Dicom/Tiff Image segmentation

Code is highlighted in green

Yellow highlight = Working on currently

Task: Tiff/Dicom files to 3D model

My current plan:

Tiff/Dicom (opener), Image Segmentation, Cropping the image segmentation, Save as Dicom/Tiff, Open it again, and use surface img as 3d model

**Tiff/Dicom(status): Done**

Opens Tiff/Dicom files and translates into Numpy array in 3D

**Image Segmentation(status):Done**

Code Explination: Using Moduels from Scipy, Numpy, and Skimage in order to segmentate image:

Starting image:

C:\Users\P633A\Documents\test1mri.tif

The imports:

import numpy as np

import matplotlib.pyplot as plt

from skimage.filters import sobel

from skimage import morphology

from scipy import ndimage

from skimage.color import label2rgb

ar = your array opened in numpy

elevation\_map = sobel(ar)

Elevationmap – Is the first step in the process, It finds the “elevation” of the pixals based on their intensity values

Markers

#get markers, start filling from where the extreme parts of the graph are,

markers = np.zeros\_like(ar)

markers[ar < 30] = 1

markers[ar > 150] = 2

Segmentation is done useing the watershed algorithm, Starting from the smallest intensity value, and fill the water to find the valleys and mountains

segmentation = morphology.watershed(elevation\_map, markers)

Label/colour

segmentation = ndimage.binary\_fill\_holes(segmentation - 1)

#fills the holes in the images using segmentation

labeled\_ar, \_ = ndimage.label(segmentation)

#label the coins, into a numpy array, with different values

image\_label\_overlay = label2rgb(labeled\_ar, image=ar)

#using labels to assign each a different colour

End result is:

C:\Users\P633A\Documents\test1mrif.tif

The one on the left is the segmentation without the colours, the one on the right is with the colours assigned

**Image Segmentation Part2:Cropping (status): Analyzing Code/Figureing out what to do:**

Progress:

Printing (label\_ar) yields Numbers from 0 to 58, could be useful, possible segmentation:

C:\Users\P633A\Documents\test1.tif1-59

C:\Users\P633A\Documents\test1.tif

C:\Users\P633A\Documents\test1.tif

Save the file as a Tiff or Dicom:

Dicom is preferable.

Saved as dicom and tiff, complete

**Surface Imaging(status):Dicom Complete**[**http://www.vtk.org/doc/nightly/html/classvtkImageReader.html**](http://www.vtk.org/doc/nightly/html/classvtkImageReader.html)

import vtk

import numpy

Opening file Tiff:

Currently Browsing VTK moduels in search of paths that can turn typical VTK tiff openers into volumes

Opening file Dicom:

PathDicom = "./vhm\_head/"

reader = vtk.vtkDICOMImageReader()

reader.SetDirectoryName(PathDicom)

reader.Update()

Opens the file and creates a 3D volume

Decompresser:

\_extent = reader.GetDataExtent()

ConstPixelDims = [\_extent[1]-\_extent[0]+1, \_extent[3]-\_extent[2]+1, \_extent[5]-\_extent[4]+1]

ConstPixelSpacing = reader.GetPixelSpacing()

Dicom uses a compressed file stage that needs to be decompressed before VTK can work with it.

Thresholding Based on Values:

threshold = vtk.vtkImageThreshold ()

threshold.SetInputConnection(reader.GetOutputPort())

threshold.ThresholdByLower(700) # remove all soft tissue

threshold.ReplaceInOn()

threshold.SetInValue(0) # set all values below 400 to 0

threshold.ReplaceOutOn()

threshold.SetOutValue(1) # set all values above 400 to 1

threshold.Update()

The pixel intensity varies and therefore you can threshold based on what is light vs what is darker

Surface Imageing:

dmc = vtk.vtkDiscreteMarchingCubes()

dmc.SetInputConnection(threshold.GetOutputPort())

dmc.GenerateValues(1, 1, 1)

dmc.Update()

Uses the matching cubes algorithm in order to create a 3D model

Writer:

writer = vtk.vtkSTLWriter()

writer.SetInputConnection(dmc.GetOutputPort())

writer.SetFileTypeToBinary()

writer.SetFileName("bones.stl")

writer.Write()

Writes the file into a STL file that can be further opened as a mesh in other programs:

**End of Project**