4ICEGE Thessaloniki-GREECE 2007

4th International Conference on
Earthquake Geotechnical Engineering
June 25-28, 2007
Paper No. 1341

A 3-D VISUALIZATION SYSTEM FOR LARGE-SCALE EXPERIMENTAL GEOTECHNICAL EARTHQUAKE DATABASES

Jorge MENESES [1], Masayoshi SATO[2] and Akio ABE[3]

ABSTRACT

A robust visualization system is developed to view, interpret and analyze large experimental databases from experiments of geotechnical systems using earthquake simulators. The visualization system offers time-based presentations of the sensor data in both two-dimensional (2D) and three-dimensional (3D) graphical representations. Visualization of the experimental information includes graphs, plots, scaled real-world representative 3D models, and time-based 2D/3D animations. Through animations, interaction and manipulation of images, the visualization system enables an analyst to understand relations between measured quantities in space- and time-domains that are difficult to understand by using other means, especially in a large scale experiment environment where hundreds of sensors are deployed. The visualization system can handle up to one thousand sensors and accommodations can be made to handle more as required. Capabilities and features are described in applying the visualization system to large-scale experiments studying the seismic interaction of foundation piles and liquefied soil.

Keywords: visualization, large-scale earthquake experiment, visual analysis, ground earthquake behavior

INTRODUCTION

Management, viewing, interpretation, and correlation of large experimental databases from large-scale earthquake geotechnical experiments can be accomplished by a visualization system (Abe et al. 2004a, Abe et al. 2004b, Meneses and Morón 2004). The visualization system should give an analyst the ability to interact with visual information and visualization environment parameters in order to establish the desired analysis context and perspective. Through animations, interaction and manipulation of images, the visualization system should enable an analyst understand relations between measured quantities in space- and time-domains that are difficult to understand by using other means. This is true especially in a large-scale experiment environment where hundreds of sensors are deployed. *QuakeVis* is a new visualization system that provides an analyst the resources and tools to unveil all retrieved apparent and even concealed information from large experimental databases; i.e., large-scale experiments of seismic interaction of foundation piles and liquefied soil.

QuakeVis uses Trolltech's Qt development environment for the development of the Graphical User Interface (GUI). Qt provides a professional finish and higher level functionality for common tools needed in GUIs. Qt is cross platform compatible, allowing for development on the developer's platform of choice and deployment on the user's platform of choice, without compromising functionality. *QuakeVis* uses OpenSceneGraph (OSG) as the platform for 3D development work. OpenSceneGraph is built upon the industry standard OpenGL, which renders primitives (points, lines,

[1] Staff Professional, Kleinfelder Inc., San Diego, California, USA, Email: jmeneses@kleinfelder.com
[2] Deputy Director, Hyogo Earthquake Engineering Research Center, NIED, Miki, Japan.
[3] Tokyo Soil Research Co. Ltd., Tsukuba, Japan

triangles, polygons), providing spatial object abstraction for higher level management and optimization of a 3D scene. The configuration file for *QuakeVis* is based on an XML (E**X**tensible **M**arkup **L**anguage) format to allow for ingestion of the experimental database by the program. XML is a self-describing file format easy to create and edit. The metadata file provides the interface between the raw data, exported to .csv files from Microsoft Excel spreadsheet format, and *QuakeVis*, putting sensor and sensor array information into an arrangement that allows organized user interaction with the experimental data. Development of *QuakeVis* is in C++, native to both Qt and OSG. Standard C++ is adhered to and deviation to non-standard, platform specific functionality is strictly avoided. The code, specifically in OSG, makes wide use of the Standard Template Library (STL), which requires RTTI (Run Time Type Identification) to be enabled on Windows based development platforms. Build environments on both Qmake and VisualStudio solution files are provided in the source code. *QuakeVis* can work under Linux or Windows systems.

QuakeVis is being developed to respond to the need to visualize experimental geotechnical databases produced by the world's largest earthquake simulator located in Miki City, Japan. An experimental setting typically includes a 6-degree-of-freedom shake table (earthquake simulator), a container (encloses the ground model and could be composed by rigid walls or by a stack of laminates that shakes with minimum interference with ground response), structural specimens (i.e., piles, pile cap, columns founded on the ground model), ground model and sensors. This paper describes some capabilities, functionalities and features of *QuakeVis* during its application to two large-scale experimental setups.

METADATA FILE

A large-scale experimental geotechnical earthquake database may typically contain huge amounts of numerical information obtained from large number of sensors, and this volume of information can even increase depending on earthquake duration and sampling frequency. An efficient data management system should be implemented to handle hundreds of time histories retrieved from accelerometers, pore water pressure transducers, strain gauges, and displacement transducers, which are sensors typically found in a geotechnical earthquake experiment. This section describes the file that *QuakeVis* uses for storing the metadata (data about data) or information about all experimental components and features.

The XML Format

The *QuakeVis* metadata file is an XML document, following some very simple XML formatting rules. XML is a set of rules for creating a markup language. It is intended for documents that are *self-describing*, by placing *tags* around content. *Tags* have the form <*identifier*>Content</*identifier*>. *Identifier* describes what **Content** is. The tag that begins with the character **'/'**, denotes the *end* tag. For example, this is a tag: <message>Hello *QuakeVis*!!</message>. <message> denotes the *start* tag, </message> denotes the *end* tag and **Hello *QuakeVis*!!** is the content. In essence we have described something called a *message*, where it starts and where it ends. Tags may contain other tags, arranging information in a hierarchical form much like the outline of a document, speech, or essay.
Tags may also have attributes. These attributes are specified as *name/value pairs* and are specified inside the **< >** delimiters following the identifier of the tag. For example,

```
<Text color="blue"
    position="-5.5,-3.0,-1.0"> accelerometer</Text>
```

In this example, the tag with the identifier *Text* has attributes *color* and *position*. *Color* and *position* are the *names* of the attributes, followed by an "=" sign, and then the *value* of the attributes contained within double quotes. Every *QuakeVis* metadata file should have at the top the following:

<?xml version="1.0" encoding="UTF-8"?>

This simply identifies the document to the XML parser used internally by *QuakeVis*. That is all the user should need to understand about XML in order to successfully create, edit and modify *QuakeVis* metadata files.

Metadata File Organization

The following describes the metadata file organization by hierarchy of XML tags. The role of each tag is described, including any attribute definitions within the tag. The "children" tags are then noted below by indented text.

Experiment

This is the top level of the organizational tree of each of the experiments ingested by *QuakeVis*. It contains all other entities and has no attributes.

- Name. Name of the Experiment. The content of this tag is used to identify the current experiment and for spurious labeling within the *QuakeVis* user interface.
- Description. The content of this tag provides a brief description of the current experiment.
- Annotation. The content of this tag describes a group of text entities that are used to place visible text in the 3D view. The **Annotation** tag can have the attributes *defaultSize* and *defaultFont* defined, which are then applied to the **Text** entities within the **Annotation** definition. *defaultFont* may be one of the following True Type fonts: "arial.ttf," "dirtydoz.ttf," "fudd.ttf," or "Times New Roman.ttf"
- Text. This tag defines the actual text of the **Annotation**. Each **Text** tag may have the attributes, *color*, *font*, *size*, and/or *position* defined. The content of the **Text** tag is the text that is displayed in the 3D view.

Container

This encapsulates the attributes that describe the container of an experiment. The container can be a laminar or rigid wall container. It has no attributes, but contains the following child entities:

- Alpha. The level of transparency of the container. Content is a real number with range 0.0 to 1.0.
- Color. The color of one laminate of the container. There may be more than one **Color** tag within a **Container** definition, defining alternating colors for each laminate to provide for visual distinction. **Color** is defined here with a triple of three real numbers ranging from 0.0 to 1.0, and represents a scaling of the red, green and blue components respectively.
- Height. The **height** of the container.
- LaminateThickness. The thickness of each laminate. This assumes a uniform thickness of all laminates. If the container is a rigid wall container, one laminate with a thickness the same as the height of the container can be specified.
- Length. The **length** of the container.
- Shape. The designated **shape** of the container. Maybe one of *Box* or *Cylinder*.
- Width. The **width** of the container.

Model

A 3D model contained in an external file. The file format should be supported by OpenSceneGraph.

DisplacementGroup

A DisplacementGroup couples models or subparts of a 3D model with sensor arrays for the purpose of doing real-time articulations of the geometry.

- Model. In this context (subject to *DisplacementGroup*), this refers to the geometry that will be articulated by the sensor array by name. The 3D model should have already been specified with the **Model** tag. The content of this tag can take the form of either just the 3D model file name, e.g.: walls.osg, or as a combination of *filename:component*, i.e.: walls.osg:eastWall, where the *component* is a named node in the 3D model graph.

- SensorArray. In this context (subject to *DisplacementGroup*), **SensorArray** refers to the sensor array that will be effecting the articulations on the model or submodel in the DisplacementGroup.

DefaultSensorSize
Without this tag, *QuakeVis* will assign a default size of 1.0 to all sensors. This may be completely out of proportion with the rest of the experiment in terms of scale. If, for example, the experiment is using millimeters, the sensor will be very small. If it is using meters, it will be very large. A value should be designated here to provide for a size for the sensors that is appropriate for the scale of the experiment.

SensorGroup
It is a group of sensors with a specific designation or category. Examples of sensor categories are: X-Acceleration, Y-Acceleration, Z-Acceleration, X-Displacement, Y-Displacement, Z-Displacement, X-Strain, Y-Strain, Z-Strain, Pile curvature, Soil pressure, and Effective stress, and Pore Water Pressure.

SensorType
A designation of the type of sensors contained in this sensor group. A label commensurate with the above categories is appropriate. The following syntax for the categories is required: X-acceleration, Y-acceleration, Z-acceleration, X-displacement, Y-displacement, Z-displacement, X-strain, Y-strain, Z-strain, pile curvature, soil pressure, effective stress, and pore water pressure.

Units
The units of the sensors in this *SensorGroup*. These may be *m* (meters), *mm* (millimeters), m/sec^2 (meters per second squared), or kPa (kilo Pascals) for example.

SensorArray
This is the array of sensors within a *SensorGroup*. *SensorArrays* are normally a set of sensors that are closely related to models or areas of the test volume within the 3D scene and consist of a set of *Sensors* measuring the area or model of interest at their position in 3D space.
- ArrayName. The name of the **SensorArray**.
- FileName. The name of the file containing the data captured by the **Sensors** in this **SensorArray**. This file is in *csv* format.
- Sensor. This entity describes an individual sensor in the **SensorArray**.
- SensorName. This is the name of the **Sensor**.
- Position. This is the position of the **Sensor** in 3D space.

GRAPHICAL USER INTERFACE

Figure 1 shows a typical experimental setting to study the seismic interaction between a 2-by-3 pile group, and a liquefiable sand deposit enclosed by a 6m-high laminar container. This container sits on a shake table or earthquake simulator. The small spheres and cubes represent the location of different sensors, which are arranged in vertical arrays. It is apparent the higher density of sensors especially pore water pressure transducers around the piles to closely monitor the development of effective stresses. Figure 1 shows the standard default view of *QuakeVis*. The top contains a Menu Bar with several menu choices. Below the menu bar is the Tool Bar with convenient access to tasks that are performed frequently. Below the toolbar is an area with controls for the data animation, visualization and interaction. And finally below the Data Controls, the viewing area.

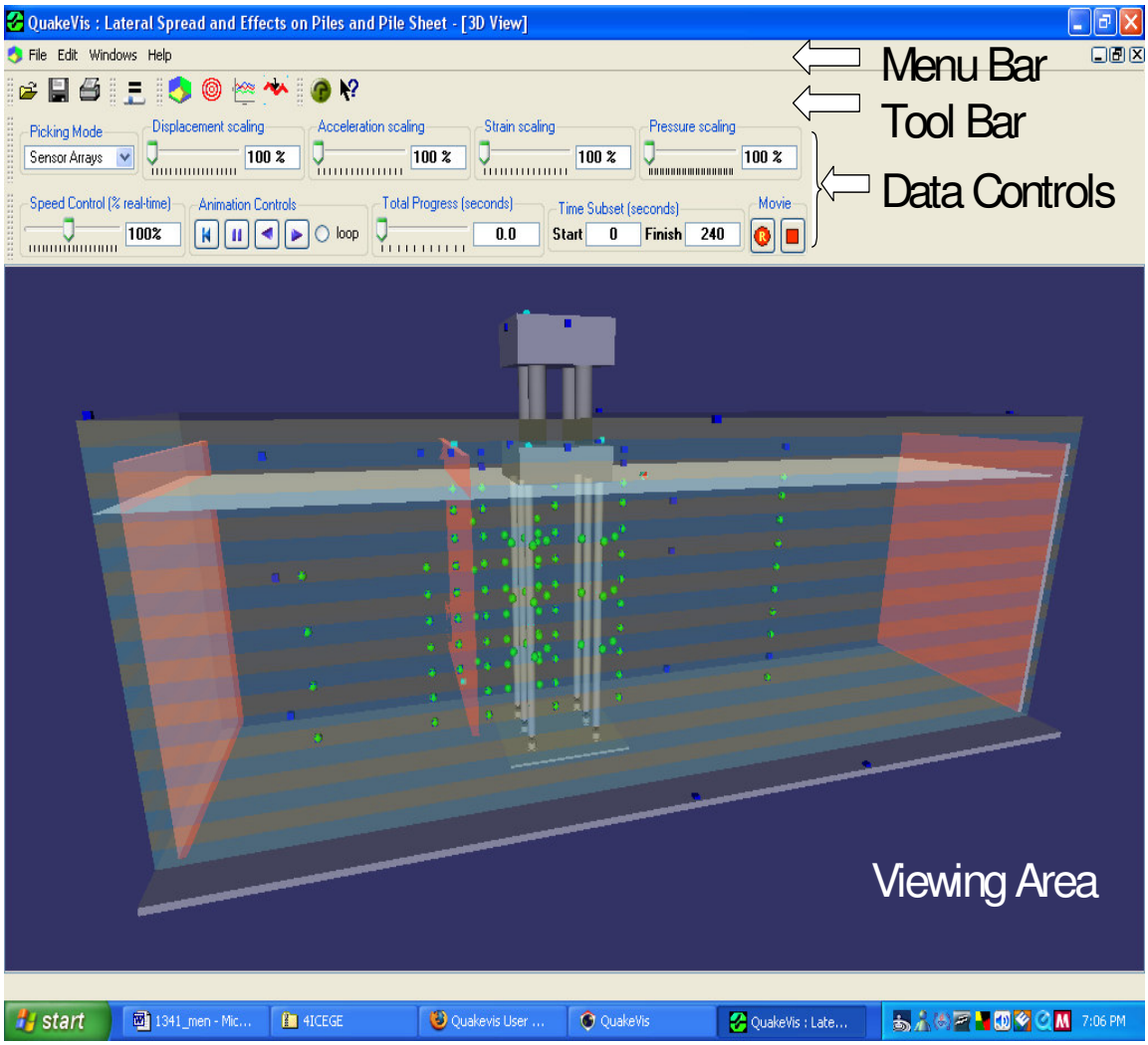

Figure 1. *QuakeVis* Standard Default View

Data Controls

Figure 2 shows the control mechanisms of the **Data Controls** panel. *QuakeVis* allows data from an *experiment* be visualized in a time-based manner. These controls allow the user to select the parameters that allow best visual interaction and understanding of the data.

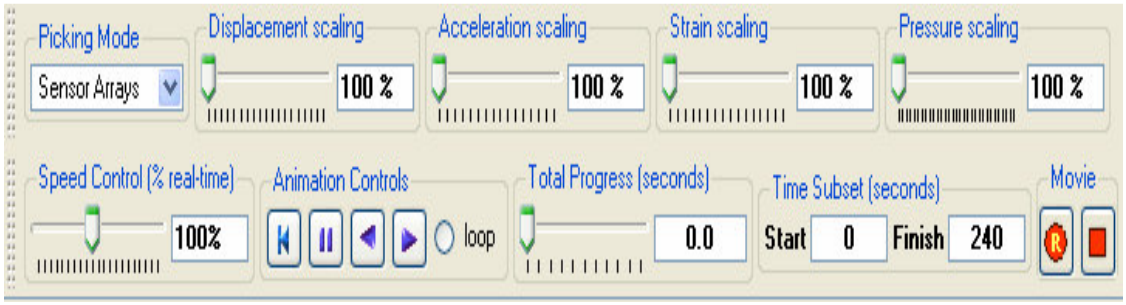

Figure 2. Data Controls Panel

Picking Mode

In the 3D View window, visible sensors may be selected by placing the mouse pointer over them and clicking the left mouse button. The different modes for sensor selection include:

- Sensor Groups. Selecting **Sensor Groups** will allow the user to select an entire group of sensors. For example, selecting on one *pressure transducer* will select all *pressure transducers*.
- Sensor Arrays. Selecting **Sensor Arrays** allows the user to select one array of sensors by picking any one of them.
- Sensors. Selecting **Sensors** sensor selection will be limited to the single sensor selected by the user.

Scaling Controls
The **Scaling Controls** allow the user control over scaling of the values displayed in the 3D view when the data from an *experiment* is played through.
- **Displacement scaling** will exaggerate the movement of sensors and models by the percentage selected by the user.
- **Acceleration scaling** will exaggerate the values of the data gathered at the accelerometers.
- **Strain scaling** will exaggerate the values of the data gathered at the strain gauges.
- **Pressure scaling** will exaggerate the values of the data gathered at the pressure transducers.

Animation Controls
The **Animation Controls** provide the user with control over the way in which the **experiment** data is played through. "VCR" type buttons provide the user with data play control. The user can **Rewind**, **Pause**, **Play backward**, and **Play forward. Speed Control** allows the user to play the data at varying rates. Setting the value to 100% would play the data in real-time. Setting it below or above 100% plays the data by the percentage set of real-time. The **Total Progress** control both reports the progress of the play through the **experiment** data while it is being played, and allows the user to drag the progress bar to the time position in the data that is of interest. The **Time Subset** control allows the user to define the **Start** and **Finish** time of the subset of data that is of interest. For example, setting **Start** to 2 and **Finish** to 6 would display the data between seconds *2* and *6*. This also provides finer control for the dragger in the **Total Progress** control. **Movie** creates an .avi movie file of the 3D visualization for a selected time frame.

Viewing Area
The viewing area accommodates several types of windows. One of these is the **3D View**, of which there is only one. Multiple **Contour Slice View** windows may also be opened in the **Viewing Area**, as well as **2D Chart View** windows. All of these are stacked and arranged by use of the **Window** menu.

TWO-DIMENSIONAL DATA GRAPHING AND PLOTTING

Figure 3 shows the time histories recorded by one particular vertical array of pore water pressure transducers. Simultaneous animated views of time histories allow a better understanding of the temporal and spatial distribution, and vertical propagation of effective stresses. If a particular vertical array of accelerometers is selected, occurrence of amplification or de-amplification of seismic accelerations can be easily identified. Figure 3 shows the default view of the **2D Chart View** window. The 2D Chart view provides two-dimensional (X, Y) graph analysis of the sampled data with time-history plotting. There are tabs for each of the option charts to visualize the experimental data; i.e., **Box Whisker**, **Array History**, **Value/Depth**, and **p-y Curves** (Reese et al. 1974, Reese 2006). Each chart will plot the data available from the different data sources available for the **experiment**. The data source is selectable in two different manners: by either picking a **Sensor**, **Sensor Group**, or **Sensor Array** from the 3D View with the left mouse button, or a combination of the left mouse button and shift keys; or by selecting from the data source selection menu at the right. The **Box Whisker** (Tukey 1977) is a histogram-like method of displaying the experimental data of a particular Sensor Array. The **Array History** displays in one chart all time histories of the selected sensor array. The **Value/Depth** chart displays the variation with depth of the time histories of the sensors of the

selected array. The **p-y Curves** displays animated p-y curves at different depths for the selected pile. By clicking on the View button, a window pops-up with an enlarged chart of the selected time history. This time history can be edited and the Fourier spectrum is automatically calculated.

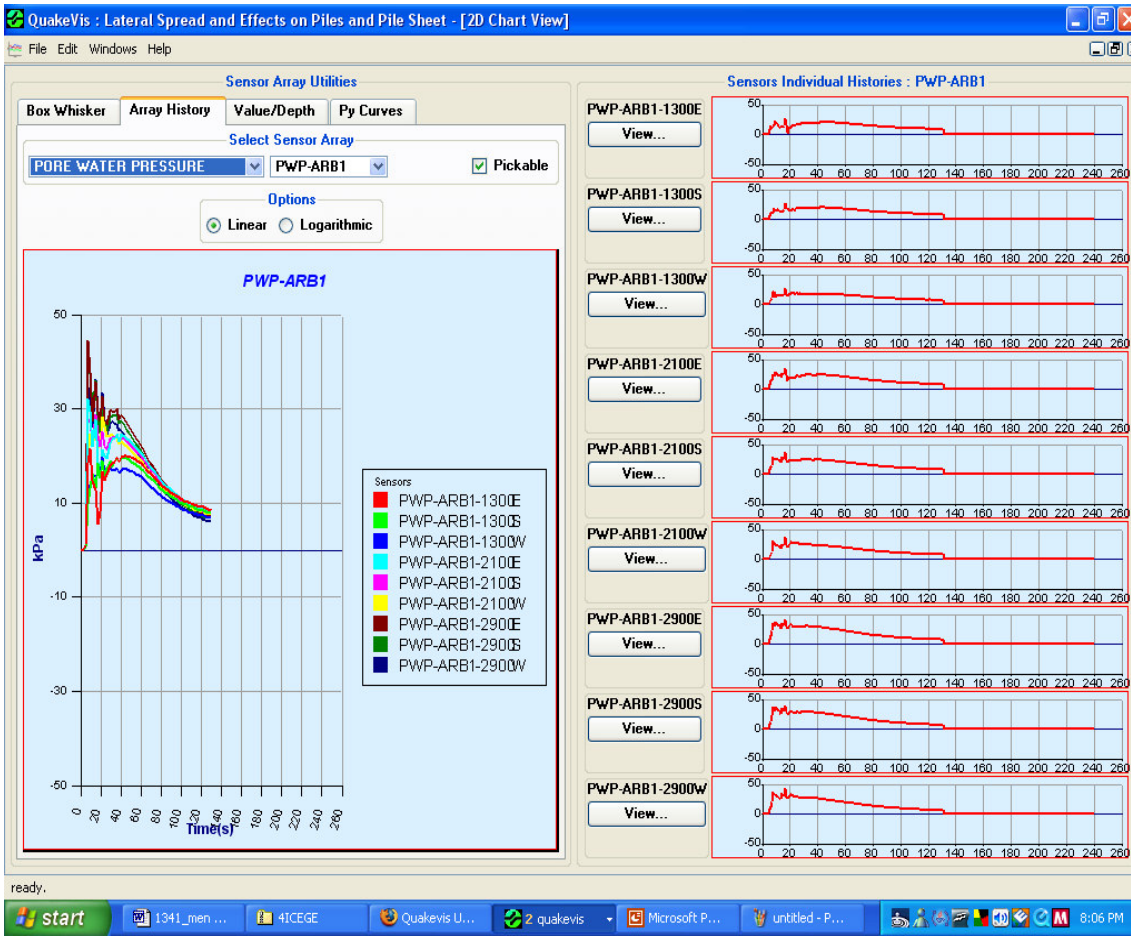

Figure 3. Two-Dimensional Chart View Window

THREE-DIMENSIONAL VISUALIZATION

Figure 4 provides a 3D view during seismic shaking of the experiment space containing the shake table, the laminar container, the 2-by-3-pile group and the sensors. The **3D View** is highly interactive allowing the user to move the view's position in 3-space by manipulating it with the mouse buttons and mouse motions. It also allows the user to select items in the scene, such as sensors and contour slices.

Navigation in 3D View

The 3D View allows the user to control the “camera” around the view of the model by rotating the model around a point, translating (or "panning") the view on a vertical plane, or by zooming in and out of the view. To **rotate** the model, press the left mouse button with the pointer placed anywhere within the 3D View window. By moving the mouse back and forth, the model may be rotated around the vertical axis. By moving the mouse up and down, the model may be moved around a horizontal axis. In fact, the mouse movements will manipulate the model as a trackball rotating it around nearly any axis, depending on the direction of movement of the mouse. To **translate (or "pan")** the model, click the middle mouse button or scroll wheel and move the mouse up, down, left or right. This will

translate the model up, down, left or right along a vertical plane orthogonal to the eye point. To **zoom** the view in and out of the model, press the right mouse button with the pointer placed anywhere in the 3D View window. By moving the mouse down (like pulling the model toward you), the eye point will zoom in to the model. Likewise, by moving the mouse up (like pushing the model away from you) the view will move away from the model.

Note that any of these modes may be used to *throw* the model. For example if you click the left mouse button, rotate, and while rotating, release the left mouse button, the model will continue to rotate freely until you click on the window again. Anytime it is desired to recenter the view, simply press the **<SpaceBar>** of the keyboard. All rotations, translations and zooming will stop and the model will be centered in the view.

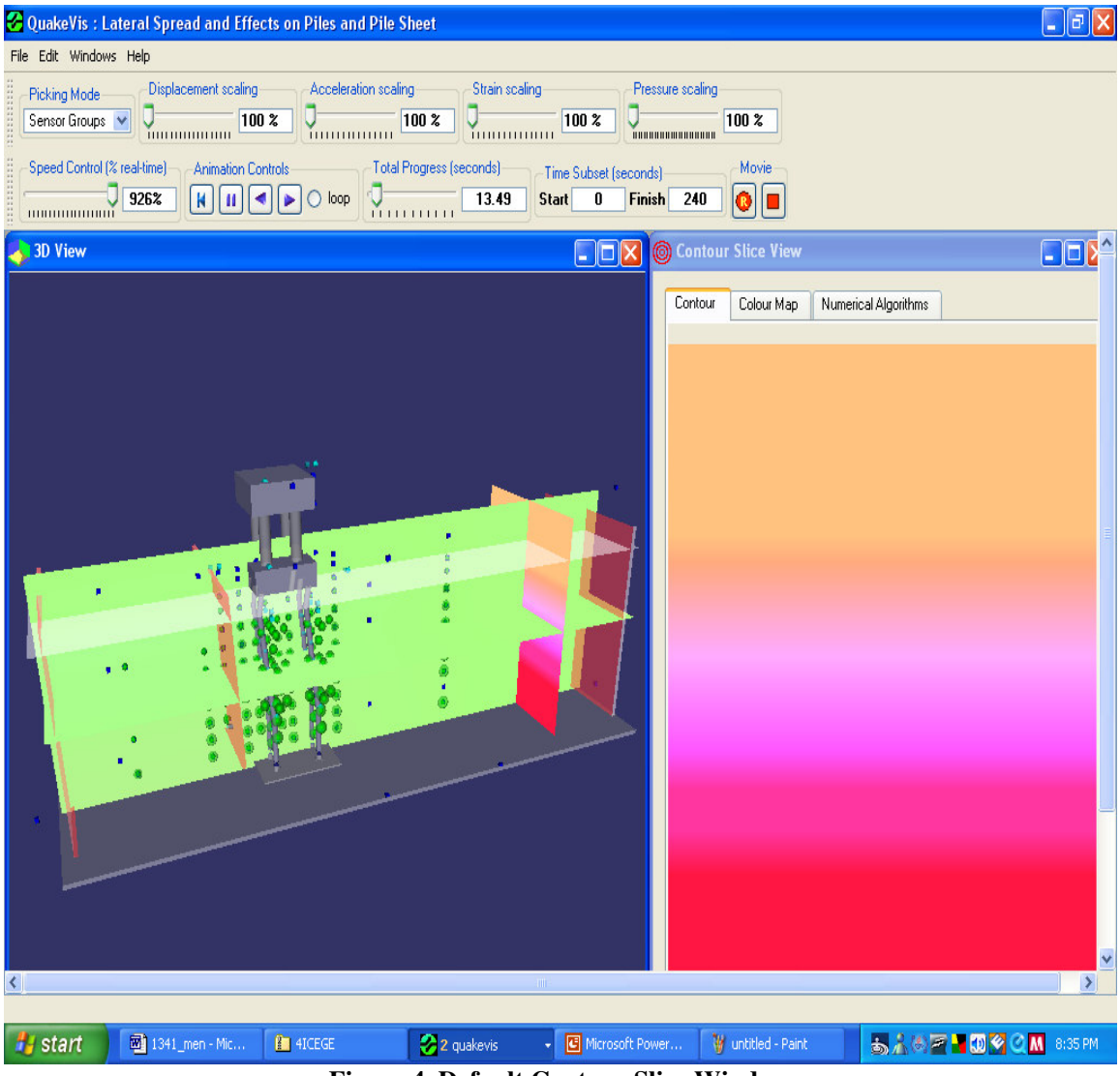

Figure 4. Default Contour Slice Window

Picking in 3D View

Items in the scene may be picked, by placing the mouse pointer over them and clicking the left mouse button. Depending on the **Picking Mode**, either a single **Sensor**, a **Sensor Array**, or a **Sensor Group** may be selected. Further, one can select multiple items by using the **<Shift>** key in combination with left mouse button selection. For example, if the **Picking Mode** is set to **Sensors** a single sensor can be

selected by clicking on it. Two sensors can then be selected by holding the **<Shift>** key and selecting the second one, and so on.

Contour Slice Views

Figure 4 shows the default view of a **Contour Slice View** window. A contour slice is a plane intersecting the data volume displayed in the 3D View. The plane contains colors representing the values at that 2D slice of the data volume. Note that each **Contour Slice View** window inserts another contour slice in the 3D View. The user may bring up multiple **Contour Slice View** windows, and in fact, it is useful to have multiple windows up allowing for views of multiple contour slices (Figure 5). Considering the Cartesian coordinate orientation of the data, where the **X** axis runs the length, **Y** runs the width, and **Z** runs the height of a container, slices may exist on the **XY** plane, **XZ** plane, or the **YZ** plane. The user can pick the plane of the contour slice by choosing from the pull down menu choices shown at the left. As an example, Figure 5 shows a snapshot of three **Contour Slice View** windows for Pore Water Pressures on XY, XZ and YZ planes. One of the windows shows the intervals of the values of Pore Water Pressures and the colors assigned to each of the intervals. During the animation of the entire duration of the earthquake shaking, the user can choose a specific time for the snapshot and visualize the spatial distribution of the pore water pressures inside the entire container.

To move the contour slice throughout the volume of data in the 3D view, select the dragger depicted to the right and move it to the desired location in the data volume. Data values will be represented by the different colors interpolated across the face of the contour slice. The contour slice can be made to visualize the different data as collected by the various sensors. By selecting a choice from the menu shown at the bottom left, in the **Contour Slice View** window, one can choose which set of data is visualized in the contour slice. By selecting the **Color Map** tab on the top of the **Contour Slice View** window, you can control the settings for the visual representation of the contour slices. First, select the range of values for the selected sensor type by entering the **Minimum value** and the **Maximum value**. Next, select the number of colors representing each value interval by setting **Number of intervals**. Note that the contour slices are dynamically updated when the data is being animated (Figure 5, top right window). This allows you to navigate through the data with the **Animation Controls** or the **Total Progress** controls and see the contour slices update in real-time.

PREPARING AND RUNNING AN EXPERIMENT

As of this version of *QuakeVis*, the experimental data is captured and placed into Microsoft ™ Excel format. For use with *QuakeVis*, this data must be converted to tab delimited csv. The user should name files with the same names as in the **FileName** tag in the metadata file under **SensorArray** category. Save the files in a folder set aside for this experiment, or a sub-folder named "Data". Each data file has data for one sensor array. Internal to this data file, the data for each of the sensors in the arrays is arranged by columns, with the left-most column indicating time. The first row in the data file contains the labels for each of the columns and each column is comma (**,**) delimited. Each row in the data file is end-of-line delimited. This data file will be associated with a sensor array specification in the metadata file. A correspondence of sensor labels in the metadata file, along with their positions, and other information will be associated with the labels in this file. It is important, therefore, that the labels be formatted correctly. All characters, spaces and digits between the delimiters in the .cvs file are taken into consideration.

Preparing the 3D Models

The 3DView of *QuakeVis* imports 3D models from various popular file formats. These models are typically created with a modeling or CAD tool. In the case of OpenSceneGraph, they can also be created programmatically. Using the 3D modeler of choice, the 3D model representation of each of the visual models in the experiment can be created. High tessellation for dynamic models may be used. The user can divide dynamic models up into components, or label components that correlate

with the sensor array that will articulate them. Then the user can save the files in a folder set aside for this experiment, or a sub-folder named "Models". There are a wide variety of modeling tools available for creating 3D models. These tools then save the models to a file format on disk. While the choice of tools is entirely the user's preference, it is important that the resulting model be in a format supported by OpenSceneGraph. Some formats supported by OpenSceneGraph include OpenSceneGraph Native, AC3D, DirectX, Design Workshop, DXF, OpenFlight, GEO, 3Dstudio, Lightwave Object/Lightwave Scene, Milkshake3D, Wavefront OBJ, and Performer binary. There is a long list of other 3D modelers as well, which export files to some of the formats listed in the table above.

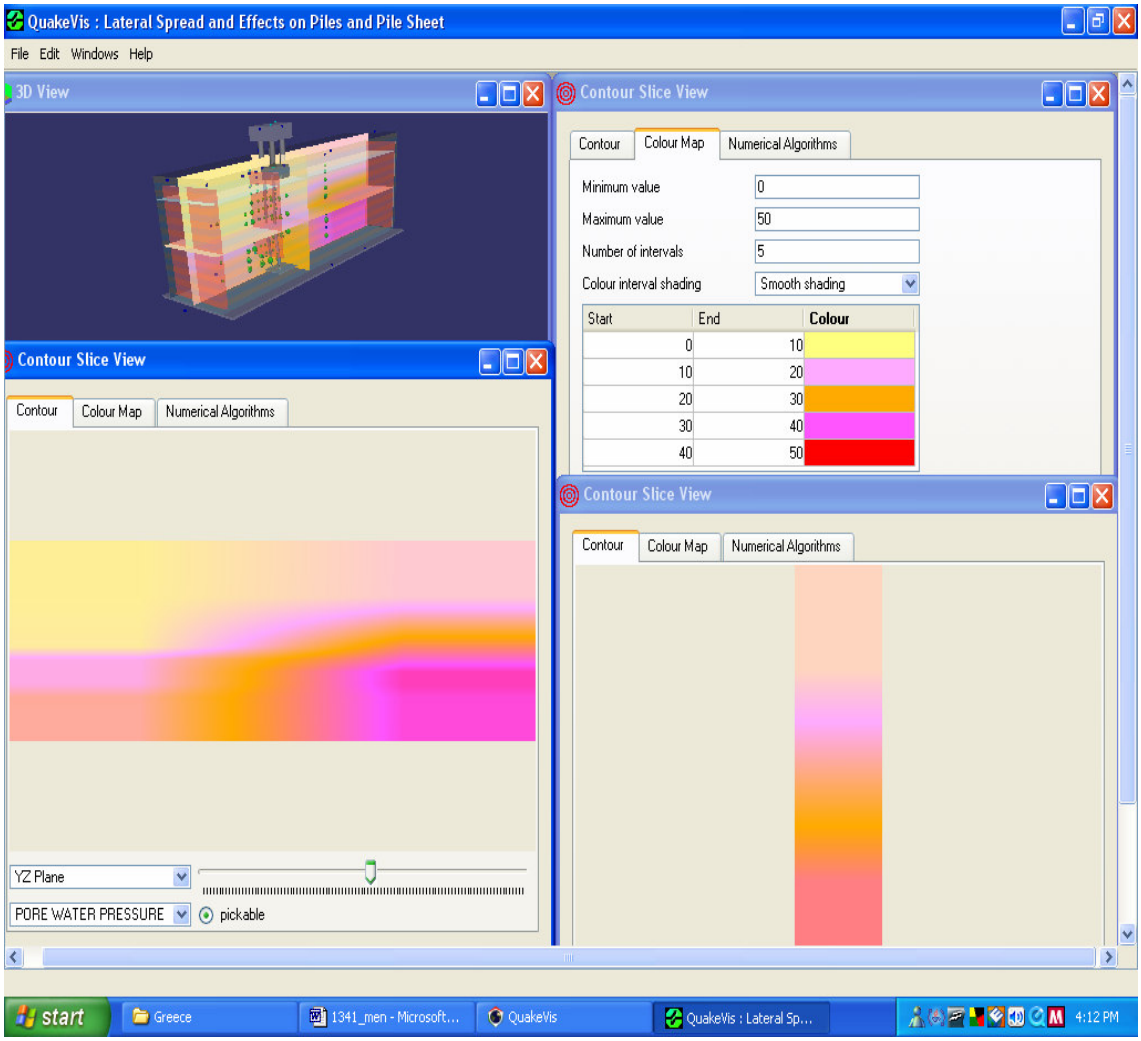

Figure 5. Multiple Contour Slice View windows displaying spatial and temporal variations of Pore Water Pressures

Dynamic Models

Dynamic models are 3D models that will be articulated by the displacement sensors. When the sensor moves, the 3D model will move in relation to the sensor. Models that are to be articulated by sensors need to be tessellated with a high enough level to adequately represent the movement of the real world object in 3D space. Hence, there should be at least one vertex division along profiles per sensor spacing. For example, if a pile has 12 sensors, it should be modeled with a minimum of 12 cylindrical sectors. More sectors will better represent the movement of the object, but too many may have an impact on display performance. In the metadata file, dynamic models will be represented as follows:

```
<Model>pile.osg</Model>

<DisplacementGroup>
   <Model>pile.osg:northwest</Model>
   <SensorArray>DISPLACEMENT X:S3_no_depth</SensorArray>
</DisplacementGroup>
```

The first tag <Model> determines a 3D model to import. The <DisplacementGroup> tag then defines the mapping of the 3D model subcomponent named *northwest* in the file named *pile.osg* and the sensor group named *DISPLACEMENT X:S3_no_depth*. It is recommended that files that will be dynamically articulated during animation be converted to *.osg* file format. The reason for this is that the *.osg* file format is a human readable ASCII format, allowing for easy editability. In the case of models that have subcomponents that will be articulated, and the modeler may not have assigned a name to the subcomponent, the user can simply add the name in the file.

The Container

QuakeVis comes with an internal mechanism to create a container and directives in the metadata file that define a container. However, there may be instances of specialized containers that require parameters that are not covered specifically in the internal mechanism. In this case the metadata file should omit the use of the Container specification and a 3D model representing the container should be made. The model may then be specified in the metadata file along with a corresponding DisplacementGroup for articulating it. Figure 1 shows an experimental setting using a rigid wall container sitting on a shake table. Figure 6 shows a snapshot during the first seconds of seismic shaking of another experimental setting using a cylindrical laminar container. This experiment includes a 3-by-3 pile group with a mass on top of the pile cap, dry sand, and sensors distributed

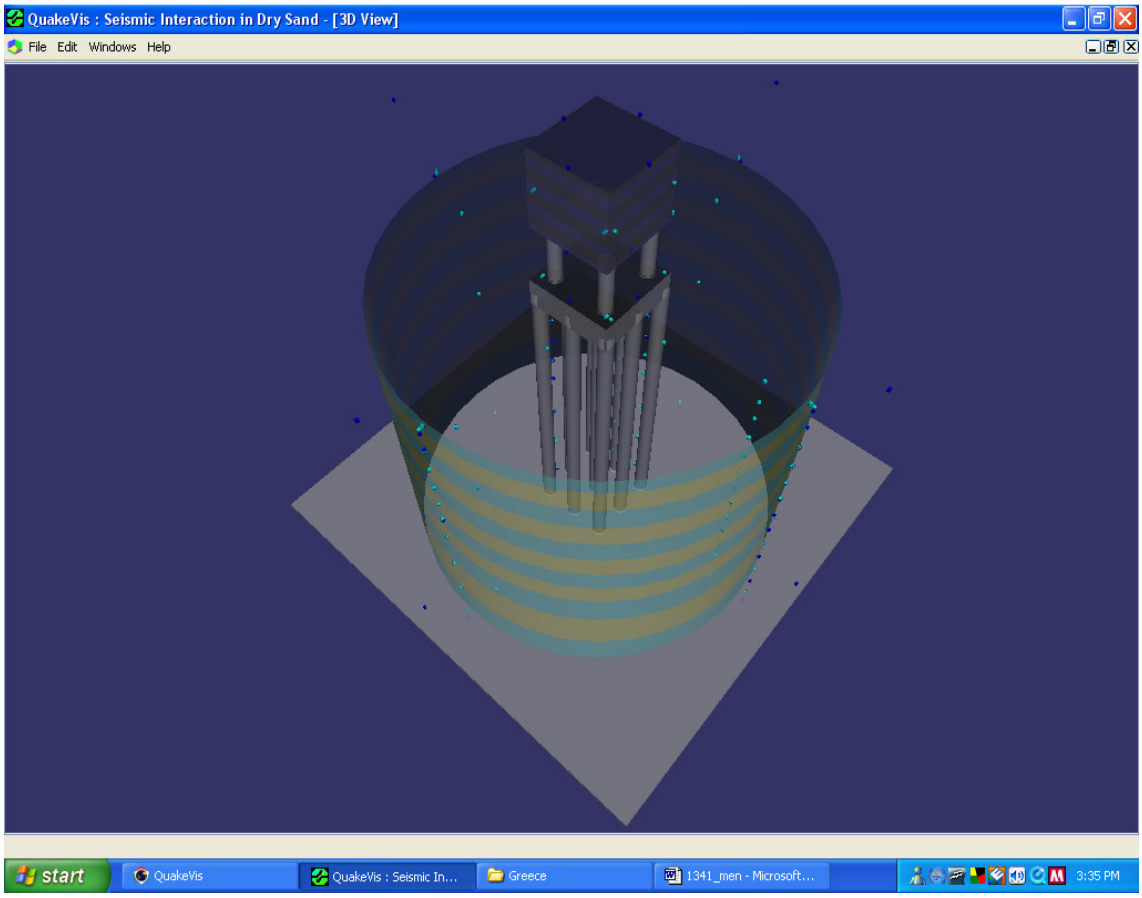

Figure 6. 3D View of a Cylindrical Laminar Container

around piles, sand deposit and container. This setting aims at studying the seismic interaction between piles and dry sand, an important issue observed during the 1995 Kobe Earthquake (Tokimatsu et al. 1998). *QuakeVis* can control the opacity, dimensions, thickness of the laminates and scale of the displacements of the laminates during simulated earthquake shaking.

Running an Experiment

With the data prepared and the 3D models and metadata file created, the user can now turn attention to running *QuakeVis*. Just double-click the *QuakeVis* icon to start the application, click on File menu and choose Open, and navigate to the metadata file containing the experiment definition and then enjoy the animations and visualizations.

CONCLUSIONS

QuakeVis is a robust visualization system capable of handling hundreds of sensors and of rendering 2D and 3D representations of large experimental databases such as the two seismic experimental settings briefly described in presenting the capabilities and features of *QuakeVis*. *QuakeVis* provides an analyst with the tools and resources to manage, interpret and visually analyze different quantities recorded during seismic experiments and compare and correlate these quantities for analyses purposes. Issues such as seismic interaction of soil (or liquefied soil) and structures are better studied in a large scale or near-full scale setting, and QuakeVis aids in data interpretation, visualization and analyses. QuakeVis In addition *QuakeVis* has the potential of being used for visualization of computer simulation of seismic behaviour of geotechnical systems.

REFERENCES

Abe, A., Meneses, J., Sato, M., and Kuwabara, F. "Web-based visualization of liquefied sand-pile interaction in near-full scale testing," Proc. 11th ICSDEE and 3rd ICEGE, Vol. 1, 828-835, Berkeley, January 2004.

Abe, A., Meneses, J., Sato, M. and Mohajeri, M. "Near-full scale testing and analysis of saturated sand-pile interaction under earthquake condition," Proc. 13th World Conference on Earthquake Engineering, Paper No. 2490, Vancouver, Canada, August 1-6, 2004.

Meneses, J. and Morón, I. "Interactive web-based visualization tools for teaching and learning experimental earthquake physical modeling," Proc. International Conference on Engineering Education and Research "Progress Through Partnership," VSB-TUO, Ostrava, Czech Republic.

Reese, L.C., Cox, W.R. and Koop, W.D., "Analysis of Laterally Loaded Piles in Sand," Paper No OTC 2080, Offshore Technology Conference, Houston, pp 473-482, 1974.

Reese, L.C., "Handbook on Design of Piles and Drilled Shafts Under Lateral Loads," US Department of Transportation, Federal Highway Administration, Paperback 2006.

Tukey, J.W., "Box-and-Whisker Plots," Exploratory Data Analysis. Reading, MA, Addison-Wesley, pp. 39-43, 1977.

Tokimatsu K. et. al., "Effects of Liquefaction-Induced Ground Displacement on Pile Performance in the 1995 Hyogoken-Nanbu Earthquake", Special Issue 2, Soils and Foundations, pp 163-177, 1998