MM804 GRAPHICS AND ANIMATION Assignment 3 Solution

Question 1:

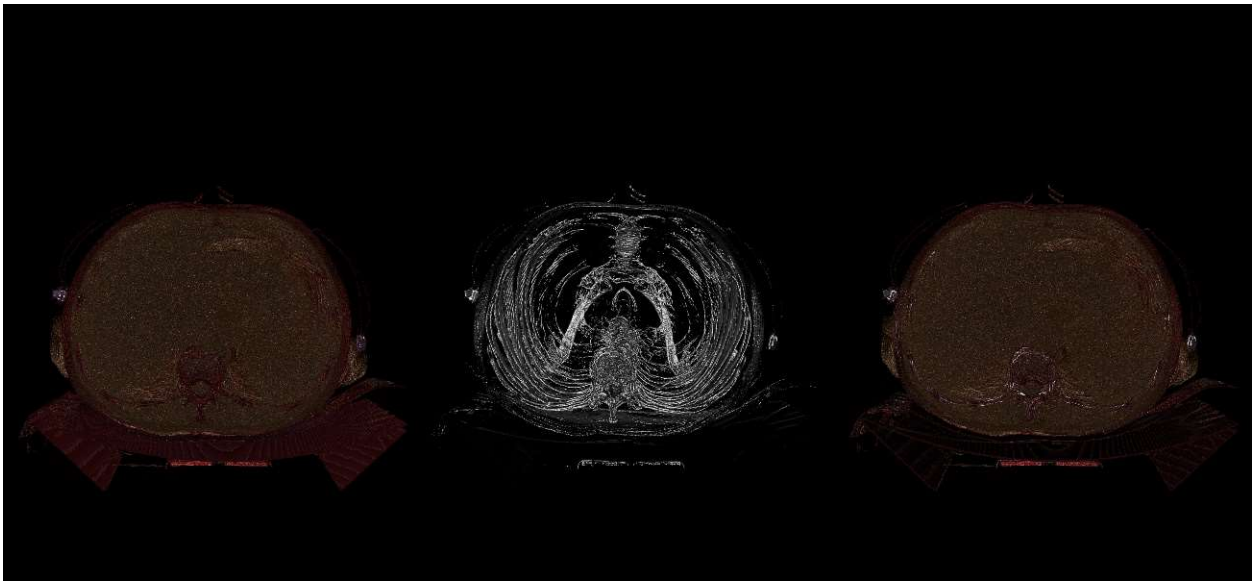
The medical imaging dataset used for volume rending the model is

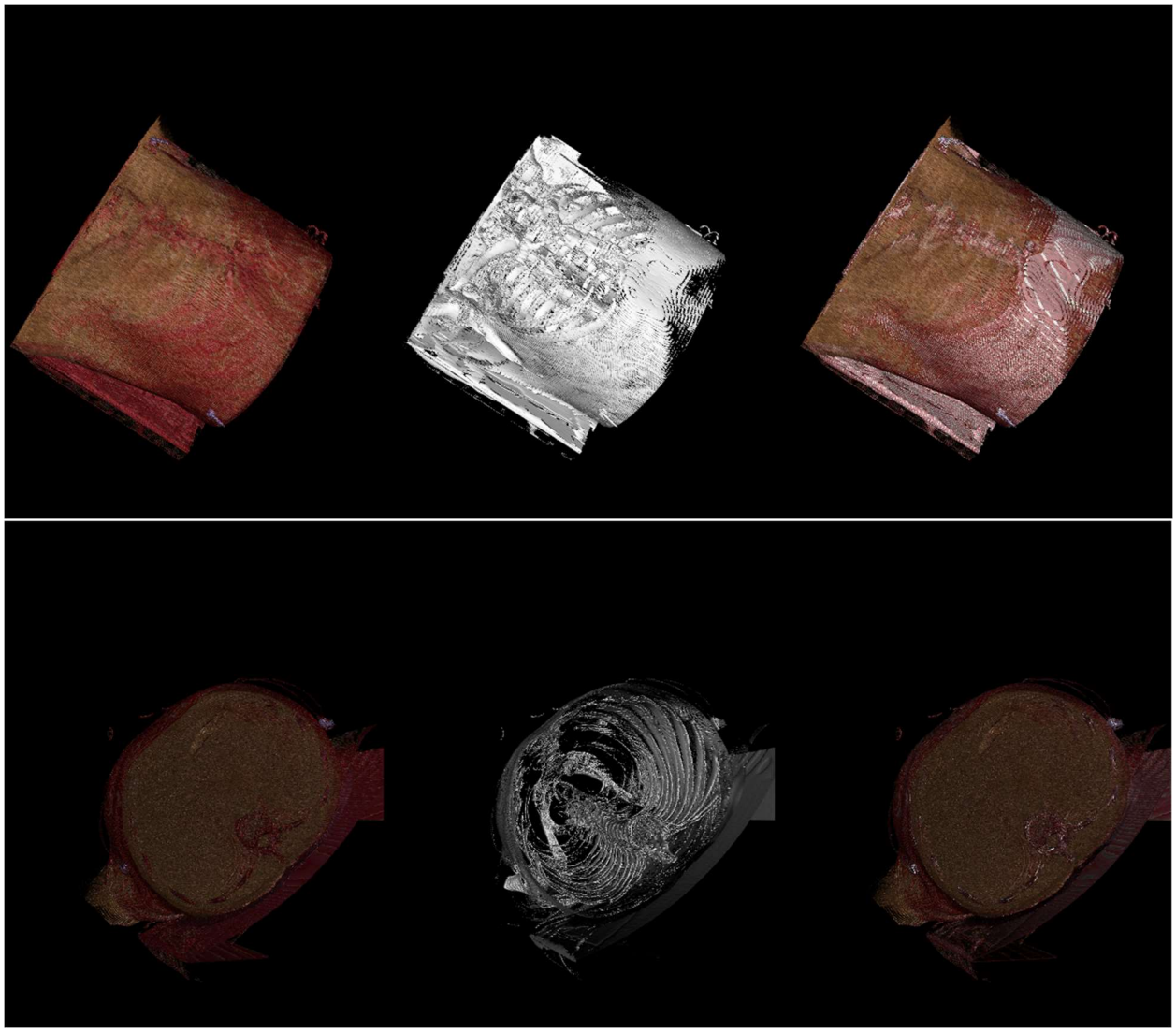
https://nbia.cancerimagingarchive.net/nbia-search/?MinNumberOfStudiesCriteria=1&CollectionCriteria=Lung-PET-CT-Dx

- The size of the dataset is 32 MB which contains a total of 64 images that are used for volume rendering of the medical images.
- Dimensions of the images are 512*512 and voxel resolution is 5mm.
- Min and Max pixel Intensities are 0 and 512.
- Size of the individual image DICOM file is 515kb

Question 2:

Multiple views of the Output Image





Question 3:

Source Code with comments in Human Readable Format

```
#import vtk library
import vtk

# 1. Reading the Lungs dataset using vtkDICOMImageReader class.
inputDataReader = vtk.vtkDICOMImageReader()
inputDataReader.SetDirectoryName("Lungs")
inputDataReader.Update()

# 2. Creating a random a colour transfer function using the following values.
colorTransferFunction = vtk.vtkColorTransferFunction()
colorTransferFunction.AddRGBPoint(-3024, 0.0, 0.0, 0.0)
colorTransferFunction.AddRGBPoint(-77, 0.5, 0.2, 0.1)
```

```
colorTransferFunction.AddRGBPoint(94, 0.9, 0.6, 0.3)
colorTransferFunction.AddRGBPoint(179, 1.0, 0.9, 0.9)
colorTransferFunction.AddRGBPoint(260, 0.6, 0.0, 0.0)
colorTransferFunction.AddRGBPoint(3071, 0.8, 0.7, 1.0)

# 3. Create a opacity transfer function using the following values.
opacityTransferFunction = vtk.vtkPiecewiseFunction()
opacityTransferFunction.AddPoint(-3024, 0.0)
opacityTransferFunction.AddPoint(-77, 0.0)
opacityTransferFunction.AddPoint(180, 0.2)
opacityTransferFunction.AddPoint(260, 0.4)
opacityTransferFunction.AddPoint(3071, 0.8)

# 4. Create viewports and render the dataset in multiple view ports
# Using Volume rendering for Viewport 1
ctVolumeMapper = vtk.vtkSmartVolumeMapper()
ctVolumeMapper.SetInputConnection(inputDataReader.GetOutputPort())

# Add the opacity and colour transfer functions
ctVolumeProperty = vtk.vtkVolumeProperty()
ctVolumeProperty.SetScalarOpacity(opacityTransferFunction)
ctVolumeProperty.SetColor(colorTransferFunction)
ctVolumeProperty.ShadeOn()

# Define a volume actor
ctVolume = vtk.vtkVolume()
volumeRenderer = vtk.vtkRenderer()

# Set volume actor properties
ctVolume.SetMapper(ctVolumeMapper)
ctVolume.SetProperty(ctVolumeProperty)

volumeRenderer.AddVolume(ctVolume)

# 5. In viewport 2, display the iso-surface extracted at suitable intensity value
of 300 using marching cubes algorithm
isoSurface = vtk.vtkMarchingCubes()
isoSurface.SetInputConnection(inputDataReader.GetOutputPort())
isoSurface.ComputeGradientsOn()
isoSurface.ComputeScalarsOff()
# print (isoSurface.GetValue(0))
isoSurface.SetValue(0, 300)

# Polydata mapper for the iso-surface
isoMapper = vtk.vtkPolyDataMapper()
```

```
isoMapper.SetInputConnection(isoSurface.GetOutputPort())
isoMapper.ScalarVisibilityOff()

# Actor for the iso surface
isoActor = vtk.vtkActor()
isoActor.SetMapper(isoMapper)
isoActor.GetProperty().SetColor(1.,1.,1.)

## renderer and render window
isoSurfaceRenderer = vtk.vtkRenderer()
## add the actors to the renderer
isoSurfaceRenderer.AddActor(isoActor)

# 6. Create a Volume Rendering + Iso Surface for View Port3
combineRenderer = vtk.vtkRenderer()

# Reuse actor and volume
combineRenderer.AddActor(isoActor)
combineRenderer.AddVolume(ctVolume)

#Rendering window
# Create a render window with three viewports
minX=[0,0.33,0.66]
maxX=[0.33,0.66,1]
minY=[0,0,0]
maxY=[1,1,1]

mainWindow = vtk.vtkRenderWindow()
windInteract = vtk.vtkRenderWindowInteractor()
mainWindow.SetSize(1300,600)
windInteract.SetRenderWindow(mainWindow)

# SetActiveCameras to the ActiveCamera of the first renderer
# This allows the visualization to be viewed from same angel in all three
viewports
isoSurfaceRenderer.SetActiveCamera(volumeRenderer.GetActiveCamera());
combineRenderer.SetActiveCamera(isoSurfaceRenderer.GetActiveCamera());
volumeRenderer.ResetCamera()

# 7. Add the renderers to main window
mainWindow.AddRenderer(volumeRenderer)
mainWindow.AddRenderer(isoSurfaceRenderer)
mainWindow.AddRenderer(combineRenderer)

# Set the location for the 3 view ports
```

```
volumeRenderer.SetViewport(minX[0],minY[0],maxX[0],maxY[0])
isoSurfaceRenderer.SetViewport(minX[1],minY[1],maxX[1],maxY[1])
combineRenderer.SetViewport(minX[2],minY[2],maxX[2],maxY[2])

mainWindow.Render()

windowToImage = vtk.vtkWindowToImageFilter()
windowToImage.SetInput(mainWindow)
windowToImage.Update()
imageWriter = vtk.vtkJPEGWriter()
imageWriter.SetInputConnection(windowToImage.GetOutputPort())
imageWriter.SetFileName('output.jpg')
imageWriter.Write()

windInteract.Initialize()
windInteract.Start()
```

Question 4:

Readme File attached in the submission folder and can be found in the GitHub link

Development Environment

- Python - 3.10.0

- VTK - 9.1.0

- OS - Windows 11

How to run:

Open the file volumeRendering.py either in Google Colab or Jupyter-Notebook and run the file using the below command for the desired output.