

WP4a: Development of diagnostics and visualization tools

Survey on VCS and VTK

Patrick Brockmann
IPSL/LSCE

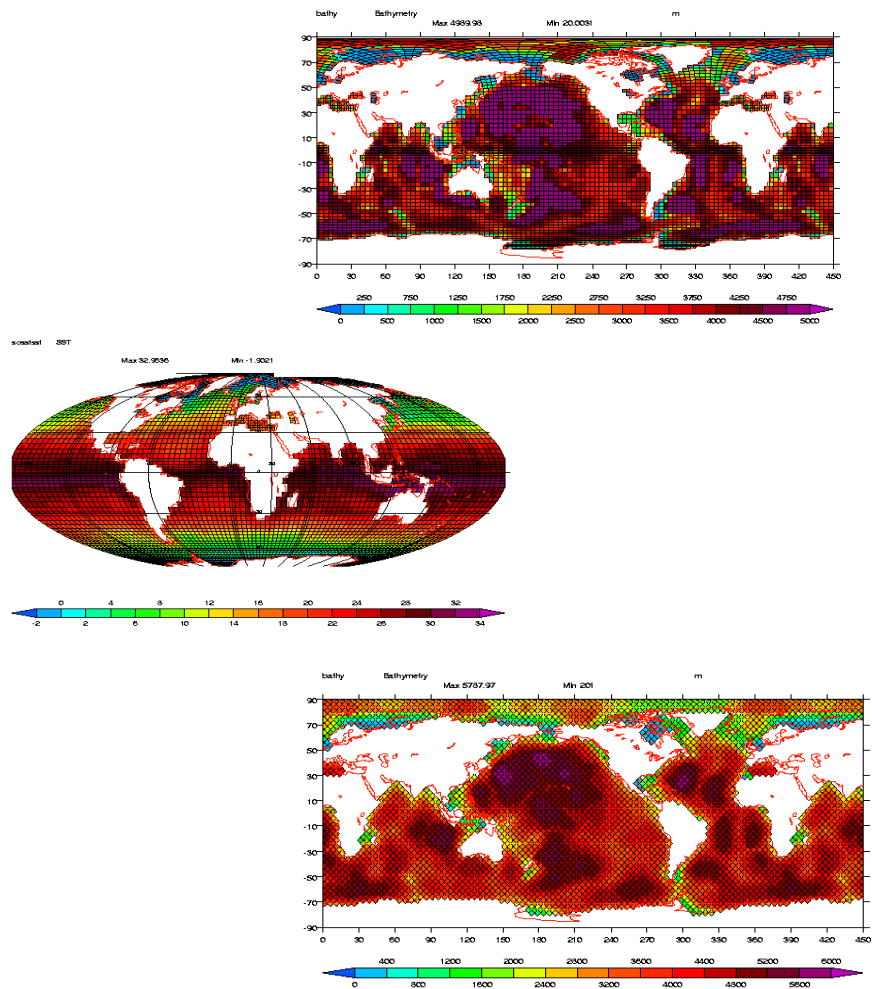
Bracknell 21-22 October 2002

Topics

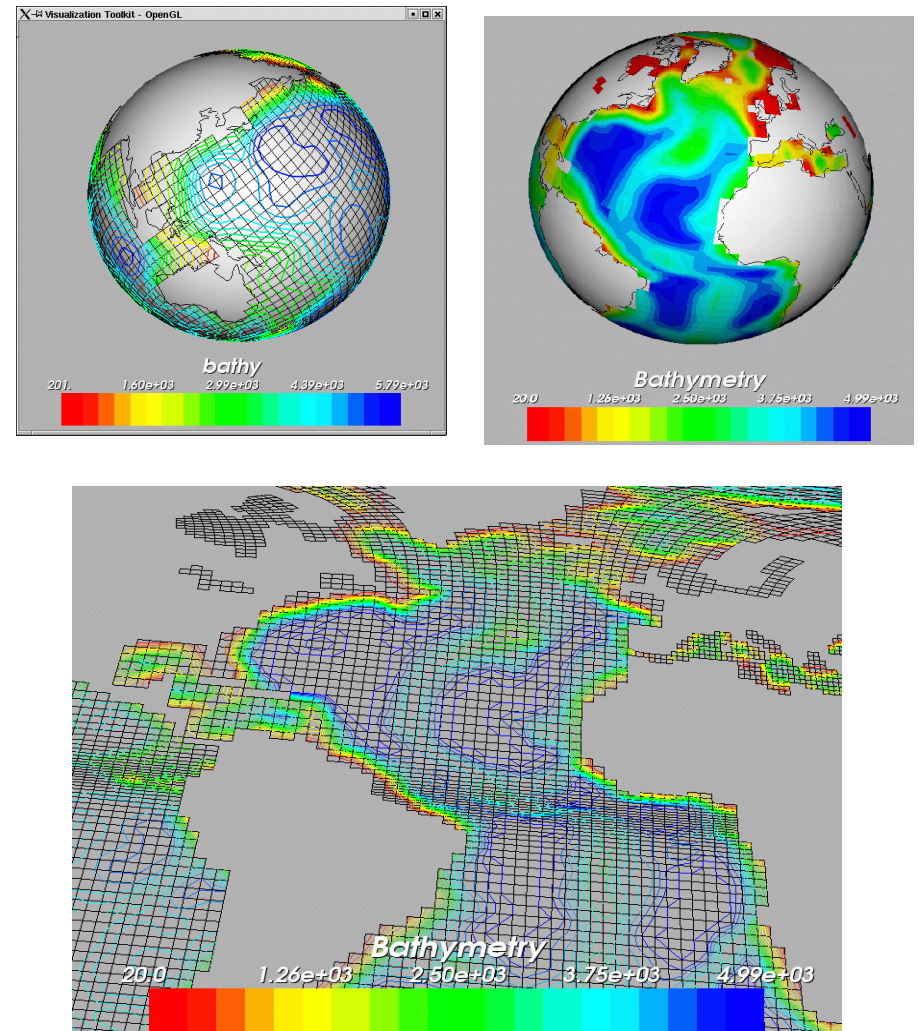
- **Purpose**
 - **Exploring VCS**
 - **Exploring VTK**
 - **Interfacing with CDMS/COCO**
 - **Continuing with Ferret**
 - **DODS server**
 - **Live Access Server**

Some tests

VCS



VTK



What is VCS ?

- VCS for Visualization Control System
- Different from the Visualization and Computation System (also VCS) developed previously by the same people from Program for Climate Model Diagnosis and Intercomparison (PCMDI) of the Lawrence Livermore National Laboratory
- Ressource site: <http://esg.llnl.gov/cdat/>



VCS - Features

VCS (through CDAT) provides capabilities to:

- View, select and modify attributes of data variables and of their dimensions
- Create and modify existing template attributes and graphics methods
- Save the state-of-the-system as a script to be run interactively or in a program
- Save a display as a Computer Graphics Metafile (CGM), GIF, Postscript, Sun Raster, or Encapsulated Postscript file
- Perform grid transformations and compute new data variables
- Create and modify color maps " zoom into a specified portion of a display
- Change the orientation (portrait vs. landscape) or size (partial vs. full-screen) of a display
- Animate a single data variable or more than one data variable simultaneously
- Display different map projections

VCS and CDAT

VCS is used in CDAT

- VCS is based on XGKS
- VCDAT = CDAT+GUI
- VCS does not allow today isocontours with irregular grids
- CDAT and in particular CDMS is not distributed as a DODS client by default

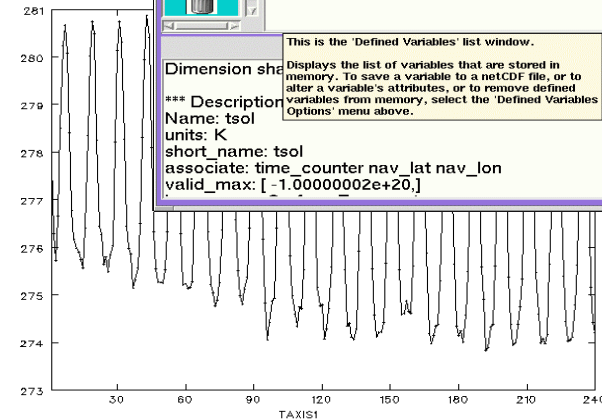
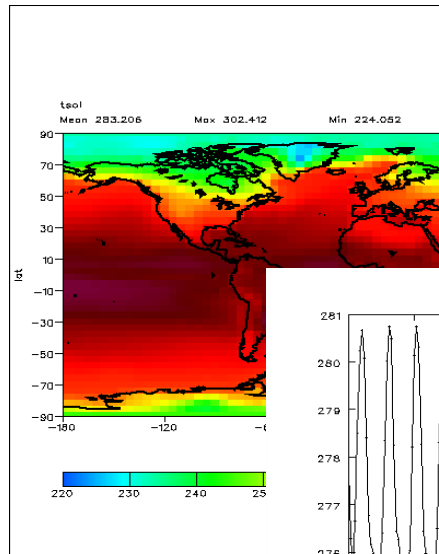
```
File Edit Windows Help
import cdms, vcs, cdutil, genutil, os, sys
from cdms import MV

# Initialize the four VCS Canvases by creating
# a VCS Canvas dictionary
vcs_canvas_dict = {}

# Loop (from 0 to 3) to create VCS Canvas 1, 2, 3, and 4
for i in range(4):
    vcs_canvas_dict[ i ] = vcs.init()

# Run VCS script that contains the appropriate VCS template
# graphics methods, and color maps
for i in range(4):
    vcs_canvas_dict[ i ].scriptrun( '/home_local/brocksce/c
    vcs_canvas_dict[ i ].setcolormap('ASD')
    vcs_canvas_dict[ i ].mode = 0

# Create a dictionary that will contain the defined
# variables that are in memory
slabm={}
os.chdir( '/home_local/brocksce/output' )
```



The Visual Climate Data Analysis Tools - (VCDAT)

File Options Tools PCMDITools Help

Select Variable

Directory: /home_local/brocksce/output

File: CPL06_LMDZ.Y1_8.nc

Variable: tsol

Plot Boxfill VCS Canvas 1 Options Define

time_counter 1999-1-2 0:0:43.0: 1999-1-2 0:0: 1999-1-2 0:0: def

Y-lat -90 : 90 by 1 lat -90 90 def

X-lon -180 : 175 by 1 lon -180 175 def

Defined Variables

This is the 'Defined Variables' list window.

Dimension shows Displays the list of variables that are stored in memory. To save a variable to a netCDF file, or to alter a variable's attributes, or to remove defined variables from memory, select the 'Defined Variables Options' menu above.

*** Description Name: tsol units: K short_name: tsol associate: time_counter nav_lat nav_lon valid_max: [-1.00000002e+20.] nc***

VCS - Demos

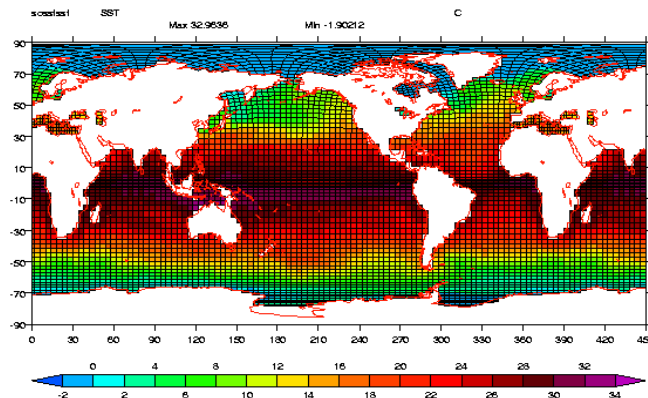
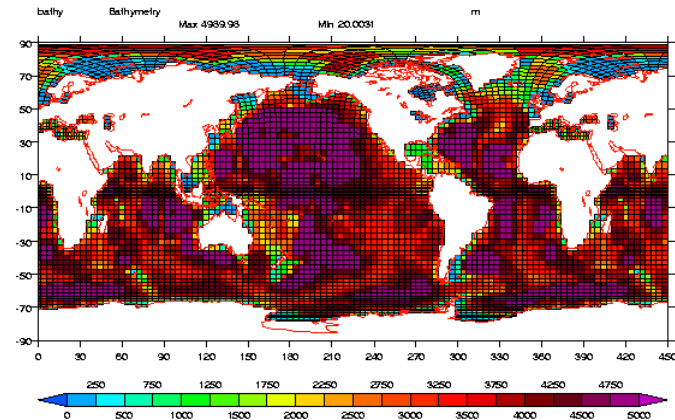
Demos are based on a current work by Charles Doutriaux (LLNL) on the meshfill graphics method which allows display of generalized grids through the vcs interface.

Plotting an array on a generalized grid requires at least 3 arguments:

- 1) The data array to be plotted
- 2) The mesh representing the data array (in y/x space)
- 3) The meshfill graphics method object

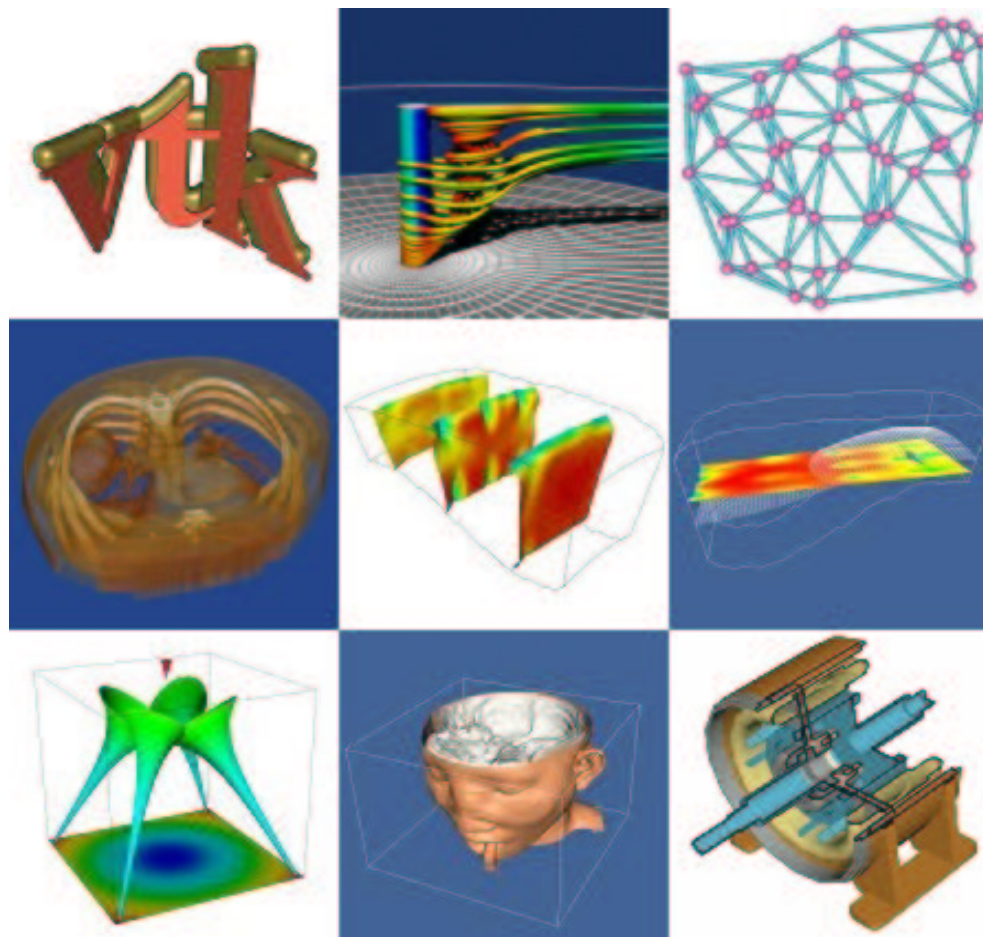
On its simplest form, plotting the array A on the generalized mesh M using the meshfill method m would be:

```
import vcs
x=vcs.init()
x.plot(A,M,m)
```



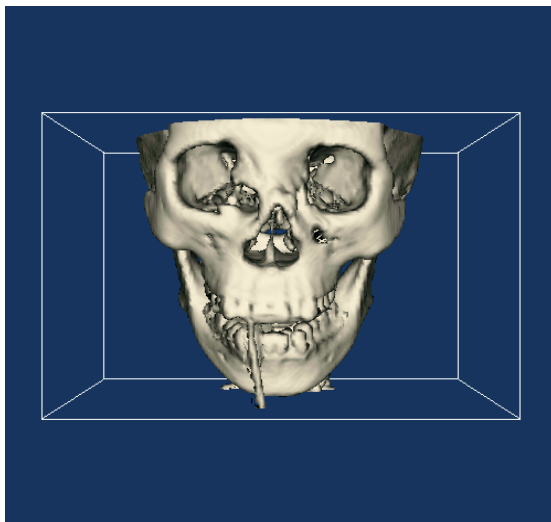
What is VTK ?

- **VTK: The Visualization Toolkit**
- **The Visualization Toolkit (VTK) is an open source, freely available software system for 3D computer graphics, image processing, and visualization.**
- **Support for hundreds of algorithms in visualization and image processing fields**
- **Object Oriented design allows Easy to Use programming interface for access to core components**
- **Several different interpreted language bindings to allow for fast development**

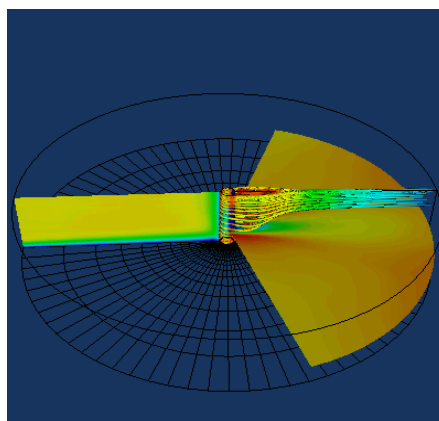


VTK - Applications

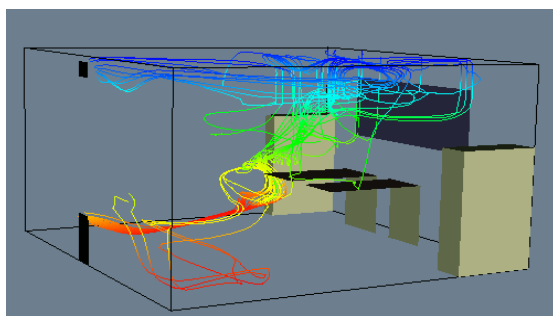
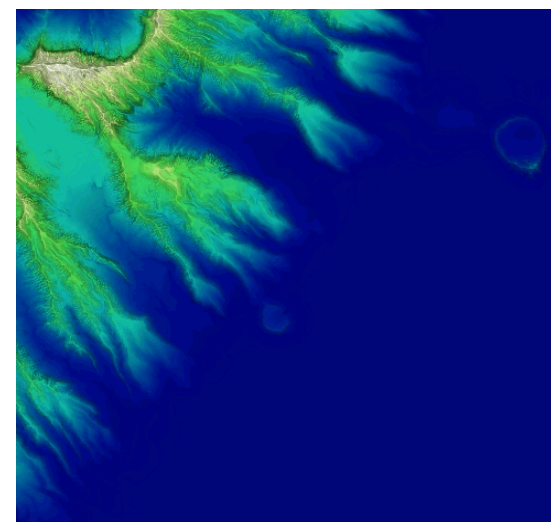
Medical Visualization



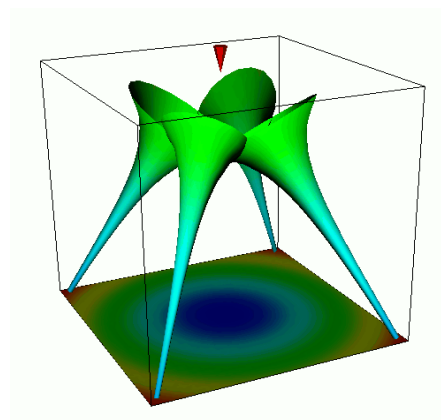
C.F.D



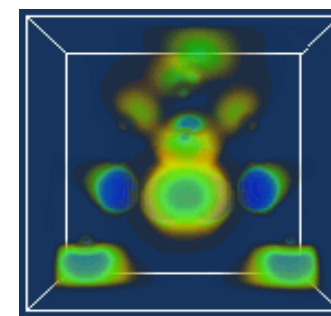
G.I.S



Flow Visualization



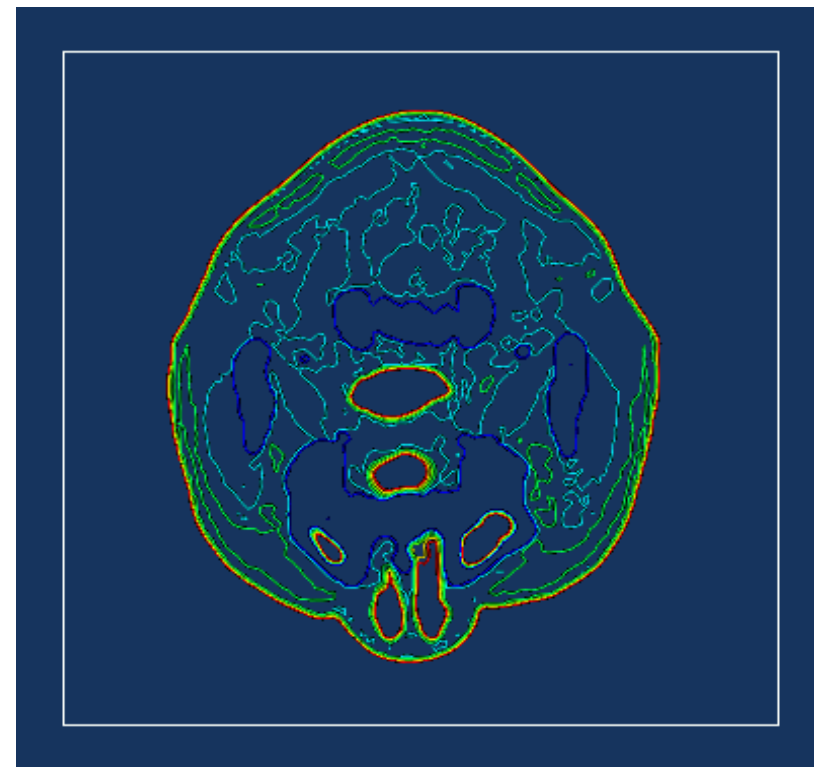
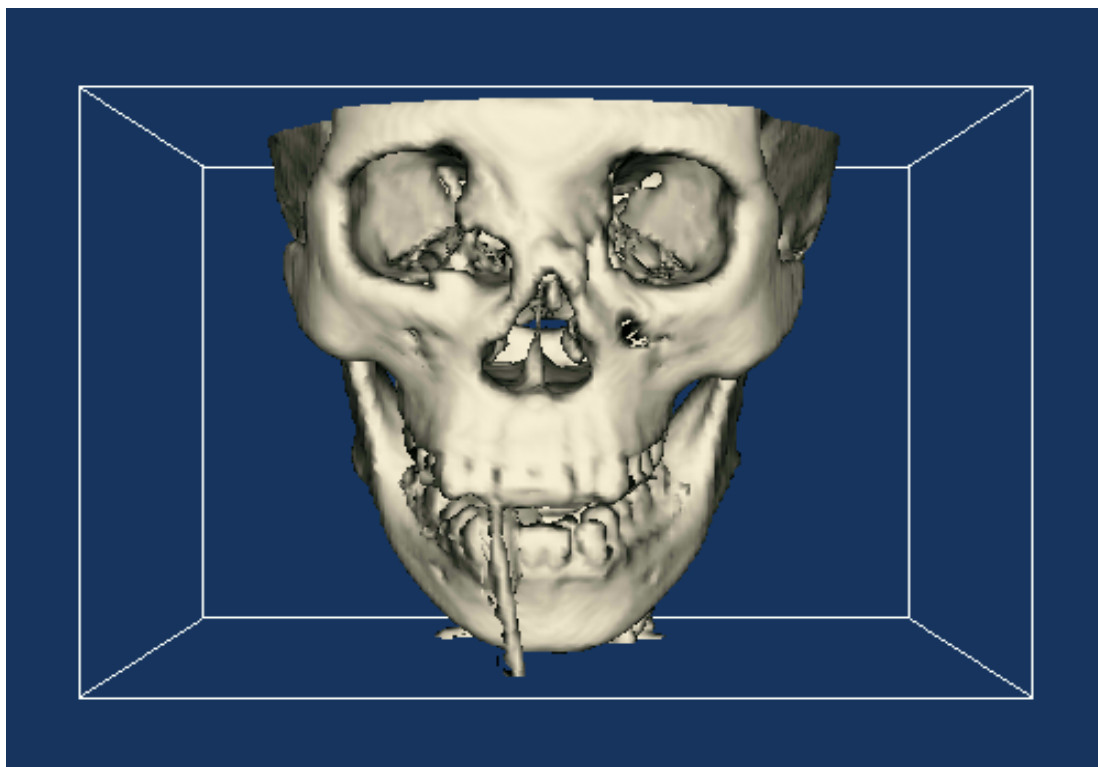
Tensor Visualization



Volume Rendering

VTK - Algorithms

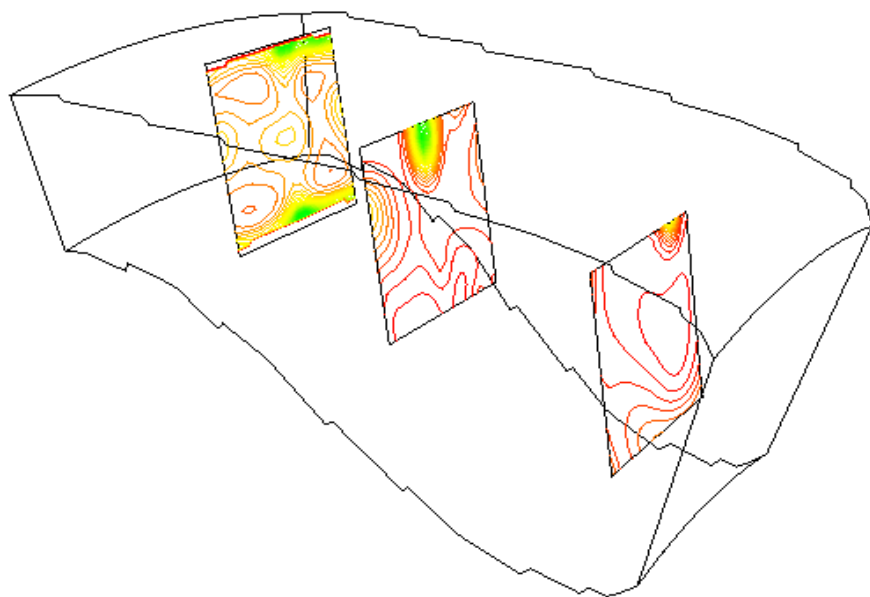
CT Scan of a Human Head



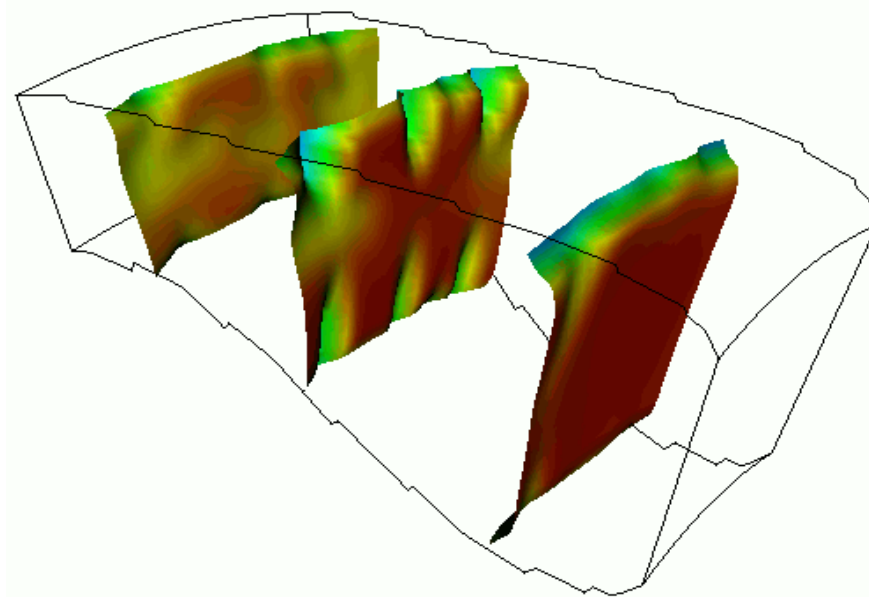
vtkContourFilter & vtkCutter

VTK - Algorithms

Flow density and energy in a Combustion Chamber



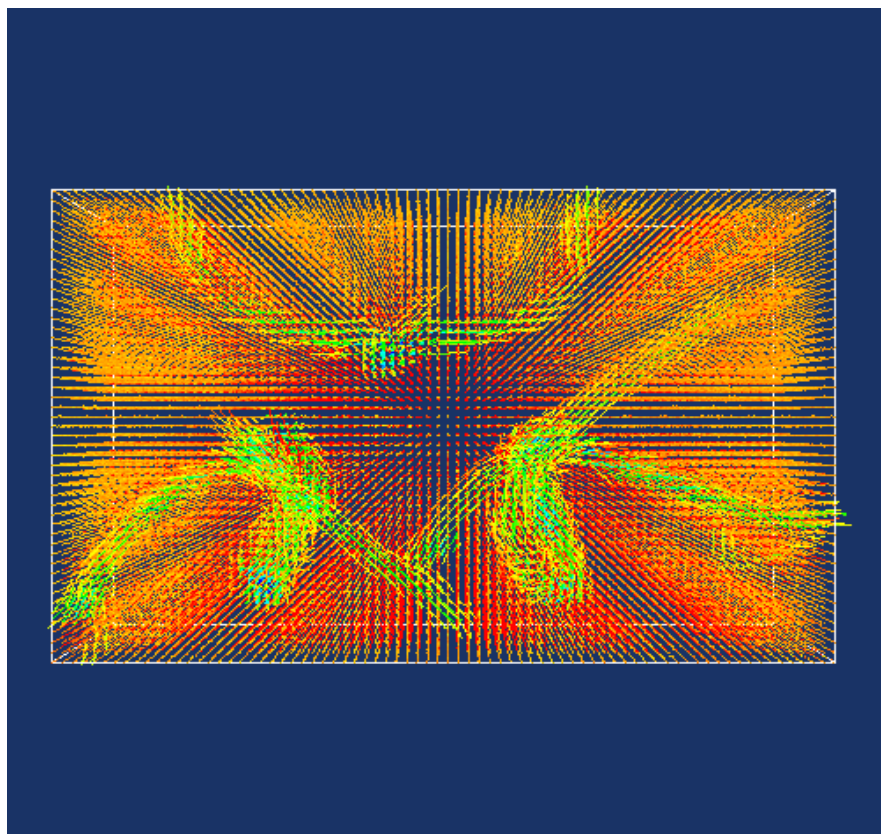
vtkProbeFilter & vtkContourFilter



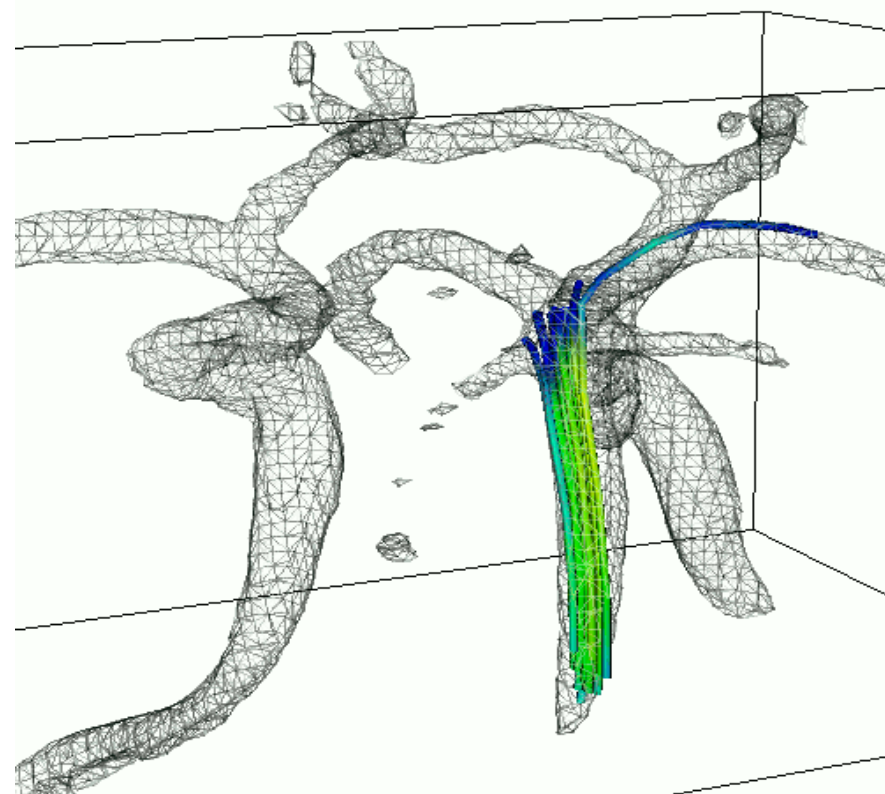
vtkCutter & vtkWarpVectors

VTK - Algorithms

Blood flow in the Human Carotid Arteries



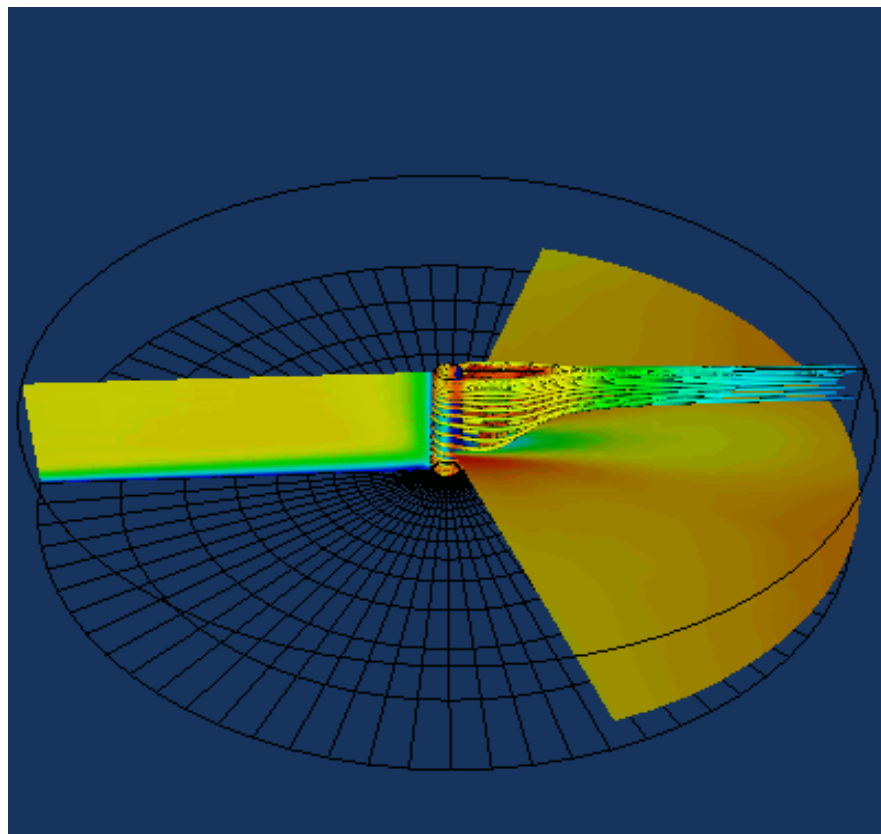
vtkHedgeHog



vtkContourFilter, vtkStreamLines & vtkTubeFilter

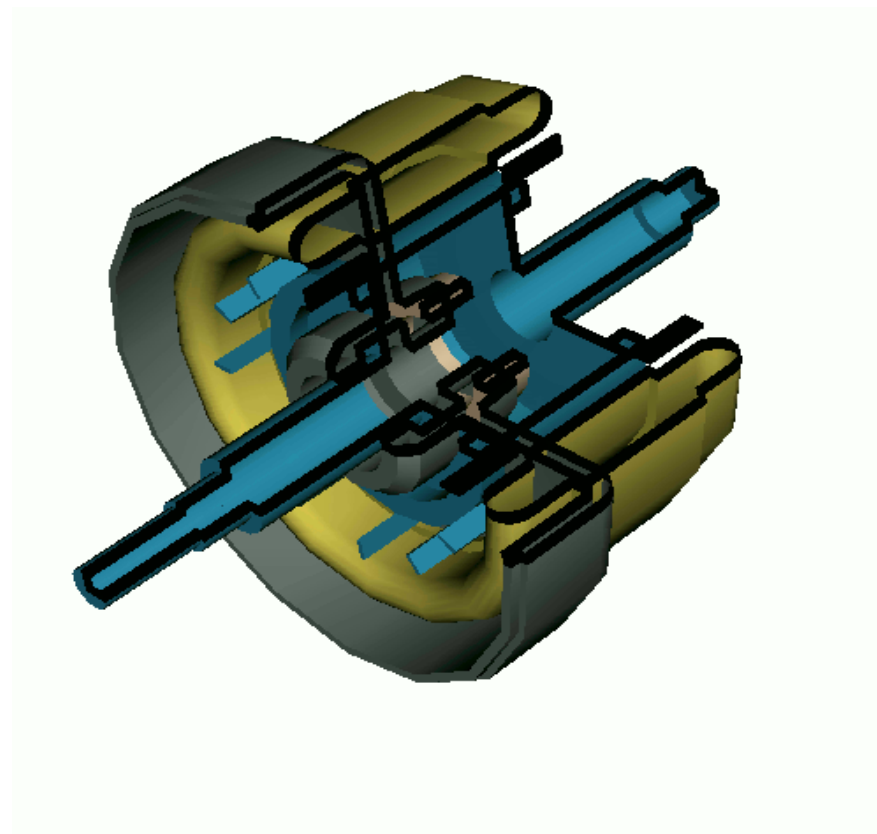
VTK - Algorithms

Verticle Post in Fluid Flow



**vtkStreamLines, vtkTubeFilter,
vtkDataSetMapper, & vtkCutter**

CAD Model of a Motor



**vtkPolyDataMapper, &
vtkTextureThreshold**

VTK - Philosophy

Interpreted Language Interface

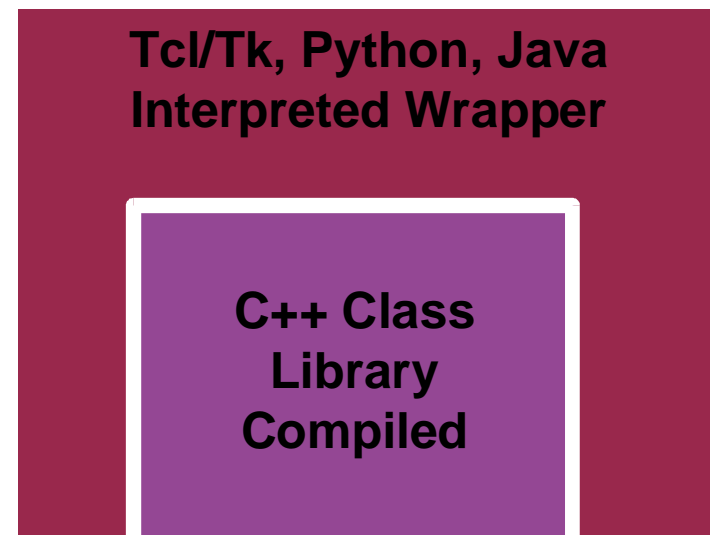
Compiled languages

- Faster
- Low level access

Interpreted

- Flexible
- Rapid app building
- Higher level
- Simpler, compact
- Easier to debug

Compiled C++ core



VTK - Pipeline data-flow approach



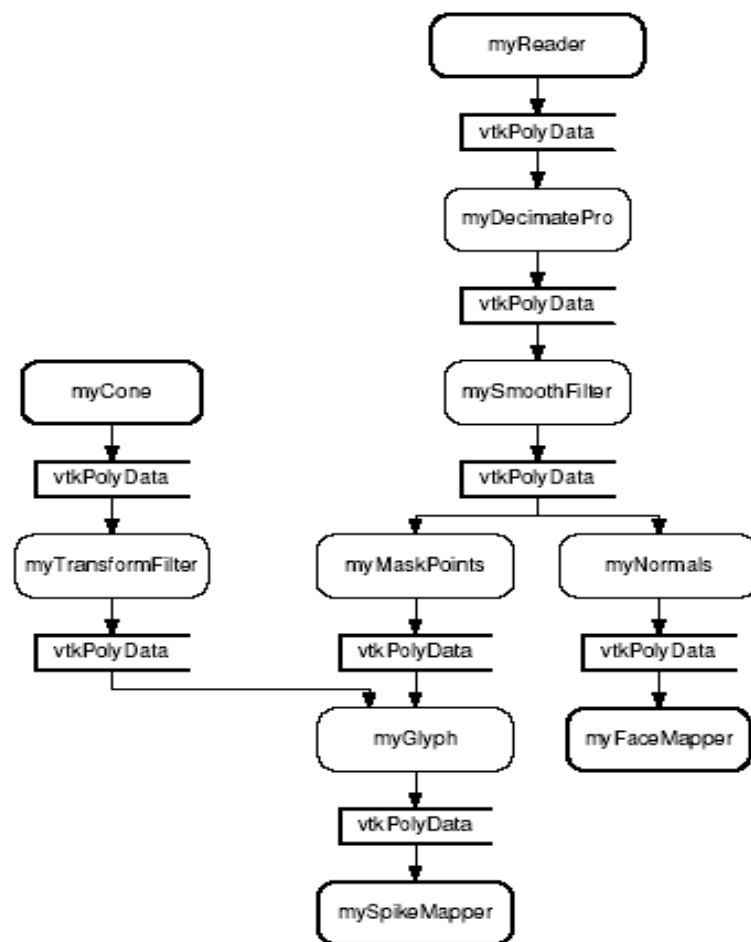
A Pipeline Consists Of:

Source Objects: interface to external data, or generate data based on local parameters (e.g. `vtkPLOT3DReader`)

Filter Objects: operate on data from Source objects and generate geometry and/or images (e.g. `vtkContourFilter`)

Mapper Objects: take the resulting data from a Filter Object and process it for display or writing to a file (e.g. `vtkPolyDataMapper`)

VTK - Example of pipeline



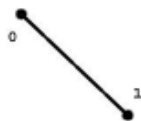
VTK - Data Structures



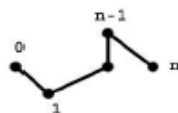
(1) Vertex



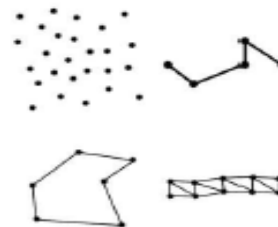
(2) Poly-vertex



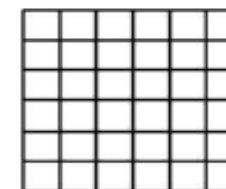
(3) Line



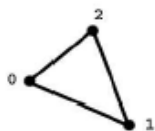
(4) Poly-line



(1) Polygonal data



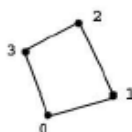
(2) Structured points



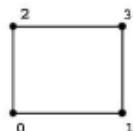
(5) Triangle



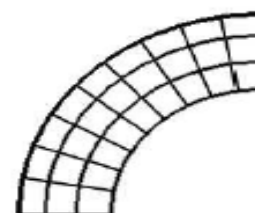
(6) Triangle strip



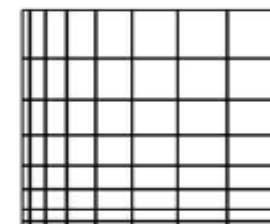
(7) Quadrilateral



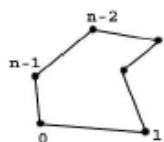
(8) Pixel



(3) Structured grid



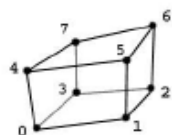
(4) Rectilinear grid



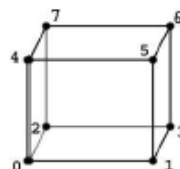
(9) Polygon



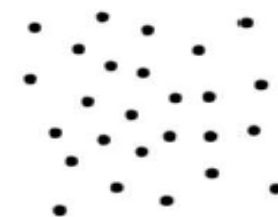
(10) Tetrahedron



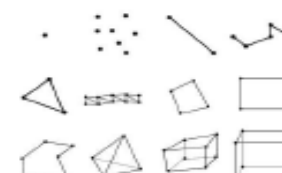
(11) Hexahedron



(12) Voxel



(5) Unstructured points



(6) Unstructured grid

VTK - 1st example

Python script

```
#!/usr/bin/env python

from vtkpython import *

# create a rendering window and renderer
ren = vtkRenderer ()
ren.SetBackground ( 0.5, 0.5, 1. )

renWin = vtkRenderWindow ()
renWin.AddRenderer ( ren )
renWin.SetSize ( 300, 300 )
renWin.StereoCapableWindowOn ()

# create an interface interactor
iren = vtkRenderWindowInteractor ()
iren.SetRenderWindow ( renWin )

# create an actor and give it cone geometry
cone = vtkConeSource ()
cone.SetResolution ( 8 )

coneMapper = vtkPolyDataMapper ()
coneMapper.SetInput ( cone.GetOutput () )

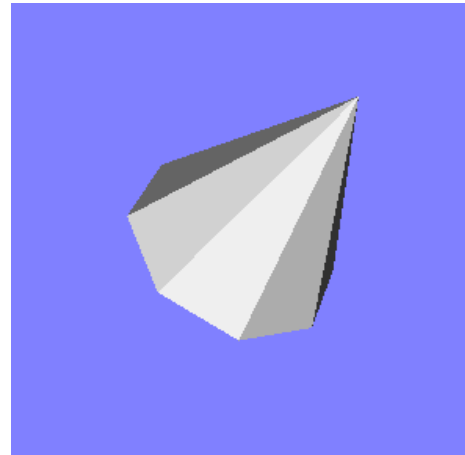
coneActor = vtkActor ()
coneActor.SetMapper ( coneMapper )

# assign our actor to the renderer
ren.AddActor ( coneActor )

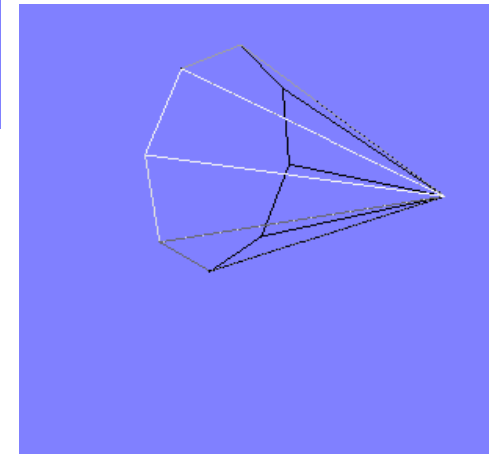
# enable user interface interactor
iren.Initialize ()

iren.Start ()
```

Output

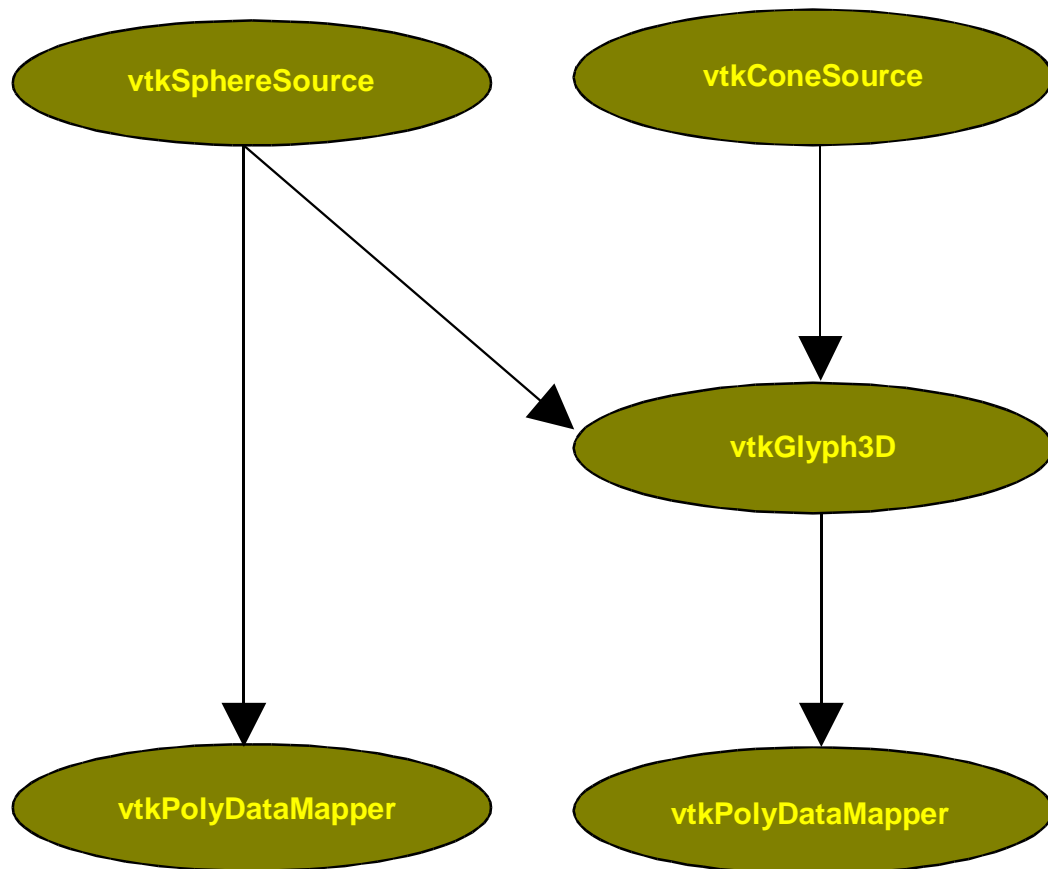


Cone.py



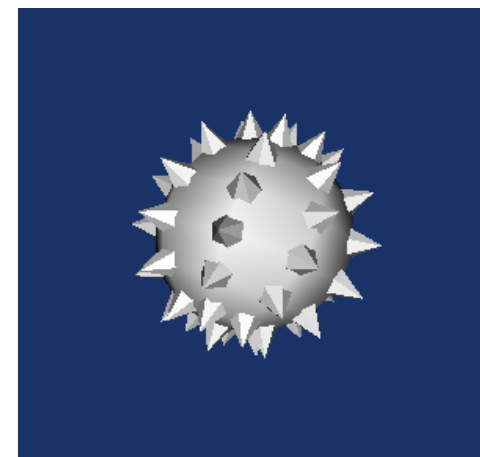
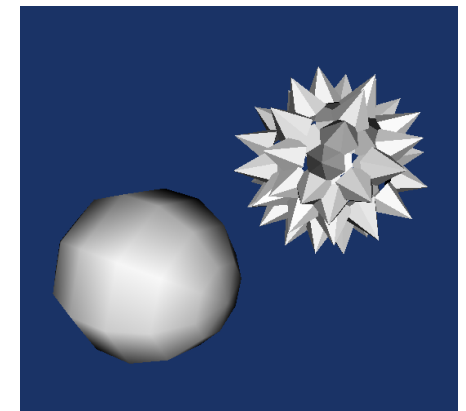
VTK - 2nd example

PIPELINE



OUTPUT

Mace.py



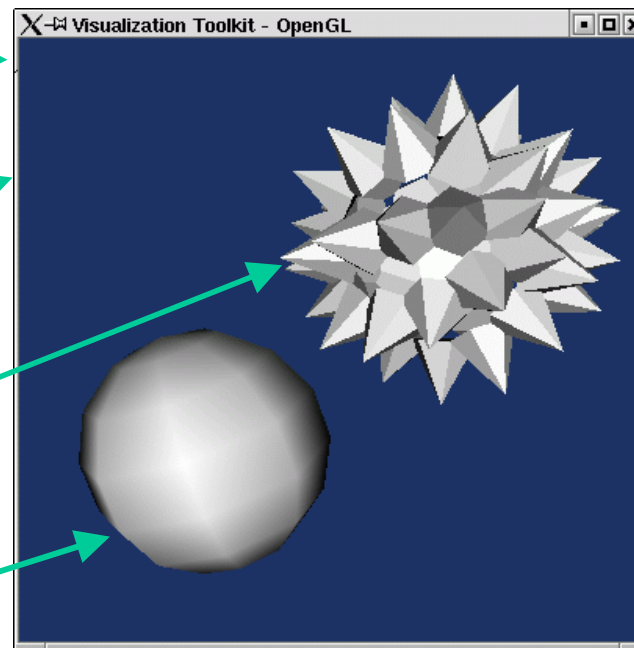
VTK - The rendering

RenderWindow

Renderer

Actor 1

Actor 2

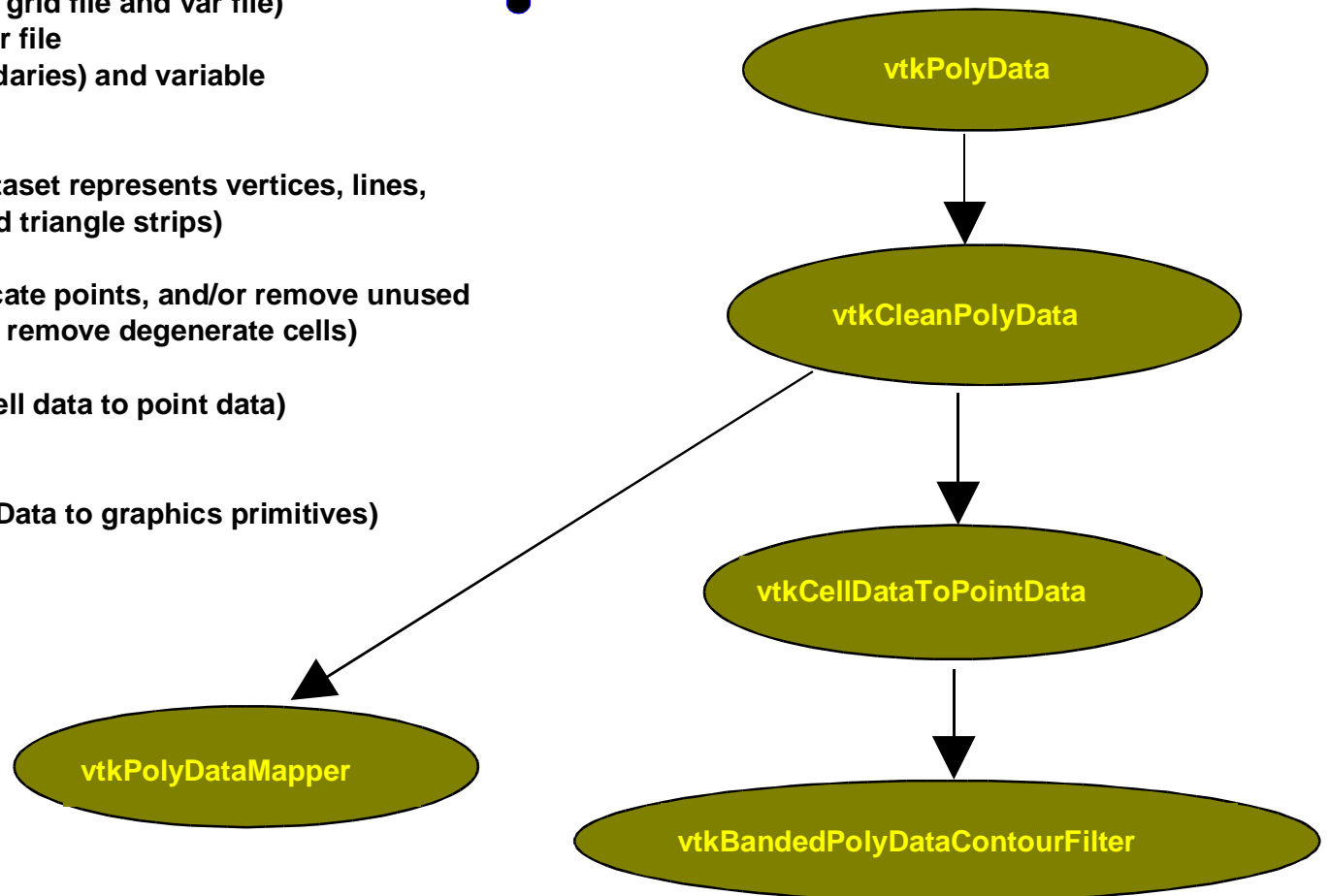


VTK - Demos



- Reading with CDMS netCDF files (CF grid file and var file)
- Extracting mask from grid file and var file
- Flat and Compress points (cell boundaries) and variable

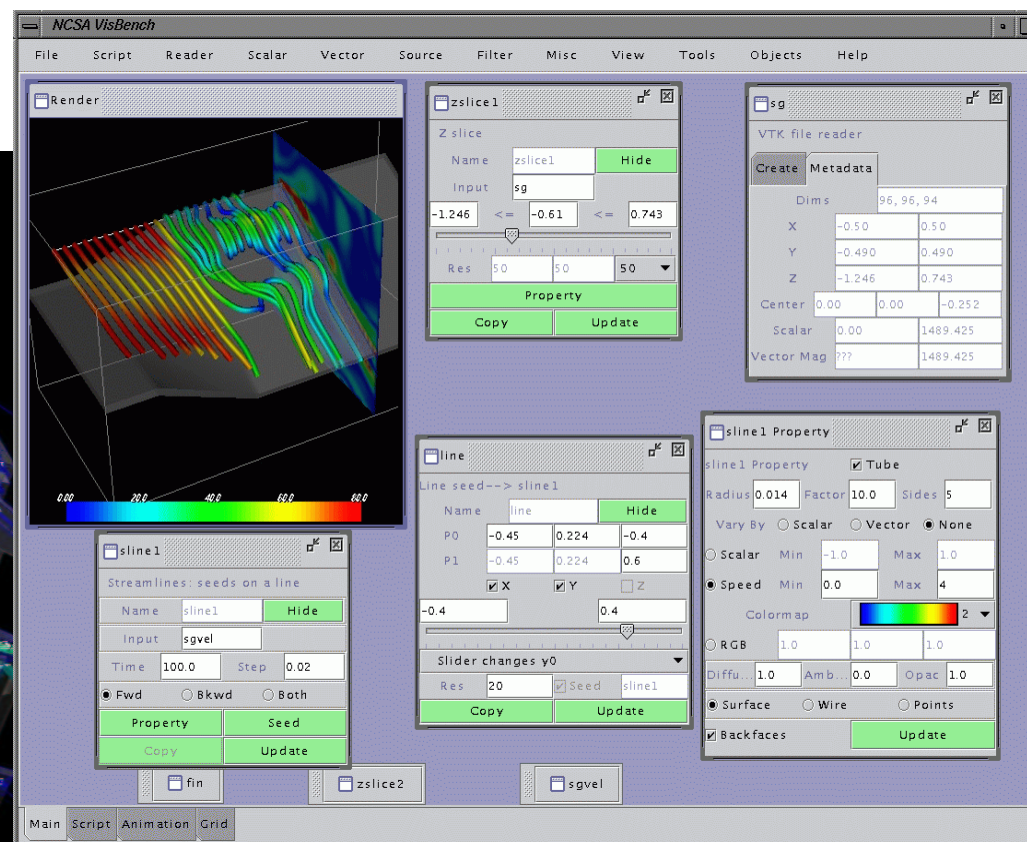
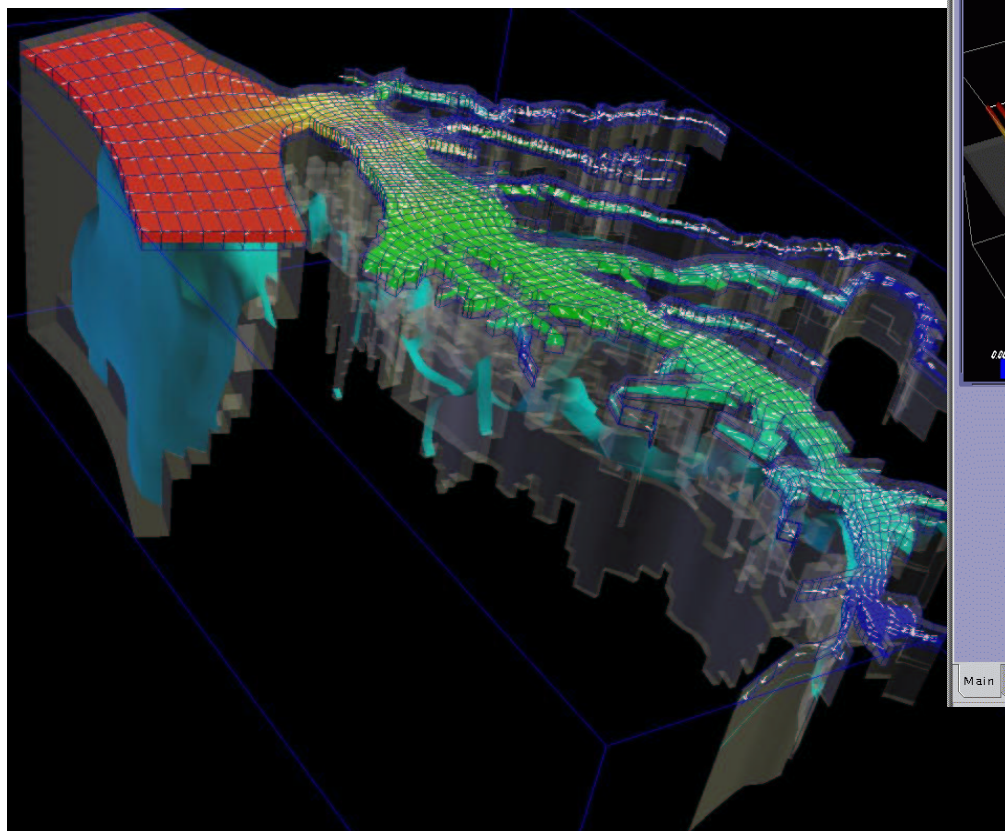
- **vtkPolyData** (Concrete dataset represents vertices, lines, polygons, and triangle strips)
- **vtkCleanPolyData** (Merge duplicate points, and/or remove unused points and/or remove degenerate cells)
- **vtkCellDataToPointData** (Map cell data to point data)
- **vtkPolyDataMapper** (Map vtkPolyData to graphics primitives)



Projects - Activities based on VTK

From the NCSA - National Center for Supercomputing Applications

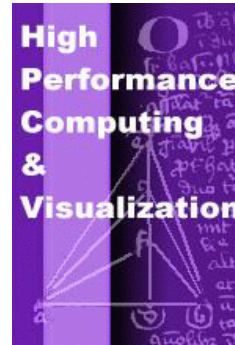
- VisBench <http://www.ncsa.uiuc.edu/Vis/Projects/VisBench/>
- VizGen <http://wasatch.ncsa.uiuc.edu/~rstein/Projects/VizGen.html>



Projects - Activities based on VTK

From the University of Groningen (RuG)

« Visualisation software

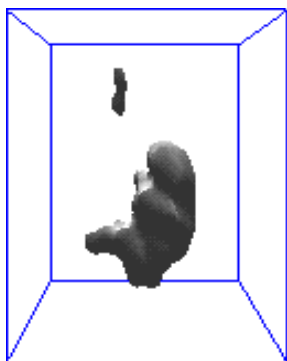


AVS and its successor AVS/Express are supported by the Rekencentrum. Due to a Site-Wide Campus Agreement, the use of these products is especially attractive for members of the FWN faculty.

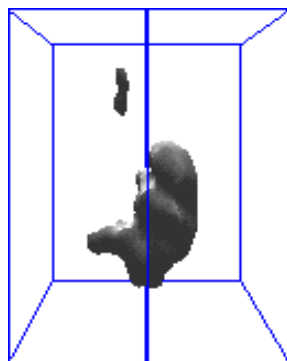
Although the user interface of both AVS-products is very attractive, **some users prefer to apply the public domain Visualisation ToolKit VTK for advanced problems**. The Rekencentrum developed a large collection of examples of VTK: the RugVTK-examples. These examples embody experience with a lot of practical scientific visualisation applications. »

Projects - Activities based on VTK

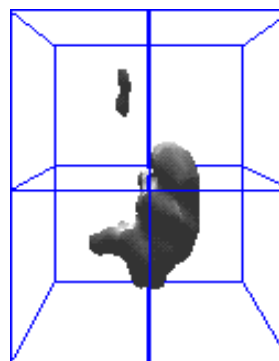
1 pipeline



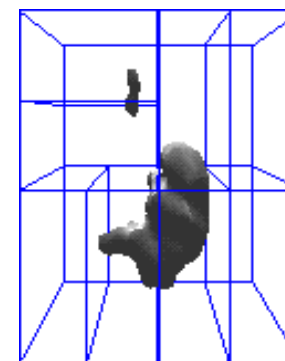
2 pipelines



3 pipelines



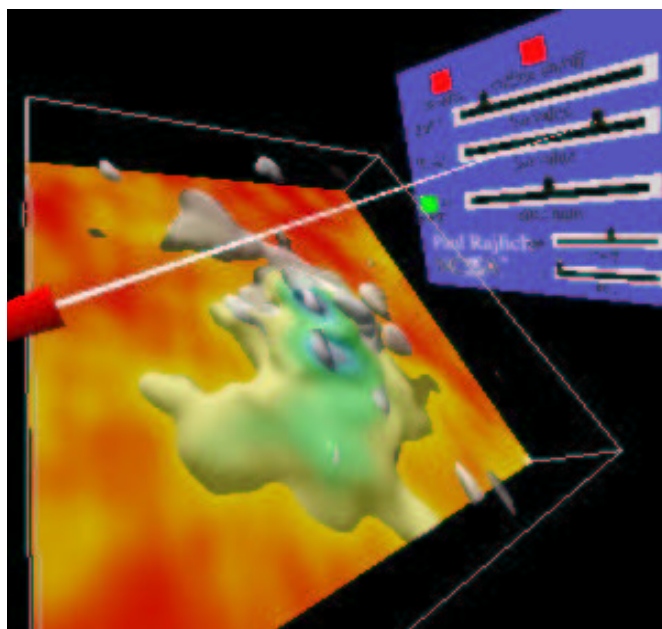
4 pipelines



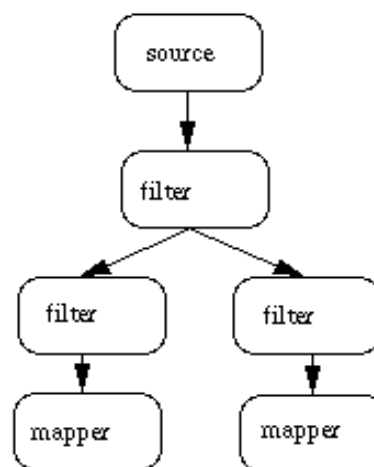
Parallel VTK

**Uses the `vtkActorToPF` framework to parallelize VTK.
This allows users to view very large datasets interactively**

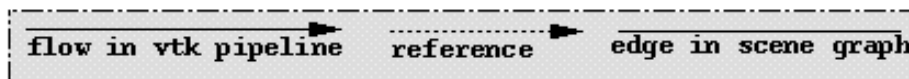
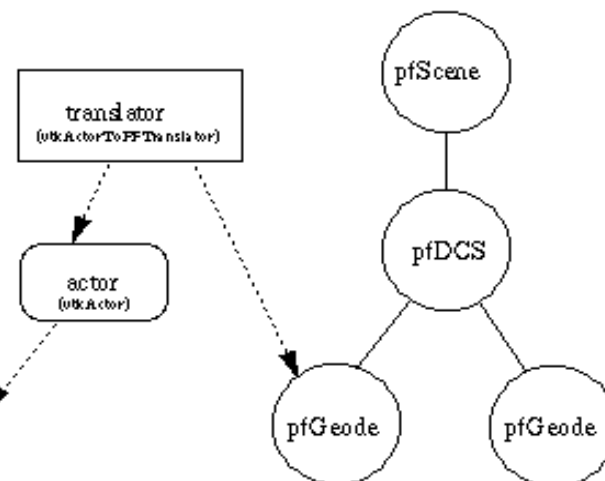
Projects - Activities based on VTK



VTK pipeline



Performer scenegraph



VTK in a CAVE

vtkActorToPerformer allows users to display geometry generated by VTK in the CAVE via an interface to the IRIS Performer Scene Graph

CAVE = CAVE Automatic Virtual Environment

VTK within PRISM

- **Evaluation/development should be continued, at least to choose between 2 « high end » visualization packages**
- **Extremely powerful, but ...**
 - **Also complex with a steep learning curve**
(Everything should be as simple as possible, but no simpler)
- **Works directly with CDMS and all Python libraries**
 - **Numpy, CDMS, PIL (Python Image Library)**
 - **Pmw (Python Megawidgets) based on Tkinter**
- **As also 1D graphic primitives**
- **Clipping, decimation usefull for zonal averaging**
- **Simultaneous probing**

VTK within PRISM

- Vector postscript output
« VTK uses mainly OpenGL for rendering, so it can not produce vector graphics or high quality postscript output. »
- Existence of a C++ library gl2ps

Objet: Re: [vtkusers] Writing Postscript files from VTK

Date: Fri, 04 Oct 2002 14:05:28 +0100

De: Goodwin Lawlor <goodwin.lawlor@ucd.ie>

A: "Patrick Brockmann (LSCE)" <Patrick.Brockmann@ipsl.jussieu.fr>

Hi Patrick,

I'm just finishing a vtk class "vtkPostScriptExporter" which uses the gl2ps code to write out vector eps files... I could mail it to you when its finished and tested. You should be able to use the class in tcl and python as with any vtk class.

*Goodwin
University College Dublin*

VTK vs DX

- OpenDx is a framework, VTK is an API
- Py-OpenDX

« OpenDX is the open-source version of the IBM Data Explorer (DX). DX is a visualization system providing a full set of tools for manipulating, rendering and animating data, especially 3D data from simulations or acquired from observations. It provides a GUI, a scripting interface and the API C libraries. Py-OpenDX is a Python binding for the OpenDX API. Currently only the DXLink library is wrapped. That wrapper allows one to start up a DX executive and communicate with it via the DXL API. »

VTK vs DX

- **Visualization Development Environments 2000 Proceedings**

« Integration of C++ digital processing libraries and VTK through Tcl/Tk dynamic loadable extensions »
Javier Suárez-Quiros, Daniel Gayo-Avello, Juan David González-Cobas, Rafael Pedro García-Díaz, Pedro Ignacio Álvarez-Peñín
Area of Graphic Expression in Engineering GIworks (University of Oviedo)

« ... a very similar approach as Ousterhout's: to provide very efficient compiled code that is invoked by simple commands on a very high level script language. (...)

VTK doesn't only allow to develop fast but rather it provides really advanced characteristics that make of it a solid base for any kind of scientific visualization:

- * The working way follows the classic rendering pipeline, so any person who knows the basic 3D graphics theoretical concepts can understand the way the library classes are used and how to obtain results from just the first moment.
- * The objects provided by library kernel are easy to understand; this way, it provides `vtkActor`, `vtkLight`, `vtkCamera`, `vtkProperty`, `vtkTransform`, `vtkRenderer` and `vtkRenderWindow` among others. Without knowing anything about VTK, the reader can already imagine what they make and how they are linked together to obtain the results (remember the pipeline).
- * It provides high performance characteristics such as level of detail actors, implicit modelling, hierarchical assemblies, real-time interaction, stereography, etc.
- * It is extensible so a big number of classes are contributed by different people, so the package is always on development, earning more and more functionality.
- * There are image, geometry and scene importers and/or exporters available for the most extended formats. »

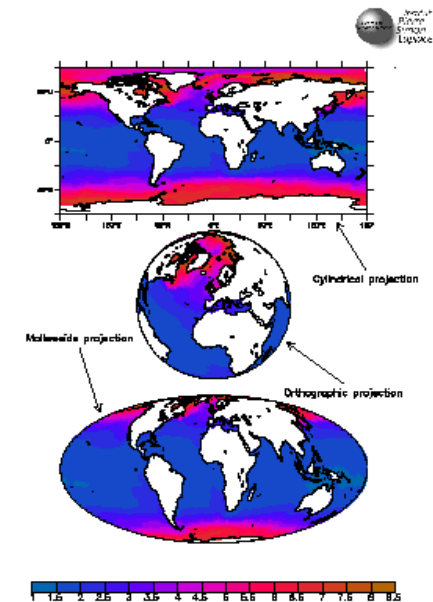
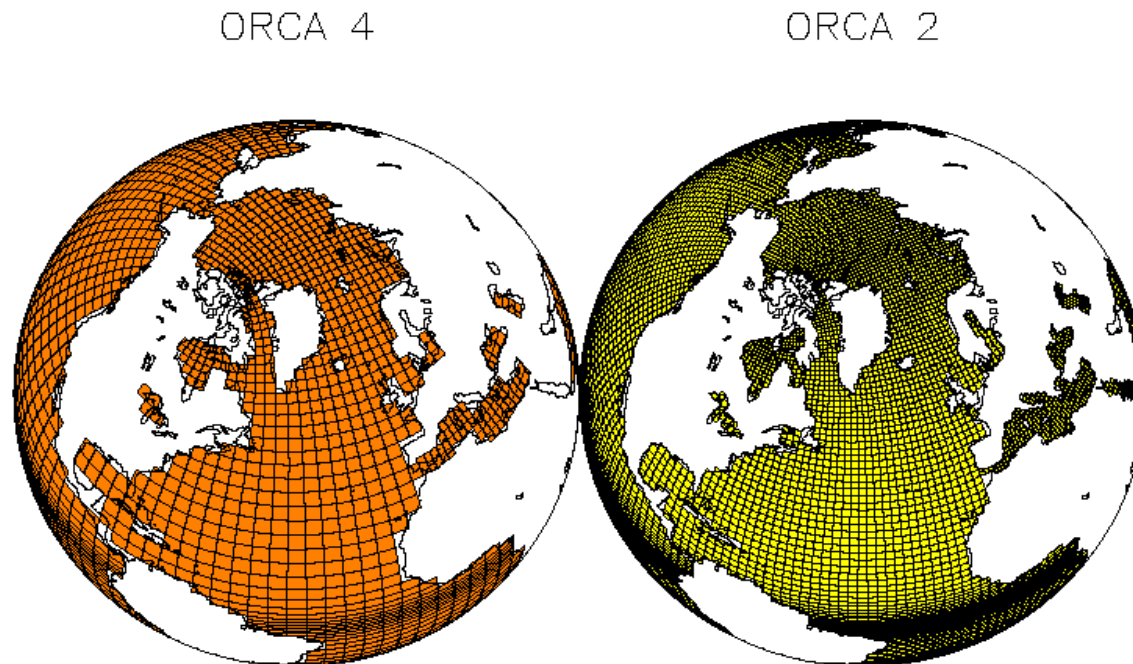
VTK - Ressources

- VTK HomePage - <http://www.kitware.com>
- William J. Schroeder, Kenneth M. Martin, and William E. Lorensen
The Visualization Toolkit: An Object Oriented Approach to 3D Graphics, 2nd edition. Prentice Hall PTR, 1998
- William J. Schroeder, Kenneth M. Martin, and William E. Lorensen.
The design and implementation of an object-oriented toolkit for 3D graphics and visualization. In *IEEE Visualization*, pages 93-100, 1996

Ferret

Daily used

- Added with FAST (Ferret Analysis Scripts Toolbox), local scripts library
- To explore datasets
- To make automatic diagnostic and validation with model output

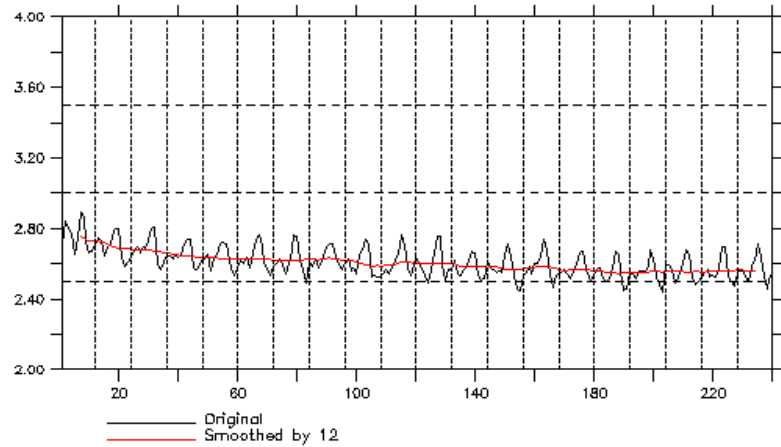


Ferret - Online diagnostics

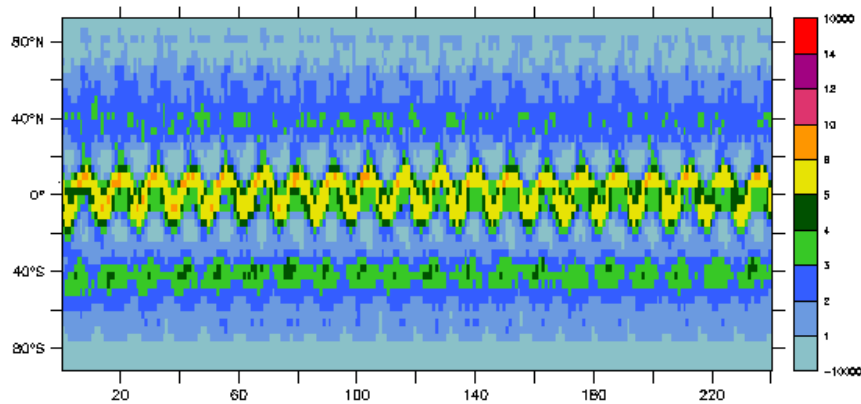
CPL84_tg_precip.nc
(tg_precip[l=1:240]*86400)



Global average



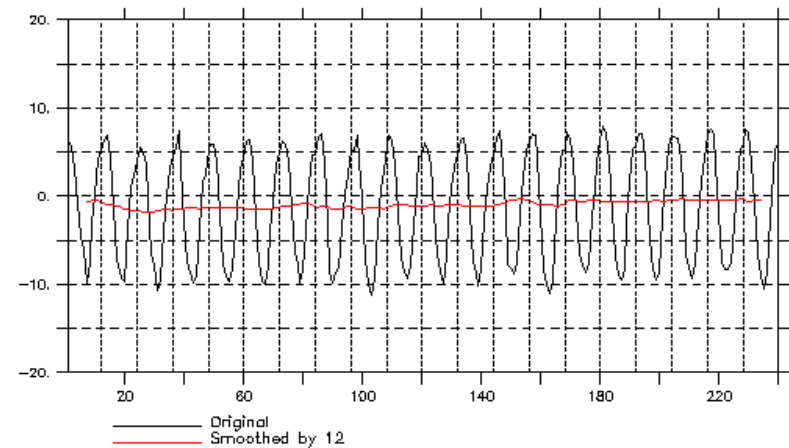
Meridional average



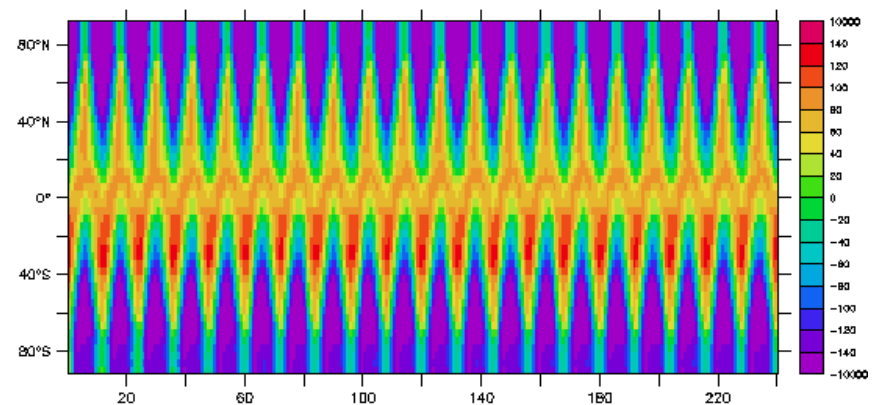
CPL84_tg_nettop.nc
tg_nettop[l=1:240]



Global average



Meridional average



Ferret - Atmosphere model diagnostics

LH (W/m²)

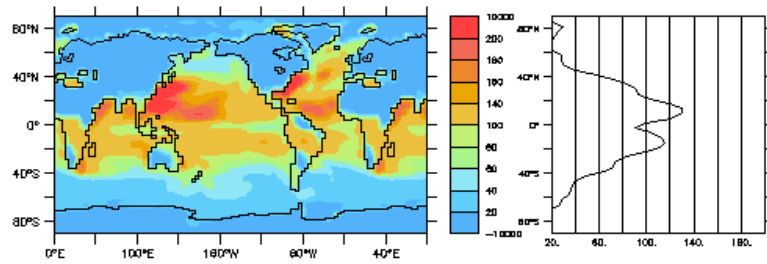


SST (C)

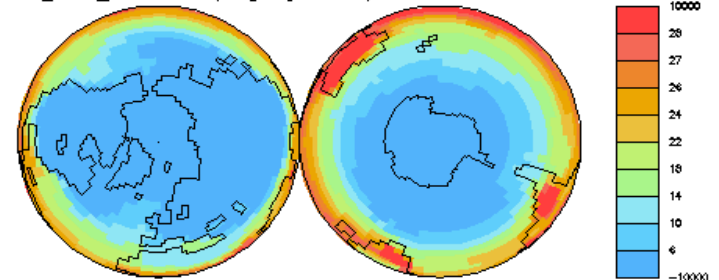


CPL84_SE_0011_0020_histmth.nc: (evap[l=1])*86400/0.03456)

CPL84_SE_0011_0020_histmth.nc: (tsol[l=1])-273.15)



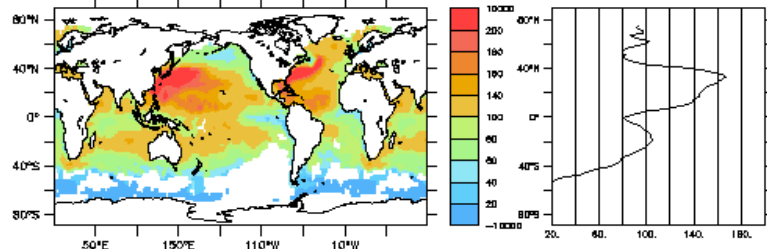
Avg: 58.853 Std: 54.913 Min: -4.824 Max: 327.873



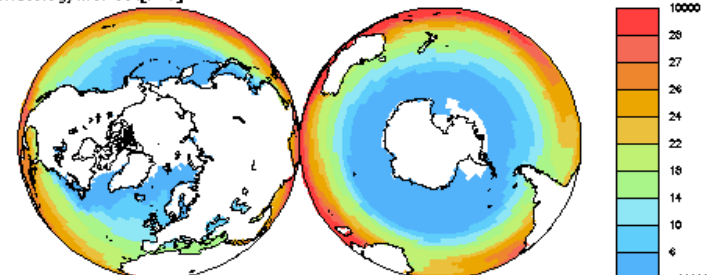
Avg: 1.052 Std: 23.456 Min: -45.854 Max: 37.724

oberhuber_climatology.nc: (-1*lh[l=1])

set8590_climatology.nc: set[l=1]



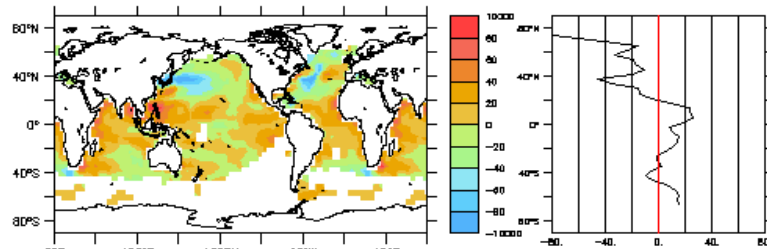
Avg: 95.985 Std: 50.778 Min: -5.7 Max: 338.4



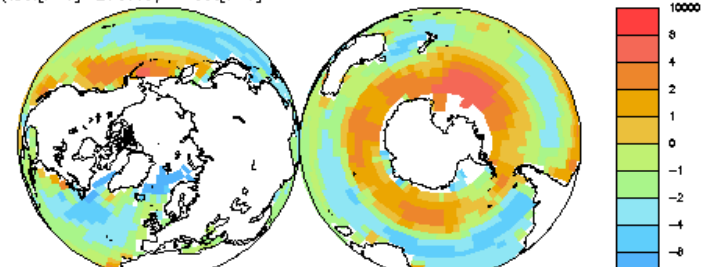
Avg: 16.464 Std: 10.262 Min: -1.79 Max: 30.053

Difference: (evap[l=1])*86400/0.03456) - (-1*lh[l=1])

Difference: (tsol[l=1])-273.15) - set[l=1]



Avg: 0.783 Std: 30.458 Min: -185.85 Max: 118.04



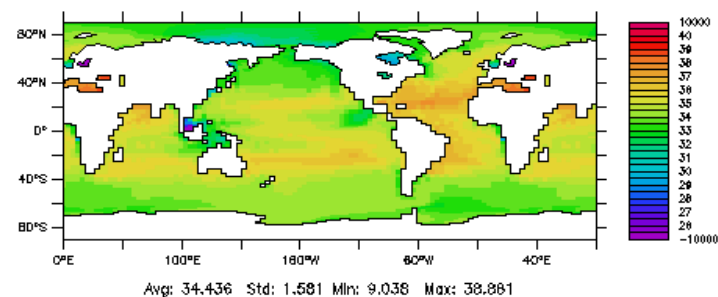
Avg: -1.163 Std: 3.64 Min: -38.416 Max: 6.22

Ferret - Ocean model diagnostics

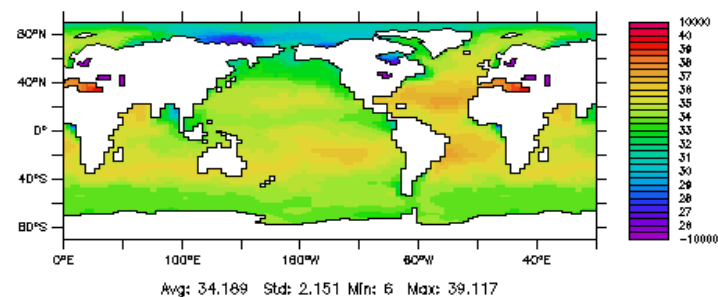
Sea Surface Salinity



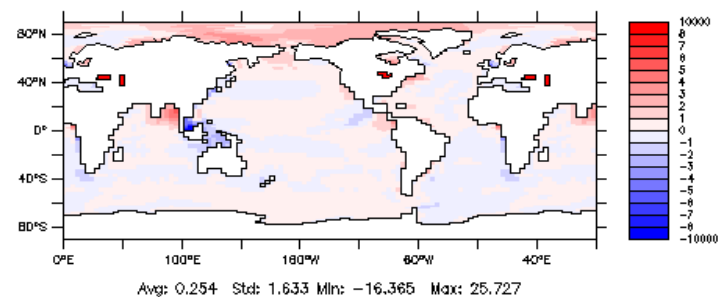
CPL84_SE_0011_0020_grid_T.nc: vosaline[k=1,l=1]



ORCA4.0_Levitus.nc: vosaline[k=1,l=1]



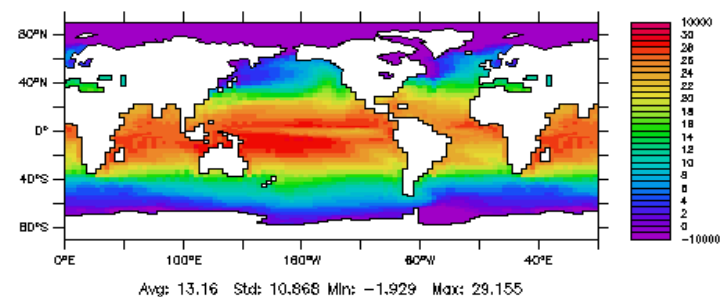
Difference: vosaline[k=1,l=1] - vosaline[k=1,l=1]



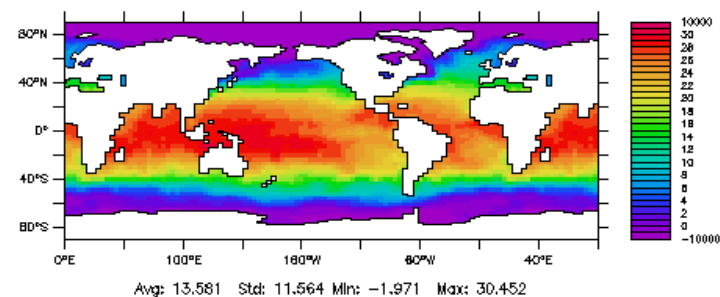
Sea Surface Temperature



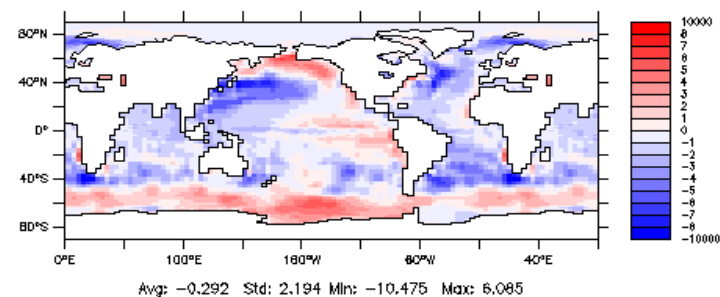
CPL84_SE_0011_0020_grid_T.nc: votemper[k=1,l=1]



ORCA4.0_Levitus.nc: votemper[k=1,l=1]



Difference: votemper[k=1,l=1] - votemper[k=1,l=1]



DODS server

How to use a DODS file ?

- Use a DODS client with an URL with a cgi-bin part
http://dods.ipsl.jussieu.fr/cgi-bin/nph-dods/prism/gridCF/IPSL.ORCA4_gridCF.nc
- For downloading the all file use the same URL but without the cgi-bin part

LAS - Live Access Server

Live Access Server provides a framework

- To produce plots on the fly
- By default use Ferret but other graphic engines such as CDAT can be installed

<http://esg.llnl.gov/las/>

