AIAA 2003-5020

Development of A Virtual Testing Environment for Electric Propulsion

L.Brieda, J. Pierru, R. Stillwater, J. Wang
Computational Advanced Propulsion Laboratory
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061-0203

39th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit
July 20-23, 2003/Huntsville, AL

Development of A Virtual Testing Environment for Electric Propulsion

L.Brieda*, J. Pierru†, R. Stillwater‡, J. Wang§
Computational Advanced Propulsion Laboratory
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061-0203

Introduction

Physics based modeling is becoming an ever more important element in electric propulsion research and development activities. In recent years, particle simulation models have been developed to investigate various aspects of electric propulsion, from thruster component design to spacecraft-thruster plume interactions. The sophistication of the model and the capability of state-of-the-art supercomputers have reached to such a level that one can produce simulation results that are in good quantitative agreement with experimental data for many problems. For instance, the 3-D ion thruster plume model by Wang et al.[2001] produced results that are in excellent agreement with data from Deep Space 1 in-flight measurements and the 3-D ion optics model by Wang et al.[2003] produced results that are in excellent agreement with erosion measurements taken during the long duration tests of the NSTAR thruster.

Large-scale, 3-D simulation models generate a wealth of data output. However, in order to apply these models as engineering design tools, one must be able to quickly transform "data rich" simulation results to "information rich" results for engineering applications. To achieve this goal, one needs a data analysis environment which would satisfy at the minimum the following requirements: 1)allow easy visualization and interpretation of complex multi-dimensional data generated by particle or CFD codes; 2)enable interactive access to information from different geographic locations and online collaborations; and 3)be machine independent. A data analysis tool that satisfies these requirements in essence can be used as a "virtual" design and testing environment. This paper discusses the initial development of a virtual testing environment for electric propulsion applications.

The CAP Lab Virtual Testing Environment (capVTE), recently developed at Virginia Tech, is a cross-platform visualization and analysis tool. Visual analysis can be performed on a wide variety of hardware platforms, ranging from high-end PCs, through SUN and SGI workstations, to virtual reality facilities such as CAVE. In addition to graphics capability visualizing multi-dimensional data set, capVTE offers immersed visualization for users with access to virtual reality environments as well as a shared, collaborative environment which allows remotely connected users to interact with each other over the same data objects.

This paper is organized as follows: section II discusses the design of capVTE; section III concentrates on immersed visualization; section IV concentrates on collaborative visualization; section V presents some examples of using capVTE for electric propulsion simulations through the desktop interface (examples of immersed visualization are shown in the presentation); Section VI discusses future work and contains a conclusion.

capVTE

The primary data output produced by a particle simulation code or fluid simulation code includes distributions of scalar and vectors defined on simulation grid points, such as potential ϕ, density ρ, and velocity $\vec{V}$. For particle codes, the output also includes distributions of macro particles in physical space and phase space. CapVTE is designed to visualize the geometry, grid data, and particles generated by a particle or CFD code. For easy interpretation of complex data, capVTE allows a user to be "immersed" into the data set using virtual reality technology. In order to use capVTE as an engineering design environment, the visualization process needs to be in real-time and data-sharing needs to be as simple as possible. Hence, capVTE does not use the "extract plane, set camera view, render, wait" procedure commonly used by visualization tools. Instead, it processes multiple data frames, and plays them back to display time-dependent trends. It also includes a fully integrated network support module for data sharing.

The source code of capVTE was designed to be machine independent so that it may be run from any computing platforms. The graphic engine used by capVTE is the open-source Visualization Toolkit (VTK) library by Kitware and QT by Trolltech. Both VTK and QT are platform independent. VTK inter-

*Undergraduate Research Assistant; Student Member AIAA
†Undergraduate Research Assistant; Student Member AIAA
‡Graduate Research Assistant; Student Member AIAA
§Associate Professor; Associate Fellow AIAA; jowangvt.edu

acts with the OpenGL package, and versions of VTK are available for Windows, UNIX/Linux and Mac OS X. QT is the basis of GUI, which is used as the user interface for capVTE. QT is used to handle issues associated with cross-platform development, such as interaction with windows, user input, and the creation of timers. Therefore, deployment of capVTE for a particular platform involves only linking the source code with the VTK and QT libraries made for that specific platform.

In capVTE, graphics generation is through geometry blocks, grid blocks, and glyphs block. The geometry block is used to define object shapes and dimensions. CapVTE can also easily read object geometries generated by commercial CAD tools. Geometry blocks are specified by its nodes and their respective connectivity. Each node can be individually colored. The geometry block serves two roles. First, it simplifies the task of visually deducing results from the data set. Changes in the contour lines on a cutting plane, for instance, can be easily translated into the corresponding effect by observing their relationship to the surface geometry. Second, capVTE supports a real-time scalar gradient mode. In this mode, the coloring of the geometry is by extrapolation of color-values from the surrounding grid nodes. Hence, every surface of the geometry acts as an individual cutting planes. The geometry may be made transparent, allowing "see through" of objects.

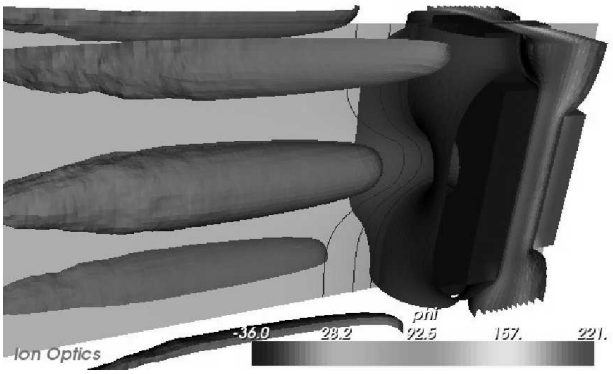

Fig. 1 View of a cutting plane, isosurfaces and geometry scalar gradient.

The grid block is used to visualize spatial distributions of grid variables. The grid data is defined on a Cartesian mesh. Grid data are visualized using cutting planes and isosurfaces (see, for example, Figure 1). In conjunction with geometry blocks, the grid data can be used to generate a scalar gradient along the geometry surface.

The glyphs block specifies small objects that can be placed anywhere in the domain. The geometry representing each glyph can be one of the many predefined types, ranging from point clouds through wedges and cubes to cones and spheres of varying resolution. Each glyph can be scaled and oriented according to the vector data. The number of glyphs can vary from frame to frame. Glyphs have many applications. One application of the glyphs is the representation of macro-particles used in the PIC simulation or vector fields defined on grid nodes. An example of the use of glyphs and a cutting plane is given in Figure 2.

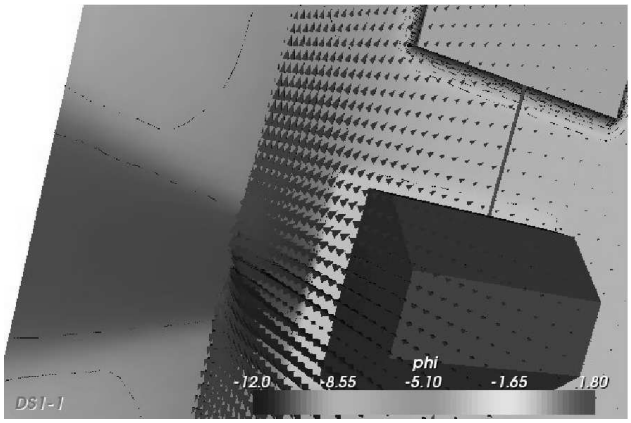

Fig. 2 View of glyphs, a cutting plane and an object geometry.

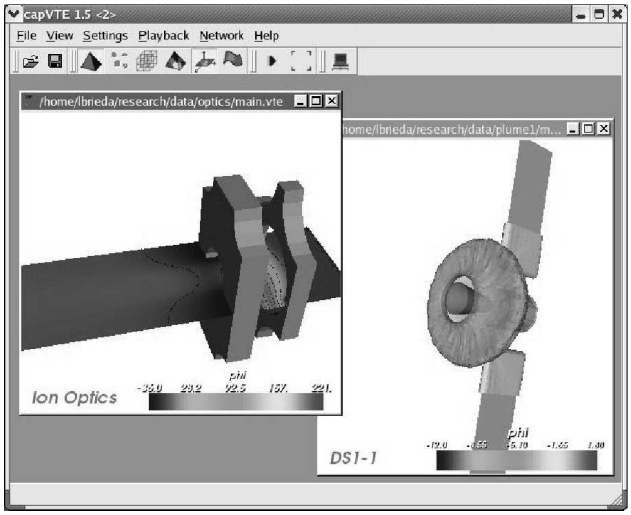

Fig. 3 capVTE user interface, showing two loaded datasets

The user interface of CapVTE, as shown in Figure 3, is a multi-window GUI application. The user navigates through each data set in real-time using the mouse. Menus and toolbars are used to select the active tools and to change their properties. The data input consists of a single ASCII file which contains the appropriate header, followed by program commands and data blocks. However, the *INSERT* command allows for the dataset to span over several physical files, in a fashion similar to that of C/C++'s *#include* directive. The main input file can be a just a short collection of *INSERT* statements pointing at a geometry file created from an output of a 3D modeling application, and the grid and particle files, generated by particle codes or CFD codes.

CapVTE allows a data export in three formats: JPEG, VRML (Virtual Reality Markup Language), and Inventor. JPEG is used to save images from static snapshots. Both VRML and Inventor are used for interactive visualization of 3-dimensional images. Unlike the JPEG format, VRML and Inventor capture the polygons making up isosurfaces, cutting planes, glyphs, geometry, etc, on display and allow a user to navigate through 3-dimensional graphics. As VRML is commonly used to publish 3D data on the Internet and plugins are available for the popular browsers, capVTE uses the VRML format for online collaborative visualization through network. Inventor is compatible with DIVERSE, a software used to run the virtual reality facility CAVE. CapVTE uses the Inventor format for immersive visualization. Details of immersive and collaborative visualization will be discussed further in the next two section.

For interactive visualization, the grids and glyphs are loaded as a collection of frames. In the playback mode, the program automatically advances the current frames. Cutting planes and isosurfaces are automatically updated and glyphs are moved to their new position. For visualization with cutting planes and isosurfaces generated on a grid with several thousand nodes, the playback rate is about several frames per second on a typical PC, sufficient for real-time navigation of images. The record mode exports a bitmap image at a chosen interval. These pictures can then be converted into a movie using software such as Adobe Premier or Flash MX.

Immersive Visualization

CapVTE offers the capability of immersed visualization. Immersed visualization currently is performed using the DIVERSE software to drive a variety of immersed visualization equipment such as the CAVE (Cave Automatic Virtual Environment), Immersa Desk, Head Mounted Display (HMD), etc. DIVERSE can also be used to drive visualization on conventional desktop or laptop computers.

CAVE

The CAVE[1] is a multi-person, room-sized, high-resolution, 3D video and audio environment used as a "virtual reality theater". The CAVE facility at Virginia Tech, shown in Fig 4, has a size of 73.5x84 inch footprint and a 104-inch high ceiling. The three side faces along with the floor act as screens, on which an image is projected from four Electrohome Marquis 8000 projectors. The false-color images are shifted such that, when viewed using the CrystalEyes goggles, they create a perception of the visualized object floating in the center of the cube.

Inside the CAVE, one "walks" into an image. Navigation is through a tracking headset. Its position and view angle is tracked by an array of sensors. The central computer then adjusts the projections to create a realistic image. Further control can be accomplished using a hand-held wand. Its translation as well as rotation are both tracked, and thus the wand can act as a virtual pointer. A small joystick located on the wand allows the user to travel greater distances than that allowed by the limited physical size of the CAVE.

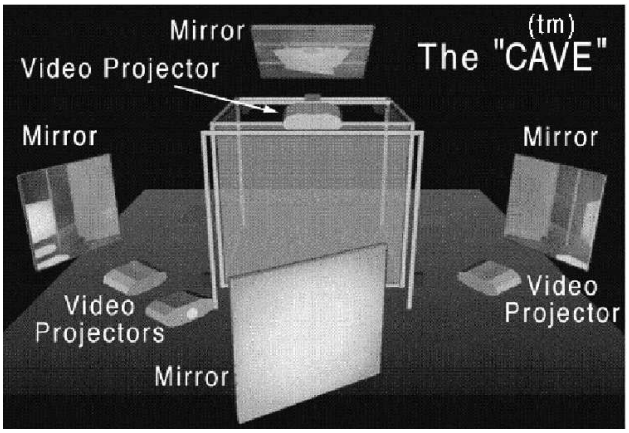

Fig. 4 The Virginia Tech CAVE facility

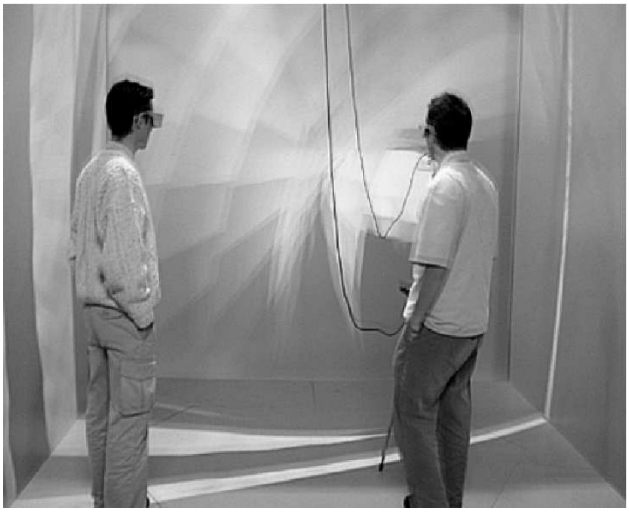

Fig. 5 Immersive Visualization inside the CAVE

DIVERSE

DIVERSE[1] (Device Independent Virtual Environment Reconfigurable Scalable Extensible) is a collection of Application Programming Interfaces (APIs) developed at Virginia Tech by the University Visualization and Animation Group. It is written in C++ and runs on GNU/Linux and SGI IRIX systems. Diverse is a free software distributed under the licenses LGPL for the libraries and GPL for the software. DIVERSE consists of three major components:

- Diverse graphics interface for Performer or DPF, uses OpenGL Performer to implement 3D Virtual Environment applications.

[1]Available at http://diverse.sourceforge.net

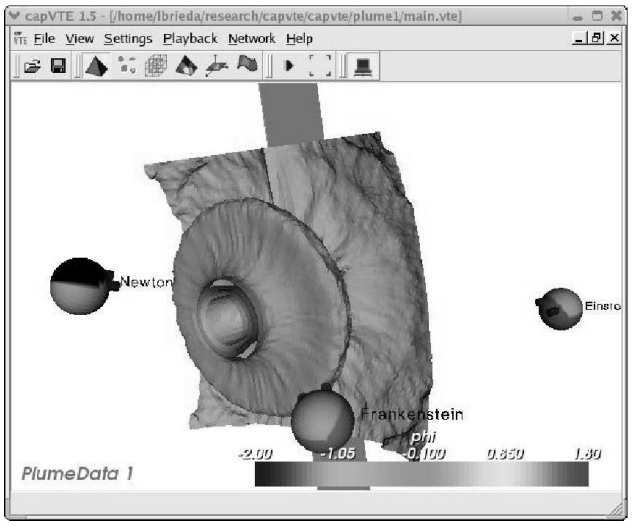

Fig. 6 Network collaboration showing three connected clients.

- Diverse GL or DGL, supports rendering with the OpenGL programming interface.
- Diverse ToolKit or DTK, allows DPF and DGL to access local and networked interaction devices.

The flexibility of DIVERSE comes from its use of dynamically-loaded shared objects(DSO's). The DIVERSE program provides the general framework, onto which additional objects can be loaded as needed.

Collaborative Visualization

The real-time data immersion capability of capVTE allows the creation of a collaborative virtual environment on internet. CapVTE includes a network module, based on a client/server and TCP/IP architecture. The network module simplifies data sharing and allows interactive visualization by remotely connected users. The user specifies whether a new server should be established, or whether the program should connect as a client to an already running collaboration. Each user specifies an "avatar", a small graphical object, which is located and oriented according to that user's viewport. Thus, every member of the online collaboration can see what part of the dataset their colleagues are looking at, as illustrated in Figure 6. The network is also partially synchronized. When one client toggles cutting planes, for instance, cutting planes will be toggled for all other clients as well. Thus, all members share the same virtual space with the only difference being the position from which the view is made. Some synchronization problems may occur, however, if the hardware performance of the used machines varies widely. This synchronization problem is being resolved.

Visualization Examples

This section shows a few visualization examples of snapshot images from the desktop interface of capVTE. Examples of immersed visualization using the CAVE will be discussed in the presentation.

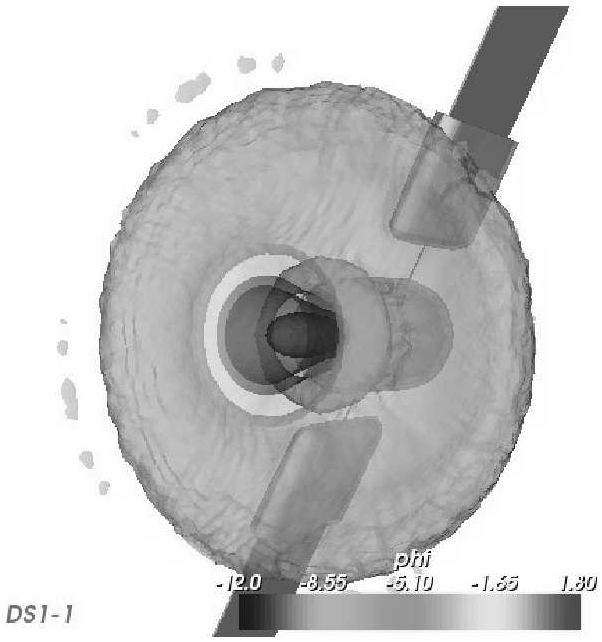

Fig. 7 Deep Space 1 ϕ isosurfaces.

Figure 7 is a visualization of ion thruster plume simulation. The data is taken from the 3-dimensional, particle-in-cell simulations of ion thruster plume environment for Deep Space 1 (DS1) presented in Wang et al.[2001]. The plot shows 3-dimensional potential isosurfaces surrounding DS1. Unlike cutting planes, which limits analysis of data to a two dimensional plane and may miss important features hidden in the data due to the choice of cutting plane, the isosurface tool connects equal scalar values through the entire domain.

Figure 8 shows the flow field of macro-particles representing charge-exchange ions. The flow field is generated by averaging the velocity of individual macro-particles onto the grid nodes. Scaled and oriented glyphs are then used to indicate the flow direction and magnitude. To reduce the visual clutter of the resulting animation, Fig. 8 only shows visualization on a single plane with a location of 11 computational cells downstream from the thruster exit. The top panel shows charge-exchange plasma expansion during the initial stage of the simulation, and the bottom one shows the steady state. The expansion of charge-exchange plasma is visible from the outward orientation of the glyphs.

Figure 9 is a visualization of ion optics simulation. The data is taken from the 3-dimensional, particle-in-cell simulations of ion optics plasma flow for the NSTAR ion thruster presented in Wang et al.[2002]. The plot shows 3-dimensional potential isosurfaces. Potential contours on a plane orthogonally to the optics grids and cutting through the aperture are also shown. The cutting plane has a 50% opacity.

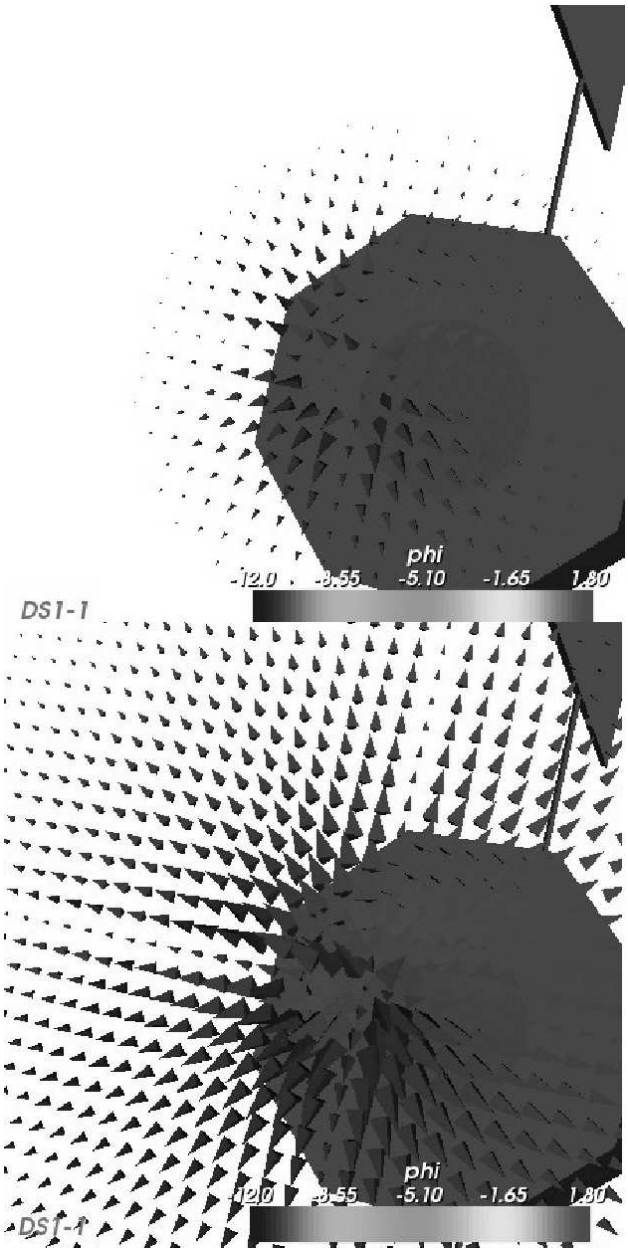

Fig. 8 Averaged flow velocity from early stage to steady state.

Figs. 7 through 9 only show the images as viewed from a 2-D computer monitor. Full three dimensional data immersion can be accomplished by exporting the geometry/isosurface combination of these figures to an Inventor file. The Inventor file is then loaded into the CAVE using DIVERSE for immersive visualization. The same data can also be exported to a VRML file, allowing collaborative visualization through internet.

Conclusion

In conclusion, a new cross-platform visualization and data analysis environment, capVTE was developed. capVTE can be used on a variety of hardware platforms, ranging from high-end PCs, through SUN and SGI workstations, to virtual reality facilities such as CAVE. capVTE offers immersed visualization for users with access to virtual reality environments as well as a shared, collaborative environment which allows remotely connected users to interact with each other over the same data objects. Hence, capVTE may be used as a virtual testing environment for electric propulsion applications.

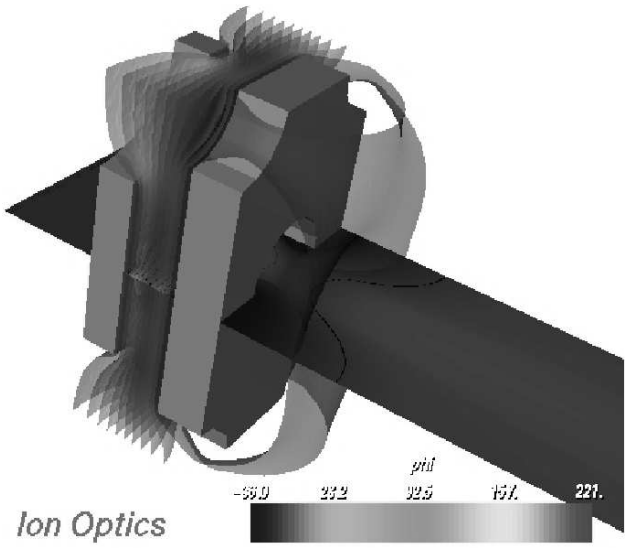

Fig. 9 Deep Space 1 Ion Optics.

The current version of capVTE already contains most of the designed features. Several remaining software issues will be resolved in future work. First, an extensive testing needs to be carried out to verify its stability in the cross-platform environment. Second, the network interface needs to be optimized. Currently, the central server relays the messages pertaining to the state of each client as received. This results in a large network traffic. The network interface will be redesigned so the server collect the state information, and communicate it at a preset interval in a single network packet. This will also improve the overall synchronization of the clients. Further, the network communication relies on open sockets. Open sockets can be a potential security risk, and could prevent clients behind a strong firewall from connecting to other capVTE users. Thus, the possibility of implementing a secured connection using the OpenSSL package is being investigated. Dialog window will be added allowing users to send text messages across the network, as well to "jump" into somebody else's viewport, and thus share the view of the dataset with their collegeau.

References

1. Arsenault, L.E. et al., "DIVERSE: a Software Toolkit to Integrate Distributed Simula-

tions and Heterogeneous Virtual Environments", Joint Aerospace Weapon Systems: Support, Sensors, and Simulations Symposium Exhibition, San Diego, CA, July 22-27, 2001

2. Brieda, L. and Pierru, J., "Development of a Virtual Testing Environment and Applications to Spacecraft Electric Propulsion", AIAA Regional Student Conference, University of Maryland, MD, 2003

3. Virginia Tech Cave. "What is the Cave?". 1998.<http://www.sv.vt.edu/future/vt-cave/whatis/> (June 25, 2003)

4. Wang, J., et al., "Three Dimensional Particle Simulations of Ion Optics Plasma Flow and Grid Erosion", accepted for publication in *J. Propulsion & Power, 2003*

5. Wang, J., et al., "Deep Space 1 Investigation of Ion Propulsion Plasma Environment", *J. Spacecraft & Rockets, 37(5), pp545*