Assigned: *1-31-2023*
Due Date: ***2-14-2023 by Noon***

CS 6635/5635 Spring Semester 2024

Assignment 2 (Scalar field visualization with Paraview and Python)

Goal

The goal of this assignment is to visualize different 1D, 2D, and 3D datasets with **ParaView and Python**.

This assignment will help you develop intuition and understanding of the color mapping and other scalar field visualization techniques described in class.

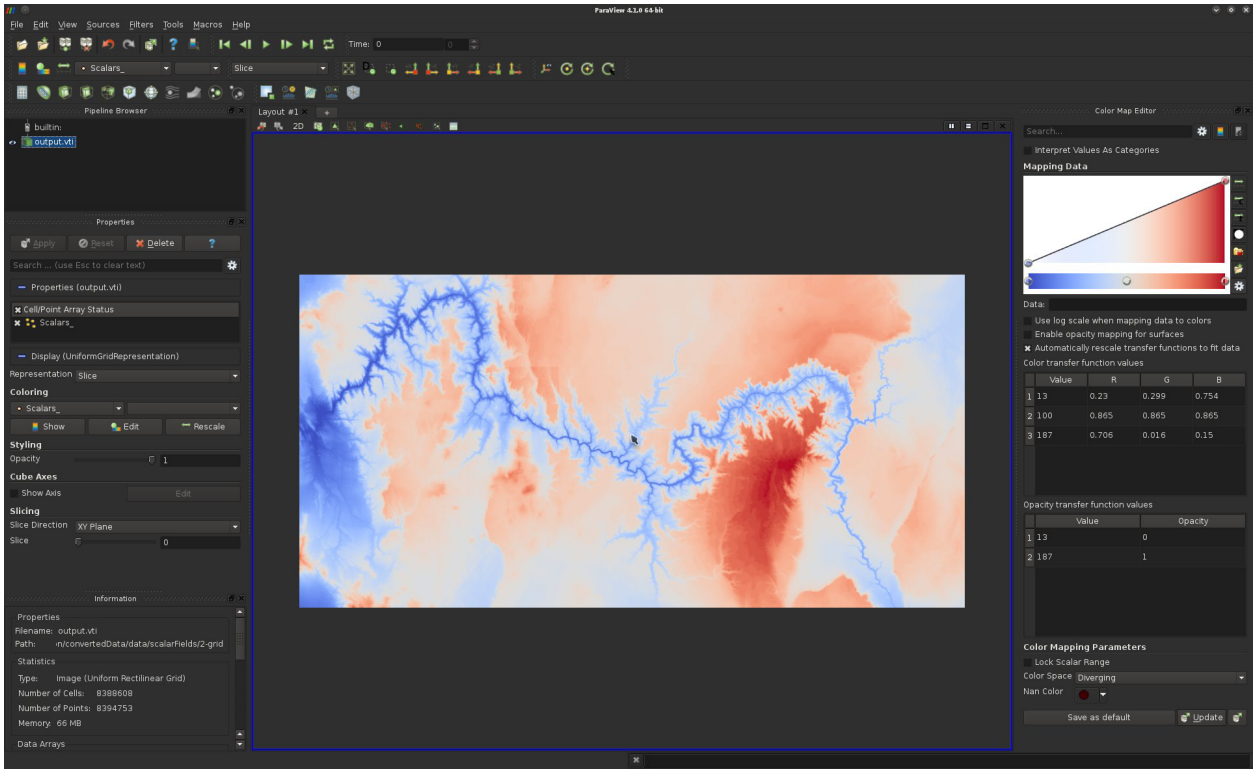
Prerequisites

This assignment requires you to install a recent version of ParaView (https://www.paraview.org). Binary installers exist for Windows, Linux, and Mac.
In addition you will need the following datasets: Assignment2-Data.zip (https://my.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).

1. Visualization of 2d Images [25 pts]



We'll be working with the data file 2d.vti. 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.

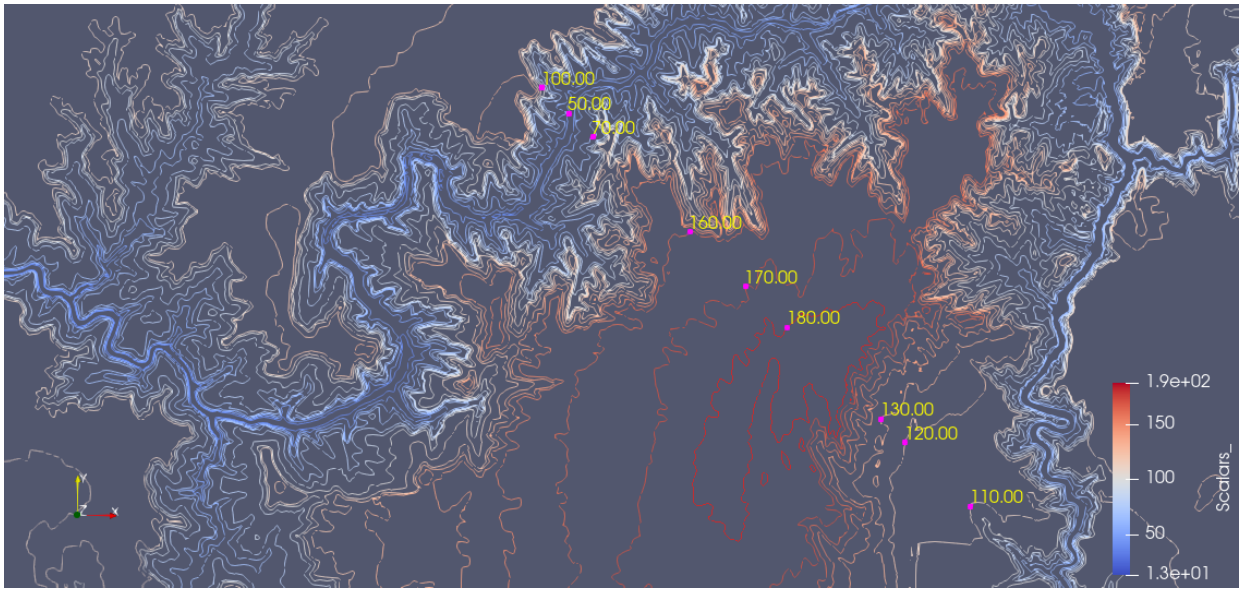
This image has pixels that correspond to height values associated with the Grand Canyon (see a map to compare). Note that if you click and drag with the mouse you can reposition it. Since this data is two-dimensional, ParaView by default loads the render view in 2D mode (you'll see this in the upper 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.
3. Create and label a contour plot of the data using multiple isocontour values



2. Exploring Data on Polygonal Meshes [15 pts]

Next, 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?

3. Visualization of 3D Images [20 pts]

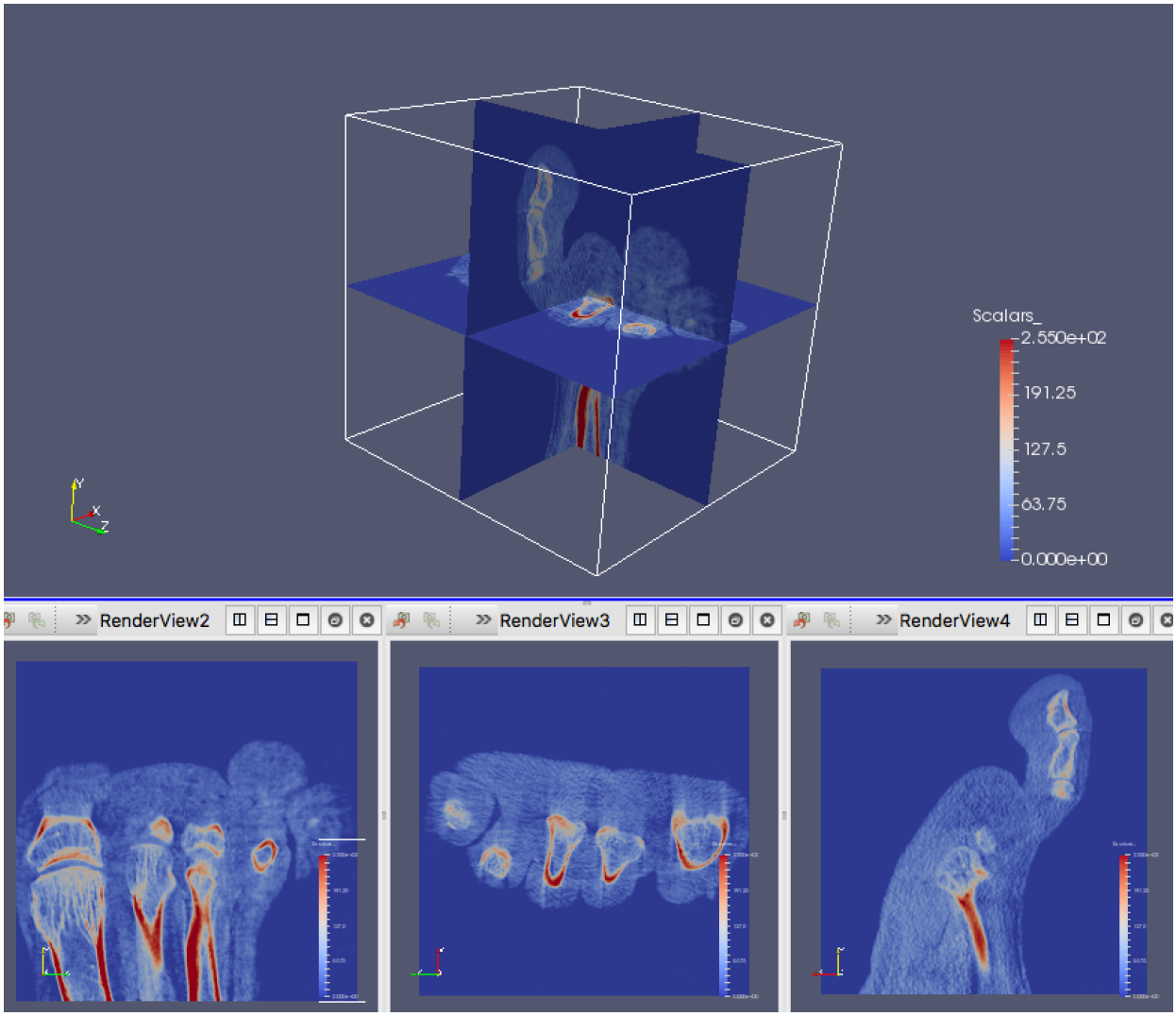
Finally, we'll try out visualizing a three-dimensional image in ParaView. Load 3d.vti. You should see only an outline of the bounding box at this point.

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.

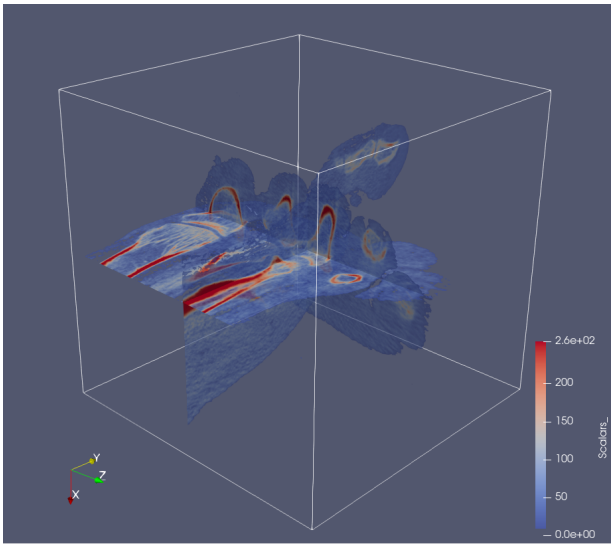
Slicing along a single axis is limited in that it only shows you a certain view of the data. Sometimes, it's helpful to create slices along multiple axes at the same time to get a 3D feel of the data (sagittal, coronal, and axial in the context of medical imaging). Create two additional slice filters, 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:

Now edit the multi-slice view using the colormap editor, such that the background is thresholded out and the remaining data ranges from slightly transparent to fully opaque at the highest value. You will need to enable opacity mapping in the colormap editor. The histogram view in this editor may also be useful.



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.

4. Marching Squares using Python

You are given a 2D scalar field of size 13x13 on a square grid:

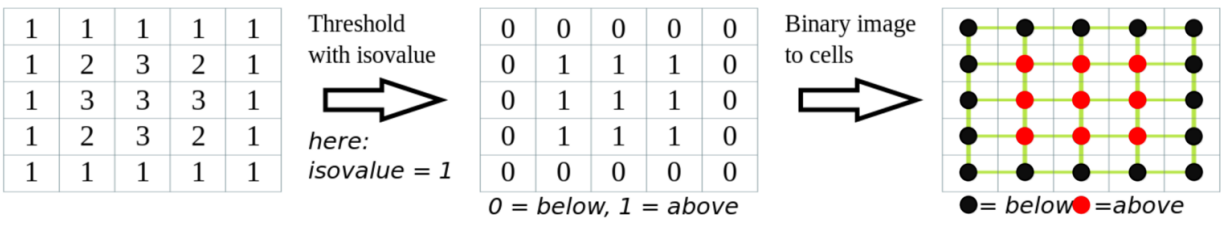
https://my.eng.utah.edu/~cs6635/scalars_2D.npy

You will use this Python code outline:

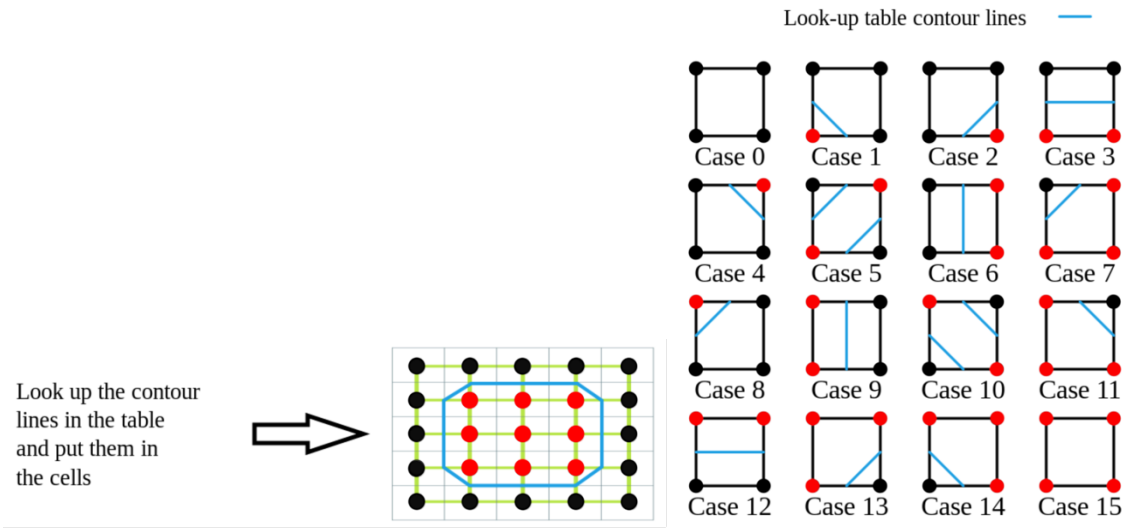
https://my.eng.utah.edu/~cs6635/march_sq_code.py

to implement the Marching Squares algorithm to draw contour lines.

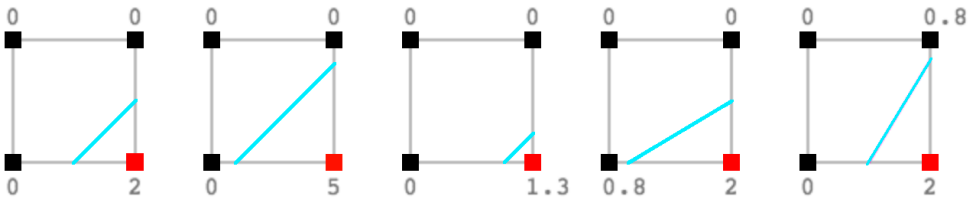
4a. [10 points] Draw a dot at the center of each cell. Color the dot **red** if it is above the isovalue and **black** if it is below. The isovalue is set to 50 by default. You are given a helper function, “draw_dot,” to add a point to the plot. Show an image of your plot.



4b. [15 points] Create a naive implementation of marching squares where lines start and end at the center of cell edges. You will need to loop through groups of 4 cells and use the lookup table to determine where the contour line will be drawn. You are given a helper function, “draw_line,” to add the contour line segment to the plot. Show an image of your plot.



4c. [15 points] Modify your marching squares algorithm to use linear interpolation for placement of line vertices. Show an image of your plot.



Even though each of these cells would be classified as type "0100 = 2", the green line approximating the set of points within the cell where $f(x, y) = 1$ can vary dramatically depending on the underlying sample values.

Time-Dependent Isosurface Extraction (Visualization Handbook Chapter 3)

https://my.eng.utah.edu/~cs6635/VH.pdf

(Only for CS6635)

Isosurface Animation [5 pts]

1. Load the data "headsq.vti" from ParaViewTutorialData Folder to Paraview.
2. Click "Contour" for extracting Isosurfaces.
3. Open "Animation View", set Mode to sequence, Start Time to the minimum value and End Time to the maximum value in the volume and you can change the No.Frames to 100 to adjust animation speed.
4. Find the approximate range of isovalues for the transition between skin and skull. At what isovalue the spine vanishes? What's the approximate isovalue for the teeth. Attach screenshots with your answers.

Reading Questions [15 pts]

1. What is time-varying data? What challenges do we face when extracting isosurfaces from time-varying data.

2. Briefly explain the need for temporal hierarchical index tree data structure for isosurfaces extraction in time-varying data.

3. Given the temporal hierarchical index tree, briefly describe isosurfaces extraction algorithm in time-varying data in your own words.

What to turn in:

Write a **report documenting your results**, including any necessary plots/figures, and answering any questions asked above. Be sure to explain any figures you submit and to write a **conclusion** at the end of your report. Your homework is primarily graded upon your report. Please submit your report on Canvas in PDF format. Please also submit your code as compressed zip files.

- Your report should be in PDF format and should stand on its own
- It should describe the methods used.
- It should explain your results and contain figures.

- It should also answer any questions asked above.
- It should cite any sources used for information, including source code.

Note: Any figures/plots in the report should be captioned appropriately. Also be sure to include axis labels in all plots.

This homework assignment is due on **February 14, 2023 by 11:59 am**. If you don't understand these directions or have questions, please send questions to teach-cs6635@sci.utah.edu or come see one of the TAs or the instructor during office hours **well in advance of the due date.**