SIMULATION TOOLS IN CONTROL ENGINEERING EDUCATION

Jenő Kovács and Imre Benyó
University of Oulu
POB 4300, Linnanmaa, 90014 Oulun yliopisto, Finland
Jeno.Kovacs@oulu.fi, Benyo@rit.bme.hu

György Lipovszki
Budapest University of Technology and Economics
Goldmann Gy. tér 3, V2 ép. 1111 Budapest, Hungary
Lipovszki@rit.bme.hu

KEYWORDS
Simulation, control, mathematical modelling, computer-aided education.

ABSTRACT

The paper introduces simulation package developed for facilitating the education of discrete-time control theory at undergraduate level. The aim was not only to provide an interactive demonstration tool, but also to provide a better understanding of the applied algorithms. That aim was fulfilled by the choice of the LabVIEW programming environment, which alloys the simplicity of graphical programming with the traceability of the analogue devices.
The simulation package provide basic blocks for identification and control: discrete-time filter, discrete-time state space models, identification blocks for output error method and equation error method, recursive least square estimation, Kalman-filter, state observer and general predictive control. Furthermore, the package contains a training purpose simulator demonstrating the so-called RST control structure. Beside demonstration, the simulator provides a good basis for control design.
Beyond classroom demonstration, the simulator can be an effective tool to intensify self-study and distance education. Several examples are shown to demonstrate the features of the new tool.

INTRODUCTION

During the last decade, the development of educational and industrial software and simulation tools has been accelerated. Industrial applications focus on the replacement of expensive equipment by software tools (virtual equipment) and parallel the new technologies, e.g. Fieldbus, are strongly supported by high-tech software solutions. Also, numerous flexible software solutions for industrial human machine interface (HMI) and supervisory control and data acquisition (SCADA) have appeared in the market. The university education increasingly integrates such industry-standard programming-environment tools mainly in laboratory processes but more and more frequently also in the research and the classroom education. In education, the demonstration is the most common utilisation. Considering engineering education, demonstration involves process modelling and simulation, imitated data acquisition and process control. It requires high-level graphical user interface providing efficient communication.
One of the most widespread industrial software used in education is the LabVIEW, a National Instrument product (National Instrument, 2003). The LabVIEW is a block-oriented graphical programming environment developed at first place for data acquisition and monitoring, but process control and modelling are also fully supported. Due to the additional toolboxes, the application area is continuously expanding.

Control engineering toolbox, TUBSIM, for mainly continuous-time modelling and control has been earlier developed in (Lipovszki and Aradi, 1995). The TUBSIM is a system simulation extension of LabVIEW interpreting an analogue computer in graphical programming environment. Besides the typical analogue computers elements, the TUBSIM Library contains different Boolean blocks, typical system engineering elements (low order transfer function elements, continuous time controllers like PI, PID, time delay) and some discrete time blocks. The TUBSIM Library is successfully used in the control engineering education at the Budapest University of Technology and Economics (Aradi, 1996).

Simulation-demonstration tools for discrete-time control engineering are presented in this paper. The tools are developed for the "Discrete-time control design" and the "Advanced control design" courses at the University of Oulu (Finland). A training purpose simulator supports the first course, providing a demonstration and control-design environment. The second is an advanced course, where one general-purpose simulator cannot cover all the topics. Therefore not complete simulators, but rather elementary blocks (*e.g.* state-space models, identification toolboxes) are developed and the users should build their own simulators. This way, the discussed algorithms can be better understood.

The paper first introduces the training purpose simulator – the RST simulator. Then the new blocks for advanced control engineering are described and the utilisation is demonstrated via a general predictive control example.

RST CONTROL DESIGN

Applying the input-output, polynomial approach for system description, the training purpose simulator is based on the general presentation of a discrete-time

Proceedings 16th European Simulation Symposium
György Lipovszki, István Molnár © SCS Press, 2004
ISBN 1-56555-286-5(book) / ISBN 1-84233-106-x(CD)

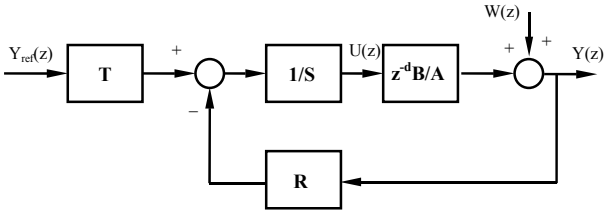

Figure 1: General two-degree-of-freedom control system.

controller: the two-degree-of-freedom controller, see in Figure 1. The process to be controlled is here defined by the pulse transfer function $z^{-d}\frac{B}{A}$. The controller is constructed from the three control polynomials, R, S and T; therefore it is called the RST control structure. The structure is very attractive, since most of the discrete-time controllers can be described by or transformed to the RST structure. The control design aims to define the R, S, and T polynomials to achieve certain required (dynamic and steady-state) performance for regulation and tracking, defined respectively by the pulse transfer functions

$$H_D\left(z^{-1}\right)=\frac{Y(z)}{W(z)}=\frac{A\left(z^{-1}\right)S\left(z^{-1}\right)}{A\left(z^{-1}\right)S\left(z^{-1}\right)+z^{-d}B\left(z^{-1}\right)R\left(z^{-1}\right)}$$

$$H_{CL}\left(z^{-1}\right)=\frac{Y(z)}{Y_{ref}(z)}=\frac{B\left(z^{-1}\right)T\left(z^{-1}\right)}{A\left(z^{-1}\right)S\left(z^{-1}\right)+z^{-d}B\left(z^{-1}\right)R\left(z^{-1}\right)}$$

The required regulation dynamics may be defined by the closed-loop characteristic polynomial, $P_D\left(z^{-1}\right)$. Solving the Diophantine equation

$$A\left(z^{-1}\right)\cdot S\left(z^{-1}\right)+z^{-d}B\left(z^{-1}\right)\cdot R\left(z^{-1}\right)=P_D\left(z^{-1}\right)$$

one can obtain the S and R polynomials. If required, additional n integrators can guarantee zero-steady state error in response to disturbance by replacing polynomial S by $(1\text{-}z^{-1})S$. Zero cancellation, which has a role in tracking, can be achieved by applying P_DB^+ on the right hand side; B^+ denotes the zeros to be cancelled as a factor of B, $B = B^+\cdot B^-$. Further tracking requirement can be satisfied by the proper choice of polynomial T. Guaranteeing zero-steady state error in response to reference signal:

$$T(1)=\frac{P_D(1)}{B^-(1)}.$$

Utilising the two-degree-of-freedom feature of the RST structure, the polynomial T may (partly) cancel the P_D dynamics, allowing the user to define different tracking dynamics via an external reference model, B_m/A_m as:

$$T\left(z^{-1}\right)=\frac{P_D\left(z^{-1}\right)}{B^-(1)}\ \text{ and }\ \frac{Y(z)}{Y_{ref}(z)}=\frac{B_m\left(z^{-1}\right)B^-\left(z^{-1}\right)}{A_m\left(z^{-1}\right)B^-(1)}.$$

Pole-placement with implicit reference model is another alternative in (Åström and Wittenmark, 1997). In that case, the R, S and T polynomials are designed to satisfy the requirement

$$H_{CL}\left(z^{-1}\right)=\frac{Y(z)}{Y_{ref}(z)}=z^{-d}\frac{B_m\left(z^{-1}\right)B^-\left(z^{-1}\right)}{A_m\left(z^{-1}\right)B^-(1)}.$$

RST simulator

The RST simulator was developed in LabVIEW graphical programming environment. The LabVIEW provides a block-oriented programming structure, well facilitated with additional toolboxes. The current simulator requires special continuous- and discrete-time blocks, which are available from the TUBSIM toolbox (Lipovszki and Aradi, 1995), while others are described in (Benyó *et al.* 2003).

The user interface of the simulator, shown in Figure 2, has three main areas: the *RST structure* in the upper right hand corner, the *graphical windows* below it, and the *parameter windows* on the left hand side.

The *RST structure* illustrates the control structure and provides several option for:

- choosing the type of process model (continuous- or discrete-time),
- setting the parameters of the reference signal, Y_{ref}, (amplitude, period, start time) and the disturbance signal, W, (period and amplitude of a deterministic stepwise signal, amplitude and mean value of a stochastic noise),
- applying (or not) integral action,
- choosing the type of T (polynomial or scalar),
- utilising (or not) a reference model, $\frac{B_m}{A_m}$.

The *graphical windows* plot the reference signal, the output signal and the sampled output (in case of continuous-time process) in the upper window, and the control input is shown in the lower one.

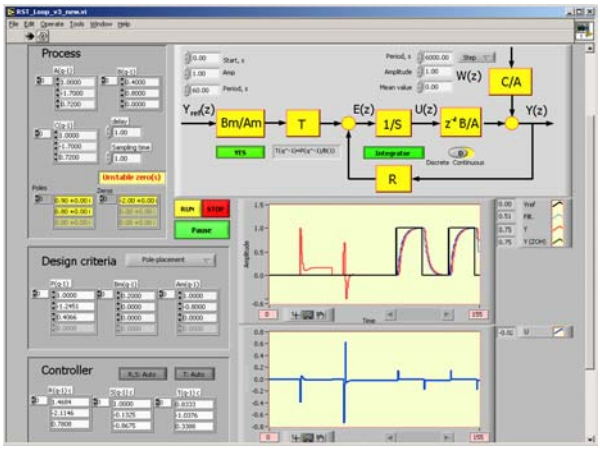

Figure 2: The user interface of the RST simulator.

The user can define the process, set the control design criteria and calculate or set manually the controller parameters in the *parameter windows*. The process parameters are the polynomials (A, B, and C) of a pulse transfer function (in discrete-time) with the time delay (d) or those of a transfer function (in continuous-time). the zeros and poles of the process are automatically calculated and warning is given when any of those is unstable. It has an importance, *e.g.*, when considering zero cancellation. After the process was defined, the control design shall be started. The user may choose from the list of offered controllers:

- pole placement controller,
- pole-zero placement controller,
- dead-beat controller, and
- minimum-variance controller.

The user sets the polynomial P_D for pole- and pole-zero placement controller (see under *Regulation criteria*), but for dead-beat and minimum variance controllers, the simulator automatically selects the P_D as $P_D(z^{-1}) = 1$ and $P_D(z^{-1}) = C(z^{-1})$, respectively.

The *tracking criteria* are determined by the choice on:

- polynomial T: if scalar then it effects only the steady-state error, if polynomial it allows to utilise the 2dof property,
- the existence of explicit reference model; if chosen, the B_m and A_m polynomials are set here.

It is important to emphasise that the choice of polynomial T and the reference model allows setting the tracking behaviour to be different from the regulation dynamics.
Based on the selected control methods, the *Controller window* automatically plots the resulted control polynomials. However, the user has the freedom to choose those manually to test any other controllers, which are not necessarily offered by the simulator. Especially, when continuous-time process is chosen, since it requires manual introduction of the control polynomials.

Several features results in a user-friend interface, such as displaying only the necessary elements; *e.g.* the polynomials of the reference model appear only when the user selects to apply one; or any modification in process parameters or design criteria is immediately updates the R, S and T polynomials. The simulation can be paused at any time for changing signal parameters.

Demonstration example

The RST simulator offers numerous possibilities for demonstrating or designing control structures. During an introductory course to control theory, simple example helps to understand the performance degradation caused by a stepwise output disturbance signal and how to eliminate it by applying an integrator in the forward loop. The difference between the regulation and tracking trajectories can be easily visualised by selecting the proper controller parameters in a fast and simple manner. Later, advanced students may design their own controllers and easily test those using the simulator. Two demonstration examples are here discussed: the design steps of a pole-placement controller and the illustration of intersampling ripple phenomenon.

Pole-placement controller
The design steps of the pole-placement controller can be summarised the following way: a) design of the output disturbance elimination based on the required closed-loop characteristic polynomial, b) introducing additional integration action to avoid non-zero steady-state error in response to output disturbance, c) ensuring zero-steady state error in reference signal following and finally d) selecting a tracking dynamics to emphasise the two-degree-of-freedom feature of the RST control structure.

Let the process to be controlled, using h=1 sec sampling time:

$$A(z^{-1}) = 1 - 1.7z^{-1} + 0.72z^{-2},$$
$$B(z^{-1}) = 0.4 + 0.8z^{-1}, d = 1.$$

The desired regulation dynamics is

$$P_D(z^{-1}) = 1 - 1.2451z^{-1} + 0.4066z^{-2}.$$

First, the simulation is run without integral action, therefore non-zero static error remains when a step output disturbance occurs. Applying the integration, the error can be eliminated. As the last step, the response to change in reference signal is tested with an explicit reference model. The responses in these three steps can be shown in the same graphical window, as seen in Figure 2, to support the full understanding.

Intersampling ripple
Intersampling ripple is a specific, discrete-time control phenomenon. Due to the sampling, information about the process performance can be obtained only at the sampling instants. If the continuous-time signal oscillates between the sampling instants, intersampling ripple occurs. Although this phenomenon has a remarkable importance in real-time application, it can be easily overlooked during control design when the user applies a discrete-time model for the process.

Avoiding this mistake, the RST simulator allows to define the process in continuous-time and to apply discrete-time controller. Consider the following double integrator process to be controlled, $G(s) = 1/s^2$. The control task is to achieve an open-loop behaviour defined by

$$H_{OL}(s) = \frac{0.2}{s(1+s)}.$$

The discrete-time transfer functions assuming ZOH and

sampling-time h = 1 sec are

$$G(z^{-1}) = 0.5\frac{z^{-1}+z^{-2}}{(1-z^{-1})^2}$$

$$H_{OL}(z^{-1}) = 0.074\frac{z^{-1}+0.718z^{-2}}{(1-z^{-1})(1-0.368z^{-1})}.$$

The simplest open-loop controller is

$$H(z^{-1}) = \frac{H_{OL}(z^{-1})}{G(z^{-1})} = 0.148\frac{(1+0.718z^{-1})(1-z^{-1})}{(1+z^{-1})(1-0.368z^{-1})}.$$

In RST structure it can be applied as:

$$T(z^{-1}) = 0.148(1+0.718z^{-1})(1-z^{-1})$$
$$S(z^{-1}) = (1+z^{-1})(1-0.368z^{-1})$$
$$R(z^{-1}) = 0.$$

The control inputs response to an impulse reference signal is

$$U(z) = H(z^{-1})Y_{ref}(z) = 0.148\frac{(1+0.718z^{-1})(1-z^{-1})}{(1+z^{-1})(1-0.368z^{-1})}1$$

The critically stable pole of the control input causes the oscillation in the control signal as therefore in the controlled output, as shown in Figure 3.

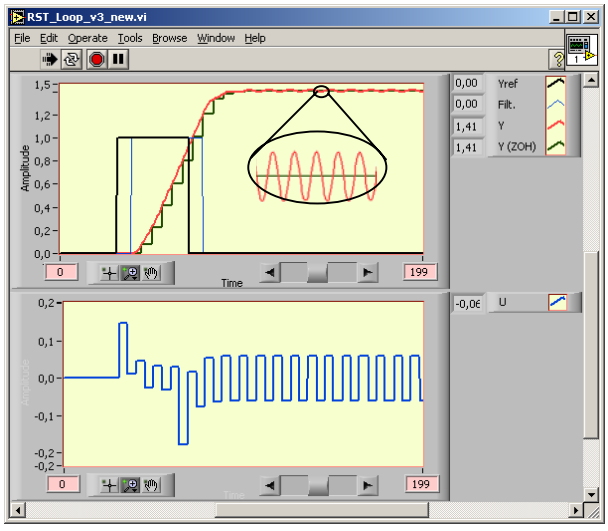

Figure 3**:** Intersampling ripple phenomenon. The upper graphic window also shows how the continuous-time signal oscillates while the sampled signal is constant.

ADVANCED CONTROL DESIGN

The aim of the development of the new blocks was to facilitate the simulation of the advanced identification and control techniques. In the present state of the development, the new block library includes tools for SISO discrete time systems. The algorithms of the implemented methods can be found in (Ikonen and Najim, 2002).

The new blocks can be divided into four subgroups according to their task: process modeling, identification tools, state observers, general predictive controller. A pop-up help window provides a brief description of each block.

Blocks for process modelling

The new set of blocks provides several possibilities for process modelling: the input-output approach and the state-space approach. The input-output approach defines the process by its pulse transfer function. The block "$\frac{B(q^{-1})}{A(q^{-1})}$" represents the pulse transfer function, defined by the coefficients of the B and A polynomials.

The state-space subgroup consists of three different blocks. The "sys SS" block defines the process model in state-space equation. The user shall define the state-space matrices.

In control and observer design, the controllable and observable canonical forms of the state-space equation are generally applied. If the process model is only available in the form of pulse transfer function (e.g. identification in input-output approach), then the necessary state-space equation can be obtained by the "SS contr" (State-Space controllable) and the "SS obs" (State-Space observable) blocks.

Blocks for identification

The identification subgroup contains three blocks: the recursive least square algorithm (RLS), the equation error (EE), and the output error method (OE).
The "RLS" block estimates the parameters of a given regression model using the recursive least square method. The input of the block is the regression vector and the output is the correlation vector. The covariance matrix of the parameter adaptation algorithm can be maintained by applying the offered methods: forgetting method, constant trace method or the bounded information algorithm. Since, the "RLS" block requires only the regression vector as an input, the user has the freedom to choose the model structure; consequently, the user has to create that vector.
In the "EE" and "OE" blocks, the structure of the regression model is a priori defined by the equation error method and the output error method, respectively. The parameter adaptation algorithm is in both cases the RLS algorithm. The block's inputs are the input and output signals of the process to be identified and the order of the model. The blocks return the estimated transfer functions given by the coefficients of the numerator and denominator polynomials, the output of the regression model, and the covariance matrix.

Blocks for State observer

Among the state observer blocks, one can choose between the Kalman-Filter ("$\hat{X}KF$") nd the "Fixed Gain State Observer ("$\hat{X}FGF$").

The implemented Kalman-Filter estimates the state variables at the $(k+1)^{th}$ sampling instant based on k^{th} process input and $(k+1)^{th}$ output measurement: $x(k+1) = f(u(k), y(k+1))$. The outputs of the block are the state vector and the trace of the covariance matrix.

The Fixed Gain State Observer estimates the state variables in the $(k+1)^{th}$ instant based on k^{th} process input and k^{th} output measurement. The output of the block is the $(k+1)^{th}$ state variables vector: $x(k+1) = f(u(k), y(k))$.

The state-observers utilise the observable canonical form of the state-space model. The required form can be generated by the "SS obs" (State-Space Observable) block, which transforms the transfer function into observable canonical state-space form.

Blocks for General Predictive Controller

There are two blocks relating to the GPC control algorithm. The "GPC M" calculates the gain matrices of the controller; the "GPC" block is the controller. The block realizes a GPC for SISO process. The controller is based on the state-space model of the controlled process.

The inputs of the block are the state variables, the controlled process output signal and the reference signal. The main output of the block is the control signal. (There is another signal for graphical purpose, the desired future process outputs on the prediction horizon.)

Besides the general parameters of the GPC algorithm (prediction, minimum and control horizons, weighting factors of the control error and control signal in the cost function; there is possibility to use weighting matrices for enabling weighting the terms in the appearance of time), the pulse transfer function of the desired closed-loop behavior can be defined.

In the case of online identification, it is possible to use the new identified model of the process in every time step, otherwise the time demanding computation of the gain matrices is performed only in the first time step (Adaptive button on/off).

Demonstration examples

The outlook and the utilisation of the developed blocks are demonstrated in the following two examples. The first example is an on-line identification combined with state observation, while the second example presents a GPC control structure.

Identification and state estimation

In this example, a simple parameter estimation and state estimation problem are solved. The "unknown" process is on-lined identified using the transfer function approach (output error method), and the identified model is transformed to state-space model that is used for the estimation of the state variable (Kalman-filter). The LabVIEW realisation, shown on Figure 4, clearly demonstrates the simplicity of programming.

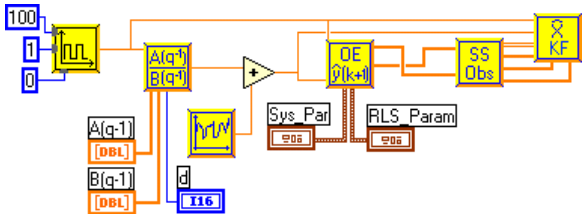

Figure 4: The program of the simulation without the time setting and graphical blocks.

The process is modelled by the output error method as

$$y(k) = \frac{B\left(q^{-1}\right)}{A\left(q^{-1}\right)}u(k) + e(k)$$

The pulse transfer function is changed at the 75^{th} sampling instants as

$$\frac{B\left(q^{-1}\right)}{A\left(q^{-1}\right)} = \frac{bq^{-1}}{1+aq^{-1}} = \frac{0.2q^{-1}}{1-0.8q^{-1}} \Rightarrow \frac{0.2q^{-1}}{1-0.6q^{-1}}$$

The e(k) noise is a Gaussian white noise with the variance 0.1. Within the RLS algorithm of the OE parameter estimation, the forgetting factor $\lambda = 0.98$ was applied to maintain the covariance matrix.

The state-space model is simple since the process is a first order,

$$x(k+1) = ax(k) + bu(k)$$
$$y(k) = x(k)$$

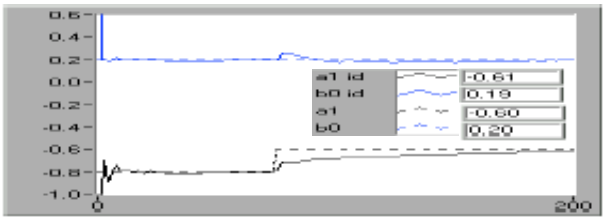

Figure 5a: LabVIEW graph: parameter estimates; true parameters (dashed lines), estimated ones (solidlines)

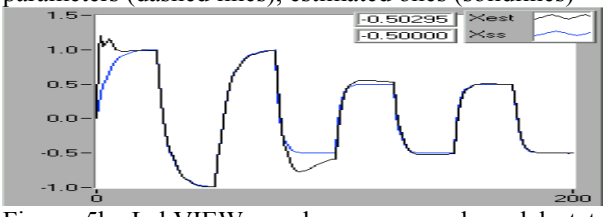

Figure 5b: LabVIEW graph: process and model state variables "Xss" and "Xest".

The initial values of the parameters were set $[a\ b]_{t=0} = [-1\ \ 0.6]$. Figure 5a demonstrates the parameter convergence while Figure 5b shows the process and model state variables, $x(k)$ and $\hat{x}(k)$, respectively.

General Predictive Control

This example demonstrates the application of the General Predictive Control block. The process output (y) is controlled by the GPC to follow the reference signal (r) with a prescribed tracking behaviour. The reference signal is a square wave signal, and the process has Gaussian white noise type measurement noise. The Fig. 8 shows the LabVIEW program of the simulation, with all of its accessories for graphical purpose. The Control Panel of the Simulation is shown on the Figure 6. On the control panel there are all of the numerical controls with which it is possible to change the parameters even in run-time.

Furthermore, all the process and control parameters are displayed on the control panel. The results of the simulation can be followed through the graphs and the numerical displays.

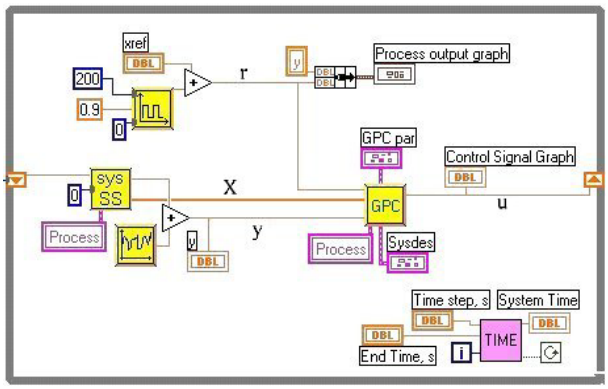

Figure 6: The program diagram of the simulation, with the same signal notation as on the previous figure.

CONCLUSION

Computer-aided education may be an advantageous solution for increasing the efficiency of control education. The classroom education, similarly to laboratory exercises, may be further visualised by introducing target-oriented simulation/demonstration environment.

A package of advanced simulation tools is developed to serve particular courses on digital control theory and advanced control engineering. The developed simulation elements provide an easy-to-use and easy-to-understand manner to deepen the enhanced knowledge of the course.

The presented tools support the simulation of discrete-time modelling, identification, state-observer and general predictive control design. The programming environment was chosen to be the LabVIEW due to its attractiveness.

The RST simulator facilitates the discrete-time control education, especially supporting the polynomial input-output approach for system description. The basic structure is the two-degree-of-freedom formulated by the control polynomials, R, S and T.

Further advantage of the LabVIEW based simulator is that it can be transformed into a self-executable file and run it in any Windows based operation system. It requires only a runtime engine, which is freely available from the web site of the National Instrument.

The proposed simulator has been evaluated in practice in a course "Digital control theory" at the University of Oulu and will be used in co-operation with the Budapest University of Technology and Economics.

REFERENCES

Benyó, I.; Lipovszki, Gy. And Kovács, J. 2003. "Advanced Control Simulation Tools In LabVIEW Environment", *Proc. 6th IFAC Symposium on Advances in Control Education*, Finland, 275-279.

Lipovszki, Gy. And Aradi, P. 1995. "General Purpose Block Oriented Simulation System Using LabVIEW", *NIWeek'95*, Austin, TX, USA.

National Instruments, 2002. *LabVIEW User Manual* at http://www.ni.com/pdf/manuals/

Åström, K.J. and Wittenmark, B. 1997. *Computer-Controlled Systems* (Prentice-Hall)

Aradi, P. 1996. "*Using LabVIEW in Education of Systems and Control Engineering",* NIWeek'96, Austin, TX, USA.

Ikonen, E. and Najim, K. 2001. *Advanced Process Identification And Control*, Dekker.

JENŐ KOVÁCS, born in Hungary in 1967, (M.Sc. 1991 Budapest, Hungary, Ph.D. 1998 Oulu, Finland) is a senior assistant at the Systems Engineering Laboratory, University of Oulu, Finland. His research interests include adaptive control, constrained control, advanced modelling and their application to energy systems and power plant control problems.

IMRE BENYÓ is born in Budapest, Hungary in 1975. He was graduated at the Technical University of Budapest, as mechanical engineer. He is researching at the System Engineering Laboratory, University of Oulu, Finland. His research area covers the predictive control, system identification problems, and its applications in the power plant control.

GYÖRGY LIPOVSZKI was born in Miskolc, Hungary and went to the Budapest University of Technology and Economics, where he studied electronics and graduated in 1975. He is now an Associate Professor at the Department of Department of Production Informatics Management and Control and his research field is development of different simulation frame systems.