Abstract: An event-driven distributed control system for conducting continuous seismic response simulations using geographically distributed hybrid models is presented. Hybrid models, comprising appropriately scaled experimental and numerical substructures in networked geographically distributed laboratories or computers, can be used to realistically and cost-effectively evaluate the performance of complex structures at large scales. The principal advantage of the proposed event-driven control system is its ability to mitigate the adverse effects of random completion times of communication, computation, actuation and measurement tasks during a hybrid simulation on the stability, accuracy and reliability of the simulation results. The efficiency of the proposed continuous extrapolation/interpolation hybrid simulation method is demonstrated by examining the earthquake response of a two-story shear building model that comprises two experimental substructures and a numerical integrator connected through the internet. An evaluation of results from these hybrid simulations suggests that distributed hybrid simulation conducted using the proposed procedure provides accurate and reliable results.

DOI: 10.1061/(ASCE)0733-9445(2006)132:1(68)

CE Database subject headings: Dynamic tests; Experimentation; Simulation; Control systems; Seismic effects; Hybrid methods; Structural response.

Introduction

Hybrid simulation is an experiment-based method for investigating the response of structures. This response is obtained by solving the equations of motion of a hybrid experimental and numerical model of the structure. The hybrid model is designed using similitude laws and appropriate partitioning of the equations of motion. Thus, the hybrid model can be conceptualized as an assemblage of one or more numerical and one or more experimental consistently scaled substructures. The experimental substructures are scaled models of the prototype substructures physically instrumented and actuated in a laboratory. The numerical substructures are scaled numerical models of prototype components instantiated in software running on one or more computers. For hybrid simulation of seismic response of structures, the hybrid model is designed to represent various behavioral aspects of the prototype. Typically, though not always, components in the equations of motion associated with inertia, viscous damping and imposed dynamic excitations are modeled numerically, whereas components associated with the restoring forces developed by the structure are evaluated either experimentally or numerically. The equations of motion are solved numerically for specified dynamic excitations using well known step-by-step integration operators. In this way, the dynamic response of large structures, especially ones where complex, nonlinear behavior is expected to occur only in a few locations, can be simulated realistically and cost effectively at large or full scale.

Hybrid simulation stems from the pseudo-dynamic testing method developed over the past 30 years (Takanashi et al. 1975; Takanashi and Nakashima 1987; Mahin et al. 1989; Shing et al. 1996; Royal Society 2001). Early pseudodynamic tests were based on explicit numerical integration procedures, involved only one experimental substructure, and utilized a ramp-and-hold loading procedure to obtain the experimental restoring force component in the equations of motion. These early results showed that the pseudodynamic test method is comparable to the shaking table test method when propagation of experimental errors is successfully mitigated (Takanashi and Nakashima 1987; Mahin et al. 1989; Magonette and Negro 1998). Improvements in computer and testing hardware enabled further advances in the pseudodynamic test method, such as: substructuring (Demitzakis and Mahin 1985), use of implicit integrators (Thewalt and Mahin 1987; Shing et al. 1991), multidirectional testing (Thewalt and Mahin 1987), and error analysis (Shing and Mahin 1987; Thewalt and Roman 1994). Perhaps the most important improvement of the method was the development of the continuous (Magonette 2001) and real-time (Nakashima et al. 1992; Nakashima 2001) testing techniques. These techniques successfully mitigate a dominant source of errors in pseudodynamic tests: The strain-rate sensitivity of structural material response that manifests itself as relaxation of the restoring force during displacement hold portions of the test.

Utility of the pseudodynamic test method can be further extended by geographically distributing experimental and numerical substructures onto a network of laboratories, then linking them
into the test using a long-range communication network (Campbell and Stojadinovic 1998). The infrastructure of the George E. Brown Jr. Network for Earthquake Engineering Simulation (NEES Consortium, Inc., (http://www.nees.org) 2004 provides the experimental equipment, the analytical modeling tools and the network interface to conduct such simultaneous testing of multiple large-scale experimental substructures and complex numerical models using resources distributed on the internet. Geographically distributed pseudodynamic tests have been carried out between Japan and Korea (Watanabe et al. 2001), in Taiwan (Tsai et al. 2003) and in the United States as part of the NEES efforts (MOST 2003). However, these tests used the conventional ramp-and-hold procedure to load the experimental substructures. As such, they are not benefiting from the improvements in test accuracy and reliability afforded by the continuous testing techniques. In addition, these experiments usually lasted a long time, some taking over 5 h to complete.

Real-time continuous algorithms are difficult to implement on a network of distributed resources because some tasks, such as integration of the numerical substructures or communication between hybrid model substructures, have random completion times. Stability of real-time algorithms may be compromised because tasks, whose execution is triggered by a clock rather than by a task-completion event, may start before the preceding tasks have been completed. The ramp-and-hold actuator motion technique is suitable for handling random delays since the hold period can be as long as needed to insure completion of preceding tasks. However, long hold periods introduce unrecoverable relaxation errors.

In order to enable continuous testing on a geographically distributed hybrid model, an event-driven hybrid simulation control strategy is proposed herein. The event-driven controller mitigates the randomness and uncertainty introduced by geographic distribution of the hybrid model components and strives to minimize, if not eliminate, the hold phase at each integration step. This approach reduces simulation errors caused by relaxation of experimental substructures. The proposed control strategy also reduces the duration of the test by reducing the time it takes to accurately actuate and measure the response of experimental substructures. The proposed distributed hardware architecture utilizing event-driven controllers is validated experimentally using hybrid simulations on a hybrid model of a frame building with two remote experimental substructures and a numerical integrator. The hybrid simulation results are evaluated using two procedures. First, the function of the controller is verified by comparing the experimental results to purely numerical simulations. Second, incurred experimental errors and the effect of these errors on the response of the hybrid model are examined. Both of these verification procedures suggest that continuous hybrid simulation on a geographically distributed hybrid model provides reliable and accurate results. These experiments are, to the best knowledge of the writers, the first-ever continuous hybrid simulation on a distributed hybrid model connected using the general purpose internet network. A doctoral dissertation written by the second author (Mosqueda 2003) details the research leading to the findings presented herein.

Hybrid Simulation Test Method

The components of a conventional hybrid simulation test setup and equipment interconnections are illustrated in block diagram form in Fig. 1. The required equipment and tools are: (1) a servohydraulic system consisting of a controller, servovalve, actuator and pressurized hydraulic oil supply; (2) a test specimen with the actuator(s) attached at the location where the displacement degrees of freedom are to be controlled; (3) instrumentation to measure the response of the test specimen; and (4) a computer connected to the servohydraulic system capable of computing a command signal using feedback from the transducers as well as computing successive solutions of the equations of motion of the hybrid model using a time-stepping numerical integration method. With the exception of the on-line computer, this equipment is readily available in most laboratories with capabilities for quasi-static tests. Note that the on-line computer does not need to be connected to a long-range network in order to run conventional hybrid simulations.

The primary task of the on-line computer is to integrate the equation of motion utilizing the restoring force vector, \( r \), which is assembled using forces measured at the degrees of freedom of experimental substructures or computed at the degrees of freedom of numerical substructures. A time-stepping integration procedure is used to solve the discretized equation of motion for displacement, \( d \), velocity, \( v \), and acceleration, \( a \), at the degrees of freedom of the structure at time intervals \( t_i = i \Delta t \) for \( i = 1 \) to \( N \)

\[
M \dot{a}_i + C \dot{v}_i + r_i = f_i
\]  

(1)

The subscript \( i \) denotes the time-dependent variables at time \( t_i \); \( \Delta t \) integration time step in the numerical simulation; and \( N \) = number of integration steps. The mass matrix, \( M \), damping matrix, \( C \), and applied loading, \( f \), are typically modeled numerically in the computer. Numerical methods used to solve the equation of motion in a hybrid simulation are discussed by Mahin and Shing (1985).

Continuous Testing

The computed actuator commands can be applied to the specimens in a ramp-and-hold or a continuous fashion. Applying a continuous load history, rather than a ramp-and-hold load history, improves the measured behavior of the specimen and reduces the actuator tracking error (Magonette 2001). The improvements are largely due to the elimination of relaxation of the specimen during the hold phase and the associated drop in measured force. Continuous algorithms, however, require a real-time platform to ensure commands for the servohydraulic controller are updated at deterministic rates. Such constant update rates allow for control of the actuator velocity, thus enabling application of a continuous load history on the experimental substructures. The difference between the ramp-and-hold and continuous load histories for one
simulation step is shown in Fig. 2. Note that the continuous procedure utilizing a predictor phase (extrapolation) followed by a correction phase (interpolation) reduces the actuator velocity demands for the same time step, or alternatively can shorten the time required per step. A polynomial based extrapolation/interpolation algorithm for continuous hybrid simulation is summarized next.

In their algorithm for real-time continuous testing, Nakashima and Masaoka (1999) separate the computations in the on-line computer into two tasks running at different sampling rates: (1) the response analysis task with time step $\Delta T$, during which the integration of the equation of motion for the structure is carried out; and (2) a signal generation task with time step $\delta T$, during which displacement commands to the servo-hydraulic actuator are computed using polynomial trajectory approximation procedures. $\Delta T$ is the actual time it takes for one simulation step: it should be sufficiently longer than the response analysis task time, as discussed below. $\delta T = \Delta T / j_{\text{max}} = \text{actuator command update rate}$, where $j_{\text{max}} > 1 = \text{number of command updates per simulation step}$. In this paper, $T$ is used to denote the actual time in the simulation and $t$ denotes the simulation time with respect to the integration algorithm, where $t = T$ for real-time testing.

The integration task and the signal generation task run on a digital signal processor (DSP) in real time using interrupts in accordance with the conventional multirate control approach. After achieving the target displacement in an integration step, the actuator is kept in motion using a predicted (provisional) displacement target computed by polynomial extrapolation of the existing trajectory. Meanwhile, the interrupt task is carrying out computations for the actual (considered more accurate) target displacement. Once the next actual target displacement is known, the controller switches to the interpolation phase and interpolates to correct the displacement trajectory toward the actual target value. Nakashima and Masaoka (1999) showed that third order polynomial interpolation and extrapolation to compute actuator command values using data from previously completed time steps provide sufficiently accurate displacement and velocity predictions for the next target. The key advantages of the polynomial approximation procedure are that it is explicit (uses only known data) and that it requires a short time to compute the actuator commands enabling updates at short and constant time intervals. Ideally, actuator commands should be generated at the update rate of the actuator controller, typically 1,000 Hz.

The accuracy of a continuous hybrid test critically depends on the response of the actuators. The inherent time lag in actuator response results in a delay of the measured restoring force signal relative to the command signal. If such asynchronous force measurements are introduced into the numerical integration algorithm effective negative damping will result. Horiuchi et al. (1999) proposed a procedure to compensate for actuator lag by computing commands advanced in time by the estimated lag time using a feed-forward control approach based on polynomial extrapolations. Nakashima and Masaoka (1999) incorporated this lag compensation technique into their extrapolation/interpolation algorithm. For signal generation steps $j = 1$ to $j_{\text{max}}$, the displacement command based on the $n$th order polynomial extrapolation procedure with time delay compensation, $\delta T_{\text{comp}}$, is

$$d_i = \sum L_i^j(j) d_{i-j}$$

where

$$\bar{j} = j / j_{\text{max}} + \delta T_{\text{comp}} / \Delta T$$

and functions $L_i^j = \text{Lagrange polynomials obtained with } \delta T_{\text{comp}} = 0$.

Although the polynomial extrapolation/interpolation and lag compensation methods were originally developed for real-time testing, these techniques can be extended to time in continuous testing at slower-than-real-time rates. This is done by extending the simulation time scale by a factor $\lambda = \Delta T / \Delta t > 0$. For a simulation with $N$ integration steps and prototype duration $N \Delta t$, the expected test duration of hybrid simulation is then $\lambda N \Delta t$.

In a slow test, the limitations of the polynomial based extrapolation/interpolation (prediction/correction) methods are also relaxed. For example, a test stability issue noted by Nakashima and Masaoka (1999) is associated with the sudden change in actuator velocity as the algorithm switches from extrapolation to interpolation. In a slow test, the change in velocity is still present for small $\delta T$. However, two modifications can be made to mitigate this effect. First, velocity limits can be placed on the command signal prior to being sent to the actuators, effectively limiting the allowable displacement command increment per step. These velocity limits should be sufficiently larger than the expected peak velocity from pretest simulations divided by the time scale expansion factor, $\lambda$. Second, actuator gains in the servocontrol loop can be reduced to effectively slow the response of the actuators to a sudden change in command velocity. The reduced control gains will also reduce overshooting to a step response, which can be beneficial in case a hold is necessary. However, reduction in control gains increases the actuator response lag, which can be, to a large extent, compensated for using the feed-forward technique. Horiuchi et al. (1999) also noted a limitation of their lag compensation method: the compensated delay time, $\delta T_{\text{comp}}$, should be less than $1.59 / \omega_0$, where $\omega_0$=highest loading frequency. In a slow test, $\omega_0$ can be approximated as $\omega_0 = \lambda$, where $\omega_0$=highest natural frequency of the structural model under consideration. Therefore, a careful balance between sudden command velocity changes and control system gains must be found to conduct a successful test.

### Distributed Hardware Architecture

The hybrid simulation controller for conventional tests typically consists of two feedback loops: the outer integration loop commanding the inner servohydraulic controller loop as shown in Fig. 1. A single processor can be used to compute both the integration of the equation of motion and the generation of the actuator commands. However, separation of these two tasks onto different processors enables an expandable distributed architecture for simultaneous testing of multiple substructures as shown in Fig. 3. Additional advantages of task separation are: (1) more processing time can be dedicated to the integration task allowing for more...
time-consuming computation needed for complex models; and (2) actuator commands are generated by a dedicated real-time processor, which can then command more than one actuator to test multiple-degree-of-freedom substructures. In a local testing configuration, the information exchange link between the processors hosting the integrator and the experimental substructures can be accomplished using a shared memory resource, a fast but short-range memory link such as SCRAMnet (Systran 2003), to maintain practically immediate information transfer for real-time tests. In the case of geographically distributed testing, a long-range, but slower, network may replace the shared memory network. Ethernet and the associated TCP/IP communication protocol that form the widely available internet network are the natural choice. However, the TCP/IP standards do not guarantee delivery of all messages. Further, transmission time for delivered messages is random. Such communication network is, therefore, not suitable for real-time algorithms. To facilitate continuous hybrid simulation algorithms using the internet, a distributed architecture for hybrid simulation is proposed herein. This architecture introduces an intermediate control loop hosted on the real-time digital signal processor that controls each substructure. The control loop buffers the communication delays using an event-driven control strategy discussed in the next section.

**Event-Driven Controller**

In cases where task execution times are random, clock-based control schemes could fail if the required tasks are not completed within the allotted time. An event-driven controller, based on the concept of finite state machines (Harel 1987), is proposed to respond to events based on the state of the hybrid simulation system rather than a clock cycle. The event-driven controller can be programmed to account for the complexity and randomness of the distributed hybrid model and, thus, take action to mitigate random disturbances and complete the simulation incurring as little error as possible. The programming procedure is based on defining a number of states in which the controller can exist in and the transitions between these states that take place as specified events occur.

The state transition diagram in Fig. 4 shows the implementation of an event-driven version of the polynomial approximation method by Nakashima and Masaoka (1999). This algorithm continuously updates the actuator commands using the conventional approach under normal operation conditions and takes action in case of excessive delays. The state diagram consists of five states: extrapolate, interpolate, slow, hold, and free_vibration. The state changes from extrapolate to interpolate after the controller receives the next target displacement and generates the event $D_{update}$. Event $D_{target}$ is generated once the experimental substructure achieves this target displacement. The controller then transitions back to the extrapolate state and sends the updated measurements of the state of the substructure to the integrator. Smooth execution of this procedure is dependent on having a reliable network connection and selecting the run time of each integration step sufficiently long for all of the required tasks to finish. Small variations in completion times for these tasks will only affect the ratio of time the controller spends in the extrapolate and interpolate states.

The advantage of the event-driven controller is that logic can be included to handle excessive delays. For example, if the system is in the extrapolate state for a time greater than $\Delta T$, the actuator can deviate substantially from the intended trajectory. Therefore, limits need to be placed on the duration of the extrapolate state. A simple solution is to generate the Timeout event to transition to the slow state. In the slow state, extrapolation continues at a pre-determined reduced velocity, thus allowing more time to receive a command update. Upon receiving the next target displacement, the controller switches to interpolate. If the update is not received within a set amount of time, the slow state generates its Timeout event to place the actuator on hold until the target displacement is received: the continuity of actuator motion during the tests is, thus, lost incurring additional error, but the test itself can continue from the hold state once the command update is received. Longer delays, possibly due to the integrator crashing or network failure, could indefinitely delay the controller receiving an updated displacement target. In this rare event, the hold state also times out, automatically or through user intervention, and
force the system into free vibration to dissipate the energy in the physical specimens and end the test. The free vibration state is intended to unload the experimental substructures in highly damped free vibration using the current state as the initial condition. Other shut-down procedures can also be used.

The state transition diagram presented in Fig. 4 serves as an example of the event-driven logic that can be programmed into local substructure controllers to improve the measured behavior of the experimental substructures and salvage the simulation in case of major communication failures. Other failure scenarios can also be handled by the event-driven substructure controller. For example: (1) local failure of the servohydraulic system can be handled by replacing the experimental substructure with a numerical substructure model (located at the experimental site or at the integrator site), calibrated using specimen response data recorded during the test before failure occurred; and (2) loss of the network connection to the remote integrator means that the distributed test may no longer be possible, but the individual experimental substructures can continue moving to complete the experiment and shut down using local models of the remote integrator. Such local models can be generated and calibrated using system identification algorithms on the data received from the remote integrator during the hybrid test before the breakdown of communications occurred.

**Experimental Verification of the Distributed Hybrid Simulation Architecture**

The hybrid simulation control system for geographically distributed testing was experimentally verified by conducting hybrid simulations involving numerical and experimental substructures located at the Davis Hall Structures Laboratory on University of California (UC) Berkeley Campus and the micro-NEES reaction frame at the UC Berkeley Richmond Field Station located 5 miles apart. The numerical integration procedure for the structural model (the integrator) was programmed in MATLAB (Mathworks 2003) and ran on a computer located on campus. The integrator was linked to two independent experimental substructures using Ethernet and TCP/IP toolkits implemented in MATLAB (Andrade 2001; Rydesater 2001). The experimental substructures were controlled by separate computers, each with an on-board DPS. The DSPs were powered by Motorola 250 MHz processors (dSPACE 2003) and had analog to digital and digital to analog converters for data acquisition and command output enabling real-time interaction with the experimental substructures by connecting the dSPACE outputs and inputs to the actuator servocontroller inputs and outputs, respectively. Identical event-driven controllers, discussed in the previous section of this paper, were implemented in both DSPs. The hardware architecture sketch is shown in Fig. 3.

The hybrid model represents a two-degree of freedom shear frame. The frame is assumed to deform in pure shear with rigid beams, as shown in Fig. 5(a). Thus, the columns have a point of inflection at mid-height of the story, allowing for the extraction of two single-degree-of-freedom experimental substructures, one representing the first story and the other representing the second story of the frame [Fig. 5(b)]. Each substructure is a half-story-high column cantilever loaded transversely by an actuator and is represented by a 1/3-scale physical specimen in the laboratory. The resisting shear forces for each story are obtained experimentally by applying the command displacement and measuring the reaction force at the tip of the cantilever. The other column in each story is assumed to behave the same as the tested column. Given the experimental substructure length scale factor of 3, the consistent force scaling factor is 9.

Each test specimen consisted of a 1,270 mm long S4X7.7 cantilever column with an idealized plastic hinge connection (Rodgers and Mahin 2002) at the base. Average initial stiffness of the specimens was approximately 0.490 kN/mm. The vibration periods of the model were 0.62 and 0.24 s for the first and second mode, respectively. These periods were obtained using this measured stiffness of the experimental substructures and the mass assigned numerically to the model, as shown in Fig. 5(a). The model damping matrix was taken as stiffness-proportional with 5% of critical damping in the first mode to quickly decay the transient response of the structure.

The equations of motion for the hybrid model was solved using the explicit integration algorithm of Newmark (1959) modified for hybrid simulation (Mahin and Shing 1985). This algorithm was selected because it is simple to implement and because its stability limits are suitable for the structural model under consideration. The appropriate geometric transformations were included in the numerical procedure to convert the global degrees of freedom shown in Fig. 5(a) to the actuator degrees of freedom. The combined experimental and analytical structural model was subjected to the 1978 Tabas historical earthquake acceleration record whose duration was consistently scaled to the experimental substructure scale using a factor of $\sqrt{3}$. The amplitude of the ground motion was scaled to a peak ground acceleration (PGA) of 0.378g for an elastic level simulation (Tabas-50%) and to a PGA of 1.133g to obtain non-linear response from the experimental substructures (Tabas-150%). Both earthquake simulations were executed for 30 s using an integration time step of $\Delta t=0.01$ s for a total of $N=3,000$ steps.

The extended time scale for the slow continuous test was se-
lected to accommodate about 95% of the simulation steps without having to slow down or hold the actuators. The time scale factor \( \lambda \) was determined as follows. First, the TimeOut event triggers were set to allow the extrapolate state to last up to 60% of the integration time step and to allow the slow state to last up to 80% of the integration time step duration. This procedure guaranteed that at least the last 20% of the time step was executed in the interpolate state, meaning that at least 20% of the time-step commands were generated using the actual target displacement and the polynomial interpolation procedure: this decision was made to reduce trajectory tracking errors. The free vibration state was not implemented for these verification tests. Instead, the tests were terminated by a simple disconnect and the experimental substructures were unloaded manually.

Using preliminary test data to characterize the behavior of the network, it was estimated that about 95% of the steps could complete the integration task and the network communication task within 0.7 s. Based on these results and the goal to achieve 95% of the steps without delays, each integration step was scaled from \( T_i = 0.01 \) to \( T_i = 1.2 \) s, using the simulation time expansion factor \( \lambda = 120 \). Note that an actual time step could take longer if network communication delays occurred. The actuator command update rate was set to \( \delta T = 0.001 \) s for a total of \( j_{\max} = 1200 \) substeps in each integration time step. The timing limitations programmed into the event-driven controller are specified in Fig. 6 in terms of the expected duration for each step, \( T_i \), and \( T_f \), which denotes the sum of integration task time and network communication task time. The maximum duration of the extrapolate state was set to 0.72 s. In this state, extending the time scale reduced the actuator velocity by a factor of 120 compared to a real-time simulation. In the slow state, the actuator velocity was further halved, making the total duration of the slow state equal to 0.48 s. In the switch to the slow state, the sudden change in velocity is not expected to have adverse effects on the actuator control for the slow rates of testing considered here. However, changes in velocity commands may become significant as the test rate approaches real-time. Such changes must be mitigated to insure a successful real-time test. If the delay in receiving the target displacement was greater than 1.2 s, the controller was programmed to switch to the hold state. The interpolate state, activated after receiving the actual target data, was set to execute at the same velocity as the extrapolate state. All displacement commands were modified to compensate for the measured response lag in the actuators using the feed-forward procedure recommended by Horiuchi et al. (1999). Note that events at each distributed hybrid simulation site are synchronized only to the scaled integration step duration, \( \Delta T \) (Fig. 6).

**Hybrid Simulation Results**

The results from distributed hybrid simulation tests were evaluated by comparing them to results of purely numerical simulations. To facilitate this comparison, the numerical models and algorithms used in hybrid and purely numerical simulations were the same. In the purely numerical simulations, the experimental elements were replaced by linear-elastic or Bouc–Wen (Bouc 1971; Wen 1976) spring elements. The numerical models were calibrated to the force and displacement data measured during the corresponding experiments to eliminate variability in specimen behavior. Note that the numerical integration algorithm in hybrid simulations uses the measured force versus command displacement to account for the behavior of the experimental substructures. Therefore, the principal difference between the two simulations is that the hybrid simulations include measurement and actuator tracking errors.

The resulting displacements response of the two-story shear frame structure during the elastic-level simulation Tabas-50% is shown in Fig. 7. In this hybrid simulation, the response of experimental substructures was linear elastic, with stiffness values of 0.490 and 0.494 kN/mm for the first and the second story, respectively. Numerical linear-elastic spring models with these stiffness values were used to compute the substructure resisting forces in the purely numerical simulation of the frame model response to Tabas-50% ground motion. The virtually identical responses indicate that the distributed hybrid simulation is accurate within the limitations of the numerical models and that experimental errors, including actuator tracking errors and friction in the test setup,
were negligible. Further, these tests verified that the distributed controller functioned effectively in the presence of random network delays.

The results from the distributed hybrid simulation of the model structure subjected to Tabas-150% ground motion are shown in Fig. 8. In this hybrid simulation, the response of the second story substructure was linear, but the first story substructure behavior was nonlinear. Accordingly, a numerical linear-elastic spring model was used to replace the second story substructure and a nonlinear Bouc–Wen model was used to replace the first story substructure in the purely numerical simulation. The nonlinear restoring force computed by the Bouc–Wen model is given by

\[ r(t) = \alpha kd(t) + (1 - \alpha)kd_jz(t) \]  

\[ d_jz(t) + \gamma v(t)z(t)z(t)^{\nu+1} + \beta v^4(t) - v(t) = 0 \]  

The definition of the parameters in the Bouc–Wen model and calibrated values used in these simulations are listed in Table 1.

The experimental results in Fig. 8 correlate well with the numerical simulation, particularly at the second story level. The maximum drift error between the two simulations occurred at the first story level and was equal to 10% of the maximum drift value. The Bouc–Wen model [Fig. 8(d)] captures the dominant characteristics of the first-story experimental substructure [Fig. 8(c)]. The negative peak force values were similar in both simulations, but the Bouc–Wen model predicted a larger positive peak force. The positive peak force was smaller in the experiment because the strength of the specimen degraded after yielding, while the strength of the numerical model did not degrade. Nonetheless, the numerical simulation and the hybrid simulation are quite similar.

### Experimental Errors

Experimental errors in the distributed hybrid simulations are examined to further ensure that the test results were accurate within the limitations of the numerical models. Propagation of experimental errors during a hybrid simulation results in the cumulative change of energy in the hybrid model. For each integration time step, \( i \), the energy errors can be approximated as (Thewalt and Roman 1994)

\[ E_{\text{error}} = \frac{1}{2}(K_i^{\text{T}}d_{\text{error}}^i + K_{i-1}^{\text{T}}d_{\text{error}}^{i-1})(d_i - d_{i-1}) \]  

where \( d_{\text{error}} = \text{target displacement minus the measured displacement} \) and \( K_i^{\text{T}} = \text{tangent stiffness matrix at step} \ i \). A formula to compute the tangent stiffness matrix for multiple degree-of-freedom experimental substructures was also presented by Thewalt and Roman (1994). The total energy error accumulated during the test at time \( n \Delta t \) is then

\[ E_{\text{error}}(n\Delta t) = \sum_{i=1}^{n} E_{i,\text{error}} \]  

The computed energy error history, \( E_{\text{error}} \), using the experimental data from the nonlinear earthquake simulation Tabas-150% is shown in Fig. 9(a). The cumulative energy error for both experimental substructures grew monotonically in the negative direction, indicating that energy was being added to the numerical model. However, the energy input due to experimental errors is

<table>
<thead>
<tr>
<th>Table 1. Calibrated Parameters for the Bouc–Wen Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td>( K )</td>
</tr>
<tr>
<td>( d_j )</td>
</tr>
<tr>
<td>( \alpha )</td>
</tr>
<tr>
<td>( \gamma )</td>
</tr>
<tr>
<td>( \beta )</td>
</tr>
<tr>
<td>( \eta )</td>
</tr>
</tbody>
</table>
only 0.57% of the total energy input from the applied loading. The energy components and the total input energy in Fig. 9 were computed using the relative energy formulation described by Uang and Bertero. Therefore, the experimental errors in distributed hybrid simulations were small and had a negligible effect on the hybrid simulation results.

**Performance of Event-Driven Controller**

During the distributed tests, there were several instances when the event-driven controller did not receive data from the integrator within the 0.72 s extrapolate state duration limit and activated the slow and hold states. Fig. 10 shows the distribution of the time taken by a local DPS to receive updated data from the integrator after reaching a target displacement. The time captured is the integration task time, $T_i$, including the network communication time, measured during the simulation Tabas-50%. It should be noted that the computation time during these tests was estimated to be under 0.01 s, while the remainder of the time was consumed by network communication. The dashed lines in Fig. 10 represent the limits where the slow and hold states were activated according to the selected timing scheme (Fig. 6). Table 2 provides a summary of the network delay statistics including: the maximum delay, the total run-time for the test and the percentage of the steps when slow and hold states were activated. It is interesting to note that the test with the most delays, Tabas-50%, overran the total target simulation time of 3,600 s by only 67 s. More importantly, the experimental specimens were in the hold state for less than 2% of the simulation steps. The simulation Tabas-150% had half as many delayed steps compared to Tabas-50%, regardless of the fact that the experimental substructure response was nonlinear. The difference in delays during these two simulations is due to variations in internet network congestion occurring during the tests.

**Effects of Delays on Experimental Substructures**

Fig. 11 provides a close look at the behavior of the yielded first-story substructure during the Tabas-150% simulation for steps when network delays occurred. The data in the figure concentrate on only 20 s of the more than 3,600-s-long simulation comprising of 10 simulation time steps. The time scale shown in Fig. 11 corresponds to the DSP clock time, $T$, and not the numerical integration time, $t$. The state of the event-driven algorithm is shown in Fig. 10(a). The Y-axis labels are the (E)xtrapolate, (I)nterpolate, (S)low, and (H)old states. As indicated in Fig. 11(a), the first few steps executed smoothly by switching directly from extrapolate to interpolate. At approximately $T=1,785$ s, two short delays occurred followed by two longer delays. The length of the delay is identified by the amount of time spent in the hold state.

The measured displacement history and force history are shown for the same 20 s of DSP clock time in Figs. 11(b) and c, respectively. The measured force history and the force-displacement data in Fig. 11(d) illustrate the consequences of the hold state on the behavior of the yielded experimental substructure. During the hold, the displacement remains constant as expected, but the force decreases in magnitude because of material

**Table 2. Summary of Delays in Distributed Hybrid Simulation Tests**

<table>
<thead>
<tr>
<th>Test</th>
<th>Number of steps</th>
<th>Maximum delay (s)</th>
<th>Run time (s)</th>
<th>% delayed steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tabas-50%</td>
<td>3,000</td>
<td>6.59</td>
<td>3,667</td>
<td>12.4</td>
</tr>
<tr>
<td>Tabas-150%</td>
<td>3,000</td>
<td>5.9</td>
<td>3,638</td>
<td>5.9</td>
</tr>
</tbody>
</table>

**Fig. 9.** Energy error and total energy history in the Tabas-150% simulation

**Fig. 10.** Histogram of target displacement update time, $\Delta T_i$, during the Tabas-50% simulation

**Fig. 11.** Behavior of yielded experimental substructure during hold states for the Tabas-150% simulation
relaxation. The corresponding segment of the hysteresis provides further evidence of force relaxation, particularly during the two long delays. The circle markers in Figs. 11(a–d) indicate the instance force measurements were taken at the target displacement. This particular step had a 0.2 s hold period, resulting in negligible force relaxation and sufficiently accurate measurements. The square marker indicates the instance force measurements were taken for a step with a much longer (4.7 s) hold period. The force-displacement data in Fig. 11(d) indicates the measured force at the target displacement was taken while the specimen recovered from force relaxation. Consequently, this measured force value used in the integration algorithm included an error. In contrast, steps in which the hold state was not activated result in a smooth force-displacement response, including the steps in which the rate of loading was reduced by switching to the slow state due to short communication delays. Thus, continuous motion of actuators is crucial to obtaining more realistic force measurements from the experimental substructures and achieving accurate simulation results.

Conclusions

An event-driven control strategy for continuous hybrid simulation using geographically distributed hybrid models instantiated using resources located on the internet was presented in this paper. This control strategy provides a fault-tolerant mechanism to handle random delays and to minimize force relaxation and rate-related errors in the experimental substructures. The proposed control strategy was experimentally verified using a hybrid model of a two-story shear frame structure comprising two experimental substructures and a numerical integrator connected using the internet. Evaluation of the test data confirms that the proposed procedure for continuous hybrid simulation using distributed hybrid models provides reliable and accurate results. More details are available in Mosqueda (2003).

The hybrid simulations described in this paper are, to the best knowledge of the writers, the first-ever continuous tests on hybrid models with geographically distributed experimental and numerical substructures. Completion of the George E. Brown Jr. Network for Earthquake Engineering Simulation (NEES Consortium, Inc., http://www.nees.org) 2004 provides the facilities needed to implement and use the proposed hybrid simulation method in earthquake engineering research practice. NEES facilities also provide the incentive to further develop the method. These efforts should focus on: (1) additional verification using shaking table test data; (2) establishment of theoretical and practical bounds on the error of the method; (3) development of better and more general extrapolation and interpolation procedures for generating actuator commands; (4) use of more reliable communication network protocols; and (5) achieving reliable and accurate real-time simulation rates for arbitrarily complex structures.

Acknowledgments

Support of this work through an NSF Grant No. CMS-0086621 made within the George E. Brown Jr. Network for Earthquake Engineering Simulation (NEES) Program of the National Science Foundation is gratefully acknowledged. The writers also acknowledge the assistance of University of California, Berkeley graduate student Mr. Andreas Schellenberg and EERC laboratory technicians Mr. Don Clyde and Mr. Wes Neighbor during experiment setup and hybrid testing. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the writers and do not necessarily reflect those of the National Science Foundation.

Notation

The following symbols are used in this paper:

- $a$ = acceleration;
- $C$ = damping matrix;
- $d$ = displacement;
- $d_{err}$ = measured actuator displacement error;
- $d_y$ = yield displacement;
- $E_{err}$ = energy error;
- $f$ = applied force;
- $j$ = command step counter;
- $j_{max}$ = number of command steps per integration step;
- $k$ = initial elastic stiffness;
- $L^m$ = $m$th order Lagrange polynomial functions;
- $M$ = mass matrix;
- $N$ = number of integration steps;
- $n$ = integration step counter;
- $r$ = restoring force;
- $v$ = velocity;
- $\alpha$ = postyield/elastic stiffness;
- $\beta$ = Bouc–Wen model parameter;
- $\Delta T^*$ = expected time for one simulation step;
- $\Delta T^c$ = integration and communication time;
- $\Delta t$ = integration time step;
- $\delta T$ = command update rate;
- $\delta T_{comp}$ = lag compensation time;
- $\gamma$ = Bouc–Wen model parameter;
- $\lambda$ = time scale expansion factor;
- $\eta$ = Bouc–Wen model parameter;
- $\omega_n$ = loading frequency; and
- $\omega_{n}$ = highest natural frequency of structural model.

References


