Artificial Intelligence and Image Processing

Introduction:

In this assignment, there will be a development of image processing methods using MATLAB software to automatically segment and measure the volume of the lung, which the assumption will be of 0.75mm px * 0.75mm px. There will also be a short report which includes that methods taken place (MATLAB functions), intermediate (results of each steps), and final result. Also, further discussions and conclusion will be included, as well as the final printed programming codes from MATLAB.

1. Thoracic CT Angiography

The aim of this project is to develop image processing methods using MATLAB to automatically segment and measure the volume of lung (assumption: pixel size is 0.75*0.75 mm).

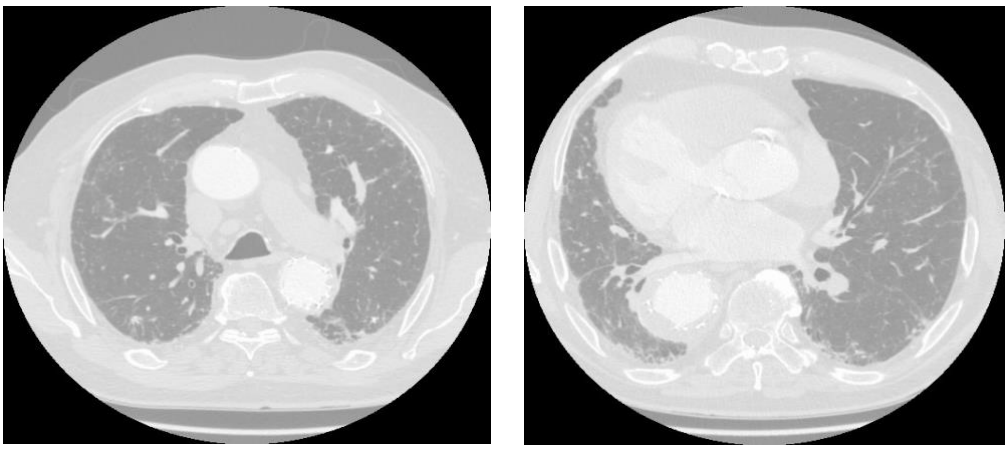

Figure 1: The above two images are used to segment and measure the volume of the lung.

Firstly, the software needs to get path to the images.

```
%% Get path to Images
cwd_ = pwd;
path_dataset = cwd_+"\dataset_lungs\"
```

By this function the images are obtained and are ready to be developed.

The next step of the process is to read the images added to the software. This is done by the **A = imread** function. This function allows the software to read weather the image is a greyscale image or a colour image.

```
%% Read image

lung_img1 = imread(path_dataset+"lung1.jpg");
```

After reading the image the image is then converted to greyscale. This is done by the **rgb2gray** function. This function helps the software to convert the RGB based image to the grayscale image.

```
%% Convert to Grey scale
lung_img1_grey = rgb2gray(lung_img1);
imshow(lung_img1_grey,[])
```

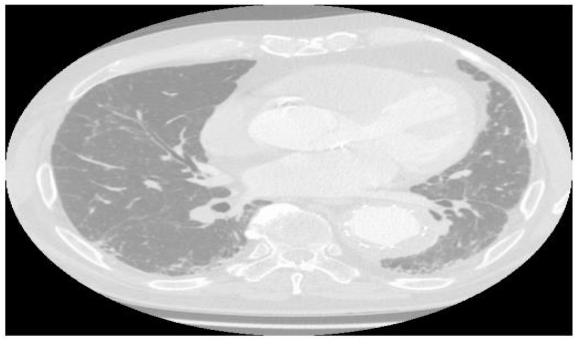

Figure 2 : result of the step taken

After converting the image to the greyscale. The OTSU method is used to find the level automatically to binarise. This method works when the image is bimodal. I have also used the **greytresh** function to convert the multidimensional arrays to 2D array.

The level = 0.2941.

```
%% From OTSU method find the level automatically to binarise

level = graythresh(lung_img1_grey)

% Convert the image into a binary image using the threshold.

thres_img1_lung_B = imbinarize(lung_img1_grey,level);

imshowpair(lung_img1_grey,thres_img1_lung_B,'montage')
```

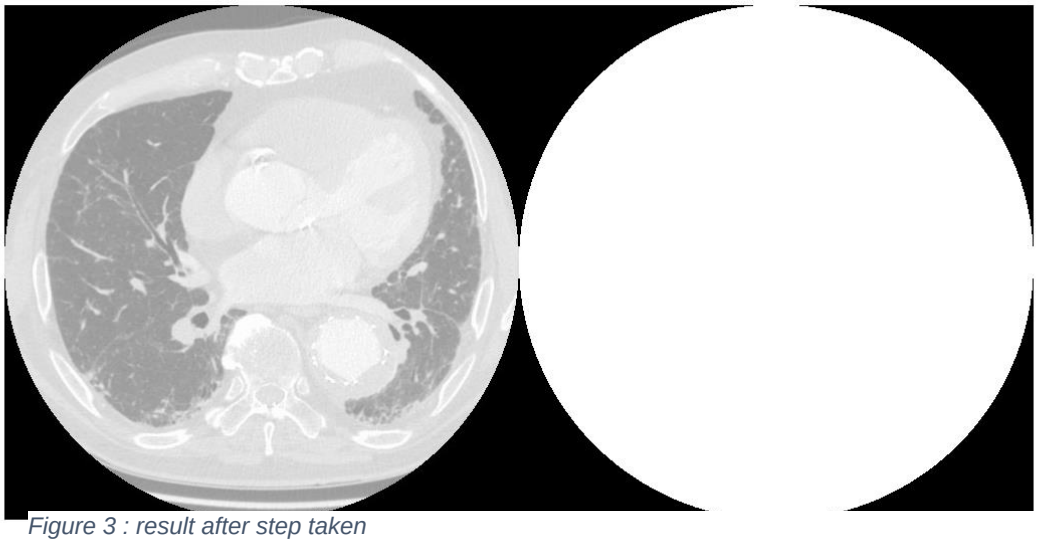

Figure 3 : result after step taken

As the automated method using OTSU is not giving a good method, the adaptive threshold method is used.

```
%% adaptive threshold

T = adaptthresh(lung_img1_grey);
% Convert image to binary image, specifying the threshold value.

thres_img1_lung_B = imbinarize(lung_img1_grey,T);
% Display the original image with the binary version, side-by-side.

figure
imshowpair(lung_img1_grey, thres_img1_lung_B, 'montage')
```

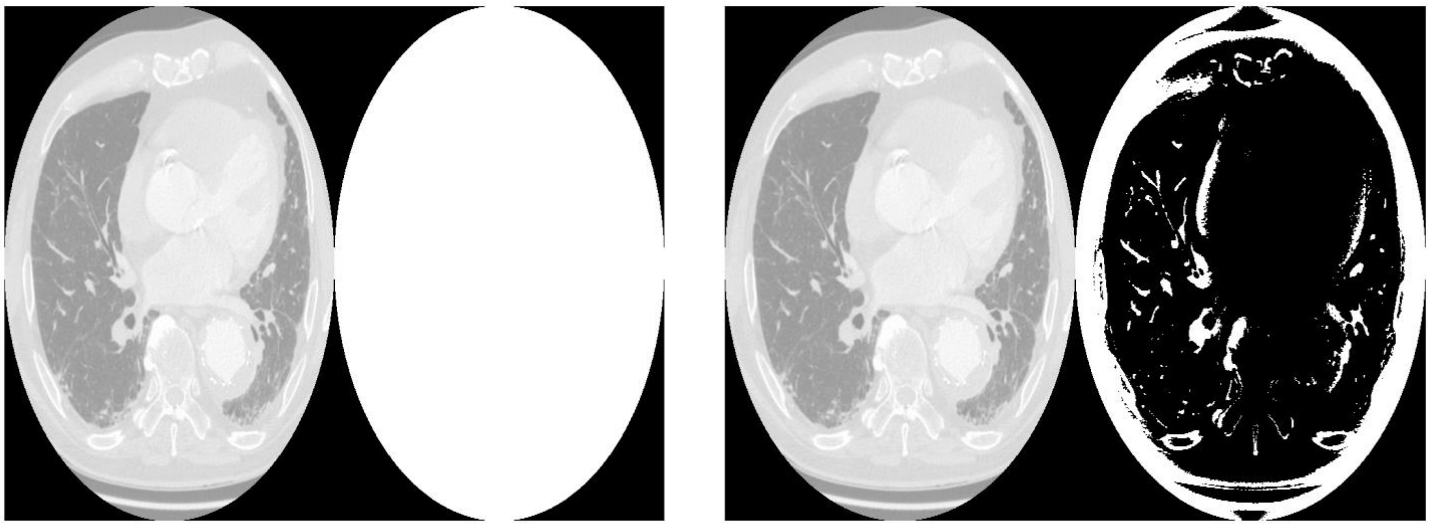

Figure 4 : result of the step taken

There is also a histogram analysis of the program development which is done through the image processing ToolBox. The function used for this is the **imhist** function which lets the software to create a histogram which represents the different range of data and then calculate the number of px's inside the range.

```
figure, imhist(lung_img1_grey);
% requires Image Processing Toolbox.
% Error in image_analysis_lung (line 21)
figure, histogram(lung_img1_grey);
```

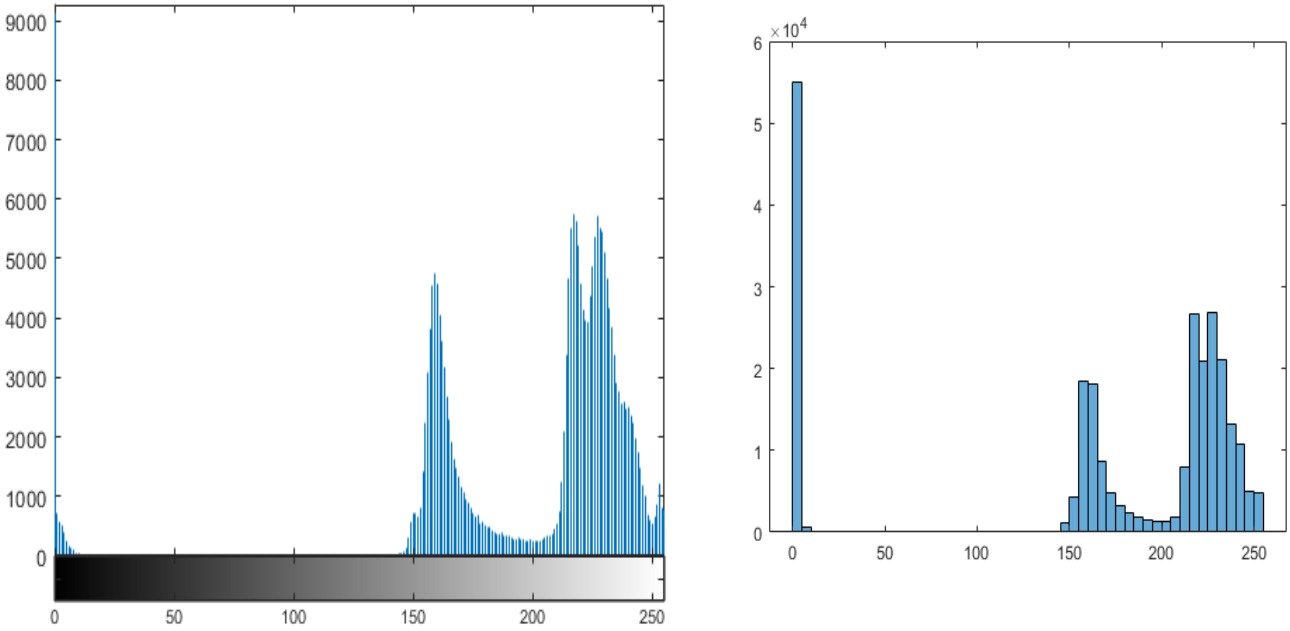

Figure 5 : result of the step taken

As there is an error in the threshold is then set to 200 => 200/256. From the histogram there are three minimums in the histogram, less than. First low is at 50 and there is another low at 200. Then the **imshow** function is used to display the value low as black and it also shows the value high as white. The display range has been specified in the function.

```
% from the histogram there are three mimimumns in the histogram. less than
% 50 and then at 200 there is a low

thres_img1_lung_B =  im2bw(lung_img1_grey,50/256);
figure, imshow(thres_img1_lung_B, []);

% The OTSU method may taken 50 as threshold

thres_img1_lung_B =  im2bw(lung_img1_grey,203/256);
figure, imshow(thres_img1_lung_B, []);

figure, imshowpair(lung_img1_grey,thres_img1_lung_B,'montage');
```

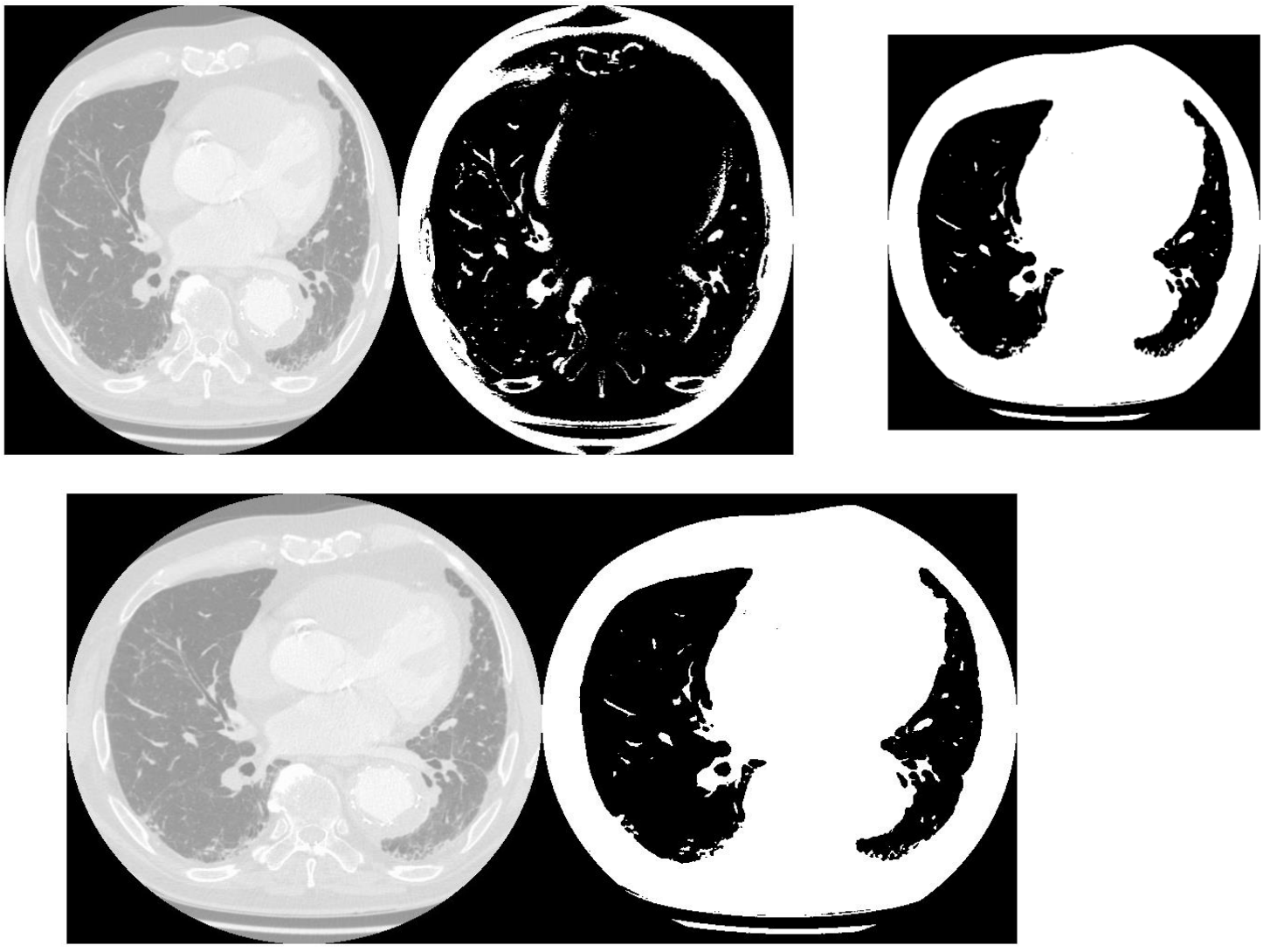

Figure 6 : result after step taken

After this there is a connect component analysis for region of interest selection. For this the **bwlabel** is used. This function gives the label matrix of the image which includes labels for the 8-connected objects which are found in BW. This method also requires parallel computing toolbox.

```
conn = 8;

labeledImage = bwlabel(thres_img1_lung_B,conn);

blobMeasurements = regionprops(labeledImage, lung_img1_grey, 'all');
numberOfBlobs = size(blobMeasurements, 1);

allBlobAreas = [blobMeasurements.Area];
[values_ar,inx_ar] = sort(allBlobAreas,'desc');
```

```
%%
allBlobsImage = ismember(labeledImage, inx_ar(1));
imshow(allBlobsImage)
```

Figure 7 : result of the step taken

After each blob is assigned to different colour. This is done by the **label2rgb** function, which sets the colour to set to each object which is based on the number og objects in the label matrix. This also gets the colour from the range as a whole on the colour map.

```
%%
% assign each blob a different color
coloredLabels = label2rgb (allBlobsImage, 'hsv', 'k', 'shuffle'); % pseudo random color
imshow(coloredLabels);
```

```
%%

Area = sum(allBlobsImage(:))*0.75*0.75
```

This image is the final piece of the lung development. However, the result is not correct and the or as accurate it can be, the sum of the are was set to 0.75 * 0.75 pixels and the whole area of the image is 7.7406e+04. All the methods were used to help in order to complete the aim of the project, which was to automatically segment and measure the volume of the lung in assumption of the pixel size being 0.75*0.75 mm.

MATLAB printed programming code:

```
%% Start the editor by clearing workspace
close all
clear all
clc
%% Get path to Images
cwd_ = pwd;
path_dataset = cwd_+"\dataset_lungs\"

%% Read image

lung_img1 = imread(path_dataset+"lung1.jpg");

%% Convert to Grey scale
lung_img1_grey = rgb2gray(lung_img1);
imshow(lung_img1_grey,[])

%% From OTSU method find the level automatically to binarise

level = graythresh(lung_img1_grey)

% Convert the image into a binary image using the threshold.

thres_img1_lung_B = imbinarize(lung_img1_grey,level);

imshowpair(lung_img1_grey,thres_img1_lung_B,'montage')

% Itseems the automated method usig OTSU is not giving good result
%% adaptive threshold

T = adaptthresh(lung_img1_grey);
% Convert image to binary image, specifying the threshold value.

thres_img1_lung_B = imbinarize(lung_img1_grey,T);
% Display the original image with the binary version, side-by-side.

figure
imshowpair(lung_img1_grey, thres_img1_lung_B, 'montage')

% not giving good result
%% lets do the Histogram Analysis

figure, imhist(lung_img1_grey);
% requires Image Processing Toolbox.
% Error in image_analysis_lung (line 21)
figure, histogram(lung_img1_grey);


%% Set threshold to 200 => 200/256

% from the histogram there are three mimimumns in the histogram. less than
% 50 and then at 200 there is a low

thres_img1_lung_B =  im2bw(lung_img1_grey,50/256);
```

```
figure, imshow(thres_img1_lung_B, []);

% The OTSU method may taken 50 as threshold

thres_img1_lung_B =  im2bw(lung_img1_grey,203/256);
figure, imshow(thres_img1_lung_B, []);

figure, imshowpair(lung_img1_grey,thres_img1_lung_B,'montage');



%% Connected component analysis for region of interest selection
conn = 8;

labeledImage = bwlabel(thres_img1_lung_B,conn);

blobMeasurements = regionprops(labeledImage, lung_img1_grey, 'all');
numberOfBlobs = size(blobMeasurements, 1);

allBlobAreas = [blobMeasurements.Area];
[values_ar,inx_ar] = sort(allBlobAreas,'desc');

%%
allBlobsImage = ismember(labeledImage, inx_ar(1));
imshow(allBlobsImage)


%%
% assign each blob a different color
coloredLabels = label2rgb (allBlobsImage, 'hsv', 'k', 'shuffle'); % pseudo
random color labels
imshow(coloredLabels);

%%

Area = sum(allBlobsImage(:))*0.75*0.75
```