Experimental Software Framework for Hybrid Simulation

Andreas Schellenberg & Stephen Mahin

With contributions from:
Gregory Fenves, Bozidar Stojadinovic, Frank McKenna, Tony Yang, Yoshikazu Takahashi and Hong Kim

Department of Civil and Environmental Engineering
University of California, Berkeley
Pioneering efforts at Berkeley

- Initial theoretical foundations (1973)
- Random & systematic error propagation studies - Williams (1979-81)
- Refinement and implementation of stable explicit formulations - Shing (1980-84)
- Experimental/numerical partitioned simulations with mixed implicit/explicit operators - Dermitzakis (1983-85)
- Multi-directional testing - Thewalt (1984-87)
- Force- and mixed force- and displacement-based control - Thewalt (1984-87)
- Implicit formulations based on numerical and analog correctors - Thewalt (1984-87)
- “Effective force” dynamic testing - Thewalt (1984-87)
- Error bound estimation - Thewalt, Stojadinovic, Mosqueda (1992-04)
- Theory and implementation of network-based local and distributed simulation - Thewalt, Stojadinovic, Campbell, Mosqueda (1987-04)
- Event-driven digital control - Stojadinovic & Mosqueda (01-04)
- Force/displacement based control switching - Mosalam (2003-05)
Hybrid Simulation

\[ M \ddot{u} + C \dot{u} + P_r(u) = P(t) \]

- Analytical model of structural energy dissipation and inertia
- Physical model of structural resistance

Quasi-Static Loading
- Dynamic Loading
  - Seismic
  - Wind
  - Blast/Impact
  - Wave
  - Traffic
Hybrid Simulation

- Model the well understood parts of a structure in a finite element program on one or more computers
- Leave the construction and testing of the highly nonlinear and/or numerically hard to model parts of the structure in one or more laboratories
- Can be considered as a conventional finite element analysis where physical models of some portions of the structure are embedded in the numerical model
Testing Methods

- Conventional hybrid simulation test where specimen is loaded using a ramp-and-hold loading procedure
- Continuous test where specimen is loaded at a continuous slow to moderately slow rate to avoid load relaxations
- Real-time test where specimen is loaded at correct velocities to account for rate-dependent material behaviors
- Geographically distributed network test
Advantages

- Enables dynamic testing of full-scale specimens
- Quasi-static testing equipment sufficient
- Fewer restrictions on size, weight and strength of a specimen
Advantages

- Geometric nonlinearities, three-dimensional effects, multi-support excitations and soil-structure interactions can be incorporated into the analytical model.

- Internationally, geographically distributed testing is made possible.
Main Challenge

- Lack of a common framework for development and deployment
- Problem specific implementations which are site and control system dependant
- Such highly customized software implementations are difficult to adapt to different structural problems

Need a robust, transparent, adaptable, and easily extensible framework for research and deployment
OpenFresco

- **Open** source **Framework** for **Experimental Setup** and **Control**
- Enable domain researchers to carry out Hybrid Simulations without specialized knowledge
- Allow IT and hybrid simulation specialists to extend frontiers of methodology, focusing only on their portions of interest
  - Facilitate additions and extensions for new equipment and procedures

- **Object-oriented programming approach**
Rethinking implementation strategies

Embed test specimen(s) in an existing computational framework of users choice

Typical features of an analysis framework

 Proper numerical model u
Rethinking implementation strategies

Embed test specimen(s) in an existing computational framework of users choice

Typical features of an analysis framework

Define element as an “Experimental Element”

OpenFresco

Laboratory

ADMINISTRATIVE FUNCTIONS

RECORDERS

COMMUNICATION

NODAL GEOMETRY

BOUNDARY CONDITIONS

MASS AND DAMPING PROPERTIES

LOADING NUMERICAL ELEMENT 1

ELEMENT TYPES & NUMERICAL ELEMENT 2

EXPERIMENTAL ELEMENT

STATE DETERMINATION

OPENFRESCO

LABORATORY CONTROLLERS AND DAQS

nees@berkeley
OpenFresco Components

FE-Software

GenericElement

Experimental Element

Experimental Site

Experimental Setup

Experimental Control

Control System in Laboratory

provides all features of unmodified computational framework, including parallel and network computing

represents the part of the structure that is physically tested and provides the interface between the FE- software and the experimental software framework

stores data and provides communication methods for distributed testing

transforms between the experimental element degrees of freedom and the actuator degrees of freedom (linear vs. non-linear transformations)

interfaces to the different control and data acquisition systems in the laboratories

provides control of physical actuators as well as data acquisition using physical instrumentation devices
Three-Tier Software Architecture

Top-Tier / Client
The top-most level of the application is the user interface.
In the case of OpenFresco this is the FE-software.

Middle-Tier / Application Server
The middle layer provides process management services and controls transactions.
Allows FE-software using different languages (Matlab, Fortran, C, C++) to communicate with physical specimens.

Third-Tier / Back-End Server
The third tier usually stores and manages information in a database for retrieval by the user.
Structure or structural subassembly from which the FE-software is requesting information.
OpenFresco Components

local deployment
network
deployment
OpenFresco Components

Experimental Element
Transforms between the global element degrees of freedom in the FE-Software and the basic element degrees of freedom in the experimental element

Consider element in structure
Two coordinate systems used in FE analysis
OpenFresco Components

**Experimental Site**
Stores data and provides communication methods for distributed testing

LocalExpSite available for local testing and RemoteExpSite/ActorExpSite pair available for geographically distributed testing

Utilizes communication channels with TCP/IP, NHCP or UDP communication protocols
OpenFresco Components

Experimental Setup

Transforms between the basic experimental element degrees of freedom in OpenFresco and the actuator degrees of freedom in the laboratory (linear vs. non-linear transformations are available)

Controlled displacements and acquired forces

d1, q1

d2, q2

d3, q3

Cantilever Basic System

\[ T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -L_0 \\ 0 & 1 & L_1 \end{bmatrix} \]
OpenFresco Components

Experimental Control
Interfaces to the different control and data acquisition systems in the laboratories (IP addresses and port numbers)
Finite Element Software

- OpenFresco is independent of FE-software
- Can be used with any FE-software allowing:
  - the addition of elements
  - hybrid simulation compatible integration methods
  - appropriate communication channels
- Abaqus, LS-Dyna, Matlab, OpenSees, SAP, SimCor, Zeus-NL and similar programs can in principle be used with OpenFresco
- Currently, validated generic experimental elements are available for:
  - Matlab
  - OpenSees
  - Simcor
  - LS-Dyna
OpenSees

The object oriented approach of OpenSees facilitates:

- Domain researchers to add materials, numerical elements and specialized recorders
- Addition of new types of direct integration methods
- Addition of new experimental elements
- Numerical execution on super-computers
- Partitioning of numerical models and execution on networks of computers
OpenSees Components

- ModelBuilder: constructs the objects in the model and adds them to the Domain.
- Domain: holds the state of the model at time $t_i$ and $t_i+dt$.
- Recorder: monitors user defined parameters in the model during the analysis.
- Analysis: moves the model from state at time $t_i$ to state at time $t_i+dt$. 
Direct Integration Methods

\[ M \cdot \ddot{u}_n + C \cdot \dot{u}_n + P_r(u_n) = P(t_n) \]

- Mass matrix M is often singular -> second order differential equation infinitely stiff -> fully implicit numerical methods
- Make as few function calls as possible
- Use constant Jacobian in the numerical methods since tangent stiffness is not available
Direct Integration Methods

Explicit Integrators

- explicit Newmark Method
- Central-Difference Method
- explicit Alpha-Method
- generalized explicit Alpha-Method

Implicit Integrators

- Newmark Method
- Alpha-Method
- generalized Alpha-Method
- Collocation Method
Direct Integration Methods

- Implicit Integrators with sub-stepping (constant number)
  - Newmark-HybridSimulation Method
  - generalized Alpha-HybridSimulation Method
  - Collocation-HybridSimulation Method

- Predictor-Corrector Integrators
  - Alpha-OS Method
  - generalized Alpha-OS
Hybrid Simulation Procedure

Finite Element Analysis

Start the Direct Integration Analysis
analyze(numSteps, Δt);
for i=0 to i<numSteps do
  Calculate and set new Trial Response Quantities,
  Increment Loads and Time by Δt:
  theIntegrator -> newStep(Δt);

  Solve the current Time Step:
  theLinearAlg -> solveCurrentStep();

  Form Effective Tangent or Initial Stiffness A in A·x = b
  theIntegrator -> formTangent();

  Form the Unbalanced Force b in
  Ax = b
  theIntegrator -> formUnbalance();

  Solve A·x = b for x
  theSOE -> solve(

  Update the Response at t+Δt
  theIntegrator -> update(x);

  Commit the Domain:
  theIntegrator -> commit();

Analytical Substructure

Calculate Tangent or Initial Stiffness
getTangentStiff(); or
getInitialStiff();

Calculate Resisting Forces
getResistingForce();

Experimental Substructure

Return Initial Stiffness since Tangent Stiffness can not be obtained
getInitialStiff();

Command Actuators to impose Trial Displacements
execute(dsp, vel, acc);

Measure Forces at Target
acquire(dspDaq, frcDaq);
**OpenSees Components**

- **ModelBuilder**: constructs the objects in the model and adds them to the Domain.
- **Domain**: holds the state of the model at time $t_i$ and $t_i+dt$.
- **Analysis**: moves the model from state at time $t_i$ to state at time $t_i+dt$.
- **Recorder**: monitors user defined parameters in the model during the analysis.
OpenSees Components

Domain

Node
Element
MP_Constraint
SP_Constraint

ExperimentalElement
BeamColumn

OpenFresco

Control System in Laboratory

Experimental Element
Experimental Site
Experimental Setup
Experimental Control
Control System in Laboratory
OpenFresco Components

OpenSees
ExpElement
ExperimentalSetup
ExperimentalControl
Control System in Laboratory

RemoteExpSite
ExperimentalSetup
Client
TCP/IP (Channel)

ActorExpSite
ExperimentalSetup
ExperimentalControl
Control System in Laboratory

FE-Software
Experimental Element
Experimental Site
Experimental Setup
Experimental Control
Control System in Laboratory
Experimental Elements

1) EETruss (1D,2D,3D)

```
expElement truss $eleTag $iNode $jNode -site $siteTag -initStif $Kij <-iMod> <-rho $rho>
```

- **$eleTag**: unique element tag
- **$iNode, $jNode**: end nodes
- **$siteTag**: tag of previously defined site object
- **$Kij**: initial stiffness matrix elements (1 x 1)
- **-iMod**: flag for I-Modification (optional, default=false)
- **$rho**: mass per unit length (optional, default=0.0)

controlled displacements and acquired forces: $d_i, q_1$

Diagram:
- Nodes $i$ and $j$ connected by an element with controlled displacements $\Delta x$ and $\Delta y$.
Experimental Elements

2) EEBeamColumn (2D, 3D)

expElement beamColumn $eleTag $iNode $jNode $tranTag -
    site $siteTag -initStif $Kij  <-iMod> <-rho
    $rho>

$eleTag    unique element tag
$iNode,$jNode end nodes
$tranTag    tag of previously
            defined crd-transf object
$siteTag    tag of previously
            defined site object
$Kij        initial stiffness matrix
            elements (ndf x ndf)
-iMod       flag for I-Modification
            (optional, default=false)
$rho        mass per unit length
            (optional, default=0.0)

controlled displacements
and acquired forces

\[ \Delta x, \Delta y \]

\[ d_1, q_1 \]

\[ d_2, q_2 \]

\[ d_3, q_3 \]
Experimental Elements

3) EEZeroLength (1D,2D,3D)

The EEZeroLength element is used to represent zero-length elements in 1D, 2D, or 3D space. It takes the following form:

```expElement zeroLength $eleTag $iNode $jNode -dir $dirs -site $siteTag -initStif $Kij <-iMod> <-orient $x1 $x2 $x3 $y1 $y2 $y3> <-mass $m>
```

- `$eleTag`: unique element tag
- `$iNode`, `$jNode`: end nodes
- `$dirs`: force directions (1-3, 1-6)
- `$siteTag`: tag of previously defined site object
- `$Kij`: initial stiffness matrix elements (nDir x nDir)
- `<-iMod>`: flag for I-Modification (optional, default=false)
- `<-orient`: local x- and y-axis (optional, default=X,Y)
- `<-mass $m>`: mass (optional, default=0.0)

Controlled displacements and acquired forces are shown in the diagram.
Experimental Elements

4) EEInvertedVBrace (2D)

expElement invertedVBrace $eleTag $iNode $jNode $kNode -
site $siteTag -initStif $Kij <-iMod>   <-rho1 $rho1>
<-rho2 $rho2>

$eleTag    unique element tag
$iNode,$jNode  end nodes
$kNode
$siteTag    tag of previously
defined site object
$Kij    initial stiffness matrix
elements (ndf x ndf)
-iMod    flag for I-Modification
(optional, default=false)
$rho1,$rho2   masses per unit length
(optional, default=0.0)
Experimental Elements

5) EEGeneric (1D, 2D, 3D)

```
expElement generic $eleTag -node $Ndi $Ndj ...
dof $dofNdi -dof $dofNdj ... -site $siteTag -
initStif $Kij <-iMod> <-mass $Mij>
```

$eleTag: unique element tag
$Ndi, $Ndj, ...: n end nodes
$dofNdi, ...: dof for n end nodes
$siteTag: tag of previously defined site object
$Kij: initial stiffness matrix elements (ndfTot x ndfTot)
-iMod: flag for I-Modification (optional, default=false)
$Mij: mass matrix elements (optional, default=0.0)
Experimental Sites

1) LocalExpSite

```
expSite LocalSite $tag $setupTag
```

$tag unique site tag
$setupTag tag of previously defined setup object
2) RemoteExpSite

```plaintext
expSite RemoteSite $tag <-setup $setupTag> ipAddr $ipPort
```

$tag: unique site tag
$setupTag: tag of previously defined setup object (optional, provide if setup is on client side)
ipAddr: ip-address of actor site
$ipPort: ip-port of actor site
Experimental Sites

3) ActorExpSite

expSite ActorSite $tag -setup $setupTag $ipPort
expSite ActorSite $tag -control $ctrlTag $ipPort

$tag          unique site tag
$setupTag     tag of previously defined setup object if setup is on server side
(ctrlTag      tag of previously defined control object if setup is on client side
$ipPort       ip-port of actor site
Experimental Setups

1) ESNoTransformation

```
expSetup NoTransformation $tag <-control $ctrlTag> -dir $dirs ... <-ctrlDispFact $f> ...
```

- $tag: unique setup tag
- $ctrlTag: tag of previously defined control object
- $dirs: directions (1-6)
Experimental Setups

2) ESOOneActuator

\texttt{expSetup OneActuator \$tag <control \$ctrlTag> \$dir <-
  ctrlDispFact \$f> ...}

\$tag \hspace{1cm} \text{unique setup tag}
\$ctrlTag \hspace{1cm} \text{tag of previously defined control object}
\$dir \hspace{1cm} \text{direction (1-6)}
Experimental Setups

3) ESTwoActuators

expSetup TwoActuators $tag <-control $ctrlTag> $La0 $La1
$L <-nlGeom> <-posAct $pos> <-phiLocX $phi>
<-ctrlDispFact $f> ...

$tag unique setup tag
$ctrlTag tag of previously defined control object
$La0 length of actuator 0
$La1 length of actuator 1
$L length of rigid link
-nlGeom nonlinear geometry flag
$pos position of actuators (left, right)
$phi angle to local x-axis
Experimental Setups

4) ESThreeActuators

```
expSetup ThreeActuators $tag <-control $ctrlTag> $La0 $La1 $La2 $L0 $L1 <-nlGeom> <-posAct0 $pos> <-phiLocX $phi> <-ctrlDispFact $f> ...
```

- **$tag**: unique setup tag
- **$ctrlTag**: tag of previously defined control object
- **$La0**, **$La1**, **$La2**: length of actuator 0, 1, 2
- **$L0**, **$L1**: length of rigid link 0, 1
- **-nlGeom**: nonlinear geometry flag
- **$pos**: position of actuator 0 (left, right)
- **$phi**: angle to local x-axis
Experimental Setups

5) **ESI**nvertedVBraceJntOff

```plaintext
expSetup InvertedVBraceJntOff $tag <-contol $ctrlTag>
$nlGeomFlag $La0 $La1 $La2 $L0 $L1 $L2 $L3 $L4 <-
nlGeom> <-posAct0 $pos> ...
```

- **$tag**: unique setup tag
- **$ctrlTag**: tag of previously defined control object
- **$La0**: length of actuator 0
- **$La1**: length of actuator 1
- **$La2**: length of actuator 2
- **$L0**: length of rigid link 0
- **$L1**: length of rigid link 1
- **$L2**: length of rigid link 2
- **$L3**: length of rigid link 3
- **$L4**: length of rigid link 4
- **$L5**: length of rigid link 5 …
Experimental Setups

6) ESAggregator

```
expSetup Aggregator $tag <-contol $ctrlTag> -setup $setupTags ...
  -sizeTrialOut $sizei ...
  -sizeTrialOut $sizej ...
  <-ctrl1DispFact $f> ...
```

- **$tag**: unique setup tag
- **$ctrlTag**: tag of previously defined control object
- **$setupTags**: tags of setups to be aggregated
- **$size**: sizes of trial and output vectors of each setup (10 values: [disp, vel, accel, force, time]) for trial and output
Experimental Controls

1) ECdSpace

```
expControl  dSpace  $tag  $type  boardName
```

$tag          unique control tag
$type          predictor-corrector type
boardName     name of dSpace board
Experimental Controls

2) ECxPCtarget

```
expControl xPCtarget $tag $type ipAddr $ipPort appName appPath
```

- `$tag` unique control tag
- `$type` predictor-corrector type
- `ipAddr` IP address of xPC Target
- `$ipPort` IP port of xPC Target
- `appName` name of Simulink application to be loaded
- `appPath` path to Simulink application
Experimental Controls

3) ECScramNet

expControl SCRAMNet $tag $memOffset $numActCh

$tag
unique control tag

$memOffset
SCRAMNet memory offset in bytes

$numActCh
number of actuator channels in control system
Experimental Controls

4) ECNI Eseries

expControl NIEseries $tag $device

$tag unique control tag
$device id of device
Experimental Controls

5) ECMtsCsi

`expControl MTSCsi $tag cfgFile <$rampTime>`

$tag
unique control tag

cfgFile
CSI configuration file name

$rampTime
time to ramp up actuator commands to requested target values (optional)
Experimental Controls

6) ECLabVIEW

expControl LabVIEW $tag ipAddr <$ipPort>
  trialCP $cpTags ... -outCP $cpTags ...

$tag  unique control tag
ipAddr  IP address of controller
$ipPort  IP port of controller
  (optional)
$cpTags  tags of previously defined control point objects used for control
$cpTags  tags of previously defined control point objects used for data acquisition
Experimental Controls

7) ECSimUniaxialMaterials

```
expControl SimUniaxialMaterials $tag $matTags ...
```

$tag  
unique control tag

$matTags  
tags of previously defined uniaxial material objects
Event-Driven Approaches

Three-loop-architecture

by Prof. G. Mosqueda
Event-Driven Approaches

- Predictor-corrector algorithms

![Graph](image-url)
Event-Driven Approaches

**Predictor-corrector algorithm**

![Diagram showing event-driven approaches with signal generation, integration, and prediction stages.]

**One single DSP**

**Two machines**
Event-Driven Approaches

Improved event-driven-controller

```
HybridController

Initialize
en: i = 1; isD = 0;
en: diSD = 1;
en: initData(hAct, dtCon);
en: zeroDsp(com);
[flag==1]

Correct
en: state = 0;
en: setCurDsp(com, iN);
en: dspLocal = dsp; setNewDsp(dspLocal);
en: di = min(max(dISD+1/(0.2*N)*(i-iSD), 0.001), 1.0);
en: i = min(i+di, N);
en: correctP3(com, iN);
du: di = min(max(dISD+1/(0.2*N)*(i-iSD), 0.001), 1.0);
du: i = min(i+di, N);
du: correctP3(com, iN);

Predict
en: state = 1;
en: diSD = 1;
en: i = 1; isD = 0;
en: predictP3(com, iN);
du: i++;
du: predictP3(com, iN);

AutoSlowDown
en: state = 2;
en: diSD = 4-i/(0.2*N);
en: i += diSD; isD = i;
en: predictP3(com, iN);
du: diSD = 4-i/(0.2*N);
du: i += diSD; isD = i;
du: predictP3(com, iN);
```

[>=N]

[flag==1]

[>=0.6*N & & flag==1]
Event-Driven Approaches

Predictor-corrector algorithms

\[ f(x) \]

Prediction

Correction

\[ f_0 = f(x) \]

\[ dw \]

\[ x_p = 0.5 \]

\[ x = 0.75 \]
Event-Driven Approaches

Counter behavior for smooth velocity transitions: $N = 16$

Counter behavior for smooth velocity transitions: $N = 256$
MATLAB® One Bay Frame Example

Model

- Element Types
- Damping
- Time Integration
- TCPSocket.mex

Ground Motion - 1940 El Centro
MATLAB® - Local Test

FE-Software
ExpElement

Client

TCP/IP (Socket)

Middle-Tier-Server

SimAppSiteServer
LocalExperimentalSite
ExperimentalSetup
ExperimentalControl

Control System in Laboratory

MATLAB®

xPC Target
MicroNEES Results

Displacement [in.]

Time [sec]

Base Shear [kips]

Deformation [in.]
FE-Software: UI-Simcor 2.6

OpenFresco 2.0

m = 10 N/(mm/sec²)

3,500 mm
OpenFresco and MiniMost
SixDOFModel
Conclusions

- Environment-independent framework for development and deployment will boost the use of hybrid simulation (on-site and tele-operation)

- Modularity and transparency of the framework permits existing components to be modified and new components to be added without much dependence on other objects.
  - Speed development of refined hybrid simulation procedures
Conclusions

Large library of hybrid simulation direct integration methods, experimental elements, experimental setups, controller models, and event-driven solution strategies will be available to the user to choose or adapt from.

Need:

- User feedback on refinements and new features
- Developer contributions to extend libraries
- NEESit assistance in streamlining network communications
OpenFresco on NEESforge

Welcome to OpenFresco project!
Local and distributed hybrid simulations enabled through an object oriented software framework for experimentation and control.

OpenFresco 2.0 & Manual have been released
Andreas Schellenberg - 2007-01-03 08:40
(0 Comment) [Read More/Comment]

Manual for OpenFresco 1.0-beta
Andreas Schellenberg - 2006-11-30 08:24
(0 Comment) [Read More/Comment]

OpenFresco 1.0-beta Release
Andreas Schellenberg - 2006-10-02 01:58
(0 Comment) [Read More/Comment]

[News archive]

Manual
Subversion Repository
Downloads

Project Summary

Tracker
- Bugs (1 open / 1 total)
  Bug Tracking System
- Support (0 open / 0 total)
  Tech Support Tracking System
- Patches (0 open / 0 total)
  Patch Tracking System
- Feature Requests (0 open / 0 total)
  Feature Request Tracking System

Forums (2 messages in 2 forums)

Doc Manager
Mailing Lists (1 public lists)
Task Manager
  - To Do
  - Next Release
Surveys (0 surveys)
SCM Tree (57 commits, 116 adds)
Released Files
### OpenFresco on NEESforge

Below is a list of all files of the project. Before downloading, you may want to read Release Notes and ChangeLog (accessible by clicking on release version).

<table>
<thead>
<tr>
<th>Package</th>
<th>Release &amp; Notes</th>
<th>Filename</th>
<th>Date</th>
<th>Size</th>
<th>D/L</th>
<th>Arch</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Examples</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.0</td>
<td>EXAMPLES.zip</td>
<td></td>
<td>2006-12-19 19:12</td>
<td>474 KB</td>
<td>94 KB</td>
<td>.zip</td>
<td></td>
</tr>
<tr>
<td>1.0-beta</td>
<td>ThreeSpanBridge_Tcl.zip</td>
<td></td>
<td>2006-10-02 12:24</td>
<td>59 KB</td>
<td>26 Any</td>
<td>.zip</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TwoDOFModel_Tcl.zip</td>
<td></td>
<td></td>
<td>13 KB</td>
<td>20 Any</td>
<td>.zip</td>
<td></td>
</tr>
<tr>
<td><strong>OpenFresco-sourceCode</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.0</td>
<td>OpenFresco2.0.zip</td>
<td></td>
<td>2006-12-19 19:10</td>
<td>2.38 MB</td>
<td>74 KB</td>
<td>.zip</td>
<td></td>
</tr>
<tr>
<td>1.0-beta</td>
<td>OpenFresco1.0-beta.zip</td>
<td></td>
<td>2006-10-03 18:56</td>
<td>6.06 MB</td>
<td>29 Any</td>
<td>.zip</td>
<td></td>
</tr>
<tr>
<td><strong>OpenFresco-win32</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.0</td>
<td>OpenFresco2.0_win32.zip</td>
<td></td>
<td>2006-12-19 17:25</td>
<td>2.62 MB</td>
<td>59 KB</td>
<td>.zip</td>
<td></td>
</tr>
<tr>
<td>1.0-beta</td>
<td>OpenFresco1.0-beta_win32.zip</td>
<td></td>
<td>2006-10-02 22:24</td>
<td>1.07 MB</td>
<td>28 KB</td>
<td>.zip</td>
<td></td>
</tr>
<tr>
<td><strong>OpenSees-win32</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.7.3</td>
<td>OpenSees1.7.3_win32.zip</td>
<td></td>
<td>2006-10-02 22:32</td>
<td>1.66 MB</td>
<td>47 KB</td>
<td>.zip</td>
<td></td>
</tr>
</tbody>
</table>

**Project totals**

7 8 14.32 MB 377
Thank you!

Development and operation of the nees@berkeley Equipment Site is sponsored by NSF George E. Brown Jr. NEES grants.

http://nees.berkeley.edu
http://openfresco.neesforge.nees.org