Code:

# %%

import cv2

import numpy as np

import matplotlib.pyplot as plt

from PIL import Image

from matplotlib import pyplot as plt

from matplotlib.gridspec import GridSpec

import numpy as np

filepath = 'jawad.jpg'

# load image

imageObj = cv2.imread(filepath)

img=cv2.cvtColor(imageObj,cv2.COLOR\_BGR2RGB)

img=cv2.cvtColor(imageObj,cv2.COLOR\_RGB2GRAY)

hist = cv2.calcHist([img], [0], None, [256], [0,256])

img\_eq = cv2.equalizeHist(img)

hist\_eq = cv2.calcHist([img\_eq], [0], None, [256], [0,256])

row = 3

col =3

#plotting image with the histogram

fig = plt.figure(figsize=(30,30))

gs = GridSpec(row, col)

fig.add\_subplot(gs[0,0])

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

fig.add\_subplot(gs[0,1])

plt.plot(hist)

# %%

#plotting EQUALIZED images with the Equalized Histogram

fig = plt.figure(figsize=(30,30))

gs = GridSpec(row, col)

fig.add\_subplot(gs[0,0])

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

fig.add\_subplot(gs[0,1])

plt.plot(hist\_eq)

# %%

#adaptive image eqalization with adaptive histogram

cl = cv2.createCLAHE(2.0, (8,8))

img\_ad\_eq = cl.apply(img)

hist\_ad\_eq = cv2.calcHist([img\_ad\_eq], [0], None, [256], [0,256])

fig = plt.figure(figsize=(30,30))

gs = GridSpec(row, col)

fig.add\_subplot(gs[0,0])

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

fig.add\_subplot(gs[0,1])

plt.plot(hist\_ad\_eq)

# %%

fig = plt.figure(figsize=(20,20))

gs = GridSpec(row, col)

fig.add\_subplot(gs[0,0])

plt.xlabel('Original Grayscale Image')

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

fig.add\_subplot(gs[0,1])

plt.xlabel('Equalized Image')

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

fig.add\_subplot(gs[0,2])

plt.xlabel('Adaptive Equalized Image')

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

fig.add\_subplot(gs[1,0])

plt.xlabel('Histogram')

plt.plot(hist)

fig.add\_subplot(gs[1,1])

plt.xlabel('Equalized Histogram')

plt.plot(hist\_eq)

fig.add\_subplot(gs[1,2])

plt.xlabel('Adaptive Equalized Histogram')

plt.plot(hist\_ad\_eq)

plt.savefig('1.jpg')

# %%

#gaussian noise

#salt-pepper noise

#random/uniform noise

#gaussian noise simulation

gu\_n = np.zeros((img.shape[0], img.shape[1]), dtype = np.uint8)

cv2.randn(gu\_n, 128, 20)

gu\_n = (gu\_n\*0.5).astype(np.uint8)

gu\_img= cv2.add(img, gu\_n)

fig = plt.figure(figsize=(40,40))

gs = GridSpec(row, col)

fig.add\_subplot(gs[0,0])

plt.xlabel('Noise Matrix')

plt.imshow(gu\_n, cmap='gray')

fig.add\_subplot(gs[0,1])

plt.xlabel('Image with gaussian Noise')

plt.imshow(gu\_img, cmap='gray')

# %%

#uniform noise simulation

rand\_n = np.zeros((img.shape[0], img.shape[1]), dtype = np.uint8)

cv2.randu(rand\_n, 0, 255)

print(rand\_n)

rand\_n = (rand\_n\*0.30).astype(np.uint8)

print(rand\_n)

rand\_n\_img= cv2.add(img, rand\_n)

print(rand\_n\_img)

fig = plt.figure(figsize=(40,40))

gs = GridSpec(row, col)

fig.add\_subplot(gs[0,0])

plt.xlabel('Random Noise Matrix')

plt.imshow(rand\_n, cmap='gray')

fig.add\_subplot(gs[0,1])

plt.xlabel('Image with Random Noise')

plt.imshow(rand\_n\_img, cmap='gray')

# %%

#salt pepper noise simulation

im\_n = rand\_n.copy()

ret, im\_n = cv2.threshold(rand\_n, 10, 100, cv2.THRESH\_BINARY)

im\_n = (im\_n\*0.8).astype(np.uint8)

im\_img = cv2.add(img, im\_n)

fig = plt.figure(figsize=(40,40))

gs = GridSpec(row, col)

fig.add\_subplot(gs[0,0])

plt.xlabel('Salt-Pepper Noise Matrix')

plt.imshow(im\_n, cmap='gray')

fig.add\_subplot(gs[0,1])

plt.xlabel('Image with Salt-Pepper Noise')

plt.imshow(im\_img, cmap='gray')

# %%

fig = plt.figure(figsize=(20,20))

gs = GridSpec(row, col)

fig.add\_subplot(gs[0,0])

plt.xlabel('Gaussian Noise')

plt.imshow(gu\_img, cmap='gray')

fig.add\_subplot(gs[0,1])

plt.xlabel('Uniform Noise')

plt.imshow(rand\_n\_img, cmap='gray')

fig.add\_subplot(gs[0,2])

plt.xlabel('Salt-Pepper Noise')

plt.imshow(im\_img, cmap='gray')

fig.add\_subplot(gs[1,0])

plt.xlabel('Gaussian Noise Matrix')

plt.imshow(gu\_n, cmap='gray')

fig.add\_subplot(gs[1,1])

plt.xlabel('Uniform/Random Noise Matrix')

plt.imshow(rand\_n, cmap='gray')

fig.add\_subplot(gs[1,2])

plt.xlabel('Salt-Pepper Noise Matrix')

plt.imshow(im\_n, cmap='gray')

plt.savefig('2.jpg')

# %%

#IMAGE SMOOTHING WITH GAUSSIAN BLUR AND MEDIAN BLUR

gu\_sm = cv2.GaussianBlur(gu\_img, (3,3), 5)

im\_sm = cv2.medianBlur(im\_img,5)

fig = plt.figure(figsize=(20,20))

gs = GridSpec(row, col)

fig.add\_subplot(gs[0,0])

plt.xlabel('Gaussian noise added image')

plt.imshow(gu\_img, cmap='gray')

fig.add\_subplot(gs[0,1])

plt.xlabel('Salt-pepper noise added image')

plt.imshow(im\_img, cmap='gray')

fig.add\_subplot(gs[1,0])

plt.xlabel('Gaussian blurred image')

plt.imshow(gu\_sm, cmap='gray')

fig.add\_subplot(gs[1,1])

plt.xlabel('Median blurred image')

plt.imshow(im\_sm, cmap='gray')

plt.savefig('3.jpg')

Output:





