PROGRESS IN NATURAL SCIENCE Vol. 15, No. 12, December 2005

Network platform for remote structural testing and shared use of laboratories*

XIAO Yan[1,2]**, HU Qing[2,3], GUO Yurong[2], ZHU Pingsheng[1] and YI Weijian[2]

(1. Department of Civil Engineering, University of Southern California, Los Angeles CA90089, USA; 2. School of Civil Engineering, Hunan University, Changsha 410082, China; 3. FutureNet Technologies Corporation, Monrovia, California, USA, and Nanchang, China)

Received January 25, 2005; revised June 20, 2005

Abstract This paper presents a network platform developed for remote testing of scaled model or prototype structures and structural elements. The network platform, NetSLab, was developed based on the client/server concept along with a proposed data model and the communication protocols. The platform is capable of transferring the control and feedback data as well as signals among remotely located structural testing laboratories or computers connected by Internet. Several concepts were introduced to develop the platform and to provide relatively easy and friendly interface for applications and further enhancement. Trial and actual tests were successfully carried out at the Hunan University, China and the University of Southern California, USA. Models simulating bridge piers and piles or building substructures were subjected to recorded earthquake ground motions, and the tests were controlled remotely over the Internet using the platform.

Keywords: communication protocol, data model, earthquake simulation, Internet, remote testing.

The development and evolution of modern structural engineering highly depend on experimentation. Particularly, due to the complexity of earthquake mechanisms and the highly nonlinear performance of structures under extreme earthquake loading, experimental research plays an important role in developing seismic design methodology for engineering structures. Experimental research serving as an efficient mean to fill the gap between any analysis and the reality can lead the way to advance theories. For large-scale and complicated structures or new systems which are not covered by existing design codes and specifications, their behaviors cannot be confidently predicted by existing analytical tools, however, experimental testing can provide performance data needed for design, construction as well as operation of such structures.

With the development of mechanical engineering, computer, control and sensor technologies, modern structural experimentation has become capable of testing larger-scale systems with more proper, realistic and accurate simulations of complicated working conditions, using advanced equipment such as actuators, shake tables, centrifuges, hybrid testing facilities, etc.[1—4],1).

Recent development of Internet technology provides new opportunities for structural experimentation. The Internet based communication enables the transfer and sharing of the vast data and particularly the control signals, feedback information among geographically separated laboratories and facilities. Scientific laboratories thus networked can then be utilized to test more complicated systems. Several attempts have been made to carry out collaborative tests on structural systems with remotely located laboratories[5—8],2). One of the most comprehensive efforts might be the Network for Earthquake Engineering Simulation (NEES) project[7], which invested several expensive facilities and established a network for potentially more efficient sharing of resources for earthquake engineering research.

To facilitate the structural research collaborations between the University of Southern California (USC) and the Hunan University (HNU) of China as well as

---

* Supported by National Natural Science Foundation of China (Grant No. 50338020), Hunan University and the University of Southern California

** To whom correspondence should be addressed. E-mail: yanxiao@usc.edu, cipres@hnu.cn

1) Assessment of earthquake engineering research and testing capabilities in the United States. Earthquake Engineering Research Institute, Proceedings: Document WP-01A, Summary Report: Document WP-01. September 1995.

2) Xiao Y. and Yi W.J. Proceedings of Workshop on Modern Structural Experiment (in Chinese). Hunan University, Changsha, China, November 28-29, 2002.

other institutes, a research effort was initiated in 2000 to establish an Internet based network platform, now named the Networked Structural Laboratories, or Networked Scientific Laboratories (NetSLab). This paper presents the development of the network platform NetSLab and its application.

1 Networked structural laboratories (NetSLab)

1.1 Proposed data model for NetSLab

The network platform is developed for earthquake response simulation of structures or models using several remotely located laboratories and computers connected by Internet. Such distributed structural testing can be organized as a series of steps. In each step, each participant completes one event according to the mutual communication among the participants, generates and sends additional resulting data. In other words, a step can contain several events. In order to make this process work properly, a controller is introduced. The controller organizes the testing procedure and coordinates the working flow of all participants as well as the data flow in the testing.

In this paper, we propose a generalized data model to abstract such distributed testing procedures and attempt to cover all possible situations in this area. The data model consists of the following three components: testing results, three types of participants, and communication protocols and dataflow.

1.1.1 Testing results The ultimate goal of distributed structural testing is to obtain a series of testing results. According to the proposed data model, the testing results are represented by an array TR,

$$TR = TR(i,j), \quad i = 1,N;\ j = 1,M. \quad (1)$$

Here, N is the maximum number of the steps and M is the maximum number of the events within a step. Each row of the array represents the testing results in one step, and each element of a row represents the testing result of a testing event, i.e. $TR(i,j)$ is the result from step i and event j. Note that $TR(i,j)$ may depend on the results from previous stages.

By this definition, the testing process can be considered as how to generate TR according to certain initial conditions and certain roles in the proposed data model.

1.1.2 Three types of participants Based on the definition of TR, the participants' tasks in a testing are processing TR with different means. Based on their roles in the testing, all participants can be categorized into three groups:

1) Controller: who organizes the testing procedure, controls the testing progress and data communication among all participants, and stores or publishes the testing results. There is only one controller in one testing for the current study.

2) Testers: who carry out the detailed task event(s) for each testing step to create new results according to previous results. There may be multiple testers in one testing. The testers can be further categorized into two types: virtual testers, who use computation to provide analytical TR, and physical testers, who operate the actual testing equipment to generate results TR.

3) Observers: who monitor the process and share the testing results without any interference of the test processing.

From the point of view of Data Model, the controller initiates and maintains TR, designates the data flow, decides how to transfer data among the testers and publishes the partial or final consolidated TRs. Each tester reads the data of partial or whole TR, creates one or more new values based on their testing facilities or algorithms and forwards the result to the next tester (or controller).

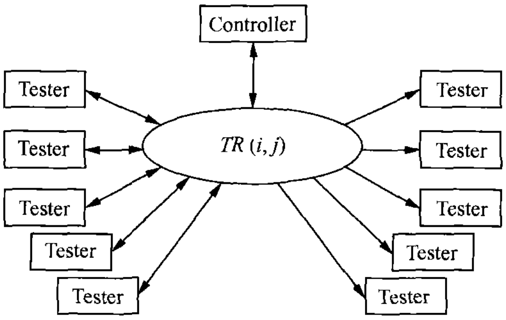


Fig. 1. Roles of participants and their interactions in the proposed data model.

1.1.3 Communication protocols and dataflow The data to be circulated among all the participants correspond to all kinds of testing results and control information. In order to keep the testing running smoothly and efficiently, it is important to specify how these data are circulated. We used the Finite State Automata (FSA) model to describe the dataflow during the testing process. In general, an FSA is decided by its state transfer function,

$$F(C, u) = N, \tag{2}$$

where C is the current state, u is the input, and N is the next state.

In the proposed data model, the state transfer function F is

$$F(C, u, H) = (N, \alpha), \tag{3}$$

where H is the historical data (partial TR), i.e. each node has a certain memory to maintain some historical data, and α is an action which specifies what to send and send to whom.

According to the definition and properties of action α, three types of data communication protocols are introduced:

1) Centralized sequential communication: controller sends control data (called request) to each tester one by one and a tester responds back to the controller only after it finishes the controller's request. The order of data transfer is determined before the testing or testing steps, i.e. F is essentially independent of u and H, and at any moment, only one tester communicates with the controller.

2) Centralized parallel communication: controller sends requests to multiple testers simultaneously and waits for their response. As soon as obtaining the responses from the testers, the controller decides and starts the next step to send out the next round of requests.

3) Distributed parallel communication: controller only sends out the start signals and initial data to certain testers. When a tester receives the partial TR and request, it executes certain action and decides its own action to send request and the testing result to someone else. At certain point, a tester may report the results back to the controller.

1.2 Implementation of NetSLab

Since the final testing system will be used by many non-computer professionals for many different needs of structural testing, it is important to keep the simplicity of the platform for understanding, further development and maintenance. Many tests may require a near real-time response which also requires the final system to operate as quickly as possible under the given network environment. Therefore, we introduce two new concepts to fulfill these goals.

1) Dynamic Unified Data Packet (DUDP). This data packet covers all communication related components: current stage data of TR, controller's request, testers' current status and communication requirements, etc. With such concept, only one type of data packet is needed for traveling among all participants, providing the simplicity and reliability to the system. DUDP Parser is the software module to parse a given DUDP and to help any entity (controller, tester or observer) understand DUDP efficiently and correctly.

2) Generalized Data Communication Agency (GDCA). This is the unique communication components in our testing system, which can be attached to its master entity, such as a controller, testers or observers. It accepts DUDP from its master entity, sends it to the given destination and activates the corresponding entity if necessary. On the other hand, any incoming DUDP also goes through this agency to activate its master entity.

Based on these two concepts, data communication in a testing procedure becomes clear and simple. Any entity, whether it is a controller, tester or observer, always contains a DUDP Parser and a GDCA. For a tester, it is always waiting there, and an incoming DUDP or a user command can activate this entity. It then parses the DUDP to understand the request and corresponding data, carries out all necessary operations (including using the actual testing equipment or theoretical model with the input data to create testing results TR), determines the destine entity, generates the new DUDP and submits it to GDCA. All the testers have the same procedure for data communication and the differences are their own operation contents. This scheme is very helpful for achieving simplicity, reliability and efficiency. The controller can also be established on the tester model. Besides the tester's function, the controller has more functions, such as designating the participants for a given testing, organizing the communication protocol, specifying the initial conditions, evaluating the testing results and deciding the termination of a test.

In NetSLab, DUDP is represented by an XML formatted data structure, which provides a good flexibility for data representation, well defined format standard and a large number of processing tools. The proposed GDCA can be considered to be an ISO layer 7 protocol, which has both data presentation and communication function. GDCA receives the standard

XML formatted data DUDP from the application programs, and can be configurable to support the three communication models mentioned above. The implementation of GDCA is fully open to system designers. In NetSLab, GDCA has been implemented based on a general interface engine, UniPipe1), which provides many sophisticated lower level communication functions and a very flexible ActiveX control (including event sink) interface. Using UniPipe's diagram based script language, Action Tree, the design and maintenance of GDCA become much easier. As an ActiveX controller, it naturally supports any application programming language with standard ActiveX function, such as VB, VC++, VJ, etc.

1.3 Other system functions

As an efficient practical application system, many other issues are also considered in the implementation of NetSLab.

1.3.1 Data repository For any scientific experiment, data repository is an important issue. In NetSLab, the data repositories are considered at two levels. The controller establishes and maintains the overall data repository for the overall experiment. Each tester establishes and maintains its own data for the assigned task. GDCA provides the database interface to allow applications to easily access certain database system to store/retrieve current and historical testing results, *TR*. Since the adopted interface engine UniPipe has a built-in database interface through Microsoft ADO, it is readily for GDCA to provide and expand its database functions. The ADO is currently the mainstream database interface in the Windows/Linux systems, which supports almost all commercial databases, such as Access, MS SQL Server, Oracle, MySQL, as well as any ODBC components. Thus, the NetSLab is a database independent system with the capability to readily use any database system.

1.3.2 Fault tolerance For a reliable application system, it is important to handle the network errors during data communication. GDCA provides fault tolerance functions to application programs through two aspects:

1) The built-in fault tolerance communications inside the Interface Engine. UniPipe utilizes all current reliable communication functions of TCP/IP networks to provide the most possibly reliable communications,

2) The complete error-handling capability to report the accurate error message and keep the on-error site. The application program can easily repeat/resume the last failed operation without any problem.

Based on GDCA's fault tolerance capability, NetSLab can handle the most Internet related communication errors quite smoothly without interfering the normal operation.

1.3.3 Multimedia communications During a remotely distributed testing, many video and audio information may need to be shared among participants besides normal DUDP. In order to handle such case, interface engine UniPipe provides the auxiliary data communication channel besides the XML data and GDCA. The auxiliary data channel can be established upon the request of application program between certain points. This is a fast binary data channel which can support a large amount of binary files, DirectX stream data, and any other user-defined data. NetSLab uses such auxiliary channel to transfer A/V data among testers and/or controller. Controller can also post such A/V data to the Internet through the Web Servers.

1.3.4 Communication security Running over the public domain, secured communication in NetSLab is also an important issue. NetSLab is built with its security functions over its interface engine. Currently, three security features have been introduced to secure the data communications:

1) Windows level C3 security: Interface engine UniPipe fully supports Windows system's native security feature and password protected access-control by using Windows user access-control. Each data communication can be configured as required user authentication.

2) Standard HTTPS security channel: Besides regular channels, interface engine UniPipe also supports an additional HTTPS channel to complete its data communication. To meet this demand, a secured Web Server in the public domain will be required to serve as the data communication buffers.

3) Capability to pass through firewalls: With the help of HTTP or HTTPS data buffers, interface en-

---

1) Hu Q. et al. UniPipe: User's Manual. FutureNet Technologies Corporation, Monrovia, California, USA, 2001

gine UniPipe also supports the communications between two parties inside different firewalls. With this function, NetSLab can allow some testers to be inside certain firewalls.

Obviously, security functions are typically implemented at the cost of communication speed. In real applications, users may need to settle with some kind of trade-offs between the speed and the security.

1.4 Application issues of NetSLab

According to the proposed Data Model, DUDP and GDCA provide the necessary foundations for communication and data processing module of NetSLab. The remaining tasks are the application software developments, i.e. structural testing implementations. Specific application programs are required to be developed for testing or computation, in order to handle different application needs.

Developed by the interface engine UniPipe script, GDCA has implemented an ActiveX Control Interface. This is a language independent programming interface to the high-level application programs. Any Windows programming tools can be used to develop such application programs, as long as it fully supports ActiveX Controller, such as Visual Basic, VC++, Visual Java and C++ In the current study, most application programs are developed using Visual Basic and VC++, in which the ActiveX control is naturally integrated into the system. In addition, the development of NetSLab is targeted to the flexibility and suitability in various network environments. This is considered to be one of the important factors for the technology transfer and the promotion for wide applications.

2 Remote simulation analysis of earthquake response

Non-linear time-history analysis is generally accepted to study earthquake response of a structure. However, due to its complexity and large amount of computation, this method is used mainly in research or earthquake designs of the important and complex structures. For a non-linear analysis program, one of the most difficult parts is how to construct the hysteretic models for various structural elements. Because different types of structural members (beams, columns, and walls) may use different materials and configurations, different hysteretic models have to be used to describe their behaviors. To develop a non-linear time-history program, one needs to construct a model for the structural system, select appropriate numerical integration method and equation solver, and develop various hysteretic models.

Based on the characteristics of non-linear time history analysis, we can separate the hysteretic modeling from the main program. Computers with programs handling one or several types of hysteretic modeling are considered to be virtual testers in NetSLab, and can be located remotely[9]. A remote analysis to simulate the earthquake response of a structure can be considered to be a virtual remote testing.

The concept of remote analysis of nonlinear time-history simulation of structural response is schematically shown in Fig. 2. The virtual testing controller located in PC-A handles the input of the structure's data and ground accelerations, selects solution methods for the equation of motion, sends displacement data to the remote virtual testers PC-B, PC-C or others, reassembles the equation of motion once receives the restoring force data, and decides the events for the next step.

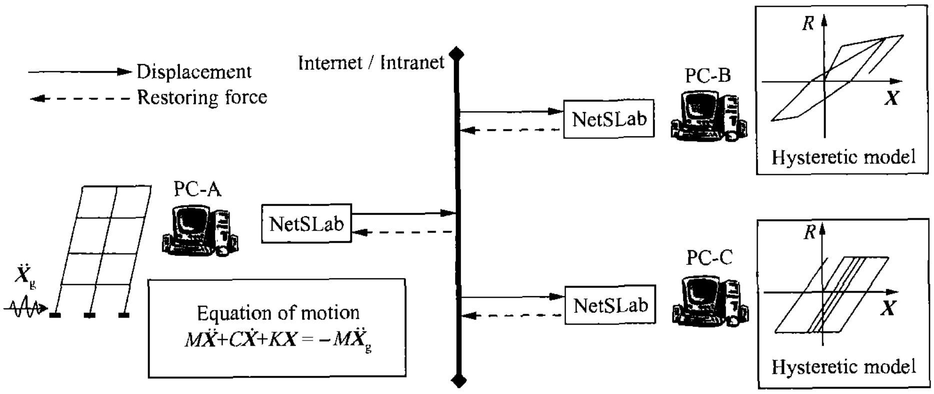


Fig. 2. Schematic diagram of remote earthquake response simulation using NetSLab. ($\ddot{X}_g$: ground acceleration; M: mass; C: damping; K: stiffness; X: displacement)

In the authors' experience, such virtual tests are particularly important and useful in evaluating the adequacy of the remote testing platform NetSLab. Remote analyses allow the investigators to conveniently evaluate the platform under various simulated possible testing conditions, thus can significantly reduce the need for expensive physical tests at the development stage. Needless to say, remote analysis has its own significance in engineering applications.

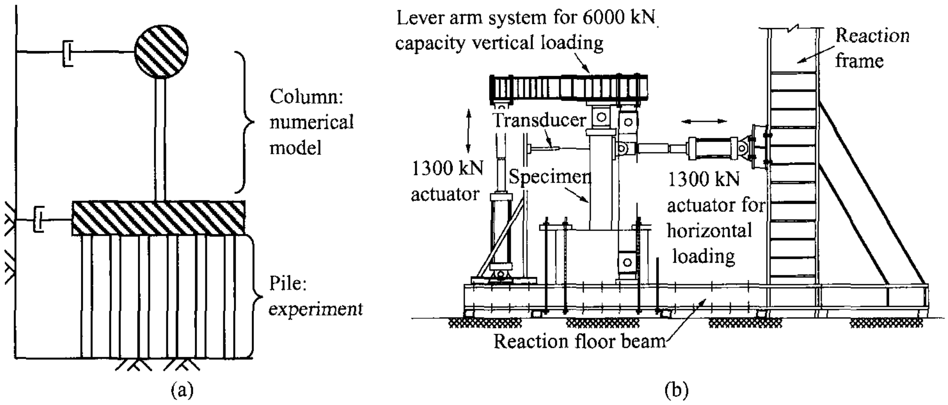
3 Application of NetSLab for remote pseudo dynamic testing

3.1 Equipment

Trial applications of NetSLab were carried out with the development of several specific application programs for simulating earthquake response of structures. The current study is a collaboration effort between the University of Southern California and the Hunan University, and both have adequate structural testing facilities. At this stage, only the large-scale pseudo dynamic testing functions of the two laboratories are mobilized. The participating USC equipment is a large-scale structural element testing facility[10]. Two 150 ton capacity Parker actuators are controlled by a two-axial control box. The participating equipment of the Hunan University is its multiple actuators and multi-axis loading system. The loading system currently has eleven Schenck actuators with capacities ranging from 10 ton to 65 ton, and two MTS actuators of 120 ton capacity. The control system includes a 4-axis MTS controller and a custom-developed multi-axis control platform, suitable for controlling actuators with different manufacturers.

3.2 Pseudo-dynamic testing

One of the trial applications of NetSLab for remote pseudo dynamic testing was carried out for testing a full-scale model of single column bent bridge with precast prestressed concrete pile foundation, using the large-scale testing frame at USC. The prototype bridge was selected from one of the typical California bridges[11,12]. The experimental concept for simulating the earthquake response of single-column bent bridge structure in the direction transverse to the bridge axis is shown in Fig. 3(a). The super structure was analyzed as a concentric mass which includes all the tributary mass from the neighboring spans. The bridge column was simulated by numeric analysis which includes a hysteresis model and the lateral force and deformation envelop. The pile response was experimentally simulated using the test setup shown in Fig. 3 (b), based on the assumption that multiplying the resistance of the pile under testing with the number of the total piles can represent the response of the pile group.



(a) (b)

Fig. 3. Experimental validation of NetSLab. (a) Virtual and actual tests; (b) test setup of precast concrete pile to pile-cap connection assembly model[12].

A program was written in Visual Basic for applying the NetSLab to the USC equipment, on the basis of the standard program Module for testers, as discussed above. This program is used to carry out pseudo-dynamic testing through Compumotor 6270 motion control-box for the Parker actuators. In the testing system, the control-box controls motions of actuators and collects responses, which include displacement and force. The program for controller to organize the test is essentially the same as that used for remote analysis of earthquake response. As shown in Fig. 4, during a test, the controller first reads an earthquake record. Based on the input information about testing structure, the controller can calculate the displacement step for the testing element and the analytical element. The controller then sends the displacement command to the physical tester for operating the actuator, and to the numerical analyzer (virtual tester). After the actuator achieves the command position, the tester and the analyzer send the feedback data to the controller, and the controller then decides the command for the next step of a testing.

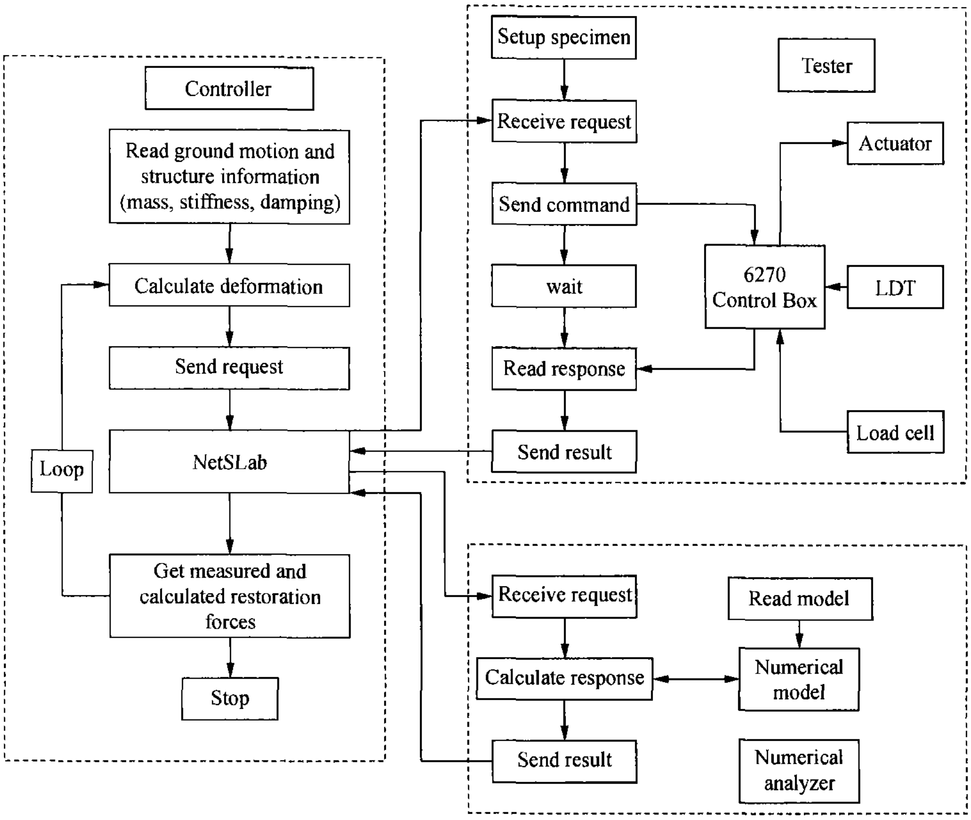


Fig. 4. Flow chart of the main algorism for three-party pseudo dynamic testing.

Figure 5 shows an example chosen from a series of validation tests conducted by the authors. The test used a part of the ground acceleration from the 1979 Imperial Valley earthquake record, which is 39 seconds long with an increment of 0.01 second. The test was conducted for the first 1000 steps of the ground acceleration record. During the testing, the computer serving as controller was placed at the Hunan University, China, whereas both virtual and physical testers were located at the University of Southern California. The time required for each step was composed by three parts: communication time between remote controller PC and the local tester PCs, actuator response time, analyzer time and the waiting time set for the equipment control card, which for this case was 2.5 s, due to the limited capability in the hydraulic pump used for the actuator. Standard MS DOS ping test showed an average communication ping speed of 0.581 s—0.671 s during the process of this testing. It was confirmed that the time required for each step was mainly the actuator response time and the set waiting time. The communication time between the controller and the tester using NetSLab for each step was about 0.7 s, which was close to the standard ping test for the 0.5 Mbps to 1 Mbps network environment. The tests successfully validated the adequacy of the remote testing platform NetSLab. Further applications of NetSLab are currently underway at the Hunan University, the University of Southern California and elsewhere.

4 Conclusion

A network platform (NetSLab) has been developed for remote response simulation of structures and structural elements under earthquake loading, using geographically distributed structural laboratories or computer facilities. The NetSLab (Networked Structural Laboratories) was developed with a proposed data

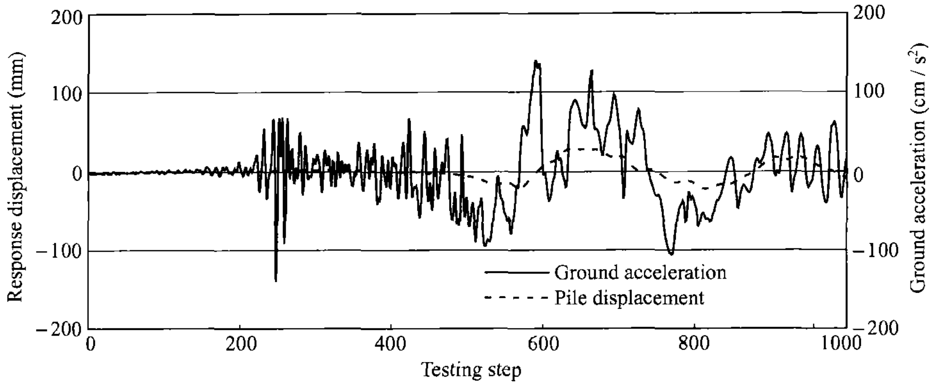


Fig. 5. Results from remote pseudo-dynamic testing over NetSLab of a bridge pier and pile foundation model.

model and the communication protocols. The concepts of Dynamic Unified Data Packet (DUDP), and Generalized Data Communication Agency (GDCA) were introduced to implement the platform along with an interface engine. Thus, the platform was made particularly easier to use by structural researchers using commonly available software. The adequacy of the NetSLab platform has been validated by several virtual and actual tests at and between two universities in China and the USA.

Developing application packages of NetSLab for various structural testing purposes is still underway. One of the major challenges is to provide smooth interface software for a variety of different equipments at different testing facilities. Another challenge is to deal with the complexity of network environment in China, where most universities use the China Education and Research Network. Promising progress has been made at the collaborating universities under the NSFC National Key Project. In addition, the authors have initiated a research effort to expand the functions of NetSLab for remote education of experimental subjects.

Acknowledgements The authors would like to express their appreciations to the Cheung Kong Scholarship of the Ministry of Education of China. FutureNet Technologies Corporation, Monrovia, California, USA and its China subsidiary division provided generous supports under an agreement for joint development, academic promotion and applications. The pile foundation testing specimen described in this paper was from a project sponsored by the MCEER-FHWA Bridge Research Program.

References

1 Mahin S. A. and Shing P. S. B. Pseudo-dynamic method for seismic testing. Journal of Structural Engineering, 1985, 111(7): 1482—1503.

2 Takanashi K. Japanese activities on on-line testing. Journal of Engineering Mechanics, 1987, 113(7): 1014—1032.

3 Mahin S. A., Shing P. S. B. et al. Pseudo-dynamic test method-current status and future directions. Journal of Structural Engineering, 1989, 115(8): 2113—2128.

4 Nakashima M. and Masaoka N. Real-time on-line test for MDOF systems. Earthquake Engineering and Structural Dynamics, 1999, 28(4): 393—420.

5 Nagata K., Watanabe E., Sugiura K. et al. Development of remote parallel pseudo-dynamic testing system by using INTERNET. In: Proceedings of Fifth Pacific Structural Steel Conference. Seoul, Korea, October 13—16, 1998, 589—594.

6 Pauschke J., Anderson T. L., Goldstein S. N. et al. Construction status of the George E. Brown, Jr. network for earthquake engineering simulation. In: Proceedings of the Seventh U. S. National Conference on Earthquake Engineering (7NCEE). Boston, MA, July 21—25, 2002.

7 Yang Y. S., Wang S. J., Wang K. J. et al. ISEE: internet-based simulations for earthquake engineering, part I: the database approach. In: Proceedings of the International Workshop on Steel and Concrete Composite Construction (IWSCCC-2003). Taipei, China, October 8—9, 2003, 301—310.

8 Wang K. J., Wang S. J., Yang Y. S. et al. ISEE: internet-based simulations for earthquake engineering, part II: the application protocol approach. In: Proceedings of the International Workshop on Steel and Concrete Composite Construction (IWSCCC-2003). Taipei, China, October 8—9, 2003, 311—320.

9 Guo Y. R. and Xiao Y. Remote collaborative analysis of structures (in Chinese). In: Proceedings of the Twelfth National Conference on Structural Engineering. Chongqing, China, October 19—22, 2003, 522—525.

10 Xiao Y. and Yun H. W. Full-scale experimental studies on high-strength concrete columns. ACI Structural Journal, 2002, 99(2): 199—207.

11 Xiao Y., Priestley M. J. N. and Seible F. "Seismic Performance of Bridge Footings Designed to Current Standards," American Concrete Institute, ACI Special Publications on Seismic Response of Concrete Bridges, SP-187, Farmington Hills, Michigan, 1999, 131—157.

12 Xiao Y. Experimental studies on precast prestressed concrete pile to cip concrete pile-cap connections. PCI Journal, 2003, 48(6): 82—91.