Cent. Eur. J. Comp. Sci. • 2(3) • 2012 • 354-366
DOI: 10.2478/s13537-012-0025-5

Central European Journal of **Computer Science**

Process models using hardware in simulation loop

Research Article

Mikuláš Alexík*

Department of Technical Cybernetics, University of Žilina,
Univerzitna 8215/1, 010 26 Žilina, Slovak Republic

Received 10 February 2012; accepted 17 August 2012

Abstract: This paper describes two methods for realization of configurable time continuous linear and non-linear dynamical models. The first method is based on classical PC plus A/D and D/A converters and the second one is based on a microcontroller (Atmel ARM7). Both methods can be used for verification of algorithms for control and identification of processes, modeling of transport processes dynamics using simulation experiments, if simulations run in real time. The application of a microcontroller allows a simple realization of changing parameters and also non-linearity of the process. In the paper we also pay attention to the realization of real time in a programmable environment under Microsoft Windows and the realization of precise calculations in floating point on microcontrollers.

Keywords: microcontroller • real time • A/D and D/A converters • time delay

1. Introduction

Based on former experience from research and teaching [3–6] we can confirm that laboratory verification of all algorithms with processes dynamics using simulation experiments presents verifiable results provided the simulation experiments are executed in real time. Such a manner of simulation experiments execution means that inside the simulation loop runs a dynamical process in time independent from the simulation software. Then simulation software has to respect a change of dynamics and variables in the independent process, in other words, the simulation software (its speed) has to be adapted to the speed and precision of the real physical process. The most commonly used method is applying special laboratory equipments that are physically realized models of real systems, e.g. a helicopter model or inverse pendulum, which are used in a laboratory for teaching. These laboratory models together with A/D and D/A converters, which for us represent "hardware in the loop of simulation" are connected with a PC where programmed algorithms are controlled and there is software which enables us to realize verification using simulation experiments. This method is very attractive for the teaching process, but is not very applicable in research, because there is no way to change process parameters and process non-linearity, especially time (transport) delay. The application of a microcontroller as the base of the model allows a simple realization of changing parameters and non-linearity, and especially transport delays in the process [1]. In the paper special attention is paid to realization of real time in a programmable environment

* *E-mail: mikulas.alexik@fri.uniza.sk*

under Microsoft Windows and the realization of precise floating-point calculations on a microcontroller. The paper is organized as follows. Section 2 describes proper models of the system dynamics. Section 3 describes real time response models of the processes. Section 4 describes real time simulation experiments with a PC and hybrid simulation. The paper ends with conclusions and outlook in Section 5.

2. Models of the system dynamics

For describing the dynamics of "driver in the car", the author of this article verified the models described as (1) and (2) in [3] and [5]. Both models are transfer functions of first or second order, where parameters "a" represent relative damping on the system step response. The time constant T_1 is always bigger than zero. If the time constant T_2 in (1) is equal to zero, this mean that the driver behavior can be characterized by a model with first order dynamics. Parameter T_3, which is frequently equal to zero, is called "the driver prediction time constant" and it is the most changeable parameters in the dynamics of the driver. All parameters of this model usually change their values between two steady states or during a time response. This rapid changeability is caused by a mixture of relative permanent dynamic of nerves and muscles in the hands (legs) and dynamic prediction from models in the mind [5]. It is self-evident that "driver in the car" acts in real time, therefore the "measurement" or real data acquisition can only be done in real time. Therefore, models for verification of algorithms for measurement and parameters identification of dynamics behavior of the driver in the car have to be realized this way, so that the calculation of the output variable as response to the input variable was done in real time.
For verification of control algorithms, authors of [2] recommend to carry out verifications by the models described by transfer functions (3), (4). The transfer functions structure is changed from second order to eighth order and it is convenient, if the parameters can change their values. This is needed to verify the adaptive control algorithm. The authors of [2] are very respected authors in the area of "control systems". They declared that these transfer functions can cover large scale of dynamics of technological systems behavior.

$$S_1(s) = \frac{K(1+T_3s)e^{-d*s}}{(1+T_1s)(1+T_2s)};\ K = 0.8-1.2;\ T_1 = 0.05-0.9;\ T_2 = 0.0-0.9;\ T_3 = 0.0-0.9;\ d = 0.1-0.9; \tag{1}$$

$$S_2(s) = \frac{K(1+T_3s)e^{-d*s}}{(T_1s^2+2aT_1s+1)};\ K = 0.8-1.2;\ T_1 = 0.05-0.9;\ T_3 = 0.0-0.9;\ a = 0.2-0.99;\ d = 0.1-0.9; \tag{2}$$

$$S_3(s) = \frac{e^{-d*s}}{(1+Ts)^2};\ T = 0,1,...10;\ d = 0,1,...10; \qquad S_4(s) = \frac{1}{(1+s)^n};\ n = 3,4...8; \tag{3}$$

$$S_5(s) = \frac{1}{(1+s)(1+\alpha s)(1+\alpha^2 s)(1+\alpha^3 s)};\ \alpha = 2-7; \qquad S_6(s) = \frac{(1-\alpha s)}{(1+s)^3};\ \alpha = 0.1, 0.2, 0.5, 1, 2; \tag{4}$$

All of the models (1) to (4) described dynamics with type of non-linearity "d" named "transport delay" or "time delay". In the analytical mathematics this kind of models is represented by non-linear differentials equations. Non-linearity transport delay cannot be modeled (in real time) with analogue electronic environments, because we do not have an analogue electronic component, which can remember more different values of a variable for a period of time. It is common knowledge that digital components like a "register" or "memory" can be used for storing and collecting many values of arbitrary variables. Therefore, for this kind of models we have to use some realization based on a microcontroller or computer, where the models (1) to (4) are represented as difference equations. For these reasons, we are only interested in the kind of models which could realize the calculation of the analogue dynamics in real time. Two such environments are described in detail below.

3. Real time response models of the processes

Analogue transfer functions from (1) to (4) can be in real time realized with several modes:
a) Analogue model of the processes realized with analogue hardware.
This possibility was realized and verified by the author, more details about the realization can be found in [1]. The author has been using this mode for a long time to verify some kind of adaptive control algorithm [8, 9]. In this mode, continuous

change model variables cannot be realized, only the switching between models, with different dynamics. In this mode the time delay cannot be realized either, which is needed for the verification of control processes with time delay and generating the "driver in the car" dynamics. Therefore, the author tried to verify step by step other possibilities for real time realization of the required models. The experiences from the realization of the analogue mode were very important for examining the possibility to realize other modes of models.

b) Analogue model of the processes realized by FPAA - Field Programmable Analogue Array.
This mode was described by the author in [1]. There, one can also find the reason why this mode was not realized and verified (the number of analogue arrays realized in the one field compared to the whole number of arrays needed for the demanded model).

c) Hybrid model of the processes.
This mode can be realized on a PC with A/D and D/A converters. The digital computation in real time on the PC represents control/measured processes and through A/D and D/A convert as continuous input/output to the/from this processes. The control/identification algorithm is computed in real time on the same PC, and the A/D and D/A converters connected with variables to this PC, as the connection to the process. This mode is described in Subsection 3.2.

d) Hardware in loop (HIL) models of the processes.
Models of the controlled/measured processes are realised as "hardware" on the base of a microcontroller which has its own A/D and D/A converters. The input to the processes are analogue values, which are transformed to the digital values by A/D converters. The microcontroller can calculate difference equations, which represent the demanded dynamics of processes. Calculated values of output variables are transformed to the analogue values by D/A converters. These independent real time working "processes" are connected in the "simulation loop" with the PC, in which the controller/identifier algorithm is calculated. The model is described in Subsection 3.2.

3.1. Hybrid environment with computation on PC

This mode is a computation of output from the controlled dynamical system and computation of controller algorithms, both of which are computed on the same PC in real time. This mode also contains the realization of input/output for the controlled system and controller by A/D and D/A converters. This means that in comparison to the measurement mode, twice as much A/D and D/A converters are needed, because output/input to the controlled system has to be in analogue mode. The block scheme of this mode is depicted in Figure 1.

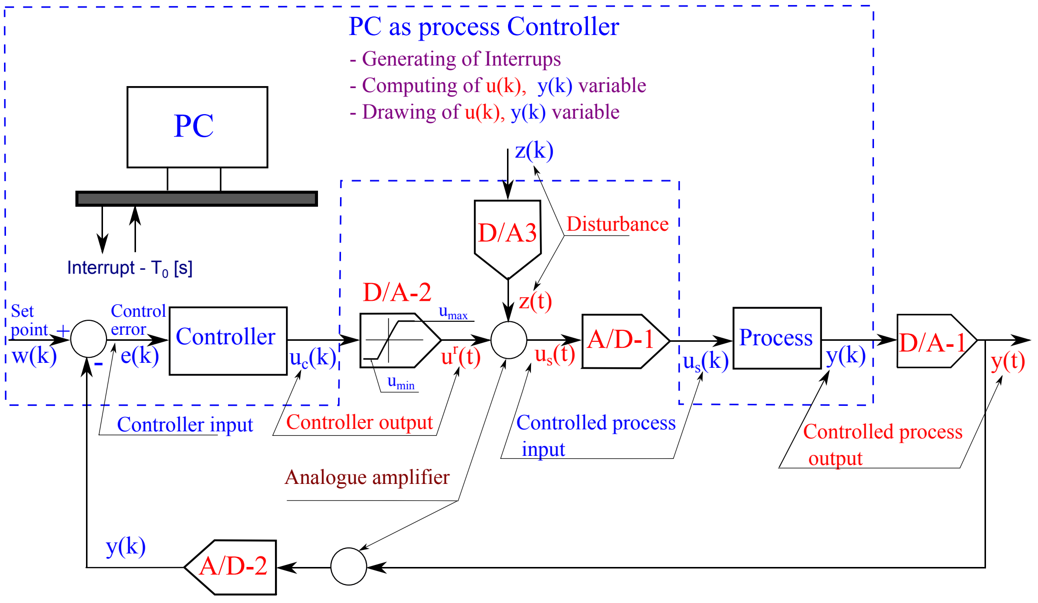

Figure 1. Real time hybrid simulation of control loop on the PC.

On the block scheme we can see the analogue amplifier with D/A as input and A/D as output. Applying D/A3 as "disturbance" input is more suitable for a mode where "disturbance" is an analogue signal (see Figure 3). It is simple to generate in analogue mode the disturbance signal "unit step", but is not possible to generate in this mode the pseudorandom signal, which can with no problem be generated from the PC as input to the D/A3 converter. For both modes, applying the analogue amplifier is necessary for summation of two analogue voltage signals before the A/D converter (summation of two outputs from D/A converters or D/A converter with potentiometer output summation). The block scheme can be used also without the second amplifier, which is placed before A/SD-2 converter. If the analogue amplifier is realized on a good level, then it can reduce the "quantization error" (Figure 2, error for 12-bit A/D) which is for a 12-bit D/A converter bigger than for a 14-bit A/D converter. The application of four models controlled systems and four summing amplifiers enables to use this hybrid model as controlled system with two inputs, two outputs and two disturbances, which is commonly called MIMO (Multiply Input, Multiply Output) controlled system.

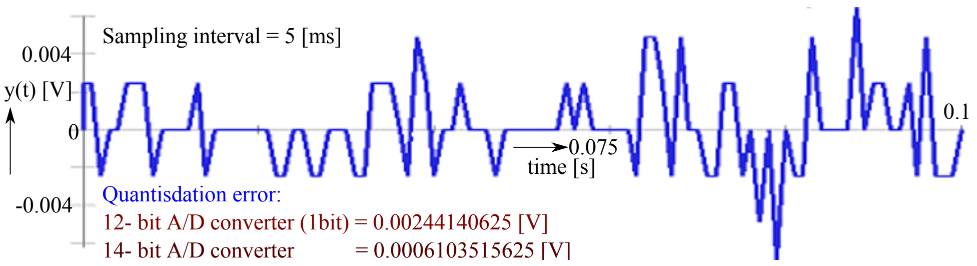

Figure 2. Quantization error on analogue amplifier output.

3.2. Hybrid models realized on microcontroller

When verifying the analogue model of the controlled process, working in real time, it is needed to apply A/D and D/A converters. Computations needed for the controller algorithm are realized as parallel in real time as well. In this "hardware in loop simulation" (HIL), while measuring is done with A/D converters, there is present an unpredictable signal noise, which is different in every simulation experiment. The same qualitative situation occurs in real process control. That is why the verifying quality in HIL control algorithm is higher than in digital simulation. The differences between model and real process, including quantization signal noise, influence the quality of the control process quite a lot. The influenced can also be negative, from delay caused by the duration of computing the control algorithm. Therefore it is useful for processes with delay (especially for control through internet environment) to have algorithms verified on analogue models in real time. The scheme with connecting blocks of a hybrid model is pictured in Figure 3. In Figure 4 there is a microcontroller part of the model and in Figure 5 there is the model's block scheme. This hybrid model can be used as 2x2 MIMO controlled process, or as two SISO (Single Input and Single Output) processes with auxiliary controlled variable or auxiliary control variable, or as four independent SISO controlled processes.
The connection of the microcontroller with the PC through RS 232 is used for sending the Z-transformation coefficient computed on the PC to the microcontroller. USB connection can be an alternative connection with the PC. There is also a connector for connecting the programmer device between microcontroller circuits and the PC. For verifying the hybrid model, simulation experiments with controlled systems and various controlled loops were carried out. Controlled processes dynamics were compared with batch processes against [2], detailed description is in [3, 5, 7]. Examples of simulation experiments with the hybrid model in control loop are commented in the next part of this paper. For the used microcontroller, the 32-bit arithmetic is available directly. The coefficients for the model of controlled process in the Z-transformation are computed on the PC with 64-bit floating point accuracy (relative - it depends on the sampling interval). Rewriting the coefficients into the microcontroller where ther is only 32-bit arithmetic causes many precision problems when calculating the difference equation for the controlled variable. This problem is explained in the next part of the paper. In the model we used a 32-bit microcontroller AT91SAM7A256 with 256KB flash memory. Therefore, we used also the operation system IAR PowerPac RTOS (RTOS - Real Time Operation System), which is placed directly in the microcontroller's memory and enable us to use computation with 64-bit floating point precision. But this way is not possible for older 16-bit microcontrollers, for example based on ATmega128, which we used for this model some years ago.

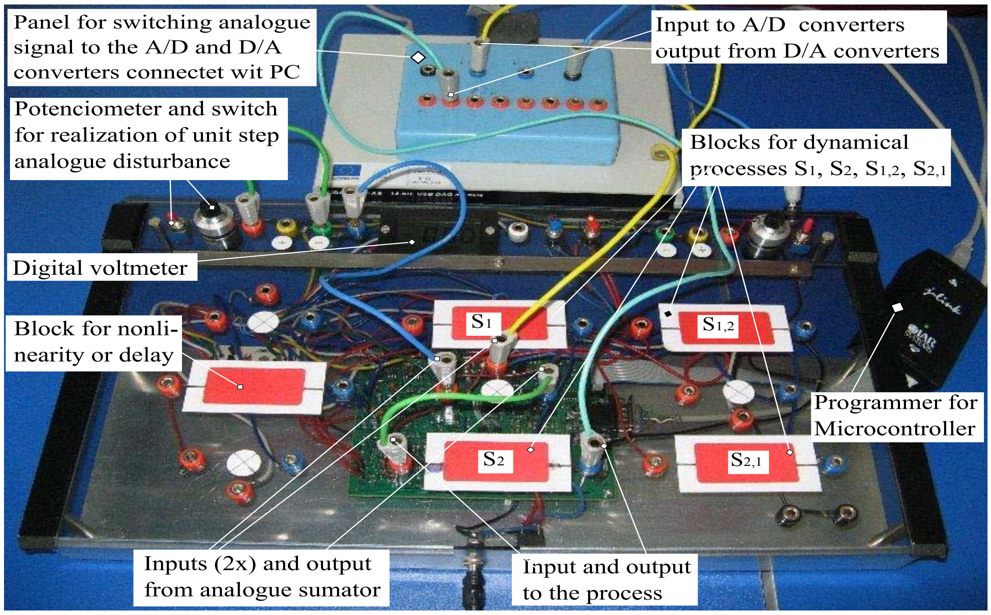
Panel for switching analogue signal to the A/D and D/A converters connectet wit PC
Input to A/D converters output from D/A converters
Potenciometer and switch for realization of unit step analogue disturbance
Blocks for dynamical processes S_1, S_2, $S_{1,2}$, $S_{2,1}$
Digital voltmeter
Block for nonlinearity or delay
S_1
$S_{1,2}$
Programmer for Microcontroller
S_2
$S_{2,1}$
Inputs (2x) and output from analogue sumator
Input and output to the process

Figure 3. Structure of Hybrid model based of microcontroller.

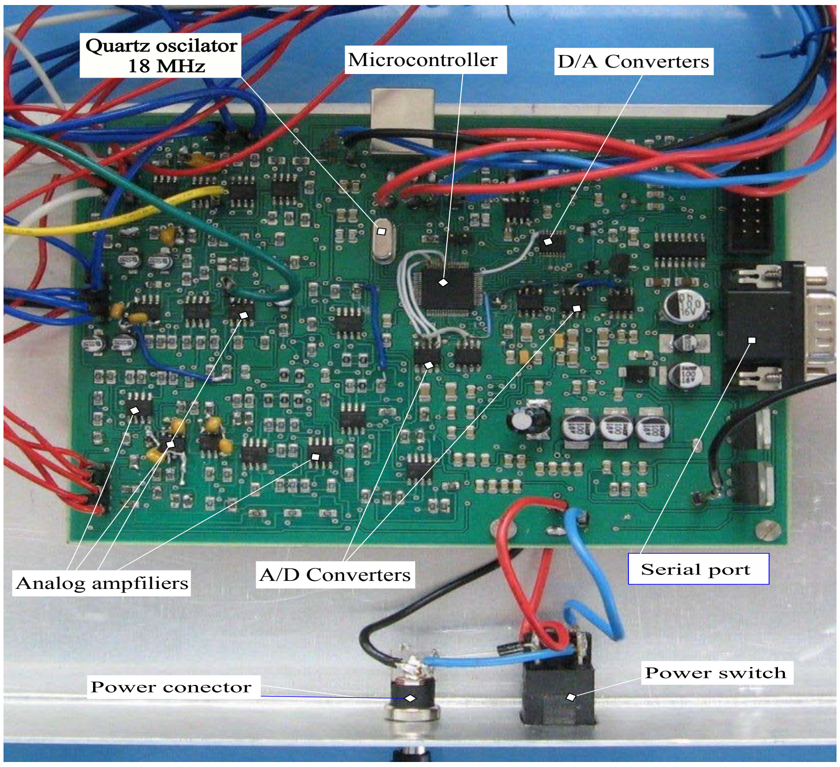

Figure 4. Microcontroller part of hybrid model.

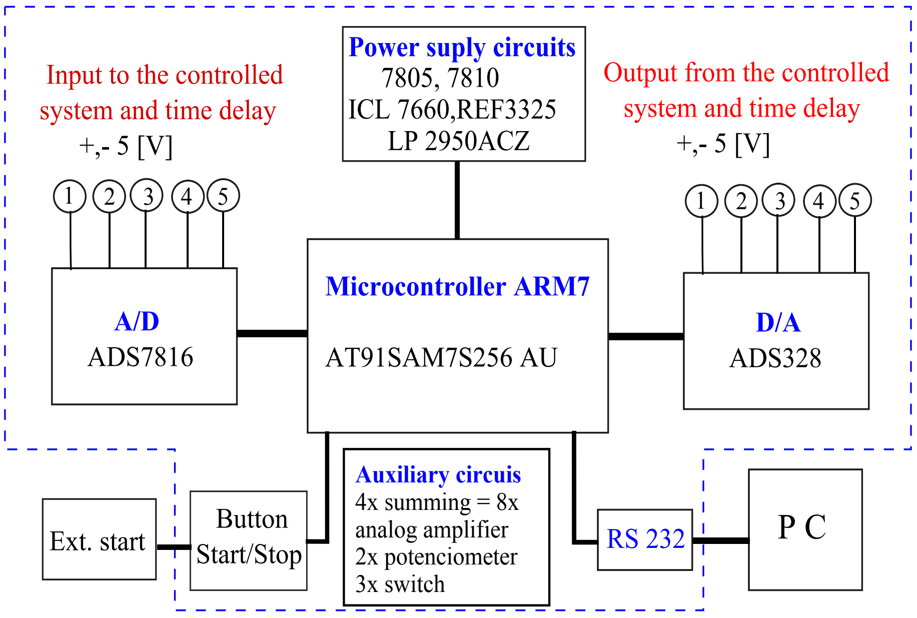

Figure 5. Hybrid model based on microcontroller block scheme.

3.3. Calculation of dynamic processes responses

Dynamic response of the used model is computed on the PC independently of the microcontroller, as output from difference equation (6), which is rewriting the Z-transform function (5). (In the sequel it is assumed that the delay $d = 0$, which has no influence on the results about the precision of calculating the Z-transform). This Z-transform function is computed on the PC for an exactly and uniquely determined sampling interval T_{0y}. The outgoing values from the model of dynamical process have to be identical for computation in the analogue mode (1) or discrete mode (5), for an identical input signal. As can be seen from (5) the situation in calculation from digital mode strictly depended on the precision of calculation of parameters a_i, b_i for the Z-transform function. And its precision depends on the used floating point and also, as can be seen in (5), on the used sampling interval T_{0y} . For clarity, we consider the situation in the steady state mode only. If the input signal is unit step, then for the dynamic process described by (1), steady state has to be gain K for analogue and digital mode, but also for calculation from the transform function (1) or (5), which is documented in (7) and (8).

$$S_1(s) = \frac{K(1+T_3 s)}{(1+T_1 s)(1+T_2 s)}; S_1(z) = \frac{b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} = \frac{Y(z)}{U(z)};\ a_1 = -(D_1 + D_2);\ a_2 = D_1 D_2;\ D_i = e^{\frac{-T_i}{T_{0y}}}$$

$$b_1 = K[(D_2+1)\frac{T_1 - T_3}{T_1 - T_2} + (D_1+1)\frac{T_2 - T_3}{T_1 - T_2} - (D_1 + D_2)]; \qquad b_2 = K(D_1 D_2 - D_2 \frac{T_1 - T_3}{T_1 - T_2} + D_1 \frac{T_2 - T_3}{T_1 - T_2}) \tag{5}$$

$$y(k) = b_1 u(k-1) + b_2 u(k-2) - a_1 y(k-1) - a_2 y(k-2);\ u(k-i) \to u(k-i-1);\ y(k-i) \to y(k-i-1) \tag{6}$$

$$\text{if } [u(t) = 1(t), U(s) = \frac{1}{s}] \Rightarrow \lim_{t\to\infty}[y(t)] = \lim_{s\to 0}[sY(s)] = \lim_{s\to 0}[sS_1(s)U(s)] = \lim_{s\to 0}\left[sS_1(s)\frac{1}{s}\right] = \lim_{s\to 0}[S_1(s)] = K \tag{7}$$

$$\text{as well: } \lim_{t\to\infty}[y(t)] = \lim_{k\to\infty}[y(k)] = \lim_{s\to 0}[S_1(s)] = \lim_{z\to 1}[S_1(z)] = \lim_{z\to 1}\left[\frac{b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}\right] = \frac{b_1 + b_2}{1 + a_1 + a_2} = K \tag{8}$$

The influence of the precise calculation of parameters a_i, b_i on the precise calculation of gain K is clear from Equation (7) and (8), it is also documented in the next example and step response in Figure 6, where results of calculation in 32-bit and 64-bit floating point are presented. The parameters a_i, b_i are calculated on the PC in 64-bit arithmetic (the mode double precision). After calculating the output from difference equation (6) and before the output is drawing or

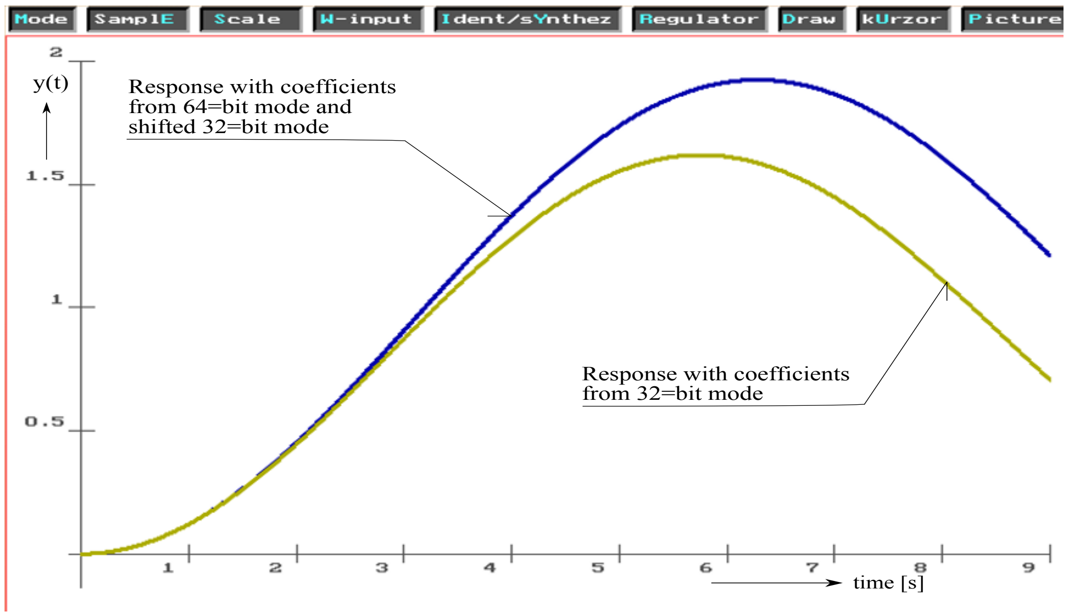

Figure 6. Step response for 64-bit and 32-bit floating point arithmetic.

typing, it is needed to check the calculation of gain K in 64-bit and 32-bit arithmetic. If the precision of gain calculation is not suitable, it is needed to use a bigger sampling interval T_{0y} for calculating parameters of the Z-transform. From a practical point of view, we know that a maximal sampling interval is around 25 [ms] (it depends on the quartz oscillator used by a microcontroller). The best solution is to use computation in 64-bit floating point in the microcontroller. Then, more microcontroller internal memory for calculation will be used and the duration of computation will be larger.

In Figure 6 step responses for the transfer function from the next example is presented. It is clearly shown that the computation of dynamics by difference equations with 32-bit arithmetic cannot secure precise calculation.

Example:

Recalculation of $S_1(s)$ to $S_1(z)$ for parameters $K = 1$; $T^2 = 4$; $2aT = 0.1$; $T_3 = 0$; Sampling $T_{0y} = 1$ [ms]

Parameters of the Z-transform in 64-bit mode are: (Gain K under (7) is $K = 1.000$)

$b_1 = 1.24998955766E^{-}7$; $b_2 = 1.249979140575E^{-}7$; $a_1 = -1.9999747503156$; $a_2 = 0.99997500031249$;

Parameters of the Z-transform in 32-bit mode are: (Gain K under (7) is $K = 0.83885$)

$b_1 = 1.24998955766E^{-}7$; $b_2 = 1.249979140575E^{-}7$; $a_1 = -1.9999747503156$; $a_2 = 0.99997500031249$;

Because computation on a microcontroller is independent from the PC, and the control algorithm is computed on the PC, it is necessity to ensure the synchronization between the PC and the microcontroller. This problem is described in the next subsection.

3.4. Realization of real time under environment Windows

The realization of real time is the main problem in the hybrid and HIL simulation mode. This paper is focused on the problem of dynamic processes realization by the difference equation computed in the microcontroller with 64-bit floating point arithmetic. The importance of the realization of controller/identification algorithm calculation on the PC was commented about many times before. Only if the verification of the control/identification algorithm in real time mode together with real A/D and D/A converters can confirm the application of the algorithm for using in the real PLC system.

The list of main modes for realization of real time on the PC is as follows:

1. MS DOS environment + interrupt service routine controlled by 8253 timer.
2. Windows 98 + MS DOS + interrupt, or Windows CE.
3. Special edition of Linux environment

4. Windows 2000, XP, 7 + multimedia timer in environments like Borland C, Borland Delphi.
5 Windows higher than 98 + RTX environment.
6. Windows higher than 98 + Zyl timer.
7. Windows higher than 98 + TSVA timer.
The author has a lot of experiences with first 5 modes. Most often he used the second mode, because in this mode there is direct time interrupt and also better environment for manipulation with pictures than in the DOS mode. In the first three modes, the handler for A/D and D/A converters is being executed in time interrupt and it makes possible to check the duration of control algorithms computation. From the point of view of the real time simulation experiment duration (3-150 [s]) the sampling interval 1-50 [ms] was used. The top time 50 [ms] was fixed by a simple realization of the interrupt service program and hardware realization of time register in 8253 timer.
The application of a multimedia timer is not a secure realization of a strict sampling interval. Its precision can be improved by a suitable realization of an application program and optimalization of the Windows environment (special environment with only needed service programs of the OS, without all services for network). From the author's experience, this mode can be used, for a well-organized application program, with a 10 [ms] sampling interval, even without any modification of the Windows OS.
The application of real time mode with the RTX (Real Time Extension) environment is conditioned by the application of multicore processors. One of the processor's cores is reserved for RTX and is handling interrupted input/output. Usually RTX was applied together with the serial RS 232 port as input/output to the PC. A disadvantage is that it cannot by used for interrupting the USB port. The RTX environment is not free and is quite complicated to use.
Zyl Timer is a high resolution, long-term Delphi and C++ Builder timer component which provides a higher precision than the standard Delphi/C++ Builder TTimer component. Zyl Timer is a thread based timer and thanks to that the architecture provides a higher precision, close to 1 millisecond (it is possible that you cannot obtain a clear resolution of 1ms on all the systems, but it must be under 2ms) which is inevitable in time critical applications. TSVA Timer is similar to Zyl Timer. The programming of parts of a program with the control/identification algorithm for real time mode requires from the programmer a totally different approach than programming other parts of programs. It is also best to implement own drivers for A/D, D/A converters and especially time interrupts procedures.

4. Real time simulation experiments

The author of this paper has verified several new modifications of known control algorithms by using working modes, where controlled processes run in real time, independently from the control algorithm, which runs in real time on the PC. Some of these new modifications of control algorithms are described in [4, 6–9]. In the papers mentioned, a detailed information with the control algorithms derivation is described and also more simulation experiments in real time, for comparison of initial and modified algorithms, are illustrated. In the current paper, there is only basic information about control algorithms, because we focus on the hybrid and HIL working modes, using which the controlled processes were realized. Therefore, only three simulation experiments are illustrated and described. The first simulation experiment was realized using the analogue mode of controlled processes, the second experiment was realized using the HIL mode and the third experiment was realized using the hybrid mode of controlled processes.
For verification of adaptive control algorithms for controlled processes without "time delay", the author has used analogue models of controlled processes described in [1]. There was used real time continuous identification of controlled system parameters and after identification the controller's algorithm parameters synthesis. By using this approach the controller is updated in every sampling interval and the incoming input-output information is capable to improve the parameter estimation process. In principle, the on-line mathematical model of the controlled process and on-line synthesis of the control algorithm for demanded quality of control process were suggested. This procedure is called Self-Tuning Control (STC), or indirect adaptive control. As noted previously, the type of processes with "time delay" is not possible to realize by using the analogue environment, therefore the hybrid and HIL modes were made-up for its realization. The second simulation experiment was realized under HIL environment by using a control algorithm called adaptive extended Dead-Beat algorithm (marked as eDB). The classic Dead-Beat algorithm called also "discreet version of time optimal control" is marked as DB(n), where "n" in brackets means that after "n" steps of control there occurs the steady state mode, where "n" is the "order" of the controlled process model. This kind of control produces very big controller output (the value of first output depends also on the set point and sampling interval of controller output) therefore using this algorithm

is very problematic if the controller output is limited. The author made a derivation of extension of DB(n) algorithm [4], working also if the controller output is limited, and this modification was marked as the eDB algorithm. The third simulation experiment was realized under the hybrid mode with the controlled process where dominant "transport delay" occurs and with control algorithm eDB-d, where letter "d" in the algorithm characterizes that algorithm is suitable for processes with dominant time delay. The new modification of the DB(n) algorithm marked as eDB-d was derived by author of this article and described in [4] in more detail. From the author's point of view, it is very important that the last two modifications of the control algorithm would not be verified without using the hybrid and HIL simulation modes, the principles of realization of which are described in this paper.

In each of the simulation experiments, the values of the controller output were plotted five times smaller, in order to keep nonoverlapping lines representing the outputs from the controller and from the controlled process. Also it is seen that the controller output is the "step variable", and therefore it may change in every sampling interval of the controller output, especially in the hybrid and HIL modes at the steady state. This "oscillation" in the steady state is caused by the properties of A/D and D/A converters. Also in the steady state mode the A/D converter is generating the "quantization error" (see Figure 2) and this error is amplified with the proportional gain of the controller and modified on the 12-bit D/A converter, which produce the output from the controller. Also in the "real control loops", every real control loop contains the stochastic noise, produced during the measurement of the controlled variable and by its transformation over A/D converters. But in the digital simulation on the PC all variables are represented as numbers in the 48-bit or 64-bit floating point, therefore the output from the controller in the steady state mode is the straight line without oscillations. Therefore, when we look at the illustration of the simulation experiments, the presence of oscillations in the controller output in the steady state is the confirmation that this simulation experiment is done with A/D and D/A converters. The illustration of the controlled variable in the steady state also contain the oscillations, but the amplitude of the oscillation of the controlled variable is smaller with comparison to the controller output, as can be seen in the presented simulation experiments. The last note is about the execution of simulation experiments in real time. It is clear that real processes are executed in real time independent from the control systemwith which they are connected through an interface, most often through A/D and D/A converters. The control system has to be adapted to the function of the controlled process. In the HIL mode, controlled processes are executed in real time and are physically independent from the controller realised on the PC, therefore the "controller - PC" has to be adapted to the running of the controlled processes and to use the real time. In the hybrid mode, parallel execution of the controlled process and the controller depends on the programmer's ability, but real time has to be running in parallel mode based on timer - the PC hardware.

Comparison of computer simulation and HIL simulation of adaptive PIDD^2 control, including tuning of identified parameter by HIL simulation, is shown in Figure 7.

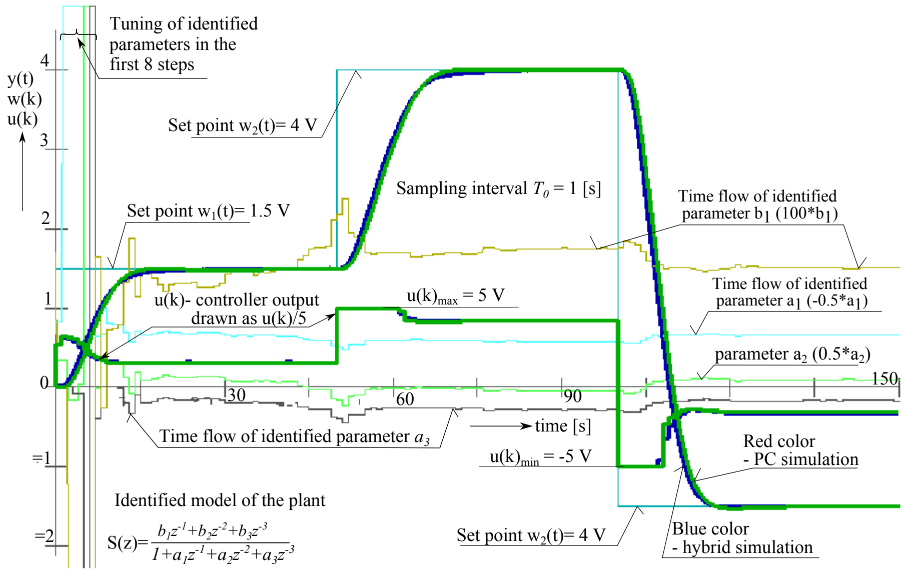

Figure 7. Comparison of PC and hybrid simulation on adaptive PID control.

For the synthesis of the controller the third range-identified model was assumed, although the controlled process has behaviour of fifth order. The algorithm of PIDD^2 in contrast to the PID algorithm contains "element" D^2 and is the "best" hybrid control algorithm for proportional controlled processes of third order. The synthesis of the continuous and discrete PIDD^2 control algorithm from identified data, derived by the author, was described in [7] and [8]. On the picture it can be seen that the hybrid and discrete PC simulation produce very similar responses. The response from the hybrid mode can be recognizable by a small oscillation of the controller output value and identified parameter values in steady state modes. From identified parameters of the process model, which are in the nominator of the Z-transfer function, only the b_1 parameter is illustrated in Figure 7. This parameter has a very small value (see equation (5) and example), therefore it is illustrated a hundred times bigger, and is written in the callout in Figure 7. Parameters a_1 and a_3 in the denominator of the Z-transfer function are always negative with values around one–two, if identification is made correctly. Due to easy and quick checking of correct identification of the models parameters, a_1 is illustrated with the negative sign. The identification algorithm is based on the recursive version of least-square and for the first regular calculation of 6 parameters of the Z-transfer function has to be executed first 8 steps of algorithm. This process of tuning the identified parameters can be seen in the left side of Figure 7. Because the controlled variable measured of the analogue model contains also the quantization error from the A/D converter and time flow of identified parameters was registered during real time simulation mode, "tuning of identified parameters" continues during the whole simulation experiment.

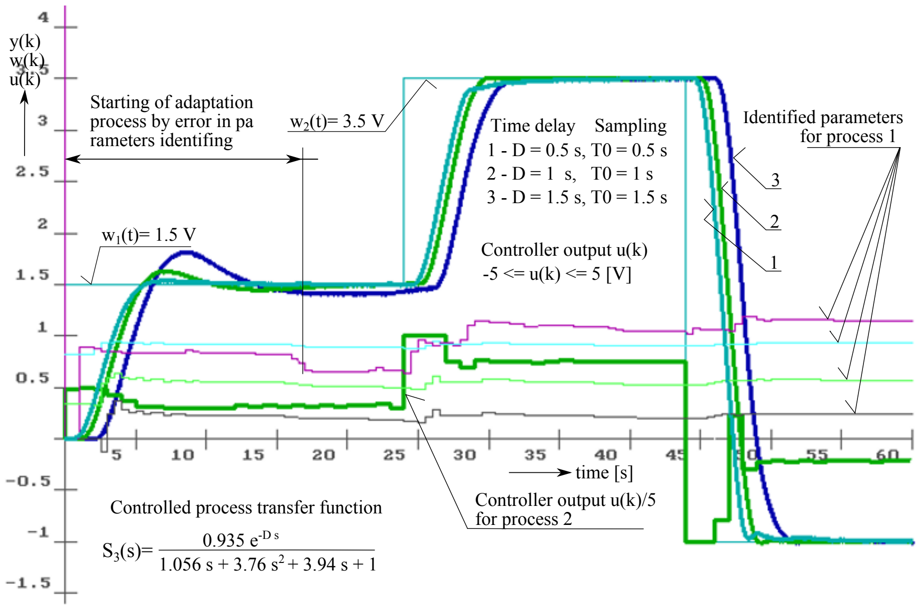

Figure 8. Hybrid simulation on adaptive eDB (time delayed) control.

The simulation experiment shown in Figure 8 is a hybrid simulation of adaptive control where the controlled system has time delay. Time delay has the same value as the sampling interval of the controller and simulation experiments were done for three different sizes of sampling intervals. Such type of a controlled system is the most difficult task for control, but from Figure 8 it is evident that the eDB adaptive algorithm provides very good quality of control loop response, even if the controlled process has dead time with value around the sampling interval. The derivation of the eDB algorithm was done by the author and is described in [4]. The verification of this type of algorithms in real time by simulation experiments can be prepared only after realization of this hybrid mode or the HIL mode. In Figure 8 it can be seen that the controller output and identified parameter values have the "oscillations" in the steady state, which indicate that A/D and D/A converters were used in simulation experiments. On the left side of Figure 8 we can see that, in contrast to Figure 7, there is little change of the identified parameters values. It is due to the fact that initial values of the identified parameters are very close to the final value. In the time 24 [s] and 44 [s], where the set point is changed (1.5 [V] to the 3.5 [V], 3.5 [V] to the - 1 [V]) it can be seen that the output from the controller was changed at the same time as the set point value, but the controlled variable has the "time delay" according to the controlled process transfer function, which is the model of the controlled process. In contrast to Figure 7, Figure 8 illustrated the parameters of a continuous transfer function.

Figure 9 shows the simulation experiment in real time at hybrid mode on the closed control loop system with the controlled process whose dynamics has dominated time delay. In this experiment the verification of the eDB-d control algorithm is presented. The algorithm derivation was made by the author of this paper and is described in [4]. The simulation experiment also shows that the response on load-disturbance is excellent, but only for the case when disturbance is measured. Also it confirmed the dominant opinion that time delay produces response on load-disturbance with big overshoot and with big settling time. The presented eDB-d control algorithm, as can be seen in Figure 9, can realize very small overshoot and small settling time in the transient mode due to the ability of the classical DB(n) algorithm, which was preserved even in the case that the controller output has limitation and the controlled process has big time delay. In this simulation experiment the condition was used that time delay is the whole multiple of the sampling interval. In real processes this can be keep by means of necessary change of the sampling interval.

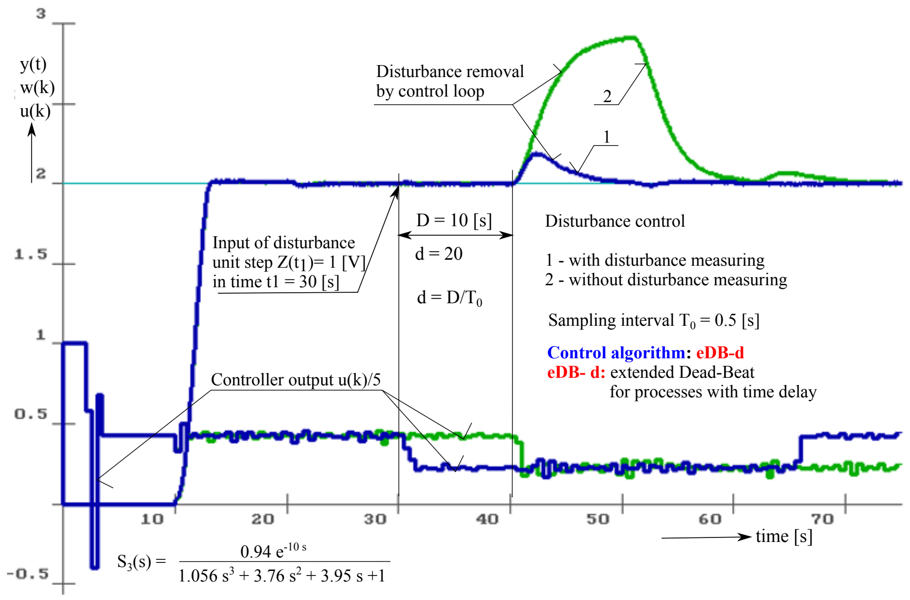

Figure 9. Hybrid simulation of process with dominating time delay.

The fact that simulation experiments were realized in real time mode and with the A/D and D/A converters can be seen in the figures with time responses of controlled variables (y(t)) and controller variables (u(k)). Both variables, especially in the steady state mode, have chalk-line in the PC mode (red color line in Figure 7). But in the hybrid mode A/D and D/A converters generate the quantization error, which is clearly seen in Figures 8 and 9 on the controller output u(k) and the controlled variable y(t). In all real time simulation experiments, the sampling interval for the controlled variable y(k) is 10–50 times smaller than the sampling interval for u(k). Therefore, frequency of the noise on y(k) is larger than for u(k). The presence of noise in documentation of simulation experiments is a suitable character for recognition whether the simulation experiment was prepared only as a PC simulation or as a hybrid simulation. The derivation of new modifications of the control algorithm, which were created and verified in the described hybrid modes for controlled systems, is sufficient verification to confirm usefulness of simulation experiments with the mode "hardware in the loop of simulation".

5. Conclusion

Based on our present experience, the described HIL simulation experiments provide good results for the laboratory verification and teaching of control algorithms. This way, classical and adaptive PID, PIDD^2, DB(n), eDB, eDB-d algorithms, algorithms with state variables and sliding mode algorithms were verified. This paper described two simulation environments for control algorithms verification and realization of dynamic processes for verifying of identification algorithms. Simulation experiments with digital models of controlled systems and real time HIL simulation were realized in a program environment of ADAPTLAB, developed and realized by author. This environment is suitable for developing and verifuing classical as well as adaptive control algorithms for SISO and MIMO control loops. The program can be used in three basic modes: simulation, measurement and hybrid mode. The simulation mode works with continuous transfer function set by operator, it is a classical discrete simulation on a PC. In the measurement mode the output (input) from a model of the controlled system realized on a microcontroller or by the human reactions, is measured with A/D (D/A) converter with a sampling interval controlled by real time clock from a PC or from A/D converters. The measurement mode is used for identification of parameters for the model of human reaction, or for real time identification of the controlled process model for adaptive control. The hybrid mode is a digital calculation of the controlled system behaviour and control algorithms on a PC in real time, combined with the realization of input/output for the process and the controller over the A/D (D/A) converters, described in this paper.
The analogue part of the described environment is successfully used in laboratory practice in subjects "Direct Digital Control", "Computer Controlled Systems" and "Simulation of the Dynamical Systems" at the University of Zilina. The environment also helps developing the verification of new adaptive control algorithms. Actual experience of the author showed that the HIL simulation is more appropriate than digital simulation, both for verification and for teaching. A new control and identification algorithm has to be verified in a real time environment, because they will be used in the praxis always in an environment of real time. For teaching, the realization of own "programming environment" which can control measurement and algorithms running in real time is a hard work expanded to several semesters. But students after its realization can better understand the meaning of every constant and modifications in the verified algorithms. External "hardware" in the models can correctly cooperate with the software on the PC only if it is written correctly. In future, the author plans to focus on self-tuned control algorithms and the problem of variable transport delay identification. In the area of "intelligent transport systems" the research will focus on the measurement, identification and modeling of dynamics in the environment "driver/car/traffic situation".

Acknowledgment

This contribution was prepared under the project VEGA 1/1099/11 "Modeling and Simulation of Dynamical Interaction in Environment of Driver/Car/Traffic Situation".

References

[1] Alexík S., Alexík M., Simulation Environment for Real Time Verification of Control Algorithm. In: Troch I., Breitenecker F. (Ed.), 5th Vienna Symposium on Mathematical Modelling (MATHMOD) (Vienna University of Technology, Austria, 2006)
[2] Åström K.J., Hägglund T., PID Controllers, 2nd Edition (Addison-Wesley Longman Publishing Co., Boston, MA, USA, 1994)
[3] Alexík, M., Modelling and identification of eye-hand dynamics, Simulat. Pract. Theory, 8, 25-38, 2000
[4] Alexík M., Modification of Dead Beat Algorithm for Control Processes with Time Delay, paper no. 3402, In: 16th IFAC WORLD CONGRESS (Prague, Czech Republic, 2005)
[5] Alexík M., Modelling and Simulation of Interaction in Driver/Vehicle Dynamics, In: Troch I. Breitenecker F. (Ed.), 6th Vienna Conference on Mathematical Modelling (MATHMOD'09), ARGESIM Rep., 35, 500-510, 2009
[6] Alexík M., Modelling of Process Control that have Time Delay. In: Al-Dabbas D. (Ed.), 12th International Conference on Modelling and Simulation (UKSIM 2010), IEEE Computer Society, 221-226, 2010

[7] Alexík M., Usage of Real Time Hybrid Simulation for Verification of Control Algorithms In: Dekker L. (Ed.) 4th International EUROSIM 2001 Congress "SHAPING FUTURE WITH SIMULATION" (TU Delft, The Netherlands, 2001)

[8] Alexík M., Simulation Experiments with Self Tuning PSD Control Algorithm. In: Control Systems Design 2003. In: Proceedings from the 2nd IFAC Conference Bratislava, Slovak Republic, 7-10 September 2003, Kozak Š., Huba M. (Ed.) (International Federation of Automatic Control by ELSEVIER LTD, Kidlington, Oxford OX5 IGB, UK, 2004)

[9] Alexík M., Simulation Experiments with Self Tuning PSD Control Algorithm, In: Al-Dabbas D. (Ed.), IFAC Tenth International Conference on Modelling and Simulation (EUROSIM/UKSim2008), IEEE Computer Society, 34-39, 2008