Teaching PID and Fuzzy Controllers with LabVIEW*

J. P. KELLER
Oensingen Institute of Technology, Switzerland. E-mail: juerg.keller@isoe.ch

This paper presents an educational concept for teaching PID and fuzzy controllers with LabVIEW. The simulation trainers for PID and fuzzy controller design are described. In addition, personal experiences with the trainer are summarised. Finally, programming considerations and cost estimates are presented.

INTRODUCTION

USING ATTRACTIVE simulations, students can efficiently acquire a lot of experience in controller tuning. Suitable tasks are in designing fuzzy controllers or tuning PID controllers in various control structures. Different types of plants, measurement noise or nonlinearities such as actuator saturation form a large field of simulation problems. Simulation is an inexpensive and fast way to practice many problems unimaginable in laboratory experiments. Simulation does not replace laboratory experiments. Analysis of plant structure and dynamics is better done with a real plant, but with the simulation experience and the already familiar control panel, students need only about $\frac{1}{4}$ of the time to complete an experiment. This also shows that simulation experience is transferable to real-life problems.

This contribution presents simulation trainers for PID and fuzzy-controller design. The PID-trainer is described and possible problems are formulated. The trainer for fuzzy controller design is documented, personal experience with the simulation trainer is summarised, and programming considerations and estimation of costs are given.

EDUCATIONAL CONCEPT

Simulation isn't a priori an efficient tool in education. Incorporating simulation into a demonstration is most likely a waste of time. The risk is that students only remember a fancy simulation, but the information, if even realised, is forgotten very quickly. This changes completely if students have to work with simulation, e.g. solve a challenging problem. The role of simulation is then twofold. First, it may contain an unwritten part of the problem formulation. For controller design, analysis of plant dynamics is an important step. This can be carried out by simulating the plant. Second, simulation is the student's tool to verify their solution. Without any risk of hazardous situations, students can explore the whole range of controllers.

The consequences are that simulation is not very suitable as an education tool for fundamentals and basic problems, but is best used in the training phase. In contrast to laboratory experiments, simulation is a cheap and fast way to experience a diverse field of control problems. The dimensions of our field are:

- *Plant parameters:* defining sets of plant parameters, several typical controller design problems can be formulated.
- *Noise:* disturbance rejection is one of the fundamental control tasks. Simulating the plant with different noise levels, controller performance can easily be investigated. Furthermore, including noise into simulation is a very important issue in bringing simulation closer to reality. In a simulation, sensor signals are generally at least 10 digits too accurate in comparison to real world signals. This leads to controller designs with unrealistic amplification of high frequencies using controllers with a large derivative part. As a consequence it is possible, for instance, to achieve similar, although unrealistic closed loop performance with a single PID controller as with a cascaded control structure. This is due to the fact that plant disturbances can easily be monitored at the 16-digit plant output signal.
- *Selection of control structure:* in a laboratory experiment, the control structure is usually predetermined by installed equipment. More degrees of freedom can be incorporated into a simulation. Since almost all possibly measurable variables are available in simulation, it is very easy to realise a simulation, where the task of selecting the measured variables is left to the student.

The educational risk of simulation exercises is that a solution is found by trail and error method. Either the exercises have to be so difficult that

* Accepted 9 September 1999.

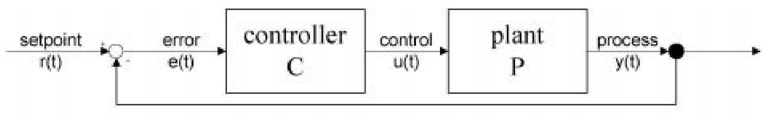


Fig. 1. Feedback control.

they can't be solved by trial and error, or a mean for quality assurance has to be used. The following tools can be recommended: a working sheet in which the student has to document and justify the design steps and an assessment sheet which focuses the student's work on the principal goals of the simulation exercise.

Finally, simulation can be replaced with a laboratory experiment. Due to the simulation experience and being familiar with the HMI (human-machine-interface), the rate of success in the laboratory work is almost 100% and completed in $\frac{1}{4}$ of the time without simulation training. But having several simulation exercises done, the students are very demanding, so they have to be convinced of the additional value of a laboratory experiment. The laboratory exercises have to be adapted to the changed conditions. For example, more emphasis can be put onto the problem of controller realisation on a PC.

LabVIEW Gsim Control and Simulation toolkit is not a well known simulation tool, but there are several reasons for using LabVIEW for simulation as well. If you have to make a simulation tool for somebody who is not interested in simulation techniques or modelling, who does not want to spend \$10,000 on a simulation tool or spend one week learning how to use the simulation environment, then you are well advised with LabVIEW. The only limits of an HMI may be the screen size of a notebook. A LabVIEW VI is easy to run and if each control and indicator has its description, online-help is available. The VI Info may be used for general explanations of the simulation problem. With the application builder, an exe-file can be created and distributed to the students. Since no additional software has to be installed, the simulation-exe can be copied to any location. There are no installation problems.

A LabVIEW programmer familiar with system theory can create LabVIEW simulations by studying some of the examples, i.e. the Gsim Tool is easy to use.

SIMULATOR FOR PID CONTROLLER TUNING

Since there is large variety of PID controllers, the implemented control law has to be documented. Afterwards the simulation tasks are described. They cover the following topics:

- Test of a PID controller.
- Tuning of a single PID controller for different plant models.
- Design of a feedforward controller.
- Tuning cascaded controllers.

The PID controller

A parallel configuration of the proportional, integral and derivative part is used. The derivative part is calculated with respect to a filtered error signal. The PID controller C in Fig. 1 basically implements the following control law:

$$u(t) = K \times \left(e(t) + 1/T_n \times \int (e(t)\,dt + T_V \times de_f(t)/dt \right) \quad (1)$$

$$\frac{de_f(t)}{dt} = \frac{1}{NT_V}(e(t) - e_f(t)) \quad (2)$$

with K = controller gain, T_n = reset time, T_v = raise time, N = time constant of derivative filter with respect to T_v, (although: derivative gain limit). The variables are defined in Fig. 1.

The controller output is limited to the range $[-100, 100]$. As anti-windup strategy, the controller implements the strategy proposed by Aström. For a complete study on anti-windup strategies,

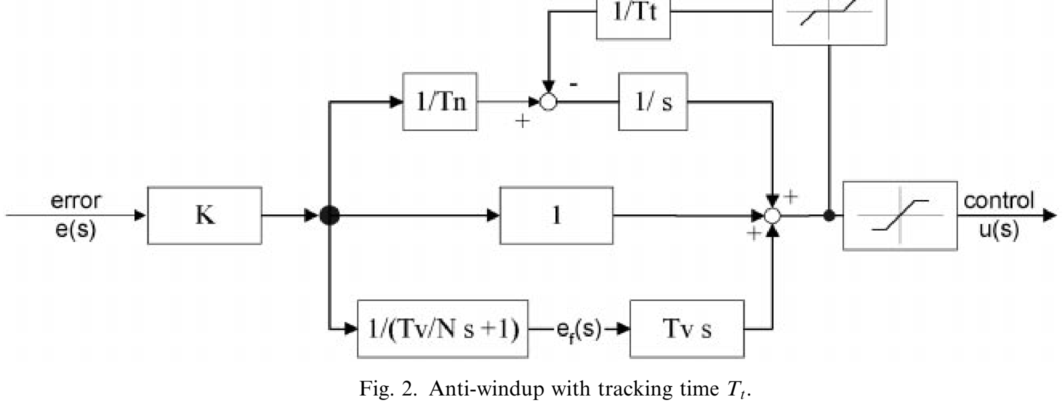


Fig. 2. Anti-windup with tracking time T_t.

see [1]. The amount the control signal exceeds the actuator limits modifies the integral part. Figure 2 shows a block diagram of the controller with anti-windup. With negative feedback and weighted with a constant called tracking time the difference between the limited and the unlimited controller output is added to the integrator input. For a constant controller error $e(t)$, the ratio between the rise time and the tracking time determines the limits of the integrator. This anti-windup strategy is insensitive to short time changes in the control error i.e. the integral part is not reset to an arbitrary value as with some other anti-windup methods. Since anti-windup problems are not the main topics of this simulation trainer, the tracking time is fixed as $Tn/3$. This gives a reasonable controller behaviour and the students do not have to be concerned with this value.

If the controller is operated in manual mode, the controller output can be set manually. This is useful to produce step responses of the plant. A bumpless switch to the auto-mode is possible.

PID test

Better than believing is testing the functionality of the implemented PID controller. Three signals are available to change the controller setpoint automatically. The process value can be changed manually to produce any desired error signal $e(t)$. This VI can be used as introduction to the controller trainer. The well known textbook step and ramp responses can easily be reproduced by simulation. The students get familiar with the controller panel and have confidence that the control law is calculated reliably. This becomes important when a reason for problems with controller tuning has to be found.

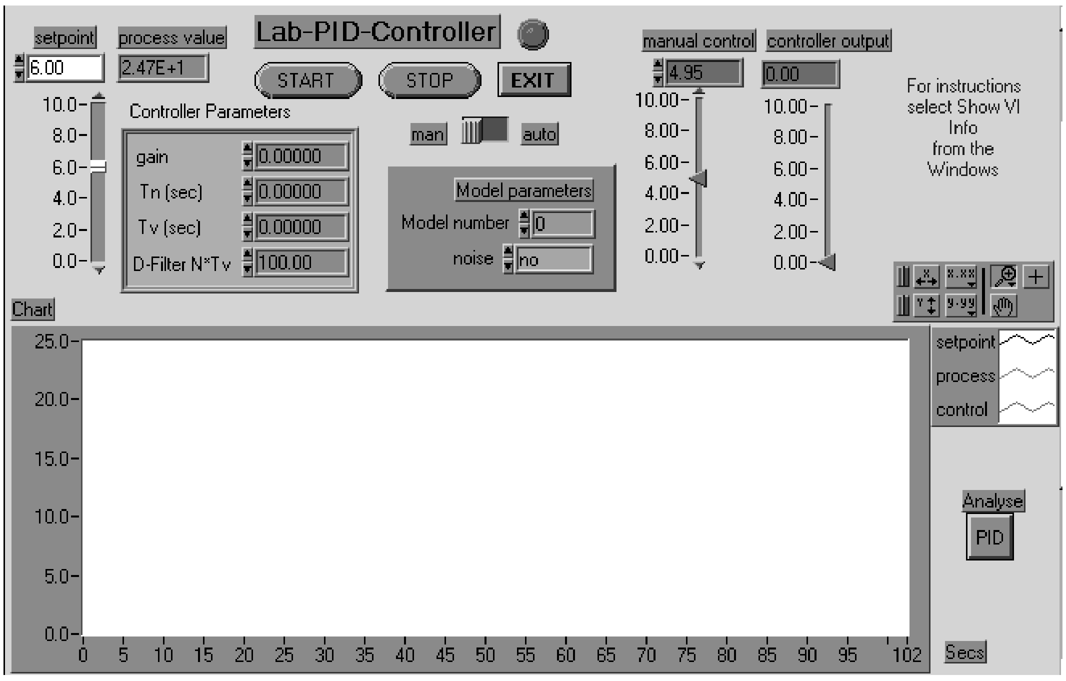
PID trainer

Tuning a PID controller is one of the most frequent controller design problems. It is therefore a compelling chapter in basic control education. Based on tuning rules, such as the well known tables of Ziegler/Nichols or Chien/Hrones and Reswick or more sophisticated schemes, a good initial guess for the controller parameters can be determined. Real-life control problems usually force an engineer to optimise the controller parameters. In my opinion there is a lack of documented methods for controller optimisation. By means of simulation, a student can acquire a lot of experience in optimising existing controllers.

A set of 6 plants is available. Random noise can be added at the plant output. The plant transfer functions are documented in Table 2 (Appendix). A brief qualitative description of the plants is the following:

- Plant 0 and 1: typical plants for step response controller tuning of stable plants.
- Plant 2: typical plant for step response controller tuning of an unstable plant.
- Plant 3: first-order plant, a method other than Ziegler/Nichols is required.
- Plant 4: third-order plant, unstable control system for tuning rules with small phase margin. Can be used to determine critical gain and frequency according to Ziegler/Nichols.
- Plant 5: second-order with damping factor 0.6.
- Noise: a random signal is added at the plant output. Two nonzero noise levels can be selected.

Step response controller tuning can be done with all the plants. In order to simulate a step response the PID controller is set to manual operation


Lab-PID-Controller

setpoint 6.00
process value 2.47E+1

Controller Parameters
gain 0.00000
Tn (sec) 0.00000
Tv (sec) 0.00000
D-Filter N*Tv 100.00

Model parameters
Model number 0
noise no

manual control 4.95
controller output 0.00

For instructions select Show VI Info from the Windows

Fig. 3. PID-control panel.

mode. The 'manual control' slider or the digital control can be used to change the controller output. The simulated response is plotted on a chart. Using the chart measurement tools of LabVIEW the step response parameters can be identified. Different control objectives lead to a wide palette of tuning problems. Minimising rise time, settling time or rise time with no overshoot are typical examples. Always let the students check disturbance attenuation. To avoid trial and error tuning, a worksheet as in the following example is recommended:

1. Analysis: draw a sketch of the step response and indicate important step response measures.
2. Select a controller and explain the selection. Determine the controller parameters for minimal rise time.
3. Draw a sketch of the closed-loop response. Explain how to change controller parameters to improve performance.
4. Give an estimate of the variance of the plant output when the noise level is set to large.

It is preferable to let the students draw the sketch manually than to produce prints of the simulation window.

The control panel of the PID Trainer is shown in Fig. 3. All controls and indicators have a description and the VI Info is used for a general explanation of the exercise.

Start the main panel and select the exercise 'PID Trainer'. You may select the real-time option to get a time feeling during simulation as well. Pressing the start button on the main panel, the PID Trainer becomes visible. The PID Trainer is started with the start button and stopped with the stop button. If a simulation is stopped, all internal variable shift registers are reset to their initial value. All controller parameter changes will also have effect during simulation. For a change of the model number to become effective, the simulation has to be stopped and restarted.

In order to analyse the controller behaviour, you can press the 'Analyse PID' button. The PID panel is opened and two charts are available for controller analysis. The upper chart shows the usual controller signals. In the second chart, the individual contributions of the P, I and D parts of the controller are shown. This chart gives a lot of insight into how a PID controller works. For instance you might use this tool to demonstrate the problem of steady-state control errors for controllers without integral action. This can be done with the following procedure. Put the controller in manual mode and set the control output to 20 or any other value. The process variable will reach some final value. Use this value as setpoint. For a given proportional gain, it is easy to calculate the steady-state control error of a P controller necessary to achieve the controller output of 20. This can be verified with simulation. Now, using a PI controller, you restart the simulation and verify that the integral part will 'learn' the value 20 of the controller output. You need to restart the simulation, because as you can see in the analysis chart, the integral part is set equal to the manual controller output for a bumpless switch to auto-mode.

Feedforward controllers

During the exercises with the PID Trainer, it becomes obvious that a controller design with no step response overshoot will have a reduced disturbance attenuation. In order to circumvent this problem, a feedforward controller is advisable. The feedback controller can be optimised with respect to disturbance attenuation and a low pass feedforward controller is used to achieve the acquired step response. Typically setpoint ramp or first-order filters are available on classical controllers. The parameterisation of such controllers can be practised with simulation. In a first step the feedback controller is optimised with respect to disturbance attenuation. In the second step, the parameters of the feedforward controller are chosen to minimise rise time, but to avoid overshoot. The simulation model is model 0 of the PID Trainer exercise.

The control panel is similar to all PID control panels. In addition, two types of feedforward controllers can be chosen: first-order filter (PT1) and setpoint ramp. With the control labelled 'time constant' both controllers are parameterised. For the first-order filter, the value is the time constant; for the ramp function it is the rise time for one setpoint unit.

Cascaded control

PID controllers are often applied in multi-loop structures. A frequently used candidate is the cascaded control structure. Tuning cascaded controllers can be easily explained in theory, but when faced with a real-life cascaded controller the procedure is not always obvious. This might be due to the fact that the controller interface is more complex, mainly in compact controllers with a small display. It is also possible that the controller implementation or the plant do not allow the tuning of the controllers sequentially.

Figure 4 shows a diagram of a cascaded control loop. One motivation to use cascaded control is to attenuate noise, labelled d_1, within the first subsystem P_1. This is possible and controller tuning is easy, if the dominant time constant of the first subsystem P_1 in Fig. 4 is much smaller than the time constant of the second subsystem P_2. If the dominant time constants are similar, the controllers interact strongly and controller tuning is not simple. Advantages of cascaded control structures become questionable. If the dominant time constant of P_1 is large compared to the time constant of P_2, cascaded control is useless.

These well known properties of cascaded control can also be explored with the simulator. Three different plant models are available:

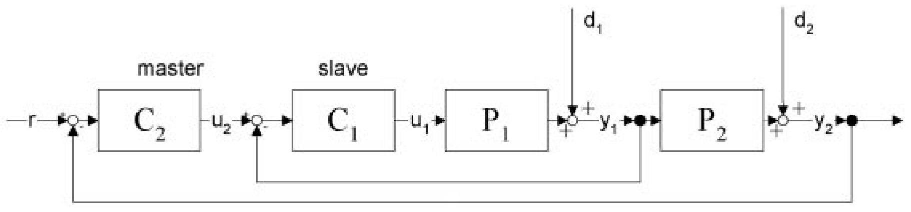


Fig. 4. Cascaded control structure.

fast-slow:
$P_1(z) = 0.4/(z - 0.8)$;
$P_2(z) = 0.05/(z - 0.99)^2$
(sample time: 0.1 seconds)
equal:
$P_1(z) = 0.02/(z - 0.99)$;
$P_2(z) = 0.05/(z - 0.99)^2$
slow-fast:
$P_1(z) = 0.002/(z - 0.999)$;
$P_2(z) = 0.05/(z - 0.99)^2$

Disturbance: d_1 is a small sine function added to a random signal and d_2 is a purely random signal. The amplitude of d_2 is only about 10% of the amplitude of d_1 in order to make the attenuation of d_1 evident at the plant output. Unless there is a disturbance d_2, good control can also be achieved by a single PID controller.

Since a cascaded control structure consists of two controllers, it is not easy to design a clearly arranged control panel. It is reasonable that both controllers have the same controls and indicators as the single loop controller. Moreover, time charts showing the controller behaviour are mandatory. As a consequence the control panel is slightly overloaded. Figure 5 shows the relations of the controls and indicators to the signals in the control structure diagram.

Tuning the cascade starts with identification of the inner loop plant P_1. Select the 'Slave only' operation mode, set the slave controller to manual operation mode and simulate a step response of plant P_1. The 'manual control' slider or the digital control can be used to change the controller output. The simulated response $y_1(t)$ is plotted in a chart. Using the chart measurement tools of LabVIEW the step response parameters can be identified. Select controller type (P, PI, PD or PID) and determine the controller parameters. Put the slave controller in 'auto' mode and check the performance of the inner loop (slave controller). In the 'slave-only' mode, the setpoint of the slave controller can be entered at the 'setpoint slave' control.

Which plant is controlled by the master controller? The students must be able to answer this question to be able to determine the controller parameters. The master controller's plant is the inner control loop in series with the second part of the physical plant. Therefore, the plant output

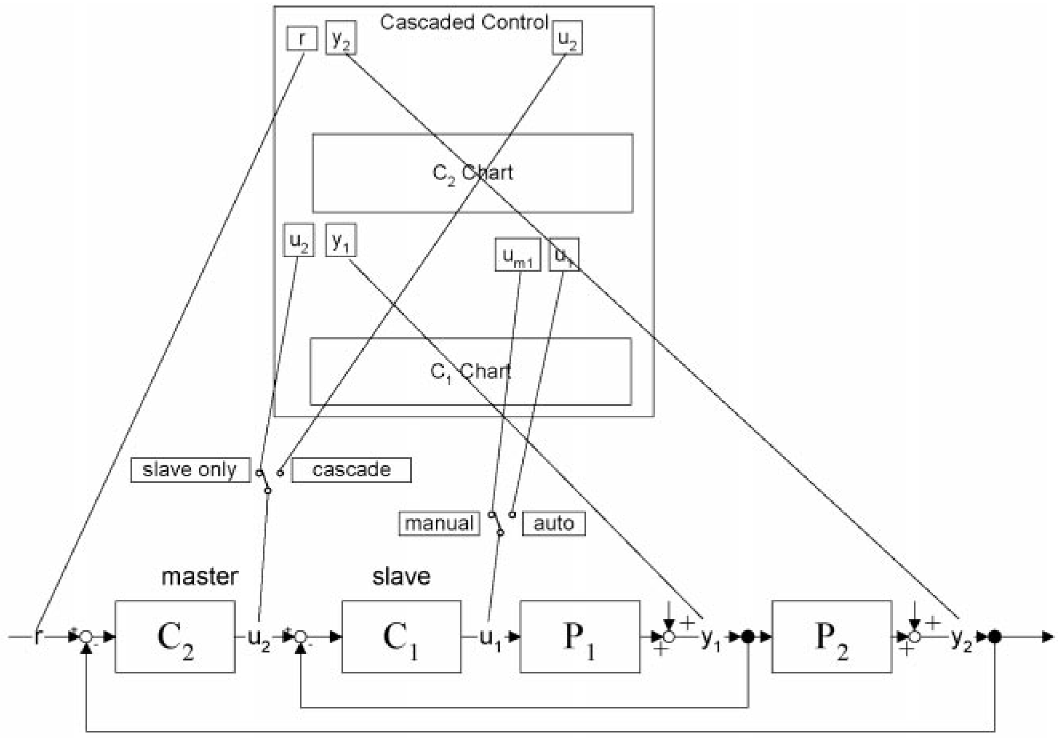


Fig. 5. Signals on the panel of the cascaded controller.

response $y_2(t)$ to a setpoint change at the inner control loop must be produced. This can be done in the same configuration as before, where performance of the slave controller was checked. The signal of interest is now $y_2(t)$ and its step response parameters can be measured in the master controller's chart. Configure the master controller, change the operation mode to 'cascade' and test the controller performance.

This tuning procedure can be done for all plant configurations, i.e. for fast-slow, equal and slow-fast. The students might experience difficulties with the equal and slow-fast plant configurations and hopefully will remember suitable plant structures for cascaded control. It is clear that controller performance can not be compared with respect to the different plants. But for each plant, it is interesting to compare the cascaded control structure with a single-loop PID controller. This can be done by setting the operation mode to 'single PID'. The differences become evident when noise is added to the simulation.

FUZZY CONTROLLER DESIGN

The fuzzy controller

The fuzzy simulator is based on the fuzzy controller of the LabVIEW Fuzzy Logic Toolkit for G. A fuzzy membership function editor allows the user to quantitatively define linguistic terms for input variables. A rule-base editor is used to define rules for the controller output based on the linguistic terms defined. The Fuzzy Logic Toolkit for G is used to implement rule-based feedback controllers. A fuzzy controller VI is used in the application VI to process input data based on the fuzzy controller designed. The toolkit is well suited for control applications on nonlinear or complex systems that are difficult to model mathematically but may be controlled by human operators.

House temperature control

Many students have some experience with house temperature control. There is no need to explain why and how to heat a house. With house temperature simulation, two problems can be practised. The first is to select physically sensible controller inputs from the set of directly measured or derived variables. The following variables can be chosen:

- indoor temperature, its time derivative and integral;
- outdoor temperature and its time derivative;
- control error, its time derivative and integral;
- indoor minus outdoor temperature and its time derivative.

It is clear that not all the available variables are meaningful, only a few are well suited as fuzzy controller inputs.

The second problem is how to systematically compensate disturbances. The outdoor temperature changes with time. The heating power necessary to compensate the heat loss can be determined experimentally. Ideally the heat loss is represented as a function of the indoor to outdoor temperature difference. This knowledge can be incorporated into the fuzzy controller design. With suitable membership functions and rules the input/output characteristics of the fuzzy controller can be tuned to compensate the heat loss.

The simulation model of the house is the following:

$$P_{el} = k(T - T_o) + c\frac{dT}{dt} \tag{3}$$

$$\tau\frac{dT_i}{dt} = T - T_i \tag{4}$$

with: P_{el} = heating power range: $0\ldots 10,000\,\mathrm{W}$; k = heat loss coefficient = $400\,\mathrm{W}/^\circ$; c = heat capacity, $330{,}000\,\mathrm{J}/^\circ$; T_o = outdoor temperature; T_i = measured indoor temperature; τ = sensor time constant, 3 min.

The temperature setpoint is reduced during the night. Setpoint during the day is 20°C, at night time 15°C. The outdoor temperature is a sine function. The simulation creates the setpoint and the outdoor temperature automatically.

Controller performance: the simulation VI determines a measure for controller performance. It is a weighted sum of the squared control error and a measure for the control effort integrated over the simulation time range of 3 days. The control effort is measured with the variance of the 8 last control values. This aims to punish fast changing, though unrealistic control signals.

The worksheet of the simulation is as follows, in analysis:

- Disturbance compensation: a) What is the major disturbance of the system? b) Find a function (equation or graph) and its arguments to determine the amount of heating power required to compensate the disturbance.
- Chose sensible fuzzy controller inputs and justify each selection with sound physical arguments. The controller function, as far as is possible, must be independent of the temperature setpoint.

In design and optimisation:

- Design a fuzzy controller for the temperature control problem. The fuzzy controller must implement the disturbance compensation function of 1b. Verify it with the I/O characteristic tool.
- Minimise the performance measure!

The assessment sheet in Table 1 is given to the students with the exercise. It should meet the following goals: minimise trial and error approaches and avoid frustration after having spent several hours on the exercise.

Table 1. Assessment of the fuzzy simulation exercise

Topic	Criteria
Disturbance compensation	I/O characteristics: sensible function for disturbance compensation (meets function in 1b)
Choice of input variables	physically sound justification controller independent of setpoint value distinctness of arguments
Linguistic variables	ranges naming of terms sensible number of terms membership function sensible
rules	appropriate
controller performance	

Race track

The race track simulation is suitable for a fuzzy controller design competition. Students are highly motivated to take part in such a competition. The simulation time is given and the car with the best fuzzy controllers will run the longest distance.

- *Sensors:* the car speed is available. Furthermore the racing car is equipped with three sensors, one looking to the right, one straight ahead and the other to the left. All three measure the distance to the edge of the track. When the car exits the track, the sensor value will most likely be -1. To adapt the sensors to the control strategy, the angle φ between the sensor directions can be configured.
- *Controls:* the controls of the racing car is the car acceleration and the steering angle. The car is equipped with anti-sliding control, i.e. the maximum steering angle is a function of the car speed. The range of the steering angle is ± 80 degrees. The car acceleration is limited to ± 20 acceleration units. The car speed is modelled as a first-order system.

Two fuzzy controllers are to be designed, the first for steering control, the second for speed control. As in the preceding simulator, a large set of input variables are available. These are:

- distances: d_1, d_2, d_3, $d_3 - d_1$
- distance derivatives: $\partial d_2/\partial t$, $\partial(d_3 - d_1)/\partial t$
- car speed

In Fig. 5, the control panel is shown. In an x/y plot the race track and the moving car are plotted. Steering angle, acceleration and car speed are shown in the time plot on the left for analysis purposes. The example plot in Figure 6 clearly shows the limiting of the steering angle. The maximal steering angle only increases when the car speed is reduced as can be seen in the middle of analysis plot. This forces the speed controller to reduce car speed to be able to turn on curves.

Often, the simulation is too fast to allow analysis of the controller behaviour. For this purpose, a time zoom control is available. Its value is the waiting time between two simulation steps.

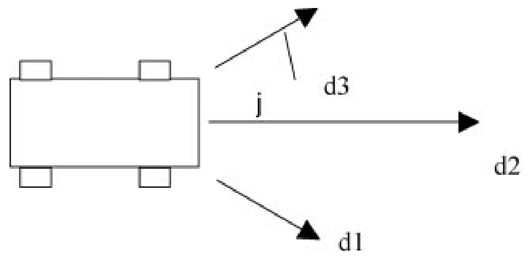


Fig. 6. Measuring directions of the car sensors.

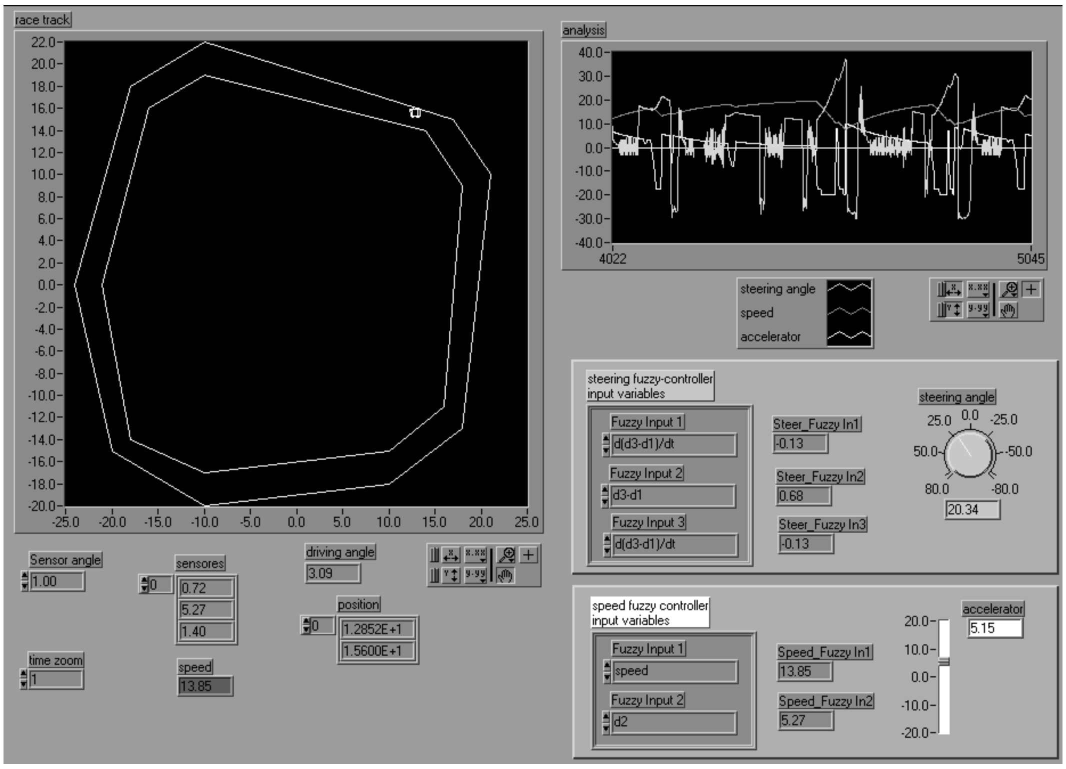
EXPERIENCE WITH THE SIMULATION TRAINER

The PID trainer has been a part of basic control education for 2 years. The students' feedback is good, but they complained about the time they have spent on simulation. The laboratory experiments showed that it was time well spent, because almost all of the students were able to solve the laboratory control problem with good results. Without simulation training, the students needed more than 8 hours to complete the experiment. Familiarity with the control panel and having the expertise of controller tuning, most students finished the experiment in less than two hours. As an engineer, they will tackle a control problem without any hesitation.

The PID trainer is also well accepted and used by colleges at the same and at other engineering schools. Due to the online help, there is no need for large documentation. New versions can be produced easily.

The content of the control theory course was changed this year and the course now starts with fuzzy control as the first topic. A training tool that does not need any knowledge of system theory was necessary. It was obvious that a LabVIEW fuzzy simulator meets these requirements. After an introduction to fuzzy control and to the LabVIEW Fuzzy Controller Design tool, the students designed a fuzzy controller for the tank example of the fuzzy toolkit. The race track competition was launched and one week later, the results were presented. It was astonishing that after 24 lessons of control theory, including an introduction to fuzzy control, the students presented good solutions for this control problem. An experienced control engineer has to spend at least two hours to produce a competitive solution. With classical PID control it is hardly imaginable to solve this multivariable, nonlinear control problem with only 24 lessons of control theory. In the laboratory, all the students were able to design a fuzzy controller for a laboratory experiment. The VIs were available and the students only had to design a fuzzy controller.

Since a lot of time was spent on simulation, the exam should test the same skills. In the exam the students had to design a fuzzy controller for the house temperature control problem. The feedback on an exam with simulation was not always good, because the students claimed that the result



Sensor angle 1.00
sensores 0: 0.72, 5.27, 1.40
driving angle 3.09
position 0: 1.2852E+1, 1.5600E+1
time zoom 1
speed 13.85

steering fuzzy-controller input variables
Fuzzy Input 1 d(d3-d1)/dt
Fuzzy Input 2 d3-d1
Fuzzy Input 3 d(d3-d1)/dt
Steer_Fuzzy In1 -0.13
Steer_Fuzzy In2 0.68
Steer_Fuzzy In3 -0.13
steering angle 20.34

speed fuzzy controller input variables
Fuzzy Input 1 speed
Fuzzy Input 2 d2
Speed_Fuzzy In1 13.85
Speed_Fuzzy In2 5.27
accelerator 5.15

Fig. 7. Front panel of the race track fuzzy simulator.

is strongly influenced by luck. But the assessment sheet revealed that non-optimal controller performance is strongly correlated to the assessment criteria of the fuzzy controller. Obscure names of variables and terms, strange membership functions were always paired with non-optimal controller performance.

The benefits of simulation training and the use of LabVIEW control panels became evident in the laboratory experiments and in this examination. Before using the simulation trainer, the laboratory fuzzy control experiments had never led to satisfactory results.

PROGRAMMING CONSIDERATIONS

Modern simulation tools try to hide the numerical simulation problem, i.e. the integration algorithms. This is usually coupled with an increasing need for computational performance. Since a simulation trainer is usually not based on sophisticated physical models, the continuous-time model can easily be discretised and implemented as difference equations. This leads to lean and easy computable models.

Simulations may produce results so fast that it is not possible to follow a 'live' simulation. Analysing a simulation after completion has some advantages, but needs more skills to reproduce situations. If it is possible to follow a live simulation, the interaction with the system is immediate and problems with a control law are easily detected. For this reason, all simulation trainers are capable of slowing down the simulation using waiting time between the simulation steps. This possibility to zoom simulation time can also be used to produce real-time simulation. In real-time simulation, the results are produced with the respect to the real time-scale, comparable to the physical plant.

The online-help capabilities of LabVIEW are very convenient tools. For each control and indicator a short description can be provided for online-help and for documentation of the VI.

A major problem of the first versions of the simulation trainer was the initialisation of variables. The simulator had the capability to stop and resume the simulation. The students rarely used this feature. When resuming a simulation, they were often irritated because of the almost unpredictable values of the state variables. As a consequence the trainer was simplified. Stopping the simulation resets all internal variables to their initial values. At the same time the problems with internal variables with value NaN were also solved. It is not easy to ensure that all variables never have a value NaN. If the variables are not checked for NaN, only closing and restarting a VI can reset the variables to useful values. This problem is solved if all variables are reset to initial values before the simulation starts. This is accomplished with the VI structure given in Fig. 8.

All sub-VIs with internal variables have the main loop counter as input signal. Using a case structure as shown in Fig. 6, the initial values can be set.

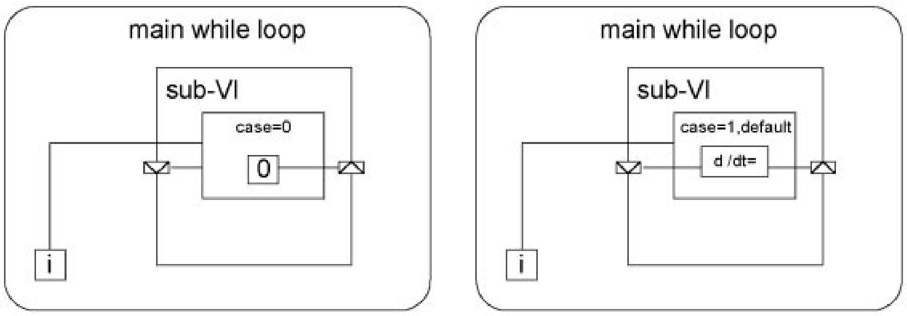


Fig. 8. VI structure for initialisation of variables.

The PID trainer now accumulates approximately three weeks of LabVIEW programming. The maintenance costs are negligible and there are no problems in distributing an exe-file. In zipped form it is still less than 1.4 Mb and can be distributed on a floppy disc. The workload for realising the race track simulation was underestimated. Moving a car and efficiently calculating the sensor signals were the main problems. The total work time is estimated at two weeks of programming. The house temperature control problem was realised in one day. Because both fuzzy applications are quite large, they were not combined on one fuzzy trainer.

Compared to C++ programming, a lot of time was saved with LabVIEW because of convenient possibilities for creating controls panel. Even more important were the very useful debugging tools.

CONCLUSIONS

Simulation is an efficient and inexpensive tool in control education. Tuning PID and fuzzy controllers are well suited problems for simulation training. After a simulation training the students are capable of solving a laboratory control problem about four times faster than without simulation training. Simulation does not replace laboratory experiments but it offers the opportunity of changing the goals of the laboratory experiments from simple controller design problems to the control of real-life plants. Results of examinations show clearly that with simulation, the students have a significantly larger expertise in controller design. These achievements are mainly due to the fact that simulation is well accepted by students. They spend more time with a fancy simulation than studying from a textbook. The instructor's challenge is to assure that the interpretation of simulation experiments are based on theoretical understanding and physical insight. Worksheets and assessment criteria are helpful tools.

LabVIEW is a powerful environment for developing simulation trainers. Besides the mathematical functions it offers a lot of possibilities for easily creating control panels. Online-Help and simulator documentation can be included without special tools. Powerful debugging tools allow the fast development and testing of simulation software.

REFERENCES

1. M. V. Kothare, P. J. Campo, M. Morari and C. N. Nett, A unified framework for the study of anti-windup designs, *Automatica*, **30**, 12 (December 1994) pp. 1869–1883.

APPENDIX

Table 2. Plant models of PID Trainer.

#	Transfer function in z, sampling time 0.1s
0	num0 = [4e-4 5e-4 5e-4]; den0 = [1 −1.94 0.9405];
1	num1 = [0.121 0.0117]; den1 = [1 −1.903 0.9048];
2	num2 = [9.900E-3 9.900E-3]; den2 = [1 −1.98 0.98];
3	num3 = 1; den3 = [1 −0.995];
4	num4 = [1e-4 1e-4 1e-4 1e-4]; den4 = [1 −3.6 4.86 −2.916 0.6561];
5	num5 = [4.9e-3 4.8e-3]; den5 = [1 −1.942 0.951];

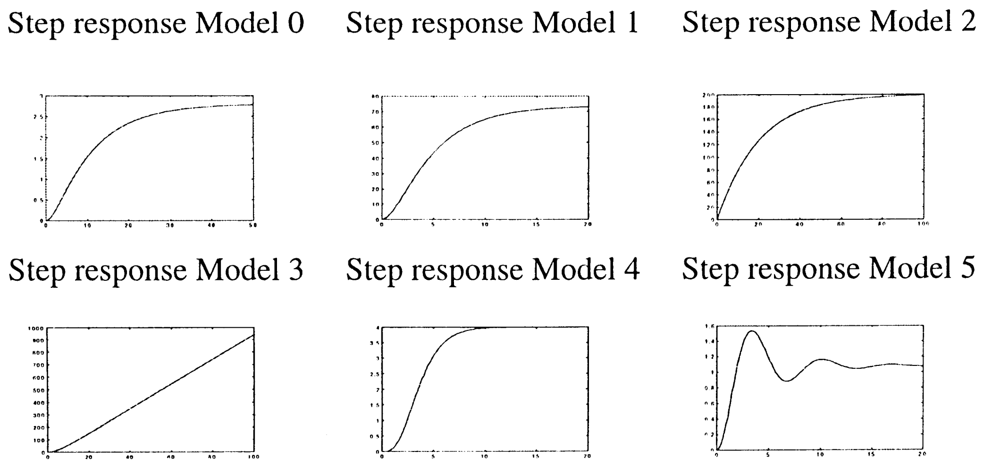


Fig. 9. Step responses of the PID trainer.

Jürg Keller was born in Switzerland in 1958. He received the diploma in Chemical Engineering in 1983 and the Ph.D. degree in 1989, both from the ETH Zürich. After 5 years of practical control experience with Hoffmann-La Roche, Basel, he is now Professor in Control Engineering at the Institute of Automation, Oensingen, Switzerland. His areas of interest are 'filling the gap between theory and application', simulation and sampled data systems.