Remote Experimentation in a Collaborative Virtual Environment

Andreas Bischoff and Christof Röhrig
Department of Electrical Engineering
University of Hagen
D-58084 Hagen, Germany

Abstract

In engineering education, concepts taught through lectures are often complemented by laboratory experimentation. Students can observe dynamic phenomena that are often difficult to explain by written material. Furthermore, interactive experimentation on real world plants improves the motivation of the students and also develops an engineering approach to solve realistic problems. This contribution presents a collaborative virtual environment for a remote laboratory. Students have access to the remote laboratory via Internet from anywhere at any time. They control the experiments exclusively with their standard Web browser, no additional software is needed. The collaborative environment allows the experimentation in a team. The group is able to interact and to discuss the results of their work. A real collaboration like in local experimentation is possible. The remote laboratory is based on a client/server architecture, which is mainly implemented in the Java programming language. The methods and software modules developed for the laboratory are generic and are frequently used in several remote experiments at some German universities. Furthermore the contribution discusses the motivation of remote experimentation.

1 Introduction

In distance teaching, laboratory experimentation is inconvenient because the students usually have to be physically present in the universities labs. One solution to avoid this disadvantage is virtual experimentation. In this paradigm the experiments are simulated and visualized by means of virtual reality [2]. Simulation is a proper way to complement engineering 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.
Another concept to avoid the disadvantages of local experimentation is teleoperation of laboratory experiments. Early implementations are reported in [3] and [4]. Providing remotely accessible experiments, unique or expensive equipment can be shared between several universities. So, a larger number of laboratory resources is available, and students can choose from a variety of lab experiments.

2 System Design

The main design idea of the remote experimentation 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 the client software. A Web Server is the interface between the client and the experiment. The Web itself provides the infrastructure to exchange the necessary information.

2.1 Analysis of Requirements

Computer based controller implementation is necessary for remote experimentation in the field of 'control theory'. Another important aspect is to transport the feeling of a real experiment to the remote user. A video and audio broadcast can provide the remote user with the feeling of being physically present at the location of the real experiment. With the visual feedback, the user supervises the experiment and checks whether the process performs as expected. For system identification purpose and controller evaluation, it is necessary to collect the relevant data of the process. These data have to be stored on the server for download and additional processing by the students. As only a group of students 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 experimentation time in advance. They carry out the whole booking procedure by themselves in order to choose the time most appropriate to their needs. The main requirement on the server side is the safety of the experiment and of the server computer. The experimental plant has to be protected against any action that can damage or destroy it. For this reason, all commands given to the plant controller must be analyzed and dangerous controller settings have to be avoided. If the controller algorithms can be defined by the user without any restrictions, system instabilities caused by the controller are hard to detect in advance. To evaluate students work, it is useful to log all communication between user and experiment.
To provide remote experimentation to a group of students, typical synchronous communication techniques like videoconferencing are not suitable because of bandwidth limitations. A video-conference with more than two communication partners is a typical point-to-multipoint application. If a true collaboration of all partners is desired, the

partner with the smallest bandwidth limits the communication. Our application of a remote lab requires real interaction between the students and the tutor, so a bandwidth-saving way of interaction is required as an alternative to the video based communication. Pure text-based communication (Chat) does not meet our requirements because a multi-user remote lab application needs the possibility of real interaction. The tutor has to be enabled to introduce and to explain the details of the experiment by some kind of visual representation of the experiment. In this collaborative virtual environment only one student at a time has active access to the experiment.

2.2 Collaborative Environment

Virtual collaborative environments brings together users, which are geographically distributed, but connected via a network. This not only means that the users will be able to easily communicate, but also collaborate. The technology behind any synchronous collaboration tool is a mechanism that enables a user to send updates to other users about the interactions that are made in the shared environment. It is necessary for the participants to have the same view of the application in real time [5]. A comparative summary of the different standards for collaborative environments is given in [6].

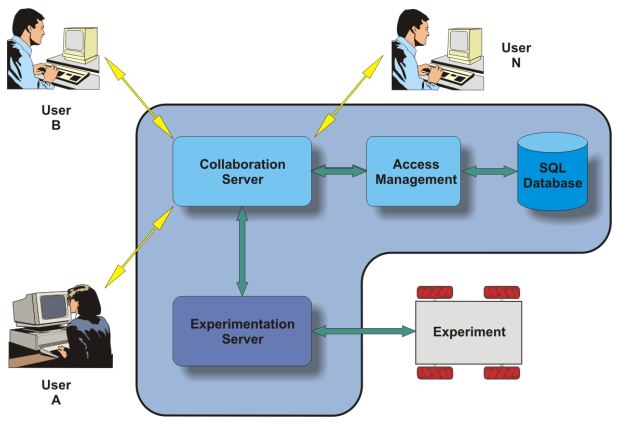


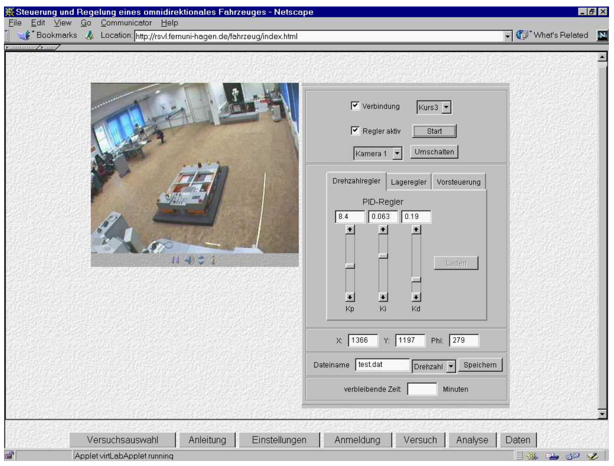
Figure 1: Communication Structure

2.3 Communication Structure

The communication structure is based on a client/server architecture written mainly in the Java programming language. Students may work on any platform that supports a Web browser with a Java runtime environment. Java is used to eliminate the operating system problem of heterogeneous environments, such that users are not restricted in their choice of a resource. This is specially important for distance education since some users might choose UNIX-workstations, while others might prefer Windows 95/98/NT PCs or MACs. The introduction of Java helped to overcome these problems. 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 Web server provides the HTML documents, the VRML scene, VRML avatars and all Java applets. The server hardware includes a video capture card and a sound card for video and audio grabbing. The real-time controller of the experiment is usually implemented on a different computer hardware with a real-time operating system. The communication structure is generic. Therefore, it can be used for 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.
The collaborative environment is divided into two main modules: a rendering and graphics part on the client side and a communication middleware on the server side. On the client side VRML is used to display the virtual 3D environment. VRML as a text-based language is a powerful, nevertheless simple language to build 'virtual worlds', which include 3D objects, light sources and animations. VRML specifies an External Authoring Interface (EAI) which can be used by external applications to monitor and control the VRML environment. This is used to update the virtual world with data of the experiment and positions of the other users. User interfaces via comfortable Java applets can be build in order to give access to the VRML environment and to allow higher-level modifications. The communication middleware is based on the open-source VRML-Multi-User-Software VNET which realize its functionality by Java-VRML coupling via the EAI. VNET itself is a pair of client and server software implemented in Java [7]. The server is realized as a Java application which communicates with all clients and provides them with updates of the 3D-scene. The client applet controls the local VRML-browser-plugin via the EAI to update the scene (the positions of other users) and senses the local user movements to send new positions to the server. The communication protocol between client and server is well documented, so it is possible to interface the VNET multi-user software with the remote lab server-system. The lab server acts as an additional client of the VNET server. This additional client module provides the VNET server with the position and orientation changes during the remote experiment. The remote user is able to see the virtual representation (the avatar) of the experiment (a vehicle) in her/his VRML-browser window.

2.4 Access Management

Scheduling of experimentation time is done by an access management system. Appointments are stored in a SQL database. The user interface of the access management system is implemented in Java and executed in the Java runtime environment of a Web browser. It is subdivided into two parts, an administrator interface and a student interface. The administrator interface consists of several dialog boxes for creating and deleting accounts, for setting up individual time-quotas, for defining time slots and for analyzing the logging messages. More details of the access management system are described in [8].



Location: http://rsvl.fernuni-hagen.de/fahrzeug/index.html

[✓] Verbindung: Kurs3
[✓] Regler aktiv
Kamera 1

Kp: 8.4 Ki: 0.063 Kd: 0.19

X: 1366 Y: 1197 Phi: 279

Dateiname: test.dat Drehzahl

verbleibende Zeit: Minuten

Applet virtLabApplet running

Figure 2: Experimentation Interface

2.5 Experimentation Interface

At the date of the experiment, the user connects to the server with a Web browser. The browser loads the Web page of the experiment. A live video stream for viewing the experiment and an optional audio stream helps to provide a laboratory feeling. Two Java applets are included in the page. One applet receives the live video and audio stream of the laboratory and the other applet controls the experiment. Details of the implementation are described in [9]. The user interface for controlling the experiment is realized in Java. After the Web page is loaded, the applet starts and an authentication dialogue and a control window pop up. The user 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. The server reports the total time still available for the current experimentation. If no more time is left, the server cuts the students connection to the experiment. The communication between client and server is executed as data telegrams using the open TCP/IP connection. Figure 2 shows the Web browser with the video/audio and control applet. Details of the experiment itself are reported in [10].

3 Conclusion

The contribution shows that distance education can be applied to real laboratory experiments. Even collaboration in a team is possible. Remote laboratory users are able to interact and discuss the results of their work. A real collaboration like in local experimentation is possible. On the client side, there are only some minor requirements. Students are able to use the experiments with a Web browser and Java runtime environment. In the laboratory, the experiment must be adapted to the requirements of remote control. The server exclusively uses software that is available for free. All software developed for the lab has been implemented in the programming language Java. Therefore, it can be easily adapted to different platforms.

References

[1] "Project Page: Real Systems in the Virtual Lab," http://prt.fernuni-hagen.de/rsvl/.

[2] C. Schmid, "A Remote Laboratory Using Virtual Reality on the Web," *Simulation*, vol. 73, no. 1, pp. 13–21, 1999.

[3] B. Aktan, C.A. Bohus, L.A. Crowl, and M.H. Shor, "Distance Learning Applied to Control Engineering Laboratories," *IEEE Transactions on Education*, vol. 39, no. 3, pp. 320–326, 1996.

[4] J. Henry, "Controls Laboratory Teaching via the World Wide Web," in *Proceedings of the ASEE Annual Conference*, Washington, USA, 1996.

[5] S. Shirmohammadi, J.C. Oliveira, and N.D. Georganas, "Applet-Based Telecollaboration: A Network-centric Approach," *IEEE Multimedia*, vol. 5, no. 2, pp. 64–73, 1998.

[6] J.C. Oliveira, S. Shirmohammadi, and N.D. Georganas, "Distributed Virtual Environment Standards: A Performance Evaluation," in *Proceedings of the 3th IEEE/ACM International Workshop on Distributed Interactive Simulation and Real Time Applications*, Greenbelt, USA, Oct. 1999.

[7] S. White and J. Sonstein, "VNet," http://www.csclub.uwaterloo.ca/symbol126sfwhite/vnet/.

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

[9] C. Röhrig and A. Jochheim, "The Virtual Lab for Controlling Real Experiments via Internet," in *Proceedings of the 11th IEEE International Symposium on Computer-Aided Control System Design*, Hawaii, USA, Aug. 1999, pp. 279–284.

[10] A. Jochheim and C. Röhrig, "The Virtual Lab for Teleoperated Control of Real Experiments," in *Proceedings of the 38th IEEE Conference on Decision and Control*, Phoenix, USA, Dec. 1999, vol. 1, pp. 819–824.