Shelly Gamlielly
20633689

Ex1 Medical Image Processing Report

Code can be found in colab notebook:
https://colab.research.google.com/drive/1k50W3d5Y6NwOE3viHB8T42Boow_OdVDM#scrollTo=2w9LLSN-tsjC

Part 1

Libraries : numpy for calculations , matplotlib for plotting, skimage (new) for morphological operations.

Solution's description : perform threshold segmentation of the skeleton in a contrast CT (CT using radiocontrast). For this purpose, I implemented two functions – SegmentationByTH, which performs the segmentation using its input Imin and Imax thresholds, and SkeletonTHFinder which is used to find the best suited thresholds.
I used numpy.where to find voxels between Imin and Imax for binary segmentation. I used remove_small_holes and remove_small_objects from skimage library to be left with a single connectivity component. The threshold or min_size that I used for those functions were calculated by counting black pixels and divide it by 2 * depth of image. Since at least half of the black pixels are part of the skeleton.

Functions description :

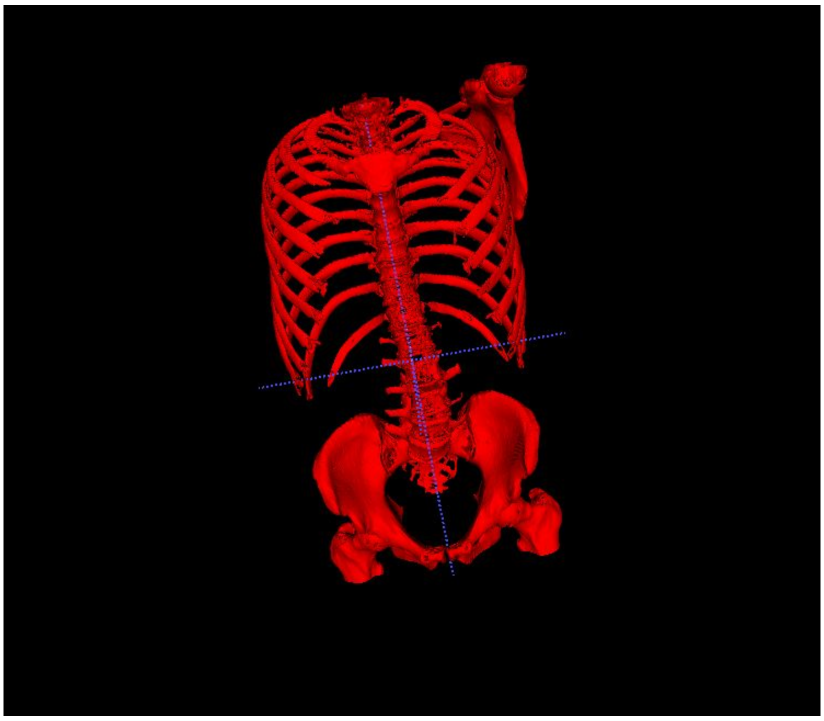
Name : SegmentationByTH_Helper(img,Imin, Imax=1300):
Explanation : this function is a helper for SegmentationByTH. It finds voxels between Imin and Imax and returns an array with elements with value 1 where the condition is True, and elements with value 0 elsewhere.
Input : image data - numpy array, and two integers – the minimal and maximal thresholds which is by default 1300.
Output : numpy array with 1 if condition is true and 0 else.

Name : SegmentationByTH
Explanation : the function generates a segmentation NIFTI file of the same dimensions, with a binary segmentation – 1 for voxels between Imin and Imax, 0 otherwise. This segmentation NIFTI file is saved under the name seg_.nii.gz. The function returns 1 if successful, 0 otherwise. And raise descriptive errors when returning 0.
Input : path to a grayscale NIFTI file, and two integers – the minimal and maximal thresholds which is by default 1300.
Output : a segmentation NIFTI file of the same dimensions, with a binary segmentation.

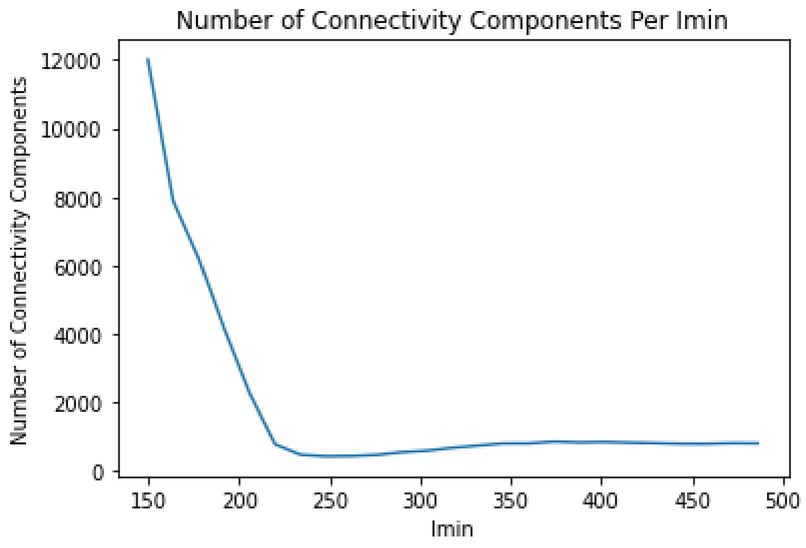
Name : post_processing
Explanation : the function performs post-processing (morphological operations – clean out single pixels, close holes, etc.) until we are left with a single connectivity component.
Input : data img - a numpy array
Output : a numpy array after removing small holes and objects.

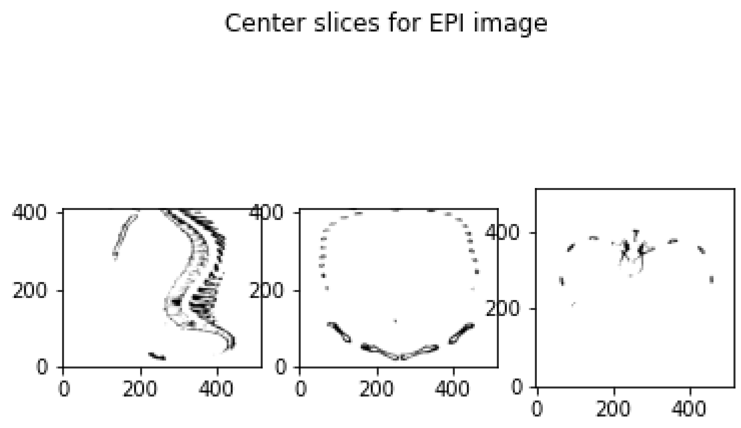
Name : SkeletonTHFinder
Explanation : This function iterates over 25 candidate Imin thresholds in the range of [150,500] (with intervals of 14). In each run, it uses the SegmentationByTH_Helper function, and counts the number of connectivity components in the resulting segmentation with the current Imin.It plots the number of connectivity components per Imin. This function finds the Imin which is the first minima in the plot. This function saves a segmentation NIFTI file called "_SkeletonSegmentation.nii.gz" and returns the Imin used for that.
Input : path to a grayscale NIFTI file
Output : Imin with the minimal connectivity components

CT Scan 1 :

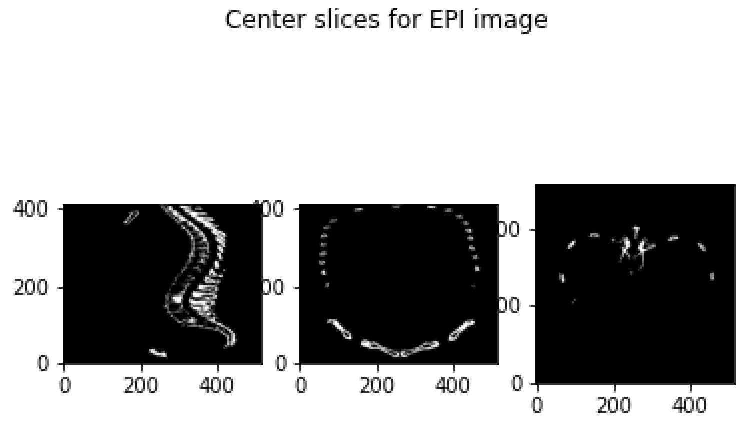


Imin : 248



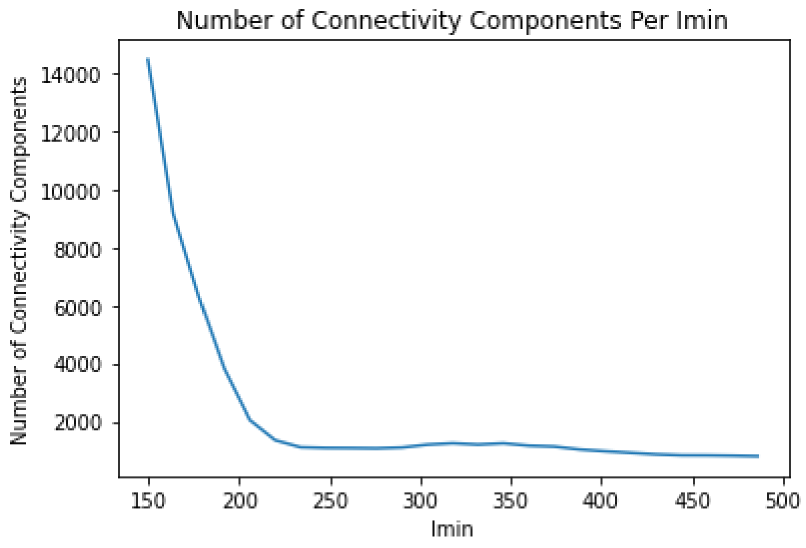


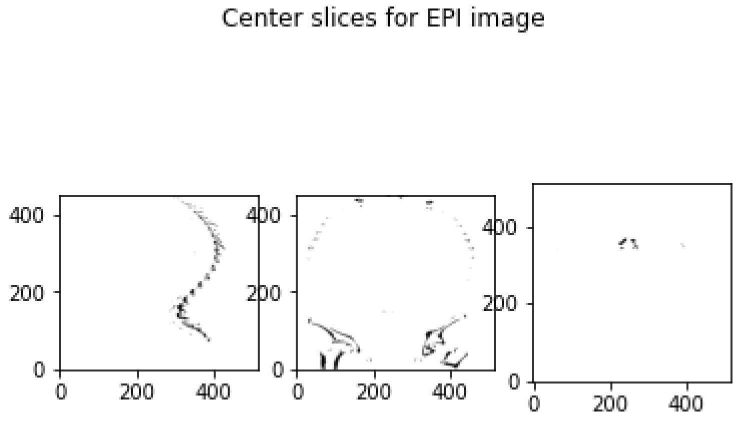
Before morphological operations:



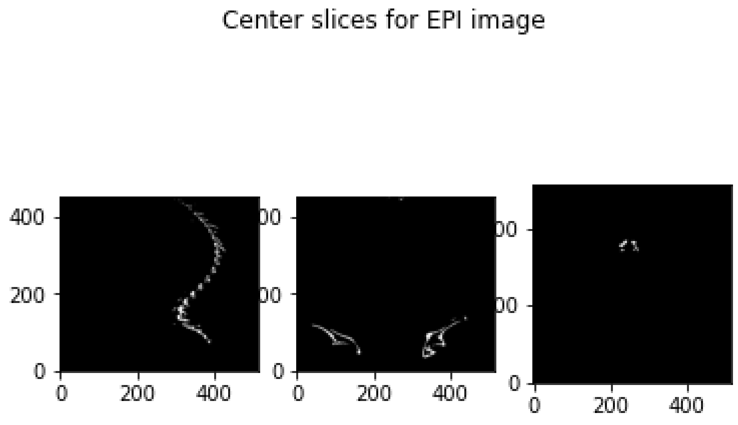
After morphological operations:

CT Scan 2 :
Imin: 486



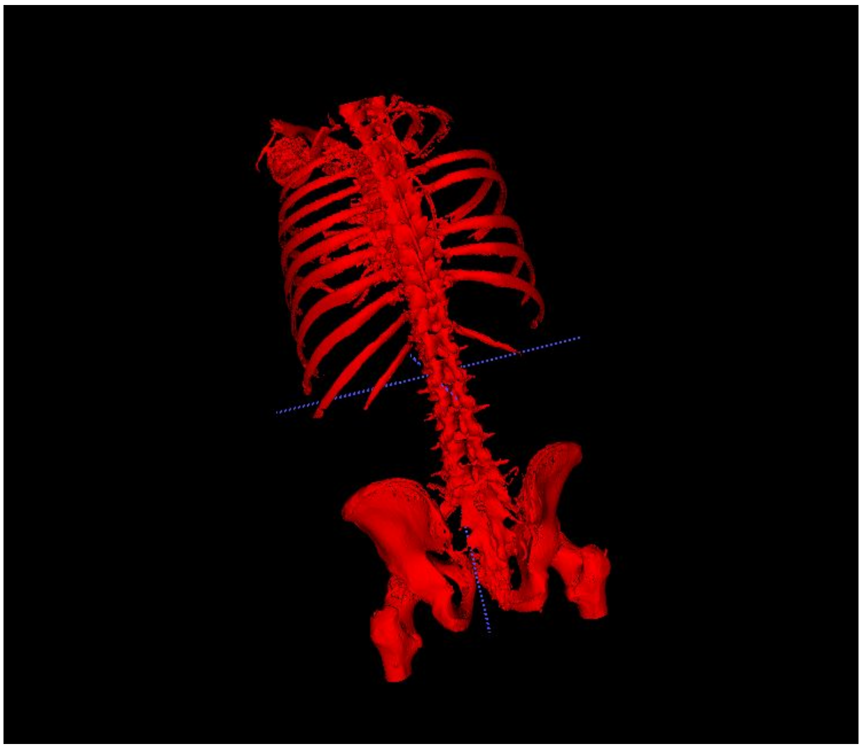


Before morphological operations:

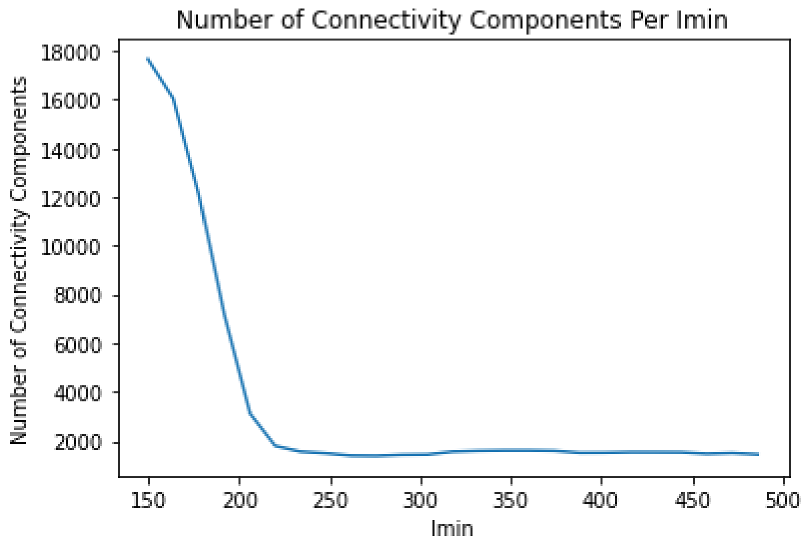


After morphological operations:

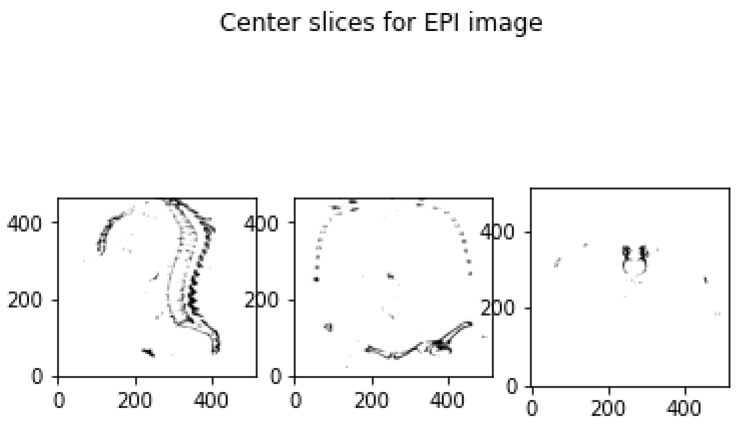
<u>CT Scan 3 :</u>

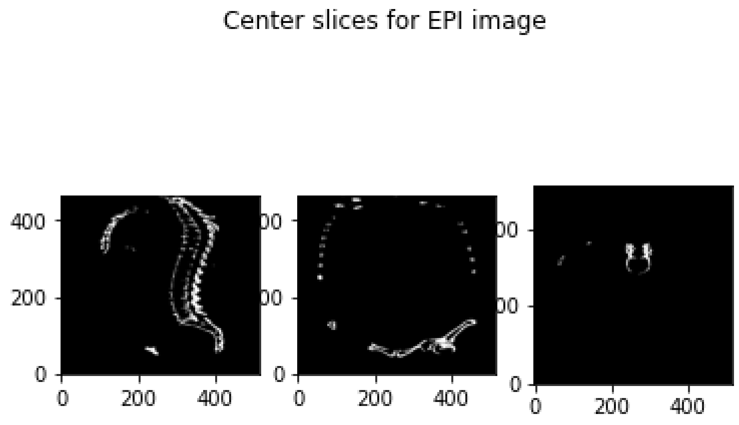


Imin: 276



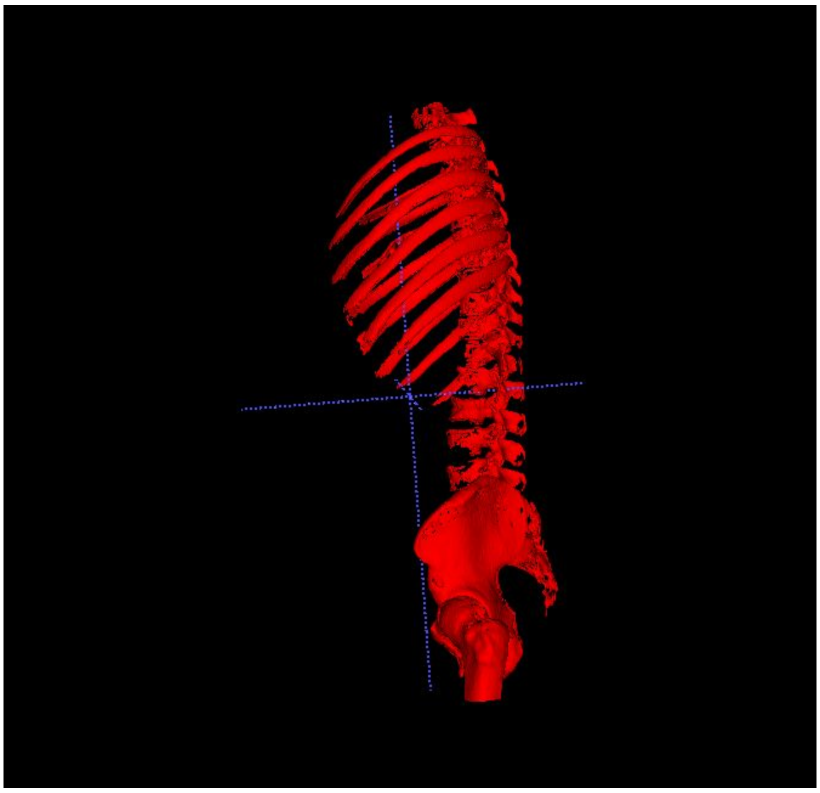
Before morphological operations:

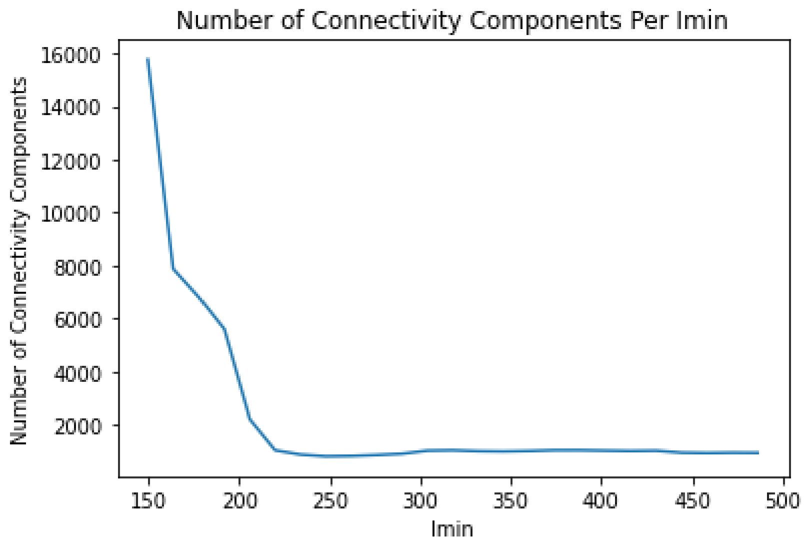




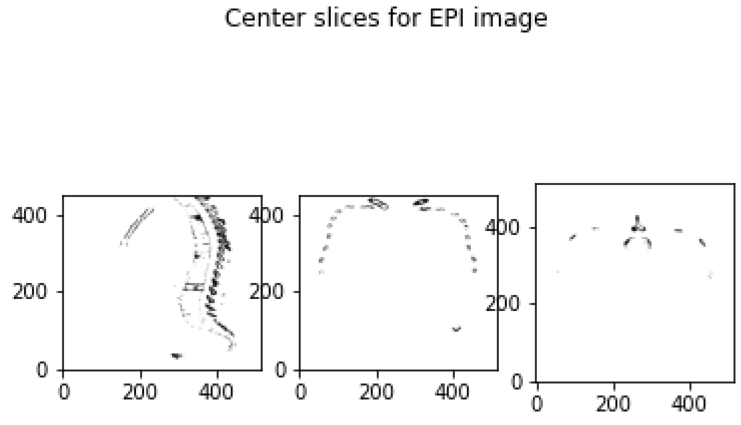
After: morphological operations

CT Scan 4 :
Imin = 248

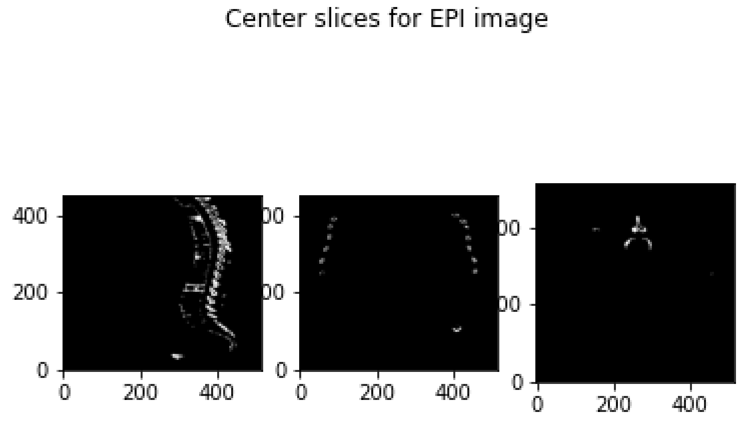




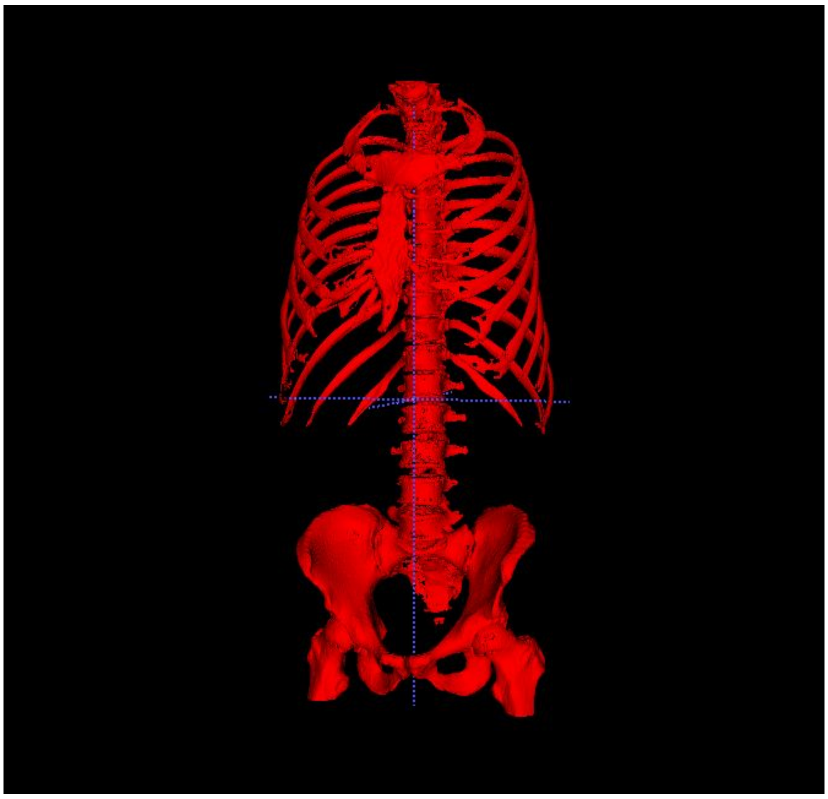
Before morphological operations:

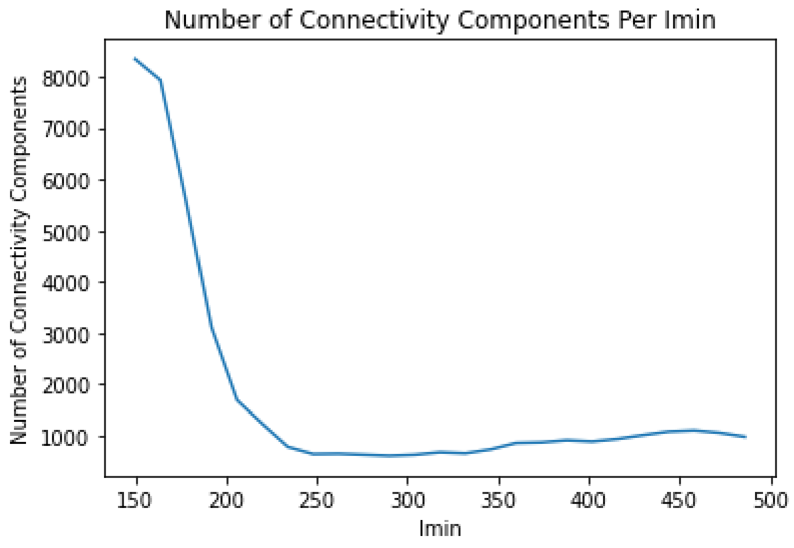


After morphological operations:

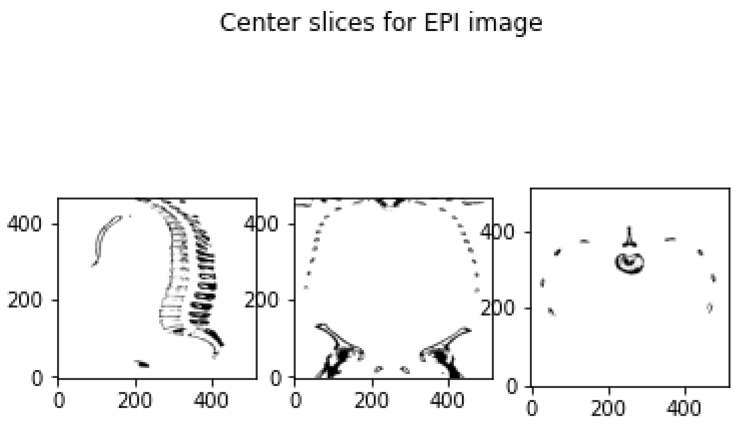


CT Scan 5 :
Imin = 290



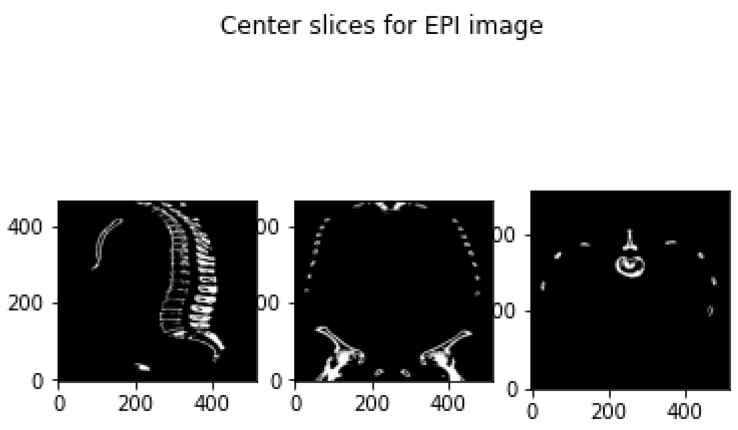


Number of Connectivity Components Per Imin
Number of Connectivity Components
8000
7000
6000
5000
4000
3000
2000
1000
150
200
250
300
350
400
450
500
Imin



Before morphological operations :

After morphological operations:



Part 2

Libraries : numpy for calculations , matplotlib for plotting, skimage (new) for morphological operations, find circles with hough_circle, and for finding edges with canny

Solution's description : perform segmentation of the aorta given ct image and L1 segmentation. L1 is a single vertebra, so using it's segmentation we could find the axial slices for aorta segmentation.
First. I found the region of interest by finding the indexes to run on the sagittal and coronal planes (the axial is defined by L1). I took only the indexes which are non zero on each plane, found the minimal and the maximal voxel on each plane, calculated the middle between max and min. Next, I changed the values until I got the region I was looking for, and cropped the input image according to the relevant indexes.
Second, I blurred the image in order to remove noise using the skimage gaussian.
Third, for each 2D axial slice, I found edges using skimage **canny**, and performed morphological operation using skimage's **dilation** and **erosion**. Afterwards I detected a circle using skimage's **hough_circle**, and selected the most prominent circle using **hough_circle_peaks**. Finally, by using skimage **draw.circle** I assigned value 1 to each pixel inside the circle, and zeros elsewhere. Evaluating results is done by calculating Dice and VOC on ROI for ground truth image and estimated image that we got by segmenting the aorta given L1.

Functions description :

Name : read_images
Explanation : this function read a CT scan image and a L1 segmentation image.
Input : path of nifty file and path of L1 segmentation nifti file
Output : ct scan image, image data of the ct scan , image data of L1 segmentation.

Name : find_RIO
Explanation : this function finds the region of interest in a CT scan image according to L1 segmentation. The axial plane is defined by L1 segmentation, and the sagittal and coronal indexes are calculated.
Input : image data of L1 segmentation.
Output : each plane‘s indexes of region of interest - min_x,max_x, min_y,max_y, min_z,max_z

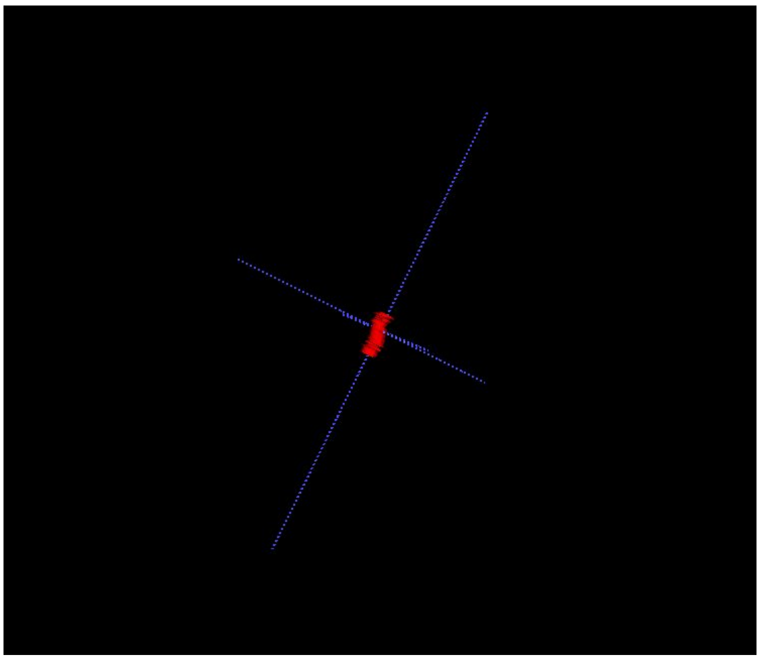
Name : create_box
Explanation : this function crops a CT scan image using L1 segmentation to decide on which axial slices segment the aorta.
Input : each plane‘s indexes of region of interest - min_x,max_x, min_y,max_y, min_z,max_z
Output CT image box to perform segmentation on.

Name : find_circles
Explanation : this function finds a circle on the cropped CT image and assigns 1 inside a circle and 0 elsewhere. It removes noise using the skimage gaussian. For each 2D axial slice, It detects edges using skimage’s canny, performs morphological operation using skimage’s dilation and erosion. A circle is detected using skimage’s hough_circle, and the most prominent circle is selected using hough_circle_peaks. Finally, by using skimage draw.circle it assigns value 1 to each pixel inside the circle, and zeros elsewhere.
Input : img_data the original CT image data, and the cropped blurred CT image.
Output numpy array with 1 inside circles and 0 outside.

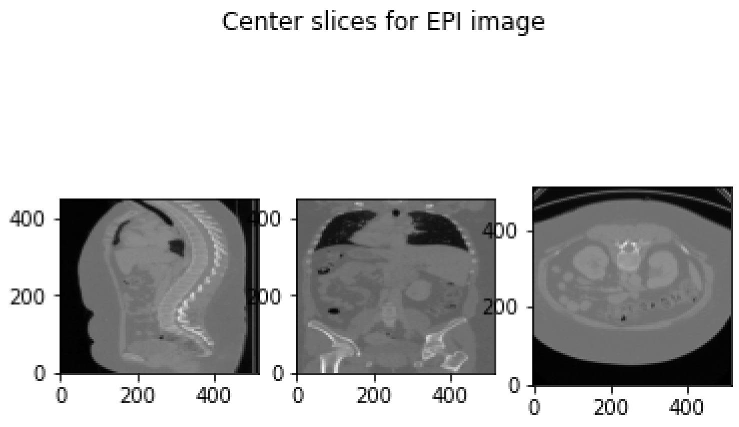
Name : AortaSegmentation
Explanation : This function is given a grayscale NIFTI file and an L1 segmentation NIFTI file. It uses the L1 segmentation to tell on which axial slices segment the aorta. It works on 2D axial slices to perform the segmentation, and finds a circle in each slice.
Input : path of nifty file and path of L1 segmentation nifti file
Output numpy array with 1 inside circles and 0 outside with the dimensions of the original CT image.

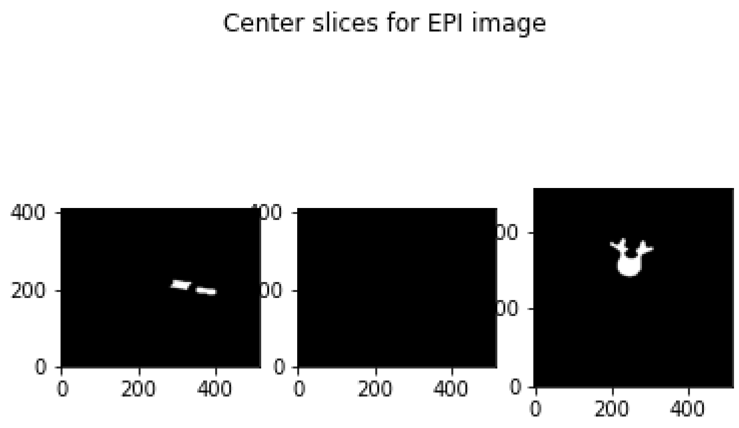
Functions outputs:

Case 1



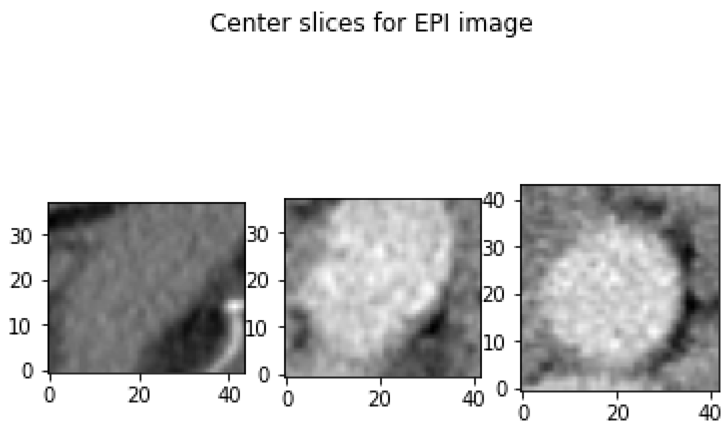
Input CT image :

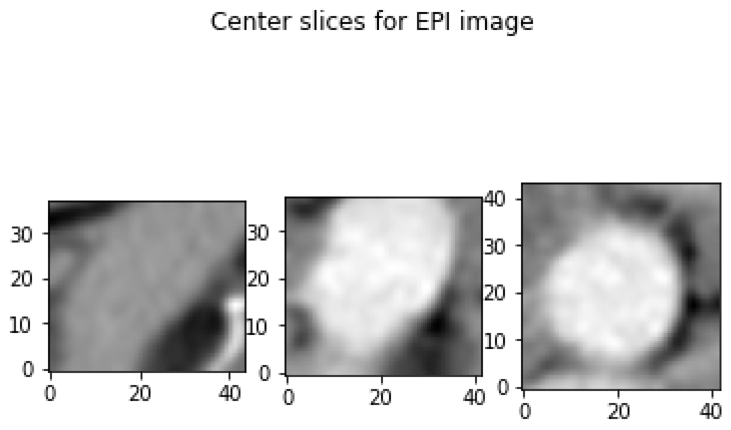




L1 :

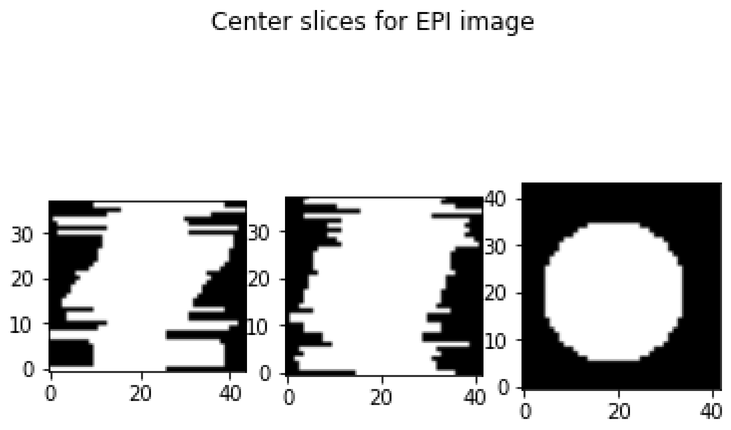
ROI:





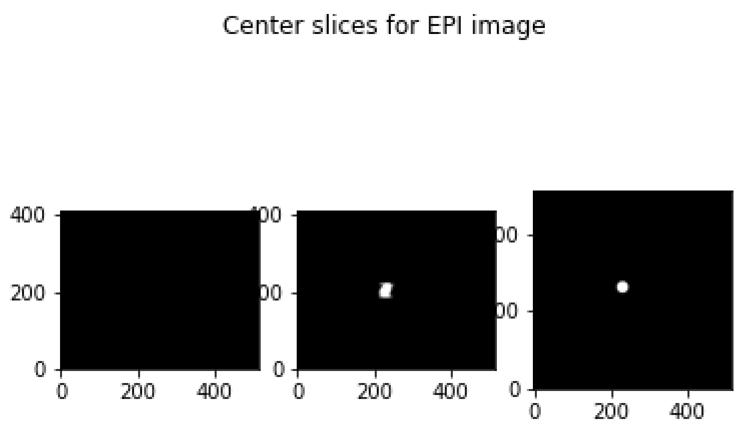
Blurred ROI:

Find edges using canny , Find circles using hough_circle and morphology operations (dilation, erosion)

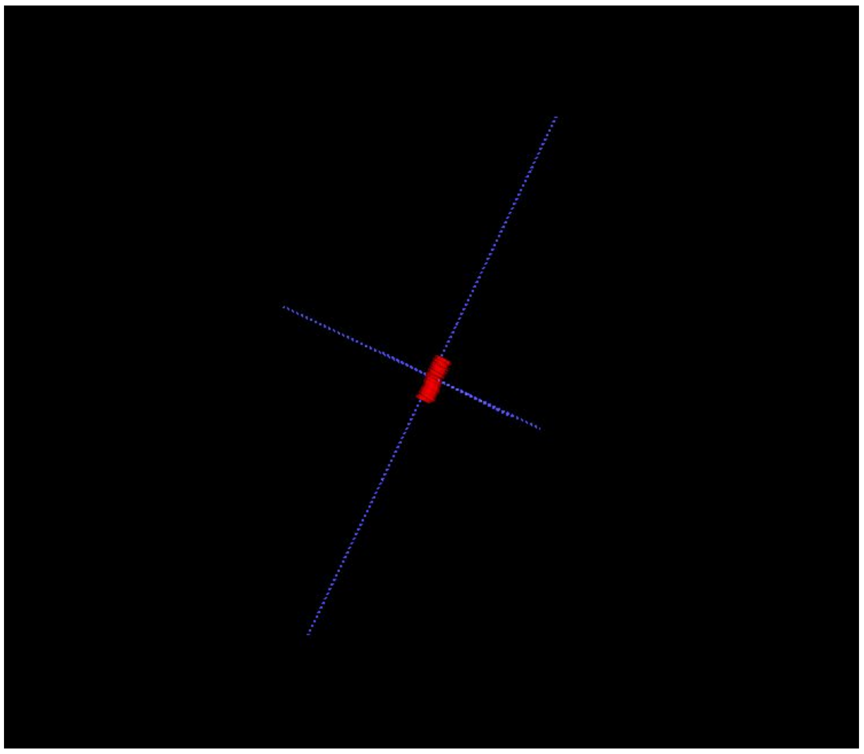


in blurred box:

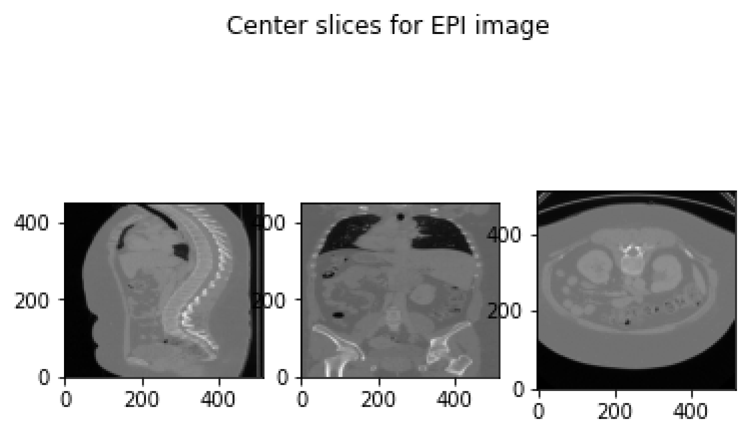
Output - in full image:



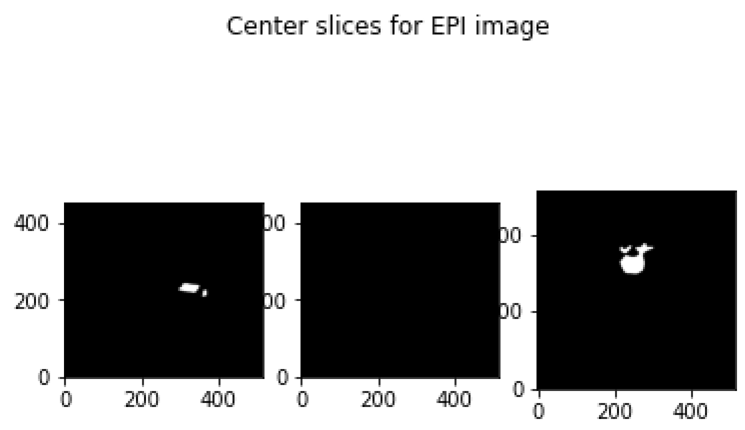
Case 2



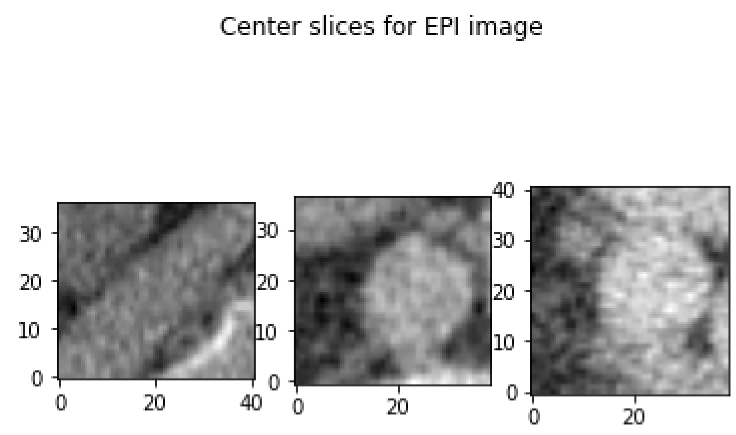
Input CT image :



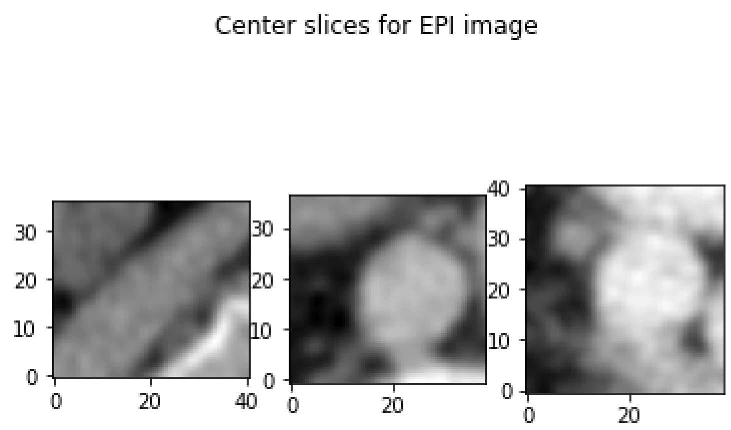
L1 :



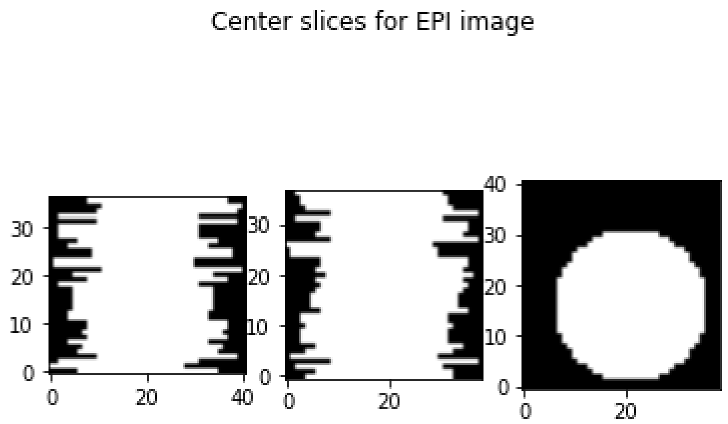
ROI:



Blurred ROI:

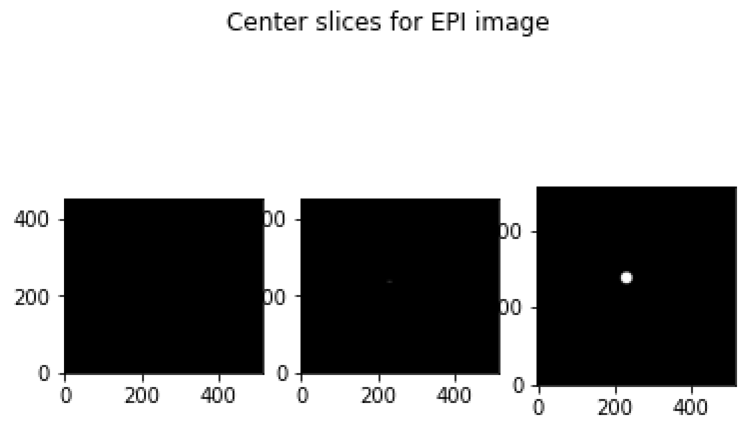


Find edges using canny , Find circles using hough_circle and morphology operations (dilation, erosion)

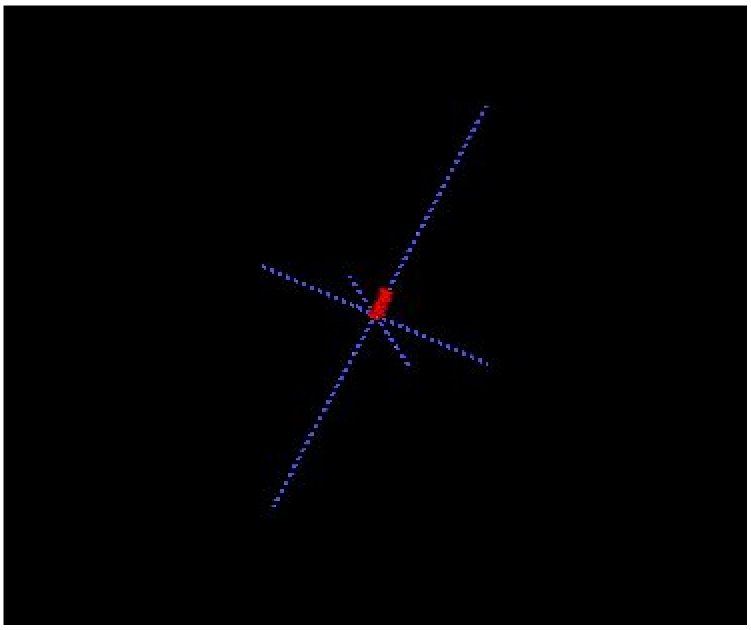


in blurred box:

Output - in full image:

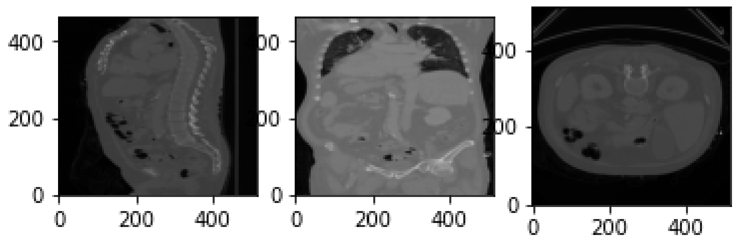


Case 3



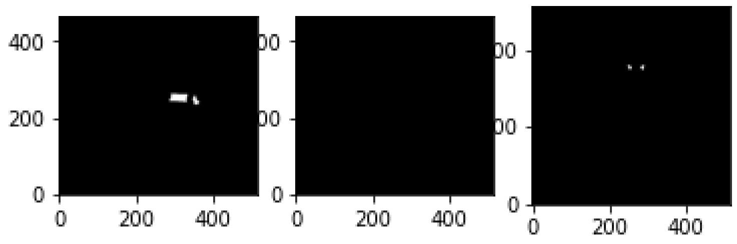
Center slices for EPI image

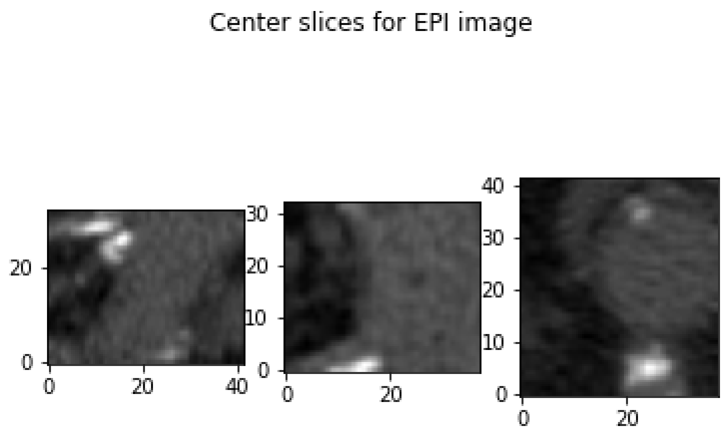
Input CT image :



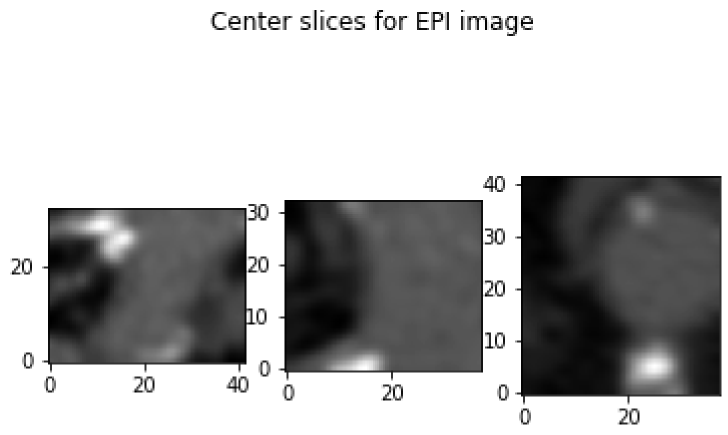
Center slices for EPI image

L1 :



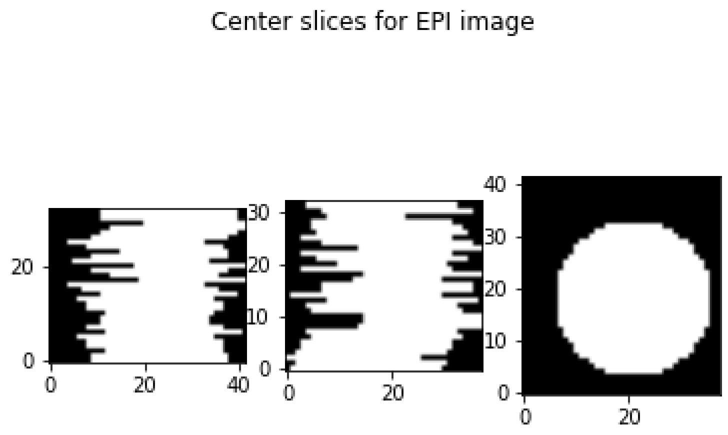


ROI:



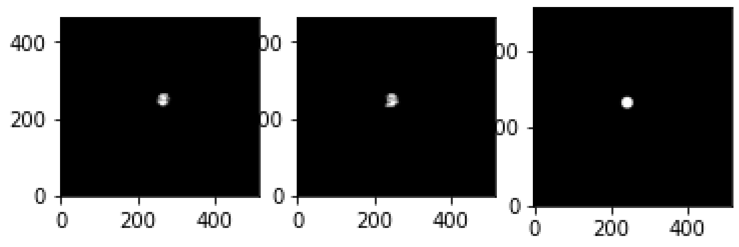
Blurred ROI:

Find edges using canny , Find circles using hough_circle and morphology operations (dilation, erosion)



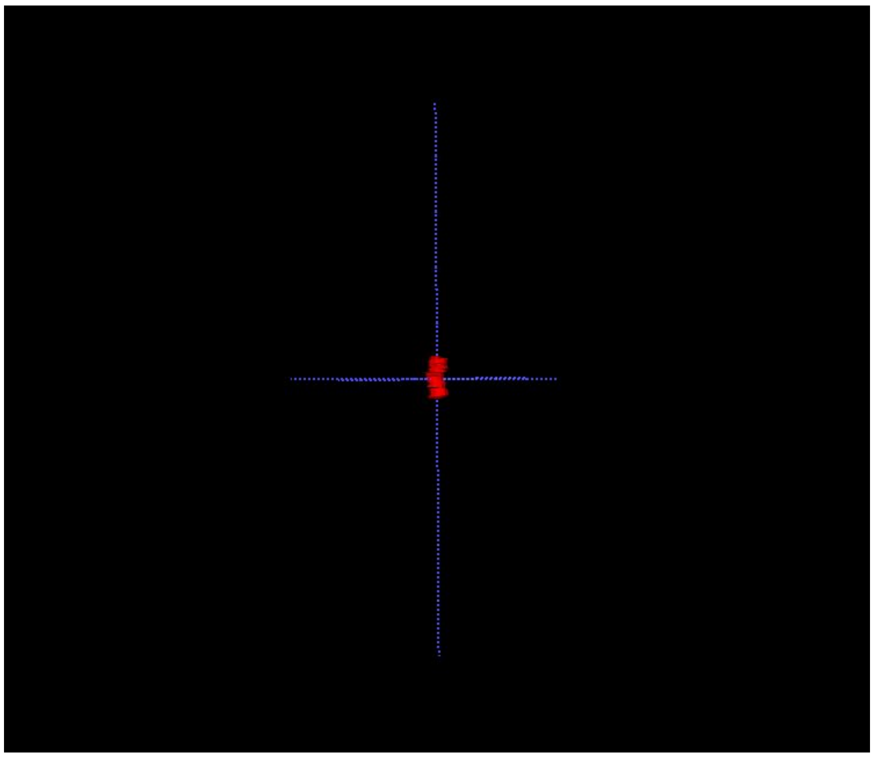
in blurred box:

Center slices for EPI image

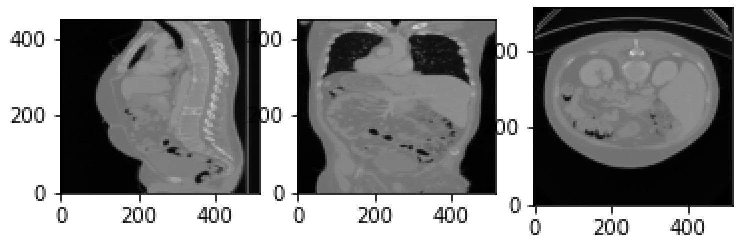


Output - in full image:

Case 4

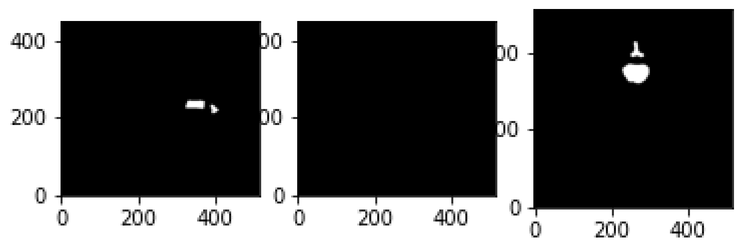


Center slices for EPI image



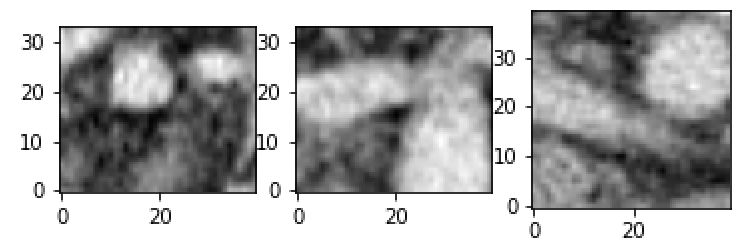
Input CT image :

Center slices for EPI image



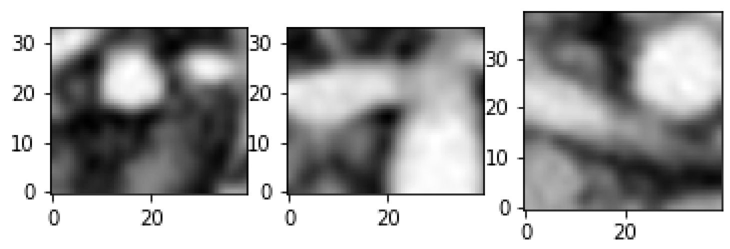
L1 :

Center slices for EPI image



ROI:

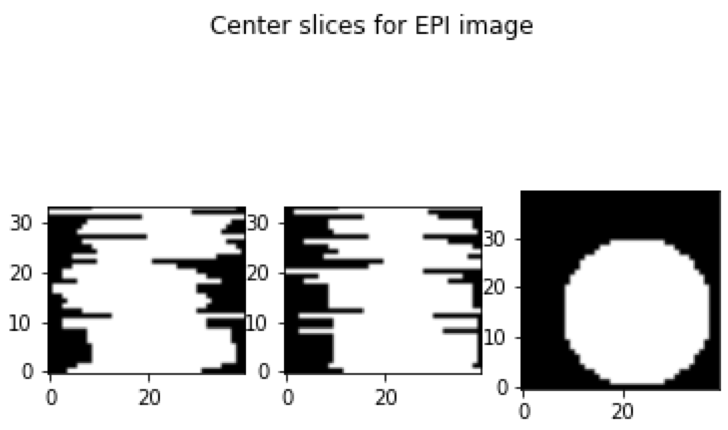
Center slices for EPI image



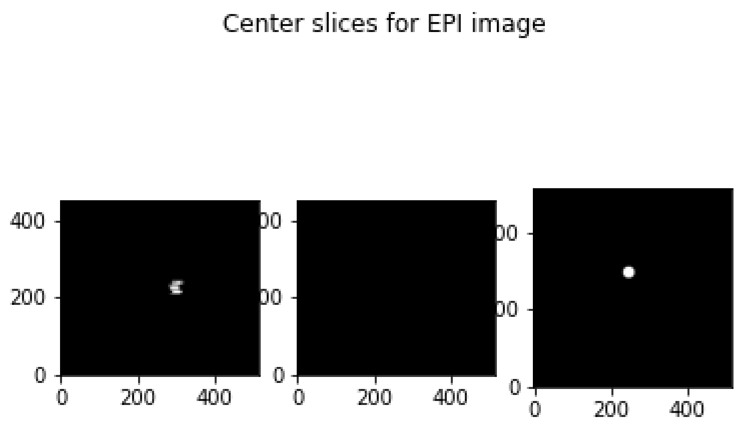
Blurred ROI:

Find edges using canny , Find circles using hough_circle and morphology operations (dilation, erosion)

in blurred box:



Output - in full image:



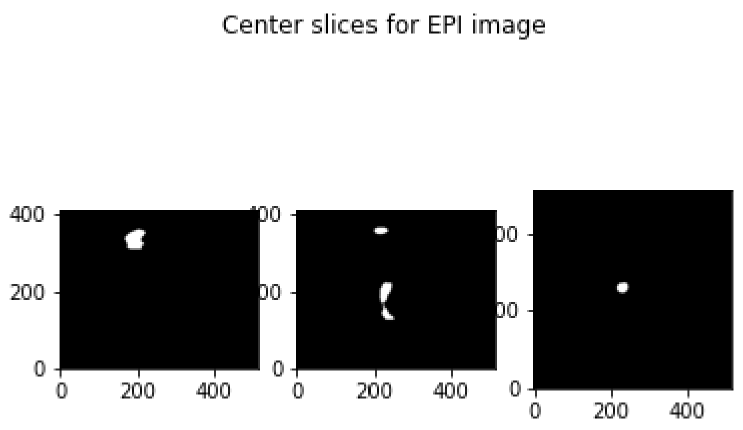
Function description :

Name : evaluateSegmentation
Explanation : This function is given a Ground Truth segmentation NIFTI file and an estimated segmentation NIFTI file computes the VOD and Dice results. For this purpose it reads the images to data images and calculates the intersection and union.
Input : path to ground truth segmentation NIFTI files of the aorta, and path to estimated segmentation created by AortaSegmentation function.
Output a tuple of (VOD_result, DICE_result).

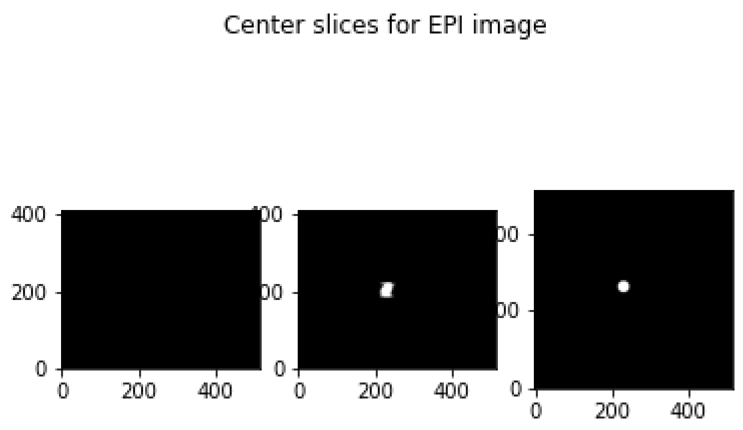
Function outputs:
.

	VOD result	DICE result
Case 1	0.4357943864652881	0.7213957150553231
Case 2	0.7372307033813668	0.41617941982317885
Case 3	0.5747872986934063	0.5967006902433174
Case 4	0.786961995436305	0.35124704050854605

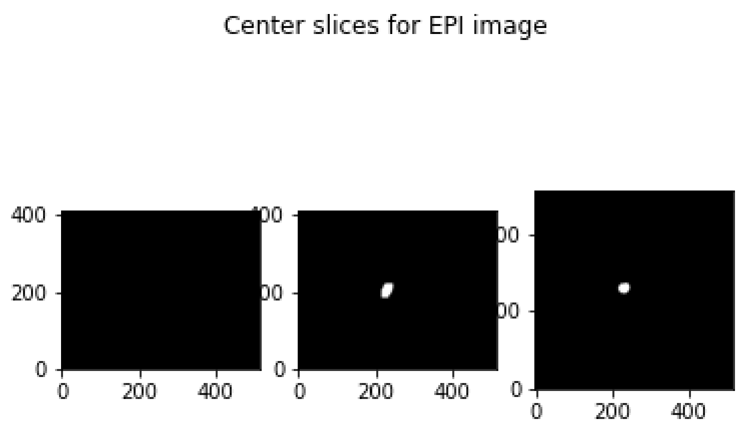
Case 1 :



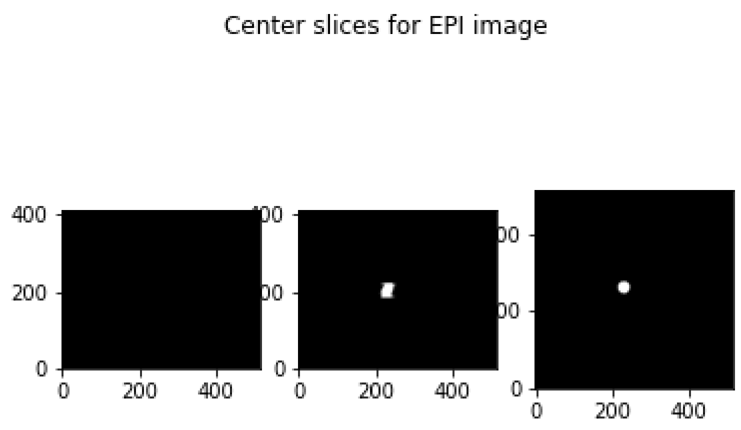
Ground Truth:



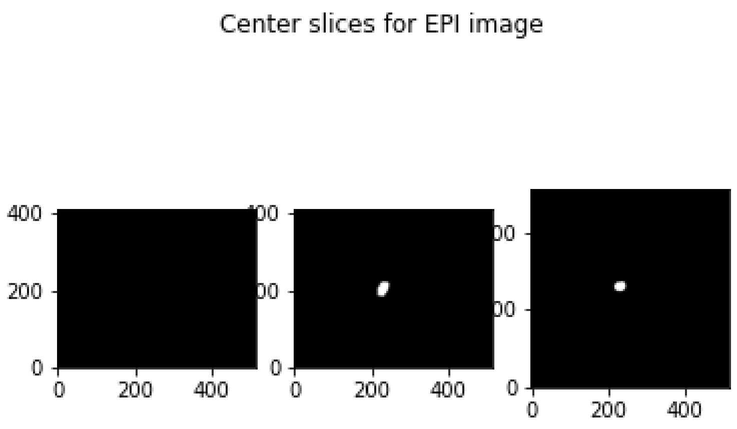
Estimated Segmentation:



GT RIO :

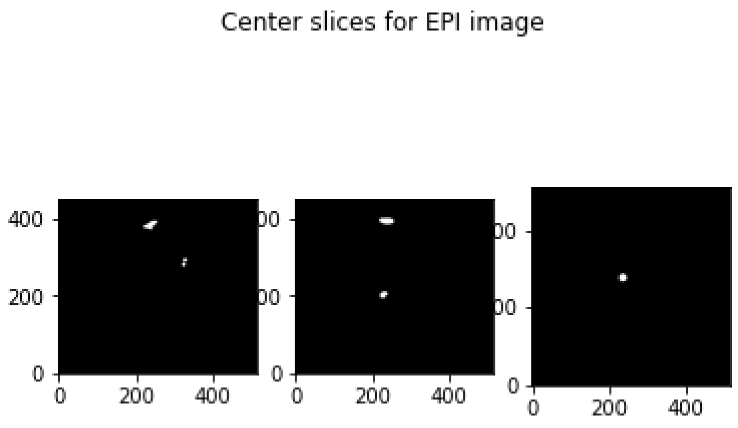


Union:

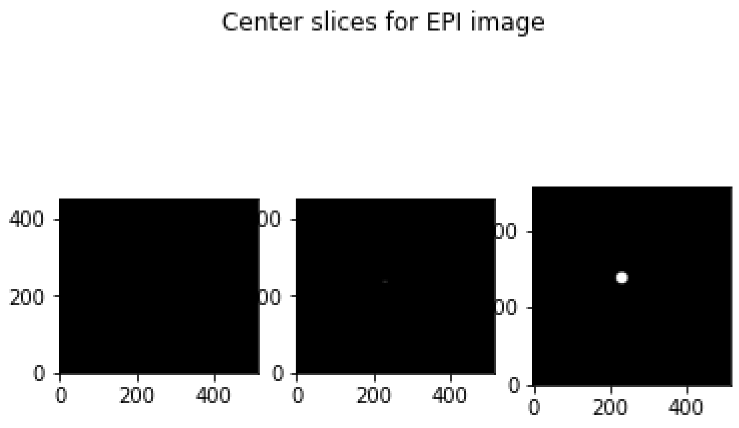


Intersection :

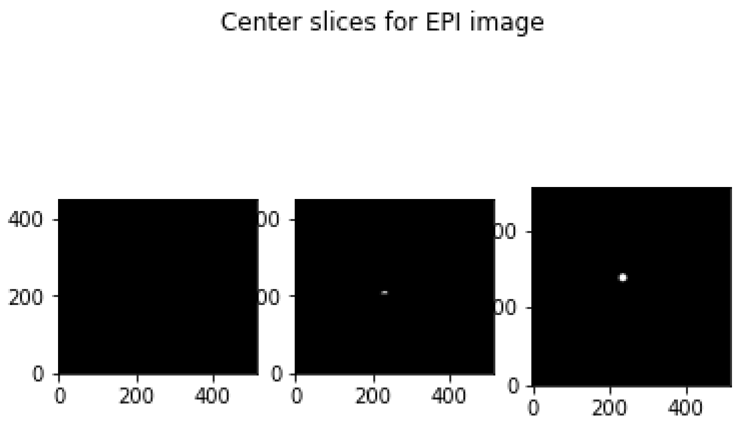
Case 2 :



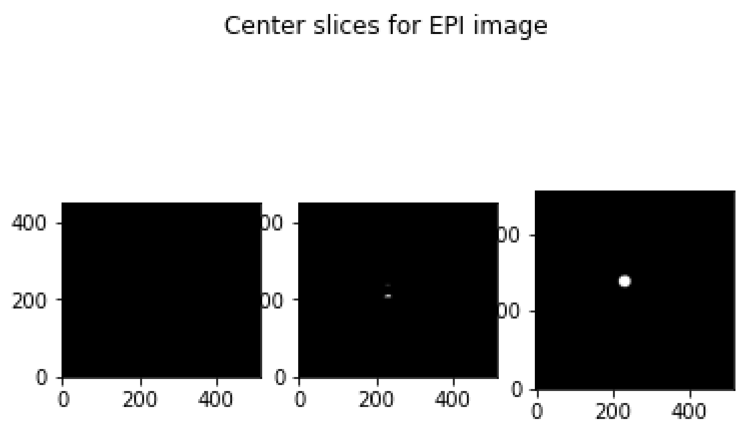
Ground Truth:



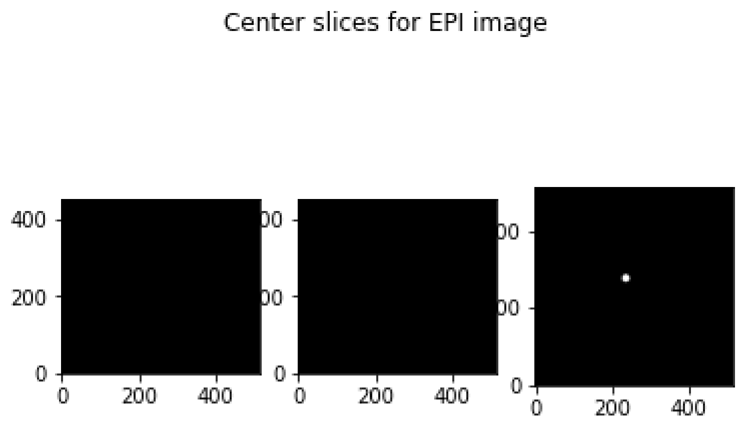
Estimated Segmentation:



GT RIO:

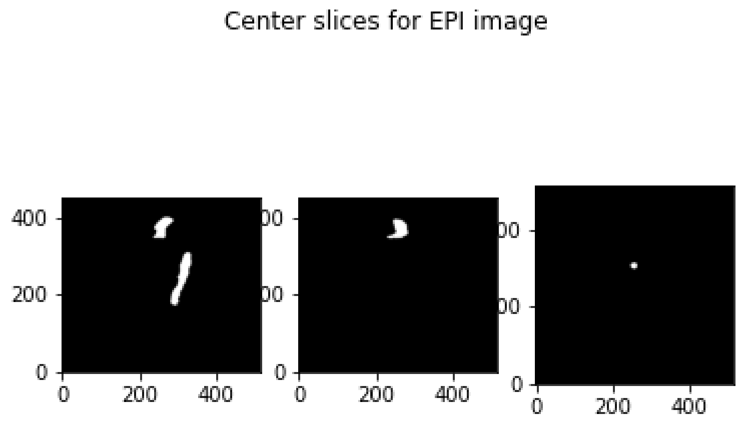


Union:

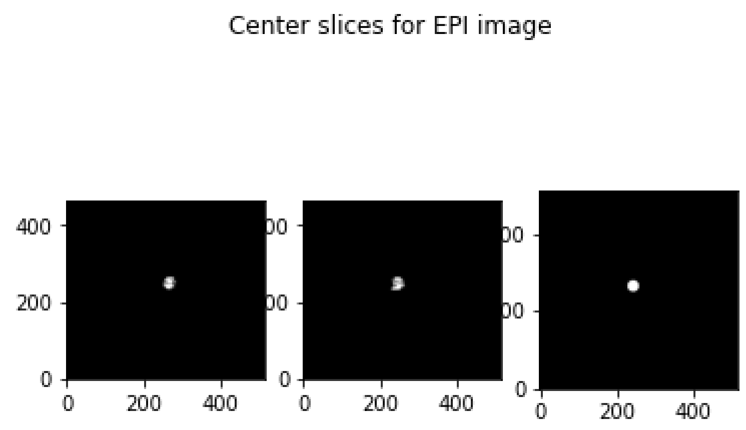


Intersection :

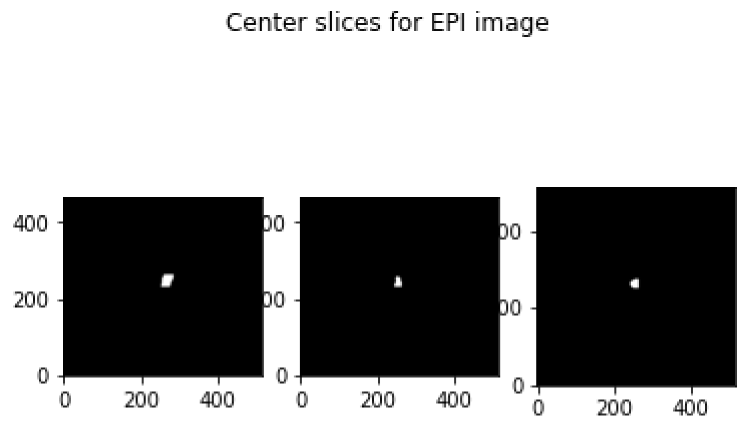
Case 3 :



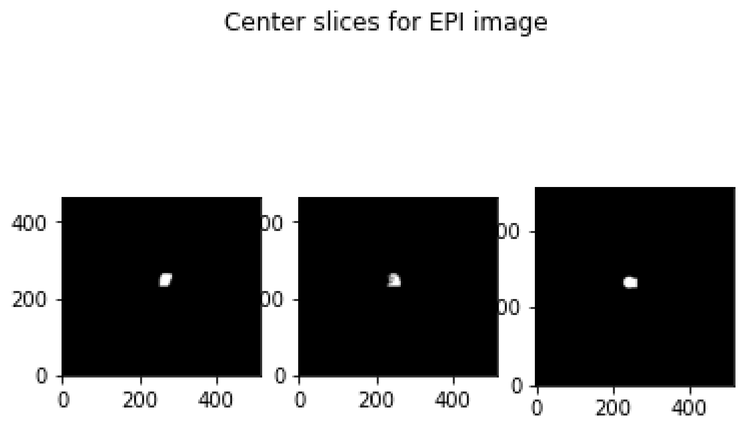
Ground Truth:



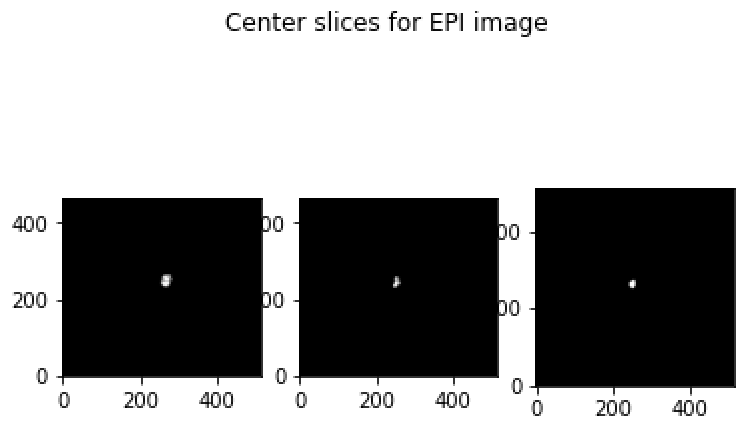
Estimated Segmentation:



GT RIO:

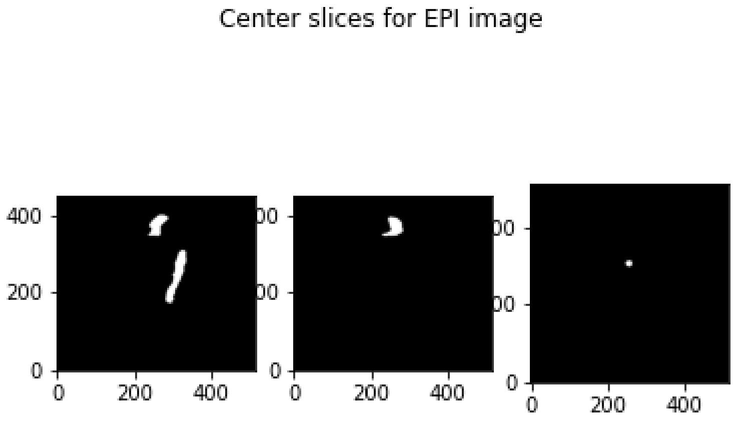


Union:

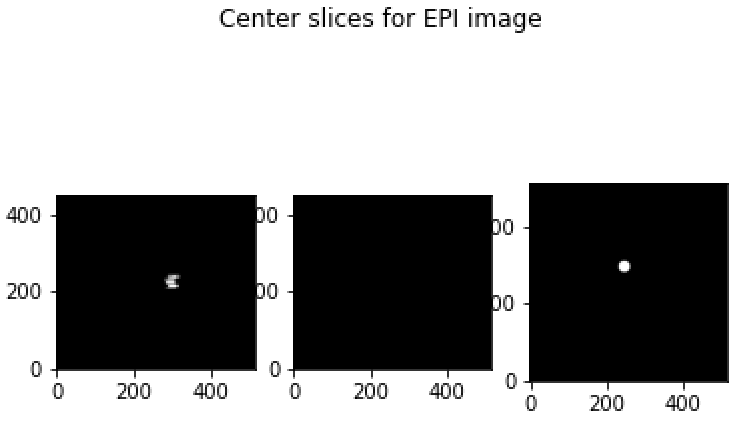


Intersection :

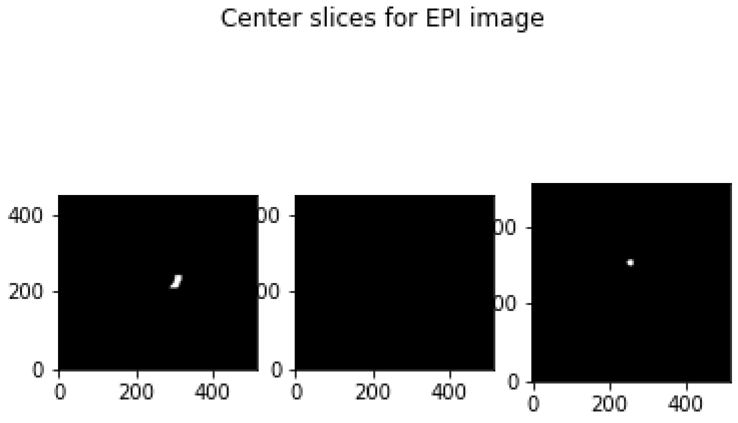
Case 4 :



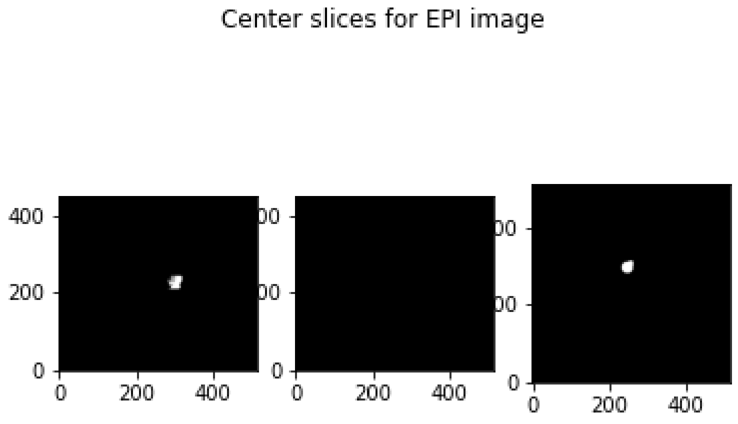
Ground Truth:



Estimated Segmentation:

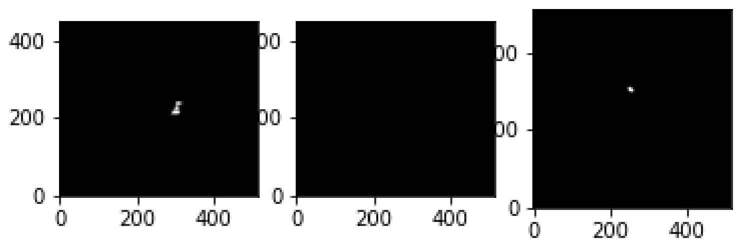


GT RIO:



Union:

Center slices for EPI image



Intersection :