**PYTHON FOR VISION TECHNIQUES**

**WEEK - 1**

**Name:** Vatsal Panchal

**Registration number:** 18BEC1125

**### Sample code for studying the color-plane information**

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.image as mpimg

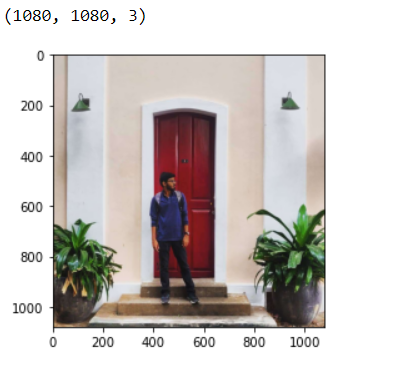
img = mpimg.imread('C:/Users/WELCOME/Downloads/IMG\_20190902\_190953\_399.jpg')

plt.figure(1)

plt.imshow(img)

img.shape

**OUTPUT -**



**### Sample code for studying the color-plane information**

imgR = img[:,:,0]

imgG = img[:,:,1]

imgB = img[:,:,2]

plt.figure(2)

plt.imshow(imgR, cmap = 'gray')

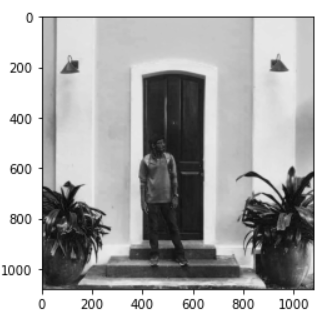
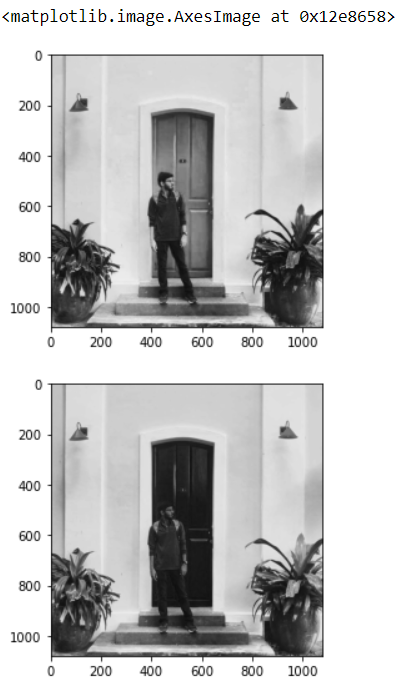
plt.figure(3)

plt.imshow(imgG, cmap = 'gray')

plt.figure(4)

plt.imshow(imgB, cmap = 'gray')

**OUTPUT -**



**INFERENCE­ ­-** As we can see from the original image, red is dominant on the door whereas green is dominant in the leaf areas. The colour blue is lightened in picture 3

**### Sample code for studying the histogram**

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.image as mpimg

img = mpimg.imread('C:/Users/WELCOME/Downloads/IMG\_20190902\_190953\_399.jpg')

def rgb2gray(rgb):

r, g, b = rgb[:,:,0], rgb[:,:,1], rgb[:,:,2]

gray = 0.2989 \* r + 0.5870 \* g + 0.1140 \* b

return gray

img\_gray = rgb2gray(img)

plt.figure(1)

plt.imshow(img\_gray,cmap='gray')

plt.figure(2)

plt.show()

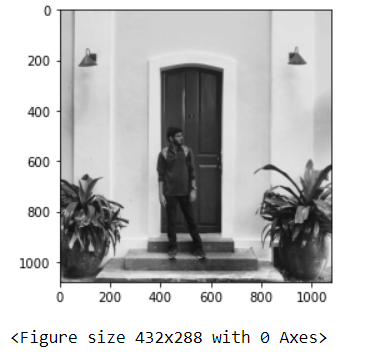
plt.hist(img\_gray)

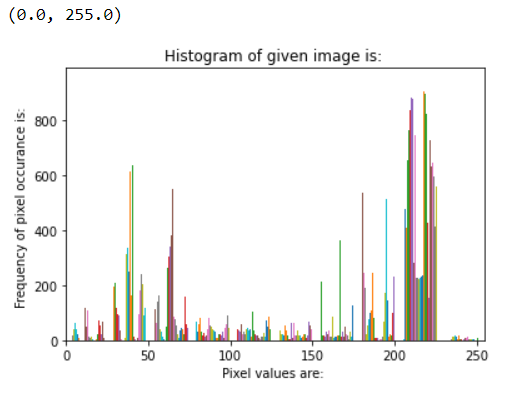
plt.title("Histogram of given image is: ")

plt.xlabel("Pixel values are: ")

plt.ylabel("Frequency of pixel occurance is: ")

plt.xlim([0, 255])





**INFERENCE­ ­–** This is for normal brightness normal contrast image as is evident from the graph.

**HIGH** **CONTRAST IMAGE**

**### Sample code for studying the histogram**

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.image as mpimg

img = mpimg.imread('C:/Users/WELCOME/Downloads/Image.jpg')

def rgb2gray(rgb):

r, g, b = rgb[:,:,0], rgb[:,:,1], rgb[:,:,2]

gray = 0.2989 \* r + 0.5870 \* g + 0.1140 \* b

return gray

img\_gray = rgb2gray(img)

plt.figure(1)

plt.imshow(img\_gray,cmap='gray')

plt.figure(2)

plt.show()

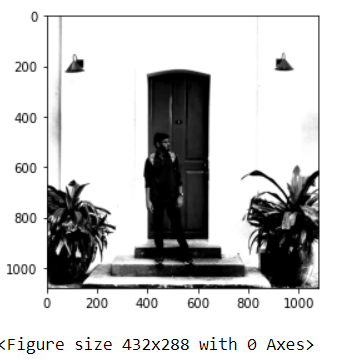
plt.hist(img\_gray)

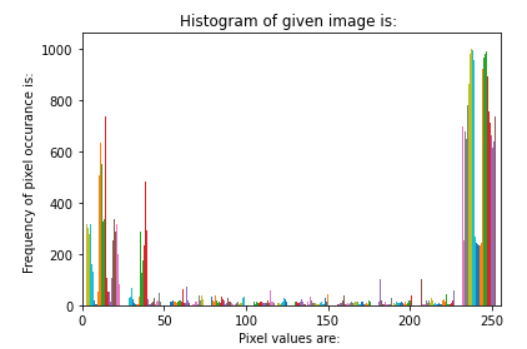
plt.title("Histogram of given image is: ")

plt.xlabel("Pixel values are: ")

plt.ylabel("Frequency of pixel occurance is: ")

plt.xlim([0, 255])





**INFERENCE­ ­–** This is for high contrast and the values are concentrated in far maximum and far minimum of X axis.

**LOW** **CONTRAST IMAGE**

**### Sample code for studying the histogram**

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.image as mpimg

img = mpimg.imread('C:/Users/WELCOME/Downloads/lol.jpg')

def rgb2gray(rgb):

r, g, b = rgb[:,:,0], rgb[:,:,1], rgb[:,:,2]

gray = 0.2989 \* r + 0.5870 \* g + 0.1140 \* b

return gray

img\_gray = rgb2gray(img)

plt.figure(1)

plt.imshow(img\_gray,cmap='gray')

plt.figure(2)

plt.show()

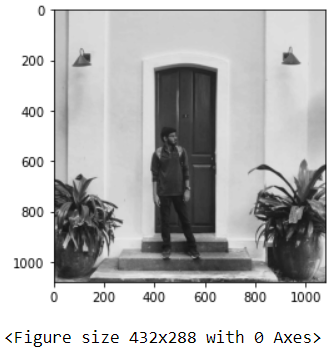
plt.hist(img\_gray)

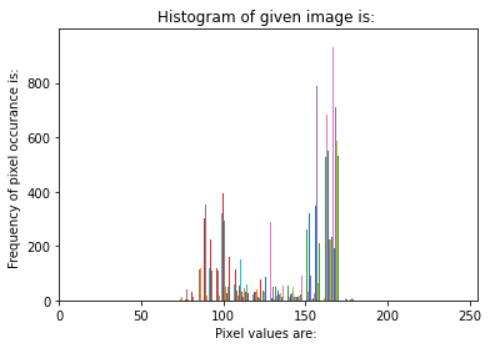
plt.title("Histogram of given image is: ")

plt.xlabel("Pixel values are: ")

plt.ylabel("Frequency of pixel occurance is: ")

plt.xlim([0, 255])





**INFERENCE­ ­–** This is for low contrast image as evident from the graph it is heavily concentrated in the middle portion of the graph.

**BRIGHT IMAGE**

**### Sample code for studying the histogram**

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.image as mpimg

img = mpimg.imread('C:/Users/WELCOME/Downloads/lmao.jpg')

def rgb2gray(rgb):

r, g, b = rgb[:,:,0], rgb[:,:,1], rgb[:,:,2]

gray = 0.2989 \* r + 0.5870 \* g + 0.1140 \* b

return gray

img\_gray = rgb2gray(img)

plt.figure(1)

plt.imshow(img\_gray,cmap='gray')

plt.figure(2)

plt.show()

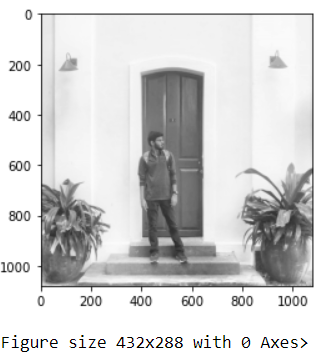
plt.hist(img\_gray)

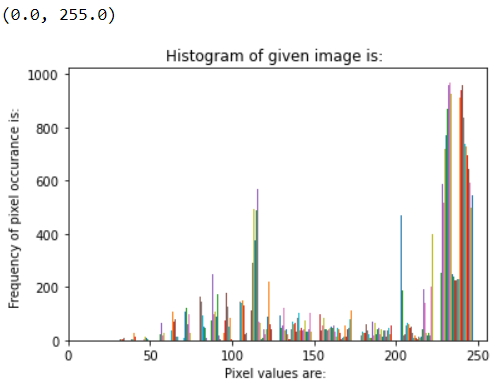
plt.title("Histogram of given image is: ")

plt.xlabel("Pixel values are: ")

plt.ylabel("Frequency of pixel occurance is: ")

plt.xlim([0, 255])





**INFERENCE­ ­–** For very high brightness the values get concentrated to the very right of of graph and pixel values increases as is evident from graph.

**LOW BRIGHTNESS IMAGE**

**### Sample code for studying the histogram**

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.image as mpimg

img = mpimg.imread('C:/Users/WELCOME/Downloads/lmfao.jpg')

def rgb2gray(rgb):

r, g, b = rgb[:,:,0], rgb[:,:,1], rgb[:,:,2]

gray = 0.2989 \* r + 0.5870 \* g + 0.1140 \* b

return gray

img\_gray = rgb2gray(img)

plt.figure(1)

plt.imshow(img\_gray,cmap='gray')

plt.figure(2)

plt.show()

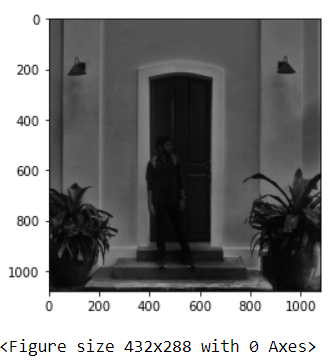
plt.hist(img\_gray)

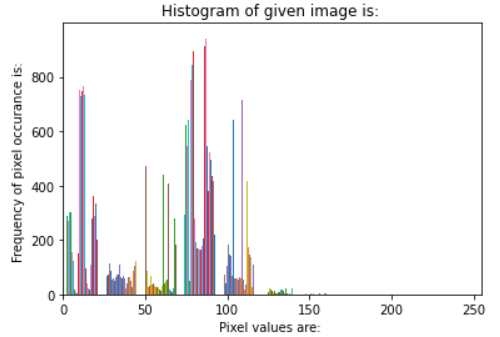
plt.title("Histogram of given image is: ")

plt.xlabel("Pixel values are: ")

plt.ylabel("Frequency of pixel occurance is: ")

plt.xlim([0, 255])





**INFERENCE­ ­–** As seen from the graph the pixel values decrease for really low brightness image.

**### Sample code for converting a gray scale image to binary image (thresholding)**

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.image as mpimg

img = mpimg.imread('C:/Users/WELCOME/Downloads/IMG\_20190902\_190953\_399.jpg')

def rgb2gray(rgb):

r, g, b = rgb[:,:,0], rgb[:,:,1], rgb[:,:,2]

gray = 0.2989 \* r + 0.5870 \* g + 0.1140 \* b

return gray

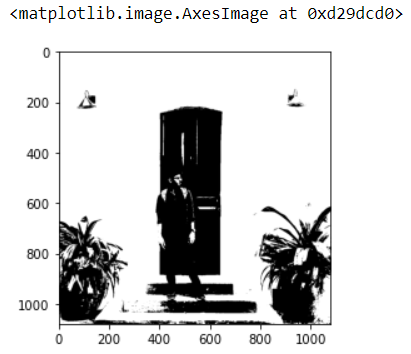
img\_gray = rgb2gray(img)

img\_temp = np.ones((1080, 1080), dtype=np.int16)

img\_gray = img\_temp\*(img\_gray > 93)

plt.imshow(img\_gray , cmap ='gray')

**OUTPUT -**



**INFERENCE­ ­-** When we compare to the original image we can see that there is a lot of white background in the image so when I keep the value at 93 it captures all the darker portion of the image while the lighter portion of the image gets whiter.

**### Sample code for converting a gray scale image to binary image (thresholding)**

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.image as mpimg

img = mpimg.imread('C:/Users/WELCOME/Downloads/IMG\_20190902\_190953\_399.jpg')

def rgb2gray(rgb):

r, g, b = rgb[:,:,0], rgb[:,:,1], rgb[:,:,2]

gray = 0.2989 \* r + 0.5870 \* g + 0.1140 \* b

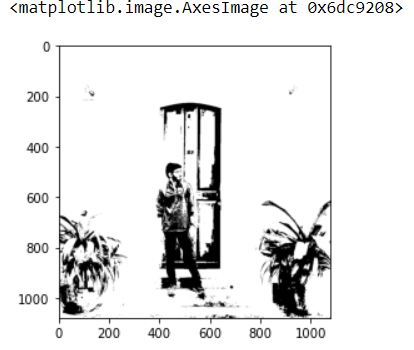
return gray

img\_gray = rgb2gray(img)

img\_temp = np.ones((1080, 1080), dtype=np.int16)

img\_gray = img\_temp\*(img\_gray > 50)

plt.imshow(img\_gray , cmap ='gray')



**INFERENCE­ ­-** When we compare to the original image we can see that there is a lot of white background in the image so when I keep the value at 50 it is not able to captures all the darker portion of the image and thus gives us more whiter portions.