Name : Ashutosh Ardu

Reg no : 20BRS1262

DA-2

1. Take your picture, convert into a grey scale image and perform the following point

operations (choose appropriate constants):

Converting to GreyScale image first :-

Initial Image



After Conversion



Code for GrayScale Conversion :-

import numpy as np

import cv2 as ll

img = ll.imread("D:\VIT\Second-Yr-Winter\SignalProcessing\_TH\DA2\messi.jpg",0)

ll.imshow("output",img)

ll.imwrite('new.jpg',img)

ll.waitKey(0)

ll.destroyAllWindows()

1. Image Negation

Code :-

import cv2 as cv

img = cv.imread('new.jpg',0)

img1 = 255 - img

cv.imshow('input',img)

cv.imshow('image negated',img1)

cv.imwrite('new1.jpg',img1)

cv.destroyAllWindows()

Output



1. Log transformation

Code :-

import cv2 as cv

import numpy as np

img = cv.imread('log.png',0)

img1 = np.uint8(np.log1p(img))

th = 1

out = cv.threshold(img1,th,255,cv.THRESH\_BINARY)[1]

cv.imshow('input',img)

cv.imshow('out',out)

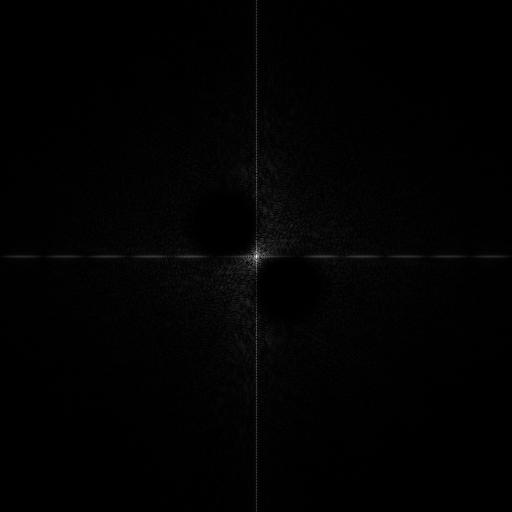
cv.imwrite('new2.jpg',out)

cv.destroyAllWindows()

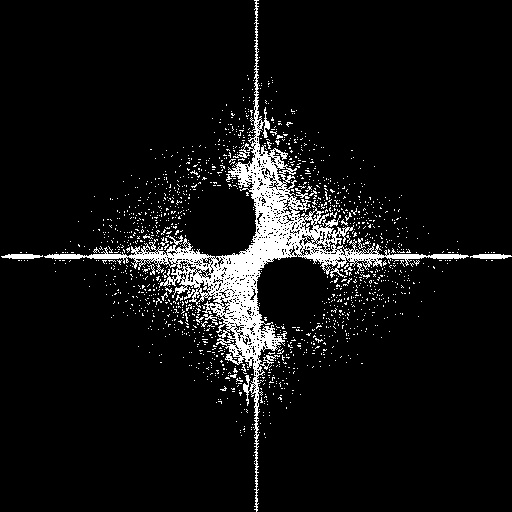
Output

C = 50

Initial Image



After Log Transformation



1. Gamma correction

Code :-

import cv2 as cv

import numpy as np

img1 = cv.imread('log.png',0)

gamma = 2

img2 = np.power(img1,gamma)

gamma = 3

img3 = np.power(img1,gamma)

gamma = 4

img4 = np.power(img1,gamma)

cv.imshow('input',img1)

cv.imshow('img2',img2)

cv.imshow('img3',img3)

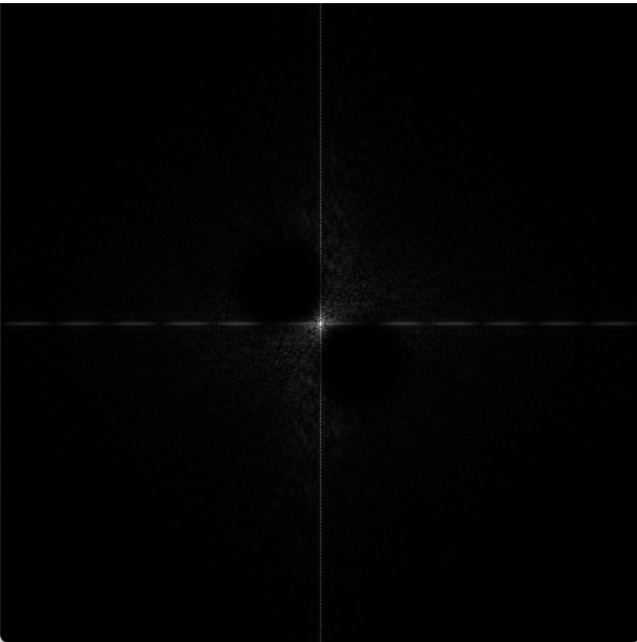
cv.imshow('img4',img4)

cv.waitKey(0)

cv.destroyAllWindows()

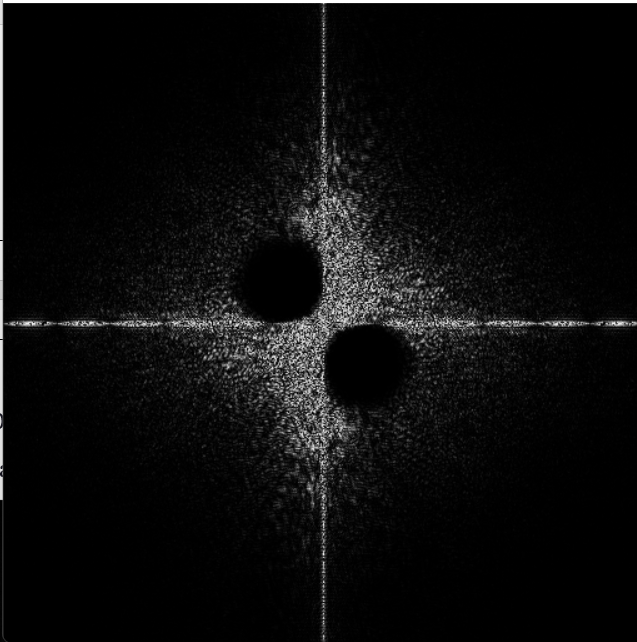
Output

Initial Image



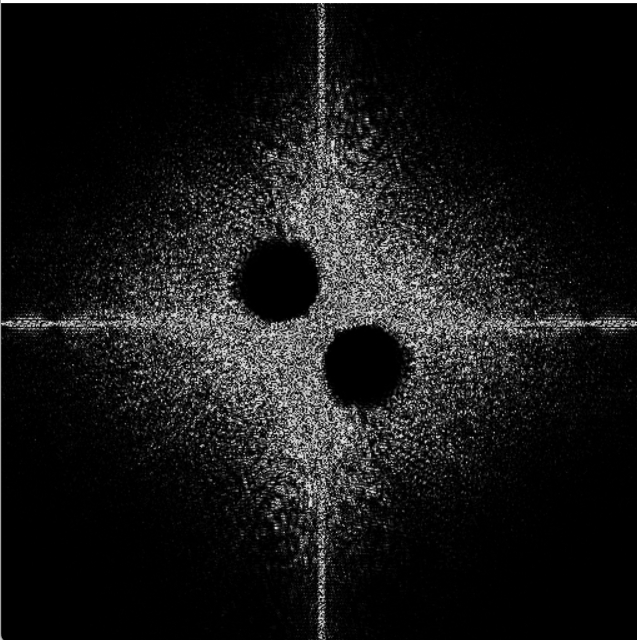
After applying Gamma Transformation

(Gamma = 2)



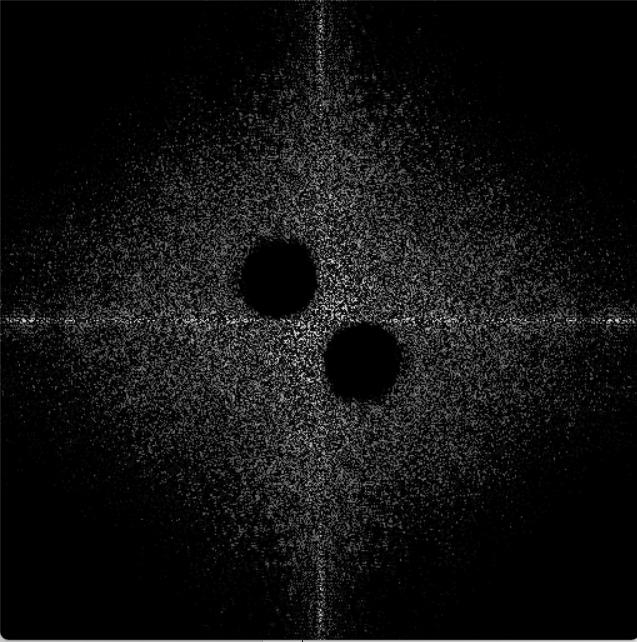
After applying Gamma Transformation

(Gamma = 3)



After applying Gamma Transformation

(Gamma = 4)



2. Consider an image of your choice and perform histogram equalization and histogram matching. Choose a gray scale image or a colour image

Histogram Equalization

Code :-

import cv2 as cv

import numpy as np

img = cv.imread('histogram.jpg',0)

equ = cv.equalizeHist(img)

res = np.hstack((img,equ))

cv.imwrite('new3.png',res)

Output

Initial Image Final Image



Histogram Matching

Code :-

import matplotlib.pyplot as plt

from skimage import data

import cv2 as cv

from skimage import exposure

from skimage.exposure import match\_histograms

# loading data

reference = cv.imread('reference.png')

image = cv.imread('source.png')

matched = match\_histograms(image, reference,

                           multichannel=True,)

fig, (ax1, ax2, ax3) = plt.subplots(nrows=1,

                                    ncols=3,

                                    figsize=(8, 3),

                                    sharex=True,

                                    sharey=True)

for aa in (ax1, ax2, ax3):

    aa.set\_axis\_off()

# displaying images

ax1.imshow(image)

ax1.set\_title('Source image')

ax2.imshow(reference)

ax2.set\_title('Reference image')

ax3.imshow(matched)

ax3.set\_title('Matched image')

plt.tight\_layout()

plt.show()

Output



3.Take any of the image from the above link and perform both mean, median and Gaussian filters on it. Compare the results with respect to PSNR (peak signal to noise ratio)

Using Mean Filters

Code (Done Using Matlab) :-

img = imread('salt.png');

filter = ones(3,3)/9;

img1 = imfilter(img,filter);

subplot(2,2,1),imshow(img),title('with salt&pepper noise');

subplot(2,2,2),imshow(img1),title('after removing salt&pepper noise');

imwrite(img1,'final.png');

[psnr,snr] = psnr(img1,img);

fprintf('PSNR value : %0.5f',psnr);

Output

Initial Image



Applying a Mean Filter





Using Median Filter

Code :-

import cv2 as cv

from skimage.filters.rank import median

from skimage.morphology import disk

img = cv.imread('salt.png',0)

out1 = cv.medianBlur(img,3)

cv.imshow('Original',img)

cv.imshow('CV2 Median',out1)

cv.imshow('Using Skimage',out2)

cv.imwrite('new6.png',out1)

cv.waitKey(0)

cv.destroyAllWindows()

Output

Initial Image



Applying a Median Filter





Using Gaussian Filter

Code :-

import cv2 as cv

from skimage import img\_as\_float,io

from skimage.filters import gaussian

img = cv.imread('salt.png',0)

out1 = cv.GaussianBlur(img,(3,3),0,borderType = cv.BORDER\_CONSTANT) # using cv2 package

out2 = gaussian(img,sigma = 1,mode = 'constant',cval = 0.0) # using skimage package

cv.imshow('Original',img)

cv.imshow('Using CV2 Gaussian',out1)

cv.imshow('Using Skimage',out2)

cv.imwrite('new5.png',out2)

cv.waitKey(0)

cv.destroyAllWindows()

Output

Initial Image



After applying Gaussian Image

(Using CV2 Package)



(Using Skimage Package)



