بسمه تعالی

**تمرین سری 2 – مرضیه امیری**

1. الگوریتم histogram equalization را روی تصویر زیر پیاده سازی کرده (بدون استفاده از دستور histeq) و بر روی تصویر زیر اعمال کنید.

import cv2

import matplotlib.pyplot as plt

import numpy as np

img = cv2.imread('1.jpg') # reading image

gray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY) #convert to gray

plt.hist(gray.ravel(), bins=256) #showing histogram of original image

plt.show()

w , h = gray.shape

n = w \* h #n is the number of pixels

My\_dict = dict()

for i in range (0,256): #create a dictionary to keep every pixel intensity as a key

    My\_dict[i] = 0

for item in gray.ravel(): #keep the number of each pixel intensity as dictionary values

    My\_dict[item]+=1

for i in My\_dict: #calculate cumulative normalized histogram

    if i > 0:

        My\_dict[i] += My\_dict[i-1]

for i in My\_dict: #calculate cumulative normalized histogram and transformed intensity

        My\_dict[i] = round(( My\_dict[i] / n ) \* 255)

equalized=gray.copy()

for i in range(w): #apply the equalization to original image

    for j in range(h):

        k=equalized[i,j]

        equalized[i,j]=My\_dict[k]

plt.hist(equalized.ravel(), bins=256) #showing equalized histogram

plt.show()

cv2.imshow('original',gray) #showing main image

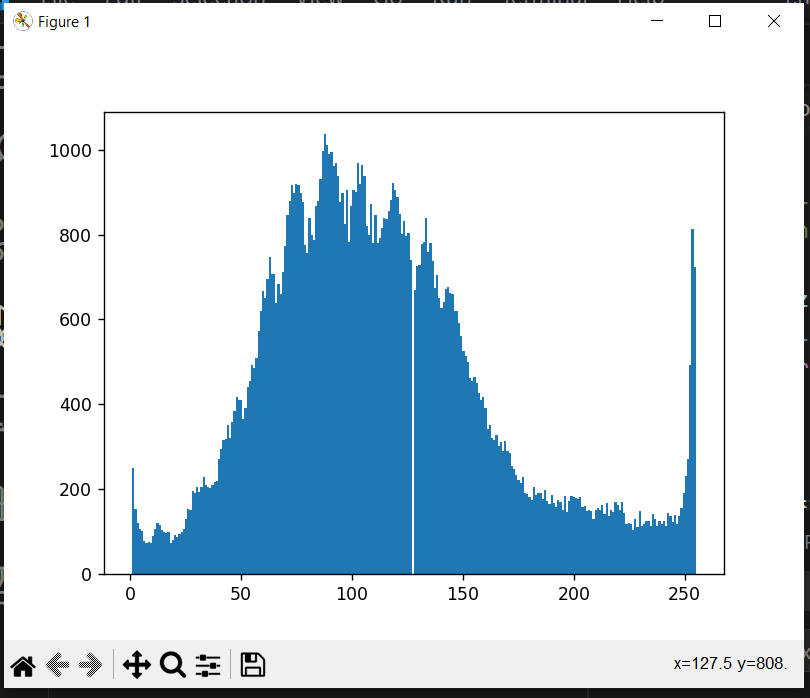
cv2.imshow('equalized',equalized) #showing equalized image

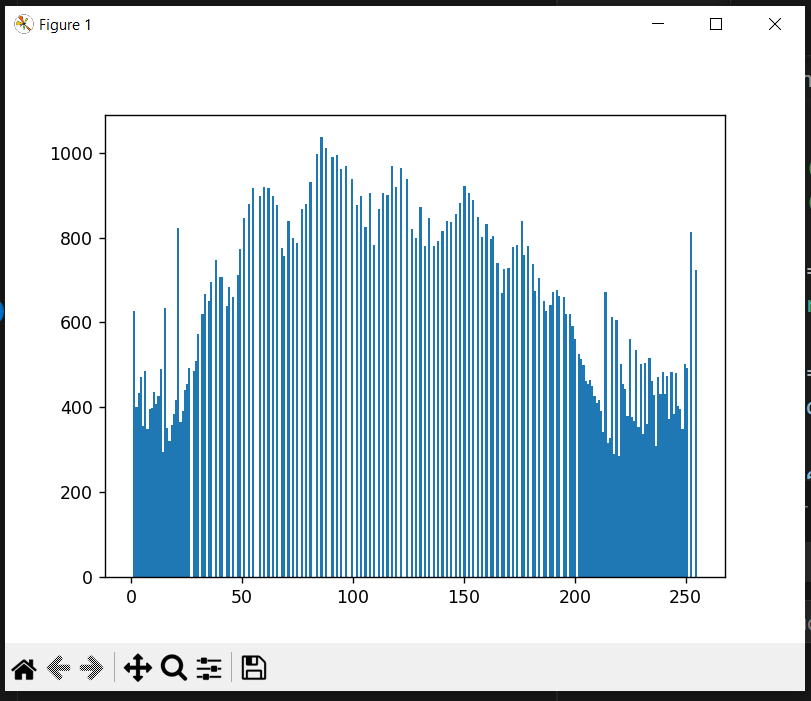
cv2.waitKey(0)

cv2.destroyAllWindows()

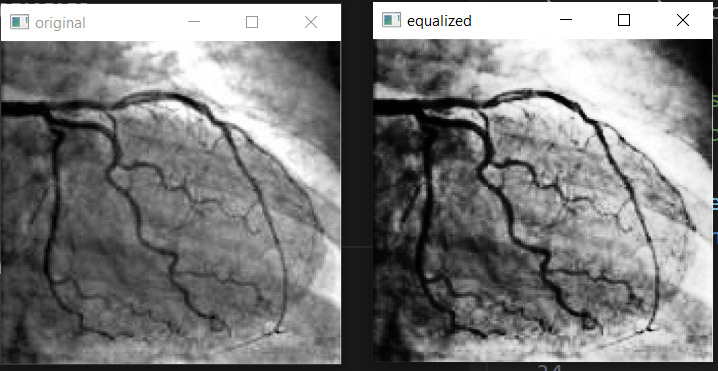
**خروجی :**

هیستوگرام تصویر اولیه:





هیستوگرام equalize شده:

تصویر اصلی و

ایکوالایز شده

ب) نتیجه الگوریتم خود را با نتایج دستورات equalizeHist و createCLAHE مقایسه کنید. (تصویر را به gray تبدیل کنید)

import cv2

import matplotlib.pyplot as plt

import numpy as np

img = cv2.imread('1.jpg')

gray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY)

img2 = cv2.equalizeHist(gray) #equalizeHist

clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8)) #createCLAHE

img3 = clahe.apply(gray) #createCLAHE

cv2.imshow('original',gray)

cv2.imshow('equalizedHist',img2)

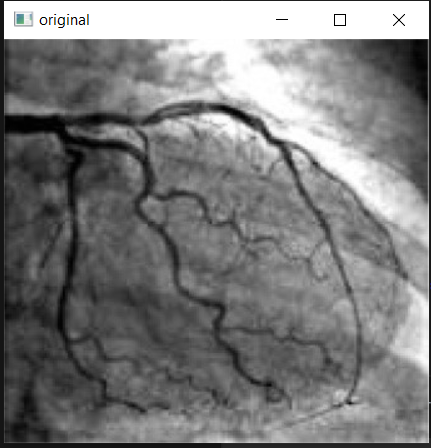
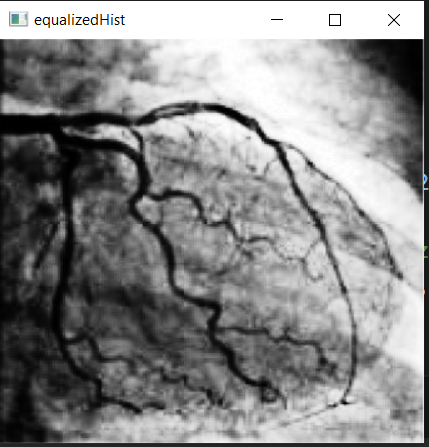
cv2.imshow('createCLAHE',img3)

cv2.waitKey(0)

cv2.destroyAllWindows()

**خروجی:**

**Original: equalizedHist:**

****

**createCLAHE**

1. تصویر زیر را خوانده و با منحنی زیر contrast stretching بر روی آن انجام دهید.

مقادیر پارامترها را بصورتی در نظر بگیرید که تصویر خروجی بهترین کیفیت را از نظر شما داشته باشد. (مقادیر پارامترها را ارسال کنید)

import cv2

a=50

b=150

alpha=0.1

beta=0.5

gama=0.8

Ya=20

Yb=160

img = cv2.imread('2.jpg')

h,w,z=img.shape

stretched=img.copy()

for i in range(h):

    for j in range(w):

        for c in range(z):

            if stretched[i,j,c]<a:

                stretched[i,j,c]\*=alpha

            elif stretched[i,j,c]<b:

                stretched[i,j,c]=beta\*(stretched[i,j,c]-a)+Ya

            else:

                stretched[i,j,c]=gama\*(stretched[i,j,c]-b)+Yb

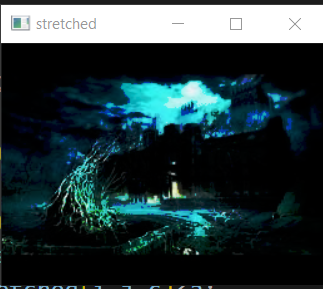
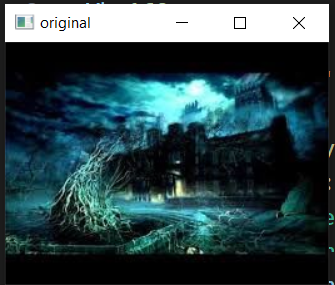
cv2.imshow('original', img)

cv2.imshow('stretched', stretched)

cv2.waitKey(0)

cv2.destroyAllWindows()

خروجی:



1. میخواهیم تصویر ورودی زیر را با تیز کردن لبه ها بهبود ببخشیم 1) با اعمال تبدیل مستقیم لاپلاس جمع با کردن با تصویر اصلی 2) با استفاده از unsharp فیلترینگ اینکار را انجام دهید. (بدون استفاده از دستورات پایتون و با استفاده از این دستور نتایج را نمایش داده و مقایسه کنید)

import cv2

import numpy as np

#laplacian function

def laplacian\_addition\_function(img):

    h,w=img.shape

    filtered\_img=img.copy()

    #applying laplacian filter to original image

    # -1 -1 -1

    # -1  9 -1

    # -1 -1 -1

    for i in range(1,h-1):

        for j in range(1,w-1):

            filtered\_img[i,j]=-img[i-1,j-1]-img[i-1,j]-img[i-1,j+1]-img[i,j-1]-img[i,j+1]-img[i+1,j-1]-img[i+1,j]-img[i+1,j+1]+9\*img[i,j]

    return filtered\_img

#bluring function

def bluring\_function(img):

    h,w=img.shape

    blured\_img=img.copy()

    blured\_img-=blured\_img

    #average filter

    # 1/16 \* 1 2 1

    #        2 4 2

    #        1 2 1

    for i in range(1,h-1):

        for j in range(1,w-1):

            blured\_img[i,j]=1/16\*(img[i-1,j-1]+2\*img[i-1,j]+img[i-1,j+1]+2\*img[i,j-1]+2\*img[i,j+1]+img[i+1,j-1]+2\*img[i+1,j]+img[i+1,j+1]+4\*img[i,j])

    return blured\_img

def unsharp\_masking(img):

    h,w=img.shape

    k=1

    masked\_img=img.copy()

    g\_mask=img-bluring\_function(img)

    masked\_img+=(k\*g\_mask)

    return masked\_img

img = cv2.imread('3.jpg', 0)

filtered\_img = laplacian\_addition\_function(img)

unsharpe\_masked = unsharp\_masking(img)

g\_mask=img-bluring\_function(img)

blured=bluring\_function(img)

cv2.imshow('Original', img)

cv2.imshow('laplacian',filtered\_img)

cv2.imshow('unsharpemasking',unsharpe\_masked)

cv2.waitKey(0)

cv2.destroyAllWindows()

خروجی :

تصویر اصلی

laplacian :

unsharped:

استفاده از دستورات پایتون برای unsharp کردن:

from PIL import Image, ImageFilter

import cv2

# Open an image file

with Image.open("3.jpg") as im:

    # Apply unsharp mask filter

    im\_sharp = im.filter(ImageFilter.UnsharpMask(radius=2, percent=150, threshold=3))

    # Save the filtered image

    im\_sharp.save("image\_sharpened.jpg", "JPEG")

imgsh = cv2.imread("image\_sharpened.jpg")

img = cv2.imread("3.jpg")

cv2.imshow('original',img)

cv2.imshow('mage\_sharpened',imgsh)

cv2.waitKey(0)

cv2.destroyAllWindows()

خروجی :

تصویر اصلی :



تصویر unsharp شده :

