1. Visualization of 2D images

1.1

The threshold value was approximately 98 - 330000, I found from the histogram plot (Figure 1):

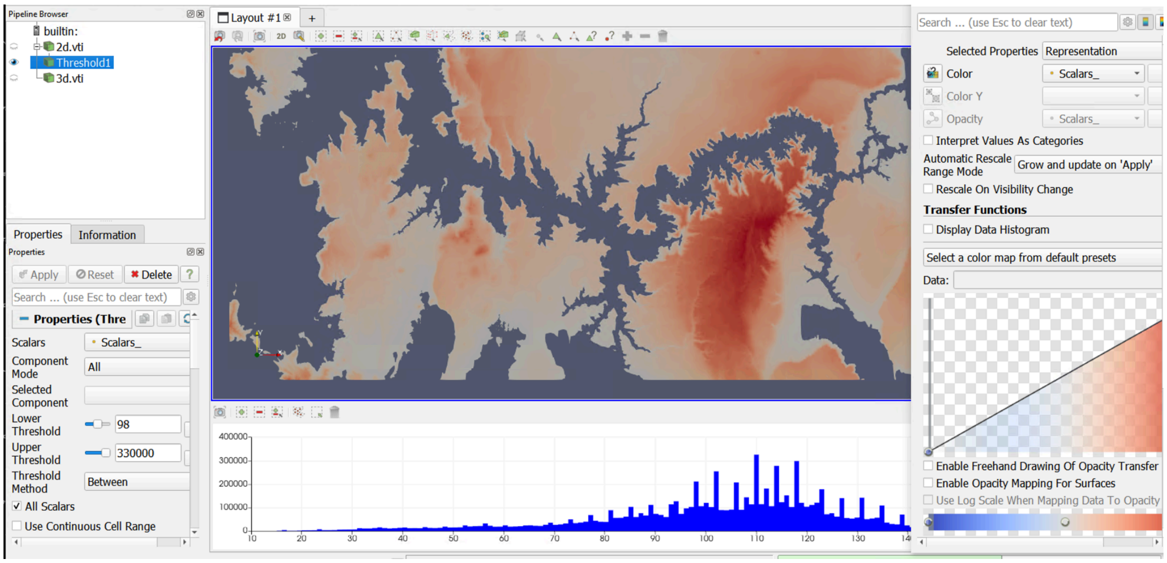

Figure 1: Threshold for Riverbed

If the minimum threshold is very low, then no riverbed is distinguishable, if the maximum threshold is very high, all other canyon heights

1.2

The information is presented in Figure 2:

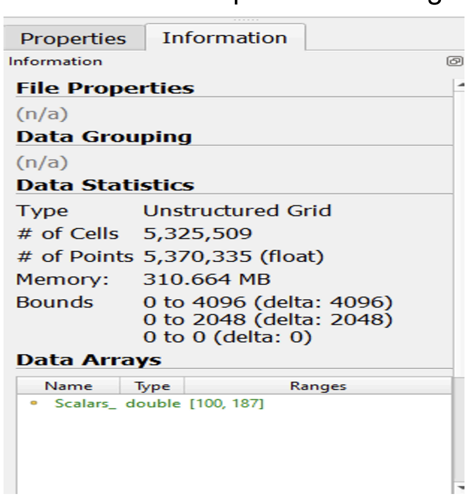

Figure 2: Number of points in threshold image

1.3

In Figure 3, I have generated a contour for a value range of 13 to 330000, and in Figure 4, the range is 13 to 187, which is low in terms of detecting riverbeds. For Figure 4, I have toggled in the interaction point selection and then edit->find data. When I select a point in contour, the find data function returns a point, and I label the value with the point labels feature. Note that the same can be done with Figure 3, too.

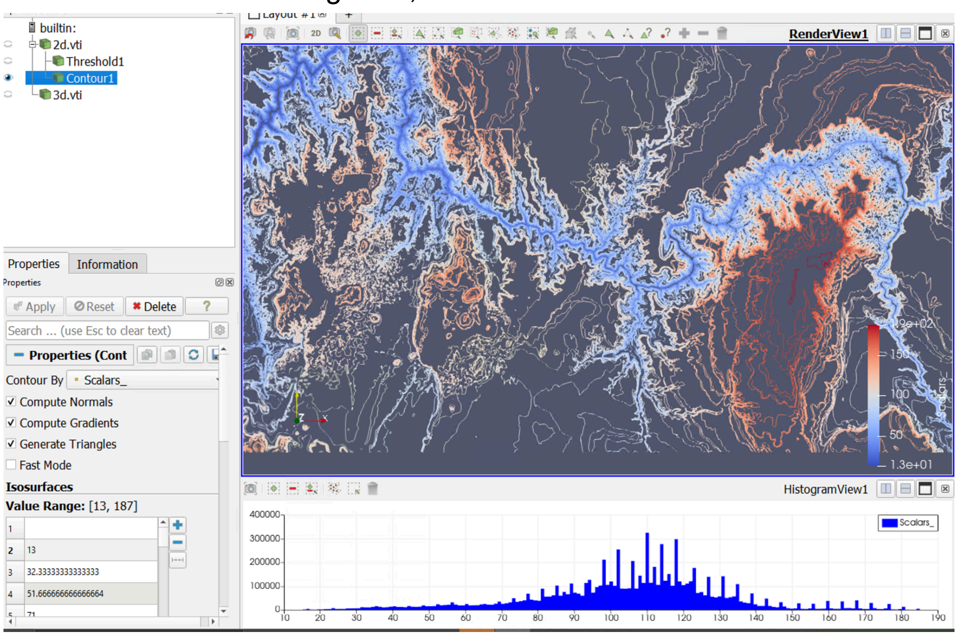

Figure 3: Contour selection

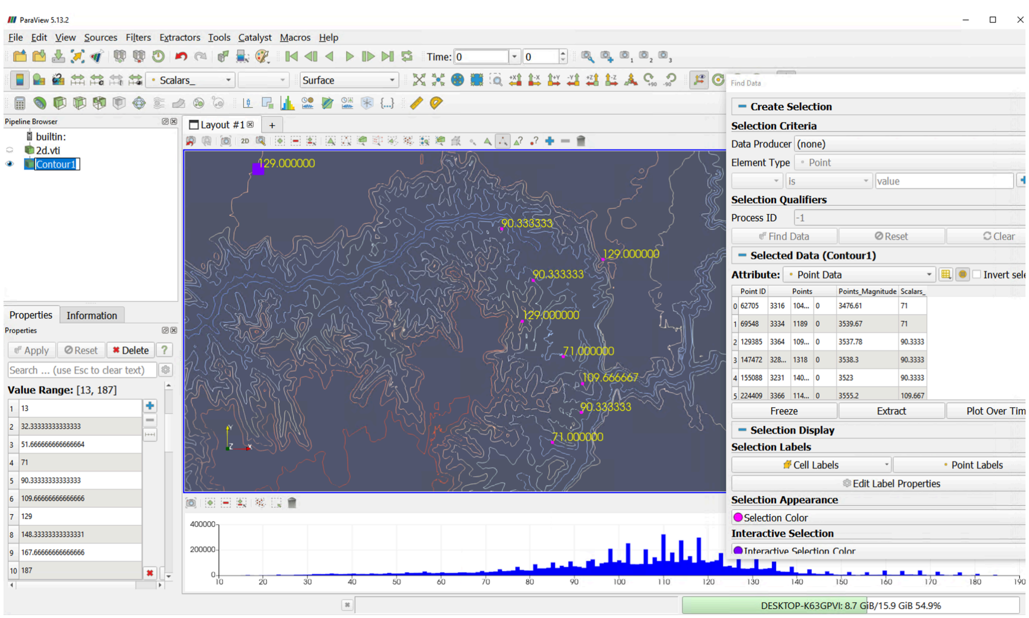

Figure 4: Contour with labels

2. Polygonal Mesh

2.1

Figure 5 shows one cylinder, as shown in the threshold view. The threshold value I found for this is 0-44

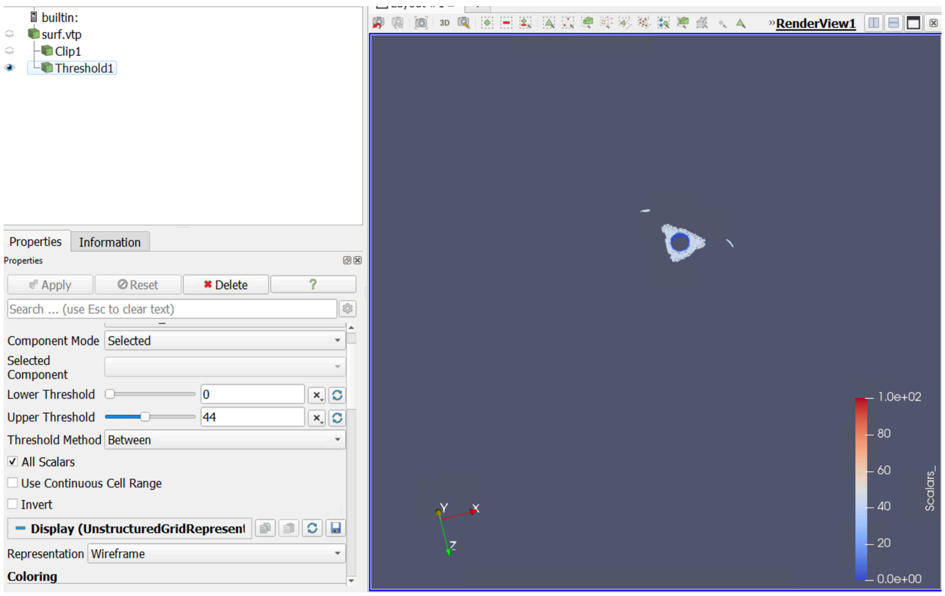

Figure 5: Threshold selection for one cylinder

2.2

From Figure 6, we can see 48 ventilation slots with the clip plane settings (Origin: 0, 12.2771, 0 with Y Normal) :

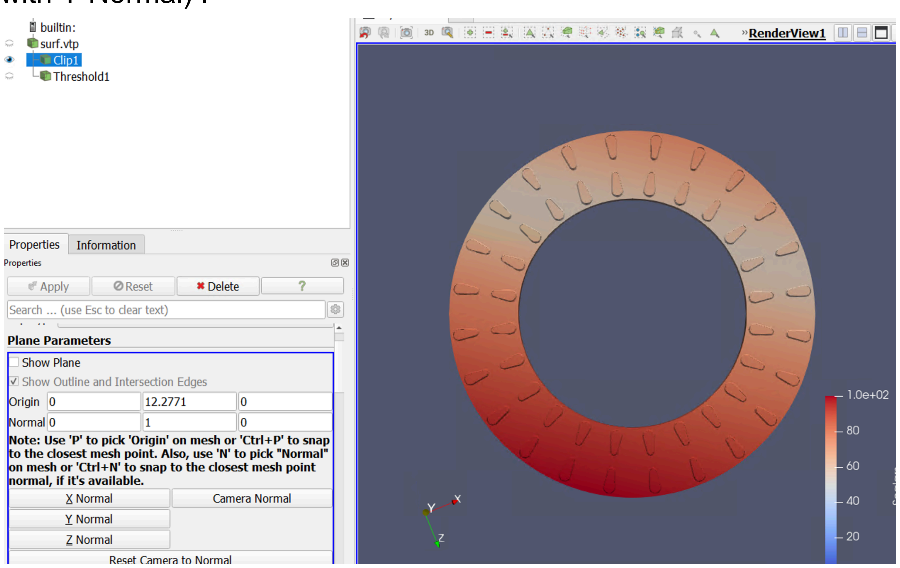

Figure 6: Ventilation slots

3. Visualization of 3D Images

In Figure 7, I have indeed made my best attempt to generate opacity mapping in the colormap editor for the 3D data. At the colormap editor in Figure 7(a), we can see the transfer function histogram and opacity value that resulted in this output (Figure 7(b)):

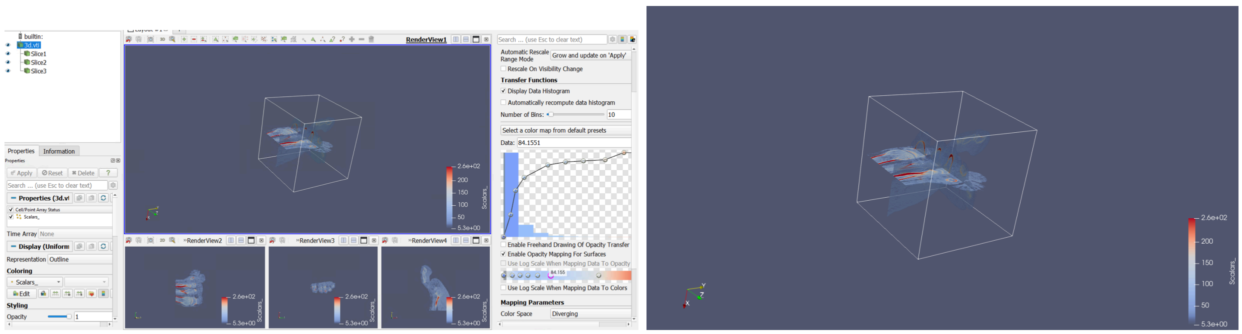

Figure 7: Opacity colormap editing (a) the transfer function values (b) the output

4. Time-dependent Isosurface

4.1

After doing all the steps mentioned in the assignment, found below results:
In Figure 8, we can see the value for face is 918, in figure 9, the value for Skull is 1458. So, the approximate range is 918 - 1458 for me:

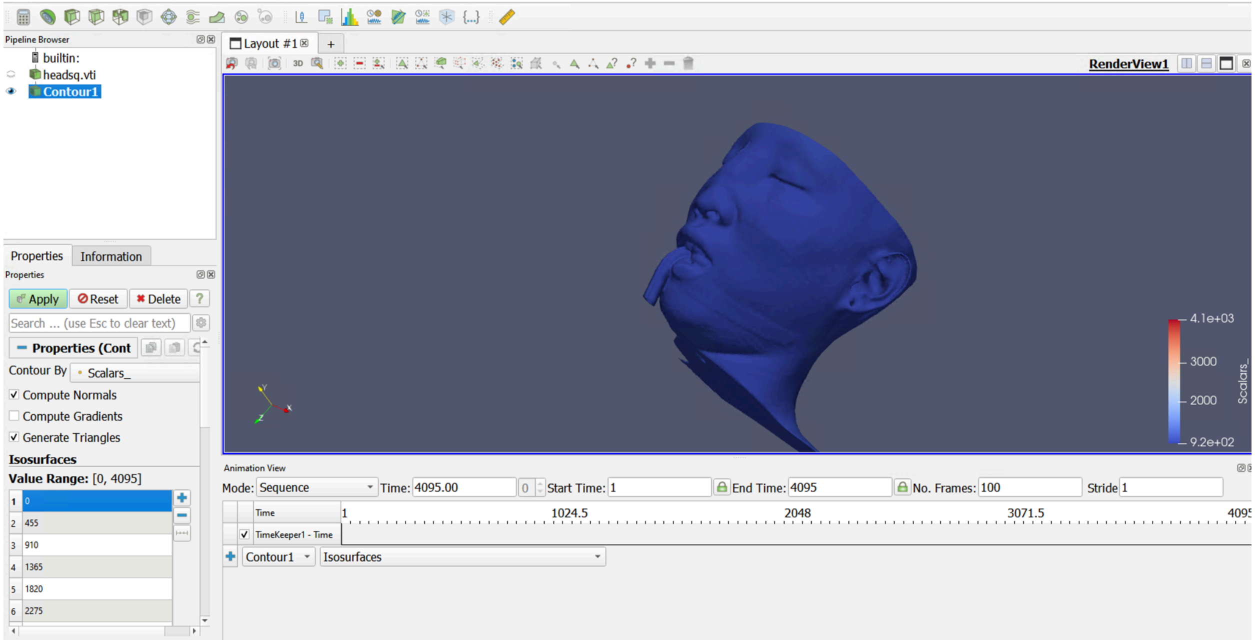

Figure 8: Value to extract face

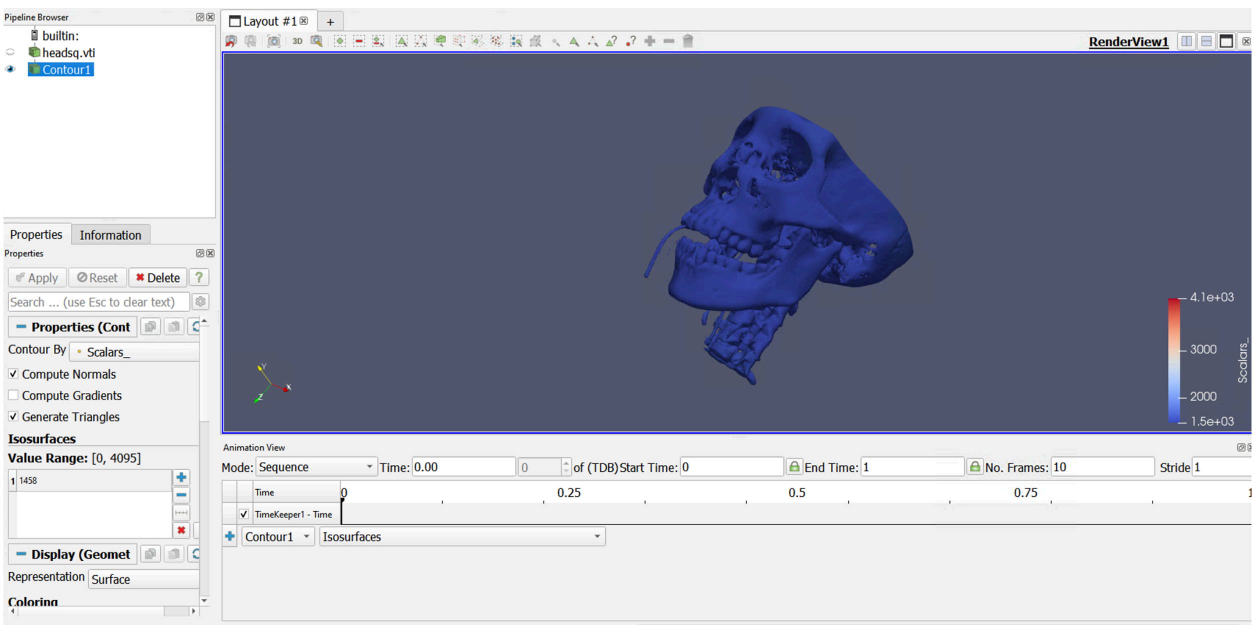

Figure 9: The value to extract Skull

The value when the spine vanishes is 2048 (Figure 10)

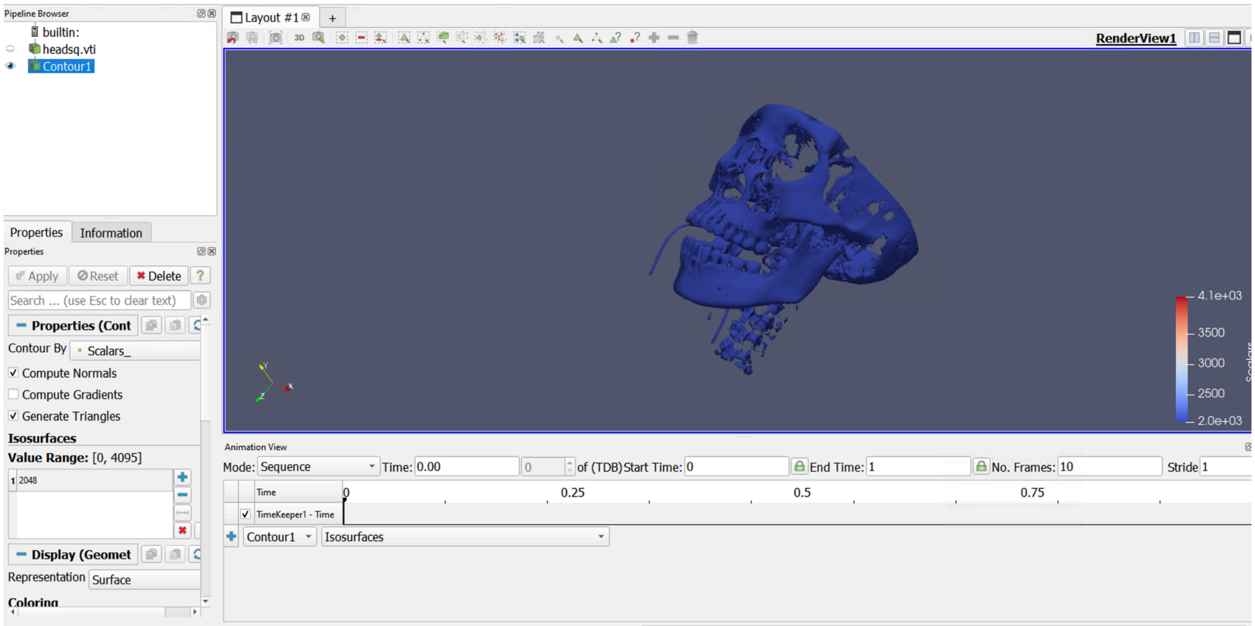

Figure 10: The value to vanish skull

4.2

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

Answer:

Scalar, vector, or tensor data that changes with time is time-varying data. Sometimes, it is represented as physical phenomena that vary constantly in space and time. Medical imaging, weather patterns, fluid dynamics, and financial market trends are a few examples of time-varying data.

The challenges we face when extracting isosurfaces from time-varying data are:

1. Temporal resolution:
The accuracy of the isosurfaces may be impacted by the data's time resolution. The isosurfaces could appear to skip between frames or be discontinuous if the temporal resolution is too low. On the other hand, if the time resolution is set too high, it may be too expensive to perform the computation necessary to extract the isosurfaces.
2. Temporal coherence:
It might be challenging to keep the isosurfaces' temporal coherence when dealing with time-varying data that contains noise or other irregularities. This may cause isosurfaces to disintegrate or disappear over time.
3. Topology changes:
The isosurfaces' topology might alter over time, making it challenging to follow the isosurfaces' development.
4. Computational complexity:

Extracting isosurfaces from time-varying data can be computationally inefficient when working with huge datasets. When real-time visualization is required, this can be difficult.

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

Answer:
Isosurfaces are frequently extracted from time-varying data using a particular form of data structure called temporal hierarchical index trees. These structures are required due to the computational difficulty of handling huge and intricate time-varying information.
It takes a lot of data searching to find the values that correspond to the desired isosurface in time-varying data, which is required for isosurface extraction. When the data is updated in real-time, this can be both time- and computationally expensive. This procedure can be sped up using temporal hierarchical index trees, which arrange the data into a tree structure that makes searching for and retrieving isosurface values easier. Each node in these trees represents a sub-region of the data, dividing the data into more manageable, more compact sub-regions. The amount of data that has to be searched can be decreased by quickly identifying the nodes that are most likely to have the necessary isosurface values using the tree structure.

In conclusion, it is crucial to employ temporal hierarchical index trees for isosurface extraction in time-varying data since it offers a technique to speed up the computation by lowering the quantity of data that has to be processed and enabling more effective and real-time isosurface extraction.

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

Answer:
The isosurfaces extraction algorithm in time-varying data is:
1. The time-varying data are first organized into a temporal hierarchical index tree by breaking up the time-varying data into smaller sub-regions and arranging them into a tree structure.
2. Then, the algorithm searches the tree structure for the nodes that are most likely to hold the desired values given the desired isosurface value. Starting at the tree's root and working our way down to the leaf nodes, we check to see if the isosurface value is contained within each node's sub-region.
3. After that, the approach uses interpolation to estimate the real isosurface values once the nodes that contain the desired isosurface value have been identified.
4. Finally, the isosurface is extracted by joining the estimated values to create a surface.

5. Marching Squares in Python ([code:https://github.com/elizaan/Viz-Scientific-Data-HWs/tree/main/hw2](https://github.com/elizaan/Viz-Scientific-Data-HWs/tree/main/hw2))

5a.

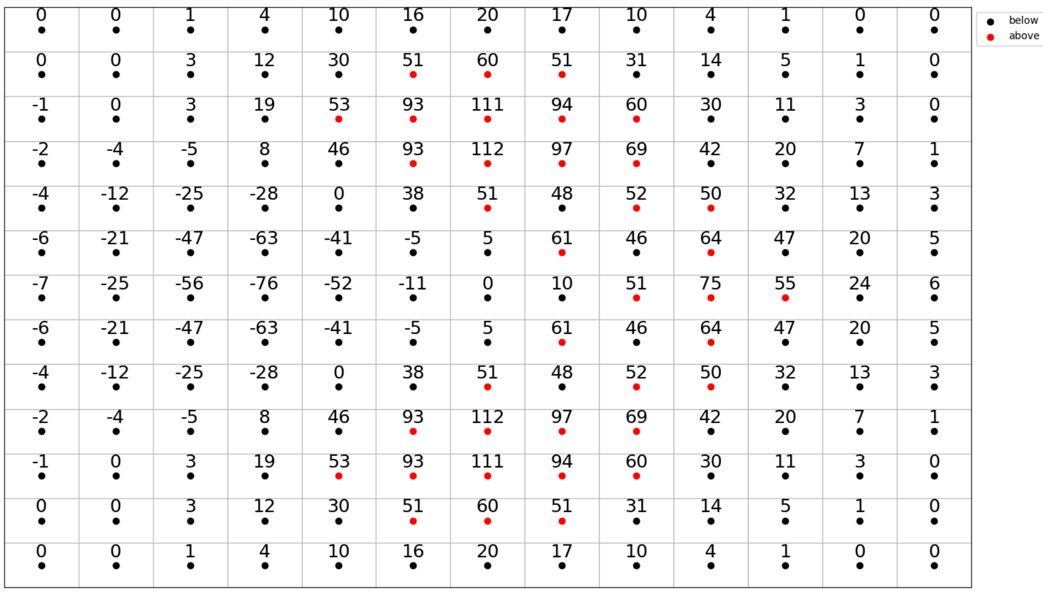

Figure 11: The labels based on iosovalues

5b.

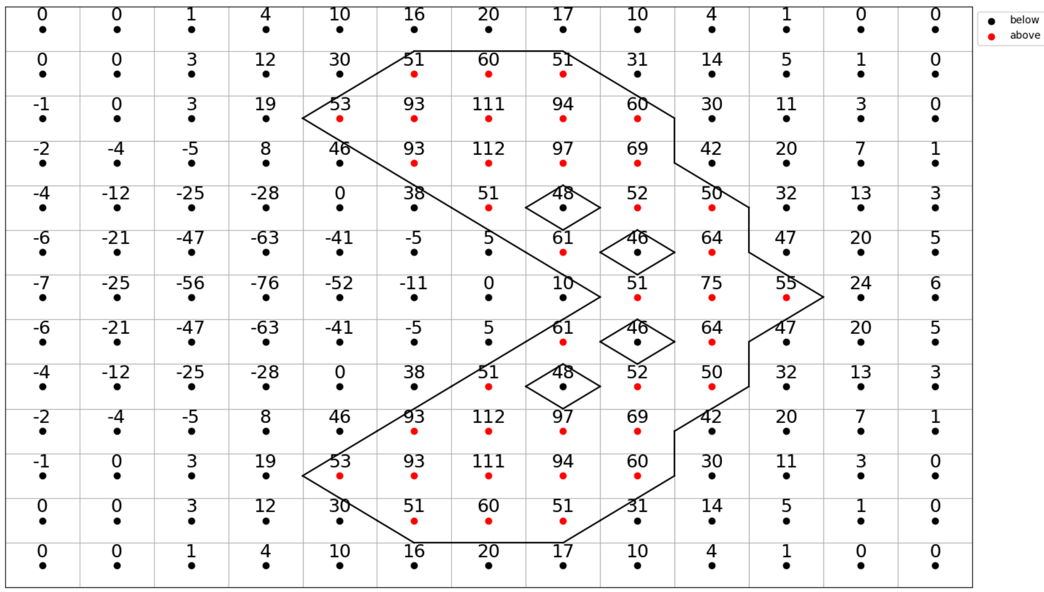

Figure 12: The Naive middle point algorithm

5c.

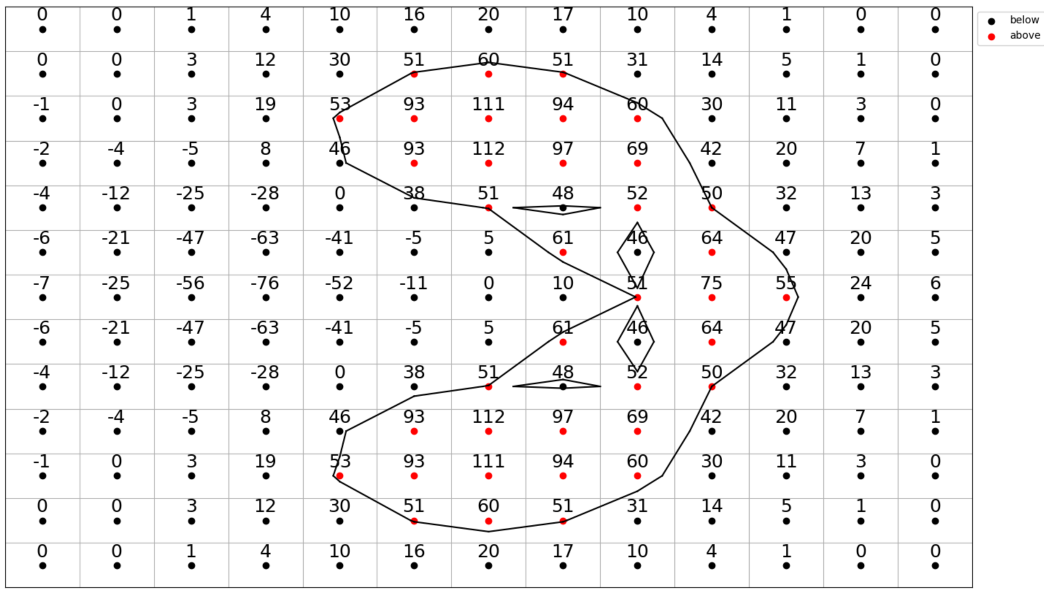

Figure 12: Marching square using linear interpolation

Conclusion:

The animation view doesn’t work in 5.12 or 5.13 paraview. Had to install 5.10