Engineering with Computers (2008) 24:405–415
DOI 10.1007/s00366-008-0089-y

ORIGINAL ARTICLE

Computational framework for remotely operable laboratories

Prakash Kripakaran · Abhinav Gupta · Vernon C. Matzen

Received: 13 July 2006 / Accepted: 29 October 2007 / Published online: 29 February 2008

Abstract Decision-makers envision a significant role for remotely operable laboratories in advancing research in structural engineering, as seen from the tremendous support for the network for earthquake engineering simulation (NEES) framework. This paper proposes a computational framework that uses LabVIEW and web technologies to enable observation and control of laboratory experiments via the internet. The framework, which is illustrated for a shaketable experiment, consists of two key hardware components: (1) a local network that has an NI-PXI with hardware for measurement acquisition and shaketable control along with a Windows-based PC that acquires images from a high-speed camera for video, and (2) a proxy server that controls access to the local network. The software for shaketable control and data/video acquisition are developed in the form of virtual instruments (VI) using LabVIEW development system. The proxy server employs a user-based authentication protocol to provide security to the experiment. The user can run perl-based CGI scripts on the proxy server for scheduling to control or observe the experiment in a future timeslot as well as gain access to control or observe the experiment during that timeslot. The proxy server implements single-controller multiple-observer architecture so that many users can simultaneously observe and download measurements as a single controller decides the waveform input into the shaketable. A provision is also created for users to simultaneously view the real-time video of the experiment. Two different methods to communicate the video are studied. It is concluded that a JPEG compression of the images acquired from the camera offers the best performance over a wide range of networks. The framework is accessible by a remote user with a computer that has access to a high-speed internet connection and has the LabVIEW runtime engine that is available at no cost to the user. Care is taken to ensure that the implementation of the LabVIEW applications and the perl scripts have little dependency for ease of portability to other experiments.

Keywords Internet-enabled experiments · Remote control · Remote observation · Web-based experiments · LabVIEW · Distance education

P. Kripakaran (✉)
Applied Computing and Mechanics Laboratory, Ecole Polytechnique Fédérale de Lausanne, GC G1 567, Station 18, 1015 Lausanne, Switzerland
e-mail: prakash.kripakaran@epfl.ch

A. Gupta · V. C. Matzen
Department of Civil, Construction and Environmental Engineering, North Carolina State University, Raleigh, NC 27695, USA

1 Introduction

Recent developments in web technologies provide an excellent opportunity to form a network of laboratories that allow control and observation of experiments via the internet. Experimental research laboratories that are geographically far apart can be networked together to create research collaboratories that promote engineering research and decision-making. With this goal in mind, NSF is currently sponsoring the network for earthquake engineering simulation (NEES). NEES (http:// www.nees.org) is a simulation resource that is composed of geographically distributed state-of-the-art experimental research equipment sites that are specifically designed to advance earthquake engineering research and education. Its objective is to make experimental laboratories with facilities like

shaketables for simulating earthquakes, available for remote participation and control. Such a network of laboratories can also be useful to provide a complete educational experience in distance-education programs in which theory is complemented with hands-on experiments [1–4].

Even though the internet is readily available as the underlying infrastructure for communication, significant modifications to the existing experimental configurations are required. Sophisticated hardware for data acquisition and control are needed to interface measurement and control devices with a computer. Appropriate software that can graphically display acquired measurements and allow manipulation of control and measurement devices must be developed. Novel IT architectures that create a secure framework by integrating web technologies with the system for data acquisition and control are needed. Moreover, sufficient safety and security protocols are required to prevent intentional or unintentional damage to the experiment and its surroundings. The framework should also support the single controller-multiple observers concept. This capability will enable geographically distributed engineers or students to simultaneously monitor and download measurements from the experiment as one person controls the loading on the test specimen.

Another facet of an internet-enabled experiment is to provide the remote user with real-time video of the experiment. Typically, structural engineering-related experiments that involve earthquake motion evaluate structural behavior at a frequency of 1–20 Hz. Capturing this motion and communicating it in real-time requires a high frame-rate digital camera and appropriate computer hardware for acquiring images from the camera. Image compression techniques have to be explored to reduce the amount of data transmitted to the remote user. In most cases, there is a trade-off between image compression and image quality. As with the measured data, multiple users may want to simultaneously watch the video of the experiment and download it on to their personal computer. Also, decision-makers sometimes employ strobe-lights while doing vibration tests in the laboratory for viewing mode shapes of the test structure. A similar capability in an internet-enabled experiment is highly desirable for educational purposes.

In this study, we propose a computational framework that enables the creation of such research collaboratories. We illustrate the framework by implementing it for the remote control and observation of a shaketable experiment. The framework has two key components: (1) a local network with computers that perform data/video acquisition and shaketable control using LabVIEW virtual instrument (VI) applications, and (2) a proxy server that controls access to the local network. The proxy server uses a user-based authentication protocol to provide security to the experiment. This protocol also provides the proxy server with the identify of the user, which is used to process user actions like request for a timeslot to observe the experiment and request to gain control of the experiment. The proxy server has forms that allow users to schedule time for observing or controlling the experiment on a future date. A user can access the experiment at this scheduled timeslot by using the forms provided on it. The proxy server uses the schedule along with the authentication protocol to control access to the LabVIEW computers. The implementation of the forms on the proxy server and that of the applications for data acquisition and shaketable control have little dependency. The system ensures that only that user scheduled as controller can decide the waveform input into the shaketable. On the other hand, multiple users are permitted to schedule for simultaneously observing and downloading the measured data from the experiment. Provision is also created for multiple users to watch a live video feed of the experiment. The images for the video are captured using a high frame-rate camera. We have evaluated two different approaches to transmit real-time video through LabVIEW applications. One approach utilizes JPEG compression of the acquired images and the other employs converting the image into an array of numerical values that represent the intensities of the gray-scale image. The former, while providing faster video than the latter, requires the remote user to download additional software. The only requirements this framework places on the remote user's computer are the following two, (1) a high-speed internet connection like DSL or cable, and (2) the LabVIEW run-time engine [5] which is available at no cost to the user.

2 Shaketable experiment

One way the seismic behavior of structural systems and various structural control architectures is studied by using a shaketable to simulate earthquake excitations. The shaketable has a platform to mount the structural system under consideration, and accelerations and displacements at different locations on the structural system are measured while the ground motion is applied through the shaketable. In this study, we consider a shaketable experiment that is used in a laboratory course to illustrate concepts in structural dynamics for undergraduate and graduate students. However, the illustrated computational framework is and can be easily adapted to other laboratory experiments.

Figure 1 shows the laboratory setup of the shaketable experiment considered for illustration of the computational framework. It consists of a $12'' \times 34''$ (0.3 m $\times$ 0.86 m) one-dimensional shaketable and a 100 lb (445 N) electromagnetic shaker. The test specimen is a single or multi-story

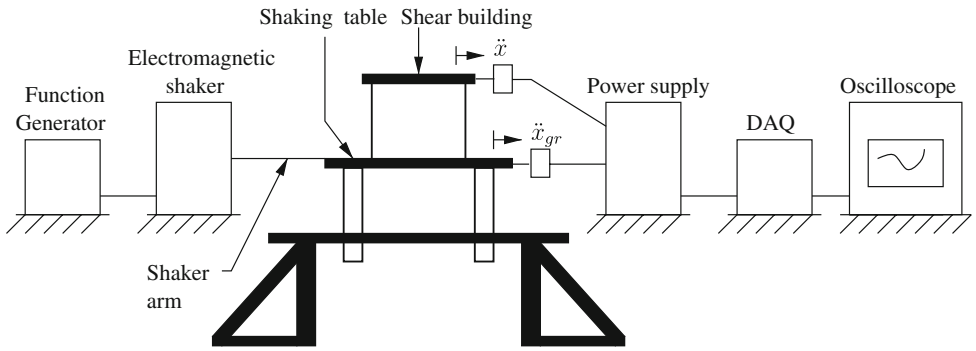

Fig. 1 Laboratory setup of shaketable experiment

shear building having wide but thin aluminum columns and heavy steel girders. Forced vibration tests are conducted by applying a harmonic excitation to the table using a function generator. The input frequency of the excitation is increased in steps from a value that is lower than the natural frequency of the structure to one that is much higher. For each input frequency, the table is excited for a reasonable duration to ensure that the structure vibrates in steady state motion, and then the acceleration response is measured using accelerometers mounted on the different floors and viewed using an oscilloscope.

To transform this experiment to one that can be controlled and monitored through the internet, significant monetary investment on hardware as well as programming effort for appropriate software is needed. Remote monitoring of the response of the test specimen requires sophisticated data-acquisition hardware that can enable communication between the measurement devices like strain gauges and accelerometers, a high-speed camera and a computer. Similarly, remote operation of a shaketable requires hardware that enables the computer to operate the electro-magnetic shaker.

Wirgau et al. [6] studied the viability of using national instruments (NI) hardware and software for data acquisition and control. They developed programs to communicate with the NI hardware in the form of VI using NI's LabVIEW development system [7]. In particular, two VIs were created for generation of waveform input and for display of measurements. For remote control, a LabVIEW technology referred to as Remote Panels [8] was employed. In this method, the LabVIEW webserver on a NI PXI [9], which is directly connected to the experiment, supported the previously described two VIs. It permitted access to these VIs from only those computers whose IP addresses were already registered on the PXI. A remote user was required to register the IP address of his or her computer with the PXI by providing it to the administrator. Then only could a user control the experiment or observe the response through an internet browser from that computer. Real-time video of the experiment was provided by using Microsoft's netmeeting software with a web camera.

The implementation by Wirgau et al. [6] was restricted in generality due to the following reasons:

- The remote panels technology allowed only one user to use the VI for viewing and downloading the response of the structure.
- The system was designed to accept connections only from computers whose IP addresses were on a list maintained by the LabVIEW webserver. Such a scheme is inflexible as the system is intended for use by authorized engineers via the internet irrespective of their IP address.
- A higher likelihood of intentional damage to the system by malignant users on the internet existed, since the PXI was directly connected to the internet.
- A scheduling facility did not exist. Such a facility would allow users to sign up for controlling or monitoring the experiment on a future date.
- The web camera used did not sufficiently capture the high-frequency motions resulting from a shaketable experiment.
- Netmeeting software did not ensure the high video transmission rate that is required for remotely viewing the real-time video of a dynamics experiment.

In the following sections, we propose a computational framework that addresses all of these aspects. We describe first the hardware and network setup of the framework and then the software architecture implemented over this hardware.

3 Hardware setup of framework

The computational framework is composed of a network of three computers: (1) a Linux-based proxy server, (2) a NI PXI [9] for data acquisition/control, and (3) a Windows-based PC with hardware for video acquisition. The networking of the computers is illustrated in Fig. 2. An IBM Thinkpad with the Fedora Linux operating system is used as the proxy server. Figure 2 shows that the proxy server serves as the gateway to the experiment for a remote user

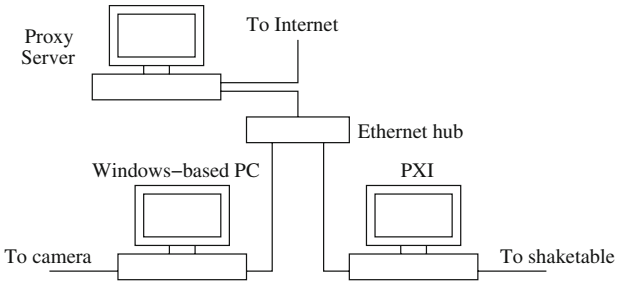

Fig. 2 Network setup of the computers

on the internet. It has two network adapters, one for communication with the internet and another for the computers on the local network.

The PXI uses an NI 8176 controller and runs LabVIEW in a Windows environment. The PXI does not directly perform the data acquisition or real-time control. Instead, it has a control board with a dedicated processor and memory. LabVIEW programs can be embedded into the board for real-time control as well as data acquisition. This facility protects the experiment from any software or hardware related accidental failure in the host PXI as the embedded program can safely shutdown the experiment during such an event. The accelerometers mounted on the test specimen are connected to this board through BNC connectors. Similarly, the board is also wired to the shaketable so that the generated waveforms can be communicated to the shaketable for real-time control. This PXI is the same one that was previously used in the study by Wirgau et al. [6].

Real-time video of the experiment is obtained using a Windows-based PC that has an NI-IMAQ card (PCI-1429) mounted on its PCI express bus. The IMAQ card offers a high data transfer rate of 680 MB/s. A high-speed Basler camera (Model: A504 K) that uses a progressive scan CMOS sensor is used to capture the video. The camera produces 8-bit monochrome images and can capture images at a maximum speed of 500 frames per second and a resolution of 1,280 × 1,024 pixels. This frame rate is sufficient for shaketable experiments as the frequencies that are of interest often lie within 20 Hz. A zoom lens, which can be controlled using software on the computer, is attached to the basler camera. Since the camera requires a bright setting for producing good images, powerful commercial video lamps are employed for additional lighting.

4 Software architecture

The software architecture essentially consists of two types of components: (1) LabVIEW applications for data acquisition, video and control, and (2) web technologies for

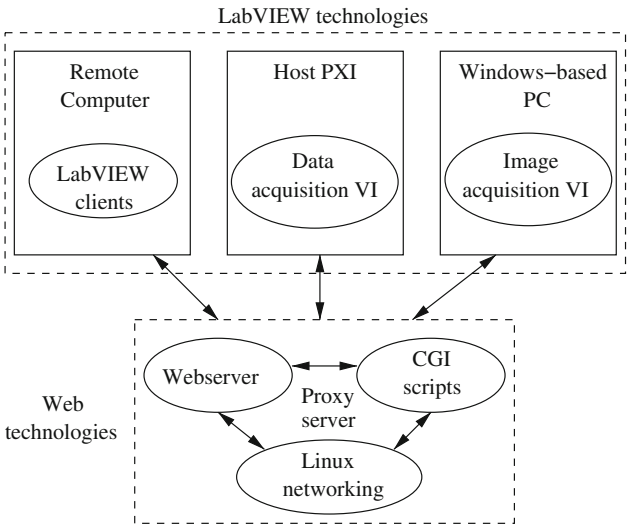

Fig. 3 Software architecture

experiment scheduling and authentication. The interaction between these components is illustrated in Fig. 3. While the LabVIEW applications are completely independent of the web technologies, the web technologies do use some information about the implementation of the LabVIEW applications. Consequentially, we first describe the LabVIEW applications, which is followed by a description of the web technologies.

4.1 Data acquisition and control

As described earlier in this paper, Wirgau et al. [6] implemented VIs for data acquisition and control and used them in combination with LabVIEW Remote Panels for remote observation of the experiment. The architecture of their VIs is schematically illustrated in Fig. 4. For control of the shaketable, a VI that generates the waveform with given input amplitude and frequency is run on the host PXI. Another VI that is embedded on the real-time board of the PXI receives the generated waveform and communicates it to the shaketable. These two VIs always maintain an active TCP/IP connection for communication. The VI on the host PXI is made available for the remote user using LabVIEW Remote Panels technology. The LabVIEW webserver is activated on the host PXI and it is configured to accept connections from a specific remote computer. The remote user can control this VI through a internet browser by connecting to the LabVIEW webserver on the host PXI. Our framework uses a similar implementation for granting control of the experiment to the remote user.

Wirgau et al. [6] proposed a similar implementation for data acquisition. This is schematically shown in Fig. 5. A VI was run on the real-time board of the PXI to read measurements from the accelerometers. These were communicated

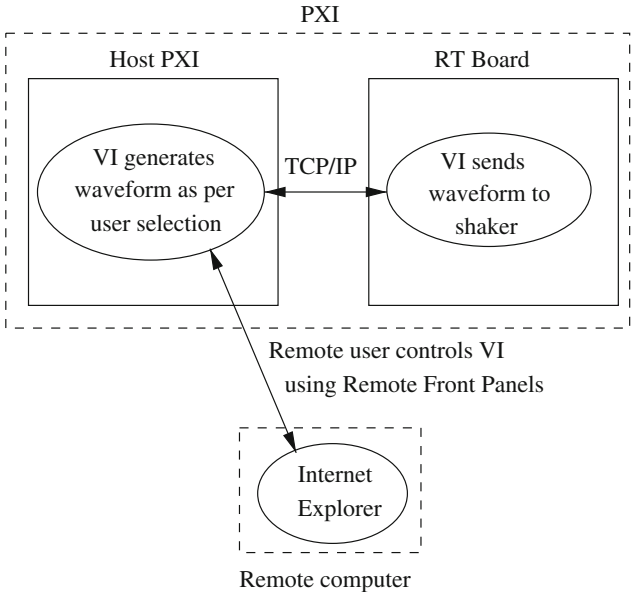

Fig. 4 Wirgau et al.'s [6] framework for shaketable control

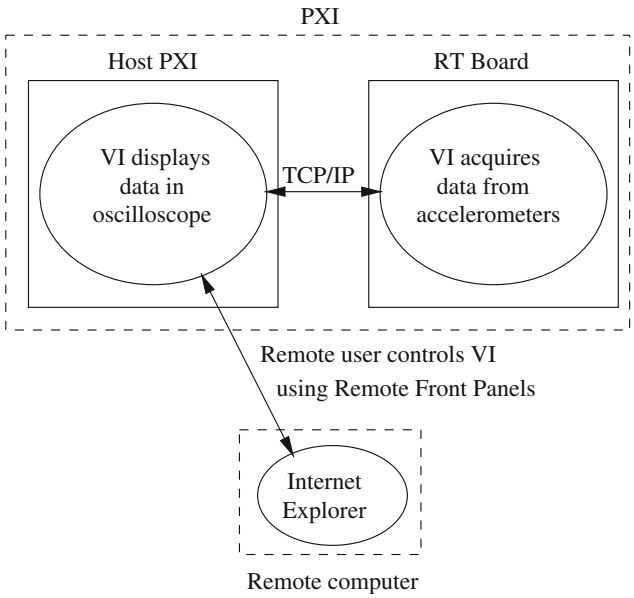

Fig. 5 Wirgau et al.'s [6] framework for data acquisition

using TCP/IP to another VI on the host PXI. The VI on the host PXI displays the measured data using a simulated oscilloscope. It also provides interfaces to manipulate the display in the oscilloscope. For instance, the user can freeze the display to examine a specific frame, say, for calculating the dynamic magnification factor. The remote user gains access to this VI using LabVIEW remote panels. When implemented in this manner, only one remote user can control this VI and hence, observe the measurements. While it was suitable for controlling the shaketable, it is desirable to provide the capability for multiple observers to simultaneously monitor the experiment. Therefore, we have modified the implementation to use LabVIEW's Datasocket technology. The new implementation enables multiple users to simultaneously view and download the acceleration measurements.

LabVIEW's Datasocket technology is specifically aimed at distributing measurements to geographically distributed users. The computer that is used to make the measurements writes them to a Datasocket server that may be running on the same computer or on a different computer. All users irrespective of geographical location can obtain the measurements by using a LabVIEW application that subscribes to the Datasocket server. Thus, our implementation of data acquisition using datasockets consists of two types of applications: one for the PXI that will gather data using the appropriate devices and publish it to a LabVIEW Datasocket server, and a second that will be used by each of the remote users for observing the measurements.

The proposed framework for data acquisition is given in Fig. 6. The VI that is embedded on the real-time control board reads data from the accelerometers. It always maintains a TCP/IP connection with the VI on the host PC. The VI sets the acquisition rate of the control board to 1,000 scans/s. During each scan, the control board acquires the voltage values returned by the accelerometers. The VI uses a loop structure such that it acquires data corresponding to 100 scans during every loop iteration. The acquired data is stored in a $100 \times m$ array, where the number of rows correspond to the number of scans and the number of columns m to the number of accelerometers, respectively. This array of data acquired in each loop iteration is communicated to the VI on the host PXI. The VI on the host PC uses the accelerometer calibration information to convert the acquired voltage values into meaningful acceleration values. It then writes the acceleration values to the Datasocket server that is running on the Windows-based PC. To reduce the number of write

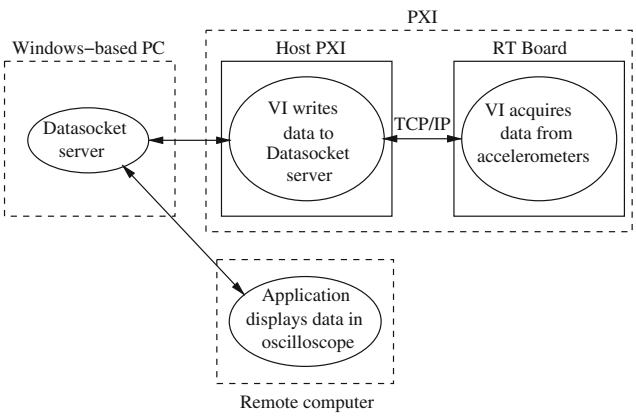

Fig. 6 LabVIEW components for data acquisition

operations to the Datasocket server, the VI accumulates 1,000 scans of data and then writes it to the datasocket server. The Datasocket server is configured to accept connections from computers with VIs that wish to read the acceleration values.

To view the accelerations, the remote user downloads a LabVIEW executable that is created from a VI. This executable connects to the Datasocket server running on the PXI and reads the accelerations. The executable also possesses various graphical interfaces that enable the user to examine the acceleration measurements. In particular, there is a knob control that the user can adjust to set the number of data points and hence, the number of wavelengths to be shown in the oscilloscope. The user can visualize a fewer number of datapoints in the oscilloscope using this control, even though the Datasocket server provides acceleration measurements in chunks of only 1,000. The client executable makes this possible by breaking the data read from the Datasocket server into smaller pieces as determined by the setting on the knob control. These pieces are sequentially processed by the oscilloscope. Additional interfaces like a button control to freeze a particular dataframe in the oscilloscope and a tool for moving the scales in the oscilloscope are also present. The VI also provides facilities to download the acquired data in a format that can be viewed in a spreadsheet application like Microsoft Excel. The application writes the acquired data to a specified spreadsheet file whenever the corresponding switch is turned on.

4.2 Video acquisition

Programs for video acquisition are developed using NI's image acquisition (IMAQ) drivers. The camera produces images at a rate that can be set using IMAQ drivers. Real-time video is provided by continuously streaming these images to the remote user. These images, which are obtained through the IMAQ hardware, are in a unique format that requires installation of IMAQ drivers for viewing. This can be a hindrance to the remote users as they may have to download a fairly large-sized set of drivers. To avoid this issue, the images can be converted to a different format that the remote user can view without installing additional software. Another important consideration while implementing real-time video, is related to the size of the images transmitted to the remote user. If the images are large, the remote user may experience significant delay between consecutive images due to the increased communication time. The images can be sent in a compressed form as a solution. Multiple alternatives are available with regard to handling these issues, and these alternatives determine the factors like size and quality of the image that is communicated to the remote user. For example, when the images are converted to a JPEG form, the reduction in size of the image is often achieved at the expense of some information loss in the final JPEG image. These factors play a significant role in determining the performance of the real-time video of the experiment.

It must be stated that the video quality also depends on many other factors in addition to the above-mentioned. Some of these are mentioned below:

1. Network bandwidth: the video speed is limited by the weakest link (lowest bandwidth network) in the route taken by the video packets.
2. Number of observers: it is expected that the communication time will increase if the number of observers simultaneously using the experiment increase.
3. Network traffic: network traffic varies through the day and so, the observer may experience different video speeds at different times.

While the effect of the number of observers can be understood using some simple experiments, the effect of network bandwidth and network traffic are much more difficult to examine. The authors are currently testing the system in a distance education course with the aim of understanding the effect of these factors. However, the results presented in this study are from tests conducted within the university network with only few observers simultaneously using the system.

In this study, we have evaluated two techniques to provide real-time video. These two methods only differ in the image processing algorithms used to modify the images before communicating them to the remote user. Otherwise, both the methods function in a similar manner. The working of this framework is illustrated in Fig. 7. As in the case of measurement communication, both methods use LabVIEW's Datasocket technology for communicating real-time video of the experiment. There are two VIs: (1) the VI on the Windows-based PC continuously acquires images from the IMAQ interface, processes them and writes the resulting images to the Datasocket server, and (2) the VI on the remote client connects to the Datasocket server and displays the images that are received. The image processing techniques that differentiate the two methods are described below.

- *JPEG streaming:* In this method, the VI on the host PXI converts the acquired IMAQ images to a JPEG stream. The conversion algorithm accepts an integer parameter between 0 and 100 that corresponds to the desired

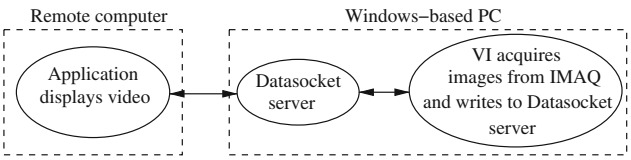

Fig. 7 LabVIEW components for video acquisition

quality of the resulting image. The image quality increases with increase in the value of the parameter but so does the size of the resulting image. Since the camera produces an 8-bit monochrome image with a resolution of 1,280 × 1,024 pixels, the size of the original image is approximately 1 MB. After experimenting with the value of the parameter, we have found that a value of 90 gives a good compromise between the size and the quality of the image. The size of the image after compression is close to 42 KB, which is considerably smaller than the size of the original image. This image is written to the Datasocket server. The main advantages of this method are: (a) faster communication because of the smaller image size, and (b) absence of any noticeable delay between consecutive images at the remote computer. The main disadvantage of this implementation is that it requires the remote computer to have IMAQ drivers to display the received images even though the images were converted to a JPEG stream format at the PXI.

- *Intensity graph:* One approach that avoids having IMAQ drivers at the remote computer involves using LabVIEW's intensity graph interface. The acquired image is converted to a two-dimensional array of integer values that represent the color intensities of the corresponding pixel in the image. The VI on the host PXI writes these arrays to the Datasocket server. The application on the remote computer continuously downloads these arrays from the Datasocket server. The application generates the images by plotting these arrays in an intensity graph. The color scale of the intensity graph is adjusted to gray-scale as the camera generates a monochrome image. The intensity graph is continuously refreshed with new arrays from the Datasocket server. Thus, the remote user will see the real-time video of the experiment in the intensity graph. While the IMAQ drivers are not required for the remote user, this implementation can involve higher communication time between the remote client and the server than the previous method. The reason is that the size of the communicated arrays can be significantly larger, i.e., of the order of 1 MB. One way of reducing the size of the communicated data is to shrink the acquisition window for the camera. For example, if the experiment can be captured in a window of 850 × 800 pixels, then the size of the communicated data is only 425 KB. This is considerably smaller than the amount of data generated from a full acquisition window.

Table 1 summarizes the differences between the two techniques. The table gives estimates of the amount of data required to communicate a single image to the remote user when using the two techniques. It is observed that the method using intensity graph sends more data and hence requires significantly larger network bandwidth. Moreover, the JPEG compression does not significantly reduce the quality of the image. Therefore, the JPEG compression technique is a better overall solution and will deliver better performance over a wide variety of networks. The intensity graph method is a better alternative only in the presence of very high bandwidth such as in the case when the remote user and the experimental site are located in the same LAN.

5 Web architecture

The LabVIEW applications described earlier in the paper use one of the following two technologies for communication with the remote user: (1) Datasockets, and (2) remote panels. Datasockets are used for sensor and video data communication while the remote panels are used for remote shaketable control. The Datasocket server and the LabVIEW webserver, which are the programs that support the two technologies, operate by listening on certain network ports. The proxy server controls access to these network ports using perl-based CGI scripts within a webserver and thereby provides security to the experiment. The key actions performed by the proxy server from the moment a user logs into the system to observe or control the experiment until the time when a user disconnects from the system is illustrated using a flowchart in Fig. 8. As seen from the flowchart, the following sequence of steps are involved:

1. The remote user is authenticated by the proxy server using WRAP [10], a web-based authentication mechanism.

Table 1 Comparison of two video transmission techniques

Features	JPEG compression	Intensity array
IMAQ drivers	Required at both host and remote computer	Required only at host
Data size	42 KB	450 KB
Effect of reducing acquisition window	Reduces data size	Reduces data size
Computation	Compression is computation intensive	Computationally inexpensive
Suggested use	Over any network	Suitable only for high band width networks

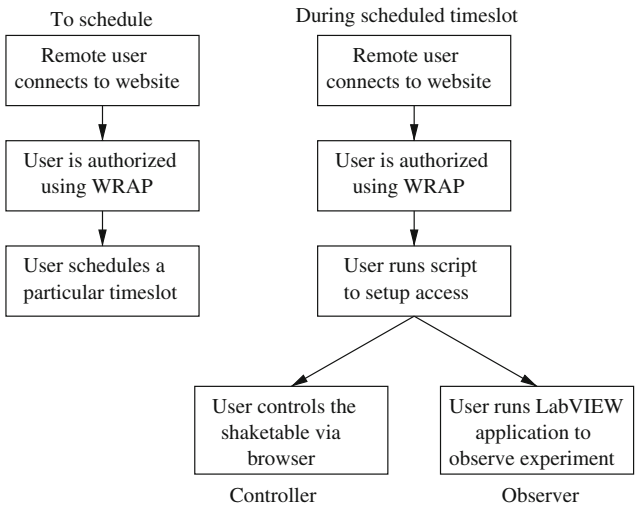

Fig. 8 Flowchart showing remote user actions to access the experiment

2. The user schedules a particular timeslot for controlling or observing the experiment on a future date.
3. On the scheduled date, the user connects to the proxy server and is authenticated.
4. The user requests access to control or monitor the experiment by running a CGI script on the webserver.
5. The CGI script checks if the user is scheduled to monitor or control the experiment during the current timeslot.
6. If the user is scheduled, the script then sets up port forwarding on the proxy server so that the user can access the appropriate server on the LabVIEW host.
7. The user runs a LabVIEW application that is downloadable from the website to observe the experiment. To control the experiment, the user uses a browser as explained previously in the paper.
8. Port forwarding that was setup in the previous step is disabled when the current timeslot expires.

These steps are performed using the following components, which constitute the web architecture.

5.1 User authentication

North Carolina State University uses a cookie-based authentication protocol referred to as WRAP [10]. This authentication protocol is used to verify the authenticity of users with access to the university's computing facilities. Students and university employees have a username and password that can be used in various computing labs around campus. We have used this authentication protocol for providing internet security to the experiment.

WRAP is a web-based authentication mechanism to verify the identity of a user without requiring the user to login to each individual webserver within the university domain. In this authentication mechanism, the user obtains an encrypted cookie, called the WRAP cookie, from a SSL-secured server by using his/her username and password. Whenever the user visits a WRAP-protected website, the browser sends the WRAP cookie to the website. The website verifies if the cookie is genuine and also obtains the username for that user. If the user does not already possess a WRAP cookie or possesses an invalid cookie, the user will be forwarded to the SSL-secured server that will issue a WRAP cookie to the user. WRAP cookie components like the username are available as environment variables within CGI scripts. The CGI scripts can, therefore, recognize the user making the request.

The apache webserver on the proxy server is configured to use WRAP. File directories that contain the CGI scripts are protected using WRAP. Thus, the remote user is forced to obtain a WRAP cookie before attempting to run the scripts. If an authorized user is making a request, the CGI scripts use the environment variables to recognize the user. This information is later used in scheduling the user for the experiment as well as setting up remote access to the experiment for the user.

5.2 Experiment scheduling

A user can schedule to either control or observe the experiment in a timeslot within the next 10 days. The system provides eight timeslots for the experiment each day. Each timeslot is $2\frac{1}{2}$ hours long and there is a 30 min interval between the timeslots for the laboratory personnel to make adjustments, if necessary. The system maintains two timetables that keep track of the users currently registered to use the experiment. One timetable is for users registered to observe the experiment while the other is for users registered to conduct the experiment. The latter timetable has only a single user for a given timeslot whereas the former timetable can have multiple users registered in any given timeslot. This is consistent with the objective of allowing multiple observers but only a single controller.

When a user wants to schedule for controlling the experiment, the user can navigate to the 2-week timetable from the homepage of the proxy server website. The user can choose a timeslot by filling in the two text boxes on the form and hitting the submit button. If the timeslot is currently unused, the user will be registered into the timeslot and the updated table will be listed. If the timeslot is already taken, the user will be displayed a corresponding message. To schedule as an observer to the experiment, the procedure is similar with the only difference being that the forms do not check for availability of a selected timeslot because multiple observers are permitted.

5.3 Single-user control

To initiate experiment within their scheduled timeslot, the user first visits the website. The activities performed by the proxy server to setup access for the remote user are given in a flowchart in Fig. 9. As shown in the figure, once the user is authenticated using WRAP, the proxy server checks whether the user is already scheduled to control the experiment. If scheduled, the script permits the remote user to access the LabVIEW webserver on the host PXI. The remote user uses a browser to access an http link returned by the script. The user must use internet explorer as the browser since the LabVIEW runtime engine is only configured to work with it.

Access to the LabVIEW webserver is setup using the networking concept of packet forwarding. All TCP/IP packets arriving from the remote user's computer of the proxy server are forwarded to the host PXI on the listening port of the LabVIEW webserver. There are two software components involved in setting up packet forwarding. One is the CGI script that is run on the webserver when the user visits the website. The other is a perl script that is always running on the proxy server. Figure 10 shows the interaction between the two scripts in a sequence diagram. The CGI script identifies the IP address of the remote user's computer. It then verifies if remote access is already enabled for the user. For this purpose, the script uses a data file that keeps track of the various users currently using the system. If the user is already logged in, the system simply informs the user that the connections are already active and that the user needs to visit the returned http link. If the user is not logged into the system, it communicates the IP address and the duration left in the timeslot to the perl script. The perl script then enables packet forwarding for packets arriving from the remote user's computer. The perl

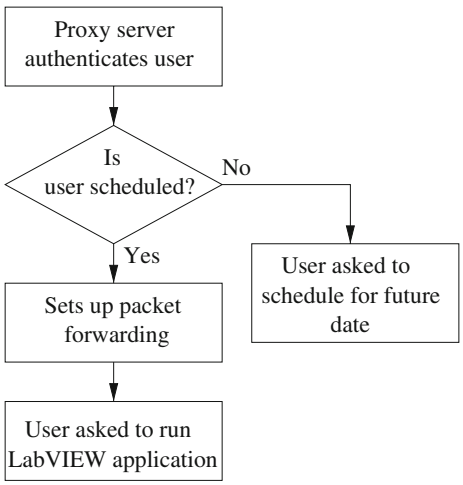

Fig. 9 Flowchart showing proxy server actions for enabling remote access

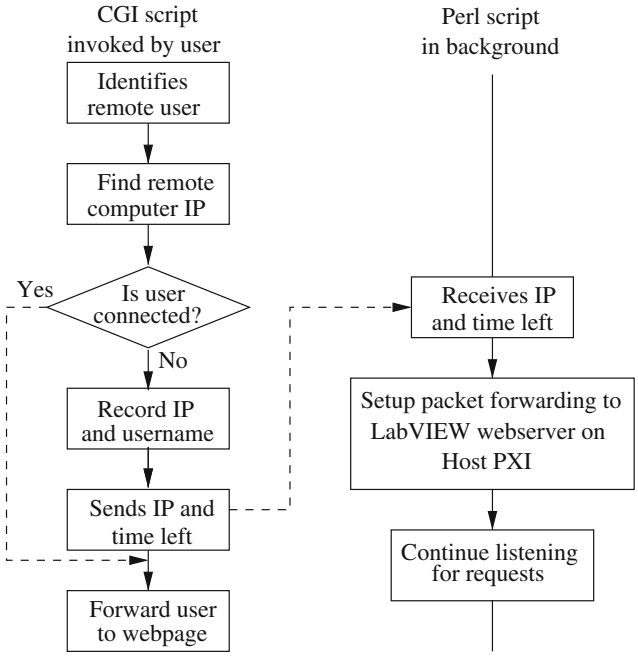

Fig. 10 Sequence diagram of actions in giving remote user access to control the experiment

script also ensures that the packet forwarding is disabled after the expiry of the allotted time for the user.

The reason for using a perl script different from the CGI script to setup packet forwarding lies in concerns regarding network security. Also, using a single script that is always active on the proxy server provides a way of queuing multiple requests. Even if the user runs the CGI script multiple times, the requests are processed sequentially as all the CGI requests are finally processed by a single perl script.

5.4 Multi-user observation

The implementations for access to monitor the measurements and to view the real-time video are similar in many ways to the implementation discussed above for an access to control the experiment. As discussed earlier in the paper, the remote user will view the video and acceleration measurements using two different LabVIEW applications. These applications will attempt to connect to the Datasocket server on the Windows-based PC in the local network. Therefore, the scripts in this case enable access to the TCP/IP port on which the Datasocket server listens.

Figure 11 shows a sequence diagram representing the interaction between the CGI script activated by the remote user and the perl script that is always active on the proxy server. The key difference between Figs. 11 and 9 lies in the TCP/IP port to which packet forwarding is enabled. The scripts in this case setup packet forwarding to the listening port of the datasocket server. It is noted that this

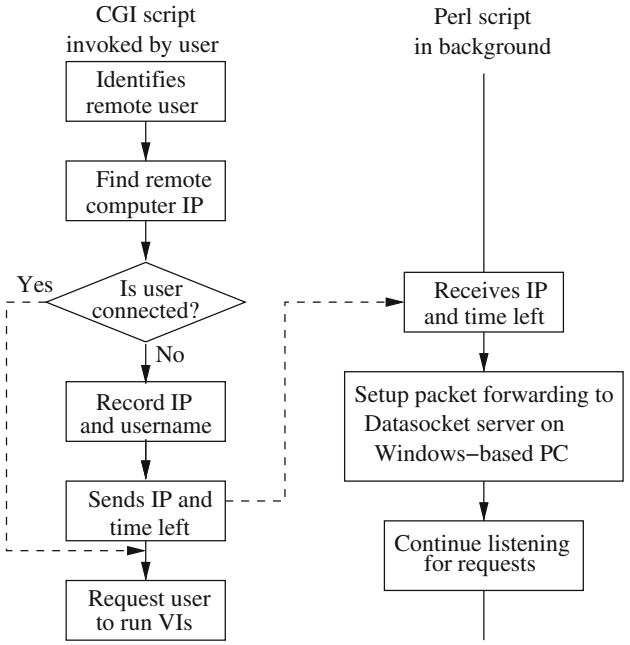

Fig. 11 Sequence diagram of actions in giving remote user access to monitor the experiment

mechanism enables access to both video and acceleration measurements as both are delivered via the Datasocket server.

5.5 Experiment administration

The software architecture, while automating the operation of the laboratory, should also provide sufficient administrative tools to the laboratory administrator. In the event that the experiment needs to be stopped, the laboratory administrator must be able to gain control of the experiment. Since the remote user who is controlling the experiment accesses the corresponding VI through remote front panels, the laboratory administrator can easily gain control of the VIs by directly accessing them through the host PXI. Also, the CGI scripts record the username and the IP address of the remote users currently using the system in a data file. The administrator can use the information in the file to disconnect any user from the system by disabling packet forwarding between the remote user's computer and the appropriate computer on the local network. The administrator can modify the weekly schedule as well as the daily schedule to make any changes.

6 Summary and conclusions

Recent computing advances in web technology have given rise to new ideas in web-based experimentation for supporting engineering research and education. The National Science Foundation (NSF) is sponsoring the development of a national Network for Earthquake Engineering Simulation (NEES) that will make large-scale laboratories with seismic testing facilities available for control and observation to geographically distributed researchers. Numerous hurdles exist in the various tasks from creating applications that will enable a remote user to perform data acquisition and control of the experiment to developing novel security and safety protocols for preventing intentional or unintentional damage to the laboratory experiment. In this study, we have proposed a computational framework as a solution. The framework uses a combination of National Instruments hardware and software, and web technologies to create a comprehensive internet-based environment that implements a single-user multiple-observer model for remote control and observation of laboratory experiments. The proposed framework is illustrated for a shaketable experiment. LabVIEW applications that use Datasocket technology and remote panels are developed for data acquisition and control, respectively. Video of the experiment is acquired using high-speed cameras and NI's IMAQ software. Video transmission using two techniques—JPEG compression and intensity graph, is explored. The study concludes that the JPEG compression method is relatively more reliable. The computers that operate the shaketable experiment are in a local network, which is protected using a proxy server. The webserver uses NC State University's WRAP authentication protocol to provide web security. Remote users are able to access the laboratory schedules on the website, which is hosted by the proxy server, and schedule for controlling or observing the experiment on a future date. Within a scheduled timeslot, the appropriate remote user is given permission to control or monitor the experiment after the user is authenticated by the website. The main conclusions from this study are:

- LabVIEW remote panels technology is an efficient solution if only a single user needs to remotely access the laboratory experiment. Such an implementation is suitable for the controller in a single-controller multiple-observer framework.
- LabVIEW Datasockets technology enables simultaneously communicating data to users distributed across the internet. This technology is ideal for communicating sensor and video data to remote observers in a single-controller multiple-observer framework.
- The JPEG compression technique is relatively better than the intensity graph method for video transmission. It provides a suitable compromise between the size of the communicated data and the image quality received at the remote user.
- A webserver with suitable user-authentication protocol, CGI scripts and Linux networking concepts are

sufficient to create a secure web-based environment to control access to laboratory experiments.

- The web technologies and LabVIEW VIs are decoupled to a large extent. The dependencies between the two components are primarily with respect to the TCP/IP listening ports of the LabVIEW webserver and the Datasocket server.
- The computational framework is easily extendable to other laboratory experiments. The major task in adapting the framework to a new experiment is the development of new LabVIEW VIs that are necessary for control and data acquisition with respect to the new experiment.

Acknowledgments This material is based upon the work supported by National Science Foundation under grant no. DUE-0310845 and by the Department of Civil Engineering as well as the College of Engineering at the North Carolina State University.

References

1. Corradini ML, Ippoliti G, Leo T, Longhi S (2001) An internet based laboratory for control education. In: Proceedings of the 40th IEEE conference on decision and control, Orlando
2. Bohus CL, Crowl A, Aktan B, Shor MH (1996) Running control engineering experiments over the internet. In: Proceedings of the 13th IFAC world congress, San Francisco, paper no. 4c-03
3. Poindexter SE, Heck BS (1999) Using the web in your courses: What can you do? what should you do? IEEE Control System 9(1):83–92
4. Gupta A, Gabr MA, Matzen VC (2004) Alternatives in the implementation of internet-enabled laboratory. In: 2004 ASEE annual conference and exposition, Salt Lake City
5. NI: NI LabVIEW Runtime Engine (2005) http://digital.ni.com/softlib.nsf/-websearch/
6. Wirgau S, Gupta A, Matzen V (2006) Internet-enabled remote observation and control of a shake table experiment. J Comput Civil Eng 20(4):271–280
7. Wells L, Travis J (1997) LabVIEW for everyone. Prentice Hall, Englewood Cliffs
8. NI: NI LabVIEW Remote Panels (2005) http://sine.ni.com/nips/cds/view/p/la-ng/en/nid/11017
9. NI: NI PXI (2006) http://www.ni.com/pxi/
10. NCSU: WRAP (2005) https://webauth.ncsu.edu/wrap/