# PPPL Fall/Winter CO-OP 2008-09

**Savraj Deep**

# Focus of the CO-OP

**The co-op focused on improving various graphical elements of ElVis, an application for the visualization and monitoring of scientific data.**

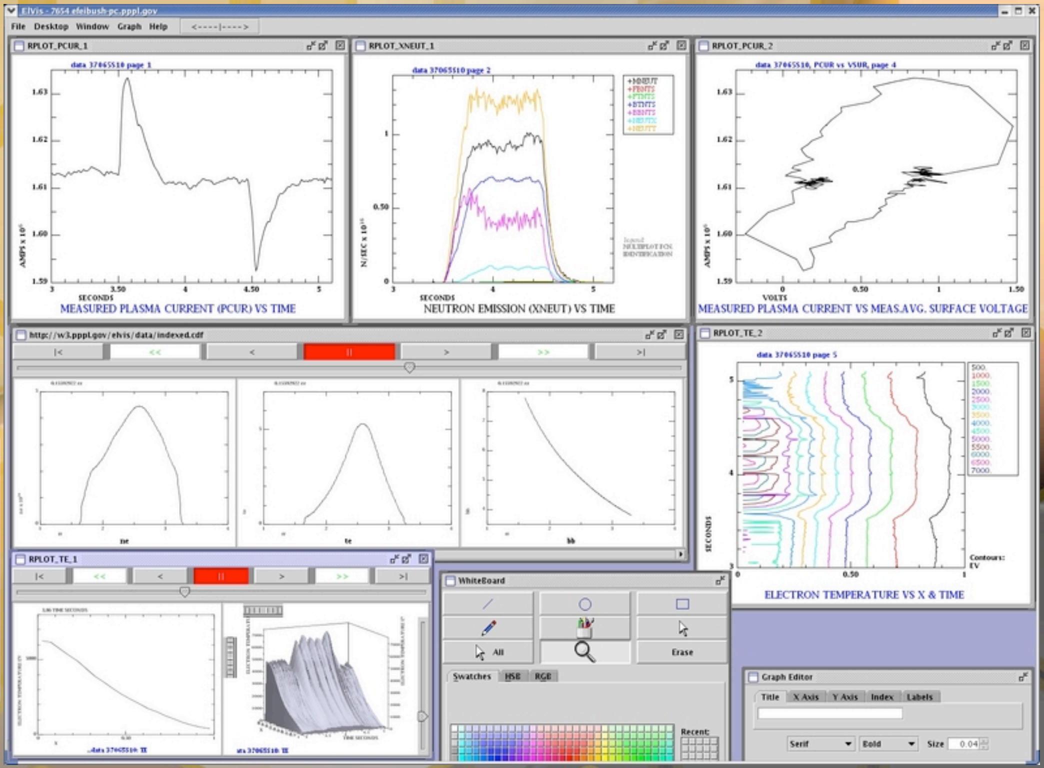

ElVis - 7654 efeibush-pc.pppl.gov
File Desktop Window Graph Help
RPLOT_PCUR_1
data 37065510 page 1
AMPS x 10^6
SECONDS
MEASURED PLASMA CURRENT (PCUR) VS TIME
RPLOT_XNEUT_1
data 37065510 page 2
N/SEC x 10^15
SECONDS
NEUTRON EMISSION (XNEUT) VS TIME
RPLOT_PCUR_2
data 37065510, PCUR vs VSUR, page 4
AMPS x 10^6
VOLTS
MEASURED PLASMA CURRENT VS MEAS.AVG. SURFACE VOLTAGE
http://w3.pppl.gov/elvis/data/indexed.cdf
ne
te
bb
RPLOT_TE_2
data 37065510 page 5
SECONDS
Contours: EV
ELECTRON TEMPERATURE VS X & TIME
RPLOT_TE_1
WhiteBoard
All
Erase
Swatches
HSB
RGB
Recent:
Graph Editor
Title
X Axis
Y Axis
Index
Labels
Serif
Bold
Size
0.04

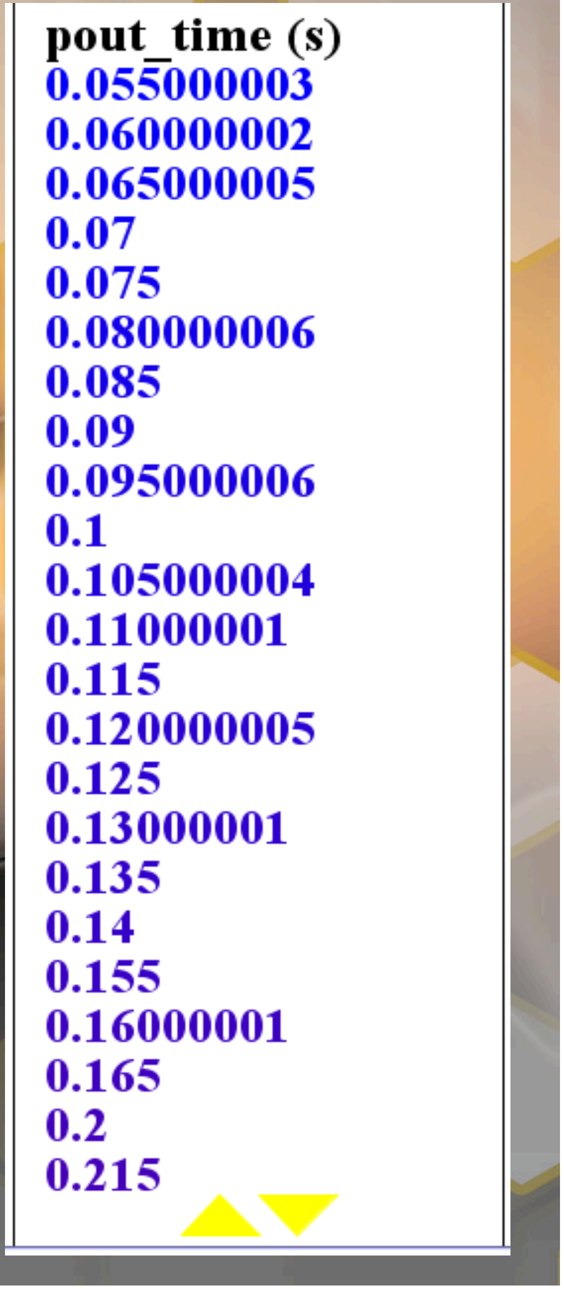
**Fig 1. A Snapshot of ElVis**

# Areas of Attention

- **Fixing legend, Graph Panel resizing and other issues**
- **Plotting multiple animated f(x.t) variables simultaneously**
- **ColorMap smooth shading**
- **Saving ColorMap legend preferences**
- **Implementing filmstrip view**
- **Explore potential OpenGL implementation of surface maps**
- **Installing NVIDIA's CUDA (Compute Unified Device Architecture) libraries.**

# Fixing legend and other issues

- The legend had various issues, such as a dysfunctional scroll and erroneous movement around the graph panel.
- The graph panels had resizing problems and failed to recognize certain movements that had to be fixed.
- These issues were fixed to make the legend and Graph Panels move as desired.
- The user is now able to select the desired background color for the Graph Panels.
- The PDF file that prints the graphs now reads in the new colors and prints out the graph in the desired format.



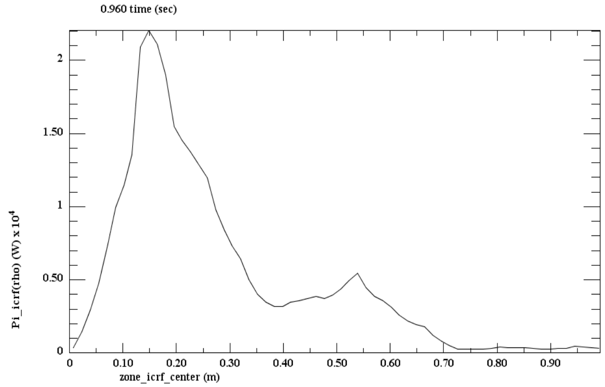
pout_time (s)
0.055000003
0.060000002
0.065000005
0.07
0.075
0.080000006
0.085
0.09
0.095000006
0.1
0.105000004
0.11000001
0.115
0.120000005
0.125
0.13000001
0.135
0.14
0.155
0.16000001
0.165
0.2
0.215

▲▼

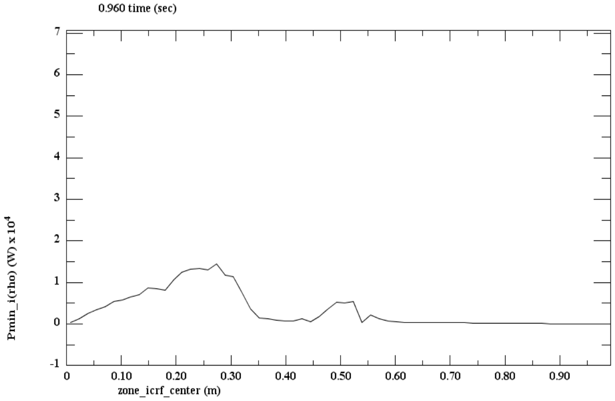
# Multiple animated f(x,t) data plotting

- Elvis could previously load just one f(x.t) data variable and did not draw two or more of these on the same Graph Panel when selected.
- Plotting of multiple f(x,t) data was implemented which could be animated as well.
- To implement this, multiple f(x,t) data of the same type are selected and plotted. The legend displays the different data points as different colours that can be highlighted and selected.
- The data can be animated as in the case of a single f(x,t) data variable. In this case, multiple data can be animated simultaneously.

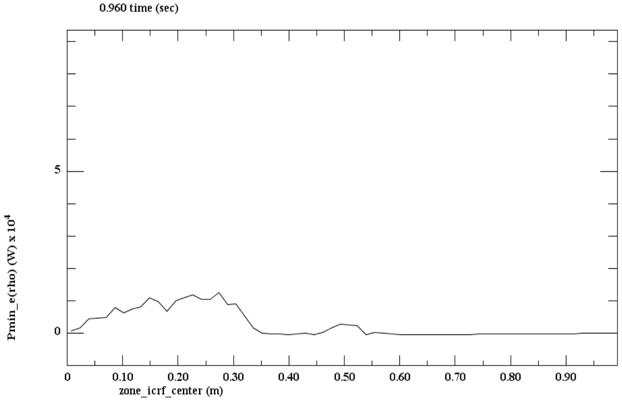
# Plotting Multiple f(x,t) Variables



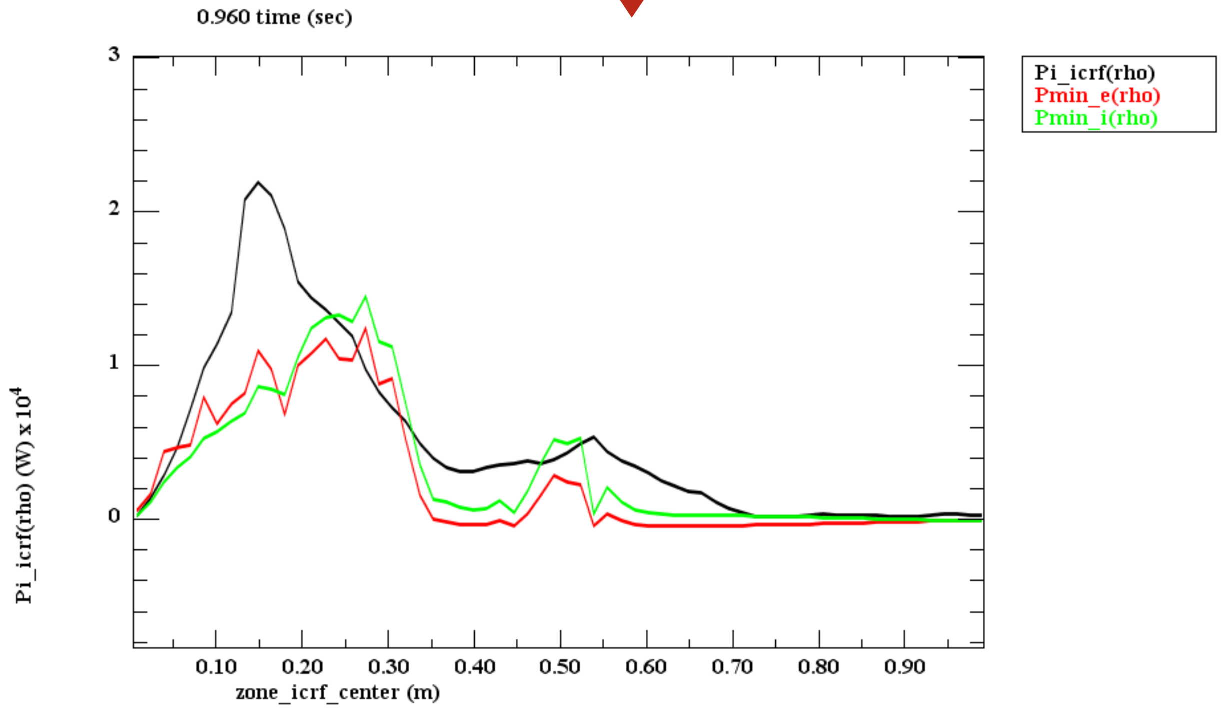
Pi icrf(rho)



Pmin_i(rho)



Pmin_e(rho)

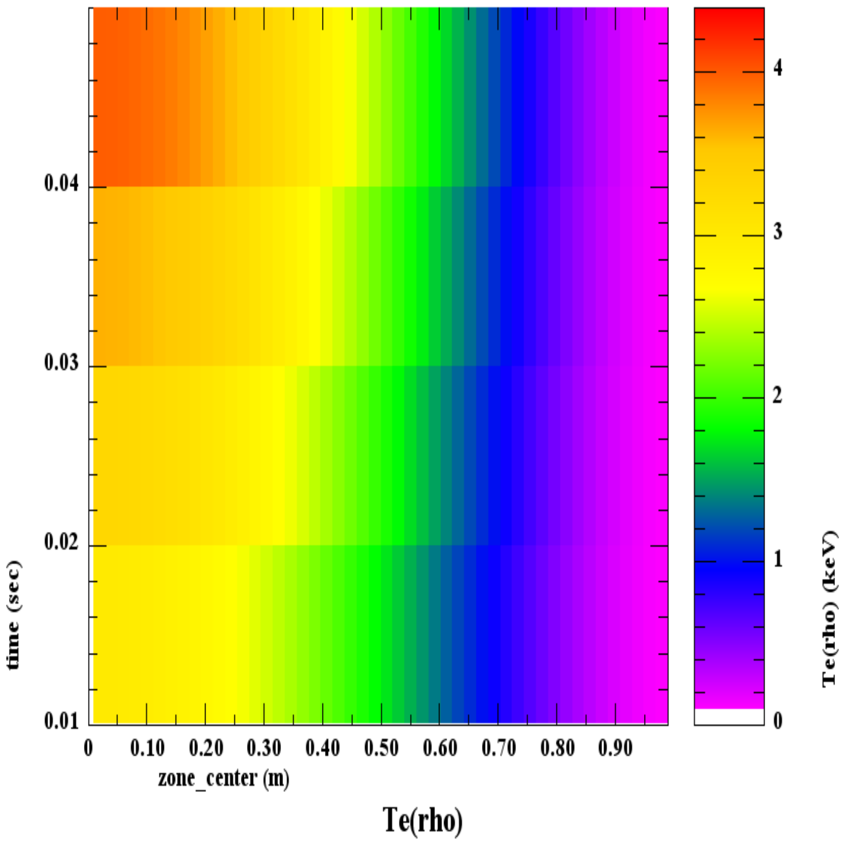


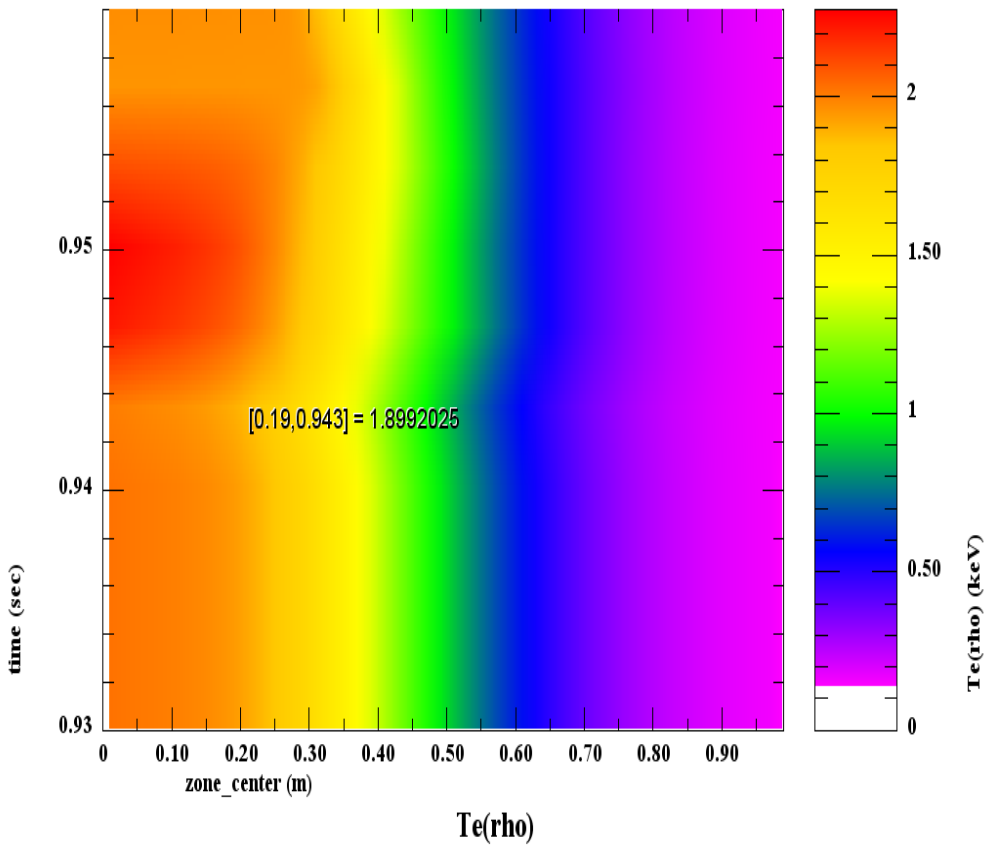
Multiple 2D Parameters Plotted

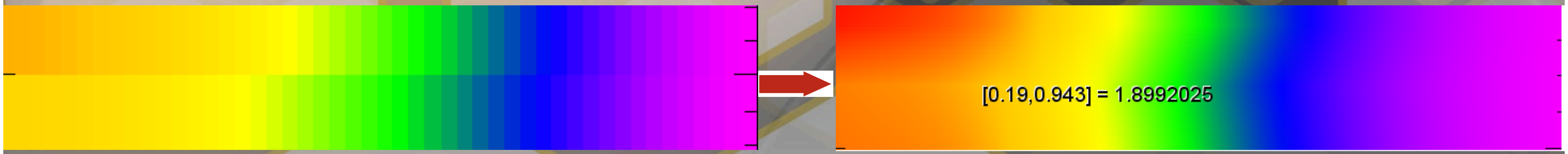
# ColorMap Smooth Shading

- The resolution of ColorMaps was increased by liner interpolation
- Issues with respect to loading of data in ColorMap plots were fixed
- The resolution or degree of interpolation of the new ColorMaps can now be increased or decreased as desired, whereby the optimal balance between performance and speed can be achieved.
- The color scheme that the users select for a particular ColorMap graph can now be saved permanently when the graph is saved.

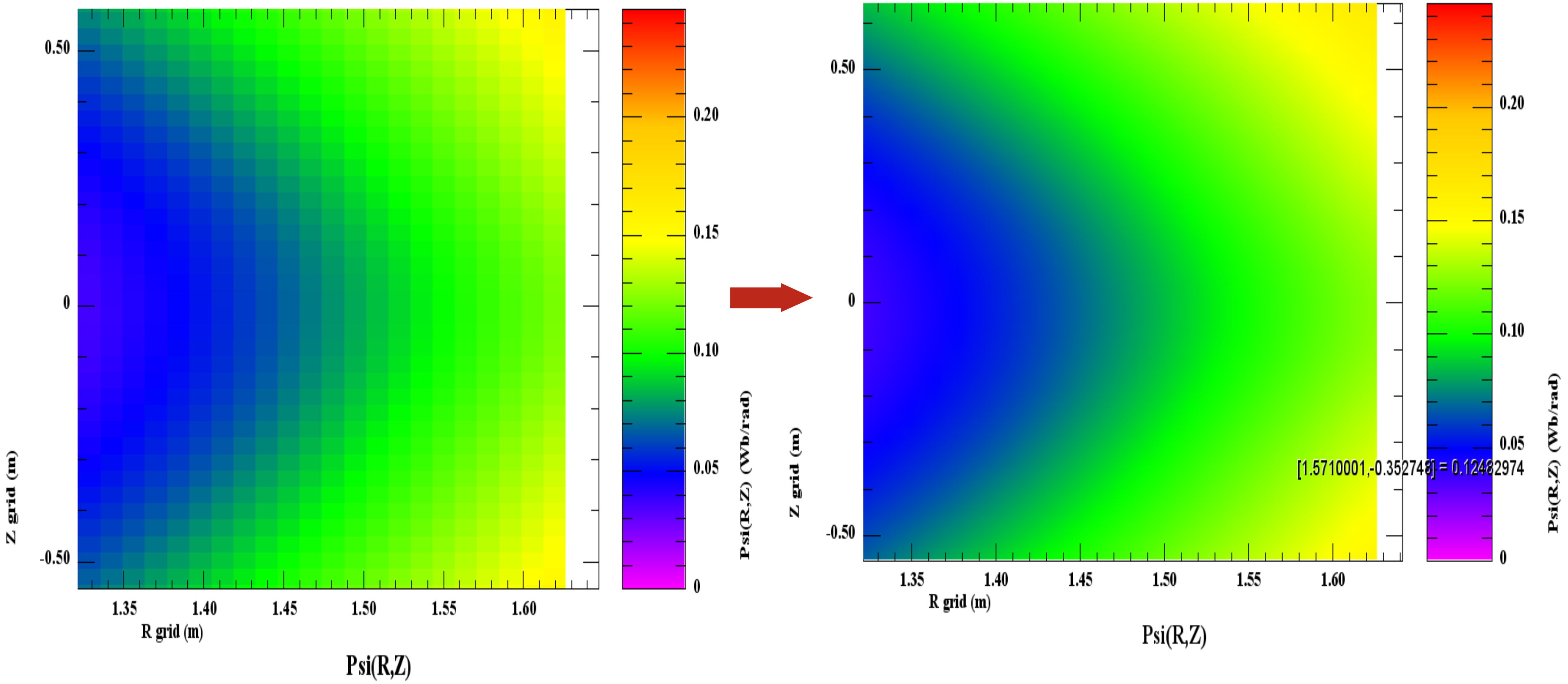
# ColorMap Comparison





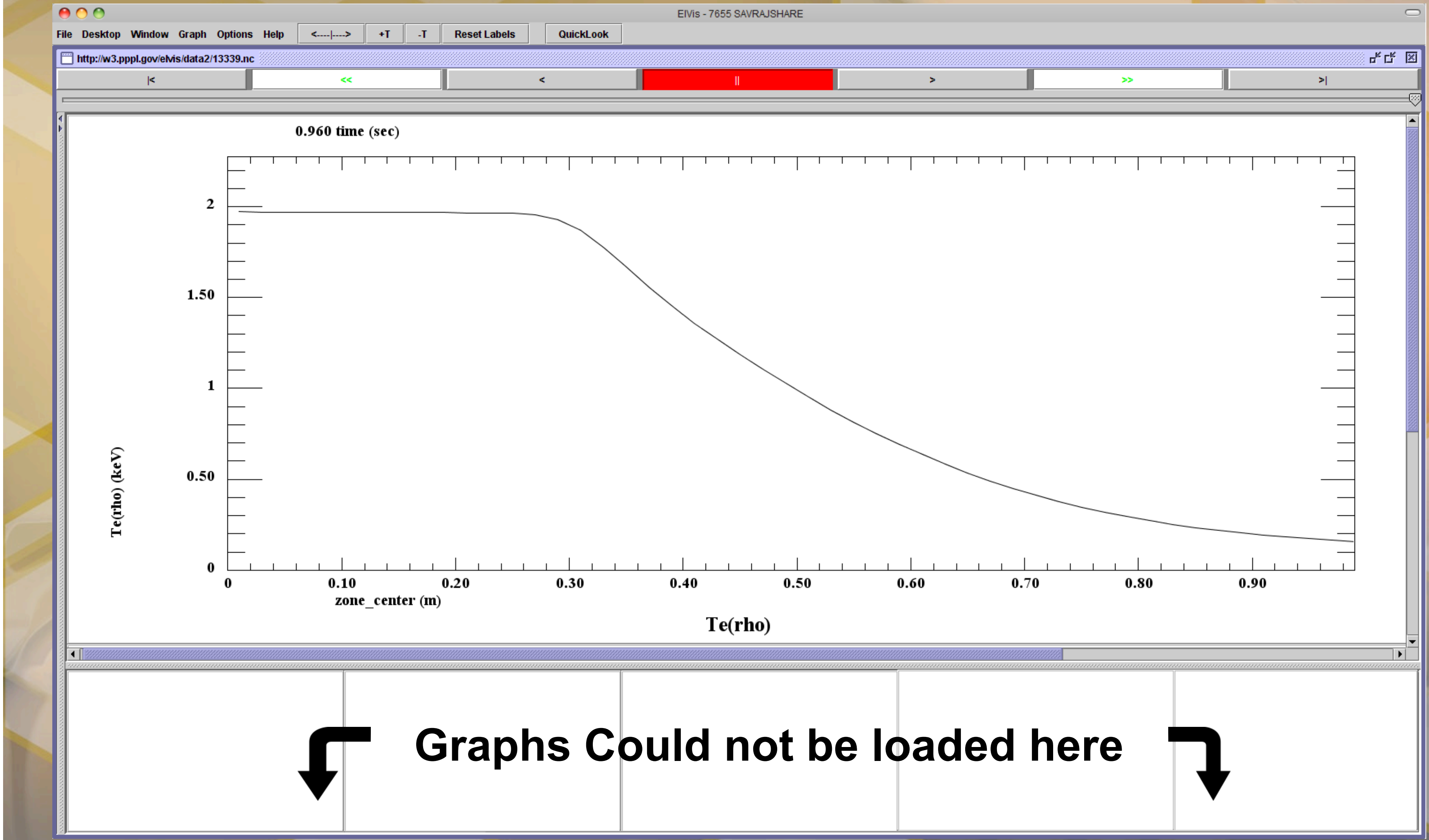


# ColorMap Comparison 2



# Implementing filmstrip view

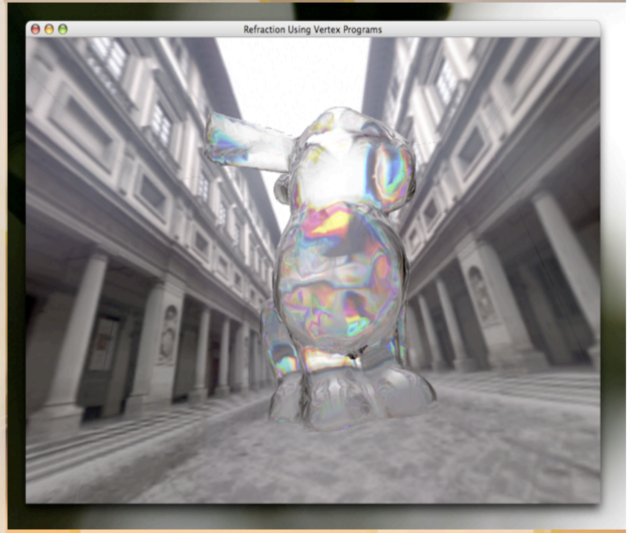
- Filmstrip View was proposed. The framework was laid out but we couldn't get the graphs to load in the lower part of the screen.

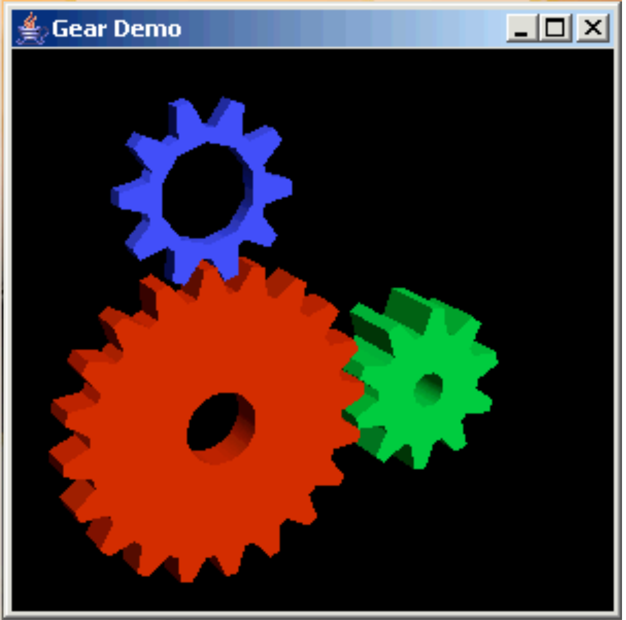


Graphs Could not be loaded here

# OpenGL (JOGL) – What is it ?

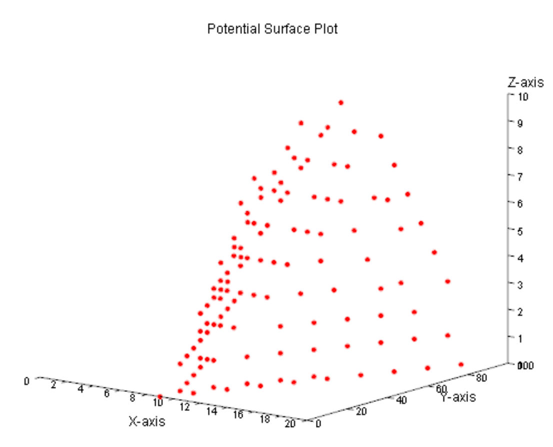
- JAVA OpenGL (JOGL) is a wrapper library that allows OpenGL to be used in the JAVA programming language.
- Allows access to most features available in the C programming language.
- SUN supported.

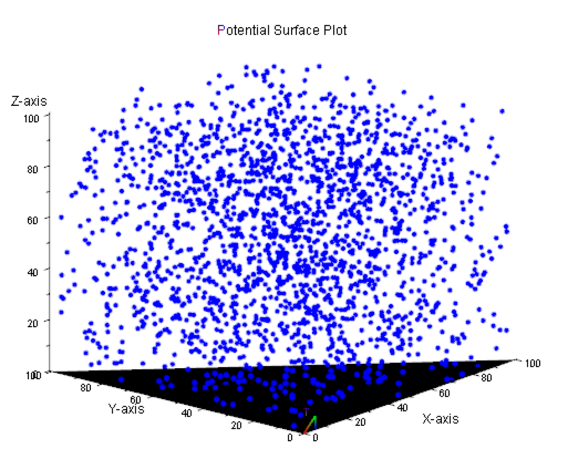




# JOGL for surface plot rendering

- Explored potential implementation of Java Open Graphics Library for implementation of surface plots.
- The present implementation uses expensive software computation to render the plots.
- Using JOGL would shift the weight onto faster hardware rendering resulting in faster loading and rendering of surface plots.
- Various open source platforms already exist that could help with this. ZG3D is one of the examples that was looked at.

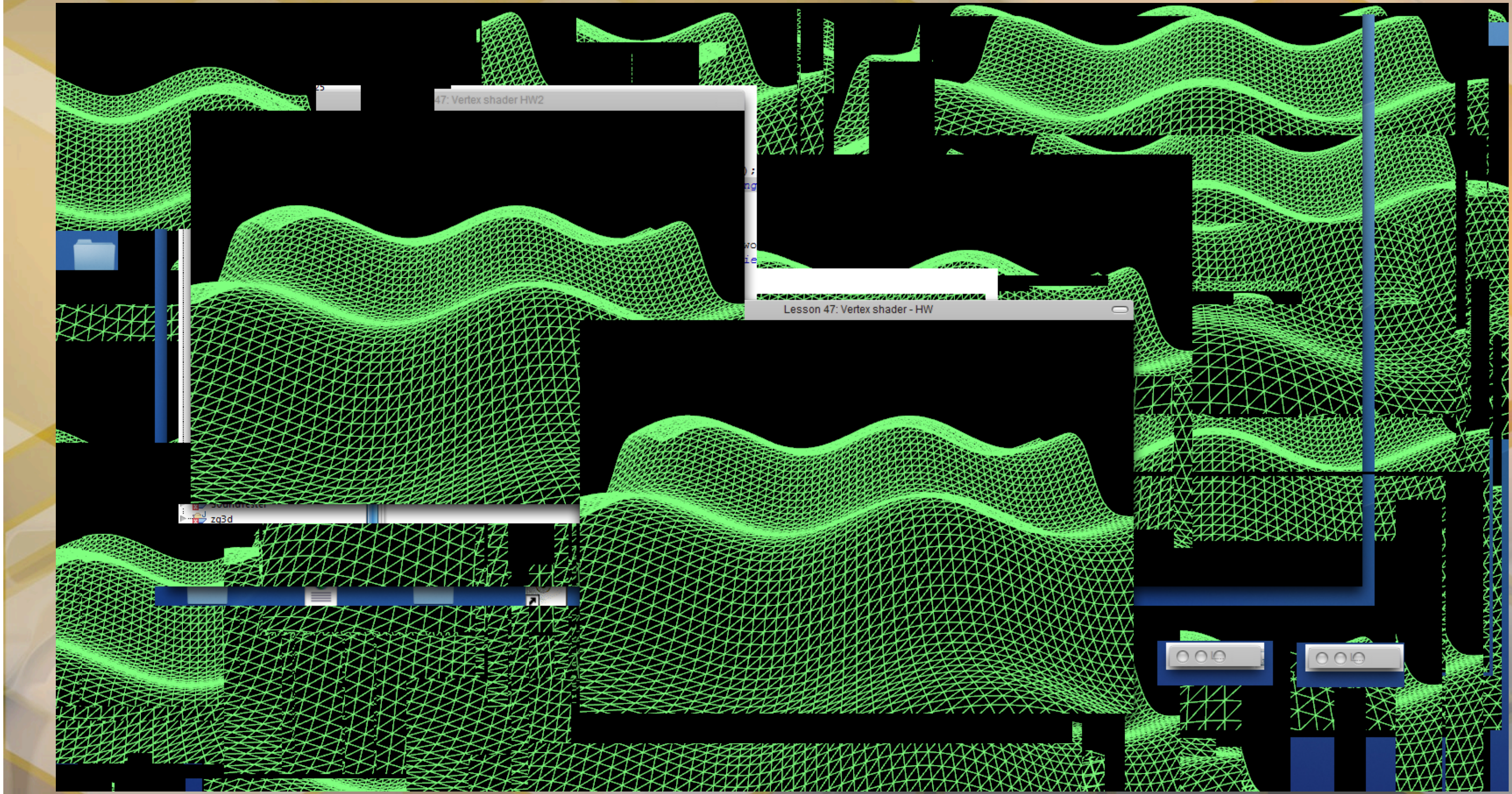




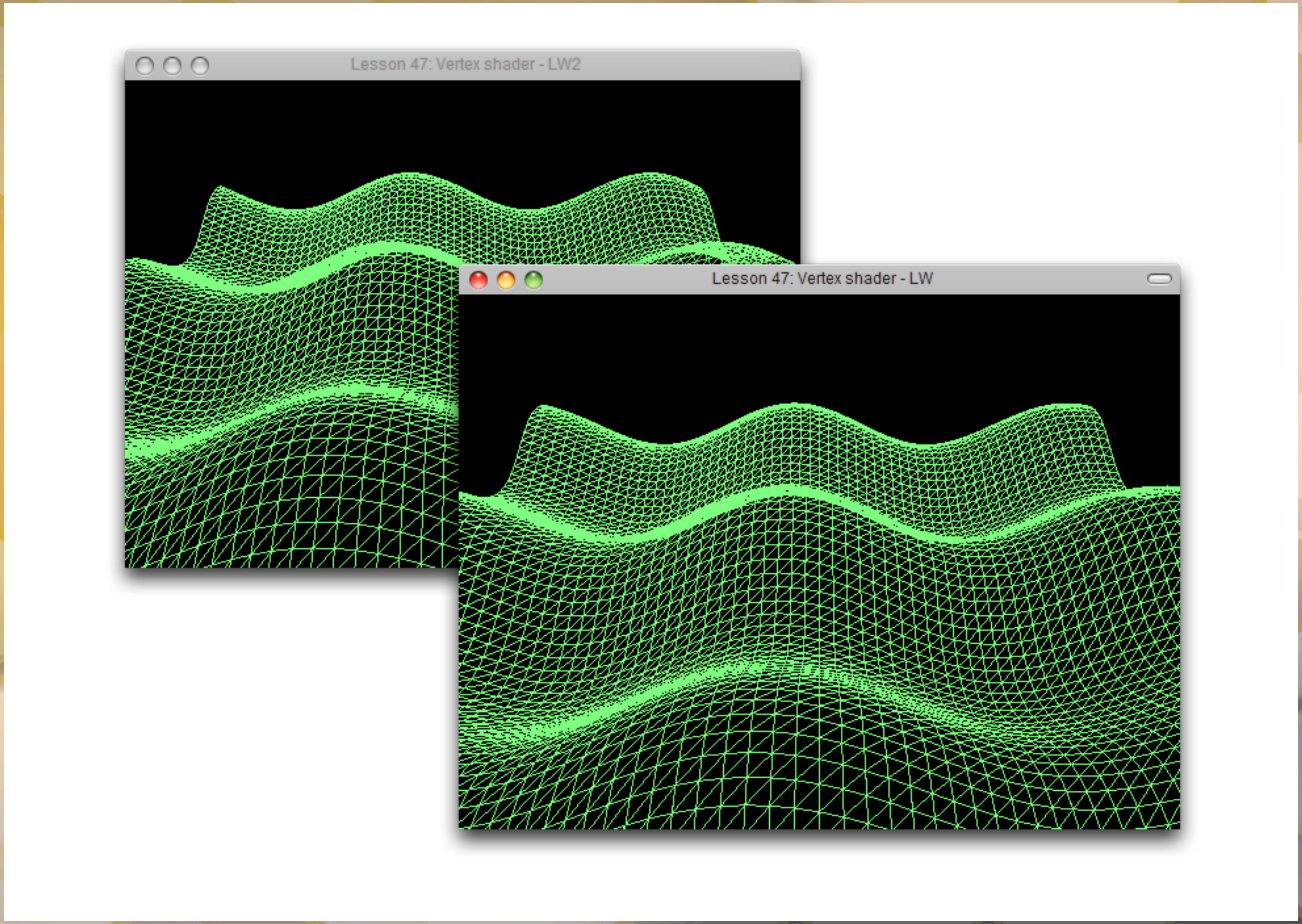
# Lightweight vs. Heavyweight

- **Graphics in JOGL can be implements as heavyweight or lightweight components.**
- **The heavyweight component is GLCanvas and the lightweight component is GLJPanel.**
- **GLCanvas is the preferred method of implementation and is faster than GLJPanel but heavyweight windows cause some graphical artefacts.**
- **Good News : Lightweight and Heavyweight components in JOGL code are quite easily interchangeable with a high degree of inter-compatibility.**

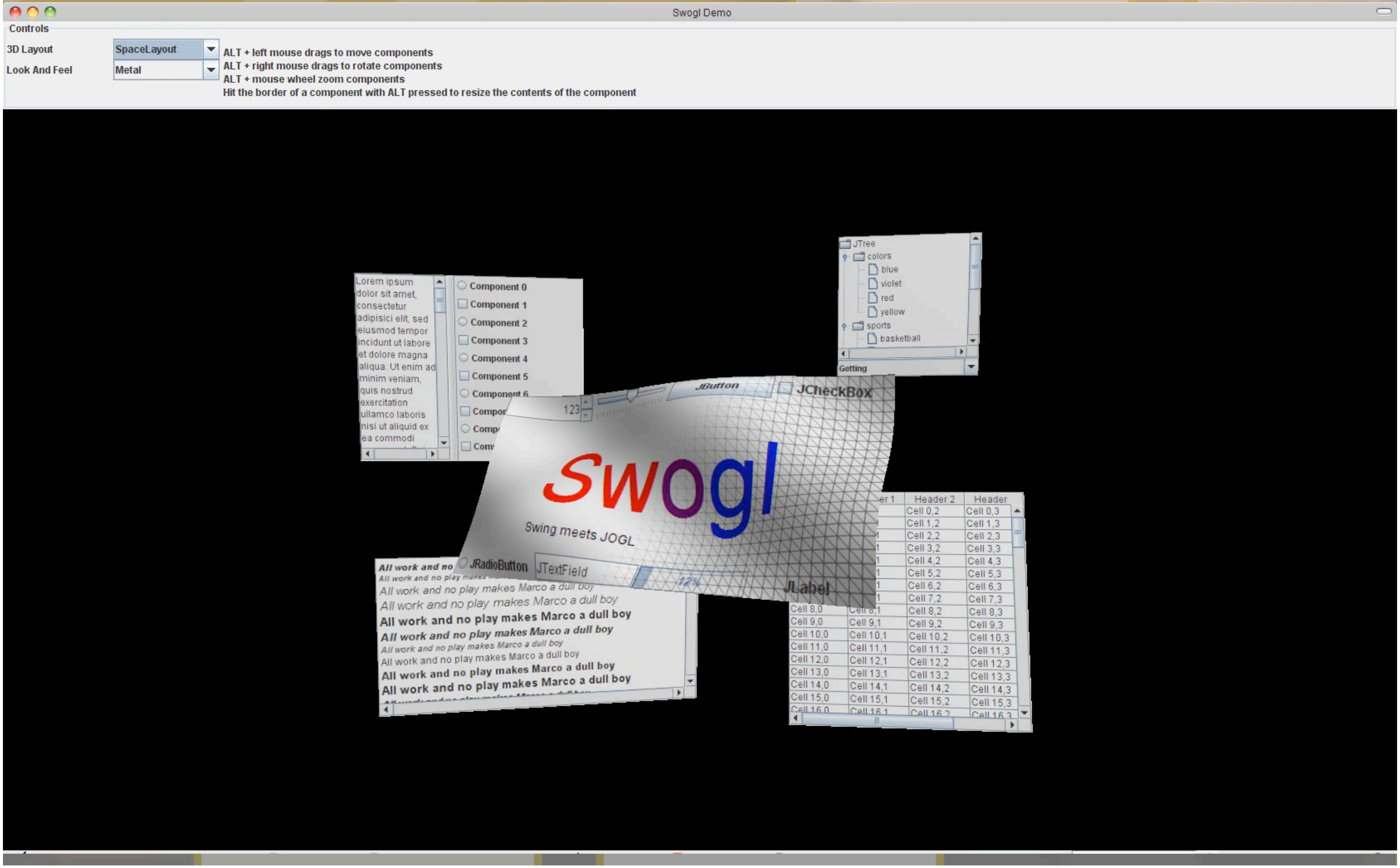
# Heavyweight Drawing(GLCanvas)



# Lightweight Drawing(GLJPanel)



# SWOGL – Swing on JOGL in 3-D



Swogl Demo

Controls

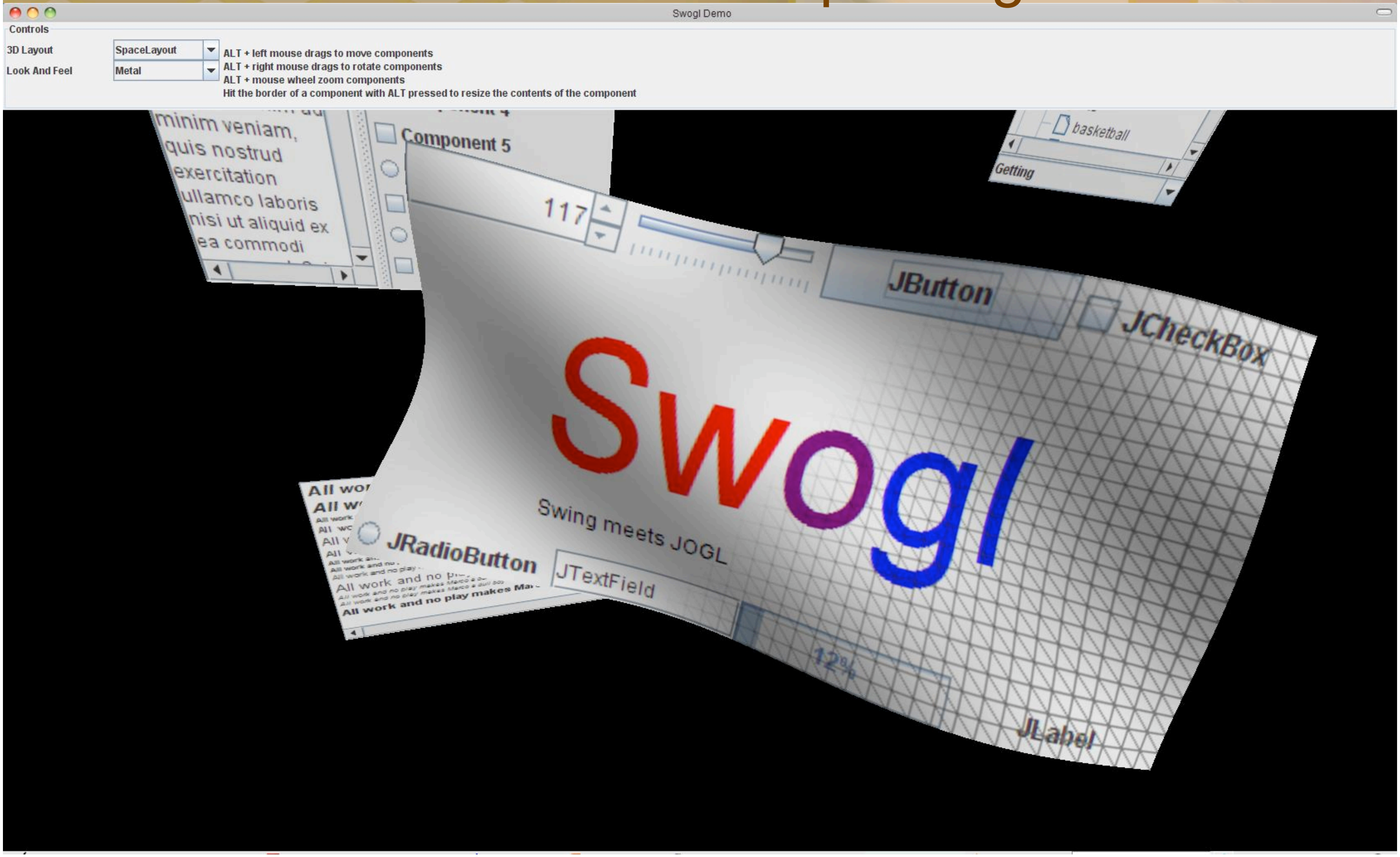
3D Layout: SpaceLayout

Look And Feel: Metal

ALT + left mouse drags to move components
ALT + right mouse drags to rotate components
ALT + mouse wheel zoom components
Hit the border of a component with ALT pressed to resize the contents of the component

# SwoglComponent class displays any Swing component in 3-D.

# Basis for new user interface paradigms.



Swogl Demo

Controls

3D Layout: SpaceLayout

Look And Feel: Metal

ALT + left mouse drags to move components
ALT + right mouse drags to rotate components
ALT + mouse wheel zoom components
Hit the border of a component with ALT pressed to resize the contents of the component

# NVIDIA’s CUDA

- CUDA(Compute Unified Device Architecture) is a parallel computing architecture developed by NVIDIA.
- Accessible to programmers through standard programming languages.
- Native implementation in C.
- Offers both low level and a higher level API.
- JAVA has open source implementations of CUDA. Eg – JCublas and JCufft.
- If used properly, the implementation of CUDA can speed up several processes.
- Drawbacks - CUDA enabled card must be present. Only (relatively) new NVIDIA cards supported, and no ATI cards supported.

# JCublas and  JCufft

- JCublas provides JAVA bindings for the NVIDIA CUDA BLAS( Basic Linear Algebra Subprograms). implementation, thus making the parallel processing power of modern graphics hardware available for JAVA programs.

- JCufft provides JAVA bindings for the NVIDIA CUDA Fast Fourier Transforms(FFT) implementation.

# Summary

- Multiple 2D animated variable plotting
- Fixing graphical issues
- Potential OpenGL implementation of Surface Plots
- Colormap smooth shading and preference saving implementation
- Fixing issues with PDF printing and fixing resizing and color selection issues
- Exploring NVIDIA’s CUDA.

# Thank You

- **In particular, I'd like to thank Eliot Feibush, Doug McCune, Drexel University and the Princeton Plasma Physics Lab.**