CS 6635 Spring Semester 2018

Date: Thursday, February 1, 2018

Due Date: Thursday, February 15, 2018

Goal

The goal of this assignment is to explore different dataset with **Paraview**.

This assignment will help you have a concrete intuition and understanding of the techniques described throughout the class.

After completing this assignment, you will be able to import into ParaView supported dataset and to produce insightful interactive visualizations.

Prerequisites

This assignment requires you to install a recent version of ParaView (https://www.paraview.org/download/). Version 5.4 or higher should be sufficient. Binary installers exist on most platforms, but you do need administrator privileges to install it.

In addition you will need the following datasets: Assignment2-Data.zip (http://www.eng.utah.edu/~cs6635/CS6635-Assignment2-Data.zip)

Document

ParaView tutorials and sample data sets can be found HERE (https://www.paraview.org/Wiki/The_ParaView_Tutorial).

Part 1: Load the data

Q1. Visualization of Statistics for 1-D data [15 pts]

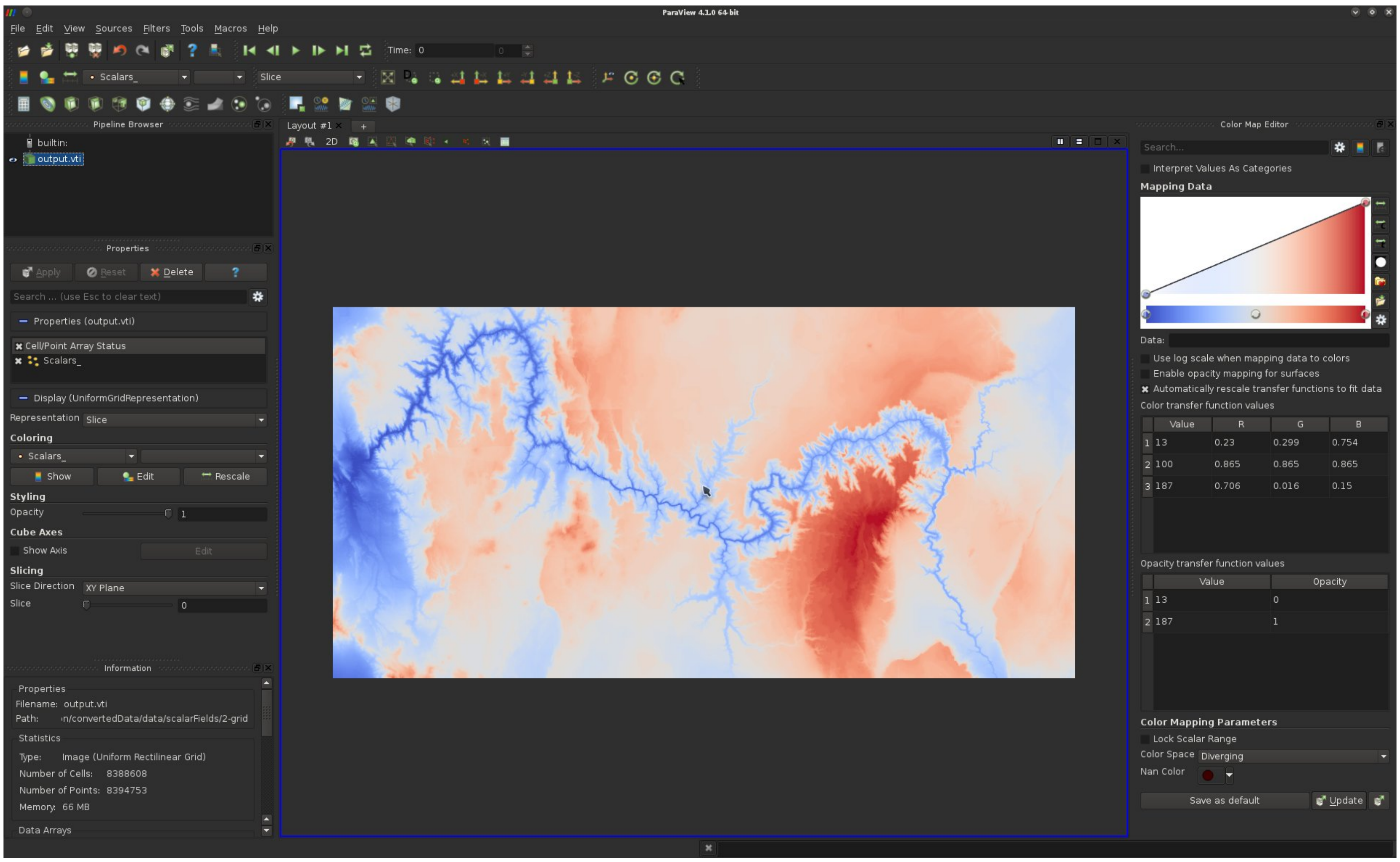
ParaView can visualize many types of datasets, from both very simple to very complicated. First, File->Open the dataset data01.txt in ParaView. In the Properties panel, make sure that you uncheck Have Headers before you hit Apply, since this text file is just a flat list of numbers. You'll quickly see…a SpreadSheetView! Given that ParaView does not know much about the data you're trying to visualize, the best it can do is show you the raw data. If you've loaded this data correctly, you should that the maximum row ID is 229 (as there are 230 rows, counting from 0). 1) Histograms: To get some basic visualization up, split the center window vertically/ horizontally. In the dialogue for a split window, click on "Histogram View". With this view highlighted (you should see a blue border around it), click on the (greyed-out) eye next to CS6635-Assignment2-P1Q1-Data.txt (http://www.eng.utah.edu/~cs6635/CS6635-Assignment2-P1Q1-Data.txt) in the Pipeline Browser to enable this dataset to be drawn in the histogram. One issue is that the histogram, by default, has too many bins. In our case, we have exactly 100 possibilities (numbers between 0 and 99), so in the Properties panel, set the bin count to 100.

In your report please answer:

1. Which number occurred the most frequently and how many times did it occur?
2. How many numbers were never used by the class? 2) Line Charts: Follow the same method as for histograms to render the line chart. Please add the left title as “Value” and the bottom axis title as the “Row ID” through the properties panel. Also, change the line thickness to 3 units.

Q2. Visualization of 2d Image [15 pts]

Next, we’ll be working with the data file 2d.vti in Assignment2-Data.zip (http://www.eng.utah.edu/~cs6635/CS6635-Assignment2-Data.zip). Files that end in .vt* are VTK file formats, the last letter of which indicates what type of file. .vti files are Images. Open up ParaView and load 2d.vti. This is a grayscale image that samples a 2D scalar field. By default, ParaView automatically maps the scalar values to color.



Let’s try working with a simple Filter. Go to Filters->Common->Threshold (or alt+space on MAC/ ctrl + space on Windows and search for Threshold) and add a Threshold filter. This filter selects only points that have values in a specified range. You’ll see the minimum and maximum of data range for initial loading of the filter.

This image has pixels that correspond to height values associated with the Grand Canyon (see a map (https://azraft.com/wp-content/uploads/2012/05/1-Azra-Map-B-web.jpg) to compare). Note that if you click and drag with the mouse you can reposition it. Since this data is two-dimensional, ParaView by default loads the render view in 2D mode (you’ll see this in the upper left of the view)

What we’d like to do is try to only select the pixels that correspond to the canyon itself (so that non-selected pixels would represent the river bed). To aid in deciding which values are above the canyon flow, it might be helpful to use a histogram. Split the view vertically, and in the view below again create a histogram. Select this view and click on the eye next to 2d.vti. Adjust the number of bins based on the minimum and maximum of the data so that you have exactly the right number of bins (you'll know you're correct where there are no gaps between bars in the histogram).

Based on the histogram, select the top render view, disable the view of the entire dataset and enable the view of the threshold. Set the maximum value to something reasonable based on the histogram (I chose a minimum threshold value which had a bin count of 200000). You should get a nice blue outline of the riverbed. Save your state file for this view as 2dImageVis.pvsm. In your report answer:

1. What threshold did you use for capturing the riverbed? Experiment with other thresholds and explain what features you may or may not have missed with this approach.
2. Using the Information panel, report the number of points in the thresholded image. Note that ParaView automatically creates cells from an input image, implicitly forming a structured quad mesh.

Q3. Exploring Data on Polygonal Meshes [15 pts]

Next up, load surf.vtp. VTP files encode polygonal mesh data. You should be seeing something that looks roughly like a rotor for an automobile disc brake. In many scenarios, it may be interesting to visually inspect the structure of the mesh on which is defined our data. In the Properties panel, adjust the right option to visualize the mesh as represented as a wireframe drawn on top of the surface cells.

The color coding of the cells turns out to be based on a measure of distance from the center of one of the five cylinders use for bolting the brake to a car. A cylinder can be identified through a wireframe/solid view along with colormap. You can use the Threshold filter to extract this cylinder through a careful setting of the minimum and maximum value.

Most disc brakes have ventilation slots that are used to transfer heat away during braking, this one is no different. Ventilation slots can be seen in the wireframe view. Nevertheless, they are quite hard to see and the Threshold filter does a poor job of extracting them – the problem is that the scalar value defined on vertices does not correlate to their geometry. Instead, we need to address this manually.

ParaView has a filter that can be used to clip arbitrary geometry. Remove the Threshold filter and instead add a Clip filter. Set the clipping plane to align with the Y normal and configure it to slice through the ventilation slots (these look like tiny pillars). Click apply. Save this state file as meshVis.pvsm

1. What were the minimum and maximum values that best captured the single cylinder associated with the bolt's cylinder?
2. How many ventilation slots are there?

Q4. Visualization of 3D Images [15 pts]

Finally, we'll try out visualizing a three-dimensional image in ParaView. Load 3d.vti. You should see only an outline of the bounding box at this point.

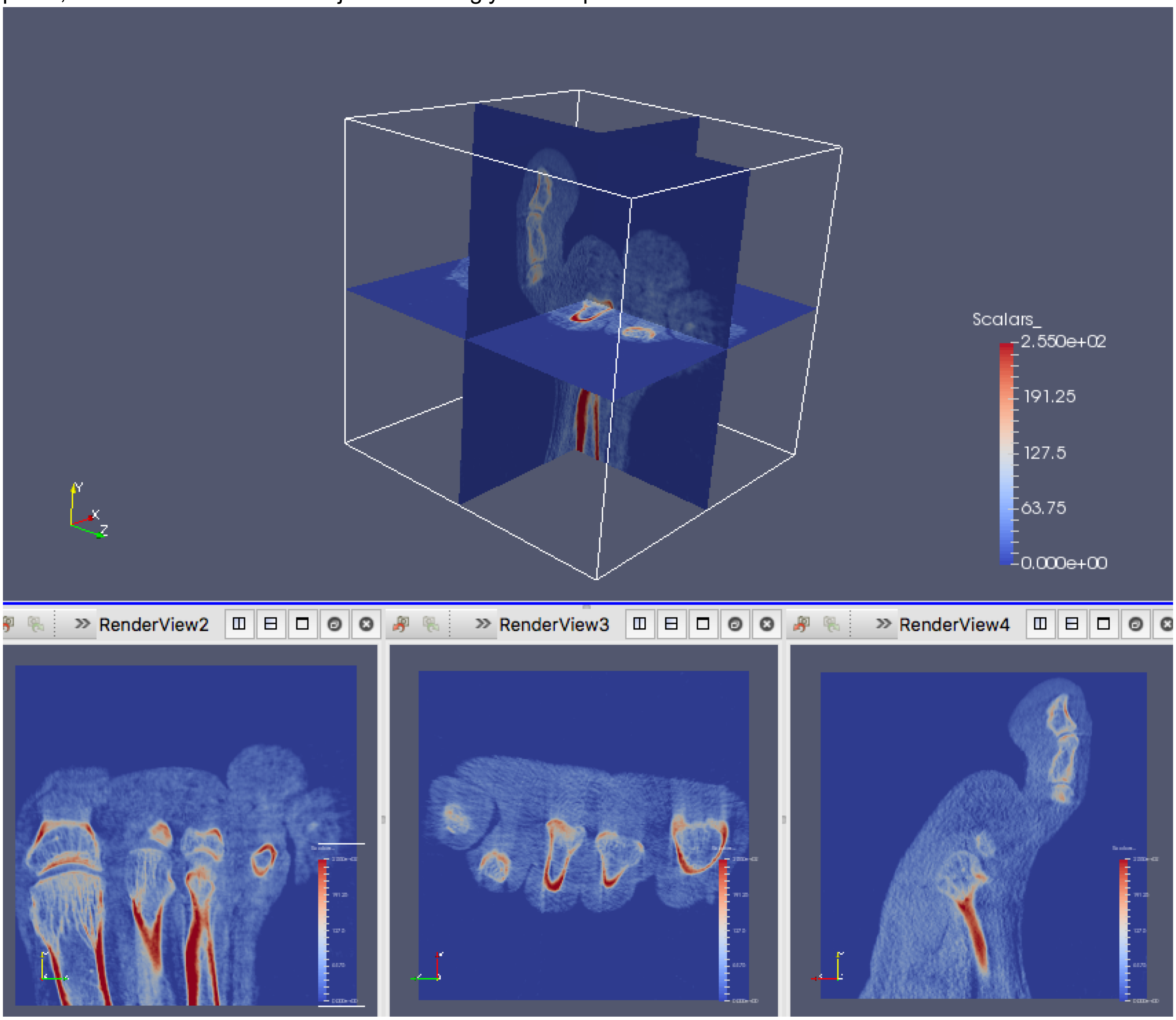
3D images are commonly visualized using axis-aligned slices. ParaView has this capability built in with the Slice filter. Try it out now. Create a slice that is aligned with the X normal of the image.

Rotate the dataset and look at the slice from both sides. Note that where you click is important – if you click inside the red square you can actually adjust the depth of the slice. You can also adjust this manually in the Properties panel. You can also rotate the slice if you click on the arrow widget.

Slicing along a single axis is limited in that it only shows you a certain view of the data. Sometimes, it's helpful to create slices along multiple axes at the same time to get a 3D feel of the data (sagittal, coronal, and axial in the context of medical imaging). Create two additional slice filters, one aligned on the Y normal and Z normal of dataset and view all three simultaneously. Rotate the volume around and investigate it.

Finally, we'll create a linked view in ParaView. To do so, we'll view the same element of the Pipeline Browser in multiple renders. First, split the view vertically so that you have a top and bottom view. Next, in the bottom view, split it horizontally twice to create 3 small views.

One at a time, click on each of the smaller views and make only one of the slices visible. You'll see that it will default to a 3D render view for these slices. Change this to a 2D view and use the camera controls so that you'll see the slice head on. You've now created a multiple linked view. If you adjust the properties of the slice in one panel, the other views should adjust accordingly. The expected result looks like:



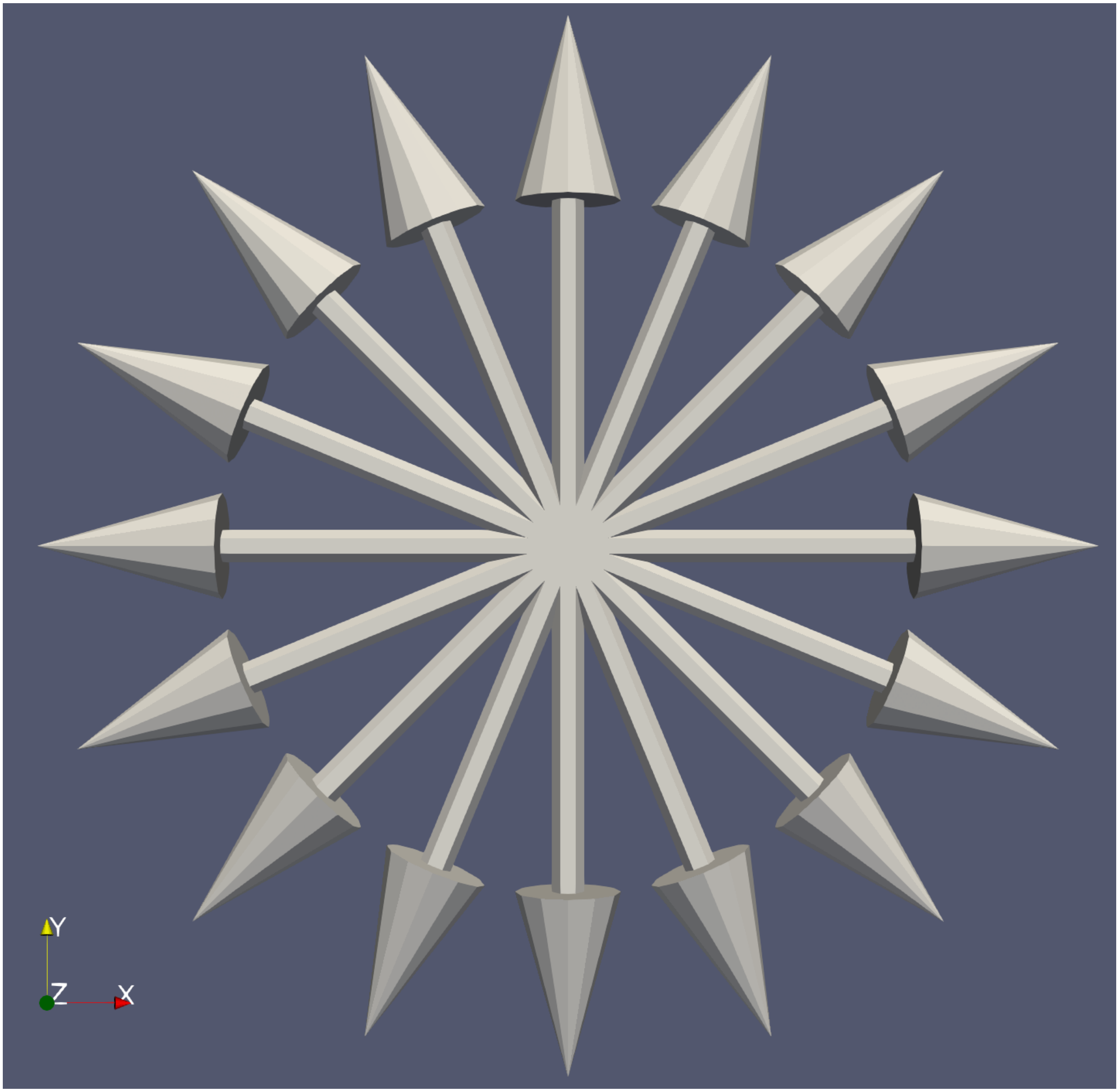
Save your best attempt at viewing this as a state file, 3dImageVis.pvsm

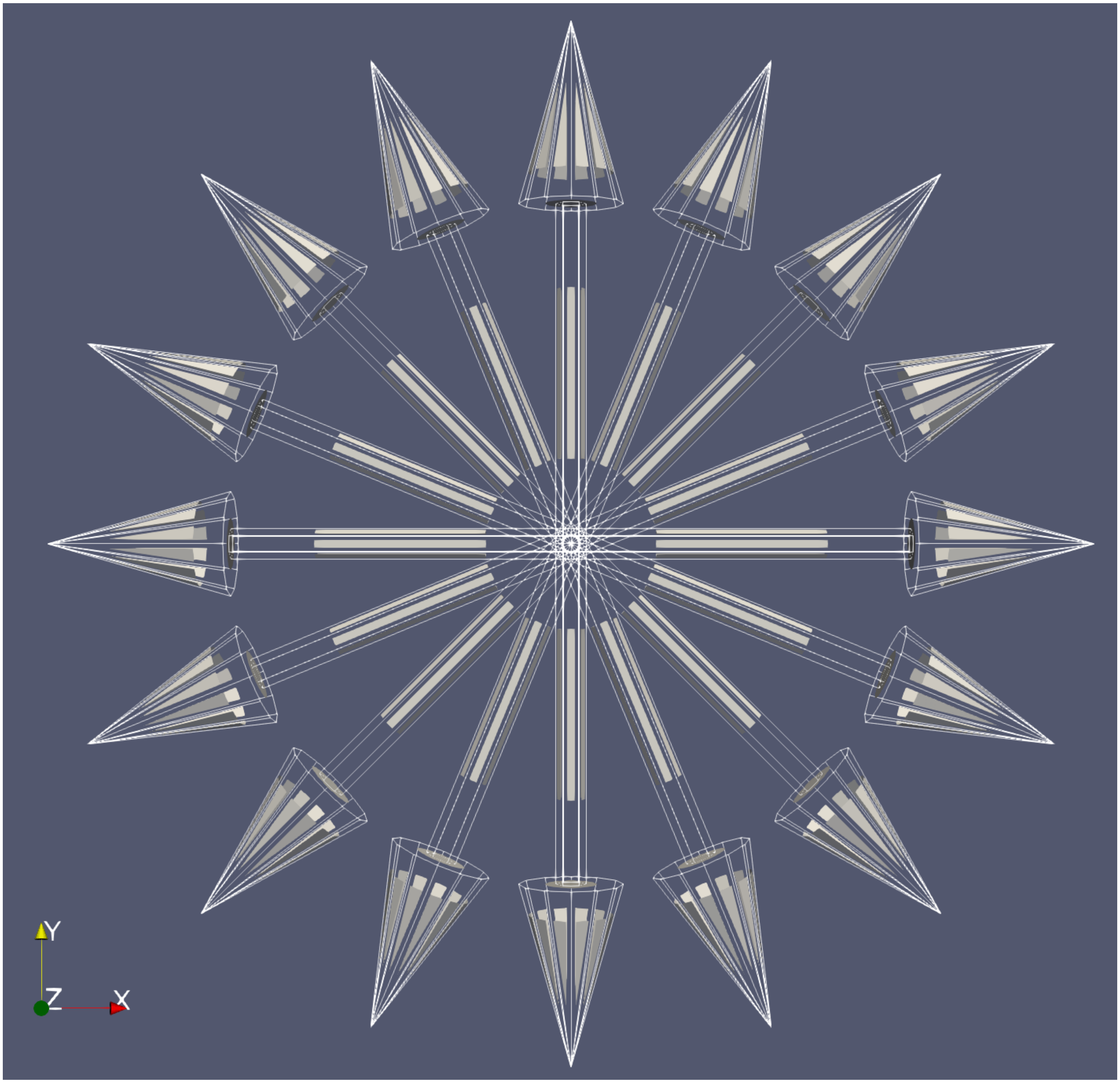
Please submit PVSM files and include images for all parts into your report.

Part 2: Code with Python Script

Q1. Use batch script to create a pipeline [20 pts]

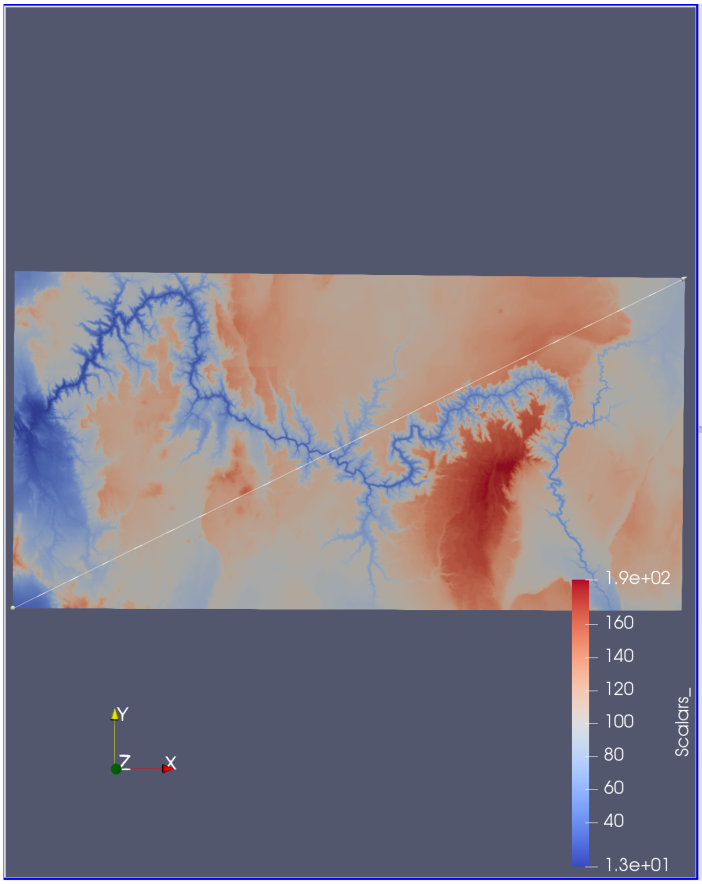
1. Use batch script to render 16 arrows in the renderview. Set the Orientation of the arrow and let them rotate 360° in XY plane. Change the propoerty "TipResolution" to 12. If you render correctly, You will see the first figure below in Paraview.
2. Hide the arrows, Then apply shrink filter & extractedge filter to the arrow. Render the filter in the renderview. you will see the second figure below if you render correctly. Please save the state with "A2P2Q1_1.pvsm" & "A2P2Q1_2.pvsm" and submit thoes two pvsm file with the python script.

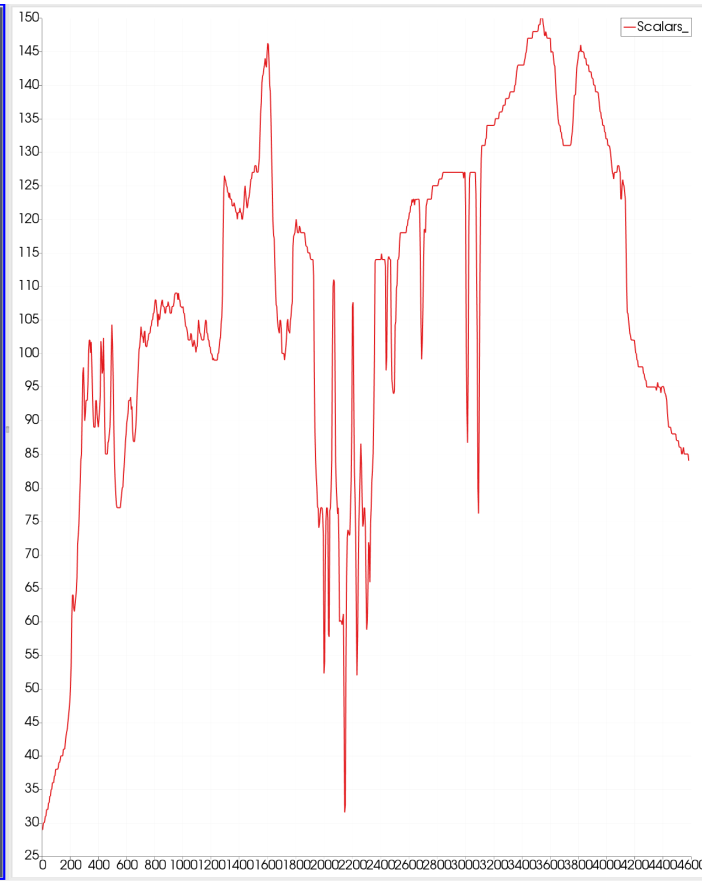


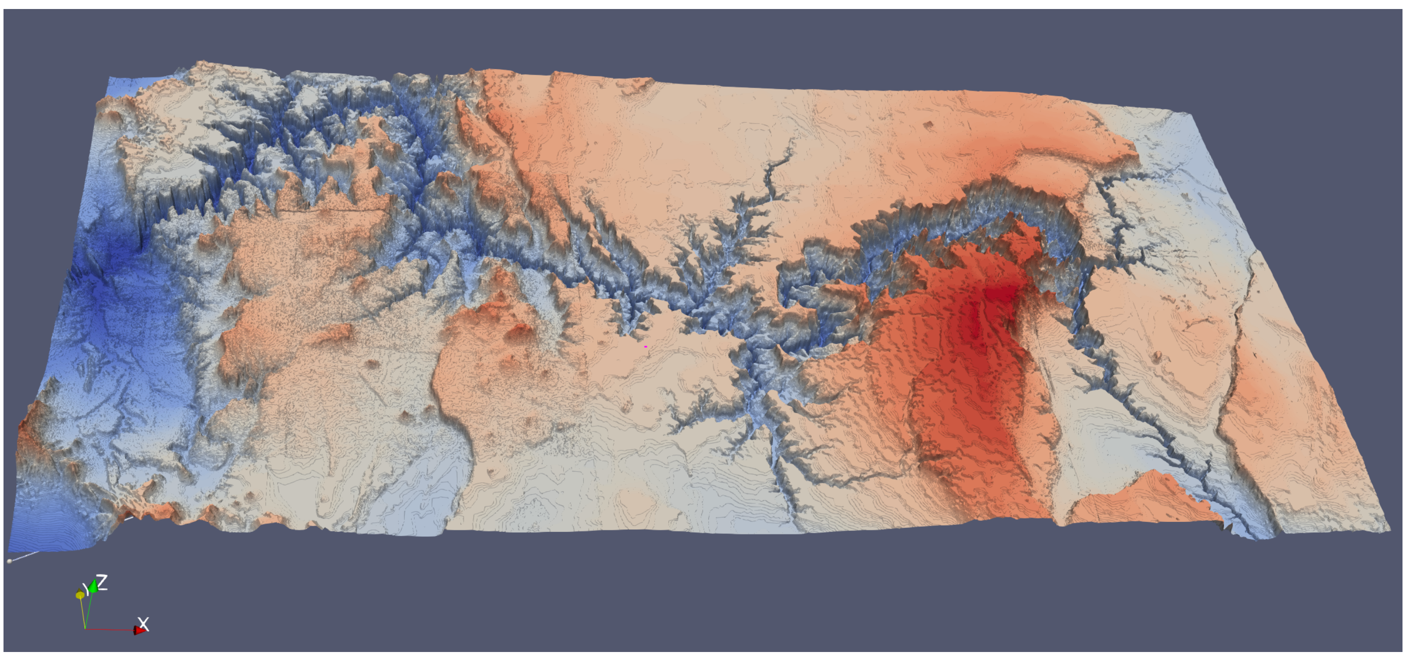


Q2. Read file and process it with Python script [20 pts]

1. Load the data "2d.vti" in Assignment2-Data.zip (http://www.eng.utah.edu/~cs6635/CS6635-Assignment2-Data.zip) into the view with python script. Plot the scalar data use filter "PlotOverline" with script and render it a multiple view. You will generate a figure like this:
2. Find a filter and apply it with Python script to generate a 3D map from the 2D scalar data. Change the filter's property in the script to change the scale factor of scale data. Please comment in the python script that which filter you use in this task. Please submit the python script file. If you find the right filter, you will generate a figure like this:


1.9e+02
160
140
120
100
80
60
40
1.3e+01
Scalars_
Y
Z
X


Scalars_
150
145
140
135
130
125
120
115
110
105
100
95
90
85
80
75
70
65
60
55
50
45
40
35
30
25
0 200 400 600 800 1000 1200 1400 1600 1800 2000 2200 2400 2600 2800 3000 3200 3400 3600 3800 4000 4200 4400


Y
Z
X