Grayscale Image Histogram Equalization

Source code:

# -\*- coding: utf-8 -\*-

"""

Created on Wed Feb 6 10:38:21 2019

@author: eagle

"""

import cv2

import numpy as np

from matplotlib import pyplot as plt

img = cv2.imread('orig\_gray\_img.jpg',0) # use this to change the grayscale image to manipulate

# Get CDF of original image

hist,bins = np.histogram(img.flatten(),256,[0,256])

cdf = hist.cumsum()

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

# Performing Histogram Equalization Equation with the use of Numpy masked array

cdf\_m = np.ma.masked\_equal(cdf,0)

cdf\_m = (cdf\_m - cdf\_m.min())\*255/(cdf\_m.max()-cdf\_m.min())

cdf = np.ma.filled(cdf\_m,0).astype('uint8')

# Now we have the look-up table that gives us the information on what is the

# output pixel value for every input pixel value

# we apply the transform here

img2 = cdf[img]

# Plot of Before and After Histogram of the Image

plt.hist(img.flatten(),256,[0,256], color = 'r')

plt.hist(img2.flatten(),256,[0,256], color = 'b')

plt.legend(('original','equalized'), loc = 'upper left')

plt.title('Original and Equalized Histogram Plots')

plt.show()

# Resize the Before and After Images to fit on laptop screen

height, width = img.shape[:2]

height2, width2 = img2.shape[:2]

if height > 800 or width > 800:

img = cv2.resize(img, (int(width/1.5), int(height/1.5)), interpolation = cv2.INTER\_CUBIC)

if height2 > 800 or width2 > 800:

img2 = cv2.resize(img2, (int(width/1.5), int(height/1.5)), interpolation = cv2.INTER\_CUBIC)

# Show Before and After Image

res = np.hstack((img, img2))

cv2.imshow('Before and After', res)

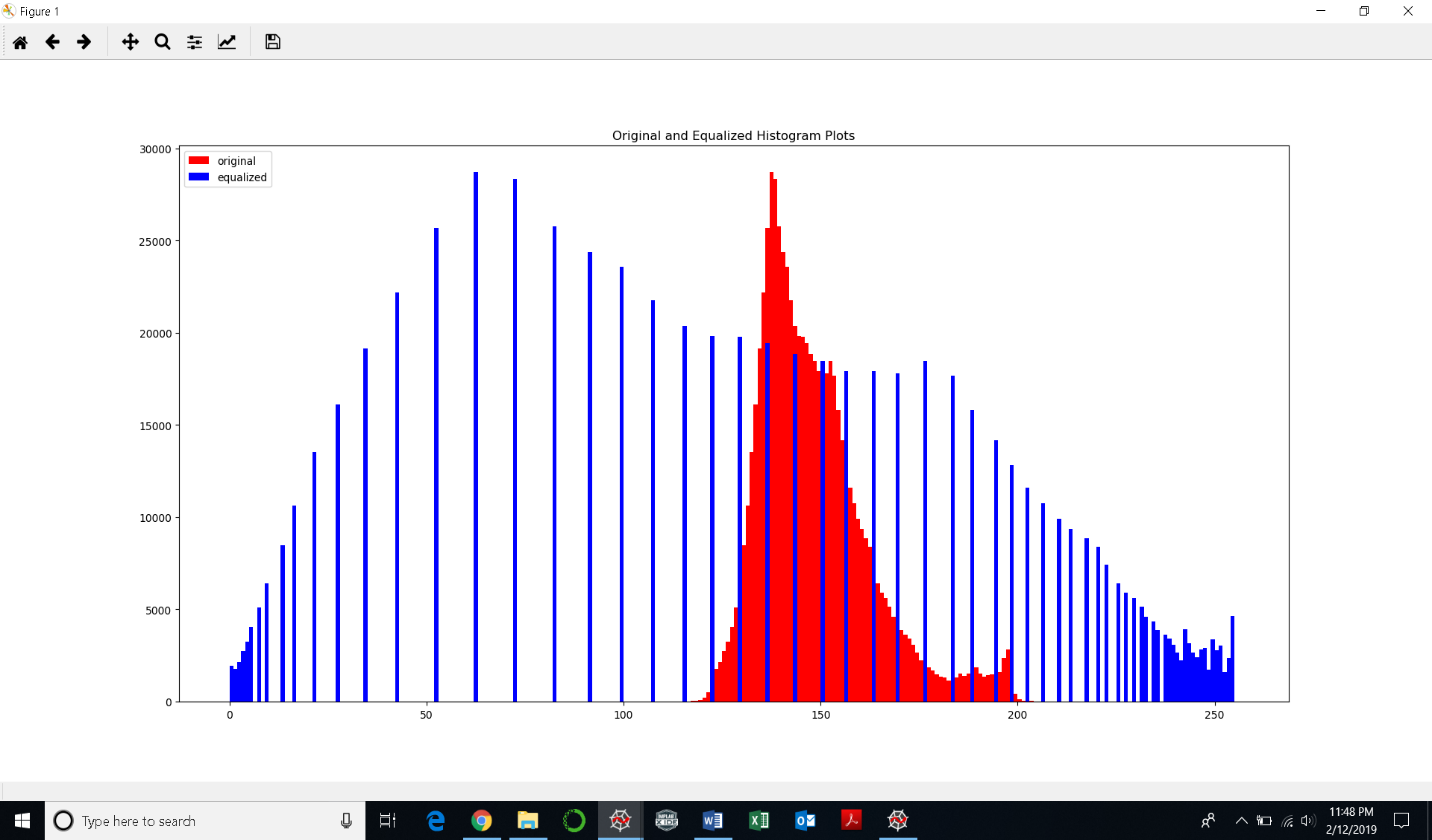
cv2.waitKey(0)

cv2.destroyAllWindows()

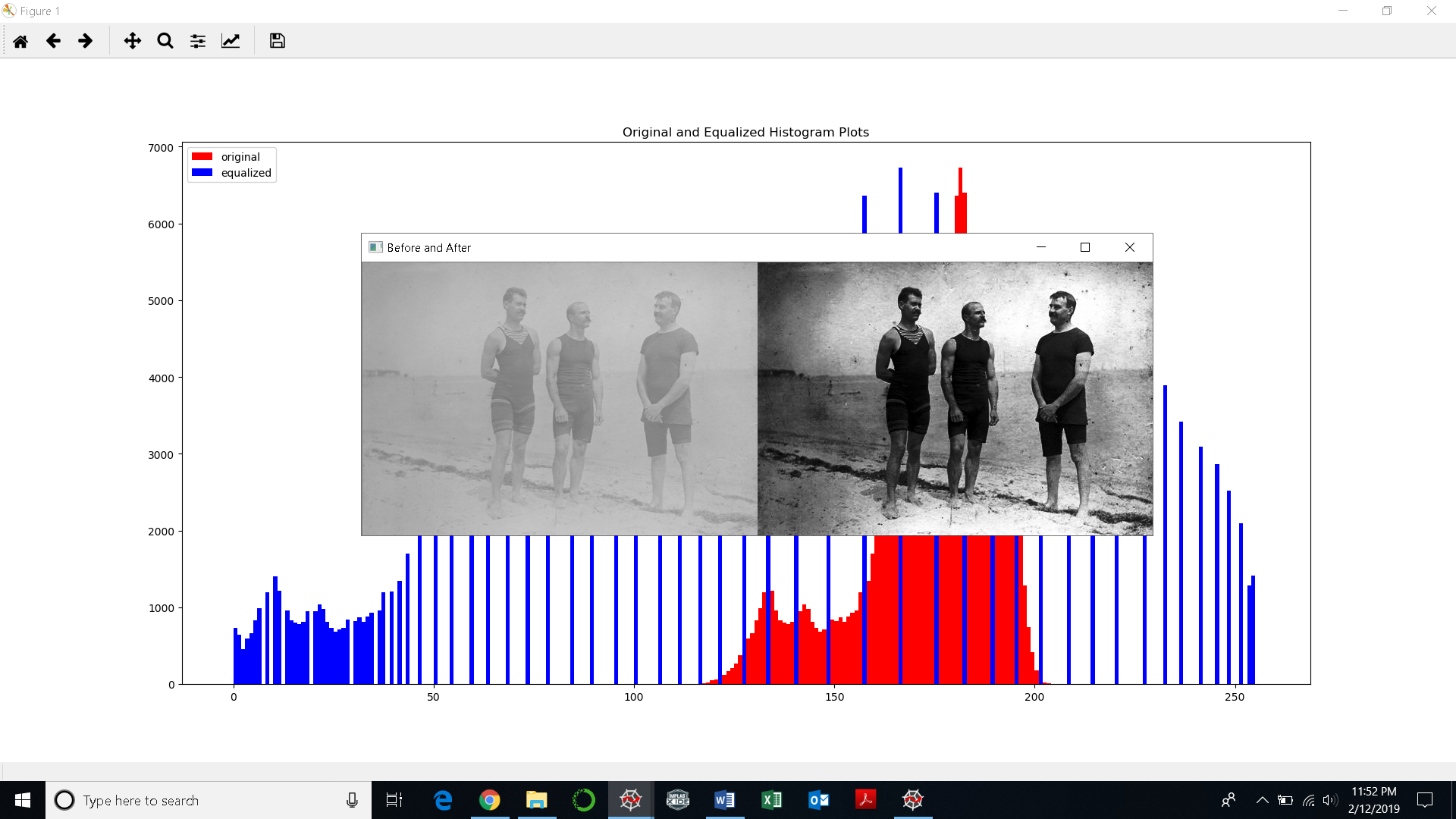
Outputs:

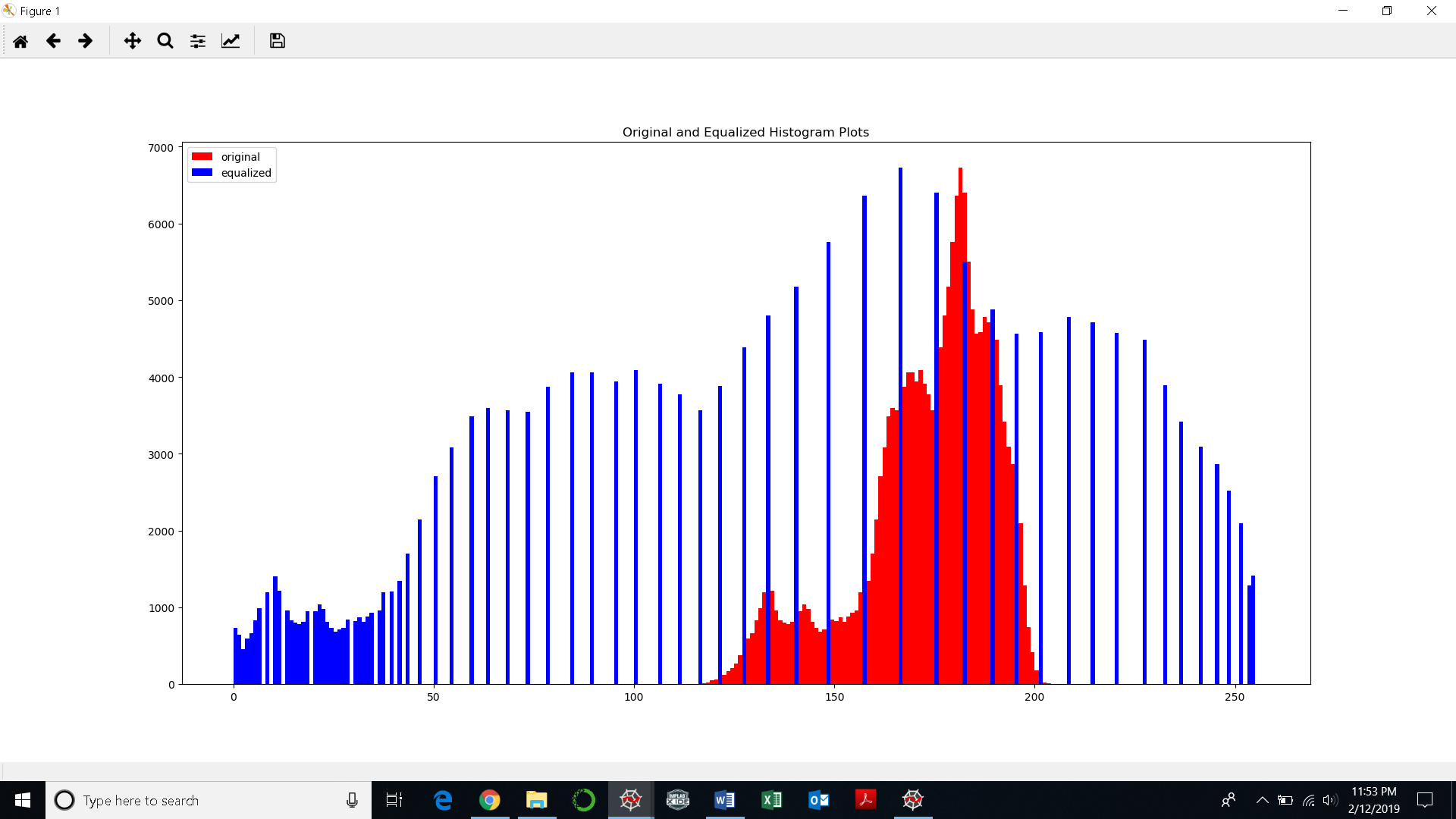
Example 1:





Example 2:





Color Image Histogram Equalization

Source code:

# -\*- coding: utf-8 -\*-

"""

Created on Wed Feb 6 11:35:04 2019

@author: eagle

source: opencv.org

"""

import cv2

import numpy as np

import argparse

# run this code using the command prompt

parser = argparse.ArgumentParser(description='Code for Histogram Equalization tutorial.')

parser.add\_argument('--input', help='Path to input image.', default='orig\_color\_img.jpg') # loading the image

# when running this code in command, put --input orig\_gray\_img\_2.jpg

# after python HistoEqOpenCV.py to run it on that image

args = parser.parse\_args()

src = cv2.imread(args.input)

# if no image is loaded

if src is None:

print('Could not open or find the image:', args.input)

exit(0)

img\_yuv = cv2.cvtColor(src, cv2.COLOR\_BGR2YUV)

# equalize the histogram of the Y channel

img\_yuv[:,:,0] = cv2.equalizeHist(img\_yuv[:,:,0])

# convert the YUV image back to RGB format

img\_output = cv2.cvtColor(img\_yuv, cv2.COLOR\_YUV2BGR)

height, width = src.shape[:2]

height2, width2 = img\_output.shape[:2]

if height > 800 or width > 800:

src = cv2.resize(src, (int(width/1.5), int(height/1.5)), interpolation = cv2.INTER\_CUBIC)

if height2 > 800 or width2 > 800:

img\_output = cv2.resize(img\_output, (int(width/1.5), int(height/1.5)), interpolation = cv2.INTER\_CUBIC)

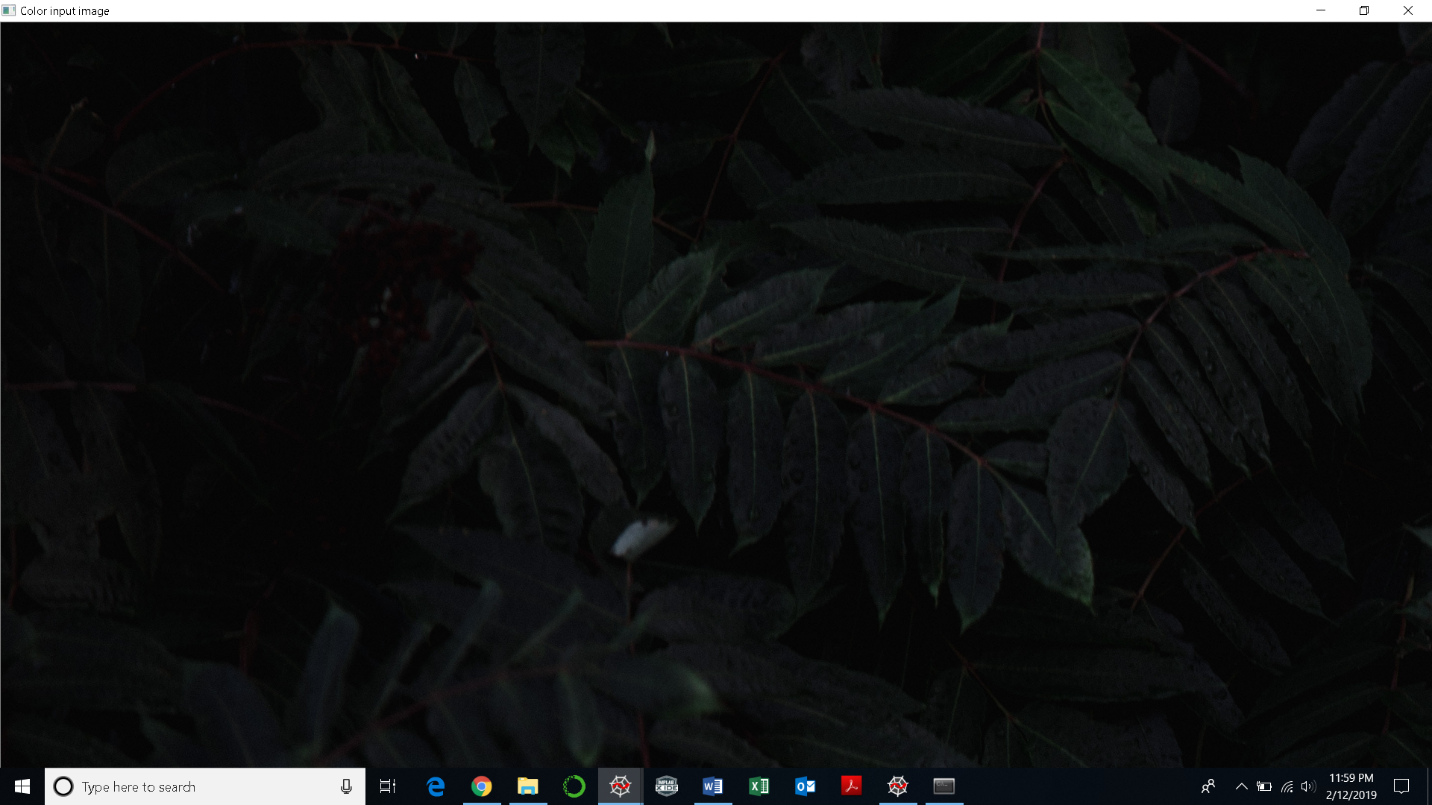
cv2.imshow('Color input image', src)

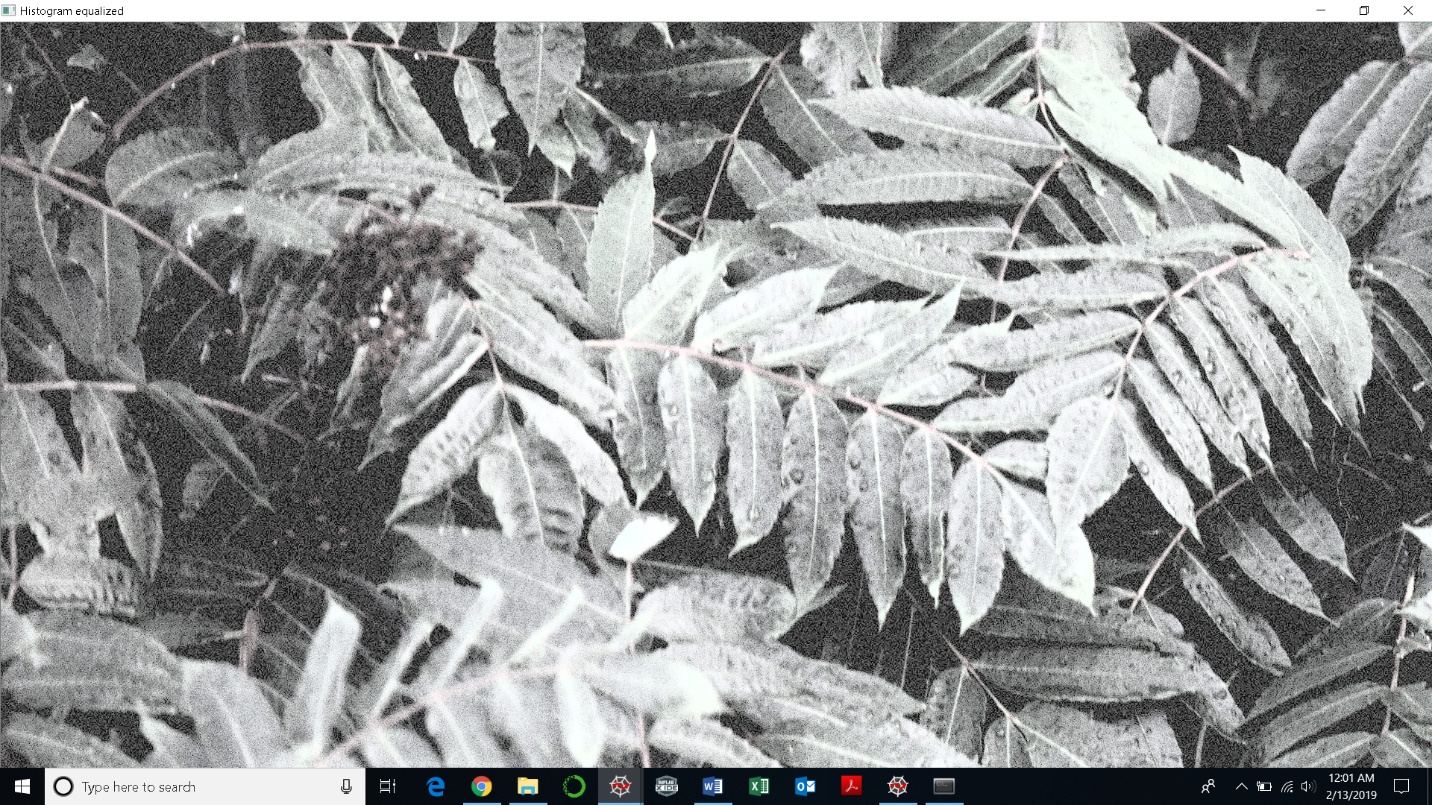
cv2.imshow('Histogram equalized', img\_output)

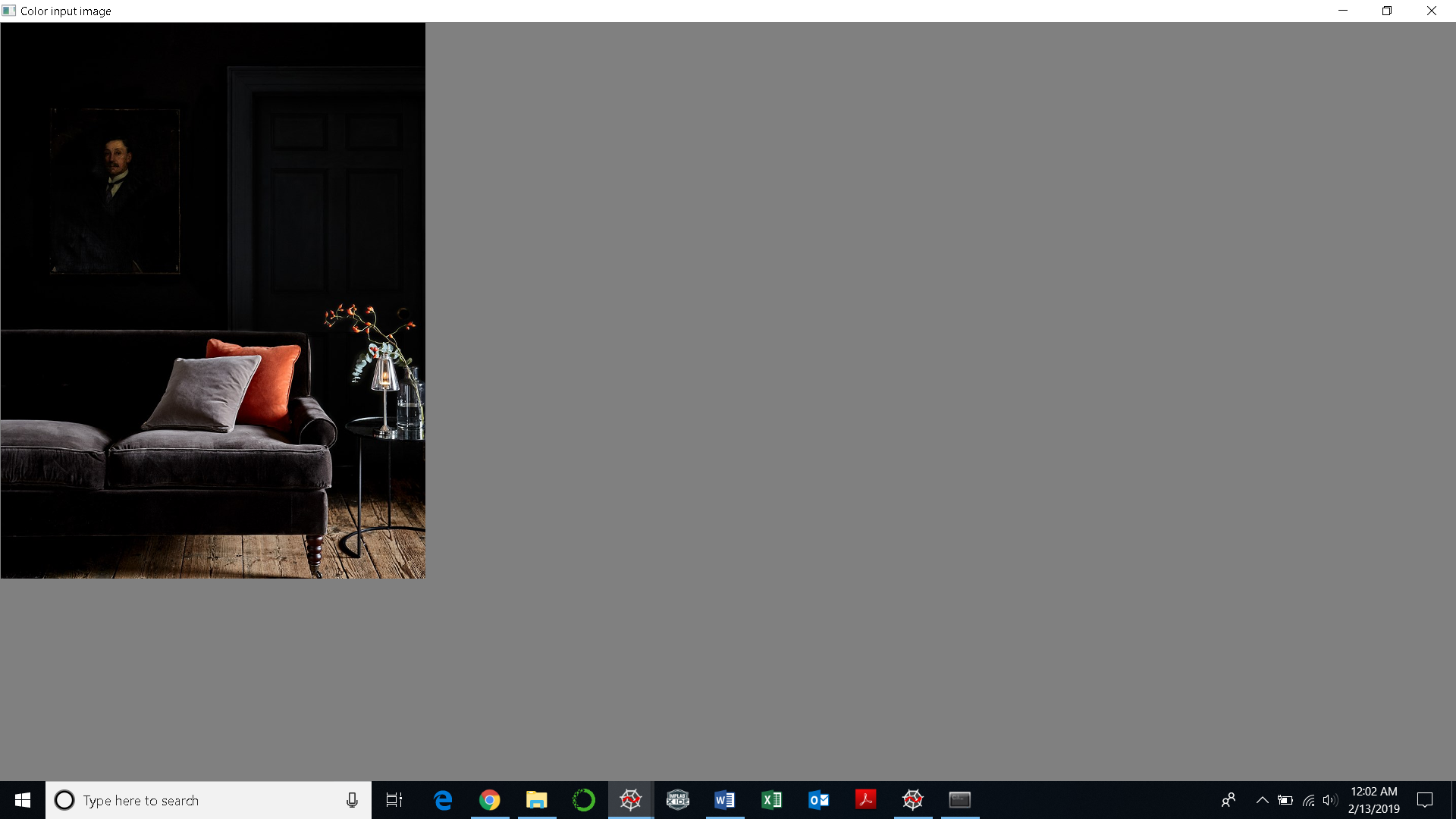
cv2.waitKey()

Outputs:

Example 1:





Example 2:

