ECE63700-Lab4

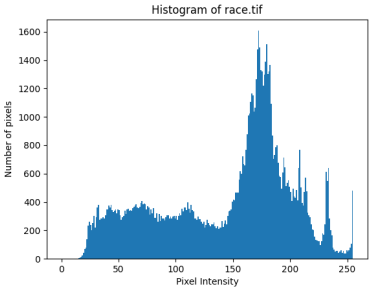
Sriranga R

February 4, 2023

1 Histogram of an Image

1.1 Images and their histogram

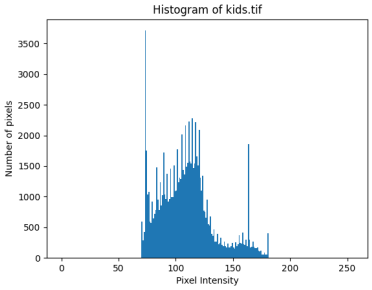
(a) race.tif



(b) Histogram of race.tif

Figure 1: race.tif and its histogram

(a) kids.tif



(b) Histogram of kids.tif

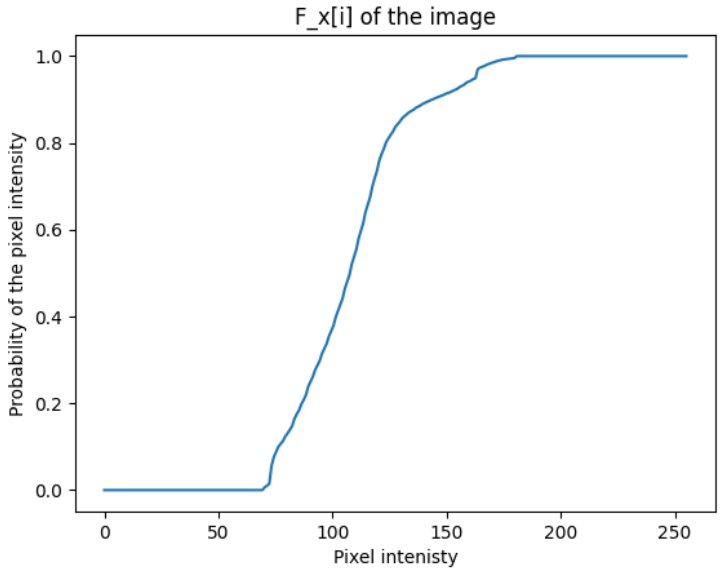
Figure 2: kids.tif and its histogram

2 Histogram Equalization

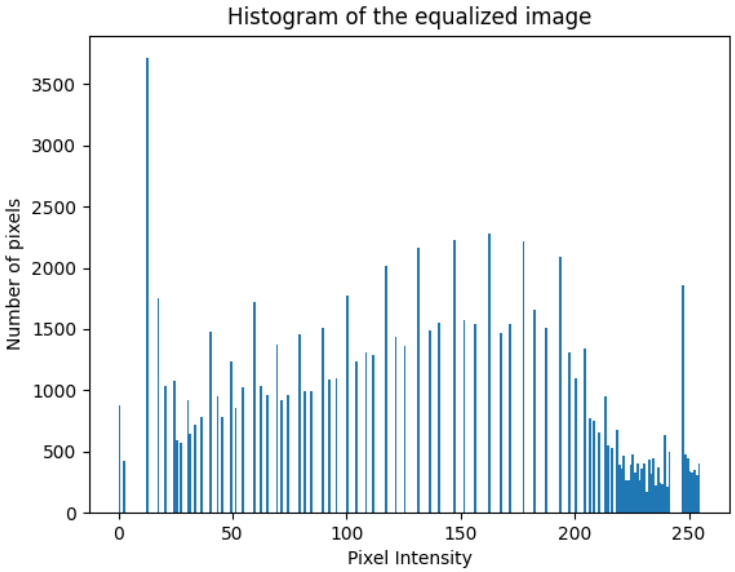
2.1 equalize.py

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
from PIL import Image

def equalize(image) :
    gray = cm.get_cmap('gray', 256)
    im = Image.open(image)
    x = np.array(im)
    plt.imshow(im, cmap=gray)
    plt.title("Original image")
    plt.show()
    height_im, width_im = x.shape
    print(height_im, width_im)
    height, width = np.histogram(x.flatten(), bins=np.linspace
    (0,255,256))
    print(len(height))
    total_sum = sum(height)
    print('Sum = ', total_sum)
    cdf = np.zeros((256, 1))
    value = 0.0
    for i in range(0,len(height)):
        value += height[i]
        cdf[i] = value / total_sum
    plt.plot(np.linspace(0,255,255), cdf[0:255])
    plt.title("F_x[i] of the image")
    plt.xlabel("Pixel intenisty")
    plt.ylabel("Probability of the pixel intensity")
    plt.show()
    Y_s = [cdf[i] for i in x.flatten()]


    print(len(Y_s))
    min_ys = min(Y_s)
    max_ys = max(Y_s)
    print(min_ys, max_ys)
    Z_s = [((255)*((Y_s[i] - min_ys)/(max_ys - min_ys))) for i in
    range(0, len(Y_s))]
    Z_image = np.reshape(Z_s, (height_im, width_im))
    plt.imshow(Z_image, cmap=gray)
    plt.title("Equalized image")
    plt.show()
    plt.hist(Z_image.flatten(), bins=np.linspace(0,255,256))
    plt.title("Histogram of the equalized image")
    plt.xlabel("Pixel Intensity")
    plt.ylabel("Number of pixels")
    plt.show()
    #for i in range(0,len(height)):

equalize("kids.tif")
```

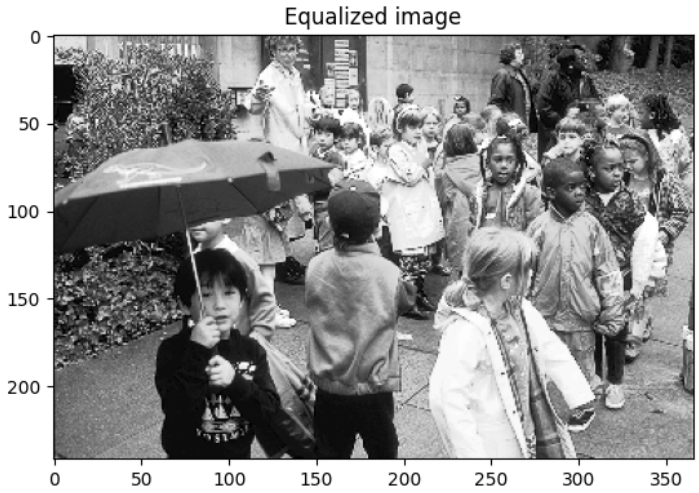
2.2 Plot of CDF of $\hat{F}_x(i)$ for kids.tif



2.3 Histogram of the equalized image



2.4 Equalized image



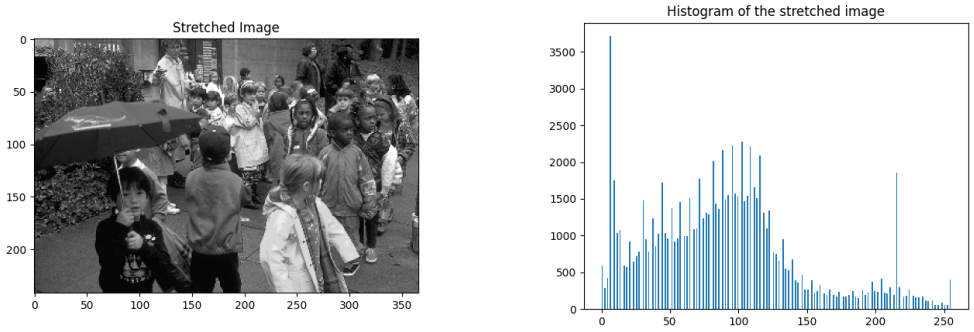
3 Contrast Stretching

3.1 Code for stretch.py

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
from PIL import Image

def stretch(input_img, T1, T2):
    input = input_img.flatten()
    print(input.shape)
    output = np.zeros(input.shape)
    for i in range(0, len(input )):
        if input[i] < T1:
            output[i] = 0
        elif input[i] > T2 :
            output[i] = 255
        else :
            output[i] = ((255 * (input[i] - T1)) / (T2 - T1))
    output = output.astype(np.uint8)
    output_img = output.reshape((input_img.shape[0], input_img.
    shape[1]))
    plt.imshow(output_img, cmap=gray)
    plt.title("Stretched Image")
    plt.show()
    return output_img

```

```
24
25 gray = cm.get_cmap('gray', 256)
26 im2 = Image.open("kids.tif")
27 x = np.array(im2)
28 plt.hist(x.flatten(), bins=np.linspace(0,255,256))
29
30 plt.title("Histogram of kids.tif")
31 plt.xlabel("Pixel Intensity")
32 plt.ylabel("Number of pixels")
33 plt.show()
34
35 height, width = np.histogram(x.flatten(), bins=np.linspace
       (0,255,256))
36 print(len(height))
37 print(len(width))
38 plt.plot(np.linspace(0,255,len(height)), height)
39 plt.show()
40
41 height, width = np.histogram(x.flatten(), bins=np.linspace
       (0,255,256))
42 output_img = stretch(x, 70, 180)
43 plt.hist(output_img.flatten(), bins=np.linspace(0,255,256))
44 plt.title("Histogram of the stretched image")
45 plt.show()
```

3.2 Stretched image and histogram



(a) Stretched image

(b) Histogram of stretched image

Figure 3: Stretched image and histogram

4 Gamma (γ)

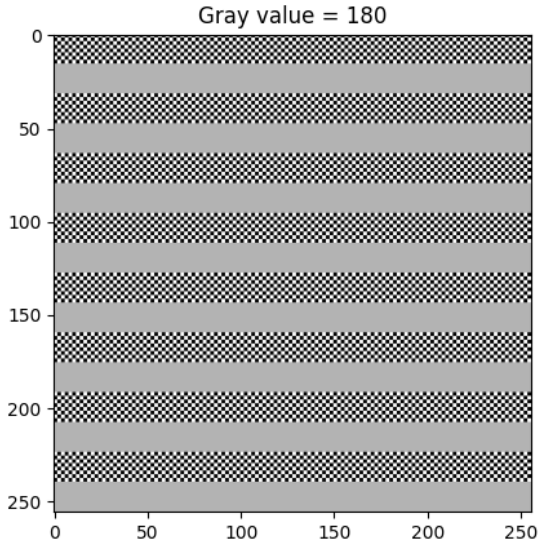
4.1 Setting the Black Level and Picture of Your Monitor

Nothing to report in this section

4.2 Determining the Gamma of Your Computer Monitor

The gray scale value g is set at 180 after performing the experiment described in the lab instructions.

4.2.1 Image corresponding to the matching gray level



4.2.2 Derivation of Gamma

$$I_c = \frac{I_{255}}{2}$$

$$I_g = I_{255}(\frac{g}{255})^\gamma$$

$$I_c = I_g$$
$$\frac{1}{2} = (\frac{g}{255})^\gamma$$

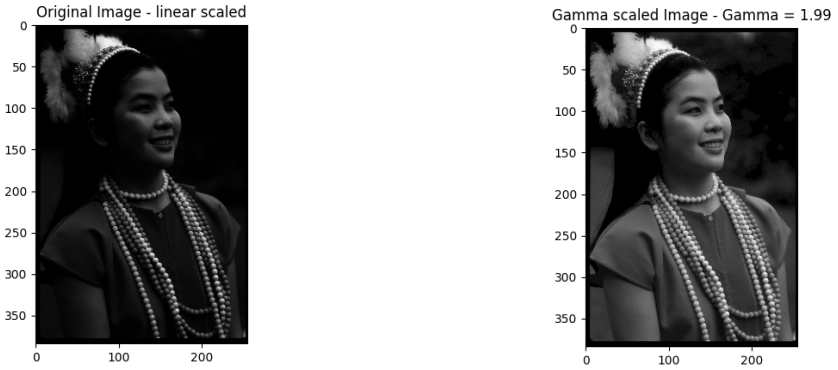
$$\ln \frac{1}{2} = \gamma \ln \frac{g}{255}$$
$$\gamma = \frac{\ln \frac{1}{2}}{ln \frac{g}{255}}$$
$$We\ obtained, g = 180$$
$$\gamma = \frac{\ln \frac{1}{2}}{ln \frac{180}{255}}$$
$$\gamma = 1.99$$

4.2.3 Gray value and Gamma measured

$$gray\ value, g = 180$$
$$\gamma = 1.99$$

4.3 Gamma Correction

4.3.1 Gamma scaled image



(a) Original Image

(b) Gamma scaled image, $\gamma = 1.99$

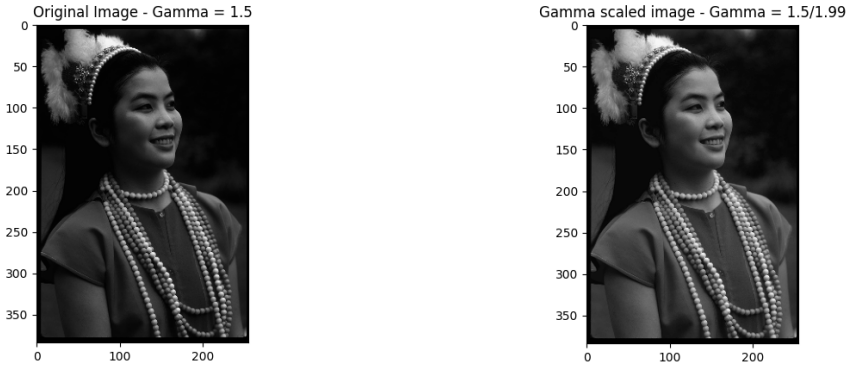
Figure 4: Gamma Scaling

4.3.2 Formula used

$$y = 255(\frac{x}{255})^(1/\gamma) \quad (1)$$

4.4 Gamma Scaled 2

4.4.1 Gamma scaled images



(a) Original Image

(b) Gamma scaled image, $\gamma = \frac{1.5}{1.99}$

Figure 5: Gamma Scaling

4.4.2 Procedure you used to change the gamma correction of the original image

Gamma scaled image as input

$$y = 255(\frac{x}{255})^{\gamma_1}$$

$$z = 255(\frac{x}{255})^{\frac{1}{\gamma_2}}$$

Substitute for y in the above equation, we get

$$z = 255(\frac{x}{255})^{\frac{\gamma_1}{\gamma_2}}$$

5 Additional Code Listings

5.1 Gamma Scaling

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
from PIL import Image
import sys

```

```
def getGammaValue(grayval):
    gray = cm.get_cmap('gray', 256)
    checkerBoardPattern = np.array([[256, 256, 0 , 0], [256, 256,
    0,0], [0,0,256,256], [0,0,256,256]])
    #plt.imshow(checkerBoardPattern, cmap=gray)
    #plt.show()

    tile_count = int(256/4)
    #plt.imshow(np.tile(np.tile(checkerBoardPattern, tile_count),
    (4,1)), cmap=gray)
    output = np.tile(np.tile(checkerBoardPattern, tile_count),
    (4,1))
    grays = (np.ones(16*256) * int(grayval)).reshape(16,256)
    output = np.concatenate((output, grays), axis=0)
    final_output = np.tile(output,(8,1))
    plt.imshow(final_output, cmap=gray)
    plt.title("Gray value = " + str(grayval))
    plt.show()

getGammaValue(sys.argv[1])

gray = cm.get_cmap('gray', 256)
im = Image.open("linear.tif")
plt.imshow(im, cmap=gray)
plt.title("Original Image - linear scaled")
plt.show()
x = np.array(im)
gamma = 1.99
#set at 180.

y = np.array([(255 * (i/255)**(1/gamma)) for i in x.flatten()]).
    reshape((x.shape[0], x.shape[1])).astype(np.uint8)
plt.imshow(y, cmap=gray)
plt.title("Gamma scaled Image - Gamma = 1.99")
plt.show()


im2 = Image.open("gamma15.tif")
plt.imshow(im2, cmap=gray)
plt.title("Original Image - Gamma = 1.5")
plt.show()
x = np.array(im2)
gamma = 1.99

y = np.array([(255 * (i/255)**(1.5/gamma)) for i in x.flatten()]).
    reshape((x.shape[0], x.shape[1])).astype(np.uint8)
plt.imshow(y, cmap=gray)
plt.title("Gamma scaled image - Gamma = 1.5/1.99")
plt.show()
```