

General John Kotelawala Defence University

Faculty of Engineering

Intake 38

Electronic and Telecommunication Engineering

Name = A.T.M Arachchige

Reg No. = D/ENG/21/0081/ET

In [ ]:

*# Question 1*

**%matplotlib** inline

**import** cv2 **as** cv

**import** matplotlib.pyplot **as** plt

**import** numpy **as** np

t1 **=** np**.**linspace(0,100,101)**.**astype('uint8') print (len(t1))

t2 **=** np**.**linspace(125,249,125)**.**astype('uint8') print (len(t2))

t3 **=** np**.**linspace(225,255,30)**.**astype('uint8') print(len(t3))

transform **=** np**.**concatenate((t1,t2),axis **=** 0)**.**astype('uint8')

transform **=** np**.**concatenate ((transform,t3),axis **=** 0)**.**astype('uint8') print (len(transform))

fig,ax **=** plt**.**subplots() ax**.**plot (transform)

ax**.**set\_xlabel (r'Input, $f(\mathbf{x})]$')

ax**.**set\_ylabel ('Output,$\mathrm{T}[f(\mathbf{x})]$') ax**.**set\_xlim (0,255)

ax**.**set\_ylim (0,255)

ax**.**set\_aspect ('equal') plt**.**show ()

im **=** cv**.**imread ('/examples/natasha grayscale.png',cv**.**IMREAD\_GRAYSCALE) transform\_im **=** cv**.**LUT (im,transform)

fig ,ax **=** plt**.**subplots (1,2, figsize**=** (10,10)) ax[0]**.**imshow(im, cmap**=**"gray")

ax[0]**.**set\_title("Without transformation") ax[1]**.**imshow(transform\_im, cmap**=**"gray") ax[1]**.**set\_title("With Transformation")

plt**.**show()

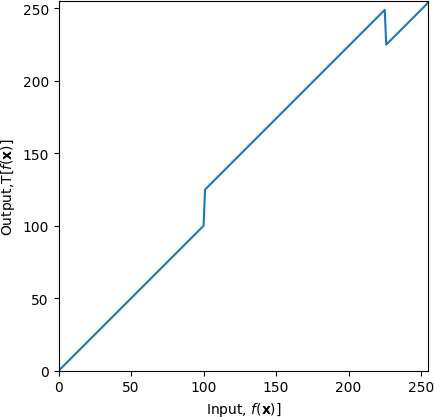
**Question 01**

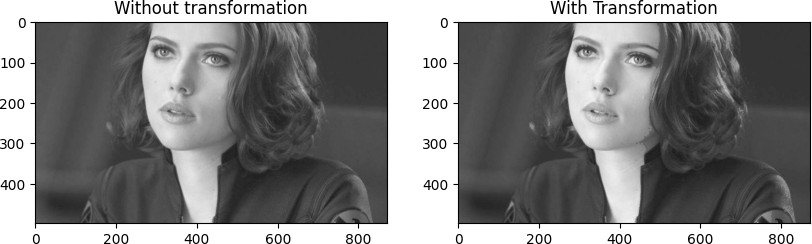
101

125

30

256





*# Question 2*

**import** cv2 **as** cv

**import** numpy **as** np

**import** matplotlib.pyplot **as** plt

im **=** cv**.**imread ('/examples/spiderman.png',cv**.**IMREAD\_COLOR) im1 **=** cv**.**cvtColor(im,cv**.**COLOR\_BGR2HSV)

h\_im,s\_im,v\_im **=** cv**.**split(im1)

**assert** im **is not None**

x**=** np**.**arange(0,256)**.**astype('uint8') a **=** 80

*#part (c)*

*## Adjusting the 'a' value to get a clear output The most accurate image i got when*

print('To get a Visually pleasing Output we can use a =',a) sigma **=** 70

y **=** np**.**minimum(((x)**+**(a**\***(np**.**exp(**-**(x**-**128)**\*\***2**/**(2**\***sigma**\*\***2))))**/**128), 255)**.**astype('uint8 trasnformation **=** cv**.**LUT(s\_im,y)

plt**.**title('INTENSITY TRANSFORMATION')

plt**.**plot(y)

**Question 02**

plt**.**show()

newHSV **=** cv**.**merge([h\_im,trasnformation,v\_im])

result **=** cv**.**cvtColor(newHSV, cv**.**COLOR\_HSV2BGR) addition\_of\_3\_planes **=** cv**.**add(newHSV, im)

trasnformation **=** cv**.**LUT(s\_im,y)

fig ,ax **=** plt**.**subplots (3,3, figsize**=** (10,10))

*#part (a)*

*## Splitting the image into 3 planes here we use the inbuilt split command inorder*

ax[0,0]**.**imshow(h\_im,cmap **=** 'gray') ax[0,0]**.**set\_title('HUE')

ax[0,1]**.**imshow(s\_im,cmap **=** 'gray') ax[0,1]**.**set\_title('SATURATION')

ax[0,2]**.**imshow(v\_im,cmap **=** 'gray') ax[0,2]**.**set\_title('VALUE')

ax[1,0]**.**imshow(im, cmap**=**"gray")

*# part (b)*

*##The given gaussian transformation is plotted below using numpy and then the trans*

ax[1,0]**.**set\_title("WITHOUT TRANSFORMATION")

ax[1,1]**.**imshow(trasnformation, cmap**=**"gray") ax[1,1]**.**set\_title("WITH TRANSFORMATION")

*#part (d)*

*## the 3 planes are added using the merge functions and the result is added with th*

ax[1,2]**.**imshow(addition\_of\_3\_planes, cmap**=**"gray")

ax[1,2]**.**set\_title("AFTER ADDING THE 3 PLANES (HSV NEW)")

*#part(e)*

*## We can see the difference when the 3 planes are merged and when they are addd to*

ax[2,0]**.**imshow(im, cmap**=**"gray")

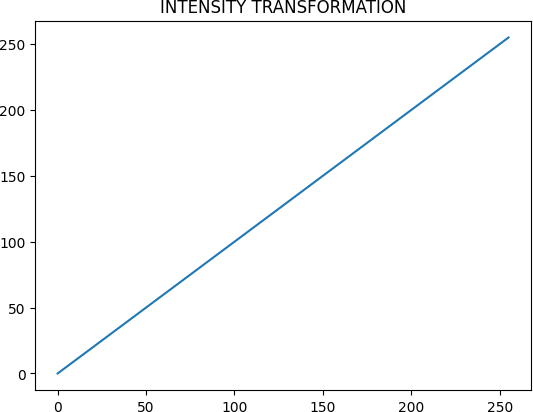
ax[2,0]**.**set\_title('INITIAL IMAGE') ax[2,1]**.**imshow(newHSV, cmap**=**"gray")

ax[2,1]**.**set\_title('VIBRANCE TRANSFORMATION')

ax[2,2]**.**imshow(addition\_of\_3\_planes, cmap**=**"gray") ax[2,2]**.**set\_title('VIBRANCE ENHANCED IMAGE')

plt**.**show()

To get a Visually pleasing Output we can use a = 80



*# Question 03*

**import** cv2 **as** cv

**import** matplotlib.pyplot **as** plt

**import** numpy **as** np

im **=** cv**.**imread ('/examples/highlights\_and\_shadows.png', cv**.**IMREAD\_COLOR)

**assert** im **is not None**

*# CONVERTING THE IMAGE TO LAB Planes*

im1 **=** cv**.**cvtColor(im, cv**.**COLOR\_BGR2LAB) L\_im,A\_im,B\_im **=** cv**.**split(im1)

gamma **=** 0.4

print('The Gamma value is =',gamma)

t **=** np**.**array([(i**/**255.)**\*\***gamma**\***255 **for** i **in** range (256)], np**.**uint8)

*## Part (a) applying the gamma corection*

g **=** t[L\_im]

plt**.**suptitle("GAMMA CURVE") plt**.**plot(t)

plt**.**show()

fig, ax **=** plt**.**subplots(1,2, figsize**=**(10,5)) ax[0]**.**imshow(L\_im, cmap**=**"gray")

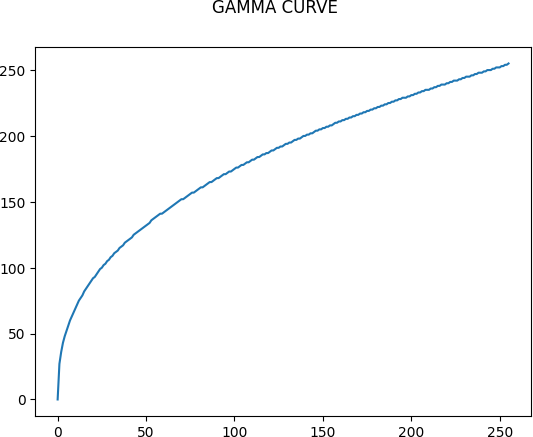
ax[0]**.**set\_title("ORIGINAL IMAGE") ax[1]**.**imshow(g,cmap**=**"gray")

ax[1]**.**set\_title("AFTER GAMMA CORRECTION")

plt**.**show()

*## Part (b) showing the histograms or of original and corrected images using the cv*

In [ ]:



**Question 03**

plt**.**figure(figsize **=** [10, 5])

plt**.**subplot(1, 2, 1)

plt**.**gca()**.**set\_title('ORIGINAL HISTOGRAM')

original\_histogram **=** cv**.**calcHist([im],[0],**None**,[256],[0,256]) *## hist = cv.calcHist*

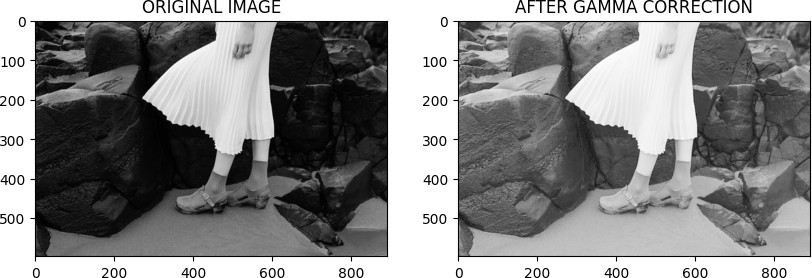
plt**.**plot(original\_histogram) plt**.**subplot(1, 2, 2)

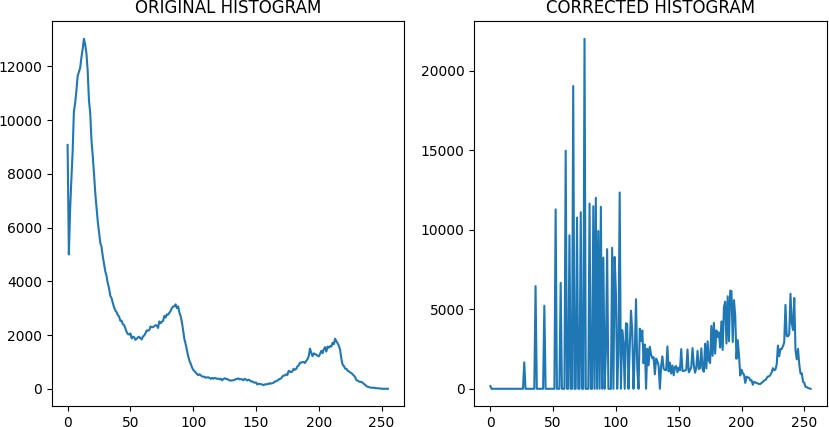
plt**.**gca()**.**set\_title('CORRECTED HISTOGRAM')

corrected\_histogram**=** cv**.**calcHist([g],[0],**None**,[256],[0,256]) plt**.**plot(corrected\_histogram)

plt**.**show()

The Gamma value is = 0.4





In [ ]:

*# Manually Calculating The Histogram equalization Function*

**import** cv2 **as** cv

**import** numpy **as** np

**import** matplotlib.pyplot **as** plt

**def** my\_function(img):

gray\_img **=** cv**.**cvtColor(img, cv**.**COLOR\_BGR2GRAY)

hist, bins **=** np**.**histogram(gray\_img**.**flatten(), 256, [0, 256]) cdf **=** hist**.**cumsum()

cdf\_normalized **=** cdf **/** cdf**.**max()

lut **=** np**.**interp(np**.**arange(256), bins[:**-**1], cdf\_normalized **\*** 255) equalized\_img **=** cv**.**LUT(gray\_img, lut)

hist\_equalized, bins\_equalized **=** np**.**histogram(equalized\_img**.**flatten(), 256, [0, plt**.**figure(figsize**=**(10,5))

plt**.**subplot(1,2,1)

plt**.**hist(gray\_img**.**flatten(), 256, [0, 256], color**=**'b') plt**.**title('Original Histogram')

plt**.**subplot(1,2,2)

plt**.**hist(equalized\_img**.**flatten(), 256, [0, 256], color**=**'g') plt**.**title('Equalized Histogram')

plt**.**show()

**return** equalized\_img

img **=** cv**.**imread('/examples/washed\_out\_arial.png') equalized\_img **=** my\_function(img)

plt**.**figure(figsize**=**(10,5)) plt**.**subplot(1,2,1)

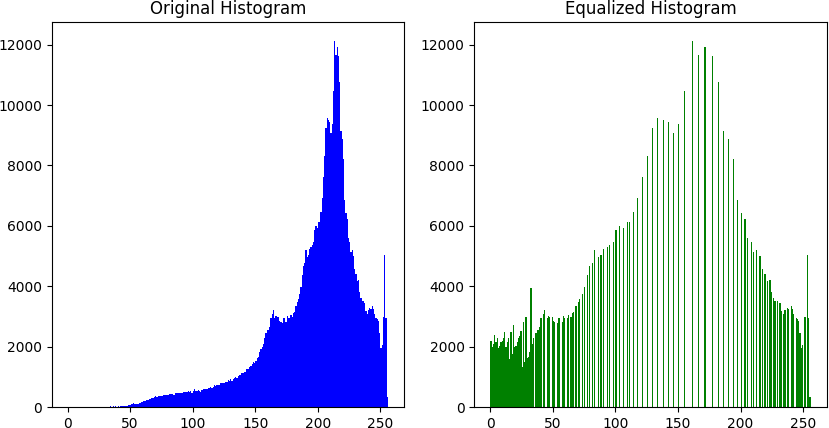
plt**.**imshow(cv**.**cvtColor(img, cv**.**COLOR\_BGR2RGB)) plt**.**title('Original Image')

plt**.**subplot(1,2,2)

plt**.**imshow(equalized\_img, cmap**=**'gray') plt**.**title('Equalized Image')

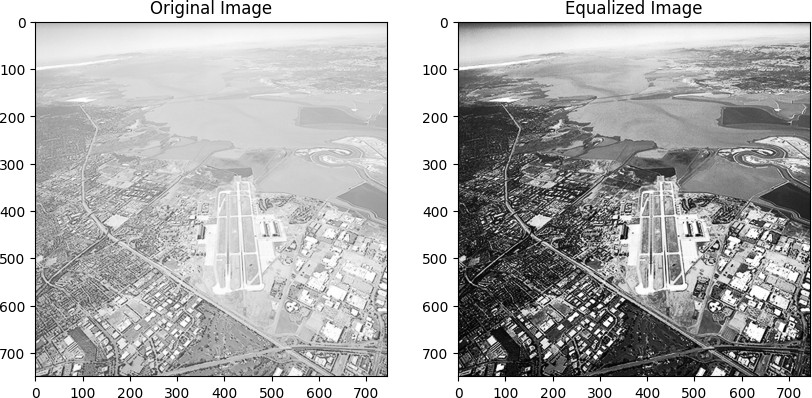
plt**.**show()

*## In the above code initially it calculates the normal histogram of the original i ##Then the CDF is calculated using cumsum() function and then the histogram is norm*



**Question 04 part I**

*## Then A lookuptable function is used to map the pixels to their equalized values. ## The values arent reduced as expected*



In [ ]:

**Question 04 part II**

*# Question 04 using the inbuilt equalizeHist() Function*

**import** cv2 **as** cv

**import** matplotlib.pyplot **as** plt

**import** numpy **as** np

im **=** cv**.**imread('/examples/washed\_out\_arial.png', cv**.**IMREAD\_GRAYSCALE) plt**.**figure(figsize **=** [10, 4])

plt**.**subplot(1, 2, 1)

plt**.**gca()**.**set\_title('INITIALLY') f **=** np**.**zeros(256)

f **=** [np**.**sum(im**==**i) **for** i **in** range (256)] plt**.**bar(range(256), f)

plt**.**subplot(1, 2, 2)

plt**.**gca()**.**set\_title('AFTER EQUALIZATION') im2**=** cv**.**equalizeHist(im)

plt**.**hist(im2)

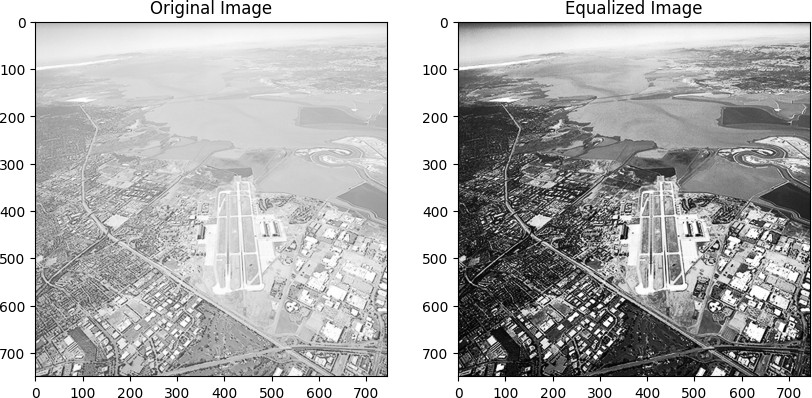
plt**.**show()

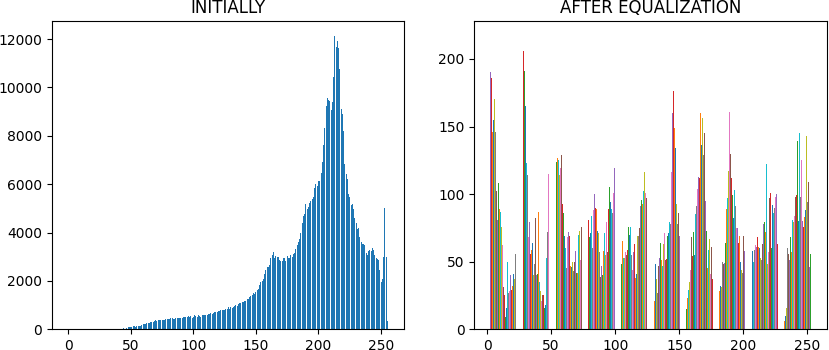
fig, ax**=** plt**.**subplots(1,2, figsize**=**(10,20)) ax[0]**.**imshow(im, cmap**=**"gray")

ax[0]**.**set\_title('Original Image') ax[1]**.**imshow(im2, cmap**=**"gray")

ax[1]**.**set\_title('After Equalization') plt**.**show()

*# There seem to be no difference in the outcome eventhough there is a huge reductio # we can use either methods equalize histogram*





*# Question 05*

**import** cv2 **as** cv

**import** matplotlib.pyplot **as** plt

**import** numpy **as** np

im **=** cv**.**imread ('/examples/jeniffer.png',cv**.**IMREAD\_COLOR) im1 **=** cv**.**cvtColor(im,cv**.**COLOR\_BGR2HSV)

*#part (a)*

*##Splitting into HUE SATURATION and VALUE Using the split functio in cv2*

H\_im,S\_im,V\_im **=** cv**.**split(im1)

*#part (b)*

*##the pixls between 20 to 245 are set to 0 and used the getStructuringElemnt functi ##Then the edges were smoothned by the morph function*

*##*

threshold **=** cv**.**inRange(S\_im, 20, 245)

el **=** cv**.**getStructuringElement(cv**.**MORPH\_ELLIPSE, (20,20)) mask **=** cv**.**morphologyEx(threshold, cv**.**MORPH\_CLOSE, el)

*# part (c)*

*## Using bitWise function in cv2 to obtain the foreground*

res1 **=** cv**.**bitwise\_and(im, im, mask**=**mask)

*#part (e)*

*## Histogram equalizing the fore ground*

plt**.**figure(figsize **=** [10, 2.5])

plt**.**subplot(1, 2, 1)

plt**.**gca()**.**set\_title('ORIGINAL HISTOGRAM')

fg\_h **=** cv**.**calcHist([res1],[0],**None**,[256],[0,256]) plt**.**plot(fg\_h)

plt**.**subplot(1, 2, 2)

plt**.**gca()**.**set\_title('CORRECTED HISTOGRAM') res2 **=** cv**.**cvtColor(res1, cv**.**COLOR\_BGR2GRAY)

equalized\_histogram\_1 **=** cv**.**equalizeHist(res2)

eh1 **=** cv**.**calcHist([equalized\_histogram\_1], [0], **None**, [256], [0,256]) plt**.**plot(eh1)

plt**.**show()

*# part (f)*

plt**.**gca()**.**set\_title ('CUMILATIVE SUM')

cumulative\_sum **=** np**.**cumsum(equalized\_histogram\_1) plt**.**plot(cumulative\_sum)

plt**.**show()

Question 05 part 2

*# part (f)*

mask1 **=** 255**-**mask

im\_bg1 **=** cv**.**bitwise\_and(im,im,mask**=** mask1)

im\_bg2 **=** cv**.**cvtColor(im\_bg1,cv**.**COLOR\_BGR2GRAY) im\_add **=** cv**.**add(im\_bg2,equalized\_histogram\_1)

fig,ax **=** plt**.**subplots(3,3,figsize **=** (10,10)) ax[0,0]**.**set\_title('HUE')

ax[0,0]**.**imshow(H\_im,cmap **=** 'gray') ax[0,1]**.**set\_title('SATURATE')

ax[0,1]**.**imshow(S\_im,cmap **=** 'gray') ax[0,2]**.**set\_title('VALUE')

ax[0,2]**.**imshow(V\_im,cmap **=** 'gray')

ax[1,0]**.**set\_title('Threshold Image (mask)') ax[1,0]**.**imshow(mask,cmap **=** 'gray')

ax[1,1]**.**set\_title('Using bitWise\_and Function') ax[1,1]**.**imshow(result,cmap **=** 'gray')

ax[1,2]**.**set\_title('Original Image') ax[1,2]**.**imshow(im,cmap **=** 'gray')

ax[2,0]**.**set\_title('Bckground Image') ax[2,0]**.**imshow(im\_bg1,cmap **=** 'gray') ax[2,1]**.**set\_title('Foreground Image')

ax[2,1]**.**imshow(equalized\_histogram\_1,cmap **=** 'gray') ax[2,2]**.**set\_title('Added Image')

ax[2,2]**.**imshow(im\_add,cmap **=** 'gray') plt**.**show()

