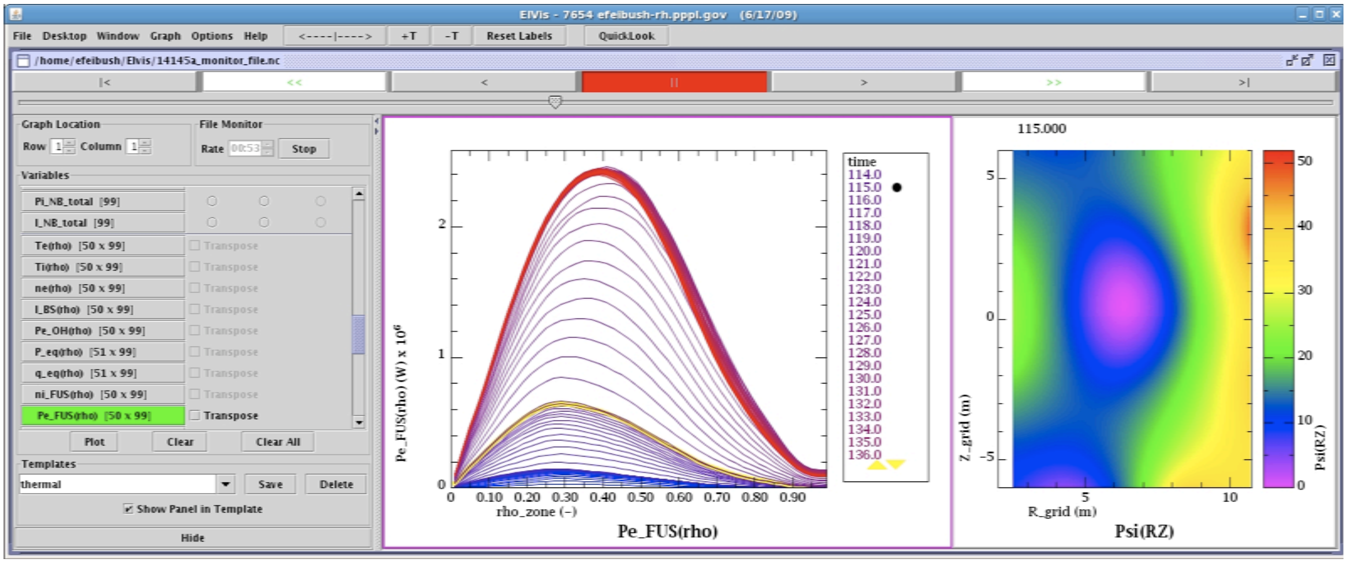
# Visualization Techniques for Monitoring Simulations

Eliot Feibush, *Member, IEEE*



ElVis - 7654 efeibush-rh.pppl.gov (6/17/09)

/home/efeibush/Elvis/14145a_monitor_file.nc

Graph Location: Row 1 Column 1

File Monitor: Rate 00:53

Variables:
- Pi_NB_total [99]
- I_NB_total [99]
- Te(rho) [50 x 99] □ Transpose
- Ti(rho) [50 x 99] □ Transpose
- ne(rho) [50 x 99] □ Transpose
- I_BS(rho) [50 x 99] □ Transpose
- Pe_OH(rho) [50 x 99] □ Transpose
- P_eq(rho) [51 x 99] □ Transpose
- q_eq(rho) [51 x 99] □ Transpose
- ni_FUS(rho) [50 x 99] □ Transpose
- Pe_FUS(rho) [50 x 99] □ Transpose

Templates: thermal

☑ Show Panel in Template

Fig. 1. Visualizing *f*(x,t) and *f*(x,y,t) data for monitoring a fusion simulation. The graph of Pe_FUS(rho) shows all the *f*(x,t) time step curves in a single plot (center panel). A color gradient from blue to red is applied to the curves to convey the time ordering in a single graph. Time index 115. has been selected in the legend and highlighted in the plot. The yellow triangles at the bottom of the legend are user interface controls for scrolling the list of time indices. The *f*(x,y,t) data for Psi(R,Z) is downloaded and visualized in the client program (right panel) so it can be explored interactively. All available variables are listed in the control panel (left).

**Abstract**—Visualization techniques for time indexed data are described for monitoring long running simulations. Approaches that emphasize data exploration and interaction have been developed for simulation data that can be downloaded to the client. The data is stored in a netCDF file on the server and automatically monitored by the client. When larger amounts of data are produced the visualization is computed on the server side and an image is rendered for each time step. The client displays the images in sequence to show an animation of the simulation. The netCDF files and the images are stored in HTTP served directories so scientists can monitor long running simulations from anywhere on the Internet.

**Index Terms**—Data visualization, simulation/analysis monitoring, scientific computer graphics, interactive data exploration.

## 1 Introduction

Monitoring long running simulation and analysis programs is important for conserving computer resources and human time. Data typically evolves over the time base of the simulation program. Periodically checking the validity of the output data as the simulation runs improves efficiency. An errant run can be stopped or possibly steered without having to let it run to completion.

Our goal is to enable scientists to monitor their run from anywhere on the Internet. Given the long run time and variable start time it is necessary to monitor from any location. The visualization client program has been written in Java for portability across platforms while maintaining only one version of source code. It can be easily downloaded and installed on a personal computer.

## 2 Related Work

The motivation for monitoring is described in [1] along with a complex, client-server architecture for processing and transmitting large amounts of data to a single, remote monitoring station. Another approach to monitoring [2] utilizes the licensed AVS software to display the computed data.

A framework for accessing monitor data over the web is described in [3]. The architecture is designed for a visualization client that downloads data from an HTTP server and renders on the user's local computer. The details of the visualization client, ElVis, are described here.

## 3 Monitoring Variables with Graph Templates

The ElVis display client automatically monitors variables stored in netCDF files. These structured data files contain variables and attributes. The API for reading the data includes specifying the source as a URL or a local file [4]. Storing the netCDF file in an HTTP served directory makes the data readable over the web. This simplifies the overall system architecture because it does not require a specialized server program or protocol and leverages the HTTP server. The client simply checks the URL for new data. This

- *Eliot Feibush is with the Princeton Plasma Physics Laboratory, E-Mail: efeibush@pppl.gov.*

*Manuscript received 1 July 2009.*
*For information on obtaining reprints of this article, please send email to: tvcg@computer.org .*

facilitates collaboration among users at different locations because the netCDF file can be served to multiple clients at the same time.

Variables are defined with a specific number of samples per dimension plus an “unlimited” dimension that corresponds to simulation time steps. At the end of each time step the simulation appends data to the variables so they grow in the unlimited dimension. If the simulation produces very large amounts of data then a representative subset must be extracted for monitoring. ElVis periodically reads the netCDF URL, checks for new data, and adds it to the visualizations. A key advantage of this approach is that any data source that writes data in this style can be automatically monitored and visualized.

The polling rate is programmatically adjusted to the production rate. If no new data is found then the rate is decreased. When new data is read the rate increases until monitoring catches up with production. A session can start at any time during a run and show all accumulated data. A completed run can be reloaded for playback.

An output file can contain hundreds of different variables which could lead to dozens of graphs. One approach to organizing a large number of graphs is the “filmstrip” or “slide” layout shown in Fig. 2. Users are familiar with this paradigm for organizing slides and digital photos so it is easily learned when applied to graphs. The graph compositions are stored in a template that describes the contents of each graph, such as the variables to plot, choice of presentation style, and annotation. The template accomplishes visualizing and comparing new runs and automatically reloading the variables in the desired appearance. This approach was requested by physicists using ElVis for monitoring fusion simulation programs.

Having the data locally enables exploration entirely in the client to achieve fast interaction. A new technique was developed for controlling the animation of *f*(x,t) variables: as the crosshair for the digital readout is moved across a scalar *f*(x) plot vs. time, the animation of *f*(x,t) advances to the closest time step as in Fig. 3. The user can simultaneously examine the values in the scalar plot and see the curves for the corresponding time.

## 4 Monitoring with Sequences of Rendered Images

A complementary approach to computing visualizations in the client is to render images on a visualization server. This approach is well suited to simulations that produce very large amounts of data which are impractical to transmit over the web. Instead, an image is rendered on the server side as each time step is computed. A typical image sequence is shown in Fig. 4. This approach offers considerable flexibility to visualization. Any visualization program that produces an image file on the server can be part of the system. Storing each image in a web served directory enables the client to access it as a URL from anywhere on the Internet. ElVis checks for new images and displays them in sequence. Users can scroll and zoom into the images during playback. Our experience is that informative images (in terms of size and content) can be downloaded fast enough to achieve animation. When the simulation is complete the images are recalculated based on the global range of all the data. Then the images are combined into one movie file that can be downloaded for playback. A portable, command-line program was developed for creating movie files to avoid copying all the image files to another platform and running an interactive movie maker.

## References

[1] J. Insley, M. Papka, S. Dong, G. Karniadakis, and N. Karonis. Runtime Visualization of the Human Arterial Tree. IEEE Transactions on Visualization and Computer Graphics, 13:810–821, July/August 2007.

[2] T. Watanabe, E. Kume, and K. Kato. Simulation Monitoring Using AVS., volume 2329, pages 990-999. ICCS, 2002.

[3] D. Aswath, M. Thompson, M. Goode, X. Lee, and N. Kim. Secure Federated Light-weight Web Portals for FusionGrid. Second International Workshop on Grid Computing Environments, 2006.

[4] John Caron, *NetCDF-Java (version 2.2) User’s Manual*, www.unidata.ucar.edu, 2004.

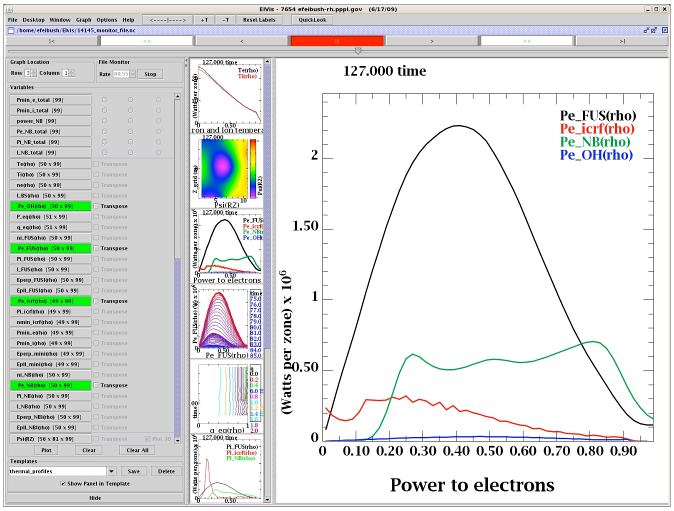
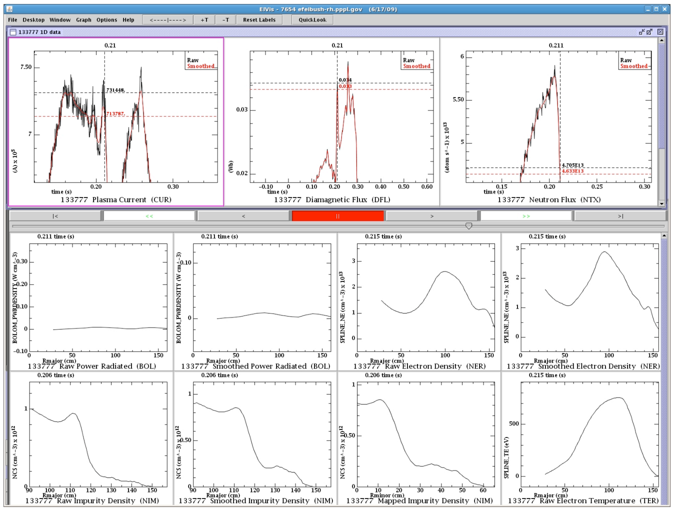


Fig. 2 A “filmstrip” layout is useful as the number of graphs increases. The user can scroll through the thumbnail views and select a graph to display in the main panel. The graph is composed by choosing variables listed in the control panel on the left.



ElVis - 7654 efeibush-rh.pppl.gov (6/17/09)
File Desktop Window Graph Options Help
Reset Labels
QuickLook
133777 1D data
Raw
Smoothed
133777 Plasma Current (CUR)
133777 Diamagnetic Flux (DFL)
133777 Neutron Flux (NTX)
133777 Raw Power Radiated (BOL)
133777 Smoothed Power Radiated (BOL)
133777 Raw Electron Density (NER)
133777 Smoothed Electron Density (NER)
133777 Raw Impurity Density (NIM)
133777 Smoothed Impurity Density (NIM)
133777 Mapped Impurity Density (NIM)
133777 Raw Electron Temperature (TER)
Rmajor (cm)
time (s)

Fig. 3 As the *f*(x) data is examined with the digital readout in the top row, the animation is simultaneously advanced to the closest time step for the *f*(x,t) graphs in the second and third row. This technique was requested by fusion scientists.

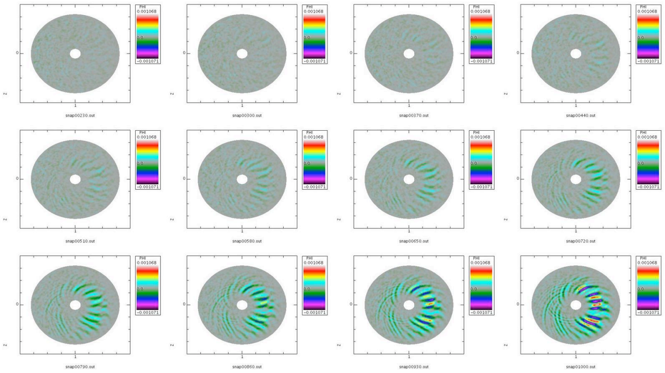


Fig. 4 An image is rendered on the server as each timestep is computed. The monitoring client checks for new image URLs and displays them sequentially to achieve an animated movie over the web.