Centre for Mathematical Sciences, Mathematics, LTH,
November 2023

FMNA30 - Medical Image Analysis, Assignment 2

1 Introduction

The purpose of this assignment is to give hands-on experience with handling DICOM images and fast marching methods. The assignment is structured into four parts, where the last part is optional and required for higher grades.

2 The Rules

The assignment is **handed out** at the lecture **Monday November 13**. The deadline is on **Sunday December 3rd** (23:59 CET). Each student should hand in his or her own individual solution and should, upon request, be able to present the details in all the steps of the used algorithm. You are, however, allowed to discuss the assignment-problem with others. You may also ask your teachers and the course assistants for advice, if needed.

The report. Present your work in a report of approximately three-four A4-pages written in English or Swedish. The report should (at least) contain an introduction, a short description of theory and the methods used, a section with the results and a conclusion where the results are discussed. A well-written report will have a positive influence on the grade you obtain.

Submitting your work. Submit the report as a pdf-file with the name format `assignment_3_yourname.pdf` on the Canvas course page. Don't submit any code. Your code should be made available for inspection upon request during grading or for the oral exam. Everything you wish to communicate should be written in the report.

Data, Matlab code, and c-code for the assignment is given in the file `Assignment-3.zip` and can be downloaded from the course

homepage. For assistance and questions regarding this assignment, please contact `einar.heiberg@med.lu.se`.

3 Part 1 - Loading DICOM data

Review the lecture notes on DICOM and PACS. In this first part of the assignment the task is to complete code for a Matlab based DICOM reader to load the data used in this task. Of note, there is a DICOM reader in Matlab in the Image Processing Toolbox. That loader is quite generic (and slow) and should not be used (other than as possible inspiration in this assignment). As a starting point there are three files provided; `mydicominfo.m`, `mydicomread.m`, `mydicomreadfolder.m`.

- The purpose of the file `mydicominfo.m` is to read DICOM tags from a file. In this file you need to complement the DICOM dictionary with suitable DICOM tags that will be required for future processing of the images (or otherwise useful and relevant DICOM tags). You may therefore need to iterative this task so that you also can solve the part 2 in this assignment. A good list of some of the important DICOM tags are found in the PACS - DICOM lecture handouts. A more complete list of DICOM tags are found in [1]. Test this function on the files `MR-heart-single.dcm`,
  and `CT-thorax-single.dcm`

- The file `mydicomread.m` uses `mydicominfo.m` to read DICOM tags as well as image data. The code needs to be complemented with code that reshapes the image data and rescales the data to true image intensities. A hint on reshaping the data is to use Matlab command `reshape` and `transpose`. If you are unused to using structs in Matlab you may want to review the on-line help on this. Also, look into how Matlab stores images and how images are stored in the DICOM standard (see lecture notes). New DICOM tags are entered in the subfunction `taglist` in the file `mydicominfo.m`. For each DICOM tag you need to specify a type, which corresponds *loosely* to VR. Possible types are `char`, `num`, `uint8`, `uint16`, `single`. As described on the lectures finding out VR is much of a guessing game. Of particular usage is the type `num` as this converts from strings to numeric data that can be used by

Matlab later. This tag is one of the most common for numbers that may be floating point information. See below for hints on how to rescale image intensities. Test this function on the file `MR-heart-single.dcm` and `CT-thorax-single.dcm`.

- The file `mydicomreadfolder` should read all DICOM files in a folder to a 3D image volume. You need to complement this file in order to find out voxel sizes etc and loop over all files in the folder. Test this function on the folder `MR-thorax-transversal`.

Hint on scaling to true image intensities: In this assignment we will work with data from CT scanner. Image intensities in CT are given in Hounsfield scale after Sir Godfrey Hounsfield (who shared the Nobel price with Allan Cormack for the invention of the CT scanner). The Hounsfield scale measures radiodensity and is a linear transformation from attenuation coefficient into one expressed in terms of radiodensity of distilled water. The scale is defined so that air has a HU of -1000 and water has a HU of 0. Soft tissue then get a value of around 100-300. Use appropriate DICOM tags to scale the image intensities to correct values. Please note that it is important that you get the correct value representations (VR) for the rescaling tags, often they are encoded as decimal strings. Do not use the DICOM tags WindowCenter and WindowWidth as they (typically) only are used to define how the window should be displayed, not how the values in the DICOM files should be scaled to get the intended values.

For the report: Document the DICOM tags added to the dictionary and how they are used in the code in a table format. Include value representation (VR) for each DICOM tag. Include the graphical output of the two suggested test images in the report. Describe the scaling of image intensities used. For the 3D DICOM reader document the assumptions (at least some of them) for the loaded DICOM images in the folder. Examples on assumptions would be under which circumstances would the 3D reader fail. You do not need to solve all cases where your 3D reader go wrong but document when it will. When displaying the images, please also use the Matlab command `colorbar` to add a color legend to your scaling. The images should be displayed in a gray scale (hint: `colormap`).

4 Part 2 - Display of medical images

In medical imaging there are three standard general traditional views called coronal, sagital, frontal views as illustrated in Figure fig:planes. In this task you should use your DICOM reader from Task 1 to load a 3D MR thorax image. In the folder `MR-thorax-transversal` there are MR images collected in the transversal plane. The convention within Radiology (the medical specialty that uses imaging to diagnose disease) is to display transversal images as seen from the feet, coronal images from the front, and sagital images from the patient's left side. Images should be presented where the axis of the images should be in the unit mm.

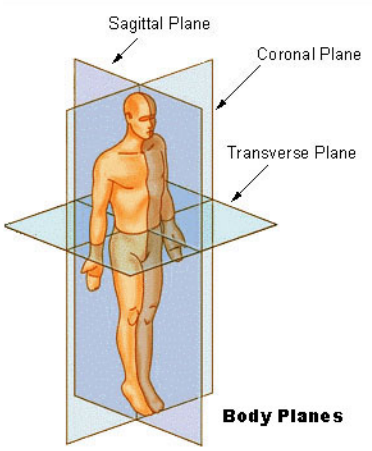

Figure 1: Standard Radiology image planes.

The first task is to display the three standard images in three separate figures with correct labeling. The scaling of the images should be correct and the unit should be in **mm**. Display the most central slice in each of the three views; transversal, coronal, and sagital.

Hint 1: The heart is towards the left side and towards the chest of the body (unless you have situs inversus where the organs are placed mirrored in the thorax which occurs in about 1/10000 individuals). For each output image look at the image content and use this knowledge to determine if the image is correctly displayed or not.

Hint 2: In Matlab you can use the `image(x,y,im)` or `imagesc(x,y,im)`

where `x`, and `y` are vectors specifying locations of pixels centers in the unit mm. See help `image`, or `imagesc` for further details. You also may want to use the Matlab command `axis image` to ensure correct aspect ratio in the display.

Hint 3: To enhance the ticks that gives the size in the figure you may want to use:

```
set(gca,'xcolor',[1 1 1],'ycolor',[1 1 1]);
set(gca,'ticklength',[0.05 0.05]);
```

Hint 4: To potentially make it easier to identify the planes, calculate the physical size of the image box in the three directions Right/Left, Anterior/Posterior, and Feet/Head (in the unit mm). Also ready the Matlab documentation for the `image` command and `dataaspectratio` it may be helpful depending which approch you choose.

The second task is to perform the same thing with the data set `MR-carotid-coronal`, but now instead of displaying the central slice perform Maximum Intensity Projection (in the three specified image planes). This data set contains an image of the carotid vessels (vessels that goes through the neck and supplies the brain with blood).

Hint 1: Please note that the images in this task are in the coronal view in the DICOM files.

Hint 2: Use the Matlab `max(X,[],dim)` syntax to compute the maximum intensity projection along dimension `dim` of the array `X`.

Hint 3: To help you with the anatomy a schematic image of carotid arteries is shown in Figure 2.

For the report: Include all three views for the two data sets in the report. State the size of the image in all three directions (right/left, anterior/posterior, feet/head) in **mm**. In the images also include a color legend as in the previous tasks.

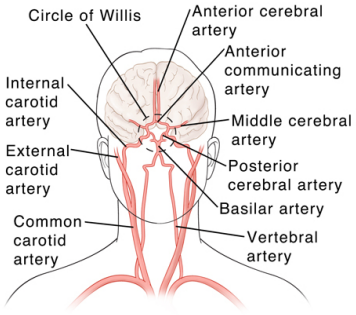

Figure 2: Schematic image of the carotid arteries.

5 Part 3 - Fast marching algorithm 2D

Write a generic 2D interactive image segmentation algorithm. In this task you will get introduced to the notion of making user interfaces in Matlab. The theoretical framework for the fast marching implementation is given in [2] and the practical implementation is detailed in the report [3] and [4]. Both documents are included in the .zip-file in the folder `Documents`. In addition there is also an very fast, but approximate fast marching code. The accurate fast marching code (stencil based) it is implemented both as normal `.m` files as well as `.mex` files.

Last year some students (specific to Matlab version and operating system) had problems where the supplied mex-files did not work. If this happens, try to more all mex-files to a different folder where Matlab can not find them. This will give much slower execution time but will work. Please contact teacher if this seem to be a problem so that we can troubleshoot this more carefully.

Study the function `interactive.m`. When the code is run it open the figure `interactive.fig`. Note that the Mathworks suggested method of writing user interfaces is not followed. The reason for that is that the author of this assignment consider the coding style for user interfaces recommended to be slow and complicated. Instead, a global variable called `GUI` is used. Usually it is considered ugly to use global variables as debugging and tracking data flows may be difficult. The rationale for using global variables in user interfaces is that the user interface itself represent a global state space.

Left click in the user interface sets the seed point for the fastmarching and a **right click** sets the ArrivalTime threshold to the arrivaltime of that pixel. In the user interface there are three slides that **could** be used to tune parameters in your speed map. It is up to your discretion to use any number of them to control how the speed image is generated and how, and if the sliders should be used. You could also use the position of the seed point (and image intensity around that point) as an input for the calculation of the speed image. A key issue in this part of the assignment is to write the speed image function so that the interactive tool can be flexible to segment many different types of images. The speed image is a mapping from intensity (or other features in the image) to the speed of the propagating surface in the fastmarching algorithm. The speed should be high for objects to be included and low for other objects. This is found in the function `calculatespeedimage` in the file `interactive.m`. The fourth slider ArrivalTime is used to control what arrival time that is taken as the threshold for the object. This slider is set as the percentage of the maximum arrival time in the image.

Specific tasks:

- Load the image `MR-heart-single.dcm`. Try to segment the a) left ventricle, b) right ventricle.
- Load the image `CT-thorax-single.dcm`. Try to use your interactive segmentation tool to segment the left lung.
- Load the image `MR-knee-single.dcm`. Try to use your interactive segmentation tool to segment the femur bone. This may likely be the hardest image to segment, and if you get a poor result it is still be acceptable as long as you state why you think that it fails and why it is difficult to segment.

Hint 1: The key is to figure out how to map the input image into a suitable speed image. Where do you want to have high velocity and where do you want to have low velocity? Once you figured out where in the image you want to have low and high speed, try to come up with a simple formula that converts image intensities to suitable speed. Such mapping should ideally use center image intensity which is the intensity of the seed point.

For the report: The focus on this section in the report is on your selection on how to construct the speed image. Use pseudo code or equations to express the speed image. Please discuss how it was designed and especially what was the rationale for the design. What did you want to accomplish? Remember to discuss possible pre-processing of images such as smoothing of the image(s), possible use of edge selectors etc.

Show the segmentation results with both the speed image, and the resulting segmentation (screen shots on the interface works fine). Discuss what is the challenge with the images are and how your definition of the speed image may help to overcome them. Add some short discussion on execution time and time complexity.

6 Part 4 - Fast marching algorithm 3D (extra for high grades)

The purpose of this part is to explore three dimensional segmentation using fastmarching. For this task write automated code to load the images and perform the segmentation. Initialization may either be implemented as image specific (i.e. hard code starting coordinates) or using an automated algorithm that initiates the fast marching algorithm. Perform a segmentation on one or both the carotid arteries in the data set `MR-carotid`. The starting position of the fast marching may be hard-coded in your script. The segmentation should be visualized as a three dimensional surface. Hint on visualization is to look at the matlab commands `isosurface`, `lighting`, `imshow3d`, and `patch`. Especially look at the example code provided in the help text for these three functions. You may also want to use `cameratoolbar` to rotate your images. It is suggested to use the provided function fastmarch that takes a cost image, start index, and finally the maximum cost as a termination condition. The latter can be set to 1e9 or any high number. Note that cost image and start index should be singles. Start index can be computed by usage of the function sub2ind. Note that the input image is **cost image**, i.e 1/ speed image.

Hint: It is sometimes helpful to first construct an own digital phantom where you know the geometry and the shape, that helps you better understand the visualization and look for errors in a known

geometry.

For the report: This part is only needed for students seeking a higher grade. Present a 3D image of at least one of the two carotids. Document process of finding seed point and your chosen speed function.

For higher grades part 4 should be implemented and parts 1-3 should be well documented and motivated in the design.

References

[1] https://exiftool.org/TagNames/DICOM.html. Last accessed on 2019-12-02.

[2] M. Sabry Hassouna, Aly A. Farag, Multistencils Fast Marching Methods: A Highly Accurate Solution to the Eikonal Equation on Cartesian Domains, *IEEE Transaction on pattern analysis and machine intelligence*, vol. 29, issue 9, 2007.

[3] 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.

[4] http://www.mathworks.com/matlabcentral/fileexchange/24531-accurate-fast-marching. Last accessed on 2019-12-02.