Int. J. Engng Ed. Vol. 21, No. 5, pp. 790–797, 2005
Printed in Great Britain.

0949-149X/91 $3.00+0.00
© 2005 TEMPUS Publications.

Platform for Distance Development of Complex Automatic Control Strategies using MATLAB*

ALDO R. SARTORIUS CASTELLANOS, LUIS HERNÁNDEZ, IVÁN SANTANA, ERNESTO RUBIO
Universidad Central 'Marta Abreu' de Las Villas, Departamento de Automática y Sistemas Computacionales, Carretera a Camajuani Km. 5½, Santa Clara, Villa Clara, Cuba.
E-mail: sartorius@uclv.edu.cu

RAFAEL ARACIL
Universidad Politécnica de Madrid, División de Ingeniería de Sistemas y Automática, José Gutiérrez Abascal 2 E-28006 Madrid, Spain

In this paper is presented a Distance Laboratory System (DLS), which is a distance laboratory focused in automatic control study developed in Universidad Central 'Marta Abreu' de las Villas. The main objective of this laboratory is to permit the users to learn to adjust predefined controllers and to design their own controllers for testing them later over a set of physical devices through the Internet and to analyze their performance. The targeted controllers (designed by the users) can use S-functions created using C language, permitting the creation of complex controllers in an easy and fast way. The DLS uses MATLAB-Simulink for practice processing and the Real Time Workshop Toolbox (RTW) such as real time kernel. The DLS permits integration of new processes for carrying out controller tests. At present three devices are available: a DC motor and a robot manipulator in Cuba as well as an electropneumatic cylinder in Spain. Since the last year the DLS has been used in identification and control theory undergraduate courses in Universidad 'Marta Abreu' de las Villas and Universidad de Cienfuegos in Cuba, as well as in postgraduate courses in México in the fields of advances control theory and robot manipulator control.

INTRODUCTION

ENGINEERING EDUCATION has had a dramatic change in the last few years mainly because of the development of computational tools such as MATLAB and Simulink, which permit the solution of complex problems in an easy way. Due to this change it has been possible to create more stimulating and interesting laboratory practices; for example in [1] are presented some experiments that use MATLAB/Simulink which support control theory and linear systems courses, while a laboratory for temperature control using MATLAB/Simulink is presented in [2]; there the students can carry out experiments with different types of controllers in a simulated form and later prove them in a real device and see their performance.

The development of Web technologies has permitted incorporation of these computational tools in distance education programs, with automatic control being one of the technical areas that has greatly exploited these new technologies for developing tools to facilitate distance learning [3]. This permits the creation of tutorials like the one shown in [4] and the development of Web-based distance laboratories, which can be classified into two great classes: virtual distance laboratories and real distance laboratories. Virtual distance laboratories are based on a physical system simulation in a remote way. Through computer animation and the use of specialized software, these physical systems can be represented in a graphic and an analytical form. In [5] is shown a virtual laboratory that uses two plug-ins: one for visualizing the data in the client browser and another for communicating with MATLAB in the client computer while the experiments are in the system server; in [6] is shown a virtual laboratory which uses the tools that MATLAB Web Server gives for analyzing the performance of a direct current motor in a remote way.

Real distance laboratories are laboratories where the users can interact with real devices in a remote way. Usually the users can change some control parameters, make experiments, see the results and download the experiment data through a Web interface. These types of experiments are used, for example in [7], where a laboratory for the distance control of a direct current motor is presented; in [8] where a laboratory for controlling an optical tracker device is described and in [9] which provides some level control experiments in tanks.

* Accepted 10 May 2005.

As the Web technology has been providing more options, distance laboratories have been evolving mainly in the form of giving feedback information to the users and in the use of more complex devices for controlling; for example, the system presented in [10], where it is shown a laboratory for controlling an inverted pendulum and a torsion disk. In [11] and [12] several experiments are presented for controlling mechanical systems while in [13] a laboratory is presented for controlling the level in interconnected tanks. This evolution towards more complex devices has also needed the implementation of more complex controllers to fulfill the control objectives like those shown in [14] where some types of PID controllers are used as well as inverse dynamic controllers for controlling different mechanical devices or those shown in [15] where auto tuning PID controllers and predictive controllers are given for carrying out experiments with a heat exchanger.

Nevertheless the tendency in the development of distance laboratories is letting the users to develop their own controllers in a remote way; in such laboratories the complexity in the hardware and software design is drastically increased. An example of this is the case of the system presented in [16], where the users can develop controllers using scripts in MATLAB language for controlling an inverted pendulum. In the laboratory proposed in [15] the users can develop their own controllers using MATLAB-Simulink in the client computer; the final block diagram is compiled using the tools provided by Real Time Workshop (RTW) for generating an application able to communicate with the remote server which transmits the process data that is being controlled in real time. Another approach is presented in [10], where it describes the use of MS NetMeeting program for accessing to a computer with MATLAB/Simulink in a remote way, which is connected to an inverted pendulum. In this way the users can develop controllers on the server side for carrying out the experiments. In [17] is presented a distance laboratory that allows the creation of controllers in a remote way using the MATLAB/Simulink environment for carrying out some process control experiments, while in [18] the same approach is used, but for controlling several mechanical devices in a remote way.

This article describes the design and implementation of a distance laboratory focused on the automatic control study that has been developed in Universidad Central 'Marta Abreu' de las Villas. The Distance Laboratory System (DLS) lets the users change the references, modify the control parameters and design their own controllers in a simple way using very well known tools in the automatic control field such as MATLAB and Simulink. One of the main features of DLS is that the remote users can create controllers using the blocks given by Simulink, as well as to incorporate S-functions defined by the users in C language. This makes it possible to test complex controllers in a simple way using the most modern techniques of computer-assisted control systems design.

CHARACTERISTICS OF THE SYSTEM

The DLS is a distance laboratory whose main objective is to allow the users to learn how to adjust predefined controllers and to design their own controllers in order to test them in real devices through the Internet

Common characteristics of distance laboratories

Like other distance laboratories, the DLS exhibits some features in common with most distance laboratories in operation at present:

- *Ease of use.* For using the DLS the users should only have some knowledge about robotics and control systems, such as kinematics and dynamic robot manipulator modeling as well as controller adjustment and design. In this way the users are focused on reinforcing these topics and avoid all implementation and operation problems of devices used in the practices.
- *Availability.* Web-based learning systems should be available 24 hours a day. This implies that the system should have self-protection rules for accomplishing this requirement. All the experiments must be equipped with hardware and software devices to prevent damage to the components and the people working in the laboratory.
- *Accessibility.* Since the DLS is mounted on a Web-platform, the users can access the system from any place in the world. Thus, only one computer connected to the Internet and a browser (Netscape Navigator or Microsoft Internet Explorer) are needed.

Specific characteristics of the DLS

Apart from the characteristics described above, which are common to most of the distance laboratories, the DLS has some additional characteristics which are given below:

- *Easy and fast user interface.* A very important part in the development of a Web-based learning system is the user interface. The main function of this part of the system is making the practice order and sending it to the Web server. The DLS user interface is based on HTML pages that use JavaScript and ASP functions; this allows the users to access the system fast and without downloading or installing any additional software. The system also has help pages which provide the users with information about the mathematic modeling of devices used in the practices, manufacturer data and controllers adjustment as well as any other relevant information for the practices.
- *Controller development using MATLAB and Simulink in a remote way.* One of the most important features of the DLS is that it allows

the users to design their own controllers using the MATLAB/Simulink environment These programs are standard tools in the automatic control field, so the users don't need to waste time learning new programming languages for implementing a new controller; they only need some basic knowledge of MATLAB-Simulink environment. Through the Simulink graphic interface, a large number of blocks can be chosen and connected in a very simple way, allowing the users to create analog, digital or hybrid controllers very fast. Furthermore, the DLS lets the users, as an option, create complex controllers including S-functions defined by the users. These S-functions must be implemented using the C programming language, a very powerful and widely known language that almost all engineering students are familiar with at present.

- *Controller type*. All the experiments in the DLS can be controlled in two different ways: (a) with predefined controllers or (b) with controllers defined by the users. In the first case the users should assign values to the controller gains; for example, the user can choose a PID controller and change the proportional, integral and derivative gains parameters to analyze the system response. In the second case, the users can design their own controllers (as previously explained) and send it to the DLS where it will be implemented and used for controlling the experiment.
- *Reference change*. The system allows the experiment's references to change for verifying the controller performance in the presence of different input signals.

GENERAL OPERATION OF THE DLS

The DLS is divided into three parts:

1. User interface.
2. Practices, order management.
3. Practices, processing.

Figure 1 shows the interaction among these three parts. The users interact with the system through the Internet. After accessing the system website, the desired practice is chosen. There the user can fill all the data in the form associated to the practice in a correct way and finally choose whether to carry out the practice in a simulated or a real way. The CGI located in the web server receives the data and sends them to the Practice Management Server (PMS) as a new order. The PMS verifies which workstation can carry out the practice order and when found, the order is extracted from the list and sends the order to the Practice Management Client (PMC) installed in the workstation. When the order arrives at the PMC, the data are processed and the practice is carried out using MATLAB/Simulink together with RTW Toolbox. Once the practice has been processed, the result is transmitted inversely with

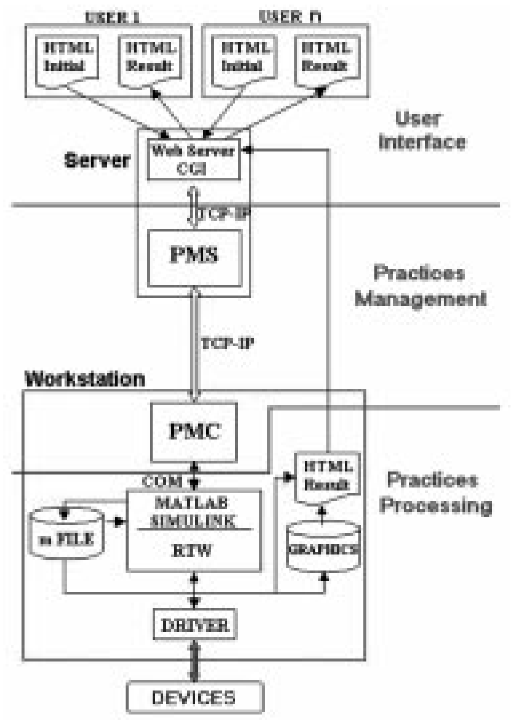


Fig. 1. General functioning of the DLS.

respect to the order that brought in the practice response. The response is a Web page showing the processing results.

The web server and the PMS are installed in a computer with Windows 2000 as operating system. The workstations need to have the PMC installed as well as MATLAB and Simulink for the practices processing. The workstations use Windows 98 as the operating system. At present the system has three processes in which the users can test controllers in a remote way: a DC motor, a robot manipulator and an electropneumatic cylinder. A DC motor is a very simple and single-input, single-output process (SISO) that enables identification, position control and velocity control experiments without loading disturbances; the electropneumatic cylinder is also a SISO process but has a nonlinear behavior, where it is possible to carry out piston position control practices. Finally the robot manipulator is a system with multiple inputs and multiple outputs (MIMO) that exhibits more coupled and stronger nonlinear dynamics and which enables more complex control position and tracking trajectory practices.

DISTANCE LABORATORY SYSTEM USE

When the users access the DLS website, they must register in the system giving their user's name and password. This permits the system to have two types of users: practice users and administrators. Once inside the system, the practice users can carry out several operations, such as seeing the theory pages, contacting with the system developers as well as seeing the practices available in the system.

All these web pages are located in the system web server, which is common for all the practices. After the user chooses one practice, he is shown a web page containing the practice diagram block, an explanation about the practice symbols, links to theory web pages associated with the practices as well as a form where the user can modify the controller parameters, the sampling time, the reference value or type or create a controller defined by the user.

Practices with a predefined controller

In this type of practice, the users only need a Web navigator (Netscape, Internet Explorer or any other) to access the DLS website. This type of page is shown in Fig. 2, where gives a practice for testing decoupled PID controller performance in a robot manipulator with two degrees of freedom. In addition, the users can choose two execution ways: (a) a simulated way, where it is simulated the execution and is obtained an idealized response of the practice or (b) a real way.

Practices with a controller created by the user

When the users access some practices in which they can define the controller that they will use, it is shown a page as the one given in Fig. 3. From this page the user can download a Simulink file containing the practice diagram block. For carrying out these types of practices the users need to have the MATLAB-Simulink software installed in order to modify the downloaded Simulink file, as it will be explained in the creation of controllers section.

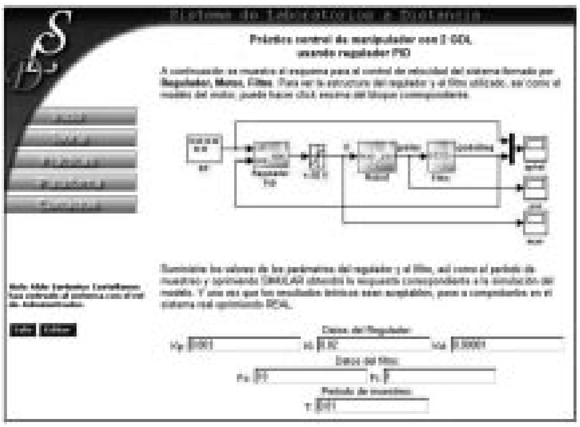


Fig. 2. Web page and form for carrying out the practice with decoupled PID controller.

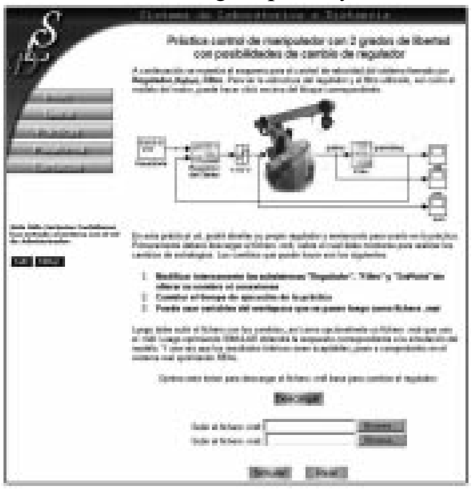


Fig. 3. Web page and form for carrying out practice with a controller created by the user.

RESPONSE WEB PAGES

The response web pages format for all practices is shown in a summarized form in Fig. 4; these pages contain the following fields:

1. *Warnings about practice accomplishment*. In this page section the user is informed if the practice has been successfully carried out; equally, he is informed about any error.
2. *System block diagram*. This part of the web page shows the user the block diagram of the robot-controller system used in the practice
3. *Practice data*. Both the trajectory data used in the practice and the controller parameters are shown here. For practices with controllers created by the users, this section does not appear in the response web page.
4. *Graphics*. The graphics associated with the executed practice are shown in this part. These graphics show the real and desired joint position, the joint positions errors and the real and desired Cartesian trajectory that the robot manipulator followed during the practice. Each graphic provides a zoom option for analyzing data in more detail.
5. *Data file*. In each response web page, there is a link that permits the users to download a text file that contains, grouped by columns, the

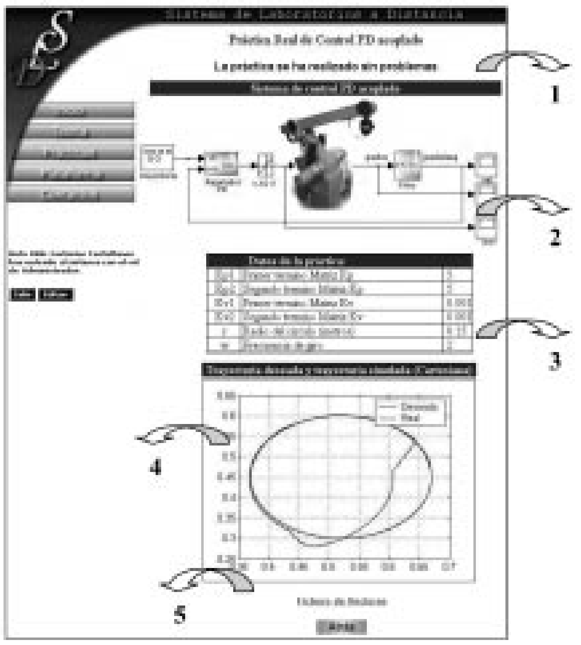
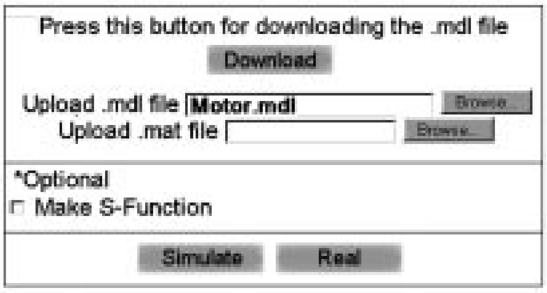


Fig. 4. Practice response web page (summarized).


Press this button for downloading the .mdl file

Download

Upload .mdl file Motor.mdl Browse...

Upload .mat file Browse...

*Optional

☐ Make S-Function

Simulate Real

Fig. 5. Form for carrying out practices with controllers created by the user using Simulink blocks.

real and desired joint position, the joint positions errors and the real and desired Cartesian trajectory that the robot manipulator followed during the development of the practice.

CREATION OF CONTROLLERS

One of the most important features of the DLS is letting the users create their own controllers in a remote way. These controllers can be created using only Simulink blocks or, as an option, a combination of Simulink blocks with functions defined by the users. Examples of creating controllers defined by the user for both cases—first in a DC motor and then for an Asea IRB-6 robot manipulator—are presented for a better understanding.

Controller creation using only Simulink blocks (DC motor)

When the user chooses a practice that permits the creation of a new controller, he or she is shown a Web page with a form such as the one in Fig. 5. In this page the user must download a Simulink file that contains the block diagram for the practice (in this case motor.mdl). In this file the user should modify the *reference* and *controller* subsystems using the MATLAB/Simulink software, without changing the input and output subsystems' names, as shown in Fig. 6, where the user uses a three sinus sum as reference into *reference* subsystem and creates a PID controller inside the *controller* subsystem.

When the modification is carried out, the user should upload the modified Simulink file to the server and decide whether to carry out a simulation or to control the real process. When the user has requested to execute the real process, the DLS first makes a system simulation and, based on the data obtained from the simulation, several tests are made to determine if the controller can be implemented in the real system. In the DC motor these tests are intended to determine if the system shows high-frequency oscillations that can be either a mechanical or a temperature risk for the motor. Once it has been determined that there are no risks for the motor, the DLS is in charge of implementing the controller, compiling the system using the RTW Toolbox and carrying out the practice in real time.

If the DLS detects that the designed controller represents a risk for the motor, the user is informed the cause and is given the simulation graphs and data produced so that he or she can inspect the system performance.

Controller creation using Simulink blocks and S-functions defined by the user (robot manipulator)

Through this mechanism the DLS allows the creation of complex controllers which use Simulink blocks and S-functions written in C language. When the user selects a practice with the possibility to create a controller and to make active the S-function creation check box, a Web page is shown that contains a form similar to the above example (Fig. 5) but with five additional fields as shown in Fig. 7.

The additional fields are the following:

- *Name.* Here the S-function name should be specified. This will be the block name defined by the user.

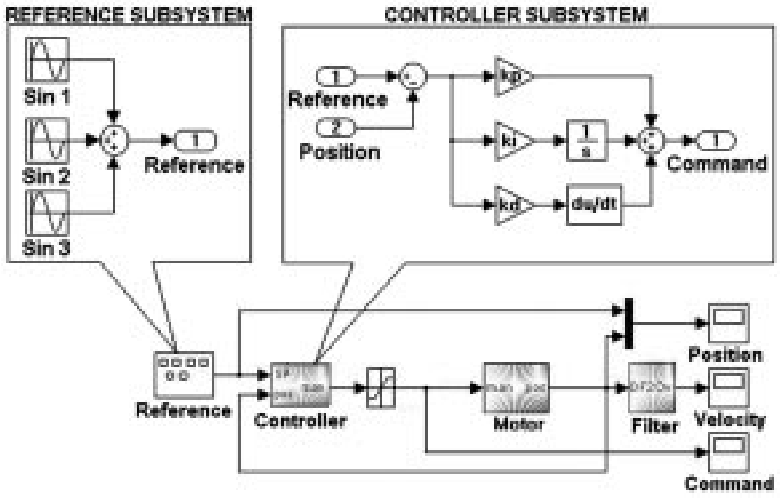
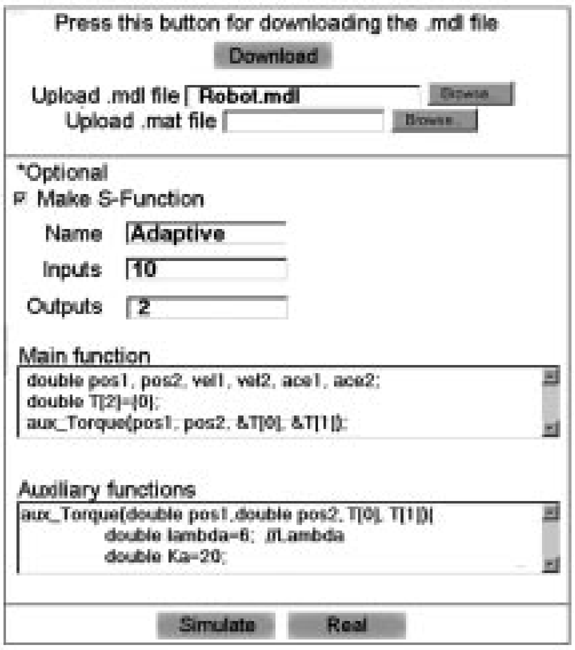


Fig. 6. Reference and PID controller implemented by the user using Simulink blocks for controlling a DC motor.



Press this button for downloading the .mdl file

Download

Upload .mdl file Robot.mdl Browse

Upload .mat file Browse

*Optional

☑ Make S-Function

Name	Adaptive
Inputs	10
Outputs	2

Main function

```
double pos1, pos2, vel1, vel2, ace1, ace2;
double T[2]={0};
aux_Torque(pos1, pos2, &T[0], &T[1]);
```

Auxiliary functions

```
aux_Torque(double pos1,double pos2, T[0], T[1])
          double lambda=6;  //Lambda
          double Ka=20;
```

Simulate Real

Fig. 7. Form for carrying out practice with controller created by the user using Simulink and S-function defined by the user.

- *Inputs.* Specifies the number of multiplexed inputs that will enter the block defined by the user.
- *Outputs.* Specifies the number of demultiplexed outputs that will get out of the block defined by the user.
- *Main function.* Here the user should write the S-function code specifying the block inputs defined by the user as u[0], u[1]. . .u[n] and the outputs as y[0], y[1] . . . y[n]. This code should be written in C language.
- *Auxiliary functions.* Here the user can write the auxiliary functions declaration, which will be called from the main function. This code should be written in C language.

As in the previous example, in this web page the user should download a Simulink file containing the practice block diagram (robot.mld in this case). In this file the user should modify the controller subsystem using the MATLAB/Simulink software without modifying the inputs and outputs name as shown in Fig. 8, where the user has implemented a PD controller with adaptive compensation proposed in [19]. This controller uses a user-defined function through an S-function called 'adaptive' in the controller subsystem. In the five additional fields of the form the user must write the S-function name (adaptive in this case), the inputs number (10 inputs), the outputs number (2 outputs) and the main function code as well as the auxiliary functions code, as shown in Fig. 7.

Once the modification is carried out and the S-function specified, the user must upload the modified Simulink file to the server and decide whether to carry out a simulation or to control the real process. When the DLS is commanded to execute the real process, it first makes a system simulation as well as several tests that, in the case of the robot manipulator, are focused on three aspects:

1. Verify that the Cartesian trajectory should not surpass the robot manipulator workspace.
2. Conform that the link positions should not exceed the mechanical limits of each link.
3. Analyze, through a signal frequency analysis, that the system should not present high-frequency oscillations that can mechanically misadjust

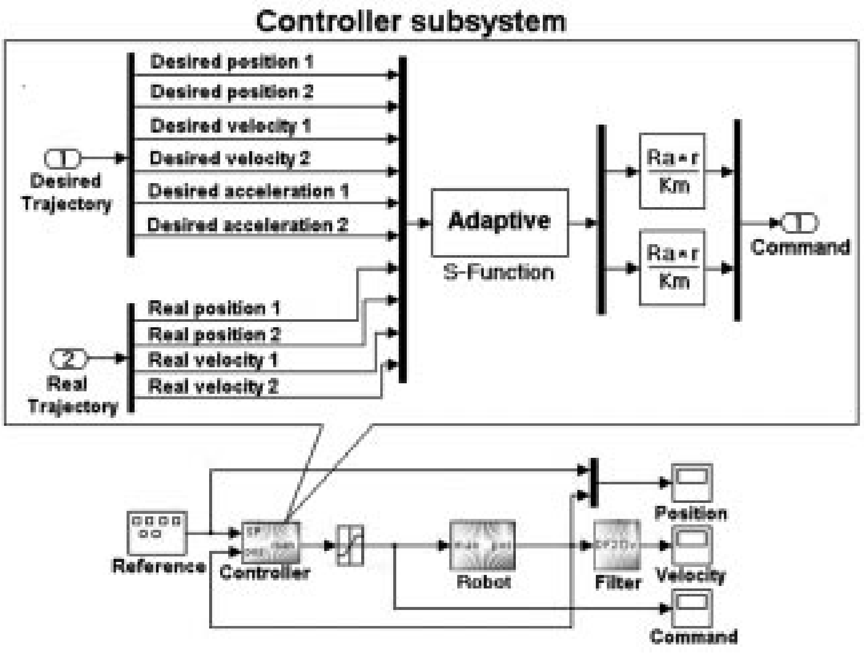


Fig. 8. PD with adaptive compensation controller implemented by the user using Simulink blocks and S-function for controlling an Asea IRB-6 robot manipulator.

the robot manipulator and/or damage its actuators.

Once these aspects have been determined, the DLS implements the controller, compiles the system using the RTW Toolbox and carries out the practice in real time. In case the DLS detects that the designed controller represents a risk for the robot manipulator, the user is informed of the cause and is given the simulation graphs and data performed so that he can inspect the system performance.

TEACHING EXPERIENCE USING DLS

Throughout the year 2003 the first DLS tests from Mexico and Spain were carried out in order to implement controllers in the devices located in Cuba. The DLS were used in postgraduate courses in the fields of advanced control theory and robotics in Instituto Tecnológico de Minatitlán, Mexico. In these courses the system worked satisfactorily but it presented a long delay time between the moment of sending the controller to the server and the moment in which the user receives the answer, so we began to work in two directions:

1. Modifying the images format in order to have a faster image load, as well as to revise the algorithms in charge of implementing the controllers so that the processing time could be reduced.
2. Giving feedback to the user with the controlled process data, so that while waiting for a response web page, he can see the process state through the control in the Java applet. This permits users to know that the information is being processed and stops them from thinking that the system is not working.

The modifications referred to in the first aspect have been completed, while those in the second aspect are being developed at present.

The present year the DLS has been used in different practices in identification and control courses for Mechanical Engineering and Automatic Engineering students in Universidad Central 'Marta Abreu' de las Villas and in Universidad de Cienfuegos, both located in Cuba. In these practices the users have shown a great interest for controlling devices in a remote way through Internet. After a brief explanation about the practice theoretical foundations, the users registered into the system and began to carry out experiments in less than five minutes. One of the most stimulating things for the students was that they could evaluate the differences between using mathematical models and real plants. On average, with 15 students per group, it is possible to carry out from 60 to 80 practices in less than 30 minutes; this shows the high rate of device utilization when accessing them in a remote way using the Internet.

CONCLUSIONS

The Distance Laboratory System (DLS) gives the users the facility of using packets such as MATLAB and Simulink together with the RTW Toolbox for the creation and testing of controllers in real devices in a remote way. Through this mechanism it is possible to implement controllers that use complex algorithms which can be easily created using the power and flexibility given by both the C language and the S-functions.

Since the system software is easily adaptable for new processes, the incorporation of new devices for testing controllers can be easily accomplished. Due to the fact that the system is in a developmental phase, the number of devices available is limited; at present we have a DC permanent magnet servomotor series PI 8.03, developed by Dynamo-SL [20] connected to an incremental encoder with 10,000 pulses per revolution of precision, an Asea IRB-6 robot manipulator with five degrees of freedom with incremental encoders with 10,000 pulses per revolution of precision and a Megliani 40.400 electro pneumatic cylinder equipped with a Festo MPYE-5-1/8-HF valve [21], and an LX-EP-40 incremental encoder [22] that has 2.44 pulses/mm. We are currently working for the future incorporation of new processes including experiments for both temperature and level control, among other processes, as well as the future incorporation of a better feedback to the user through Java technology which can make the system more interactive and more encouraging for the users.

REFERENCES

1. A. Coelho, O. Almeida, R. Sumar and J. Santos, Learning Lab for Understating Control Theory of Signals and Linear Systems, *Conference of Decision and Control*, 2001.
2. A. Leva, A hands-on experimental laboratory for undergraduate courses in automatic control, *IEEE Trans. Education*, 2003, pp. 263–272.
3. M. Casini, D. Prattichizzo and A. Vicino, E-learning by remote laboratories: a new tool for control education, *6th IFAC Symp. Advances in Control Education*, 2003.
4. D. M. Tilbury and W. C. Messner, Control tutorials for software instruction over the World Wide Web, *IEEE Trans. Education*, **4**(42), 1999, pp. 237–241.
5. C. Schmid and A. Aly, A web-based system for control engineering education, *American Control Conf.*, 2000.
6. M. Valles, A. Valera, J. L. Diez and J. L. Navarro, Setting up a virtual MATLAB control laboratory, *IFAC Workshop Education in Automatic Control*, (2001).

7. R. Puerto, O. Reinoso, R. Ñeco, N. Garcia and L. M. Jimenez, Remote lab for control applications using MATLAB, *Proc. Internet Based Control Education*, 2001.
8. T. F. Junge and C. Schmid, Web-based remote experimentation using a laboratory-scale optical tracker, *Proc. American Control Conference*, 2000, pp. 2951–2954.
9. R. Saco, E. Pires and C. Godfrid, Real time controlled laboratory plant for control education, *32nd ASEE/IEEE Frontiers in Education Conf.*, 2002, pp. 12–16.
10. N. Swamy, O. Kuljaca and F. L. Lewis, Internet-based educational control system lab using NetMeeting, *IEEE Trans. Education*, 2002, pp. 145–151.
11. W. E. Dixon, D. M. Dawson and B. T. Costic, Towards the standardization of a MATLAB-based control system laboratory experience for undergraduate students, *American Control Conference*, 2001.
12. R. M. García, F. Wornie, B. G. Stewart and D. K. Harrison, Real-time remote network control of a inverted pendulum using ST-RTL, *32nd ASEE/IEEE Frontiers in Education Conference*, 2003.
13. C. Hopp, S. Stoll and U. Konigorski, Remote control design and implementation using the Internet, *World Automation Congress*, 2002.
14. H. H. Hahn and M. W. Spong, Remote laboratories for control education, *Proc. 39th IEEE Conf. Decision and Control*, 2000.
15. D. Srinivasagupta and J. Babu, An Internet-mediated process control laboratory, *IEEE Control Systems Magazine*, 2003.
16. Y. Piguet and D. Guillet, Java based remote experimentation for control algorithms prototyping, *American Control Conference*, 1999.
17. C. Bonivento, L. Gentili, L. Marconi and L. Rappini, A Web based laboratory for control engineering education, in *Second Int. Workshop Tele-education in Engineering Using Virtual Laboratories*, (2002).
18. M. Casini, D. Prattichizzo and A. Vicino, The Automatic Control Telelab: a user friendly interface for distance learning, *IEEE Trans. Education*, 2003, pp. 252–257.
19. J. J. Slotine *Applied Nonlinear Control*, Prentice-Hall, (1991).
20. Dynamo Slaven (2004). http://www.dynamo-slaven.com
21. Festo pneumatics (2004). http://www.festo.com
22. Unimeasure (2004). http://www.unimeasure.com

Aldo R. Sartorius Castellanos graduated from Universidad Veracruzana, Mexico (Electromechanical Engineering) in 2000. He has an M.Sc. in Automatics (Computational Systems) from Universidad Central 'Marta Abreu' de Las Villas. He is currently completing a Ph.D. in Automatic Control at the same university. At present he is the head of the Computer Integrated Manufacturing Laboratory at Instituto Tecnológico de Minatitlán in México. His interests include adaptive control systems, remote laboratories and robotics.

Luis Hernandez Santana is the leader of the Mechatronics Research Group in Universidad Central 'Marta Abreu' de Las Villas. He is in charge of the University Collaboration Program between Universidad Central 'Marta Abreu' de Las Villas and the Council of Flemish Universities from Belgium. He graduated from Universidad Central 'Marta Abreu' de Las Villas (Automatic Control Engineering) in 1981 and holds a Ph.D. in Science and Technology from the same university.

Rafael Aracil Santonja is Full Professor in Departamento de Automática, Ingeniería Electrónica e Informática Industrial at Universidad Politécnica de Madrid. He graduated (Electrical Engineering) from this University in 1971 and received a Ph.D. in engineering from the same university in 1975. He is currently working in the development of robots for new applications and for intelligent teleoperation. He is the leader of the IFAC Computer Vision Spanish Group and of the Latin American Robotics Network. He has worked for several EU-founded projects (ESPRIT, BRITE, and EUREKA). His interests include automation, robotics and image processing. Dr. Aracil has published several books and articles on the above mentioned topics.

Ernesto Rubio Rodriguez graduated (Automatic Engineering) from Universidad Central 'Marta Abreu' de Las Villas in 1997. He received an M.Sc. in Automatics (Robotics and Intelligent Control) from the same university in 2000. At present he is working on his PhD in robotics (pneumatic articulations) at Universidad Central 'Marta Abreu' de Las Villas in collaboration with Universidad Politécnica de Madrid.

Ivan Santana Ching graduated (Automatic Engineering) from Universidad Central 'Marta Abreu' de Las Villas in 1999. He received an M.Sc. in Automatics (Computational Systems) from that university in 2004. His main interests are distance laboratories, data basis and Internet programming.