The Virtual Lab for Controlling Real Experiments via Internet

Christof Röhrig and Andreas Jochheim
Department of Electrical Engineering
University of Hagen
D-58084 Hagen, Germany
christof.roehrig@fernuni-hagen.de
andreas.jochheim@fernuni-hagen.de

Abstract

The aim of the Virtual Lab project is to provide students access via the Internet to various experiments in control engineering, which are situated in control laboratories at several universities. Three German universities are currently developing the Virtual Lab as a network of remotely accessible laboratories in order to set up a prototype experimental environment. Students under consideration are usually located at geographically distributed location (e.g. at home) and have remote access to our experiments. The Virtual Lab is based on a distance education concept due to the fact that certain students (e.g. professionals) may be interested in studying even at places which are far away from campus eliminating the necessity to be there in person. In the Virtual Lab they are able to gain some practice in control theory at their convenience thereby saving travel time and cost. An approach based on a client/server architecture written in Java is proposed. This paper discusses the requirements of remote experimentation and presents the technical structure and first results of the project.

1 Introduction

Control engineering education should combine theory and practice. Students should achieve knowledge and skills of control systems modelling in order to develop controllers that enforce performance requirements. After a controller is designed and implemented, observation of the resulting dynamics gives valuable insight into design concepts. Students can observe dynamic phenomena that are often difficult to explain in writing. Furthermore, interactive experimentation on real-world plants improve the motivation of the students and also develop an engineering approach to solve realistic problems. Simulation is a proper way to complement control education but in general, it cannot replace experiments on real plants. Since simulation is only as good as the model, experimentation has the advantage of making the user aware of phenomena that are hard or impossible to simulate. With remote experimentation, unique or expensive equipment can be shared between different universities. Owing to the fact that a wider range of laboratory resources can be made accessible, students have the choice of a variety of experiments. They can use the Virtual Lab at any time and at any place in the Internet saving travel time and cost.

2 System Design

The main design idea of the system is to use the World Wide Web as communication structure and a Web browser as user interface. The Web browser provides a platform for transmitting information as well as an environment to run Java applets. The Web itself provides the infrastructure to exchange the necessary information.

2.1 Requirements

Computer based controller implementation for the experiment is necessary for remote experimentation. Another important aspect is to transport the feeling of a real experiment to the remote user. A video and audio broadcast provides the remote user with the feeling of being present at the experiment location. With the visual feedback, the user can supervise the experiment and check whether everything runs as expected. For system identification and performance evaluation, it is necessary to collect the relevant data of the process. The data should be stored at the server side for later download and additional processing by the students. As only one student at a time receives access to an individual experiment, schedules and exclusive access procedures to the experiment are necessary. The main requirement of the server is the security of the experimentation plant and of the server computer. The plant has to be protected against any action that can damage or destroy it. For this reason, all commands to the controller of the plant must be analysed and dangerous controller parameters must be avoided. If the

controller algorithms can be defined by the user without any restrictions, any controller instability is hard to detect in advance. From the users point, of view there are other important features that have to be fulfilled. Students may not have the choice of which computer to use at their site. They usually do not want to install specific software before they can start the experiments. Therefore, cross-platform client software is an important user demand. Since students do not want to pay for additional software, free client software is a large advantage, too.

2.2 Procedure of Remote Experimentation

After a student has requested for a particular experiment, he receives his user ID and password. With that identification, he can log into the server and make an appointment for the chosen experiment. Appointments are stored in a database and are used for access control. The student can download the instructions and tools for offline simulation and preparation of the experiment. At the date of the experiment, the student connects to the server with the help of a Web browser. He uploads his prepared data onto the plant and starts the experiment. A live video and audio stream help to provide the student with a laboratory feeling. The output data of the experiment are stored on the server, the user can download them to analyse the results.

2.3 Communication Structure

Figure 1 shows the communication structure of the system. The communication structure is based on a client/server architecture written mainly in Java. The student may work on any platform that supports a Web browser with a Java runtime environment. The local Web browser is the only user interface to the experiment. The browser loads the client software as Java applets from the server and starts them. Due to the modular structure of the system, extensions with new features are easy to implement. The communication server runs on a PC with the operating system Linux. The HTTP server Apache provides the HTML documents and the Java applets. The server contains a video capture card and a sound card for video and audio transmission. The real-time controller of the experiment itself usually runs on a different computer hardware with a real-time operating system. The connection between the server and the real-time controller could for example be a serial link or an Ethernet. If the real-time operating system RT-Linux [3] is used, the controller can be run directly on the server. The communication structure is generic. Therefore, it can be used in different experiments. Java applets on the client's side, allow the continuous improvement of the software since the applets are loaded when they are needed. Applets are always up-to-date so no user software upgrading is necessary when the software is exchanged.

2.4 Video and Audio Transmission

A live video stream for viewing the experiment and an optional audio stream helps to provide a laboratory feeling. Early Web implementations could not handle video streams. Netscape introduced the server-push mechanism which sends images in equidistant intervals to form a video stream. Server-push technology has the disadvantage that it takes a long time to transmit images to the client. Therefore, the maximum possible frame rates are too slow for high dynamic systems. The throughput of this mechanism is far to high to be used in wide area or modem connections. Video compression standards mainly used over the Internet (H.261 and MPEG) apply motion detection and differential encoding to reduce the throughput of the video stream [4]. MPEG uses a better compression algorithm than H.261 but was designed for stored videos. H.261 was designed for video conferences over ISDN lines and allows encoding of the video stream in real-time.
Without the help of plug-ins or Java applets, Web browsers cannot handle audio streams. Plug-ins are dynamically loaded modules which are usually written in the native programming language of the computer. Compared to Java applets, plug-ins are capable of using all of the local computer resources. A software failure in a plug-in can damage the student's local computer. Furthermore, plug-ins are not platform independent and must be installed on the client's side. For user convenience and client computer security, a Java applet solution was chosen. In order to employ a different software product on the server's side, standard protocols are used. The Real-Time Transport Protocol (RTP) [10] allows the transmission of real-time video and audio via the Internet. It does not standardize any audio or video formats. Therefore, RTP can be used for transmitting standardised video and audio formats [11] as well as for transmitting proprietary formats.

2.4.1 Client: The Java Media Framework (JMF) [5] was chosen to display the media streams in the browser. JMF is an application programming interface (API) for incorporating media data types into Java applications and applets. The JMF API is a cross-platform solution written entirely in the Java programming language. JMF supports RTP receiving in the video and audio streams via the Internet. For real-time video decoding, there is a performance pack which must be installed on the client's side. The performance pack includes native code in dynamically loaded libraries. In the current version of JMF, this performance pack is only available for Windows and Solaris platforms. For real-time audio encoding, a version in pure Java is available. It enables installation of JMF 1.1 on Web servers to facilitate Web deployment of the necessary JMF library classes. In this case, there is no need for the user to install JMF software on his computer. On account of the fact that we need the video stream too,

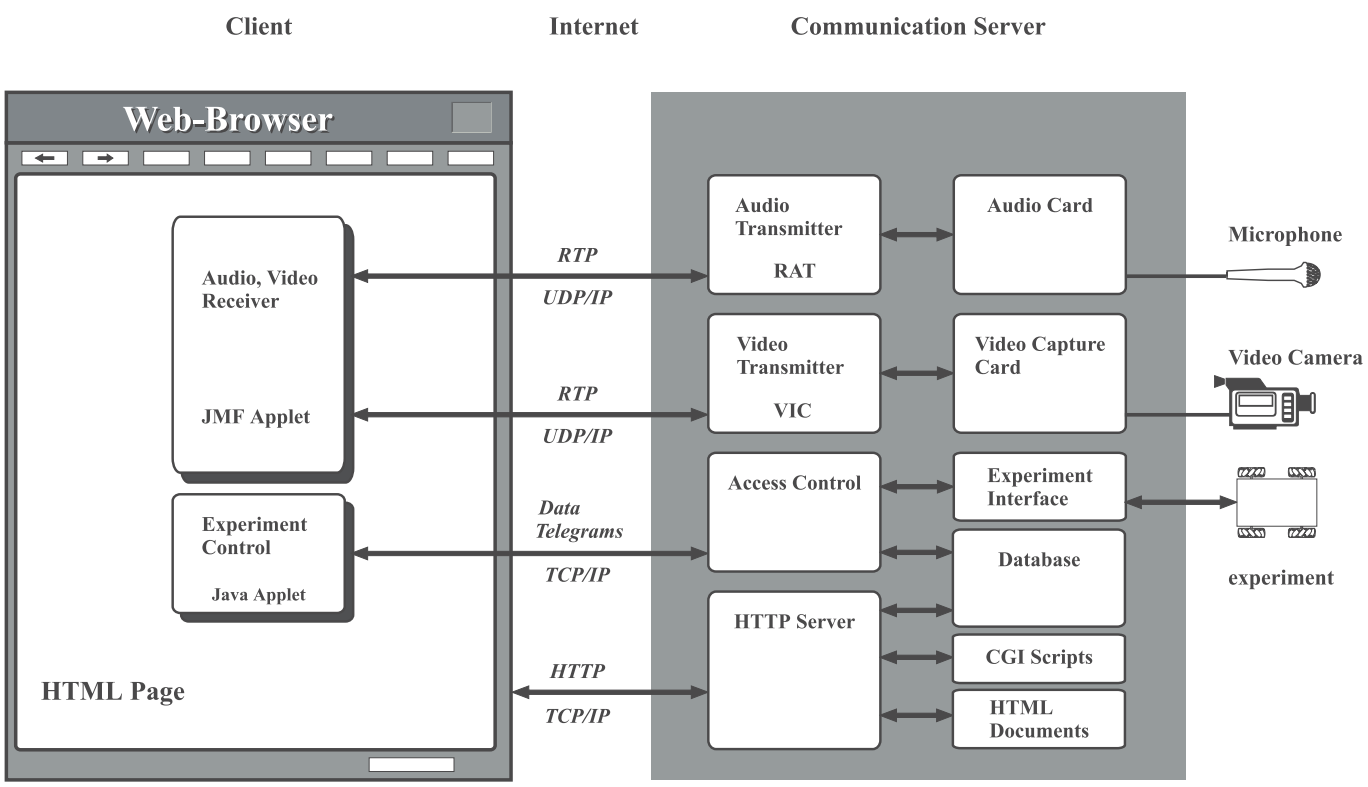

Figure 1: Communication Structure

the inconvenient software installation on the client's side is necessary. Just one single applet is needed to receive the video and audio stream from the server. This applet is stored on the server eliminating the need to install it at the client. For platforms other than Windows or Solaris, the use of plug-ins to display the media streams is necessary.

2.4.2 Server: The used JMF 1.1 is not able to transmit multimedia streams but the upcoming JMF 2.0 will support video and audio capture capability. As this new version is not yet available, MBone tools [6] were chosen to transmit the media streams. These tools are available for free and run on most Unix systems and Windows. They are specially developed for multicast transmission of the data streams but they support unicast transmission too. If unicast transmission is used, only one client can watch the media streams. To support team work, multicast transmission is necessary. Since modem connections usually are not able to receive multicast streams, a multicast to unicast translator is used to connect the server to several unicast clients. For this task, the mTranslator [7] has successfully been tested. VIC (Video Conferencing Tool) [8] is used to produce the video stream. VIC supports different video encodings but only the H.261 encoding is supported by both the used Linux release and the JMF 1.1. The throughput of the video stream depends on the frame rate, the frame size and on the image quality. The Common Intermediate Format (CIF) is often used for video conferencing. CIF produces 352 x 288 pixel at 15 frames per second. In order to reduce the throughput of the video stream, smaller sizes or fewer frame rates are possible. A quad of CIF (QCIF 176 x 144) or a sub-quad (SQCIF 128 x 96) is usual. When using VIC with a modem connection and CIF only a few frames per seconds (1-3) are feasible. When using VIC with a unicast stream it is necessary to know the IP-Address of the client. This is done with the Common Gateway Interface (CGI) of the Web server. After the client is connected to the server, the Web server starts VIC. This is done with a shell-script that uses the IP-Address of the client as a parameter for the destination address of VIC. Since the connection to the user may differ, he can chose the frame rate and the maximum throughput of VIC.
RAT (Robust Audio Tool) [9] was chosen for the audio stream. Table 1 shows the different encodings available

Table 1: RAT Encodings

Name	Bit rate	JMF	RTP payload type
L16	128.0 kbit/s	-	11
PCMU	64.0 kbit/s	+	0
DVI	32.0 kbit/s	+	5
GSM	13.2 kbit/s	+	3
LPC	5.8 kbit/s	-	7

with RAT. The throughput and the sound quality differ very much between the encodings. Not all encodings are supported by JMF. In Local Area Networks (LAN), PCMU encoding is used. This encoding produces a good sound quality and doesn't need too much processing power for compression and decompression. For modem connections, PCMU is not usable because of the high requirement of bandwidth. In that case, GSM encoding is a good choice. It was specially designed for voice signals and it needs a high processing power. If it is used in experiments with technical noise, the sound is not as good as the PCM encoding but works well over a modem connection. DVI is a good compromise between processing requirements and bandwidth. All encodings of RAT use a fixed 8kHz sampling rate.

2.5 User Interface

The user interface for controlling the experiment is realised as a Java applet. After the web page is loaded, the applet starts and an authentication dialogue and a control window pop up. The student must enter a valid login name and password to access the server. When the login process is completed, the applet opens a TCP/IP connection to the access control module of the server. This server module is also implemented in Java. The communication between client and server is executed as data telegrams over the previously opened TCP/IP connection. These telegrams contain commands and optional parameters. Each command is stored in a database with its valid range of parameters. The access control verifies the telegrams and if the command and its parameters are valid, our experiment interface transforms it and sends it to the experimental plant. The server acknowledges each command with a status telegram. This telegram contains status information of the experimentation plant which are shown in the Java applet on the client side. The control applet and the plant interface depend on individual experiments and have to be adapted to each additional experiment. The access control and the telegram structure are generic and can be used in all cases.

2.6 Data Acquisition

While the experiment is running, the input and output data of the experimentation plant are measured and stored as an ASCII file on the server. The students can download the files with the help of their browsers. The files include the input and output values of the plant and controller. Since the values are stored in ASCII format, they are platform independent. A Java application has been developed for the analysis of the controller performance. The program runs as applet in a browser or as an application in a Java Runtime Environment.

3 Controller Design for an Omnidirectional Vehicle

In one of our experiments, students have to design a speed controller for an omnidirectional vehicle with Mecanum wheels that provides three degree of freedom in Cartesian space. The vehicle is controlled by a VME-Bus computer with the operating system pSOS+. The connection to the communication server is built with a radio Ethernet bridge. The communication is based on the TCP/IP protocol and is similar to the communication between the client applet and the communication server. Students can generate a trajectory to learn the kinematic behaviour of the vehicle. The trajectories are created with a CAD tool that supports paths with linear and circular shapes. In a first step, we enhance

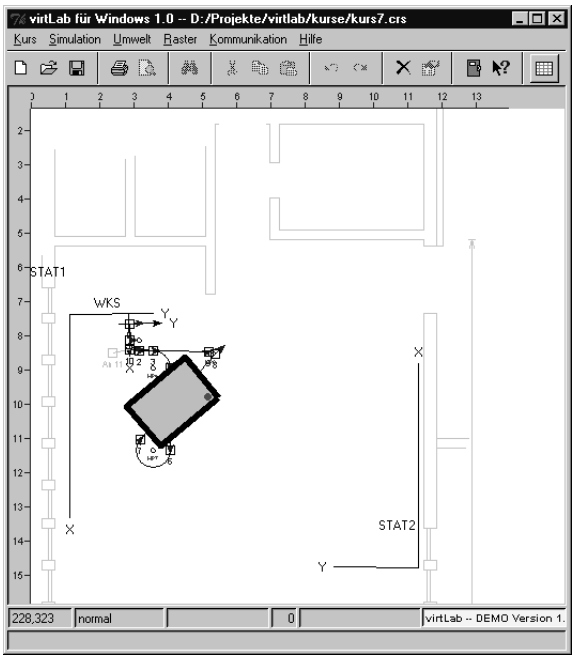

Figure 2: Simulation

the existing CAD tool with communication facilities for remote experimentation [2]. The communication module has been implemented in the programming language Java. In the actual software release, it is reused as Java Applet in the browser. After path generation, students can simulate the corresponding motion of the vehicle. Figure 2 shows the CAD tool during simu-

lation. If the behaviour is satisfactory, students can download the path on the control computer of the vehicle itself and restart the motion. The audio feedback shows the student the different noise emission of the vehicle depending on speed and direction.

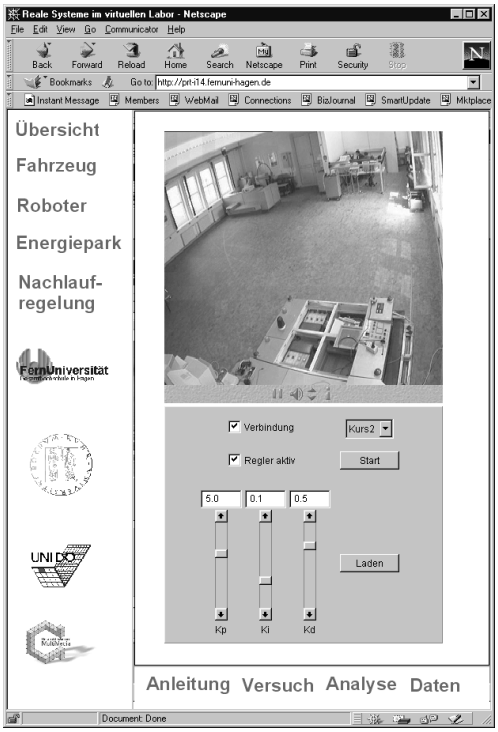

Figure 3: Web Browser with JMF and Control Applet

The first part of the experiment is to model the plant and to identify the parameters of the plant with the help of a step response signal. The measured values of the step response are stored on the server. The student analyses them with the help of the analyser applet. The analyser applet can identify the model parameters of the vehicle by means of several methods. After modelling the plant, the student has to design the parameters of the speed controllers for the wheels. He can adjust the controller parameters with sliders in the control applet. He has the choice of different paths before he starts the motion of the vehicle. Figure 3 shows the Web browser with the audio, video and control applet. While the vehicle is moving, the desired and actual velocities of the four wheels are measured and stored as an ASCII file on the server. The student analyses the data with the analyser applet. Figure 4 shows the Web browser with two analyser windows. The controller can be modified interactively until the expected performance is achieved.

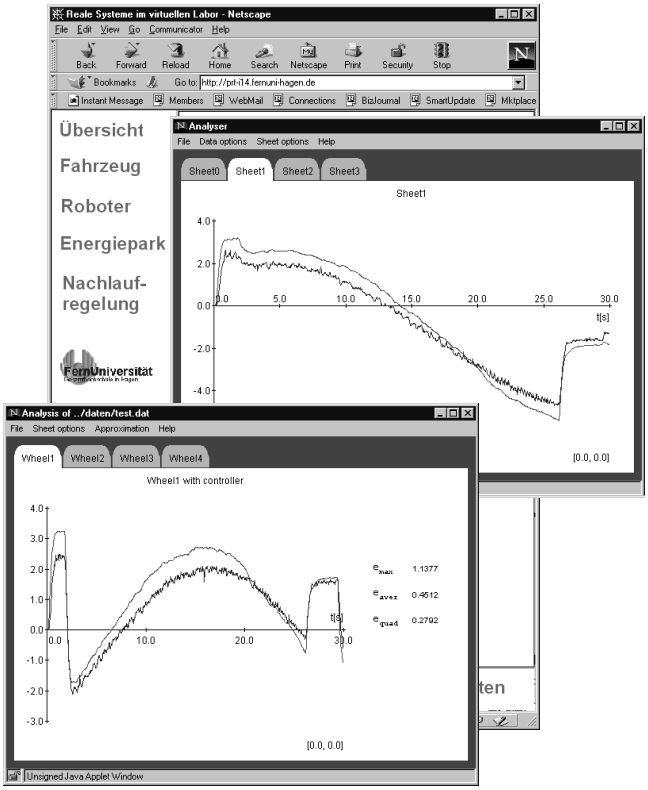

Figure 4: Analysis of the Output Data

4 Conclusion

The paper shows that distance education can be applied to laboratory experiments. On the client part, there are only some minor requirements. A student can use the Virtual Lab with a Web browser and Java runtime environment. In the laboratory, the experiment must be adapted to the requirements of remote control. The Communication server contains only software that is freely available. Since open standard protocols for the video/audio transmission are used, different software products on the client's side are applicable. The developed software has been implemented in the programming language Java. Therefore, it can be easily adapted to other platforms. To support teamwork, the Virtual Lab can be extended with video conferencing tools. In this case, only one student at a time has access to the experiment. At different stages of the experiment, various students can and should have access to the experiment. The other students can observe the experiment and discuss the results in order to modify the controller interactively.

References

[1] Project Page http://prt.fernuni-hagen.de/virtlab

[2] Röhrig, C.; Jochheim, A.: Remote Control of

Laboratory Experiments, Proc. of the 19th ICDE World Conference on Open Learning and Distance Education, Vienna, June 99

[3] Real-Time-Linux: http://www.rtlinux.org

[4] Bolot, J.; Hoschka, P.: Sound and Video on the Web, http://www.inria.fr/rodeo/personnel/hoschka/WWW5tutorial.ps.gz

[5] Sun Microsystems: Java Multimedia Framework, http://www.java.sun.com/products/java-media/jmf/index.html

[6] UCL Networked Multimedia Research Group: MBone Conferencing Applications, http://www-mice.cs.ucl.ac.uk/multimedia/software/

[7] Parnes, P.: mTranslator http://www.cdt.luth.se/~peppar/progs/mTranslator/

[8] Network Research Group at the Lawrence Berkeley National Laboratory: http://www-nrg.ee.lbl.gov/vic/

[9] UCL Networked Multimedia Research Group: http://www-mice.cs.ucl.ac.uk/multimedia/projects/rat/

[10] RFC 1889 RTP: A Transport Protocol for Real-Time Applications http://www.normos.org/rfc/rfc1889.txt

[11] RFC 1890 RTP: Profile for Audio and Video Conferences with Minimal Control http://www.normos.org/rfc/rfc1890.txt