Class: CS 6635
Assignment: 2
Name: Trupti Mohanty

1. Visualization of 2d Images [25 pts]

Q1. 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.

Answer: To capture the riverbed of the Grand Canyon, I used the lower threshold value as 13 and upper threshold value as 90. While increasing the lower threshold value > 13, not able to capture entire riverbed as some of the riverbed areas not visible clearly. Furthermore, increasing the upper threshold > 90 captures other regions apart from riverbed and the color is more towards red.

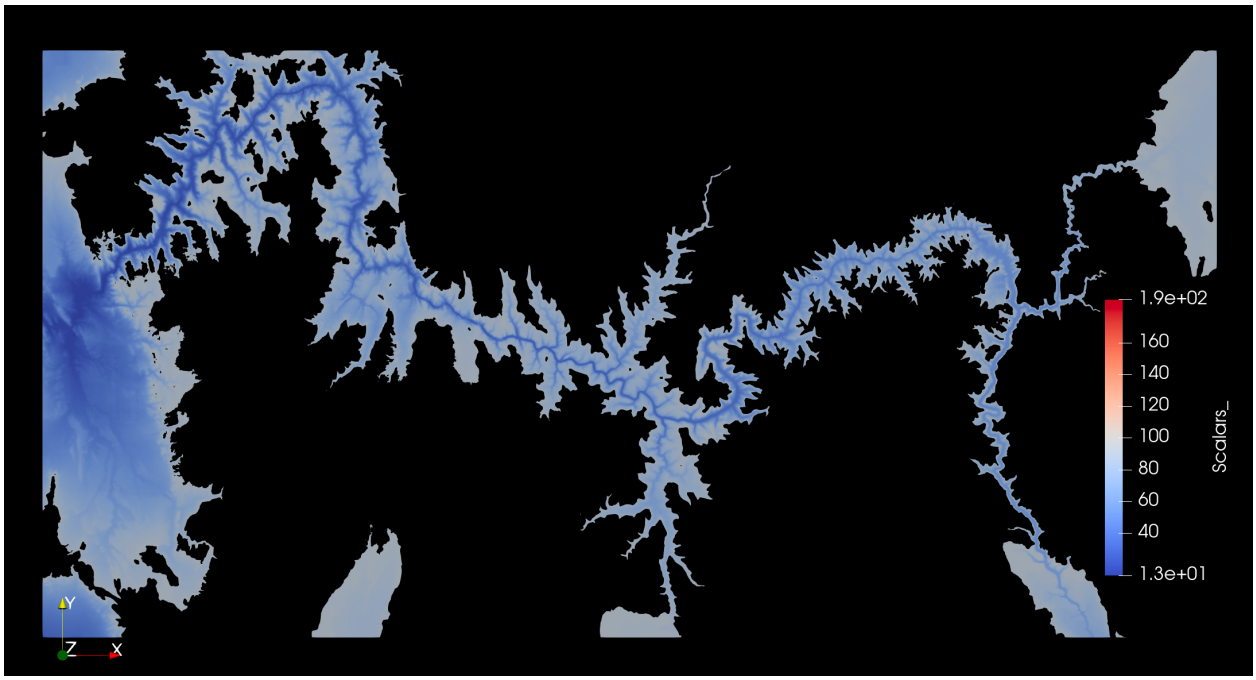

Fig 1. Shows the riverbed of the Grand Canyon using a threshold filter range 13-90

Q2. 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.

Answer: Number of points as per the information panel in the thresholded image: **2,053, 863**

Q3. Create and label a contour plot of the data using multiple isocontour values.

Answer: Applied the contour filter and choose different iso-values in the range 13 – 187 to show the contour lines. Figure 2. shows the contour plot with multiple iso-contour values of the Grand Canyon data.

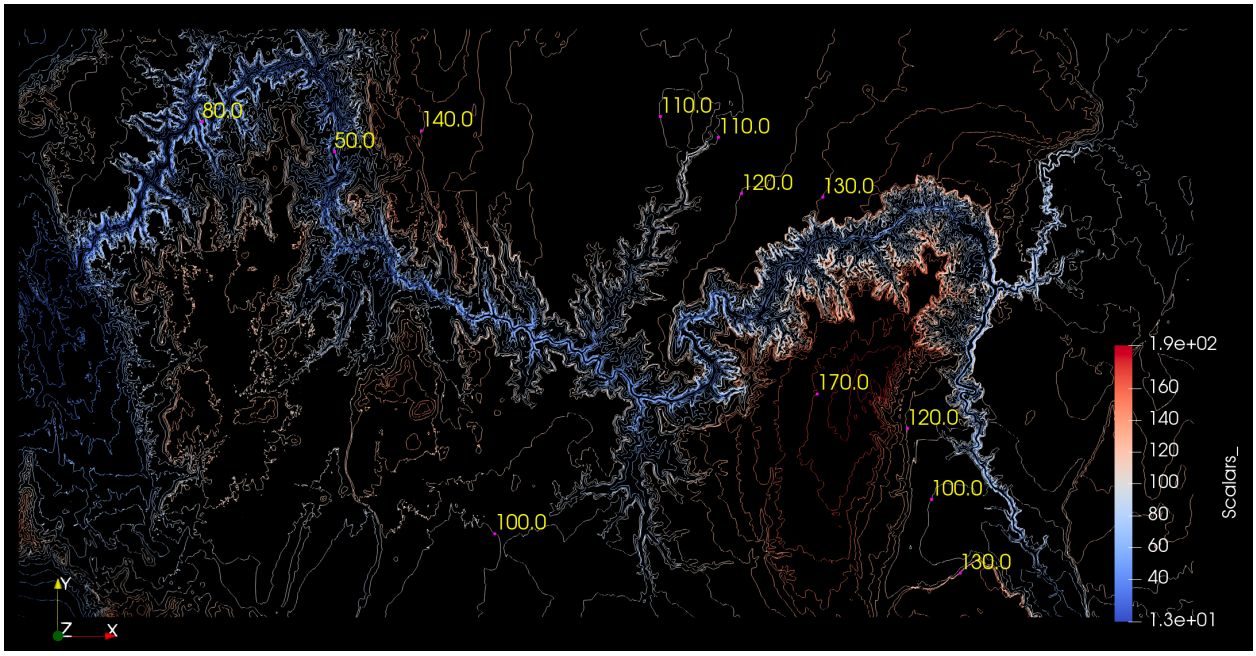

Fig 2. Shows the contour plot with multiple iso-contour of Grand Canyon data with labels.

2. Exploring Data on Polygonal Meshes [15 pts]

Q1. What were the minimum and maximum values that best captured the single cylinder associated with the bolt's cylinder?

Answer: Applied the threshold filter and the minimum and maximum values that best capture the single cylinder associated with the bolt's cylinder are **0 and 40** respectively. When I increase the upper threshold value > 40 other parts also get visible apart from the bolt cylinder and when the upper threshold is < 40 not able to capture the entire bolt cylinder. Furthermore, increasing the lower threshold value not able to visualize the bolt cylinder. Fig 3. Shows the details of the surface view with threshold filter ranging 0 – 40 indicating the bolt cylinder. Furthermore, we can see the ventilation slots in the cylinders in wireframe view. To make the ventilation slots more clear I used the clip filter.

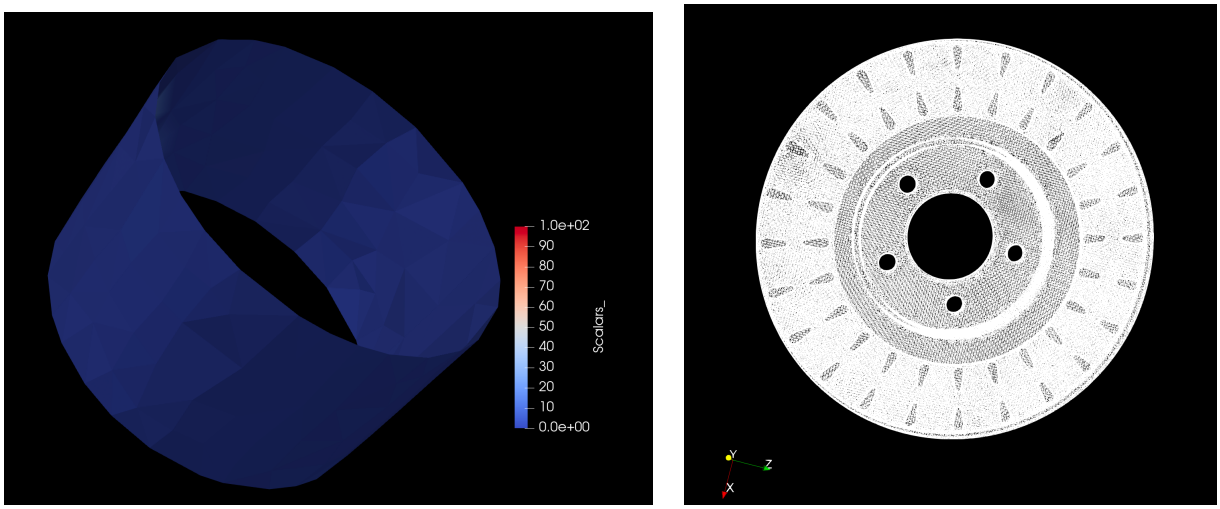

Fig 3. (Left) shows the surface view of bolt cylinder after using a threshold filter (0-40) and (Right) shows the wireframe view of the cylinders.

Q2. How many ventilation slots are there?

Answer: There is total 48 ventilation slots (Inner 24 / Outer 24) as shown in figure 3 and 4.

Figure 4. shows the ventilation slots after using the clip filter with Y normal direction. The ventilation slots are looking like tiny pillars and more clear than wireframe view.

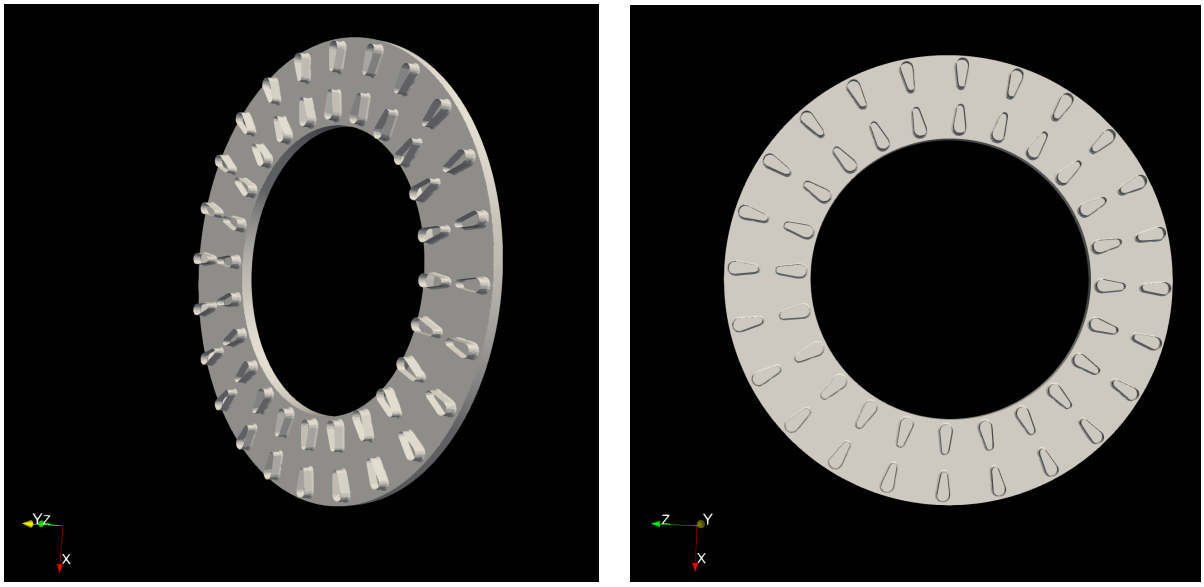

Fig 4. Shows the ventilation slots after using the clip filter in Y normal direction.

3. Visualization of 3D Images [20 pts]

Took the 2D slices of the 3D image with X, Y and Z normal direction. Represented all the three slices together and individually in figure 5.

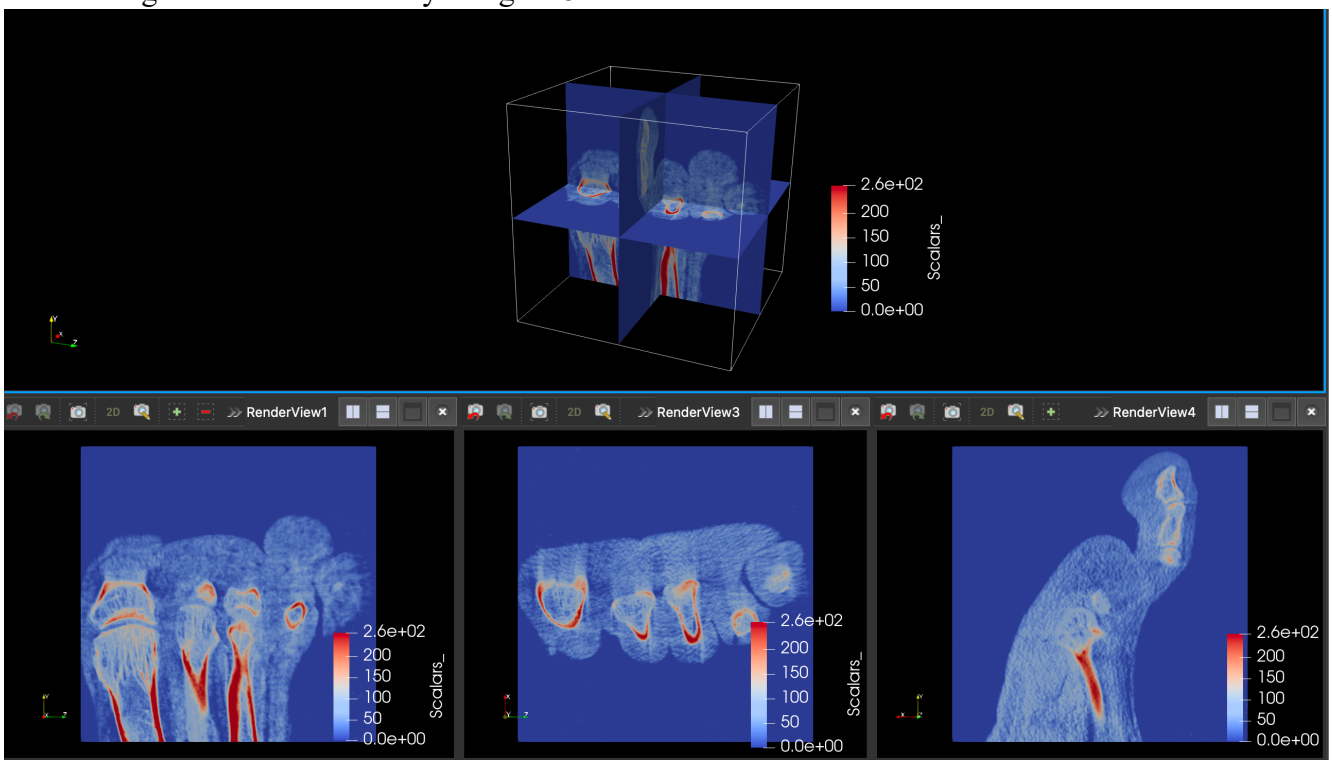

Fig 5. Shows all 3 slices together (with X, Y, Z normal direction) and their individual slices.

Enabled the opacity mapping in the color map to remove the background. Highly dense tissues are represented in red color which are the bones as shown in figure 6.

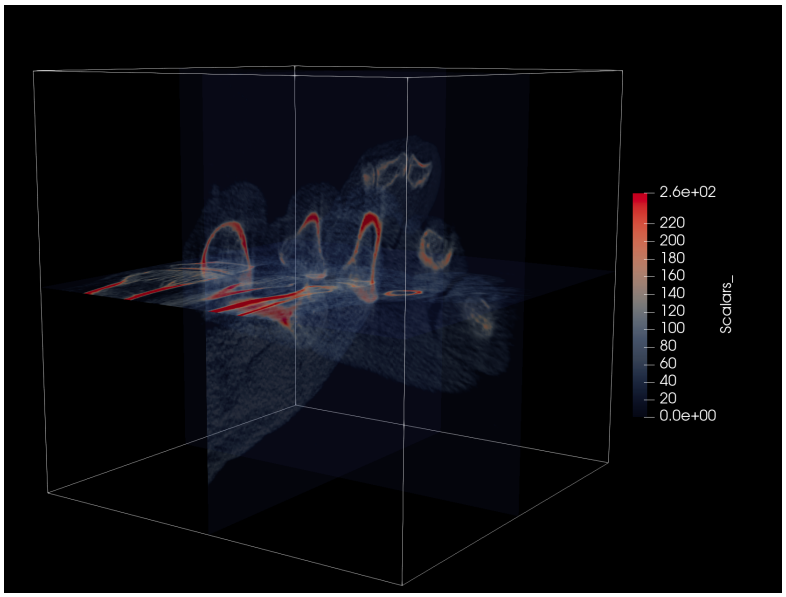

Fig 6. Used opacity mapping to threshold out the background.

4. Marching Square Implementation using python

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.

Answer: The isovalue is 50. The vertices above 50 are red in color and rest are black implemented in the marsq.py file.

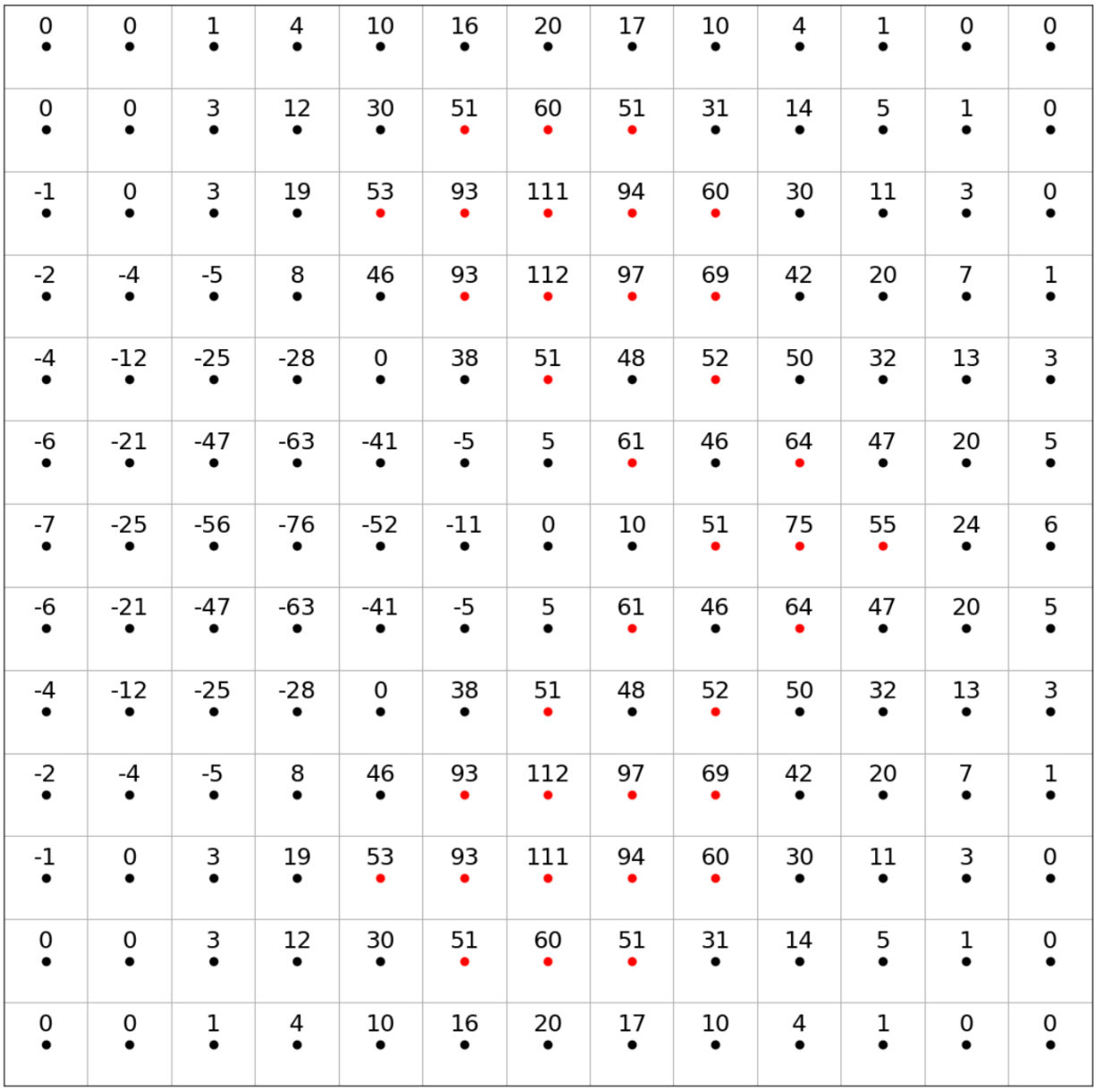

Fig 7. Red dots have values greater than isovalue (50) and black dots have values less than isovlaue (50)

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.

Answer: Implemented the function "march_sq_midpoint" which takes care of all the cases mentioned in the look up table. Figure 8 shows the contour line drawn using the mid point method for isovalue 50.

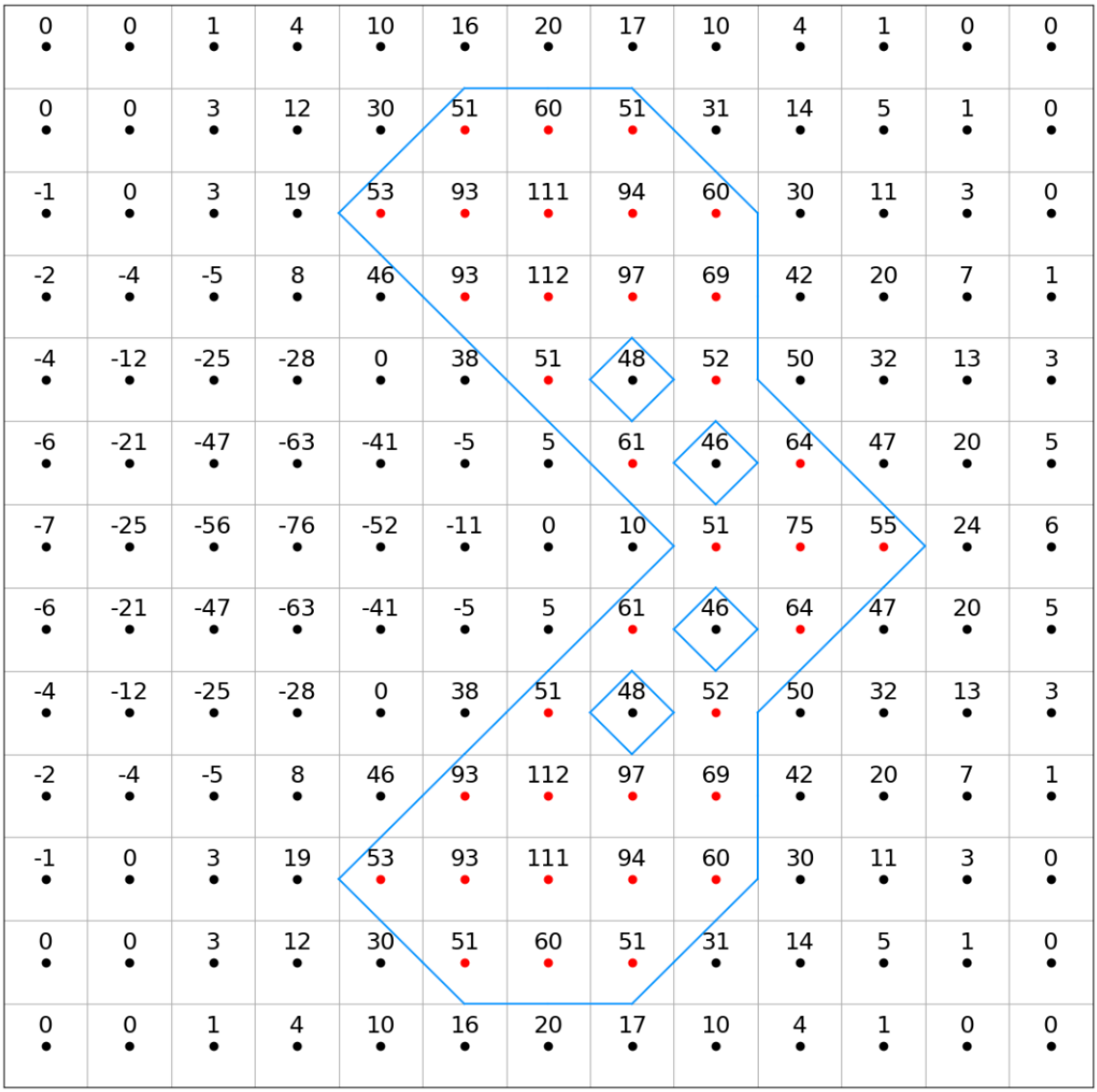

Fig 8. Contour line segments are added to the plot using the mid-point method and the look up table.

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

Answer: Implemented the function "march_sq_lin_interp" which takes care of all the cases mentioned in the look up table and also uses the interpolant value instead of the midpoint. Figure 9 shows the contour line drawn using the interpolant method for isovalue 50.

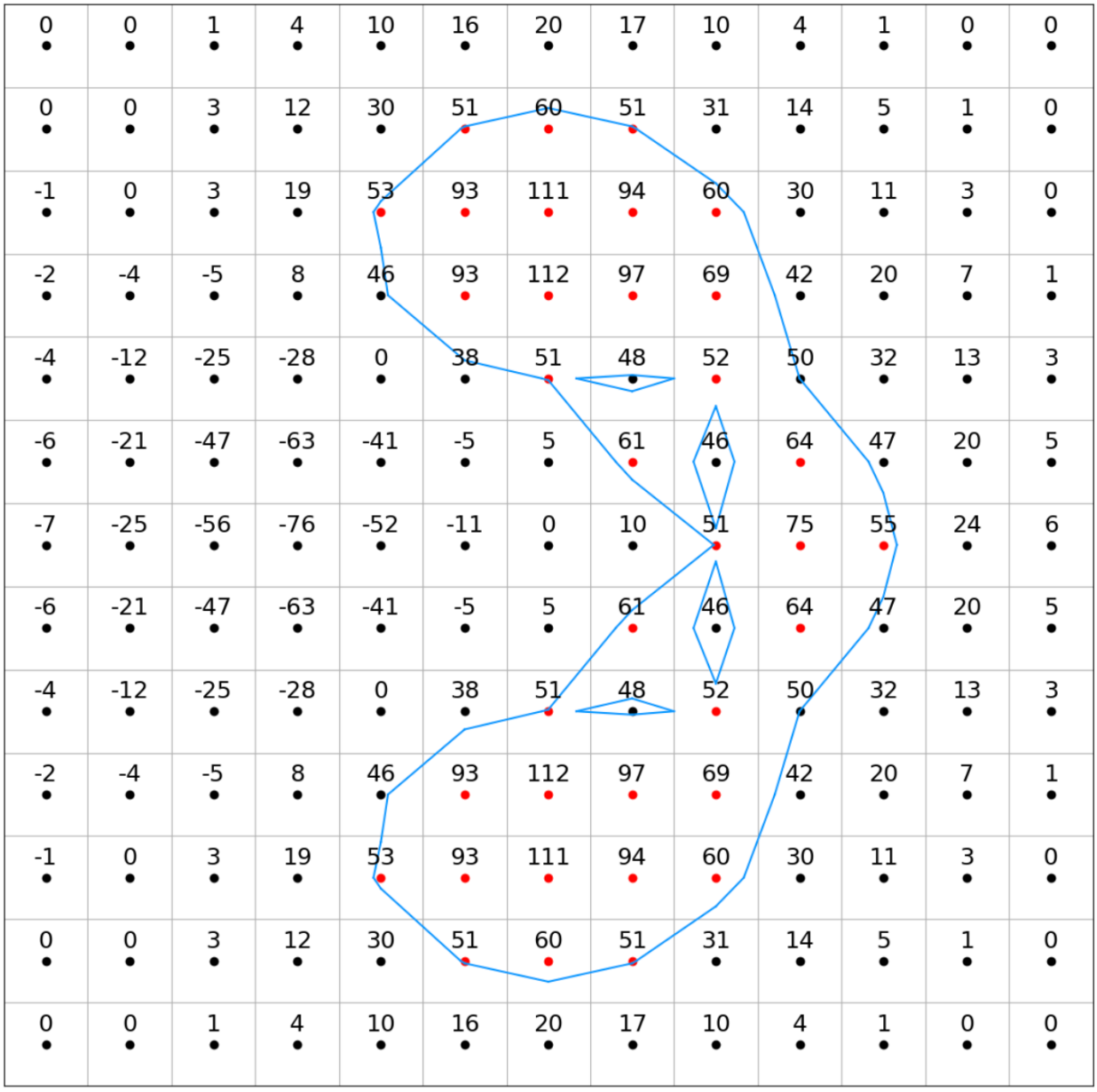

Fig 9. Contour line segments are added to the plot using the interpolant points and the look up table.

Isosurface Animation [5 pts]

Answer: Loded the data "headsq.vti" in paraview and used the contour filter for extracting the isosurfaces. Used the animation view and set the mode to sequence and used 100 frames per second. Further choose the contour with isosurfaces in animation view properties.

The approximate range of isovalues for the transition between skin and skull (with spine) is 620 to 1158 and 620 to 2523 (without spine).

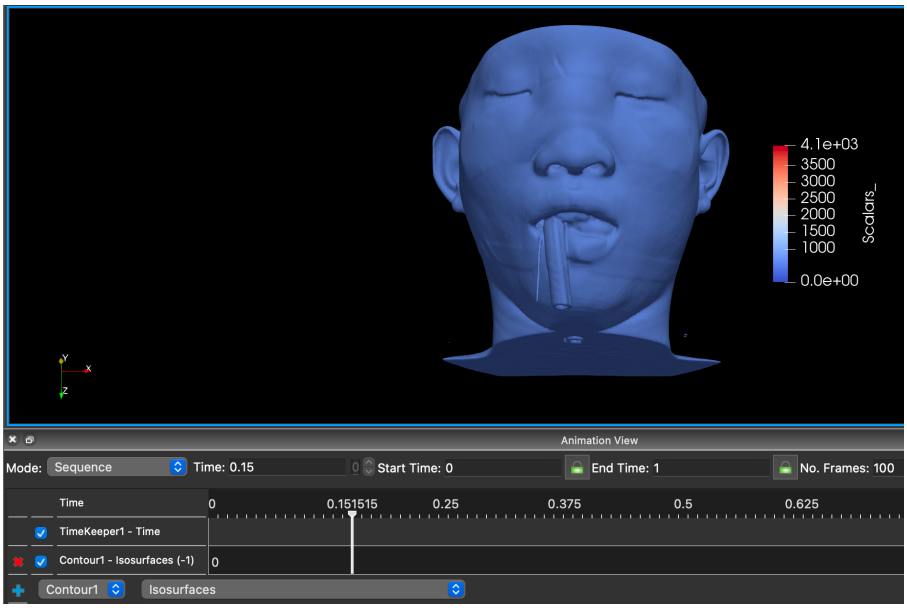

Fig 10. Isosurface animation view, when the skin is visible. Isovalue around 620

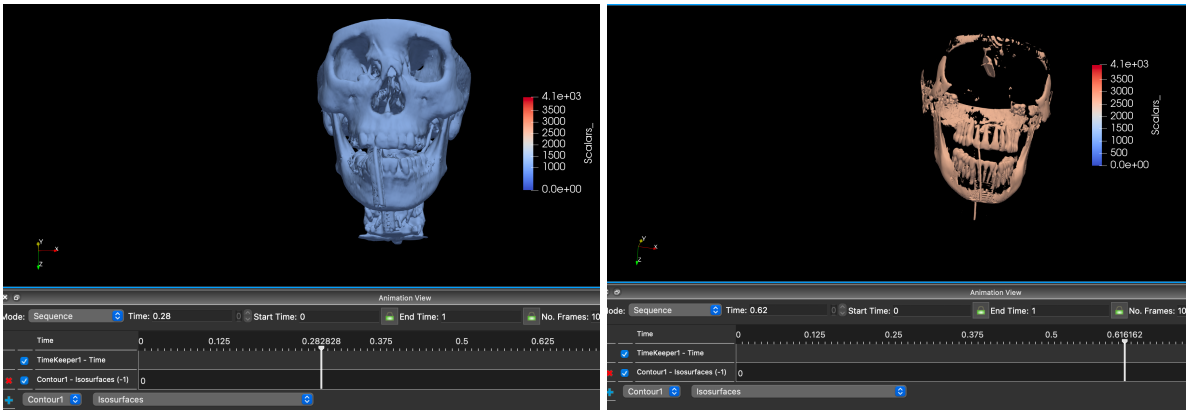

Fig 11. Isosurface animation view, (left) when the skull started visible Isovalue around 1158, (right) skull (without spine) Isovalue around 2523.

Answer: The spine vanishes at approximate isovalue **2523** as shown in figure 12.

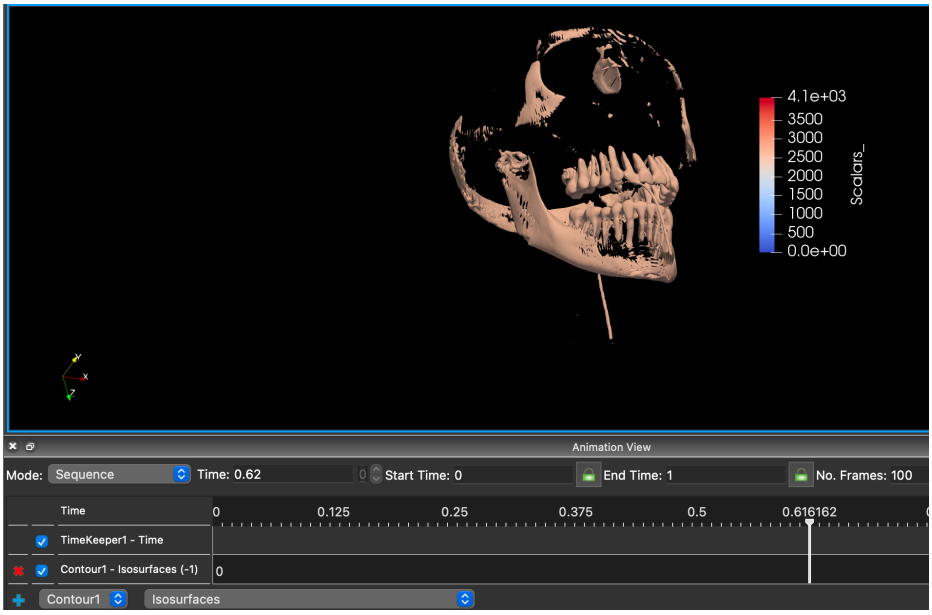

Fig 12. Isosurface animation view, when the spine is invisible at approximate isovalue 2523

The isovalue where only teeth are visible is **2895.**

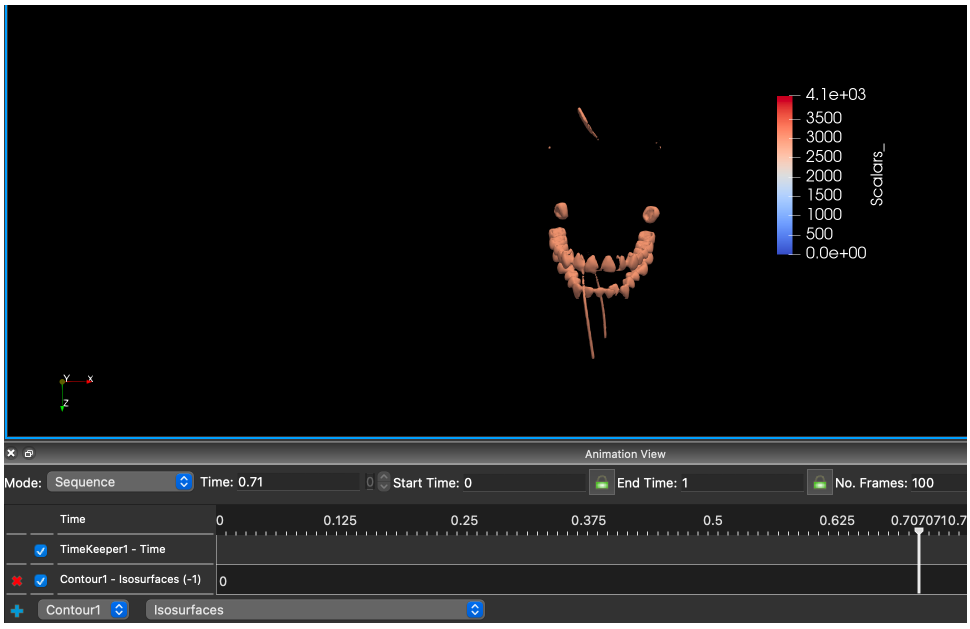

Fig 13. Animation view, when only teeth are visible, Isovalue 2895

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

Time varying data are the data which changes over time. It may include the state of chemical system at different time step, weather patterns or a patient's medical record. The challenges associate with time varying data are to navigate huge amount of data in space and temporal domain, selecting data in different resolution, working with different visualization parameters and animating certain features over time.

The process of extracting the isosurfaces from time varying data is often challenging. Marching cube algorithm is the simple and robust algorithm however it is computationally expensive due to the linear search required for the extraction isosurface cells. To address this issues, various accelerated algorithms have been developed. However, these algorithms require larger storage space to access the data indices associating with a tradeoff between performance improvement and increased I/O overhead.

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

As we discussed in the previous question, though the marching cube algorithm is robust and straightforward, it is associated with computational complexity, and the accelerated algorithms are associated with higher storage space to access the data indices while extracting the isosurface from time-varying data. Therefore, the data structure is needed for the efficient extraction of cells required for the iso surface generation without compromising the performance and using less storage space. The goal of the temporal hierarchical index tree data structure is to provide an adaptive approach that minimizes the storage demand associated with indexing for isosurface extraction in time-varying data. This method involves classifying the cells based on the degree of variation in their values over time.

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

In the temporal hierarchical index tree, cells exhibiting minimal changes in the values over time are grouped into a single node, while those with significant variation are distributed across multiple nodes at different labels of the tree. The structure employs uniformly distributed rectangles known as lattice points to organize the cells. It utilizes the lattice subdivision to categorize the cells based on temporal variations of their extreme values. At the root of this data structure are the cells having low scalar variability over the time interval. The time interval is then divided into half, and the process of classifying the cells based on their scalar value is recursively applied till we obtain the leaf node, which contains the cells with higher variability.

During the extraction of cells for isosurface generation, the algorithm systematically traverses the tree starting from the root node corresponding to the broader time interval and progressively refining the search to the nodes consisting of cells of higher scalar variability with narrower time intervals. As the cells are categorized based on their temporal variation, the algorithm selectively

processes only those cells associated with the required isosurface, thus making the extraction process efficient with less computation.

Conclusion:

- In this homework assignment, I systematically studied how to use different filters in Paraview.

  Threshold filter - To highlight certain part of the image based on its scalar value range.
  Contour filter - To get an idea of different isolines and isosurfaces.
  Clip Filter – To visualize the cross-sectional view in different direction.

- How to create 2D slices in X, Y, and Z normal directions from a 3D image and use the opacity mapping to avoid the background and focus on the actual data.

- Animation view is used to visualize the isosurfaces in time-varying data.

- The marching square implementation helps to understand how to draw the contour lines based on given isovalue using mid point method and linear interpolant method.

- The reading assignment helped me to understand the efficient temporal hierarchical index tree data structure used for the time-dependent isosurface extraction.

References:

1. https://www.paraview.org/Wiki/images/b/bc/ParaViewTutorial56.pdf