Design and Implementation of Web-Based Control Laboratory for Test Rigs in Geographically Diverse Locations

Wenshan Hu, Guo-Ping Liu, *Senior Member, IEEE*, David Rees, and Yuliang Qiao

***Abstract*—This paper compares current remote laboratories and describes the design and implementation of the Networked Control System Laboratory (NCSLab) in the University of Glamorgan on http://www/ncslab.net, which provides a unified and flexible Web-based interface to access test rigs located in different countries of the world. All the test rigs are connected and managed together by the NCSLab system. They are well cataloged by their characteristics, and their geographical locations are not necessarily known to the users. A three-layer structure, which consists of the main server, subservers, and test rigs, is adopted to organize the distributed facilities. All the control algorithms for the test rigs are generated by using the Matlab Real-time Workshop. Users can design and implement their own control algorithms for the test rigs. The Web interface is designed using Java JSP/Servlet technology which gives the users great flexibility, including remote tuning, remote monitoring (both data and videos), and remote control logics. In order to manage the massive information and support concurrent access, MySQL database is also integrated into the system.**

***Index Terms*—Internet, networked control, remote handling, remote laboratories, teleoperation.**

I. Introduction

EXPERIMENTATION plays an important role in science and engineering education. In the traditional environment for engineering education, lectures are supported by laboratory work, where students carry out experiments. This enables them to experience the practical application of knowledge they gain from the lecture. However, not all educational institutions can afford all the expensive experiment test rigs that their students need, and often with specialized test rigs, their utilization level is low and difficult to justify. However, if these resources can be shared across institutions, then it provides a stronger justification for purchase. Nevertheless, the geographical distance between the service provider and the service receiver is a big obstacle to enable this to be achieved.

With the rapid development of industrial electronics and network technology in the last few years, many Internet-based telemonitor and telecontrol technologies have been used in industrial applications. Telerobotics [1]–[3], process control [4], automated building [5], manufacturing [6], telemedicine [7], and teleoperations [8], [9] are examples of such applications.

Thus, with the growth of the technology, the Web accessible remote experiment via Internet has become an economical solution for the increasing number of students. Encouraged by the low cost and the great effectiveness of Web-based laboratories, design methods for various kinds of experimental instruments have been developed in the last few years, including electric drivers [10], coupled tanks [11], vibration measurements [12], linear systems [13], digital electronic systems [14], virtual labs [15], etc. Recently, some new technology has been used to improve the performance and accessibility of remote laboratories. To simplify maintenance, a remote laboratory through a bootable device is introduced in [16]. The remote laboratory can be run from a bootable CD which contains the operating system (OS) and software. In [17], the remote test rigs can be accessed via mobile phones.

However, for geographically diverse institutions to share their control experimental facilities, it is necessary to design a more universal remote laboratory platform which provides unified interfaces for different test rigs located in different places. To achieve this, the following issues need to be considered carefully.

1) Global Scale Remote Laboratory: The Web-based laboratory should have a distributed structure to support test rigs geographically located all over the world. Most of the current laboratories only support the test rigs located in one place, but there are still several exceptions. Peer–peer structures are proposed and implemented in the ALFA project [18] and Vienna University of Technology [19]. The peer–peer structure is good for expansion, but it is difficult for maintenance because of the lack of a central server. In [17], a server/microserver structure is proposed, in which both the server (central) and microservers (distributed) provide TCP/IP services to end users:

2) Flexibility for the Users: The remote laboratory should provide as much flexibility as possible to the users. Most (if not all) of the jobs that can be done in hands-on laboratories should also be realized in remote laboratories. For a control laboratory, the following are the important issues.

1) Parameter tuning. This is the most basic function of the remote laboratory. Most of the current laboratories allow the users to turn the parameters remotely. In the

Manuscript received March 21, 2007; revised February 11, 2008.

W. Hu, G.-P. Liu, and D. Rees are with the Faculty of Advanced Technology, University of Glamorgan, CF37 1DL Pontypridd, U.K. (e-mail: whu@glam.ac.uk; gpliu@glam.ac.uk; drees@glam.ac.uk).

Y. Qiao is with the Complex Systems and Intelligence Science Laboratory, Institute of Automation, Chinese Academy of Sciences, Beijing 100080, China (e-mail: aqiaojoe@163.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIE.2008.920646

Control Telelab of the University of Siena [20], some important parameters (for instance, P, I, and D in a PID control) can be manipulated by the users. However, for a laboratory with good flexibility, the users' options should not be restricted to these "important" parameters. All the parameters related to the control algorithm, such as the reference value and the variables for signal conditioning, should also be accessible in a very convenient way.

2) Real-time signal monitoring. All the real-time signals in the control algorithm should be available for the users. The users can select the signals and monitor them in the Web interface [21]. From the change in the signals, the control performance can be judged.
3) Real-time video monitoring. Monitoring the signals through charts cannot provide the same impact as in hands-on experiments. However, real-time video can effectively overcome this by helping the users to understand the dynamics of the test rigs. In the remote laboratory for a brushless dc motor, introduced in [22], both visual image and sounds are transmitted to the users.
4) Customized control algorithms. A good remote laboratory should allow the users to design their own control algorithms and implement them on the remote test rigs as conveniently as they do in a hands-on laboratory. The works in [17] and [20] are two good examples of this. In [17], the electronics laboratory at the University of Deusto allows the users to upload their own programming logic for remote complex programmable logic devices and field programmable gate arrays.

   In [20], Matlab Simulink and Real-time Workshop (RTW) are deployed for the users to generate user-defined control algorithms. The user needs to download a template file which contains two subsystems. One represents the reference, and the other is the controller. The user fills in the two subsystems with his/her algorithm and uploads them to the server. The server merges the two subsystems and generates the executable program in RTW. It is an innovative idea but a little bit inconvenient for the users. The users have to separate their algorithms into reference parts and controller parts. If they could upload their simulation diagrams directly to the server without considering the subsystems, it would be more straightforward and convenient.
5) Simultaneous access. In a hands-on experiment, an operator (teacher) works on the test rigs, and others (students) can monitor how the experiment is going on. Therefore, in remote laboratories, the users should also have the same flexibility. Many users should be able to access a test rig at the same time, but among them, only one of them can actually tune the parameter or change the control algorithms, and others can only watch. More than one user modifying the experiments concurrently makes no practical sense and should be prohibited.

3) GUI Design: The Web GUI is the interface interacting between the client users and the remote laboratory. From the user point of view, the GUI design should be friendly and easy to use. The remote laboratories in [23] and [24] are good examples. They provide various visual components for the users.

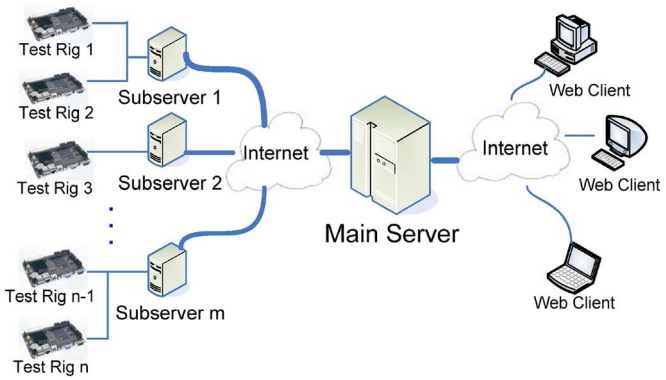


Fig. 1. Three-layer structure of the NCSLab.

The user interface should also provide sufficient documents covering every test rig. Rich documentation helps the users to understand the test rigs before they do experiments, which can reduce the chances of making errors.

The aforementioned list is of the targets set to the designers of the Networked Control System Laboratory (NCSLab). The object of the NCSLab project is to build a global scale remote laboratory providing a user-friendly remote experiment service to end users all over the world. The following technologies are adopted in the design of the NCSLab.

a) Three-layer structure: In order to manage geographically diverse test rigs, a three-layer structure (server/subserver/test rigs) is adopted. As shown in Fig. 1, the subservers are placed locally with the test rigs. They collect the real-time data and videos from the test rigs via the local network and pass them to the main server. A similar three-layer structure is proposed in the DEUSTO system [17]. However, in that system, every microserver (subserver) provides TCP/IP service to the end user. Because the DEUSTO system does not support concurrent connections, only one user can access a particular test rig at any one particular time.

In the NCSLab, the subservers do not provide direct TCP/IP service to the users. They only communicate with the test rigs and main server. This feature is good for concurrent access. The load on the subserver keeps constant no matter how many users access a particular test rig simultaneously. The increase in the number of users only increases the load on the main server, which can be a very powerful dedicated server. The loads on subservers are relatively constant and predictable. Therefore, they can be low-cost computers with limited computing capacities.

The main server provides centralized management and maintenance for the diverse test rigs. Because only the main server provides services to the users, it can easily guarantee one unified interface for accessing every test rig. All the test rigs are cataloged by their functions, and their locations are not necessarily known to the users.

b) Database: There is a lot of information that needs to be managed in the NCSLab, such as the users' details, test rigs' information, help documents, real-time experimental data, etc. It is natural to use the database to simplify the information

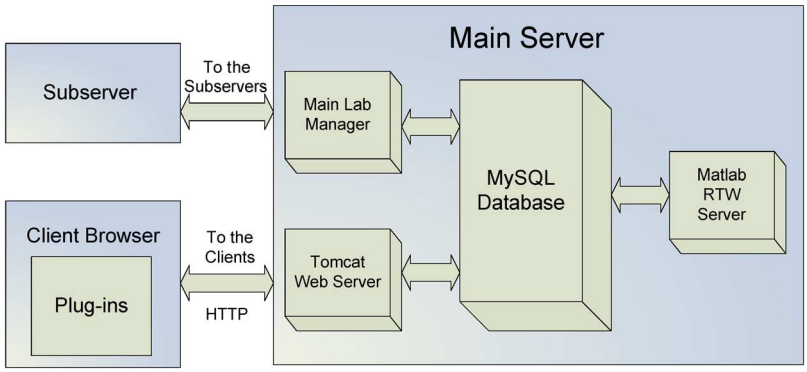


Fig. 2. Structure of the main server.

management. All the data including both real time and nonreal time are stored in the database. The Web server only derives the data from the database.

The introduction of the database makes the system configuration and maintenance easier. By manipulating the database, it is very easy to add, delete, or update the test rigs without modifying the Web scripts.

c) AJAX: AJAX is a new label to describe rich desktop-like Internet applications that run in standard Web browsers and do not require any special plug-ins [25]. Many functions that used to be achieved only by plug-ins can be realized by AJAX. However, the idea of AJAX is relatively new (first introduced in 2005) and far from fully developed. It still has some restrictions. For example, AJAX does not support the persistent HTTP streaming very well [25], which may be necessary for the real-time data acquisition. Therefore, plug-ins have to be used sometimes.

d) Matlab RTW: RTW, which is an accessory of Matlab, can generate optimized ANSI C codes from control system blocks in Simulink. All the control algorithms in the NCSLab are generated by Matlab RTW, because it can provide a unified communication interface. The users can upload their own Matlab Simulink diagrams to the server. On the server side, these diagrams are transformed into executable codes by Matlab RTW.

Benefiting from the technologies listed previously, the NCSLab has successfully integrated six test rigs located in four institutions (University of Glamorgan, Chinese Academy of Sciences, Tsinghua University, and Central South University) from two countries (United Kingdom and China). Users can log on the Website to access the six test rigs without considering their geographical locations. They are allowed to do most of the experimental work which they can do in a hands-on laboratory, such as implementing the control algorithms, tuning the parameters, and monitoring the control results through both dynamic charts and real-time videos.

II. Structure of the NCSLab

In order to organize and manage the remotely located test rigs, the NCSLab Web-based laboratory applies the three-layer structure that is shown in Fig. 1. There is a subserver layer between the test rigs and the main server in this structure. Their task is to collect the real-time data from the test rigs via a local network and pass them to the main server. The function of the subserver is normally performed by a Java program.

This structure secures reliable communication channels for the control modules of the test rigs. Normally, they are embedded computers and have limited computing resources. Long-range Internet communications may be too resource-costly due to the long delay and data dropout. Subservers are placed locally with the test rigs. They are connected through the local network rather than the Internet, so the connections are more reliable and cause less uncertainties. It also benefits the security of the test rigs. Because there is no direct communication between the test rigs to the hosts out of the local network, the test rigs can be placed behind firewalls which protect the test rigs from malicious attacks.

A. Structure of the Main Server

The main server is the central part of the whole NCSLab system. On the one hand, it provides the Web interface for the users to log on and carry out remote experiments; on the other hand, it also has a machine–machine interface to collect the real-time data from the subservers and pass the users' instructions to the lower layer. As in the structure shown in Fig. 2, the main server consists of four parts: the Tomcat Web Server, Main Lab Manager, MySQL database, and Matlab RTW Server.

1) Tomcat Web Server: The Web server for the NCSLab is based on Tomcat 5.5 which supports JSP/Servlet technology. It provides the Web interface for the clients to log on and access the remote laboratories. The interface includes plug-ins as well as Web pages. These plug-ins are designed to provide dynamic interfaces for the clients to monitor the experimental results and real-time videos. They are loaded automatically by the client's browser when the client logs on.

The communication interfaces between the plug-ins (Applets and Flashes) and the Web server are also developed. When the plug-ins are running on the client's side, there are persistent HTTP communication channels established between them and the Web server. Through these channels, the Web server sends the real-time data to the plug-ins constantly, so they can display the experimental results in real time. It also receives the client's instructions from the AJAX scripts, such as tuning parameters or changing the control algorithm, and stores them to the database.

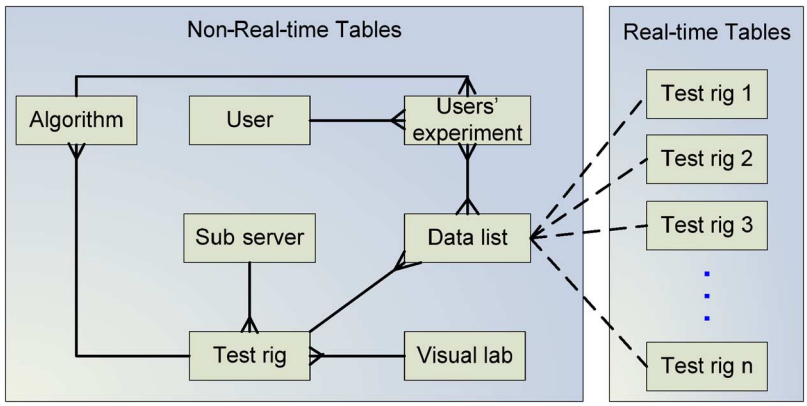


Fig. 3. Database structure.

2) Main Lab Manager: Main Lab Manager is a Java program whose task is to provide a machine–machine interface to the subservers, by which the main server can collect the real-time data and pass the clients' instructions to the lower layers. It continually receives the real-time data collected by the subservers from the test rigs and stores them into the database. The Main Lab Manager also scans the status of the database periodically (every 1 s). When new clients' instructions arrive, they are passed to the corresponding subservers for execution.

3) MySQL Database: The database is the kernel of the main server. All the important information, such as the users' and the test rigs' details and the real-time data, is stored in it. If it crashes, the whole system would stop working. Therefore, stability is of great importance for the whole system. MySQL is a free database software with high reliability and good performance, so it is an excellent solution for the database platform. The tables in the database are divided into two parts by their real-time features: the non-real-time part and linear part, as is shown in Fig. 3.

Non-real-time data such as the users' and test rigs' details, etc., are stored in the non-real-time tables. The "Data list" is a dynamic table. When a test rig is connected to the NCSLab, the system gets all the information about the real-time data (signals and parameters), which includes the data type and position, etc., from the test rig and stores them in the table. The old list for the test rig is overlapped by the new one. The users' experiment configurations, such as which signals he/she wants to supervise, which parameters he/she wants to tune, etc., are stored in the table "Users' experiment." The table "Visual lab" is added into the database to catalog the test rigs by their inherent characteristics. For examples, all the water tank test rigs are cataloged into the "Water tank lab," and all the servo control test rigs are cataloged into the "Servo control lab," and so on.

The real-time experimental results are stored in the real-time tables. When a test rig is connected into the NCSLab, a corresponding table is created to save the real-time data obtained from it. These data are stored consecutively according to a sequence indicated by the value of the corresponding field in table "Data list." The database keeps the last 10-min data. The data outside of this are discarded automatically.

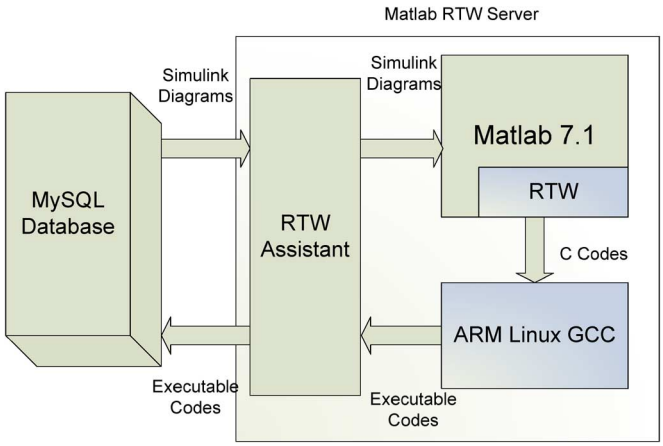


Fig. 4. Structure of the Matlab RTW Server.

Therefore, the non-real-time tables are just like buffers between the Tomcat Web Server and Main Lab Manager. No matter how many users access a certain test rig concurrently, the Web scripts only needs to obtain the real-time data from the database.

4) Matlab RTW Server: The Matlab RTW server consists of three parts: Matlab 7.1 with Simulink and RTW, ARM Linux GCC, and an interface program, RTW Assistant. Their relationship with the MySQL database is described in Fig. 4.

Matlab RTW can generate optimized ANSI C codes from control system blocks in Simulink. ARM Linux GCC, the special compiler version for embedded applications, can compile these C codes into executable programs. Therefore, based on this software, Matlab RTW Server for NCSLab is designed. The RTW Assistant, which is a Java program, obtains diagrams from the database. The RTW transforms the diagram into C codes, and ARM Linux GCC compiles these codes. The RTW Assistant sends the final executable program back to the database.

In a control program generated by RTW, there are two kinds of accessible real-time data: the parameters, which are tuneable, and the signals, which can only be monitored. The RTW provides C-API to access these data. By using C-API, the signals can be monitored, and the parameters can be tuned while the generated program is running. Therefore, the communication interface is built based on C-API, which is the bridge between

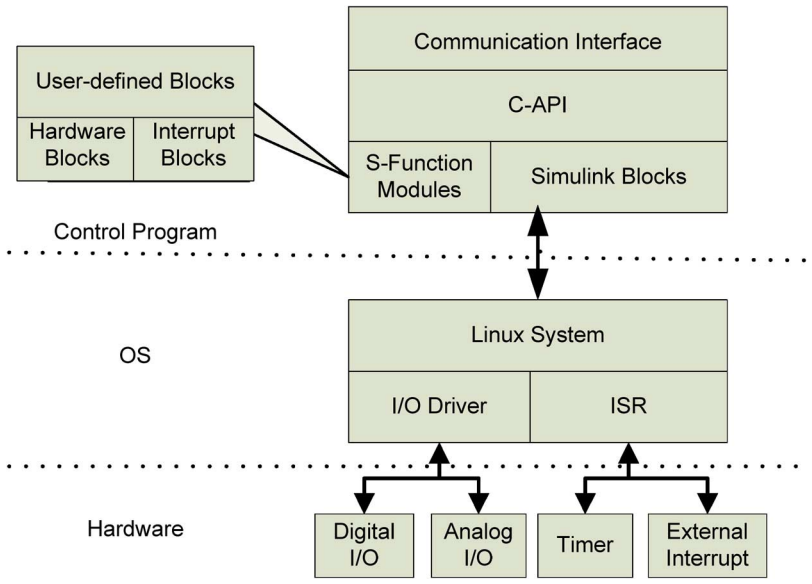


Fig. 5. Structure of the control program and its relationship with the OS.

the internal status of the algorithm and the external monitoring software.

Some blocks relative to the hardware are necessary to generate the final executable program but not available in the Simulink library, so the S-function is employed to develop and mask more general-purpose blocks. With these blocks, users can access the expanded peripheral units in the control module, such as analog input, analog output, digital input, digital output, the network data sender, and the network data receiver. Users can also adopt some advanced control algorithms by writing their own S-functions.

From the point of view of the OS theory, the S-function I/O blocks cannot access the hardware directly. The access to physical peripheral units is essentially achieved by their respective device drivers, which are created in the kernel layer. Therefore, several programmed device drivers in Linux kernel are developed, from which generated control programs get the ability to manipulate I/O interfaces. The structure of the control program and its relationship with the OS are shown in Fig. 5.

B. Subserver

The subserver is a computer which supports Java virtual machine. As long as the subserver has enough resources left for running the Sub-lab Manager program, it has not got to be a dedicated host. Unlike the DEUSTO system [17], the subserver layer does not provide direct TCP/IP services to the end users. Therefore, no matter how many users access a certain test rig, the load on the subserver can be kept constant. The Sub-lab Manager program is the bridge between the test rigs and the submain server. The main tasks for it are to collect the real-time data from the local test rigs and send them to the main server, so it needs two network interfaces. One serves the test rigs, and the other is for the main server. It requires only a very small program with a few kilobits of code. If the number of test rigs locally is less than five, normally, it only takes less than 5% CPU time for a typical Pentium 4 PC, so the host is still free for other tasks.

C. Test Rigs

The test rigs are located throughout the world, as was intimated in Section I. However, each of them is controlled by a similar control module. It is mainly based on a 32-b ARM RISC microprocessor, which is a cost-effective high-performance microcontroller solution for Ethernet-based systems. It is composed of a main board, an AD/DA board, an LCD board, and an I/O board. The main board has a 32-b ARM CPU (200 MHz), 64-Mb memory, a network port, and two USB ports. The AD/DA provides 12 analog–digital channels and two digital–analog channels.

Various real-time control algorithms can run on the control module. The control algorithm can be generated by using visual control configuration software based on Matlab RTW and the GCC compiler. The OS for the control module is embedded Linux. There is an algorithm receiver program running as a service, which allows the users to change the control algorithm at any time. When a new algorithm is used, the service stops running the previous one and starts the new one. Because all control algorithms are generated by the visual configuration software, every control algorithm supports a unified monitoring interface, which enables the external programs, such as NCSLab to monitor the internal parameters and signals via the network. By using the three-layer structure, the test rigs diversely located in a different part of the world are connected together elegantly.

As an example, Fig. 6 shows the magnetic levitation test rig in the University of Glamorgan. The device on the left-hand side is the control module with an LCD; the right-hand side is the test rig which is controlled by the control module.

III. Design of the NCSLab

A. Machine–Machine Communications

The machine–machine communications refer to the communication channels among the test rigs, subservers, and main server. This part has two tasks. One is to collect the real-time

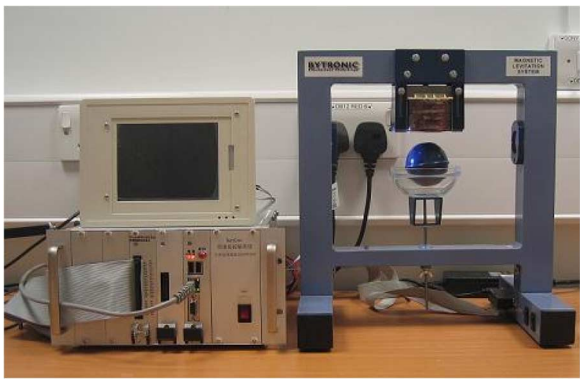


Fig. 6. Magnetic levitation test rig.

data from the test rigs and store them into the database, and the other is to pass the users' instructions to the test rigs.

In order to make the structure more clear and simplify the design, the implementation of this part takes advantage of Java's Multithread technology. When a test rig is first added to the NCSLab system, the local subserver will try to connect the test rig. If the connection is established, a new thread which deals with communication with the test rig is established. Then, the new thread tries to make a connection with the main server. The main server receives the request and then also creates a new thread to keep the TCP connection. Therefore, the communication between the test rig and the main server via the subserver is established.

In this model, every test rig has a dedicated corresponding thread in both the local subserver and the main server, as shown in Fig. 7. These threads have the same functions but work independently. That makes the system maintenance very convenient because if one or a few test rigs or threads fail, it only affects the functionality of parts of the system. The stability of the whole system is not risked.

Once the communication channels are established, the list of all the parameters and signals of the test rigs is collected by the subserver and sent to the main server. The main server receives them and stores them in the database table "Data List." Then, according to the sequence of the list, real-time data are sent to the main server through this communication channel every sample time. These data are stored in the database by the main server.

Also, when an instruction has been made by the clients, it is stored in the database. The corresponding thread at the database perceives the change on the database and then passes the instruction to the test rig for execution.

The use of multithread programming makes the logic of the communication program clearer. If a new test rig is added to the laboratory, only a new thread is created in the communication programs. According to the experiences obtained from the current NCSLab system, an average test rig normally increases the CPU utilization time by less than 1% in a Pentium 4 main server or subserver.

B. Web-Based Interface Design

The user accesses the server through a single Web address (http://www.NCSlab.net), which activates the Web server application. The Web-based interface written in HTML, JSP/Servlet, AJAX scripts, Flash, and Java Applets integrates the functions of user authorization, experiment configuration, experiment implementation, and test rig management. The Apache Tomcat 5.5 open source Web server has been used to develop the Web-based interface.

1) User Authorization: Only authorized users can access the NCSLab system. Before a user does an experiment, he/she must log on the Website and provide the right username and password. A request with the wrong password is refused. The information of users' details is listed in the database table "User."

2) Experiment Configuration: Before the user can carry out the experiments, he/she needs to set up the experiment configuration first. In the RTW control program, signals and parameters are organized as a tree structure according to their internal relationships on Simulink diagrams. The tree structure is stored in the database when a test rig is newly connected to the system or reconnected after changing the control algorithms.

The RTW provides various signals and parameters for the users, which gives the users great flexibility to manipulate the experiments. However, for a particular experiment, the user may not want to access all of them. Too many of them would cause unnecessary confusion during the experiment. Therefore, the user needs to choose the parameters and signals he/she is going to access before the experiment starts. A pop-up window in which the tree structure is displayed is developed to help the users to select the data they need. The internal relationships between all the data are clearly listed in the pop-up window. In the main page, the users can also add or delete new tunable parameters, add or delete charts, and add or delete signals for a chart. The Web pages provide a very flexible interface for the users. There is no limitation on the maximum number of charts and tunable parameters. Once the configuration is finished, the user can click the "Save" button, and then the configuration is stored in the database and the browser jumps to the online experiment automatically.

3) Online Real-Time Experiment: Fig. 8 shows the main interface for the experiment. At the left-hand side of the Web page, the user can select the test rigs from a combo box and a tree structure. On the right-hand side is the main working area. The user can tune the selected parameters and see the real-time experimental results from the dynamic charts when the experiment is going on.

This Web interface is developed by using a Flash plug-in, AJAX, and JSP/Servlet together. It is shown in Fig. 8.

The dynamic chart is based on the Flash plug-in. The use of the plug-in should be limited in the Web GUI. However, sometimes, a tradeoff has to be made. In this case, the dynamic charts need to obtain real-time data from the main server persistently. However, the AJAX does not support persistent HTTP streaming very well. Flash or Applet supports long-live HTTP connection and provides rich GUI API for the developers. Therefore, before the next generation of AJAX with enhanced functionalities emerges, the use of plug-ins is the best choice at this moment.

The Flash plug-in running on the client side keeps persistent HTTP connections with a Servlet on the server. The Servlet gets new real-time data from the database and transfers them

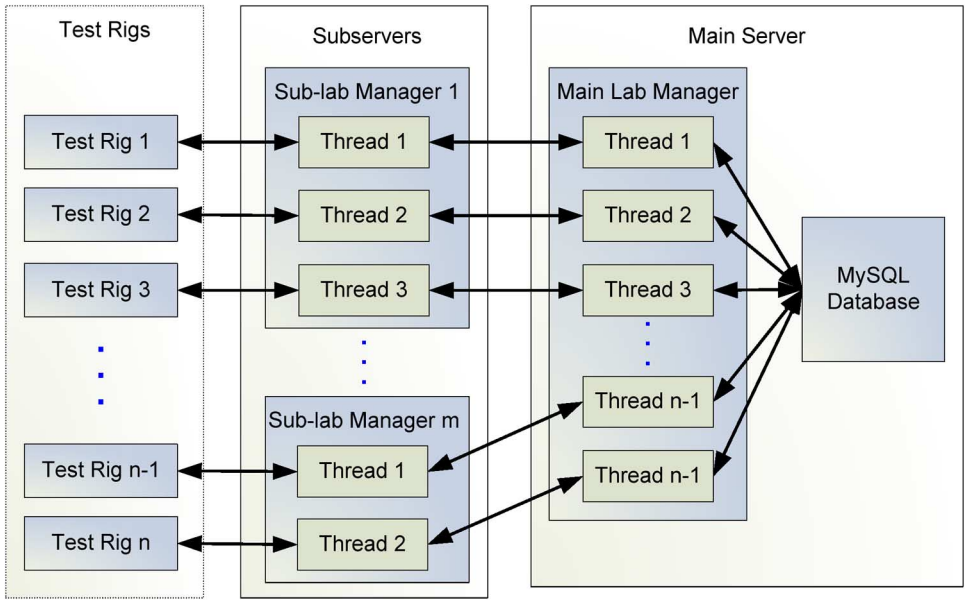


Fig. 7. Multithread communications.

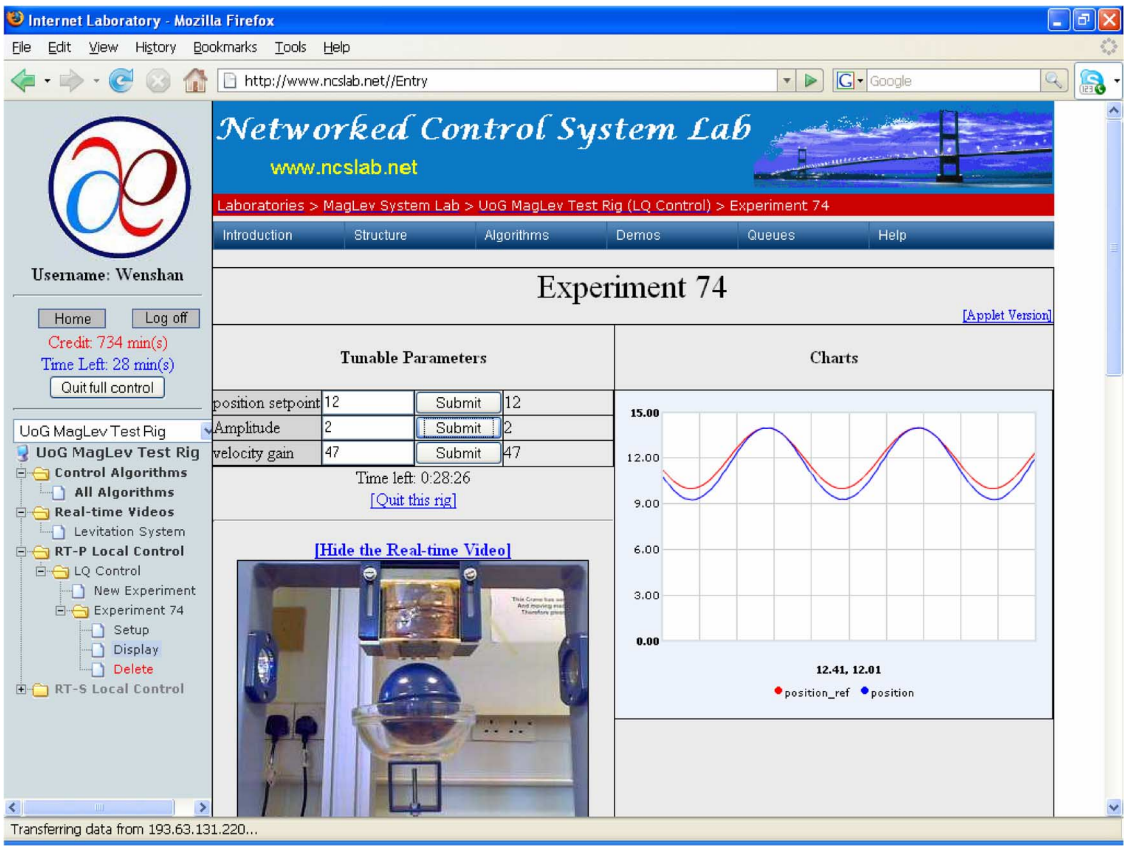


Fig. 8. Online real-time experiment.

to Flash via the Internet. According the data received, Flash displays the dynamic charts to the users. Because the data are fetched from the database rather than the test rigs, concurrent access can be realized in a very easy way.

The tunable parameters are also listed in the working area. The user can change the parameter by typing in the new value into the text box. After he/she clicks "Submit," this instruction is passed to a corresponding Servlet running at the Web server. The Servlet receives the instruction and saves it in the database. The Main Lab Manager notifies the change in the database and passes the instruction to the corresponding test rig via machine–machine communication channels. This part is achieved by AJAX.

4) Queuing Algorithm: To avoid collisions and confusions, the NCSLab supports concurrent access to certain test rigs, but only a single user has full control (the right of tuning the

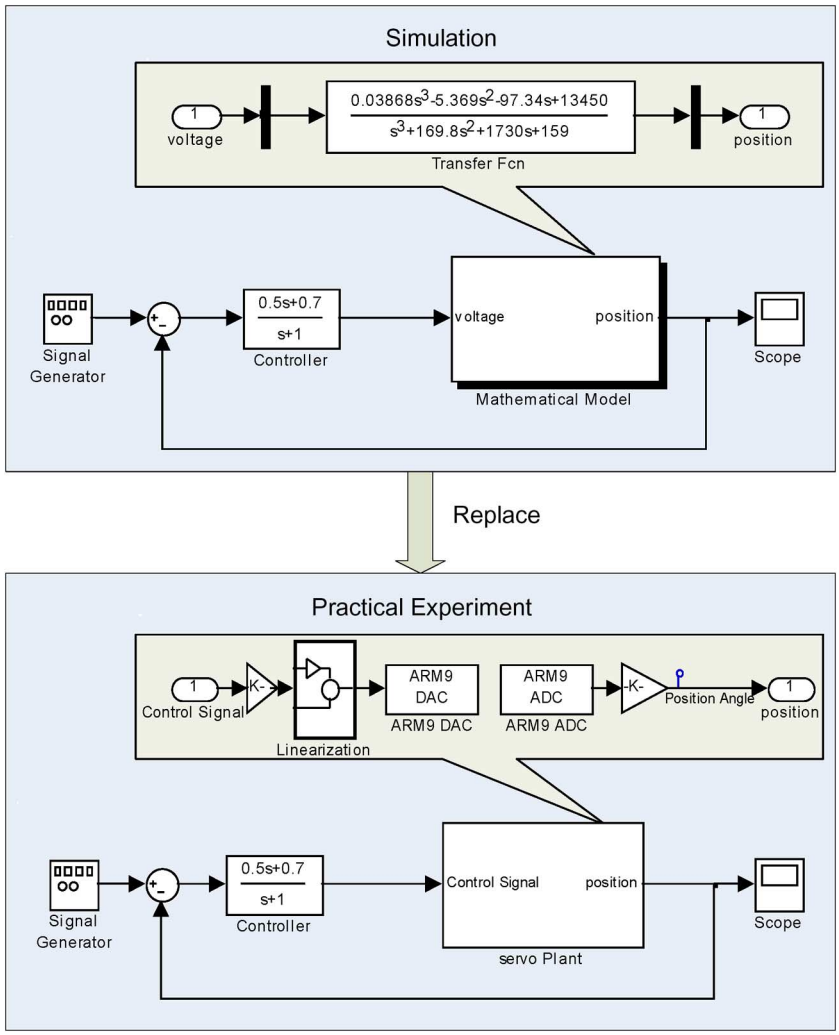


Fig. 9. Modification from the simulation to practical experiment.

parameters and changing the control algorithm) to that test rig at one time. The users without full control can only watch the real-time experimental results. They can make no changes to the test rig. If they want to get full control, they must wait in a queue. The estimated waiting time is presented on the Web interface when the user is waiting. When his/her time comes, the Web will refresh itself, and a new interface for full control will appear automatically.

Once a user is granted full control of a test rig, he/she has about 30 min of experiment time which is deemed sufficient to perform an experiment. The actual experiment time for different test rigs varies according to their complexities. When the time expires, the system withdraws the right of access. If the user wants to go back to the experiment, he/she has to request it again.

5) Customized Control Algorithms: The NCSLab provides a few control algorithms (for example, PID, LQG, etc.) for every test rig. The users can implement them. They also have the flexibility to design their own control algorithms.

For every test rig, the NCSLab provides a Simulink template file. Inside this file, there is a subsystem which encloses the mathematical model of the test rigs. The user can download this template and design his/her own simulation diagram in the Simulink based on this mathematical model. They can modify the control algorithms in Simulink and test the simulation results. They can do it again and again without worrying about the damage to the physical test rigs, until the simulation results are satisfactory. Then, the whole simulation diagrams can be uploaded to the server without any changes using the Web interface.

In every subsystem enclosed in the template files, there is a special "TagID" which can be easily identified. At the server side, the block for the mathematical model can be found from the unique "TagID." Then, the simulation model is replaced by a block which represents the physical test rigs.

Fig. 9 shows a sample example of a customized control algorithm for the servo control test rig at the University of Glamorgan. The upper part of the diagram is the simulation model designed by the user and running on the user's computer. In that diagram, the subsystem, which is derived from the template, includes the mathematical model for the servo test rig. The lower part of the diagram is the model for a practical experiment. Within that diagram, the mathematical model is replaced by the subsystem which represents the physical test rig, which includes two S-functions "ARM9 ADC" and "ARM9 DAC" for the input and output of the test rig, respectively. In

the final executable algorithms, the two S-functions are linked to the actual A/D and D/A devices on the control module.

Simulink diagrams are encoded in text files with the suffix model (MDL). The MDL format is quite clear and not difficult to interpret. However, different versions of Matlab have slightly different MDL formats. Fortunately, the information of the Matlab version is also included in the MDL files. Therefore, the NCSLab RTW server can identify the MDL format first and then make the corresponding modifications according to the Matlab version detected, which solves this problem. Currently, the NCSLab supports Matlab versions from 6.5 to 7.1.

After the diagram is modified, then it is sent to the Matlab RTW Server where it is finally transformed into executable codes.

All the transformation procedures are not necessarily known to the users. From their point of view, they only need to concentrate on the simulations in Simulink and then upload the diagrams to the server. Then, the server will generate the control algorithms automatically for them. There is no programming needed in the whole procedure. It is both flexible and convenient for the users.

6) Algorithm Selection: To change the current control algorithm, the user can click the "All Algorithm" in the tree structure on the left-hand side. It leads to the algorithm selection Web page. The name of the typical and user-defined control algorithms is listed at the right-hand side of the page. The user clicks the algorithms he/she wants to implement, and then, the Web page, which gives a brief introduction of the algorithm, appears. If the user has full control of the test rig, he/she clicks the "Download" button, then the new algorithm is transferred to the test rig after waiting for a while. Then, the user can configure the experiment for the newly downloaded algorithm.

All the executable codes for the algorithms are stored in the database. When a user clicks the "Download" button, his/her instruction would be sent to the Servlet on the Web server. The Servlet saves this instruction in the database. The Main Lab Manager notices the change in the database. It gets the executable code of the required algorithm from the database and passes it to the subserver. When the subserver receives the new algorithm, it breaks the old connection with the test rig and downloads the new algorithm through the test rig's Algorithm Receiver. Then, the control algorithm running at the test rig is updated, and the subserver makes the new connections to both the test rig and the main server. After the new machine–machine communication channel is established, the downloading process is completed.

C. Video System

In order to let the users get a more visual window on the experiments via the network, a video system is used in the NCSLab. In every subserver, there is a Visual C++ video-collecting program which obtains the video images from the USB cameras and compresses them into JPEG format. Then, these JPEG images are sent to the main server and temporarily stored in the MySQL database.

At the client side, a Java Applet is developed. It holds a persistent HTTP stream with the main server, getting the real-time video from the main server and displaying them in the user's browser.

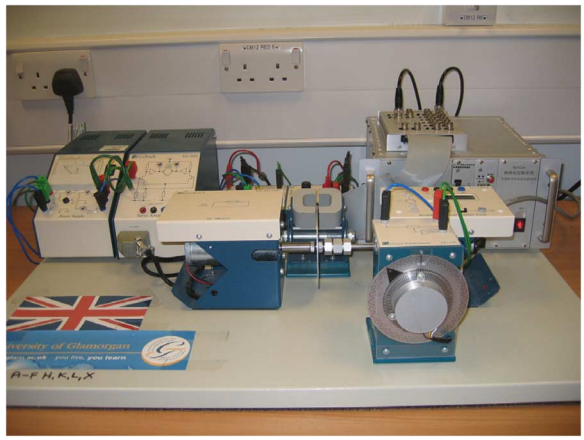


Fig. 10. Servo control test rig.

This video system structure keeps the loads on the subservers constant. The increase in the number of users watching the real-time videos does not increase the burden on the subservers.

IV. Implementation of the NCSLab

A. Virtual Laboratory

The architecture described in the previous section has been implemented. The main server is located at the University of Glamorgan, and there are six test rigs integrated around the world so far. These include two servo control test rigs, three water tank rigs, and a magnetic levitation test rig. The six test rigs are cataloged by their inherent characteristics. They are put into three virtual laboratories: Servo Control Lab, Water Tank Lab, and Meglev Lab. If a user wants to do an experiment on a particular kind of test rig, he/she can go to the corresponding virtual lab. The user can choose any test rigs available at that moment to do the experiment.

In fact, all the test rigs in a virtual lab might not be at one place. For instance, the two test rigs in Servo Control Lab are located in two different locations. One is at the University of Glamorgan, U.K.; the other is at the Chinese Academy of Sciences, China. However, from the users' point of view, they do not have to know where the test rigs are located geographically. They can find what they want by going to the relative virtual laboratory.

B. Real-Time Experiment and Real-Time Simulation

The NCSLab provides online real-time simulations as well as real-time experiments. In real-time simulation, the control object is not the actual test rigs but their mathematical models. These models are well identified, so they can represent the characteristics of the practical test rigs closely.

In real-time simulation, both the control algorithm and the mathematical model are running in the control module. Because the control object is the model, the control module does not apply any control signal to the practical test rig. Real-time simulations are very useful for some fragile test rigs. Bad

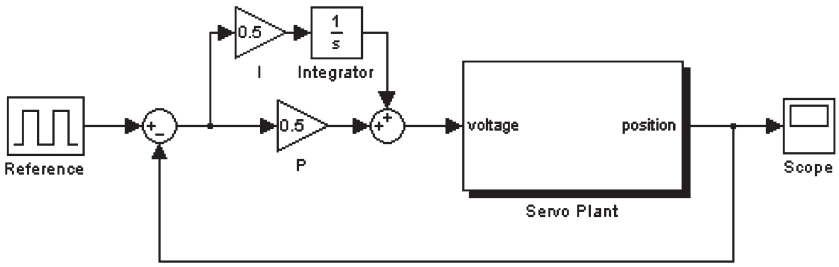
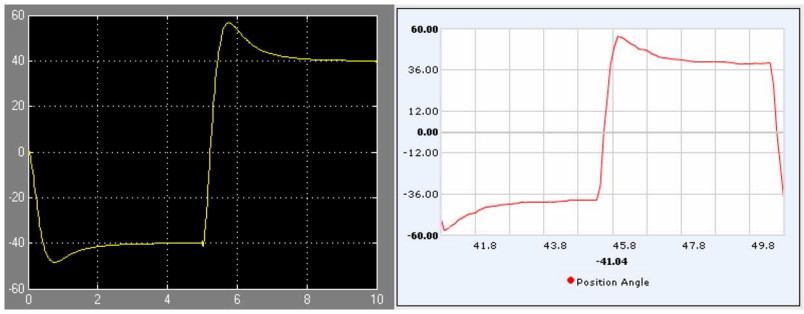


Fig. 11. Simulation diagrams in Simulink.



(a) (b)

Fig. 12. Comparison between the simulation and experiment. (a) Simulation result in Simulink. (b) Practical experimental result in NCSLab.

control practice may cause damages to them. Therefore, real-time simulation provides a good platform for the users to test their ideas beforehand. Only after the new idea is tested in the real-time simulation and proved to be safe can it be applied to the practical experiment.

V. Example

To illustrate how the users use the NCSLab system to do experiments, the servo control test rig at the University of Glamorgan is selected as an example. This rig consists of the following elements: a dc motor, a magnetic load, a gear box, an angle position plate, an amplifier, and a power supply. These individual elements are magnetically assembled on a base plate which is shown in Fig. 10. The task of the test rig is to control the angle of the position plate following the preset angle.

This test rig can be applied to both the local and network predictive [26]–[28] control experiments. In local control experiments, both the controller and the test rig are at the same place. In networked control experiments, the controller and test rig are located in two different places. The control loop is closed through the network.

A. Local User-Defined Experiment for the Servo Control Test Rig

To carry out a user-defined experiment, the user needs to download the template file for the server control test rig. The template encloses a subsystem which represents the mathematical model of the test rig identified as

$$G(s) = \frac{0.03868s^3 - 5.369s^2 - 97.34s + 13\,450}{s^3 + 169.8s^2 + 1730s + 159}.$$

Based on this subsystem, the user can build his/her simulation diagram. Fig. 11 shows a sample example of PI control, in which the block "Servo Plant" is the subsystem. The user can work in Simulink and get simulation results.

If the simulation results are satisfactory, the user can upload the diagram to the NCSLab server without any changes. At the server side, the mathematical model is replaced by a real physical plant, and the modified diagram is compiled into the executable program.

Then, the user can select the newly generated user-defined algorithm and implement it to the practical servo control test rig. Fig. 12 shows the comparison between the simulation result in Simulink and the practical experimental result in the NCSLab. The comparison shows that the two results are very close. It indicates that if the identified mathematical model is accurate enough, the transformation from the simulations to the practical experiments could be very straightforward and convenient for the users in the NCSLab.

B. Networked Control Experiment for the Servo Control Test Rig

In this case, the test rig is at the University of Glamorgan, but the controller is at the Chinese Academy of Sciences, as shown in Fig. 13. Both of the two parts are registered to the NCSLab. As most of the NCS systems, the two parts are connected by using user datagram protocol (UDP) communication protocol. Due to the long distance, there is a long time delay (the round-trip delay is around 400 ms) and serious data dropout between the two parts. If they are not well compensated for, the control system is unstable.

However, the round-trip networked predictive control (NPC), which was introduced in [29], is an effective method to solve this problem. The idea is to take advantage of the network

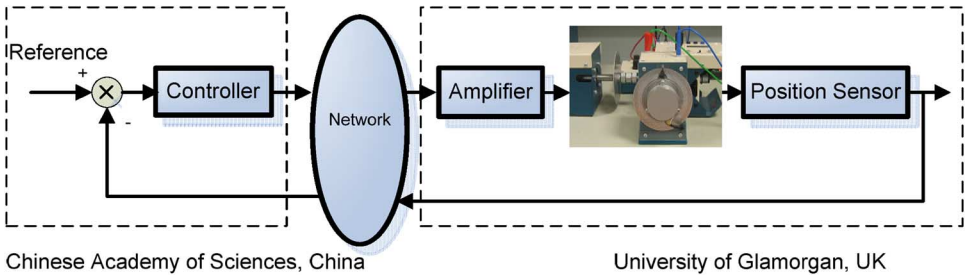


Fig. 13. Networked servo control.

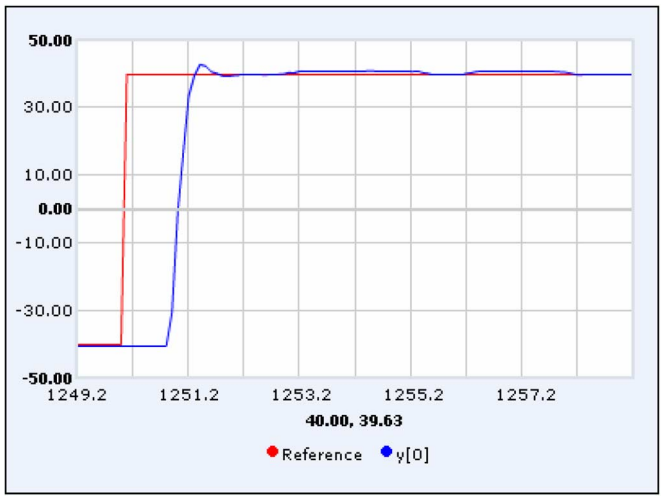


Fig. 14. Networked servo control experiment.

feature that a sequence of data can be transferred through a network simultaneously. Therefore, in this method, a sequence of future control signals are predicted, packed into one packet, and sent to the plant side together. When the plant side receives the sequence, the right predictive signal would be selected according to the measured round-trip delay. A control cycle is initiated at the plant side, and the controller is an event driver. This unique design avoids the synchronization between the controller and plant side, which is always required by most of the other NCS control methods. The NCSLab system provides this algorithm for the users to experience the networked control experiment.

To carry out the NPC experiment, both the algorithms for the controller and plant side need to be downloaded. Both sides can be found in "Servo Control Lab." A simple algorithm for the plant side is downloaded to the "Servo Control Test Rig" in the U.K. so that the test rig enables itself to be controlled by a remote controller. The NPC algorithm for the controller side is downloaded to the "Remote Controller" in China.

After both the controller and plant side running the algorithms are downloaded, the UDP communication channel between them is established. In this case, the test rig is controlled by the remote controller, so all the parameters are tuned on the remote controller. Fig. 14 shows a snap shot of the chart which displays the real-time control result.

VI. Conclusion

The NCSLab, the Web-based remote laboratory, which provides access to geographically scattered test rigs, has been presented in this paper. The three-layer server structure provides enhanced communication channels, so that test rigs from all over the world can be easily integrated into the NCSLab. It is a convenient way for teaching institutions to share their experimental facilities. It also provides friendly GUI and great flexibility (the remote tuning, remote monitoring, and remote under defined algorithms) for the users. There are six test rigs belonging to four institutions from two countries already involved so far, and the open architecture enables more to join in the project in the near future.

To avoid potential risk of damaging experiment facilities, these user-made algorithms have to be validated before they can be applied to the experiment. Moreover, the users may want to compare their algorithms with others. Therefore, the design of a validation and benchmark module will be designed in a future work.

References

[1] K.-B. Sim, K.-S. Byun, and F. Harashima, "Internet-based teleoperation of an intelligent robot with optimal two-layer fuzzy controller," *IEEE Trans. Ind. Electron.*, vol. 53, no. 4, pp. 1362–1372, Jun. 2006.

[2] Y. Matsumoto, S. Katsura, and K. Ohnishi, "Dexterous manipulation in constrained bilateral teleoperation using controlled supporting point," *IEEE Trans. Ind. Electron.*, vol. 54, no. 2, pp. 1113–1121, Apr. 2007.

[3] T. Chang, P. Jaroonsiriphan, M. Bernhardt, and P. Ludden, "Web-based command shaping of cobra 600 robot with a swinging load," *IEEE Trans. Ind. Informat.*, vol. 2, no. 1, pp. 59–69, Feb. 2006.

[4] S. H. Yang, X. Chen, L. S. Tan, and L. Yang, "Time delay and data loss compensation for Internet-based process control systems," *Trans. Inst. Meas. Control*, vol. 27, no. 2, pp. 103–108, Jun. 2005.

[5] P.I.- Hai Lin and H. L. Broberg, "Internet-based monitoring and controls for HVAC applications," *IEEE Ind. Appl. Mag.*, vol. 8, no. 1, pp. 49–54, Jan./Feb. 2002.

[6] Z. Y. Wang, K. P. Rajurkar, and A. Kapoor, "Architecture for agile manufacturing and its interface with computer integrated manufacturing," *J. Mater. Process. Technol.*, vol. 61, no. 1, pp. 99–103, Aug. 1996.

[7] C. H. Salvador, M. P. Carrasco, and M. A. G. de Mingo, "Airmed-cardio: A GSM and Internet services-based system for out-of-hospital follow-up cardiac patients," *IEEE Trans. Inf. Technol. Biomed.*, vol. 9, no. 1, pp. 73–85, Mar. 2005.

[8] H. Li and A. Song, "Virtual-environment modeling and correction for force-reflecting teleoperation with time delay," *IEEE Trans. Ind. Electron.*, vol. 54, no. 2, pp. 1227–1233, Apr. 2007.

[9] R. Marin, P. J. Sanz, P. Nebot, and R. Wirz, "A multimodal interface to control a robot arm via the web: A case study on remote programming," *IEEE Trans. Ind. Electron.*, vol. 52, no. 6, pp. 1506–1520, Dec. 2005.

[10] D. Hercog, B. Gergic, and V. Matko, "Remote lab for electric drives," in *Proc. IEEE Int. Symp. Ind. Electron.*, 2005, vol. 2, pp. 1685–1690.

[11] C. C. Ko, B. M. Chen, J. Chen, Y. Zhuang, and K. C. Tan, "Development of a Web-based laboratory for control experiments on a coupled tank apparatus," *IEEE Trans. Educ.*, vol. 44, no. 1, pp. 76–86, Feb. 2001.

[12] S. Qin, L. Bo, and X. Liu, "Development of the networked virtual instrument lab for vibration measuring based on Microsoft .NET," in *Proc. 21st IEEE Instrum. Meas. Technol. Conf.*, 2004, vol. 2, pp. 1286–1289.

[13] G. Viedma, I. J. Dancy, and K. H. Lundberg, "A web-based linear-systems iLab," in *Proc. Amer. Control Conf.*, 2005, vol. 7, pp. 5139–5144.

[14] L. S. Indrusiak, M. Glesner, and R. Reis, "On the evolution of remote laboratories for prototyping digital electronic systems," in *IEEE Trans. Ind. Electron.*, vol. 54, Dec. 2007, pp. 3069–3077.
[15] R. U. Garbus, I. J. Oleagordia Aguirre, R. C. Sanchez, and O. R. Pureco, "Virtual remote lab for control practice," in *Proc. Electron., Robot. Autom. Mech. Conf.*, 2006, vol. 2, pp. 361–366.
[16] M. Casini, D. Prattichizzo, and A. Vicino, "Operating remote laboratories through a bootable device," *IEEE Trans. Ind. Electron.*, vol. 54, no. 6, pp. 3134–3140, Dec. 2007.
[17] J. Carcia-Zubia, D. Lopez-de-Inina, P. Orduna, U. Hernandez, and I. Trueba, "Questions and answers for designing userful webLabs," *Int. J. Online Eng.*, vol. 2, no. 3, 2006.
[18] G. R. Alves, J. M. Ferrerira, D. Muller, H. Erbe, and N. Hine, "Remote experimentation network-yielding an inter-university peer-to-peer e-service," in *Proc. IEEE Int. Conf. Emerging Technol. Factory Autom.*, 2005, vol. 2, pp. 1023–1030.
[19] C. Kerer, G. Reif, T. Gschwind, E. Kirda, R. Kurmanowytsch, and M. Paralic, "ShareMe: Running a distributed systems lab for 600 students with three faculty members," *IEEE Trans. Educ.*, vol. 48, no. 3, pp. 430–437, Aug. 2005.
[20] M. Casini, D. Prattichizzo, and A. Vicino, "The automatic control telelab: A web-based technology for distance learning," *IEEE Control Syst. Mag.*, vol. 24, no. 3, pp. 36–44, 2004.
[21] J. W. Overstreet and A. Tzes, "An Internet-based real-time control engineering laboratory," *IEEE Control Syst. Mag.*, vol. 19, no. 5, pp. 19–34, Oct. 1999.
[22] T. Kikuchi, T. Kenjo, and S. Fukuda, "Remote laboratory for a brushless DC motor," *IEEE Trans. Educ.*, vol. 44, no. 2, p. 12, May 2001.
[23] J. Sanchez, F. Morilla, S. Dormido, J. Aranda, and P. Ruiperez, "Virtual and remote control labs using Java: A qualitative approach," *IEEE Control Syst. Mag.*, vol. 22, no. 2, pp. 8–20, Apr. 2002.
[24] H. Hassan, C. Dominguez, J. M. Martinez, A. Perles, and J. Albaladejo, "Remote laboratory architecture for the validation of industrial control applications," *IEEE Trans. Ind. Electron.*, vol. 54, no. 6, pp. 3094–3102, Dec. 2007.
[25] M. Mahemoff, *Ajax Design Patterns*. Sebastopol, CA: O'Reilly Media, Inc.
[26] G. P. Liu, J. X. Mu, D. Rees, and S. C. Chai, "Design and stability analysis of networked control systems with random communication time delay using the modified MPC," *Int. J. Control*, vol. 79, no. 4, pp. 288–297, Apr. 2006.
[27] W. S. Hu, G. P. Liu, and D. Rees, "Event-driven networked predictive control," *IEEE Trans. Ind. Electron.*, vol. 54, no. 3, pp. 1603–1613, Jun. 2007.
[28] G. P. Liu, Y. Xia, J. Chen, D. Rees, and W. S. Hu, "Networked predictive control of systems with random network delays in both forward and feedback channels," *IEEE Trans. Ind. Electron.*, vol. 54, no. 3, pp. 1282–1297, Jun. 2007.
[29] W. S. Hu, G. P. Liu, and D. Rees, "Design and implementation of networked predictive control systems based on round trip time delay measurement," in *Proc. Amer. Control Conf.*, Minneapolis, MN, Jun. 2006, pp. 674–679.

Wenshan Hu received the B.Sc. and M.Sc. degrees from Wuhan University, Wuhan, China, in 2002 and 2004, respectively. Since 2005, he has been working toward the Ph.D. degree at the University of Glamorgan, Pontypridd, U.K.

His current research interests are in the fields of networked control systems and Web-based remote laboratories.

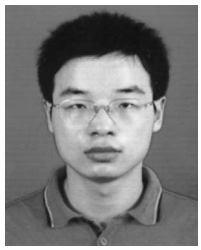
Guo-Ping Liu (SM'99) received the B.Eng. and M.Eng. degrees in electrical and electronic engineering from Central South University of Technology (now Central South University), Changsha, China, in 1982 and 1985, respectively, and the Ph.D. degree in control engineering from the University of Manchester Institute of Science and Technology (now the University of Manchester), Manchester, U.K., in 1992.

He did postdoctoral research at the University of York, York, U.K., from 1992 to 1993. He worked as a Research Fellow with the University of Sheffield, Sheffield, U.K., in 1994. During 1996–2000, he was a Senior Engineer with GEC-Alsthom and ALSTOM and then a Principal Engineer and a Project Leader with ABB ALSTOM Power. He was a Senior Lecturer with the University of Nottingham, Nottingham, U.K., from 2000 to 2003. He is currently a Chair of control engineering with the University of Glamorgan, Pontypridd, U.K., where he has been a Professor since 2004. He has been a Visiting Professor with the Chinese Academy of Sciences since 2000, and a Visiting Professor with Central South University since 1994. His main research areas include networked control systems, modeling and control of fuel cells, advanced control of industrial systems, nonlinear system identification and control, and multiobjective optimization and control.

David Rees received the B.Sc. degree in electrical engineering from the University of Wales, College of Swansea, U.K., in 1967, and the Ph.D. degree in signal processing applied to composite frequency testing from the Council of National Academic Awards, U.K., in 1976.

He is an ex-Associate Head and Director of Research with the School of Electronics, University of Glamorgan, Pontypridd, U.K. His industrial experience includes five years with British Steel and two years with Imperial Chemical Industries. His research interests include system identification, modeling, and control, and he has published over 170 scientific papers, including contributing to a number of research monographs.

Dr. Rees is a Fellow of the Institution of Engineering and Technology and a Chartered Engineer. He is a past Chairman of the IEE Control Applications Committee. In 1996, he was awarded the IEE F. C. Williams Premium Prize, which was the highest award of its computing and control division.



Yuliang Qiao received the B.Sc. degree from Tianjin University, Tianjin, China, in 2005. He is currently working toward the Ph.D. degree at the Complex Systems and Intelligence Science Laboratory, Institute of Automation, Chinese Academy of Sciences, Beijing, China.

His current research interests are networked control systems and Web-based remote laboratories.