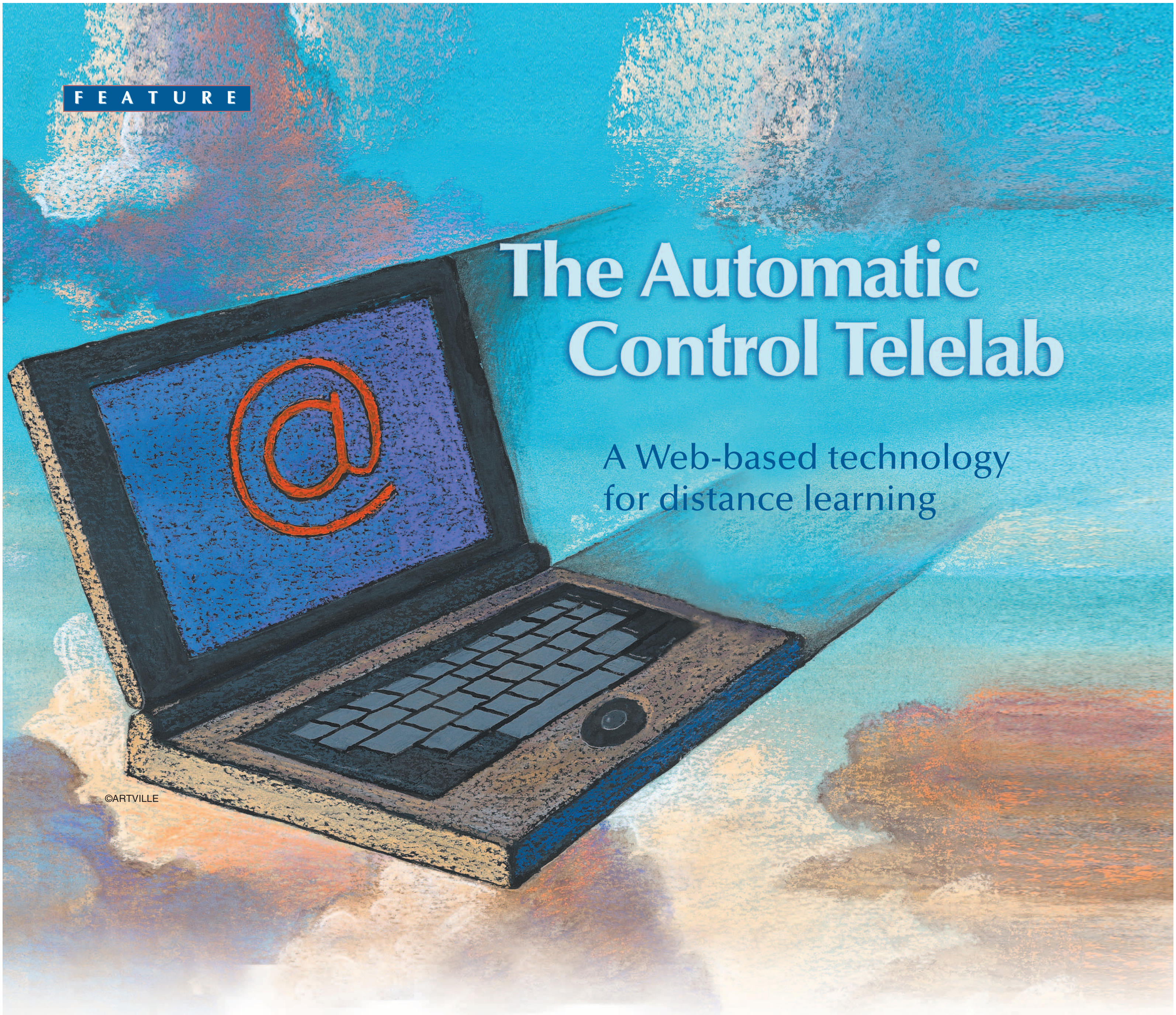
FEATURE

The Automatic Control Telelab

A Web-based technology for distance learning



©ARTVILLE

By Marco Casini, Domenico Prattichizzo, and Antonio Vicino

Control systems education is currently exploiting the advantages of Internet and Web technologies to develop distance learning paradigms. Remote user interaction with online experiments is one of the hottest topics today in distributed computing, Web applications, and distance learning. The results of a recent survey about how information technology is being applied to control education are reported in [1]. Applications range from games to telelaboratories, telemedicine, network management, and telecommerce.

For a survey on Web technologies used in control systems courses, the reader is referred to [2], where the authors describe the use of virtual [3]–[5] and remote labs. A distinguishing feature of remote labs as compared to virtual labs is that users can interact with real physical processes through the Internet, making them more attractive than controlling software simulations. At the same time, the design and implementation of a remote lab is more challenging due to safety and fault tolerant aspects.

In a telelab, remote operators can run the experiment, change control parameters, observe results, and download data through a Web interface, as in [6], where a remote lab for testing analog circuits is described. A remote chemical control process is implemented in [7], while several laboratory experiments, such as voltage, pressure, and temperature control, are available in [8].

The complexity of the hardware and software architec-

0272-1708/04/$20.00©2004IEEE

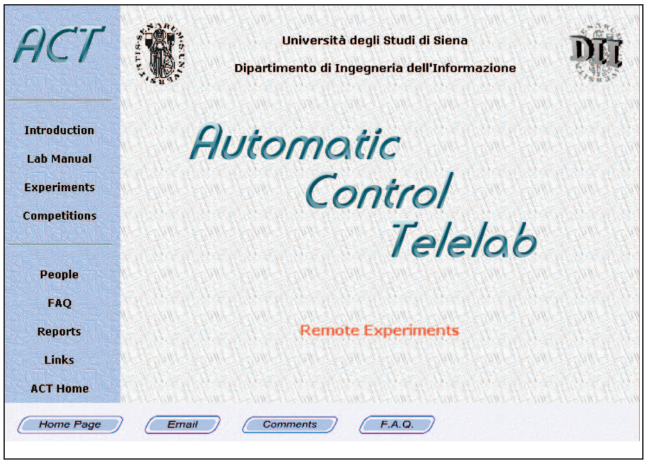


Figure 1. *The ACT home page. In addition to the set of experiments available for remote control, information such as a user guide can be accessed. All of the Web pages are in English.*

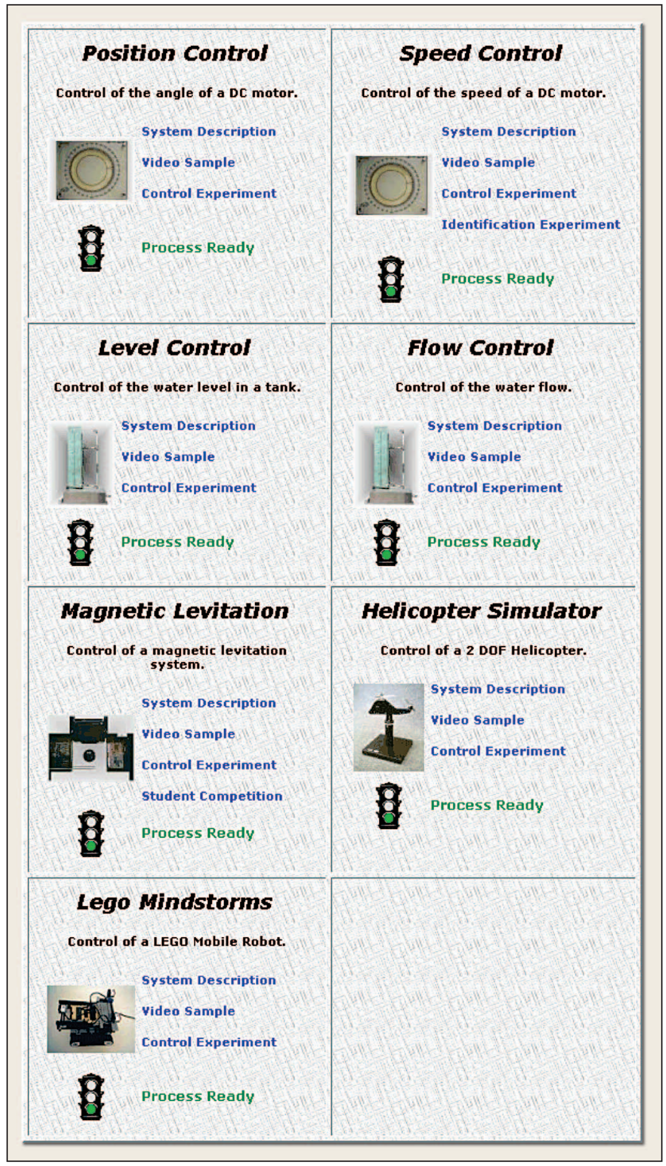


Figure 2. *ACT's online experiments. Five processes are available for remote experiments: a dc motor, a water tank, a magnetic levitation system, a 2-DOF helicopter simulator, and a LEGO mobile robot.*

ture design increases when the remote lab allows the user to design the controller within the remote Web session, as seen in the remote lab developed at the College of Engineering at Oregon State University [9], [10]. In this implementation students can remotely control a robot arm by changing parameters and, more interestingly, by transmitting the control program that changes the dynamics of the closed-loop system.

A two-link, direct-drive robot arm is available at the University of Illinois at Urbana-Champaign, where special pan-tilt controllable cameras are used to enhance the atmosphere of telepresence [11]. Students can perform lab assignments involving Lagrangian dynamics, proportional-integral-derivative (PID) control, and computed torque control of robots. At the University of Pisa [12], an anthropomorphic robot arm is remotely programmable at a higher level than the inner joint control. The remote user can choose the trajectory to be followed by the end-effector and can learn features of high-level robot programming languages. The Mercury project [13] is an early example of robotic distance programming on the Web, dating back to 1995.

A different approach has been analyzed in [14], where the user can run experiments using a controller that resides on the client machine. Network reliability issues and delays are also addressed.

From a technological and pedagogical point of view, remote labs that allow the user to define controllers, rather than choose from a predefined set of controllers, are more stimulating. The software architecture design for this type of facility is more difficult. The software environment used to synthesize controllers is the challenging aspect for the overall project.

Generally speaking, a remote laboratory can use a well-known software environment, such as LabVIEW [15], MATLAB/Simulink [16], [17], special purpose software [18], or a line command syntax [12]. We believe that well-known software environments are better suited for increasing laboratory usage since students do not want to learn control languages that are tailored for a particular remote lab.

This article describes a remote laboratory, the Automatic Control Telelab (ACT), being designed and built at the University of Siena (http://www.dii.unisi.it). The MATLAB/Simulink environment has been chosen to implement ACT. ACT allows the user to choose a predefined controller or synthesize a new controller through the MATLAB/Simulink environment.

A feature of the project is a simple user interface, which requires knowledge of Simulink for designing the controller to

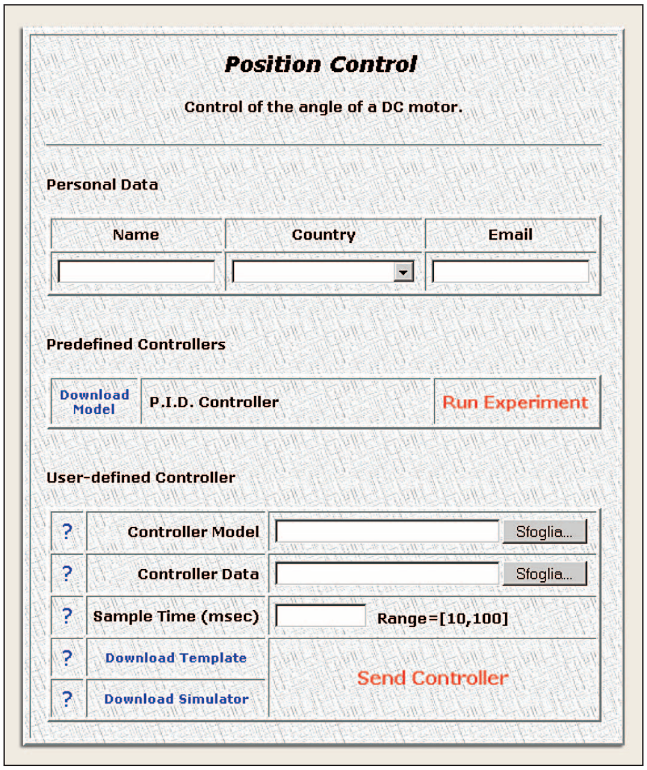


Figure 3. *The control type interface. A user can choose a predefined or custom controller for use during the experiment.*

be tested through ACT. During the experiment, it is possible to change controller parameters and the reference signal. Experimental results can be checked through online plots and a live video window showing the running experiment.

Features of ACT

The extension of teaching capabilities through the Internet is the core philosophy of the ACT project, whose home page is shown in Figure 1. Since 1999, ACT has been used in control systems courses at the University of Siena [19]–[21]. The goal of the project is to enable students to apply their theoretical knowledge of control theory without restrictions due to laboratory hours and experiment avail-

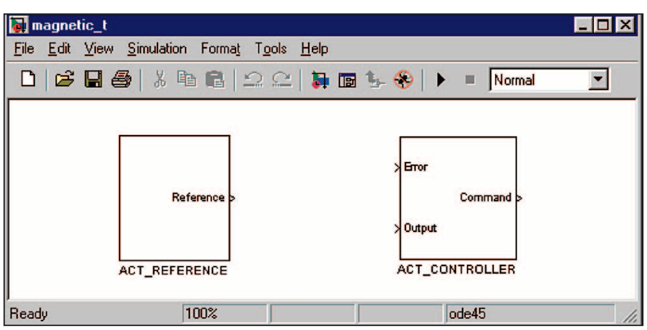


Figure 4. *The Simulink template model. This model helps the user design a controller and choose the reference signals to be applied during the experiment.*

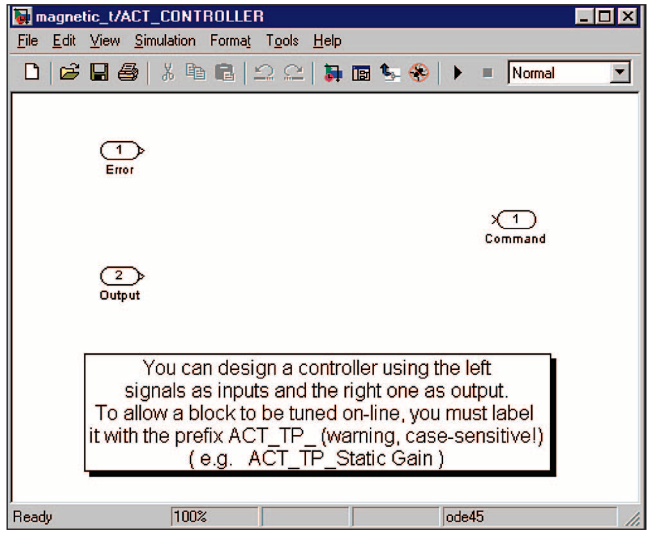


Figure 5. *The ACT_Controller Simulink subsystem. Designing a controller involves connecting the error, output, and command signals with suitable blocks.*

ability. ACT is accessible 24 hours a day from any computer connected to the Internet using browsers such as Netscape Navigator or Microsoft Internet Explorer. MATLAB/Simulink software is required for the user to design her/his own controller. Accessibility through the Internet makes ACT an enabling tool for students with disabilities.

The ACT remote lab is continually upgraded with new software versions and experiments. In the present stage, five processes are available for online experiments: a dc motor, a tank, a magnetic levitation system, a two-degree-of-freedom (2-DOF) helicopter, and a mobile robot (Figure 2). The dc motor involves control of the shaft angle or rotation speed. Through the tank process a user can perform level or flow experiments. In spite of its simplicity, the dynamics of the tank are nonlinear. The magnetic levitation process, which is nonlinear and unstable, has properties that can be analyzed at the undergraduate level, whereas the 2-DOF helicopter, which is a nonlinear, unstable multi-input, multi-output (MIMO) system, can be used in graduate level courses. In addition, the LEGO robot can be used for experiments in mobile robotics.

ACT Features

- *Easy-to-use interface.* Simplicity is essential for realizing an interface that is easy to use. ACT is based on intuitive and simple HTML pages and Java applets, which are fully supported by the latest versions of browsers. Help pages are provided for detailed information.
- *Simulink-based interface for controller design.* The Simulink-based interface is used to design controllers that drive the real process. Only a basic knowledge of the MATLAB/Simulink environment is required. At

the end of the experiment the user can download a file in the MATLAB workspace format .mat, where all data of the experiment is stored for off-line analysis.

- *Predefined and user-defined controller types.* Every experiment of the remote lab can be controlled using either predefined or user-defined controllers. In the first case, a student chooses a control law from a given list and then assigns the values of typical parameters. For example, a student can select a PID controller to run the experiment and choose the values of proportional, integral, and derivative coefficients. Alternatively, by using the Simulink graphical interface, the user can design her/his own controller to drive the experiment and then send the controller to the ACT server. A Simulink user-defined template is available to help the remote user in this phase.
- *Predefined and user-defined reference signals.* The remote user can choose reference signals from a given list or create new reference signals by building a Simulink subsystem. It is possible to change the reference signal while an experiment is running. Thus, the user does not have to start a new experiment to verify the response of the system to different input signals.
- *Controller parameter change.* While an experiment is running, ACT provides a mechanism that allows the user to change controller parameters online, such as the coefficients of the PID controller. Working over the Internet, parameters are updated when the packets reach the ACT server. The resulting time lags depend on the distance, the type of Internet connection, and network congestion. These delays do not adversely affect the process since the control law resides on a local PC connected to the process (see the "ACT Architecture" section). The only consequence of these time lags is a delay between the user parameter change request and the execution of the command. Tunable parameters can also be included in the user-defined controller by naming the parameter variable according to a special and simple syntax, as described in the next section.
- *Lab presence.* For effective distance learning, it is important for the user to have a sense of presence in the laboratory. A live video and online data plots are thus provided, making it possible for students to view the real process while the experiment is in progress.
- *Resource management.* As with other remote labs, the experiment hardware is controllable by one user at a time. To prevent process monopolization, a fixed amount of time (5–10 minutes) is assigned to each experiment session. After that time the user is automatically disconnected, and the process is available for the next user. The Web page provides a list of available experiments (Figure 2) indicating which processes are idle as well as the maximum delay time for the busy experiments.

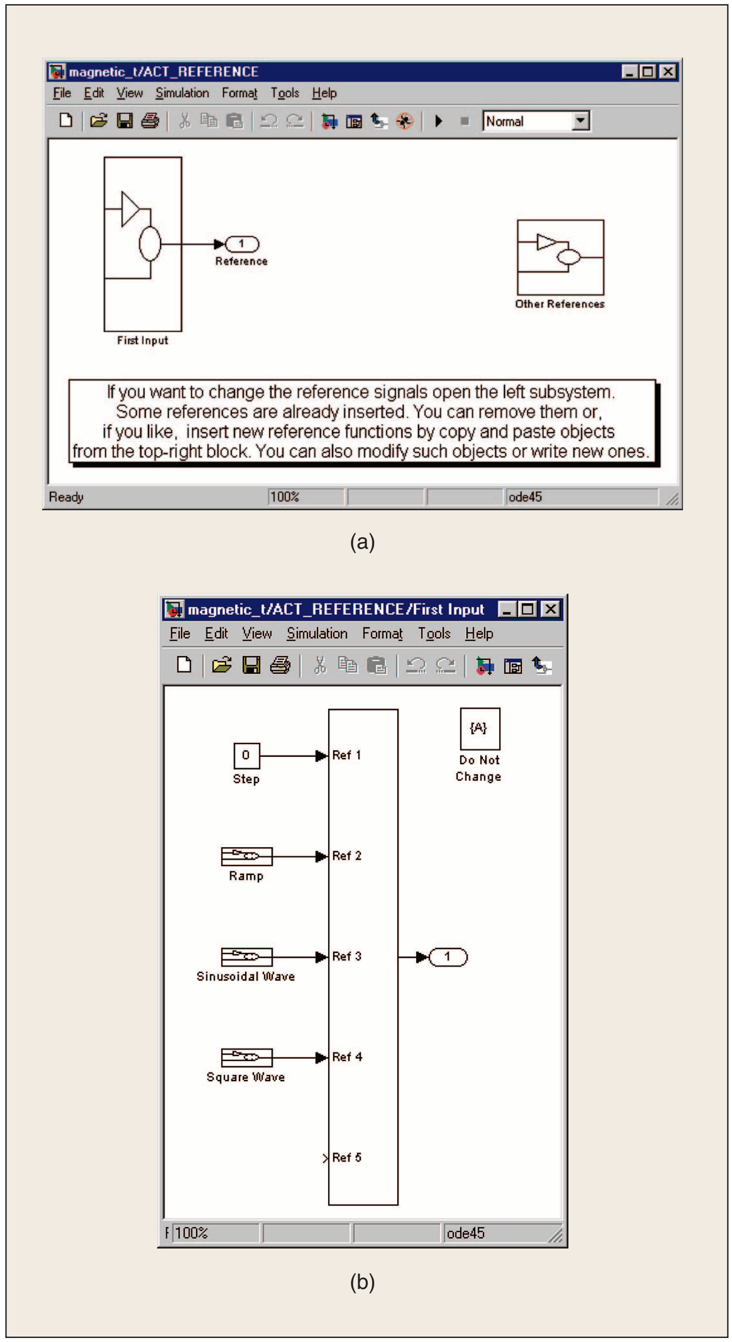


Figure 6. *(a) ACT_reference subsystem. (b) Detail of the first input block. Through these blocks the user can include new reference signals for use during the experiment.*

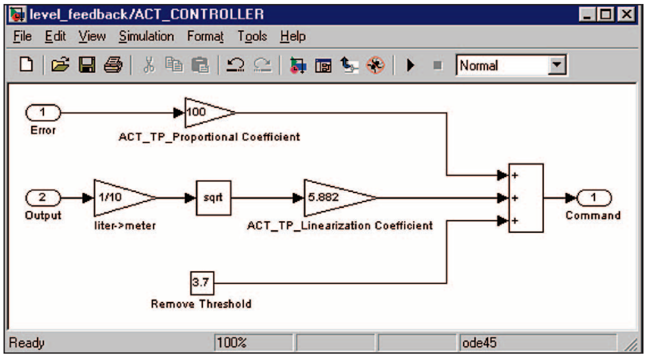


Figure 7. *A controller Simulink model. This model represents a controller for the tank level process with tuning parameters based on feedback linearization.*

- *System safety*. Regarding system safety, hardware and software actuator saturation is enforced to prevent users from performing dangerous operations. Similarly, saturation is enforced on input reference. Moreover, an instability detection system has been implemented in software to stop the experiment when the system becomes unstable.
- *Simplicity of adding new processes.* The software and hardware architectures of ACT have been designed to simplify the connection of new processes to the remote laboratory. Concerning software, only a Simulink model and a text file must be created to add a new process to ACT.

ACT provides a sophisticated yet easy-to-use structure to control a remote process through the Internet.

A Session Description

From the home page of ACT, it is possible to access general information pages, such as the laboratory user guide, and the list of available experiments. After choosing which experiment to run, the control type interface appears as shown in Figure 3. Through this interface, the user completes a personal-data form (used to gather statistics about users) and chooses a controller for implementation.

User-Defined Controller

To simplify the controller design, the user can download a Simulink template model that contains two subsystems, one for the controller (ACT_Controller), and one for the reference input (ACT_Reference); see Figure 4. The control error, output, and command signals are available in the "ACT_Controller" subsystem, as shown in Figure 5. The task for the user consists of joining the signals by means of suitable blocks that define the controller structure. Such blocks can be provided by any of the available Simulink toolboxes. Moreover, it is possible to set "constant" and "gain" blocks as variable parameters that can be modified on-line while the experiment is in progress. This feature is obtained by using the prefix "ACT_TP_" (ACT tuning parameter) to name the variables described in the bottom window of the Simulink template in Figure 5. The "ACT_Reference" subsystem of the template file in Figure 6(a) is used to build new reference signals for implementation during the experiment. A set of reference signals is available by default, such as constant and ramp signals or sinusoidal and square waves. The user can remove some of these blocks or add new ones; see Figure 6(b). To help the user in this task, reference blocks are provided inside the "Other References" subsystem. For advanced users, it is possible to design special reference input signals in the Simulink environment.

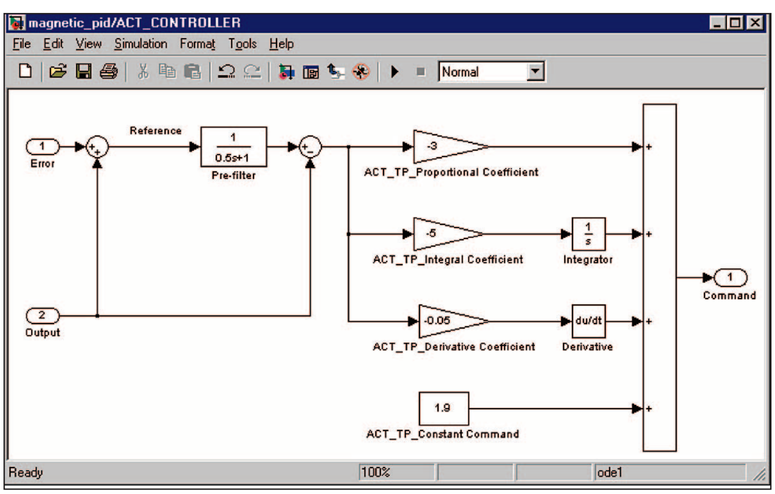


Figure 8. *A controller for the magnetic levitation system. The controller is a PID with a prefilter on the reference signal. All PID coefficients are tunable online.*

The Tank Level and Magnetic Levitation Examples

The mathematical model of the tank shown in Figure 2 is

$$\dot{h}(t) = -0.008\sqrt{h(t)} + 100\,q(t), \qquad (1)$$

with

$$q(t) = \begin{cases} 0 & \text{if } V(t) \leq 3.7 \\ 1.36 \times 10^{-5}\,(V(t) - 3.7) & \text{if } V(t) > 3.7, \end{cases}$$

where h is the water level inside the tank as measured by a pressure transducer on the bottom of the tank, q is the input flow, and V is the voltage command applied to the pump. Due to a threshold on the pump actuator, the input flow is zero when the voltage applied is less than 3.7 V. Moreover, an input saturation of 8 V is present. Since the dynamics of the tank process are nonlinear, a potential controller based on *feedback linearization* is used to cancel the nonlinear part of (1). Problems due to the threshold can be avoided by applying a constant voltage of 3.7 V to the pump. The Simulink model implementing the feedback linearization controller is shown in Figure 7. The model has been obtained from the "ACT_Controller" template in Figure 5 by linking the error, output, and command nodes through Simulink block functions. Two parameters have been set to be tuned on line. The *proportional coefficient* is the proportional gain on the system error, while the *linearization coefficient* is used to cancel, or at least reduce, the effect of the nonlinearity. Since the model described in (1) is an approximation of the true plant, online tuning of these parameters is mandatory for optimizing performance.

The levitation process consists of a magnetic suspension, a ball whose vertical position is to be controlled, and an electromagnetic coil. The position of the ball is sensed by an optical sensor. The minimum and maximum distance of the ball from the coil is 0.5 and 2.5 cm, respectively. The power amplifier supplies the coil with current that is proportional to the command voltage V_u of the actuator. A protection circuit sets the current to zero when it reaches 3 A. Letting z be the height of the ball of mass m, the system dynamics are modeled by

$$\ddot{z}(t) = g - \frac{F_m}{m},$$

with the magnetic force $F_m = k_{ma}(V_u^2/z^2)$ and system constant k_{ma}. The PID is a predefined controller with a prefilter on the reference; the Simulink model is shown in Figure 8. The coefficients of the proportional, integral, and derivative actions can be tuned while the experiment is running.

Running the Experiments

Once the user-defined controller has been built, the controller model is uploaded to the ACT server through the *send controller* button; of course, this operation is not needed for predefined controllers. If the Simulink model does not contain syntax errors, the "Experiment Interface" is displayed, as shown in Figure 9, whereby it is possible to run the remote experiment through the "start experiment" button. When the

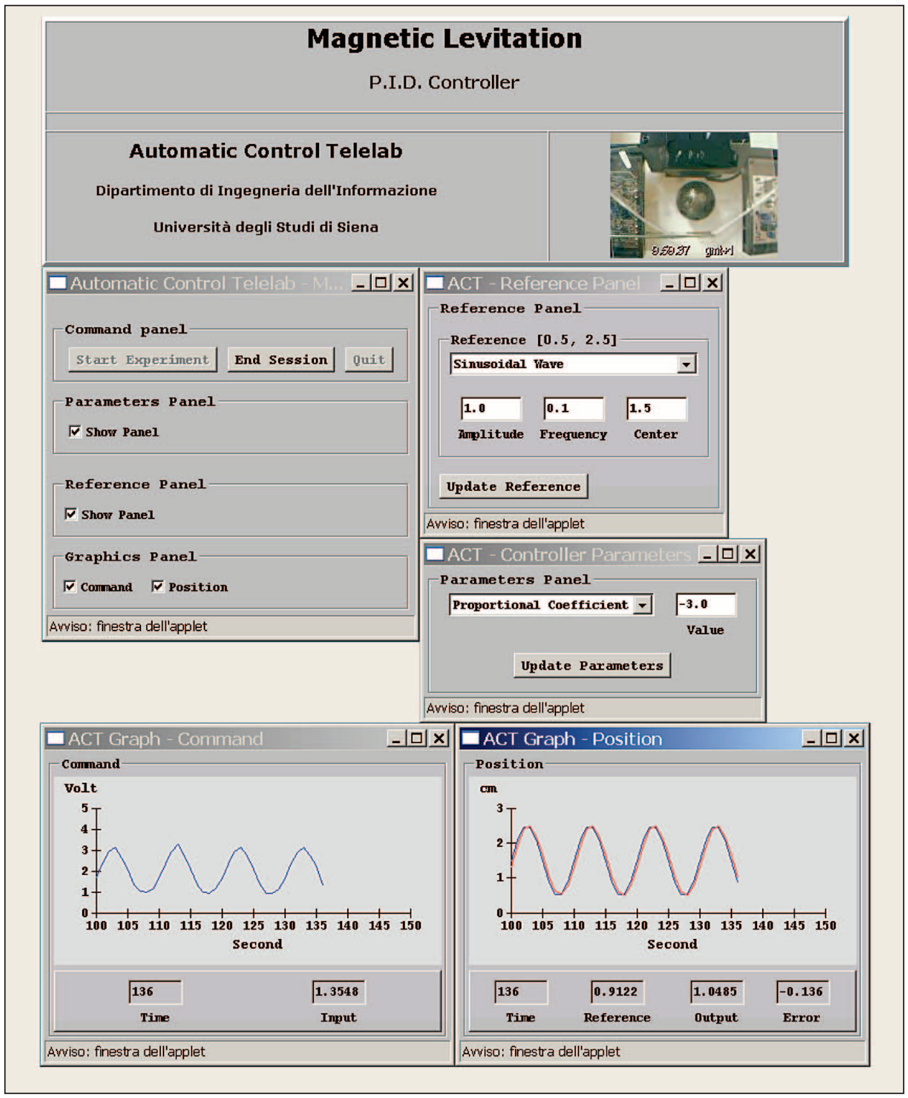


Figure 9. *The experiment interface. An experiment involving the magnetic levitation system is shown. With this interface, it is possible to open windows that allow one to change controller parameters and reference signals, as well as view data plots and an online video of the experiment.*

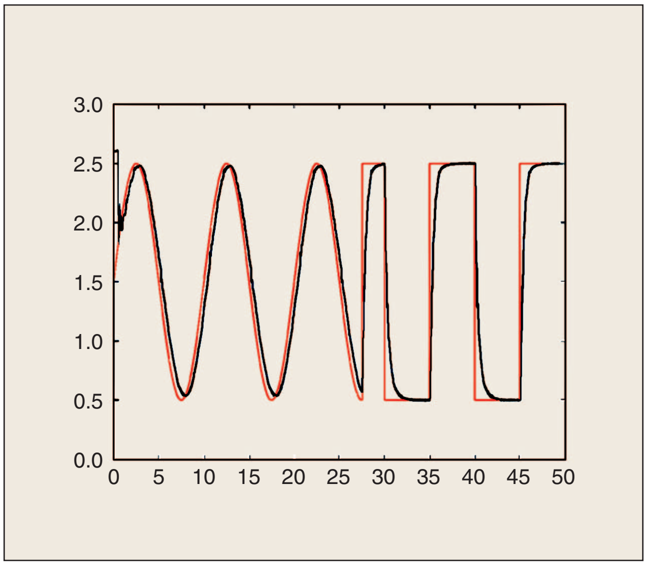


Figure 10. *A time plot of an experiment involving the magnetic levitation system. The red line represents the reference signal, while the black line is the real output.*

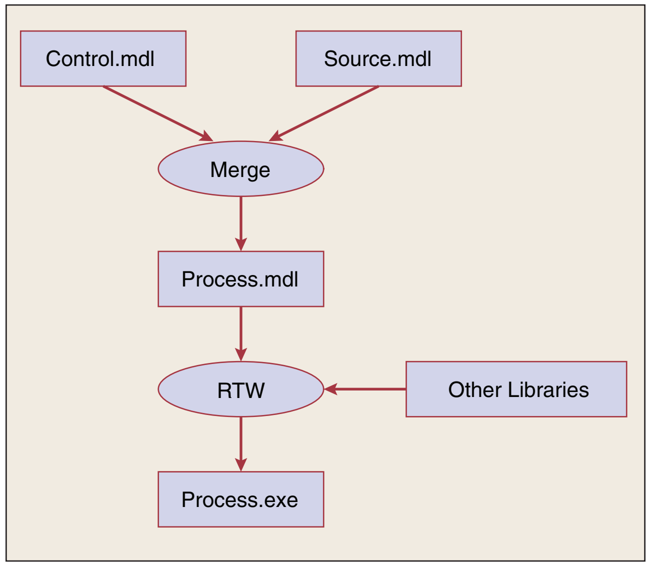


Figure 11. *Integration of the user-defined controller. After merging the controller model with the hardware interface model, the resulting model is compiled by RTW along with ACT special libraries.*

experiment is in progress, the user can look at the signals of interest in a window displaying the control input, the reference input, and the output along with their numerical values as shown in Figure 9.

A live video window is provided to view what is occurring in the remote lab. The video window is an important feature because the user can look at the real process, obtaining a sense of presence in the laboratory.

ACT has been used in control system courses at the University of Siena since 1999.

When the user stops the experiment, it is possible to download a file in MATLAB format where the signal dynamics have been stored. This file can be used to perform offline analysis, such as evaluation of the maximum overshoot and settling time, as shown in the time plot in Figure 10.

The ACT Architecture

The software architecture consists of two parts: the first part concerns control of the physical process (server side), and the second part relates to the user interface (client side).

The ACT server runs on the Microsoft Windows 2000 platform and is based on the MATLAB/Simulink environment, allowing the user to design her/his own controller. The steps necessary to obtain the executable file from a controller model are shown in Figure 11. The first phase involves merging the user-defined controller (control.mdl) with a Simulink model representing the process (source.mdl). Once the output model (process.mdl) has been obtained, the MATLAB Real-Time Workshop (RTW) routine converts the Simulink model into a C source file, which is compiled to obtain executable code. The compilation task is integrated with libraries that allow the executable file to perform special functions, such as communication with the user and real-time control of the process.

The client side is based on HTML pages and Java applets to maximize portability across various platforms. The home page and other descriptive pages are static HTML pages generated by PHP. The "Control Type Interface" in Figure 3, which changes with the chosen experiment, has been implemented as a dynamic page through the use of PHP language. The integration of the user-defined controller in executable code is handled by another PHP script. All data regarding the experiments, user access, and controllers are stored in a MySQL database.

The overall software architecture is summarized in the block diagram of Figure 12. Once the executable file process.exe has been compiled, the "Experiment Interface" window pops up onto the client machine. This interface is a Java applet that allows the user to communicate with process.exe through a TCP connection. Through this connection it is possible to change the references and the controller parameters online and to send the experimental data over the Internet. A Web cam is used to send streaming video to the remote user.

As illustrated in Figure 12, the controller (process.exe) resides on the computer connected to the process, allowing safe execution of the experiment despite network delays.

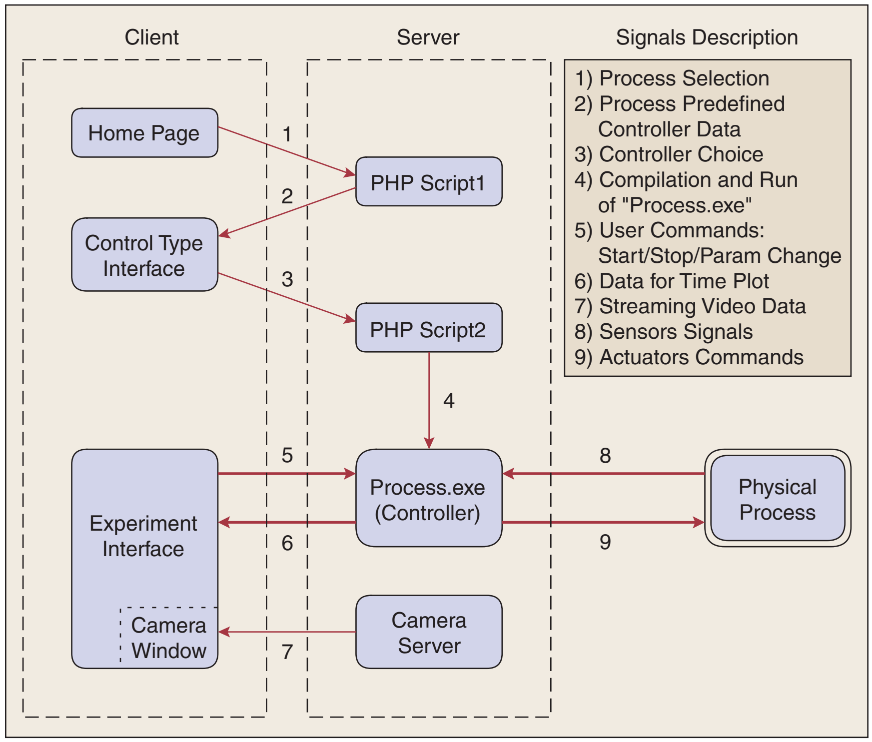


Figure 12. *The ACT software architecture. Since process.exe resides on the computer directly connected to the process, delays due to Internet time lags do not affect the performance of the controlled system.*

Teaching Experiences

ACT has been used in control system courses at the University of Siena since 1999. A typical student ACT session consists of studying the physical and mathematical model of an ACT process, synthesizing a controller through the Simulink controller template, validating the control system on a Simulink model of the process, running the ACT experiment, downloading data, and analyzing the control system performance off line.

Future plans include using ACT in robotics courses, allowing students to perform experiments with the LEGO mobile robot, and designing and testing control laws on the real process. Work is in progress on the design of path-planning algorithms to enable the LEGO robot to move in an environment with obstacles.

An important step toward the educational objectives of ACT has been realized by the student competition component of the telelab [22], which is designed to stimulate groups of students to compare their user-designed controllers with those of other students. In the competition, performance indexes are automatically computed, and the results are stored and ranked. At present, this feature is available only for the magnetic levitation process. Students are enthusiastic about the competition, and their comments are positive.

Users have connected to and used ACT from around the world. From January 2003 to October 2003, ACT was accessed by 2,640 users, mainly from Italy, Brazil, and the United States. The overall running time for the available experiments was 3,850 minutes.

Other institutions that use ACT in their courses include the University of Pisa, the Polytechnic of Milan, and the University of Bologna. Additionally, the ACT server has been installed in the Department of Aeronautics and Astronautics at MIT to allow remote experiments on 3-DOF helicopter simulators.

Conclusions

ACT provides a sophisticated yet easy-to-use structure to control a remote process through the Internet. The telelaboratory allows control system students to operate real processes without being physically present in the lab. One of the distinguished features of ACT is that users can design their own controllers through Simulink. This feature helped develop a student competition mechanism, where groups of students can compare their own user-designed controllers.

In designing the ACT architecture, special attention has been paid to simplifying the procedures for adding new online experiments. Work is in progress both to upgrade ACT with new processes and to allow the ACT server to run under Linux.

References

[1] S. Dormido, "Control learning: Present and future," in *Proc. 15th IFAC World Congress '02*, Barcelona, Spain, 2002, pp. 81–103.

[2] S.E. Poindexter and B.S. Heck, "Using the Web in your courses: What can you do? What should you do?" *IEEE Contr. Syst. Mag.,* vol. 19, no. 1, pp. 83–92, 1999.

[3] C.M. Merrick and J.W. Ponton, "The ECOSSE control hypercourse," *Comput. Chem. Eng.*, vol. 20, sup., no. 972, pp. S1353–S1358, 1996.

[4] K.M. Lee, W. Daley, and T. McKlin, “An interactive learning tool for dynamic systems and control,” in *Proc. Int. Mechanical Engineering Congr. Expo.*, Anaheim, CA, 1998, pp. 71–76.

[5] C. Schmid, “The virtual lab VCLAB for education on the Web,” in *Proc. American Control Conf.*, Philadelphia, PA, 1998, pp. 1314–1318.

[6] C.D. Knight and S.P. DeWeerth, “World Wide Web-based automatic testing of analog circuits,” in *Proc. 1996 Midwest Symp. Circuits and Systems*, Ames, IA, 1996, pp. 295–298.

[7] M. Shaheen, K. Loparo, and M. Buchner, “Remote laboratory experimentation,” in *Proc. American Control Conf.*, Philadelphia, PA, 1998, pp. 1314–1318.

[8] J. Henry. (Nov. 1, 2003) “Engineering lab web resource center” [Online]. Available: http://chem.engr.utc.edu

[9] B. Aktan, C.A. Bohus, A. Crowl, and M.H. Shor, “Distance learning applied to control engineering laboratories,” *IEEE Trans. Educ.*, vol. 39, no. 3, pp. 320–326, 1996.

[10] A. Bhandari and M. Shor, “Access to an instructional control laboratory experiment through the World Wide Web,” in *Proc. American Control Conf.*, Philadelphia, PA, 1998, pp. 1319–1325.

[11] H.H. Hahn and M.W. Spong, “Remote laboratories for control education,” in *Proc. Conf. Decision and Control*, Sydney, Australia, 2000. pp. 895–900.

[12] A. Bicchi, A. Coppelli, F. Quarto, L. Rizzo, F. Turchi, and A. Balestrino, “Breaking the lab’s walls, tele-laboratories at the university of Pisa,” in *Proc. 2001 IEEE Int. Conf. Robotics and Automation*, Seul, Korea, 2001, pp. 1903–1908.

[13] K. Goldberg, M. Masha, S. Gentner, and N. Rothenberg, “Desktop teleoperation via the World Wide Web,” in *Proc. IEEE Int. Conf. Robotics and Automation*, Nagoya, Japan, 1995, pp. 654–659.

[14] J.W. Overstreet and A. Tzes, “An Internet-based real-time control engineering laboratory,” *IEEE Contr. Syst. Mag.*, vol. 19, no. 5, pp. 19–34, 1999.

[15] V. Ramakrishnan, Y. Zhuang, S.Y. Hu, J.P. Chen, C.C. Ko, B.M. Chen, and K.C. Tan, “Development of a web-based control experiment for a coupled tank apparatus,” in *Proc. American Control Conf.*, Chicago, IL, 2000, pp. 4409–4413.

[16] T.F. Junge and C. Schmid, “Web-based remote experimentation using a laboratory-scale optical tracker,” in *Proc. American Control Conf.*, Chicago, IL, 2000, pp. 2951–2954.

[17] J. Apkarian and A. Dawes, “Interactive control education with virtual presence on the Web,” in *Proc. American Control. Conf.*, Chicago, IL, 2000, pp. 3985–3990.

[18] G. Choy, D.R. Parker, J.N. d’Amour, and J.L. Spencer, “Remote experimentation: A web-operable two phase flow experiment,” in *Proc. American Control Conf.*, Chicago, IL, 2000, pp. 2939–2943.

[19] M. Casini, “Designing a tele laboratory for control of dynamic systems through the internet,” Master’s thesis (in Italian), Università di Siena, 1999.

[20] M. Casini, D. Prattichizzo, and A. Vicino, “The Automatic Control Telelab: A remote control engieering laboratory,” in *Proc. Conf. Dec. Control*, Orlando, FL, 2001, pp. 3242–3247.

[21] M. Casini, D. Prattichizzo, and A. Vicino, “The automatic control telelab: A user-friendly interface for distance learning,” *IEEE Trans. Educ.*, vol. 46, no. 2, pp. 252–257, 2003.

[22] M. Casini, D. Prattichizzo, and A. Vicino, “E-learning by remote laboratories: A new tool for control education,” in *Preprints 6th IFAC Symposium on Advances in Control Education*, Oulu, Finland, 2003, pp. 95–100.

Marco Casini received the Laurea degree in computer science engineering and the Ph.D. in control systems from the University of Siena in 1999 and 2003, respectively. He is currently a research associate at the Dipartimento di Ingegneria dell’Informazione of the University of Siena. Since 1999 he has held several fellowships at the Dipartimento di Ingegneria dell’Informazione of the University of Siena. In 2001 he was a visiting scientist at the Laboratory for Information and Decision Systems (LIDS) at Massachusetts Institute of Technology, Cambridge. His research interests include distance learning, remote laboratories, set membership identification, and identification for control.

Domenico Prattichizzo received the M.S. degree in electronics engineering and the Ph.D. degree in robotics and automation from the University of Pisa in 1991 and 1995, respectively. In 1994 he was visiting scientist at the Artificial Intelligence Lab at MIT. Since 2002 he has been an associate professor at the University of Siena. He was cochair of the Second IEEE International Workshop on Control Problems in Robotics and Automation in 2002. His research activities include distance learning, optimal control, robotic manipulation, and visual servoing. He is an associate editor for *IEEE Transactions on Robotics and Automation*, an Editorial Board member of the *Journal of Dynamics of Continuous, Discrete and Impulsive Systems Series B*, and a member of the Conference Editorial Board of the IEEE Control Systems Society. He is coauthor of more than 90 papers in the area of control systems and robotics.

Antonio Vicino (vicino@ing.unisi.it) received the Laurea in electrical engineering from the Politecnico di Torino, Torino, Italy. From 1987 to 1990 he was associate professor of control systems at the Università di Firenze. In 1990 he joined the Dipartimento di Ingegneria Elettrica, Università di L’Aquila, as professor of control systems. Since 1993 he has been with the Università di Siena, where he is currently dean of the Engineering Faculty. In 2000 he founded the Center for Complex Systems Studies of the University of Siena, where he is currently director. He was an associate editor for *IEEE Transactions on Automatic Control* from 1992 to 1996. Presently he is associate editor for *Automatica* and associate editor at large for *IEEE Transactions on Automatic Control*. He is a Fellow of the IEEE. He is the author of 190 technical publications and coeditor of two books on robustness in identification and control. He has worked on stability analysis of nonlinear systems and time series analysis and prediction. His present main research interests are in robust control of uncertain systems, robust identification and filtering, applied system modeling, and mobile robotics. He can be contacted at Dipartimento di Ingegneria dell’Informazione, Università di Siena, Via Roma 56, 53100 Siena Italy.