Laboratories and Real-Time Computing

Årzén, Karl-Erik; Blomdell, Anders; Wittenmark, Björn

Published in:
Control Systems Magazine

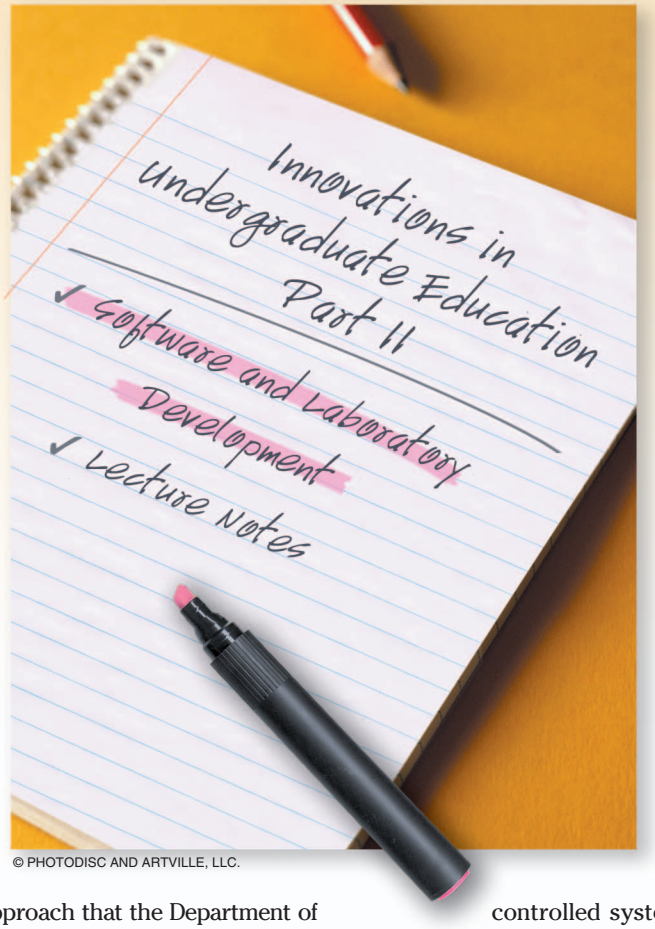
DOI:
10.1109/MCS.2005.1388797

2005

FEATURE

Laboratories and Real-Time Computing

Integrating experiments into control courses



© PHOTODISC AND ARTVILLE, LLC.

By Karl-Erik Årzén, Anders Blomdell, and Björn Wittenmark

Laboratory experiments using real, rather than simulated, processes are of vital importance in control education. Experiments help the students understand the theoretical material and provide important motivation. However, laboratory equipment is costly to develop, maintain, and operate, and laboratory space can be expensive.

This article describes the approach that the Department of Automatic Control at Lund Institute of Technology uses to maintain a high level of practical laboratory experiments. Engineering studies in the Swedish education system consist of a 4.5 year program leading to a master of science degree. No separation is made between undergraduate and graduate levels. The engineering program is divided according to subjects, for example, electrical engineering, mechanical engineering, and chemical engineering. The departments, which are orthogonal to the study programs, are smaller in size and give courses within several of the study programs. The Department of Automatic Control is responsible for all control education, offering courses within most of the programs offered by the Lund Institute of Technology.

For all of these programs a basic control course is mandatory, except for chemical and biotechnical engineering where control is an elective. In addition to the basic courses, we have elective courses in computer-controlled systems, real-time systems, system identification, nonlinear control, and adaptive control. The department also gives two smaller project courses in control, one of them in cooperation with Ecole des Mines de Nantes in France. Every year around 900 students take our courses and, hence, pass through our laboratories. More information about the course content can be found in [1].

Laboratory Experiments

Laboratory experiments are integrated into all of our courses

0272-1708/05/$20.00©2005IEEE

[2], [3]. Each course typically has three mandatory four-hour laboratory exercises. The laboratory experiments in our large courses are based on two fundamental principles: mobile desktop processes and the use of standard computing equipment. Students normally work in teams of two persons, each team outfitted with a Linux PC equipped with an input/output (I/O) interface and a laboratory process. The processes are small enough to fit on the table on the side of the computer, hence, the name desktop processes. Some of the processes are in-house designs, for instance, the double-tank process, while others are commercially available laboratory processes from different vendors. An overview of some of our processes can be found in [4].

Our largest laboratory hall can accommodate 24 students working in teams of two (Figure 1). Two teaching assistants, normally a Ph.D. student and a final year student, supervise each laboratory session. During peak periods, the laboratory halls are in use 14 hours per day, five days per week, with three four-hour laboratory sessions per day. The desktop laboratory processes make it easy to change between different laboratory experiments. The processes are designed so that they need only electrical power, and thus no special facilities are needed. This feature and the high utilization help keep costs down.

Computer Environment

The laboratory computer environment is based on off-the-shelf PCs running Linux 2.4.18 with the UTIME extension from KURT [5]. UTIME allows an application to sleep with microsecond resolution, but in practice, more than 2 kHz sampling is hard to achieve on a standard PC. However, this rate suffices for most of the experiments. The advantage of this approach is that all Linux/POSIX timing primitives have access to the improved timer resolution, making it possible to run all programs with better timing resolution. Programs such as Java and Scilab/SciCos can directly benefit from the increased timing resolution, while Simulink requires a specific S-function block to synchronize simulated time with wall-clock time, making it necessary to run all blocks at multiples of a fixed sampling frequency.

The main part of the software used in the laboratory exercises is based either on stand-alone Java applications or MATLAB/Simulink programs. The laboratory programs for our double-tank laboratory experiments are written entirely in Java. The user interface is shown in Figure 2. Ordinary Linux and Java do not provide an ideal real-time control platform due to the nondeterminism caused by the automatic memory management in Java. However, because of the high speed of current PCs, the solution works well for almost all of our laboratory processes. It is only for our ABB industrial robots and for a DVD-player process, sampled at 40 kHz, that we must use special-purpose hardware.

The laboratory experiments based on MATLAB/Simulink use the high resolution timer described previously. The controllers are implemented in Simulink, which is run in real time with special A/D and D/A blocks that connect to the I/O on the PC. Hence, code generation tools such as Real-Time Workshop are not needed.

Real-time systems are of vital importance to control engineers, since control systems are real-time systems.

I/O is handled by the COMEDI device interface [6], which allows us to choose from a large assortment of I/O cards without modifying the programs. Currently, we use Analog Devices RTI-815, ComputerBoards DAS-1600, IOTech DAQBoard/2000, Quanser MultiQ-3, Advantec PCI-1711, and home-built serial I/O. I/O support is provided in the form of Java classes, Scilab/SciCos blocks, and Simulink blocks.

An Atmel AVR-based remote I/O module was developed to simplify lecture demonstrations of processes with timing requirements of greater than 10 ms sample interval. The remote I/O box is connected by means of a serial cable

Figure 1. *Laboratory hall. Our largest laboratory hall, with 12 PCs and 12 tank processes, is used for three daily four-hour laboratories involving 24 students.*

to a PC or by a USB port to a serial port converter for laptops without a serial port. After the successful introduction of this I/O during 2003, some of our processes were equipped with their own built-in serial I/O, which makes it possible to interface them with only a serial cable.

Real-Time Systems

Real-time systems are of vital importance to control engineers, since control systems are real-time systems. It is therefore essential for control engineers to have a thorough understanding of computers and real-time systems. It is also important for computer engineers to understand control theory. The Department of Automatic Control has pioneered the teaching of real-time programming and real-time systems at the Lund Institute of Technology.

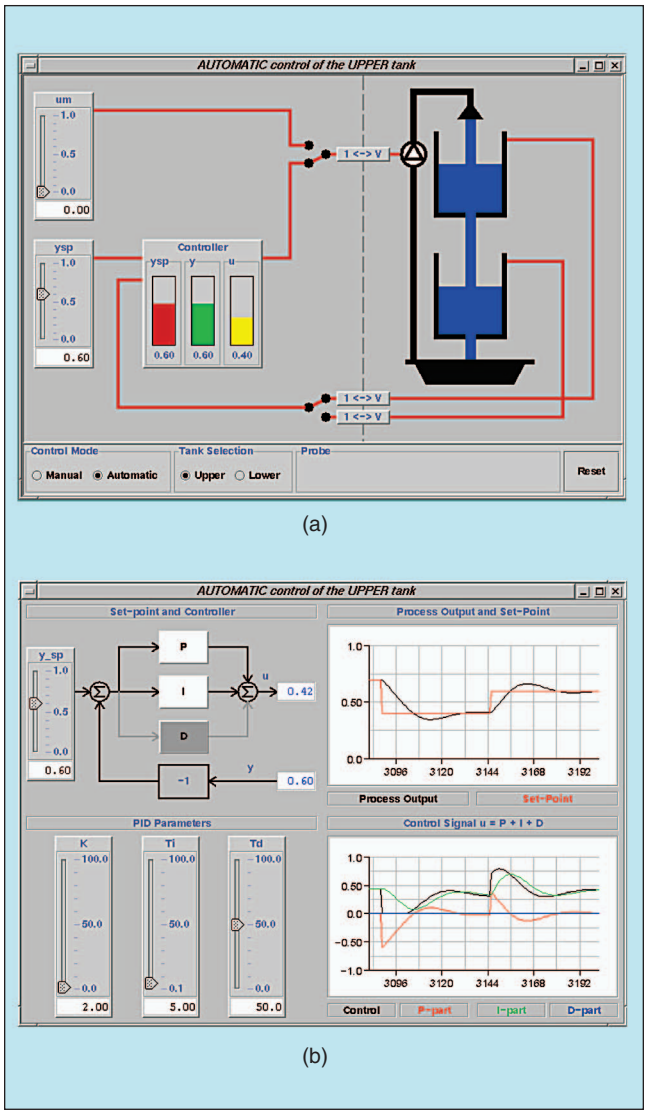


Figure 2. *Java GUI. The software for the double-tank laboratory experiment is developed in Java. This laboratory experiment is part of the basic control course.*

The aim of the real-time systems course is to study methods for designing and implementing computer control systems, with a focus on embedded systems as well as on industrial control systems of the programmable logic controller (PLC) and distributed control system (DCS) type. The course consists of three parts: real-time programming, computer implementation of control systems, and industrial control systems. Most of the time is devoted to the first two parts. The course gives the students sufficient knowledge to implement small embedded control systems on their own and to have a understanding of the system aspects of large industrial control systems.

The real-time systems course encompasses one semester spanning 14 weeks. The first half of the course consists of 17, 90-minute, lectures; computer exercises; and two laboratory exercises. During the computer exercises, the students learn concurrent and real-time programming with Java as the programming language. The students implement Java-based controllers, including graphical user interfaces (GUIs), and the controllers are applied to virtual processes. The virtual processes are real-time simulation models that include an animated user interface. One example of a virtual process is the virtual ball and beam process. During the first laboratory exercise, the task is to apply the Java-based controller developed for the virtual ball and beam process to a physical ball and beam process. This process normally goes smoothly, and the students learn that simulation is a useful tool for control system implementation. However, the students also learn that a simulation model never exactly matches reality.

In one of the computer exercises, the students use the TrueTime toolbox [7] to study how task scheduling affects controller performance. TrueTime makes it possible to simulate the temporal behavior of multitasking real-time kernels and real-time communication networks in Simulink. Tasks representing controllers are implemented as MATLAB m-files or in C. In the exercise, the students can see how scheduling strategies and scheduling parameters affect the timing of the controller tasks and how that in turn influences control performance.

The topic of the second laboratory experiment is sequential discrete-event control. We use JGrafchart, which is an in-house developed environment for graphical programming, and execution of sequential function charts (SFC) [8]. The task is to implement a sequential control program to sort and sequence different colored beads in a small bead-sorter process.

Projects

During the second half of the course, a project is assigned in conjunction with the lectures. These projects are performed by teams of four students who choose from 30–40 different projects. The projects, which comprise approximately two weeks of work, are divided into three groups,

real-time control projects, real-time programming projects, and embedded control projects.

The task of the real-time control projects is to implement a controller for one of the laboratory processes. These processes include various pendulum models, helicopters, more complex ball and beam setups, flexible servos, fluid tanks, and ABB industrial robots. Standard Java is used for controller implementation. In some projects, the focus is on networked control applying, for example, CORBA, Java-remote method invocation (RMI), or Bluetooth. The type of controller used depends on the background of the students. Students who have taken the course in computer-controlled systems can design the controller using sampled-data control theory, whereas students who have taken only the basic course design a continuous-time controller, which is then discretized. Students who have taken the course in adaptive control, or who follow the two courses in parallel, often implement an adaptive controller.

Laboratory experiments using real, rather than simulated, processes are of vital importance in control education.

The real-time programming projects focus on real-time programming issues. Past projects include implementation of a kernel supporting earliest deadline first (EDF) scheduling, implementation of the priority ceiling protocol, evaluation of RT-Linux, and application of rate-monotonic scheduling theory. The motivation for having these projects is to attract students with primary interest in control as well as students with primary interest in real-time computing.

The third type of project focuses on embedded control applications. Here, the implementation platform is a small Atmel AVR Mega8 processor. This processor has limited memory consisting of 1 kB RAM and 8 kB flash memory. The processor does not support floating point calculations, and the limited memory does not allow software emulation of floating point calculations. Hence, the students have to implement their controllers using C and fixed-point calculations. The processor has no multitasking kernel, supports timers, counters, and interrupts and is equipped with digital and analog input and output. The analog output is implemented using pulse-width modulation (PWM). Serial communication (RS 232) is used between the AVR and a PC, where a GUI is implemented. The task in the embedded projects is the same as in the real-time control projects, that is, to implement a controller for one of the laboratory processes.

The real-time systems course attracts around 110 students every year. The project part is especially appreciated, and many students are quite ambitious. During 2003, four students chose to build a complete mini-Segway process, including mechanical design, construction, and electrical circuit board fabrication, Atmel microprocessor programming, and control design. The students succeeded, although they spent considerably more time on the project than required. The process, shown in Figure 3, includes two motors, wheel encoders, a rate gyro, and an accelerometer. A Kalman filter is used to extract the angle and the angular velocity from the gyro and accelerometer signals. In parallel with the 2003 course, a project-based Ph.D. course in embedded control was given. In one of the projects, a Lego car was equipped with an anti-locking brake system implemented on Atmel AVR.

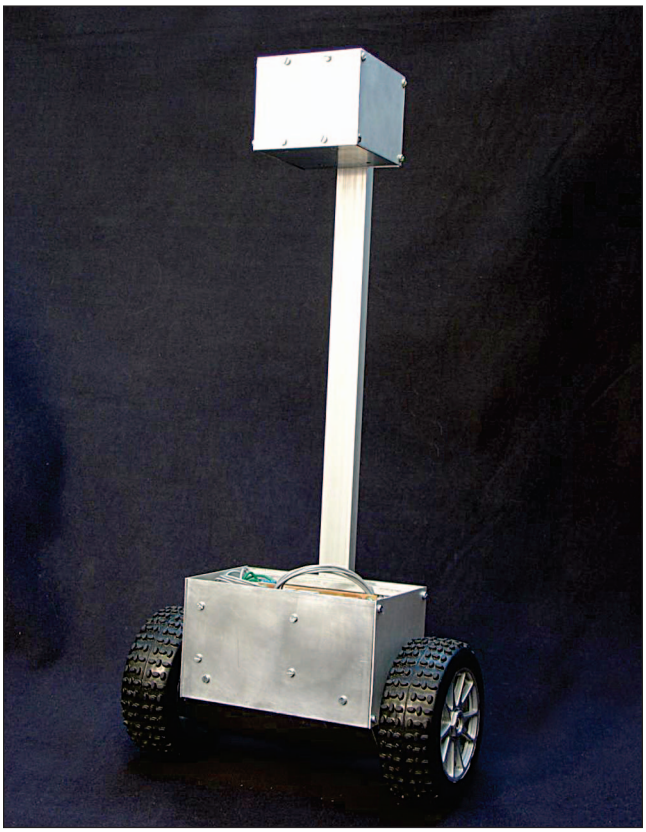


Figure 3. *Mini-Segway process. This process was developed as a project in the real-time systems course in 2003 by four students. The project included mechanical design, construction, electrical circuit board fabrication, Atmel microprocessor programming, and control design.*

Conclusions

There are strong pedagogical advantages to integrating laboratory experiments into control courses. If the laboratory exercises are properly organized and the student volume is sufficiently large, it is possible to combine a high level of practical laboratory experiments in control education at reasonable cost. The use of off-the-shelf hardware and open-source software are important. With desktop processes, it is possible to achieve high utilization of lab space and high student throughput. Technical staff within the department can administer the computing system and develop and maintain the laboratory processes.

With the current focus on embedded systems, sensor networks, and networked control, real-time systems technology is a natural component of the control curriculum. At Lund, the computer engineering students have the same courses in mathematics, signals, systems, and control as do the electrical engineering students. The focus on real-time systems in our group has contributed to real-time control and real-time systems becoming one of our major research areas.

References

[1] Department of Automatic Control home page [Online]. Available: http://www.control.lth.se/

[2] K.J. Åström and A.-B. Östberg. "A teaching laboratory for process control," *IEEE Control Syst. Mag.*, vol. 6, pp. 37–42, Oct. 1986.

[3] K.J. Åström and M. Lundh. "Lund control program combines theory with hands-on experience." *IEEE Control Syst. Mag.*, vol. 12, no. 3, pp. 22–30, June 1992.

[4] B. Wittenmark, "Laboratory processes used by the Department of Automatic Control" [Online]. Available: http://www.control.lth.se/education/processes/

[5] D. Niehaus, "Kansas University Realtime" [Online]. Available: http://www.ittc.ku.edu/ kurt/

[6] COMEDI [Online]. Available: http://www.comedi.org/

[7] A. Cervin, D. Henriksson, B. Lincoln, J. Eker, and K.-E. Årzén. "How does control timing affect performance?" *IEEE Control Syst. Mag.*, vol. 23, no. 3, pp. 16–30, June 2003.

[8] K.-E. Årzén, R. Olsson, and J. Åkesson. "Grafchart for procedural operator support tasks," in *Proc. 15th IFAC World Congress, Barcelona, Spain*, pp. 1530–1535, July 2002.

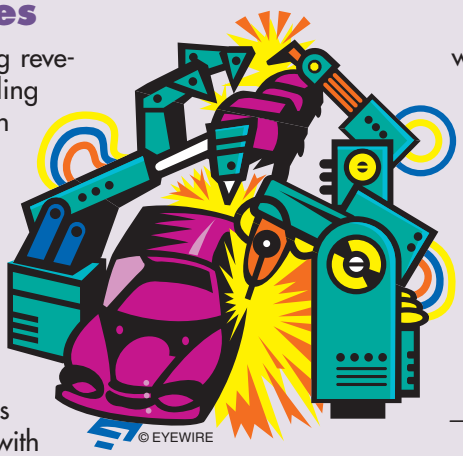
Karl-Erik Årzén (karlerik@control.lth.se) received a Ph.D. in automatic control from the Lund Institute of Technology, Sweden, in 1987. He has been a professor at the Department of Automatic Control since 2000. His research interests are real-time systems, real-time control, and programming languages for control applications. He was the chair of the IEEE Control System Society Technical Committee on Real-Time Control, Computing, and Signal Processing from 1999 to 2002. He can be contacted at the Department of Automatic Control, Lund Institue of Technology, Box 118, SE-221 00 Lund Sweden.

Ånders Blomdell has been a research engineer at the Department of Automatic Control since 1988. His main responsibilities are system administration, real-time computers, and networking.

Björn Wittenmark obtained the M.Sc. degree in electrical engineering in 1966 and the Ph.D. degree in automatic control in 1973, both from Lund Institute of Technology. Since 1989, he has been a full professor in the Department of Automatic Control at Lund Institute of Technology, Lund, Sweden, and has been vice rector of Lund University since 2003. His main research interests are in the fields of adaptive control, sampled-data control, and process control. He has written numerous papers and is the coauthor or editor of eight books.

Impotent Machines

This is surely the most astonishing revelation that a book like this dealing with mechanical possibilities can put before the layman: that there are mechanisms which perceive the external world and reason about their perceptions as premises and adjust their reactions to the outcome of their syllogisms. The only limit to machines in conquering our world of intellect is that they only solve the problems with which they have been constructed to deal. If their designer wishes them to deal with a hundred problems they will do so, but if a hundred and one problems are presented, they will reveal themselves as stupid and impotent to deal with the one extra. But after all, does not man also find himself disarmed and stupid in the face of certain problems?



— *Quoted from P. De Latil,* Thinking by Machine, *The Riverside Press, Cambridge, MA, 1957.*