TELEOPERATION OF AN INVERTED PENDULUM THROUGH THE WORLD WIDE WEB

J. Sánchez, F. Morilla, S. Dormido

Dpto. de Informática y Automática, UNED, Avda. Senda del Rey nº 9, 28040 Madrid. Spain. Phone: 34-91-3987146 Fax: 34-91-3986697, E-mail: {jsanchez, fmorilla, sdormido}@dia.uned.es

Abstract: The Departamento de Informática y Automática of the Universidad Nacional de Educación a Distancia (UNED) is working in the development of new paradigms of Internet-based laboratories. In this approach, a WWW browser is the only tool needed to conduct the remote practical experiences with real or simulate systems. An example of remote on-line control of an inverted pendulum using a Java-enabled browser is presented in this paper. *Copyright © 2001 IFAC.*

Keywords: Teleoperation, laboratory education, teaching, remote control.

1. INTRODUCTION

Now, modeling and dynamic simulation are considered the basic tools to verify the theoretical background acquired during the study of the subjects. These two tools allow to practice with certain objects that are unreachable for the students (inappropriate timetable, long distances between home and university) or for the university departments (high prices, no room for new didactical setups, scarcity of staff) (Kheir *et al.*, 96).

The experimentation with a physical system is irreplaceable for simulations or training simulators (Poulis and Pouliezos, 1997; Cooper and Fina, 1999). A student working with a real experiment has feelings that are not the same ones that those obtained against simulations. Many times the student thinks that working with a real plant instead of a simulation has more profits, but it is not true. In most of the activities with real plants, the student calculates and enters a set of parameters in a graphical user interface and obtains some numerical or graphical results. If the plant model is right, these results do not have to be different with those obtained of working against the real plant. But in certain type of hands-on exercices, as the laboratories of Digital Electronic, Computer Architecture, Process Control or Robotics, the student's activity is not limited to the manipulation of parameters. The student designs systems by using graphical environments in those a symbol or icon represents a real or simulated object. Therefore, the design stage is independent of the object regarding its existence or not.

But working with real plants produces the appearance of phenomena (nonlinearities, saturations of the actuators) or accidental situations (electrical and mechanical irregularities). These phenomena are not easily ported to a simulation due to their stochastic nature or for simplification. But these phenomena contribute to improve the practical experience acquired for the student. *The practical education needs the support of errors and irregularities of the mechanical, electrical or chemical systems in opposition to the icons and ideal environments represented in a computer screen.*

But the current face-to-face laboratories present a set of factors that avoid their application before certain situations. These factors are:

- *A great number of students enrolled in a course*. That situation obligates to fix several shifts and timetables. But that does not benefits students (lack of additional time to fulfill the projects, few possibilities to practice with new systems) and teachers (neglecting the research activities or taking time away from teachers' leisure in order to attend to the labs)
- *The distance education model*. Distance students have to move for several days to the labs because of practical experiences can take many hours and even days to perform.
- *The lack of financial resources*, in many cases, forces to that the number of available experiments in the laboratory to be scarce. This point, the student enrolment, and the reduced space of the labs cause that the hands-on exercises can not be developed in the way that the students and teachers want.

2. THE TELEOPERATION AS A SOLUTION

To solve or to play down the problems outlined previously is necessary to appeal to the *teleoperation*. This concept has already been studied for some years, but that now it is accessible to students thanks to the current diffusion of the Internet network in our society. In general, the teleoperation term is the possibility to manipulate and to control at distance certain resources with the same possibilities that one person would have if he/she were operated in a local, manual, and direct way. Obviously the definition of the term includes many cases: from the control of a guided car using a cable until the control by means of radio frequency of the supervision vehicles on the Mars surface, or even the control of our bank account by means of WAP mobile phones.

In our context, the teleoperation is the access to the physical elements of a laboratory using the resources of the Internet. But this access has to guarantee an adequate level of presence in order to allow users to develop the hands-on activities with the same validity to practice in the laboratory rooms.

If the teleoperation term is applied to the Internet-based access to the face-to-face laboratories without spatial and temporal restrictions, the concept of the *WWW-based remote laboratory* or *telelab* arises (see Figure 1).

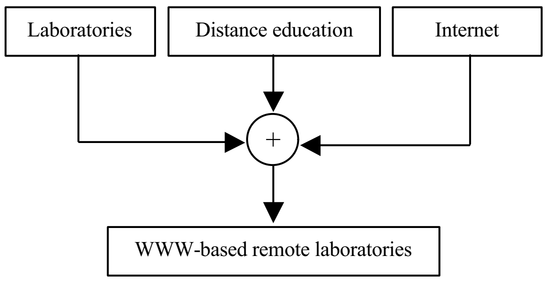


Fig. 1. Telelab is the mixture of realities and necessities

But, why is the telelaboratory a possible solution to the problems in the university laboratories? The remote access via the Internet facilitates that:

1. Laboratories have 24-hour-a-day access, every day of the year.
2. Students do not have to move to the lab for the realization of the hands-on practical exercises.
3. Bigger optimization in the use of the lab resources.
4. Access to different types of experiments, with independence that the department resources are scarce.
5. Students can prepare their experiments in advance in case that the presence in the laboratory is mandatory.
6. The learning process improves thanks to a 24-hour link between practical experience and theoretical knowledge.

In accordance with the modern society in which we are living, a consequence of the existence of remote laboratories that deserves special attention is the creation of *remote laboratory experimentation networks among universities*. If a single remote laboratory solves many of the problems outlined previously, the creation of consortia increases these benefits notably. And these benefits redound to the two parts: (1) students would have a complete set of practical activities regardless of the university resources (personnel, equipment, etc.); (2) teaching staff would have different platforms to support their lectures but reducing costs of acquisition and maintenance. Additionally, and thanks to this kind of consortia, the remote access to different didactical setups allows universities a better integration of the practical experiences in the educational curriculum (Antsaklis *et al.*, 1999).

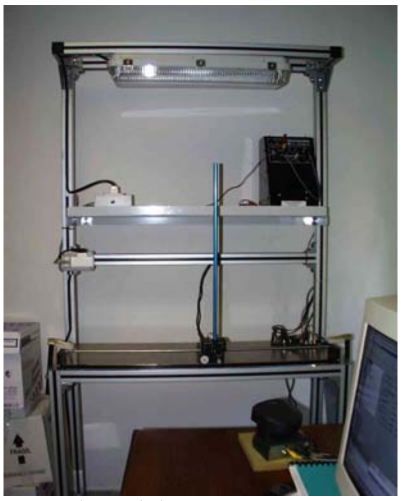


Fig. 2. Inverted pendulum

3. AN EXAMPLE OF TELEOPERATION: REMOTE CONTROL OF AN INVERTED PENDULUM

To distinguish this contribution from other similar works (Gillet *et al.*, 2000; Zolnay *et al.*, 2000) and to highlight the main differences and analogies, next some of the design options of this approach are enumerated:

- *Easy to use and understand.* The experimentation interface is very friendly and has already tested in other simulation environments developed in the department (Sánchez *et al.*, 2000).

- *Multiplatform software client.* The graphical experimentation interface is a 100% pure Java applet. This allows students to conduct the experiments from any platform just using a Java-enabled browser.

• *Global access to the remote laboratory by using a data-sharing approach*. The exchange of information between the server and the clients is carried out by means of a small application protocol that allows:

- To know user's type: student or teacher,
- To transmit the parameter and data streams closing the tuning loop across the Internet, and
- To manage the control loop (stop, start, new sampling time, etc.) from the client interface.

This data approach against the terminal-like approach gets a sustained and flexible exchange of information, minimizing the bandwidth requirements at the same time. However, the effort of software development is bigger.

• *On-line access to the physical resource*. The interaction with the system is dynamic and in realtime. Students are not just expected to tune sliders and controllers, to run the simulation, to examine the scopes, and to repeat again all the steps if they want to change some data. During the experimentation stage, every change in the input variables is immediately shown in the experimentation GUI. So, users can visualize on-the-fly how the system behavior evolves according to the values of the interactive variables. But, when the experimentation stage is over, it is possible to download and picked up the results in a file. Later on, the student could carry out a quantitative analysis of his/her experiment since the file has the samples of all the system signals. The existence of a master client has been considered. Other clients have to wait but they can visualize what is happening in the system while the master client conducts a practical exercise.

• *The control loop implements at the server side.* The real-time control loop is running on the computer that stands for the communication and operation interface with the didactical setups. The control loop is closed in the server and never across the Internet as it happens with the tuning loop.

This design approach involves two information loops: the *synchronous control local loop*, and the *asynchronous tuning loop* (see Figure 3). The last one is closed across the Internet and it takes charge of tuning and managing the control local loop and the plant state.

• *Generation of disturbances in the process variables for validation purposes*. There are two techniques to change the system signals: on-line modification due to the user's actions (movement of a slider in the process diagram) or disturbances preprogrammed by the tutor/instructor for the experiment. In this last case, the disturbances are saved in the experiment definition file (file of parameters).

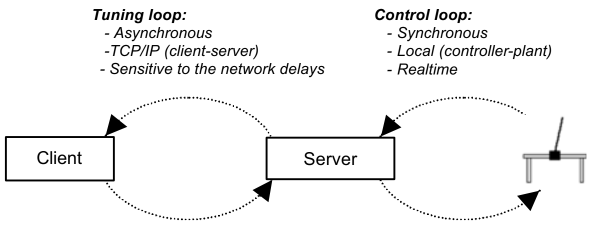


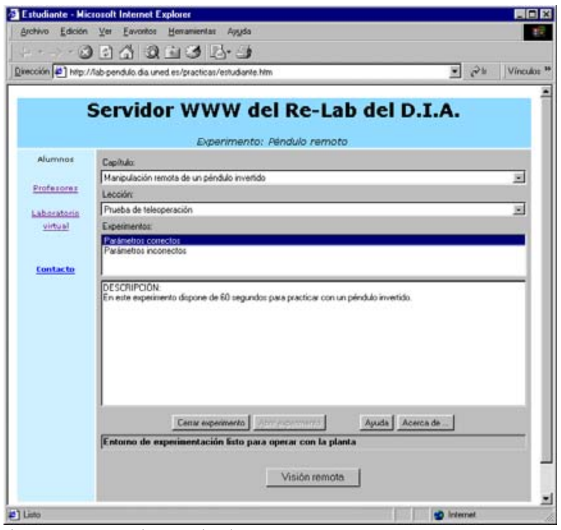
Fig. 3. Teleoperation diagram with two information loops: tuning and control.

• *Replacement of the controller*. In this environment, the software package used to design the control structure determines the change of the control law. This software is Matlab/Simulink and the Quanser WinCon environment. Different controllers can be placed in a Simulink block diagram and the instructor could choose one or another by means of the file of parameters.

4. ELEMENTS OF THE ENVIRONMENT

4.1 The client-side

At the client side, the experimentation interface is composed of two parts: the graphical user interface (GUI) and the visualization system. The GUI is a Java applet with two parts: the browsing window (see Figure 4), and the experimentaction window (see Figure 5).



http://lab-pendulo.dia.uned.es/practicas/estudiante.htm

Servidor WWW del Re-Lab del D.I.A.

Experimento: Péndulo remoto

Capítulo: Manipulación remota de un péndulo invertido

Lección: Prueba de teleoperación

Experimentos:
Parámetros correctos
Parámetros incorrectos

DESCRIPCIÓN:
En este experimento dispone de 60 segundos para practicar con un péndulo invertido.

Entorno de experimentación listo para operar con la planta

Fig. 4. Browsing window

The browsing window.

The browsing window allows the experiments to be adjusted to the hierarchical structure of a textbook or a course. Three browsing levels are considered: chapters, lessons, and experiments. In this way, the environment could have i chapters, each chapter j lessons, and each lesson k experiments (these experiments can be different, using either the same plant type or different plants). In this simple way, the tutor/instructor will be able to tailor the environment for teaching a course with a certain profile, and the student/operator will be able to select an experiment among all the existent ones, according to the concepts or situations that he/she wants to study or to observe. This window is setup with a plain text file (the browsing file).

The graphical experimentation interface

The experimentation graphical user interface (EGUI) for the remote control of the inverted pendulum is shown in Figure 5. The EGUI is characterized to offer all the elements to carry out a realtime and interactive supervision of the pendulum.

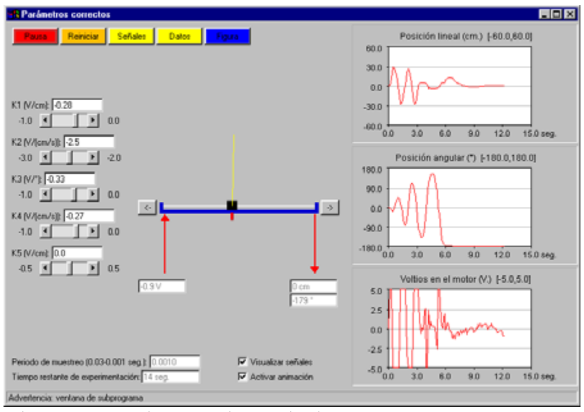

Parámetros correctos
Pausa
Reiniciar
Señales
Datos
Figura
Posición lineal (cm.) [-60.0,60.0]
Posición angular (°) [-180.0,180.0]
Voltios en el motor (V.) [-5.0,5.0]
Visualizar señales
Activar animación
Advertencia: ventana de subprograma

Fig. 5. Experimentation window

The EGUI is composed of the following parts: the process diagram, the control panels, three scopes, a multisignal scope, and the historical log. The process diagram is made up of a graphical diagram of the process with alphanumerical visualization of the most important signals and units, plus an outline of the control strategy, allowing the access to the parameters and the modes of the controllers. The control panels are composed of three types of elements (buttons, sliders, and fields) and can be grouped in three categories: main control panel (located at the top of the interface), the interactive variable panel (sliders and alphanumeric fields to modify the value of the signals and the set point), and the controller panel.

Besides scopes displaying the pendulum state with all the relevant signals, the IGUE has an animation of the mobile parts of the pendulum to enhance the user perception. This way, the user has a quantitative (scopes, numeric fields) and qualitative (animation, video) view of the physical system.

With the purpose of allowing the tutor to limit and guide the user's actions, the way in which the user interacts with these panels can be configured by means of files of parameters. There are three possible configurations for each variable: totally hidden, visible but not modifiable, visible and modifiable.

The total time that a user has to fullfil the experience depends of the type of experiment chosen in the browsing window. This time is fixed for the tutor in the file of parameters, file located in the remote server.

4.2 The server-side

There are several applications running at the server side (see Figure 6). These are:

- *A HTTP server*. It provides the HTML pages, the experimentation applets, and the files of parameters. Also it controls the access to the pages in function of user's type and it supplies the access to the remote visualization environment.

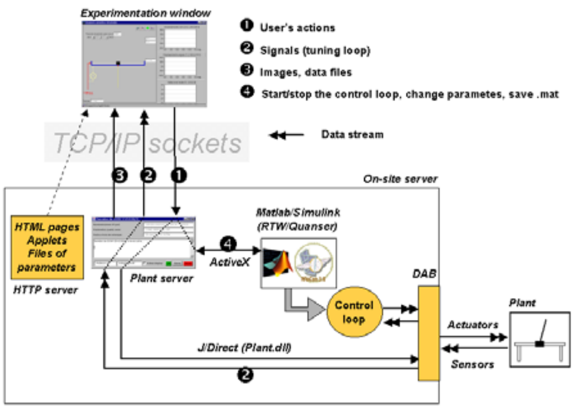


Fig. 6. Teleoperation diagram

- *A concurrent server, named plant server*, developed in Java, with a double purpose. First, it interacts with the data acquisition board to supply a stream with the plant state to the tuning loop in order to update the experimentation user interface. Second, it manages the Simulink+Wincon environment through the Matlab workspace and, so, the control loop is managed (starting, stopping) and certain parameters can be changed (controller parameters, sampling period). The dialog between this server and the experimentation applet is based on an application protocol. Using this protocol, the client can send commands and inquiries to the server in order to control the experiment and to receive the answers.

- *WinCon 3.0* is a realtime Windows application that runs Simulink generated code using the Realtime Workshop to achieve digital realtime on a PC equipped with a data adquistion board. The board (MultiQ3), the inverted pendulum, and the WinCom software are made by Quanser Consulting.

- *The Matlab/Simulink environment.* The design and construction of the control loop is carried out with the Simulink blocks and the WinCom's Simulink block library. The Simulink model is managed from the Matlab workspace by means of Matlab commands, and, therefore, the supervision of the control loop and their parameters is done easily.

The communication on Windows between the Java language and Matlab has been carried out using the ActiveX Automation protocol. Using the Matlab type definition file, named mlapp.tlb and located in the folder of the Matlab distribution, to generate automatically a group of Java classes wrapping the ActiveX-Matlab interface is an immediate action from the Microsoft Visual Java++ environment (MVJ++). The names of these classes are DIMLApp.class and MLApp.class, and the signatures of the available methods are:

```
public abstract void MinimizeCommandWindow();
public abstract void MaximizeCommandWindow();
public abstract void Quit();
public abstract void GetFullMatrix(java.lang.String, .....);
public abstract void PutFullMatrix(java.lang.String, .....);
public abstract java.lang.String Execute( java.lang.String );
```

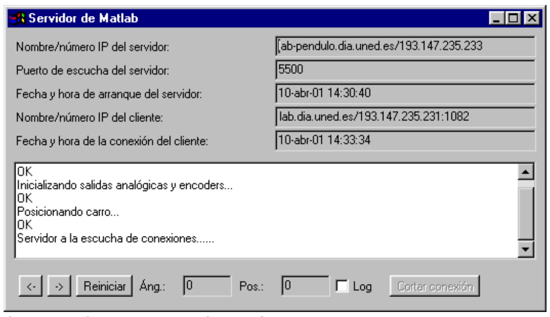
So, the Java code can communicate with the Matlab workspace using these methods and, therefore, Java can manage the control loop generated with Simulink/WinCom. Hence, the employment of the MVJ++ environment is inevitable to carry out the communication with Matlab through ActiveX

A dynamic library, Plant.dll, has been written to program from the MultiQ3 board (initialization of encoders and A/D outputs, encoder reads, A/D output/input operations, etc.) from the Java server. Because the MVJ++ is not compatible with the Sun's JNI specification (Java Native Interface), it was necessary to use the Microsoft's J/Direct specification in order to access to the Plant.dll library from the Java code.

The use of TCP sockets between the applets and the server is a highly flexible and smart solution to exchange information, and TCP level is reasonable efficient in network utilization. Unfortunately, TCP can also cause delays when an application is sending many small packets in a short amount of time. This makes the information view pretty choppy. Since the server sends to the applet a 50 bytes-sized data packet in response to a command, the NAGLE algorithm has been disabled (RFC896) in the TCP level. In this way, an improvement of the 60-90% in the data transmission is reached, and, the information exchange between applet and server is carried out in a continuous and cadenced way.

The plant server

The server interface is composed of three parts: the information fields, the message area that displays the dialogue between applet and server, and the button area (see Figure 6).



Nombre/número IP del servidor:	lab-pendulo.dia.uned.es/193.147.235.233
Puerto de escucha del servidor:	5500
Fecha y hora de arranque del servidor:	10-abr-01 14:30:40
Nombre/número IP del cliente:	lab.dia.uned.es/193.147.235.231:1082
Fecha y hora de la conexión del cliente:	10-abr-01 14:33:34

```
OK
Inicializando salidas analógicas y encoders...
OK
Posicionando carro...
OK
Servidor a la escucha de conexiones......
```

Áng.: 0 Pos.: 0

Fig. 6. Plant server interface

The plant server is designed to attend concurrently multiple connections, although in this approach the existence of a unique master client has been considered. The server operation is very similar to any concurrent server: a parent process listens to the incoming connections in a port; once established the TCP connection with the experimentation applet, the parent process forks a new thread, named *Connection*, in order to attend this new client connection (see Figure 7). At the same time that the thread is created, the Matlab/Simulink/WinCom is open in the server and the Simulink model (the control loop) is loaded. From now on, this thread will receive the commands and inquiries from the applet and transmit them to the Matlab workspace, giving back the answers to the applet. So, the tuning loop is closed across the network.

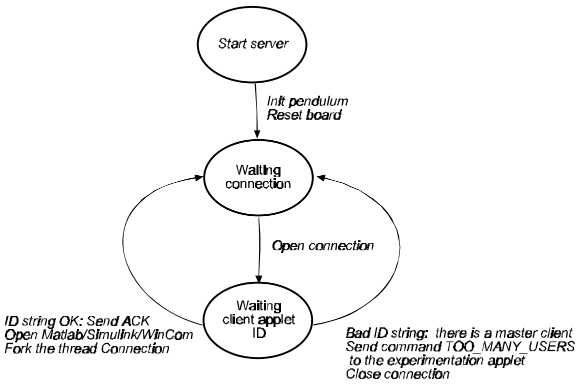

Start server
Init pendulum
Reset board
Waiting connection
Open connection
Waiting client applet ID
ID string OK: Send ACK
Open Matlab/Simulink/WinCom
Fork the thread Connection
Bad ID string: there is a master client
Send command TOO_MANY_USERS to the experimentation applet
Close connection

Fig. 7. State diagram of the server

5. REMOTE VISUAL SUPERVISION

In order to provide a 24-hour-a-day access to the lab, the hardware, two Java applets, and a server side Java application have been developed for the remote control of a system composed of lights and a video camera commanded by a computer. The system design allows the teacher to preconfigure several camera setups. So, students concentrate their attention on specific points of the physical system that they are controlling across the Internet.

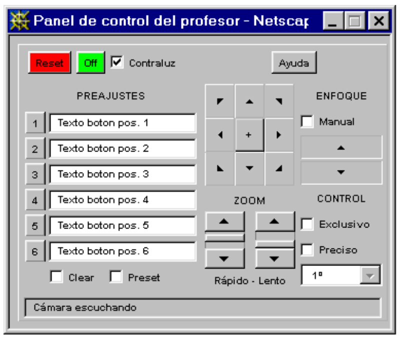


Fig. 8. Teacher applet

There are two client applets to control the camera: teacher and student. The teacher client applet (Figure 8) has a full functionality. With this applet, the teacher can configure the camera interactively according to the necessities of the conducted experiments: plant location, lighting, zoom for emphasizing some details, etc. The student applet has fewer functions than the teacher one. As it can be appreciated in Figure 9, the interface lets user program the camera with one of the six setups or take it to the home position.

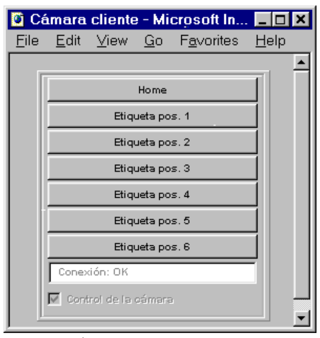


Fig. 9. Student applet

In accordance to the hardware requirements fixed for the development of the visualization environment, SONY and CANON have cameras that adapt perfectly to these specifications. After analyzing each one of them, the SONY camera EVI-D30/D31 was chosen. This camera uses a protocol called VISCA and can be full-controlled using RS-232C communications.

Like the control of the pendulum, an application protocol has been developed for the control of the camera. The server, developed in Java, receives the applet commands and inquiries and translates them to the VISCA protocol, sending this information to the camera through the on-site server serial port.

The video grabbing and transmission is carried out by means of an AXIS hardware video server. This solution discharges the server computer of the video tasks, avoiding to disturb the control local loop.

Evidently, the construction of a remote experimentation environment involves not only the physical independence of the student, but also the temporal one. For this reason, a prototype for the remote control of lighting has been designed so that, the view of the physical system is guaranteed at the very same instant in which a TCP connection is established to begin an experiment on a real plant. The prototype is based on electromechanical relays, separating the control circuit (D.C. supplies by the cards D/A) from the power circuit (A.C. to turn on, in this case, the lights).

6. CONCLUSIONS

Today, Internet is the bottleneck for this kind of applications. To reduce the communication overhead and to use augmented reallity techniques are some solutions to get better communication performance.

The system is designed to be modular. So, the Matlab/Simulink/WinCom environment allows to integrate another pilot plant into the system easily.

The use of files of parameters is a smart solution to manage different experiments with an only plant.

REFERENCES

Antsaklis, P., T. Basar, R. DeCarlo, N. Harris, M. Spong and S. Yurkovich (1999). Report on the NSF/CSS WorkShop on new directions in control engineering Education, *IEEE Control Systems Magazine*, **Vol. 19, nº 5**, pp. 53-58.

Cooper, D. and D. Fina (1999). Training simulators enhance process control education. In: *Proceedings of the American Control Conference*, pp. 997-1001.

Gillet, D., C. Salzmann and P. Huguenin (2000). A distributed architecture for teleoperation over the Internet with application to the remote control of an inverted pendulum. In: *Second Nonlinear Control Network (NCN) Workshop*, Paris, France.

Kheir, N.A., K.J. Ämstrom, D. Auslander, K.C. Cheok, G.F. Franklin, M. Masten, and M. Rabins (1996). Control system engineering education. *Automatica*, **32**, 147-166.

Poulis, D. and A. Pouliezos, (1997). Computer assisted learning for automatic control. In: *The 4th Symposium on Advances in Control Education*, pp. 181-184.

RFC986. In: *http://www.faqs.org/rfcs/rfc896.html*.

Sánchez, J., F. Morilla, S. Dormido, J. Aranda, and P. Ruipérez (2000). Conceptual learning of control by Java-based simulations. In: *IFAC Symposium on Advances in Control Education*, Gold Coast, Australia.

Zolnay, A., A. Lassó, H. Charaf and I. Vajk (2000). Configurable remote, platform independent control system. In: *IFAC Symposium on Advances in Control Education*, Gold Coast, Australia.