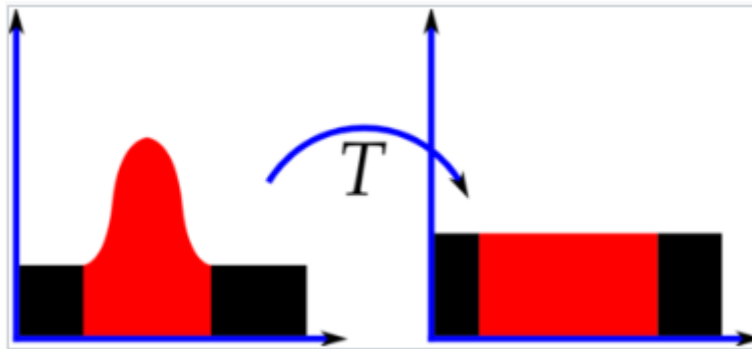


Histogram Equalization

In the [previous blog](#), we discussed contrast stretching, a linear contrast enhancement method. In this blog, we will learn Histogram Equalization which automatically increase the dynamic range based on the information available in the histogram of the input image.

Histogram Equalization, as the name suggests, stretches the histogram to fill the dynamic range and at the same time tries to keep the histogram uniform as shown below



Source: [Wikipedia](#)

By doing this, the resultant image will have an appearance of high contrast and exhibits a large variety of grey tones.

Mostly we will not be able to perfectly equalize the histogram. This is only possible if we assume continuous intensity values.. But in reality, intensity values are discrete thus perfectly flat histograms are rare in practical applications of the histogram equalization.

The transformation function used in this is

$$s_k = T(r_k) = (L - 1) \sum_{j=0}^k p_r(r_j)$$

where 's' and 'r' are the output and input pixel intensities respectively. 'L' is the maximum intensity value(for n bit image $L = 2^n$). The probability of occurrence of the intensity level r_j in

the image is approximated by

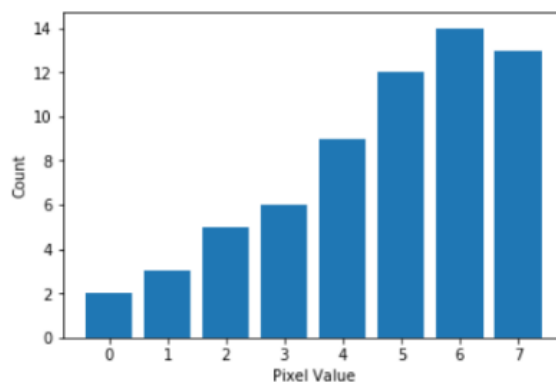
$$p_r(r_j) = \frac{n_j}{MN}$$

Here, MN is the total number of pixels in the image and n_j is the number of pixels that have intensity r_j .

Now, let's take an example to understand how to perform Histogram Equalisation using the above equations.

Suppose we have a 3-bit, 8×8 image whose pixel count and corresponding histogram is shown below

r_k	n_k
0	2
1	3
2	5
3	6
4	9
5	12
6	14
7	13



Now, using the above transformation function we calculate the equalized intensity values.
For instance

$$p_r(r_0) = \frac{n_0}{MN} = \frac{2}{64} = 0.03$$

$$s_0 = T(r_0) = (L-1) \sum_{j=0}^0 p_r(r_j) = 7 p_r(r_0) = 0.21$$

and;

$$p_r(r_1) = \frac{n_1}{MN} = \frac{3}{64} = 0.05$$

$$s_1 = (L-1) \sum_{j=0}^1 p_r(r_j) = 7 p_r(r_0) + 7 p_r(r_1) = 0.56$$


Doing this for all values we get

r_k	n_k	$P_r(r_k)$	S_k
0	2	0.03	0.21
1	3	0.05	0.56
2	5	0.08	1.12
3	6	0.09	1.75
4	9	0.14	2.73
5	12	0.19	4.06
6	14	0.22	5.60
7	13	0.20	7.00

Because the pixel values can only be integers so we round the last column(s_k) to the nearest integer as shown below

r_k	n_k	$P_r(r_k)$	S_k	Round
0	2	0.03	0.21	0
1	3	0.05	0.56	1
2	5	0.08	1.12	1
3	6	0.09	1.75	2
4	9	0.14	2.73	3
5	12	0.19	4.06	4
6	14	0.22	5.60	6
7	13	0.20	7.00	7

So, the round column is the output pixel intensity. The last step is to replace the pixel values in the original image(r_k column) with the round column values. For example, replace 0 with 0, 1 with 1, 2 with 1 and so on. This results in the histogram equalized image.

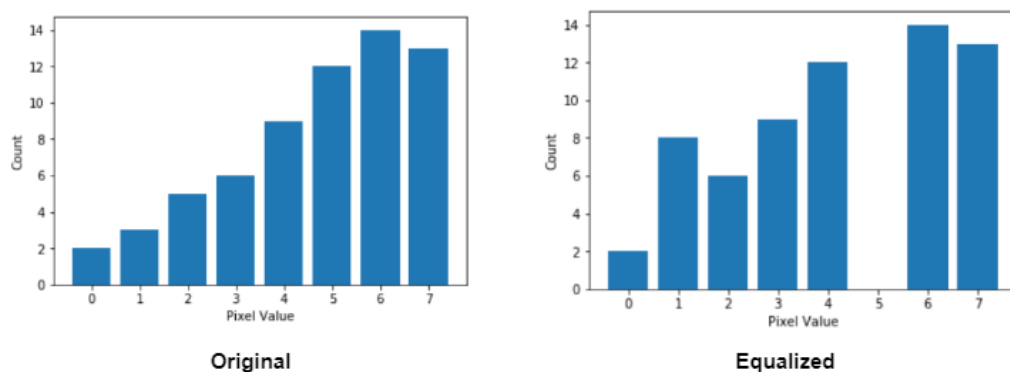


Qualidade Comprovada Di
Você. Apenas R\$76,50

Growth Supplements

To plot the histogram, count the total pixels belonging to the rounded intensity values(See Round and n_k column). For example, 2 pixels belonging to 0, 8 pixels for 1, 6 pixels for 2 and so on.

The initial and equalized histogram is shown below



Sometimes rounding to nearest integer yield non-zero minimum value. If we want the output to range from say [0,255] for 8-bit, then we need to apply stretching (as we did in Min-Max stretching) after rounding.

Histogram Equalization often produces unrealistic effects in photographs and reduce color depth(no. of unique grey levels) as shown in the example above(See pixel value 5). It works best when applied to images with much higher color depth.

Let's see OpenCV function for Histogram Equalization

```
1 equalized_img = cv2.equalizeHist(greyscale_img)
```

Its input is grayscale image and output is our histogram equalized image.

Hope you enjoy reading.

If you have any doubt/suggestion please feel free to ask and I will do my best to help or improve myself. Good-bye until next time.



Xiaomi Redmi 8A Global Version 6,22...

Ad Banggood.com

Notebook Samsung Odyssey Intel Core...

Ad Submarino.com

Histogram Matching (Specification)

theailearner.com

Oferta Black Friday Oi

Ad Oi

Adaptive Histogram Equalization (AHE)

theailearner.com

Contrast Stretching

theailearner.com

Comparing Histograms using OpenCV-Python

theailearner.com

Unsharp Masking and Highboost filtering

theailearner.com

Histogram Backprojection

theailearner.com

Intensity-level Slicing

theailearner.com

Log Transformation

theailearner.com

contrast stretching TheAILEarner

theailearner.com

Intensity Transformation

theailearner.com

This entry was posted in Image Processing and tagged contrast stretching, cv2.equalizeHist(), histogram equalization, image processing, opencv python on 1 Apr 2019 [<https://theailearner.com/2019/04/01/histogram-equalization/>] by kang & atul.
