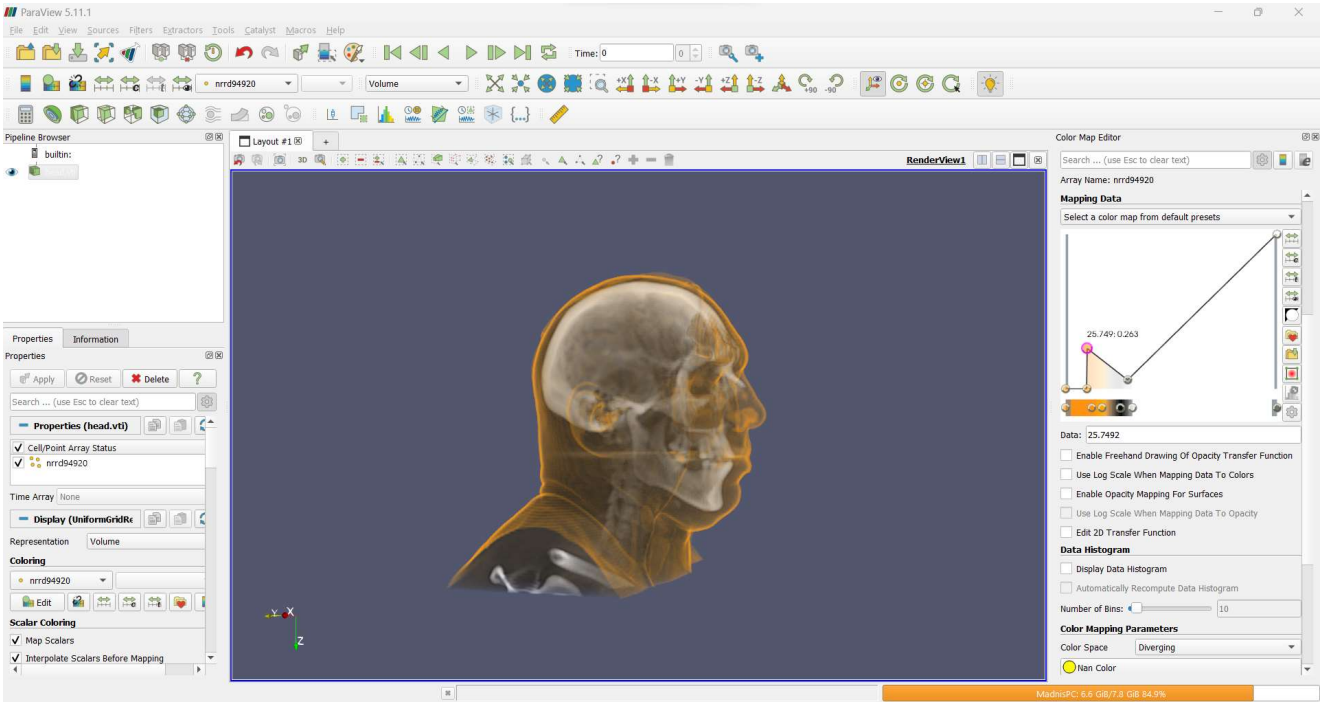
# Assignment 10

## Group Members

Abdul Wahab Madni, Hamza Badar, Aleksandr Semenikhin

## Ex.1: Direct Volume Rendering in ParaView

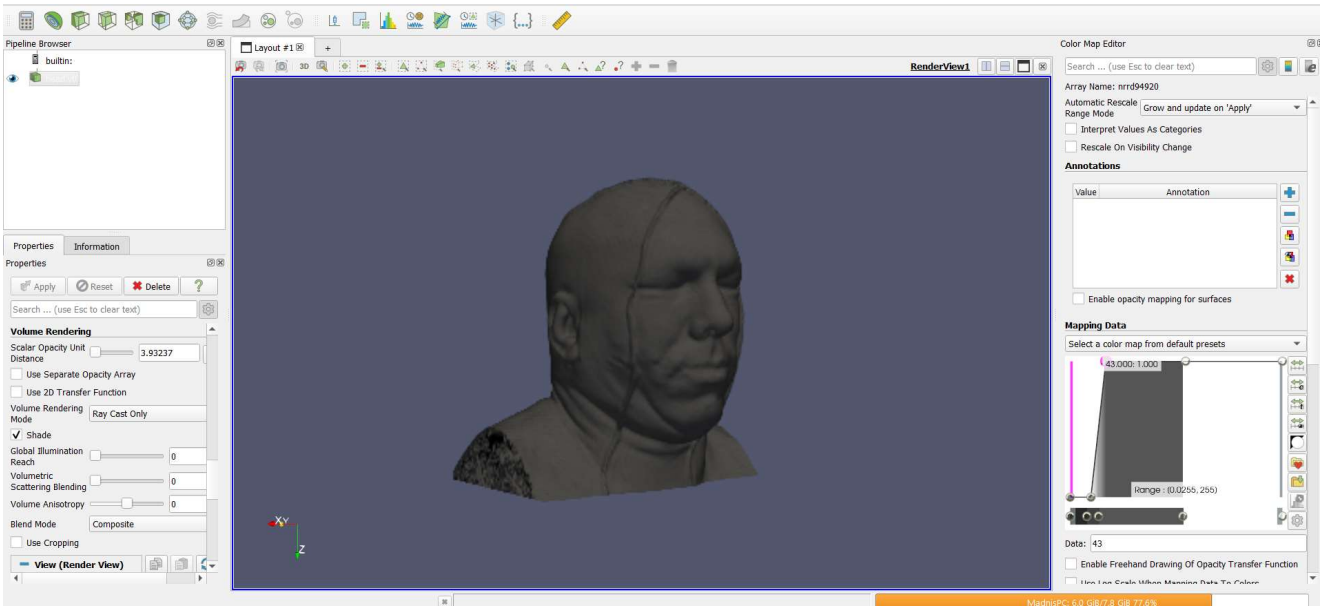
**a)**




**b)**

The intended effect of such effect is most likely to see something regarding bones or skeleton. It doesn't lead to a cleaner image because of the insufficient samples. Post-processing filters can enhance the final rendered image. Techniques like Gaussian blurring or edge-preserving filters can be applied to the rendered image to smooth out any remaining noise or artifacts, resulting in a cleaner and more polished result.

**c)**




***Shading and ray casting were missing in case 1.***

**d)**

When the image is still, the image is constant and hence the gpu have enough time to render all the steps. But when we are rotating the image, the computational data becomes very large because the angle is changing. If the gpu has to render all the the steps now then it'll take a long time to render anything proper while rotating. To solve this problem, the larger stepsize in raycasting are used. It reduces the computational effort hence giving a comprehensible image but in result produces the artifacts.

### Ex.2: Rendering Speed of Direct Volume Rendering in VTK

**a)**

Upon rotating the volume and observing the results, it was found that rendering speed increases after zoom out rotation and rendering speed decreases when rotating after zooming in.

The fps values observed on zoomed in rotation are:

fps=10.165

fps=2.866

fps=13.072

fps=14.217

fps=7.484

fps=11.859

fps=13.777

fps=0.740

fps=12.210

fps=13.939

After zooming in a little more it dropped to:

fps=1.470

fps=5.346

fps=3.965

fps=3.662

fps=2.772

fps=0.828

fps=3.959

fps=6.887

fps=2.945

The fps values observed on zoomed out rotation are:

fps=56.893

fps=28.138

fps=40.914

fps=32.891

fps=0.761

fps=53.640

fps=60.022

fps=30.030

fps=11.976

fps=60.099

**b)**

After adding another transfer function that only assigns non-zero opacity in a narrower value range for a cleaner rendering and after achieving cleaner rendering. The rendering speed seemed to be increased and rate of frames per second increased. ray casting algorithm used by 'vtkFixedPointVolumeRayCastMapper' performs early termination for regions with zero opacity. And by assigning zero opacity to a larger portion of the volume data, the modified transfer function reduces the computational workload during the rendering. The mapper can skip the calculations for the voxels that fall within the zero-opacity range, resulting in faster rendering.

**c)**

Trilinear interpolation involves computing weighted averages of neighbouring voxels, resulting in smoother, appealing with reduced pixelation but more computationally expensive rendering as fps observed in this interpolation were low in comparison with Nearest neighbour interpolation.

Nearest neighbour interpolation, on the other hand, selects the value of the nearest voxel without interpolation. Nearest neighbour interpolation is faster but generates more of a pixelated image renderings. Unlike Trilinear interpolation fps observed in this were comparatively high. This interpolation will be best suitable for large datasets and low end hardware.

We can get smoother and less pixelated renderings while maintaining the rendering speed, if we apply additional transfer function that only assigns non-zero opacity in a narrower value range with Trilinear interpolation.

**d)**

After enabling "AutoAdjustSampleDistances" property VTK started to automatically adjust the sampling distances based on the data and rendering parameters. VTK dynamically adjusts the sample distances based on the current view settings, such as the camera position, zoom level. Upon performing the previous experiments after enabling "AutoAdjustSampleDistances", the rendering quality and speed both were improved compared to when it was disable.

Moreover, fps rate didn't show much larger difference in both cases when we rotate volume while zoomed-in and zoomed-out