

described by transfer functions. Their usefulness lies in their simplicity and the possibility to compute the expected results in advance and compare them with the results from simulation experiments. For example, the courses shown in Figure 3 were obtained with the use of a linear model with the transfer function

$$G_s = \frac{1}{s^2 + 1,2s + 1} e^{-1,5s} \quad (1)$$

The second type of models used here are models supplemented by some precautions providing a better simulation of physical reality than is possible using pure linear models. Such more complicated models may contain non-linearities, and their complexity may vary from models based on linear models with added non-linearity, e.g., a saturation of capacity, tank overflow, stopping the temporary increase in temperature while changing the phase from liquid state to gaseous state, etc., to pure non-linear models, which represents phenomena observable on the physical device that are significant for control. The advantage of models of this type is their easy reconfigurability. Some problems can occur while performing simulations with such complex models. These problems are not caused by a wrong model design, but are connected with the simulation solver. When there is a possibility, that they may occur, preventive measures must be taken.

The third and most challenging type of models used here were physical models, or replica simulation models of them.

3.1 Model of a three-tank cascade laboratory set-up

Before implementing the autotuning control algorithm on a three-tank cascade set-up, we worked with a model of it in various levels of complexity, as mentioned above. One of the advantages of such modelling is that tuning results can be obtained in less time than when a physical model is directly used. The mathematical description can be nonlinear, and is based on a volumic flow rate balance,

$$\frac{d}{dt} h_1(t) = \frac{1}{A} q_1(t) - \frac{Kv_{12} \sqrt{h_1(t) - h_2(t)}}{A} \quad (2)$$

$$\frac{d}{dt} h_2(t) = \frac{Kv_{12} \sqrt{h_1(t) - h_2(t)}}{A} + \frac{Kv_{12} \sqrt{h_3(t) - h_2(t)}}{A} - \frac{Kv_{20} \sqrt{h_2(t)}}{A} \quad (3)$$

$$\frac{d}{dt} h_3(t) = \frac{1}{A} q_3(t) - \frac{Kv_{32} \sqrt{h_3(t) - h_2(t)}}{A} \quad (4)$$

where A means the cross section (the same for all three tanks) in dm^2

Kv_{xy} means the flow rate coefficients used in the equations quantifying the (volumic) flow rate through the valve from tank x to tank y

h_x means the water level in tank x

q_x means the supply flow rate, or the mutual flow rates between the tanks ($x, y = 1, 2, 3$) or to the atmosphere ($y = 0$)

The mathematical model gives a credible representation of the three-tank cascade. The only behaviour not described by the equation involves situations when the tanks overflow, so tanks of unlimited height are considered in the simulation. However, overflow is not a typical or important state in our testing.

A Simulink representation of the mathematical model enables the initial conditions to be computed before the simulation starts, so it is not then necessary to compute the initial conditions separately outside the Simulink scheme.

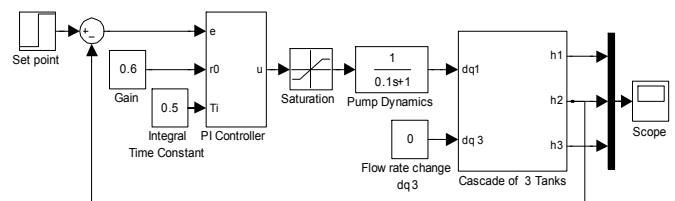


Fig.4 Simulink block scheme of the case the level in tank 2 of the cascade is controlled

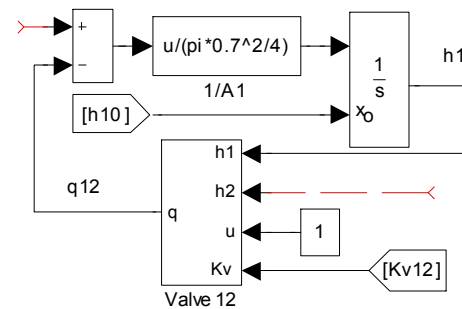


Fig.5 Water tank Simulink block scheme

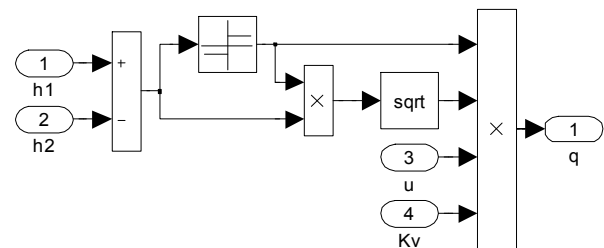


Fig.6 Simulink model for flow rate computation

Figure 4 shows the Simulink scheme of three-tank cascade laboratory model when the second tank level is controlled. Figures 5 and 6 show the main parts of the

three-tank cascade Simulink scheme, where Fig. 6 is the Simulink scheme of the equation, $|h_1 - h_2|$ is computed as $(\text{sgn}(h_1 - h_2))(h_1 - h_2)$.

3.2 Model of the control valve

Another example of the non-linear models used here is a model of the control valve used for position control by means of a linear double acting pneumatic drive.

The mass flow rate through a control valve can be described by the equations

$$M(t) = G_n \varepsilon(t) \sqrt{\frac{T_0}{T_{in}}} \frac{P_{in}(t)}{P_0} \sqrt{\frac{P_{out}(t)}{P_{in}(t)} \left(1 - \frac{P_{out}(t)}{P_{in}(t)}\right)} \quad (5)$$

when $\frac{P_{out}(t)}{P_{in}(t)} > 0,5$ (subcritical case)

$$M(t) = G_n \varepsilon(t) \sqrt{\frac{T_0}{T_{in}}} \frac{P_{in}(t)}{2P_0} \quad (6)$$

when $\frac{P_{out}(t)}{P_{in}(t)} < 0,5$ (supercritical case)

where M means the mass flow rate through the valve,
 G_n means the nominal flow capacity of the valve
 ε means relative valve opening
 T_0 means normal temperature
 T_{in} means temperature of the air flowing into the valve,
 P_0 means normal pressure,
 P_{in} means pressure of the air flowing into the

valve,

P_{out} means pressure of the air flowing out of the valve.

$$Q(t) = \frac{M(t)}{\rho} \quad (7)$$

where Q means volumic flow

M means mass flow

ρ means air density

$$\rho = \rho_0 \frac{T_0}{T} \frac{P}{P_0} \quad (8)$$

where ρ means air density

ρ_0 means normal air density

T means air temperature

T_0 means normal temperature

P means air pressure

P_0 means normal air pressure

Equations (5) and (6) can be converted into the Simulink scheme, as shown in Figure 7. Because we are considering a double acting pneumatic drive, it is necessary to use two of these schemes to model the complete control valve.

Let us assume that the pneumatic drive is connected to pressure values $P_{out1}(t)$ and $P_{in2}(t)$. Then we need to compute these values during simulation to be able to compute the volumic flows $Q_1(t)$ and $Q_2(t)$ to and from the pneumatic drive.

One way to compute values $P_{out1}(t)$ and $P_{in2}(t)$ is to derive the functions $P_{out1}(t, P_{in1}, P_{out2}, \varepsilon(t))$ and $P_{in2}(t, P_{in1}, P_{out2}, \varepsilon(t))$. This is possible; we have two conditions for joining these values

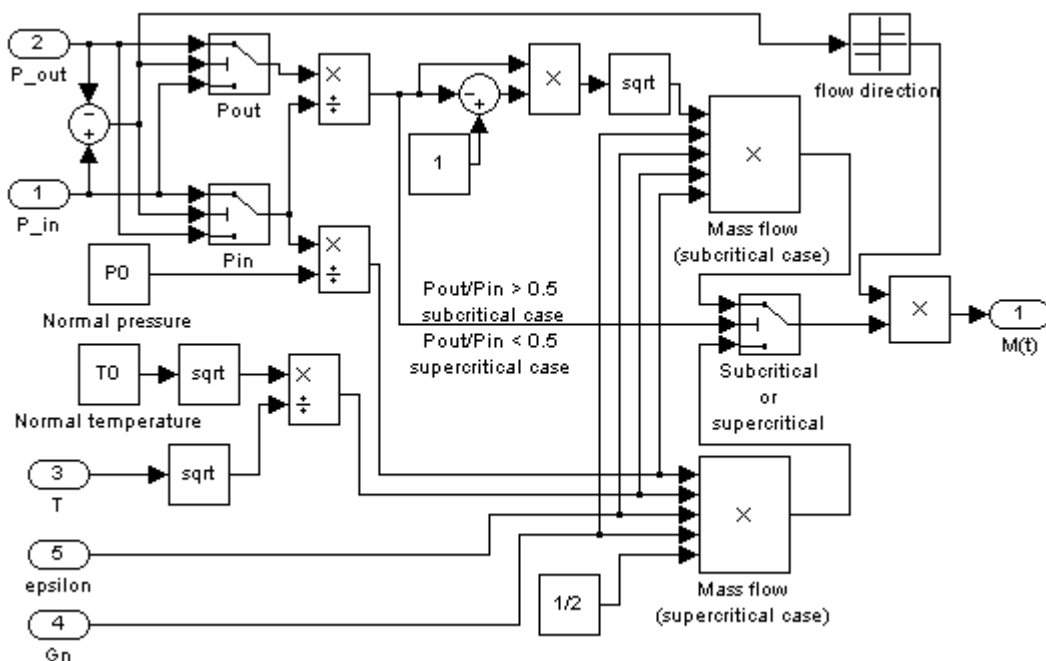


Fig.7 Simulink scheme of computation of mass flow of air flowing through control valve

$$P_{in2} = P_{out1} - \frac{m}{A} \tag{9}$$

where P_{in2} means the pressure of the air flowing out of the drive,

P_{out1} means the pressure of the air flowing into the drive,

m means the weight of the mass,

A means the area of the piston.

$$Q_1(t) = Q_2(t) \tag{10}$$

However, this solution is very complicated. The derivation of the functions $P_{out1}(t, P_{in1}, P_{out2}, \varepsilon(t))$ and $P_{in2}(t, P_{in1}, P_{out2}, \varepsilon(t))$ is very complicated, and we also get seven possible pairs of values P_{out1} and P_{in2} , one for the case when the flow is supercritical in both parts of control valve, two for the case when the flow is supercritical in the first part of control valve and in its second part the flow is subcritical, two for the case when the flow is subcritical in the first part of control valve and in its second part the flow is supercritical, and two for the case when the flow is subcritical in both parts of control valve.

The second solution is to run an iterative process that computes the values P_{out1} and P_{in2} from equations (5), (6), (7), and (8). The iterative process is based on the increase of values P_{out1} and P_{in2} , while it is tested if $Q_1(t)$ is equal to $Q_2(t)$. When they are equal, the searched values P_{out1} and P_{in2} are found.

The iterative process can be constructed as a Matlab function in the Simulink scheme. To avoid additional work with the complete iterative process in the Matlab function, we can use a second Simulink scheme of the control valve that is executed from the Matlab function. This second (slave) model is the same as the first (main) model, with the same parameters, with the exception that values P_{out1} and P_{in2} are connected to the Ramp block, which provides a continually increasing value. In this second model, the values $Q_1(t)$ and $Q_2(t)$ are compared, and when they reach the same value (when they are equal), the simulation can be stopped, because the searched values of P_{out1} and P_{in2} have been found and can be propagated to the main model. It is necessary to execute the slave model in each iteration step of execution of the main model.

A slave model executed from main model can be used not only to compute values, for computation of which it is necessary to use an iterative process. The simple slave model can also be used to show the internal values of any Matlab function

3.3 Numeric problems simulating

Models of valves, and more generally of resistances, very often form a part of simulation models that are

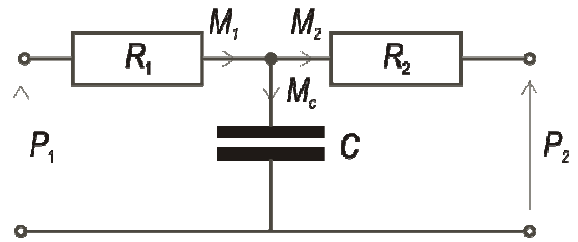


Fig.8 RC circuit scheme

created by interconnecting blocks modelling process phenomena in a simplified way. Each such block represents a certain real element, where we imagine that a certain specific physical phenomenon, lumped and separated from the others, takes place. For example, creating pressure-flowrate models we get chains of resistances (valves) and capacitors (volumes), i.e. we can create a pneumatic circuit model as an analogy of RC electric circuit models. Let us consider such a very simple RC circuit, the scheme of which is depicted in Figure 8.

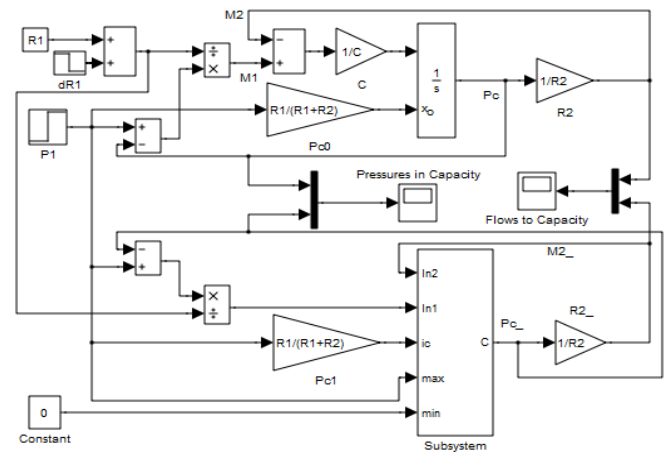


Fig.9 Standard and limited accumulation models of an RC circuit

A problem that we would like to mention in connection with simulating physical models is the numeric instability that sometimes occurs when resistances and capacities have specific values which cannot be mastered by step by step simulation with the setting of simulation conditions used here (step and solver) and in a standard simulation scheme (see the upper part of the Simulink program in Fig, 9). In such cases, something happens in each step of this numerically performed simulation that does not correspond to physical reality. Let us consider two situations in the experiments that are later demonstrated by means of results from simulation. In the first experiment, let the input pressure (voltage, temperature, etc. in analogous models of the same RC type) change from 12 to 15 kPa. As a step response to this increase after the first step, the size of which was set too large (Δt

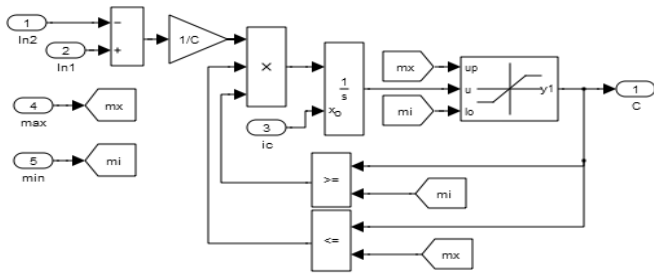


Fig.10 Subsystem performing limited accumulation

= 0,15 s for the integration method used here (Euler)), it is computed that the mass increase in the capacity is $\Delta m_1 = M_1(t)\Delta t$ and the corresponding pressure increase in the capacity is

$$\Delta P_C = \frac{RT}{V} \Delta m_1 \quad (11)$$

There is nothing to prevent pressure P_C at the time instant $t + \Delta t$ being greater than the input pressure. This cannot happen in continuous time, because each continuous increase in the pressure P_C must lead to an immediate decrease in the flow to the capacitor. Thus there is no tendency to reverse the direction that we can observe in the discrete simulation if the pressure in the capacitor becomes greater than the input pressure. Because this change in the flow orientation is impossible for physical or technical reasons, it is necessary to add to the simulation model some precautions against it. One of reasonable solutions is to stop the integral action as soon as the state variable measuring accumulation in the capacity exceeds defined limits. In our example, the input pressure P_1 is the upper limit, and atmosphere (zero overpressure) is the lower limit. The stopping mechanism performed by the Subsystem in Fig. 10 can be clearly seen under the mask in Fig. 9. It is very similar to an antiwind-up solution.

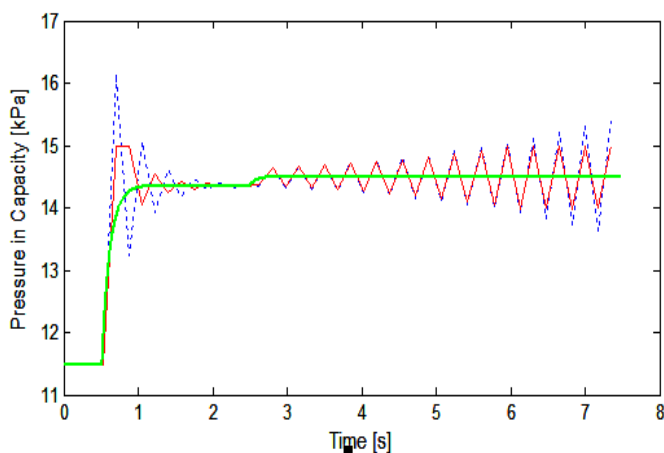


Fig.11 Step responses of an RC circuit under various simulations

The results from simulations carried out under various conditions are shown and compared in Fig.11.

The smooth course in Fig.11 is the “right” reaction to step changes in the input pressure P_1 and the value of resistor R_1 , when the step size and/or the integration method have been set in such a way that the obtained course is very close to the continuous reality. The full line with some saw teeth parts depicts the response that was obtained from the model using the block Subsystem for limiting accumulation in the capacitor. The dotted line marks the response from a standard model; both responses have been obtained under the same setting of the simulation parameters.

The use of blocks that model saturation of the capacitor is important in complex models containing more than one mutually interconnected RC submodel. They help to keep numeric stability, and the results are much closer to reality. No other model precautions are necessary to ensure inalterability of the flow orientation.

4 Laboratory model

As a laboratory model we utilize a three-tank cascade set-up. The set-up consists of three interconnected tanks (see Fig. 12). Water is supplied into Tank One and Tank Three. Each tank is equipped with a pressure sensor that is used for measuring the water level. In addition to their mutual interconnection, each of the tanks in the cascade has its own outlet valve, which enables various operational modes with different dynamics to be simulated. All valves in the laboratory model can be adjusted only manually.

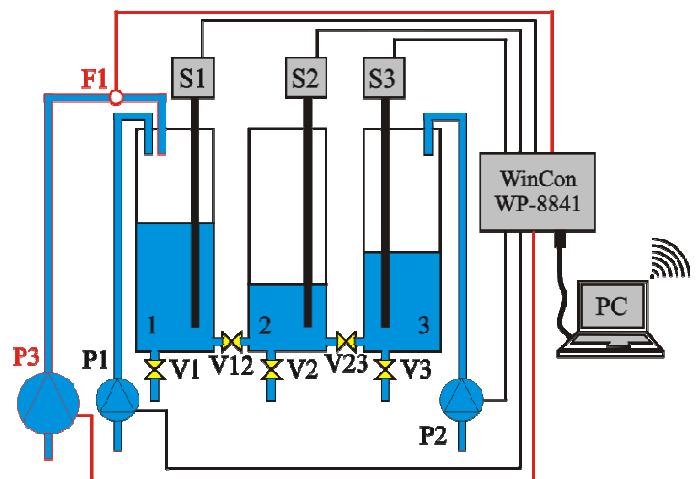


Fig.12 Three-tank cascade laboratory model scheme

Pumps P1 and P3 feed Tank One, while pump P2 feeds Tank Three. Tank Two is unfed. The hydrostatic pressure is measured by pressure sensors S1, S2, and S3, and the value is converted to the water level value. It is

also possible to measure the flow rate to Tank One through pump P3 by flow meter F1. The laboratory model is of the third order, so that an unstable control process can be achieved if parameters from the area of unstable value combinations are used for setting the controller. Originally the laboratory model contained only pressure sensors S1, S2, and S3 and pumps P1 and P2, and the circuit containing pump P3 and flow meter F1 was added later. The driving electronics of this circuit is based on a recommendation given in [6].

4.1 WinCon programmable automation controller

The fundamental hardware part is the control device. The PAC WinCon (PAC means Programmable Automation Controller) was chosen as the control device. This PAC is based on the ARM processor, and an embedded operating system is used. WinCon uses Windows CE as the operating system. We chose this type of control device because Rex control software can be used as control software, and also because of the RexLib library, which adds some block into the Simulink block library and allows direct communication between WinCon and Simulink.

The communication between Simulink and WinCon provides an opportunity to split the whole controller with the autotuning algorithm into two parts. The own control algorithm of PID type and the harmonic signal generator are placed in WinCon, and the autotuning mechanism, which is an extension of the PID algorithm, is placed in Simulink. The control algorithm placed in WinCon works independently of the connection to the autotuning algorithm in Simulink.

The selected type WinCon W-8741-G contains seven expansion module slots, two LAN connectors, two USB connectors, one RS-232 connector, one RS-485 connector, one VGA connector and one Compact Flash card slot. We use the expansion modules I-8017H (14-bit 100K sampling rate 8-channel analog input module), I-8024 (4-channel 14-bit analog output module), I-8053 (16-channel isolated digital input module - differential input), I-8064 (8-channel power relay module), I-87018 (8-channel thermocouple input module), and I-8080 module (4/8 channel counter/frequency module). The I-8017H module is used to get data from pressure sensors S1, S2, and S3 and also from flow meter F1 (the flow meter has a frequency output, and a frequency to voltage converter is used). Modules I-8024 and I-8064 drive pumps P1, P2, and P3. So both types are possible: continuous control, and on/off control of the pumps. The other modules are not used to control the laboratory model of the three-tank cascade. We use this PAC not only to control this laboratory model but also for

experiments on a biomass-fired boiler [13], where the other modules, especially module I-87018, are useful.

This PAC also exists in other types, which differ in the built-in devices that they offer. For example, W-8331-G contains three expansion module slots, one LAN connector, one USB connector, one RS-232 connector, one RS-485 connector, two PS/2 connectors (keyboard and mouse), one VGA connector and one Compact Flash card slot, while W-8701-R2 contains seven expansions module slots, one LAN connector, three RS-232 connectors, one RS-485 connector and one Compact Flash card slot.

4.2 Rex control software

The tool for WinCon algorithm development and for monitoring it is a couple of programs, RexDraw and RexView. Rex Draw is designed for developing, organizing and compiling algorithms, and RexView is designed for compiled configuration upload and for station activity monitoring. It can provide information about the activity of its own station and about the activity of individual algorithms. The algorithm parameters can be changed.

The supporting part of the software is the function block set library RexLib, which enables use of the Simulink extension of the Matlab program for algorithm development. The advantage of algorithm development in Simulink is that it enables the algorithm to be simulated. Simulation is not possible with the use of RexDraw or RexView. On the other hand, compilation is not possible in Simulink, so RexDraw has to be used.

The RexLib library contains all blocks which are necessary for developing the control algorithm and which are also contained in the RexDraw program. RexLib library is designated for use with Simulink. The algorithm configuration blocks are also included, so the whole algorithm can be simulated even in situations when the algorithm is divided into more than one individual block scheme. All particular algorithms can also be simulated separately.

When an algorithm designated for use outside Simulink is developed in Simulink, it is necessary to use only blocks included in the RexLib library. Other blocks can also be used, but only for simulation purposes. They cannot be a part of the algorithm that is developed. The extension of the Rex control system can be used, which allows to leave these blocks to be left in the scheme if they are named in such a way that their name contains the word simulation. Then it is not necessary to delete these blocks before final compilation of the algorithm.

5 Experiments comparing results, and the demands of controller tuning by various methods

Three examples are shown, in Fig. 14, Fig. 15 and Fig. 16, of a controller retuning process using three different tuning methods: the relay method [1], the momentum method [11], and the frequency-based tuning method [17]. The examples demonstrate the problems of model-free controller setting on-the-spot. They also

show what kind of information is missing if the explanation focuses on the theoretical background of these methods, which is often the case in publications and in university education. All experiments are performed in the following simulation experiment sequence: first, the desired water level in Tank Two is increased by $\Delta h_2 = 0,1$ dm, then it is decreased to the initial water level. In the second period, the controller is tuned, and, finally, the same experiment as in step one is

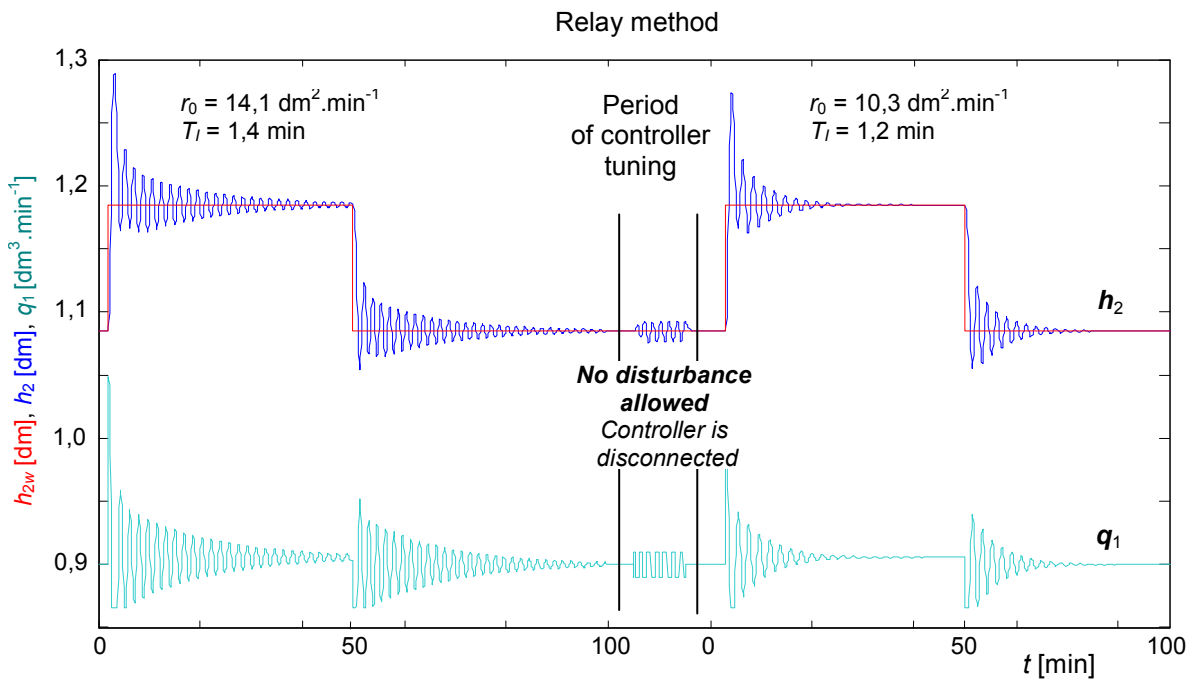


Fig.13 Relay method tuning experiment (manipulated variable q_1 depicted after dividing values by 10)

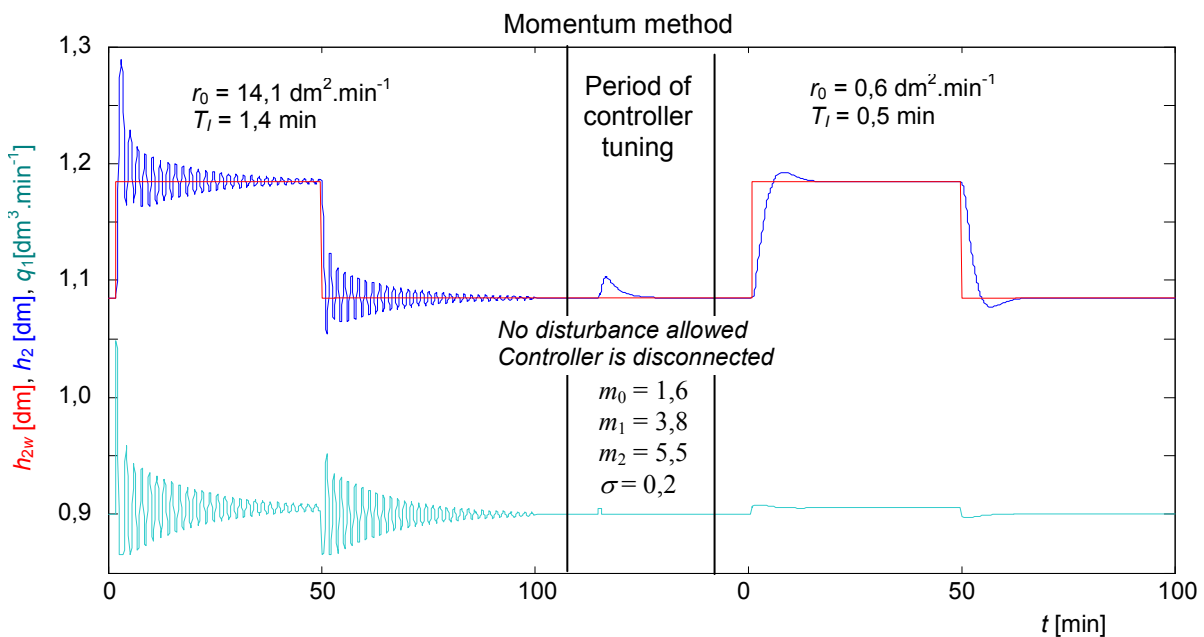


Fig.14 Momentum method tuning experiment (manipulated variable q_1 depicted after dividing values by 10)

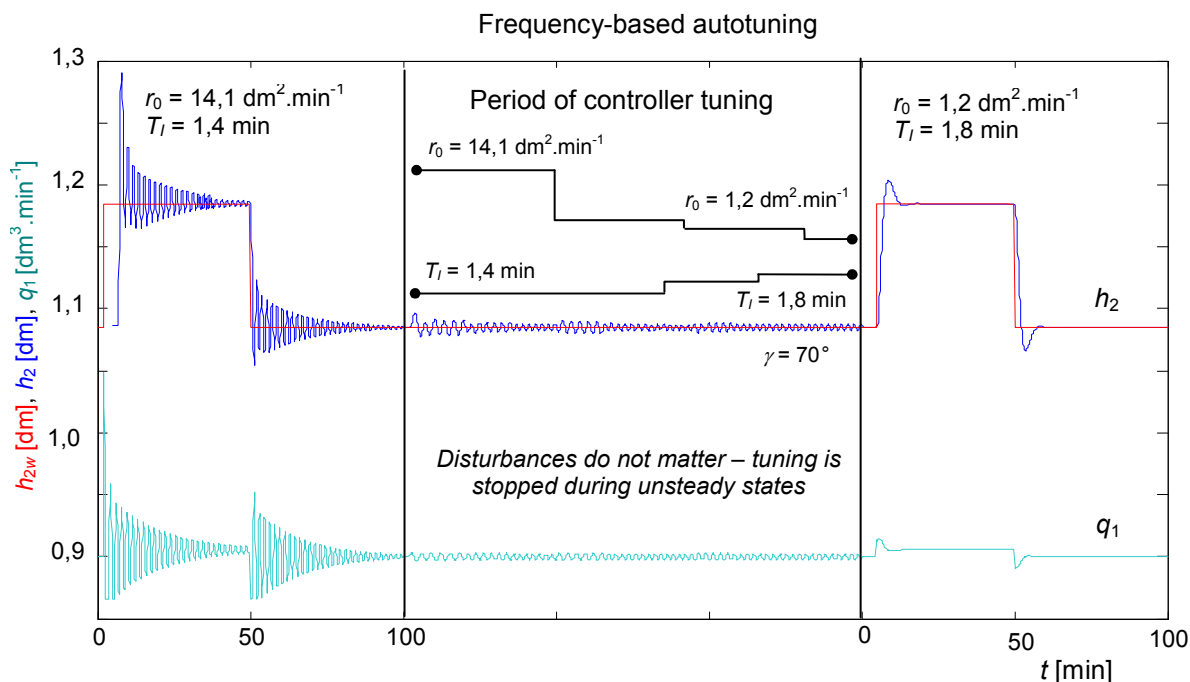


Fig.15 Frequency-based autotuning experiment (manipulated variable q_1 depicted after dividing values by 10)

repeated, but with a new controller setting. The first period demonstrates the control performance when the controller setting is unsuitable. The control process is stable, but it oscillates considerably. The second period demonstrates the retuning process, which differs for the three tuning methods used here. The third period demonstrates the control performance when the controller is retuned. The control performance is better than in the first period.

As is shown in the figures, the courses when the tuned controller is used differ significantly. The relay method produces the most oscillating response, while the response of other methods is not oscillating.

The second major difference between the tuning periods in the figures is their length. Therefore, for the controller obtained here, frequency-based autotuning can provide better results. However, when we take into account the length of the tuning period, the whole tuning process is not so much better. Nevertheless, there is another significant difference – the need for the controller to be disconnected.

When we use the relay method to tune the controller, it is necessary to disconnect it temporarily, then to perform the relay experiment and reconnect the controller into the loop. The same is necessary when we use the momentum method to tune the controller. When we use the frequency-based tuning method, however the control function is fully operable during the tuning period. Thus, the need to disconnect the controller or restrict its function is a further consideration when comparing tuning methods.

Tuning methods should be compared in several aspects, but the only considerations mentioned here are the quality of the controller setting and the information that is required about the controlled plant/process. All such considerations should be taken into account in comparing the tuning methods focused on the needs of practical application.

6 Conclusions

A frequency-based PID controller tuning method that uses control quality indicators has been presented not only in terms of its operating principles, but mainly in terms of the experience obtained during implementation and testing with various kinds of controlled plant models (linear, non-linear, and physical). We used these types of model to some problems and their solution if the simulation should run in a correct, reality reflecting way.

The PID controller tuning methods were tested on a three-tank cascade set-up and on a simulation model of this cascade with emphasis on modeling all details corresponding to real application. This circumstance is very important, because it is only when we reflect all real conditions under which autotuning starts and runs that a serious and useful comparison of various approaches and of autotuning efficiency can be made. This is what is often missing in the literature. From our tests it follows, for example, that the relay method requires the shortest tuning time but the setting that is obtained is the least suitable. The controller is disconnected during tuning. The momentum method leads to a controller setting that needs minimal control

actions, but the controller is also disconnected during tuning. Frequency-based autotuning needs more time for controller retuning than the other methods. Although the results of all tuning methods are parameters of optimal controller setting, the obtained settings differ and lead to responses of different quality.

The significant advantage of a controller equipped with the frequency-based autotuning method is that the controller is fully serviceable during tuning. In contrast to the momentum method and the relay method, when the operator cannot influence the controller setting obtained by these methods, the frequency-based method also allows the operator to change the criteria and their optimal values at any time if the response does not satisfy the expected course.

This work has been supported by Czech Science Foundation Grant No. GAČR 101/07/1667.

References:

- [1] Åström, K. J., Hägglund, T., *Advanced PID Control*. USA: ISA, Research Triangle Park, NC, 2006.
- [2] Åström, K. J., Hägglund, T., Hang, C. C., Ho, W. K., Automatic tuning and adaptation for PID controllers - a survey, *Control Engineering Practice*, Volume 1, Issue 4, August 1993, pp. 699-714
- [3] Åström, K. J., Hägglund T., The future of PID control, *Control Engineering Practice*, Volume 9, Issue 11. 2001. pp. 1163 – 1175
- [4] Emami, T., Watkins, J. M., A Unified Approach for Sensitivity Design of PID Controllers in the Frequency Domain, *WSEAS Transactions on Systems and Control*, Vol. 4, Issue 5, pp. 221-231, May 2009.
- [5] Emami, T., Watkins, J. M., Robust Performance Characterization of PID Controllers in the Frequency Domain, *WSEAS Transactions on Systems and Control*, Vol. 4, Issue 5, pp. 211-220, May 2009.
- [6] Hlava, J., Šulc, B., Advanced Modelling and Control using a Laboratory Plant with Hybrid Processes, *Proceedings of the 17th World Congress The International Federation of Automatic Control, Seoul, Korea, July 6-11, 2008*. Seoul: IFAC, 2008, p. 14636-14641
- [7] Huang, Sh., Lo, Y., Metal Chamber Temperature Control by Using Fuzzy PID Gain Auto-tuning Strategy, *WSEAS Transactions on Systems and Control*, Vol. 1, Issue 4, pp. 1-10, Jan. 2009.
- [8] Chang, H., Lin, Sh., Tang J., Su, Y., A Commercial Economy Efficiency PID Control System and the Implementation, *WSEAS Transactions on Business and Economics*, Vol. 2, Issue 4, pp. 186-191, Oct. 2005.
- [9] Chang, L., Chen, H., Tuning of Fractional PID Controllers Using Adaptive Genetic Algorithm for Active Magnetic Bearing System, *WSEAS Transactions on Systems*, Vol. 1, Issue 8, pp. 158-167, Jan. 2009.
- [10] O'Dwyer, A., *Handbook of PI and PID Controller Tuning Rules*. London: Imperial College Press, 2003
- [11] Schlegel, M., *A new approach to the proposal for robust industrial controllers* (in Czech: Nový přístup k robustnímu návrhu průmyslových regulátorů). Inaugural dissertation. Pilsen: University of West Bohemia, 2000
- [12] Schlegel, M., Exact Revision of the Ziegler-Nichols Frequency Response Method, *Proceedings of the IASTED International Conference on Control and Application, Cancun, Mexico, 2002*, Calgary: Acta Press, p. 121-126.
- [13] Šulc, B., Vrána, S., Plaček, V., Hrdlička, J., Lepold, M., Control for Ecological Improvement of Small Biomass Boilers, *Proceedings of the IFAC Symposium on Power Plants and Power Systems Control, Tampere, Finland, July 5-8, 2009* Tampere: IFAC, 2009.
- [14] Večerek, O., Schlegel, M., Design of H_{∞} PI Controller by Robustness Region Method, *Process Control 2004*, Pardubice: University of Pardubice, pp. R165-1-R165-9
- [15] Vološencu, C., Pseudo-Equivalence of Fuzzy PID Controllers, *WSEAS Transactions on Systems and Control*, Vol. 4, Issue 4, pp. 163-176, Apr. 2009.
- [16] Vrána, S., Čihák, J., Šulc B., Three Tank Cascade Improvement for Model-free Autotuning Testing, *XXXIV. Seminar ASR '2009 "Instruments and Control"*. Ostrava: VŠB - Technical University of Ostrava, 2009, pp. 349-356
- [17] Vrána, S., Šulc, B., Control Quality Indicators in PID Controller Autotuning, *The 5th International Conference on Computing, Communications and Control Technologies: CCCT 2007 Proceedings*. Volume II. Orlando: IIIS – International Institute of Informatics and Systemics, 2007, pp. 280-285
- [18] Wade, H. L., *Basic and Advanced Regulatory Control: System Design and Application*. 2nd ed. USA: ISA, 2004.