**Department of Computer Science and Engineering**

**National Institute of Technology, Hamirpur**

**Name: Aryan Puri**

**Roll Number: 22BEC032**

**Digital Image Processing Lab (CS-325)**

**Solution Laboratory Assignment -1**

**Topic: Spatial Image Enhancement functions and interpretation of results**

**Question 1: Design a program to read a bmp file and do the inversion and sub-sampling. Write the conclusion based on the observation of the output image.**

**Solution:**

1. **Inversion of Image**

**Python Program:**

import cv2

import numpy as np

import matplotlib.pyplot as plt

image=cv2.imread('./Assignment 1/MR.pgm',0)

height=image.shape[0]

width=image.shape[1]

image2=np.ones((height,width),dtype=np.uint8)

hi=[0]\*256

ho=[0]\*256

for i in range(height):

    for j in range(width):

        image2[i][j]=255-image[i][j]

        hi[image[i][j]]+=1

        ho[image2[i][j]]+=1

cv2.imshow('Orignal Image',image)

cv2.imshow('Inversion Image',image2)

for i in range(256):

    hi[i]=hi[i]/(height\*width)

    ho[i]=ho[i]/(height\*width)

plt.plot(hi)

plt.title('Input Image Histogram')

plt.show()

plt.plot(ho)

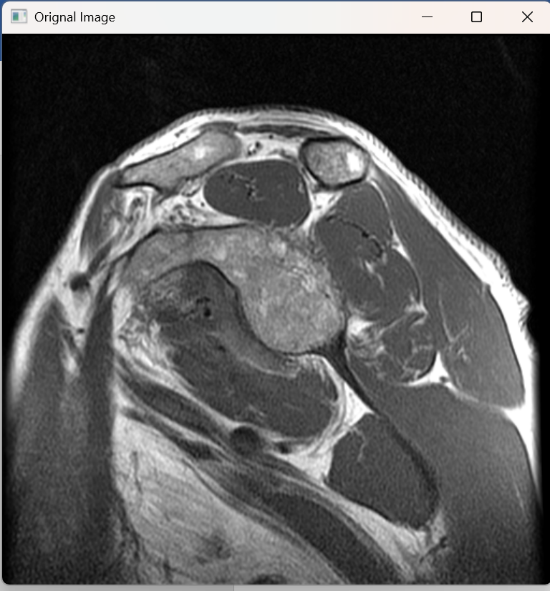
plt.title('Output Image Histogram')

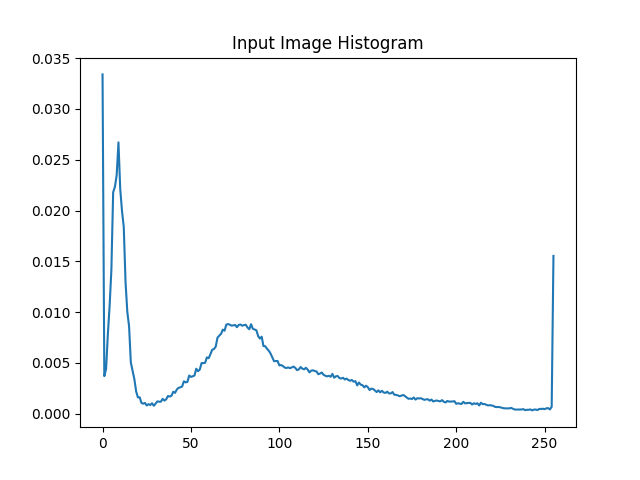
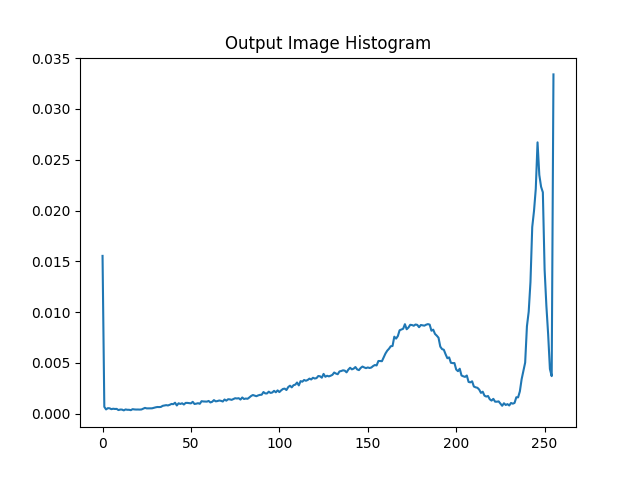
plt.show()

cv2.waitKey(0)

cv2.destroyAllWindows()

**Input Image: Output Image:**

**** ****

**Conclusion:**The Output Image is negative of the input image also the input and output histograms are inverse of each other.

1. **Sub Sampling**

**Python Program:**

import numpy as np

import cv2

image=cv2.imread('./Assignment 1/MR.pgm',0)

image=cv2.resize(image,(512,512))

height=image.shape[0]

width=image.shape[1]

image2=np.ones((int(height/2),int(width/2)),dtype=np.uint8)

for i in range(height):

    for j in range(width):

        if(i%2==0):

            if(j%2==0):

                image2[int(i/2)][int(j/2)]=image[i][j]

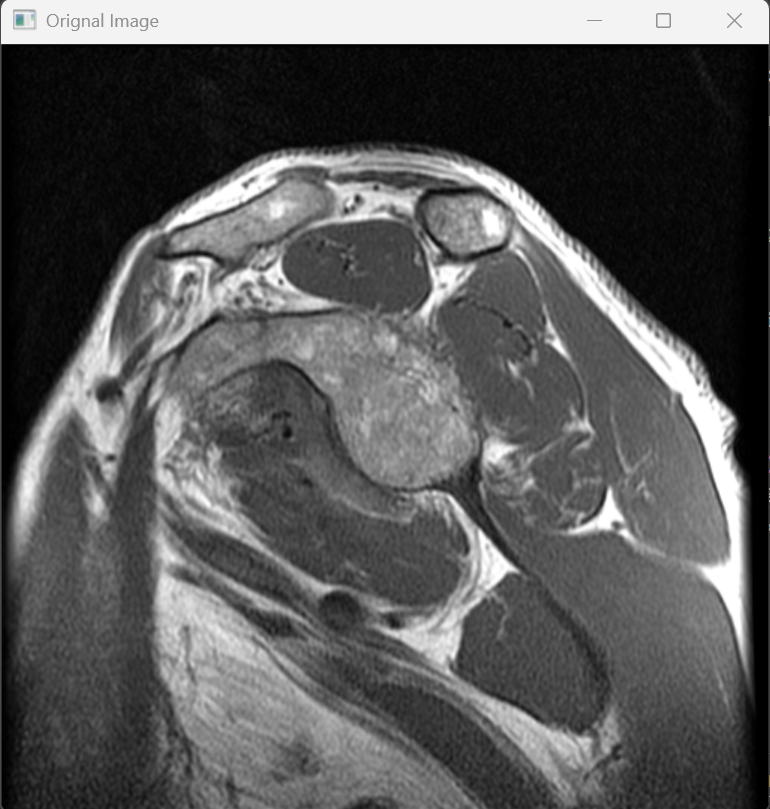
cv2.imshow('Orignal Image',image)

cv2.imshow('Sub Sampled Image',image2)

cv2.waitKey(0)

cv2.destroyAllWindows()

**Input Image: Output Image:**

**Conclusion:**The Output Image is sub sampled of the input image and having the dimensions half of the original image.

**Question 2: Design a program to read a bmp and pgm file and do the following enhancement operations in spatial domain**

**Solution:**

**A) Contrast stretching**

**Python program:**

import numpy as np

import math

import cv2

import matplotlib.pyplot as plt

image=cv2.imread('./Assignment 1/boats.bmp',0)

height=image.shape[0]

width=image.shape[1]

hi=[0]\*256

ho=[0]\*256

image2=np.ones((height,width),dtype=np.uint8)

for i in range(height):

    for j in range(width):

        if(image[i][j]<20):

            image2[i][j]=0

        elif(image[i][j]<200):

            image2[i][j]=round(255\*((image[i][j]-20)/180))

        else:

            image2[i][j]=255

        hi[image[i][j]]+=1

        ho[image2[i][j]]+=1

graph=[0]\*256

for i in range(256):

    if(i<20):

        graph[i]=0

    elif(i<200):

        graph[i]=round(255\*((i-20)/180))

    else:

        graph[i]=255

cv2.imshow('Orignal Image',image)

cv2.imshow('Contrast Enhancement Image',image2)

for i in range(256):

    hi[i]=hi[i]/(height\*width)

    ho[i]=ho[i]/(height\*width)

plt.plot(graph)

plt.title('\nS={0 if R<20;\n (R-20)/180 if 20<R<200;\n 255 if R>200}\n')

plt.xlabel('R')

plt.ylabel('S')

plt.show()

plt.plot(hi)

plt.title('Input Image Histogram')

plt.show()

plt.plot(ho)

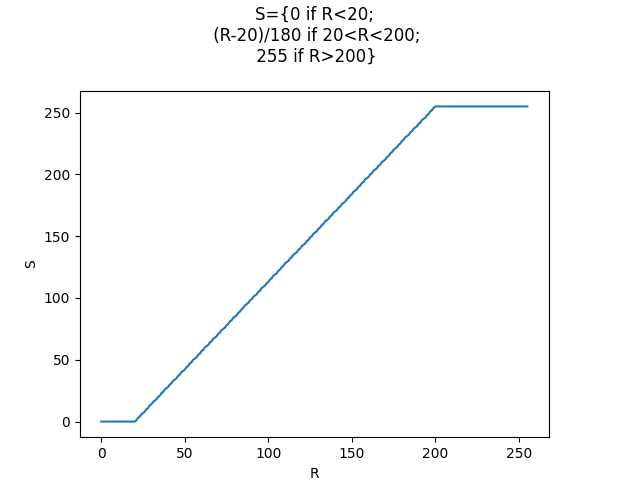
plt.title('Output Image Histogram')

plt.show()

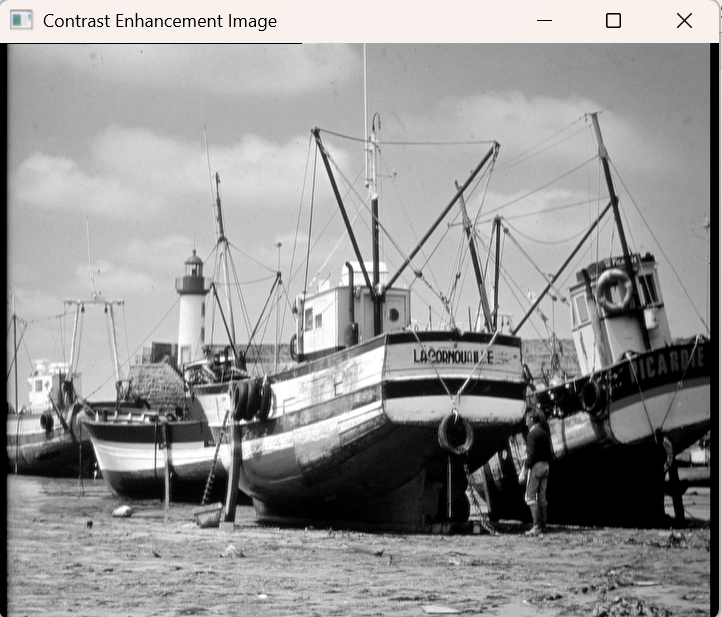
cv2.waitKey(0)

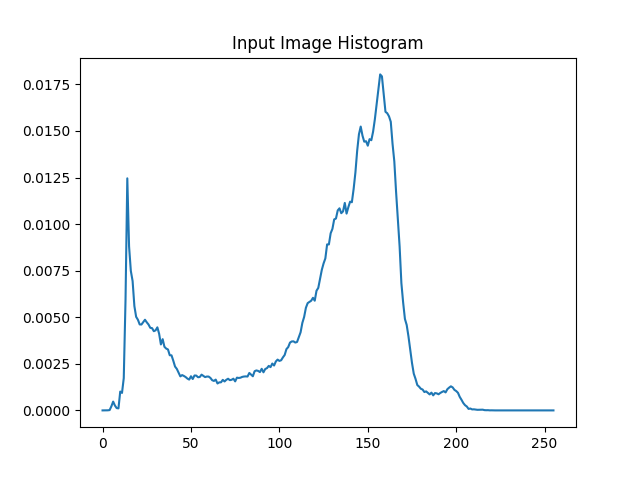
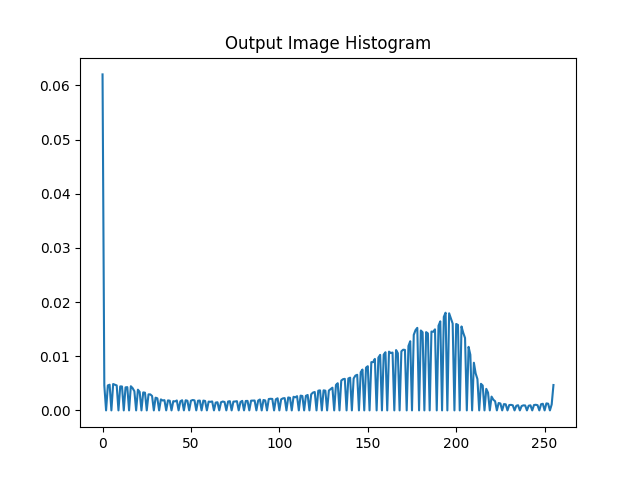
cv2.destroyAllWindows()

**Contrast Enhancement Function:**



**Input Image: Output Image:**

**Conclusion:**The Dynamic Range ( Max-Intensity/Min-Intensity ) of the output image is more than that of input image as seen from the histograms and the visual clarity of the image also increases.

**B) Log Transformation**

**Python Program:**

import cv2

import numpy as np

import math

import matplotlib.pyplot as plt

image=cv2.imread('./Assignment 1/ultrasound.pgm',0)

height=image.shape[0]

width=image.shape[1]

hi=[0]\*256

ho=[0]\*256

image2=np.ones((height,width),dtype=np.uint8)

for i in range(height):

    for j in range(width):

        if(image[i][j]>0):

            image2[i][j]=int(32 \* math.log(image[i][j],2))

        else:

            image2[i][j]=0

        hi[image[i][j]]+=1

        ho[image2[i][j]]+=1

graph=[0]\*256

for i in range(256):

    graph[i]=int(32 \* math.log(i+1,2))

cv2.imshow('Orignal Image',image)

cv2.imshow('Log Transformation Image',image2)

for i in range(256):

    hi[i]=hi[i]/(height\*width)

    ho[i]=ho[i]/(height\*width)

plt.plot(graph)

plt.title('\nS=clog(1+R)\n')

plt.xlabel('R')

plt.ylabel('S')

plt.show()

plt.plot(hi)

plt.title('Input Image Histogram')

plt.show()

plt.plot(ho)

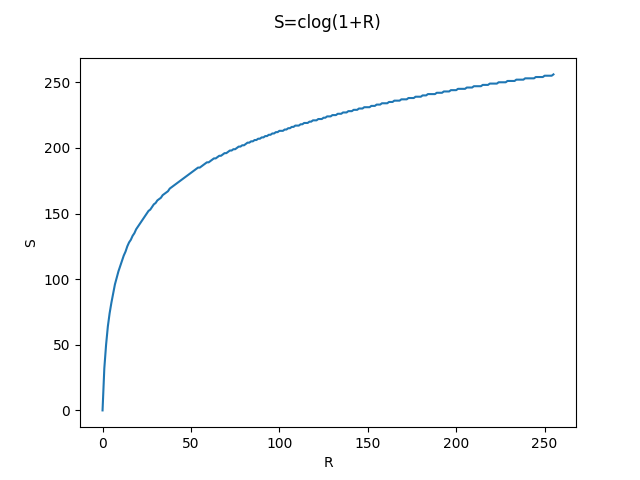
plt.title('Output Image Histogram')

plt.show()

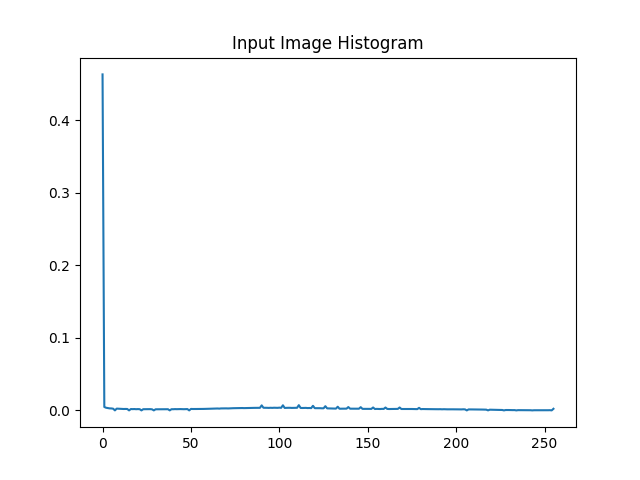
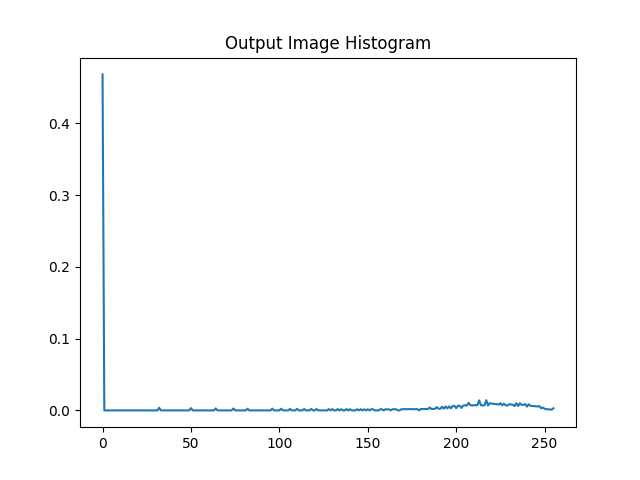
cv2.waitKey(0)

cv2.destroyAllWindows()

**Log Transformation Function**



**Input Image: Output Image:**  

**Conclusion:**Log Transformation enhances the details in darker regions of the image and compress high intensity values, due to which probability of white pixel increases in the image as seen in the histograms and that increases the Dynamic Range( Max-Intensity/Min-Intensity ) .Now the image appears more brighter.

1. **Power Law Transformation**

**Python Program:**

import numpy as np

import cv2

import math

import matplotlib.pyplot as plt

image=cv2.imread('./Assignment 1/boats.bmp',0)

height=image.shape[0]

width=image.shape[1]

image3=np.ones((height,width),dtype=np.uint8)

image4=np.ones((height,width),dtype=np.uint8)

image5=np.ones((height,width),dtype=np.uint8)

ho3=[0]\*256

ho4=[0]\*256

ho5=[0]\*256

hi=[0]\*256

for i in range(height):

    for j in range(width):

        hi[image[i][j]]+=1

        image3[i][j]=round(48\*math.pow(image[i][j],0.3))

        image4[i][j]=round(27\*math.pow(image[i][j],0.4))

        image5[i][j]=round(14\*math.pow(image[i][j],0.5))

        ho3[image3[i][j]]+=1

        ho4[image4[i][j]]+=1

        ho5[image5[i][j]]+=1

graph3=[0]\*256

graph4=[0]\*256

graph5=[0]\*256

for i in range(256):

    graph3[i]=round(48\*math.pow(i,0.3))

    graph4[i]=round(27\*math.pow(i,0.4))

    graph5[i]=round(14\*math.pow(i,0.5))

cv2.imshow('Orignal Image',image)

cv2.imshow('Power Law Transformation (b=0.3) Image',image3)

cv2.imshow('Power Law Transformation (b=0.4) Image',image4)

cv2.imshow('Power Law Transformation (b=0.5) Image',image5)

for i in range(256):

    hi[i]=hi[i]/(height\*width)

    ho3[i]=ho3[i]/(height\*width)

    ho4[i]=ho4[i]/(height\*width)

    ho5[i]=ho5[i]/(height\*width)

plt.plot(graph3,label='b=0.3')

plt.plot(graph4,label='b=0.4')

plt.plot(graph5,label='b=0.5')

plt.legend()

plt.title('\nS=c(R^b)\n')

plt.xlabel('R')

plt.ylabel('S')

plt.show()

plt.plot(hi)

plt.title('Histogram of Input')

plt.show()

plt.plot(ho3)

plt.title('Histogram of Output (b=0.3)')

plt.show()

plt.plot(ho4)

plt.title('Histogram of Output (b=0.4)')

plt.show()

plt.plot(ho5)

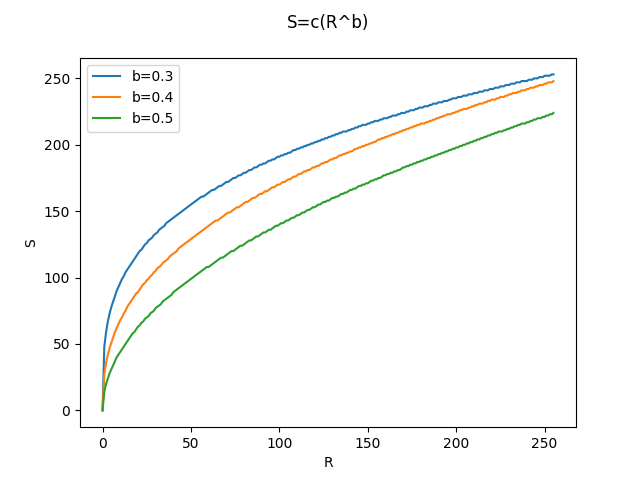
plt.title('Histogram of Output (b=0.5)')

plt.show()

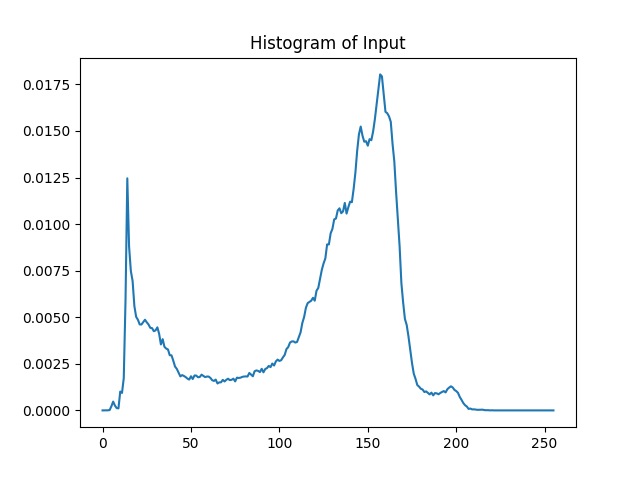
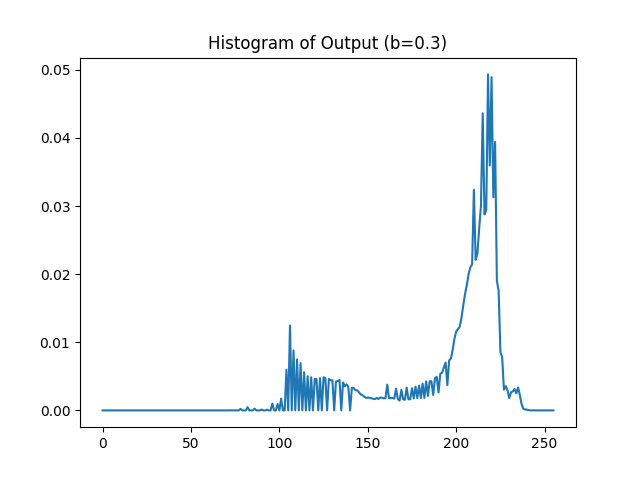
cv2.waitKey(0)

cv2.destroyAllWindows()

**Power law Transformation Function**

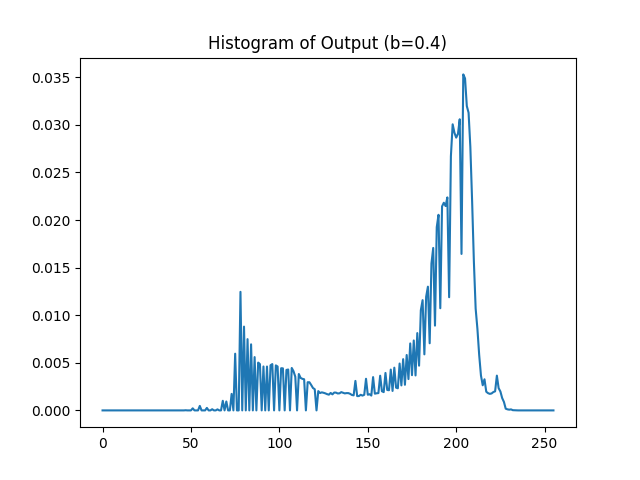
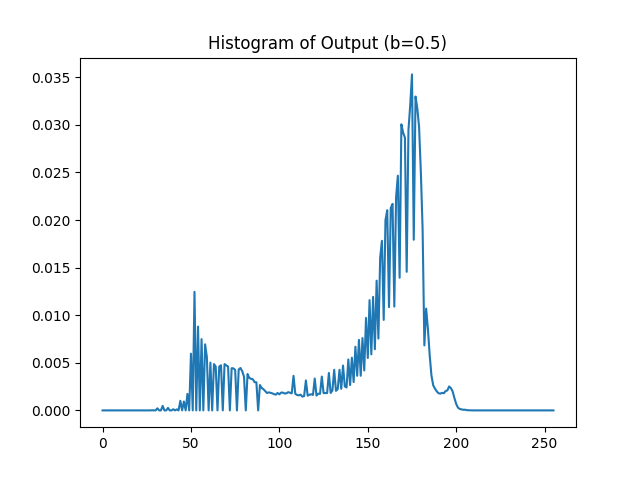


**Input Image: Output Image (b=0.3):**

 ****

**Output Image (b=0.4) Output Image (b=0.5)**

**Conclusion:**Power Law Transformation depending on the values of b perform intensity enhancement.

b<1: Enhances darker regions of the image, making details in shadows more visible.

b=1: Give output as original image.

b>1: Compresses brighter regions, reducing highlights.

So depending on the value of ‘b’ we can get the required transformation in the image as visual clarity of the image with b=0.5 is highest.