Characterization of Emission pollutants

Content page

Aims and Objectives 3

Intro 3

MATLAB 3

What is digital image prcessing 4

Different level of Processing 4

Coordinate convention 5

Image and class types 5

MATLAB functions 6

Image processing functions 6

1. Aim & Objectives

The aim of this project is to identify the different function that can be used in MATLAB which can be used for digital image processing. Understand each function then apply each function, compare the result and find the best sequence of function applied together to give the best result which can be used to determine the morphology(shape and size) of particles in a pollutant emission.

The outcome of the projects are written below:

* Identify different functions in MATLAB that can be used to digitally process micro-graphs.
* Understand each of the function you have identified and understand what the function is doing.
* Apply each function to a micro-graphs and obtain the results for each function.
* Compare are the results with one another to see which function gives the best result and then apply two functions together and then three functions together to see which sequence or series give the best result.
* After this the micrograph should be clear enough so that the morphology of the particles can be measured, then depending on size and shape, an estimate of how dangerous the particle is can be worked out.

2.1 Intro

This interim report will contain details of the progress in this project so far. Also details about the understanding which has been developed so far about the project.

First of all, what is a digital image processing and how is an image represented by MATLAB will be discussed, followed the different function in MATLAB which we will be interested in will be discussed.

2.2 MATLAB

MATLAB is short term for Matrix Laboratory. The basic component of MATLAB is Matrix. This make is easy to solve a lot of technical computational problems in no time. MATLAB was initially design to give an ease of access when dealing with matrix and linear algebra software. Whereas before to deal with matrix and linear algebra a FORTRAN programme is required to write. But today this is not the case MATLAB has evolved into becoming a main stream source when requiring a numerical computational software.

The Performance of MATLAB cannot be compare with others, the reason behind this is the many techniques that are integrated into MATLAB. Computation, visualization and programming are all in MATLAB and can be used to carry out the following;

* Math and computation
* Algorithm development
* Data acquisition
* Modelling, simulation, and prototyping
* Data analysis, exploration, and visualization
* Scientific and engineering graphics
* Application development, including building graphical user interfaces

MATLAB is used in both Universities and in the industry as well. In university MATLAB is used in many different sectors which include Engineering, Mathematics, and Science. In industry MATLAB is used for Research development and carrying out analysis.

MATLAB contains application specific TOOLBOXES. Which includes the image processing toolbox. This toolbox contains the collection of MATLAB functions which has the ability to solve image processing problem.

2.3 What is Digital image Proccessing

An image is a 2D (dimensional) function. Which can be represented by the function f of variable x and y, like this F(x,y), these variable x and y are the coordinate. The gray level of an image at any point or any coordinate is the amplitude of the function f(x,y). For an image to be a digital image the coordinate (x,y) and the function f must all have limited in size, they cannot be infinite and each the values must be individually separate and different from one another. When these requirements are completed this is when an image becomes a digital image.

For an image to become a digital image it needs to be digitized. This involves sampling of the coordinates of the image. Sampling is the digitizing process of the coordinate. Quantization is the digitizing of the amplitudes if the function f(x,y).In MATLAB a colour image is represented in RGB individual colours. A colour image is made from three individual images, a red, green and blue image. These individual images are known as monochrome images. If colour images are made from three monochrome images therefore any monochrome functions in MATLAB can be applied to a colour image.

2.4 Different level of Processing

There are different level of digital image processing; low level, medium and high level image processing. Low level image processing consists of removing/decreasing noise in an image, contrast enhancement, image sharpening. Low level processing is recognised by the fact that the input is an image and the output is also an image. Medium level processing consist of segmentation, description of these segments, classification of these individual objects. This level of processing is recognised by the fact the input is an image and the output is extracts of images. Finally high level of image processing include analysing images and working out what the image is, this sort of processing is trying to get close to the human vision. During this project, low and medium level of processing will be used.

2.5 Coordinate Convention

There is a specific coordinate system used in MATLAB when dealing with image processing. A matrix is obtained from the results of sampling and quantization. This is the matrix of real value. In MATLAB the origin is now as (1,1) or (A,B), A is representing the row and B is representing the columns. The next coordinate is represented by (1,2) which represents, row 1 and column 2, so this the coordinate convention used in MATLAB. It must be noted that these coordinates are only for MATLAB and should not be considered real or physical coordinates.

A digitalized image will be shown in the manner below;

F =

The dashes lines on the side represent a long [ open bracket and a long close Bracket ] .

| f(1,1) f(1,2) ……… f(1,N) |

| f(2,1) f(2,2) ………. f(2,N)|

| : : ……… : |

| : : ……… : |

| f(M,1) f(M,2) …….f(M,N) |

The above represents a matrix. Where M is the maximum rows and N is the maximum columns. These matrices are stored in MATLAB as variables, these variable contains letters, numbers and underscores.

2.6 Image and Class types

In Matlab the Image processing toolbox supports four types of images, these are listed below.

* Gray-scale image
* Binary images
* Indexed images
* RGB images

There are various classes in MATLAB and the Image processing toolbox that are used to represent pixel values. A image is referred to as being ‘Class image-type image’, for example unit8 gray-scale image. This provides us with the information of the image, that the image is with unit8 class pixels and it is a gray-scale image. In the table below is the list of the various classes.

|  |  |
| --- | --- |
| Name | Description |
| double | Double-precision, floating-point numbers in the approximate range ±, 8 bytes per element. |
| single | Single-precision floating-point numbers in the approximate range ±, 4 byte per element. |
| unit8 | Unsigned 8-bit integers in the range of (0,255), 1 byte per element |
| unit16 | Unsigned 16-bit integers in the range (0,65535), 2 bytes per element |
| unit32 | Unsigned 32-bit integers in the range (0,4294967295), 4 bytes per element |
| int8 | Signed 8-bit integers in the range (-128, 127), 1 byte per element |
| int16 | Signed 16-bit integers in the range (-32768,32767),2 bytes per element |
| int32 | Signed 32-bit integers in the range(-2147483648,2147483647),4 bytes per element |
| char | Characters (2 bytes per element) |
| logical | Values are 0 or 1 (1 byte per element) |

Apart from the bottom two classes are the rest are classified as numerical classes. Some of the above classes are used more often than others, which are never used or used very less. The classes that are used less often or never used include; int8, int32 and unit 32. On the other hand the classes that used more often include unit8 and logical. These are used when dealing with a TIFF or JPEG. In these two classes 1 byte is used to represent one pixel.

2.7 MATLAB Functions

Below are lists of some of the functions discovered so far, these functions are used in MATLAB to carry out different Image processing jobs.

|  |  |
| --- | --- |
| Function | Discription/function |
| clc | Used to clear the command window |
| clear | Used to clear workspace |
| Close all | Used to close all the previous work |
| Imread(‘filename’) | Used to import an image to Matlab |
| Imshow(f) | Used to show image in a figure |
| Figure, imshow(g) | Used to open another figure to show g instead of replacing previous figure f with g. |
| A(2:3,1) | This means row 2 and 3, get the first column |
| i./A | Inverse of each and every element |
| Flipud(x) | This will flip the matrix up and down, a mirror effect will take place across the middle row. |
| X’ | To get the transpose of the matrix |
| a(3,:) | Entire third row |
| a(2,4) | Row 2 and column 4 |
| a(12) | Index number 12 |

2.8 Image Processing Function

In this part I will list some of the more important functions identified which will be discussed further in more detail and used in MATLAB. They are listed below:

* Image segmentation using point, line and edge detection
* Image segmentation using thresholding
* Image segmentation using the Watershed transform

As well as this some of the filters in MATLAB are of interest and these will be looked at in detail as well, these filters include Low pass filter, medium pass filter and finally high pass filter.

This is what has been understood and looked at so far. All the above details have been obtained from study ‘Digital image processing using MATLAB’ by Rafael C. Gonzalez, Richard E. Woods and Steven L.Eddins.

The outcome of this project was to segment an image using MATLAB. This image given was a micrograph of emission from an exhaust pipe. The main objective was to get rid of the noisy background and remain with only the particles.

There are many different type of segmentation techniques which can be used segment images but all of them cannot be used to segment every image. The techniques that have been looked at in this project are; image segmentation using watershed transform and segmentation using thresh-hold values.

The different functions that can be used to modify micrographs are listed below with their functions mentioned.

I=rgb2gray(fig) : The job of this function is to convert an RGB image to a grayscale image.

hy = fspecial('sobel') : The function of fspecial job is to create a 2D linear filter. The type of filter that is create depends on what is put inside the brackets. Sobel creates a 3 by 3 filter which approximates the vertical gradient.

hx = hy' : This function is used to transpose the vertical filter so that a horizontal filter is created, which can approximate the horizontal gradient.

Iy = imfilter(double(I), hy, 'replicate') : This function filters the image by with the mask being hy and the replicate, extends the size of the image by replicating the out boundaries of the image. The ‘double(I)’ gives double-precision, floating point numbers and uses 8 bytes per element.

gradmag = sqrt(Ix.^2 + Iy.^2) : This function combines the result of two filters into one.

L = watershed(gradmag) : This returns a label matrix L that identifies the watershed regions for the input in the bracket. These can be the catchment basins or the watershed ridge lines. These lines are where two catchment basin meet each other.

Lrgb = label2rgb(L) : This function converts the label matrix of the watershed to an RGB for better visualisation.

se = strel('disk', 20) : strel is the function for creating a structuring element and disk is a type of structuring element. Disk is the shape of the structuring element, which is a flat circle with a radius R. This Disk is a matrix, the 20 shows the size of the disk, how big should the disk be. The radius R is the distance from the middle of the matrix to the outs side.

0 1 0

1 1 1

0 1 0

The matrix above is the structure element disk.

Io = imopen(I, se) : The function of this structure is to perform the morphological function opening. Opening is erosion followed by dilation. Erosion is a morphological process in which takes away material. The image below shows how erosion works.

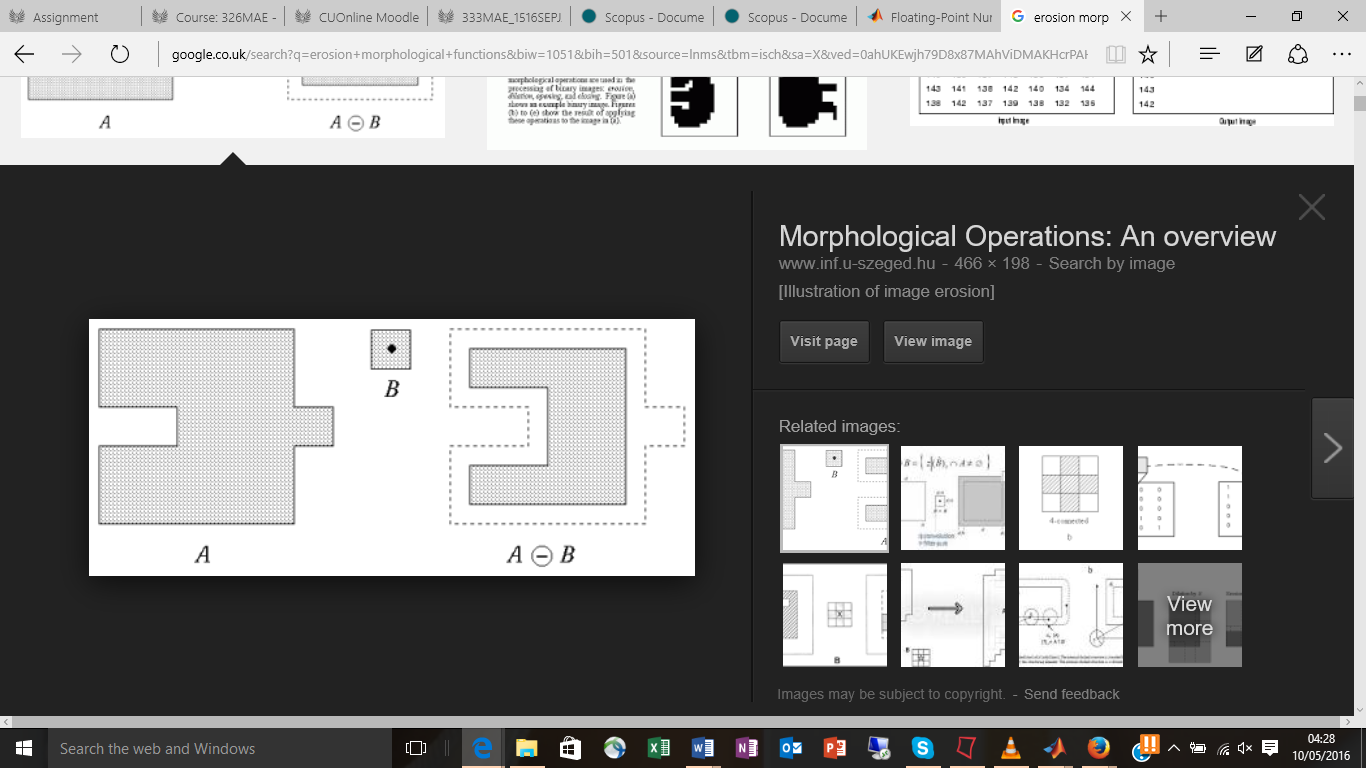


Fig.1 Erosion (Perterlin 1996)

In the image above B is the se (structuring element), and A is the shape that will be going to go through the process of erosion. Imagine the structuring element, B being dragged inside A, the shape that we will be left with will be the one on the right hand side of B. This is called erosion.

Dilation is the opposite of erosion. In erosion the structuring element B is dragged around A on the inside of the edges of A but in dilation the structuring element B is moved around the outside of the edges of A. This is shown in the figure below.

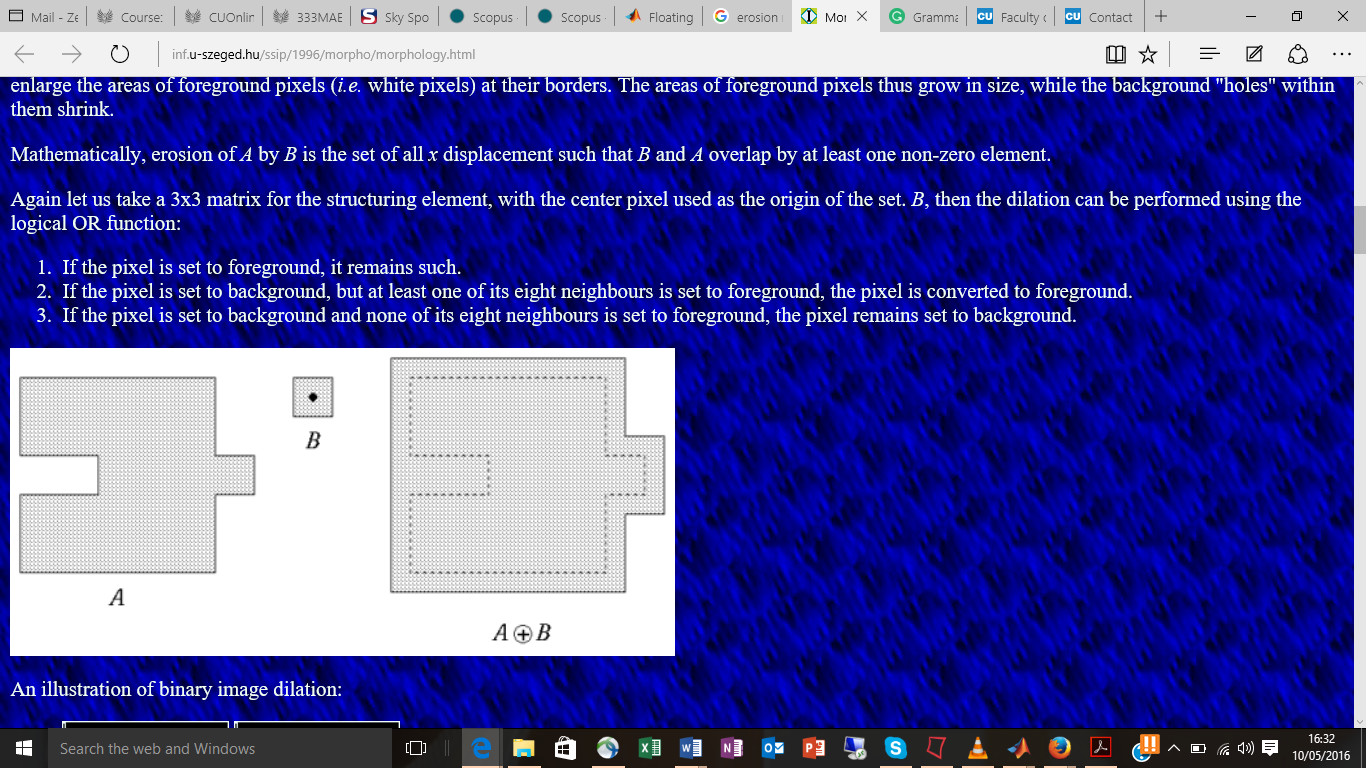


Fig.2 Dilation (Perterlin 1996)

Ie = imerode(I, se) : This function does erosion of the input image ‘I’ using the structuring element se.

Iobr = imreconstruct(Ie, I) : This function carries out morphological reconstructions which is repeatative dilation until the maker image fits unders the mask images. In this function ‘le’ is the marker image and ‘I’ is the mask image. In this function the peaks of the marker image are spread out. The marker image is constraint by the mask image.

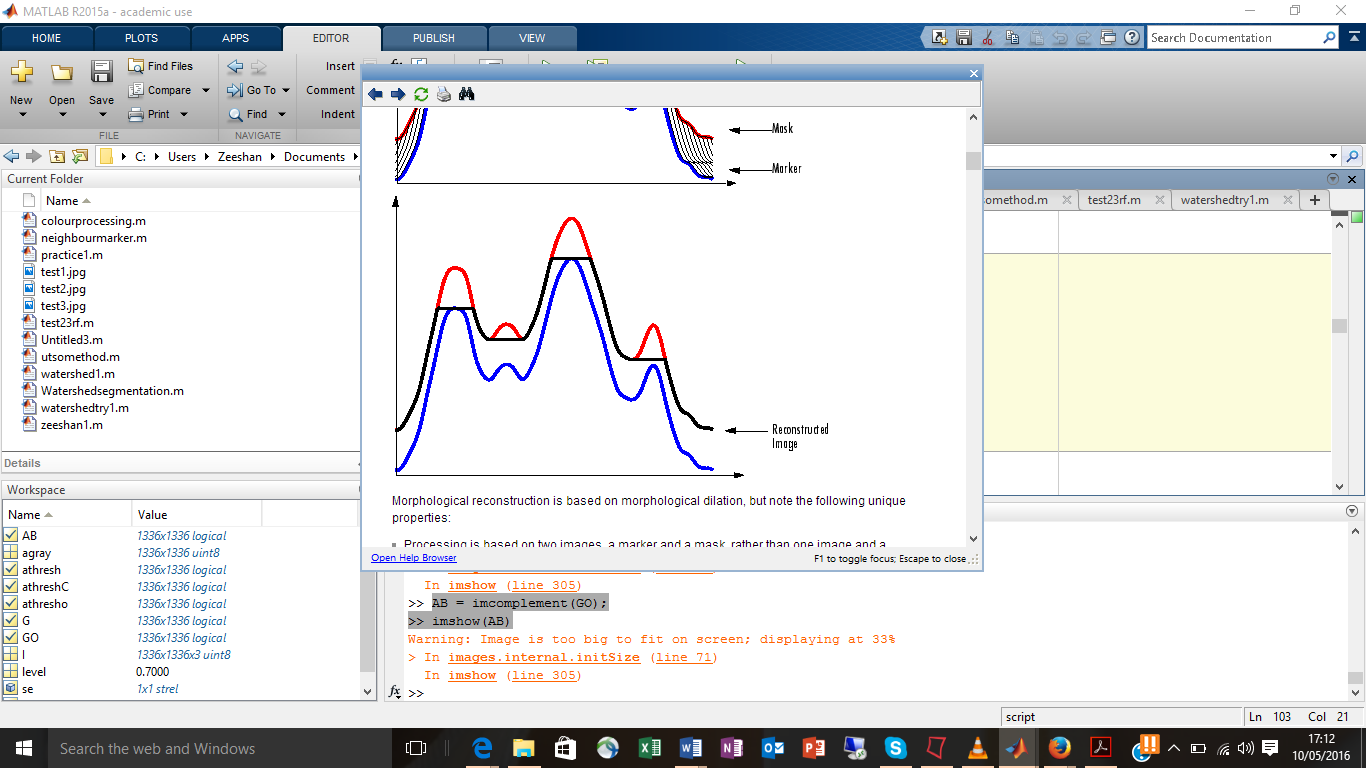
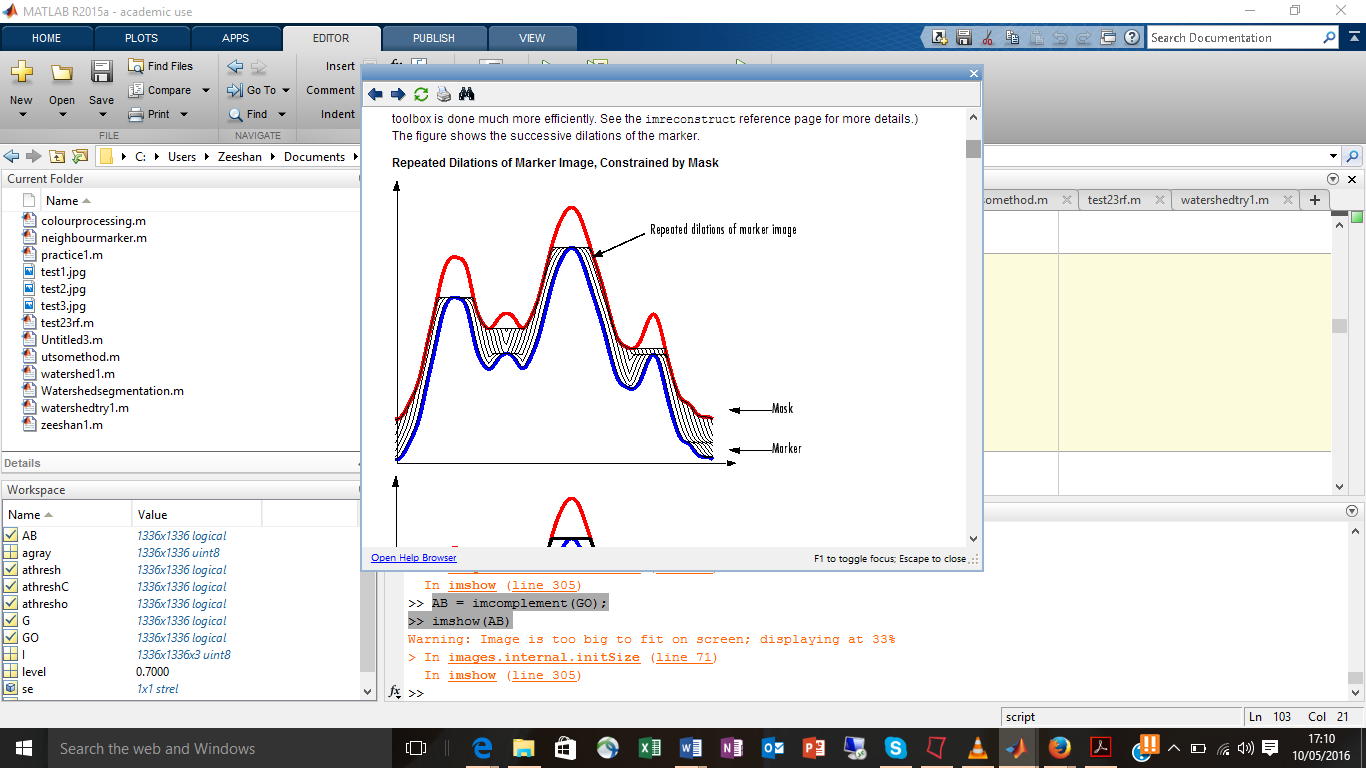


Fig.3 Repeative dilation (MATLAB 2015)

Fig.4 Reconstruction (MATLAB 2015)

The figures above show how the function above works. It can be seen that the mask acts as a constraint to the marker.

Ioc = imclose(Io, se): This function is the opposite of imopen. This function performs dilation followed by erosion, whereas, imopen performs erosion followed by dilation.

Iobrd = imdilate(Iobr, se): This function performs dilation, which has been explained earlier.

Iobrcbr = imcomplement(Iobrcbr): The job of this function is very simple. This function turns everything from black to white and everything white to black.

fgm = imregionalmax(Iobrcbr): This function helps to find the regional maxima’s, which are that markers for the particles which needs to be extracted.

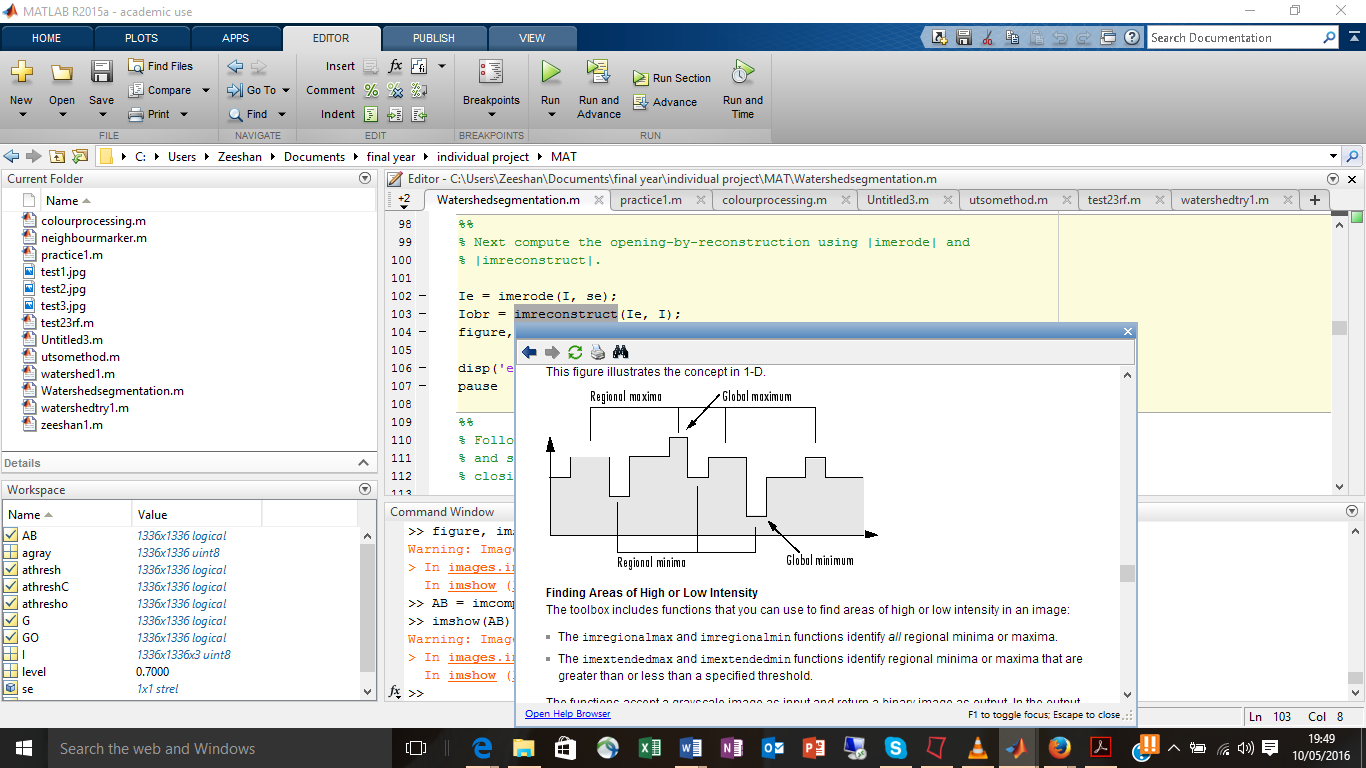


Fig.5 Max and Min regions (MATLAB 2015)

The figure above illustrates the concepts of peaks and valleys in a picture. The peaks are the high intensity and the valleys are the low intensity in an image. The above function is used to find these maximum regional intensities.

I2 = I:

I2(fgm) = 255:

The above two functions are used to superimpose the markers that has been found, on to the original image. Superimpose means to lay something on top of another so that both are still evident.

fgm4 = bwareaopen(fgm3, 20): the function of this code is to remove all the connected blobs that have an area less than a certain amount of pixels. In this case the 20 is the number of pixels, and ‘fgm3’ is the input image. This function is known as area opening.

bw = im2bw(Iobrcbr, graythresh(Iobrcbr)): This image converts an image to binary image. The outcome of this picture only has 1(white) and 0(black) in the image. The 1’s come from any of the intensity in the image greater than the level. In this case the level is ‘graythresh(Iobrcbr), anything above this thresh would be a 1 and anything below this would be a zero. If the graythresh(Iobrcbr) is not included in the code than it automatically take the value as 0.5.

D = bwdist(bw): This function computes the Euclidean distance transform of the binary image bw. This Euclidean distance transform find a value for every pixel in that image. This value is the distance between the pixel and the nearest non-zero pixel.

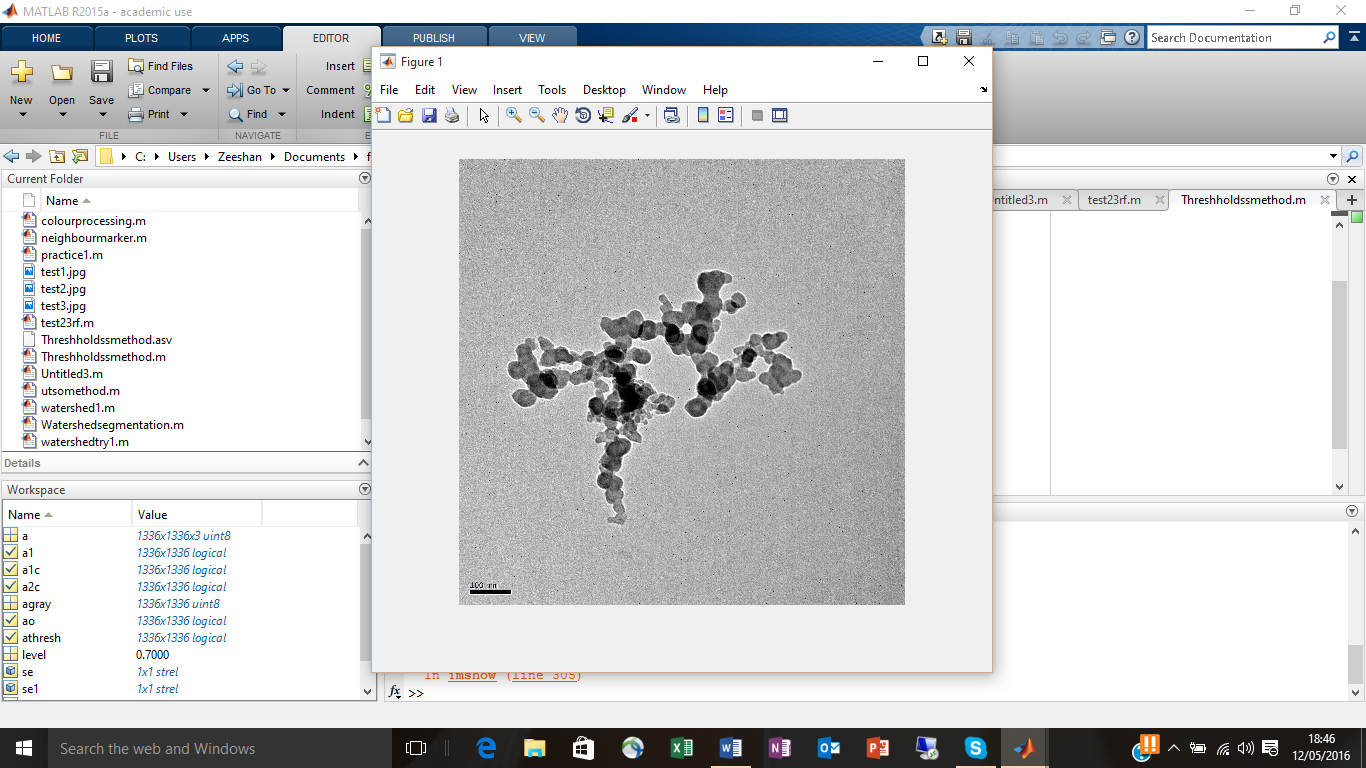
gradmag2 = imimposemin(I,BW): This function modifies the intensity in ‘I’ using the Morphological reconstruction ‘BW’ by only having regional minima’s where ever BW is a nonzero. For this BW and I must be the same size.

**Methods of image segmentation in MATLAB:**

The first method for segmenting the wanted particles from the background was developed was using threshold method and then getting using other methods to get rid of the remaining background noise. Below the code will be displayed, followed by the output and effect of it on the image.

Step 1: The first step is to import the image into MATLAB. [a=imread('test3.jpg')]

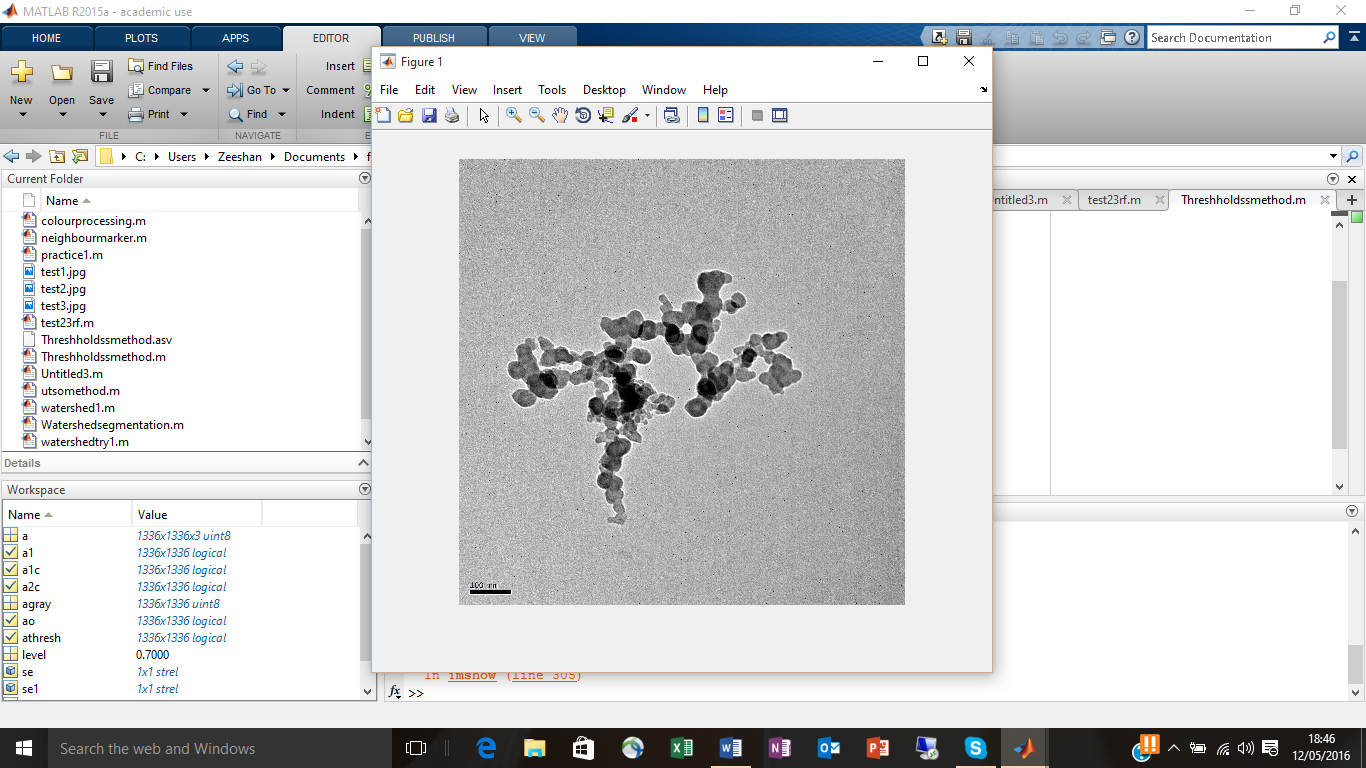
Using imshow(a), this picture will be shown.



The above image will be segmented, the particles needs to be left alone and the background removed.

Step 2: convert the image to a grayscale so that it only contains intensity from 0 to 1 only. This is done by the following function. [agray = rgb2gray(a)]

Below a display will be shown. This will be exactly the same picture as before because no change has be done to the actual picture.



Step 3: Using a threshold level between 1 and 0. The picture can be segmented according to the intensity, anything above the level will be white anything below the level will be white. This will leave the picture to only have black and white.

level = 0.7;

athresh = im2bw(agray,level);

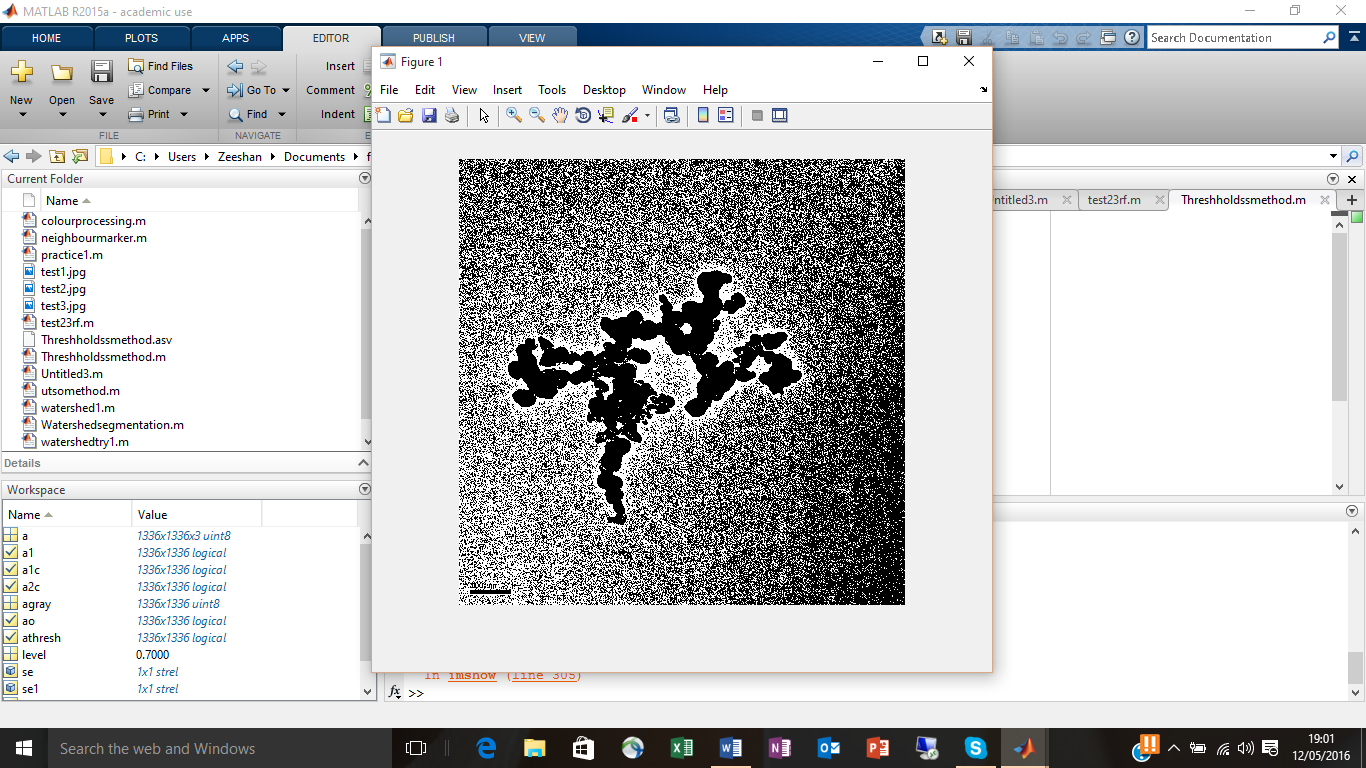


Image A

The above is the image that we are left with after applying that function.

The effect of change the level will be shown below.

Level = 0.5

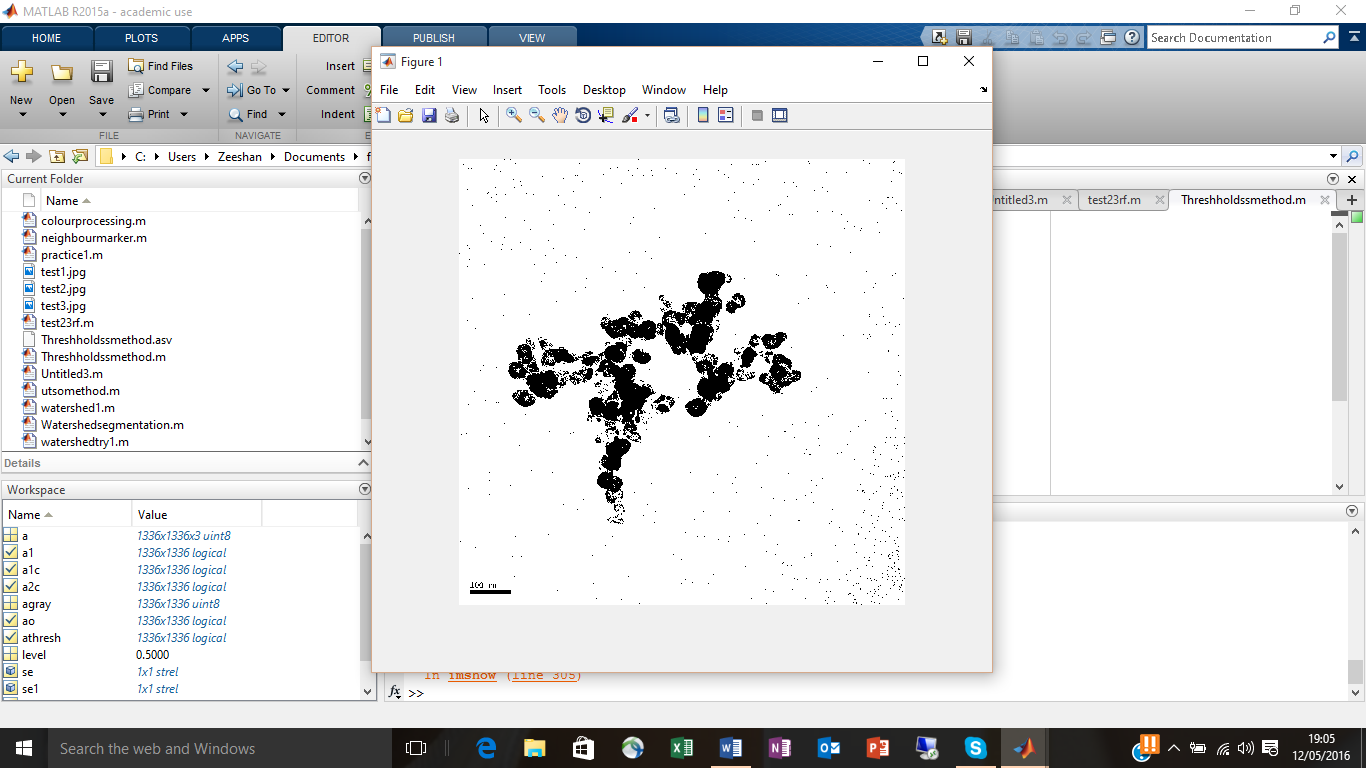


Image B

As can be seen a lot of the blackness has disappeared, this is the effect of reducing the kevel value. Therefore, increasing the level will increase the blackness in the image. This can be seen in the image below. The image below has a level of 0.9.

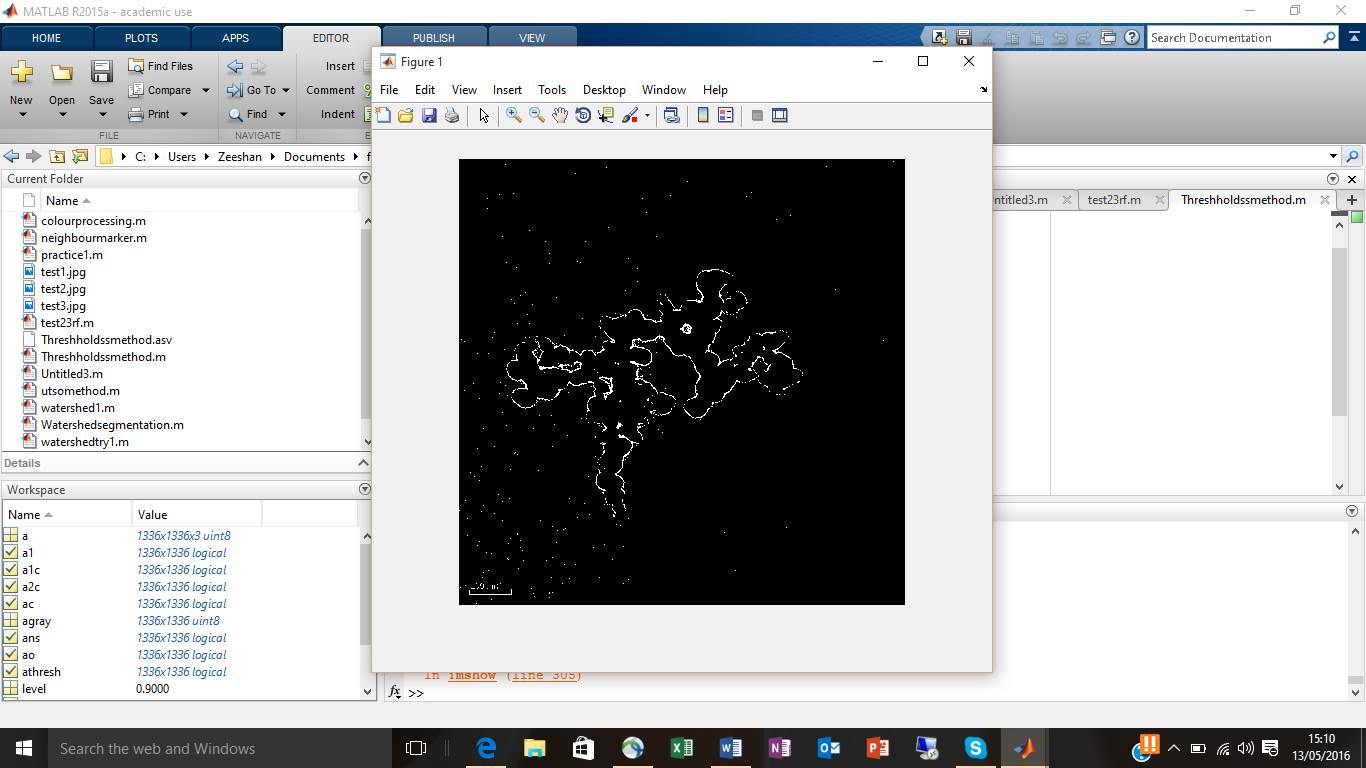


Image C

Step 4: The next step is to try and get rid of noise that is still in the background. This can be done using Morphological functions such as opening and closing. Below image A will be modified using morphological function. This process is long as the correct function needs to be created, this function will take in consideration of how much detail is loss from the particles that we want.

For this step we will create different size of structuring elements. The code for this is listed below:

se = strel('disk', 20);

se1= strel('disk', 10);

se2= strel('disk', 6);

se3 = strel('disk', 8);

Four different types of structuring elements are created. This can be seen above.

Using the morphological function opening to modify image A. The result can be seen below.

ao = imopen(athresh, se)

using the structuring element se to modify image.

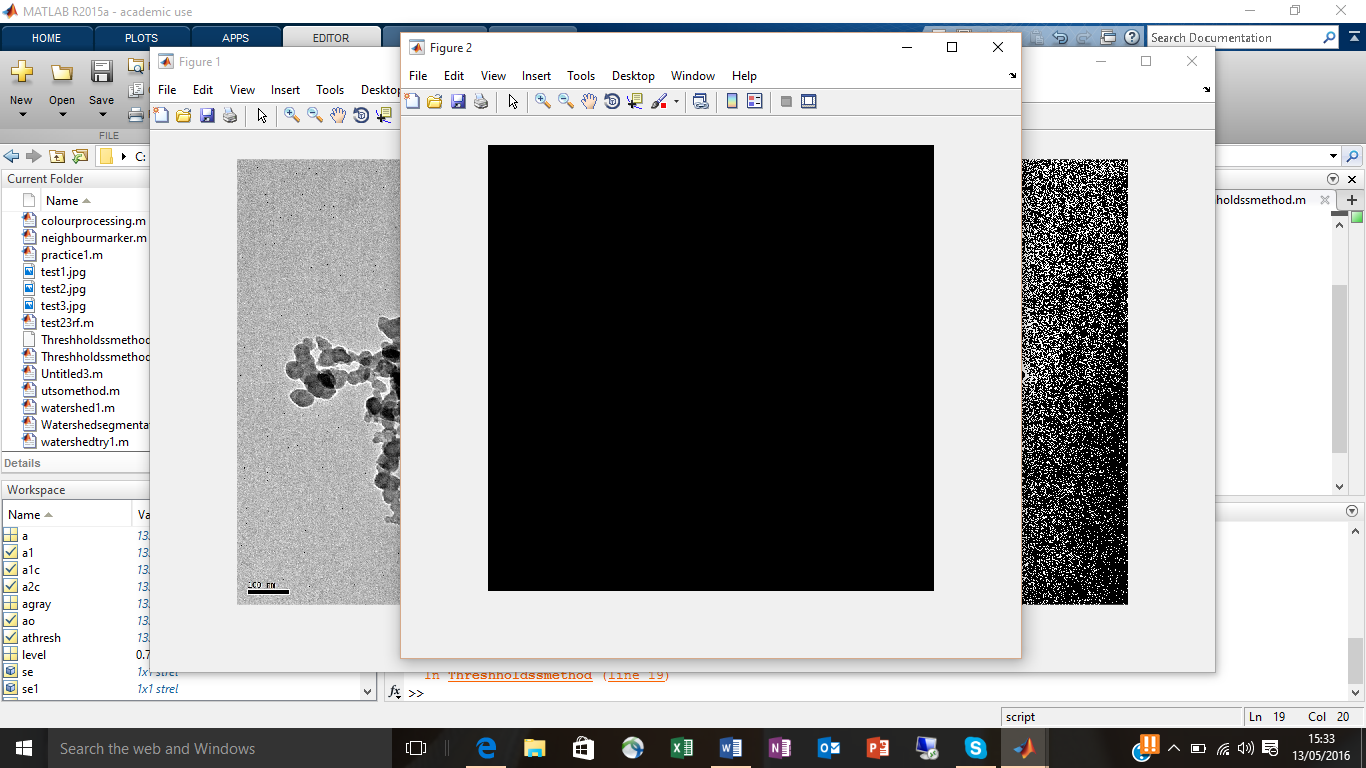


Image D (using structuring element size 20)

The result can be seen above, the whole image turned black, this is because the size of the structuring element is too big. The effect of reducing the size of the structuring element can be seen below.

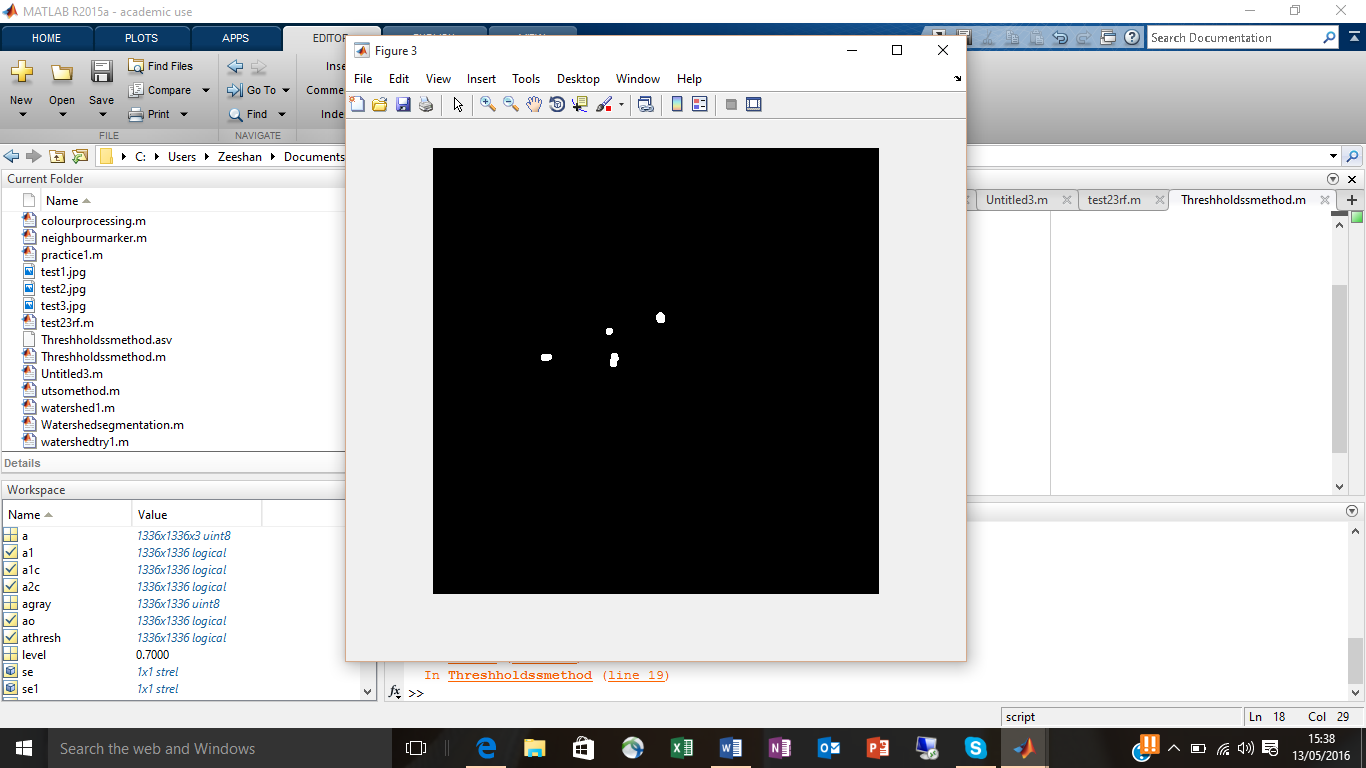


Image E (Structuring element size of 10)

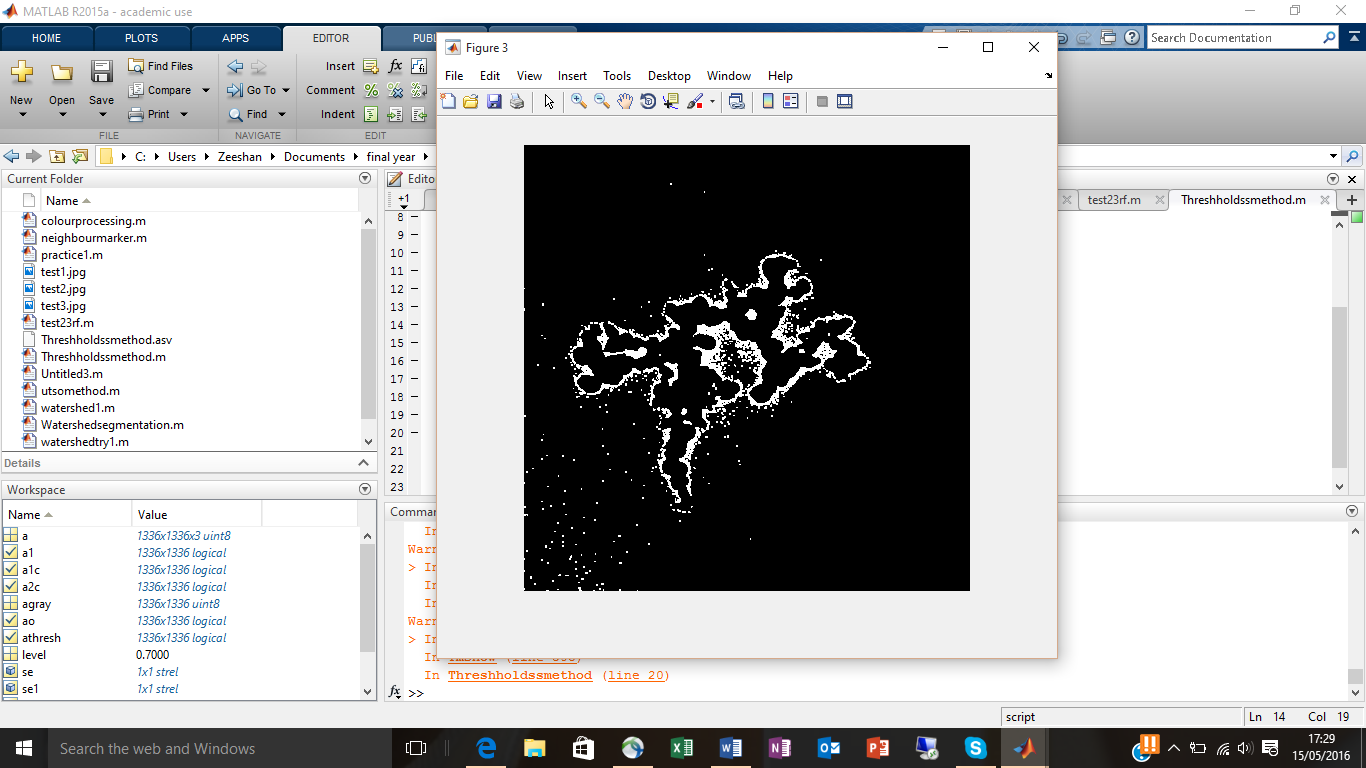


Image F with structuring element size 3

From the above results above it can be understood that, the morphological opening function will not work, therefore, moving on to the morphological closing function. Below, the results are shown.

The code and the results are shown below.

[a1c = imclose(athresh, se1)]

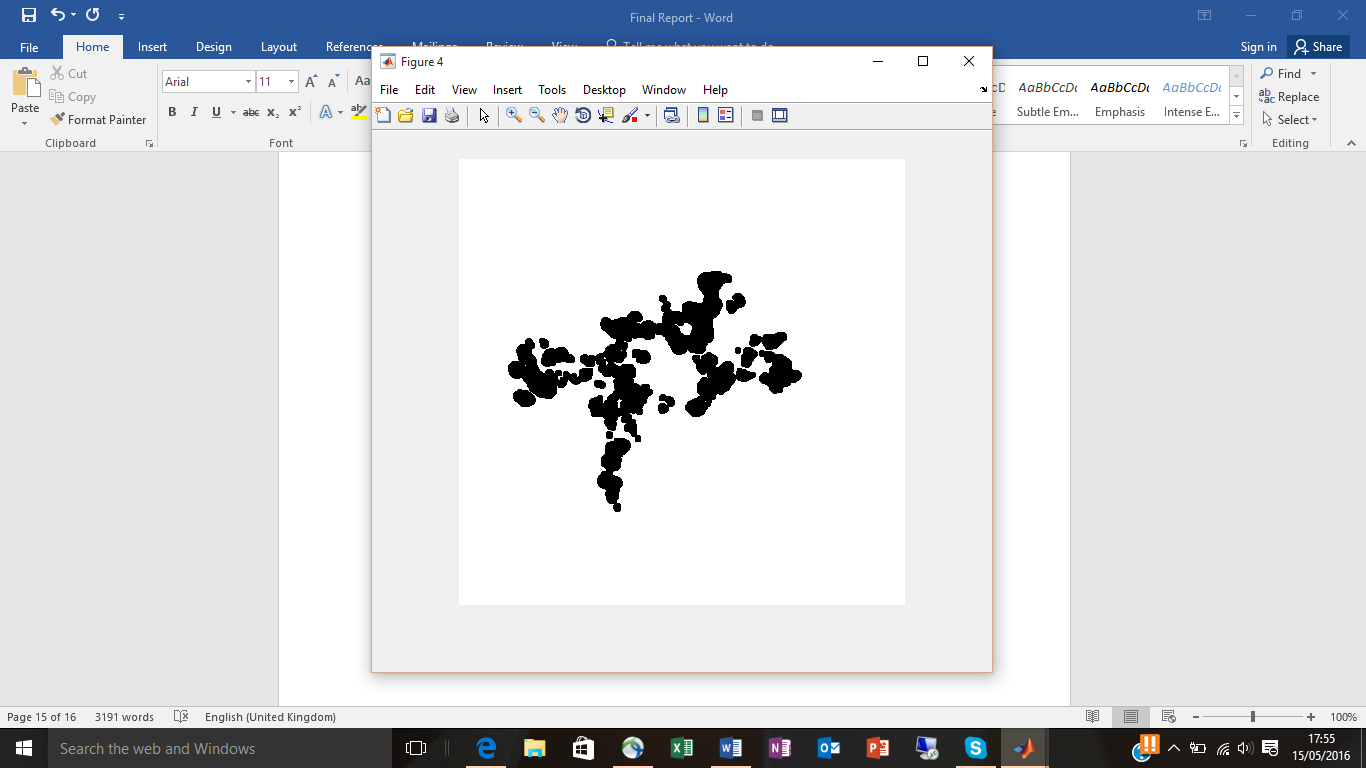


Image G with structuring element size 10

Next the effect of reducing the size of the structuring element will be shown below.

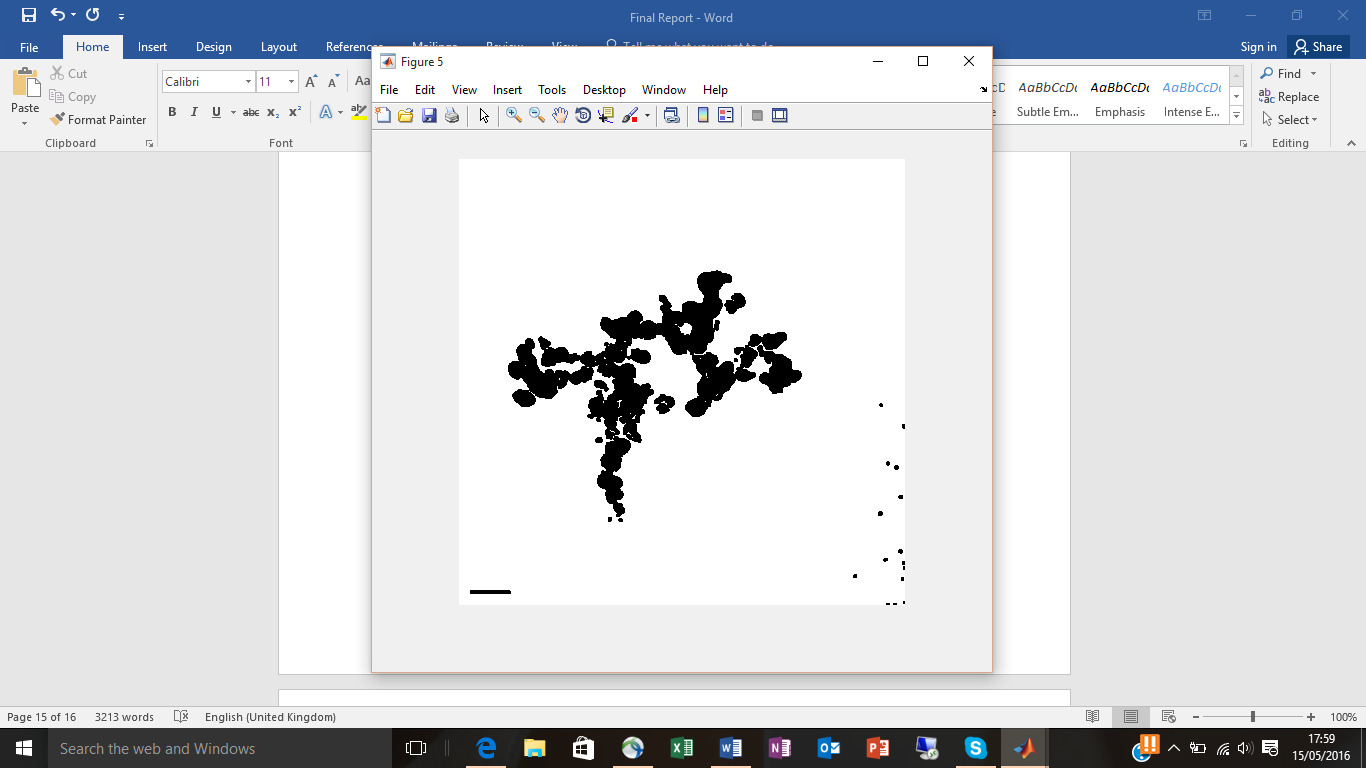


Image H with structuring size 6

From the above image, it can be understood that reducing the size of the structuring element will result in some of the background noise reappearing. From this result we can understand that the best result will be between 6 and 10. This can be seen below.

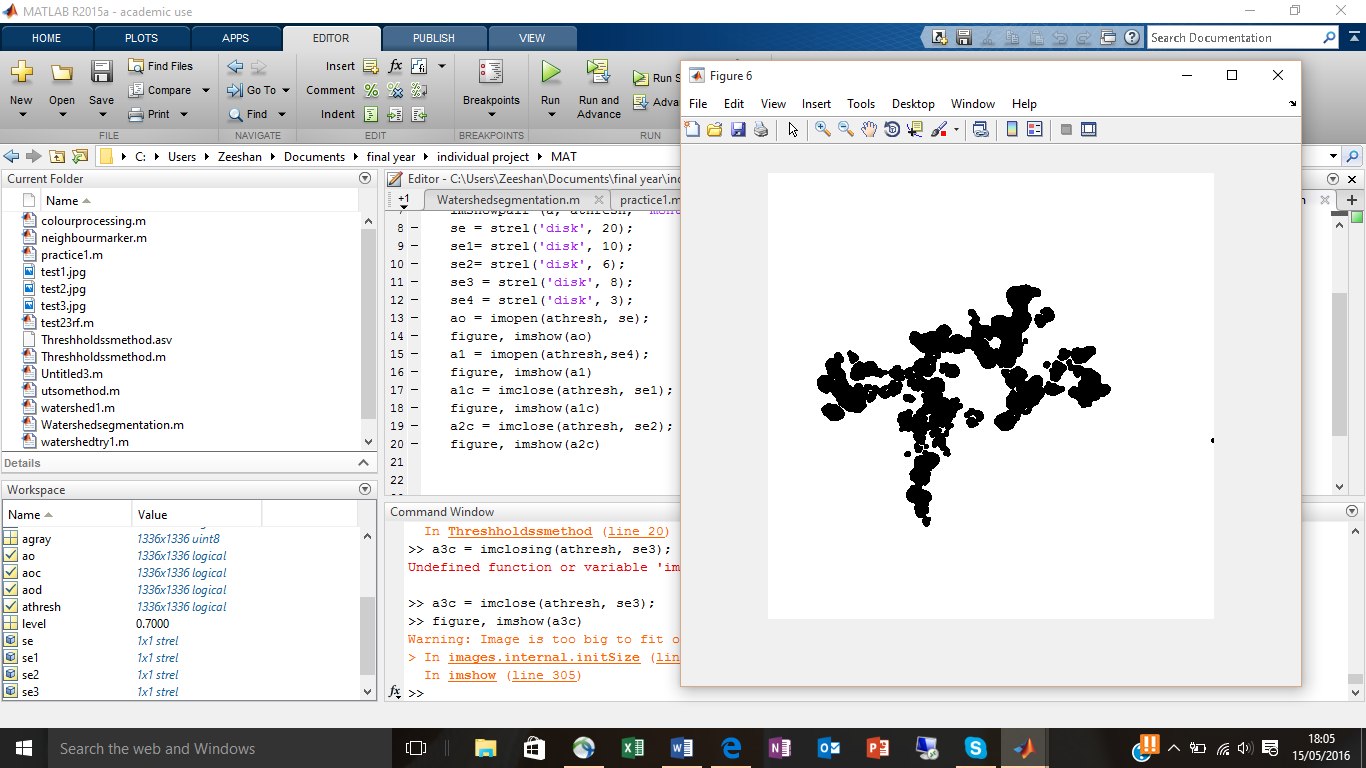


Image I with structuring size 8

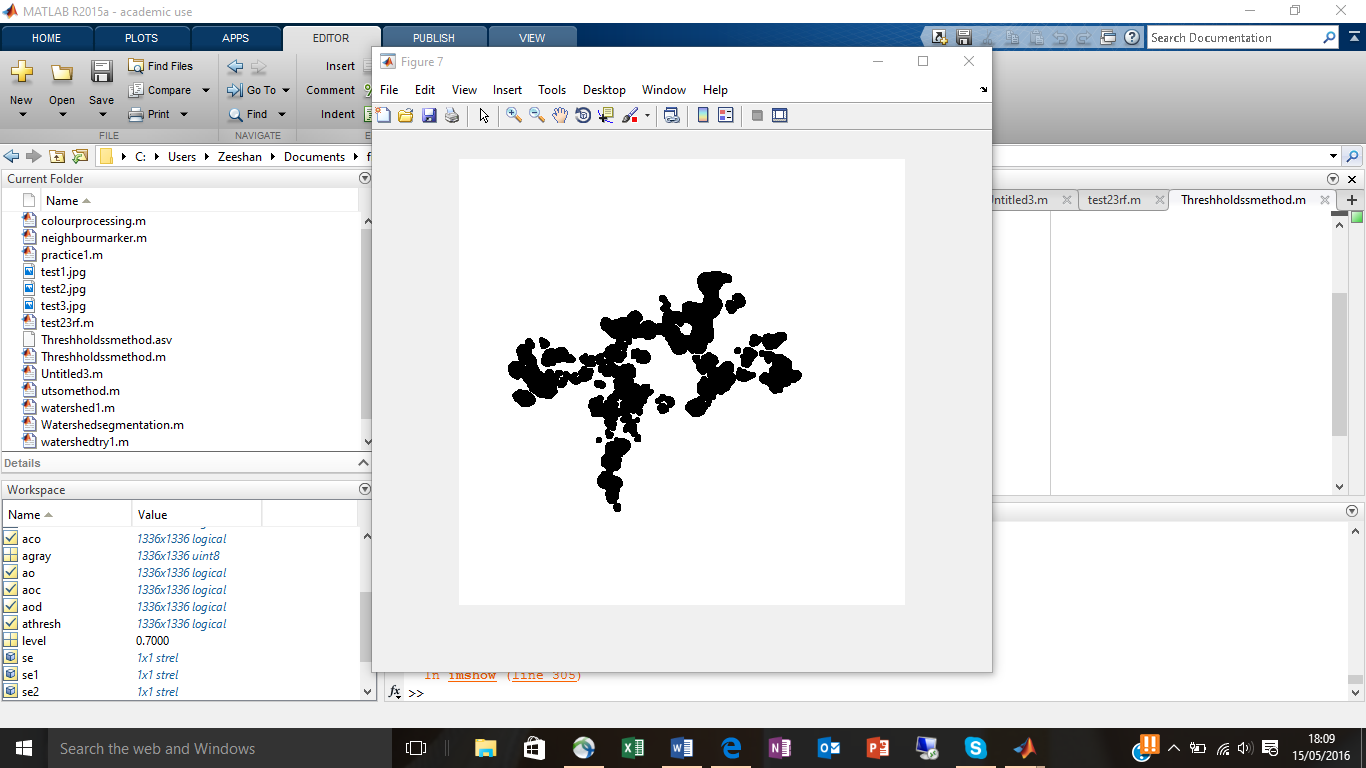


Image J with structuring element size 9

The best result is from the structuring element size 9.

To show how good the above segmentation is, superimpose can be used to overlap image J on top of the original image.



Image K

Image K show the extent to which we have done the segmentation. From this image the unshaded regions are what is left out and not included in the segmentation. Therefore this paticular techniques is not the very best technique.

Method Number 2: Watershed transformation

Segmentation using the watershed transform works better if you can identify, or "mark," foreground objects and background locations. Marker-controlled watershed segmentation follows this basic procedure:

1. Compute a segmentation function. This is an image whose dark regions are the objects you are trying to segment.

2. Compute foreground markers. These are connected blobs of pixels within each of the objects.

3. Compute background markers. These are pixels that are not part of any object.

4. Modify the segmentation function so that it only has minima at the foreground and background marker locations.

5. Compute the watershed transform of the modified segmentation function.

The first step is to import the image into MATLAB and convert it to grayscale. The code below Is used to do this.

fig=imread(strcat(pathname,filename))

I=rgb2gray(fig);

In the above the highlight part is different the usual code for reading an image in. This code is used so that when the m file is run in MATLAB it will ask the user to select the image they want to segment hence that is why there is pathname and filename inside the brackets.

The result for the above code is presented below.

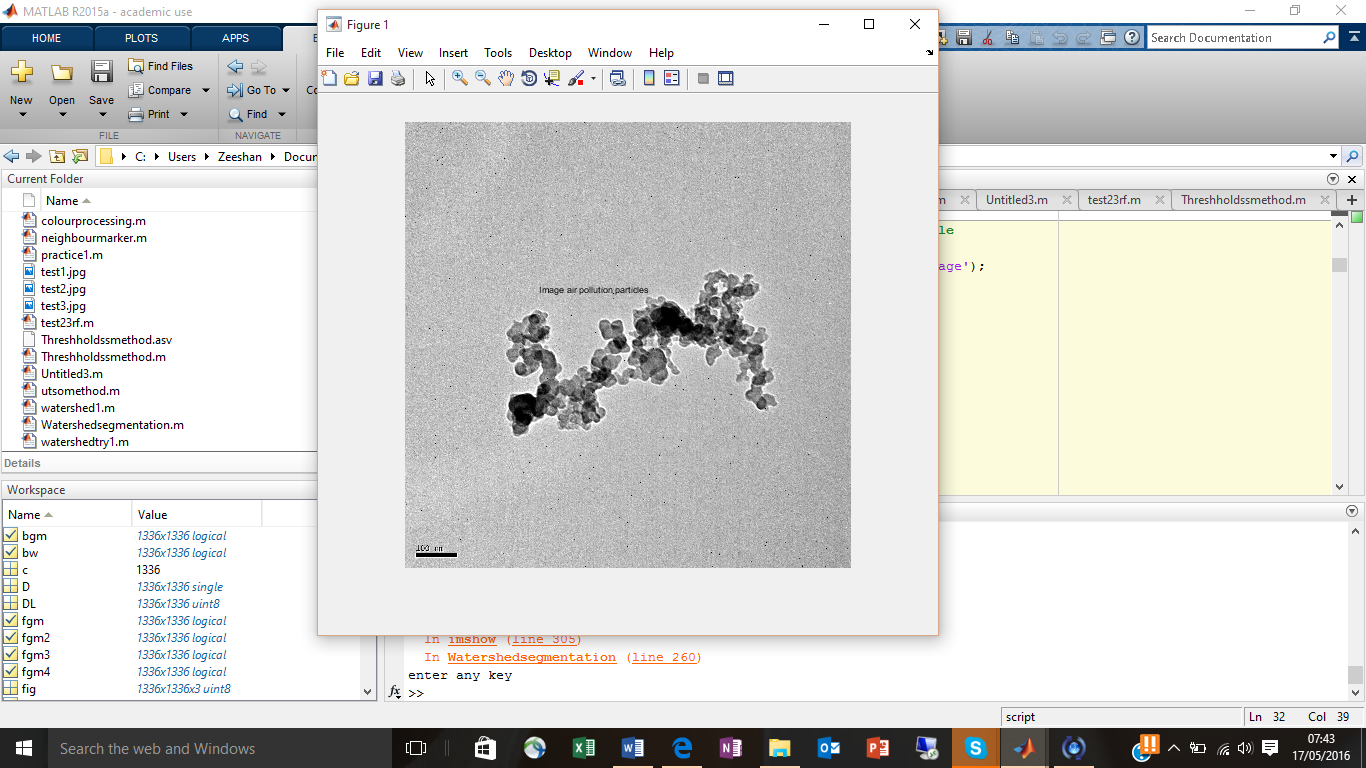


Image M2A

The next step is the created the segmentation function, this is done by finding all the edges. Edges are boundaries between discontinuities in the intensity or grey level of the image. The code and result for this step is shown below.

hy = fspecial('sobel');

hx = hy';

Iy = imfilter(double(I), hy, 'replicate');

Ix = imfilter(double(I), hx, 'replicate');

gradmag = sqrt(Ix.^2 + Iy.^2)

The first line in the code is to find all the edges in the vertical direction, the reason for find edges in vertical direction is the mask ‘sobel’ inside the brackets. This mask finds edges in the vertical direction. To find the edges in horizontal direction this hy in the first line is transposed. This is shown in the second line where, hx is equal to hy transposed. In the second and third line, “imfilter” is a function which is used to apply any filter to an image. In the finrst and second line the filters were created and then in the second and third line these filters were applied. Sum of the vertical and horizontal edges equal to the total edges in the images, therefore in the last line the gradient is found by taking the square root of the vertical and horizontal edges.

Below the result for the gradient is shown.

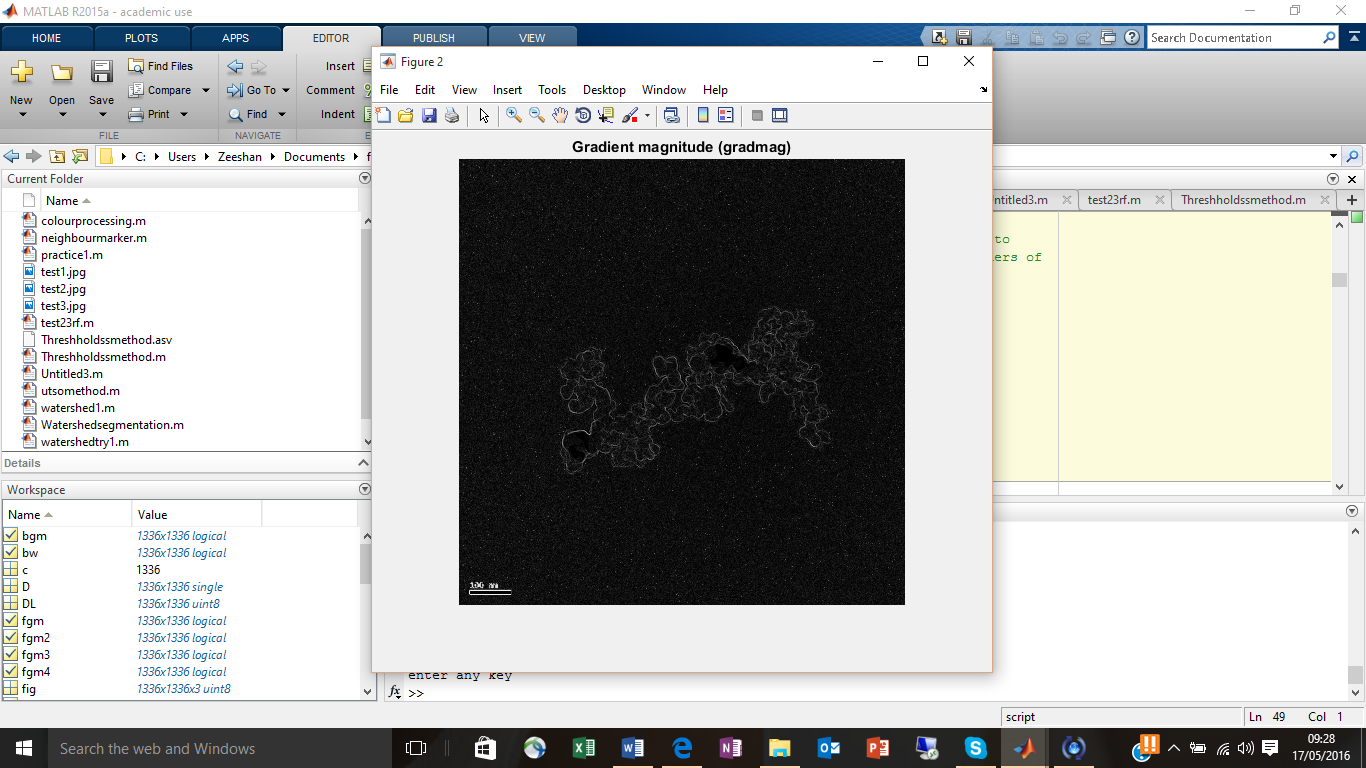


Image M2B the Detected Edges

Next the watershed transform can be applied directly on to the gradient image. The code and result is shown below.

L = watershed(gradmag);

Lrgb = label2rgb(L);

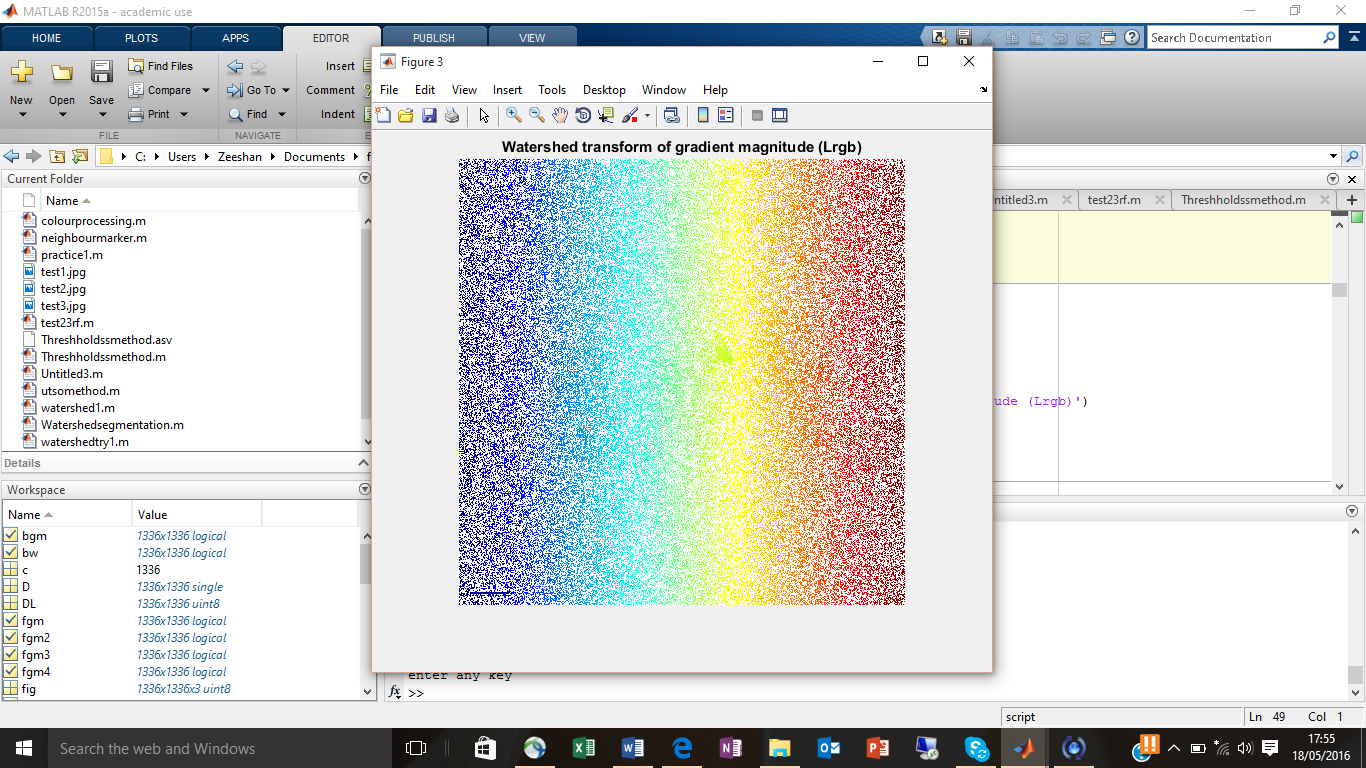


Image M2C

An image cannot be segmented by directly applying the watershed transform on top the gradients, this can be seen above. Therefore, the need for computing markers for the particles and the background are necessary.

Step 3: Computing the markers

Different methods can be applied or used to find these objects. Here the Morphological function opening by reconstruction is going to be used. After this function closing by reconstruction will applied to clean it up further. These function will turn the whole particle or the whole objectives into maxima’s, then these maxima’s can be found using the function “imregionalmax”. This is will result in a much better segmentation get before. The codes and results are shown below in steps.

se = strel('disk', 20);

Io = imopen(I, se);

figure, imshow(Io), title('Opening (Io)')

Line one of the above code is used to create the structuring element that will be used to carry out the erosion followed by dilation.

Line 2 is the function for the morphological opening, which is erosion following by dilation.

The third line is to show the result.

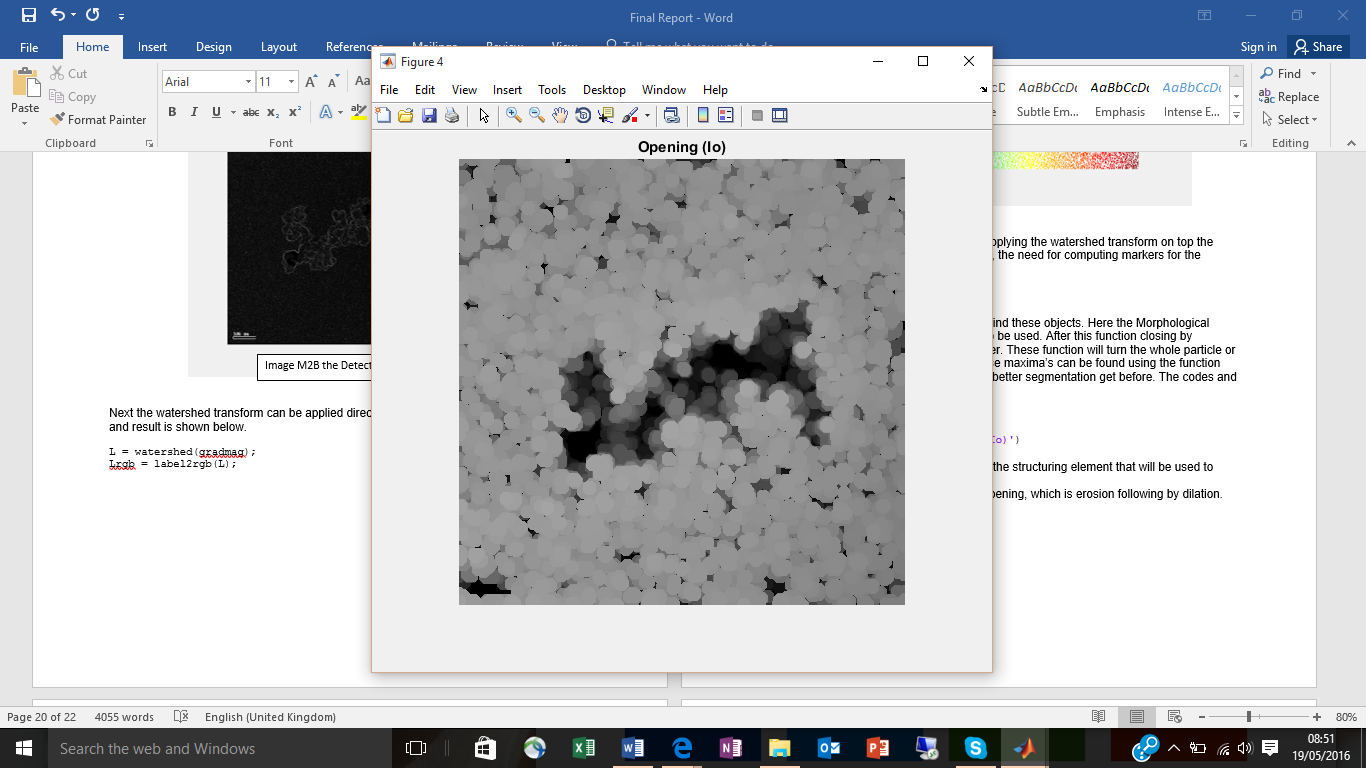


Image M2D

After doing opening, opening by reconstruction takes place, this is to compare the two opening. Opening by reconstruction is done by using the function “imerode”.

Below the code and results for opening by reconstruction will be presented.

Ie = imerode(I, se);

Iobr = imreconstruct(Ie, I);

figure, imshow(Iobr), title('Opening-by-reconstruction (Iobr)')

The first line of the code is the function for the Morphological function erosion which is explain above in the definition.

References:

Peterlin, P (1996) *Morphological operations: an overview* [online] available from <http://www.inf.u-szeged.hu/ssip/1996/morpho/morphology.html> [10May 2016]

MATLAB (2015) MATLABR2015a, The Mathworks inc

Appendix

Code 1: segmentation using threshold

a = imread('test3.jpg');

figure, imshow(a)

agray = rgb2gray(a);

imshow(agray)

level = 0.7;

athresh = im2bw(agray,level);

imshowpair (a, athresh, 'montage');

se = strel('disk', 20);

se1= strel('disk', 10);

se2= strel('disk', 6);

se3 = strel('disk', 8);

se4 = strel('disk', 3);

se5 = strel('disk', 9);

ao = imopen(athresh, se);

figure, imshow(ao)

a1 = imopen(athresh,se4);

figure, imshow(a1)

a1c = imclose(athresh, se1);

figure, imshow(a1c)

a2c = imclose(athresh, se2);

figure, imshow(a2c)

a4c = imclose(athresh, se5);

figure, imshow(a4c)

a2 = a;

a2(a4c) = 255;

figure, imshow(a2)