COMP 6771 Image Processing: Assignment 1

Student name: Yunqi Xu

Student id: 40130514

October 3, 2022

1. Question 1

   (a) Explain why the discrete histogram equalization technique does not, in general, yield a flat histogram. If we want a flat histogram, we need to guarantee each component has the same pixels in

   it. Let's set n be the total number of pixels, and L be the intensity levels. The flat histogram will require each component has n/L number of pixels. However, the Histogram Equalization process will re-mapping the pixels only based on their intensity level. So, we can not control the number of pixels in each conponent, it is difficult to yield a flat histogra.

   (b) Suppose that a digital image is subjected to histogram equalization. Please elaborate why a second pass of histogram equalization (on the histogram-equalized image) will produce exactly the same result as the first pass. Suppose an image has already pass a histogram equalization. Input this

   image into the second second pass of histogram equalization. Let n is the total number of pixels, and L is the intensity level.

$$s_k = T(r_k) = (L-1)\sum j = 0^k n_{rj}/n = \frac{L-1}{n}\sum_{j=0}^{k} n_{rj} = (L-1)\sum_{j=0}^{k} p_r(rj)$$

   we will use the equation mentioned above to calculate the s_k. Since every pixel belong to r_k is distributed to s_k. So, the $n_{sk} = n_{rk}$

$$p_s(s_k) = n_{sk}/n = n_{rk}/n = p_r(r_k)$$

   Also, no matter how many iterations,

$$new_k = (L-1)\sum_{j=0}^{k} p_s(s_k) = (L-1)\sum_{j=0}^{k} p_r(r_k) = s_k$$

   Therefore, the second pass will get the same result as the first pass.

2. Given two arbitrary images f(x,y) and g(x,y) and two arbitrary constant a and b, H is said to be a linear operator if:

$$H[af(x,y) + bg(x,y)] = aH[f(x,y) + bH[g(x,y)]]$$

   The median m of a set of numbers is such that half the values in the set are below m and the other half are above it (the mid-point value by population). Is an operator that computes the median of a set of pixels of a sub-image area linear or nonlinear? Explain your answer by giving examples.

   In general, the median operator is not linear operator. Since we can not control the order of values in $f(x,y)$ or $g(x,y)$. We will re-order the values in $f(x,y)$ and $g(x,y)$ and the the result of $af(x,y) + bg(x,y)$ before we get the median value.

   In some specific case, median operator will be linear, for example, if the the median value located in the center of $f(x,y)$ and $g(x,y)$ and the value in the left are smaller or equal then the median and the values in the right are greater or equal than the median in both $f(x,y)$ and $g(x,y)$, and after $af(x,y) + bg(x,y)$ the result still folloing the rules mentioned above, H will be a linear operator.

For example, suppose $f(x,y) = [1,-3,5,7,9]$ and $g(x,y) = [2,4,6,8,10]$ and $a = 2$, $b = 3$.

$$\begin{aligned} H[af(x,y) + bg(x,y)] &= H[[2,-6,10,14,18] + [6,12,18,24,30]] \\ &= H[8,6,28,38,48] \\ &= 28 \end{aligned} \tag{1}$$

$$\begin{aligned} aH[f(x,y)] + bH[g(x,y)] &= 2 * H[5] + 3 * H[6] \\ &= 10 + 18 \\ &= 28 \end{aligned} \tag{2}$$

Therefore, we can find that the Eq. 1 = Eq. 2.

On the other hand, If the values in $f(x,y)$ and $g(x,y)$ not following the rules mentioned about, the H is not a linear operator.

For example, Let us set $f(x,y) = [1,3,-5,7,9]$ and $g(x,y) = [2,4,6,810]$, also, $a = 2$, $b = 3$.

$$\begin{aligned} H[af(x,y) + bg(x,y)] &= H[[2,6,-10,14,18] + [6,12,18,24,30]] \\ &= H[8,18,8,38,48] \\ &= 18 \end{aligned} \tag{3}$$

$$\begin{aligned} aH[f(x,y)] + bH[g(x,y)] &= 2 * H[3] + 3 * H[6] \\ &= 6 + 18 \\ &= 24 \end{aligned} \tag{4}$$

Therefore, the Eq. 3 $\neq$ Eq. 4.

3. The purpose of this question is to perform histogram equalization to a given histogram and plot the resulting histogram. Given the following histogram where GL is Gray level, and NP is Number of pixels:

GL	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
NP	0	5	13	57	100	39	21	12	7	2	0	0	0	0	0	0

(a) Plot the histogram of the image given in the table above

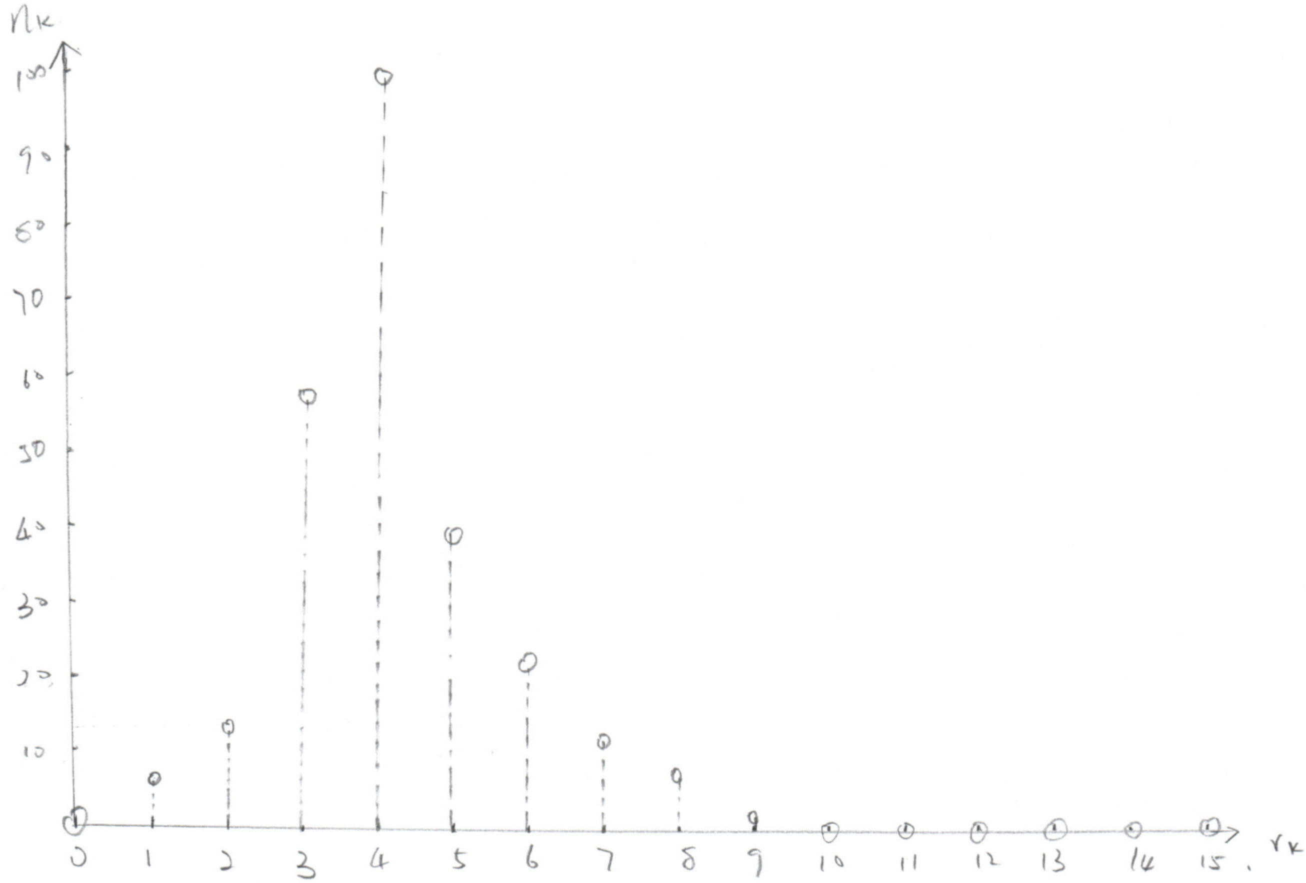

Figure 1: Histogram image

The Fig. 1 presents the histogram based on the given table.

i. Calculate sk from the table

GL	NP	$p_r(r_k)$	s_k	Round
0	0	0	0	0
1	5	0.0195	0.2929	0
2	13	0.0507	1.0546	1
3	57	0.2226	4.3945	4
4	100	0.3906	10.253	10
5	39	0.1523	12.539	13
6	21	0.0820	13.769	14
7	12	0.0468	14.472	14
8	7	0.0273	14.882	15
9	2	0.0078	15	15
10	0	0	15	15
11	0	0	15	15
12	0	0	15	15
13	0	0	15	15
14	0	0	15	15
15	0	0	15	15

Figure 2: Calculation result

The Fig. 2 shows the calculation result of s_k. In This Figure, Let GL be the Gray Level r_k, and NP is the number of pixels with value r_k, we will use n_k later. Based on the given table, the total pixel $n = 256$, and $L = 15$

$$p_r(r_k) = \frac{n_k}{n} \tag{5}$$

$$s_k = \frac{L-1}{n} \sum_{j=0}^{k} n_j \tag{6}$$

And in the Fig. 2, we utilize the Eq. 5 and Eq. 6 to calcuation the $p_r(r_k)$ and s_k respectively.

(b) Plot the probability density functions pr (rk) and ps (sk)

GL(s_k)	Mapped NP(n_{sk})	$p_s(s_k)$
0	5	0.0195
1	13	0.0507
2	0	0
3	0	0
4	57	0.2226
5	0	0
6	0	0
7	0	0
8	0	0
9	0	0
10	100	0.3906
11	0	0
12	0	0
13	39	0.1523
14	33	0.1289
15	9	0.0351

Figure 3: Calculation of ps(sk)

We following the Histogram Equalization process to calculate those values in Fig. 3.

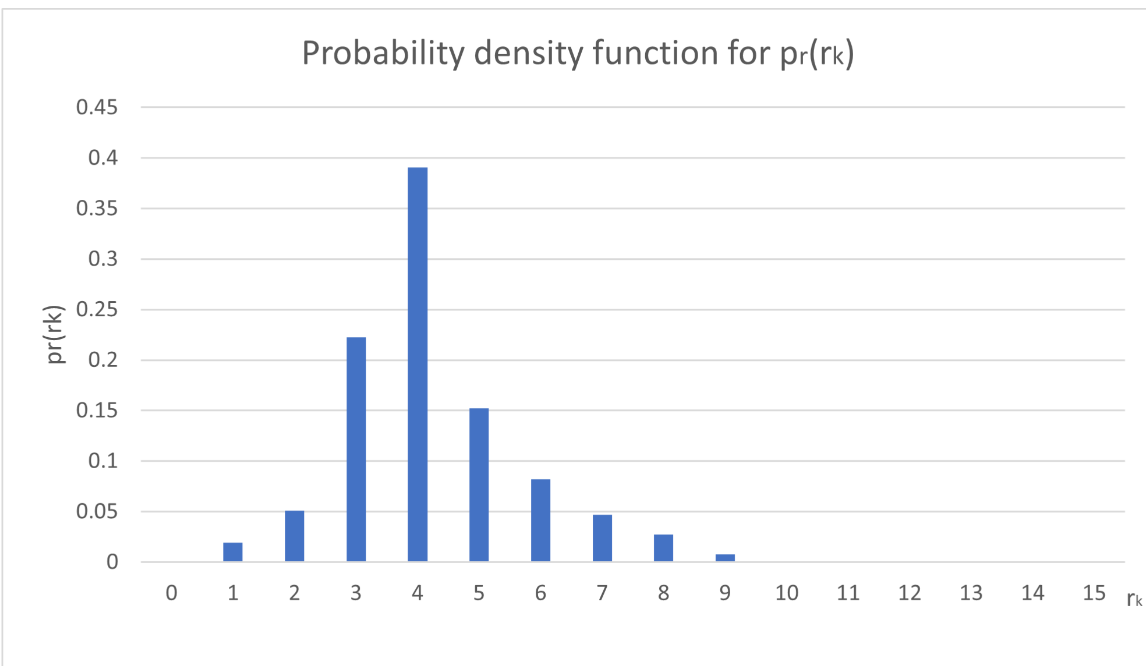

Figure 4: Probability densi function for pr(rk)

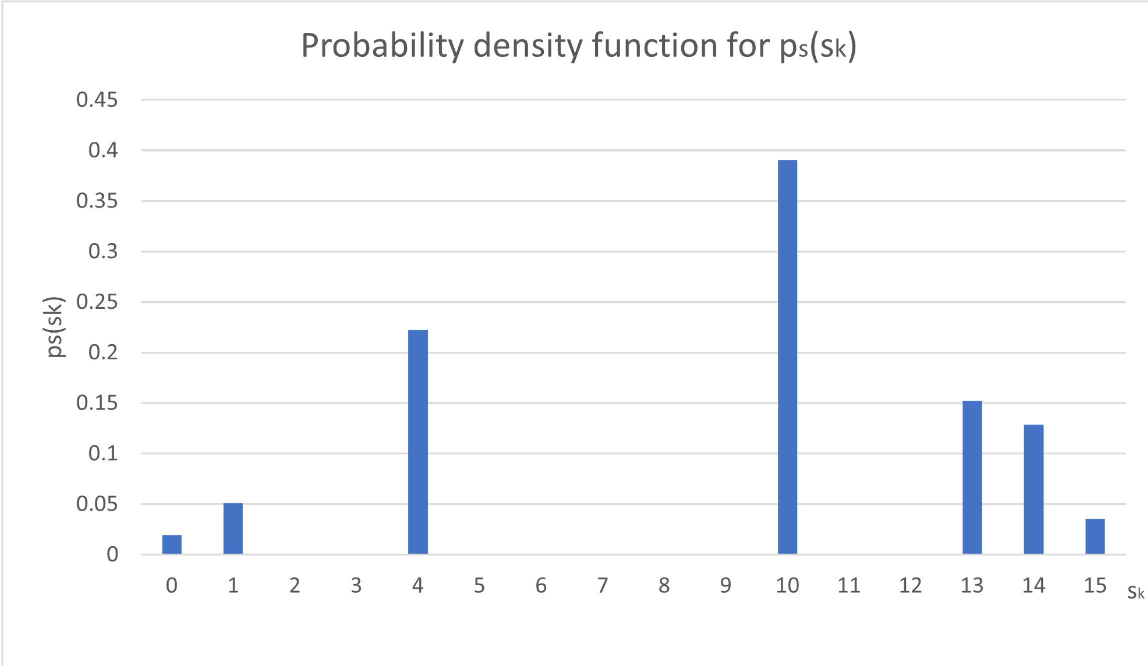

Figure 5: Probability density function for ps(sk)

Finally, we will get two probability density function present in Fig. 4 and Fig. 5.

(c) Plot the new histogram after performing the histogram equalization.

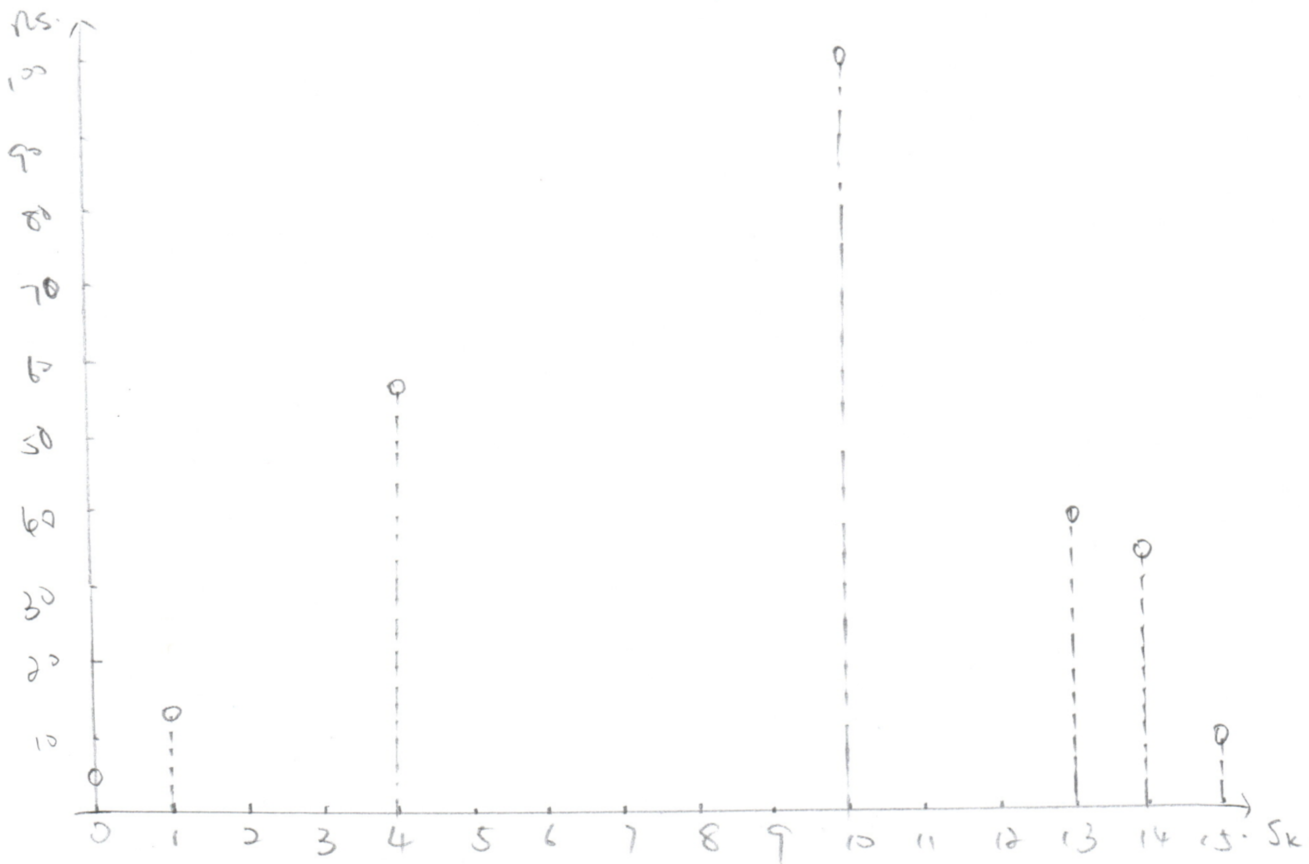

Figure 6: New histogram after histogram equalization

The new histogram is presented in Fig. 6.

4. The Given two images f(x,y) and g(x,y) with histogram hf and hg . Assuming that all the pixels of image g(x,y) have the same intensity equal to a non-zero positive constant value c. The gray levels of the pixels of both images have positive values. Please explain the relationship between the histograms of f(x,y) and the new images formed by f(x,y) + g(x,y) and f(x,y)*g(x,y)

   (a) Since all the pixels of image $g(x, y)$ have the same intensity equal to a non-zero positive constant value c. Set, $n(r_k)$ be the number of pixels belong to the intensity level r_k, then, the $f(x, y) + g(x, y) = f(x, y) + c$, So, the $n(r_k)$ will not change, but $r'_k = r_k + c$. So, the histogram of new image $f(x, y) + g(x, y)$ will have the same hight as the histogram of $f(x, y)$ but all components will move to right with c.

   (b) As mentioned above, t$f(x, y) * g(x, y) = f(x, y) * c$, So the hight will not change after the product operator, but the distance beteen each conponent will be enlarged by c, as $r'_k = r_k * c$. For example, $r_k = 1$ will go to $r_k = c$ and $r_k = 2$ will go to $r_k = 2c$, $r_k = 3$ will go to $r_k = 3c$ ...

5. Download the image from the assignment folder, and perform the following operations using MATLAB or any software packages you are familiar with. Please show your steps in the report.

```
import numpy as np
import cv2
from matplotlib import pyplot as plt
```

(a) Write a program to read the grayscales of the image.

```
def read_grayscales(img):
        img = cv2.imread(img,0)
        return img
img_name = 'HawkesBay.jpeg'
img = read_grayscales(img_name)
```

(b) Write a program to calculate the histogram of the image and display the histogram chart.

```
def calculate_histogram(img, L):
    image_flatten = img.flatten()
    new_list = np.zeros(L)
    for each_pixel in image_flatten:
        new_list[each_pixel] += 1
    return new_list

def histogram_show(hist):
    plt.bar(range(len(hist)), hist)
    plt.xlabel("rk")
    plt.ylabel("nk")
    plt.title("Histogram of imput image")
    plt.savefig("Histogram_input")

histogram_statis = calculate_histogram(img, 256)
histogram_show(histogram_statis)
```

The histogram is presented in Fig.7.

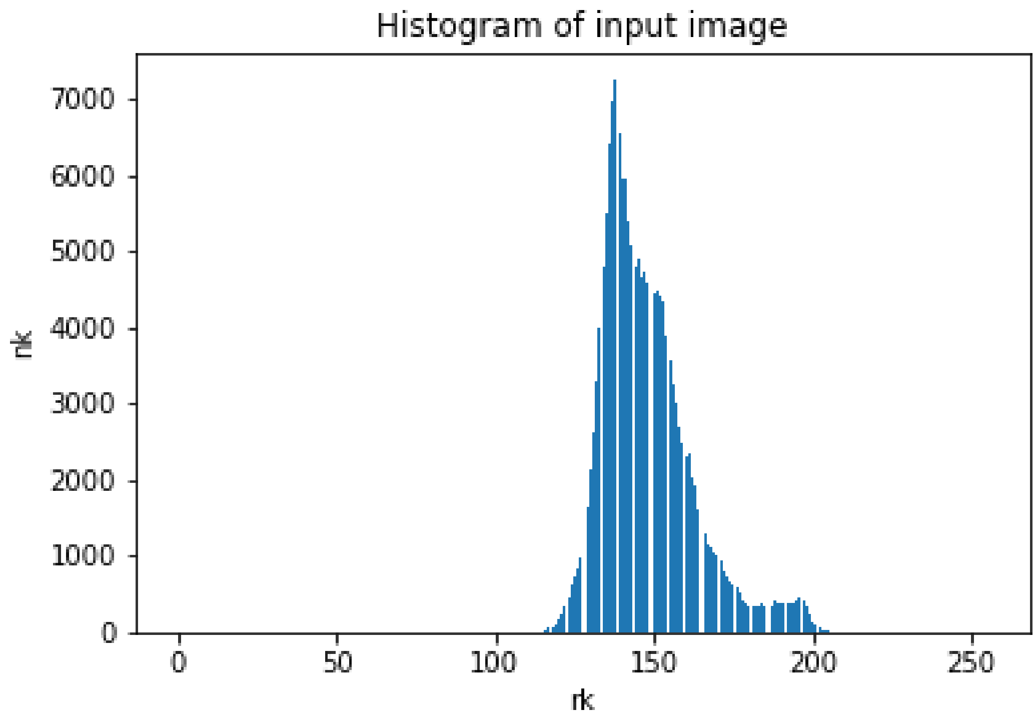

Figure 7: Histogram of input image

(c) Compare the calculated histogram obtained by using your own program with the one using the imhist function in MATLAB.

```
img = imread("HawkesBay.jpeg")
imhist(img,256)
```

The histogram from Matlab imhist function is present as Fig. 8

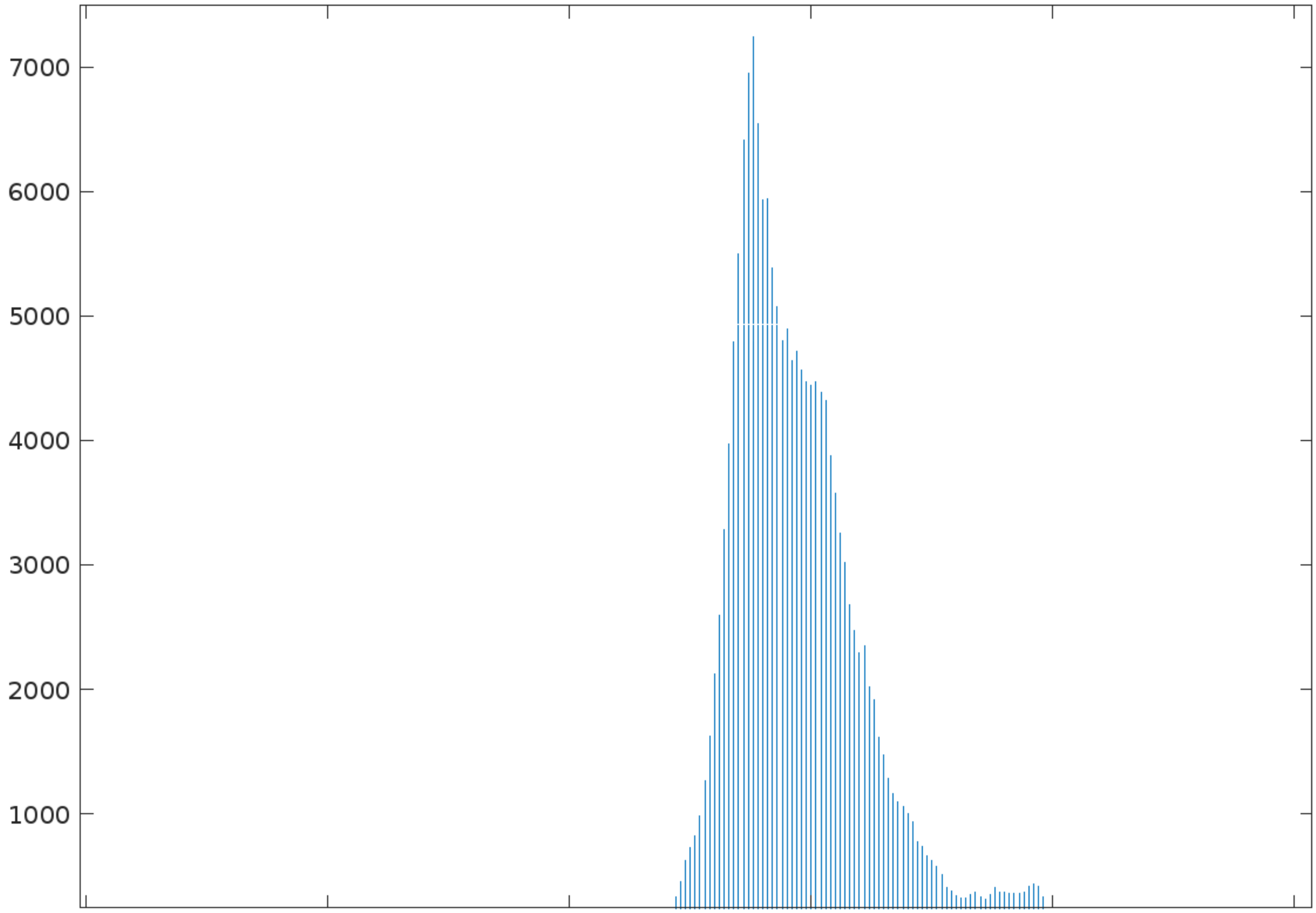

Figure 8: Histogram of Matlab imhist function

(d) Write a program to do histogram equalization on this image.

```
def calculate_prob(hist,total_pixels):
    return [values_each_bin/total_pixels for values_each_bin in hist]

def calculate_sk(cal_pr, L):
    new_list = np.zeros(L)
    for index, each_pr in enumerate(cal_pr):
        cal_result = (L-1) * sum(cal_pr[0: index+1])
        new_list[index] = round(cal_result)
    return new_list

def new_image(img, sk):
    height, width = img.shape
    new_img = np.zeros(img.shape, dtype = 'uint8')
    for i in range(img.shape[0]):
        for j in range(img.shape[1]):
```

```
                new_img[i, j] = sk[img[i,j]]
        return new_img
tp = img.shape[0]*img.shape[1]
pr = calculate_prob(histogram_statis, tp)
sk = calculate_sk(pr, 256)
new_img = new_image(img, sk)
```

(e) Compare the histogram-equalized image obtained by using your own program with the one by using histeq function in MATLAB.

```
J = histeq(I,256)
imshow(J)
```

The new image from matlet histeq function is presented in Fig. 9

Figure 9: Histogram-equalized image obtained by Matlab histeq

```
cv2.imwrite("he_newimage.png",new_img)
```

The new image from the histogram equalization problem is shown in Fig. 10

Figure 10: "histogram-equalized image obtained by my own program"

This two histogram-equalized images are very similar.