BLG 453E - Computer Vision HW1 Report

Q1) Firstly, I calculated mean value of the image.

```
## CALCULATING MEAN VALUE ##
sum = [0 for x in range(ch)]

for i in range(row):
        for j in range(col):
                for k in range(ch):
                        sum[k] += arr[i][j][k]

mean = [0 for x in range(ch)]
for i in range(ch):
        mean[i] = sum[i] / pixel

print('mean =', mean)
```

Here, all values of array are summed. Green channel values are summed within themselves, red channel within themselves and blue channel within themselves. The sum values are divided by pixel numbers. Pixel numbers is equal to row times column. By this way, mean of blue channel, mean of green channel and mean of red channel are calculated seperately. For the given image mean values are given below.

```
mean = [36.605749063670409, 50.857013108614233, 67.261132958801497]
```

Secondly, I calculated standard deviation of the image.

```
## CALCULATING STANDART DEVIATION VALUE ##
distance = [[[0 for x in range(ch)] for y in range(col)] for z in range(row)]
distsqr = [[[0 for x in range(ch)] for y in range(col)] for z in range(row)]

for i in range(row):
        for j in range(col):
                for k in range(ch):
                        distance[i][j][k] = arr[i][j][k] - mean[k]
                        distsqr[i][j][k] = distance[i][j][k] * * 2

sumsd = [0 for x in range(ch)]

for i in range(row):
        for j in range(col):
                for k in range(ch):
                        sumsd[k] += distsqr[i][j][k]

sd = [0 for x in range(ch)]
for i in range(ch):
        sd[i] = sumsd[i] / pixel
        sd[i] = sd[i] * * 0.5
```

Here, the formula which is given below is functionalized. Standard deviation is calculated separately for each channel. The result is:

$$S = \sqrt{\frac{\Sigma(X - \overline{X})^2}{N}}$$

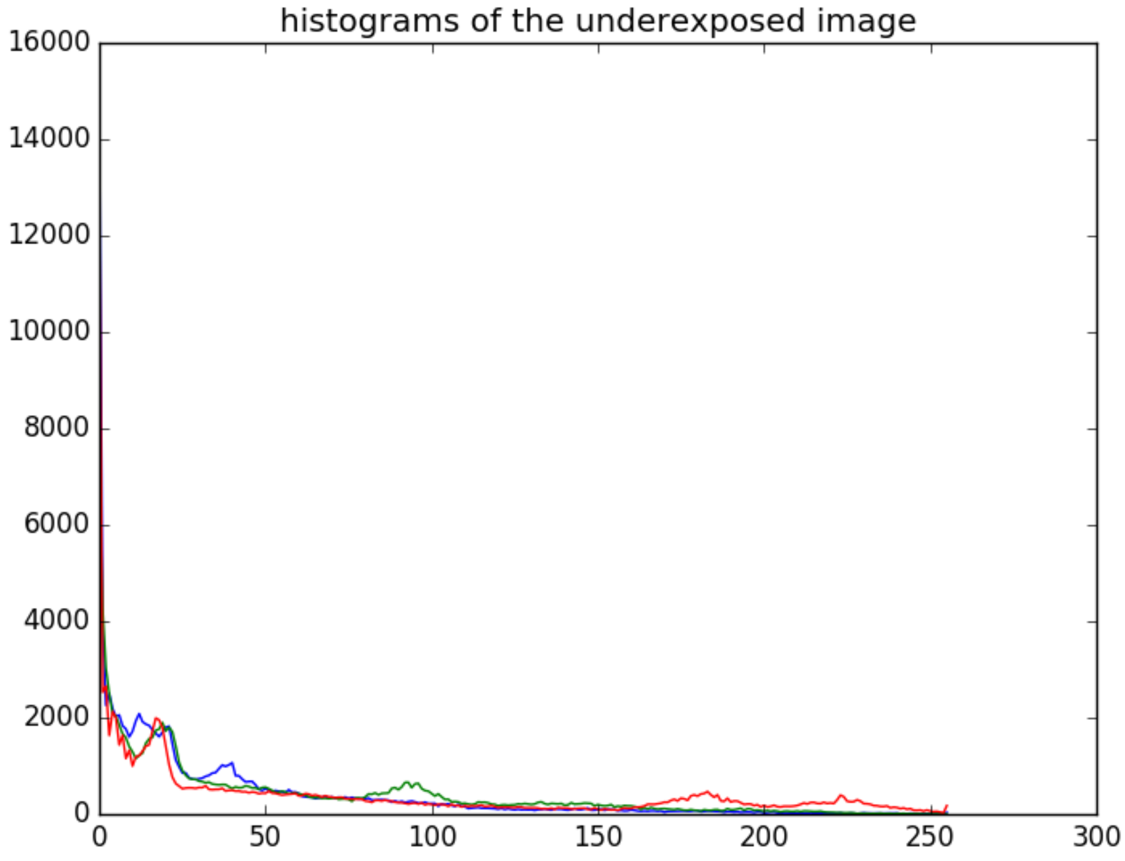
where S = the standard deviation of a sample,
Σ means "sum of,"
$\underline{X}$ = each value in the data set,
$\overline{X}$ = mean of all values in the data set,
N = number of values in the data set.

```
standart deviation = [44.241283200571814,
53.269995405993306, 75.137656833114121]
```

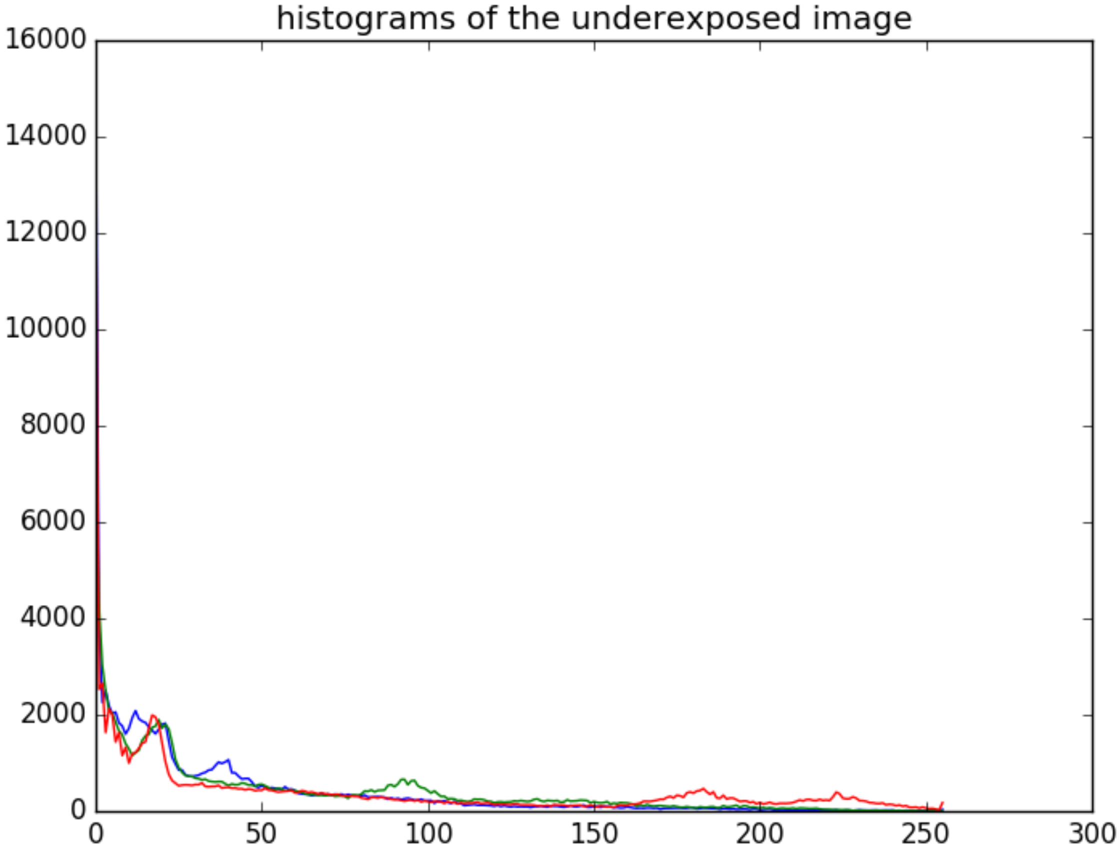
Second part for first question is about to calculate histogram of an image.

```
row, col, ch = img.shape
arr = np.array(img)

hist = [[0 for x in range(256)] for y in range(ch)]

for i in range(row):
   for j in range(col):
      for k in range(ch):
         hist[k][arr[i][j][k]] += 1
```

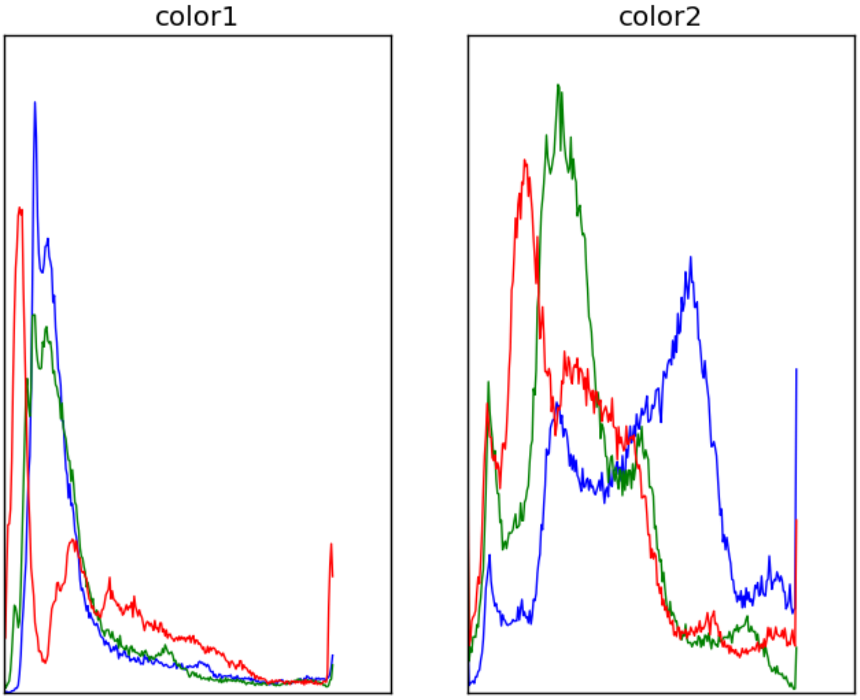
Here; row, column and number of channel info are taken by img.shape function. The image is rendered into an array by using array function of Numpy library. Elements' value of the array may be between 0 and 255. Each values' numbers are calculated for each channel. It is plotted.



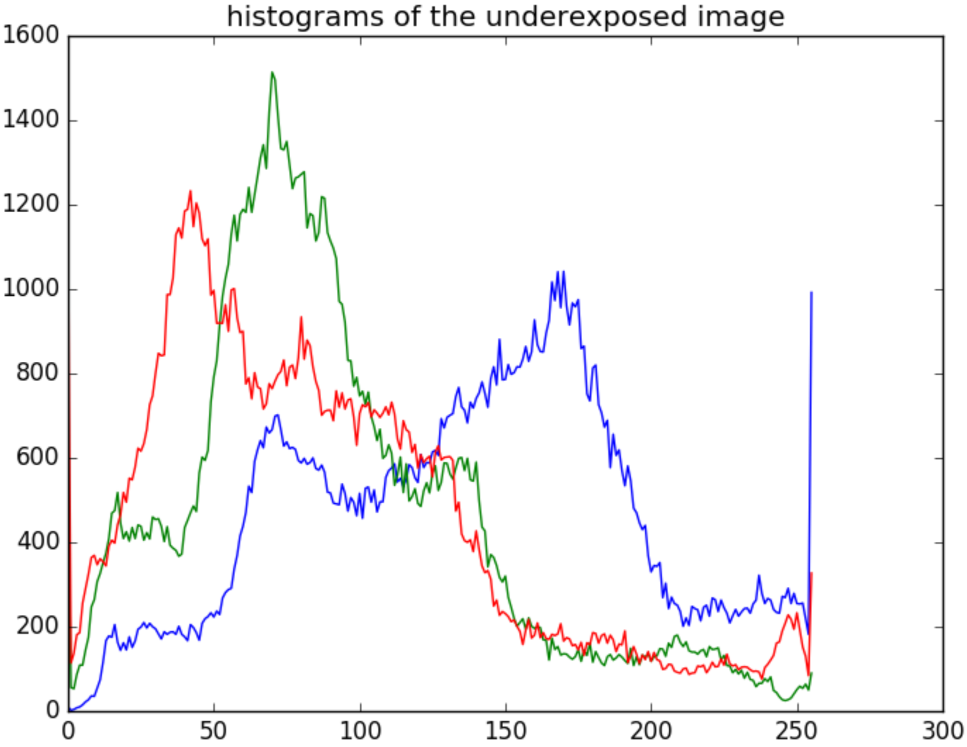
Cdf is calculated by using Numpy.cumsum function.



Q2) Histograms of the both image named Color1 and Color2 are calculated and plotted. Same method is used for this question.

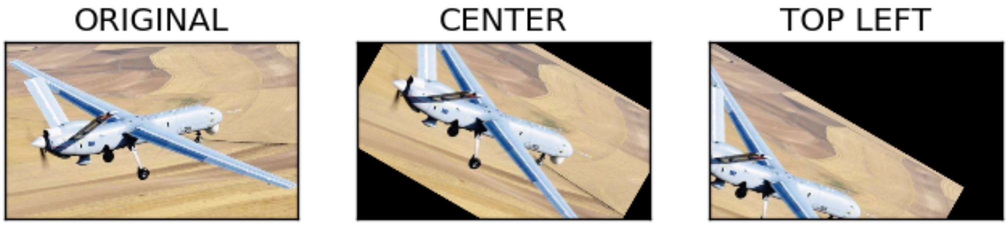


Histogram of the matched image:



Q4) You are given following image. Rotate the image by 30 degrees clockwise with respesct to: (i) image center used as the center of rotation (ii) top left corner used as the center of rotation.

```
rows,cols,ch = img.shape
M = cv2.getRotationMatrix2D((cols/2, rows/2),330,1)
dst1 = cv2.warpAffine(img,M,(cols,rows))
cv2.imwrite('cenrotanka.jpg',dst1)

M = cv2.getRotationMatrix2D((0, 0),330,1)
dst2 = cv2.warpAffine(img,M,(cols,rows))
cv2.imwrite('toprotanka.jpg',dst2)
```



ORIGINAL

CENTER

TOP LEFT