B06502149 資工三 張琦琛

(a) Original image and its histogram

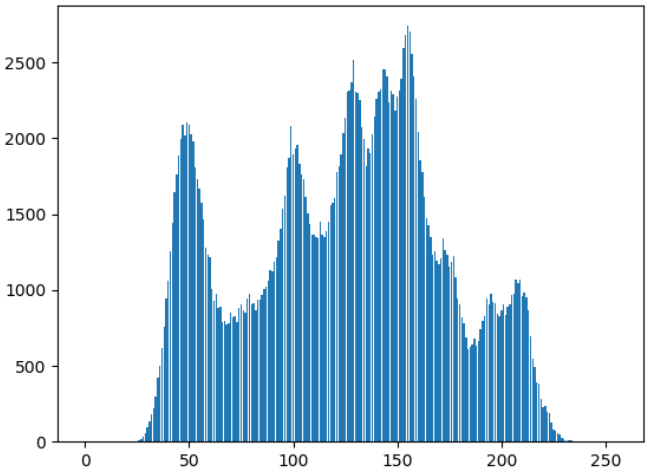
Count every pixel's intensity and save it in the intensity list x[256], and use matplotlib to generate histogram.

Code fragment:

The input image is the original lena.bmp, and K = 1

```
x = np.zeros(256, dtype=int)
for i in range(r):
    for j in range(c):
        intensity = img[i][j] // k
        x[intensity] += 1
plt.bar(np.arange(256), x)
plt.savefig("histogram"+str(k)+".png")
plt.close()
```

Result:



(b) Image with intensity divided by 3 and its histogram

For the first part, we let the intensity of each pixel divided by 3; that is, img[i][j] = img[i][j] / 3.

Code fragment:

```
def dark(img):
    r = img.shape[0]
    c = img.shape[1]
    for i in range(r):
        for j in range(c):
            img[i][j] /= 3
    cv2.imwrite('dark.bmp', img)
```

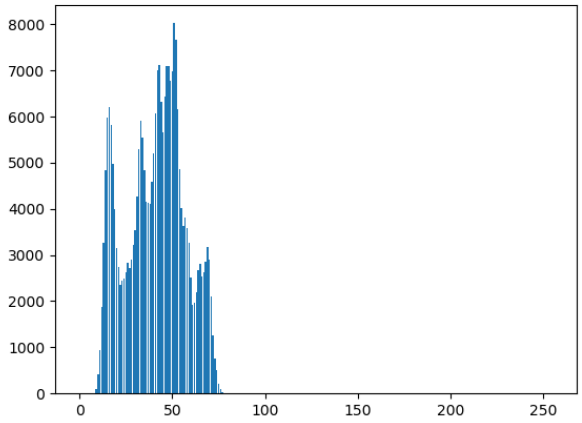
For the second part, we count every pixel's intensity of the image after division and save it in the intensity list x[256], and use matplotlib to generate histogram.

Code fragment:

The input image is the original lena.bmp, and K = 3

```
for i in range(r):
    for j in range(c):
        intensity = img[i][j] // k
        x[intensity] += 1
plt.bar(np.arange(256), x)
plt.savefig("histogram"+str(k)+".png")
plt.close()
```

Result:



(c) Image after applying histogram equalization to (b) and its histogram

We first calculate s_k,

$$s_k = 255 \sum_{j=0}^{k} \frac{n_j}{n} \qquad k = 0, 1, \ldots, 255$$

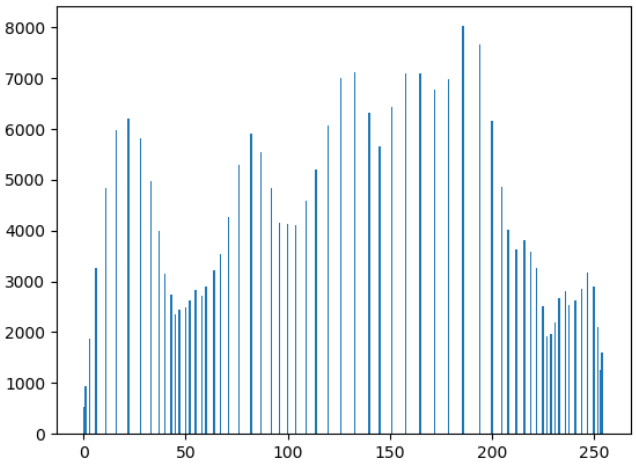
n is the total number of pixels and n_j denotes the total number of pixels with intensity j. Then, for each pixel in the image, we replace the original value with s_k; that is, if $I(i,j) = k$, then $I_{HE}(i,j) = s_k$.

Code fragment:

```
x = np.zeros(256, dtype=int)
prefix = np.zeros(256, dtype=int)
s = np.zeros(256, dtype=int)
for i in range(r):
    for j in range(c):
        x[img[i][j]] += 1
prefix[0] = x[0]
for i in range(1,len(x)):
    prefix[i] = x[i] + prefix[i-1]

for i in range(r):
    for j in range(c):
        img[i][j] = 255*(prefix[img[i][j]] / (r*c))
cv2.imwrite('equalization.bmp', img)
```

Result:


8000
7000
6000
5000
4000
3000
2000
1000
0
0
50
100
150
200
250