Remote Experimentation in Control Engineering

C. Schmid

Abstract—**This paper presents a remote laboratory approach for experimentation with real physical processes, which uses the communication techniques of the World Wide Web. The aim of this kind of distance learning environment is to facilitate the access to these plants for students and professionals in the area of control engineering. Furthermore, resource sharing among the involved universities reduces money and time for maintenance or development of new experiments. The approach for remote experimentation incorporates software that uses the powerful computational engine of MATLAB/SIMULINK together with fast prototyping real-time software to build up the laboratory experiments. It is shown how to give the student the necessary feeling of interacting with real, physical equipment. Live media and virtual reality techniques are used to support the user interface. The implementation issues are discussed and an example is given.**

Index Terms—**Control engineering education, educational technology, remote experimentation, student experiments, virtual laboratories.**

I. Introduction

During the last decades there has been a decline in the use of physical experiments in control engineering education. Learning is accomplished through a diversified set of activities like lectures, paper and pencil exercises, simulators, pilot experiments, discussions, and report writing. There are several reasons for this decline. One important reason is the fact that physical experiments are costly to develop, maintain and operate. Another reason has been a strong belief that simulators can fully replace physical experiments. The ability to simulate a process is clearly very helpful for finding solutions to many real-world problems. The knowledge about modeling and simulation are therefore important skills for an engineer. Nevertheless, it is the strong believe of the author that physical experiments constitute an important ingredient in order to carry out real hands-on learning activities.

One of the salient features of engineering education is the combination of theoretical knowledge with practical experience. The former, in conventional education, consists of lectures and exercises supplemented by lecture notes and textbooks, while, the latter comprises highly resource-demanding laboratory courses. The scarcity of lab facilities and staff limits the student enrolment. Active problem solving and visual feedback on the part of students can provide a valuable insight into the problems. One of the aims in control engineering education is to teach the techniques and possible pitfalls of theory-based design methods when applied in practice. The control system design process involving practical experiments and observing the dynamics of a real physical process gives a valuable insight. For this reason, control-engineering students have to be in a laboratory to gain hands-on experience.

The laboratory approach in this paper is based on the idea that although a physical process cannot easily be moved, it can be made available for on-line remote experimentation via the Internet. The development in communication techniques has opened new opportunities for students to take advantage of a remote laboratory, where an audio-visual interface together with a collaboration tool gives them the feeling of being in the laboratory. The World Wide Web has been used to successfully demonstrate how current technology can support information sharing among widely dispersed groups. The hypertext coupled with wide area network functionality in the World Wide Web is the major reason for exploiting this technology for educational purposes to implement open and distributed hypermedia. The integration of multimedia elements has significantly enhanced the ability to train and educate electronically. Virtual education explores a new way of teaching and learning, which, instead of restricting itself only to enrolled students, addresses a larger audience. It is capable of being used for advanced training.

It is shown herein how real control engineering laboratory experiments can be implemented and offered such that they can be performed remotely via the Web. In order to maximize the educational value of running remotely operated experiments, it is important that it as closely as possible resembles running local experiments. Some important factors in this respect are the ability to observe, manipulate and control the experiment in a variety of ways, and the ability to compensate for the remoteness by utilizing advanced techniques, like video and audio communication, animation and different virtual- and augmented-reality techniques.

This work was supported in part by the Federal Ministry of Education and Research under project contract PT-NMB-08NM101A, by the 'Universitätsverbund MultiMedia des Landes Nordrhein-Westfalen' under project contract „Multimediale Lernumgebung für projektorientiertes Lernen der Methoden der Regelungstechnik" and by the EU IST Programme under project contract IST-1999-20827.

C. Schmid is with the Institute of Automation and Computer Control (ATP) at Ruhr-Universität Bochum, 44780 Bochum, Germany. (phone: +49 234-32-24093; fax: +49 234-32-04093; e-mail: cs@esr.ruhr-uni-bochum.de).

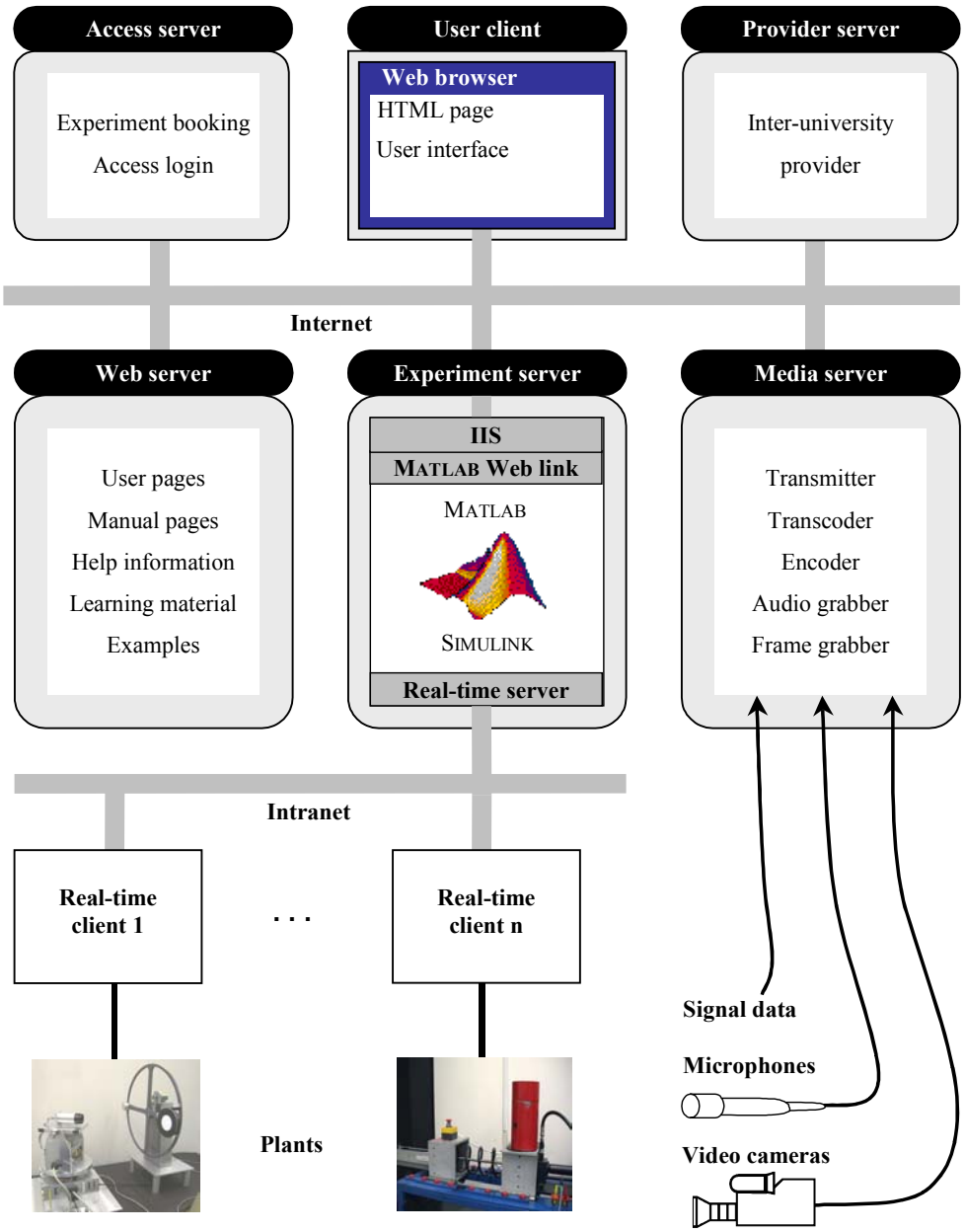

Fig. 1. Schematic diagram of the remote laboratory structure.

II. The Web-Based Laboratory Environment

A. General Aspects and Functional Units

This paper is devoted to the remote laboratory approach for experimentation with real physical processes, which uses the communication techniques of the Web. The aim of this distance learning environment is to facilitate the access to these plants for students in control engineering. Furthermore, resource sharing among universities reduces money and time for maintenance or development of new experiments.

One important aspect when designing a remote laboratory is to keep all hardware components as simple as possible to have a fast and effective maintenance. The same criterion applies for the utilization of commercially available software packages. The idea is, using software and tools that are already available at most of the educational institutions or that can be acquired free of cost. Under these viewpoints the networked structure shown in Fig. 1 is a suitable remote laboratory environment. It contains seven different types of separate functional units, which may run distributed on different hardware depending on the local availability of resources in the laboratory. In the following these units will be described and their implementations discussed.

B. User Client

The human-machine interface on the *user client* is purely Web based. The student on this client accesses and operates the laboratory system with a standard Web browser. Applets

are integrated into this interface for enhancing the Web media. More details on the user interface will be given later in this paper. The user client will send and receive information to and from the different types of servers. As far as the Web information protocol is used, Web servers will receive and handle user client requests for driving the laboratory system. Other servers are for organizational and enhanced media services.

C. Experiment Server and Real-time Clients Functionality

The operational link of the user with the real plant via Internet is only through the *experiment server*. This gateway unit uses MATLAB [1] and SIMULINK [2] combined with a *real-time server* software to communicate with *real-time clients* on an Intranet. A real-time client runs the real-time code. For each plant a separate client must be available. This client contains the process interface, which builds the hardware connection to the plant via different types of digital and analog I/O-ports using Multi-I/O cards. The real-time server and the real-time client may also run on the same computer if an Intranet solution is not available.

The real-time server uses SIMULINK and MATLAB's Real-Time Workshop [3] for generating application code that runs on a real-time client. The control system including all components for experimental purposes is modeled with SIMULINK. The experiment server holds this model and generates the real-time code, which is downloaded to the real-time clients on experiment start.

The access to the experiment server is performed only through the Web protocol. The Web server component of the *Microsoft Internet Information Server (IIS)* software is linked by a small *MATLAB Web link* dynamic link library (DLL) to MATLAB's engine interface. MATLAB commands embedded into the Hyper Text Markup Language (HTML) code of the user interface are sent to MATLAB through this DLL attached to the IIS, to get or set parameters in the SIMULINK model on the experiment server. In order to operate a plant, the real-time server software mirrors these actions between the SIMULINK model on the experiment server and the real-time code running on real-time clients. On return MATLAB functions generate JavaScript code [4], which is transferred through the same channel back and executed in the environment of the user client.

D. Real-Time Server/Client Software Solutions

The experiment server and real-time client software is based on a MATLAB and SIMULINK environment together with automatic code generation using the Real-Time Workshop of this system. For the real-time server/client part, three different software realizations are supported:

1) WinCon is a product of Quanser Consulting Inc. [5], which uses SIMULINK and MATLAB's Real-Time Workshop for generating application code that runs on the real-time client (target application). The WinCon server that implements the real-time server and the WinCon client that implements the target application may run on different computers or on the same computer under Microsoft Windows NT4.0, 2000 or XP operating system. In both cases, the server communicates with the client through the TCP/IP protocol. The WinCon server software is controlled by MATLAB and SIMULINK. During experiment operation, the target application code runs within the WinCon client software on the real-time client. The SIMULINK block diagram becomes a graphical interface to the WinCon application on the experiment server using the external mode interface of SIMULINK. By changing parameters in the SIMULINK blocks one also changes parameters in the target application. Changes of parameters in the SIMULINK model are performed by MATLAB commands and are immediately send to the target application. Signal and parameter gathering for monitoring are done similarly.
2) The xPC Target solution [6] is very similar to the WinCon system described above. The significant difference is that the real-time client does not require any standard operating system. Instead, the target system, which must be a PC, boots from a special boot disk that includes a real-time kernel. Because of this, the real-time client and server must always run on separate computers. The communication is performed only through the Intranet.
3) For experiments with lower sampling intervals both the real-time server and client can run on the experiment server. An appropriate solution for this is available using the Real-Time Windows Target [7]. The type of realization for controlling the target application is close to that of xPC Target.

E. Development and Safety Aspects

As shown above, a fast prototyping approach is applied for establishing the real-time client server configuration. The main advantage of the software solutions is that new experiments with different physical structures can be set up easily, since one can directly write new MATLAB commands embedded in JavaScript statements. On the experiment server only the SIMULINK model for the target application and the corresponding MATLAB M-files have to be rebuilt.

An additional important aspect is the safety of the experiment. The experiment has to be protected against any action that would damage or even destroy it. For this reason, all actions on the plant are analyzed by the independent and secure experiment server, which can reject dangerous commands. To evaluate students' progress, all communication between user and experiment is logged.

In this kind of experiment server implementation, the MATLAB operation is completely hidden from the user. There is no need for the experimenter to know details about the implementation techniques used. But there are restrictions. Because of safety aspects only parameter changes can be allowed for the experiments. Currently no control structure can be remotely designed and implemented by the student. In order to perform this under safety control, it is necessary to have a jacketing system that can control user actions during the

design and implementation of control system structures. For developing a SIMULINK model the current effort for the experimental auxiliary environment without the control part is already about 70% of the total. For a jacketing system this would be a multiple.

F. Media Server

One important issue that needs to be addressed in remote experimentation is how to give the user the necessary feeling of interacting with real, physical equipment. The ability to manipulate the experiment remotely, and seeing and/or hearing the results of such manipulations in real time will contribute to this aim. The feeling of being present at the experimental site can be further enhanced by the transmission of sound and video images from the experiment, and by the use of virtual-reality techniques.

Microphones and video cameras are connected to an audio and a video capture interface, respectively, which is included in the *media server* performing live audio, video and physical plant signal transmission in order to obtain the necessary closeness to the experiment, see Fig. 1. The media server has to broadcast these live media and be enabled to synchronize audio, video and signal information on one ore more user clients. Therefore multiple Internet connections must be supported and in addition it must be possible to simultaneously connect a user client with a slow and a client with a fast Internet connection.

The video frame rates using common Web server push technology are too low to visualize fast transients of dynamical systems. Compression standards used over the Internet to reduce the throughput of the streams, like MPEG, were designed for stored videos. The H.261 [8] or the improved H.263 [9] format is better suited, as it was designed for video conferences over ISDN lines and allows encoding of the video stream in real time. Without the help of some media elements, like Java applets, Web browsers cannot handle video and audio streams. As the platform of the user interface and the media server may be different, a standardized protocol is used that can be handled by Java applets. The Real-Time Transport Protocol (RTP) allows the transmission of real-time object streams via the Internet without considering specific standards of the payload formats [10]. In the current implementation of the media server, H.263 is preferred for video and GSM for audio stream encoding. For the physical plant data a new specific format has been defined. The transport is performed via RTP.

The software on the media server consists of a Java-based combined video, audio and signal capturing, encoding, transcoding and transmitting program, which uses the Java Media Framework (JMF) classes [11]. On the user client also JMF is used for a live media player that cooperates closely with the media server. As RTP uses time stamps of the payload objects, the video, audio and signal stream can be synchronized on the client. In order to reach the impression of simultaneousness of user action and reaction a maximum of transportation delay is guaranteed. This is performed on the user client by buffering the payload for about 300 milliseconds. Late arriving packages are discarded.

The payload stream delays and loads are continuously measured by the client and their statistics are reported to the media server for evaluation purposes. The media server watches over the performances of all streams and client connections. In case of an overloaded connection to a client the frame rate and/or quality of the transmitted video stream of this connection is automatically reduced such that the load is in balance with the capacity. In case of a very degraded connection speed, payload streams will be switched off according to a specified priority list. When the transmission capacity will increase again, the stream will recover automatically. This adaptive mechanism is very suitable for the different bandwidths on the Internet. Neither the users nor the experiment provider have to take care about bandwidth problems.

G. Access and Provider Servers

As only one student at a time receives access to an individual experiment, schedules and exclusive access procedures to the experiment are necessary. Users should be able to book laboratory time in advance. They should carry out the whole booking procedure by themselves in order to choose the time most appropriate to their needs. The *access server* is responsible for these tasks. The laboratory administrator can create or delete accounts and define quotas for each experiment. Before starting with experiments the students must have received access and then are able to book experimentation time by logging into the access server.

Experiment sharing offers an excellent set of didactic resources for students in engineering. A network of remote laboratories may be established on local, national, European or international level. The optional *provider server* takes care of these administrative tasks, which are necessary to link a remote laboratory to a distributed laboratory like Cyberlab [12].

The remote laboratory experiments discussed in this paper are currently available through different provider networks and can also be accessed by different access servers synchronized by the provider server. The access server for all experiments on a local and federal state level is located at the German open university FernUniversität Hagen. A detailed description can be found in [13]. On a European level access is available by Cyberlab.Org AS, a Norwegian company at Trondheim [12].

III. THE USER INTERFACE

The user interface necessary to control the experiment is organized in the form of a cockpit. The Web pages and all material are coming from the *Web server* (Fig. 1). Fig. 2 shows the example user interface for an experiment with an optical tracking system. In this experiment the student should retune controller parameters in order to achieve a better control performance. A detailed technical description of this plant and experiments is given in [14].

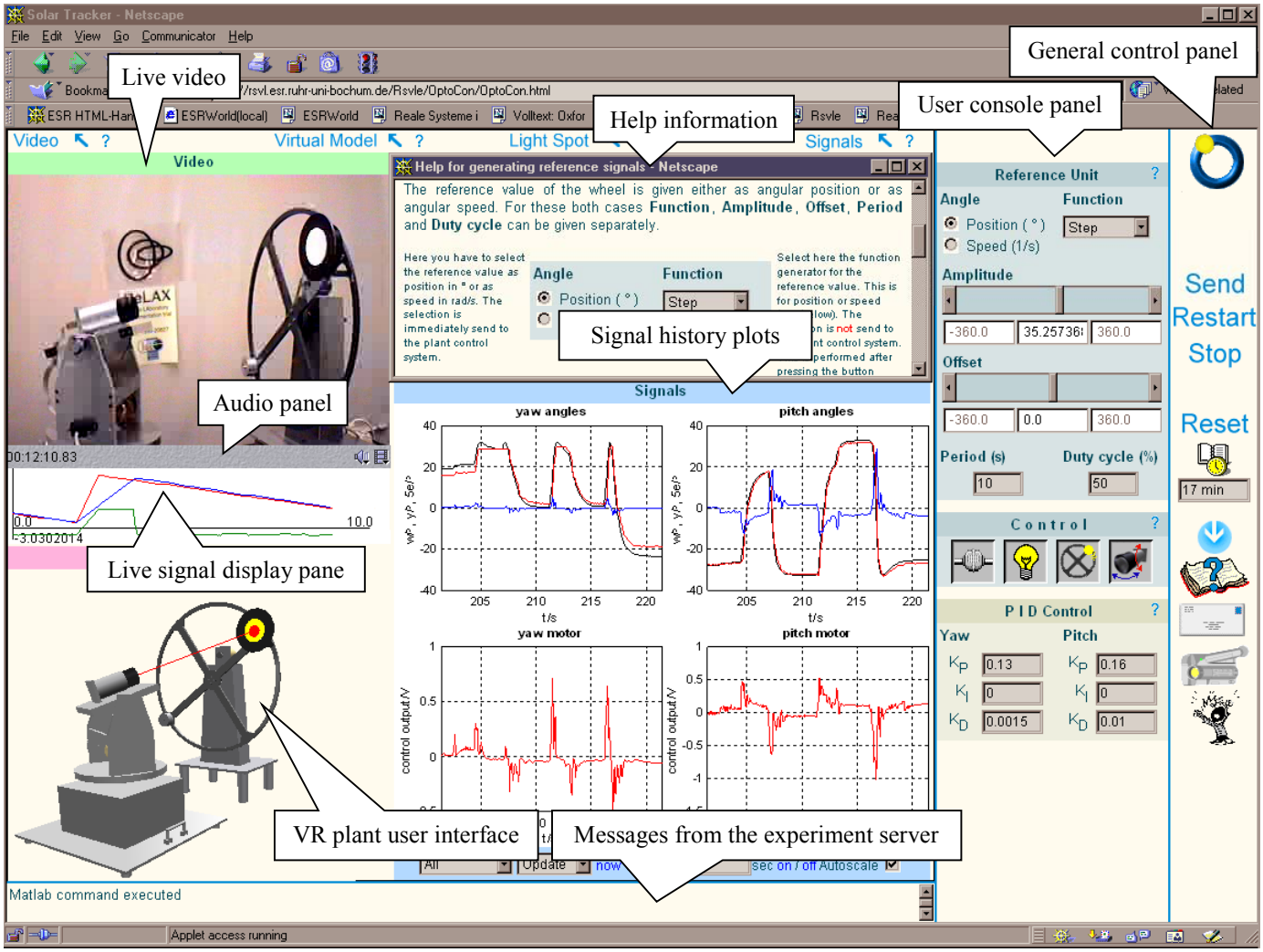

Fig. 2. Screenshot example of a user interface for an optical tracking plant.

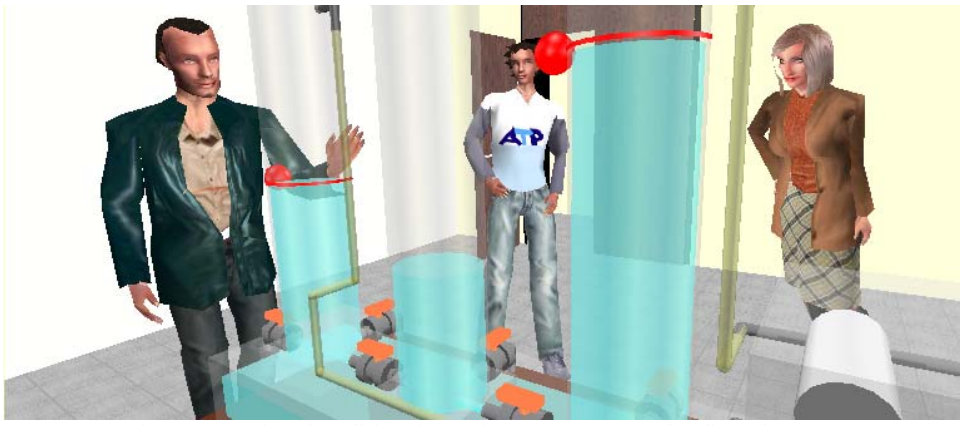

Fig. 3. Screenshot of a collaboration environment using virtual reality techniques.

Besides sliders implemented as Java applets and a graphics browser plug-in, only HTML and JavaScript have been used to organize all actions. The main operating actions (e.g. start/stop an experiment session, or reset the plant to a predefined initial state) are located in the *general control panel* frame, since these actions do not depend on the type of experiment. The specific commands to operate the plant as well as parameters to modify the characteristics of the reference signal and of the implemented controller are found in the *user console panel*. The sliders are used for adjusting values for variables smoothly in a given range and therefore facilitate operating the plant. On the left-hand side of the user interface is a workspace area, where other windows can be placed, freely arranged and controlled by the user. There is the window for displaying the live video, playing the audio signals and showing the live signals through the JMF applet.

The lack of a direct physical contact to the experiment and the use of flat operating panels may lead to a loss of practical feeling. This is compensated by enhanced multimedia components. A *VR plant user interface* enables the user to operate the real plant manually by mouse actions in the virtual reality (VR) scene of the 3-D model. The actions performed with this 3-D model are immediately converted in the environment of the user client into MATLAB commands and transferred to the real-time client by the experiment server (see Fig. 1). The 3-D plant model is written in the Virtual Reality Modeling Language (VRML) [15].

Although the user can watch the plant by the video camera or tune the controller by listening to the sound the plant produces, a graphical representation of measured signals is needed to show small variations, for the analysis of fast transients and to generate a report. The diagrams are generated remotely on the experiment server by using standard MATLAB graphical routines. In order to get fast and best results, these diagrams are directly transferred and embedded in a window on the Web page using the Microsoft Extended Metafile Format (EMF). As this is not a regular Web feature, a special graphics browser plug-in has been developed.

Typically laboratory courses are organized for and accomplished by groups. This promotes problem solutions by teamwork, which is a substantial requirement to the abilities of an engineer. Remote laboratory experiments described in this paper can currently be accomplished only by single students. In order to promote teamwork, new software tools are under development, which enable the students to collaborate in a team. Fig. 3 shows a 3-D collaboration environment, where students can met them represented by their avatars to have simultaneous access to the experiment. It requires at the same time a didactical concept that promotes the teamwork in such a environment.

IV. CONCLUSION

This paper presents a Web-based remote laboratory environment for distance education, which addresses students and learners in control engineering. The user can perform experiments using a standard Web browser. The only requirement is to install some browser extensions, which can be downloaded from the Web server and installed simply. The experiment is controlled using JavaScript statements embedded in Web pages. They contain MATLAB commands, which are sent down to the experiment server for execution in the MATLAB environment. This concept simplifies and speeds-up the implementation of new experiments due to its availability and flexibility. To add another laboratory plant only some minor sections in the user interface and the MATLAB/SIMULINK part has to be reprogrammed. The total laboratory system can be easily assembled from commonly available commercial and free components.

An evaluation form has been elaborated and given to students of control engineering to assess the quality of this remote laboratory environment. Based on the feedback received, an overall positive resonance could be noticed. The quality of the user interface as well as the graphics has been approved. Controlling a real remote plant through a virtual reality 3-D model caused a surprise among the users. They have readily accepted this kind of experimentation, which is available round the clock. This result has corroborated the development team to continue in adding other experiments and plants into this environment.

Currently three experiments using the described laboratory environment are available. Besides the optical tracker plant, mentioned above, a ball-and-beam experiment has been released. A hydraulic drive experiment is currently added. All experiments are accessible on the Web round the world and round the clock, supposed the user has been granted a login id with password, which can be received from the author. It seems that remote experiments are starting to become standard experiments in the basic control laboratory courses at many universities. Importing interesting experiments into the own courses while opening own experiments in a remote laboratory to others saves time and resources.

REFERENCES

[1] The MathWorks Inc., Natick, MA. (2003, April). MATLAB 6.5 - The Language of Technical Computing. [Online]. Available: http://www.mathworks.com/products/matlab/

[2] The MathWorks Inc., Natick, MA. (2003, April). Simulink 5 - Dynamic system simulation for MATLAB. [Online]. Available: http://www.mathworks.com/products/simulink

[3] The MathWorks Inc., Natick, MA. (2003, April). Real-Time Workshop 5. [Online]. Available: http://www.mathworks.com/products/rtw/

[4] D. Flanagan, *JavaScript: The Definitive Guide*, New York: O'Reilly, 2001.

[5] Quanser Consulting Inc., Markham, Canada. (2002). WinCon 3.3 User's Guide.

[6] The MathWorks Inc., Natick, MA. (2003, April). xPC Target 2. [Online]. Available: http://www.mathworks.com/products/xpctarget/

[7] The MathWorks Inc., Natick, MA. (2003, April). Real-Time Windows Target 2.2. [Online]. Available: http://www.mathworks.com/products/rtwt/

[8] T. Turletti, and C. Huitema. (1996, Oct.). RFC 2032. RTP Payload Format for H.261 Video Streams. [Online]. Available: http://www.normos.org/rfc/rfc2032.txt

[9] C. Zhu. (1997, Sept.). RFC 2190. RTP Payload Format for H.263 Video Streams. [Online]. Available: http://www.normos.org/rfc/rfc2190.txt

[10] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. (1996, Jan.). RFC 1889. RTP: A Transport Protocol for Real-Time Applications. [Online]. Available: http://www.normos.org/rfc/rfc1889.txt

[11] Sun Microsystems. (2003). Java Media Framework API. [Online]. Available: http://java.sun.com/products/java-media/jmf/index.html

[12] Chr. Schmid, T.I. Eikaas, B. Foss, and D. Gillet, A Remote Laboratory Experimentation Network. Prepr. 1st IFAC Conference on Telematics Applications in Automation and Robotics TA 2001, Weingarten 2001, p. 449-454.

[13] C. Röhrig, and A. Jochheim, "Java-based Framework for Remote Access to Laboratory Experiments" in Proc. *IFAC/IEEE Symposium on Advances in Control Education,* 2000, Gold Coast, Australia.

[14] Th.F. Junge, and Chr. Schmid, "Web-Based Remote Experimentation Using a Laboratory-Scale Optical Tracker" in *Proc. American Control Conference,* 2000, pp. 2951-2954.

[15] The VRML Consortium Inc. (1997). The Virtual Reality Modeling Language. International Standard ISO/IEC 14772-1:1997. [Online]. Available: http://www.vrml.org/technicalinfo/specifications/vrml97/index.htm