A Multiuser Virtual-Reality Environment for a Tele-Operated Laboratory

Helmut Hoyer, Andreas Jochheim, Christof Röhrig, and Andreas Bischoff

Abstract—In engineering studies, concepts taught through lectures are often complemented by laboratory experimentation. This contribution presents a collaborative virtual environment for a tele-operated laboratory. Students have access to the tele-operated laboratory via the 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. Group members are able to interact and discuss the results of their work. A real collaboration, such as in local experimentation, is possible. The tele-operated laboratory is based on a client/server architecture, which is implemented in the Java programming language.

Index Terms—Control education, educational aids, laboratory education, multimedia, teleoperation.

I. INTRODUCTION

IN distance teaching, laboratory experimentation is inconvenient because the students usually have to be physically present in the universities laboratories. 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 studies, but in general, it cannot replace experiments on real plants. Since simulation is only as good as the model behind it, experimentation has the advantage of making the user aware of phenomena that are hard or even 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 tele-operated experiments, unique or expensive equipment can be shared among several universities. Therefore, a larger number of laboratory resources is available, and students can choose from a variety of laboratory experiments.

In local laboratory experiments, students usually work together in groups of two or more. This learning paradigm is often called collaborative learning. Collaborative learning develops skills for solving problems in a team. The underlying concept of collaborative learning is based upon consensus building through cooperation by group members, in contrast to competition in which individuals defeat other group members. Members of the learners group will usually organize their activities themselves and decide upon the roles of the different members via consultation and negotiation [5].

With the rapid expansion and availability of communication and information technologies, collaborative learning can also be done effectively in a virtual environment from a distance. Collaborative virtual environments bring together users who are geographically distributed but connected via a network.

II. ANALYSIS OF REQUIREMENTS

In order to build a successful setup for remote experiments, a number of requirements have to be fulfilled.

- *Telecontrol*: Computer-based controller implementation is necessary for telecontrol of the experiment. In the simplest case, users alter only parameters of the controller. In more advanced approaches, students upload the controller algorithm to the experiment [6], or the controller algorithm is optionally executed on the client computer [7].
- *Telepresence*: An 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.
- *Data collection*: For system identification purposes and controller evaluation, one must collect the relevant data of the process. These data have to be stored on the server for download and additional processing by the students.
- *Scheduling*: Since only one group of students at a time receives access to an individual experiment, schedules and exclusive access procedures to the experiment are necessary. Users 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.
- *Security*: 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.
- *Logging*: To evaluate students' work, one can have all communication between user and experiment logged.
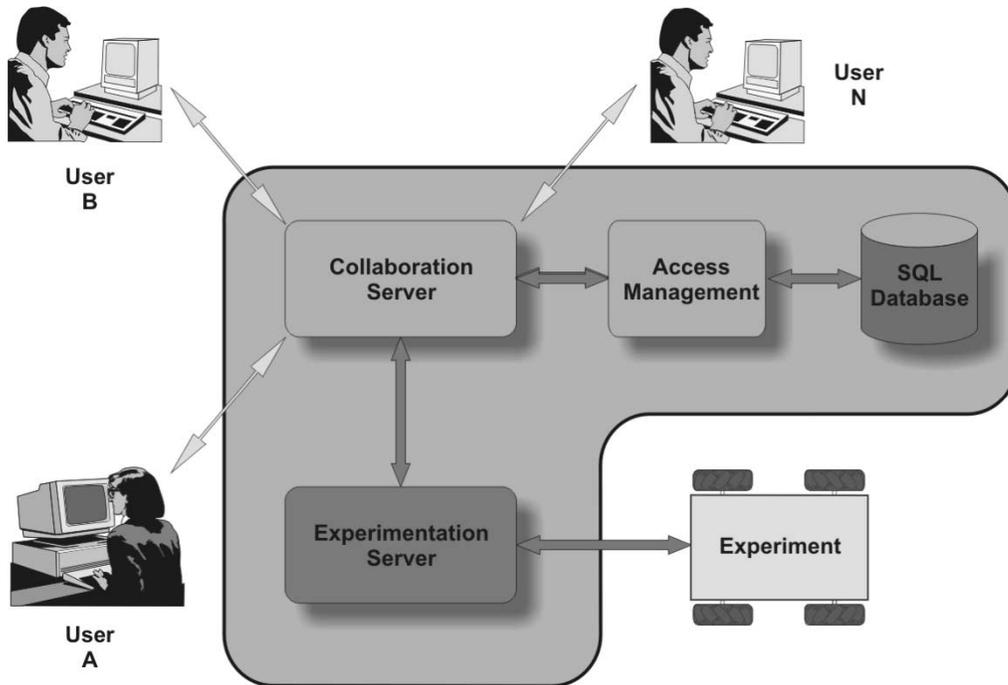
Fig. 1.   Communication structure.

- *Synchronous Communication*: To provide tele-operated experiments to a group of students, synchronous communication techniques are required.
- *Collaborative Environment*: Virtual collaborative environments bring together users who are geographically distributed but connected via a network. Thus, the users not only communicate easily, but also collaborate with each other. 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. The participants need to have the same view of the application in real time [8]. A comparative summary of the different standards for collaborative environments is given in [9].

III. SYNCHRONOUS COMMUNICATION TECHNIQUES

In conjunction with the Internet, different communication techniques are common.

- *Text Chat*: In a text chat, the users exchange textual messages by typing in a shared space. Communication in larger groups is difficult because the train of thought is difficult to follow. Text chat is suitable even with very slow Internet connections.
- *Audio Conference*: In an audio conference, the voice is captured and transmitted over Internet to other participants. The user needs a sound card with an additional microphone. The bandwidth requirement is much higher than in a text chat.
- *Video Conference*: In video conferences the additional video is used to watch other participants and to be aware of their existence and actions. A video conference has much more bandwidth requirement than a audio confer-

ence. It is not suitable for users at home with slow Internet connections.
- *3-D Chat*: In a third-dimensional (3-D) chat, an *avatar* tries to mimic the behavior of any user in a virtual reality environment. The avatar plays the role of the video in a video conference. Participants can see other users as in the real world. 3-D chats have less bandwidth requirements than video conferences do, because only events are transmitted. 3-D chat can be combined with text chat or audio conferencing.

To provide tele-operated experimentation to a group of students, typical synchronous communication techniques, such as video conferencing, 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. This application of a tele-operated laboratory requires real interaction between the students and the tutor; therefore, a bandwidth-saving way of interaction is required as an alternative to the video-based communication. Pure text-based communication (chat) does not meet these requirements because a multiuser tele-operated laboratory 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.

IV. SYSTEM DESIGN

The main design idea of the tele-operated experimentation system is to use the World Wide Web as communication structure and a Web browser as user interface. The Web browser

Fig. 2.   3-D chat interface.

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.

A.   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 run-time 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 ability is specially important for distance education since some users might choose UNIX workstations, while others might prefer Windows 95/98/NT/2000/ME/XP PCs or Macintosh computers. 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. Because of the modular structure of the system, extensions with new features are easy to implement. The Web server provides the HTML documents, the VRML (virtual reality modeling language) 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 (Fig. 1). 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 that 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 communication middleware on the server side. On the client side, VRML is used to display the virtual 3-D environment. VRML as a text-based language is a powerful but simple language to build "virtual worlds," which include 3-D objects, light sources, animations, and user representation via avatars. VRML specifies an external authoring interface (EAI) which can be used by external applications to monitor and control the VRML environment. The EAI is used to update the virtual world with the data of the experiment and positions of the other users. User interfaces via Java applets can be built 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 multiuser software VNET, which implements its functionality by Java–VRML coupling via the EAI. VNET itself is a pair of client and server software implemented in Java [10]. The server is implemented as a Java application that communicates with all clients and provides them with updates of the 3-D scene.

Fig. 3.   Experimentation interface.

The client applet controls the local VRML browser plug-in 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 that one can interface the VNET multiuser software with the tele-operated laboratory server system. The laboratory server acts as an additional client of the VNET server. This additional client module provides the VNET server with position and orientation changes during the tele-operated experiment. The remote user is able to see the virtual representation (the avatar) of the experiment (a vehicle) in her or his VRML browser window (Fig. 2).

B. Access Management

Scheduling of experimentation time is done by an access management system. Appointments are stored in a structured query language (SQL) database. The user interface of the access management system is implemented in Java and executed in the Java run-time environment of a Web browser. It is subdivided into two parts—an administrator interface and a student interface. The administrator interface consists of several dialogue 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 [11].

C. Experiment

The most important task in the experiment is the design of a position and speed control for a wheel-driven vehicle (Fig. 4). The vehicle is equipped with Mecanum wheels and electronic drives, which provide three degrees of freedom. Such a vehicle is omnidirectional, and it provides any desired motion in x and y direction, and rotation φ_z around the z axis. The drive units of the cart consist of brushless direct current (dc) servomotors with gearing and resolvers. Resolver feedback provides input to the quadrature-axis servo controller. The servo controller accomplishes commutation, control of phase currents, pulsewidth modulation, power amplification, and generation of incremental pulses. Because of the compensation of electrical time constants by means of analog current control, a nearly immediate request of a desired motor torque can be assumed. The on-board computer of the vehicle already provides different controller algorithms, which are subject to evaluation and fine-tuning during remote experiments. Details of the experiment are reported in [13].

D. 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. An optional 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 audiostream of the laboratory, and the other applet controls the experiment (Fig. 3). Details of the implementation are described in [12]. 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 log-in name and password to access the server. When the log-in process is completed, the applet opens a transport control protocol/Internet protocol (TCP/IP) connection to the access control module of the server. The server reports the total time still available for the current experiment. If no more time is left, the server cuts the student's connection to the experiment. The communication between client and server is executed as data telegrams using the open TCP/IP connection.

E. Displaying of Data

While the experiment is in progress, the input and output data of the experimentation plant are measured and stored as an ASCII file on the server. The file is platform independent and includes the input and output values of the plant and controller. Students download the file for subsequent analysis or they analyze the data online with the analyzer applet. The analyzer applet creates graphs from measured data and displays them in analysis sheets. The analyzer provides user-selected colors, automatic or manual scaling, and rastering.

Analysis sheets can be directed to a printer or stored into a file. This is a feature useful for the students' final elaboration of experimental results.
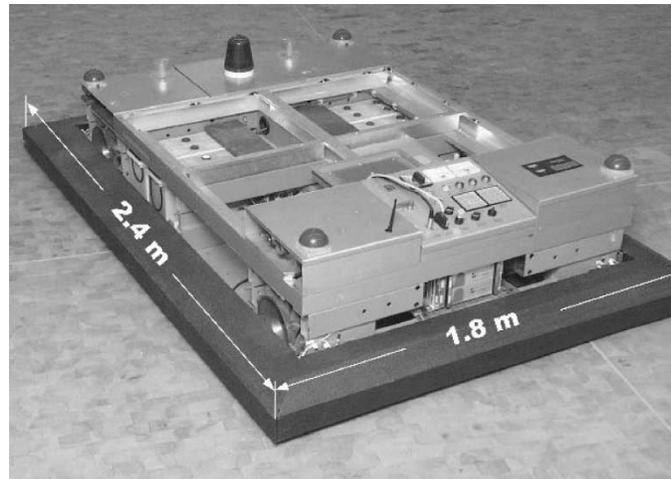
Fig. 5 shows the analyzer windows.
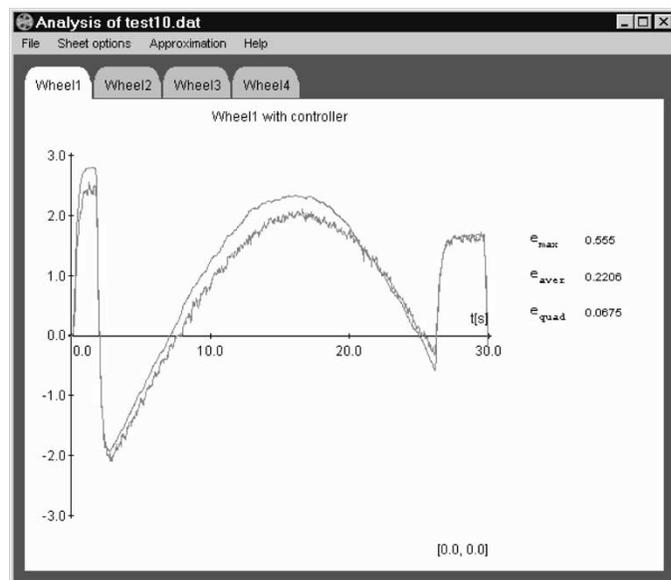

Fig. 4. Omnidirectional vehicle.

Fig. 5. Analysis of the output data.

V. 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, such as 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 run-time environment. The laboratory experiment must be adapted to the requirements of teleoperation. The server exclusively uses software that is available for free. All software developed for the laboratory has been implemented in the programming language Java. Therefore, it can be easily adapted to different platforms. All the components are usable through a low-bandwidth modem (56 kb/s) or ISDN connection. The remote experimentation without the live video stream, but including the virtual reality collaborative environment, is still usable under "worst case" conditions, such as a 9600-Bd GSM (Global System for Mobile communication) mobile-phone connection.

REFERENCES

[1] Real Systems in the Virtual Lab (2000). [Online]. Available: 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 Trans. Educ.*, vol. 39, pp. 320–326, Aug. 1996.

[4] J. Henry, "Controls laboratory teaching via the World Wide Web," in *Proc. ASEE Annu. Conf.*, Washington, 1996.

[5] P. Kirscher, "Using integrated electronic environments for collaborative teaching/learning," *Learning Instruction*, vol. 10, no. Supp./1, pp. 1–9, 2001.

[6] Y. Piguet and D. Gillet, "Java-based remote experimentation for control algorithms prototyping," in *Proc. 1999 Amer. Control Conf.*, vol. 2, San Diego, CA, 1999, pp. 1465–1469.

[7] J. W. Overstreet and A. Tzes, "An Internet-based real-time control engineering laboratory," *IEEE Control Syst. Mag.*, vol. 19, pp. 19–34, Oct. 1999.

[8] S. Shirmohammadi, J. C. Oliveira, and N. D. Georganas, "Applet-based telecollaboration: A network-centric approach," *IEEE Multimedia*, vol. 5, pp. 64–73, Apr.–June 1998.

[9] J. C. Oliveira, S. Shirmohammadi, and N. D. Georganas, "Distributed virtual environment standards: A performance evaluation," in *Proc. 3th IEEE/ACM Int. Workshop Distributed Interactive Simulation Real Time Applicat.*, Oct. 1999, pp. 14–21.

[10] S. White and J. Sonstein. (1998) VNet. [Online]. Available: http://www.csclub.uwaterloo.ca/~sfwhite/vnet/

[11] C. Röhrig and A. Jochheim, "Java-based framework for remote access to laboratory experiments," presented at the IFAC/IEEE Symp. Advances Control Educ., Gold Coast, Australia, Dec. 17–19, 2000.

[12] ——, "The virtual lab for controlling real experiments via Internet," in *Proc. 11th IEEE Int. Symp. Computer-Aided Control Syst. Design*, Aug. 1999, pp. 279–284.

[13] A. Jochheim and C. Röhrig, "The virtual lab for teleoperated control of real experiments," in *Proc. 38th IEEE Conf. Decision Control*, vol. 1, AZ, Dec. 1999, pp. 819–824.

Helmut Hoyer is head of the Control Systems engineering group and Rector at the University of Hagen, Hagen, Germany.

As rector, he promotes the University of Hagen as a virtual university. His research interests are in the area of service robotics.

Andreas Jochheim was head representative of the Control Systems engineering group at the University of Hagen, Hagen, Germany, from 1997 to 2001. He is currently head of the Control Engineering Department at Hesse & Knipps GmbH, Paderborn, Germany.

Christof Röhrig received the Ph.D. degree in electrical engineering from the University of Hagen, Hagen, Germany, in 2003.

He is a Professor of Computer Science at the University of Applied Sciences Dortmund, Dortmund, Germany. His research interests are in the area of motor control, robotics, and telematics.

Andreas Bischoff is a Research Assistant with the Control Systems engineering group of the University of Hagen, Hagen, Germany.

His research interests are in the area of virtual reality and multimedia technologies.