Medical Image Analysis Assignment 2

DICOM and Fast Marching

Jingmo Bai

Introduction

In this assignment we are going to deal with the format of medical images “DICOM” and one of the image segmentation algorithm “Fast Marching”. There are four problems in this assignment.

The first is to load DICOM data, including knowledge of using tags in DICOM files.

The second is to display 3D medical images, we have two data set, thorax and carotid, to display in three image planes, transversal, coronal, and sagittal.

The third and forth are to use the fast marching algorithm to segment objects in medical images, for 2D and 3D cases respectively. We already have the needed algorithm implemented. This task will mainly focus on the construction of speed image.

Description

In the field of medical images, almost all images are stored in DICOM format. DICOM stands for Digital Imaging and Communication in Medicine. It is both an image format and network protocol. Header and data are stored together, including the information about the patient, machine, and data acquisition. Different tags have unique tag ID, we are going to implement some tags to read and use in the task.

In medical imaging there are three standard general traditional views called coronal, sagittal, and transversal planes. Transversal images are seen from the feet, coronal images from the front, and sagittal images from the patient’s left side. Combined together they can give a clear view of different organs.

The fast marching method algorithm is used for image segmentation. We implement a speed function to construct a speed image with different speed for different parts in the image and then use the different arrival time to segment them.

Results

Part 1: Loading DICOM data

Below is the table of DICOM tags added, their ID, type, default value and value representation.

Table: DICOM tags added and VR

Name	Number	Type	Default	VR
TransferSyntaxUID	0002, 0010	char	1.2.840.10008.1.2.1.	UI
StartOfPixelData	7FE0, 0010	uint16	[ ]	OB or OW
BitsAllocated	0028, 0100	uint16	0	US
RescaleSlope	0028, 1053	char	1	DS
RescaleIntercept	0028, 1052	char	0	DS
SliceThickness	0018, 0050	char	1	DS
SpacingBetweenSlices	0018, 0088	char	0	DS
Rows	0028, 0010	uint16	[ ]	US
Columns	0028, 0011	uint16	[ ]	US
PixelSpacing	0028, 0030	char	1	DS

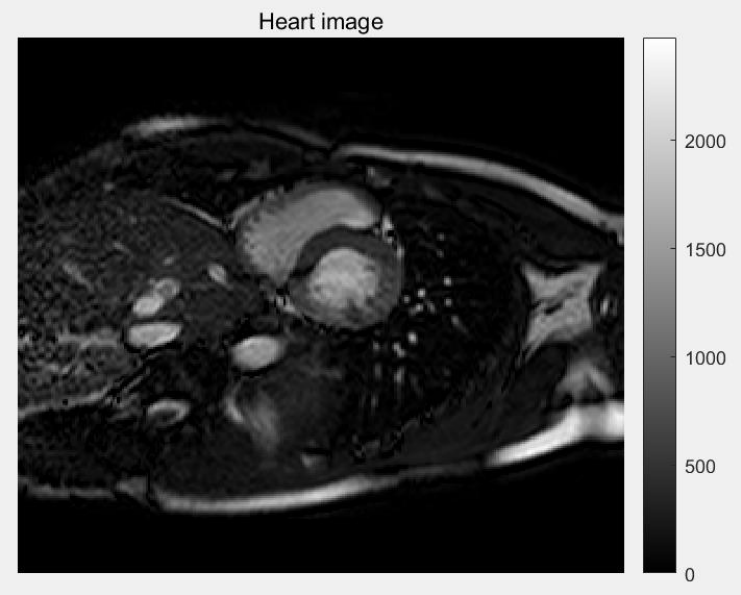
Image intensities in CT are given in Hounsfield scale.

To display the images in the true intensities values, the tags RescaleSlope and RescaleIntercept are used to construct a linear transformation.

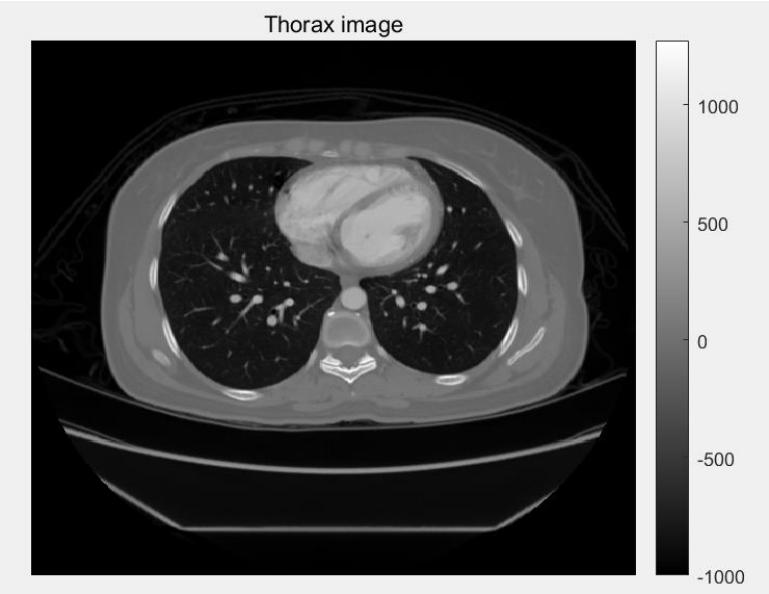
First, we need to convert the type of tag RescaleSlope and RescaleIntercept to double with *str2double*. Then we obtain the true intensities by

Im = Im * RescaleSlope + RescaleIntercept

Below are two test images from heart and thorax respectively. They are displayed in a gray scale with color legend.



MR-heart Test Image



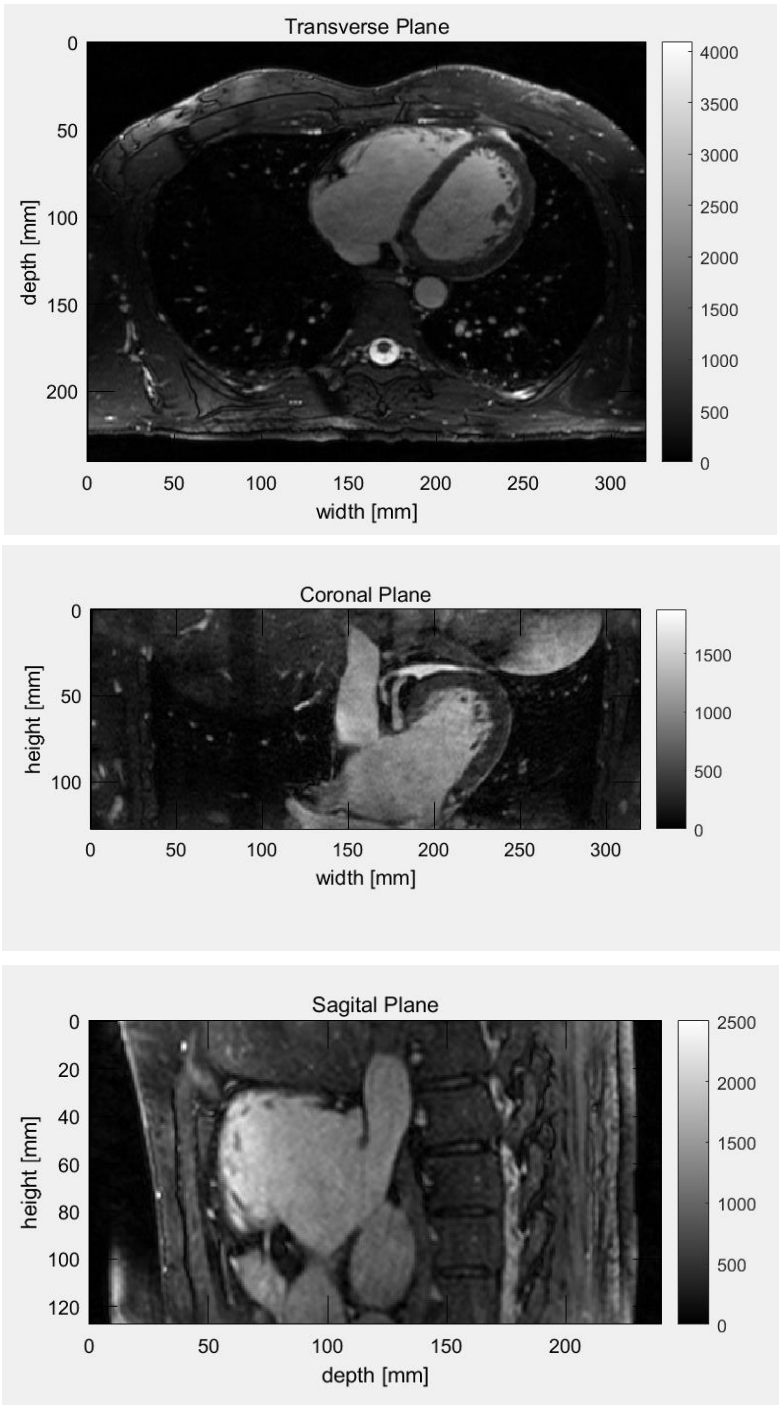
CT-thorax Test Image

For the 3D reader, my assumptions are it will generate a 3-dimension data representing the image volume. The first two size of this volume depend on the size of each file in the folder, constructing each slice. The third size is the number of files in the folder, representing how many slices are there.

One possible situation when the 3D reader would fail is the used tags' type or default value are not implemented correctly. For example, if the RescaleSlope or RescaleIntercept are implemented as not string type or the default value is not string. That's because the function *str2double* would fail in this case and return NaN.

Part 2: Display of medical images

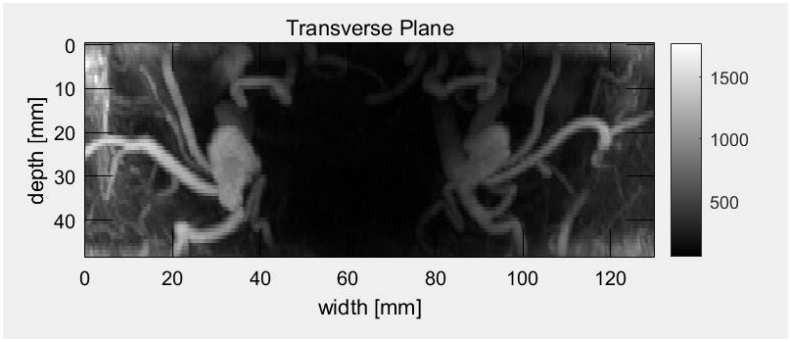
In the first task, we need to display the central slice of MR images of thorax in three standard radiology planes. We need to scale them to the physical size in the unit of mm.
The left/right width is 320mm. The Anterior/Posterior depth is 240mm. The Feet/Head height is 127mm.

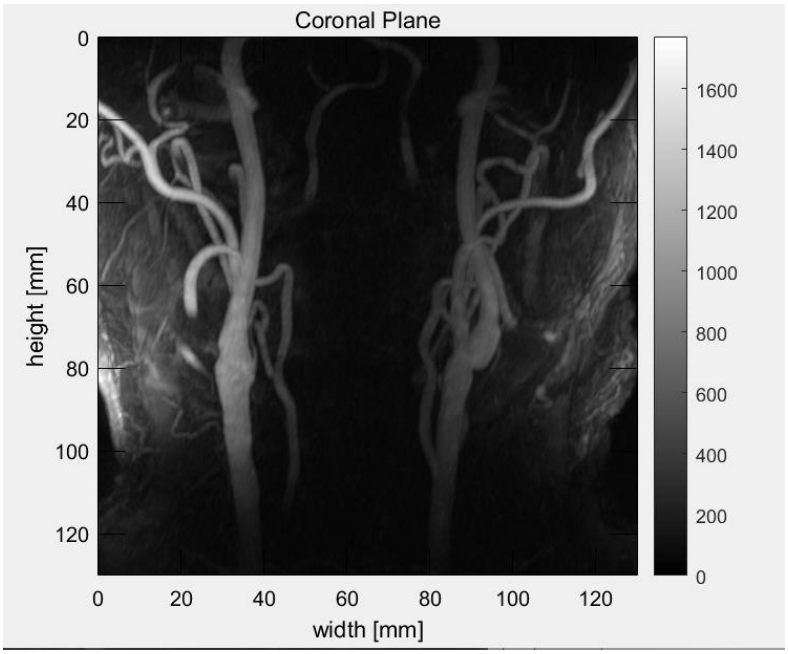


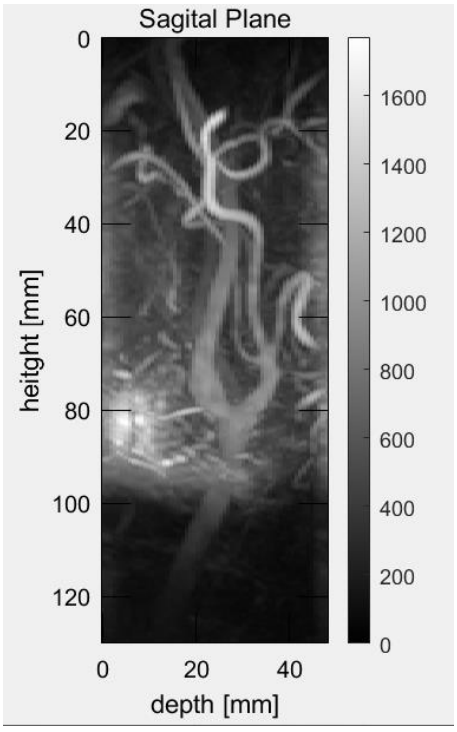
Central slice of MR images of thorax

In the second task, we need to display the Maximum Intensity Projection MR images of carotid in three standard radiology planes. We use the matlab syntax *max(X, [ ], dim)* to compute the maximum intensity projection along the dimension dim of the array X.

We need to scale them to the physical size in the unit of mm. The left/right width is 130mm. The Anterior/Posterior depth is 48mm. The Feet/Head height is 130mm.







Maximum Intensity Projection MR images of carotid

Part 3: Fast marching algorithm 2D

The focus on this part is on the selection on how to construct the speed image. In Fast marching method for segmentation, the design of speed function is to make it high for objects to include and make it low for objects not to include. Then there will be a significant difference on the arrival time.

In my design of speed image, first I perform some pre-processing techniques to the image.

J = imadjust(I) to enhance the contrast and saturation.

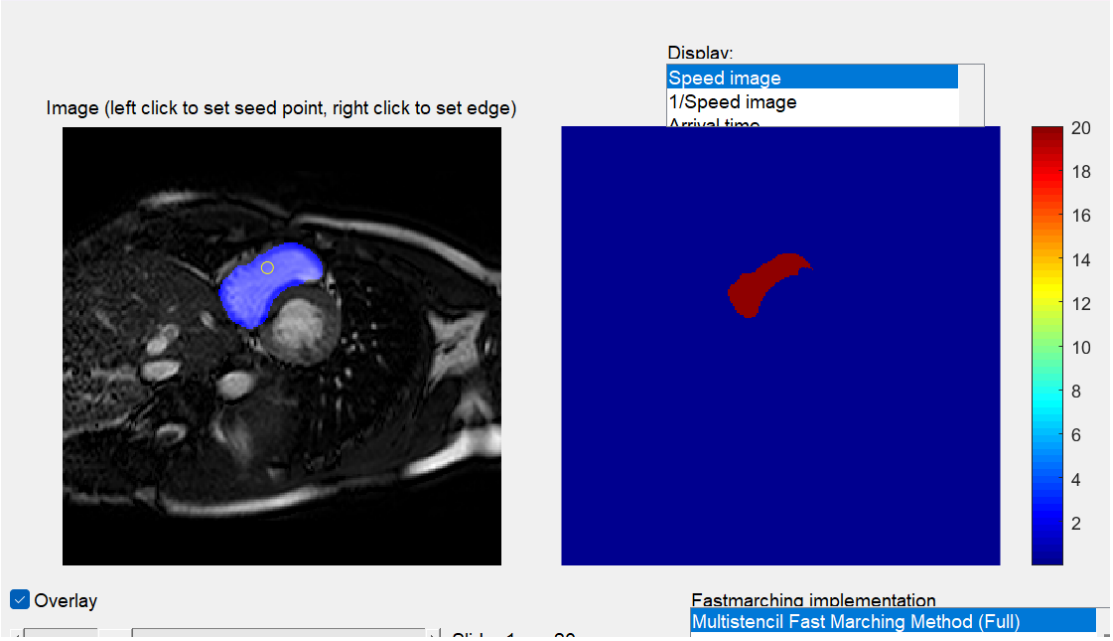
B = imgaussfilt(I,sigma) to apply gaussian filter to the image to smooth the image and reduce noise. I also normalize the image with Min-Max Normalization to shift the intensity values to range from 0 to 1. In the end, I use two functions to construct different regions of speed map.

BW = grayconnected(I, GUI.YSeed, GUI.XSeed). It selects contiguous image region with similar gray values using flood-fill technique. It returns binary mask with 1 where the region is connected with the given coordinates and 0 elsewhere.

ed = edge(I,"canny"). It finds edges in 2D grayscale image with “canny” method. Canny finds edges by looking for local maxima of the gradient of I, and the gradient using the derivative of a Gaussian filter. It returns binary mask with 1 where the function finds edges and 0 elsewhere.

I assign high speed to the pixels where grayconnected returns 1 because that is the region we want to segment. I assign low speed to the pixels where edge returns 1 because that is the edge of regions and should not be across.

The results of segmentation of three tasks are shown below.



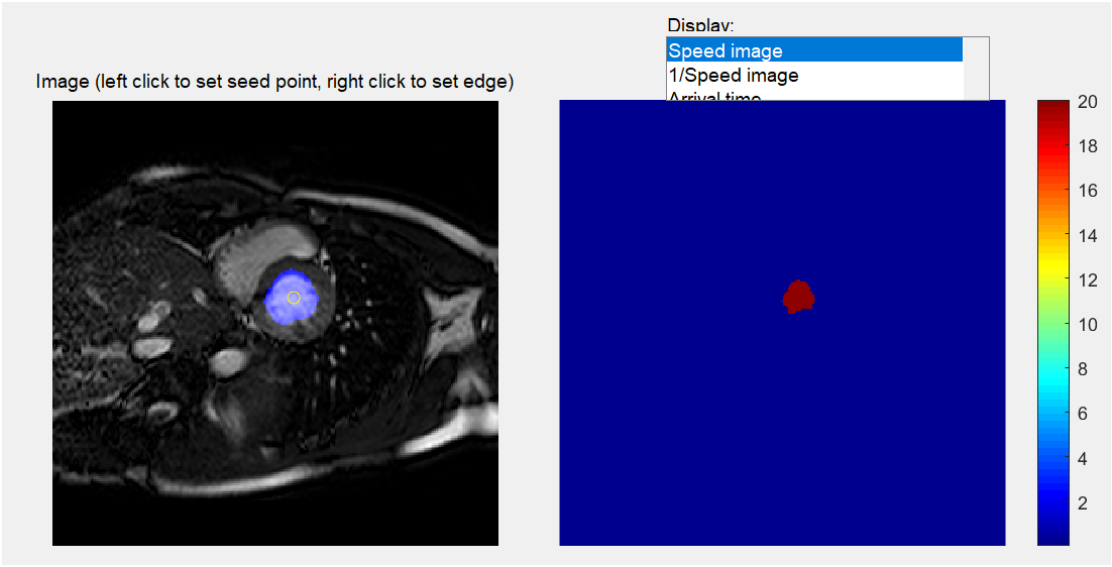


Image (left click to set seed point, right click to set edge)
Display:
Speed image
1/Speed image
20
18
16
14
12
10
8
6
4
2

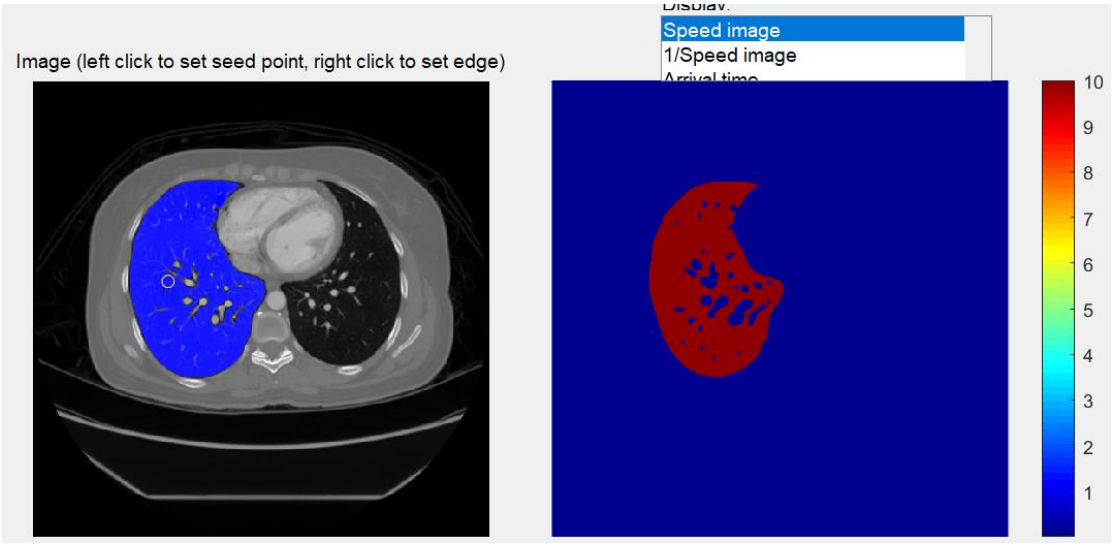


Image (left click to set seed point, right click to set edge)
Speed image
1/Speed image
10
9
8
7
6
5
4
3
2
1

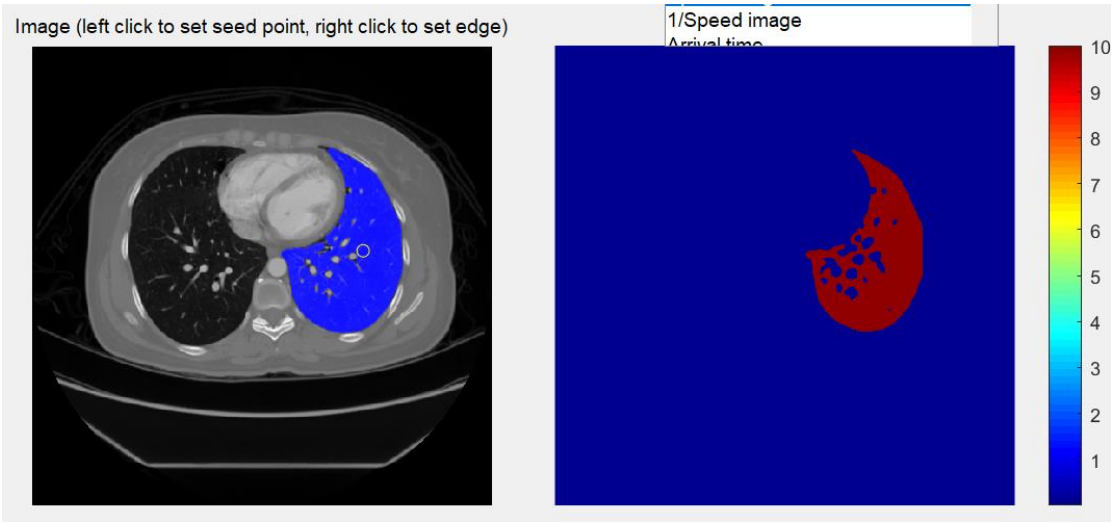
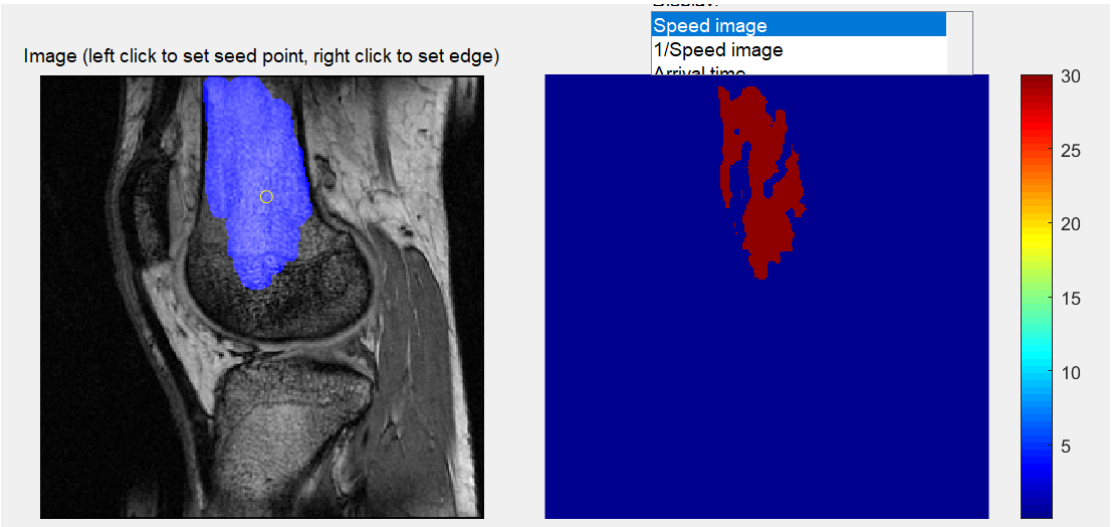


Image (left click to set seed point, right click to set edge)
1/Speed image
10
9
8
7
6
5
4
3
2
1



For the first two task, the segmentations are good. The left ventricle and right ventricle are segmented successfully, also for the left lung. We can see also from the speed image that the speed for the part that we want is relatively high to the other parts.

However, for the third task. It is very hard to segment the femur bone from the other parts. My implement does not work. I think it's mainly because the intensities of the whole femur bone differs a lot, the structure of the bone is not very dense. And there is also a edge inside the femur bone itself, where the intensity looks quite different for the two sides. So both the gray connected and edge function can not contribute a lot here.

The calculation time for the heart segmentation is about 0.02s. For the lung task , it's 0.08s For the knee task, it's about 0.02s.

The algorithm goes from the seed point to each of the N nodes in the speed image in time complexity of O(n log n). So the calculation time is much higher for the lung segmentation because the size of lung is bigger and there are much more nodes.

Part 4: Fast marching algorithm 3D

To find the seed points. I first plot a slice of the images volume im(:,:,30), then I pick two seed points from left carotid and right carotid. The coordinates are left = [290,130,30]; right = [275,368,30]; Then I use sub2ind to translate them into index. So the seed points are hard-coded.

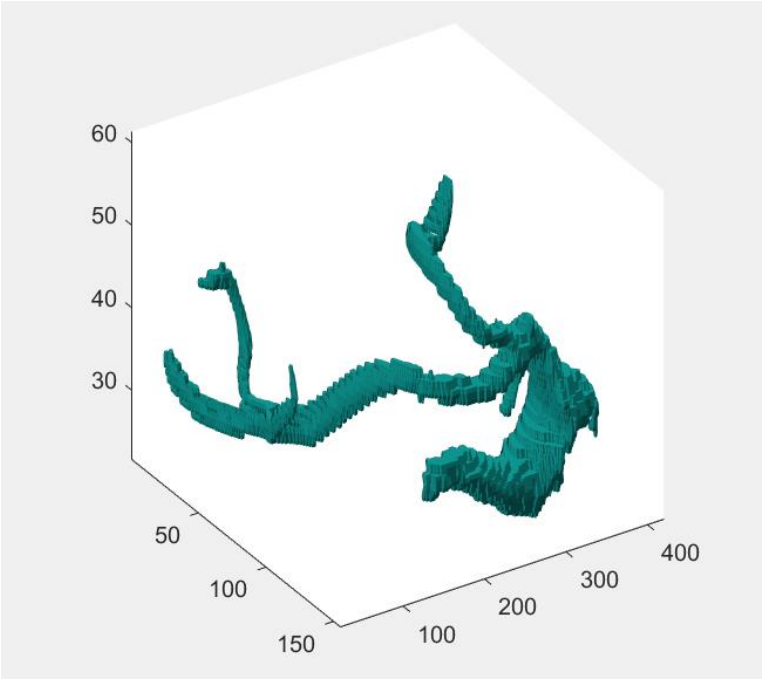
In this 3D case. I use a gaussian kernel-like speed function.

speed = exp(-(im - im(seedpoint).^2/var)

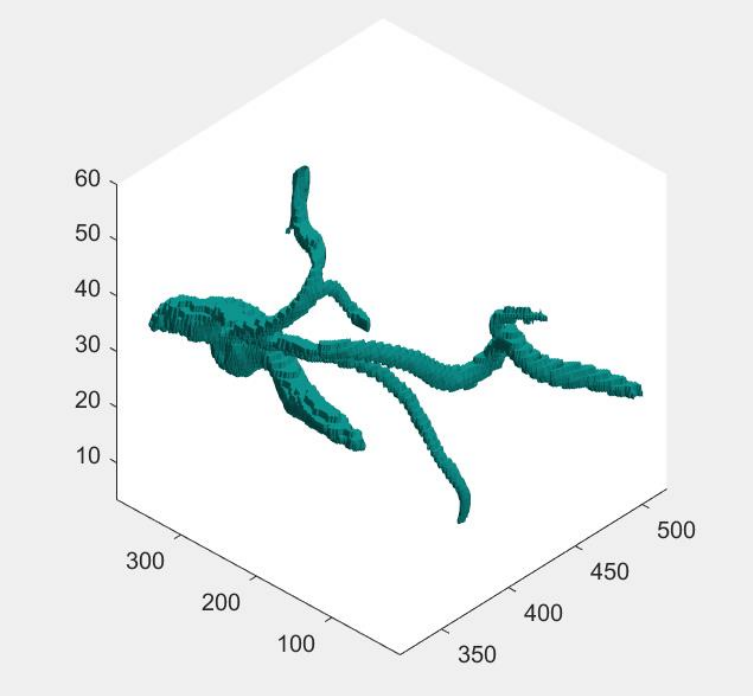
It assigns very high speed values to those points with close intensity as the seed point and very low speed values to points with different intensity. The var can be tuned to change the tolerance of the segmentation, just like changing the variance of gaussian filter.

The input of the given fastmarch function is cost image, so we take reciprocal of the speed image. The maximum cost is 1e9 as a termination condition.

Below are the two carotids segmented from the 3D images volume. They are generated by using isosurface function.



Left carotid



Right carotid