Proceedings of the 40th IEEE
Conference on Decision and Control
Orlando, Florida USA, December 2001

The Automatic Control Telelab: a Remote Control Engineering Laboratory

Marco Casini Domenico Prattichizzo Antonio Vicino

Dipartimento di Ingegneria dell'Informazione, Università di Siena
via Roma 56, 53100 Siena Italia
{casini,prattichizzo,vicino}@ing.unisi.it

Abstract

This paper describes the realization of a remote laboratory of automatic control developed at the University of Siena. The Automatic Control Telelab (ACT) allows the on–line interaction between remote users and a set of remote physical processes through the Internet. The key feature of this project is the user–defined controller facility. The remote user can design his/her own controller through the well–known Simulink environment. The overall architecture of the Automatic Control Telelaboratory has been designed with the goal of simplifying the upgrading procedure and the procedures to add new experiments. The Automatic Control Telelab can be found at http://www.dii.unisi.it/~control/act/index.html.

1 Introduction

The impressive development of Internet technologies in the last few years, contributed to increase the relevance of web-based teaching and learning in many research fields. For an exhaustive survey on web technologies used in control systems courses, the reader is referred to [1] where the authors describe the use of virtual and remote laboratories in control teaching. For virtual laboratories, a software simulation of physical processes is meant. Examples of this kind of laboratories are described in [2, 3, 4]. In remote laboratories, real physical processes can be accessed by remote users through the Internet. During an experiment session, users can normally change some parameters, observe the results and download data. Examples of these laboratories are given in [5, 6, 7, 8, 9].

One of the most important features to spread out the remote laboratory practice, consists of supplying the laboratory with well-known software interfaces. Most people do not want to spend time to learn special commands and procedures to run experiments. For instance in [10], the authors take advantage from the Labview packages while the Matlab/Simulink environment is used in [11, 12, 8]. Obviously, ad-hoc interfaces enjoy the property of being designed for the special application [13].

This paper presents the experience of the control group at the University of Siena, in designing a remote laboratory, namely the Automatic Control Telelab (ACT). The Matlab/Simulink environment has been chosen to implement the Automatic Control Telelab. The ACT is a remote laboratory where users can remotely choose a predefined controller to steer the process, or they can synthesize a new controller through the Matlab/Simulink environment along with its toolboxes.

One of the main features of this project consists in simplifying the user interface in a way that only the very basic notions of Simulink should be known in order to design the controller to be tested through the remote laboratory. During the experiment it is also possible to change some typical controller parameters and the reference signal. Experimental results can be checked through on–line plots and and a live video window showing the real running experiment.

The paper is organized as follows. Section 2 illustrates the motivations, the main features and the structure of the ACT. In Section 3 a typical working session is described along with an example of user-defined controller. Section 4 provides the general software architecture of the laboratory, while conclusions and future developments are drawn in Section 5.

2 Automatic Control Telelab overview

The ACT is mainly intended for educational use, and since 1999 it has been used in control systems courses [14, 15]. The aim of the project was to allow students to put in practice their theoretical knowledge of control theory in an easy way and without restrictions due to laboratory opening time and processes availability. The ACT is now accessible 24 hours a day from any computer connected to the Internet. No special soft-

0-7803-7061-9/01/$10.00 © 2001 IEEE

ware or plug-in are required. The ACT is accessible by means of any common browser like Netscape Navigator or Microsoft Internet Explorer. If the user wants to design his/her own controller, the Matlab/Simulink software is required.

It is common opinion that this kind of laboratories increases the teaching performance of control theory classes. To make the remote experiment sessions more stimulating a live video window is provided. Students can watch the real process in such window, having a most sense of presence in the laboratory.

One of the main features of the ACT is the possibility of integrating in the control loop of the remote process, the user-defined controller. The interface for the controller synthesis is very friendly. It is based on the Matlab/Simulink environment which is very popular in the control community. Since Matlab and Simulink packages are standard tools in control systems courses, there are no additional hurdles for students who want to design their own controller, which simply consists in a Simulink model similar to those commonly used to run a system simulation.

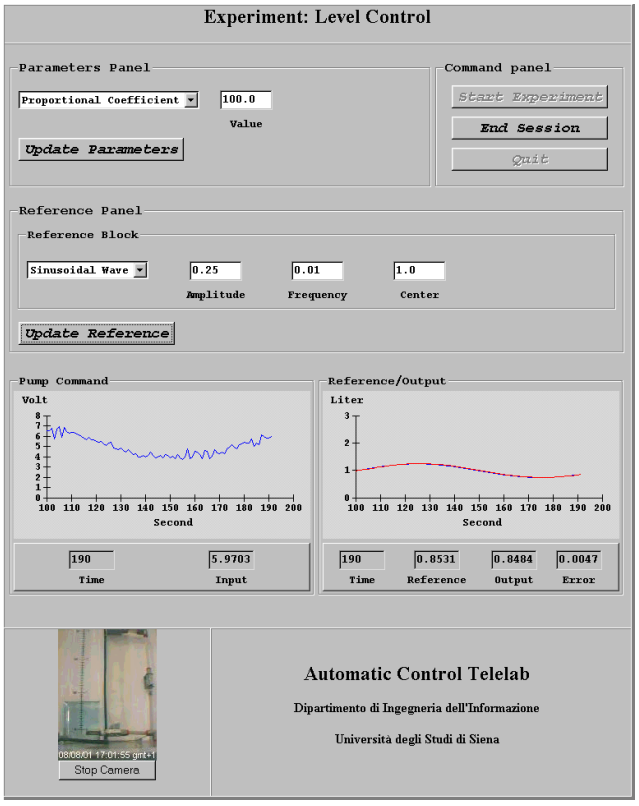

Figure 1: The *Experiment Interface* which allows for the interaction between the user and the remote process.

Other than providing the possibility of designing a user-defined controller, the ACT provides, for each available process, a set of predefined controllers whose parameters can be changed on-line by the user through the Internet. A classical example is a PID controller whose proportional, integral and derivative coefficients can be set by the user. Obviously the remote users do not need the Matlab/Simulink software at all, to run any of predefined controllers.

Figure 2: The DC motor used for angular position control.

While the remote experiment is running, the user can change on-line the reference input and some typical controller parameters, and can observe the experiment signals (command, reference and output) while live video window shows what is going on in the remote laboratory (see Fig. 1). At the end of the experiment it is possible to download data for off-line data processing.

Finally, it is worth noticing that the software and hardware architecture of the ACT has been designed with the goal of simplifying the procedure of connecting new processes to the remote laboratory.

Three processes are linked to the ACT remote laboratory: a DC motor, a tank for level control and a magnetic levitation system. The DC motor (Fig. 2) is used to control the axis angular position. The level control process (Fig. 3) has been considered because of its nonlinear dynamics. Finally, the magnetic levitation process (Fig. 4), being nonlinear and unstable, presents very interesting properties to be analyzed in control theory education. Work is in progress to add more MIMO experiments to the ACT whereby more involved control laws can be tested. In particular, a six degrees–of–freedom robot, will be soon available as an on-line experiment of the ACT remote laboratory.

3 A session description

In this section, a typical working session is described. From the home page of the ACT it is possible to access to general information pages, as for instance the

Figure 3: The tank process used for level control.

user guide of the laboratory and the list of the available experiments. After choosing the experiment to run, the *Control Type Interface* shows up as in Fig. 5. Through this interface, the user fill in a form containing personal data (used to provide statistics about ACT users), and then he chooses the controller to be integrated in the control loop of the remote laboratory process. Although some predefined control laws are available, the most stimulating aspect for the user consists in synthesizing his/her own controller.

3.1 Designing a controller

To simplify the controller design, a template model (*template.mdl*) is downloaded to the user. This template is a Simulink model which contains two subsystems, one for the controller ("ACT_Controller") and one for the reference input ("ACT_Reference").

Figure 4: The process of magnetic levitation.

With a very basic knowledge of the Simulink environment, the task of designing the controller is very easy. The control error, output and command signals are available in the "ACT_Controller" subsystem. The task which is left to the user is that of joining them by means of suitable blocks which define the controller structure. Such blocks can be dropped by any simulink toolbox available. Moreover, it is also possible

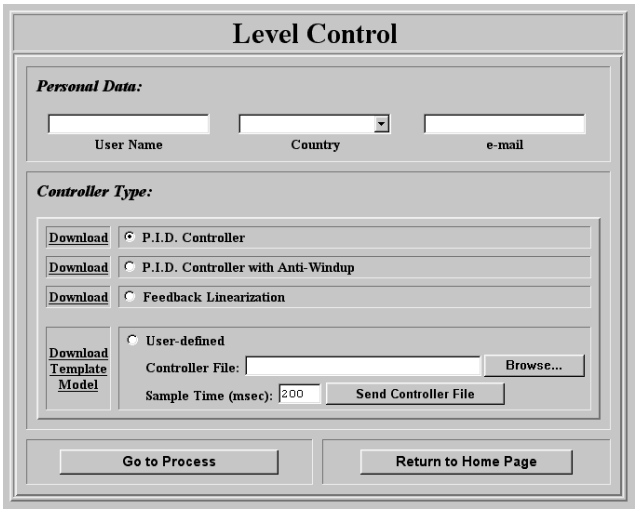

Figure 5: The *Control Type Interface*.

to set some "constant" and "gain" as on–line variables which can be modified on–line while the experiment is in progress. This feature is obtained by simply using the prefix "ACT_TP_" (ACT Tuning Parameter) to name these variables.

The second subsystem of *template.mdl* is used to choose the references which can enter the system during the experiment. A set of references is available by default, such as constant and ramp signal or sinusoidal and square waves. The user can remove some of these blocks or add new ones. To help the user in this task, other reference blocks are provided inside the "Other References" subsystem. However, for advanced users, it is also possible to design new references in the Simulink environment.

The tank level control example: In this example a controller model for the tank process shown in Fig. 3 is described. The mathematical model of the tank is

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

with:

$$q(t) = \begin{cases} 0 & \text{if } V(t) \leq 3.7 \\ 1.36 \times 10^{-5}\,(V(t) - 3.7) & \text{if } V(t) > 3.7 \end{cases} \tag{2}$$

where h is the water level inside the tank, measured by a pressure transducer on the bottom of the tank, q is the input flow and V is the voltage applied to the pump (command). Due to a threshold on the actuator (pump) the input flow is zero when the voltage applied is less than 3.7volts. Dynamics of the tank process is nonlinear. A possible type of controller is based on the so called *feedback linearization*, whose goal is to cancel the nonlinear part of (1) through a suitable action on the command. Moreover, applying to the pump a constant voltage of 3.7volts, the problems due to the

threshold can be avoided. The Simulink model implementing the feedback linearization controller is shown in Fig. 6.

Two parameters have been set as on–line variables. 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, on-line tuning of these parameters is mandatory to get better performances.

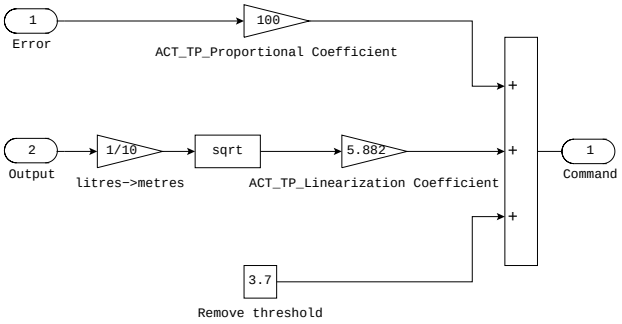

Figure 6: The Simulink model representing a controller with tuning parameters based on *feedback linearization*.

3.2 Running the experiment

Once the user–defined controller has been built, it is necessary to upload the controller model to the ACT server through the *send controller* button (this operation is not needed for predefined controllers). If the Simulink model is correct, the *Experiment Interface* shows up, as in Fig. 1, whereby it is possible to run the remote experiment by pushing the *start* button. When the 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.

Moreover, a live video window is provided to watch what is really occurring in the remote laboratory. Unlike virtual laboratories, based only on software simulations, the presence of a video window is an important feature because the user can watch the real process, having a most sense of presence in the laboratory.

During the experiment it is possible to change references on–line, as well as the controller parameters.

When the user stops the experiment, it is possible to download a file in Matlab format (*.mat*) where all the signal dynamics have been stored. This file can be used to perform off–line analysis, such as the evaluation of the maximum overshoot and the settling time.

4 The ACT architecture

The software architecture can be split in two parts: one concerns the control of the physical process (server side) and the other one relates to the user interface (client side).

The ACT server runs on the Microsoft Windows NT platform and is based on the Matlab/Simulink environment. Such an environment allows the user to design his controller through a Simulink model in a very easy way. The steps necessary to obtain the executable file from a controller model are shown in Fig. 7. The first

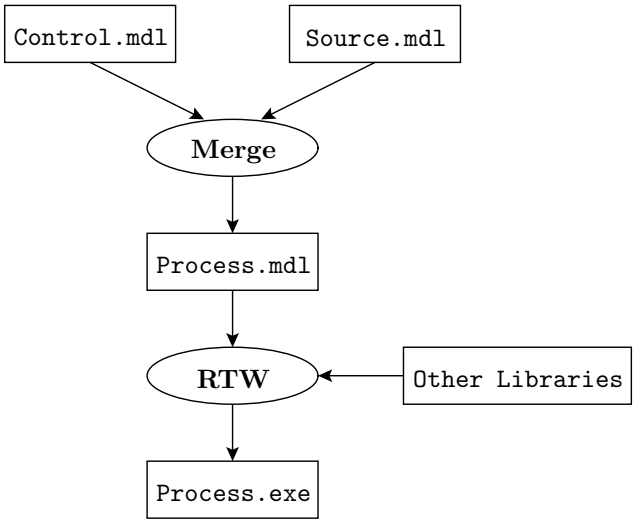

Figure 7: Integration of the user-defined controller.

phase consists in 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, it is possible to run the Matlab Real Time Workshop (RTW) routine, which allows for the conversion of a Simulink model in C source, and to compile it to get an executable code. The compiler task is integrated with some libraries which allow the executable file to perform special functions such as communications with the user and the real time control of the process.

The client side is essentially based on HTML pages and Java applets to guarantee the maximum portability across various platforms. The home page and other descriptive pages are simply static HTML pages. While the *Control Type Interface* in Fig. 5, changing with the chosen experiment, has been implemented as a dynamic page through a Java servlet (*Telelab1*). The integration of the user-defined controller in executable code, is handled by the servlet (*Interface*).

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 which al-

lows the user to communicate with *process.exe* through a TCP connection. Through this connection it is possible to change on-line the references and the controller parameters, and to send the experimental data over the Internet. A webcam is used to send streaming video to the remote user. The video software package is the *webcam32* [16]. The overall software architecture has been summarized in the block diagram of Fig. 8.

Finally, it is worthwhile to stress that the software architecture of the ACT has been designed with the goal of simplifying the upgrading procedures and the connection of new processes to the remote laboratory.

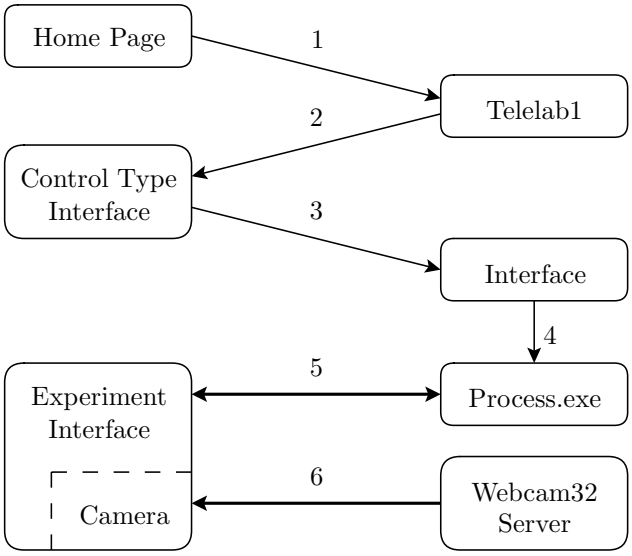

Figure 8: The ACT software architecture. 1) Process selection 2) Process predefined controllers data 3) Controller choice or user-defined controller forwarding 4) Compiling and running *Process.exe* 5) Experiment data and signals 6) Streaming video data.

5 Conclusions

The Automatic Control Telelab provides a sophisticated but easy to use mechanism to control a remote process through the Internet. This laboratory is intended to allow control system students to operate on real processes without being present in the laboratory. A student can choose a predefined controller or design a customized one (through a Simulink model) to control the process. While the experiment is running, it is possible to change controller parameters as well as the reference signals. In designing the overall ACT architecture, special attention has been devoted to simplify the procedures for adding new on–line experiments.

Work is in progress to upgrade the Automatic Control Telelaboratory with new MIMO processes. As far as system security is concerned, more advanced tools are under investigation. Actually, system security is left essentially to the actuators saturations.

References

[1] Sandra E. Poindexter and Bonnie S. Heck. Using the web in your courses: What can you do? what should you do? *IEEE Control System*, pages 83–92, Feb. 1999.

[2] C.M Merrick and J.W. Ponton. The ECOSSE control hypercourse. *Computers in Chemical Engineering*, 20, Supplement:S1353–S1358, 1996. www.chemeng.ed.ac.uk/ecosse/control/sample/index.html.

[3] K-M Lee, W. Daley, and T. McKlin. An interactive learning tool for dynamic systems and control. In *Proc. of International Mechanical Engineering Congress & Exposition*, CA, November 1998.

[4] C. Schmid. The virtual lab VCLAB for education on the web. In *Proc. of IEEE American Control Conference*, pages 1314–1318, Philadelphia, June 1998. www.esr.ruhr-uni-bochum.de/VCLab.

[5] J. Henry. *Enginering lab on line*. University of Tennessee at Chattanooga, 1998. chem.engr.utc.edu.

[6] Burçin Aktan, Carisa A. Bohus, A. Crowl, and Molly H. Shor. Distance learning applied to control engineering laboratories. *IEEE Transactions on education*, 39(3):320–326, August 1996.

[7] A. Bhandari and M. Shor. Access to an instructional control laboratory experiment trough the world wide web. In *Proc. of IEEE American Control Conference*, pages 1319–1325, Philadelphia, June 1998. www.ece.orst.edu/˜aktanb/distance-labs.html.

[8] Henry H. Hahn and Mark W. Spong. Remote laboratories for control education. In *Proc. of 39th IEEE Conference on Decision and Control*, Sydney, Australia, December 2000.

[9] Jamahal W. Overstreet and Anthony Tzes. An interntet based real–time control engineering labotatory. *IEEE Control Systems Magazine*, pages 19–34, October 1999.

[10] V. Ramakrishnan, Y. Zhuang, S.Y. Hu, J.P. Chen, C.C. Ko, Ben M. Chen, and K.C. Tan. Development of a web-based control experiment for a coupled tank apparatus. In *Proc. of IEEE American Control Conference*, pages 4409–4413, Chicago, June 2000. vlab.ee.nus.edu.sg/vlab/control.

[11] T.F. Junge and C. Schmid. Web-based remote experimentation using a laboratory-scale optical tracker. In *Proc. of IEEE American Control Conference*, pages 2951–2954, Chicago, June 2000.

[12] J. Apkarian and A. Dawes. Interactive control education with virtual presence on the web. In *Proc. of*

IEEE American Control Conference, pages 3985–3990, Chicago, June 2000. www.controlab.com.

[13] G. Choy, D.R. Parker, J.N. d'Amour, and J.L. Spencer. Remote experimentation: a web-operable two phase flow experimnet. In *Proc. of IEEE American Control Conference*, pages 2939–2943, Chicago, June 2000.

[14] Marco Casini. *Designing a Tele Laboratory for control of dynamic systems through Internet*. Master thesis (in italian), Università degli Studi di Siena, June 1999.

[15] Marco Casini, Domenico Prattichizzo, and Antonio Vicino. The automatic control telelab: a user-friendly interface for distance learning. Technical report, Università di Siena, Italy, 1999.

[16] Neil Kolban. Webcam32 - the ultimate webcam software. Technical report, http://surveyorcorp.com/webcam32.