Assignment 4
Medical Image Analysis

Azalea Alothmani

January 9, 2022

1 Introduction

The continuous growth of medical imaging technologies increases the need for efficient and accurate accurate image segmentation algorithms to enhance patient diagnosis and treatment. Magnetic resonance imaging (MRI) together with graphical display technologies are one of the popular tools used to segment medical data. The segmentation is usually performed by hand, and are often very expense and time-consuming. These methods have been used for decades by physicians to make non-invasive diagnoses. However, these methods suffer from limitations in making automatic interpretation of the captured data and thereby difficulties in making efficient segmentation of the different anatomical entities. The increasing requirement of having efficient and accurate medical diagnostics tools making the development of robust medical image segmentation tools with minimum user interaction significant.

The purpose of the assignment is to exploring one of the popular front propagation models used in image segmentation where detailed segmentation is necessary. Fast marching or level set methods are used to tracing moving objects and finding the desired regions of interest automatically. These methods have increased in popularity due to the smooth edges that can be obtained. The first two parts of this assignment covers some hands-on experience on handling DICOM images. In the third and fourth part, the fast marching method in 2D and 3D respectively are studied.

2 Theory

In this part of the report, a brief explanation of the fast marching method is provided. Furthermore, the details of implementation together with some mathematical formulas. The fast marching method is an algorithm used for finding the optimal paths in a graph, and it is similar to Dijkstra's algorithm. The main purpose of the algorithm is to solve the Eikonal equation using an uniformly sized spatial grid. Each point in the grid corresponds to the pixel in 2D and voxel in 3D in the image we are aiming to segment. The solution obtained is in form of arrival time for each point of the grid. In order to accomplish the optimal solution, a speed function at each point is used to calculate the arrival time from a chosen starting point to each point in the grid. Fast matching method has been used to a discretized version of Ekikonal equation on 2D and 3D uniformly spatial grid. Besides to time arrival computation, fast marching method can be used to construct a distance filed for 2D or 3D objects. One example of application where fast marching method is blood vessel segmentation, which is going to be investigated further in this assignment. [1]

Before going ahead with describing the theory of fast marching method in details, a brief introduction of digital imaging and communications in medicine (DICOM) is provided.

2.1 Digital Imaging ans Communications in Medicine (DICOM)

DICOM is a standard used for management of medical imaging information and data. The main usage to DICOM is to storing and transmitting of medical images and data between different medical imaging devices such as servers, scanners, workstation and ***Picture Archiving and Communication Systems (PACS)***. PACS server is a database in which all medical images are stored. The server is used by the radiologist to intierpret the files and identify diagnoses. DICOM incorporates standards for imaging modalities such as radiography, ultrasonography, computed tomography (CT) and magnetic resonance imaging (MRI).

THe DICOM medical image consists of two parts - a header and the actual image. The header contains all the data that describes the image such as patient data and image characteristics. A DICOM data object consists of number of attributes that give information about the image pixel data. The data element or tag composed of a tag that identifies the attribute with hexadecimal numbers, and a DICOM Value Representation (VR) that describes the data type and format.

2.2 Medical Image Display

For medical imaging, the standard anatomical position of a human is described by the anatomical system, also called patient coordinate system. When viewing the images, the sides are switched meaning that left is shown to the right. However, top/bottom and front/back positions remain the same. The three planes that used to describe the anatomical position are; the sagital plane (from the side), coronal plane (from the front) and transverse plane (from the top down). The describe the positions in x,y and z coordinate system, the x-axis goes from front to back, the y-axis from left to right, and z-axis from up to down. In 3D, the coordinates systems provides the dimensions of space given as depth, width and height.

The sagital plane is the yz-plane and it separates the left from the right side. This plane is also called medium plane since it the plane located in the middle of the body. The coronal plane is in the xz-plane and it separates the anterior and posterior plane. While the transverse plane is in the xy-plane and it separates the superior from the inferior plane.

2.3 Fast Marching Method

The goal of fast marching method is study the propagation and evolution of an expanding front of closed curve in 2D or closed surface in 3D by calculating the time it takes for the front to reach every point of a discrete lattice. The front separates the interior and the exterior region of the curve or the surface. The method applies only on uniformly expanding front.

To put the fast marching method in a mathematical framework, we start by considering the spreading of a closed curve denoted by γ of a 2 dimensional or higher surface. The spreading of the surface depends on the speed function $F(x)$, and we study the spreading of the curve over time. The position of the close curve γ can be described by first defining an arrival time function $T(x)$ at each point x in the grid. to describe the propagation of the initial front over time. If the speed function $F(x)$ is positive, the so called boundary value formulation, the time function satisfies the Eikonal function defined as:

$$|\delta T(x)|F(x) = 1 \tag{1}$$

This equation can be solved numerically using fast marching method. [2]

2.3.1 The Algorithm Implementation

In order to find the discrete solution of the continuous function $T(x)$, we start by discretizing the domain into a uniformly sized grid. By doing that, the continuous speed function $F(x)$ becomes discrete on every grid point and therefore can be denoted by F_{ij}.

The algorithm starts by defining an initial front by choosing a single point $(i, j$ as a start point, and defining the time function $T_{i,j}$ at this point to be zero. In order to make the algorithm implementation easier to understand, we start by dividing the points on the grid into three sets; *know, neighbours and far.* Since the time function $T_{i,j}$ is know for the starting point, this point is considered as known. The points next to the starting point are defined as neighbours, while the remaining points a far.

The fast matching method is performed by calculating the the time function Ti, j for the newly added neighbouring points. The point with the smallest $T_{i,j}$ is added to the set of known points. This step is performed attractively until the grid points are set to known.

Spatial derivatives are used to calculate the arrival times of the newly added neighbours. If we are looking at uniformly sized spatial grid where the distance between each point is given as h in x direction as k in y direction. Suppose that we are looking for value of the function $T(x, y)$. Two types of spatial derivative operators called forwards operator, eq2, and backward operator, eq.3, can be defined:

$$D^{+x}T = \frac{T(x+h, y) - T(x, y)}{h} \tag{2}$$

$$D^{-x}T = \frac{T(x, y) - T(x-h, y)}{h} \tag{3}$$

The forwards operator propagates from right to the left, while the backward operator propagates from left to right. In order to calculate the values of $T_{i,j}$ using the speed function $F_{i,j}$, discrete version of the operators is used and it is given by Godunov [3]

$$\begin{bmatrix} \max\left(D^{-x}_{i,j}T, -D^{+x}_{i,j}T, 0\right)^2 \\ + \max\left(D^{-y}_{i,j}T, -D^{+y}_{i,j}T, 0\right)^2 \end{bmatrix}^{\frac{1}{2}} = \frac{1}{F_{i,j}} \tag{4}$$

where $D^{+x}_{i,j}$ and $D^{-x}_{i,j}$ are the standard backward and forward finite difference schemes, respectively, at location (i, j). Eq.4 can be interpreted as follows

$$\begin{bmatrix} \max\left(T_{i,j}-T_{i-1,j}, T_{i,j}-T_{i+1,j}, 0\right)^2 \\ +\max\left(T_{i,j}-T_{i,j-1}, T_{i,j}-T_{i,j+1}, 0\right)^2 \end{bmatrix}^{\frac{1}{2}} = \frac{1}{F_{i,j}}$$

here, $T(i,j)$ is the arrival time for the known point at location (i,j). To calculate the $T_{i,j}$ values of the neighbours, the method presented by R. Kimmel and J.A. Sethian [4] is used given the following expressions

$$a = \min\left(T_{i-1,j}, T_{i+1,j}\right)$$
$$b = \min\left(T_{i,j-1}, T_{i,j+1}\right)$$

The quadratic equation for $T_{i,j}$ of the newly added neighbour point can be solved as follows:

$$\text{If } \frac{1}{F_{i,j}} > |a-b|, \text{ then } T_{i,j} = \frac{a+b+\sqrt{2\left(\frac{1}{F_{i,j}}\right)^2-(a-b)^2}}{2} \tag{5}$$

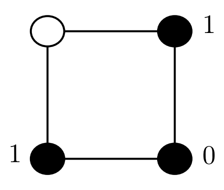
$$\text{Otherwise, let } T_{i,j} = \left(\frac{1}{F_{i,j}}\right)^2 + \min(a,b). \tag{6}$$

The equations above are only used to calculate the time function for the new added point. It the currently calculated $T(i,j)$ value is smaller than the previously calculated one for the point $(i,j) \in$ neighbours, this value will be updated.

The calculated values are stored using an efficient structure called heap, where the smallest value is stored after every iteration. The reason of using heap is due to the fact that heap has a binary tree structure, where the the children of each node is equal or greater than the parent node. Using heap makes the process of finding the smallest value efficient since it ensures that the smallest value is the first entry.

2.3.2 Mathematical Analysis

As it has been explained earlier, fast marching method is a fast method to solve the Eikonal equation. However, the precision of this method can be discussed. We assume that front propagates from the frozen black node labeled 0 in the grid shown below, and that the distances to the other two black nodes have been calculated. The white node is being updated.



According to eq.4, the value at the white node should be $1+\sqrt{1/2}$. However, it is clear from the gird above that the distance should be $\sqrt{2}$, meaning that the value is around 0.3 wrong. This error originates from the fact that the fast marching method does not know the curvature of the front. The method is implemented based on an assumption that the front is propagating in a diamond shape due to the discretization of the problem, with the vertical and horizontal axis as its corners. In continuous version, the front would have propagate in circular shape instead. With this limitation of fast marching method, the algorithm constructs a solution for eq.1 with an execution cost of *nlogn*.

2.4 Segmentation using Fast Marching Method

Fast marching method is based on theory of level set that provides a framework for tracking the evolution of a curve or surface given the velocity of the propagating front in its normal direction. The method applies specifically to the cases where there are a clear boundary and distinction between outside and inside region. The boundary between regions can be formulated as different level sets.

To segment medical images, fast marching method is applied to extract the rough boundaries of the interested object. The extracted edges are then considered as an initialization of level set method. However, this is also one of the limitation of fast marching method since the speed function only depends on the edge information (gradient information) and does not make use of the global information in the image. Having that in mind, obtaining a good segmentation for blurred edges can therefore be difficult.

Fast marching algorithm can be used as an advanced region growing segmentation approach being controlled by a speed image. One of the most important criteria of the speed image is that it should be close to zero near object boundaries and relatively high in between. To produce an edge image, gradient magnitude is used after applying anisotropic diffusion to reduce image noise without removing significant parts of the image. Gaussian is then used to enable level-set to slow down near edges. To ensure that the boundaries are near to zero and that homogeneous regions are close to one, Sigmoid filter is applied to perform linear transform on the gradient magnitude.

To start the segmentation, an initial seed point which is the starting location of the level-set. The output of fast marching method is time-crossing map indicating the time of arrival of the propagated level-set front. To form the segmentation, threshold is applied to the region that the level-set front propagated through. The contour propagation over time can be shown graphically using an interface that enables the used to select the seed point and the determine the desired segmentation.

3 Method

In the first part of the assignment, hands-on experience with handling DICOM images is given. The second part is about studying the three general traditional views of displaying medical images using the DICOM reader implemented in part 1. In part 3 and 4, a genetic 2D and 3D interactive image segmentation algorithm is implemented respectively.

3.1 Part 1 - Loading DICOM Data

In this part of the assignment, the task is to complete code for a Matlab based DICOM reader which is used to load the data. The first task is to complement the DICOM reader with suitable DICOM tags. The following tags are implemented according to [5]. The DICOM reader function was then tested on two images, and the graphical outputs of the suggested images are included later in the result part of this report.

The DICOM data elements are added as a list or dictionary. Each data element is composed of, a tag which identifies the element given in the format (xxxx, xxxx) for group number and element number respectively. For each added data element, a corresponding value representation (VR) is added which describes the data type and format of the attribute's value. A list of the added DICOM tags are shown below, together with corresponding value representations. The tags are implemented according to [5].

- name: RescaleIntercept - group number = '0028'; element number = '1052', VR: DS (char)
- name: SliceThickness - group number = '0018'; element number = '0052', VR: DS (char)
- name: SpacingBetweenSlices - group number = '0018'; element number = '0088', VR: DS (char)
- name: Rows - group number = '0028'; element number = '0010', VR: US (uint16)
- name: Columns - group number = '0028'; element number = '0011', VR: (uint16)
- name: PixelSpacing - group number = '0028'; element number = '0030', VR: DS (char)
- name: ImageOritentationPatient - group number = '0020'; element number = '0037', VR: DS(char)
- name: ImagePosition - group number = '0020'; element number = '0032', VR: DS(char)
- name: BitsStored - group number = '0028'; element number = '0101', VR: US (uint16)
- name: PixelData - group number = '7ef0'; element number = '0010', VR: US (unit32)

The next step is to use the previously implemented DICOM reader to read the image data. In order to do that the image data need to be reshaped and the data to be rescaled to true image intensities. The provided data is a CT scanner data, where the image intensities are given in Hounsfield scale (HU) that measures radiodensity. For CT data, intercept and slope define the transformation between intensity values (IV) and Hounsfield Units (HU):

$$IV = HU * Rescaleslope + Rescaleintercept$$

By applying this formula to the image data, the pixel values will be transformed into values that are meaningful for the application used. The main reason of applying scaling on image data is due to the fact that DICOM images typically contain 12-16 bits/pixel (4,096 to 65,536 shades of gray. Majority of medical displays and computer screens are often limited to 8 bits (256 shades of gray). Due to the high amount of voxel values in the DICOM images, the scaling to true image intensities is importance in order to change the appearance of the image to highlight the particular regions of interest.

3.2 Part 2 - Display of medical images

To display medical images, three standard general traditional views called coronal, sagital, frontal views are used. In this part of the assignment, the DICOM reader from part 1 is used to load a 3D MR thorax image from the folder *MR-thorax-transversal*. The MR images are collected in the transversal plane. The three standard images are displayed with **mm** unit scaling. Matlab commands such as *imagesc()* and *axis image* are used to ensure that the correct aspect ration is displayed. To it easier to identify the planes, the physical size of the image box in the three directions Right/Left, Anterior/Posterior, and Feet/Head (in the unit mm) has been calculated.

3.3 Part 3 - Fast marching algorithm 2D

In this part of the assignment, a generic 2D interactive image segmentation algorithm is implemented. A speed image function that the interactive tool can be flexible to segment many different types of images needs to be written. The speed image is a mapping from intensity (or other features in the image) to the speed of the propagating surface in the fast marching algorithm. The speed should be high for objects to be included and low for other objects.

To design the speed image, some pre-processing was performed. The following steps were performed to design the speed image function:

1. The minimum value was shifted towards the origin to get ride of negative values
2. Normalizing of the image
3. Enhancing image contrast using Matlab built-in function *imadjust()*. This is implemented to be controlled by slider 2.
4. Image blurring with Gaussian filter. The resulting image is referred to as blurred image.
5. Select contiguous image region with similar gray values as the seed using *grayconnected()*. The resulting image is referred to as grayconnected image.
6. Applying some morphological methods to extract large edges, by first calculating the 2^{nd} spatial derivative of an image using **Laplacian filter** to penalize large jumps in image. *bwareafilt()* was then used to extract all connected components from the binary image with an area in the range [4e2, 1e6]. The image is then dilated by adding pixels to the boundaries of objects in an image to make boundaries smoother to force a closed line. This is implemented in slider 3.
7. Make sure that the speed image of the object at the seed is larger than its' bounds by first calculated the speed of the object and applying a constrain; if the mean value of the speed image is less than the speed at the seed point, the image is reversed. Referred to as 'edge image'.
8. After Once figuring out where in the image we want to have low and high speed, the image intensities were converted to suitable speed using the following formula:
   $GUI.SPEED = blurredimage + GUI.Slider1 * grayconnectedimage - 5 * edgeimage$

3.4 Part 4 - Fast marching algorithm 3D

The purpose of this part of the assignment is to explore three dimensional segmentation using fast marching method. An automated code is implemented to load the images and perform segmentation. The starting point was manually selected by using an pre-implemented algorithm to display 3D grayscale image in slice by slice fashion [1]. In order to calculate the speed image, similar pre-processing steps that have been implemented in the part 3 are used. The image is normalized thesholding was applied so that pixels lower than 0.45 are set to black. The speed function should be larger than zero (min value 1e-8), otherwise some regions will never be reached because the time will go to infinity. The min value chosen is 1e-6. The segmentation was performed using the pre-implemented *msfm3d*. This function calculates the shortest distance from a list ofpoints to all other pixels in an image volume, using the Multi-stencil Fast Marching Method (MSFM). This method gives more accurate distances by using second order derivatives and cross neighbours.

4 Result

4.1 Part 1 - Loading DICOM Data

After modifying the DICOM taglist by adding additional tags, the graphical output of the *MR-heart-single.dcm* is:

```
struct with fields:

        TransferSyntaxUID: '1.2.840.10008.1.2.1.'
         StartOfPixelData: 6570
            BitsAllocated: 16
             RescaleSlope: '5.38461538461539'
         RescaleIntercept: '0.0 '
           SliceThickness: '8.0 '
    SpacingBetweenSlices: '8.0 '
                     Rows: 256
                  Columns: 256
             PixelSpacing: '1.40625\1.40625 '
   ImageOrientationPatient: '0.73984640836715\0.00664897123351\0.67274296283721\0.40173023939132\0.79774409532547\-0.4496856033802 '
            ImagePosition: '-149.52497243881\-153.74338393099\-111.97172820568'
               BitsStored: 12
                PixelData: []
```

Figure 1: The graphical output of the test image *MR-heart-single.dcm* with the name of the tags added together with the value representation.

The graphical output of the DICOM tags added together with the value representation (VR) for the testing image *CT-thorax-single.dcm*

[1] `https://se.mathworks.com/matlabcentral/fileexchange/41334-imshow3d`

```
struct with fields:

        TransferSyntaxUID: '1.2.840.10008.1.2.1.'
         StartOfPixelData: 3616
            BitsAllocated: 16
             RescaleSlope: '1 '
         RescaleIntercept: '-1024 '
           SliceThickness: '2.500000'
      SpacingBetweenSlices: '0'
                     Rows: 512
                  Columns: 512
             PixelSpacing: '0.703125\0.703125 '
   ImageOrientationPatient: '1.000000\0.000000\0.000000\0.000000\1.000000\0.000000 '
            ImagePosition: '-180.000\-180.000\58.500'
               BitsStored: 16
                PixelData: []
```

Figure 2: The graphical output of the test image *CT-thorax-single.dcm* with the name of the tags added together with the value representation.

The next step was to test the DICOM reader to read image data for two images; MR image of a heart and CT scanner of thorax. The MR image of the heart is shown in Fig.3.

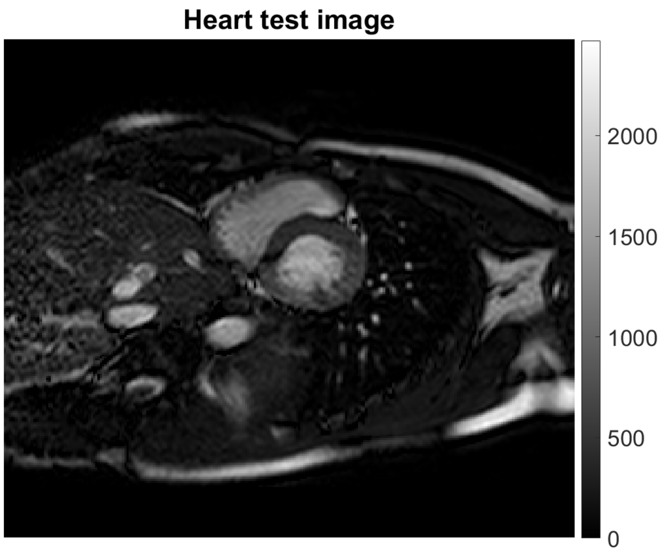


Figure 3: MR image of a heart read using the implemented DICOM reader after included additional tags to the taglist, and performing reshape and rescaling.

The CT scanner image of the thorax is shown in Fig.4. In Fig.5, the values outside domain are removed to get relevant scale/colorbar.

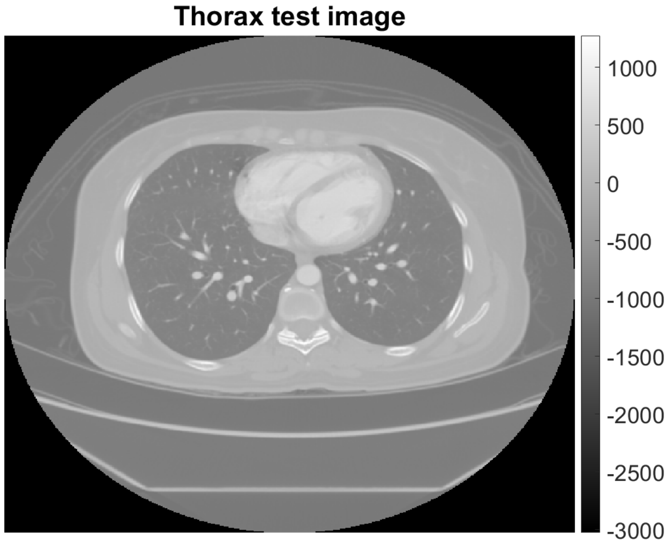


Figure 4: (a) CT scanner of a thorax. The image is read using the implemented DICOM reader after included additional tags to the taglist, and performing reshape and rescaling.

Figure 5: CT scanner of a thorax with only domain value. The values outside domain are removed to get relevant scale/colorbar. The image is read using the implemented DICOM reader after included additional tags to the taglist, and performing reshape and rescaling.

To be able to read all DICOM files in a folder to a 3D image volume, a code was written to loop over all the files in the folder *MR-thorax-transversal*. There are 160 images in this folder, which have been displayed in order to make sure that the DICOM reader works as intended. Fig.6 shows some of these images.

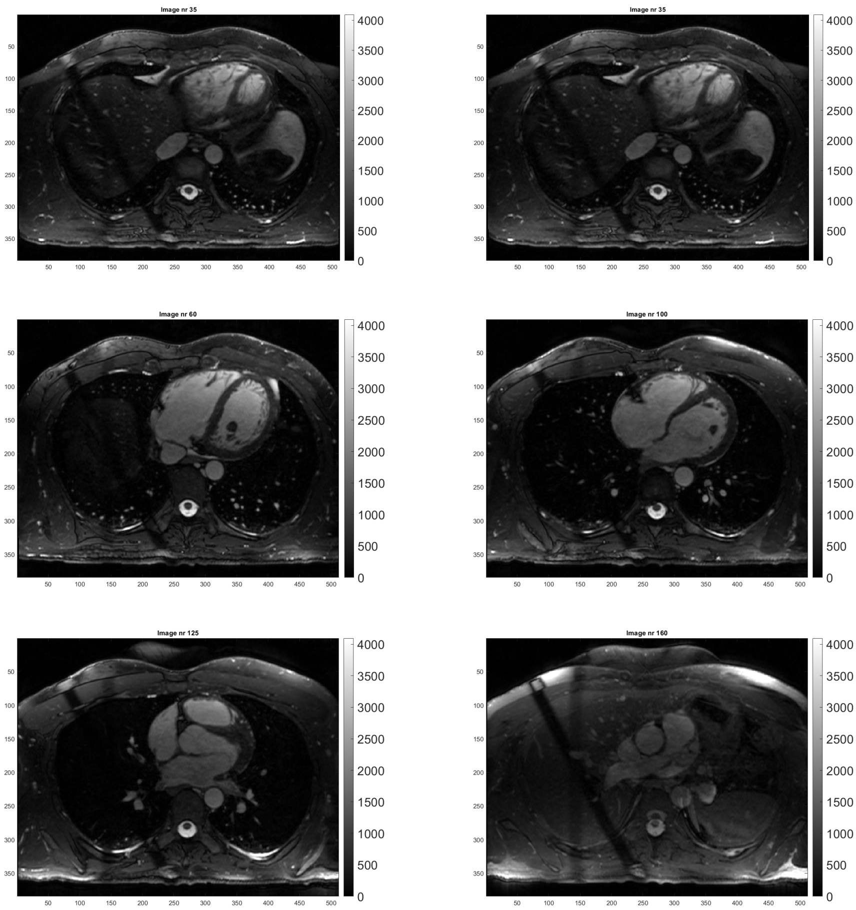


Figure 6: Displaying some of the images in the folder *MR-thorax-transversal* using the DICOM reader.

When querying PCAS systems, image information model is used where the information are stored on different levels; patient level information, study level information, series information/collection of images and image information. The PCAS systems query information on different and this can have impact on if the DICOM reader works or not. During storage communication, the source response to a request contains type of the data that the server can accept, The request can be acknowledge or a reject. If the request is not understood or can not be properly answered, it will be rejected.

In general, DICOM is a large and complex format which making it difficult to implement support. It is also important to mention that each implementation has a different level of completeness making it difficult to predict which tags and services will be available. One other problem is that most scanners add custom proprietary tags to the header and choosing a proper dictionary tags to read this information can be challenging. These custom tags might not be implemented in the standard DICOM tags. One example is the velocity encoded MR images to study the blood flow velocity. Different manufacturers implement their own specific dictionary tags in order to study blood flow, and one should compare manufacturers specifically. This

is problematic and the software can be complex due to that.
One other assumption for the loaded images is that the patient coordinate system is assumed to be centered in the image array. This assumption holds as long as the scanner operator does not perform any position displacement. However, if the patient position changes the image might appear displaced.

4.2 Part 2 - Display of medical images

After calculating the physical size of the image box and doing some processing on the images such as reshaping, transposing and flipping, the MR image of the thorax is displayed from the three different planes; transverse, coronal and sagital. Fig.7 shows the thorax in the transverse plane, seeing from the patient's feet. Fig.8 shows the thorax in the coronal plane, seeing from the patient's front. Fig.9 shows the thorax in the sagital plane, seeing from the patient's left side.

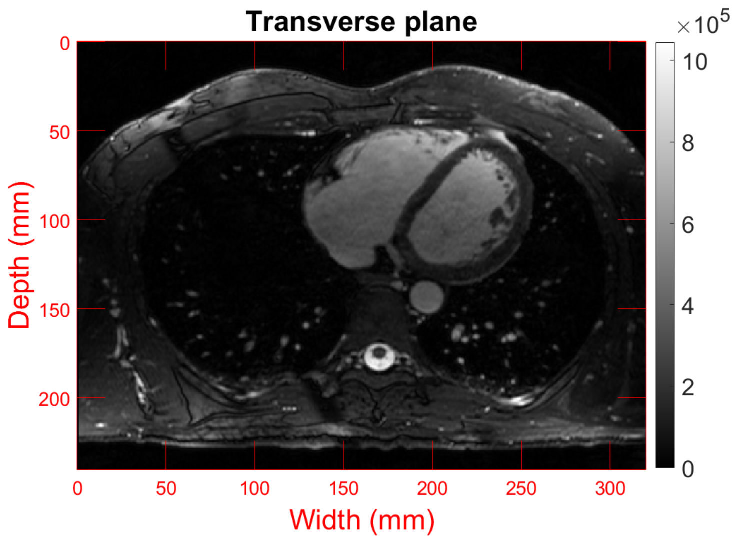


Figure 7: The MR thorax image as shown from transverse plane, seeing from the patient's feet.

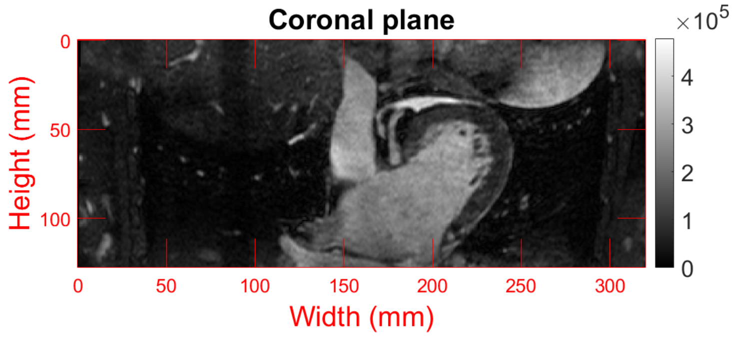


Figure 8: The MR thorax image as shown from coronal plane, seeing from the patient's front.

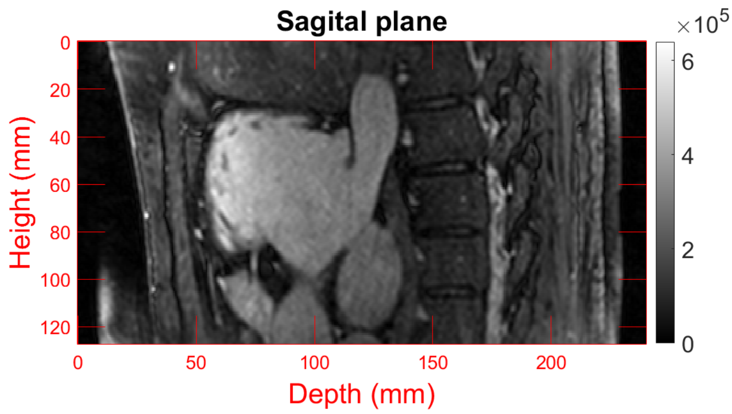


Figure 9: The MR thorax image as shown from sagital plane, seeing from the patient's left side.

The MR image of the carotid is displayed from the three different planes; transverse, coronal and sagital. Instead of displaying the central slice, maximum intensity projection is performed. Fig.10 shows the thorax in the transverse plane, seeing from the patient's feet. Fig.11 shows the thorax in the coronal plane, seeing from the patient's front. Fig.12 shows the thorax in the sagital plane, seeing from the patient's left side.

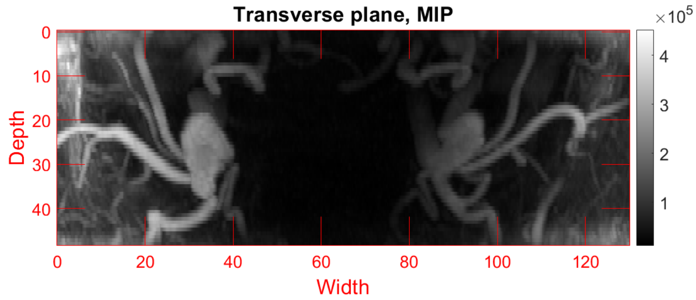


Figure 10: The MR carotid image as shown from transverse plane, seeing from the patient's feet.

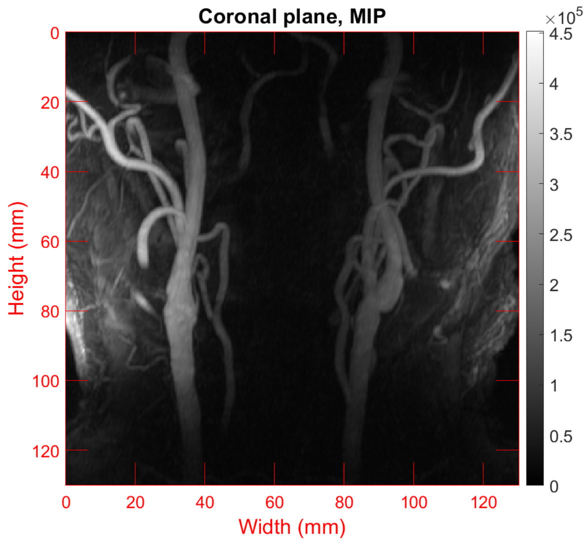


Figure 11: The MR carotid image as shown from coronal plane, seeing from the patient's front.

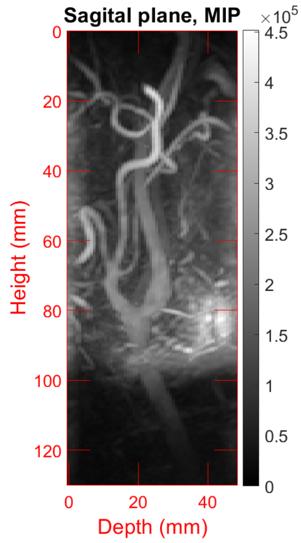


Figure 12: The MR carotid image as shown from the sagital plane, seeing from the patient's left side.

4.3 Part 3 - Fast marching algorithm 2D

After implementing the generic 2D interactive image segmentation algorithm, the left and right ventricle of the MR image of the heart were segmented. Besides, the algorithm was used to segment the left lung in a CT image of the thorax. Lastly, the femur bone in a MR image of a knee was segmented. The segmented images for the left and right ventricle are shown in Fig.13 and Fig.14, respectively. The figures show the segmented images to the left, together with the speed image to the right.

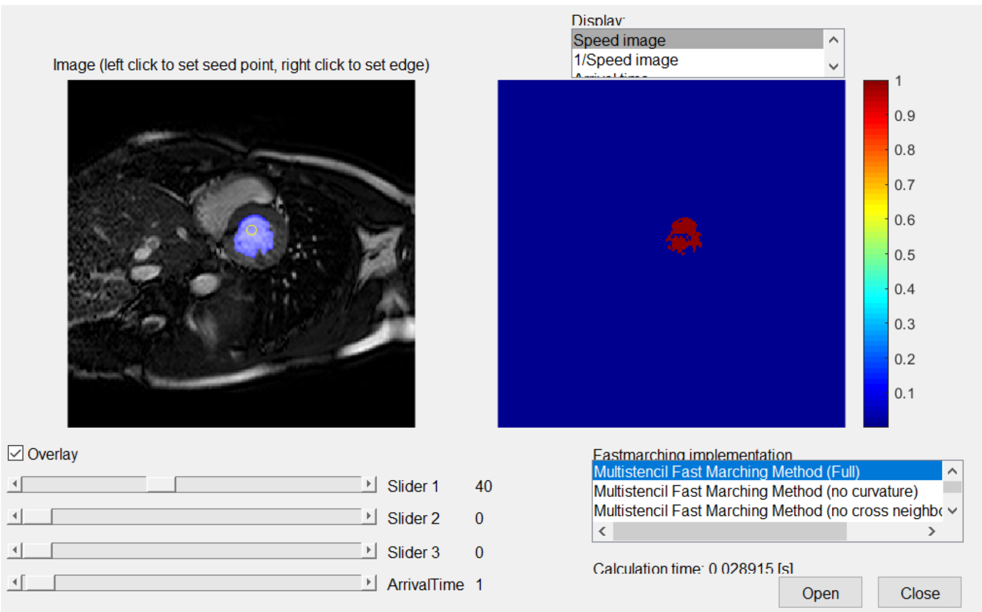


Figure 13: The segmented left ventricle to the left and the corresponding the speed image to the right.

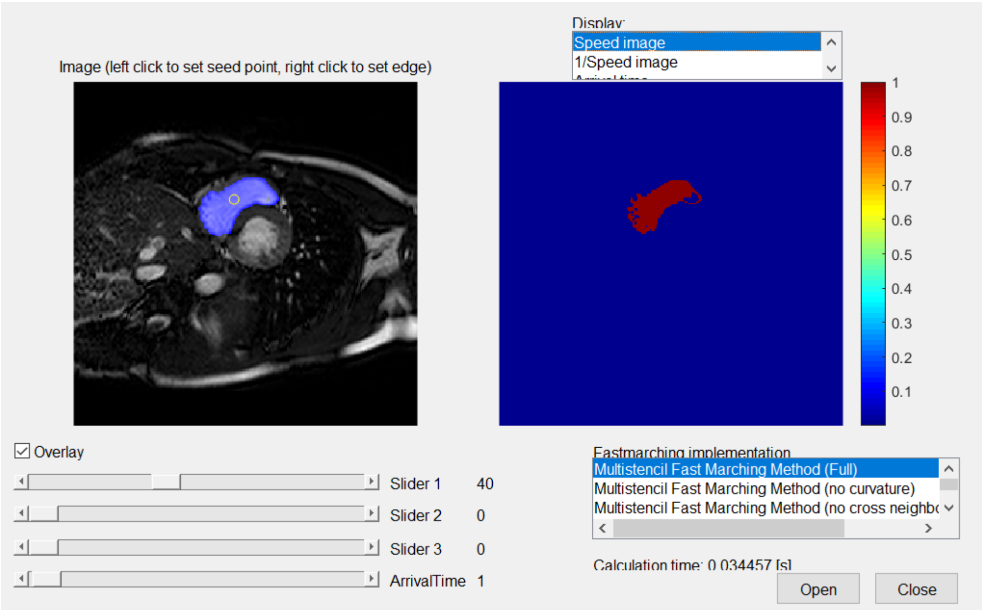


Figure 14: The segmented right ventricle to the left and the corresponding the speed image to the right.

The segmented left lung to the left, together with the speed image to the right.

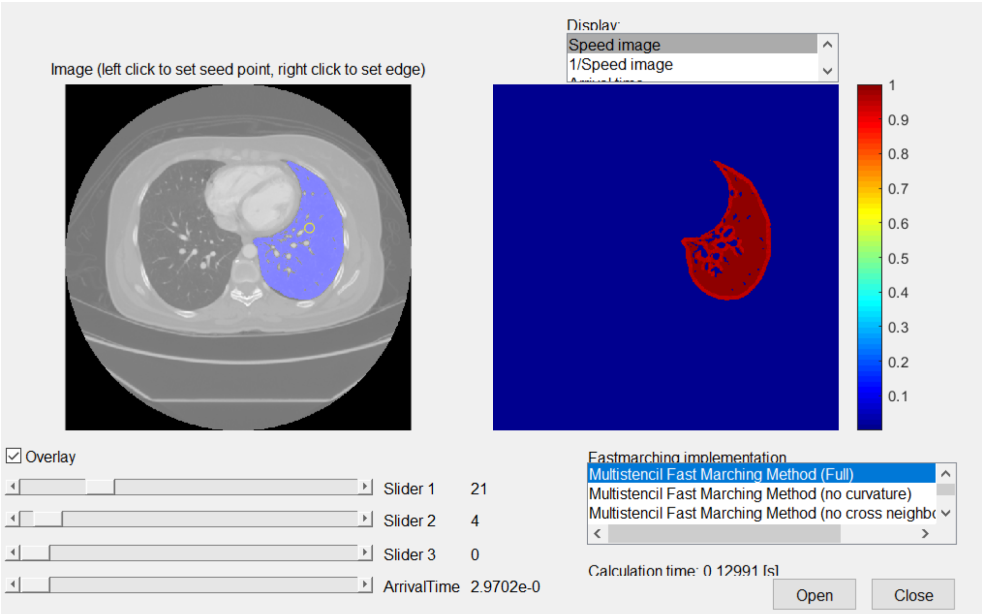


Figure 15: The segmented left lung to the right and the corresponding the speed image to the right.

The segmented femur bone to the left, together with the speed image to the right.

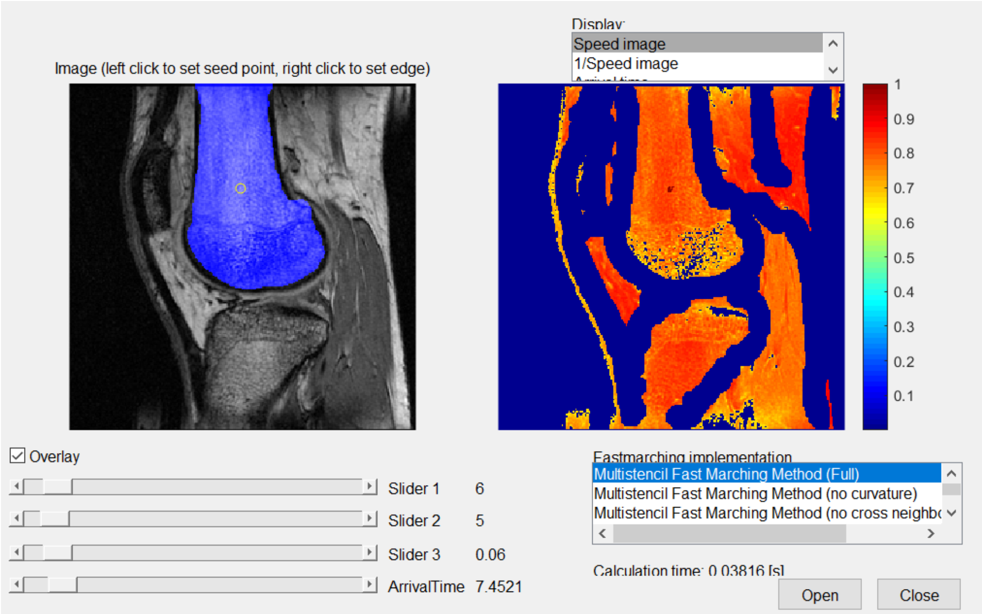


Figure 16: The segmented femur bone to the right and the corresponding the speed image to the right.

The segmentation using the implemented 2D method seems to perform good on all the three test images as we easily could segment the right and the left ventricle, the left lung and the femur bone. For the MR image of the heart, the speed image is high inside the interesting region while the speed outside the interesting region is very low-close to zero. Same applies to the CT image image of the thorax. Here, we can see that the left lung region got high speed while the outside regions have low speed. The segmentation of the femur bone could be achieved by increasing the arrival time. However, the speed image was not perfect. This might be due to the pixel intensities as the region of interest and the surrounding regions seem to have same gray shade overall. Besides, it could be difficult to distinguish boundaries between the regions which can make the segmentation a task complex. Other aspect are that we maybe do not know the ground truth, the anatomic definition of the objects and where the objects end. It is even a complex task to have a correct definition of the object in interest.

After performing the segmentation of the three suggested images, it seems to be difficult to finding good starting point(s) where T=0. Moreover, finding suitable speed map (translating intensity and edges) is challenging even after performing pre-processing on the images. The object of interest was selecting based on the arrival time. The arrival time seems to be high for objects that are difficult to define such as the femur bone for example. Since the knee image contains more edges and in general difficult to segment, the time complexity should be high. More nodes are need before finding the optimal path and there are more edges included. The time complexity of this image can be calculated using $O(E + NlogN)$. This problem has a higher time complexity than the other two, heart and thorax. For these two images, the number of nodes should be small since the speed images achieved were good and the algorithm seems to find good criteria.

The algorithm seems to select the shortest path in an optimal sense to reduce the cost. The execution time for calculating the speed image during the femur bone segmentation seems to be short in comparison to the execution time for segmentation heart ventricles and left lung. This is despite the fact that the speed image received from the femur bone segmentation is not perfect.

4.4 Part 4 - Fast marching algorithm 3D

After implementing the 3D fast marching algorithm, the segmentation was performed on the MR image of a carotid. The segmentation is visualized as a three dimensional surface for the right carotid.

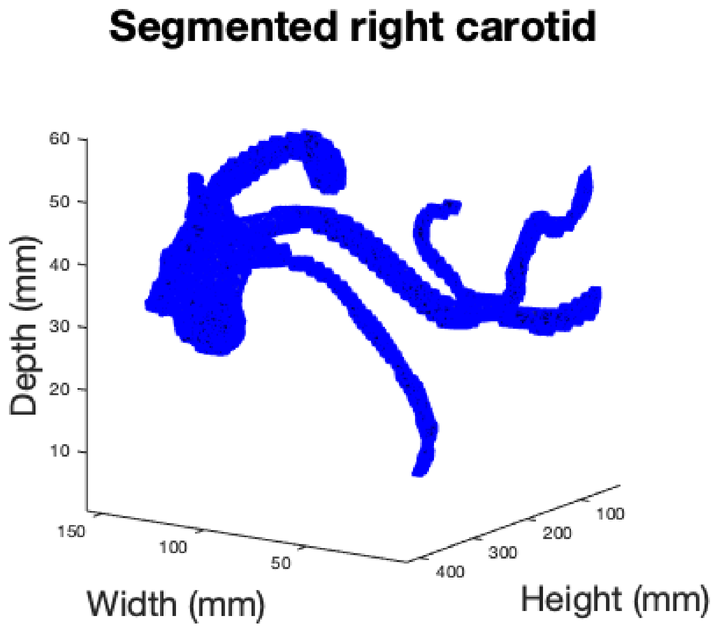


Figure 17: The segmented right carotid visualized as a three dimensional surface.

References

[1] Dieuwertje Alblas, (June 30, 2018), Implementing and Analysing the Fast Marching Method (PDF)

[2] J. Andreas Baerentzen, On the implementation of fast marching methods for 3D lattices, Report Department of Mathematical Modelling, IMM-REP-2001-13, Technical University of Denmark.

[3] S. Godunov, "Finite Difference Method for Numerical Computation of Discontinuous Solutions of the Equations of Fluid Dynamics," Matematicheskii Sbornik, pp. 47-271, 1959, translated from Russian by I. Bohachevsky.

[4] R. Kimmel and J.A. Sethian (1996). Fast Marching Methods for Computing Distance Maps and Shortest Paths.

[5] Medical Imaging in IDL, (March 06, 2007)

https://climserv.ipsl.polytechnique.fr/documentation/idl_help/DICOM_Attributes.html