Implementation of Real Time Hybrid Simulation Based on GPU Computing

Zhenyun Tang
Beijing University of Technology
Xiaohui Dong (dongxiaohui@emails.bjut.edu.cn)
Beijing University of Technology https://orcid.org/0000-0001-8268-6453
Zhenbao Li
Beijing University of Technology
Xiuli Du
Beijing University of Technology

Research Article

Keywords: graphics processing unit, real time hybrid simulation, numerical integration algorithm, shaking table, finite element analysis

Posted Date: June 10th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-596198/v1

Implementation of Real Time Hybrid Simulation Based on GPU Computing

Zhenyun Tang, Xiaohui Dong*, Zhenbao Li, Xiuli Du
(Key Laboratory of Urban Security and Disaster Engineering of Ministry of Education, Beijing University of Technology, Beijing China 100124)

Abstract

With combination of physical experiment and numerical simulation, real-time hybrid simulation (RTHS) can enlarge the dimensions of testing specimens and improve the testing accuracy. However, due to the limitation of computing capacity, the maximum degrees of freedom for numerical substructure are less than 2000 from the reported RTHS testing. It cannot meet the testing requirements for evaluating the dynamic performance of large and complex engineering structures. Taking advantages of parallel computing toolbox (PCT) in Matlab and high-performance computing of graphics processing unit (GPU). A RTHS framework based on MATLAB and GPU was established in this work. Using this framework, a soil-structure interaction system (SSI) was tested by a shaking table based RTHS. Meanwhile, the dynamic response of this SSI system was simulated by finite element analysis. The comparison of simulation and testing results demonstrated that the proposed testing framework can implement RTHS testing successfully. Using this method, the maximum degrees of freedom for numerical substructure can reach to 27,000, which significantly enhance the testing capacity of RTHS testing for large and complex engineering structures.
KEYWORDS: graphics processing unit; real time hybrid simulation; numerical integration algorithm; shaking table; finite element analysis

1. INTRODUCTION

Real-time hybrid simulation(RTHS) is a testing method combining physical experiments and numerical calculations[1, 2]. It divides the integral structure into physical and numerical substructure. The former one is tested by physical experiment, and the latter one is simulated by numerical analysis. The coordination of interface responses between two substructures are transferred in real-time. RTHS testing makes it possible to evaluate the dynamic performance of large and complex engineering structures on existing facilities [3].

Generally, the dynamic solution needs to be completed in a very small integration time step (at least 20 ms). Efficient explicit integration algorithms, such as the central difference method, are usually used to solve the response of the numerical substructure[4]. However, the stability of convergence is a disadvantage of explicit integration algorithm. In order to ensure the stability and accuracy, a smaller integration step (e.g., Δt=1 ms) [5] is often required. Correspondingly, the small integration step will limit the calculation scale of numerical substructure, which cannot meet the RTHS needs for evaluating the dynamic performance of large and complex engineering structures[6]. Optimizing the numerical integration algorithm[7, 8] and improving the numerical solution efficiency[6] are primary ways to solve the above problems.

To guarantee the stability of numerical solution for RTHS testing, Chang[9], Nakashima[10]studied unconditionally explicit pseudo-dynamic algorithms. These

methods provide only explicit target displacement but not explicit target velocity, which cannot consider the velocity-dependent restoring force in RTHS. Wu[7] proposed a target velocity formulation based on the forward difference of the predicted displacements so as to render the above methods explicit for RTHS. Based on existing integration methods, Nakashima[11] proposed to divide the numerical solution into two parts, response analysis task (RAT) and signal generation task (SGT), realized RTHS testing with 10 degrees of freedom(DOFs) under Δt=330 ms or 12 DOFs under Δt=500 ms. Cheng [12] developed a 'hybrid finite element' program by MATLAB, which combined finite element method(FEM) with hybrid structure testing, realized RTHS testing with 122 DOFs under Δt=10 ms. Chae[13] used Hybrid-FEM technology to accomplish a RTHS testing with 514 DOFs under Δt=10 ms. Saouma[14] developed a program named Mercury running on real-time hardware and completed a nonlinear model RTHS testing with 405 DOFs under Δt=10 ms. Zhu[6] used RAT and SGT in two different target computers, realized RTHS testing with 1240 DOFs under Δt=20 ms. In summary, the research of algorithms and the development of FEM improve the real-time computing capacity of numerical substructure. However, as known from the reported work, the maximum DOFs of the numerical substructure for RTHS is hard to exceed 2000.

In the research field of solution efficiency, traditional numerical substructure calculations are all based on Central Processing Unit (CPU) in computer. At present, the calculation capacity of CPU conforms to Moore's Law[15]. Due to the limitation of calculation capacity, complex numerical models are difficult to be solved in real-time using small time step. Since 2008, NVIDIA Corporation has used Graphics Processing Unit (GPU) for scientific computing successfully. Compared with CPU, there are more computing units in the GPU, it is more advantageous to use GPU in large-scale numerical calculations[16]. Papadopoulos[17], Gravvanis[18] proved that GPU parallel calculation based on the Compute Unified Device Architecture (CUDA) has higher calculation efficiency than multi-CPU processors in solving large sparse matrix. In the field of engineering, researchers were also using GPU to accelerate calculation and simulation[19, 20]. With combination of GPU parallel and discrete element method (DEM), Mohammad[21] studied displacement slip faulting through granular soils by a method of GPU-based DEM modelling. Durand[22] realized the simulation of contact between rock and concrete 30 times based on GPU (NVIDIA Fermi C2050) faster than CPU (Intel Xeon X5650 2.67 GHz). Lu[23] used GPU to simulate seismic damage of a medium-sized urban area, achieved 39 times faster than CPU, which were similarly priced. GPU in the field of civil engineering calculation has a very high computational efficiency [24-26]. There are a number of software support GPU acceleration calculation, including finite element software such as ABAQUS, OpenSEES and ANSYS, and mathematical calculation software such as MATLAB[27-30]. Hence, this work attempts to develop a RTHS framework based on MATLAB and GPU, the accuracy and efficiency of GPU (NVIDIA Tesla V100) compared with CPU (Intel Xeon E5-2690 V4) were verified by simulation and RTHS testing.

2. GPU-BASED RTHS TESTING SYSTEM

This work established a testing system framework, using GPU instead of traditional CPU as the computing hardware. There are two problems that need to be solved before establishing a testing system with GPU computing. One is how to realize the

real-time signal interaction between two substructures; the other is how to carry out dynamic analysis for large-scale numerical substructure by GPU.

2.1. Schemes of framework

The framework of the testing framework is composed of three parts, as shown in Figure 1, including numerical substructure solution, signal transmission and physical substructure loading. In RTHS, the efficiency of numerical solution must meet the requirements of real-time loading of the physical substructure, and the time of each dynamic solution step is fixed. The loop structure of time control in LABVIEW[31] and the Parallel Computing Toolbox (PCT) in MATLAB are used in numerical substructure solution. The loop structure of time control ensures the time of numerical substructure solution fixed in each step. The period of the loop structure is adjusted according to the step size of the integration algorithm. The M-file script (a type of MATLAB script file) for solving numerical substructure is imported to the MATLAB script window[32], which is in the loop structure of time control. Parallel Computing Toolbox (PCT) can be used in MATLAB R2014a or updated version[33]. It supports multi-core CPU and GPU parallel computing. PCT can significantly improve the efficiency of large matrix elementary arithmetic and signal processing.

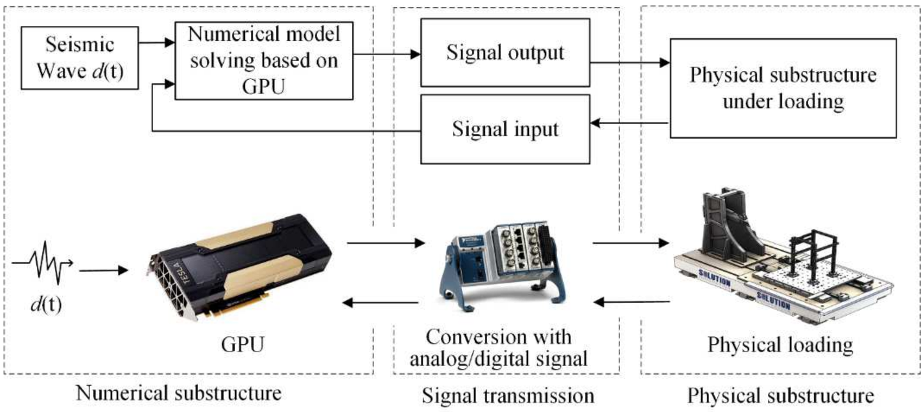


Figure 1 GPU-based framework for RTHS.

The function of signal transmission part is to ensure data communication between two substructures in RTHS. The digital signal from the numerical solution computer is converted into analog signal, and then transmitted to shaking table. Signal acquisition cards are needed to achieve the conversion of digital signal and analog signal.

In physical substructure part, the interface response of numerical substructure transmits from the signal transmission part into the controller of shaking table. The shaking table applies the interface response to the physical substructure after receiving the command, the sensors measure the response of physical substructure and return to the numerical substructure through the signal transmission part.

2.2. Solution of numerical substructure

As mentioned in introduction, there are many ways for dynamic analysis using GPU. Commercial finite element software has pre-processing functions, it can make preparations for dynamic analysis in MATLAB. Parameter matrices of numerical model are extracted from the pre-processing functions of commercial finite element software.

The numerical substructure is modelled by pre-processing of ABAQUS in this work. The steps for extracting parameters of numerical substructure from ABAQUS are as follows.

Step 1, establish the finite element model.

Step 2, output the stiffness and mass parameters: the model parameters saved in the text file with format of *'inp'* will be generated in the file directory. Add the following code to the *'inp'* file and save it. After resubmitting the job file, total mass and stiffness parameters are obtained, the parameters are not yet in matrix form.

```
*step,name=matrix
*matrix generate,stiffness,mass
*matrix output,stiffness,mass
*end step
```

Step 3, convert stiffness and mass parameters to matrices: use MATLAB to import and process the parameters file, the mass and stiffness matrices of numerical substructure are obtained.

Only the mass and stiffness parameters can be extracted from ABAQUS. The damping matrix is constructed from the stiffness and mass matrix based on Rayleigh damping as shown in Equation (1), $[C]$ is the Rayleigh damping matrix, $[M]$ is the mass matrix, and $[K]$ is the stiffness matrix. a_0 and a_1 are two proportional parameters, which are determined according to the first two modal frequencies.

$$[C] = a_0[M] + a_1[K] \tag{1}$$

Figure 2 shows the flowchart of numerical substructure solving based on GPU and MATLAB. The real-time solution of numerical substructure in GPU-based RTHS testing requires CPU and GPU, in which CPU is for transmission of the interface response and GPU is for solution.

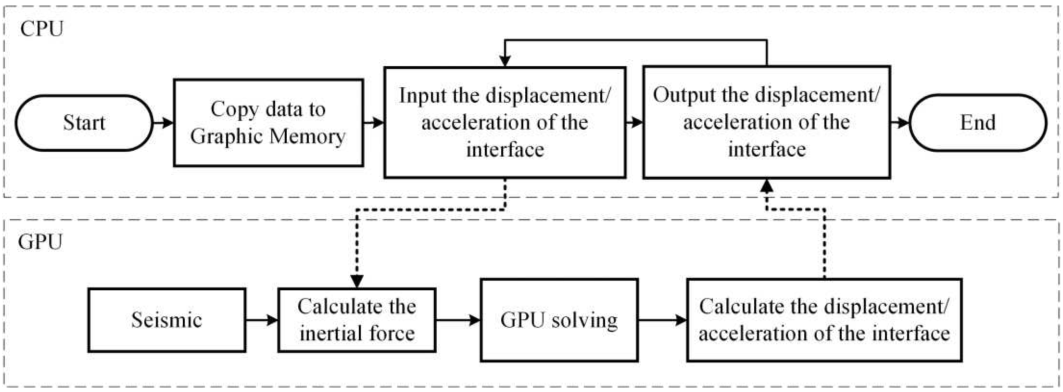


Figure 2 Solution of numerical substructure in GPU-based RTHS testing.

3. NUMERICAL VERIFICATION

In order to verify the solving capability of GPU, a soil-structure interaction (SSI) system was adopted to be numerical modal, which is solved by GPU and CPU respectively.

3.1 Parameters of simulation

As shown in Figure 3, the upper part of the SSI system is physical substructure, and

the soil is numerical substructure. The size of the numerical model is 30m×30m×15m, the density is 1×10^4 kg/m^3, the Young's modulus is 211 MPa, and the damping ratio is 0.5. Viscoelastic boundary is set to simulate far-field soil boundary conditions. The normal stiffness and damping are 20 kN/m and 1.437×10^3 kN/(m/s), The tangential stiffness and damping are 10 kN/m and 9.45×10^3 kN/(m/s). The Kobe seismic wave is adopted as excitation, the peak ground acceleration (PGA) is adjusted to 0.5g, and central difference method is used to numerical dynamic analysis method.

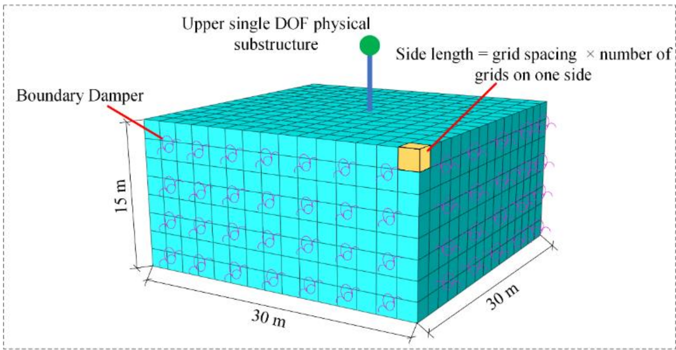


Figure 3 Soil-structure interaction system.

In the simulation, the computing capacities of GPU and CPU based on MATLAB were compared, a personal computer (PC) equipped with consumer-grade GPU and a server equipped with professional-grade GPU. Table 1 shows the hardware configuration parameters of the PC and the server.

System configuration	PC	GPU Server
CPU	Intel i5 8300H	Intel Xeon E5-2690 V4
CPU Performance	4 cores 2.3 GHz	14 cores 2.6 GHz
GPU	NVIDIA GTX 1050	NVIDIA Tesla V100
GPU Performance	3.88 GFLOPS	10.60 TFLOPS
RAM	8 GB	32 GB
Operation System	Windows 10 64-bit	
Software Platform	MATLAB 2020、CUDA 10.2	

Table 1 The configuration parameters of simulation system

3.2 Results of simulation

In order to compare the calculation efficiency of CPU and GPU, it is necessary to record their time cost in the process. There is a predictive function to record the cost of programs in MATLAB. In the program, *'tic'* is added at the beginning of dynamic analysis program, '*toc*' is added at the end, the time cost in each step of dynamic analysis is counted. In order to evaluate the calculation capability of GPU in RTHS testing, the different modal DOFs were set by changing mesh sizes. Three numerical substructures with DOFs of 3888, 6591 and 27000 are solved through CPU and GPU respectively, and the floating-point precision of the model parameters is single.

Table 2 shows the time cost of three numerical substructures and speedup ratio (SR) between CPU and GPU. SR is calculated from Equation (2). T_{CPU} is the time cost of numerical solution based on CPU in each step, and T_{GPU} is the time cost of that based on GPU.

$$SR = \frac{T_{CPU}}{T_{GPU}} \tag{2}$$

When DOFs of numerical model is 3888, the time cost taken by CPU on PC and the CPU on server are similar, it shows the calculation capacity of the CPU on server cannot be brought into full play when the computation is small. When using GPU to solve the model, the speed of the GPU on server is much faster than the GPU on PC. The speed of GPU is 2 times faster than CPU on PC, and the speed of GPU on server is 8.5 times faster than CPU on server. When the DOFs is 6591, the CPU on PC needs 68ms in each step, while GPU only needs 21 ms, the speedup effect of GPU on PC is 3.4 times faster than CPU on PC, and the speedup effect of the GPU on server is 12 times faster than CPU on server. When the DOFs is more than 6951, it will exceed the calculation capability of MATLAB on PC, but the DOFs can further increase on the server.

DOFs	PC			GPU Server		
	T_{CPU}	T_{GPU}	SR	T_{CPU}	T_{GPU}	SR
3888	20	10	2	17	2	8.5
6591	68	21	3.4	48	4	12
27000	Out of GPU memory			936	17	55

Table 2 Comparisons of solving times (Unit: ms)

With increase of DOFs, the numerical solution for each time step cost more time. In this work, when the time step is 17 ms, the maximum DOFs solved in real time is 27,000. It needs 936 ms for the time step of 20 ms to solve this model through the CPU on server. The GPU on server can achieve 55 times faster than the CPU on server. When the DOFs is more than 27,000, it is out of the memory of MATLAB on the server.

Figure 4 shows the displacement time history of the interface between two substructures. when the DOFs of numerical model is 3888, and time steps are 1 ms, 5 ms and 20 ms respectively. Figure 4 (a) shows the displacement time history solved by CPU and GPU with Δt=1 ms. In Figure 4(b-d), X-axis is the displacement time history solved by GPU, and Y-axis is the displacement time history solved by CPU. It can be seen from the Figure 4(b-d) that all of them are straight lines, which show that solution based on GPU has the same accuracy as CPU. GPU can be used instead of CPU to dynamic analysis, and its calculation capability is higher than CPU's when the DOFs is large.

4. EXPERIMENTAL VERIFICATION

The solving capability of GPU was verified by simulation in Section 3, the performance of GPU-based testing system will be evaluated by shaking table RTHS testing in this section.

4.1 Schemes of testing

Figure 5 shows the principle of RTHS based on shaking table. The frame structure was installed on the shaking table as physical substructure. It interacted with numerical substructure in real time to test the dynamic performance of integral structure. The schemes of RTHS testing in this work are shown in Figure 6. GPU was used to solve numerical model, and the solving time of numerical substructure was controlled by the loop structure of time control in LABVIEW. After solving numerical substructure based on GPU, the interface response y_N was sent to physical

substructure by signal transmission part. The physical substructure was loaded by shaking table after the loading system receiving the response. The dynamic response of the physical substructure *f* was measured by sensors and transmitted to numerical substructure. The dynamic characteristics of the shaking table were compensated by an out-loop controller, which was added between the signal transmission part and the shaking table. The command signal y'_N was generated by the controller. The out-loop controller program was written in SIMULINK and then downloaded into dSPACE to run in real time.

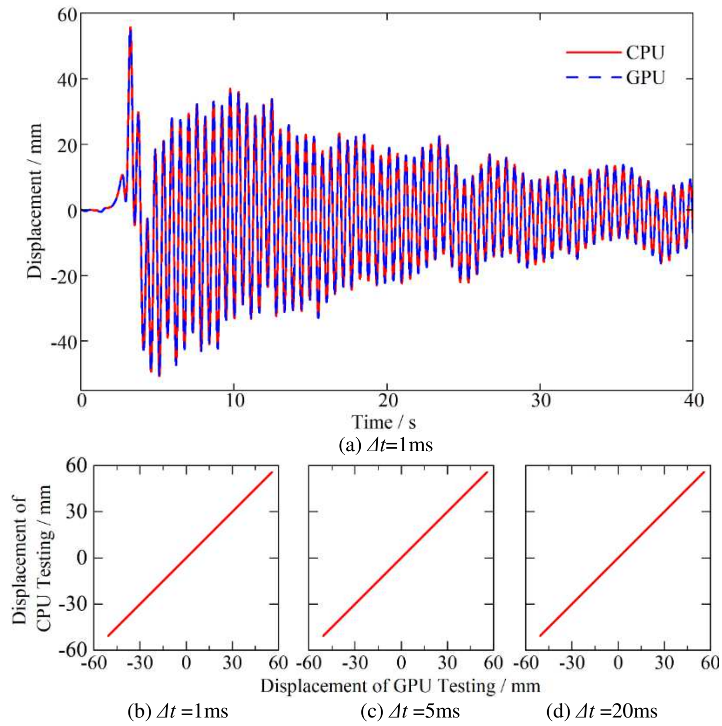


Figure 4 Displacement time histories of the interface based on CPU and GPU

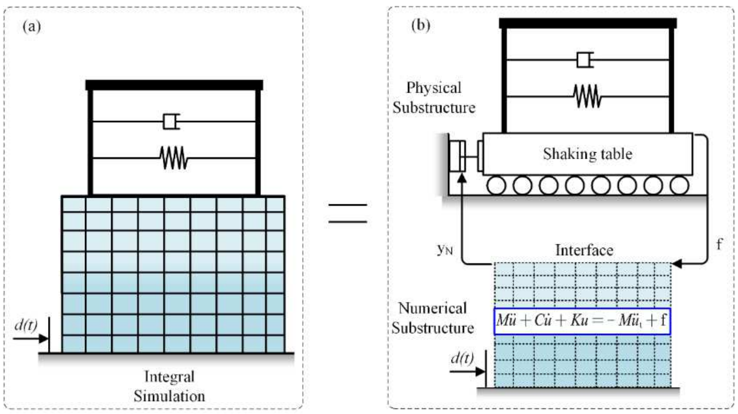


Figure 5 RTHS testing with shaking table

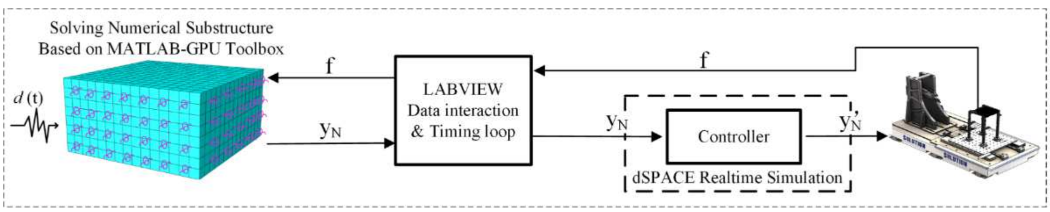


Figure 6 Schemes of the RTHS testing framework

4.2 Testing implementation

The parameters of experiment modal were same as Figure 3 in Section 3.1. In this shaking table RTHS testing, the loading facility is a 0.5 m×0.5 m electromagnetic shaking table, solving part of numerical substructure is same as the GPU server configuration in Table 1, and the other parts are shown in Table 3.

Parts	configuration parameters
Signal transmission	NICompactDAQ-9174、NI-9201、NI-9263
Real-time simulation	dSPACE MicroLabBOX-1202

Table 3 Testing system configuration parameters

The layout of the physical substructure and the sensors were shown in Figure 7.During the testing process, the sensors were connected to the control system of the shaking table. The acceleration sensors were installed on the top of the physical substructure and the shaking table. And the displacement of the physical substructure was measured by a laser displacement meter. The displacement of the shaking table was obtained from the control system of the shaking table. The physical substructure is an aluminum single-layer steel frame, and the bottom was fixed on the shaking table by bolts. The top mass of the physical substructure is 7 kg, and the mass of the four struts is 0.48 kg. As the struts mass only accounts for a small part of the total mass, the sum of the 1/2 struts mass and the top mass were added as the mass of the physical substructure, that is m=7.24 kg. White noise displacement command signal with frequency range of 0.2-15 Hz and amplitude of 2 mm was inputted to the shaking table, and the stiffness and damping of the physical substructure are identified as k=753.25 N/m and c=0.44 N/(m/s). The Kobe and El-Centro waves with PGA=0.5 g were input to shaking table, and the displacement of the physical substructure was compared with integral simulation through central difference method. Figure 8(i) shows the displacement time history results. In Figure 8(ii), X-axis is the displacement time history in simulation, and Y-axis is the displacement time history of physical substructure in RTHS testing. The inevitable error existed when the frame was treated as a linear time-invariant system. Therefore, the measured and simulated response in Figure 8 was not consistent perfectly. However, this work focuses on the feasibility of GPU for RTHS testing. The model of this frame is still accurate enough for evaluating the testing capacity of GPU-based RTHS system.

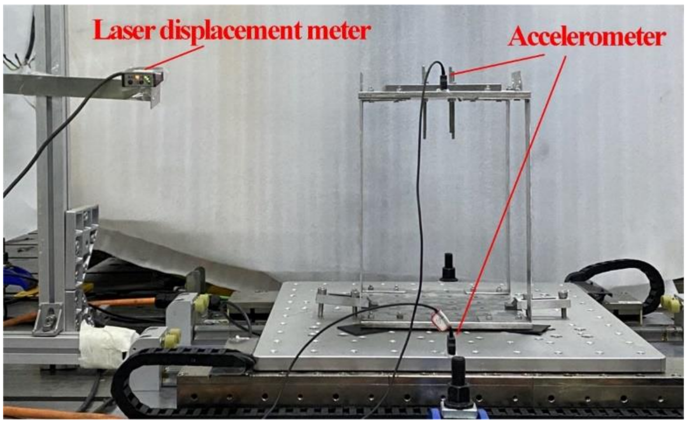


Figure 7 Shaking table RTHS

Furthermore, a white noise displacement signal with frequency range of 0.2-15 Hz and amplitude of 2 mm was input to the shaking table, the input signal and displacement of the shaking table were recorded. A fourth-order transfer function was used to identify characteristics of the shaking table, and the transfer function is shown in Equation (3).

$$G_{st}=\frac{1.023e09}{s^4+427.5s^3+1.481e05s^2+3.041e07s+1.017e09} \tag{3}$$

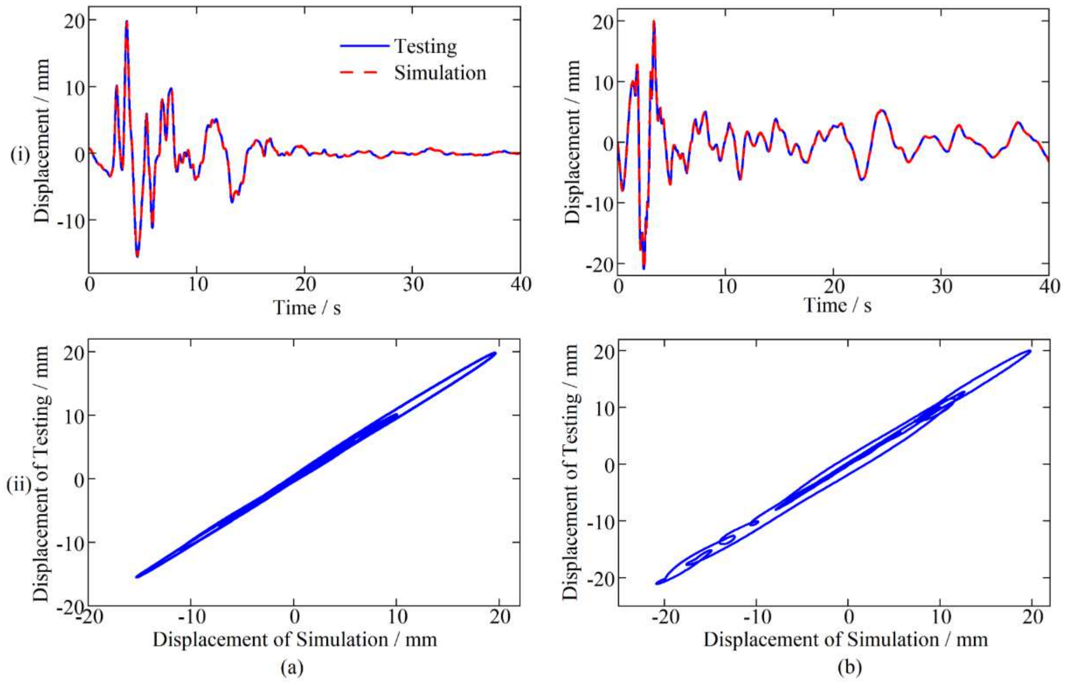


Figure 8 The verifications of model parameters under different seismic excitations: (a)Kobe; (b)El-Centro

Figure 9 shows the transfer function of the shaking table and identified results. The amplitude and phase errors of the shaking table signals are very small when the frequency between 0-3 Hz, hence the shaking table command can be accurately loaded in this frequency range. When the frequency exceeds 3 Hz, the differences of amplitude and phase increase with the rise of frequency, so it is necessary to add an out-loop controller to improve the accuracy of loading. Based on the dynamic model of shaking table shown in Equation (3), the FSCS controller proposed by Tang[34] was used for an out-loop controller. The signal measured by the acceleration sensors was interfered by high-frequency noise, and a low-pass filter (frequency range of 0-20 Hz) was added to the signal collection part. The FSCS controller and the filter are shown in Figure 10.

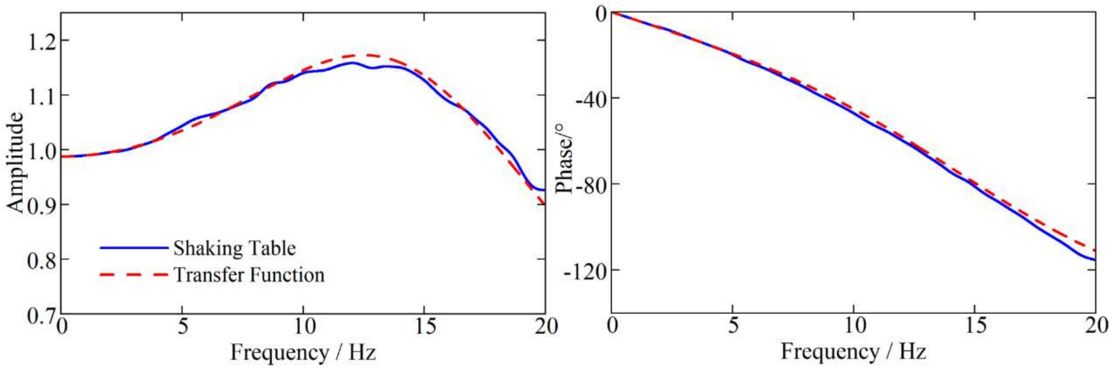


Figure 9 Transfer function of shaking table

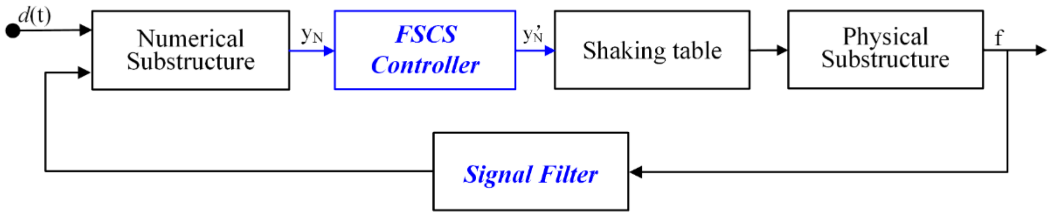
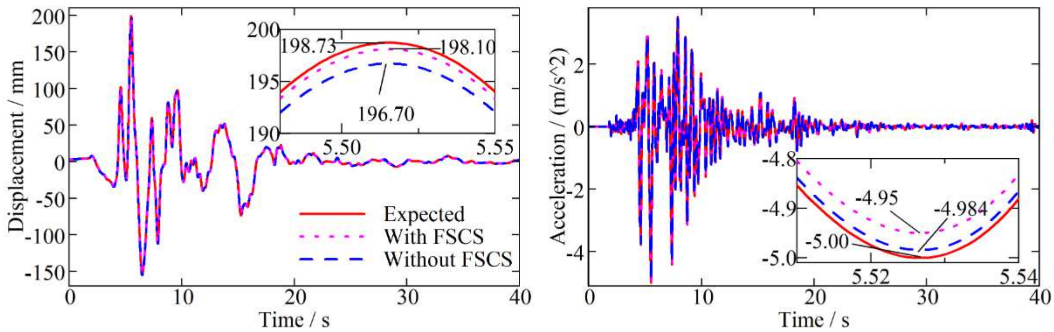


Figure 10 Out-loop control for shaking table

Kobe wave with PGA of 0.5g as input command of the shaking table. Figure 11 shows the expected time histories and the achieved response with or without FSCS controller. The time histories with FSCS controller are much closer to the expectation than those without FSCS. The results show that the FSCS controlled shaking table is suitable for the RTHS testing.



(a) (b)

Figure 11 Performance of FSCS controlled shaking table: (a) displacement, (b) acceleration

In this testing framework, LABVIEW and MATLAB used in the numerical substructure were based on Windows operating system (OS). Windows is not a real-time OS, the real-time performance is unstable[35], the maximum of Windows clock frequency is 1kHz. In order to ensure that more CPU and memory resources are allocated to the numerical calculation, the priority of LABVIEW and MATLAB can be set higher in Windows OS task manager. There is a delay in data communication when LABVIEW invoked MATLAB, which leaded to a delay of nearly 3ms in data transmission between MATLAB and LABVIEW. That is to say, after the dynamic analysis of numerical substructure is completed, it takes 3ms to send the data to the signal transmission part. The MATLAB script stops calculating within this 3ms. Therefore, the minimum time step in the RTHS testing in this work is shown as

Equation (4).

$$\Delta t = t_{solve} + 3 \tag{4}$$

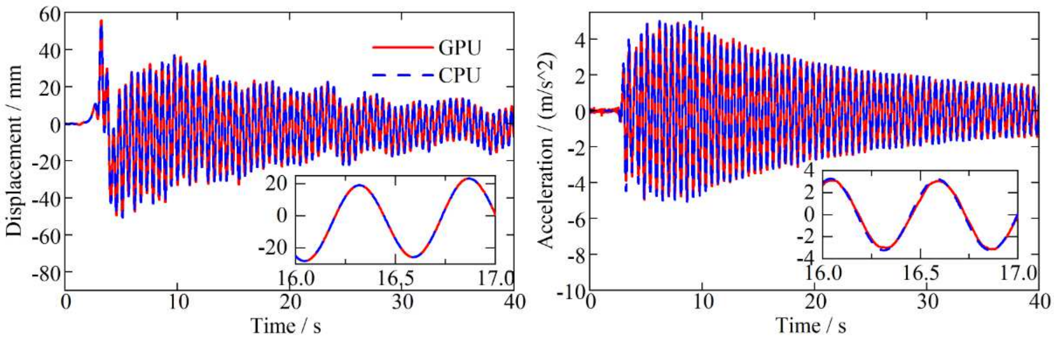
Δt in Equation (4) is the actual time step of numerical integration, and t_{solve} is the actual time cost. For example, the time step would be 4ms when the time cost of solution is 1ms. The accuracy of numerical integration is greatly affected by the time step[36], hence the maximum time step of RTHS testing is 20ms in this work, because of the 3ms in data communication, the actual time cost of solution is 17ms.

4.3 Testing results

Table 4 shows the maximum DOFs of the numerical substructure model solved by GPU and CPU, with different time steps using double and single precision.
Figure 12 shows time histories of the physical substructure solved by GPU and CPU, with Δt=4ms and double precision data. The results of RTHS testing were consistent with the results of integral simulation, which indicated that the testing framework can meet the accuracy requirements of RTHS testing. Figure 13 shows displacement time histories of physical substructure obtained from GPU-based RTHS testing and integral simulation under working conditions NO.2-9 in Table 4. Groups (i) and (ii) in Figure 13 are 4ms and 20ms respectively. The displacement time histories of simulation are taken as X-axis, and the displacement time histories of the GPU-based RTHS testing are taken as Y-axis. All plots of GPU-based RTHS and simulation are basically linear lines. In some cases, the simulated response cannot match perfectly with RTHS testing results because of the modelling errors of physical substructure shown in Figure 8 but not the calculation errors resulted from GPU or CPU. Therefore, we are still sure that, with the same numerical substructure, the precision of RTHS testing based on GPU is same as CPU.

NO.	Δt(ms)	Solving hardware	Precision	DOFs
1	4	CPU	double	1500
2	4	GPU	double	1500
3	4	GPU	single	3168
4	4	CPU	double	1080
5	4	CPU	single	1500
6	20	GPU	double	18876
7	20	GPU	single	27000
8	20	CPU	double	2904
9	20	CPU	single	3888
10	5	GPU	single	3888
11	20	GPU	single	3888

Table 4 Performance comparisons of numerical solution by GPU and CPU



(a) (b)

Figure 12 Dynamic response of physical substructure measured from RTHS testing when the numerical model solved by GPU and CPU with 1500 DOFs, double precision and Δt=4 ms: (a) displacement, (b) acceleration

In order to discuss the influence of integration step in GPU-based RTHS testing, a numerical substructure model with 3888 DOFs was solved by GPU and CPU. NO.8 and 9 in Table 4 show that the minimum Δt are 5ms and 20ms respectively. Figure 14 shows displacement and acceleration time histories of physical substructure in the RTHS testing and integral simulation. The peak error of displacement between the RTHS testing and the simulation is 6.74% when Δt=5 ms, and 13.45% when Δt=20 ms. The peak error of acceleration between the RTHS testing and the simulation is 6.76% when Δt=5 ms, and 16.91% when Δt=20ms. Therefore, GPU solution can carry out the RTHS testing with smaller time step than CPU solution and improve the accuracy of solution.

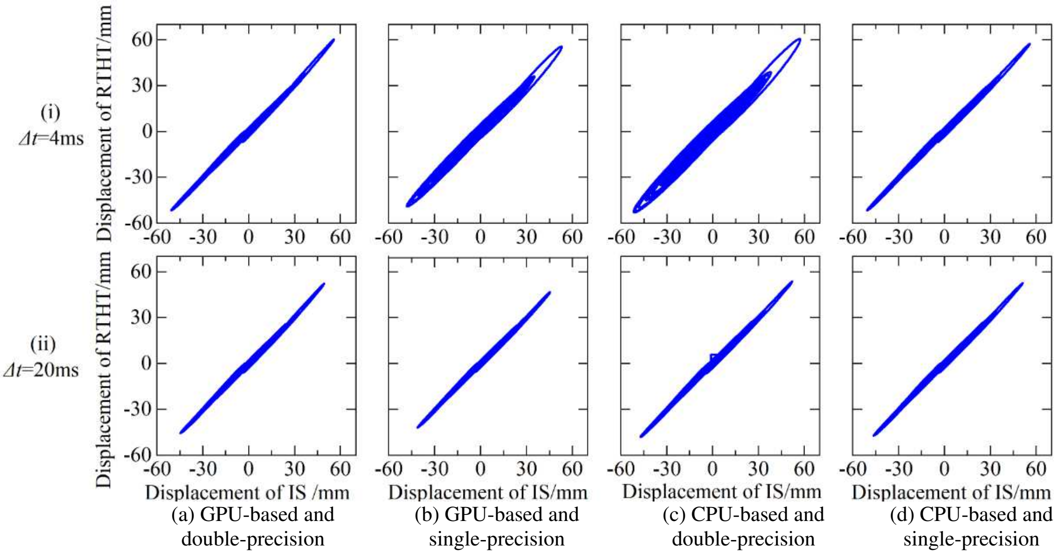
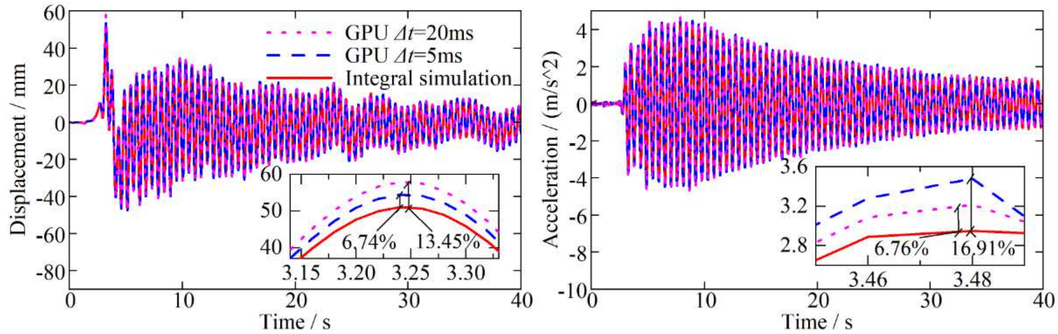


Figure 13 Displacement of physical substructure in RTHS testing and integral simulation(IS)



(a) (b)

Figure 14 The influence of time step sizes on testing accuracy: (a) displacement, (b) acceleration

4.4 Discussion of testing results

NO. 2-5 in Table 4 shows that, with Δt=4ms and double precision, the maximum DOFs is 1500 solved by GPU and 1080 solved by CPU, the advantage of GPU

solving is not obvious. When the precision is single, the maximum DOFs is 3168 solved by GPU, and 1500 solved by CPU. The advance of solution by GPU with single precision is higher than double precision.

NO. 6-9 in Table 4 shows that, with Δt=20 ms and double precision, the maximum DOFs is 18,876 solved by GPU, and 2904 solved by CPU is. When the precision is single, the maximum DOFs is 27,000 solved by GPU, and 3888 solved by CPU. It can be seen that when Δt=20 ms, the advantage of DOFs solved by GPU is obvious, the maximum DOFs solved by GPU far exceed those solved by CPU. The RTHS testing with large-scale numerical substructure can be realized by GPU solution, which cannot be solved by CPU with the same time step.

In RTHS testing, the size of integration step needs to meet the requirement of real time. Using the GPU server and PCT, the RTHS testing with 27,000 DOFs can be carried out with Δt=20 ms, it needs 926 ms to solve by CPU at least. Solution by CPU is far from meeting the real time requirement of numerical solution. When the numerical substructure is 3888 DOFs, the time step is reduced from 20 ms(CPU-based) to 5ms(GPU-based). According to results of the different time steps, the solution accuracy can be improved with smaller time step. The time step can be reduced, and the testing accuracy can be improved through the GPU-based solution in RTHS testing.

Due to the instability of clock frequency in Windows OS and the time delay caused by communication, when Δt is 4ms, the actual time cost of solution is 1ms. The performance of GPU-based RTHS testing is limited. The maximum DOFs is 27,000 in the GPU-based RTHS testing because of the limitation of MATLAB in this work. Hence, the method of GPU solution, GPU parallel calculation, resource allocation and data communication still can be optimized, which may further improve the scale and efficiency of the solutions.

5. CONCLUSION

The advantage of RTHS is to combine physical testing with numerical simulation and improve the testing capability of existing equipment. Due to the limitations of calculation cost, only large integration time step (e.g., 20 ms) and less DOFs (e.g., 2000) for numerical substructure solutions are allowed in the current RTHS testing. It is detrimental to utilize the testing capability and accuracy of RTHS testing. In order to overcome these drawbacks, this work established a RTHS framework based on GPU and Matlab. The performance of framework was verified by numerical simulation and experimental testing. The following conclusions were drawn.

(1) Under the condition of Δt=20ms and single precision data, using the RTHS testing framework based on MATLAB and GPU (NVIDIA Tesla V100, 10.60 TFLOPS), the maximum DOFs for numerical substructure can reach 27,000 in real-time solution, the speed of GPU calculation is 55times faster than that of CPU (Intel Xeon E5-2690 V4, 14 cores 2.6 GHz). And using this framework the numerical substructure based on CPU can only reach 3888 with Δt=20 ms in real-time solution. The testing capacity of RTHS is significantly enhanced by GPU solving.
(2) With the numerical substructure of 3888 DOFs and the model parameters are single precision, the minimum Δt is 5ms by the RTHS testing based on GPU, and

that is 20ms based on CPU. The RTHS testing results are compared with the integral simulation, the peak errors of displacement and acceleration time history are reduced, Using the testing framework based on GPU in this work can reduce the time step and improve the testing accuracy.

(3) MATLAB and LABVIEW under Windows operating system are used in this work. Because of instability of Windows clock frequency and delay of communication between two software, it takes 3 ms to transfer the response data, hence the performance of GPU calculation is limited. Further research is needed to consider real-time performance of operating system and optimization of software interaction, it can further decrease the time step and enhance the DOFs of numerical substructures in RTHS testing.

ACKNOWLEDGEMENT

The authors gratefully acknowledge the support of the National Natural Science Foundation of China under grant number 51978016.

REFERENCE

[1] M. Nakashima, H. Kato, E. Takaoka, Development of real-time pseudo dynamic testing, Earthquake Engineering & Structural Dynamics, 21 (1992) 79-92.

[2] D.P. Mccrum, M.S. Williams, An overview of seismic hybrid testing of engineering structures, Engineering Structures, 118 (2016) 240-261.

[3] A. Blakeborough, M. Williams, A. Darby, D. Williams, The development of real–time substructure testing, Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences, 359 (2001) 1869-1891.

[4] F. Zhu, J.T. Wang, F. Jin, Y. Gui, Comparison of explicit integration algorithms for real-time hybrid simulation, Bulletin of Earthquake Engineering, 14 (2016) 89-114.

[5] M. Wang, F.T.K. Au, Assessment and improvement of precise time step integration method, Computers & Structures, 84 (2006) 779-786.

[6] F. Zhu, J. Wang, F. Jin, M. Zhou, Y. Gui, Simulation of large-scale numerical substructure in real-time dynamic hybrid testing, Earthquake Engineering and Engineering Vibration, 13 (2014) 599-609.

[7] B. Wu, G. Xu, Q. Wang, M.S. Williams, Operator-splitting method for real-time substructure testing, Earthquake Engineering & Structural Dynamics, 35 (2006) 293-314.

[8] B. Wu, H. Bao, J. Ou, S. Tian, Stability and accuracy analysis of the central difference method for real-time substructure testing, Earthquake Engineering & Structural Dynamics, 34 (2005) 705-718.

[9] S. Chang, Y. Sung, An enhanced explicit pseudodynamic algorithm with unconditional stability, in: 100th Anniversary Earthquake Conference, 2006.

[10] M. Nakashima, Integration techniques for substructure pseudo-dynamic test, in: 4th US National Conference on Earthquake Engineering, 1990. 5, 1990.

[11] M. Nakashima, N. Masaoka, Real-time on-line test for MDOF systems, Earthquake engineering & structural dynamics, 28 (1999) 393-420.

[12] C. Chen, J.M. Ricles, Stability analysis of SDOF real-time hybrid testing systems with explicit integration algorithms and actuator delay, Earthquake Engineering & Structural Dynamics, 37 (2008) 597-613.

[13] Y. Chae, K. Kazemibidokhti, J.M. Ricles, Adaptive time series compensator for

delay compensation of servo-hydraulic actuator systems for real-time hybrid simulation, Earthquake Engineering & Structural Dynamics, 42 (2013) 1697-1715.
[14] V. Saouma, D.H. Kang, G. Haussmann, A computational finite-element program for hybrid simulation, Earthquake engineering & structural dynamics, 41 (2012) 375-389.
[15] R.R. Schaller, Moore's law: past, present and future, IEEE spectrum, 34 (1997) 52-59.
[16] D. Kirk, NVIDIA CUDA software and GPU parallel computing architecture, in: ISMM, 2007, pp. 103-104.
[17] C. Filelis-Papadopoulos, G.A. Gravvanis, P. Matskanidis, K.M. Giannoutakis, On the GPGPU parallelization issues of finite element approximate inverse preconditioning, Journal of computational and applied mathematics, 236 (2011) 294-307.
[18] G.A. Gravvanis, C. Filelis-Papadopoulos, K.M. Giannoutakis, Solving finite difference linear systems on GPUs: CUDA based parallel explicit preconditioned biconjugate conjugate gradient type methods, The Journal of Supercomputing, 61 (2012) 590-604.
[19] Z. Liu, Y. Wang, X. Hua, H.P. Zhu, Z.W. Zhu, Optimization of wind turbine TMD under real wind distribution countering wake effects using GPU acceleration and machine learning technologies, Journal of Wind Engineering and Industrial Aerodynamics, 208 (2020).
[20] S. Dazzi, R. Vacondio, P. Mignosa, Internal boundary conditions for a GPU-accelerated 2D shallow water model: Implementation and applications, Advances in Water Resources, 137 (2020) 103525-.
[21] M. Hazeghian, A. Soroush, Numerical modeling of dip-slip faulting through granular soils using DEM, Soil Dynamics & Earthquake Engineering, 97 (2017) 155-171.
[22] M. Durand, P. Marin, F. Faure, B. Raffin, DEM-based simulation of concrete structures on GPU, European journal of environmental and civil engineering, 16 (2012) 1102-1114.
[23] X. Lu, B. Han, M. Hori, C. Xiong, Z. Xu, A coarse-grained parallel approach for seismic damage simulations of urban areas based on refined models and GPU/CPU cooperative computing, Advances in Engineering Software, 70 (2014) 90-103.
[24] Y. Cai, G. Li, H. Wang, G. Zheng, S. Lin, Development of parallel explicit finite element sheet forming simulation system based on GPU architecture, Advances in Engineering Software, 45 (2012) 370-379.
[25] W. Zhang, Z.-h. Zhong, C. Peng, W.-h. Yuan, W. Wu, GPU-accelerated smoothed particle finite element method for large deformation analysis in geomechanics, Computers and Geotechnics, 129 (2021) 103856.
[26] X. Cao, Y. Cai, X. Cui, A parallel numerical acoustic simulation on a GPU using an edge-based smoothed finite element method, Advances in Engineering Software, 148 (2020) 102835.
[27] S.F. Johnsen, Z.A. Taylor, M.J. Clarkson, J. Hipwell, M. Modat, B. Eiben, L. Han, Y. Hu, T. Mertzanidou, D.J. Hawkes, NiftySim: A GPU-based nonlinear finite element package for simulation of soft tissue biomechanics, International journal of computer assisted radiology and surgery, 10 (2015) 1077-1095.
[28] G.P. Krawezik, G. Poole, Accelerating the ANSYS direct sparse solver with GPUs, in: Symposium on Application Accelerators in High Performance Computing, SAAHPC, 2009.
[29] X. Lu, Y. Tian, S. Cen, H. Guan, L. Xie, L. Wang, A high-performance

quadrilateral flat shell element for seismic collapse simulation of tall buildings and its implementation in OpenSees, Journal of Earthquake Engineering, 22 (2018) 1662-1682.
[30] J.W. Suh, Y. Kim, Accelerating MATLAB with GPU computing: A primer with examples, Newnes, 2013.
[31] S.K. Lee, E. Park, K.W. Min, J.H. Park, Real-time substructuring technique for the shaking table test of upper substructures, Engineering Structures, 29 (2007) 2219-2232.
[32] N. Patrascoiu, Modeling and simulation of the DC motor using MatLAB and LabVIEW, International Journal of Engineering Education, 21 (2005) 49-54.
[33] N. Ploskas, N. Samaras, GPU programming in MATLAB, Morgan Kaufmann, 2016.
[34] Z. Tang, M. Dietz, Y. Hong, Z. Li, Performance extension of shaking table-based real-time dynamic hybrid testing through full state control via simulation, Structural Control and Health Monitoring, 27 (2020).
[35] C. Lina, Z. Qiang, The design and implementation of real-time HILS based on RTX platform, in: IEEE 10th International Conference on Industrial Informatics, IEEE, 2012, pp. 276-280.
[36] B. Wu, H. Bao, J. Ou, S. Tian, Stability and accuracy analysis of central difference method for real-time substructure testing, Earthquake Engineering & Structural Dynamics, 34 (2010) 705-718.

Figures

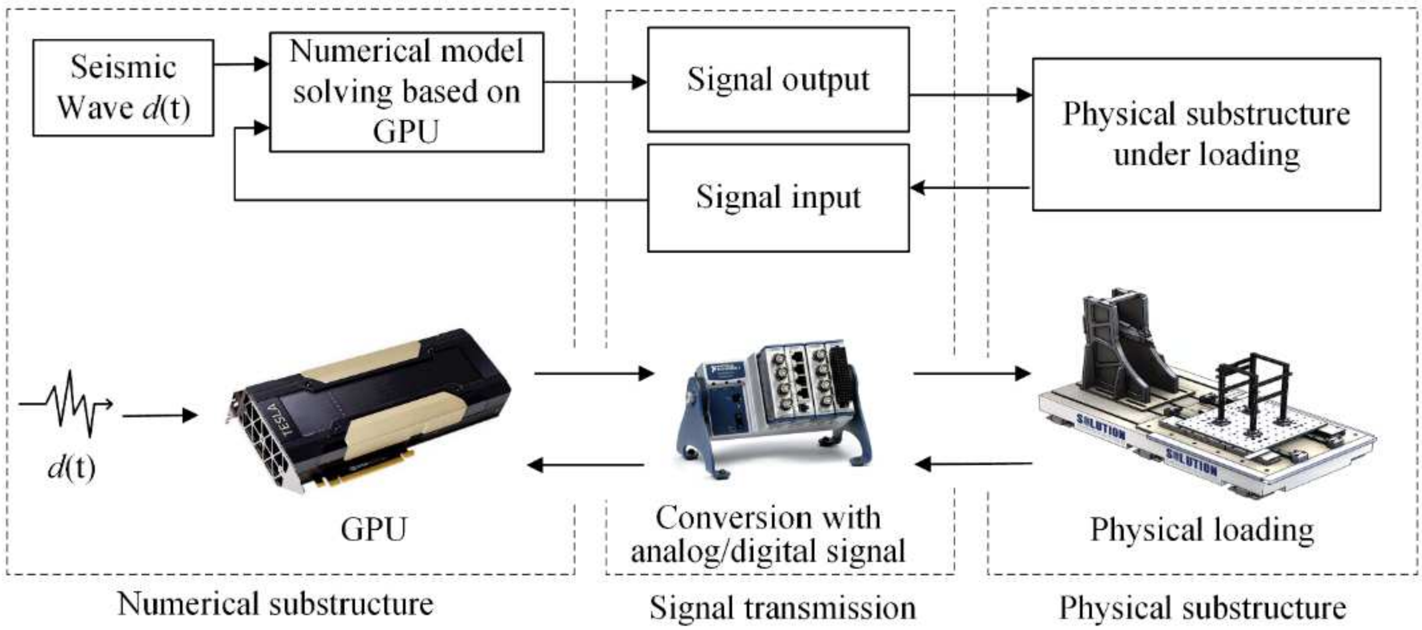


Figure 1

GPU-based framework for RTHS.

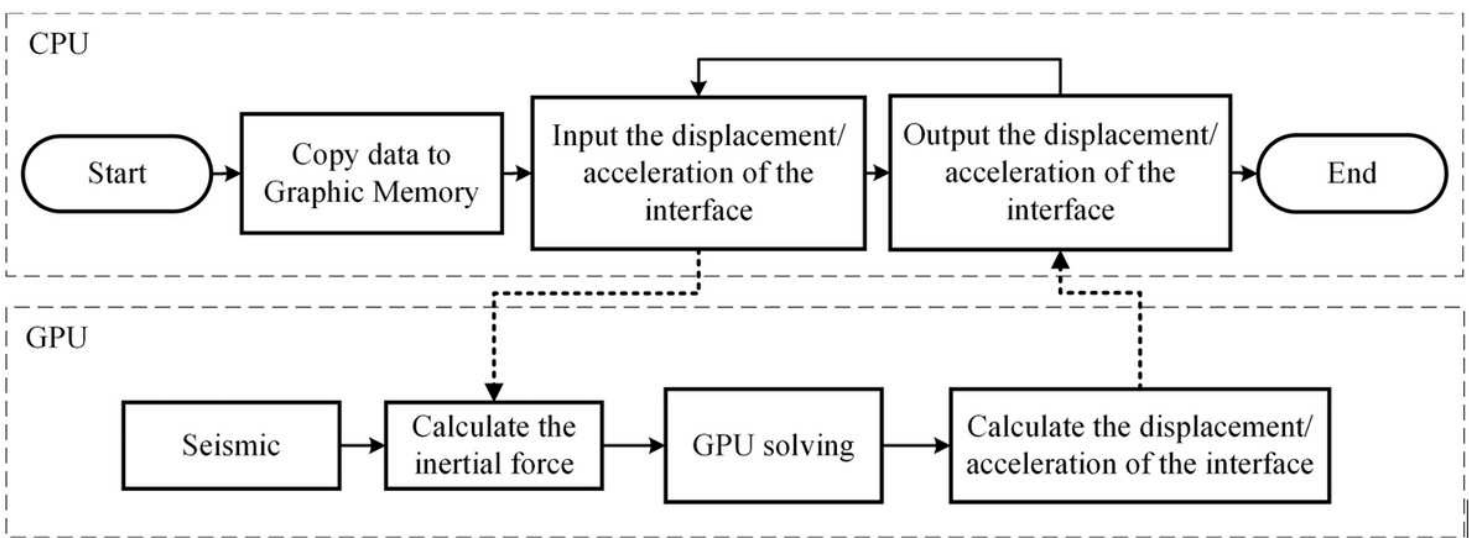


Figure 2

Solution of numerical substructure in GPU-based RTHS testing.

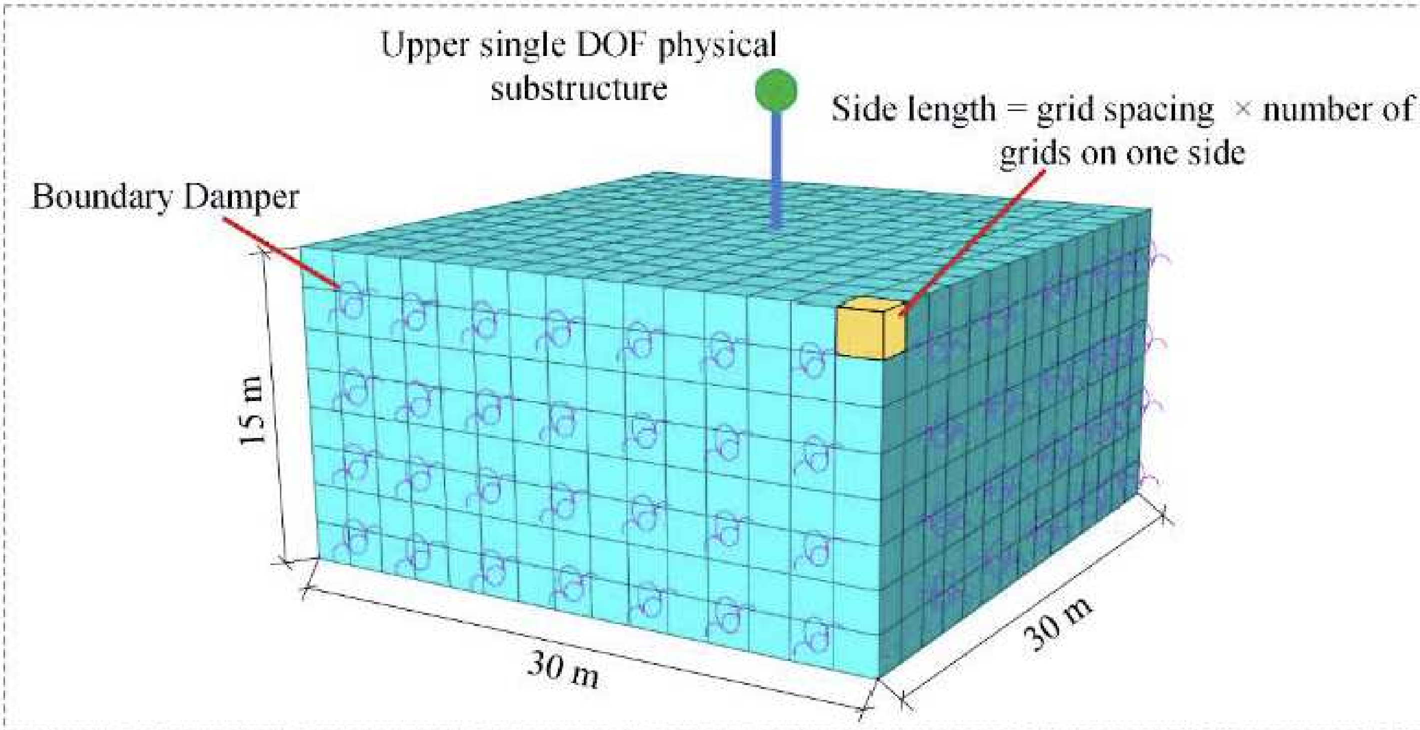


Figure 3

Soil-structure interaction system.

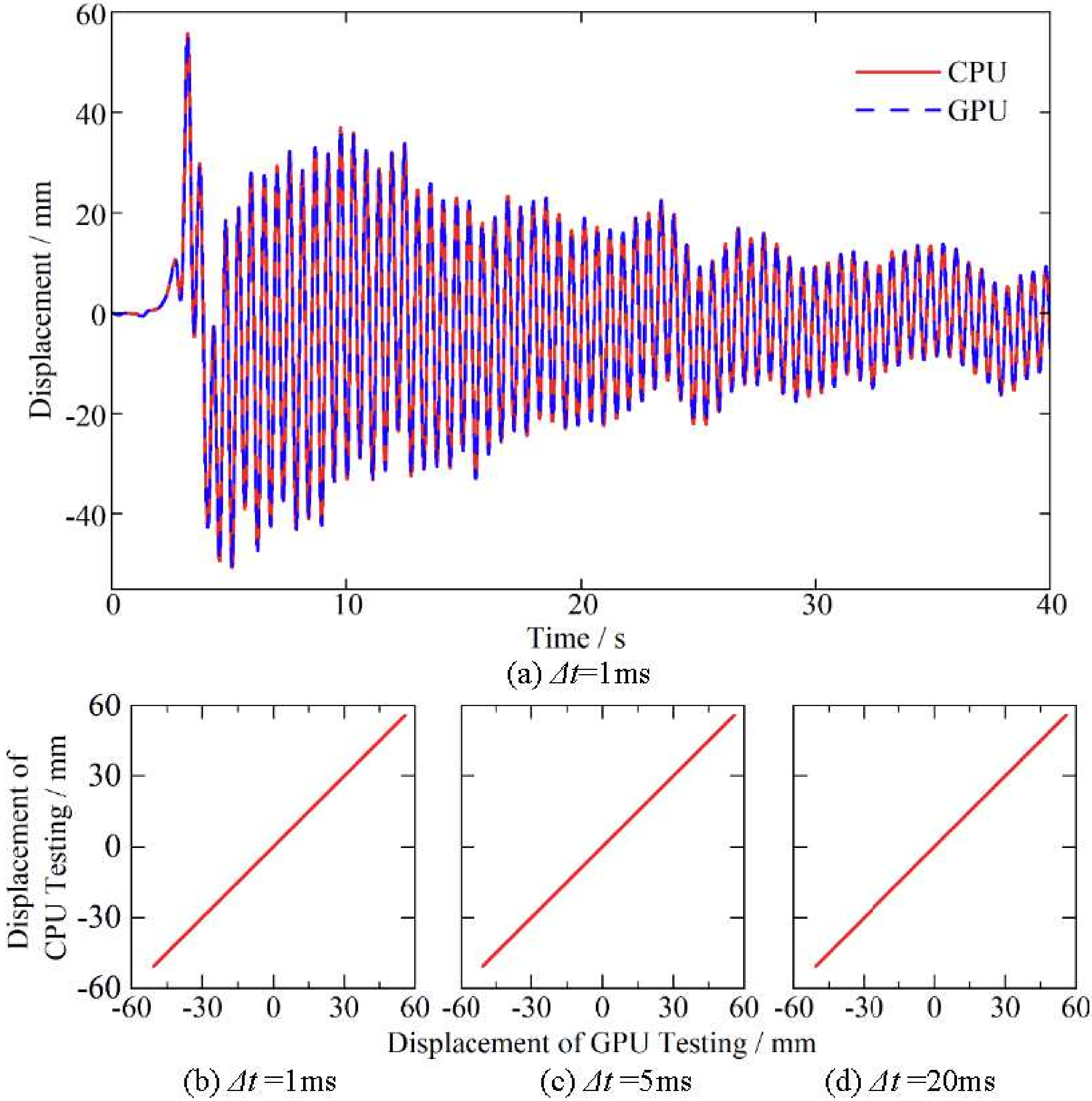


Figure 4

Displacement time histories of the interface based on CPU and GPU

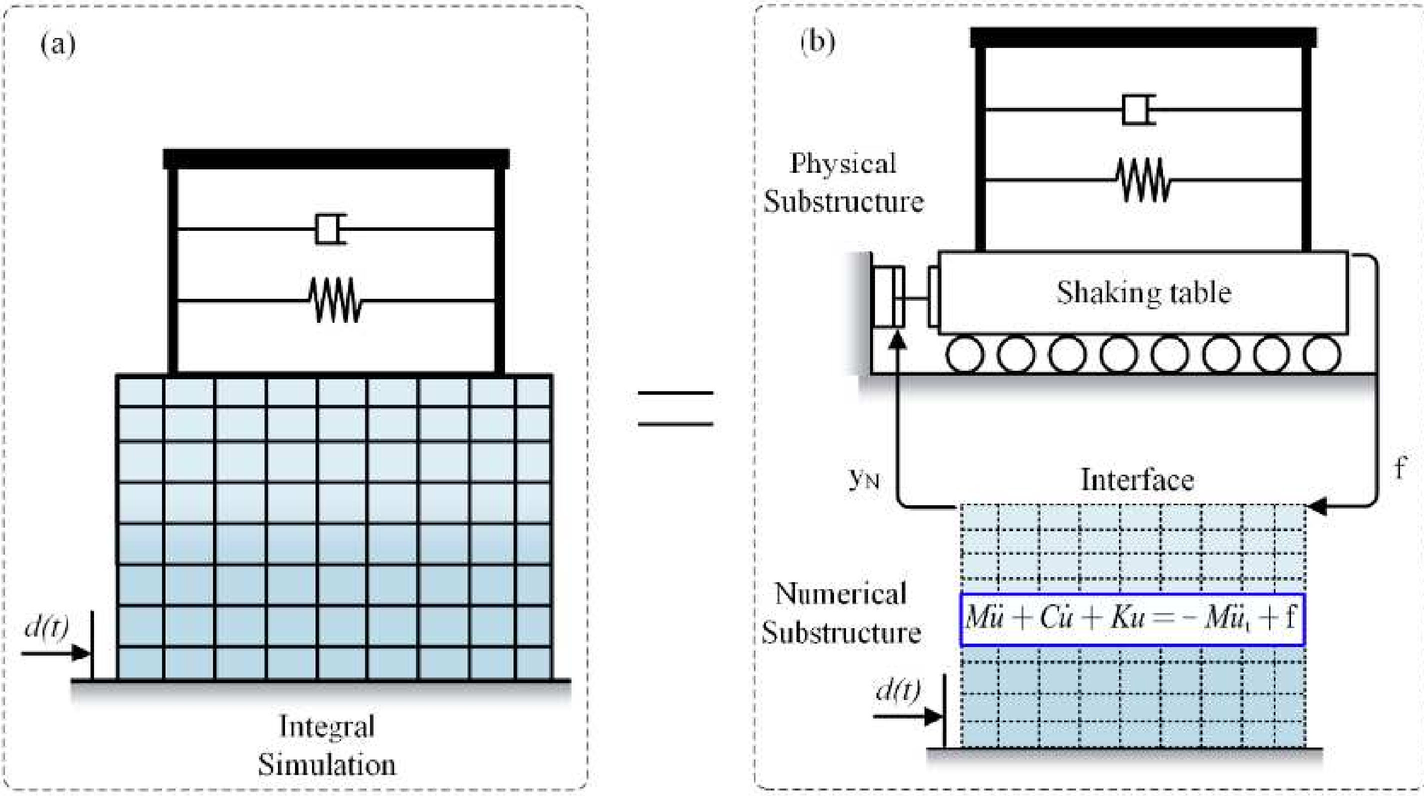


Figure 5

RTHS testing with shaking table

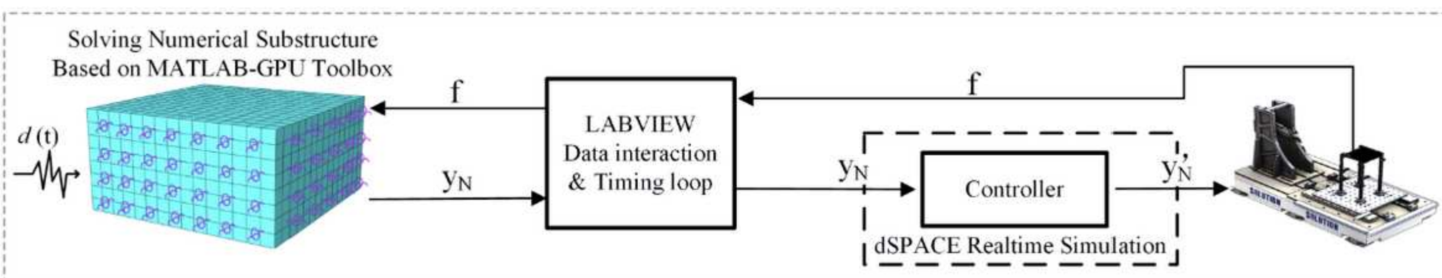


Figure 6

Schemes of the RTHS testing framework

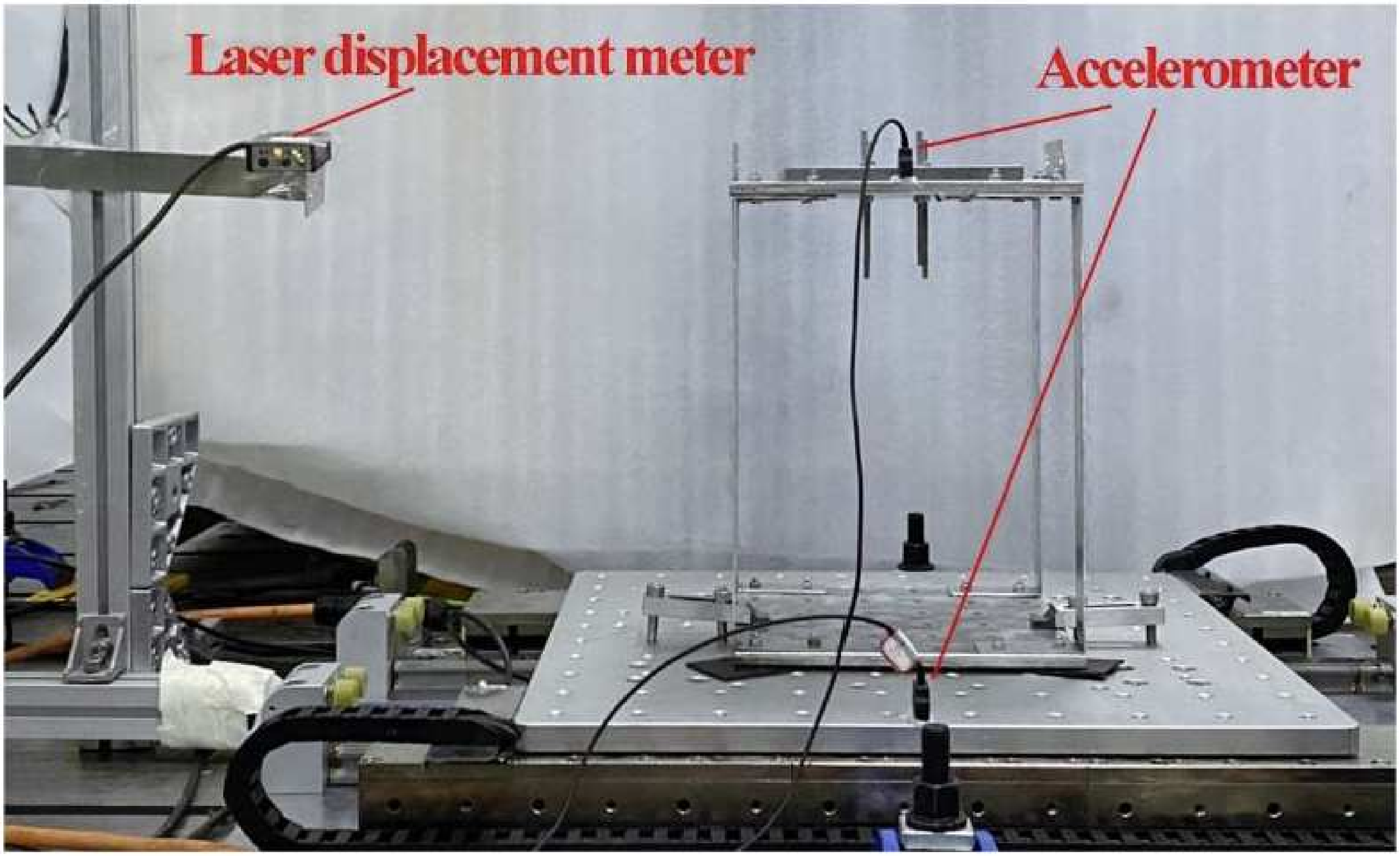


Figure 7

Shaking table RTHS

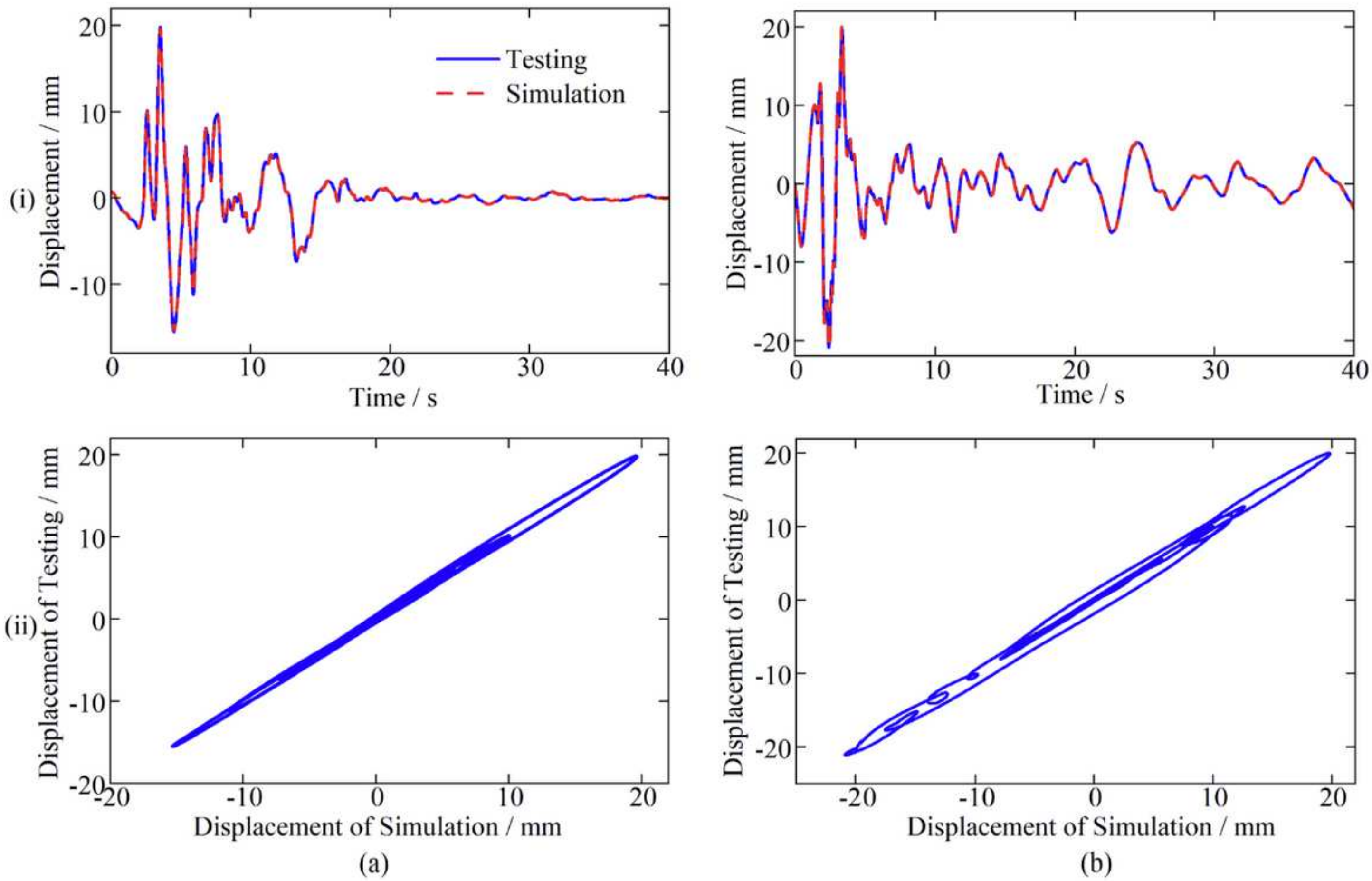


Figure 8

The verifications of model parameters under different seismic excitations: (a)Kobe; (b)El-Centro

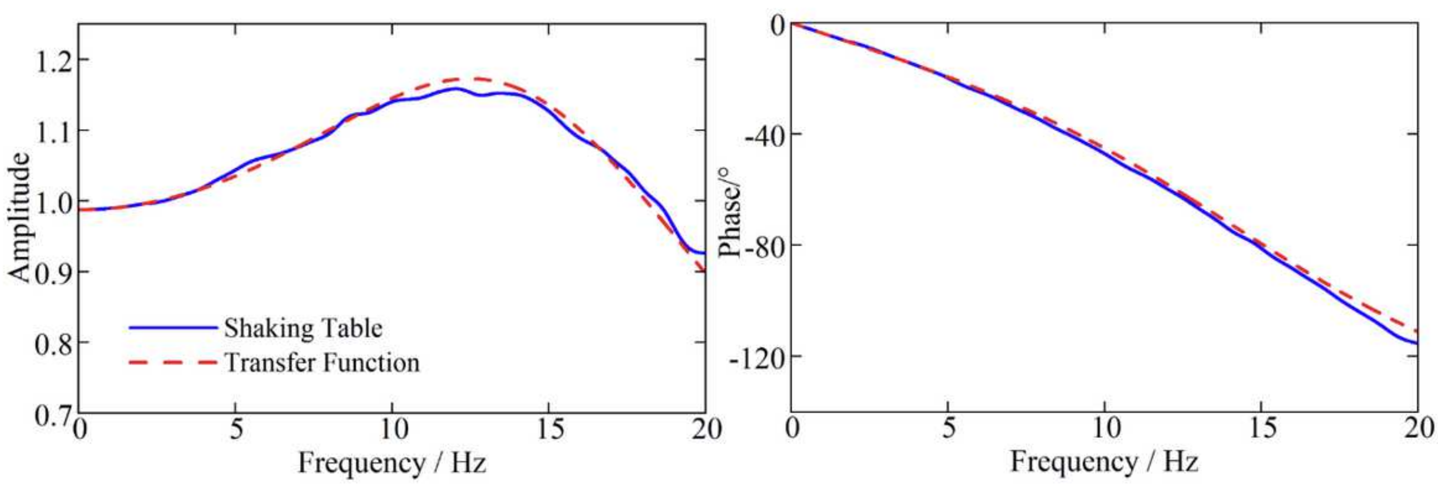


Figure 9

Transfer function of shaking table

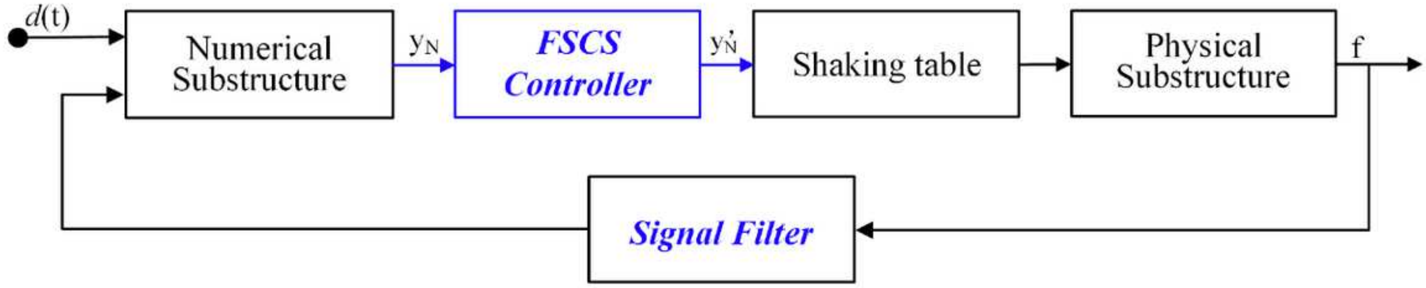


Figure 10

Out-loop control for shaking table

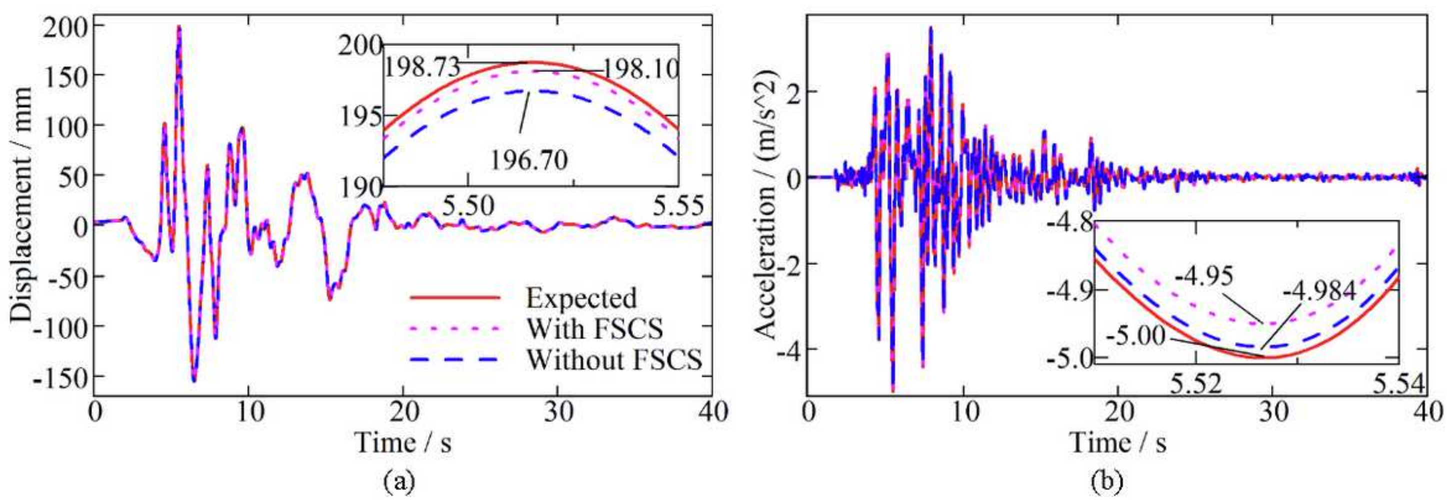


Figure 11

Performance of FSCS controlled shaking table: (a) displacement, (b) acceleration

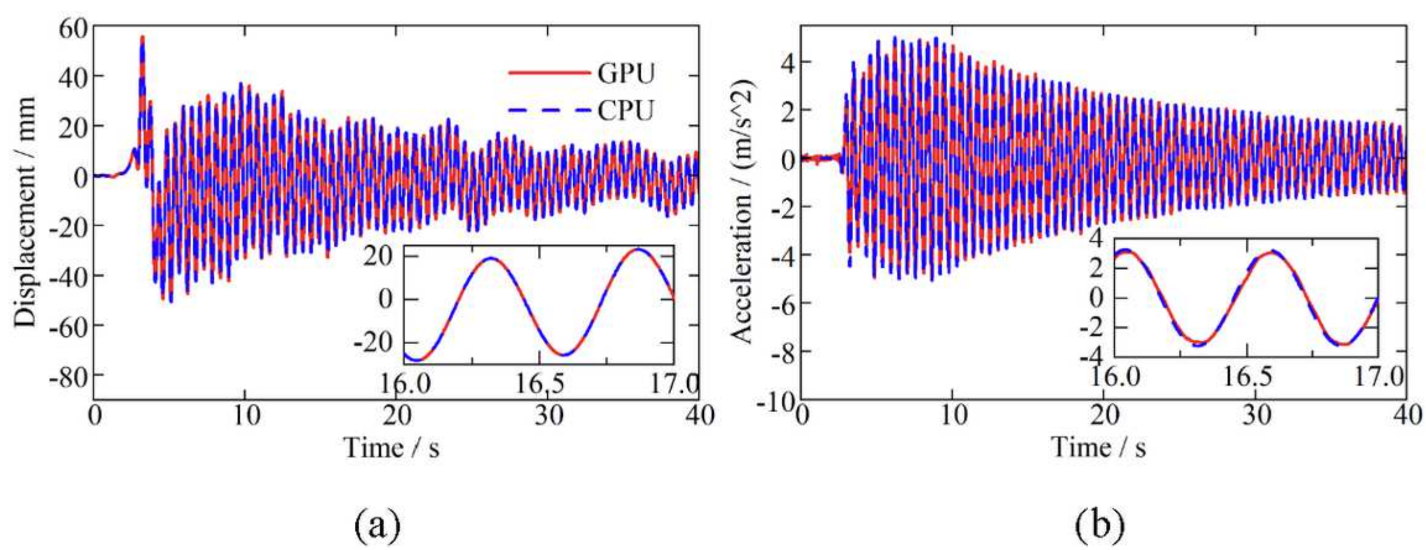


Figure 12

Dynamic response of physical substructure measured from RTHS testing when the numerical model solved by GPU and CPU with 1500 DOFs, double precision and Δt=4 ms: (a) displacement, (b) acceleration

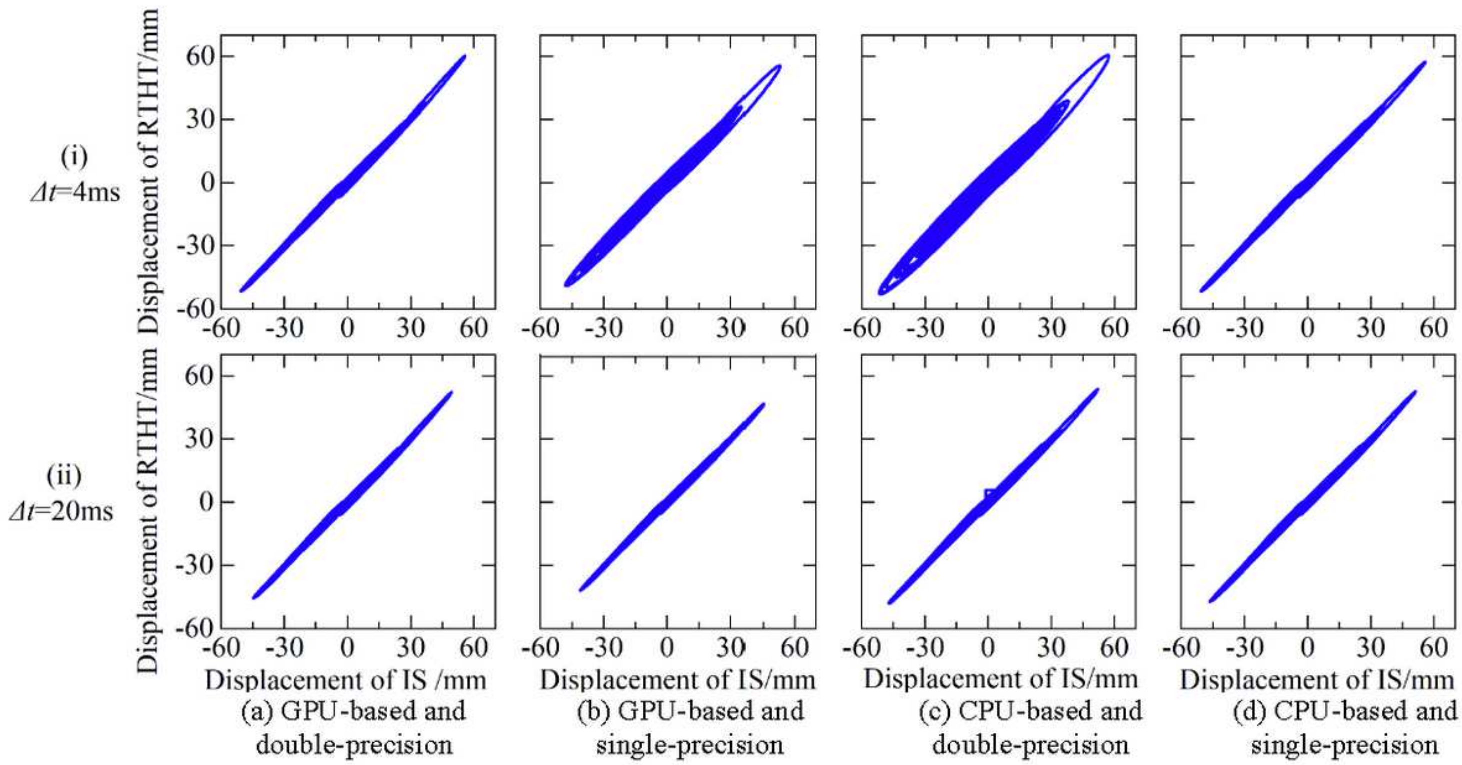


Figure 13

Displacement of physical substructure in RTHS testing and integral simulation(IS)

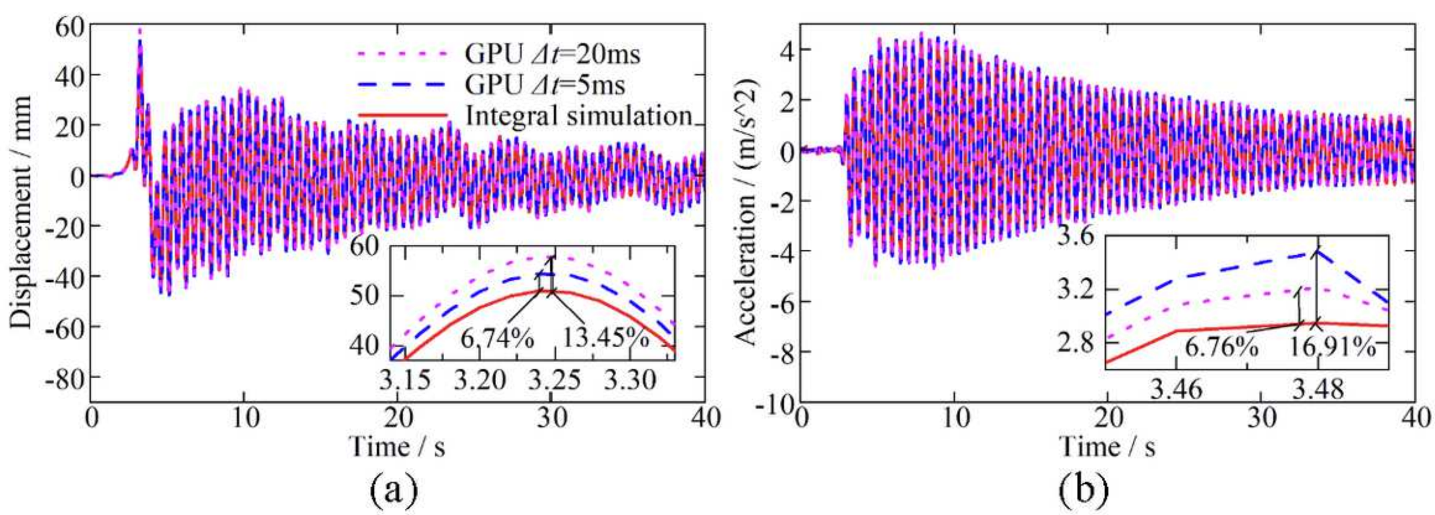


Figure 14

The influence of time step sizes on testing accuracy: (a) displacement, (b) acceleration