AI6121 Computer Vision

Assignment 1 Report

Members:
Chiam Jia Hao(G2104016B)
Li Xuemeng(G2202226E)
Ryan Tan Zhiyang(G2202174K)

1. Abstract/Introduction

For most computer vision tasks, the further-work implementations are all based on grey images. However, sometimes the images appear 'foggy' and that is because most of the grey levels are between an incorrect or narrow range, for example, 100 and 180. If we want a more clear grey image, we should adjust the grey levels to 0-255.

Histogram Equalisation is a way to enhance the contrast of an image. To achieve it, we have to turn the normal histogram into an approximately uniform-distribution histogram, which is also the core idea of the method. In the following part, we will introduce how we have implemented Histogram Equalisation and tested the image given by this assignment. In addition, the discussion and further work are included in detail.

2. Implementation of HE and Result

For the implementation part, our team has worked out two approaches. The first one mapped the CDF to the grey-scale values of the new image from its PMF. The second one is similar to the first approach but applied to the individual RGB channels of the image.

Note: All enhanced images generated by Approach 1 are shown in the table below while enhanced images generated by Approach 2 contain a prefix of "he_" followed by the sample image name (e.g. "he_sample01.jpg").

2.1. Approach 1

Firstly, consider the case where the grey value is continuous. For the grey scales of the image before processed(gray_before) and after processes(gray_after), they all have the same value boundary[0,L-1], L is always considered as 256. And for every pixel with the gray_before value from the input image, it will have the gray_after value after transition Function T(). Since both gray_before and gray_after are all random values in [0,L-1], then we can get their Probability Density Function P(gray_before) and P(gray_after). Usually, we can get P(gray_after) by using the value P(gray_before) and the function T(), but after calculation, we can know that P(gray_after) actually does not depend on P(gray_before), it is a variable that obeys a uniform distribution: P(gray_after) = 1/L-1.

$$\frac{d_{after}}{d_{before}} = \frac{dT(before)}{d_{before}} = (L-1)\frac{d\left[\int_0^{before} P_{before}(w)\,dw\right]}{d_{before}} = (L-1)\,p_{before}(before)$$

$$P_{after}(after) = P_{before}(before)\left|\frac{d_{before}}{d_{after}}\right| = P_{before}(before)\left|\frac{d_{after}}{d_{before}}\right|^{-1}$$

$$= P_{before}(before)\,\frac{1}{(L-1)\,\cancel{P_{before}(before)}}$$

$$= \frac{1}{L-1}.$$

However, in this task, we normally have the discrete image grey value. The code can be seen in the HEq1.m file. For discrete values, we have a k, which values in 0,1,2,…,L-1. Then set r_k as the grey value of the kth level, and n_k is the number of pixels with grayscale r_k in the image. The probability that r_k occurs in the image is P(r_k) = n_k/WH. Since the values are discrete, they sum to 1. Therefore, the PMF should be:

$$PMF_X(i) = \frac{n_k}{WH}, \quad 0 \le i < L$$

To describe the probability distribution of a random value X, which is also the summation of the probability mass function. Also with the Transition function T(), we can then calculate the CDF for a discrete grey image X:

$$CDF_X(x) = P(X \le x)$$

$$\Rightarrow CDF_X(i) = \sum_{j=0}^{i} PMF_X(j)$$

$$s_k = T(r_k) = (L-1)\sum_{j=0}^{k} P_r(r_j) = \frac{(L-1)}{WH}\sum_{j=0}^{k} n_j \cdot k$$

after before

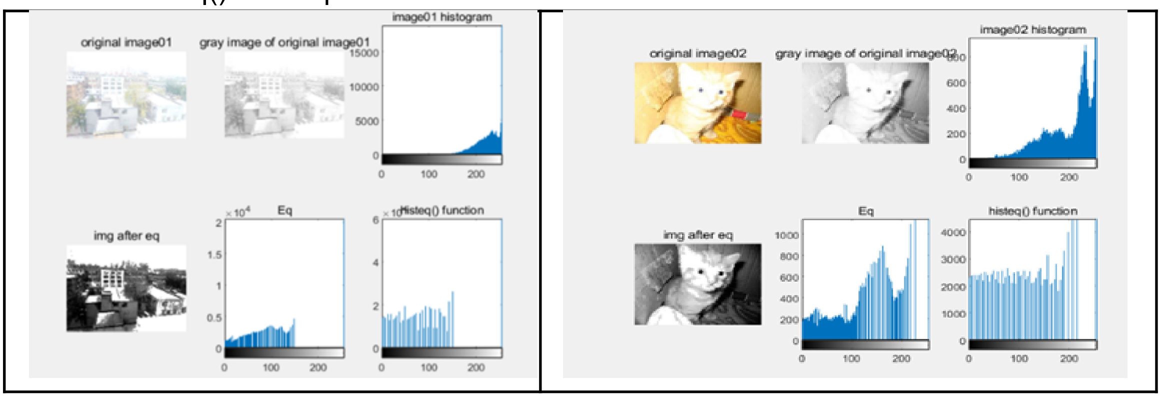
Which has the same result as the formula in the lecture slide *Spatial Image Filtering*.

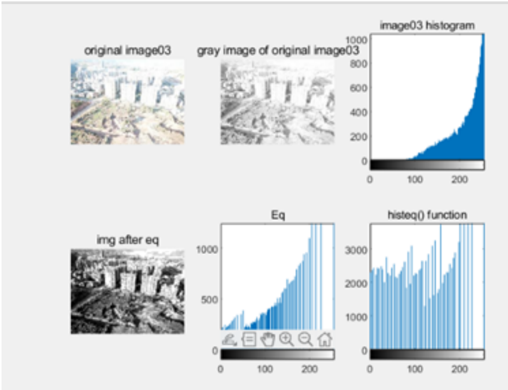
How many bins can be filled? $\sum_{j=0}^{k} N_j \div \frac{W.H}{L-1} = \frac{L-1}{W.H}\sum_{j=0}^{k} N_j$

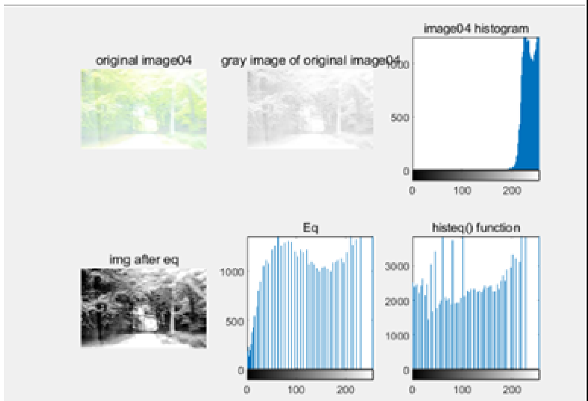
Where do pixels with intensity *k* go? $s_k = (L-1)\sum_{j=0}^{k} p_j$

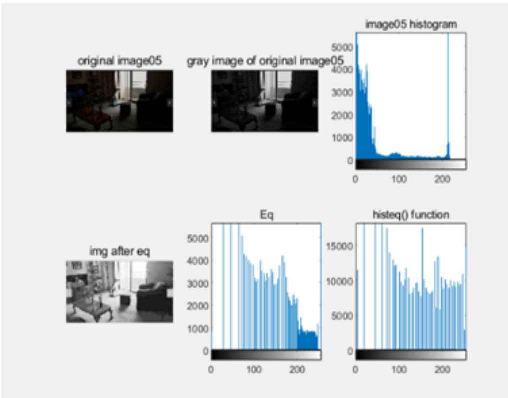
At last, we just need to Map the (L - 1) CDF to new grey-scale values.

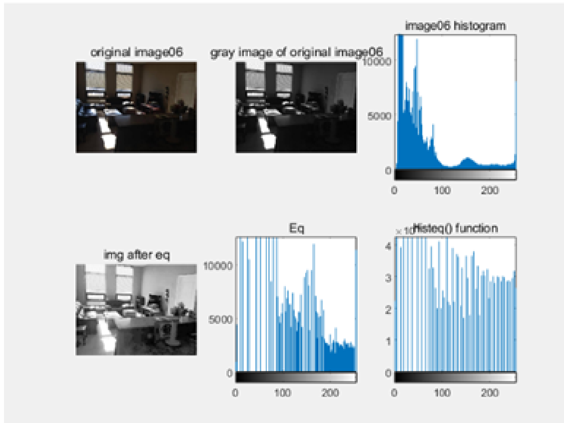
For the implementation of the code of the above functions. Firstly, the image should be turned into grey images using the function rgb2gray(), then for PMF, we performed pixel grey-scale statistics to count the number of occurrences of each pixel value, then the probability of them was also computed to get the probability histogram. For performing the CDF, the cumulative histogram was computed, and the output changed to an integer. The last step was to complete the mapping of each pixel using for loop. The results for each image are shown below, we also used the built-in function histeq() to compare the results.

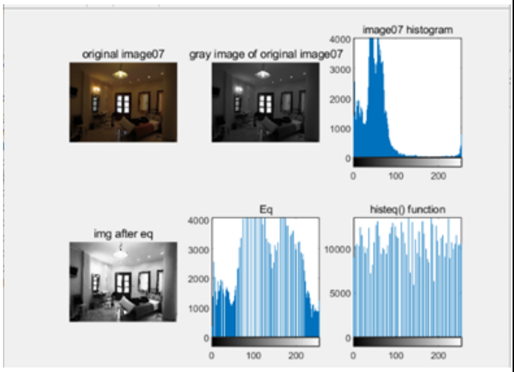












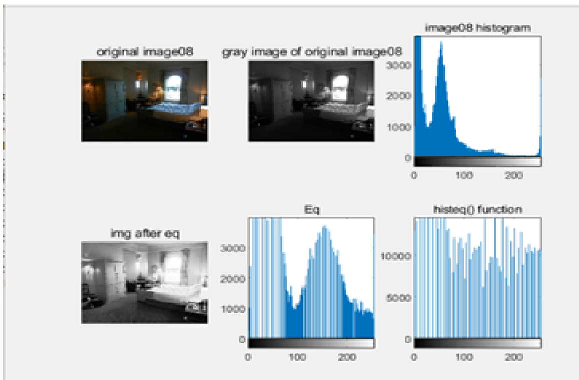
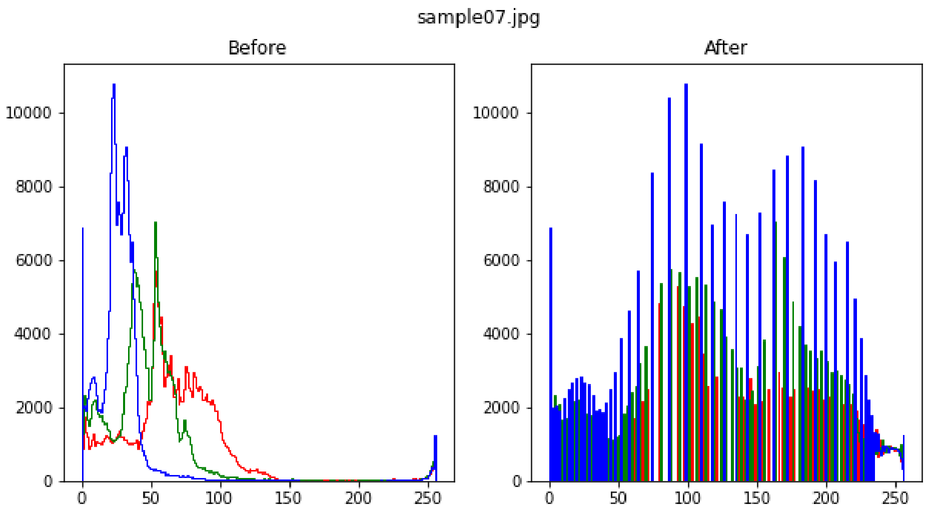


Table1 shows the results of 8 images, which indludes the original image, the grey image, the histogram of the origInal image, image after equalisation, the equalisation histogram and the test histogram of using *hiseq()* function.

Compared to the histogram by using built-in functions and original histograms, some of the results were not bad, for example, image04 and image05. But most of them did not perform as well as expected. So obviously, the algorithm should be improved. After searching and reading, we found that AHE and CLAHE may be helpful to deal with it. The implementation will be shown in the Further Work part.

2.2. Approach 2

Alternatively, we can also apply the same algorithm in Approach 1 to the RGB channels of the images individually as shown in “Assignment1.ipynb”. Below is an example of the histogram and image after the equalisation for “sample07.jpg” and more examples could be found in the Appendix “Histograms before and after the equalisation is performed on individual RGB channel”. All images processed using Approach 2 will have a prefix of “he_” in the file names.



3. Pros and cons of HE based on the enhanced sample images by the implemented HE algorithm

3.1. Pros

It is straightforward in implementation but yields reasonably good contrast enhancement. For instance, e are able to see the details of the dark regions in “sample05.jpeg” and “sample06.jpg” clearly after the histogram equalisation, as shown in “he_sample05.jpeg” and "he_sample06.jpg".

It has a lower computation cost compared to other methods such as adaptive histogram equalisation.

3.2. Cons and possible causes of unsatisfactory contrast enhancement

Since the histogram equalisation only considers global level contrast, for overexposed images, a high percentage of high-intensity values results in the background noise being increased due to over-contrast enhancement. For example, there is salt-and-pepper noise spotted on the building walls and background of "he_sample01.jpg".

For underexposed images, a high percentage of low-intensity value results in meaningful details in light regions that become unobservable after the histogram equalisation. For example, the edges of the ceiling light in "sample07.jpg" and the details outside the window in "sample08.jpg” are now difficult to tell.

For images with a very narrow range of intensity values, histogram equalisation will result in zero or low occurrence at certain intensity values. This results in images having a patchy look due to abrupt changes in intensity value over a small region.

For Approach 2, as the histogram equalisation is performed on each colour channel separately, some images will have unnatural colours. E.g. Unnatural purple hue found in "he_sample04.jpeg" which is supposed to be a forest scene.

4. Possible improvements & idea implementation

1. To overcome the issue caused by taking global level contrast, we can use adaptive histogram equalisation. Adaptive histogram equalisation is a method that performs histogram equalisation across multiple smaller regions of the image so local level contrast is accounted for.
2. We can limit the amount of stretching done by the histogram equalisation to prevent over-contrast enhancement that amplifies noise.
3. To prevent images from having unnatural colouration, we can perform histogram equalisation on other colour spaces such as CIELAB instead of RGB. Then, we can adjust the lightness while maintaining the chromaticity of the images.
4. There are also other more advanced methods that were not implemented in this report. These methods are namely, Multi-peak Histogram Equalisation (MPHE) and Multipurpose Beta Optimised Bihistogram Equalisation (MBOBHE). The focus of these two methods is to improve the contrast of the image without detail loss of the image.

4.1. Adaptive Histogram Equalisation(AHE)

From the results above, we can see that the quality of global images has improved, but some local parts could be still too bright or too dark. Adaptive Histogram Equalisation (AHE) is introduced to improve local contrast in the images and also to enhance the edges in each region of the image. AHE first computes several histograms with respect to the section of the images. By doing so, AHE redistributes the lightness values of the image locally.

However, a drawback of using AHE is that it can over-amplify noise in the image. This can be seen in the figure below.



When AHE was applied, sample image04 appeared to be worse than when normal Histogram Equalisation was used. It can be seen that the noise of the image was amplified in the white areas of the image. In order to tackle this problem, Contrast Limited Adaptive Histogram Equalisation (CLAHE) was used.

4.2. Contrast Limited Adaptive Histogram Equalisation (CLAHE)

One popular variant of adaptive histogram equalisation is Contrast Limited Adaptive Histogram Equalisation (CLAHE). Compared to ordinary adaptive histogram equalisation, this variant helps to reduce noise generated due to over-contrast enhancement in near homogeneous regions. In our case, we will use the CLAHE implemented by OpenCV, which involves 2 parameters - “clipLimit” and “tileGridSize”. “tileGridSize” determines the size of the small region that we want to perform histogram equalisation while “clipLimit” trims pixel values less than a certain count in the region. After multiple trials, we set “tileGridSize” at 2% of the width and height of the image and “clipLimit” at value of 3 since they produced reasonably good results.

Compared to the normal histogram equalisation, we did see an obvious improvement. For example, we have an image of “sample07.jpg” before applying any histogram equalisation (left), after applying normal histogram equalisation (middle) and after applying CLAHE (right). We observed that the details of the ceiling light, which were lost after applying normal histogram equalisation, can now be clearly seen with CLAHE.

Similarly, after applying CLAHE on “sample06.jpg”, the details outside the window and the bedsheet creases are now more obvious compared to when normal histogram equalisation was applied.

Lastly, with the contrast limiting component in CLAHE, the “sample01.jpg” image contains less noise (e.g. top left of the image after normal histogram equalisation) and looks less “patchy” caused by over-contrast enhancement in a near homogeneous region.

All enhanced gray images using CLAHE can be identified with the prefix of “clahe_gray_” followed by the sample image name (e.g. “clahe_sample01.jpg”) while enhanced coloured images can be identified with the prefix of “clahe_” followed by the sample image name (e.g. “clahe_sample01.jpg”).

4.3. CLAHE on CIELAB colour space

As mentioned earlier, performing histogram equalisation on the RGB channel can result in an unnatural hue since the image chromaticity is modified. Hence we can perform CLAHE on CIELAB colour space, which is defined by 3 parameters(l*, a*, b*). l* represents the perceptual lightness while a* and b* represent red, green, blue, and yellow. By performing CLAHE only on the perceptual lightness, we should be able to get more realistic images while achieving good contrast enhancement. For instance, we have an image of “sample02.jpeg” before applying any histogram equalisation (left), after applying CLAHE on RGB colour space(middle) and after applying CLAHE on CIELAB colour space(right). In the rightmost image, we observed that the cardboard colour is probably closer to reality and it does not have an unnatural blue colouration seen in the middle image.

However, we also noticed that this method seems to make the images more undersaturated compared to CLAHE on the RGB colour space, as shown in “sample04.jpeg”.

All enhanced images using CLAHE on CIELAB colour space can be identified with the prefix of “clahe_lab_” followed by the sample image name (e.g. “clahe_lab_sample01.jpg”)

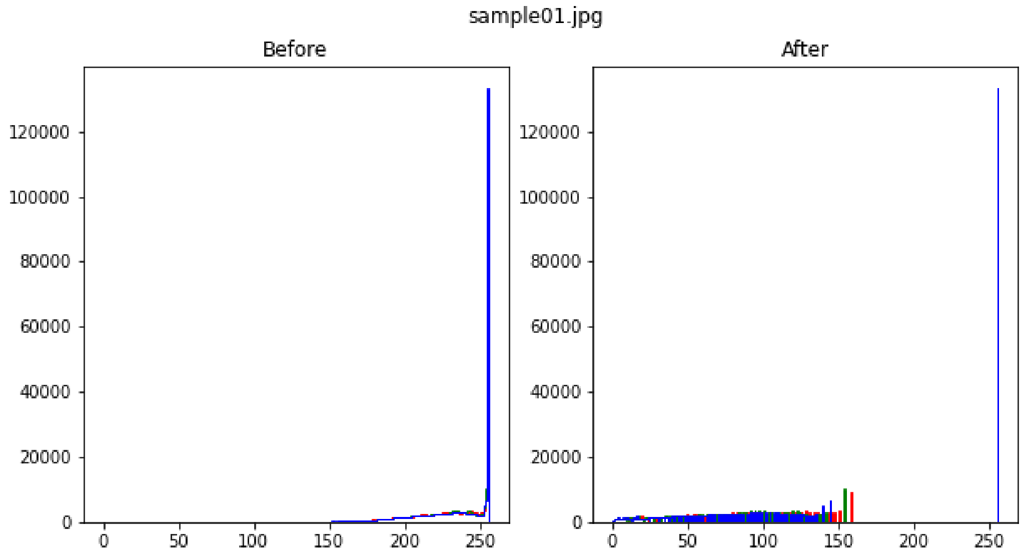
5. Summary

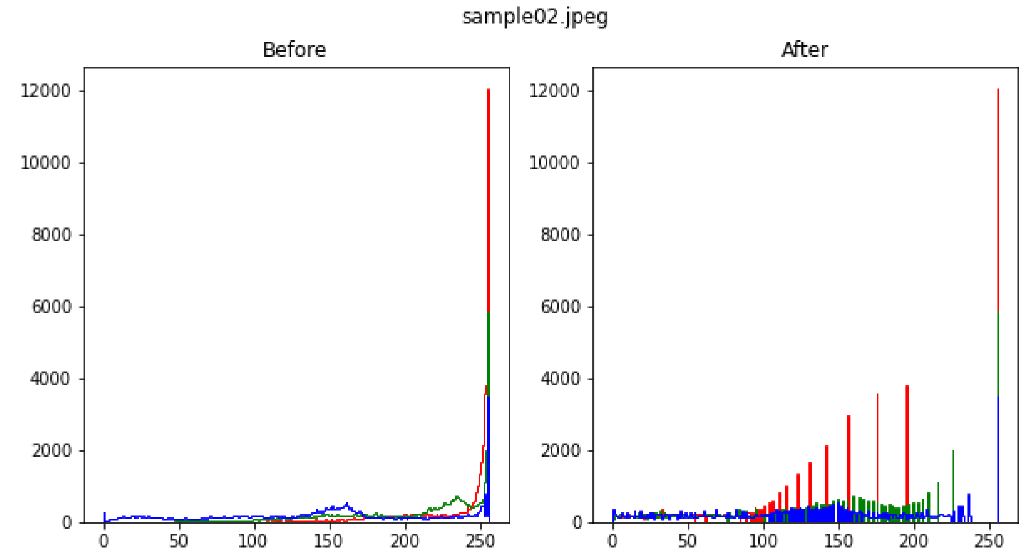
In this report, we showed details about how we understood and completed the Equalisation Histogram. The two discussions also showed our opinions depend on the source code and results. Additionally, we got more information about the AHE and CLAHE as an improvement to this task. .

Our team has implemented the equalisation histogram successfully both on normal and RGB channels. As working as a group, all of us participated in this task positively. Also, we discussed in a group chat which part to do and how to improve. From this assignment, we learnt from each other.

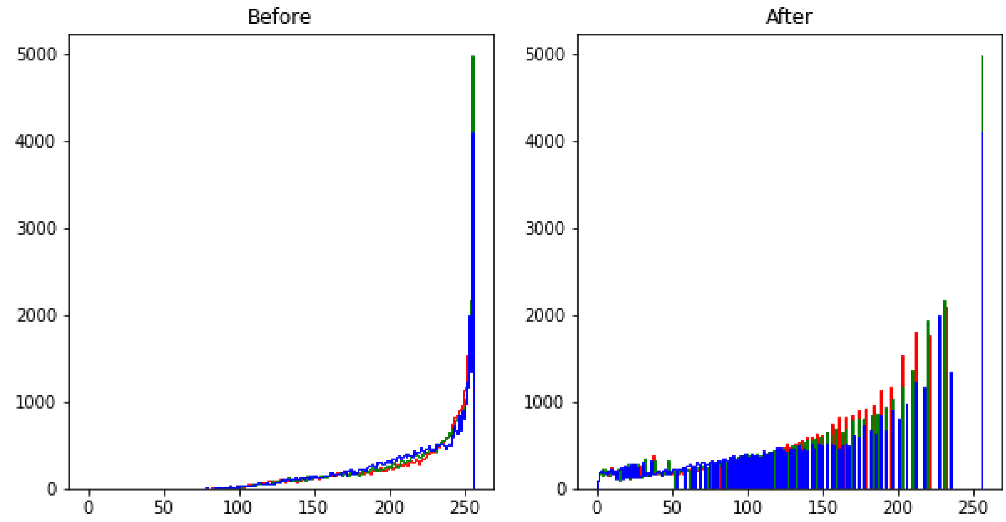
6. Appendix

6.1. Histograms before and after the normal histogram equalisation are performed on individual RGB channel

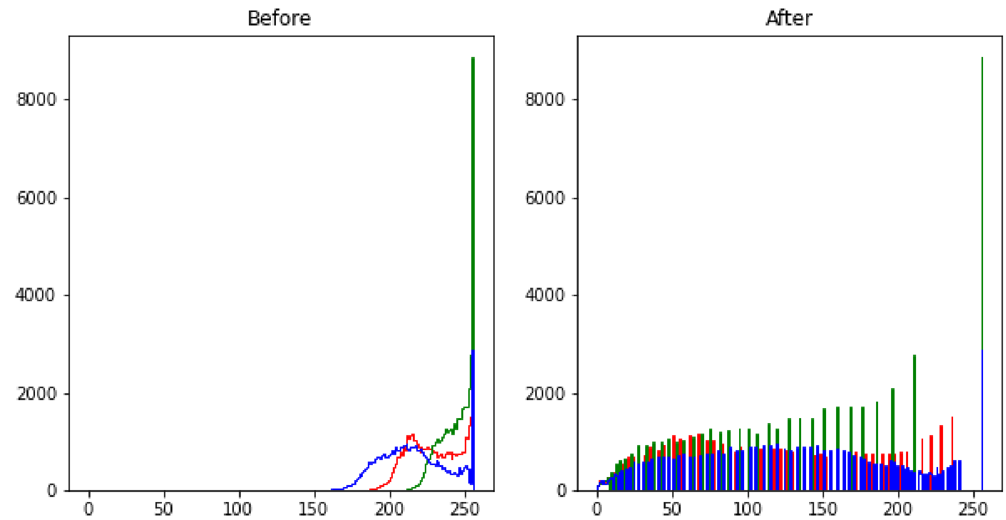




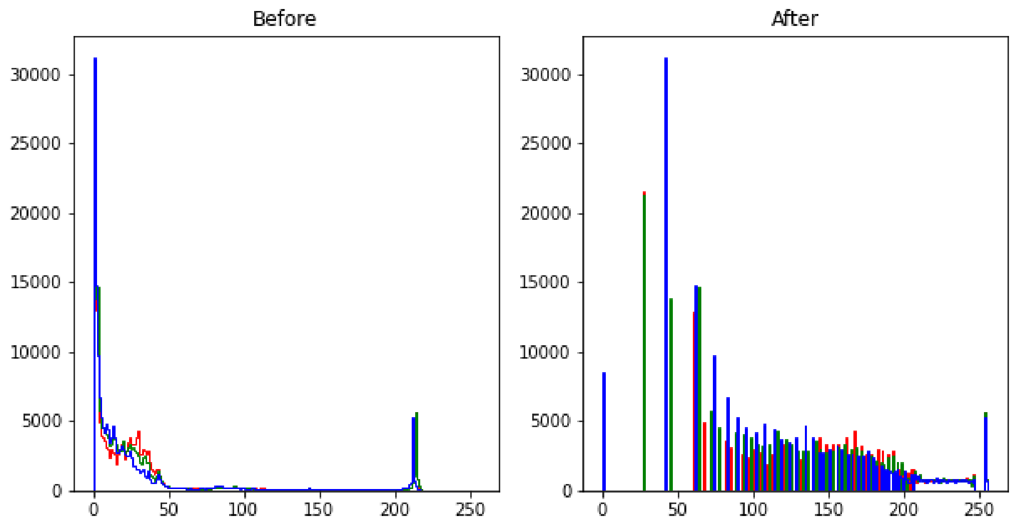
sample03.jpeg



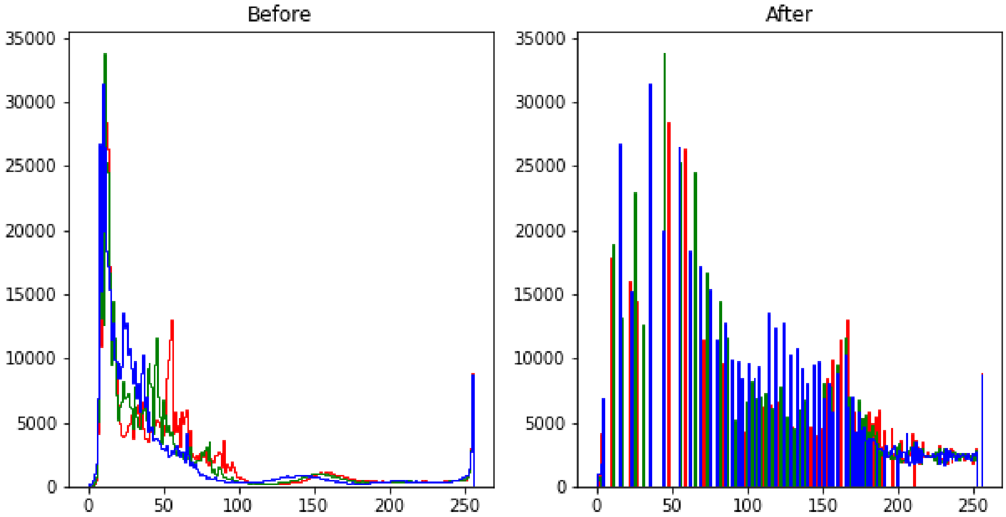
sample04.jpeg



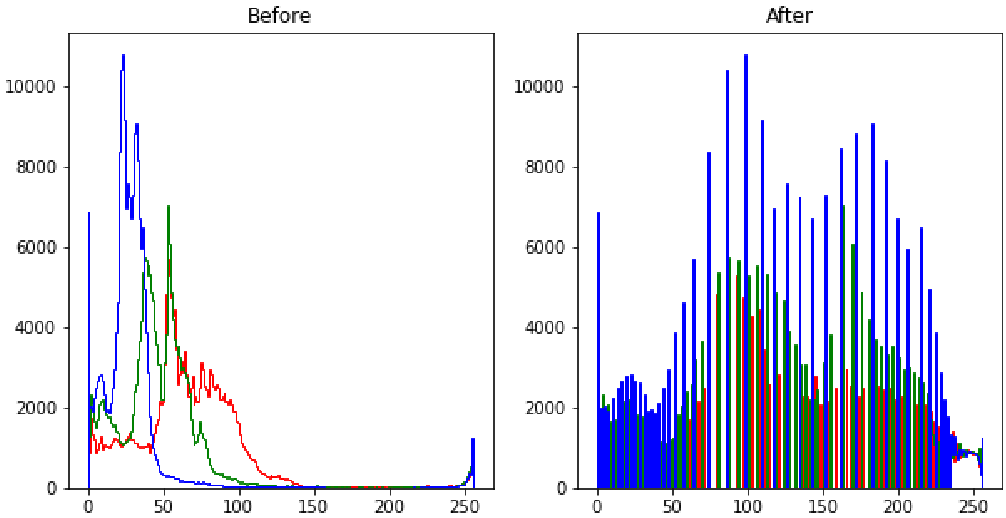
sample05.jpeg



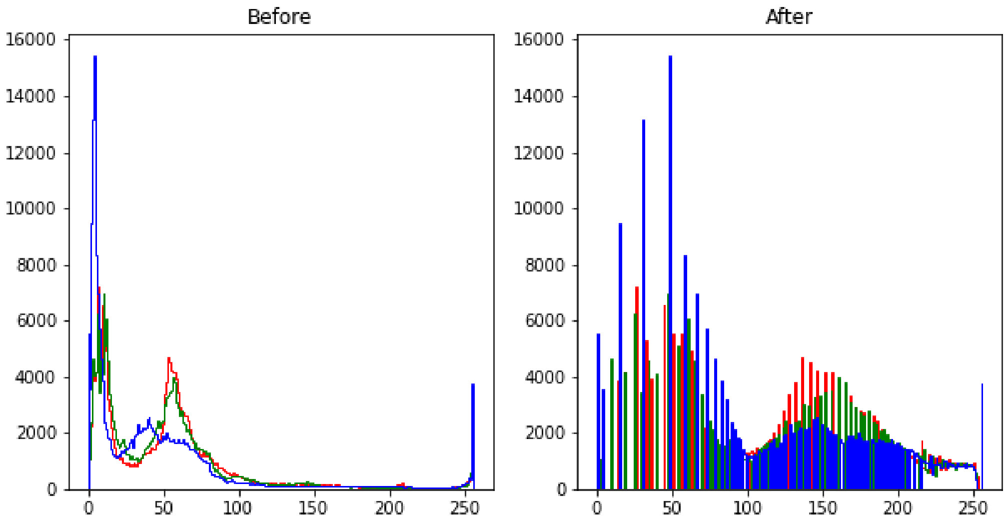
sample06.jpg



sample07.jpg



sample08.jpg



6.2. Source Code

The source code files can be checked in HEq1.m and Assignment1.ipynb within the .zip.

6.2.1. HEq1.m

6.2.2. Assignment1.ipynb