Mayavi and VTK

Introduction

Prabhu Ramachandran

Department of Aerospace Engineering
IIT Bombay

Outline

Background and Motivation

Introduction to Mayavi

- History

Outline

Background and Motivation

Introduction to Mayavi

History

Visualization?

What is visualization?

Visualization: graphics

> *Making a visible presentation of numerical data, particularly a graphical one. This might include anything from a simple X-Y graph of one dependent variable against one independent variable to a virtual reality which allows you to fly around the data.*

– from the Free On-line Dictionary of Computing

What is visualization?

Visual representation of data

3D visualization

Harder but important

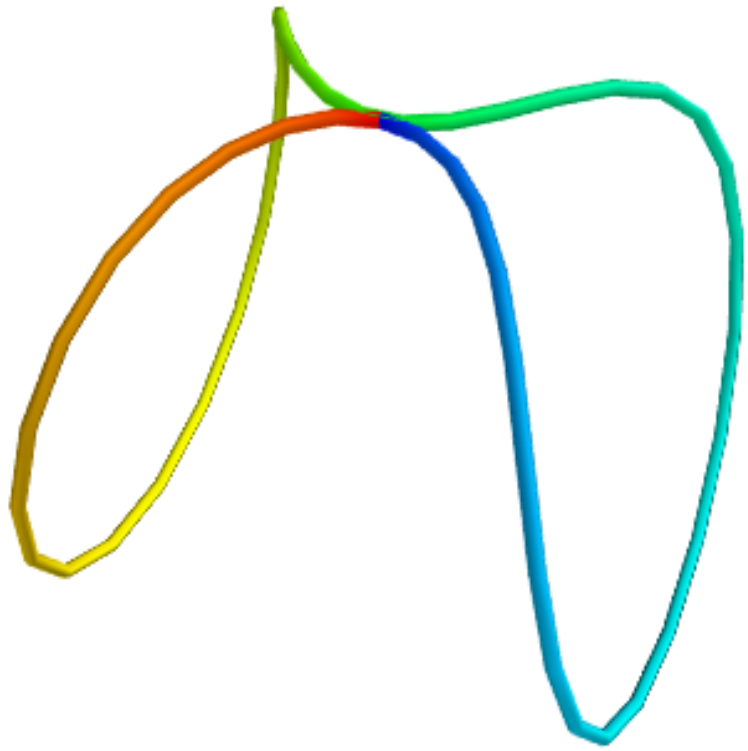
Is this Graphics?

Visualization is about data!

Examples: trajectory in space

```
>>> t = linspace(0, 2*pi, 50)
>>> u = cos(t)*pi
>>> x, y, z = sin(u), cos(u), sin(t)
```

Examples: trajectory in space



Examples: Fire in a room

Demo of data

Motivation and Needs

Scientists: not interested in graphics

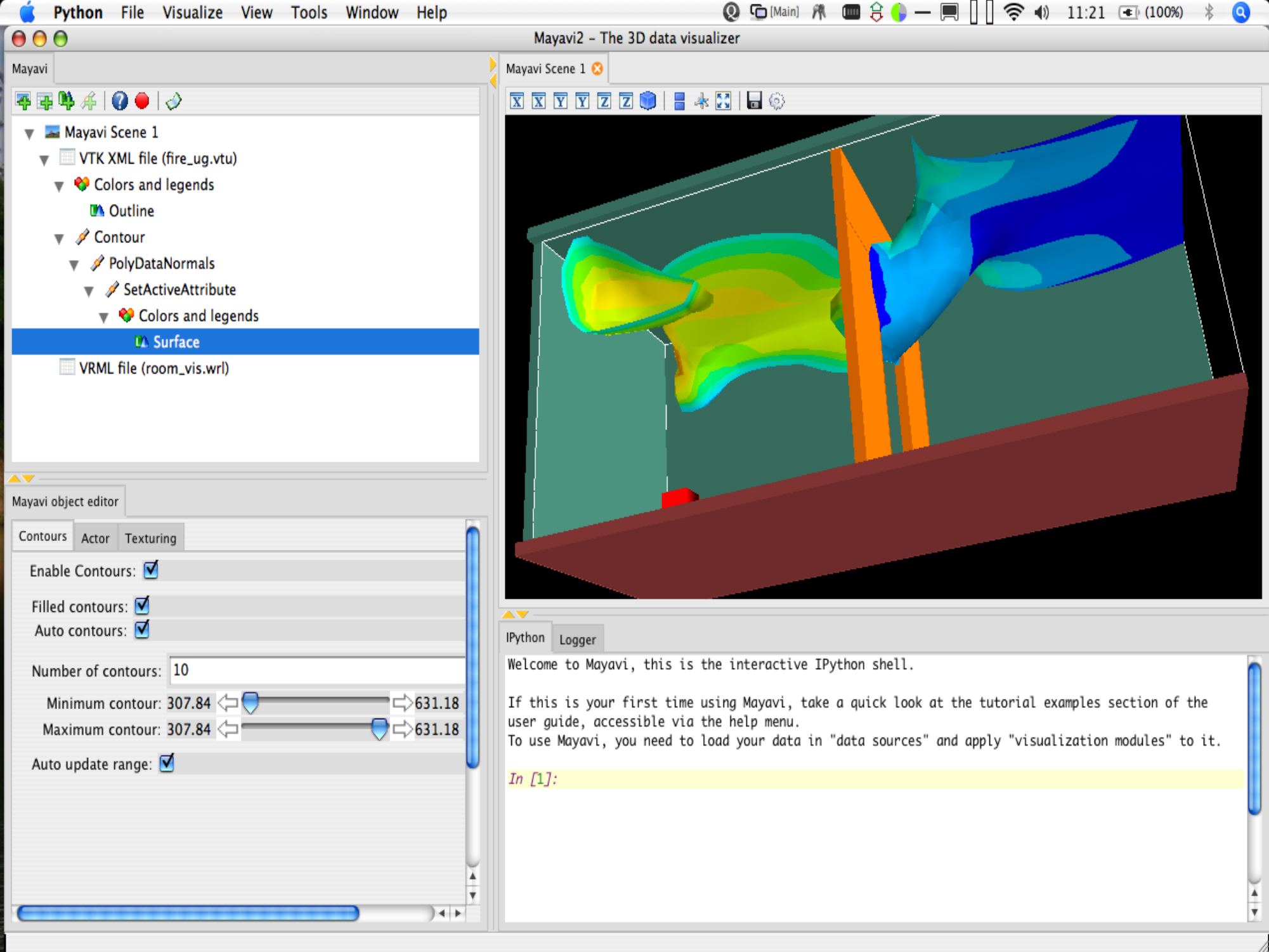
Interactive visualization of data (think Matlab)

Visualization of data files with a nice UI

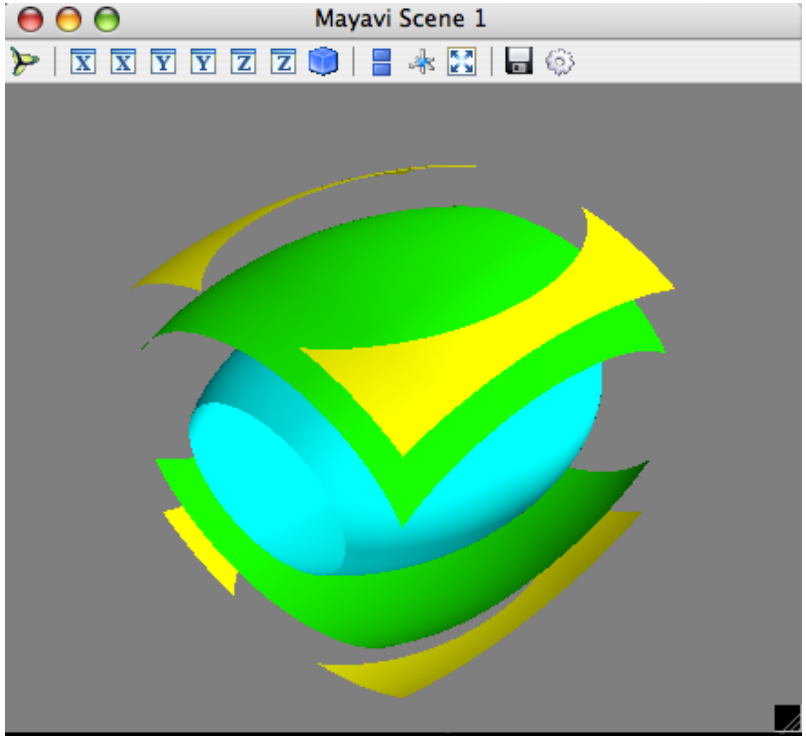
Embedding visualizations in applications

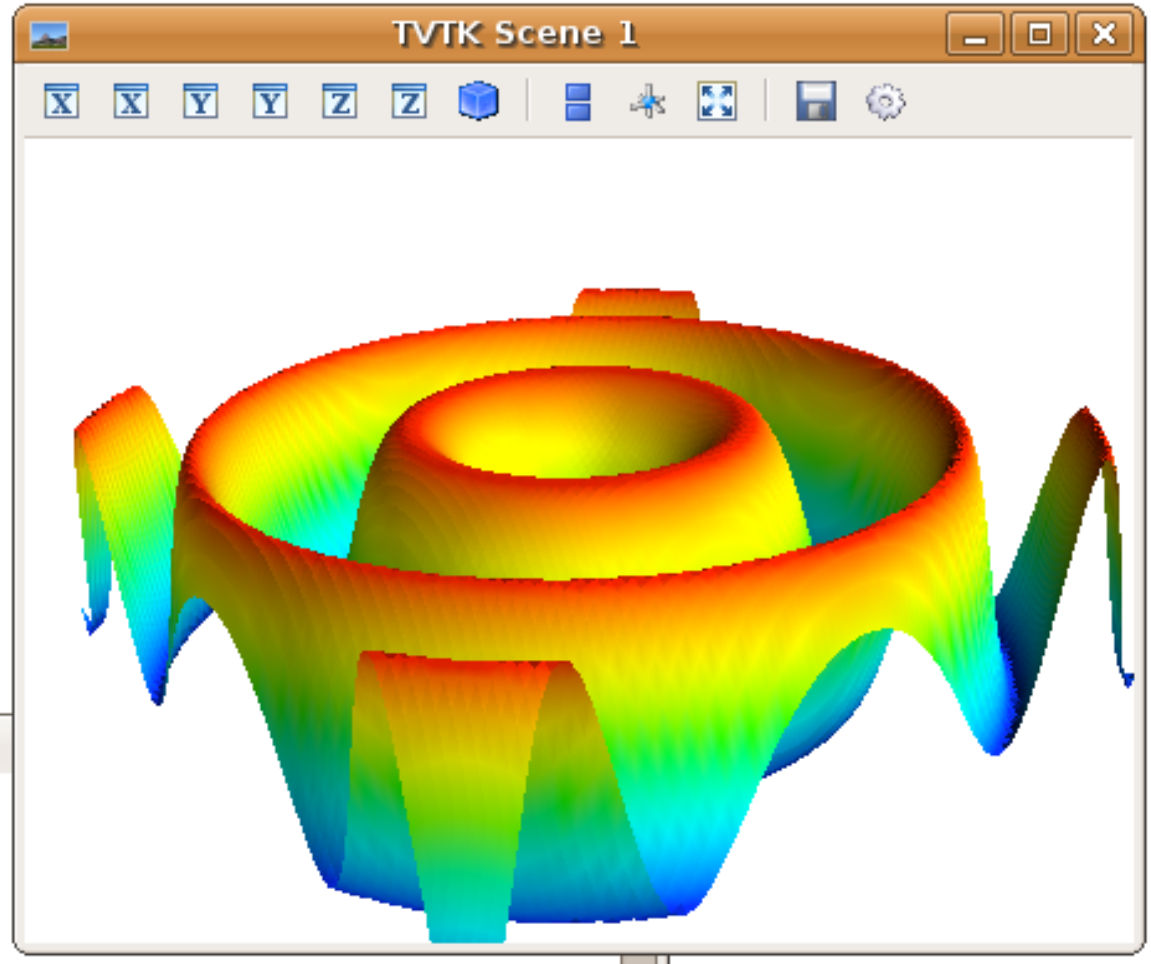
Customization

Flexible library/app for every one of these needs!


Python File Visualize View Tools Window Help
11:21
Mayavi2 - The 3D data visualizer
Mayavi
Mayavi Scene 1
VTK XML file (fire_ug.vtu)
Colors and legends
Outline
Contour
PolyDataNormals
SetActiveAttribute
Colors and legends
Surface
VRML file (room_vis.wrl)
Mayavi object editor
Contours
Actor
Texturing
Enable Contours:
Filled contours:
Auto contours:
Number of contours: 10
Minimum contour: 307.84 631.18
Maximum contour: 307.84 631.18
Auto update range:
Mayavi Scene 1
IPython
Logger
Welcome to Mayavi, this is the interactive IPython shell.
If this is your first time using Mayavi, take a quick look at the tutorial examples section of the user guide, accessible via the help menu.
To use Mayavi, you need to load your data in "data sources" and apply "visualization modules" to it.
In [1]:

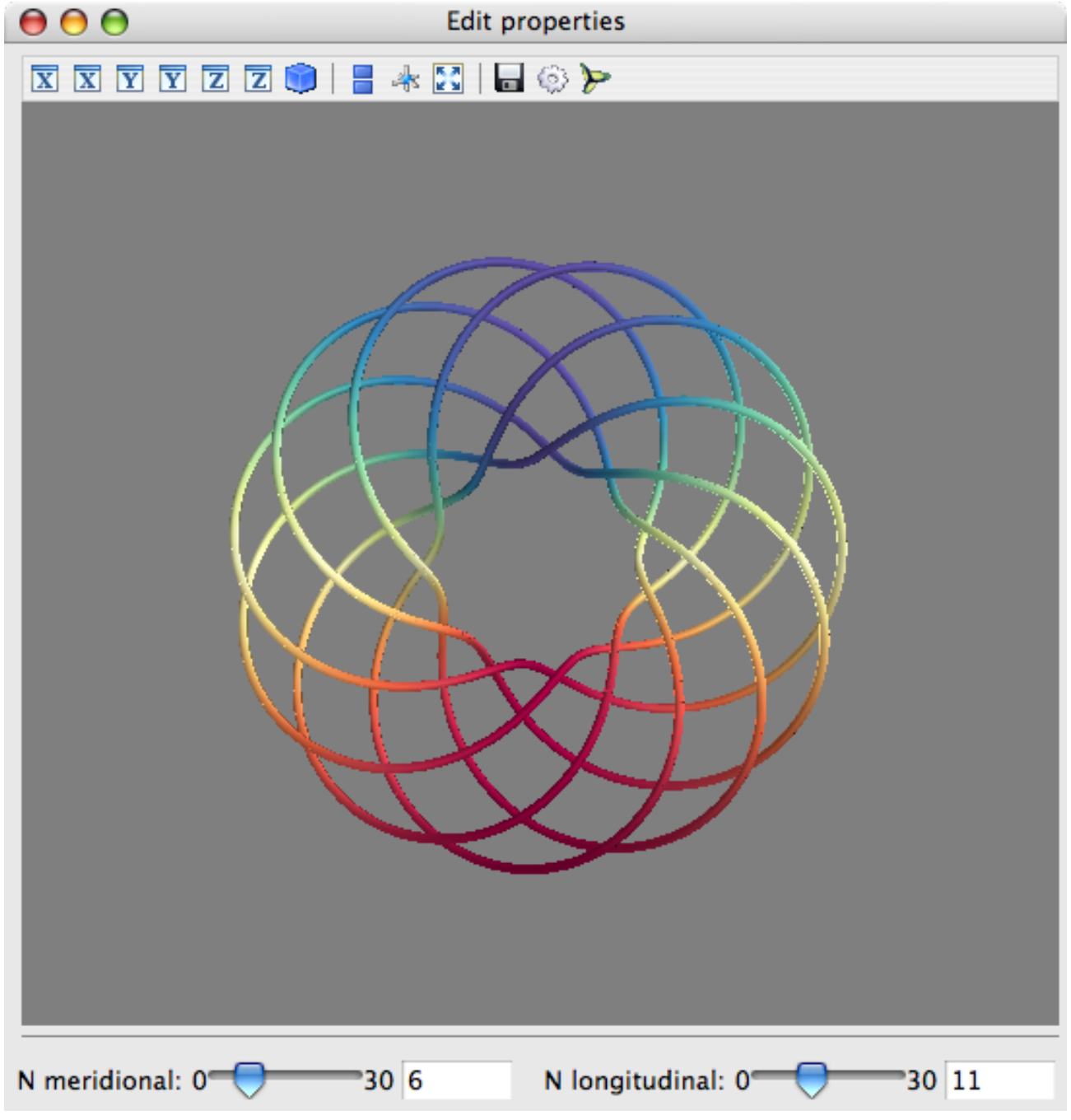
```
from mayavi import mlab
from numpy import ogrid
x, y, z = ogrid[-5:5:64j,
                -5:5:64j,
                -5:5:64j]

mlab.contour3d(
    x*x*0.5 + y*y + z*z*2
)
mlab.show()
```



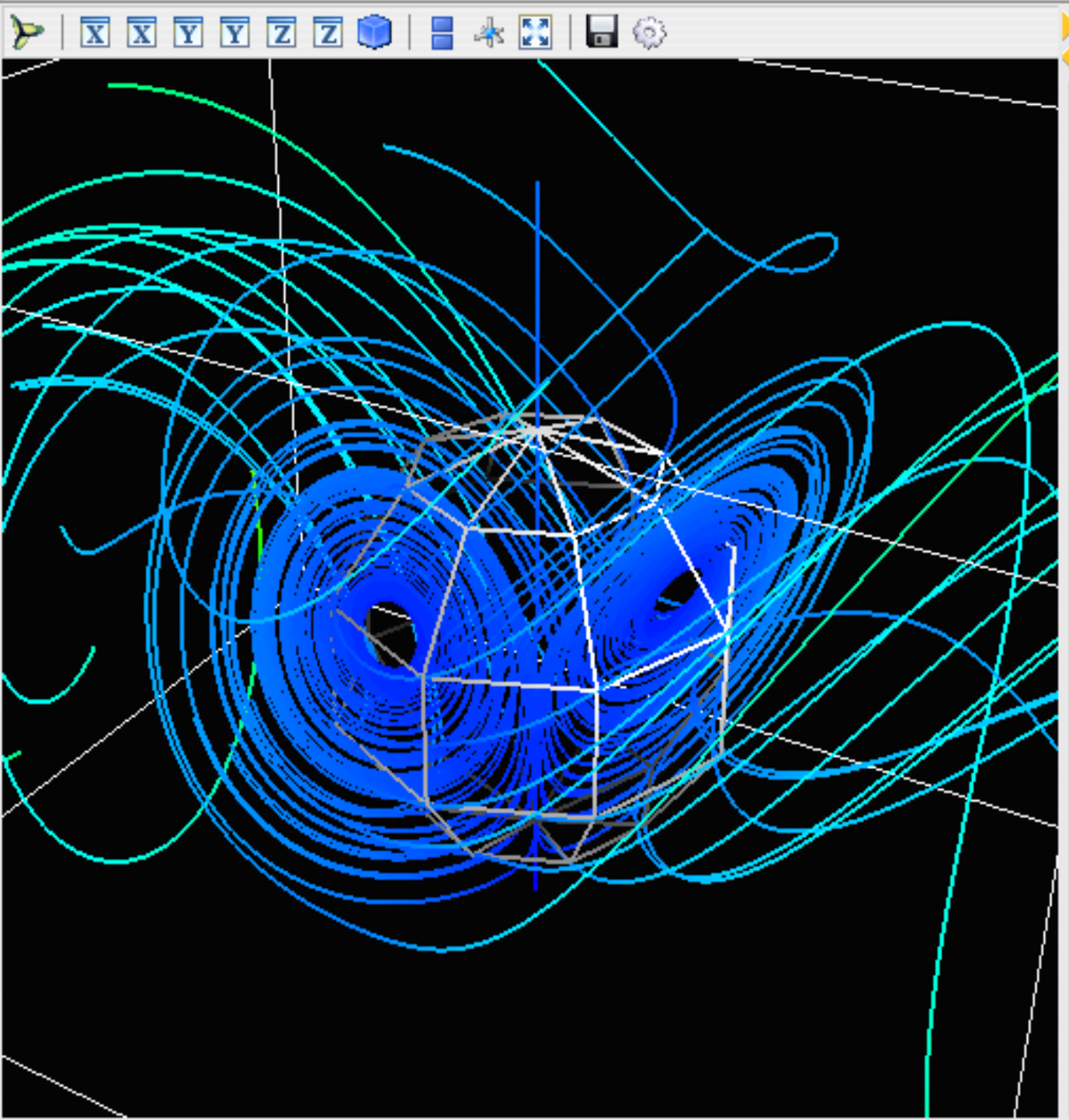


Terminal

```
In [1]: from numpy import *

In [2]: x, y = mgrid[-3:3:100j, -3:3:100j]

In [3]: z = sin(x**2 + y**2)

In [4]: from mayavi import mlab

In [5]: mlab.surf(x, y, z)
```


Edit properties
N meridional: 0
30
6
N longitudinal: 0
30
11

Edit properties



S: 0.0 20.0 10.0

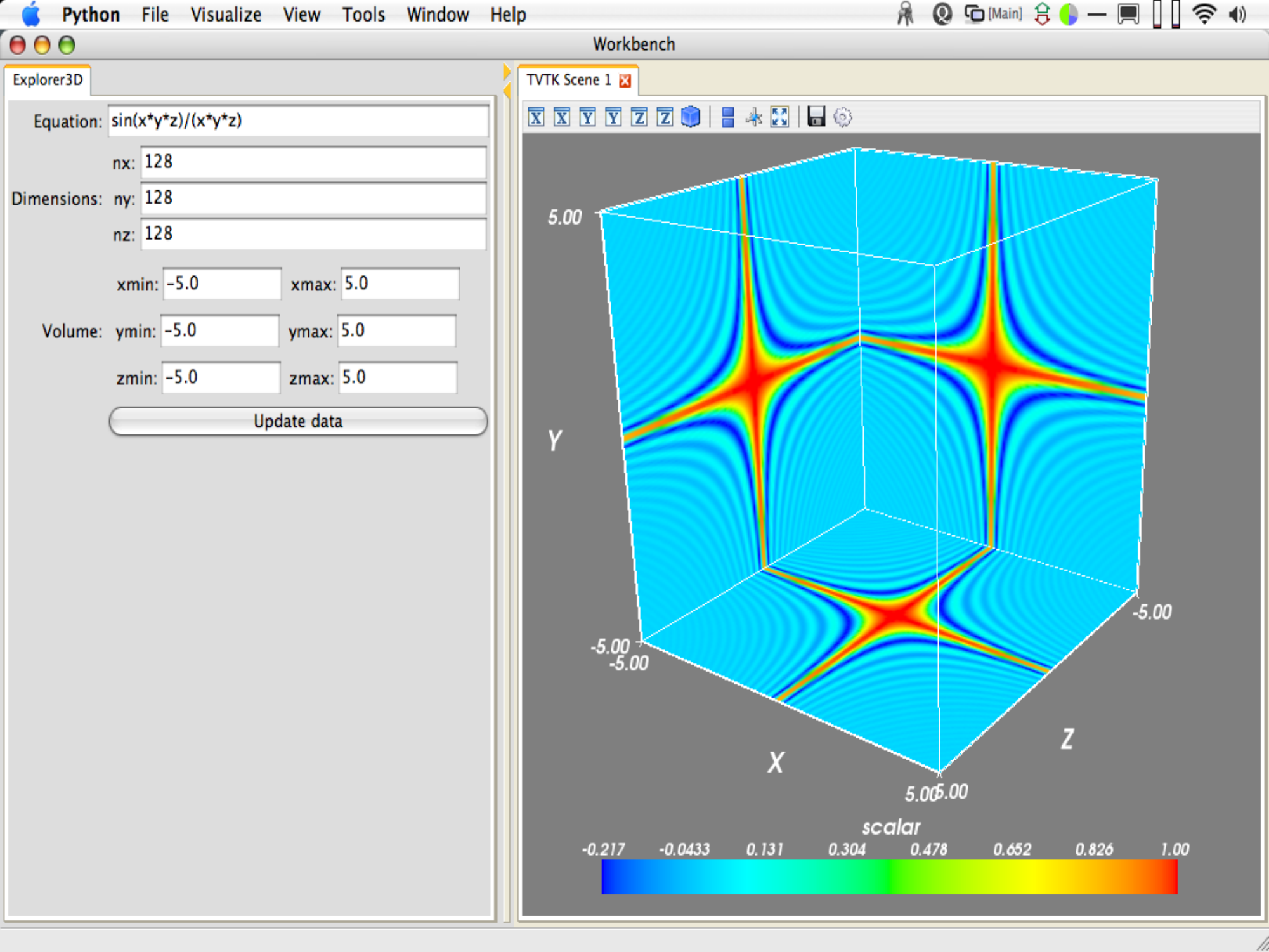
R: 0.0 50.0 28.0

B: 0.0 10.0 666667

U: s*(y-x)

V: r*x - y - x*z

W: x*y - b*z


Python File Visualize View Tools Window Help
Workbench
Explorer3D
Equation: sin(x*y*z)/(x*y*z)
nx: 128
Dimensions: ny: 128
nz: 128
xmin: -5.0 xmax: 5.0
Volume: ymin: -5.0 ymax: 5.0
zmin: -5.0 zmax: 5.0
Update data
TVTK Scene 1
5.00
Y
-5.00
-5.00
X
5.00
5.00
Z
-5.00
scalar
-0.217 -0.0433 0.131 0.304 0.478 0.652 0.826 1.00

mayavi_jupyter

Prabhu

localhost:8891/notebooks/mayavi_jupyter.ipynb

jupyter mayavi_jupyter Last Checkpoint: 2 minutes ago (autosaved)

File Edit View Insert Cell Kernel Help Python 2

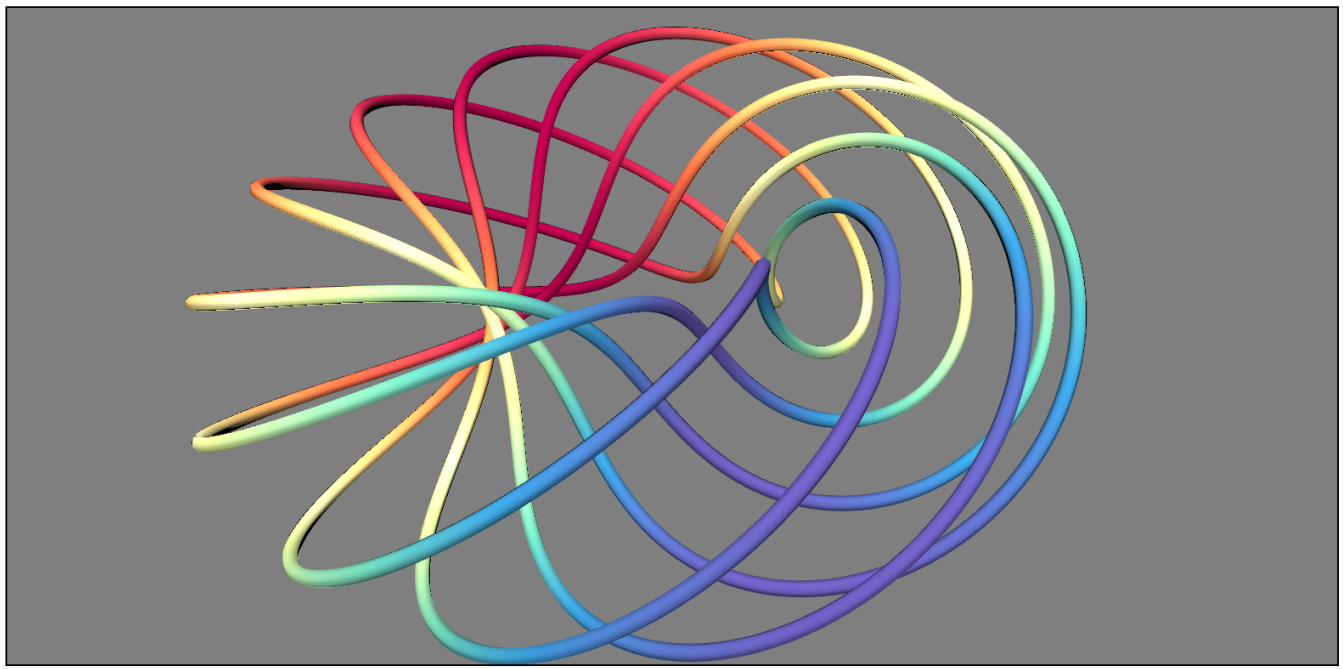
Code CellToolbar

In [1]:
```
from mayavi import mlab
mlab.init_notebook()
```

```
Notebook initialized with x3d backend.
```

In [2]:
```
s = mlab.test_plot3d()
s
```

Out[2]:



In [ ]:

Other features

- Automatic script recording
- wxPython and Qt support
- Powerful command line options
- Off-screen support

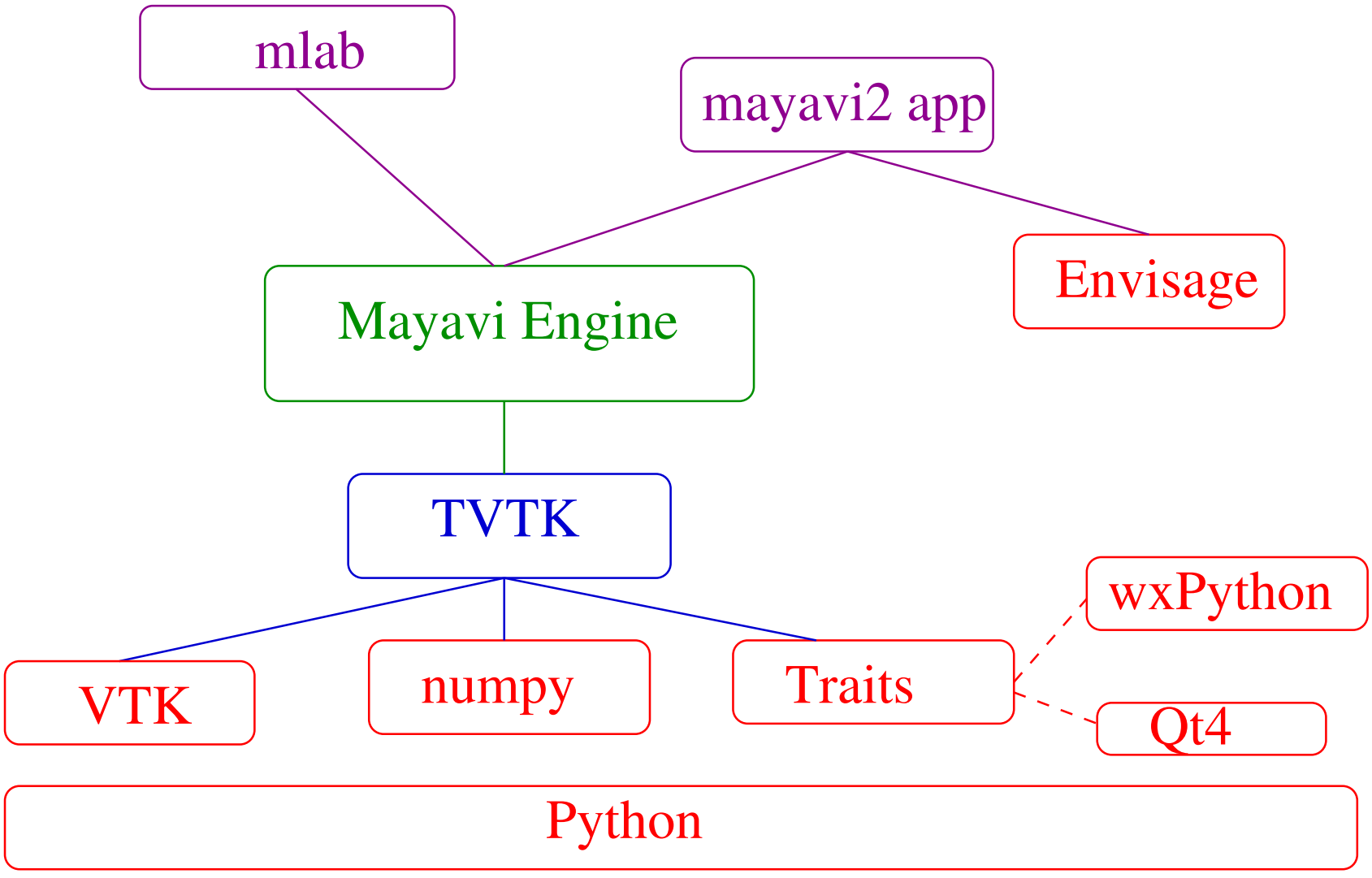
Outline

Background and Motivation

Introduction to Mayavi

History

Overview of architecture



Information

- `http://code.enthought.com/projects/mayavi`
- `https://github.com/enthought/mayavi`
- Uses VTK (`www.vtk.org`)
- BSD license
- Linux, Windows and Mac OS X
- Debian/Ubuntu/Fedora
- Canopy/Anaconda

Overview of features

- `mayavi2` application
- Python library
- OO design
- Highly scriptable
- `mayavi.mlab`: for easy scripting
- Pythonic: Seamless NumPy integration
- Embed in Traits UIs (wxPython and PyQt4/PySide)
- Envisage Plugins

Outline

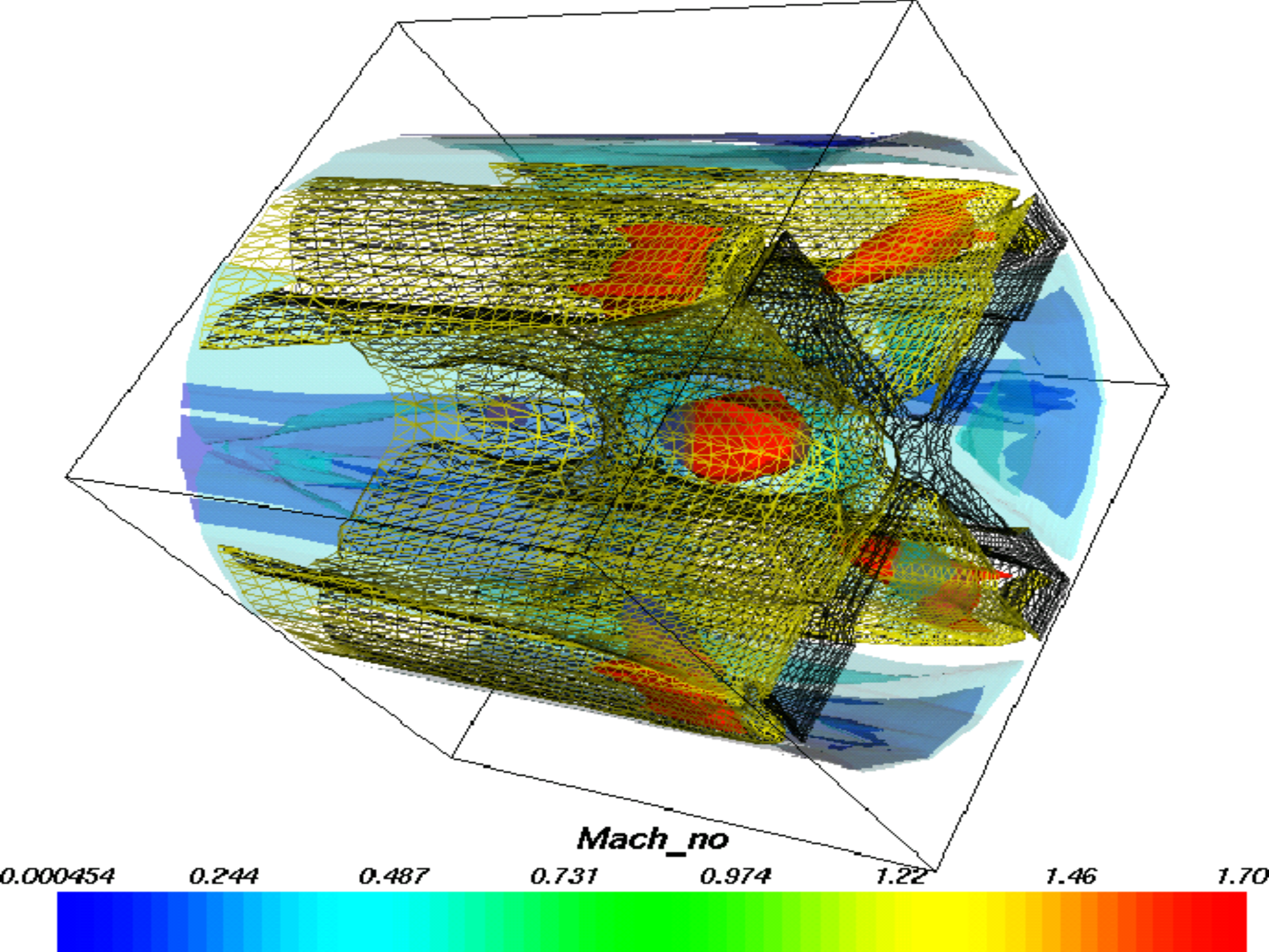
Background and Motivation

Introduction to Mayavi

History

Requirements

Colleagues at IITM needed 3D visualization for CFD data (1998)


Mach_no
0.000454
0.244
0.487
0.731
0.974
1.22
1.46
1.70

VTK: Visualization Toolkit

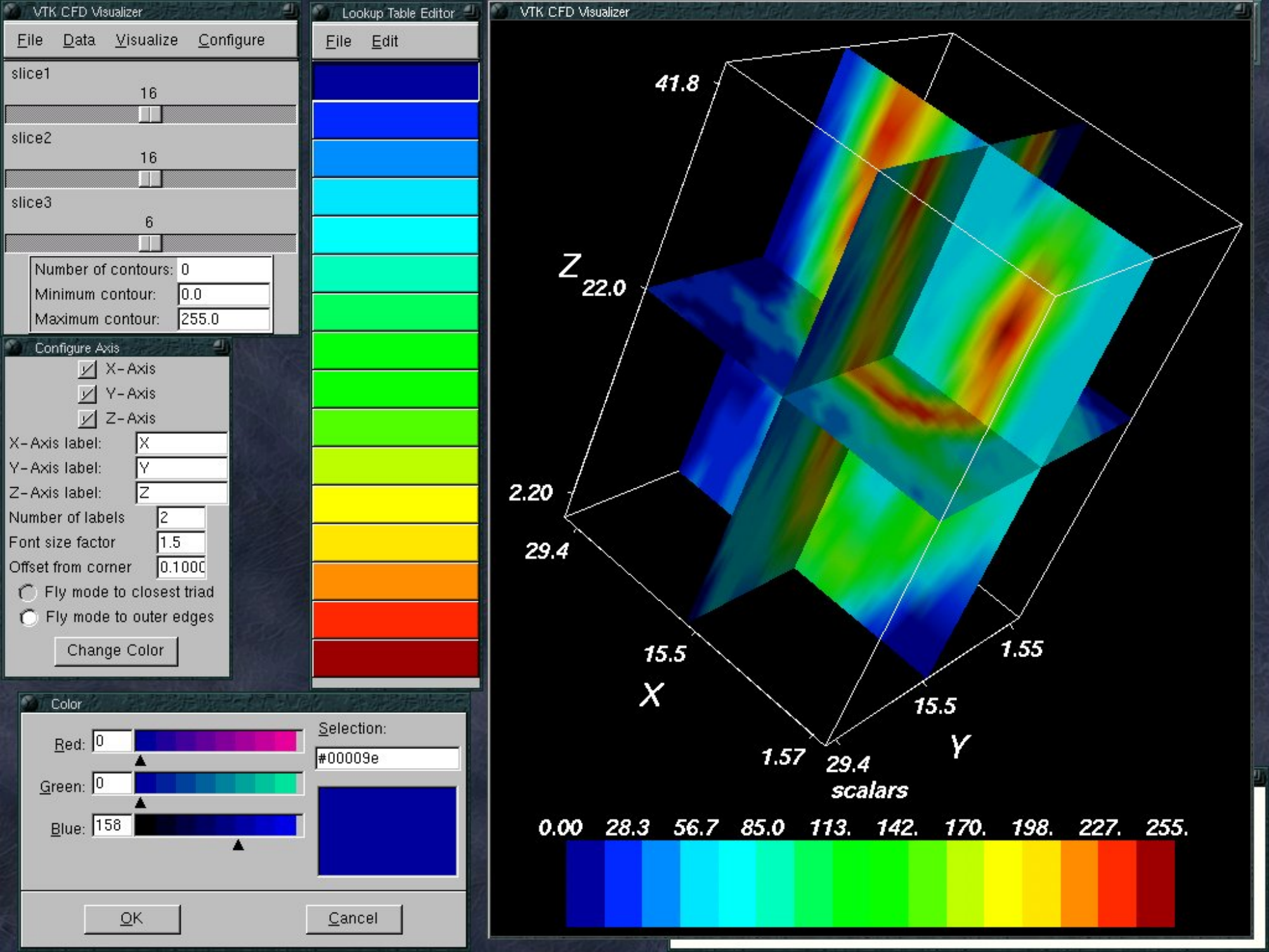
www.vtk.org

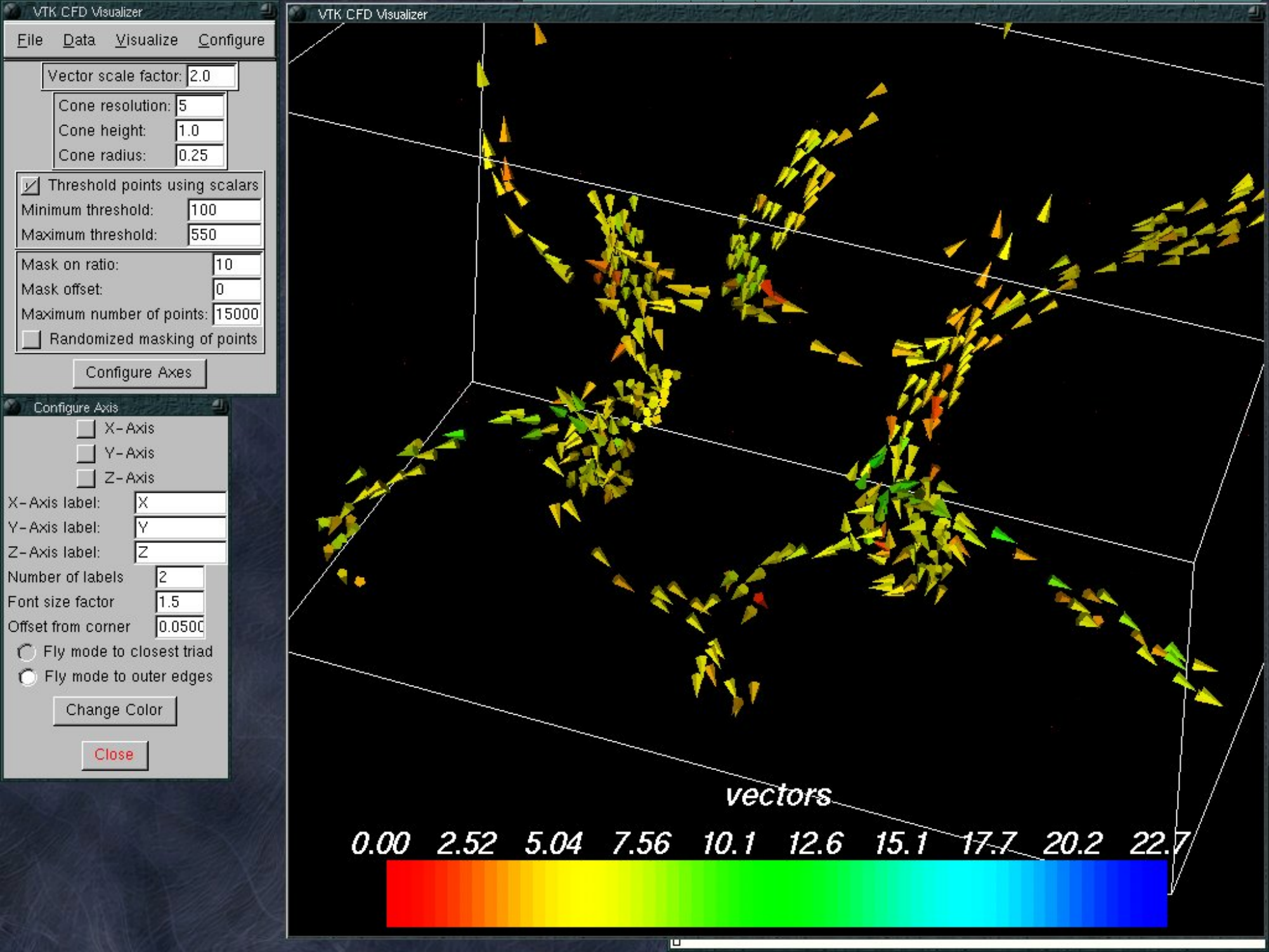
- 3D graphics, imaging and visualization
- C++ wrappers: Python (Tcl, Java)
- Pipeline architecture
- Huge: 900+ classes!
- 2000+ classes now!!

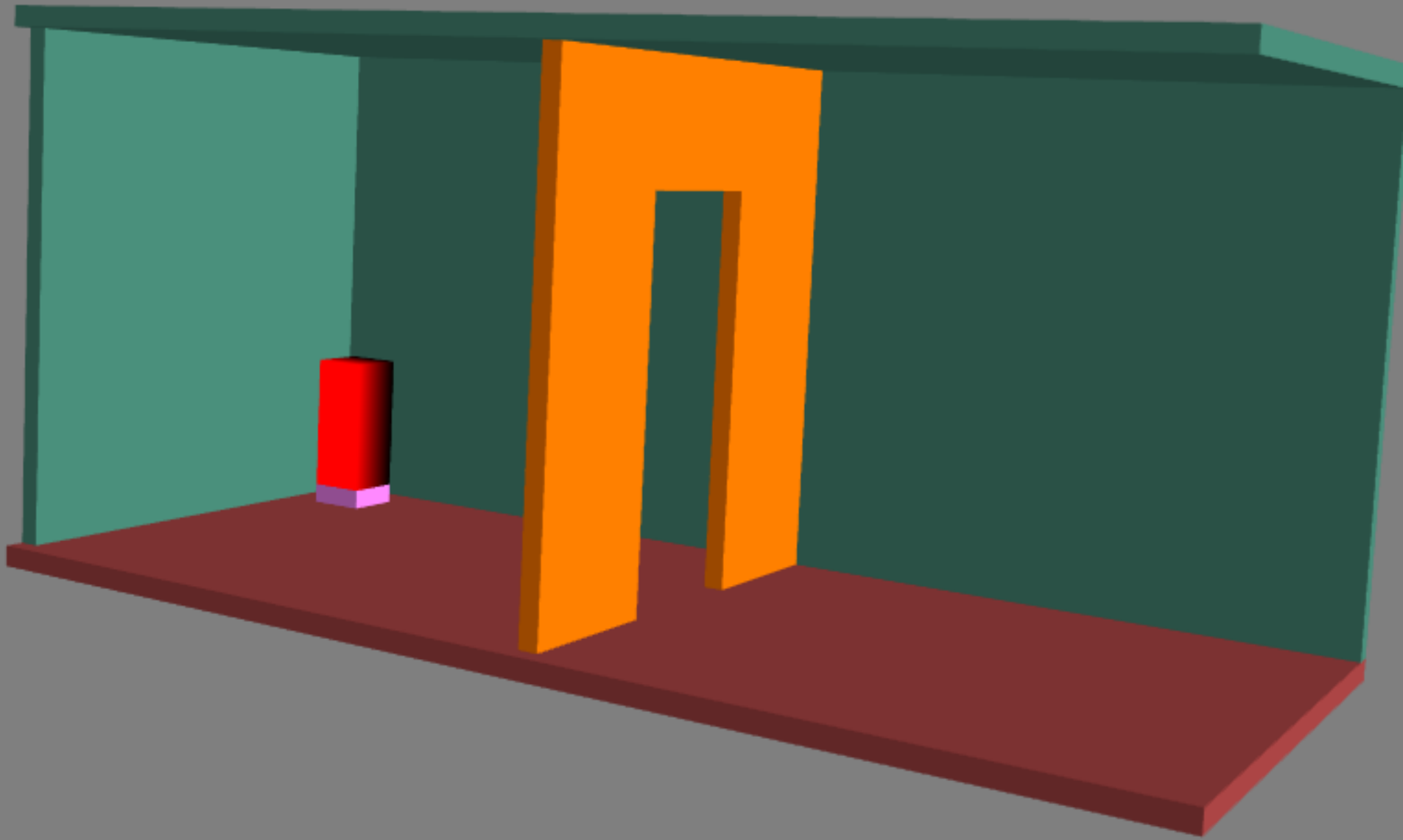
Who wants to learn a graphics library?

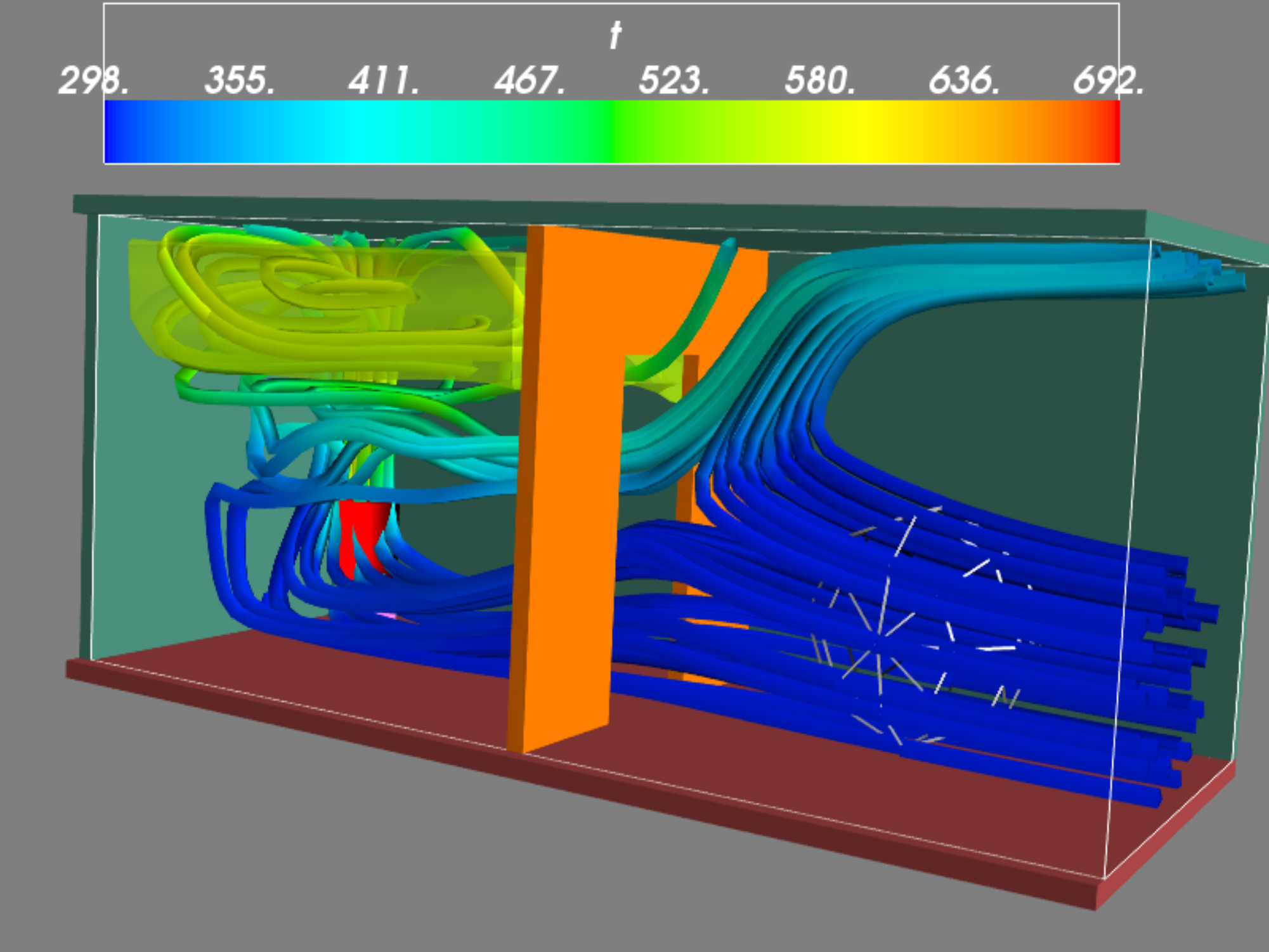
VTK-CFD (2000)

- Simple UI: Tkinter
- Free


VTK CFD Visualizer
File Data Visualize Configure
slice1
16
slice2
16
slice3
6
Number of contours: 0
Minimum contour: 0.0
Maximum contour: 255.0
Configure Axis
X-Axis
Y-Axis
Z-Axis
X-Axis label: X
Y-Axis label: Y
Z-Axis label: Z
Number of labels 2
Font size factor 1.5
Offset from corner 0.1000
Fly mode to closest triad
Fly mode to outer edges
Change Color
Lookup Table Editor
File Edit
Color
Red: 0
Green: 0
Blue: 158
Selection:
#00009e
OK
Cancel
VTK CFD Visualizer
41.8
Z
22.0
2.20
29.4
15.5
X
1.57
29.4
15.5
Y
1.55
scalars
0.00 28.3 56.7 85.0 113. 142. 170. 198. 227. 255.


VTK CFD Visualizer
File
Data
Visualize
Configure
Vector scale factor: 2.0
Cone resolution: 5
Cone height: 1.0
Cone radius: 0.25
Threshold points using scalars
Minimum threshold: 100
Maximum threshold: 550
Mask on ratio: 10
Mask offset: 0
Maximum number of points: 15000
Randomized masking of points
Configure Axes
Configure Axis
X-Axis
Y-Axis
Z-Axis
X-Axis label: X
Y-Axis label: Y
Z-Axis label: Z
Number of labels 2
Font size factor 1.5
Offset from corner 0.0500
Fly mode to closest triad
Fly mode to outer edges
Change Color
Close
VTK CFD Visualizer
vectors
0.00 2.52 5.04 7.56 10.1 12.6 15.1 17.7 20.2 22.7




t
298.
355.
411.
467.
523.
580.
636.
692.

vtkPipeline

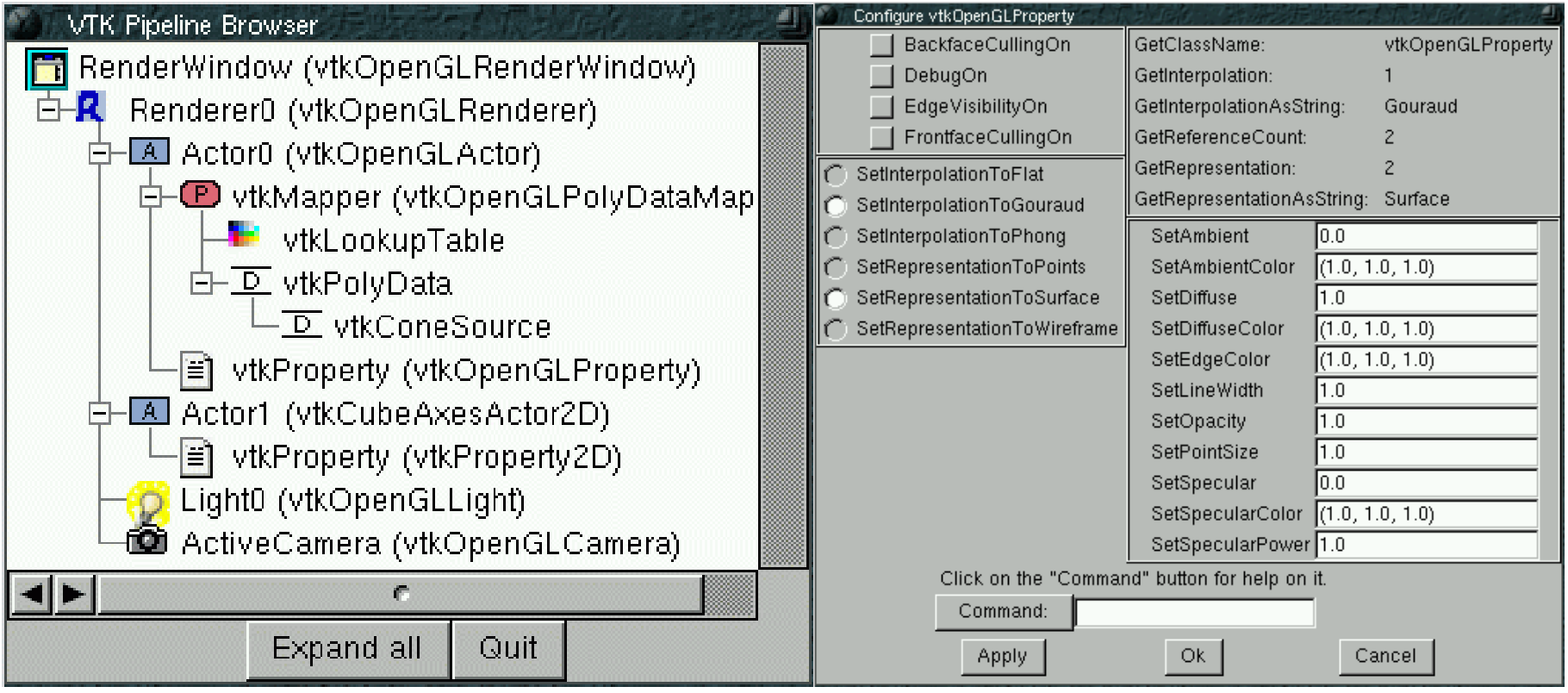
Parse VTK class on the fly
producing automatic UI

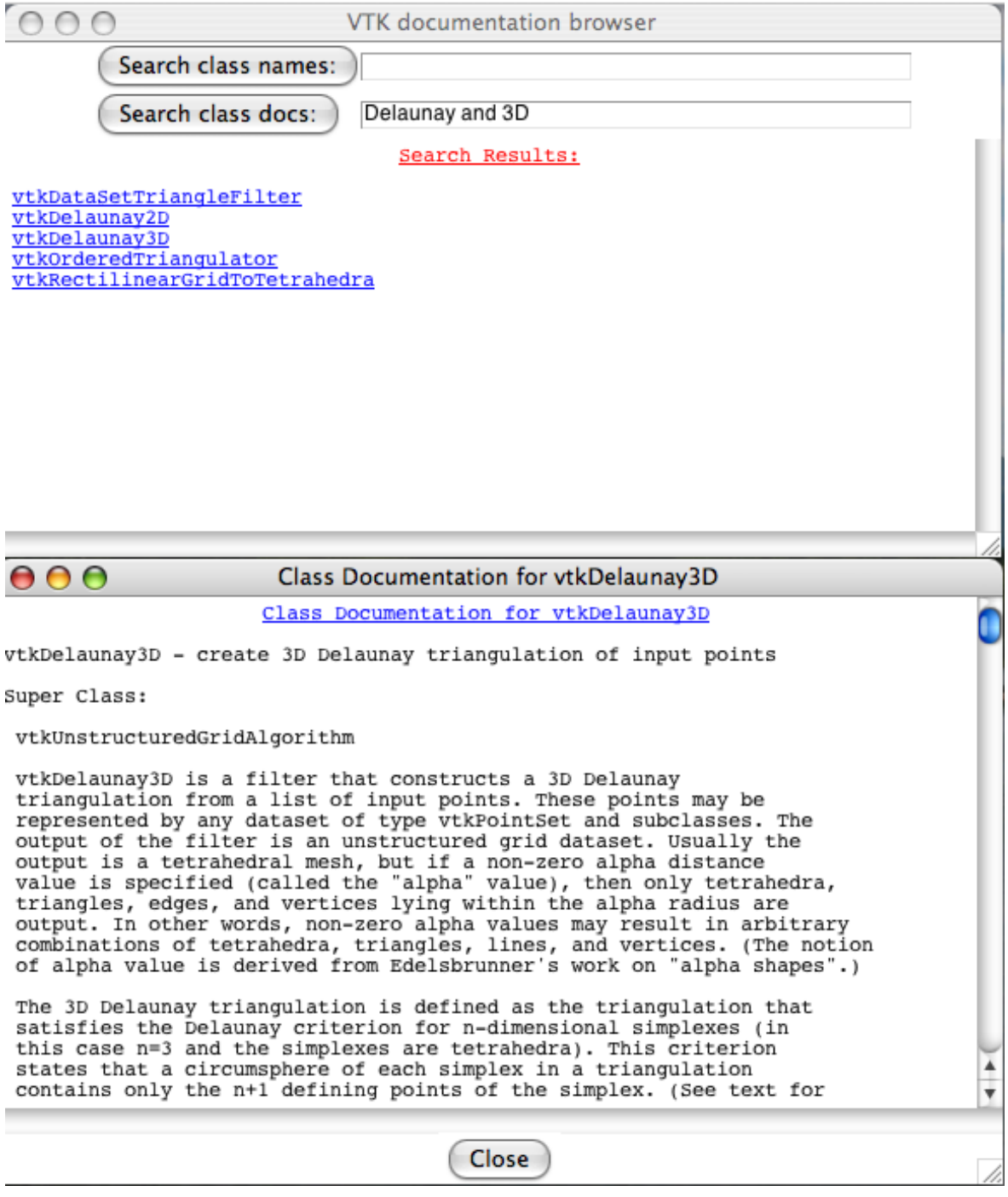
vtkPipeline

Parse VTK class on the fly producing automatic UI

Discover and generate VTK pipeline

```
from vtkPipeline import \
vtkPipelineBrowser

# Create a full VTK-Python script
# ...

# renwin is a vtkRenderWindow.
pipe = vtkPipelineBrowser(root, renwin)
pipe.browse ()
```


VTK Pipeline Browser
RenderWindow (vtkOpenGLRenderWindow)
Renderer0 (vtkOpenGLRenderer)
Actor0 (vtkOpenGLActor)
vtkMapper (vtkOpenGLPolyDataMap
vtkLookupTable
vtkPolyData
vtkConeSource
vtkProperty (vtkOpenGLProperty)
Actor1 (vtkCubeAxesActor2D)
vtkProperty (vtkProperty2D)
Light0 (vtkOpenGLLight)
ActiveCamera (vtkOpenGLCamera)
Expand all
Quit
Configure vtkOpenGLProperty
BackfaceCullingOn
DebugOn
EdgeVisibilityOn
FrontfaceCullingOn
SetInterpolationToFlat
SetInterpolationToGouraud
SetInterpolationToPhong
SetRepresentationToPoints
SetRepresentationToSurface
SetRepresentationToWireframe
GetClassName: vtkOpenGLProperty
GetInterpolation: 1
GetInterpolationAsString: Gouraud
GetReferenceCount: 2
GetRepresentation: 2
GetRepresentationAsString: Surface
SetAmbient 0.0
SetAmbientColor (1.0, 1.0, 1.0)
SetDiffuse 1.0
SetDiffuseColor (1.0, 1.0, 1.0)
SetEdgeColor (1.0, 1.0, 1.0)
SetLineWidth 1.0
SetOpacity 1.0
SetPointSize 1.0
SetSpecular 0.0
SetSpecularColor (1.0, 1.0, 1.0)
SetSpecularPower 1.0
Click on the "Command" button for help on it.
Command:
Apply
Ok
Cancel


VTK documentation browser
Search class names:
Search class docs:
Delaunay and 3D
Search Results:
vtkDataSetTriangleFilter
vtkDelaunay2D
vtkDelaunay3D
vtkOrderedTriangulator
vtkRectilinearGridToTetrahedra
Class Documentation for vtkDelaunay3D
Class Documentation for vtkDelaunay3D
vtkDelaunay3D - create 3D Delaunay triangulation of input points
Super Class:
vtkUnstructuredGridAlgorithm
vtkDelaunay3D is a filter that constructs a 3D Delaunay triangulation from a list of input points. These points may be represented by any dataset of type vtkPointSet and subclasses. The output of the filter is an unstructured grid dataset. Usually the output is a tetrahedral mesh, but if a non-zero alpha distance value is specified (called the "alpha" value), then only tetrahedra, triangles, edges, and vertices lying within the alpha radius are output. In other words, non-zero alpha values may result in arbitrary combinations of tetrahedra, triangles, lines, and vertices. (The notion of alpha value is derived from Edelsbrunner's work on "alpha shapes".)
The 3D Delaunay triangulation is defined as the triangulation that satisfies the Delaunay criterion for n-dimensional simplexes (in this case n=3 and the simplexes are tetrahedra). This criterion states that a circumsphere of each simplex in a triangulation contains only the n+1 defining points of the simplex. (See text for
Close

Doc browser: one afternoon!

Lessons

- Python rocks!
- VTK is very powerful

Lessons

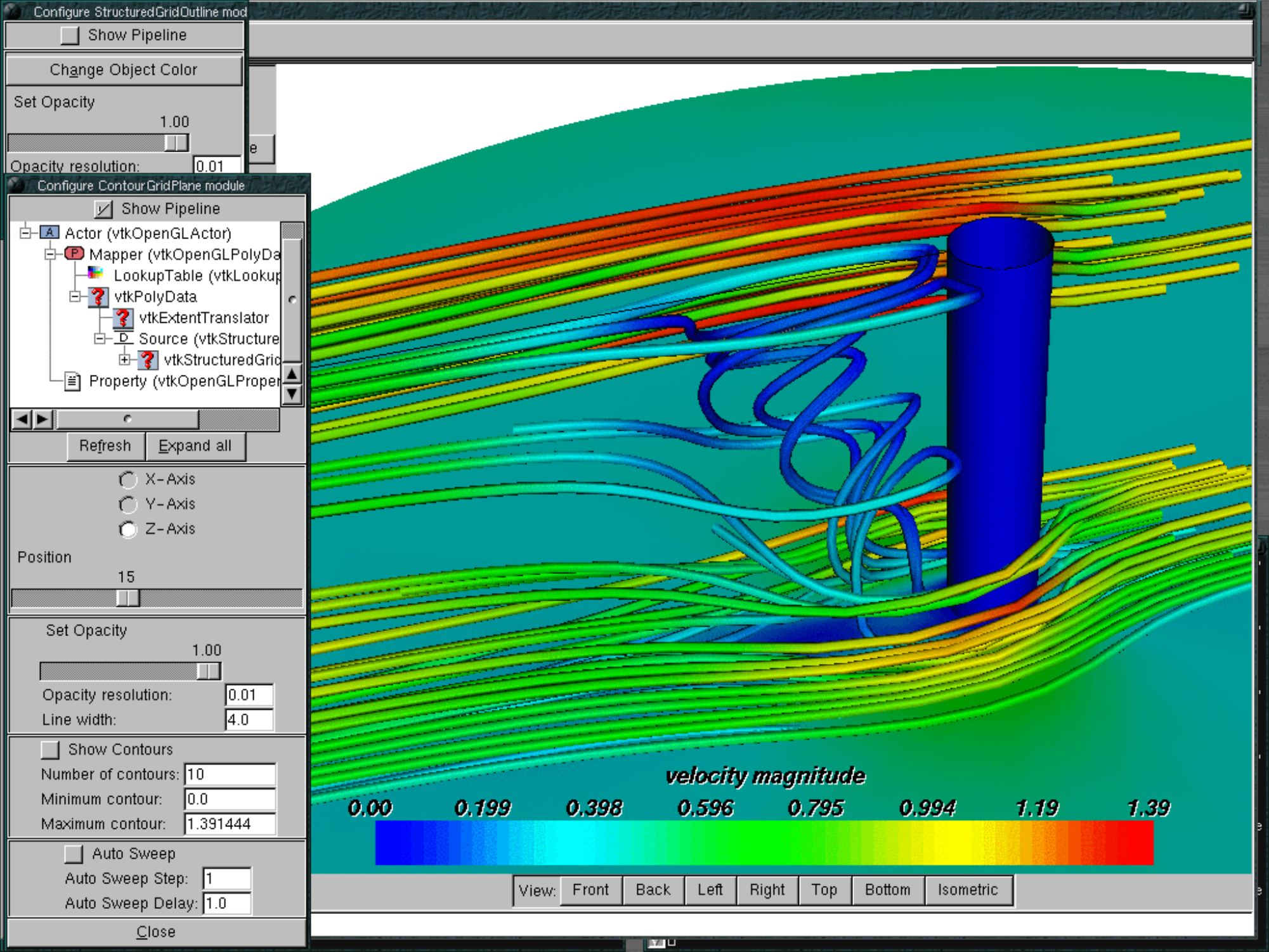
- Run-time introspection
- Dynamic programming
- Automatic UIs are cool and fun

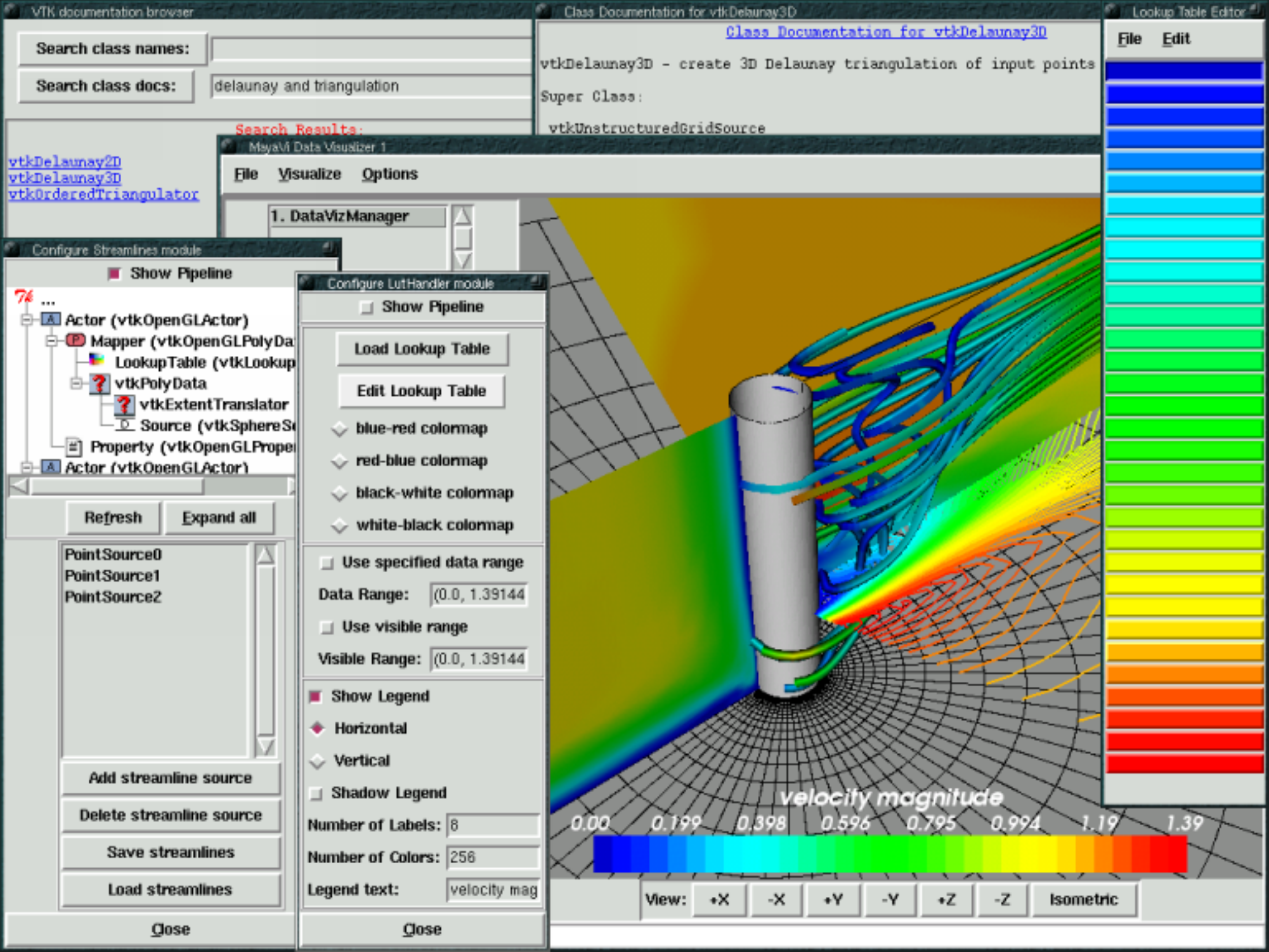
Issues with VTK-CFD

- Very specific visualizations
- Not general enough

MayaVi-1.0

mayavi.sf.net


Configure StructuredGridOutline mod
Show Pipeline
Change Object Color
Set Opacity
1.00
Opacity resolution:
0.01
Configure ContourGridPlane module
Show Pipeline
Actor (vtkOpenGLActor)
Mapper (vtkOpenGLPolyDa
LookupTable (vtkLookup
vtkPolyData
vtkExtentTranslator
Source (vtkStructure
vtkStructuredGric
Property (vtkOpenGLProper
Refresh
Expand all
X-Axis
Y-Axis
Z-Axis
Position
15
Set Opacity
1.00
Opacity resolution:
0.01
Line width:
4.0
Show Contours
Number of contours:
10
Minimum contour:
0.0
Maximum contour:
1.391444
Auto Sweep
Auto Sweep Step:
1
Auto Sweep Delay:
1.0
Close
velocity magnitude
0.00
0.199
0.398
0.596
0.795
0.994
1.19
1.39
View:
Front
Back
Left
Right
Top
Bottom
Isometric


VTK documentation browser
Search class names:
Search class docs:
delaunay and triangulation
Search Results:
vtkDelaunay2D
vtkDelaunay3D
vtkOrderedTriangulator
Class Documentation for vtkDelaunay3D
Class Documentation for vtkDelaunay3D
vtkDelaunay3D - create 3D Delaunay triangulation of input points
Super Class:
vtkUnstructuredGridSource
Lookup Table Editor
File
Edit
MayaVi Data Visualizer 1
File
Visualize
Options
1. DataVizManager
Configure Streamlines module
Show Pipeline
Actor (vtkOpenGLActor)
Mapper (vtkOpenGLPolyDa
LookupTable (vtkLookup
vtkPolyData
vtkExtentTranslator
Source (vtkSphereS
Property (vtkOpenGLPrope
Actor (vtkOpenGLActor)
Refresh
Expand all
PointSource0
PointSource1
PointSource2
Add streamline source
Delete streamline source
Save streamlines
Load streamlines
Close
Configure LutHandler module
Show Pipeline
Load Lookup Table
Edit Lookup Table
blue-red colormap
red-blue colormap
black-white colormap
white-black colormap
Use specified data range
Data Range:
(0.0, 1.39144
Use visible range
Visible Range:
(0.0, 1.39144
Show Legend
Horizontal
Vertical
Shadow Legend
Number of Labels:
8
Number of Colors:
256
Legend text:
velocity mag
Close
velocity magnitude
0.00
0.199
0.398
0.596
0.795
0.994
1.19
1.39
View:
+X
-X
+Y
-Y
+Z
-Z
Isometric

- 2001 May
- GUI, CLI
- One month, including docs
- PhD Procrastination Project!

Issues with MayaVi-1.0

- No clean scripting API
- No MVC
- Clunky Tkinter UI
- Not easy to embed

Mayavi2

code.enthought.com/projects/mayavi

2004 –


Apprentice no longer!

Enthought

TVTK + Mayavi2 (2004)

www.enthought.com

The world of Traits

Python object on steroids!

Part of the Enthought Tool Suite (ETS)

TVTK

TVTK = VTK + Traits + NumPy

The whole is greater than the sum of the parts!

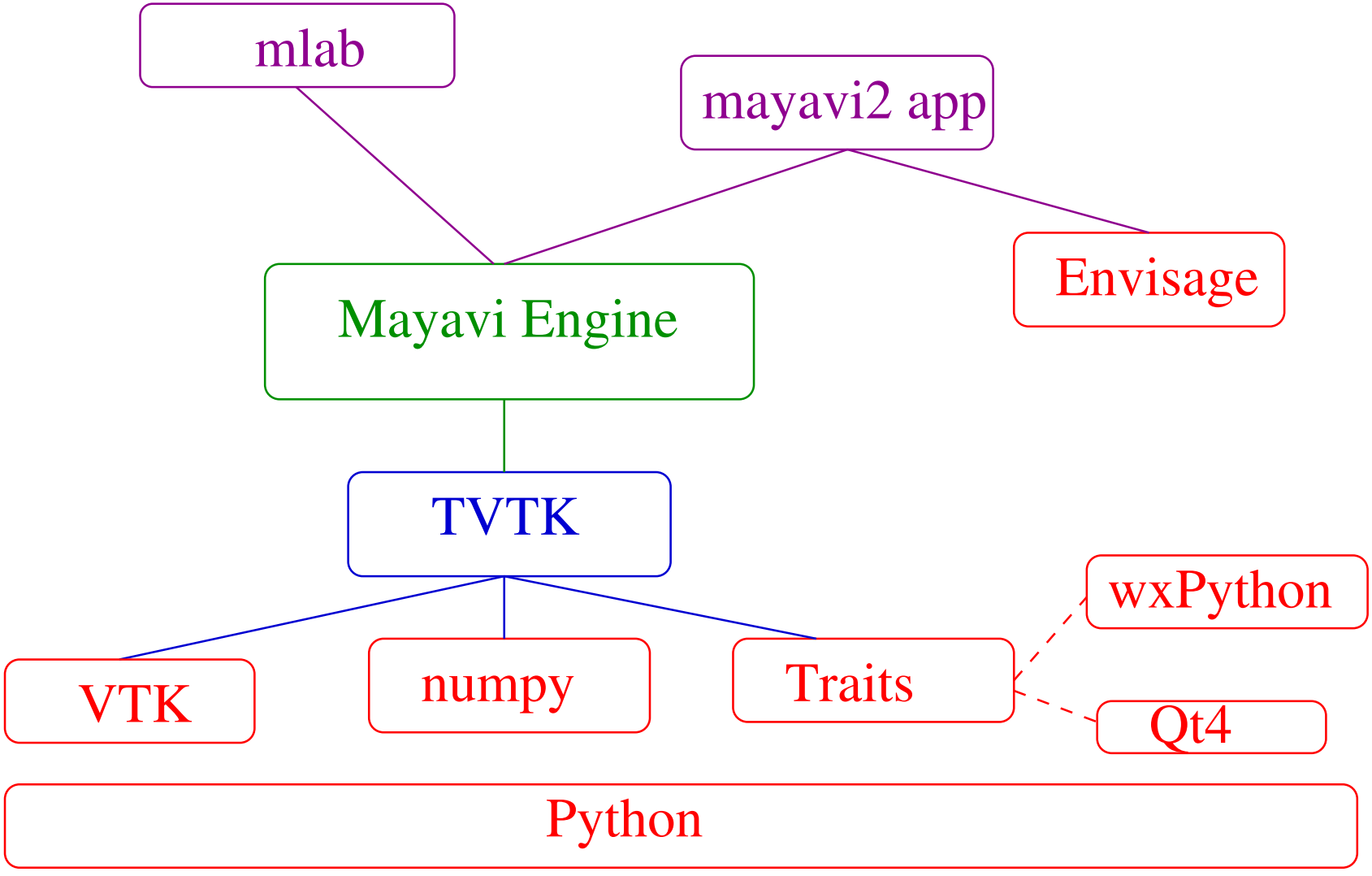
Lessons

- The API matters a lot
- TDD is quite a life-saver
- Be prepared to throw away code!

Mayavi2

- TVTK + Traits + Envisage

Recap of architecture



Developers and support

Prabhu Ramachandran	Creator and lead, 2001 –
Gaël Varoquaux	Mlab, documentation, usability, 2007 – 2012
Enthought Inc.	ETS, Hosting, support, sprints, initial funding, distribution
Deepak Surti	new pipeline support, 2014 – 2016
Enthought Devs	Bug fixes, support, testing: Kit Choi, Ioannis Tziakos
IITB	Freedom and support for PR