Simulation of physical experiments in immersive virtual environments

Ahmed K. Noor and Tamer M. Wasfy
Center for Advanced Engineering Environments, Old Dominion University, NASA Langley Research Center, Hampton, Virginia, USA

Keywords *Computational methods, Numerical modelling, Software, VR*

Abstract *An object-oriented event-driven immersive virtual environment is described for the creation of virtual labs (VLs) for simulating physical experiments. Discussion focuses on a number of aspects of the VLs, including interface devices, software objects, and various applications. The VLs interface with output devices, including immersive stereoscopic screen(s) and stereo speakers; and a variety of input devices, including body tracking (head and hands), haptic gloves, wand, joystick, mouse, microphone, and keyboard. The VL incorporates the following types of primitive software objects: interface objects, support objects, geometric entities, and finite elements. Each object encapsulates a set of properties, methods, and events that define its behavior, appearance, and functions. A "container" object allows grouping of several objects. Applications of the VLs include viewing the results of the physical experiment, viewing a computer simulation of the physical experiment, simulation of the experiment's procedure, computational steering, and remote control of the physical experiment. In addition, the VL can be used as a risk-free (safe) environment for training. The implementation of virtual structures testing machines, virtual wind tunnels, and a virtual acoustic testing facility is described.*

1. Introduction

Virtual (or synthetic) environments (VEs) are three-dimensional, computer-generated environments that can be interactively experienced and manipulated by the user in real-time. A VE is either a projection of some real environment, a fairly realistic environment that does not exist, or an unreal environment (e.g. for entertainment and games). VEs provide a natural interface between humans and computers by artificially mimicking the way humans interact with their physical environment. A VE includes facilities for interfacing with humans through output of sensory information and input of commands. Output facilities include an immersive stereoscopic display, and stereo speakers. Input facilities include hand-held 3D navigation devices, such as a wand, joystick, or 3D mouse; 2D navigation devices such as a mouse or a touch pad; haptic feedback devices such as gloves; devices for position and orientation tracking

The present research is supported by a NASA Cooperative Agreement. The IVRESS and DIS codes were provided by Advanced Science and Automation Corp. The authors would like to thank Derek Wilson, Michael Copeland and Jeanne Peters of the Old Dominion University for their assistance in producing the figures; and Richard McGinnis and Larry Matthias of the Distributed Active Archive Center (DAAC) at NASA Langley Research Center, for the use of the DAAC CAVE facility.

of parts of the user's body (such as the head and hands); a microphone for voice commands; and a keyboard.

Many VE applications have been reported in the literature. Among the engineering applications are:

(1) *Visualization of 3D models* including navigation and walk/fly-through.

(2) *Visualization of the results of physical experiments.*

(3) *Visualization of numerical simulation results.*

(4) *Simulation-based design and computational steering.*

(5) *Visual simulation of physical systems* (e.g. flight simulators). In this application, a virtual model of a physical system, such as a machine or an airplane, is constructed. The user can walk/fly-through and interact with the model as if he is interacting with the actual physical system. For example, when a user presses a button in the virtual model, the VE reacts the same way as it would in the physical environment.

(6) *Tele-presence* (e.g. virtual presence in harsh/remote environments).

(7) *Virtual product development* including solid modeling, assembly, and virtual prototyping.

(8) *Tele-collaboration* in all of the aforementioned applications.

In this paper, an object-oriented event-driven immersive virtual environment (VE) is used for the creation of virtual labs (VLs) for simulating physical experiments. The virtual labs are being constructed at the Old Dominion University's Center for Advanced Engineering Environments (ODU-AEE). They include structures testing machines, wind tunnels, and acoustic testing facilities, which are used in the design and testing of aerospace systems at NASA Langley Research Center. In the succeeding subsections, a number of aspects of the virtual labs are described, including interface devices, software objects, and applications.

2. Immersive virtual-reality facilities

2.1 Achieving a sense of presence

In order for a VE to mimic the real environment it must be able to couple the sensory output of the environment to the real-time actions (navigation) of the user. Recent review articles (Mills and Noyes, 1999; Nash *et al.*, 2000; Stanney *et al.*, 1998) and a monograph (Dai, 1998) provide an overview of the current research on coupling the visual output (for recognition, tracking of moving objects, distance judging, search, and size estimation), auditory output (for sound localization and recognition), and kinesthetic/haptic output, with the user's navigation (fly-through and manipulation of objects) in the VE. In order to achieve a "realistic" VE (a VE in which the users are fully immersed and feel as if they are actually "present"), the following capabilities are needed:

- High-resolution (more than 1,024 × 1,024) 24-bit color, flicker and ghosting free, stereoscopic display.
- Large field-of-view (FOV) ~ 180°.
- Frame rate of at least 15 frames/second.
- Head and body tracking.
- Photo-realistic rendering including light sources, textures, transparency, and shadow casting.
- Consistency (the position and appearance of objects are as predictable as in a real environment).
- No disturbance from the real-world environment.
- Navigation tool that allows accurate direction pointing and fly-through or walk-through in the environment.
- Facilities for recognizing the actions of the user, such as touching or clicking an object. Many VEs have included standard graphical user interface objects such as buttons, dials, menus, checkboxes, etc.
- Realistic stereo sound effects that can help localize the sound producing objects.
- Support for haptic feedback devices such as gloves and pressure sensitive joystick with force feedback.
- Tracking multiple points on the user's body, allowing the use of multiple body parts (e.g. hands, head, and legs) to interact with the VE. It also allows displaying a realistic "avatar," which closely mimics the user's actual movement.
- Generating secondary motion effects where a user's motion and actions in the VE generate motion of objects in the environment. This also includes computational steering, where the user can change the parameters of the model in the VE and watch the simulation respond to that change (Ryken and Vance, 2000; Ginsberg, 1999; O'Brien *et al.*, 2000).
- Two-way natural language communication, including voice commands. The VE can feature an intelligent, voice-enabled virtual assistant that can respond to the user's speech (Tarau *et al.*, 1999).

2.2 VE facility

A four-wall immersive VE facility that supports the aforementioned capabilities is shown in Figure 1. The VE facility includes output and input hardware interaction devices, computers for generating the VE, and networking facilities for communication with other computers. A review of the input (navigation) and output (display) devices used in VEs is presented in Nash *et al.* (2000). The VE facility includes the following output devices:

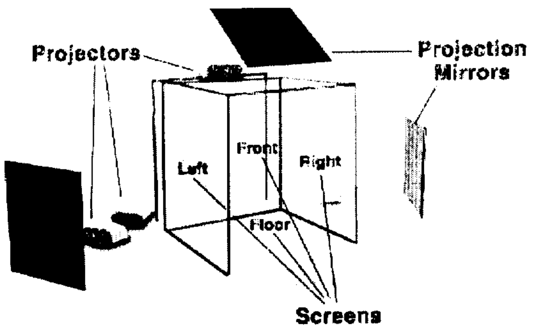

Figure 1.
VE facility used in the present study

- *Immersive stereoscopic display* is provided by using four 3 × 3m synchronized stereoscopic back-projected screens arranged as a cubicle room with a front, a floor (front-projected), a left, and a right screen (Figure 1). This configuration provides a field-of-view of at least 180°. Stereoscopic viewing is achieved by displaying the correct perspective view of the model for both eyes of the user using LCD shuttered glasses that are synchronized with the screen refresh rate. When the correct perspective view for the right eye is displayed, the left eye is blacked out, and vice versa. An infrared emitter, linked to the display output signal, sends the screen refresh trigger signal to infrared receivers in the glasses to shutter them. A refresh rate of 96Hz provides a nearly flicker-free stereoscopic display.
- *Four speakers* are used to output spoken messages, sound effects, and data sonification.

The following input devices are used:

- *Position and orientation tracking devices* for tracking the position and orientation of the user's body. Tracking receivers are usually placed on the stereo glasses for head tracking in order to calculate the correct perspective view, as well as a hand-held "wand" for navigating and pointing in the VE.
- *Tracked wand.* The wand has a pressure sensitive 2D joystick and three buttons that can be programmed to perform special functions.
- *2D navigation device* such as a mouse, touch pad, or joystick.
- *Microphone* for voice commands.
- *Keyboard.*
- *Haptic feedback gloves* that measure the position and movement of the fingers and wrist, and provide force feedback to the fingers.
- *Video camera.*

Typically, the following computers are used:

- A display computer with a high-end OpenGL graphics board for rendering the view for each projection screen.
- A master computer for storing the VE, synchronizing the display computers, and managing the changes to the VE that need to be sent to the display computers.
- A support computer for sound input/output, tracking and navigation.

The computers communicate via a standard Ethernet connection. Older virtual reality facilities used only one computer for rendering the view for all four screens. While this configuration does not require communication and model update management, it does not provide the rendering band-width of 10^6 polygons at 15 frames/second, required for the virtual labs.

3. The virtual labs

The virtual labs described herein leverage modeling, simulation, and information technologies to create an immersive, highly interactive virtual environment tailored to the needs of researchers and learners. Virtual experiments in the VLs are not exact duplicates of their real-world counterparts and, therefore, can provide educationally valuable features not available in physical experiments (for example, a fatigue test can be simulated in a few minutes).

3.1 Object-oriented tools for building VLs

Many general purpose object-oriented toolkits for constructing VEs have been developed. In an object-oriented paradigm, each object encapsulates a set of properties (data), which determines its appearance and behavior. The objects are polymorphic (i.e. objects of different type can contain or respond to the same function without regard to the object type). The objects are persistent (i.e. they behave in a natural predictable way). Also, their properties can be modified, they can be deleted, and new objects can be added. In addition, different objects can be grouped into one "group object". The last characteristic allows a hierarchical, directed, tree-type representation of the VE. The group object can be transformed (translated, rotated, and scaled) as one entity. This hierarchical object-oriented representation (including the transformation hierarchy) is called the "scene-graph" (Foley *et al.*, 1990).

Several general-purpose toolkits, based on the scene-graph approach for the construction and display of VEs, have been developed. Among the toolkits that enable the development of custom VE applications are: SGI's Inventor (Strauss and Carey, 1992), SGI's IRIS Performer 2.0 (Eckel, 1997), WorldToolKit (WKT) from Sense8, MR Toolkit (Shaw *et al.*, 1993), μuse SDK from MUSE Technologies, Karma VI (Germs *et al.*, 1999) for visualization of geographical information, and Lego Toolkit (Ayers and Zeleznik, 1996), and IVRESS (Integrated Virtual Reality Environment for Synthesis and Simulation) (Wasfy and Noor, 2000). These toolkits consist of a collection of C/C++ functions and classes for interfacing with the various hardware components, constructing the VE, and navigation in the VE. The Virtual Reality Modeling Language 2.0 (VRML 2.0) is a file format specification for scene-graph description of VEs on the Internet. VRML includes a set of primitive geometry, grouping, sensor, interpolator, texture map, light objects, which allows construction of dynamic interactive multimedia VEs.

3.2 IVRESS toolkit

IVRESS is an object-oriented stand-alone toolkit for constructing VEs. IVRESS was used to construct the virtual testing machines and wind tunnels. IVRESS script is an interpreted scripting language based on ECMA script, that allows setting the properties of the various objects, and writing custom event handling routines. In addition, custom objects can be added to IVRESS by writing C/C++ code for the object and linking that code to IVRESS either dynamically (using a dynamic link library), or statically (by linking with an IVRESS static library file).

Four types of objects (Figure 2) are used to construct the VLs. These are:

- *Interface objects* (IOs) provide various functions in the VE. Typical IOs include container, label, text box, button, check box, slider bar, dial, table and graph. These objects display information in the VE and allow the user to input data or commands in the VE. The container is a special type of object that is used to group "children" objects.

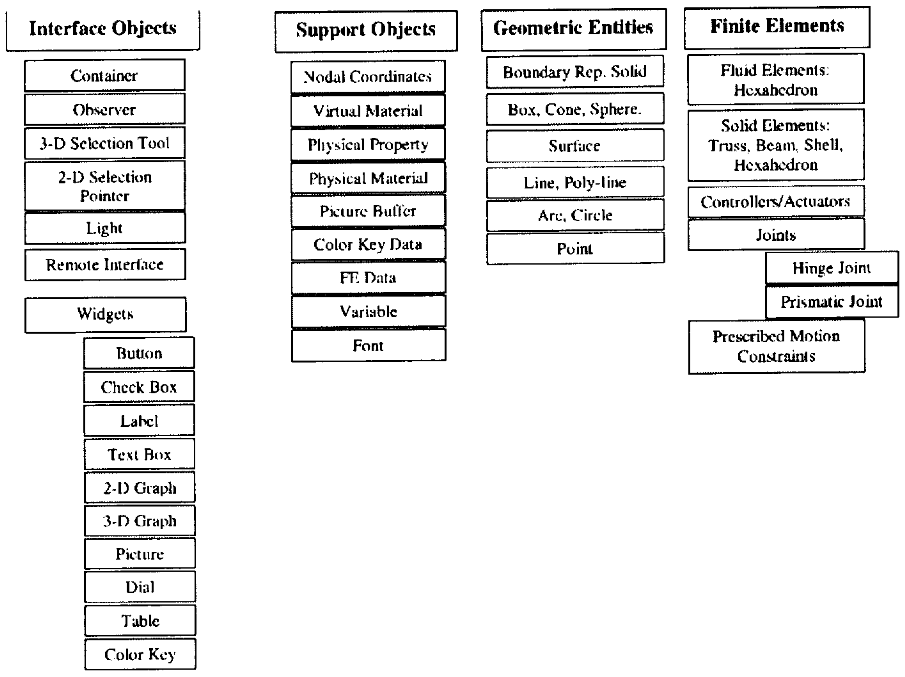

Figure 2.
The four types of objects used to build a VE

- *Support objects* (SOs) contain data that can be referenced by other objects. Typical SOs include material properties, time-history data and mode-shape data. For example, a finite element refers to a nodal positions list, a physical material and a material color support object. Arithmetic (such as addition, multiplication and division) and logical (such as "and", "or", and "not"), can be performed on SOs.
- *Geometric entities* (GEs) represent the geometry of the physical components of the VL. Typical GEs include boundary-representation solid, box, cone and sphere.
- *Finite elements* (FEs) represent the numerical model of the physical components of the VL. Typical FEs include beam, shell and hexahedral solid elements, and hexahedral fluid elements.

Each object has properties that determine its state and behavior, methods which are functions that it can perform, and events which are triggered when certain conditions, initiated by the user or the passage of time, are met. All objects have the same basic structure. Each object defined in the script file has a name and may be followed by a list of properties and property values. Property values that are not explicitly defined are set to a default value.

Common properties of IOs include translation, orientation, scale, foreground material color name, background material color name, and visibility. Common methods for IOs include *Draw* and *Check-events*. The container methods invoke the methods of all the children nodes. For example, the container *Draw* method invokes all *Draw* methods of all the objects contained within, including other containers. Typical events include *Touch*, *Press*, and *Click*. For example, the *Touch* event is invoked when a selection tool touches the object. The *Click* event is invoked when a selection tool is touching the object and the user clicks on the first wand button. An event is triggered by calling a subroutine associated with that event. The subroutine name consists of the object name concatenated with an underscore and the event name (e.g. *object-name_event-name*).

IVRESS can read and write file formats for geometry data such as VRML 2.0, Open Inventor (Strauss and Carey, 1992), DXF, and LightWave; finite element information such as MSC/NASTRAN, MSC/DYTRAN, ABAQUS, DIS, CFL3D (Krist *et al.*, 1998), and PLOT3D (Walatka *et al.*, 1990); pictures such as Bitmaps, PNG, JPEG, and GIF; and movies such as MPEG, AVI, and MNG. In addition, IVRESS has facilities for voice commands and LAN communication with other computers.

3.3 Modeling of the VLs

A hierarchical object-oriented model is used to construct each VL (see Figure 3). The model of the VLs described herein is composed of solid components and fluid domains. A *solid component* is a continuous solid which is either a bulk solid or a sheet metal surface-type solid with no internal joints or connections. Each component is represented as a container object, with the object name chosen as the name of the component. Each of the solid components is modeled using a geometric solid (GS) model and a finite element (FE) model. The dynamic response of the solid components is calculated using the FE model. A flexible multibody dynamics code, DIS (Dynamic Interactions Simulator), which can be used as an IVRESS plug-in, is used to obtain the time-history of the motion of the FE model nodes. These nodes serve as control points for the GS model. Thus, the motion of the GS model follows the motion of the FE nodes. The GS model can be either a boundary-rep solid or a NURBS surface geometric entity. The geometric entity refers to the following support objects: material color, picture texture map to be applied to the entity's surface, and physical material. The FE model is a set of finite elements. The finite elements of a component refer to a node list support object. In addition, each finite element refers to a physical material support object (which is usually the same as the GS physical material).

Two solid components can be joined together using a *connection*. A connection is modeled as a container which can include one or more of the following elements:

- *Zero-length springs.* The zero-length spring is a spring-dashpot-Coulomb friction element with the unstressed length set to zero. The parameters of the element are the spring stiffness, the damping coefficient, and the Coulomb friction force. The element connects two arbitrary points on the two components. This element can model a spherical joint, a revolute joint, or a rigid connection between two components. A spherical joint is modeled by connecting the two components using one zero-length spring. A revolute joint is modeled by connecting the two components using two or more co-linear zero-length springs. A rigid connection is modeled by connecting the two components using three or more non-co-linear zero-length springs.
- *Prismatic joint.* This element connects a point on the first component to a curve on the second component. This curve is specified using a piece-wise linear curve composed of a set of points on the second component. The parameters of this element include the value of the penalty parameter used to ensure that the point follows the curve.

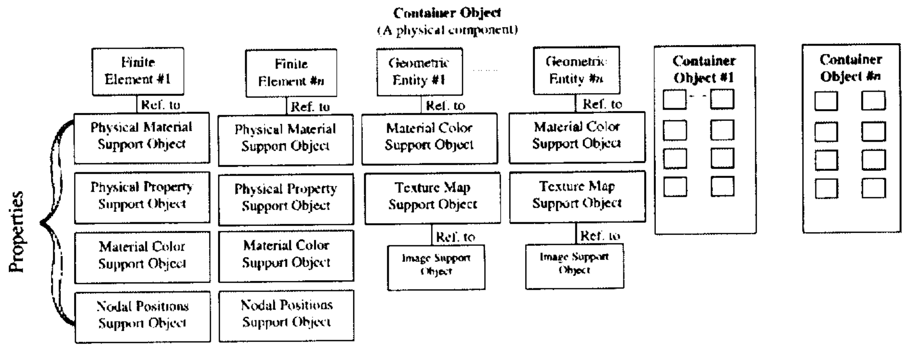

Figure 3.
Object-oriented representation of a component of a physical system

- *Contact model.* This element specifies that the two components can come into contact with each other. The properties of this element include the value of the penalty parameter used to ensure the impenetrability condition between the two components, and the friction law used along with the friction law parameters, such as the Coulomb friction coefficient.

The relative motion between two components can be controlled by using an actuator. An actuator is modeled as a container, which can include one or more of the following elements:

- *Linear actuator.* A linear actuator can be placed between two points on two components. Both points can be on one component. The properties of this element include the type of control (such as PID) and the controller parameters (such as the PID controller gains).
- *Rotary actuator.* A rotary actuator connects three points with the center point in common between the two components. The properties of this element include the type of control (such as PID) and the controller parameters (such as the PID controller gains).

A solid component container can also contain an IO such as a button, dial, slider bar or graph. Thus, the component can be used to represent the controls and displays of the lab. The control logic of the buttons is programmed by writing custom event handling script in the event subroutines associated with the IO.

A *fluid domain* is a continuous closed volume containing the fluid. Each fluid domain is represented as a container object. Similar to the solid components, the fluid domain is modeled using a GS model and an FE model. The GS model is a boundary-rep solid. The FE model is a set of fluid hexahedral elements, which can form an unstructured mesh or a block structured mesh. Each finite element refers to a node list and a physical material support object. A CFD code can be used to model the fluid flow and fluid-structure interaction dynamic response.

Components and fluid domains of the model can be grouped into a sub-assembly container. The sub-assemblies are grouped into the model container. The model container is placed in the "WORLD" container.

3.4 Viewing the experiment

The solid components can be viewed in the VE using any combination of the following:

- Photo-realistic rendering of the GS and/or FE model (including light sources, surface textures and shadows).
- Animation of the motion of the GS and/or FE model of the machine and test article.

- Color shaded view of the FE model surface of the machine and test article, using a scalar response quantity (such as stress/strain component, combined stress/strain, displacement/velocity/acceleration component, temperature, strain energy density, etc.).
- An arrow plot of a vector response quantity (such as displacement, velocity, principal strains/stresses, etc.). The length and color of an arrow can represent a scalar response quantity.

The fluid domains are discretized using a multi-block structured or an unstructured finite element or finite volume mesh. The VE allows viewing of the model using any combination of the following:

- Volume rendering using transparency.
- Surface shading of a cross-section of the fluid domain using a scalar response quantity (such as pressure, temperature, velocity magnitude, etc.).
- Surface arrow plot on a cross-section of the fluid domain using a vector response quantity (such as velocity, acceleration, etc.).
- Volume arrow plot of a vector response quantity (such as velocity).
- Stream lines, streak lines or path lines colored using a scalar response quantity.

3.5 Important features of the VLs
Some of the important features of the VLs that are required to construct the virtual labs include:

- *Standard user-interface widgets.* These are required for constructing a menu system for interacting with users as well as for simulation of the various buttons, dials, and gauges in the physical lab. User-interface widgets include button, check box, label, dial, slider bar, table, and graph.
- *Support for multiple level-of-detail models.* IVRESS can automatically extract, from a detailed high polygon count geometric solid model, a hierarchy of lower polygon count models. The fidelity level of the displayed model is proportional to the distance between the viewer and the object.
- *Voice commands.* A support computer runs a voice commands interface program, which uses a speech recognition engine (such as Microsoft SAPI 4.0). The voice command is translated to a set of IVRESS script commands. The script is sent to the master computer. A "remote commands" object is set to listen to and execute the script commands that are being sent from the support computer.
- *Collision detection.* When collision detection is turned on the user cannot set his view point to be inside a solid component such that the user's

head does not penetrate a solid component. Three types of collision detection can be used, namely, bounding box, embedded spheres, and detailed geometry. A property of the container is the type of collision detection used.

- *3D selection tool.* This tool is used to select, move and touch objects in the VE, and is controlled by using the wand. The object consists of a selection bounding box and two perpendicular vectors, indicating the current spatial orientation of the wand. The 3D selection tool can be used in absolute or incremental mode. In the absolute mode, the physical absolute position of the wand is set to the position of the selection tool bounding box in the VE. In the incremental mode, the user points the wand in the direction he wants the selection box to move (the direction vector) and then uses the pressure sensitive joystick to control the speed of motion in that direction. Once the selection bounding box touches an object, a touch event for that object is triggered which, in turn, executes an associated sub-routine. Also, a click event is triggered when the selection box is touching the object and the user clicks the first wand function key.
- *Virtual hand.* The virtual hand object interfaces in real-time with a hand motion capture and force feedback gloves. The virtual hand follows the motion, including the finger motions, of the actual hand of the user. The virtual hand allows the user to grasp virtual objects. Force feedback to the user's hand allows the user to touch and manipulate virtual objects in a natural way.
- *User avatar.* An avatar object is a geometric model of a user. Position/orientation tracking receivers are placed on various parts of a user's body including hands, head, neck, torso, and legs. The data from these sensors are used to generate the motion of the avatar using inverse kinematics to determine the avatar's joint angles. The user can select an extra body view and observe his own avatar in the VE. In addition, avatars of remote physical users and computer generated virtual users can be displayed in the VE.
- *Intelligent agents (IA).* The function of an IA is to assist the user and perform user requests. The agent can output information to the user as spoken text, written text, or images. The user can input commands to the agent either through voice, typing on the keyboard, or a screen menu. A virtual avatar of the IA can be displayed. The agent maintains a history of the user's requests. It uses this information to build an internal state which allows the agent to provide intelligent responses to the user. The agent can be used as a virtual instructor to provide information to the user based on the user's progress and performance.
- *Computational steering and simulation-based design.* IVRESS can read the input and output files of structures' finite element codes such as

MSC/NASTRAN, MSC/DYTRAN and ABAQUS, and fluid flow structured and unstructured codes such as CFL3D. In addition, IVRESS is tightly integrated with DIS (Dynamic Interactions Simulator), which is a time-accurate finite element code for modeling flexible multibody dynamics and fluid-structure interaction. IVRESS and DIS share the same object-oriented model database described in section 3.3. IVRESS can be used to start a DIS simulation as well as to change model parameters before and during a simulation. Also, IVRESS can dynamically display the simulation results as they are being computed by DIS as well as read pre-computed stored DIS results. DIS is based on the nonlinear finite element method, a global inertial reference frame, corotational/convected local element frames, and explicit-time integration. DIS can run in parallel on shared and distributed memory parallel computers. The explicit solution procedure used in DIS allows a nearly linear parallel speedup on an element-by-element basis.

- *Tele-presence and remote control of experiments.* IVRESS support objects can interface (input and output data) with a "socket" object. This object allows real-time two-way communication with remote computers. Thus, a picture taken with a remote camera can be displayed in real-time in the VE. Also, data acquired from an experiment can be read from the lab data acquisition computer and displayed in real-time as graphs, dials, surface shading, and position/orientation of virtual objects in the VE. Furthermore, the user actions in the VE can be communicated to the lab control computer. Thus, the lab can be controlled from within the VE.
- *Collaboration.* A client-server approach is used for collaboration between remote multiple users. The common server is used to store the entire VE. The clients send the actions and the state of the client's user to the server. The server integrates the actions of all the clients, updates the VE, and sends, to the clients, the information needed to update the display of the VE. In addition, a video camera in the VE facility, displaying a live picture of a user to all other users, enhances collaboration.
- *Data sonification.* Four speakers provide 3D sound so that the user can localize the sound source in 3D. This is done by manipulating the sound component volume for each speaker.

3.6 Common uses of the virtual labs

The virtual labs can be used for any combination of the following:

- *Visualization of the computer model of the test facility.* The user can walk/fly-through the virtual lab.
- *Visualization of the results of the experiment.* These include graphs of various response quantities such as applied load vs. strain at various

points for structures testing machines. Pre-stored results can be displayed. Also, results that are read in real-time from the lab data acquisition computer can be displayed. The VE allows viewing of the results from any angle, zooming on a desired feature, and replaying the results at any speed.

- *Visual simulation of the physical experiment.* The simulation results displayed can include more information than the test measurements provide. For structures testing machines, a structures finite element code can perform a dynamic simulation of the machine/specimen multibody system. Animations of the test with the specimen color shaded using stress, strain and other response quantities can be displayed. In wind tunnel tests, CFD codes can predict the response time history of the pressure, temperature, and velocity fields in the fluid. The simulation results can include shaded stream and streak lines, volume rendering of pressure, velocity, or temperature fields, volume arrow plots, and shaded cross-sections of the fluid domain. The VE allows viewing of the results from any angle, zooming in on a desired feature, and replaying the results at any speed. Numerical simulations can provide more information about the response than experimental measurements. This helps in the design of the experiment by placing the instruments and sensors in critical regions.
- *Comparing the experiment measurements with the predictions of numerical simulations.* The response measured from the experiment and that predicted using numerical simulations can be superimposed, placed side by side, or subtracted.
- *Hybrid experimental and numerical modeling.* For example, a component of an aerospace vehicle can be tested in the actual lab, while the other components of the vehicle can be numerically modeled.
- *Computational steering.* If the response of the machine or wind tunnel can be predicted at a "reasonable" speed, then the dynamic response can be continuously predicted as the user is interacting with the VE. Thus, the effect of changing the shape and/or parameters of the virtual model can be studied in real-time. DIS simulations can be run from within the VE. The user can dynamically change the problem parameters, such as the applied load, while observing the simulation results.
- *Training and education.* The VE can simulate the actual operation of the physical test facility, including buttons, gauges, knobs, etc. These objects can mimic the response of the physical objects. Thus, the VE can be used for risk-free training and preparation for the experiment. In addition, an intelligent agent can guide a user through each test step and can provide tips on what to look for in order to gain a better understanding of the test.

- *Remote control of the lab (tele-presence).* The physical test facility can be remotely controlled from within the VE. If the lab has a control computer and control actuators, the VE can interface with the control computer so that the user actions in the VE can be communicated to the lab control computer and the experiment can be remotely controlled from within the VE. Also, data acquired from an experiment can be read by the lab data acquisition computer and displayed in real-time as graphs, dials, surface shading, and position/orientation of virtual objects.

3.7 Implementation of the VLs

Virtual models of several testing facilities, including structures testing machines, wind tunnels and acoustic testing facilities at NASA Langley Research Center, have been constructed at the ODU-AEE Center. Some of these facilities are described subsequently; namely, two structures testing machines, two wind tunnels, and an acoustic flow/jet noise testing facility.

Structures testing machines. Two machines are described herein; namely, the 50 Kips HTS (Hydraulic Test Stand) and the COLTS (Combined Loads Testing System). The 50 Kips HTS is used to determine mechanical properties including strength, stiffness, and fracture of structural sub-components. Specimens up to 2ft long can be tested under tension, compression or fatigue. The maximum stroke is 6in. The machine is equipped with a 5,000× microscope and a crack measurement apparatus. Up to 40 channels of data including applied load, position, and strain gage data can be acquired at a rate of 2Hz.

A computer model of the 50 Kips HTS is shown in Figure 4. The model has the following types of objects:

- Fixed (non-moving) machine components including the frame of the machine, desk, display monitors, keyboard, and control panels.
- Moving machine components. These include the two hydraulic cylinders and the two clamping heads. The FE model of these components is a rigid body model.
- The two hydraulic cylinders are modeled as linear actuators with tracking PID control.
- The machine controls include buttons, dials, and levers. The user can manipulate the controls using the wand based 3D selection tool, or the virtual hand. Figure 5 shows the 3D selection tool touching a lever. A semi-transparent red box is drawn around the lever. The user can activate the lever by clicking the first wand button.
- An output screen shows graphs of various response quantities. A menu allows choosing the graph parameters, such as force, strain, and displacement.
- The specimen can be a general anisotropic material such as a composite. The specimen is modeled using solid brick elements.

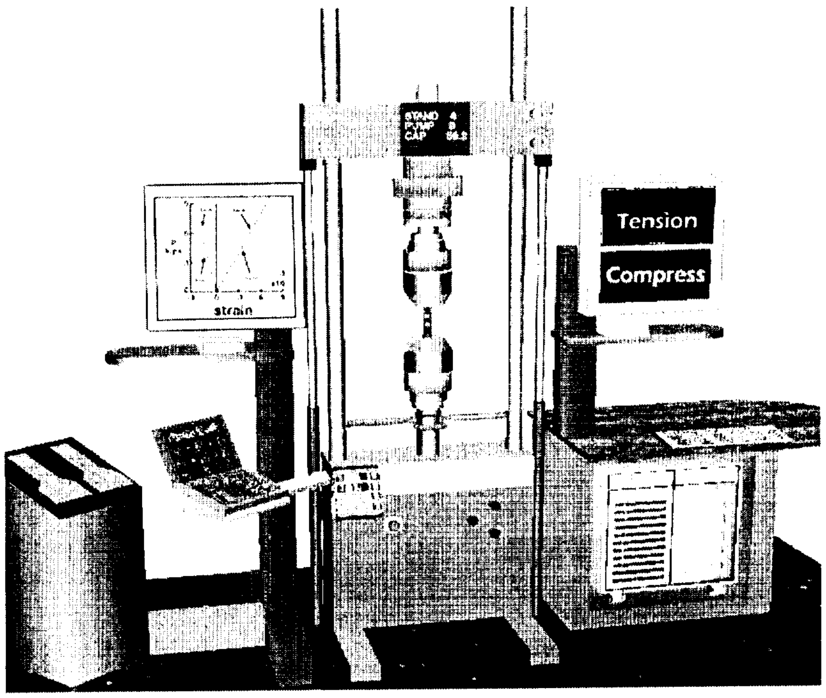

Figure 4. Computer model of the 50 Kips tension, compression, fatigue testing machine

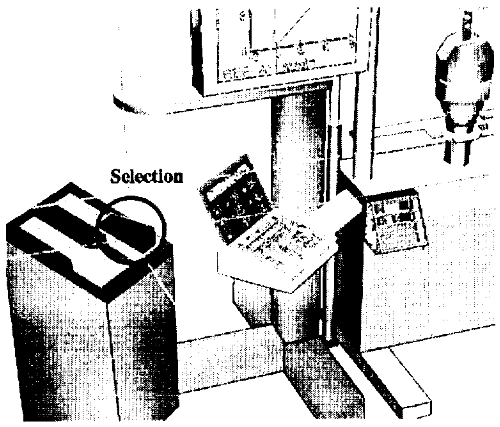

Figure 5. Selecting a machine control (clamping lever) by using the 3D selection tool

The tasks required to perform the various test steps are modeled into the virtual model. A typical structural test consists of the following steps:

- Switching on the machine and going through the machine checkout and setup checklists.
- Loading the specimen in the machine and clamping it.
- Applying the desired loading time history to the specimen.

- Collecting the test data such as applied load, displacement, and strain measured using strain gages placed on the specimen. The data is stored and can also be displayed in real-time on a display screen.
- Observing the specimen during the test including deformed shape, failure modes, and surface cracks occurring during the test.
- Removing the specimen from the machine.
- Performing the machine shutdown procedure.

DIS can be used to perform a dynamic simulation of the machine/specimen multibody system starting with the application of the load to the failure of the specimen. Strain gages are placed at various points on the specimen. Graphs of the measured applied load vs. strain at various points can be displayed in the VE. Also, a view of the specimen and cracks recorded using a camera can be shown in the VE. Pre-stored results can be displayed. Also, results which are read in real-time from the 50 Kips HTS data acquisition computer can be displayed.

The COLTS can test fuselage barrel sections (up to 4.6m diameter and 13.7m length) of aerospace vehicles under any combination of the following types of loads: axial tension, compression, and bending using six axial actuators (450 Kips each); shear using two horizontal actuators (300 Kips each); torsion using two vertical actuators (300 Kips each); and internal pressure (up to 155 KPa). All the loads can be independently cycled at up to 12 cycles per minute. Strain gages can be placed on the test article. The strain gage data is collected into a data acquisition computer.

A computer model of the COLTS is shown in Figure 6. The model has the following types of objects:

- Fixed (non-moving) machine components, which include the frame of the machine, display monitors, and control panels.
- Moving machine components, including the ten hydraulic cylinders, the two end platens, and connecting links. A rigid body FE model of these components is used.
- The hydraulic cylinders are modeled as linear actuators with tracking PID control.
- The machine controls consist of buttons and dials. The controls are manipulated by using the wand based 3D selection tool, or the virtual hand.
- An output screen shows graphs of various output quantities that are measured using the data acquisition system. A menu allows choosing the graph parameters such as applied load, strain, and displacement.
- The specimen can be a general anisotropic material. The FE model of the test article consists of brick elements and/or shell elements.

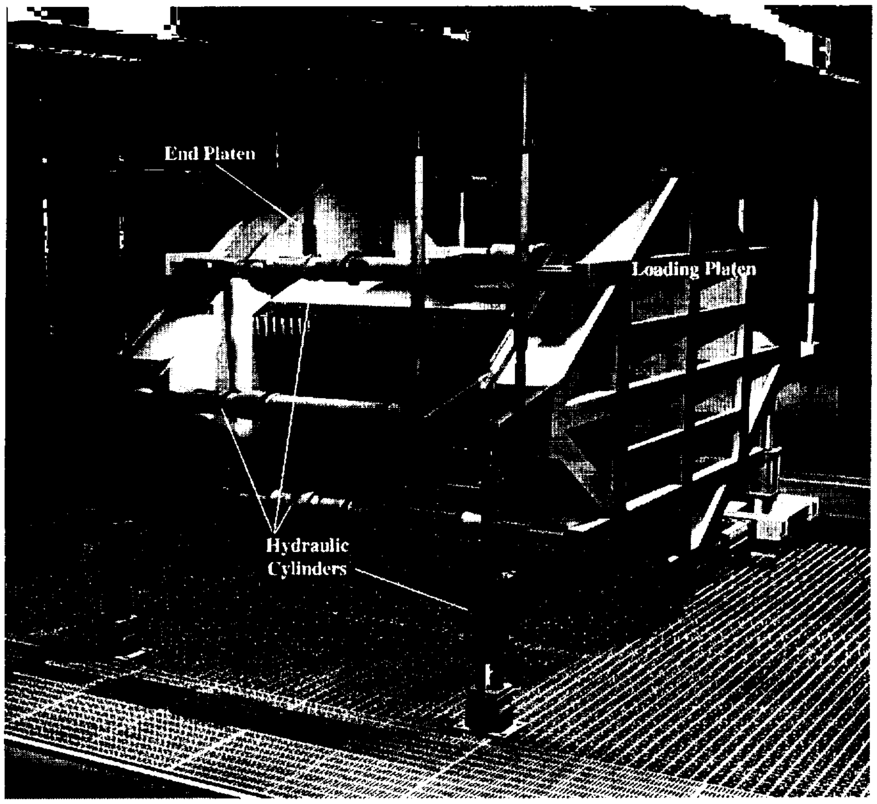

Figure 6.
Computer model of the combined load test system (COLTS)

The general test steps for the COLTS are similar to those for the 50 Kips HTS. DIS can be used to perform a dynamic simulation by predicting the response time history of the machine/test article multibody system due to the combined loading. The DIS simulation can be run from within the VE. The user can dynamically observe the simulation results and can dynamically change various applied loads. In Figure 7, a snapshot of the COLTS with a fuselage cylindrical section loaded with a twisting moment is shown. Also, a view of the COLTS machine and test article recorded using a camera can be shown in the

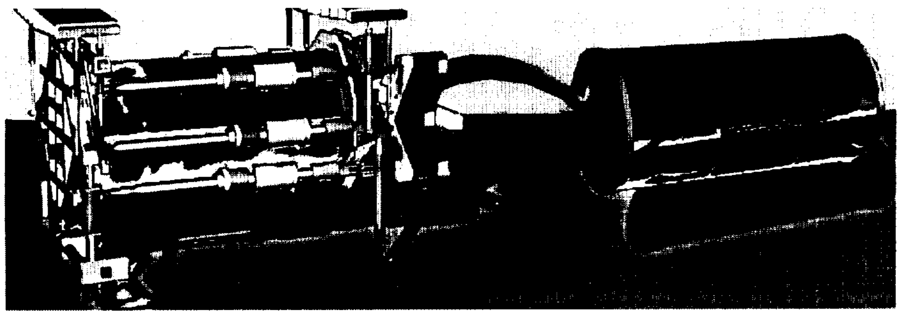

Figure 7.
Twisting of a cylindrical specimen in the COLTS

VE. Pre-stored results can be displayed. Also, results that are read in real-time from the COLTS data acquisition computer can be displayed.

Wind tunnels. Two wind tunnels are described herein; namely, the 8ft HTT (High Temperature Tunnel) and the NTF (National Transonic Facility). The 8ft HTT is the largest hypersonic tunnel in the world that provides temperature-controlled flow at up to Mach number 7. Temperature control is achieved by the combustion of methane and air at maximum pressures of 31,000KPa and maximum temperatures of 4,000K in a 30ft long combustion chamber. The facility has liquid oxygen, hydrogen and silane supply systems for propulsion tests. Among the types of tests that can be conducted in the 8ft HTT are: hypersonic propulsion systems, thermal protection systems, and aerothermal loads on large-scale high speed and aerospace re-entry vehicle components. Testing is conducted at steady-state conditions for approximately 30 seconds. About 1,500 channels of data can be acquired from the model, including velocity and pressure at various locations. A detailed description of the 8ft HTT and its test capabilities are given in Hodge and Harvin (2000).

An outside view of the computer model for the 8ft HTT is shown in Figure 8 and two inside views are shown in Figure 9. The model has the following types of objects:

- Fixed (non-moving) components. These include the test chamber, pod, nozzles, test windows, piping, frames, display monitors, and control panels.

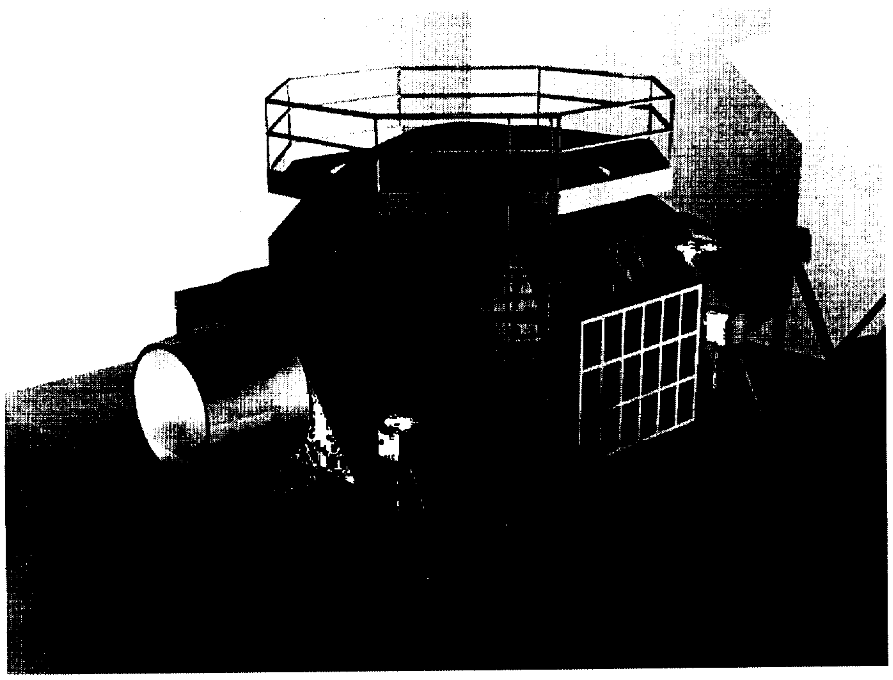

Figure 8.
Outside view of the computer model of the eight-foot wind tunnel test chamber

- Moving wind tunnel components. These include the model elevator, elevator carriage, model support strut, and the model. A rigid body FE model of these components is used.
- The hydraulic elevator is modeled as a linear actuator with PD control.
- The wind tunnel controls consist of buttons and dials. The user can manipulate the controls using the wand based 3D selection tool, or the virtual hand.
- An output screen shows graphs of various output quantities. A menu allows choosing the graph parameters such as force/moment on the model, pressure, temperature, and velocity at various positions.
- The domain is discretized using an unstructured grid of hexahedral fluid elements.

The tasks required to perform the various test steps are modeled into the virtual model. A typical aerothermal loads test consists of the following steps:

- Going through the tunnel checkout and startup checklists.
- Placing the model in the tunnel.
- The elevator is initially in the lower position (Figure 9(a)). The radiant heater system is activated to provide the desired surface temperature.
- The tunnel airflow is started. After steady state test conditions are achieved, the radiant heater is deactivated and retracted. The model is raised to the level of the nozzle center line (Figure 9(b)). The angle of attack is adjusted to the desired value. The entire process takes ten seconds from the time the heaters are deactivated to the time the model is at the desired orientation.
- The test is run for up to 60 seconds.
- The model is lowered.
- The tunnel airflow is shut down.

CFD codes (such as CFL3D) can be used to simulate the aerothermal loads test by predicting the response time history of the pressure, temperature, and velocity fields. The simulation can be run from within the VE. The user can

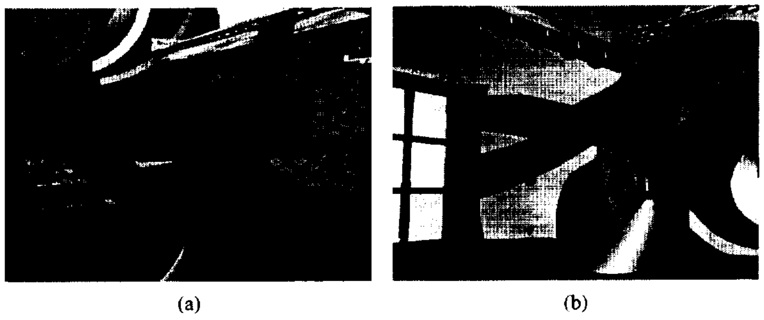
(a) (b)

Figure 9. Inside view of the computer model of the eight-foot wind tunnel test chamber with (a) the elevator lowered; (b) the model raised to the centerline of the flow

dynamically observe the simulation results and can dynamically change the upstream velocity and the surface temperature.

The NTF is a fan driven, closed circuit, continuous flow, pressurized wind tunnel. The test section is 2.5m × 2.5m and 7.6m long. The tunnel can operate at Mach numbers up to 1.2. The test temperature range is from 78-340K and the test pressure range is from 1 to 900KPa. The test gas may be dry air or nitrogen. For the air mode of operation, heat removal is achieved by a water-cooled heat exchanger (cooling coil). For the cryogenic nitrogen mode of operation, heat removal is achieved by evaporation of liquid nitrogen, which is sprayed into the circuit upstream of the fan. Full-span (string mounted) or half-span (wall mounted) models can be tested. With both temperature and pressure as test variables, three types of investigations are available: Reynolds number effects at constant Mach number and dynamic pressure; model aeroelastic effects at constant Reynolds number; and Mach number effects at constant dynamic pressure and Reynolds number. By using the cryogenic approach to generate high Reynolds numbers, the NTF achieves its performance of near full-scale conditions at lower cost and at lower model loads than concepts based on ambient temperature operation.

An outside view of a computer model of the NTF test section is shown in Figure 10 and two inside views are shown in Figure 11. The model has the following types of objects:

- Fixed (non-moving) components. These include the frame and shell of the test section, display monitors, and control panels.

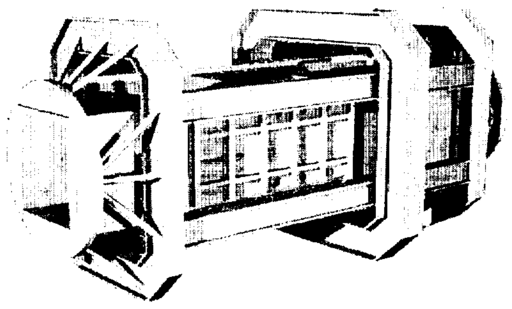

Figure 10.
Outside view of the computer model of the National Transonic Facility (NTF) test section

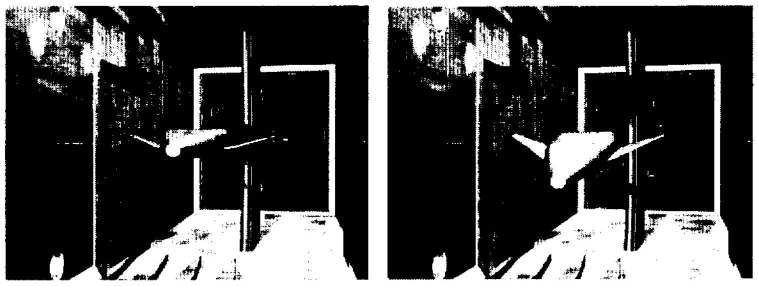

Figure 11.
Inside snapshots of the computer model of the NTF test section

- Moving wind tunnel components. These include the model support and the model. A rigid body element FE model of these components is used.
- Rotary actuators with PD control are used to control the various degrees of freedom of the model support system.
- The wind tunnel controls consist of buttons and dials. The user can manipulate the controls using the wand-based 3D selection tool, or the virtual hand.
- An output screen shows graphs of various output quantities. A menu allows choosing the graph parameters such as force/moment on the model, pressure, temperature, and velocity at various positions.
- The domain is discretized using an unstructured grid of hexahedral fluid elements.

A typical test consists of the following steps:

- Go through the tunnel checkout and startup checklists.
- Place the model support system with the model mounted at the desired angles in the tunnel.
- The model is placed at the desired orientation and position using the articulated model support (see Figure 11).
- The tunnel airflow is started.
- After steady state temperature and flow is reached, data is collected from the model including time history of velocity, pressure and temperature at various locations.
- The tunnel airflow is shut down.

CFD codes can be used to simulate the experiment. The user can dynamically observe the simulation results and can dynamically change the upstream velocity.

Acoustic flow/jet noise testing facility. One acoustic flow and jet noise testing facility is described, namely, the quiet flow facility (QFF). The QFF is a quiet open-jet anechoic acoustic testing facility. The purpose of the current experiments conducted in the QFF is to reduce the noise generated due to subsonic flow around commercial aircraft components during landing and takeoff. The QFF consists of a sound insulated test chamber. A 2×3ft rectangular nozzle fixed to the floor of the QFF provides a steady airflow onto a test article such as a flap. The air exits at the top of the QFF. A movable microphone array can be positioned at various points around the test article to measure sound amplitude and frequency in various directions. The test article can be rotated in order to change the angle of attack to the air flow. A description of the QFF along with experimental and numerical studies conducted in the QFF, is given in Macaraeg (1998).

A computer model of the QFF is shown in Figure 12. The model has the following types of objects:

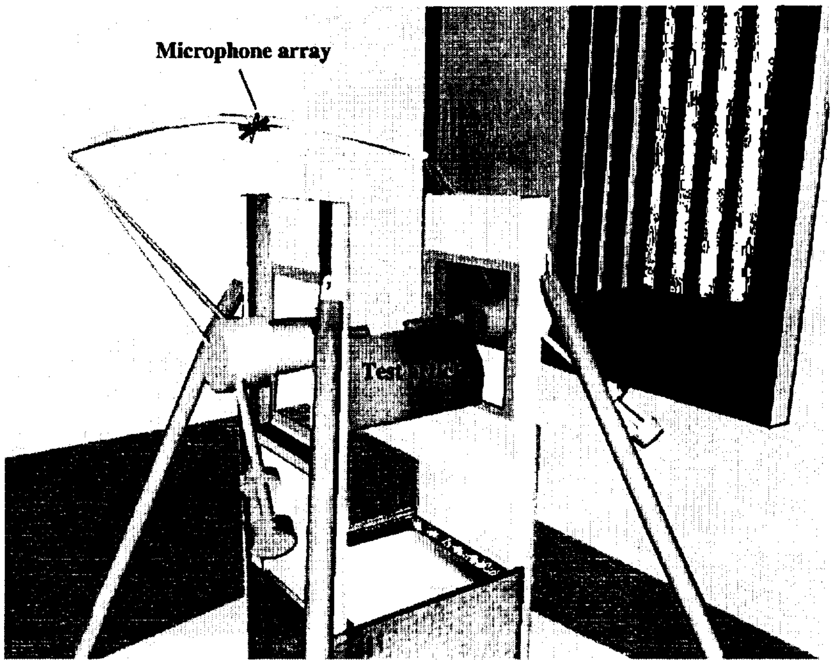

Figure 12.
Computer model of the quiet flow facility (QFF)

- Fixed (non-moving) components. These include the test chamber, frame and shell of the test section, display monitors, and control panels.
- Moving components. These include the test article, and the microphone array support structure. A flexible FE model of these components is used.
- Rotary actuators with PD tracking control are used to control the orientation of the test article and microphone array. A linear actuator controls the distance between the microphone array and the test article.
- The machine controls consist of buttons and dials. The user can manipulate the controls using the wand-based 3D selection tool, or the virtual hand.
- An output screen shows graphs of various output quantities. A menu allows choosing the graph parameters such as pressure, noise magnitude, and velocity at various positions.
- The domain is discretized using a hexahedral grid of fluid elements.

A typical acoustic test consists of the following steps:

- Close and secure the doors to the room and go through the QFF checkout and startup checklists.
- The airflow is started.
- The microphone array data is acquired at 30KHz at various positions around the test article and for various angles of attack.

- Data from pressure gages on the surface of the test article is acquired.
- The airflow is turned off.

CFD codes can be used to simulate the fluid flow around the test article by predicting the response time history of the pressure and velocity fields. The model can be color shaded using the pressure, temperature, or velocity fields. The pressure field is shaded on the surface of the test article and can be compared to the pressure observed on the actual test article coated with pressure sensitive paint. The sound generated by the test article can be generated in the VE using the four speakers. The user can listen to the simulated sound from various points around the test article.

4. Concluding remarks

The implementation of virtual labs using an object-oriented event-driven VE toolkit is described. The VLs that are being constructed at the ODU-AEE Center include, structures testing machines, wind-tunnels, and acoustic testing facilities. The VLs allow simulation of the test procedure, viewing of the results of the physical experiment, visualization of a numerical simulation of the experiment, computational steering, and remote control of the experiment (tele-presence). In addition, the VLs can be used as a risk-free training environment.

References and further reading

ABAQUS/Standard User's Manual Version 5.7 (1997), Hibbit, Karlsson & Sorensen, Inc..

AutoCAD 2000 DXF Reference (2000), AutoDESK Corp.

Ayers, M.R. and Zeleznik, R.C. (1996), "The Lego interface toolkit", *Proceedings of the 1996 ACM Symposium on User Interface and Software Technology (UIST).*

Dai, F. (Ed.) (1998), *Virtual-Reality for Industrial Applications*, Springer, New York, NY.

Dynamic Interactions Simulator (DIS) User's Manual Version 1.1 (2000), Advanced Science and Automation Corp.

Eckel G. (1997), *IRIS Performer Programmer's Guide*, Silicon Graphics, Inc..

ECMA Script Language Specification (1990), Standard ECMA-262, 3rd ed.

Foley, J., van Dam, A., Feiner, S. and Hughes, J. (1990), *Computer Graphics: Principles and Practice*, 2nd ed., Addison Wesley, Reading, MA.

Germs, R., Maren, G.V., Verbree, E. and Jansen, F.W. (1999), "A multi-view VR interface for 3D GIS", *Computers and Graphics*, Vol. 23, pp. 497-506.

Ginsberg, M. (1999), "Influences, challenges, and strategies for automotive HPC benchmarking and performance improvement", *Parallel Computing*, Vol. 25 No. 12, November, pp. 1459-76.

Hodge, J.S. and Harvin, S.F. (2000), "Test capabilities and recent experiences in the NASA Langley 8ft high temperature tunnel", Paper No. AIAA 2000-2646, 21st AIAA Advanced Measurement Technology and Ground Testing Conference, Denver, CO, June.

ISO/IEC 14772-1: 1997 Virtual Reality Modeling Language (VRML97) (1997), The VRML Consortium Incorporated.

Krist, S.L., Biedron, R.T. and Rumsey, C.L. (1998), *CFL3D User's Manual (Version 5.0)*, NASA/TM-1998-208444, June.

LightWave 3D Object File Format (1996-1999), NewTek Corp.

Macaraeg, M.G. (1998), "Fundamental investigations of airframe noise", Fourth AIAA/CEAS Aeroacoustics Conference, Toulouse, France, AIAA 98-2224, 2-4 June.

Mills, S. and Noyes, J. (1999), "Virtual reality: an overview of user-related design issues revised paper for special issue on 'virtual reality user issues', in interacting with computers", *Interacting with Computers*, Vol. 11, pp. 375-86.

MSC/DYTRAN User's Manual Version 4.0 (1997), The MacNeal-Schwendler Corporation.

MSC/NASTRAN User's Manual Version 67 (1991), The MacNeal-Schwendler Corporation.

μuSE Software Development Environment (2000), MUSE Technologies, Inc.

Nash, E., Edwards, G., Thompson, J. and Barfield, W. (2000), "A review of presence and performance in virtual environments", *International Journal of Human-Computer Interaction*, Vol. 12 No. 1, pp. 1-41.

O'Brien, J.F., Zordan, V.B. and Hodgins, J.K. (2000), "Combining active and passive simulations for secondary motion", *IEEE Computer Graphics and Applications*, July/August, pp. 86-96.

Ryken, M.J. and Vance, J.M. (2000), "Applying virtual reality techniques to the interactive stress analysis of a tractor lift arm", *Finite Elements in Analysis and Design*, Vol. 35, pp. 141-55.

Shaw, C., Green, M., Liang, J. and Sun, Y. (1993), "Decoupled simulations in virtual reality with the MR Toolkit", *ACM Transactions on Information Systems*, Vol. 11 No. 3, July, pp. 287-317; (http://www.cs.ualberta.ca/~graphics/MRToolkit.html).

Stanney, K.M., Mourant, R.R. and Kennedy, R.S. (1998), "Human factors issues in virtual environments: a review of the literature", *Presence: Teleoperators and Virtual Environments*, Vol. 7 No. 4, pp. 327-51.

Strauss, P. and Carey, R. (1992), "An object-oriented 3D graphics toolkit", *Proceedings of Siggraph '92*, Chicago, IL, 26-31 July, pp. 341-9.

Tarau, P., Bosschere, K., Dahl, V. and Rochefort, S. (1999), "LogiMOO: an extensible multi-user virtual world with natural language control", *The Journal of Logic Programming*, Vol. 38, pp. 331-53.

Walatka, P.P., Buning, P.G., Pierce, L. and Elson, P.A. (1990), *PLOT3D User's Manual*, NASA TM 101067, March.

Wasfy, T. and Noor, A.K. (2000), "Object-oriented virtual environment for visualization of flexible multibody systems", *Advances in Engineering Software*.

World Tool Kit Reference Manual, (1998), Sense8 Corporation.