BUILDING DATA VISUALIZATION CAPACITY

Summer Webinar Series
July 21 - 24, 2020

Website: https://tinyurl.com/yctfhc6l
Administrivia & Logistics

• Take a moment to provide feedback for yesterday’s session: You’ve got data now what?

• Feedback link located at the bottom of Day 2 Web page, can also be accessed from here [https://tinyurl.com/y2yjncwn](https://tinyurl.com/y2yjncwn)
About Me

• Vetria Byrd, PhD
• Assistant Professor
• Purdue University
  • Department of Computer Graphics Technology
  • Polytechnic Institute
• Byrd Data Visualization Lab, Director
• Founder and Organizer of BPViz: Broadening Participation in Visualization Workshop

Webinar Organizers

Vetria Byrd
Assistant Professor
Byrd Data Visualization Lab, Director
Purdue University

Ritu Arora
Assistant Vice President of Research Computing
University of Texas at San Antonio

Brent League
Director of Research Computing Support
University of Texas at San Antonio
Webinar Goals and Objectives

Goal

Introduce participants to data visualization

Objectives

1. Examine what data looks like, define data visualization.
2. Illustrate how data visualization can improve understanding of the story within the data.
3. Introduce the data visualization process.
4. Explore different data visualization paths.
1 minute Recap

Day 2: You’ve got data now what?
Understanding Data

1. What real life behavior does the data reflect?
2. What are the strengths of the data source(s)?
3. What are the weaknesses of the data source(s)?
4. What information is emphasized?
5. At what level of granularity is the data provided?
6. What is the scope of the data?
Recommended Readings
Quick and Easy Read

7 things you should know about...
Data Visualization

7 things you should know about...
Data Visualization II

Available on the webinar website’s Resources page

EDUCAUSE Learning Initiative
advancing learning through IT innovation
www.educause.edu/eli
We are here

<table>
<thead>
<tr>
<th>Schedule</th>
<th>Topic</th>
<th>Skill Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Day 0</td>
<td>Download and Install</td>
<td>Beginner</td>
</tr>
<tr>
<td>Day 1</td>
<td>Introduction to data visualization</td>
<td>Beginner</td>
</tr>
<tr>
<td>Day 2</td>
<td>You’ve got data now what?</td>
<td>Beginner/Intermediate</td>
</tr>
<tr>
<td>Day 3</td>
<td>Introduction to scientific Visualization</td>
<td>Intermediate/Advanced</td>
</tr>
<tr>
<td>Day 4</td>
<td>Visualization hackathon</td>
<td>Advanced</td>
</tr>
</tbody>
</table>
INTRODUCTION TO PARAVIEW

Day: Thursday, July 23, 2020
Time: 1PM - 3 PM (CST)/2PM - 4PM (EST)
Skill level: Intermediate/Advanced

This seminar provides an introduction to scientific visualization using ParaView. An overview of the visualization process is presented by exploring the scientific visualization pipeline followed by hands on experience with ParaView.
Agenda

• Introduction
  • Scientific Visualization
  • Scientific Visualization Pipeline
  • Common Sci-vis Techniques

• Introduction to ParaView

• Hands-on Exercises
An iterative process

obtain the data

provide structure

remove all but the data of interest

apply methods from statistics or data mining to discern patterns or place the data in mathematical context

choose a basic visual model, such as a bar graph, list or tree

improve the basic representation to make it clearer and more visually engaging

Add methods for manipulating the data or controlling what features are visible

Adopted from Visualizing Data: Exploring and Explaining Data with the Processing Environment by Ben Fry, O’Reilly (p 15)
Primarily concerned with the visualization of three-dimensional phenomena (architectural, meteorological, medical, biological, etc.),

Where the emphasis is on realistic renderings of volumes, surfaces, illumination sources, and so forth, perhaps with a dynamic (time) component.
Scientific Visualization Pipeline

What’s Missing?

http://www.bu.edu/tech/research/training/tutorials/introduction-to-scientific-visualization-tutorial/the-scientific-visualization-pipeline/
Scientific Visualization Pipeline

http://www.bu.edu/tech/research/training/tutorials/introduction-to-scientific-visualization-tutorial/the-scientific-visualization-pipeline/
Scientific Visualization Pipeline: Step 1 . . .

Produce Data

- Simulated Data
- Images
- Numerical
- Some measured value
- Observed Phenomena

Adopted from http://www.bu.edu/tech/research/training/tutorials/introduction-to-scientific-visualization-tutorial/the-scientific-visualization-pipeline/
Scientific Visualization Pipeline: Step 2 . . .

Analyze, Filter, Reformat

- Cleaning up the data
  - Removing noise
  - Replacing missing values
  - Clamping values to be within a specific range of interest

- Performing operations to yield more useful data

Adopted from
http://www.bu.edu/tech/research/training/tutorials/introduction-to-scientific-visualization-tutorial/the-scientific-visualization-pipeline/
Scientific Visualization Pipeline: Step 3

Apply SciVis Techniques

- Converts raw information into something more understandable
- Visually extracting meaning from a scientific data set using various techniques

Adopted from http://www.bu.edu/tech/research/training/tutorials/introduction-to-scientific-visualization-tutorial/the-scientific-visualization-pipeline/
Scientific Visualization Pipeline

Step 4 . . .

Map to Geometry

- Scalars, vectors, tensors
- 1D, 2D, 3D
- Mesh

Adopted from
http://www.bu.edu/tech/research/training/tutorials/introduction-to-scientific-visualization-tutorial/the-scientific-visualization-pipeline/
Scientific Visualization Pipeline: Step 5 . . .

Render, Post Process

Adopted from http://www.bu.edu/tech/research/training/tutorials/introduction-to-scientific-visualization-tutorial/the-scientific-visualization-pipeline/
Scientific Visualization Pipeline: Step 6 . . .

View Results

Output from ParaView

Adopted from
http://www.bu.edu/tech/research/training/tutorials/introduction-to-scientific-visualization-tutorial/the-scientific-visualization-pipeline/
Large data produced by large simulations produce large visualization results and require large visualization resources.

- Terabytes of data
- AT LEAST Terabytes of Vis
- Gigapixel Images

Resampling, Application, ... Resolution to Capture Feature Detail
Visualization Toolkit (VTK)

- Open source, multiplatform
- Supports distributed computation models
- Extensible modular architecture
- Available for 3D computer graphics, image processing and visualization

- Collection of C++ libraries
- Leveraged by many applications
- Divided into logical areas
  - Filtering
  - Information Visualization
  - Volume Rendering
- Cross platform, using OpenGL
- Wrapped in Python, Tool Command Language (Tcl) and Java
ParaView is an end-user application with support for

- Parallel Data Archiving
- Parallel Reading
- Parallel Processing
- Parallel Rendering
- Single node, Client-Server, MPI Cluster Rendering
## More ParaView Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Paraview</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standalone GUI</td>
<td>Yes</td>
</tr>
<tr>
<td>Scriptable (Python)</td>
<td>Yes</td>
</tr>
<tr>
<td>API</td>
<td>C++, Python</td>
</tr>
<tr>
<td>Ability to Embed Graphics window</td>
<td>Yes (with C++ API and Qt GUI)</td>
</tr>
<tr>
<td>custom GUI</td>
<td></td>
</tr>
<tr>
<td>Plugin Architecture</td>
<td>Yes</td>
</tr>
</tbody>
</table>

http://www.mantidproject.org/Visit_vs_Paraview
Introduction to Scientific Visualization Using ParaView
• Multi-platform parallel data analysis and visualization application
• Mature, feature-rich interface
• Good for general purpose, rapid visualization

Mac
Windows
Linux
Current Funding

- ARL
- ERDC
- US Army (SBIR)
- US Air Force (STTR)
- ONR
- Support Contracts
  - Electricity de France
  - Microsoft
- Other contributors
  - Swiss National Supercomputing Centre
  - DOE SLAC
  - Ohio State
  - Mississippi State
  - RPI
Data Ranges

• Used for all ranges of data size.

• Landmarks of usage:
  – 6 billion structured cells (2005).
  – Billions of AMR cells (2008).
  – Scaling test over 1 Trillion cells (2010).
  – 6.33 billion unstructured cells in Catalyst (2016).
ParaView Application Architecture

<table>
<thead>
<tr>
<th>ParaView Client</th>
<th>pvpython</th>
<th>ParaWeb</th>
<th>Catalyst</th>
<th>Custom App</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>UI (Qt Widgets, Python Wrappings)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ParaView Server</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VTK</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OpenGL</td>
<td>MPI</td>
<td>IceT</td>
<td>Etc.</td>
<td></td>
</tr>
</tbody>
</table>
ParaView Development

- Started in 2000 as collaborative effort between Los Alamos National Laboratories and Kitware Inc. Sandia has been a major contributor since 2005.
  - ParaView 0.6 released October 2002.
  - GUI rewritten to be more user friendly and powerful.
- ParaView 4.0 released in June 2013.
  - Properties panel redesign for smoother interaction.
- ParaView 5.0 released in January 2016.
  - Updated to OpenGL 3.2 features. Huge performance improvements.
- ParaView 5.4 latest version
**Grid** – regular structure, all voxels (cells) are the same size and shape

**Uniform Rectilinear (Image Data)**
A uniform rectilinear grid is a one-, two- or three-dimensional array of data. The points are orthonormal to each other and are spaced regularly along each direction.
Curvilinear (Structured Grid)
Curvilinear grids have the same topology as rectilinear grids. However, each point in a curvilinear grid can be placed at an arbitrary coordinate (provided that it does not result in cells that overlap or self intersect). Curvilinear grids provide the more compact memory footprint and implicit topology of the rectilinear grids, but also allow for much more variation in the shape of the mesh.

Curvilinear – regularly gridded mesh shaping function applied

Adopted from The ParaView Tutorial, The Basics of Visualization, version 3.98
**Unstructured grid** – irregular mesh typically composed of tetrahedra, prisms, pyramids, or hexahedra

Unstructured Grid
Unstructured data sets are composed of points, lines, 2D polygons, 3D tetrahedra, and nonlinear cells. They are similar to polygonal data except that they can also represent 3D tetrahedra and nonlinear cells, which cannot be directly rendered.

Adopted from The ParaView Tutorial, The Basics of Visualization, version 3.98
- Point data
- Polygonal data
- Images
- Multi-block
- Adaptive Mesh Refinement (AMR)
- Time series support
SUPPORTED VISUALIZATION ALGORITHMS

- Isosurfaces
- Cutting planes
- Streamlines
- Glyphs
- Volume rendering
- Clipping
- Height maps
- & more
- Supports derived variables
- Scriptable via Python
- Saves animations
- Can run in parallel / distributed mode for large data visualization
- ParaView Data (.pvd)
- VTK (.vtp, .vtu, .vti, .vts, .vtr)
- VTK Legacy (.vtk)
- VTK Multi Block (.vtm, .vtmb, .vtmg, .vthd, .vthb)
- Partitioned VTK (.pvtu, .pvti, .pvts, .pvtr)
- ADAPT (.nc, .cdf, .elev, .ncd)
- ANALYZE (.img, .hdr)
- ANSYS (.inp)
- AVS UCD (.inp)
- BOV (.bov)
- BYU (.g)
- CCSM MTSD (.nc, .cdf, .elev, .ncd)
- CCSM STSD (.nc, .cdf, .elev, .ncd)
- CEAucd (.ucd, .inp)
- CMAT (.cmat)
- CTRL (.ctrl)
- Chombo (.hdf5, .h5)
- Claw (.claw)
- Comma Separated Values (.csv)
- Cosmology Files (.cosmo, .gadget2)
- Curve2D (.curve, .ultra, .ult, .u)
- DDCMD (.ddcmd)
- Digital Elevation Map (.dem)
- Dyna3D (.dyn)
- EnSight (.case, .sos)
- Enzo boundary and hierarchy
- ExodusII (.g, .e, .exe, .ex2, .ex2v.., etc)
- ExtrudedVol (.exvol)
- FVCOM (MTMD, MTSD, Particle, STSD)
- Facet Polygonal Data
- ProSTAR (.cel, .vrt)
- Protein Data Bank (.pdb, .ent, .pdb)
- Raw Image Files
- Raw NRRD image files (.nrrd)
- SAMRAI (.samrai)
- SAR (.SAR, .sar)
- SAS (.sasgeom, .sas, .sasdata)
- SESAME Tables
- SLAC netCDF mesh and mode data
- SLAC netCDF particle data
- Silo (.silo, .pdb)
- Spherial (.spherial, .sv)
- SpyPlot CTH
- Spy Plot (.case)
- Stereo Lithography (.stl)
- TFT Files
- TIFF Image Files
- TSurf Files

Many more . . .
• All processing operations (filters) produce data sets
• Can further process the result of every operation to build complex visualizations
  • Extract a cutting plane,
  • Apply glyphs (i.e. vector arrows) to the result
    – Gives a plane of glyphs through your 3D volume
Three Basic Steps:

- First your data must be **read** into ParaView
- Next, you may apply any number of **filter**s that process the data to generate, extract, or derive features from the data
- Finally, a viewable image is **rendered** from the data
ParaView 5.8
Let’s get started . . .
Training Data

Webinar training data available on the Webinar Website

GETTING STARTED WITH PARAVIEW

Bluewaters Webinar: Introduction to ParaView
Estimated time: 1 hr

Summary: This seminar provides an introduction to scientific visualization using ParaView. An overview of the visualization process is presented by exploring the scientific visualization pipeline followed by hands on experience with ParaView. ParaView is an open-source, multi-platform, parallel data analysis and visualization application built upon the Visualization Toolkit (VTK) Library. Participants will work with a small dataset (provided) to become familiar with ParaView functions and capabilities.

SUMMER WEBINAR FILES

ParaView 5.8 [Download]

Data files:
headsgvti
Rectgrd2vtk

Additional training data files can be downloaded from the ParaView Website

https://www.paraview.org/Wiki/The_ParaView_Tutorial
Basic Usage
ParaView User Interface

Menu Bar
Tool Bar
Pipeline Browser
Object Inspector
Properties Panel
Advanced Toggle
3D Viewer
Getting Back GUI Components
Creating a Cylinder Source

1. Go to the Sources menu.
2. Select Geometric Shapes.
3. Select Cylinder
Creating a Cylinder Source

• A cylinder1 object should appear in the pipeline browser

Click the Apply button to accept the default parameters.
Creating a Cylinder Source

• Explore the resolution feature by adjusting the Resolution slider or type in a value.

Click the **Apply** button to accept changes.
Simple Camera Manipulation

- Drag left, middle, right buttons for rotate, pan, zoom.
  - Also use Shift, Ctrl, Alt modifiers.
  - Also try holding down x, y, or z.
Pipeline Object Controls

- Apply
- Reset
- Delete
Using Auto Apply

1. Click **Auto Apply**.

2. Change the Resolution parameter again.

3. Note that the visualization automatically updates without having to hit **Apply**.
Using Auto Apply

1. Click 📦 to apply changes to parameters automatically.

2. Change the Resolution parameter again.

3. Note that the visualization automatically updates without having to hit Apply.
Display Properties

Properties (Cylinder1)
- Resolution: 6
- Height: 1
- Radius: 0.5
- Center: 0, 0, 0
- Capping: checked

Display (GeometryRepresentation)
- Representation: Surface
Change Render Properties

1. Scroll down to the Display (Geometry Representation) section.

2. Click the Edit button. (This button is replicated in the toolbar.)

3. Select a new color for the cylinder and click OK.
Render View Options

Check to turn on the Axis Grid

Scroll down
Changing the Color Palette

1. Make sure the orientation axes are visible in the lower left corner.

2. Click the color palette button and change the colors.

3. Try several color palettes or edit the current color palette.
Color Palettes

→ Edit Current Palette...
Color Palettes

→ Edit Current Palette...

---

**Settings**

**General**

- Color used when solid coloring surfaces and faces.
- Surface
- Color used for rendering elements like wireframes, points.
- Foreground
- Color used for the edges when using 'Surface With Edges' representation.
- Edges
- Color used for text and other annotations.
- Text

**Selection**

- Color used for showing selected cells/points.
- Selection
- Color used for showing interactive selection.
- Interactive Selection
- Color used for showing interactive widget.
- Interactive Widget

**Background**

- Use Gradient Background: Enable gradient background.
- Color used as background for the view.
- Background
- Color used as the 2nd color for gradient backgrounds.
- Background 2

**Load Palette**: You can explicitly set the colors in the application's color palette above or you can load one of the predefined color palettes to initialize the active palette.

Select palette to load...
Advanced Properties

Search Properties

Toggle Advanced Properties
Searching Properties

1. Type “specular” in the properties search box
2. Changing Specular value to 1 (makes the cylinder shiny)

Try typing in the following in to the search box (one at time) to explore other properties:
- Axes Grid
- Opacity
- Lights
Undo Redo

Undo  Redo

Camera Undo  Camera Redo
Creating a Cylinder Source

1. Go to the **Sources** menu and select **Geometric Shapes**.
2. Select **Cylinder**
3. Click the **Apply** button to accept the default parameters.
4. Increase the **Resolution** parameter.
5. Click the **Apply** button (if needed) again.
6. Now, delete the Cylinder1 object (make sure it is selected in the pipeline browser)
Additional training data files can be downloaded from the ParaView Website

https://www.paraview.org/Wiki/The_ParaView_Tutorial

GETTING STARTED WITH PARAVIEW

Bluewaters Webinar: Introduction to ParaView
Estimated time: 1 hr

Summary: This seminar provides an introduction to scientific visualization using ParaView. An overview of the visualization process is presented by exploring the scientific visualization pipeline followed by hands on experience with ParaView. ParaView is an open-source, multi-platform, parallel data analysis and visualization application built upon the Visualization Toolkit (VTK) Library. Participants will work with a small dataset (provided) to become familiar with ParaView functions and capabilities.

SUMMER WEBINAR FILES

ParaView 5.8 | Download |

Data files:
headsgvti
Rectgrd2.vtk
Tutorial Datasets

- RectGrid2.vtk
- headsq.vti

Sample data file
- header.txt
- xCoordinates.txt
- yCoordinates.txt
- zCoordinates.txt
- lookUpTable.txt
What are we going to do?

- Load Data File
- Extract Isosurfaces from the data
- Create contours, clip contours, slice contours
- Volume Rendering
- Saving your Data
- Getting your data into Paraview
- Introduction to ParaView and Python Scripting
- Additional Resources
Good to Know

Reset ParaView

Undo Redo

Undo  Redo
Open Data File: RectGrid2.vtk

- Locate `RecGtrid2.vtk`
- Click File → Open
- Select `RectGrid2.vtk`
- Click OK
# vtk DataFile Version 2.0
Sample rectilinear grid
ASCII_DATASET RECTILINEAR_GRID
DIMENSIONS 47 33 11
X_COORDINATES 47 float
-1.22396 -1.17188 -1.11979 -1.06771 -1.01562 -0.963542
-0.911458 -0.859375 -0.807292 -0.755208 -0.703125 -0.651042
-0.598958 -0.546875 -0.494792 -0.442708 -0.390625 -0.338542
-0.286458 -0.234375 -0.182292 -0.130209 -0.078125 -0.026042
0.0260415 0.078125 0.130208 0.182291 0.234375 0.286458
0.338542 0.390625 0.442708 0.494792 0.546875 0.598958
0.651042 0.703125 0.755208 0.807292 0.859375 0.911458
0.963542 1.01562 1.06771 1.11979 1.17188
Y_COORDINATES 33 float
-1.25 -1.17188 -1.09375 -1.01562 -0.9375 -0.859375
-0.78125 -0.703125 -0.625 -0.546875 -0.46875 -0.390625
-0.3125 -0.234375 -0.15625 -0.078125 0 0.078125
0.15625 0.234375 0.3125 0.390625 0.46875 0.546875
0.625 0.703125 0.78125 0.859375 0.9375 1.01562
1.09375 1.17188 1.25
Z_COORDINATES 11 float
0 0.1 0.2 0.3 0.4 0.5
0.6 0.7 0.75 0.8 0.9
POINT_DATA 17061
SCALARS scalars float
LOOKUP_TABLE default
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0

Create Isosurfaces

Contour – extracts the points, curves, or surfaces where a scalar field is equal to a user-defined value.
Click on RectGrid2.vtk

Select
- Filters
- Common
- Contour
Should see

- New object in Pipeline Browser
- One (1) value in the value range box
Select the value in the **Value Range** window

Click the minus sign in the value range box to Delete the value
Click the Add a range of values button

Click **Generate**
Should see

- A range of values in the Value Range box
- Values range from the min and max values entered in the previous dialog box
Should see

- Rendering in the 3D viewer
- Use your mouse to explore (rotate) the output
Step 1

Step 2
CREATE ISOSURFACES

SUMMARY OF STEPS

- Click Filters ➔ Common ➔ Contour
- Remove all values in Value Range Box
- Click Add a range of values
- Set values (Or accept default)
- Click Apply
- Set Color By property: vectors
Clip Isosurfaces
Clip – Intersects the geometry with a half space. The effect is to remove all the geometry on one side of a user-defined plane.
- Make sure Contour 1 is selected (highlighted in blue)
- Set View Direction to +Y
Create a new Clip filter:
Select:
- Filters
- Common
- Clip
If you do not see the clipping plane click the properties button

Should see
- Clip object in Pipeline Browser
- Clipping Plane
• Using the mouse, hover over the arrow head (will change color – red)
• Press the mouse button, keep it pressed and rotate the clipping plane; arrow points out of the screen towards you
When done: Click **Apply**
- Click the **Inside Out** box to switch the clipping plane
- Click **Apply**
CLIP ISOSURFACES

SUMMARY OF STEPS

- Click +Y view button
- Click Filters → Common → Clip
- Position the clipping plane (move arrow point to desired position)
- Click Apply
- Check Inside Out check box to switch the clipping plane view
Slice Isosurfaces

Slice – Intersects the geometry with a plane. The effect is similar to clipping except that all that remains is the geometry where the plane is located.
Make visible
- RectGrid2.vtk
- Clip1

To make an object visible click the eye icon to the left of the object

Contour 1 should **NOT** be visible for the next task
Select Clip 1

- Filters
- Common
- Slice
Should see
- **Slice1** object in Pipeline Browser
- Slicing Plane with arrow in 3D window
Position the slicing plane (move arrow point to desired position)

Click **Apply** if the viewer window does not automatically update
You can move the slicing plane along the axis to see a different slice

Click **Apply** to see changes
SLICE ISOSURFACE

SUMMARY OF STEPS

- Select Clip1 Object
- Select Filter
- Select Common
- Select Slice
- Position Slicing Plane
- Click Apply

Slice - Intersects the geometry with a plane. The effect is similar to clipping except that all that remains is the geometry where the plane is located.
Recall Three Basic Steps:

- First your data must be read into ParaView
- Next, you may apply any number of filters that process the data to generate, extract, or derive features from the data
- Finally, a viewable image is rendered from the data

Opened simple data file RectGrid2.vtk

Applied filters: Contour, Slice, Clip
Introduction to ParaView
Part II: Hands-On-Exercise
Recall Three Basic Steps:

- First your data must be read into ParaView
- Next, you may apply any number of filters that process the data to generate, extract, or derive features from the data
- Finally, a viewable image is rendered from the data

Opened simple data file RectGrid2.vtk

Applied filters: Contour, Slice, Clip
Let's try a more complex data set

headsq.vti
In preparation for the next section

- Edit
- Reset Session

Your Interface should look like this
Locate File: headsq.vti
New Object in Pipeline Browser

Click **Apply** to update the viewer window if necessary.
Should see a bounding box in the 3D viewer window
Create an Isosurface

Select:
→ Filters
→ Common
→ Contour
A new object appeared in the pipeline browser (Contour 1)

Contour – Extracts the points, curves, or surfaces where a scalar field is equal to a user-defined value.

The surface is often also called an isosurface.
Value Range for the data set is now visible
➢ Value Range for the data set is [0, 4095]

➢ Only one value is showing: 2047.5

➢ Click **Apply** to see what points, curves, or surfaces in the dataset have a value of 2047.5
If you do not see anything in the 3D window click the eye icon next to Contour1 in the Pipeline Browser.

This allows you to toggle between views in the 3D Viewer.
Visually Explore Dataset
You should be here

Pipeline Browser
→ Two objects
→ Value Range [0, 4095]

[Image of a 3D skull in ParaView software]
Isosurfaces

- Select 2047.5 showing in Value Range
  - Selecting the value 2047.5 highlights the value
- Delete that value (click the minus button to remove all values)
Click the button below the minus button to: **Add a Range Of Values**

- Should see the Add Range Window
  - Use this window to set the range of values
  - For this tutorial
    - Min: 0
    - Max: 4095
  - Feel free to play around with the range (between 0 and 4095)
- Click **OK**
- Notice the Value Range: [0, 4095]

- There are 10 values (steps) showing values between 0 and 4095
This may take a few seconds to render...
Explore the rendering...
**Contour** – Extracts the points, curves, or surfaces where a scalar field is equal to a user-defined value.

The surface is often also called an *isosurface*.
<table>
<thead>
<tr>
<th>From</th>
<th>To</th>
<th>Step</th>
<th>What do you get?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1500</td>
<td>2500</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>2500</td>
<td>3500</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>2999</td>
<td>3500</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>3500</td>
<td>4000</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>2000</td>
<td>2999</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>3000</td>
<td>3999</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>
1. Click red X – Remove all entries
2. Click Add a range of values button
3. Click OK
4. Click Apply
Clip Isosurface

- CLIP - Intersects the geometry with a half space.
- The effect is to remove all the geometry on one side of a user-defined plane.
Select:
→ Contour 1 (in pipeline Browser)
→ Filters
→ Common
→ Clip
A new object appeared in the pipeline browser (Clip 1)
- Clipping plane (see red vertical line and horizontal arrow)
Clip Isosurface

- Select the arrow point (arrow turns red)
- Rotate (drag) the arrow point until the arrow is pointing out of the screen toward you
Depending on where you placed the clipping plane the results may be easily seen: see clipped ears; and area round neck

- Make sure the eye icon is not greyed out on the Clip1 object in the Pipeline Browser
Clip Isosurface

Rotating the view reveals the clipped isosurface
If this property is set to 0, then clip filter will return that portion of the dataset that lies within the clip function.

If set to 1, the portions of the dataset that lie outside the clip function will be returned instead.
Slice Isosurface

- Click the eye icon next to Clip1 in the pipeline browser *(hide the clip plot)*
- Select **Contour 1**
Slice Isosurface

Select

- Filters
- Common
- Slice
Rotated View

Drag arrow point around to point out of screen toward you.
SLICE – Intersects the geometry with a plane. The effect is similar to clipping except that all that remains is the geometry where the plane is located.
Q: How do we combine (show) the Clip and Slice views at the same time?
Many Options to Save Your Work

- Open...
- Recent Files
- Load State...
- Save State...
- Save Data...
- Save Screenshot...
- Export Scene...
- Save Animation...
- Save Geometry...
- Load Window Layout...
- Save Window Layout...
- Connect...
- Disconnect
- Exit
- Export Cinema...
Save Your Work

Save State

save the state of the visualization pipeline itself, including all the pipeline modules, views, their layout, and their properties.

This is referred to as the application state or, just, state.

In paraview, you can save the state using the File Save State... menu option.

Conversely, to load a saved state file, you can use File Load State...
You can save the dataset produced by any pipeline module in ParaView, including sources, readers, and filters.

To save the dataset in ParaView, begin by selecting the pipeline module in the Pipeline browser to make it the active source.

For modules with multiple output ports, select the output port producing the dataset of interest.

To save the dataset, use the File Save Data menu or the button in the Main Controls toolbar. You can also use the keyboard shortcut Ctrl + S (or + S).

The Save File dialog will allow you to select the filename and the file format. The available list of file formats depends on the type of the dataset you are trying to save.
Save Your Work

Save Screenshot

To save the render image from a view in paraview, use the File Save Screenshot menu option. This will pop up the Save Screenshot Options dialog (Figure 8.3).

This dialog allows you to select various image parameters, such as image resolution and image quality (which depends on the file format chosen).
Save Your Work

Save Screenshot continued . . .

The dialog also provides an option to change the color palette to use to save the image using Override Color Palette.

By default, paraview will save rendered results from the active view. Optionally, you can save an image comprising of all the views layed out exactly as on the screen by unchecking the Save only selected view button.
In preparation for the next section

- Edit
- Reset Session

You should be here
Introduction to ParaView
Part III: Getting Data Into VTK File Format
Getting Your Data Into VTK File Format

Sample File
Supported Data Formats

- VTK (http://www.vtk.org/VTK/img/file-formats.pdf)
- EnSight
- Plot3D
- Various polygonal formats
- Users can write data readers to extend support to other formats
- Conversion to the VTK format is straightforward
VTK Simple Legacy Format

VTK simple legacy format (http://www.vtk.org/VTK/img/file-formats.pdf)

- ASCII or binary
- Supports all VTK grid types
- Easiest for data conversion

---

Part 1: Header

Part 2: Title (256 characters maximum, terminated with newline \n character)

Part 3: Data type, either ASCII or BINARY

Part 4: Geometry/topology. Type is one of:
- STRUCTURED_POINTS
- STRUCTURED_GRID
- UNSTRUCTURED_GRID
- POLYDATA
- RECTILINEAR_GRID
- FIELD

Part 5: Dataset attributes. The number of data items n of each type must match the number of points or cells in the dataset. (If type is FIELD, point and cell data should be omitted.)
The data

- Simulated temperature values
- Sample size: 100 x 100
- Rectilinear Grid

<table>
<thead>
<tr>
<th></th>
<th>100</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>20.18</td>
<td>20.36</td>
</tr>
<tr>
<td>20.36</td>
<td>20.36</td>
<td>21.07</td>
</tr>
<tr>
<td>20.53</td>
<td>21.07</td>
<td>21.61</td>
</tr>
<tr>
<td>20.70</td>
<td>21.41</td>
<td>22.13</td>
</tr>
<tr>
<td>20.86</td>
<td>21.74</td>
<td>22.62</td>
</tr>
<tr>
<td>21.02</td>
<td>22.05</td>
<td>23.10</td>
</tr>
<tr>
<td>21.17</td>
<td>22.35</td>
<td>23.54</td>
</tr>
<tr>
<td>21.31</td>
<td>22.63</td>
<td>23.96</td>
</tr>
<tr>
<td>21.44</td>
<td>22.88</td>
<td>24.35</td>
</tr>
<tr>
<td>21.56</td>
<td>23.12</td>
<td>24.71</td>
</tr>
<tr>
<td>21.66</td>
<td>23.34</td>
<td>25.04</td>
</tr>
<tr>
<td>21.76</td>
<td>23.53</td>
<td>25.33</td>
</tr>
<tr>
<td>21.85</td>
<td>23.70</td>
<td>25.58</td>
</tr>
<tr>
<td>21.92</td>
<td>23.85</td>
<td>25.81</td>
</tr>
<tr>
<td>21.99</td>
<td>23.98</td>
<td>26.00</td>
</tr>
<tr>
<td>22.04</td>
<td>24.09</td>
<td>26.16</td>
</tr>
<tr>
<td>22.09</td>
<td>24.18</td>
<td>26.29</td>
</tr>
<tr>
<td>22.12</td>
<td>24.25</td>
<td>26.40</td>
</tr>
<tr>
<td>22.15</td>
<td>24.30</td>
<td>26.48</td>
</tr>
<tr>
<td>22.17</td>
<td>24.34</td>
<td>26.53</td>
</tr>
<tr>
<td>22.18</td>
<td>24.36</td>
<td>26.56</td>
</tr>
<tr>
<td>22.18</td>
<td>24.37</td>
<td>26.57</td>
</tr>
<tr>
<td>22.18</td>
<td>24.36</td>
<td>26.55</td>
</tr>
<tr>
<td>22.17</td>
<td>24.34</td>
<td>26.53</td>
</tr>
<tr>
<td>22.15</td>
<td>24.31</td>
<td>26.48</td>
</tr>
<tr>
<td>22.13</td>
<td>24.27</td>
<td>26.42</td>
</tr>
<tr>
<td>22.11</td>
<td>24.23</td>
<td>26.35</td>
</tr>
<tr>
<td>22.08</td>
<td>24.17</td>
<td>26.27</td>
</tr>
</tbody>
</table>
# vtk DataFile Version 2.0
Really cool data
ASCII | BINARY
DATASET type
...
POINT_DATA n
...
CELL_DATA n
...

Part 1: Header

Part 2: Title (256 characters maximum, terminated with newline \n character)

Part 3: Data type, either ASCII or BINARY

Part 5: Dataset attributes. The number of data items \( n \) of each type must match the number of points or cells in the dataset. (If type is FIELD, point and cell data should be omitted.)
# vtk DataFile Version 2.0
Rectilinear grid of temperature values
ASCII
DATASET RECTILINEAR_GRID

* DIMENSIONS 100 100 1

X_COORDINATES 100 float
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99

Y_COORDINATES 100 float
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99

* Z_COORDINATES 100 float
0

* Although this is a 2D grid, the z-coordinate must be included and represented in the DIMENSIONS.
# vtk DataFile Version 2.0
Rectilinear grid of temperature values
ASCII
DATASET RECTILINEAR_GRID
DIMENSIONS 100 100 1
X_COORDINATES 100 float
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52
53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76
77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99
Y_COORDINATES 100 float
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52
53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76
77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99
Z_COORDINATES 1 float
0
* POINT_DATA 10000
SCALARS temperature float
LOOKUP_TABLE default

* x-dimension * y-dimension * z-dimension
# vtk DataFile Version 2.0
Rectilinear grid of temperature values
ASCII
DATASET RECTILINEAR_GRID
DIMENSIONS 100 100 1
X_COORDINATES 100 float
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99
Y_COORDINATES 100 float
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99
Z_COORDINATES 1 float
0
POINT_DATA 10000
SCALARS temperature float
LOOKUP_TABLE default
20.18 20.36 20.54 20.73 20.93 21.13 21.35 21.58 21.82 22.09 22.38 22.70 23.06 23.46 23.92 24.44 25.05 25.77
26.63 27.68 28.99 30.68 32.90 35.99 40.50 47.61 60.00 84.65 142.03 300.00 300.00 300.00 300.00 300.00 300.00 300.00 300.00 300.00 300.00
300.00 300.00 300.00 289.04 288.50 287.82
Data Files:

Header.txt
xCordinates.txt
yCoordinates.txt
zCoordinates.txt
lookUpTable.txt

lookUpTable.txt also contains the data points

Task:

Combine these files into one file and save the file with the .vtk file extension SampleData.vtk

Remember what you name the file.

ByrdVisLab Polytechnic
• Open data file (the file that you just created and saved)
• Click **Apply**
EXERCISE: VISUALIZE SAMPLE DATA

Add **Contour Plot**

Set the **Range of Values**
- From 20.01
- To: 300
- Step 10
What should the pipeline browser look like?
You should be here
Filters in ParaView

- Filters
- Alphabetical
In preparation for the next section

- Edit- Reset Session
- Select an object in the Pipeline Browser
- Click the Delete button (or right click, then Delete)
- To select multiple objects press and hold the CTRL key while selecting objects
More Training Data

Download your preferred archive format from ParaView website

http://www.paraview.org/Wiki/The_ParaView_Tutorial

- The ParaView Tutorial Data (in tar/gzip)
- The ParaView Tutorial Data (in zip)
- ParaView Scripts (in tar/gzip)
Wrapping Up
ADDITIONAL RESOURCES

http://www.paraview.org

ParaView User’s Guide: Downloaded with ParaView

ParaView Sample Data
http://www.paraview.org/Wiki/The_ParaView_Tutorial

ParaView/Python Scripting – KitwarePublic
http://www.paraview.org/Wiki/ParaView/Python_Scripting

ParaView Server Manager
http://www.paraview.org/ParaView/Doc/Nightly/www/py-
doc/paraview.servermanager.html
ParaView Guide – Comprehensive user guide for ParaView
Help – online help for file readers and filters
Online Tutorials – in depth tutorials for ParaView
Online Blogs – informative blog posts on new features in ParaView
What’s next?

Day 4
Day 4: The pièce de résistance!