An Integrated Virtual and Remote Control Lab

The Three-Tank System as a Case Study

Internet-based technologies can supplement traditional laboratories with remote or simulated experimentation sessions. The authors describe their virtual and remote laboratory, which uses a nonlinear system that students can run from anywhere on the Internet. The implementation integrates both open source and commercial software tools.

Quality education in computing science and engineering has always been connected to two levels of knowledge: theoretical and practical. Although theoretical knowledge is easy to transfer to students in traditional classrooms, attaining the thinking and problem-solving skills inherent in engineering requires hours of work in a conventional laboratory. In particular, students in control engineering education must gain knowledge about and skills in control system modeling before they can develop and test the controllers that enforce performance requirements. Once a controller is designed and implemented, observation of the resulting dynamics gives valuable insight into design concepts.

Clearly, this requirement represents a serious drawback for higher education distance learning. Our institution, the Spanish National Distance Education University (UNED), offers distance education courses on automatic control for as many as 300 students each year. Until recently, these students had to travel to Madrid from all over the country for up to two weeks to attend day-long laboratories to complete the prescribed hands-on experiments in system identification and control courses. Fortunately, the impressive development of Internet technologies in the past few years has highlighted the importance of Web-based teaching and learning in many research fields, including automatic control.[1] We therefore decided to use Web-based laboratories in our instruction so that students could minimize their need to physically attend laboratories yet still observe the dynamic phenomena that are often difficult to explain in written form.

Traditionally, Web-based laboratories are divided into two categories, according to the system's nature: virtual and remote. A virtual laboratory simulates a model of a physical process,[2] whereas a remote laboratory provides access to a real physical process over the Internet.[3] Simulation is an appropriate way of complementing control education, but it generally can't replace experimentation with automatic control systems in real industrial plants. A simulation is only as good as its model, and a model is just an approximation (it can't reproduce every aspect of a process); but in a remote Web-based

1521-9615/08/$25.00 © 2008 IEEE
Copublished by the IEEE CS and the AIP

Natividad Duro, Raquel Dormido, Héctor Vargas, Sebastián Dormido-Canto, José Sánchez, Gonzalo Farias, and Sebastián Dormido
Spanish National Distance Education University (UNED)
Francisco Esquembre
University of Murcia

laboratory, students can change control parameters, run experiments, see results, and download data over the Internet. Recent research shows that different disciplines have successfully deployed remote laboratories in various outreach projects or to help illustrate scientific phenomena that require costly or difficult-to-assemble equipment.[4–6]

Our use of remote laboratories is fully motivated by our institution's distance education needs. For educational purposes, we proposed a three-tank system to serve as a benchmark for different purposes, as a test environment for fault detection and identification, and for reconfigurable control.[7] Traditional laboratories have taken notice because our system presents interesting properties in both control education and research.[8] In particular, our system exhibits the typical characteristics of a constrained hybrid system[9] and has proved useful as a testbed for state-estimation algorithms and control and identification of hybrid systems. For these reasons, institutions frequently use it to show the results of different control strategies and as an educational tool for teaching advanced control techniques.

This article describes our novel implementation of a three-tank plant that uses both a virtual and a remote laboratory. Our Web-based system offers both a simulation of the three-tank plant and an Internet connection to a real plant (located in our facilities in Madrid), which students can control remotely. We even let students compare the responses of the computer model with the real plant by overlaying the simulation's graphical output with real-time video captures from the plant. Using our system, students can get a high-quality remote experimentation session that compares to the manipulation of a real plant without traveling far from home.

The Three-Tank System

Our laboratory is the result of instructor interaction with experts in educational technology. We used Easy Java Simulations (EJS; http://fem.um.es/Ejs) to implement the model and serve as a laboratory GUI because the tool's low programming requirements ensure that a wide range of academic staff can access and modify the laboratory's pedagogical portions for future needs. We implemented the lower-level, server-side communication and hardware control portion of the laboratory with Labview (www.ni.com/labview/), a de facto tool for accessing computer-controlled laboratory equipment, and standard TCP/IP communication routines.

For our laboratory, we chose the DTS200 three-tank system manufactured by Amira GmbH.

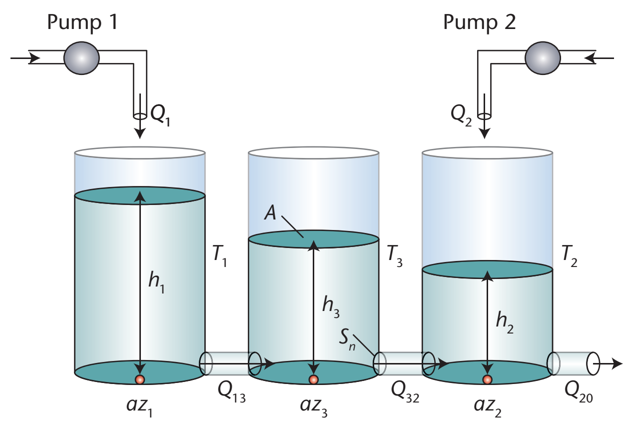

Figure 1. Amira's three-tank system. From left to right, tanks T_1, T_2, and T_3 are connected to each other serially by cross-section S_n pipes.

System Overview

The three-tank plant consists of three cylinders T_1, T_2, and T_3 with the same cross-section A. These cylinders are connected serially to each other by cross-section S_n pipes. Figure 1 shows the plant's full structure.

The right-hand side of tank T_2 has a single outflow valve (through which Q_{20} flows) that also has a circular cross-section S_n. The liquid flowing out from the system is collected in a reservoir located under the three tanks—this reservoir supplies pumps 1 and 2 with liquid, which eventually returns to the system (pumps 1 and 2 represent the input flows of tanks T_1 and T_2). In this closed system, the liquid that enters the reservoir from the tanks returns to the tanks via the pumps. However, these pumps switch off automatically when the liquid level of T_1 or T_2 exceeds a given upper limit.

In addition to the outflow valve at T_2, the system has five more valves. Two of them connect two consecutive tanks (one for the T_1 to T_3 connection, through which Q_{13} flows, and the other for the T_3 to T_2 connection, corresponding to the Q_{32} flow) and can be manually adjusted to close the link between them. The other three are leak valves located at the bottom of each tank that can be used to manually drain the tanks.

Pump flow rates correspond to process input signals; the liquid levels in tanks T_1 and T_2 are the output signals. System users can manage all these signals for control purposes.

Mathematical Model

The plant's mathematical model is equivalent to the following balance equations:

Table 1. Effects of independent proportion, integral, and derivative (PID) tuning.

Closed-loop response	Rise time	Overshoot	Settling time	Steady-state error	Stability
Increasing K_P	Decrease	Increase	Small increase	Decrease	Degrade
Increasing K_P/T_I	Small decrease	Increase	Increase	Large decrease	Degrade
Increasing $K_P T_D$	Small decrease	Decrease	Decrease	Minor change	Improve

$$A\frac{dh_1}{dt} = Q_1 - Q_{13} - Q_{1leak} \tag{1}$$

$$A\frac{dh_3}{dt} = Q_{13} - Q_{32} - Q_{3leak} \tag{2}$$

$$A\frac{dh_2}{dt} = Q_2 - Q_{32} - Q_{20} - Q_{2leak}, \tag{3}$$

where t represents time, h_1, h_2, and h_3 represent the liquid levels in each tank, A represents the tanks' cross sections, Q_1 and Q_2 denote the flow rates of pumps 1 and 2, Q_{ij} denotes the flow rates between tank T_i and T_j ($j = 0$ represents the system's output), and Q_{ileak} (i = 1, 2, or 3) represents the flows that leave the respective tank when the drain valve is open. These three balance equations mean that the volume variance in each tank is equal to the sum of the flow rates that enter and leave the tank.

Yet flows Q_{13}, Q_{32}, and Q_{20} are still unknown in Equations 1, 2, and 3. To obtain them, we use Torricelli's law:

$$Q_{ij} = az_i S_n \operatorname{sgn}(h_i - h_j)\sqrt{2g\,|\,h_i - h_j\,|}, \tag{4}$$

where az_i is the outflow coefficient, sgn(x) is the sign of the argument x, and g is the acceleration due to gravity.

The resulting equations for calculating the partial flows are thus

$$Q_{13} = az_1 S_n \operatorname{sgn}(h_1 - h_3)\sqrt{2g\,|\,h_1 - h_3\,|} \tag{5}$$

$$Q_{32} = az_3 S_n \operatorname{sgn}(h_3 - h_2)\sqrt{2g\,|\,h_3 - h_2\,|} \tag{6}$$

$$Q_{20} = az_2 S_n \sqrt{2gh_2}\,. \tag{7}$$

We modeled the valves of the pipe connections, the pumps, and the leaks with the usual relationships.[10]

Control Configuration Scheme

From a practical viewpoint, we can run controllers in two different modes: manual or automatic. In manual mode, the operator directly manipulates the controller output, typically by pushing buttons that increase or decrease it. In automatic mode, a computer or some other automatism generates the controller output signals. Manual mode is most useful during the plant's startup and shutdown or in an emergency situation. Conventional controllers have a manual/automatic switch to transfer the controller from automatic mode to manual and vice versa.

Other alternative controller design theories can control the fluid level in the tanks while in automatic mode.[11]

Decentralized control. A controller's objective in our system is to control the liquid level in both T_1 and T_2. From an inspection of the process equations, it seems natural to use the pump flow rate $Q_1(t)$ to control the level in the first tank $h_1(t)$ and $Q_2(t)$ to control $h_2(t)$. This strategy is often called *decentralized control*, a simple approach to multivariable controller design in which the matrix transfer function $G(s)$ is a square plant (2 × 2) to be controlled with a diagonal controller.

This strategy works well if $G(s)$ is close to diagonal because the plant to be controlled is essentially a collection of independent subplants, and we can design each element in the controller $C(s)$ independently, as indicated in Equation 8:

$$C(s) = diag\{C_1(s)C_2(s)\}. \tag{8}$$

A classical and popular technique for designing C_1 and C_2 is via proportional-integral-derivative (PID) control. A standard PID controller, also called a *three-term controller*, has a transfer function $C(s)$ given by

$$C(s) = K_P(1 + \frac{1}{T_I S} + T_D S), \tag{9}$$

where K_P is the proportional gain, T_I the integral time constant, and T_D the derivative time constant. For the three-term functionalities,

- the proportional term provides an overall control action proportional to the error signal through the all-pass gain factor;

- the integral term reduces steady-state errors through low-frequency compensation via an integrator; and
- the derivative term improves transient response through high-frequency compensation via a differentiator.

Table 1 summarizes the individual effects these three terms have on closed-loop performance. Researchers have developed many PID variants to improve system performance.[12]

Decoupled control. If off-diagonal elements in $G(s)$ are large, then the performance with decentralized diagonal control could be poor because no attempt is made to counteract the interaction.

Figure 2 shows that if the set-point for tank T_1 changes, then C_1 will adjust the flow rate $Q_1(t)$. However, a change in Q_1 will also affect the level in tank T_2. This phenomenon is known as *cross-coupling* and can produce a steady-state dynamic behavior worse than the one predicted with the multiloop approach.

In decoupling control,[13] we introduce a transfer matrix $D(s)$ between the controller outputs and plant inputs. Figure 3 shows the structure. This approach, which we also implemented in our laboratory, attempts to remove the dynamic interaction between the process loops as well as the steady-state interaction.

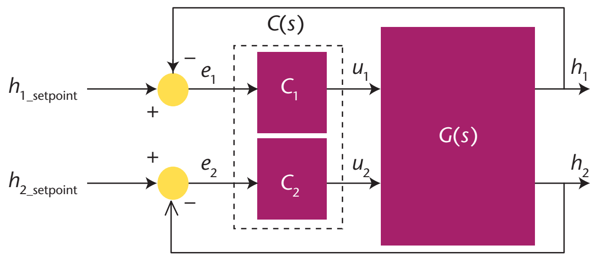

Figure 2. Decentralized control strategy. $G(s)$ is the three-tank system to be controlled, and $C(s)$ is the controller to be designed; it's composed of two elements C_1 and C_2. C_i controls the level in tank i. The variables $h_{i\_setpoint}$, e_i, u_i, and h_i are the set-point to be reached, the error signal, the control signal, and the liquid level in tank i, respectively.

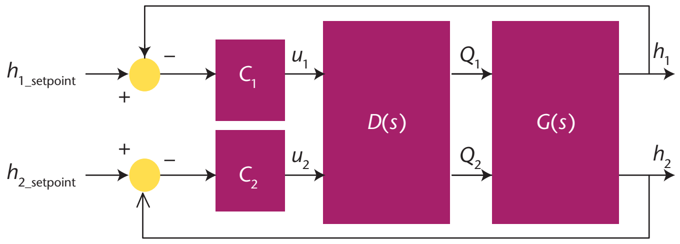

Figure 3. Decoupled control strategy. $G(s)$ is the three-tank system to be controlled, C_i is the controller that adjusts the flow rate Q_i, and $D(s)$ is a decoupling control matrix. The variables $h_{i\_setpoint}$, u_i, Q_i, and h_i are the set-point to be reached, the control signal, the flow rate, and the liquid level in tank i, respectively.

Virtual and Remote Laboratory Operation

Students first gain access to our three-tank system by visiting our Web page at http://lab.dia.uned.es via any standard Java-enabled Web browser. This page provides instructions about the experiments to be conducted as part of a course and grants access to the Java applet that serves as our laboratory's GUI. The applet can operate in both virtual and remote modes.

Plant Operation: Virtual Mode

Figure 4 shows the applet's main window, which displays the plant's schematic representation in virtual mode.

In virtual mode, the plant's behavior is simulated by using a software model based on the equations listed earlier. We tried to make the plant's schematic representation as self-explanatory as possible: students can easily visualize the system's different components and see how the liquid levels in the tanks change with different valve and pump parameters.

Students can play, pause, and reset a simulation by using standard push buttons in the GUI or by manipulating the two rows of valve parameter sliders. They can also control the liquid levels in

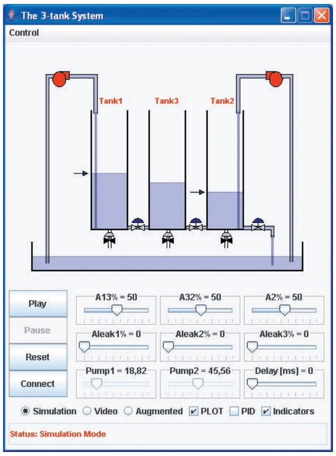

Figure 4. Three-tank system GUI. In virtual mode, we can simulate plant behavior by using a software model based on different mathematical equations.

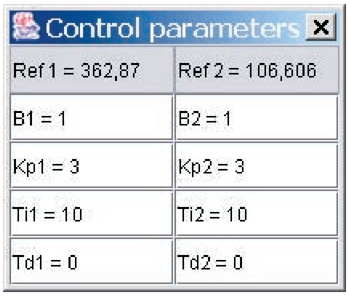

Figure 5. Control parameters. Users can manipulate this window to change the parameters in a proportional integer (PI) control strategy.

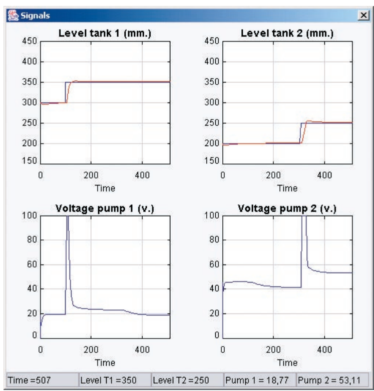

Figure 6. Controlled and manipulated variables. The top half of the figure plots the controlled variables (the levels of the two tanks, T_1 and T_2) along with their set-points, whereas the bottom half shows the voltages applied to the pumps supplying those tanks.

tanks T_1 and T_2 by changing the voltage applied to the pumps. If students control the system in manual mode, they directly set the voltages of the pumps by using the lower row of sliders provided in the applet's main window; the voltage required for the pumps to reach the prescribed set-points is indicated by two small black arrows displayed on the sides of tanks T_1 and T_2. Students can change these set-points interactively by dragging the arrows up and down. In automatic mode, students can select which controller strategy to use and then tune its parameters for optimal performance. By checking the "PID" box, a second window helps them adjust the parameter values of the controller selected. Figure 5 shows an example of the parameters for a PI (T_D = 0) control strategy for each liquid level. The student can specify proportional gain (K_P), integral time (T_I), and derivative time (T_D) for both controllers by typing in a specific value; he or she can also change a fourth parameter, B, which represents the set-point weighting parameter. This parameter reduces the overshoot in the output after following step changes to the set-point signal.[13] The window shows the reference values that the controllers follow, but the user can't change these values in this panel.

An additional window displays time plots for the system's controlled and manipulated variables. The top half of Figure 6 plots the levels of the two tanks T_1 and T_2 (the controlled variables) together with their prescribed set-points. The bottom half displays the changes in control variables—that is, the voltages applied to the pumps supplying tanks T_1 and T_2. These two variables evolve in time differently depending on the controller action selected. The bottom of this window also displays the value of the simulation time, the level of tanks T_1 and T_2 in millimeters, and the percentage values of pumps 1 and 2.

Plant Operation: Remote Mode

The bottom of the applet's main window displays the plant's operation mode: simulation or remote. By default, the applet always starts in simulation (or virtual) mode. A button labeled "Connect" lets users work in remote mode by using the real three-tank system running at our facilities. Login and password identification are required to access this equipment.

After the connection is established, the parameters the user sets in the simulation GUI go to the real plant via the Internet. At this point, users can also check the real system's response and compare it to that of the simulation: when the user connects to the plant, the time plots show the output data from both the real and the simulated plants.

The user can also switch the system display among the virtual representation, a real-time video image captured from a Web cam pointing at the real plant in Madrid (Figure 7a), and a superposition of both images (Figure 7b). We've found that both simulation and remote results are typically similar because the simulation faithfully represents real system behavior.

In video mode, six additional buttons—General View, Close View, Top Tank1, Bottom Tank1, Top Tank2, and Bottom Tank2—let the user control the video camera and focus on particular parts of the three-tank system. However, only the general view graphically matches the superimposed virtual scheme.

Implementation Issues

Now let's look at the tools we used to implement the three-tank laboratory at both the client and server sides.

The client side consists of a Java applet that runs on the student's Web browser. This applet acts either as a virtual laboratory by running the mathematical model locally (on the student's computer) or as a remote laboratory by establishing a TCP connection with the hardware controller running on the server side (in our facilities). In other words, when the interface is used as a remote laboratory, a TCP channel between the GUI and the plant replaces the local channel between the GUI and the model.

The server side consists of a standard HTTP server and a hardware controller that communicates via TCP/IP with the applet and is connected directly to the plant hardware.

Client-Side Implementation

We created the client-side applet with EJS, a freeware, open source tool developed in Java specifically to help nonprogrammers create interactive simulations in Java.[14,15]

The EJS architecture follows the successful model-view-controller paradigm and provides a simplified structure in which the author can create the simulation model by declaring and initializing the model's variables and stating the ordinary differential equations that govern how the state variables change over time (see Figure 8). The program automatically generates the Java code required to solve these differential equations by using standard solver algorithms. The algorithm we used in our three-tank laboratory is the classic fourth-order Runge-Kutta, but we could have specified additional computations by using Java expressions and sentences to obtain other output variables or to compute the required input control signals.

The view of a simulation in EJS gives both the visualization and the interaction needed to control it. Accordingly, EJS includes a wide collection of ready-to-use graphical elements ranging from buttons and video components to 2D and 3D drawings and plots. This collection includes a special section on automatic control components, such as tanks, pipes, pumps, valves, and so on. The view designer can add these view elements with an easy drag-and-drop interface to create a realistic visualization of the three-tank system. Figure 9 partially illustrates the construction of the view using these elements.

Each view element has a set of properties that the view designer can customize by using constant values or (more interestingly) model variables to

(a) (b)

Figure 7. Real vs. virtual. (a) A real-time video image matches (b) the real-time video image superimposed with the virtual representation.

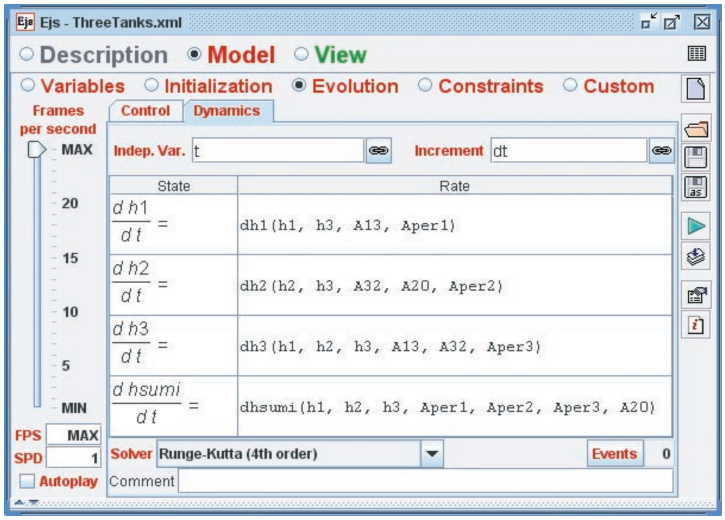

Figure 8. Easy Java Simulations (EJS). We used EJS to build the differential equations for our three-tank system.

produce a bidirectional flow of information between the view and the model. The view then automatically displays any variable change; likewise, any user interaction with the view automatically changes the corresponding variable's value.

With this high-level information, EJS generates the simulation program's Java source code, compiles the program, packs the resulting object files into a compressed file, and generates the HTML page required to run the simulation as an applet.

Server-Side Implementation

Our remote laboratory architecture is based on

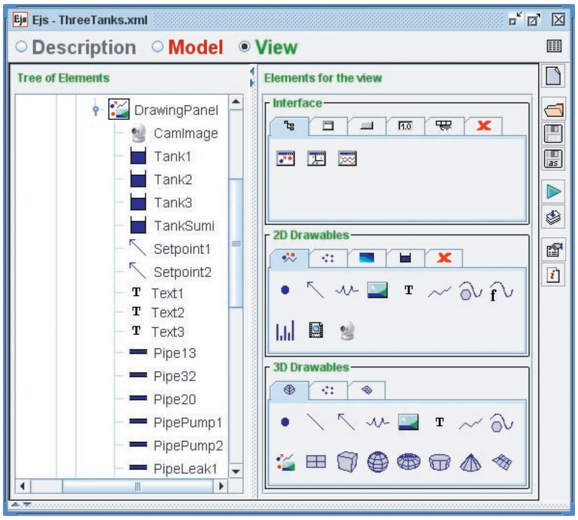

Figure 9. Easy Java Simulations. This partial view of available elements shows how easy it is to create a visualization of the three-tank system.

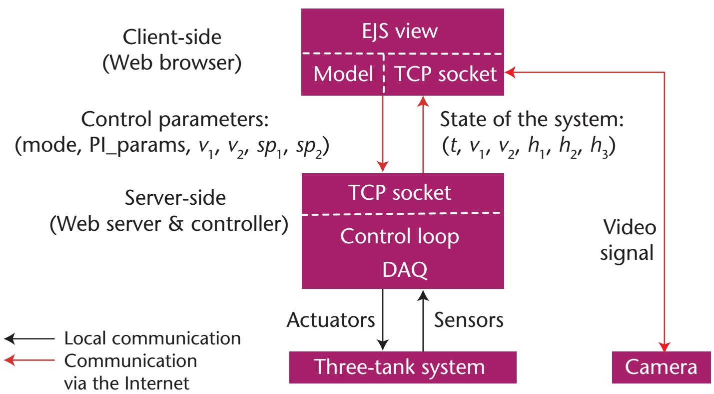

Figure 10. Single client-server structure. The client side is the applet's GUI, and the server side is a computer running an HTTP server and the local control loop. The Web server lets students download the client applet.

a single client-server structure, which means the same computer acts both as the Web server and the controller (see Figure 10).

We developed the controller in the Labview graphical programming language[16] and made it consist of two different information loops: asynchronous tuning and synchronous control. The first is closed via the Internet with the laboratory GUI and can suffer unpredictable network delays—the information circulating in this loop can contain user actions or the system state. On the server side, both loops exchange information by using common global variables, including controller parameters, set-points, pump voltages, and liquid levels.

We added special calls in the EJS model to help establish communication with Labview by using standard TCP functions and routines when the system runs in remote mode. Thus, every time a user changes any value in the EJS view, the new value travels through the information loop to the server side and is written in its corresponding Labview global variable. The control loop then reads this variable at each sampling interval and applies the new value to the system. The control loop implemented in Labview governs the system according to the data packets sent from the EJS view and returns the output values read from the plant. The control parameters sent from the view to the hardware controller are control mode (manual or automatic), PI parameters (K_{P1}, T_{I1}, K_{P2}, TI_2), pump voltages (v_1, v_2), and T_1 and T_2 set-points (sp_1, sp_2). The information returned to the view consists of six values: sampling time, current pump voltages (v_1, v_2), and tank levels (h_1, h_2, h_3).

The server-side Labview implementation and the complete client-side XML description are available by contacting us directly (see p. 59 for contact information); they're easy to inspect with Labview and EJS, respectively. The connectivity between Labview and EJS hasn't presented any problem in our laboratory because of the three-tank system's slow dynamics. Faster dynamics could cause problems due to the typical delays in TCP/IP communication.

Examples of Interactive Control Experiments

Students typically perform experiments remotely by following a prescribed protocol (accessible from http://lab.dia.uned.es) that includes theoretical readings, problem worksheets, and practical sessions in our virtual and remote laboratory. Students must keep a record of their work in a laboratory notebook that they regularly discuss with instructors over the Internet. After successfully completing the experiments, students get an additional point that adds to their final grade. (In Spain, grades range from 0 to 10.)

Reference Tracking

We can use a specific experiment to evaluate reference tracking by using the different controllers we've implemented. In this context, reference tracking means that we fix a liquid level of reference at set-point in a tank; the controller then calculates the control signal to reach that level. A good tracking is achieved when the liquid level reaches the reference set-point.

At the beginning of the experiment, the student must set the following initial conditions for the system:

- The level of tank T_1 is at 300 mm, the level of T_2 is at 200 mm, and the level of T_3 is at 250 mm. These three levels can range from 0 mm to 630 mm.
- The two connection valves between tanks T_1 and T_3 and T_2 and T_3 are 50 percent open.
- All the leak valves are completely closed.
- The tank T_1 pump works at 18.82 percent, and the T_2 pump works at 45.56 percent.
- The two PI controllers have similar control parameters (the tank 1 controller's K_P = 3.38 and T_I = 56.55, and the tank 2 controller's K_P = 8.55 and T_I = 50.895).

The experiment begins with the system working in manual mode and switching to automatic mode at 150 seconds. At 500 seconds, the student changes the set-point of tank T_1 to 400 mm; we then allow settling, and at 1,000 seconds, the student fixes the set-point of T_1 to 350 mm. The reference in T_2 is set to 300 mm at 1,500 seconds and, after settling, is stepped down to 250 mm at 2,000 seconds.

The student repeats the experiment using different controllers to compare level-control strategies. Figures 11 and 12 display the result of these experiments using a PI and a decoupled PI controller. To validate the virtual laboratory results, students must conduct the same experiments in the remote laboratory and obtain similar results. We can see this by comparing Figures 11 and 13.

As we see, the system behaves nicely when any of the controllers is running, and we still achieve reference tracking after parameters change. We can also see the effect that changes in the set-point for T_1 (T_2) have on the level in T_2 (T_1). This is due to the *cross-coupling effect*. Because of the integral action, a PI controller is sufficient to ensure that the levels of T_1 and T_2 (the process outputs) agree with the set-point in the steady state.

Figure 12 shows that decoupled control works extremely well. Even though there isn't a severe interaction among the model variables in our system, using a decoupled control strategy produces significantly better performance. The response is quicker, with no overshoot, and the control effort is greater than with PI controllers.

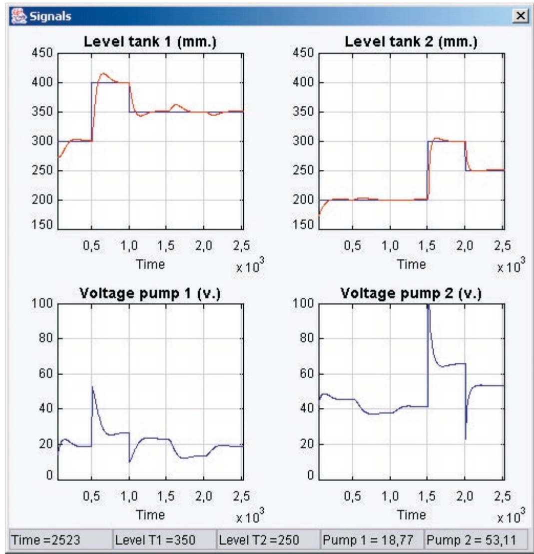

Figure 11. Two proportional integer (PI) controllers. Students can compare the evolution of controlled and manipulated process variables in different PI structures by adjusting the controller parameters.

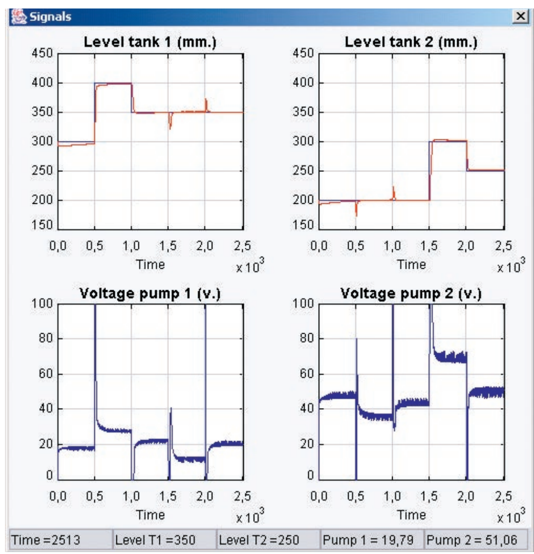

Figure 12. Decoupled controllers. Students can compare the evolution of controlled and manipulated process variables in different PI decoupled controllers by adjusting the controller parameters and then analyzing the evolution registered in the figures.

Disturbance Rejection

To prove disturbance rejection for the controllers, we can change the three system leaks to introduce a disturbance in the process. The system's initial conditions are the same as in our previous example, and the experiment begins with the system working in manual mode and switching to automatic mode at 150 seconds. At 500 seconds, the student changes the set-point of tank T_1 to 400

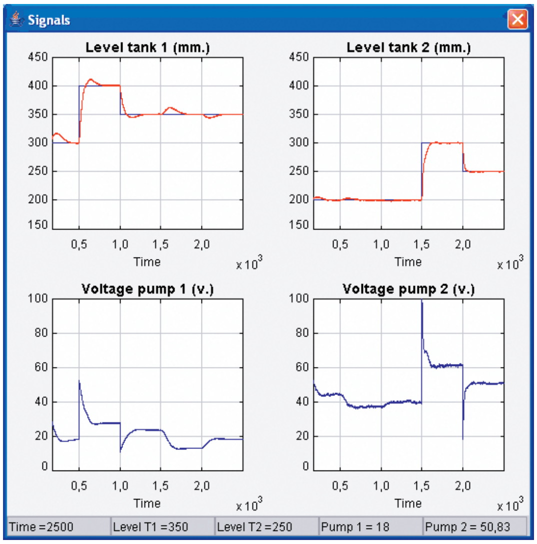

Figure 13. Real plant. Students can validate the virtual laboratory results by conducting the same experiments in the remote laboratory and getting similar results.

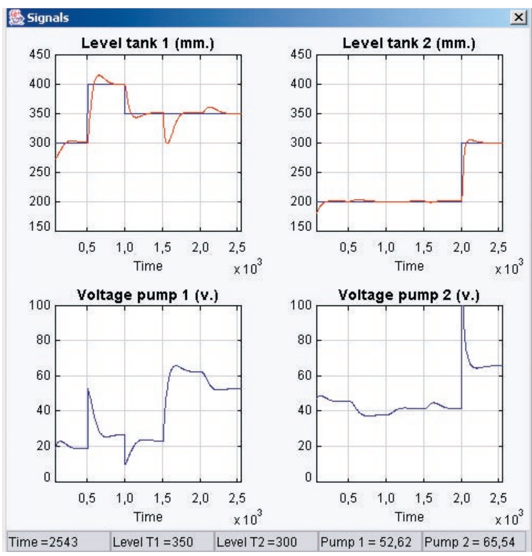

Figure 14. Disturbance rejection. If we implement the two proportional integer (PI) controllers, the controlled and manipulated process variables exhibit good performance.

mm, and then we allow settling; at 1,000 seconds, the student fixes the set-point of T_1 to 350 mm. At 1,500 seconds, the student opens the leak valve of T_1 at 30 percent. This is where we add the disturbance. Figure 14 shows the evolution of the different variables.

Similar to changing set-points, introducing a perturbation in any of the tanks implies that the controller moves the controlled variables in the right direction, toward the set-point. A disturbance could be a change in the tank leak valves or a change in the connection valves between the tanks. These manual valves can be substituted with electro-valves, which we could also consider as new controlled process variables.

Research in remote and virtual experiments for engineering education is a relatively young field, but it has grown considerably in recent years, especially in higher education distance learning. Yet, the procedure to transform a physical system into an interactive virtual and remote laboratory is far from trivial. It requires skills in advanced graphical programming to create the interactive GUI on the client side (the virtual laboratory), and, fortunately, EJS has proved to be a very useful solution for control engineering teachers.[3] But on the server side, low-level technical work is still required to create the physical connection to the hardware and the Internet communication with the client applet. We plan to extend EJS's capabilities in this direction so that programming the server-side connections is more intuitive.

So far, we've only conducted preliminary surveys about students' perceptions of the new laboratory. Our results indicate that technical functionality hasn't posed usability problems, but that design issues are the most critical feature for learners. The students' responses indicate that we can use Web-based laboratory environments in distance learning courses without compromising quality. The students found it very convenient to be able to complete their laboratory assignments without needing to travel to Madrid. From the instructors' viewpoint, it appears that students can successfully complete their laboratory assignments. Obviously, some aspects of a traditional lab aren't possible to reproduce, such as human relations.

Current and further works on the pedagogical part of our work consists of creating a battery of Web-based laboratories to offer a complete range of experiments to our students. We're also considering access rights, resource booking, and security issues in the design process.

Acknowledgments

This work was supported by the Spanish Comisión Interministerial de Ciencia y Tecnología (CICYT) under grant DPI 2004-01804 and the European Com-

mission under the project "Leonardo da Vinci Pilot Project" number 2004 N/04/B/PP 165.011.

References

1. S. Dormido, "Control Learning: Present and Future," *Ann. Rev. Control*, vol. 28, no. 1, 2004, pp. 115–136.
2. S. Dormido et al., "The Role of Interactivity in Control Learning," *Int'l J. Eng. Education*, vol. 21, no. 6, 2005, pp. 1122–1133.
3. J. Sánchez et al., "A Java/Matlab-Based Environment for Remote Control System Laboratories: Illustrated with an Inverted Pendulum," *IEEE Trans. Education*, vol. 47, no. 3, 2004, pp. 321–329.
4. B.E. Penprase, "Discovering Remote Observing in the Under-Graduate Curriculum: Remote Observing with the Pomona College 1-Meter Telescope, and Other Undergraduate Laboratories in Astronomy," *Innovation in Teaching/Learning Astronomy Methods*, 26th meeting of the IAU, Special Session 2, Czech Republic, 2006, pp. SPS2 #84.
5. G.W. Chang et al., "Teaching Photonics Laboratory Using Remote-Control Web Technologies," *IEEE Trans. Education*, vol. 48, no. 4, 2005, pp. 642–651.
6. S.C. Sivakumar et al., "A Web-Based Remote Interactive Laboratory for Internetworking Education," *IEEE Trans. Education*, vol. 48, no. 4, 2005, pp. 586–598.
7. B. Heiming and J. Lunze, "Definition of the Three-Tank Benchmark Problem for Controller Reconfiguration," *Proc. European Control Conf.*, 1999, www.uni-duisburg.de/euka/ecc99/ecc99/home.htm.
8. M. Hou, Y.S. Xiong, and R.J. Patton, "Observing a Three-Tank System," *IEEE Trans. Control Systems Technology*, vol. 13, no. 3, 2005, pp. 478–484.
9. D. Mignone, *Estimation and Control of Hybrid Systems*, Automatic Control Laboratory, ETH Zürich, 2002.
10. F.G. Shinskey, *Process Control Systems: Application, Design and Tuning*, 4th ed., McGraw Hill, 1996.
11. S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control: Analysis and Design*, Wiley, 2004.
12. K.J. Åstrom and T. Hägglund, *Advanced PID Control*, ISA, 2006.
13. K.J. Åstrom, K.H. Johansson, and Q.G. Wang, "Design of Decoupled PI Controllers for Two-by-Two Systems," *IEEE Proc. Control Theory Application*, vol. 149, no. 61, 2002, pp. 74–81.
14. F. Esquembre, "Easy Java Simulations: A Software Tool to Create Scientific Simulations in Java," *Computer Physics Comm.*, vol. 156, no. 2, 2004, pp. 199–204.
15. W. Christian and F. Esquembre, *Modeling Physics with Easy Java Simulations*, The Physics Teacher, 2007.
16. J. Travis, *Internet Applications in LabVIEW*, Prentice-Hall, 2000.

Natividad Duro *is an associate professor in the Department of Computer Sciences and Automatic Control at the Universidad Nacional de Educación a Distancia (UNED). Her current research interests are control process, modeling and simulation of continuous processes, and the design of new systems for control education. Duro has a PhD in science from UNED. Contact her at nduro@dia.uned.es.*

Raquel Dormido *is an associate professor in the Department of Computer Sciences and Automatic Control at UNED. Her current research interests are robust control, modeling and simulation of continuous processes, and the design of systems for control education. Dormido has a PhD in science from UNED. Contact her at raquel@dia.uned.es.*

Héctor Vargas *is a student in the Department of Computer Sciences and Automatic Control at UNED. His current research interests are related to the design of new systems for control education, virtual laboratories, and the use of the Internet in education. Vargas has a BS in electronic engineering from Chile de la Frontera de Temuco University. Contact him at hvargas@bec.uned.es.*

Sebastián Dormido-Canto *is an associate professor in the Department of Computer Sciences and Automatic Control at UNED. His current research interests are related to the analysis and design of control systems via intranet or Internet, high-performance interconnection networks, and optimal control. Dormido-Canto has a PhD in science from UNED. Contact him at sebas@dia.uned.es.*

José Sánchez *is an associate professor in the Department of Computer Sciences and Automatic Control at UNED. His current research interests are the design of new systems for control education, virtual laboratories, and the use of the Internet in education. Sánchez has a PhD in science from UNED. Contact him at jsanchez@dia.uned.es.*

Gonzalo Farias *is a student in the Department of Computer Sciences and Automatic Control at UNED. His current research interests include simulation and control of dynamic systems and virtual and remote laboratories. Farias has a BS in computer sciences from Chile de la Frontera de Temuco University. Contact him at gfarias@bec.uned.es.*

Francisco Esquembre *is an associate professor at the University of Murcia. His research includes computer-assisted teaching and the simulation of scientific processes for didactical purposes. Esquembre has a PhD in mathematics from the University of Murcia. Contact him at fem@um.es.*

Sebastián Dormido *is a professor of control engineering at UNED. His research activities include computer control of industrial processes, model-based predictive control, and simulation of continuous processes. Dormido has a PhD in physics from the University of the Basque Country. Since 2002, he has been president of the Spanish Association of Automatic Control CEA-IFAC (Comité Español de Automática-International Federation on Automatic Control). Contact him at sdormido@dia.uned.es.*