understanding. We are considering process and disturbance model as,

$$y(k) = \frac{B}{A} u(k) + \frac{C}{\Delta A} e(k) \quad (6)$$

where, $e(k) = y(k) - \hat{y}(k)$ is prediction error, and polynomials

$$\begin{aligned} A &= 1 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_n z^{-n}, \\ B &= b_1 z^{-1} + b_2 z^{-2} + \ldots + b_n z^{-n} \\ C &= 1 + c_1 z^{-1} + c_2 z^{-2} + \ldots + c_{nc} z^{-nc}. \end{aligned}$$

We introduce polynomial,

$$\tilde{A} = \Delta A = 1 + \tilde{a}_1 z^{-1} + \tilde{a}_2 z^{-2} + \ldots + \tilde{a}_n z^{-n} + \tilde{a}_{n+1} z^{-(n+1)}$$

From (6), we can derive

$$\begin{aligned} \underbrace{\begin{bmatrix} 1 & 0 & \cdots & 0 \\ \tilde{a}_1 & 1 & \cdots & 0 \\ & & \vdots & \\ 0 & 0 & \cdots & 1 \end{bmatrix}}_{\mathbf{A}_p} \begin{bmatrix} \hat{y}(k+1) \\ \hat{y}(k+2) \\ \vdots \\ \hat{y}(k+N) \end{bmatrix} &= \underbrace{\begin{bmatrix} b_1 & 0 & \cdots & 0 \\ b_2 & b_1 & \cdots & 0 \\ & & \vdots & \\ 0 & 0 & \cdots & b_1 \end{bmatrix}}_{\mathbf{B}_p} \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \vdots \\ \Delta u(k+N-1) \end{bmatrix} + \\ + \underbrace{\begin{bmatrix} -\tilde{a}_1 & -\tilde{a}_2 & \cdots & -\tilde{a}_{n+1} \\ -\tilde{a}_2 & -\tilde{a}_3 & \cdots & 0 \\ & & & \\ 0 & 0 & \cdots & 0 \end{bmatrix}}_{\mathbf{A}_m} \begin{bmatrix} y(k) \\ y(k-1) \\ \vdots \\ y(k-n) \end{bmatrix} &+ \underbrace{\begin{bmatrix} b_2 & b_3 & \cdots & b_n \\ b_3 & b_4 & \cdots & 0 \\ & & \vdots & \\ 0 & 0 & \cdots & 0 \end{bmatrix}}_{\mathbf{B}_m} \begin{bmatrix} \Delta u(k-1) \\ \Delta u(k-2) \\ \vdots \\ \Delta u(k-n+1) \end{bmatrix} + \\ + \underbrace{\begin{bmatrix} c_1 & c_2 & \cdots & c_{n_c} \\ c_2 & c_3 & \cdots & 0 \\ & & \vdots & \\ 0 & 0 & \cdots & 0 \end{bmatrix}}_{\mathbf{C}_m} \begin{bmatrix} e(k) \\ e(k-1) \\ \vdots \\ e(k-n_c+1) \end{bmatrix} & \end{aligned} \quad (7)$$

This can be expressed in terms of predictions as,

$$\underbrace{\begin{bmatrix}\hat{y}(k+1)\\ \hat{y}(k+2)\\ \vdots\\ \hat{y}(k+N)\end{bmatrix}}_{\mathbf{Y}} = \underbrace{\mathbf{A}_p^{-1}\mathbf{B}_p}_{\mathbf{G}}\underbrace{\begin{bmatrix}\Delta u(k)\\ \Delta u(k+1)\\ \vdots\\ \Delta u(k+N-1)\end{bmatrix}}_{\mathbf{U}} + \underbrace{\mathbf{A}_p^{-1}\mathbf{A}_m}_{\mathbf{F}_y}\underbrace{\begin{bmatrix}y(k)\\ y(k-1)\\ \vdots\\ y(k-n)\end{bmatrix}}_{\mathbf{Y}_m} +$$
$$+\underbrace{\mathbf{A}_p^{-1}\mathbf{B}_m}_{\mathbf{F}_u}\underbrace{\begin{bmatrix}\Delta u(k-1)\\ \Delta u(k-2)\\ \vdots\\ \Delta u(k-n+1)\end{bmatrix}}_{\mathbf{U}_m} + \underbrace{\mathbf{A}_p^{-1}\mathbf{C}_m}_{\mathbf{F}_e}\underbrace{\begin{bmatrix}e(k)\\ e(k-1)\\ \vdots\\ e(k-n_c+1)\end{bmatrix}}_{\mathbf{E}_m} \quad (8)$$

On substituting predictor equation into cost function (2), we get,

$$J = \mathbf{U}^T\underbrace{\left(\mathbf{G}^T\mathbf{R}\mathbf{G}+\mathbf{Q}\right)}_{\mathbf{H}}\mathbf{U} + \mathbf{U}^T\underbrace{\mathbf{G}^T\mathbf{R}(\mathbf{f}-\mathbf{W})}_{\mathbf{g}} + \underbrace{(\mathbf{f}-\mathbf{W})^T\mathbf{R}\mathbf{G}}_{\mathbf{g}^T}\mathbf{U} +$$
$$+\underbrace{(\mathbf{f}-\mathbf{W})^T\mathbf{R}(\mathbf{f}-\mathbf{W})}_{k} \quad (9)$$

If matrix $\mathbf{R}$ is symmetric we get $J = \mathbf{U}^T\mathbf{H}\mathbf{U} + 2\mathbf{g}^T\mathbf{U} + k$ and we can solve the problem by quadratic programming.

For unconstrained case explicit solution exists (if matrix $\mathbf{H}$ is positive definite)

$$\mathbf{U} = -\mathbf{H}^{-1}\mathbf{g} = \left(\mathbf{G}^T\mathbf{R}\mathbf{G}+\mathbf{Q}\right)^{-1}\mathbf{G}^T\mathbf{R}\left(\mathbf{W}-\mathbf{F}_p\mathbf{x}_p(k)\right) = \mathbf{L}\left(\mathbf{W}-\mathbf{F}_p\mathbf{x}_p(k)\right) \quad (10)$$

or for actual control action $\Delta u(k) = \mathbf{K}\left(\mathbf{W}-\mathbf{F}_p\mathbf{x}_p(k)\right)$ where $\mathbf{K}$ is the first row of matrix $\mathbf{L}$ and the control law can be seen as shown in Fig. 1.

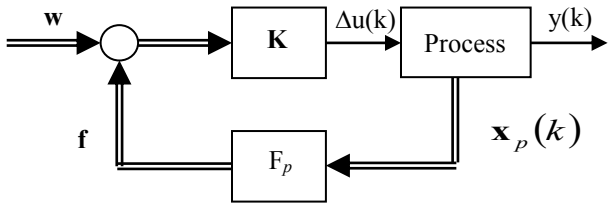

Fig. 1. Control law scheme.

We are also explaining constraints handling – especially input, output and state constraints

$$\left.\begin{aligned} u_{\min} \le u(i) \le u_{\max}, &\quad i \in \{k, k+N_u-1\}\\ y_{\min} \le y(i) \le y_{\max}, &\quad i \in \{k+1, k+N_2\}\\ \mathbf{x}_{\min} \le \mathbf{x}_p(i) \le \mathbf{x}_{\max}, &\quad i \in \{k+1, k+N_2\}\end{aligned}\right\} \quad (11)$$

which can be written in the form $\mathbf{A}\mathbf{U} \le \mathbf{b}$.

We are also mentioning the possibility how to extend controller design procedure to nonlinear MPC by free response calculation from nonlinear dynamical process model and by considering linear time varying model for forced response calculation.

2.1 MPC MATLAB scripts

Students with help of the teacher create own MATLAB scripts for both versions of predictive controllers. In the initialization part process model, horizons, penalization coefficients, sampling time, limits on manipulated and controlled variable and future set-point are specified. Poles for the state observer and filtering coefficient polynomial are also parameters for state-space or transfer function version of the controller. The rest of the script runs automatically and is general for arbitrary LTI SISO system. Predictor matrices are calculated before the main loop. In this loop process output is calculated or measured in case of real control. User can choose an unconstrained case with analytic solution or quadratic programming respecting given constraints. In case of the real control, plot is updated every sample time and manipulated variable is applied to the process.

2.2 MPC simulations

Student are testing and debugging their algorithms during the programming phase. If both algorithms are ready, students should get identical control responses for the same parameters. We are trying different models and parameters to demonstrate typical MPC behaviour and explaining the influence of the tuning parameters like penalization coefficients and length of horizons on the control response. The next step is that we introduce model mismatch, disturbance and measurement noise to simulate the phenomenon which will be in real application. This will cause different behaviour of state-space and the input-output form of MPC and questions like how to choose observer poles or filtering polynomial can be answered. Also, steady-state control error can be studied and explained why the transfer function formulation has integrating character while state-space not. The MATAB scripts for both versions can be downloaded from State-space, Transfer function. The next step will be laboratory application of both controllers.

3. LABORATORY PROCESSES

For MPC laboratory application we like to use two models from the company GUNT – level and speed control (Gunt 2016a and b). The systems are supplied with made to measure software for open loop and PID control experiments. We found out that the producer is using standard LabJack U12 data acquisition card (LabJack 2016). These cards have great software support e.g. for Dev-C++, LabVIEW, Python, MATLAB, Perl, .NET(C#,VB), Delphi, VBA Excel, Java, Visual Basic 6, PowerBASIC, Agilent VEE, TestPoint, Visual C++ OCX, Visual C++ DLL and LabWindows/CVI for Windows, Python for Mac and Python and Perl for Linux. We wrote also own DLL library and S-function for Simulink

(Honc and Dušek, 2013) and it is possible to call it from MATLAB script. This is more convenient for advance control methods like MPC than using MATLAB/Simulink.

3.1 GUNT RT 010 level control system

GUNT RT 010 is a TISO first order system for controlling the water level (output) using pump (input 1) and valve (input 2). Experimental set-up (see Fig. 2) is mounted on the housing with electronics 3. Transparent level-controlled tank 1 is fed from the storage tank 4 with the speed-controlled pump 2. Liquid level is measured using the tensometric pressure sensor. Electromagnetic proportional valve 5 in the tank outlet can serve as a disturbance variable or as a second manipulated variable.

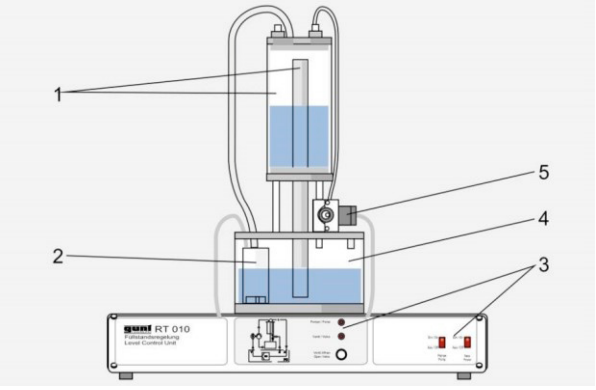

Fig. 2. Scheme of training system RT 010.

3.2 GUNT RT 050 speed control system

GUNT RT 050 is a SISO second order model for speed control (output) using motor power (input). Experimental set-up (see Fig. 3), is mounted on the housing with electronics 2. DC motor 6 drives a shaft 4 with a mass flywheel 5. The dial gauge 1 allows reading off the speed at any time. The speed is measured inductively using a speed sensor. A generator 3 acting as a mechanical resistance to a shaft rotation can be activated to study the influence of the disturbance variables. Generator load can be used in discontinuous manner as 0, 33, 66 and 100 %.

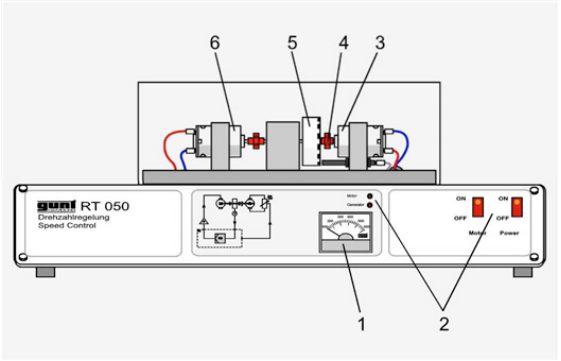

Fig. 3. Scheme of training system RT 050

4. MODELING AND IDENTIFICATION

To be able to apply MPC algorithms to GUNT models, student must identify models of both systems. It is possible to use first principle approach to get a physical model of water level system (Honc et al. 2014) but for simplicity students measure step response and find approximation transfer function by numerical identification.

The parameters of the chosen model are estimated by minimizing the squared error between the model output and measured output using least square method, after applying step inputs. Fig. 4 and Fig. 5 shows the model validation of the laboratory setups, where dotted line represents measured output and bold line is model response.

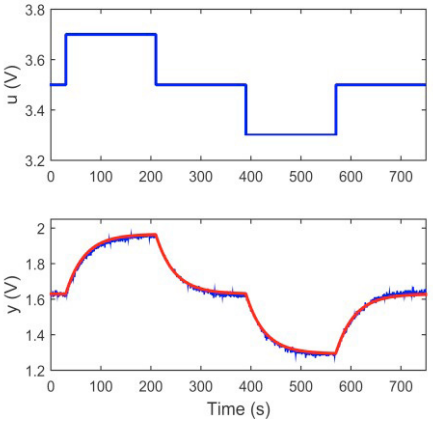

Fig. 4. Measured output and model response of GUNT RT 010 system

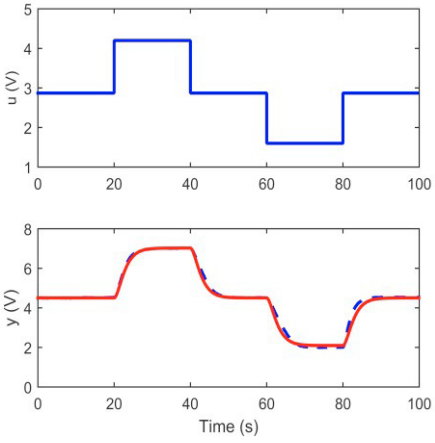

Fig. 5. Measured output and model response of GUNT RT 050 system

The transfer function model, *G(s)*, of the GUNT RT 010 and 050 system after system identification procedure is obtained as,

$$G(s) = \frac{1.7}{37s + 1} \tag{12}$$

$$G(s) = \frac{1.87}{(1.47s+1)(1.4s+1)} \tag{13}$$

5. SIMULATION AND EXPERIMENTAL RESULTS

Student simulate the MPC control response first – they will find suitable parameters like horizons and penalization for given future set-point shape (usually steps around working point). They measure real control experiment thereafter – usually they are retuning observer poles or filtering coefficient.

Both MPC approaches are applied to GUNT RT 010 and RT 050 training systems – predictive controller with predictor based on transfer function model (8) and another one based on state-space process model (5). State is estimated by Kalman filter. Constraints (11) were considered and we assumed that the future course of the set-point is known. Active set algorithm were used in Quadratic Programming to minimize the cost function.

GUNT RT 010 model is considered as first order SISO system (12) for simplicity instead of TISO system - the valve is kept open during the experiment. Hence, the level control is done by adjusting the pump power. Linearization points of pump and pressure sensor voltage are chosen as, u_0 = 3.42 V and y_0 = 1.61 V respectively. Control experiment starts from the calculated steady-state input and output of model. Multiple positive and negative set-point steps are applied at times t=50 s, 200 s and t=100 s, 150 s respectively. Response of the controller with the transfer function is in Fig. 6 and with the state-space model is in Fig. 7. Simulation response is plotted with the solid line and the dotted line shows the experimental data.

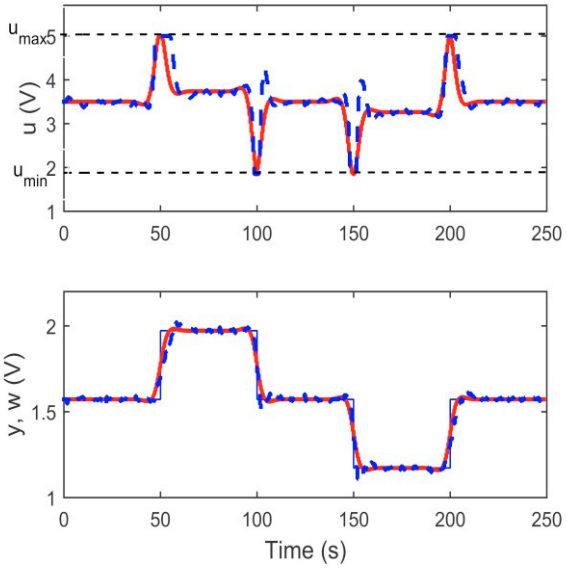

Fig. 6. GUNT RT 010 control response of controller with transfer function model

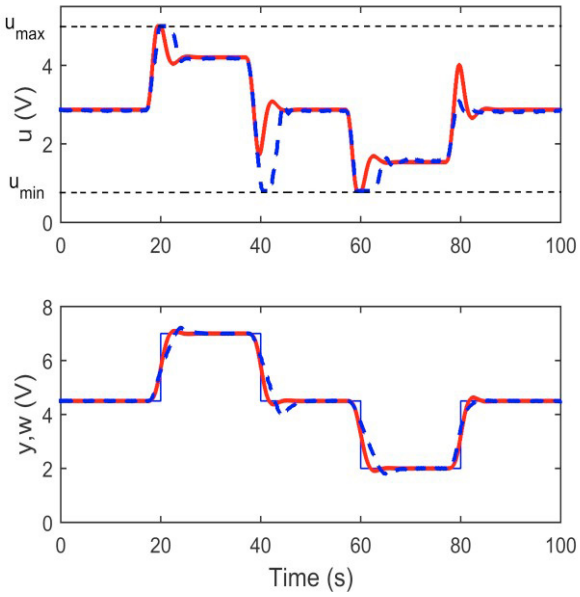

Fig. 7. GUNT RT 010 control response of controller with state space model

A second order model (13) is considered for GUNT RT 050 model. Linearization points of motor and speed sensor voltage is u_0 = 2.85 V, y_0 = 4.5 V respectively. Multiple positive and negative set-point steps are applied at times t=20 s, 80 s and t=40 s, 60 s.

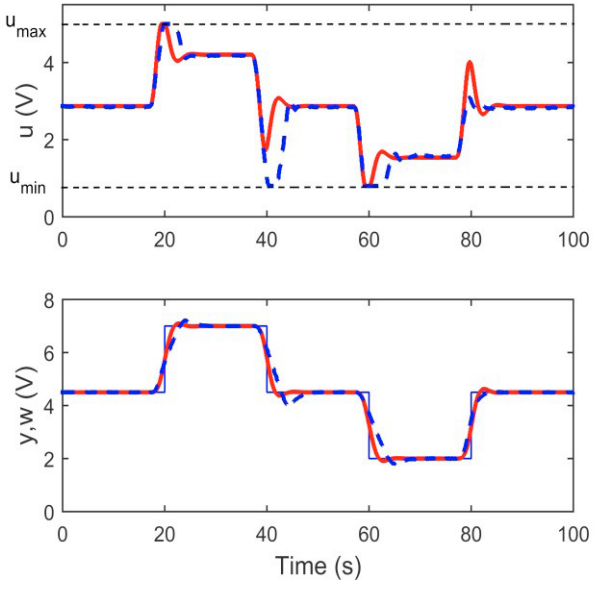

Fig. 8. GUNT RT 050 control response of controller with transfer function model

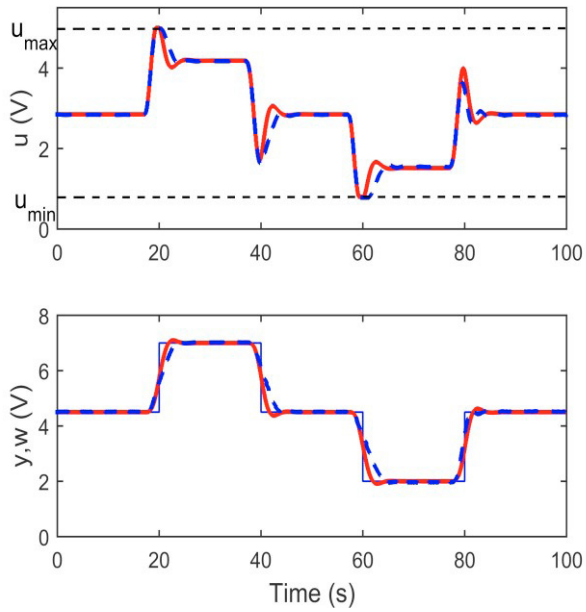

Fig. 9. GUNT RT 050 control response of controller with state space model

6. CONCLUSION

The paper describes the teaching experience with MPC at University of Pardubice and Anna University. Students learn the concepts, derive and code the state-space and the transfer function version of the MPC controller. Students simulate ideal control algorithms, add model mismatch, disturbance and measurement noise. Further, the designed controllers are applied to laboratory systems during the last stage of the course. Students' knowledge is evaluated continuously during the course. Students complete particular tasks and compare results with teachers' version. The course is concluded with an oral examination. The student appreciate that they are able to apply quite complex control method to real processes and "it works".

ACKNOWLEDGEMENT

This research was supported by project SGS_2016_021 at Faculty of Electrical Engineering and Informatics, University of Pardubice. This support is very gratefully acknowledged.

REFERENCES

Bemporad, A. (2009). Model Predictive Control: Basic Concepts. http://www.seas.upenn.edu/~ese680/papers/IntroductionMPC.pdf

Borrelli, F., Bemporad, A., Morari, M. (2015). Predictive Control for linear and hybrid systems, http://www.mpc.berkeley.edu/mpc-course-material

Boyd, S. (2008). Model Predictive Control. http://stanford.edu/class/ee364b/lectures/mpc_slides.pdf

Camacho, E.F., Bordons, C. (2007). *Model Predictive control*, Advanced Textbooks in Control and Signal Processing, Springer.

Canale, M. and Casale-Brunet, S. (2014). A multidisciplinary approach for Model Predictive Control Education: A Lego Mindstorms NXT-based Framework, Regular Papers Control Applications, *International Journal of Control, Automation and Systems, 12*(5), 1030-1039.

Honc, D. and Dušek, F.Daniel Honc and František Dušek. (2013a). MATLAB/Simulink support for GUNT control units, *In International Conference on Process Control. IEEE*, 534-539.

Honc, D. and Dušek, F. (2013b). State-Space Constrained Model Predictive Control. *In ECMS*, 441-445.

Honc, D. and Dušek, F., Sharma, R. (2014). GUNT RT 010 Experimental Unit Modelling and Predictive Control Application, *Nostradamus 2014: Prediction, Modeling and Analysis of Complex Systems*. Springer International Publishing, 175-184.

Doležel, P.; Mariška, M.; Taufer, I.; Havlíček, L. (2013). Artificial Neural Network Promotion: How to Introduce Artificial Neural Networks to Students. *In International Conference on Process Control. IEEE*. 214 – 218.

GUNT RT 010 Training system: Level control. (2016a). http://www.gunt.de/static/s3411_1.php

GUNT RT 050 Training system: Speed control. (2016b). http://www.gunt.de/static/s3405_1.php

Jay H. L. (2005). A Lecture on Model Predictive Control, http://cepac.cheme.cmu.edu/pasilectures/lee/LecturenoteonMPC-JHL.pdf

jMPC. (2016). http://www.i2c2.aut.ac.nz/Resources/Software/jMPCToolbox.html

Kouvaritakis, B. and Cannon, M. (2015). *Model Predictive Control, Classical, Robust and Stochastic*, Advanced Textbooks in Control and Signal Processing, Springer.

LabJack. (2016). https://labjack.com/products/u12

Maciejowski, J. M. (2002). *Predictive Control with Constraints*. Pearson education.

Mareš J. and Hrnčiřík P. (2012). *Základy prediktivního řízení,* VŠCHT Praha, 78 (in Czech).

MATHWORKS MPC toolbox. (2016). http://www.mathworks.com/products/mpc/

Mikleš, J. and Fikar, M. (2004). *Modelovanie, identifikácia a riadenie procesov 2*. Identifikácia a optimálne riadenie, 260 (in Slovak).

Multi-Parametric Toolbox. (2016). http://people.ee.ethz.ch/~mpt/3

Pekař, J. (2010). Short Introduction to Model based Predictive Control, https://moodle.dce.fel.cvut.cz/pluginfile.php/890/mod_resource/content/1/slides/MPC_introduction_FEL.pdf

Rawlings, J.B. and Mayne D.Q. (2009). *Model Predictive Control: Theory and Design*, Nob Hill Publishing, LLC.

Richmond, P. and Chen, D. (2012). A model predictive control package for undergraduate education, *Education for Chemical Engineers, 7*(2), e43–e50.

Rossiter, J.A. (2003). *Model-Based Predictive Control: A Practical Approach*, CRC Press, 344.

Rossiter, J.A. (2014). Videos on model predictive control, https://www.sheffield.ac.uk/acse/staff/jar/mpcmaster

Shakouri, P., Ordys, A., Collier, G. (2013). Teaching Model Predictive Control Algorithm Using Starter Kit Robot, *Engineering Education, 8*(2), 30-43.

Wang, L. (2009). *Model Predictive Control System Design and Implementation Using MATLAB®*, Advances in Industrial Control, Springer.

Yilmazlar, E. and Kaplanoğlu, E. (2012). Design of the Model Predictive Control Education and Application Interface, *Journal of Software Engineering and Applications, 5*(9), 677-681.