Assignment 1 Question 2 Report

Sai Gaurav Anugole 170070008
Titas Chakraborty 170070019
Jayesh Choudhary 170070038

August 2019

1 Binary Mask for Foreground Separation

1.1 Approach

We used a simple thresholding function to separate the foreground from the background. This would appear to work exceptionally well for this particular image as most of the background is very dark while most of the foreground is well lit. The output image files are saved in *Images* folder which include:

- q2a_fore_mask.png

1.2 Results

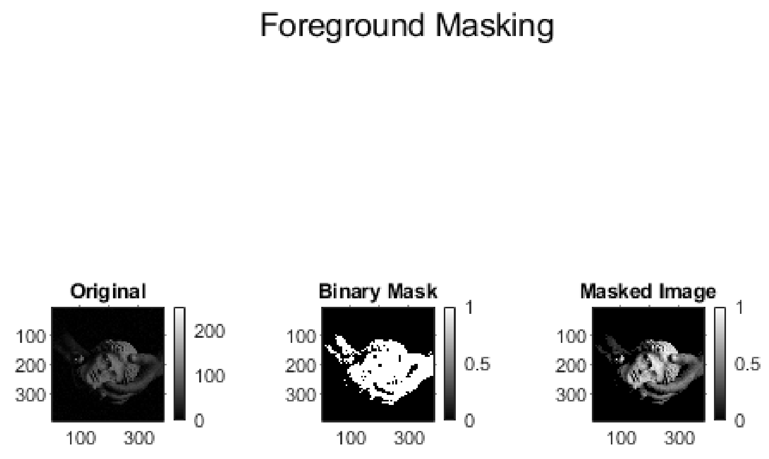


Figure 1: Foreground separation using binary mask

1.3 Inferences

- The thresholding function worked fairly well when it removed the background, but it removed some regions in the foreground which had very dark pixels

- The resultant foreground image seems fairly well segmented. The previously mentioned introduction of black pixels in the foreground is very hard to notice as those regions were very dark in any case

1.4 Conclusion

Hence we have used a thresholding function to build a binary mask for foreground and background segmentation.

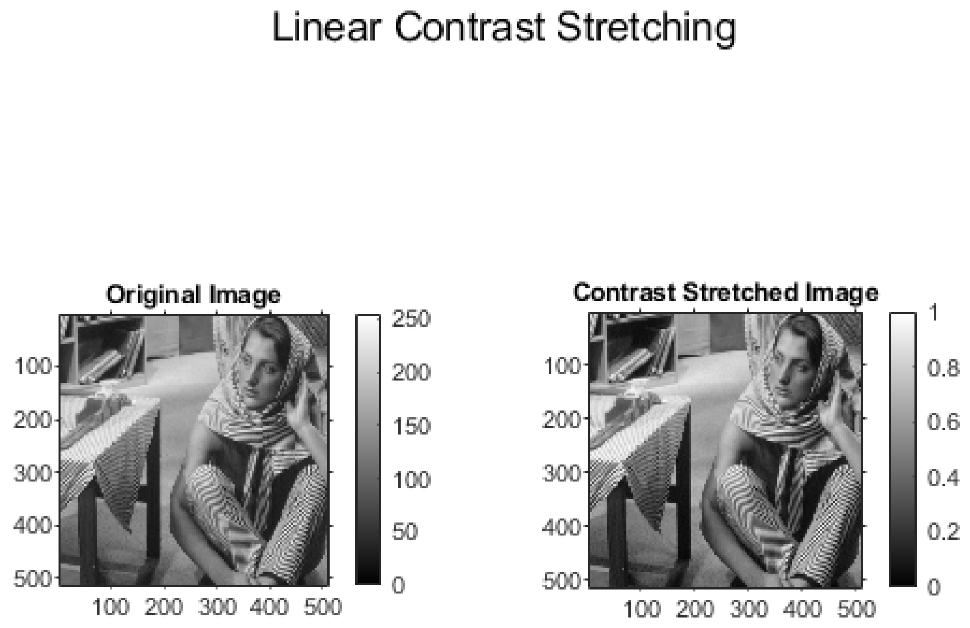
2 Linear Contrast Stretching

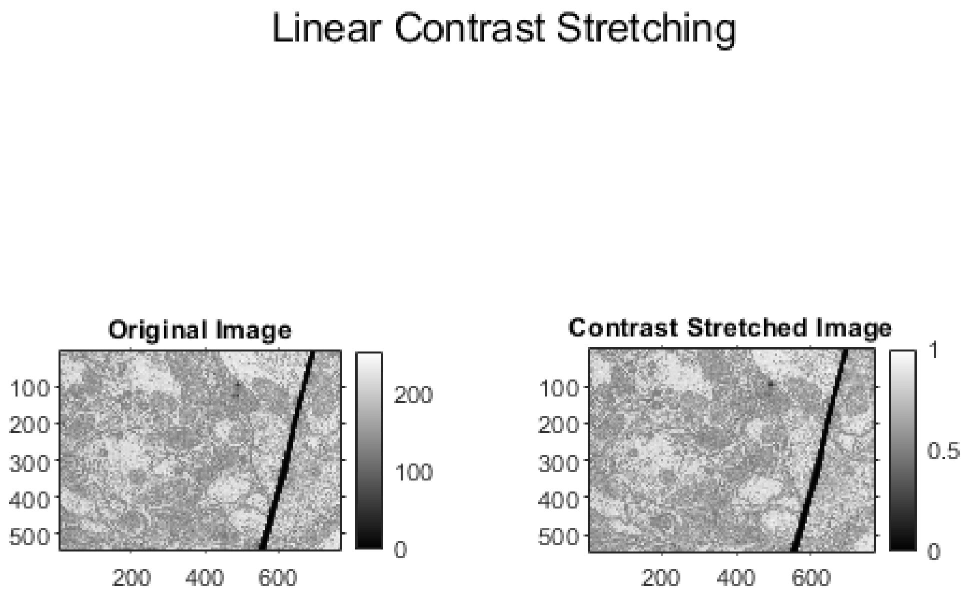
2.1 Approach

Linear Contrast Stretching is used when an image has intensities covering only a very narrow range of the available intensities. Let the range be [a,b]. We used the function: $255 * \frac{x-a}{b-a}$ to map it from 0 to 255. The output image files are saved in *Images* folder which include:

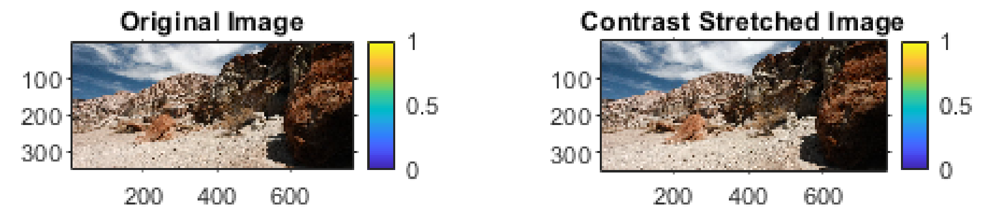
- q2b_linear_stretch_img1.png
- q2b_linear_stretch_img2.png
- q2b_linear_stretch_img3.png
- q2b_linear_stretch_img5.png
- q2b_linear_stretch_img6.png
- q2b_linear_stretch_img7.png
- q2b_linear_stretch_masked_img7.png

2.2 Results

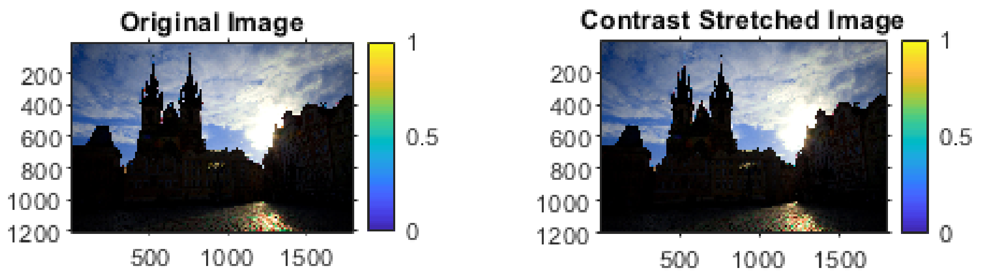

Linear Contrast Stretching
Original Image
Contrast Stretched Image
100
200
300
400
500
250
150
50
0
1
0.8
0.6
0.4
0.2


Linear Contrast Stretching
Original Image
Contrast Stretched Image
100
200
300
400
500
600
0
1
0.5

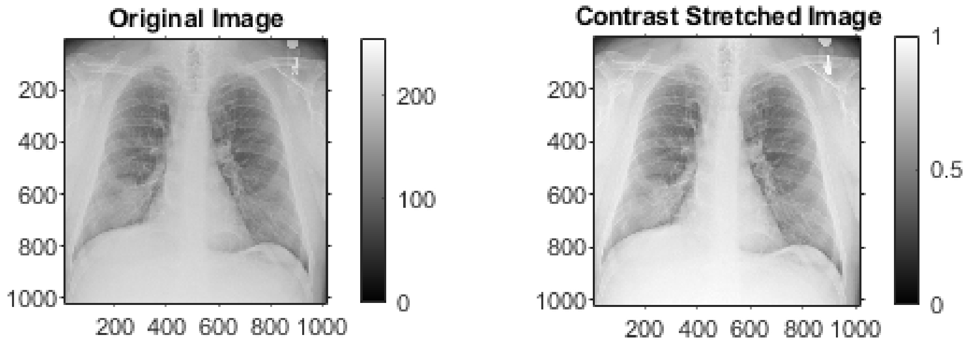
Linear Contrast Stretching



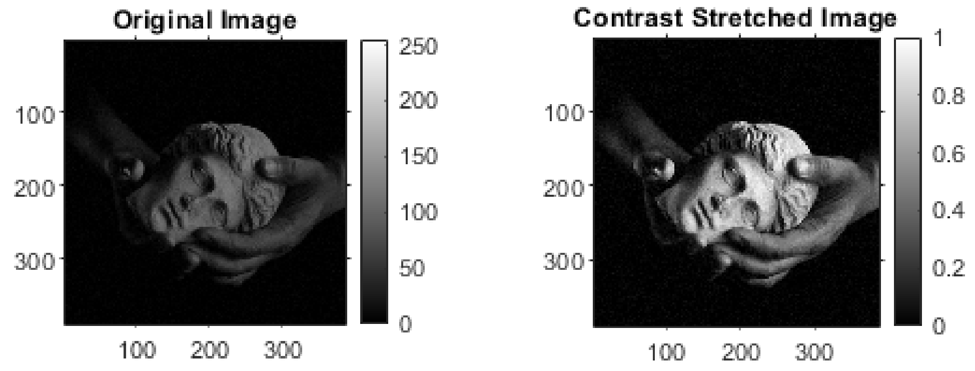
Linear Contrast Stretching

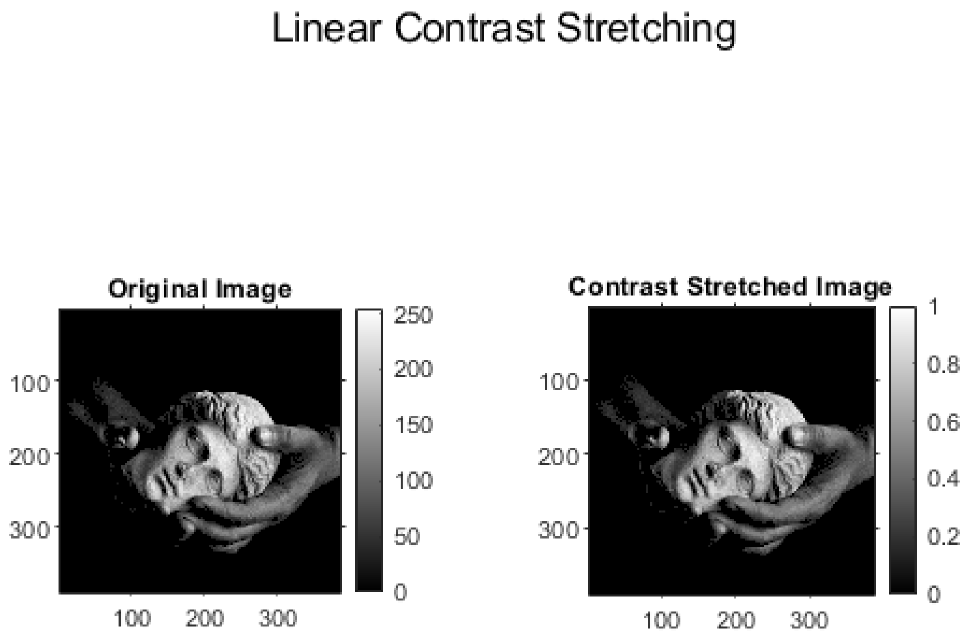


Linear Contrast Stretching



Linear Contrast Stretching





2.3 Inferences

- Linear interpolation seemed to produce little difference for some images. However, for other images, the effect was visible
- For image 5, the interpolation seemed to have no effect at all. This was because the image consisted of very dark regions where RGB was very close to [0,0,0]. But it also consisted of very bright regions(the sky) where RGB was very close to [255,255,255]. Hence since the image already occupied all of the available intensities, there was nothing which linear contrast stretching could change. Some other images had negligible change due to the same reason.

2.4 Conclusions

Hence we have successfully used linear contrast stretching to enhance our image.

3 Histogram Equalization

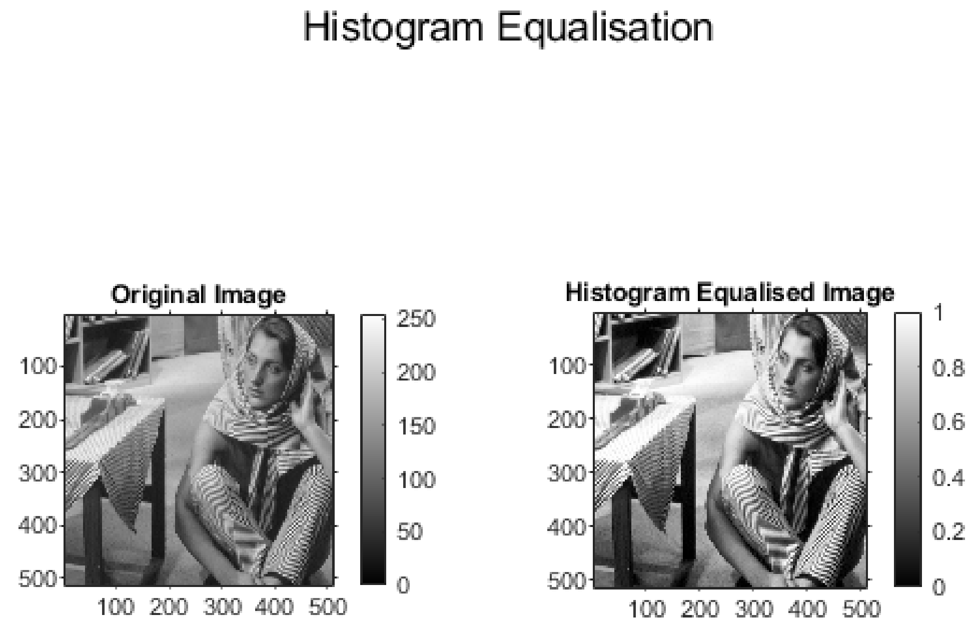
3.1 Approach

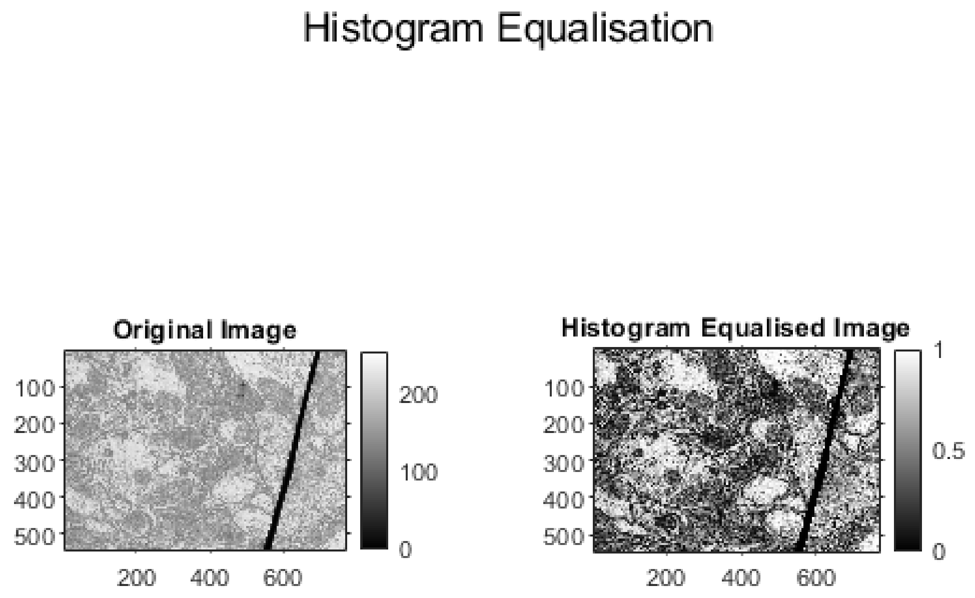
To do histogram equalization, we used the cdf() function of MATLAB to map our image to a cdf. Then for each intensity point x, $255 * cdf(x)$ would give us our new intensity to which the point is mapped. The output image files are saved in *Images* folder which include:

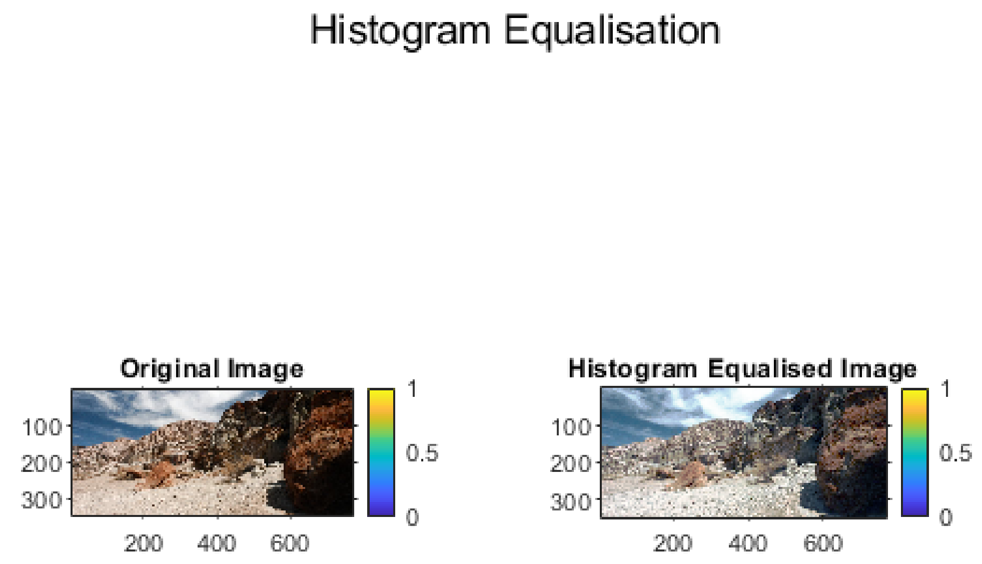
- q2c_hitogram_equalised_img1.png
- q2c_hitogram_equalised_img2.png
- q2c_hitogram_equalised_img3.png
- q2c_hitogram_equalised_img5.png

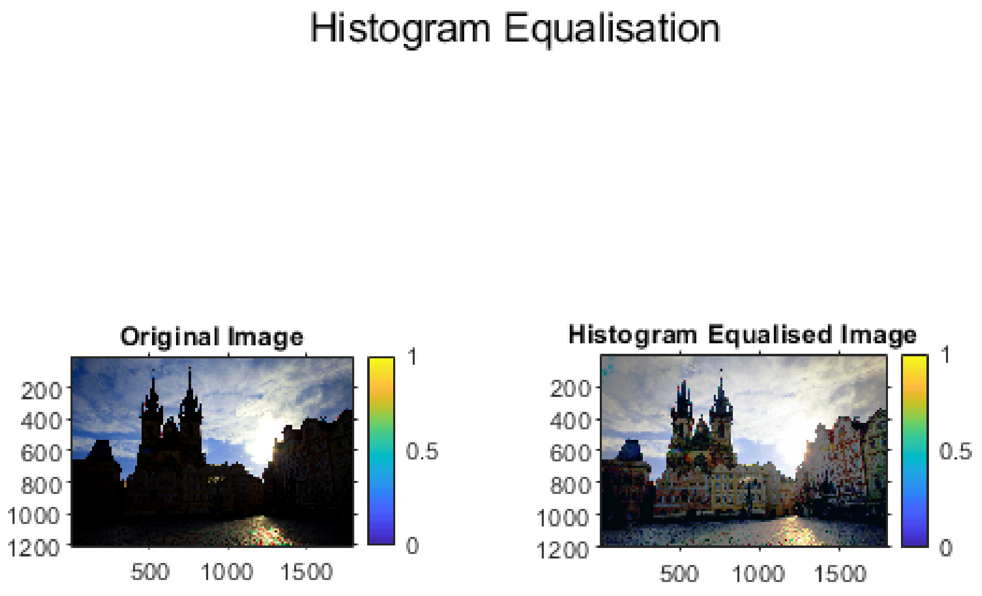
- q2c_hitogram_equalised_img6.png
- q2c_hitogram_equalised_img7.png
- q2c_hitogram_equalised_masked_img7.png

3.2 Results

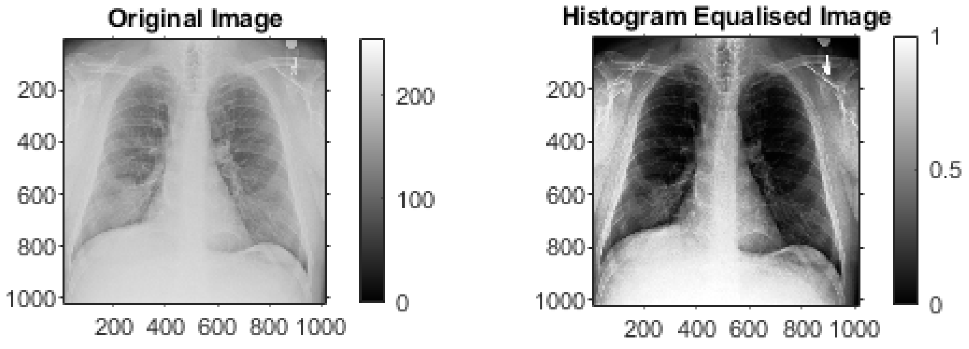

Histogram Equalisation
Original Image
Histogram Equalised Image


Histogram Equalisation
Original Image
Histogram Equalised Image

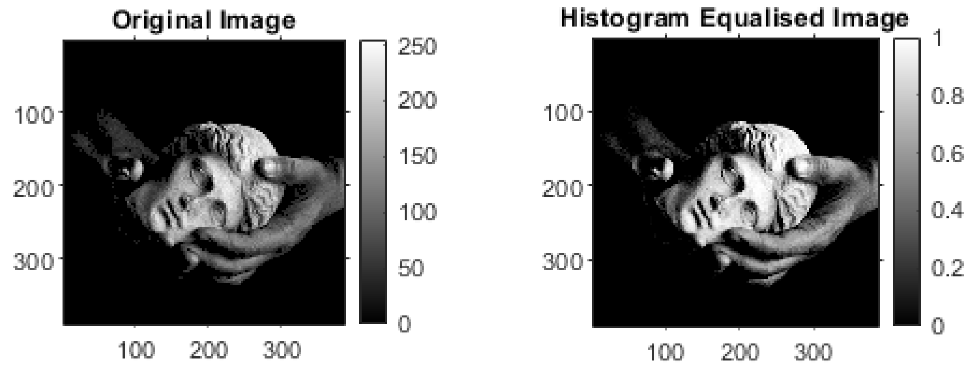

Histogram Equalisation
Original Image
Histogram Equalised Image
100
200
300
200
400
600
1
0.5
0


Histogram Equalisation
Original Image
Histogram Equalised Image
200
400
600
800
1000
1200
500
1000
1500
1
0.5
0

Histogram Equalisation



Histogram Equalisation



3.3 Inferences

- Histogram equalization works very well on most images.
- Histogram equalization works very successfully on image 5. This is because image 5 had a very large number of pixels with very high or very low intensities. Assuming half of the intensities to be very large and half to be very small, histogram equalization mapped the very small intensities to an intensity range of 0 to 127 and the very large intensities to a range of 128 to 255, greatly enhancing the contrast in the image.

3.4 Conclusions

Hence we have performed histogram equalization on the image

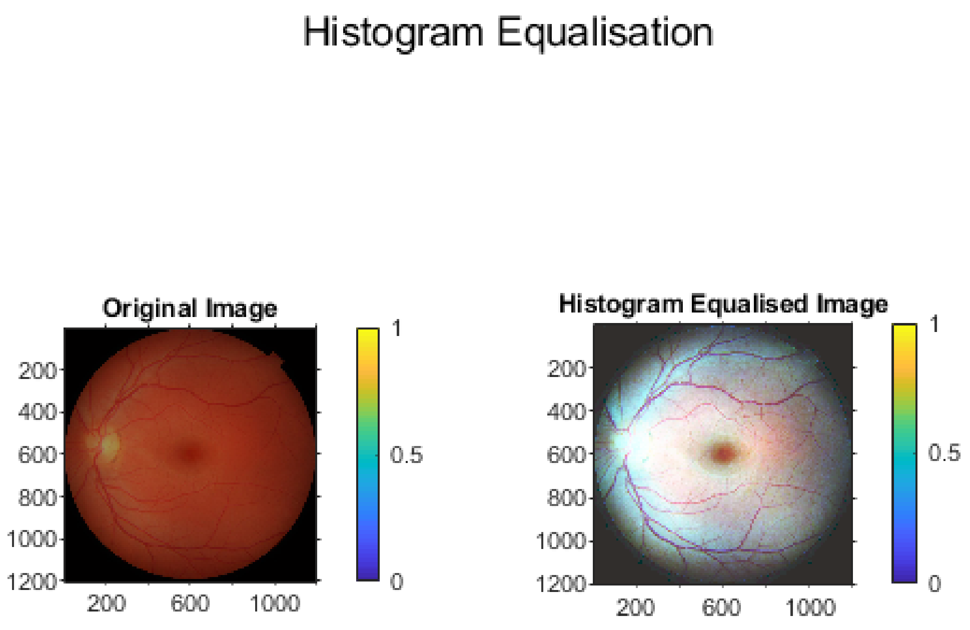
4 Histogram Matching

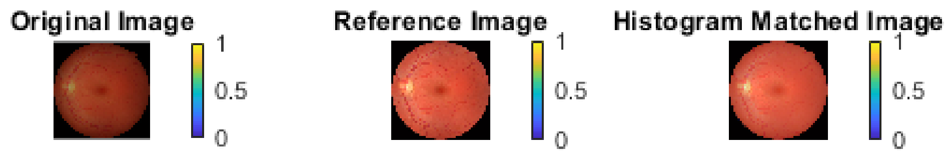
4.1 Approach

For histogram matching, we are given two images, a reference image and an image to be equalized. We compute the cdf for both. Then for each intensity x in the image to be equalized, we find $cdf_im(x)$. Then we found the next biggest element in the array of $cdf_ref(x)$ and assigned the intensity of the point in the reference image with that intensity to the original intensity in our image. The output image files are saved in *Images* folder which include:

- q2d_histogram_equalised.png
- q2d_histogram_matched.png

4.2 Results


Histogram Equalisation
Original Image
Histogram Equalised Image
200
400
600
800
1000
1200
200
600
1000
1
0.5
0


Original Image
Reference Image
Histogram Matched Image
1
0.5
0

4.3 Inferences

- Histogram matching seems to work very well in this case.
- Our sample image is dark while our reference image is considerably more well lit. After doing histogram equalization, our sample image became as well lit as the reference image

4.4 Conclusions

Hence we have performed histogram matching with satisfactory results

5 CLAHE

5.1 Approach

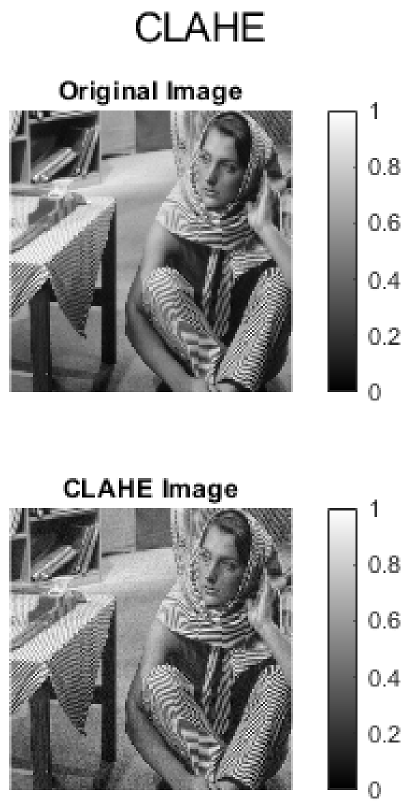
For a given pixel in an image, a function funcHE() is called which computes the CLAHE intensity corresponding to the given window size and threshold parameter (epsilon). The histogram of the window is computed iterating the pixel coordinates from (iistart, jjstart) to (iiend, jjend). Depending on the location of the pixel within the image, these start and end points are varied. For Instance, if the pixel is at a position (a,b) such that the window lies completely within the image, then iteration takes place from (a - $\frac{n-1}{2}$, b - $\frac{n-1}{2}$) to (a - $\frac{n-1}{2}$, b - $\frac{n-1}{2}$). Considering another case, when the pixel is at location say the top left corner of the image, then iistart and jjstart are set to (1,1). Like wise all corner and edge margin cases when the window protrudes out of the image boundary are considered and corresponding start and end points are given. Then, the histogram is computed. With Epsilon being a value between (0,1), normalized to the range of given histogram, sum of all values above it are computed and correspondingly subtracted forming now an upper limit. The remainder sum is now evenly distributed over all intensities to give a new histogram. This new histogram is now equalized as done before and the intensity corresponding to the input intensity from the CDF function is return back.

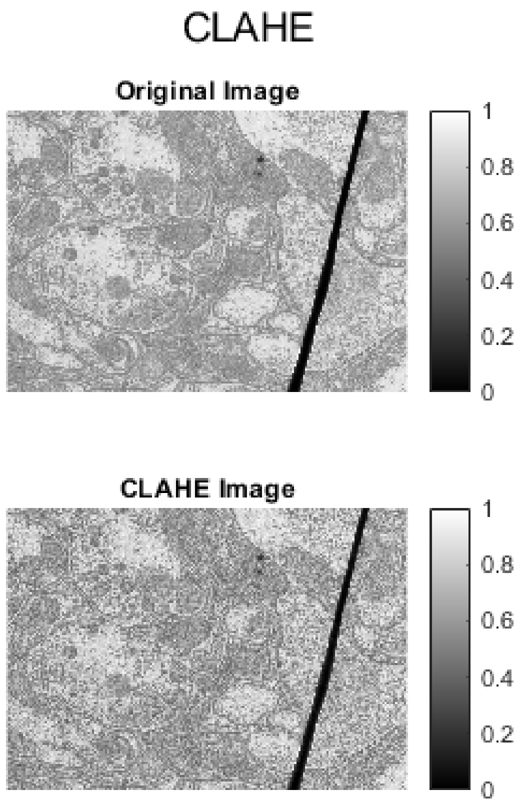
5.2 Results

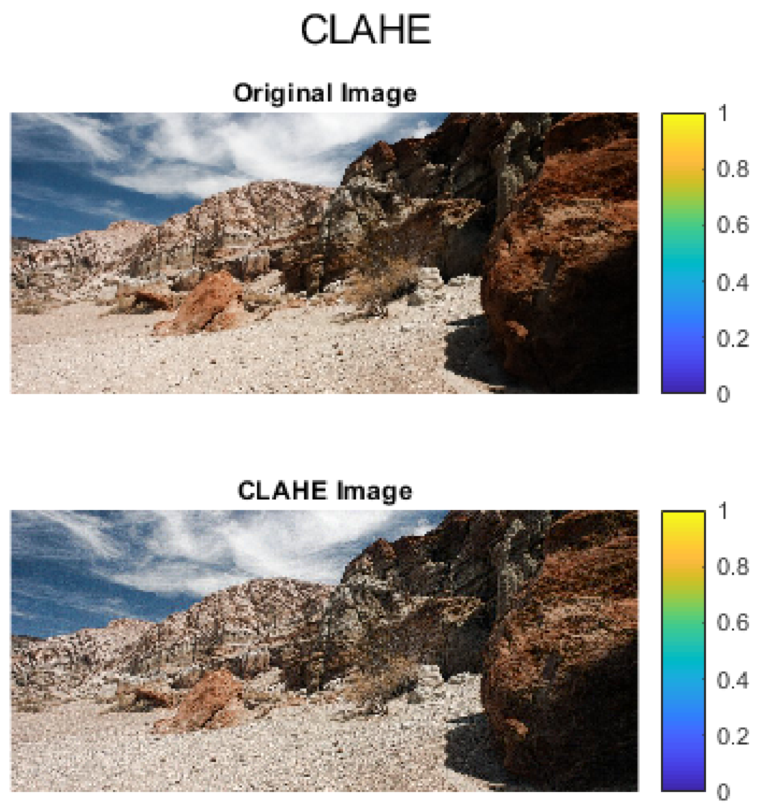
5.2.1 Window Size=5x5 and epsilon=0.1

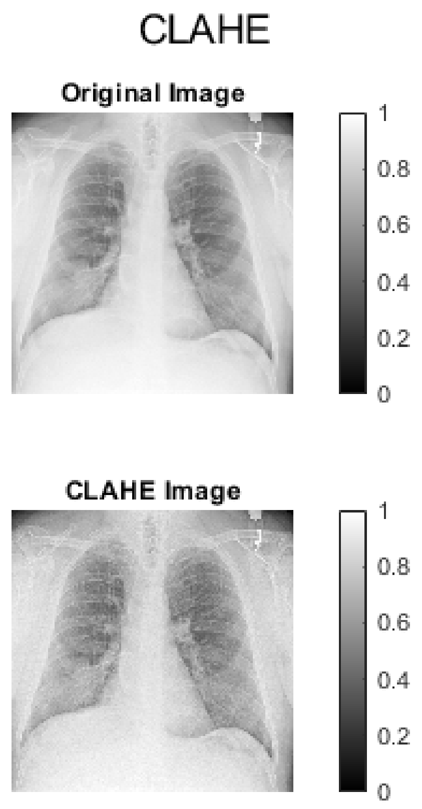
The output image files are saved in *Images* folder which include:

- q2e_clahe_5x5_1_img1.png
- q2e_clahe_5x5_1_img2.png
- q2e_clahe_5x5_1_img3.png
- q2e_clahe_5x5_1_img6.png


CLAHE
Original Image
CLAHE Image
1
0.8
0.6
0.4
0.2
0


CLAHE
Original Image
CLAHE Image
1
0.8
0.6
0.4
0.2
0

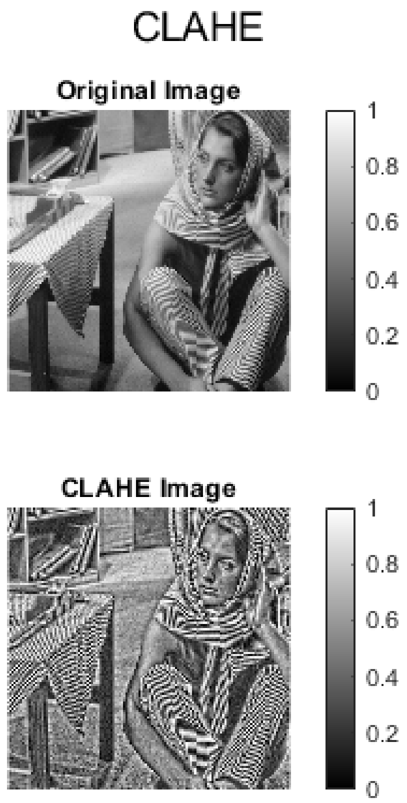

CLAHE
Original Image
1
0.8
0.6
0.4
0.2
0
CLAHE Image
1
0.8
0.6
0.4
0.2
0

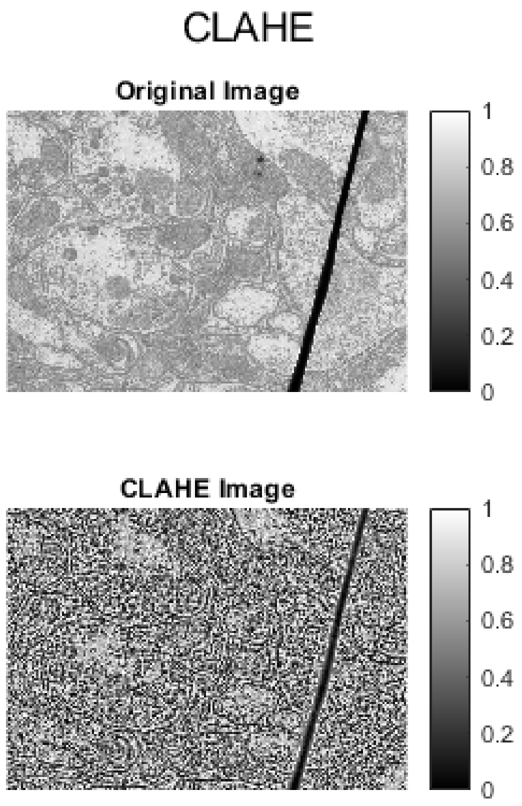

CLAHE
Original Image
1
0.8
0.6
0.4
0.2
0
CLAHE Image
1
0.8
0.6
0.4
0.2
0

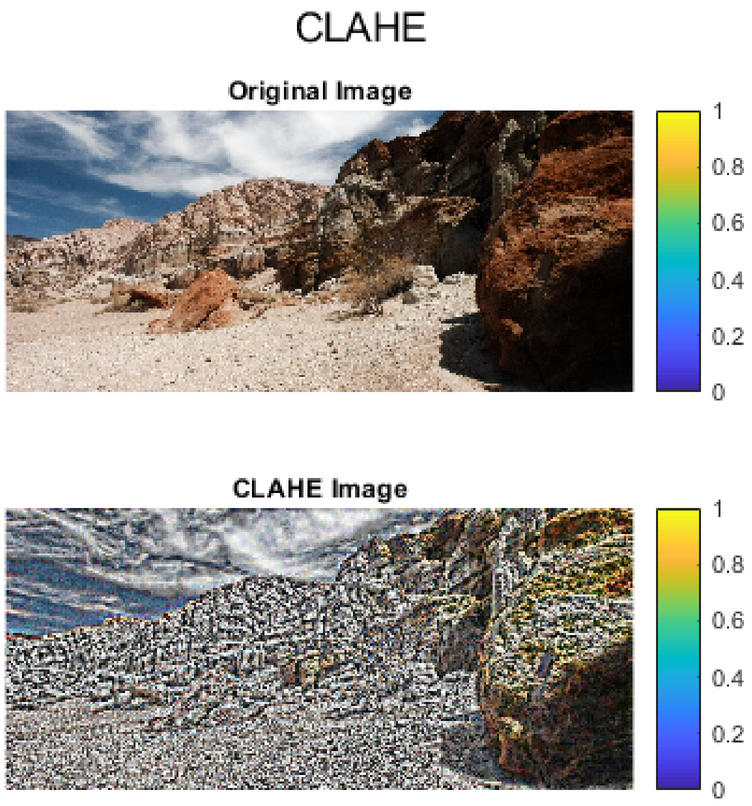
5.2.2 Window Size=15x15 and epsilon=0.1

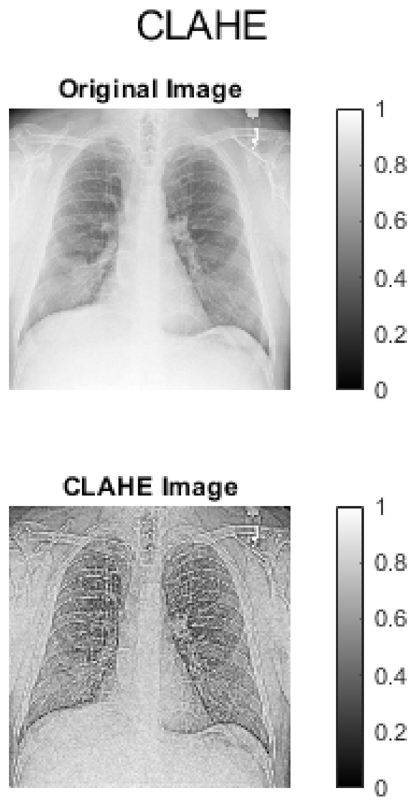
The output image files are saved in *Images* folder which include:

- q2e_clahe_15x15_1_img1.png
- q2e_clahe_15x15_1_img2.png
- q2e_clahe_15x15_1_img3.png
- q2e_clahe_15x15_1_img6.png




CLAHE
Original Image
1
0.8
0.6
0.4
0.2
0
CLAHE Image
1
0.8
0.6
0.4
0.2
0

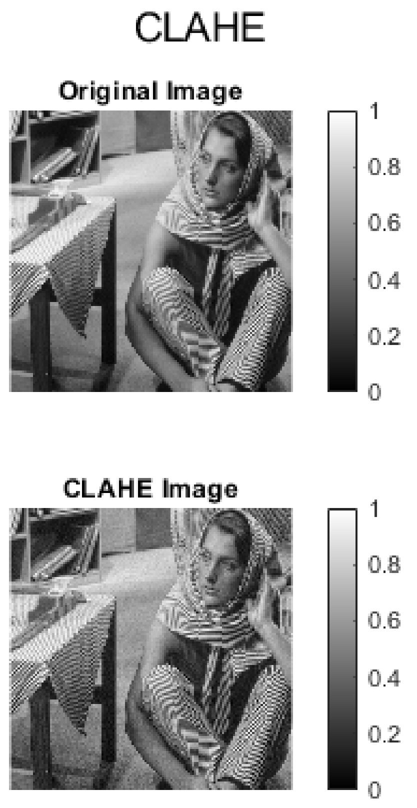

CLAHE
Original Image
1
0.8
0.6
0.4
0.2
0
CLAHE Image
1
0.8
0.6
0.4
0.2
0

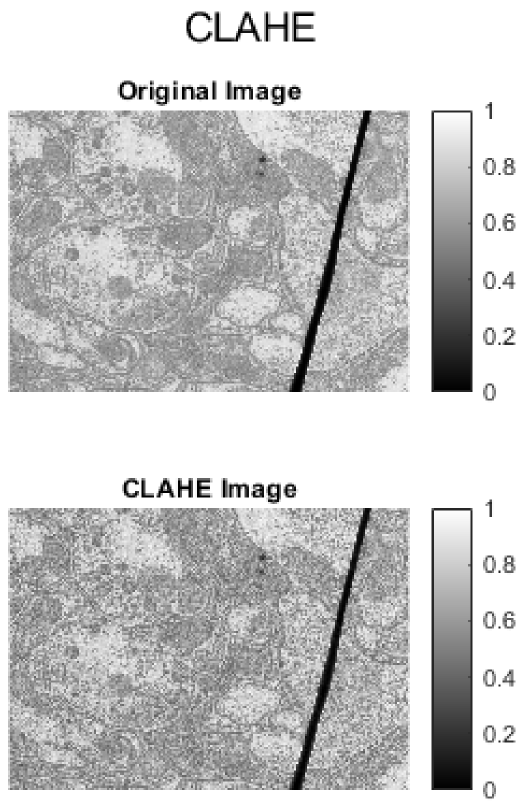


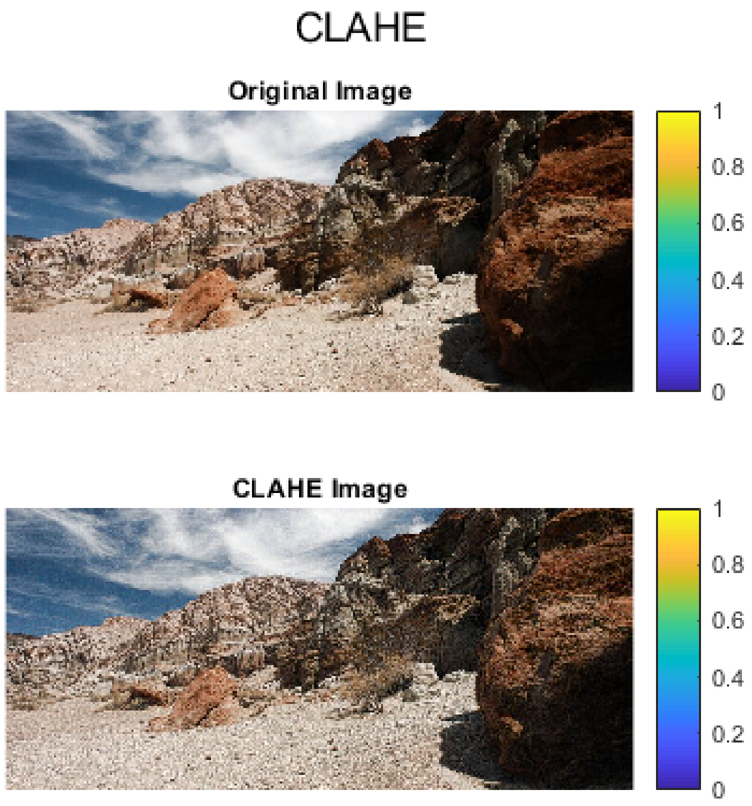
5.2.3 Window Size=5x5 and epsilon=0.05

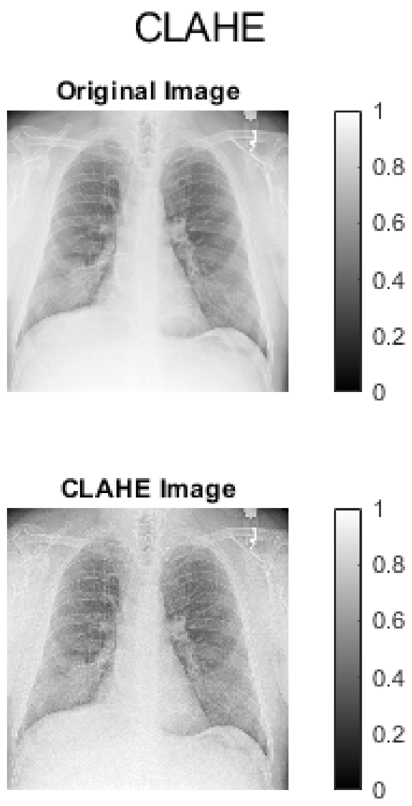
The output image files are saved in *Images* folder which include:

- q2e_clahe_5x5_2_img1.png
- q2e_clahe_5x5_2_img2.png
- q2e_clahe_5x5_2_img3.png
- q2e_clahe_5x5_2_img6.png


CLAHE
Original Image
1
0.8
0.6
0.4
0.2
0
CLAHE Image
1
0.8
0.6
0.4
0.2
0


CLAHE
Original Image
1
0.8
0.6
0.4
0.2
0
CLAHE Image
1
0.8
0.6
0.4
0.2
0


CLAHE
Original Image
CLAHE Image
1
0.8
0.6
0.4
0.2
0


CLAHE
Original Image
CLAHE Image
1
0.8
0.6
0.4
0.2
0

5.3 Inference

- For a given epsilon and window size, CLAHE seems to enhance the contrast of the given image
- With a larger window size, there is excessive noise amplication and low contrast enhancement than the original one
- With the same window size as the beginning, decreasing epsilon by factor of 2 resulted in slight increase in contrast than before