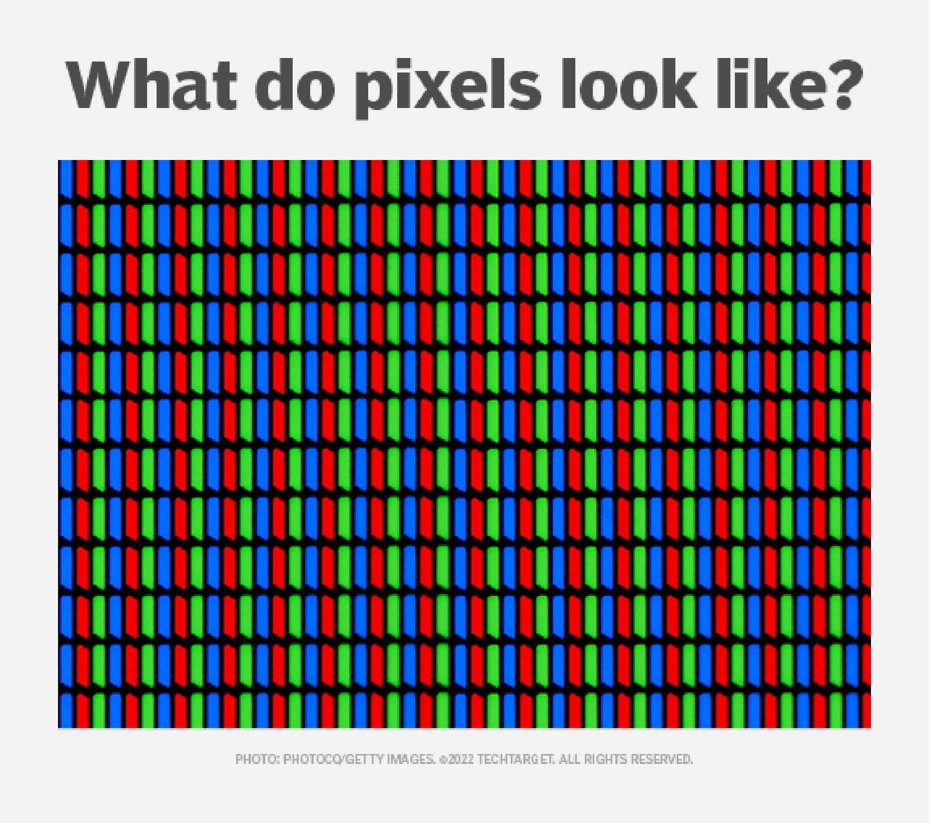
Pixels and Histogram

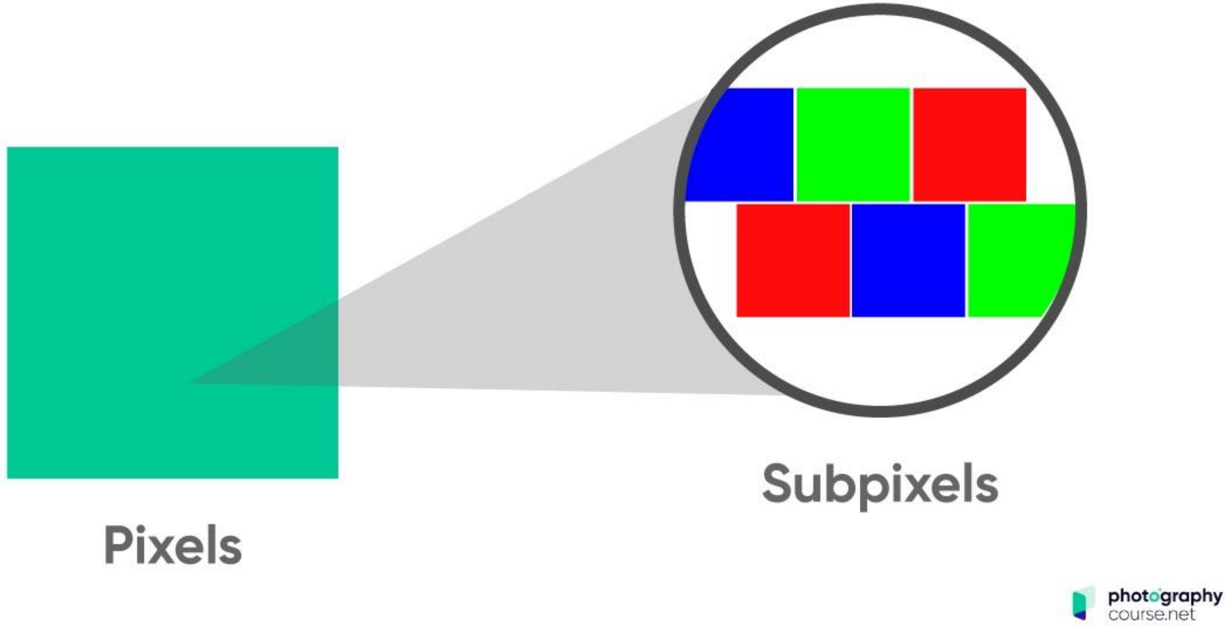
DongJoon Kim

| Phone : +82-2-940-8461
| Email : dongjoonkim@kw.ac.kr

Pixel

- Each pixel is made up of a red, green and blue subpixel that lights up at different intensities to create different colors.



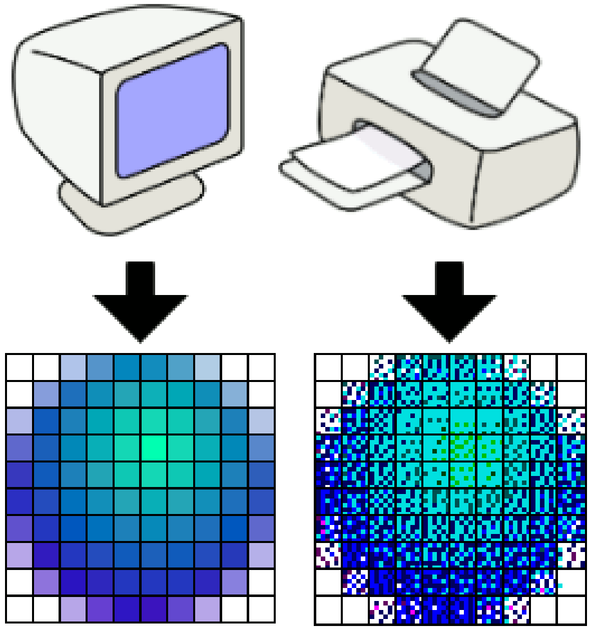


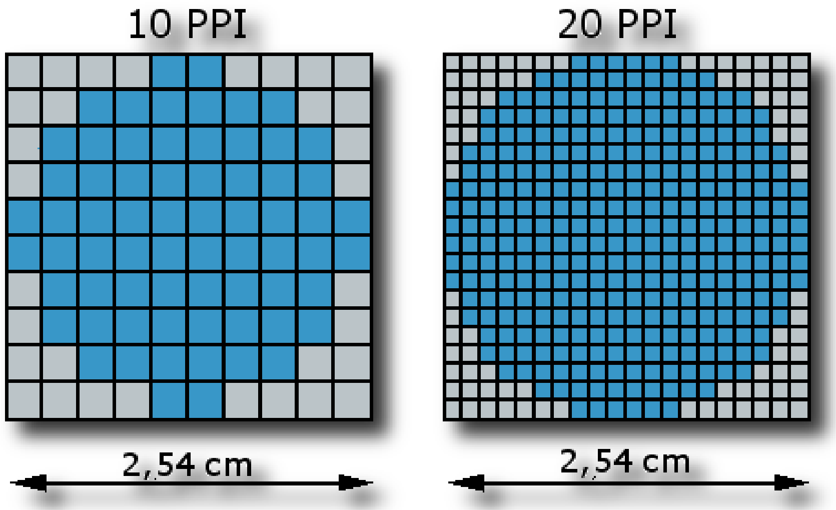
Pixel: DPI vs PPI

- DPI (Dots Per Inch) is a measure of spatial printing, video or image scanner dot density
- PPI (Pixels Per Inch) is a measure of the pixel density of an electronic image/display device
- Number of dots/pixels that can be placed in a line within the span of 1 inch (2.54 cm)

https://photographycourse.net/dpi-vs-ppi/

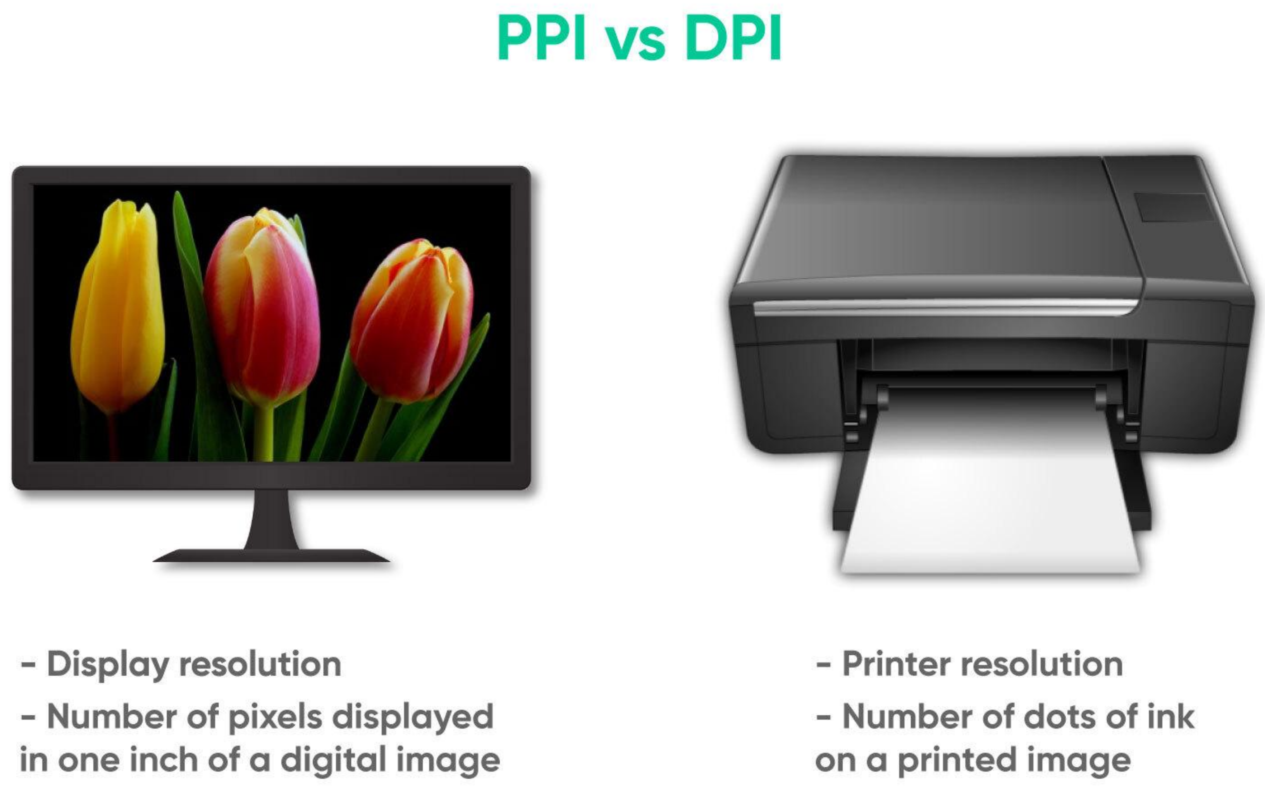
$$\text{PPI (monitor)} = \frac{\text{Number of Pixels}}{\text{Size in Inches}} = \frac{1920}{20} = 96 \text{ ppi}$$





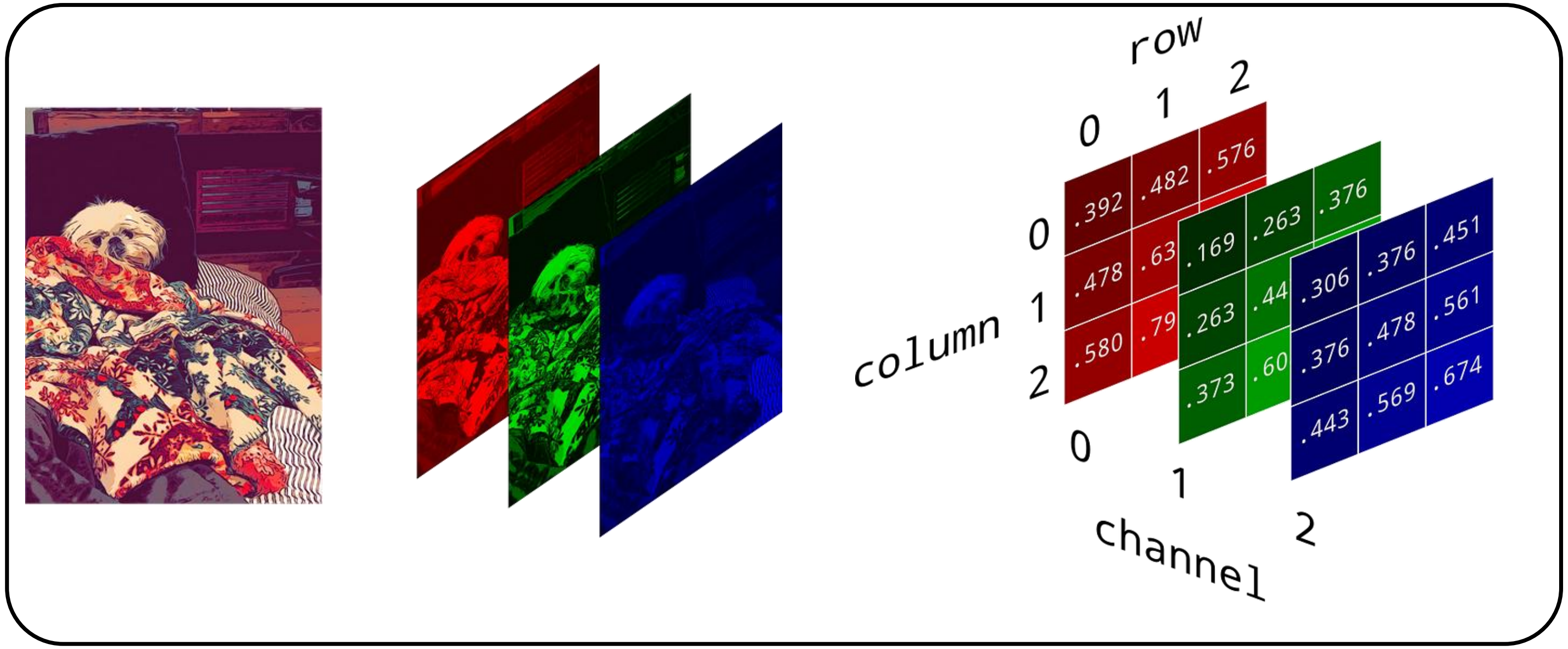
PPI vs DPI: Do they affect each other?

- Imagine you want to print a 300 PPI image at 600 DPI
  - Simply divide 600 DPI/ 300 PPI and you have your answer 2 (DPI/PPI)

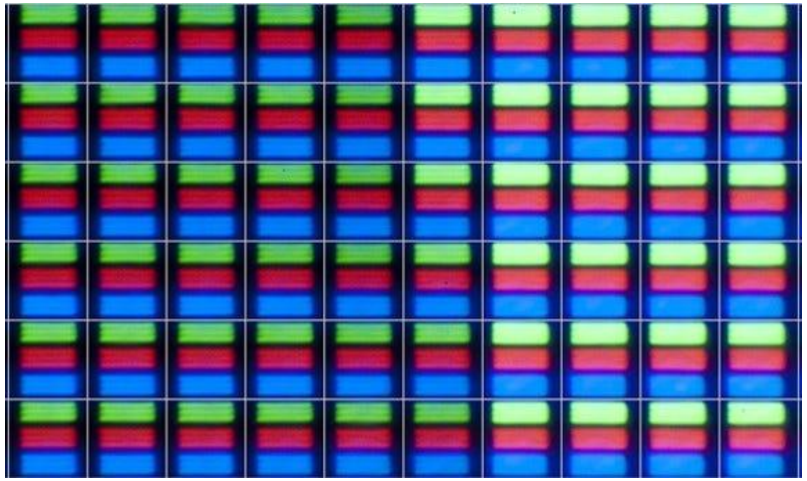


https://photographycourse.net/dpi-vs-ppi/

Pixel Values



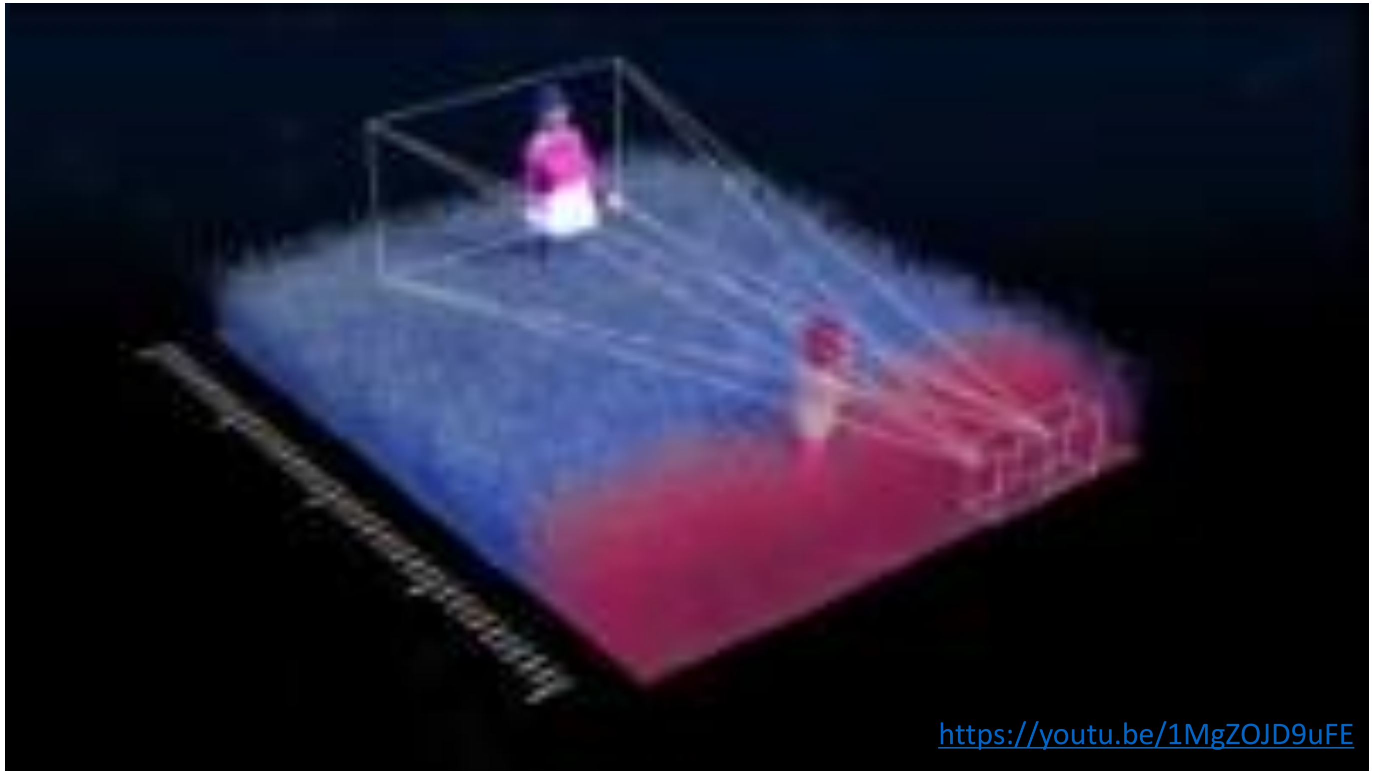
Pixel array on an Image



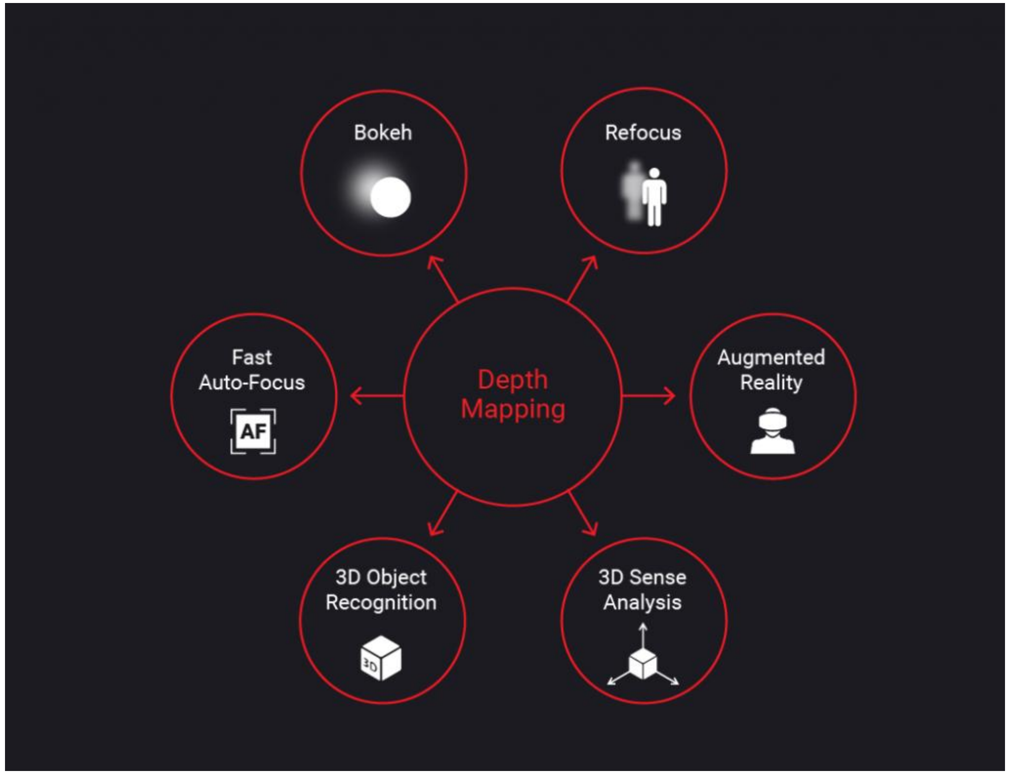
Pixel array on a display monitor

Image vs (Image) Map

- Image represents a pixel array what we see, or a picture
- (Image or 2D) Map represents an array where each element is mapped to each pixel of the target image



https://youtu.be/1MgZOJD9uFE



https://corephotonics.com/technologies/depth-mapping/

Now we have pixels and their values

Let’s analyze them!

for further processing

Where do we start? 🤔

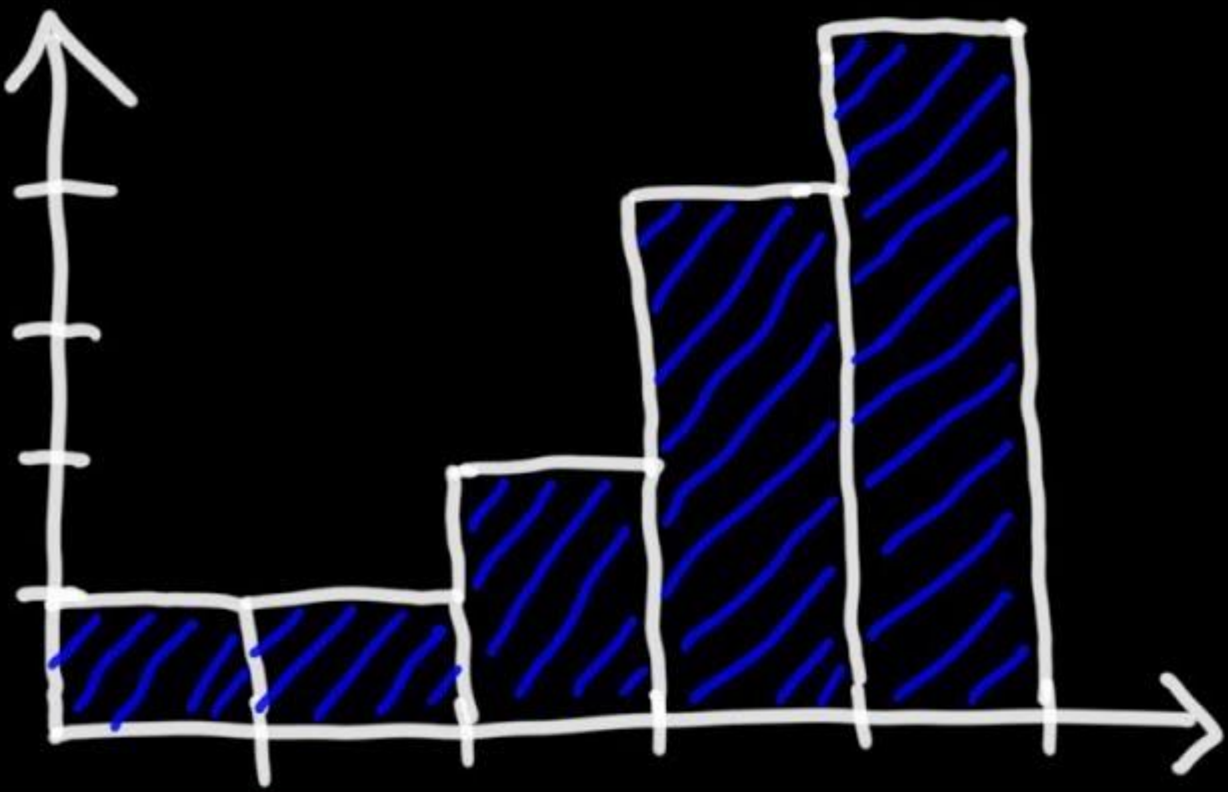
Histograms

1. How many students received, at most, a score of 69 on the exam?

2. How many students received a score of at least 80 on the exam?

3. How many students received a score between 60 and 90?

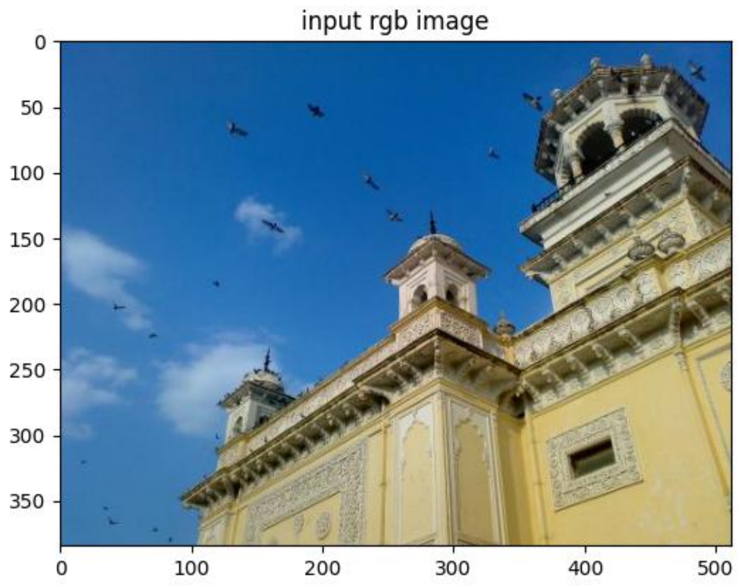
Grades vs freq.

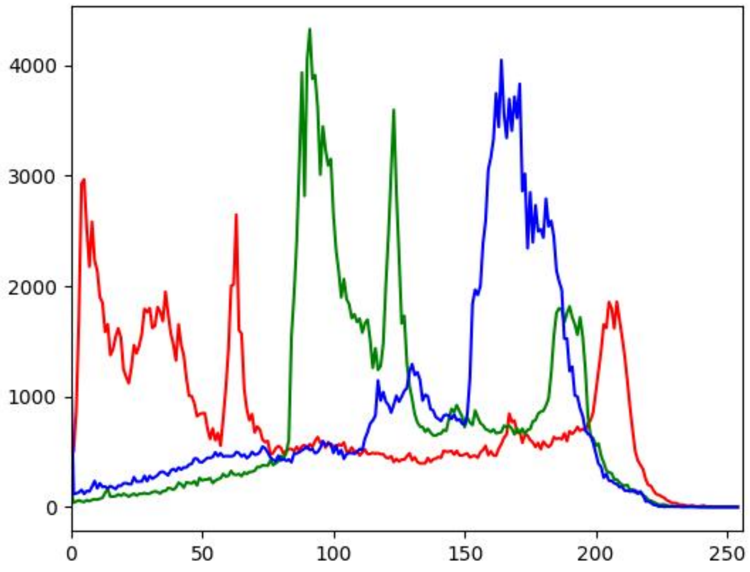


Test Scores: 74, 83, 69, 95, 78, 85, 42, 98, 73, 68, 90, 85, 84, 71, 88, 52, 94

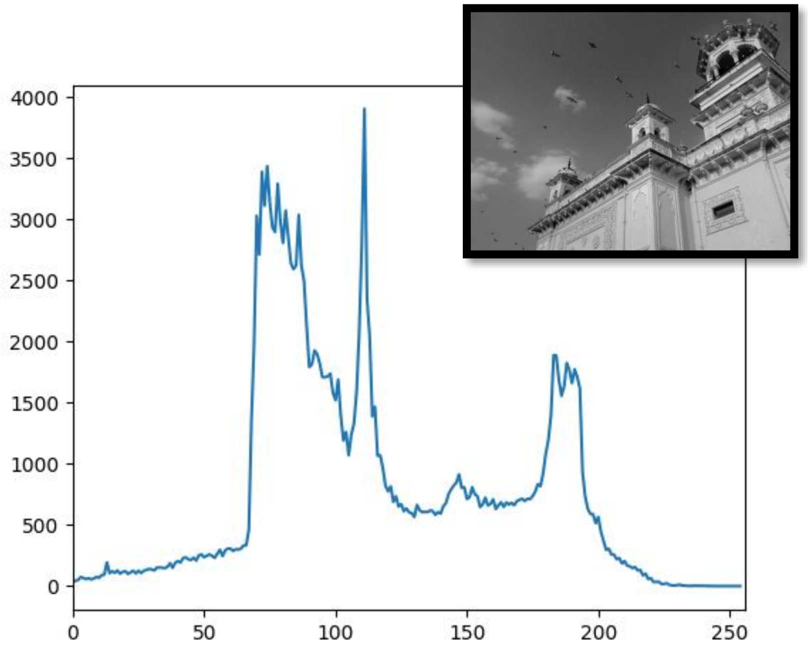
(Image) Histogram

- A plot with pixel values in x-axis and corresponding number of pixels in the image on y-axis
- For a RGB image, there are three histograms
- A representative value per pixel is normally used, i.e., gray scale image





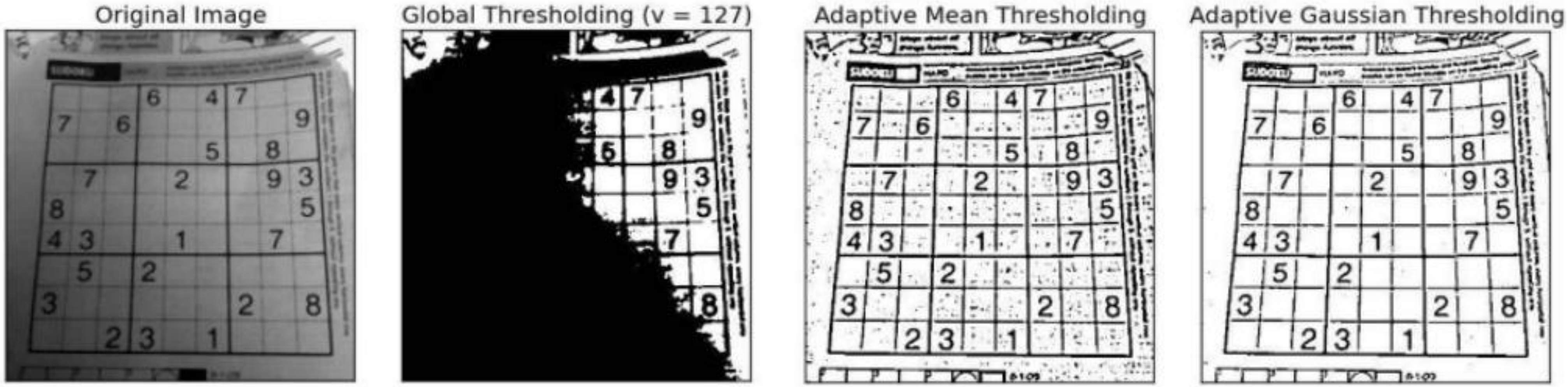
histograms for each RGB channel



1-channel histogram for gray image

Thresholding

- Do something if a metric value is larger/less than a criterion value, i.e., ***threshold value***
- Specific function is applied to pixels whose values are larger/less than a threshold value
- Simple thresholding (global thresholding) uses **a single threshold value**
- Adaptive thresholding uses **‘per-pixel-dynamic’ threshold values**
  - Locally determined by considering a target pixel’s neighborhood pixel values
  - This kind of approach is called ‘**adaptive**’ method



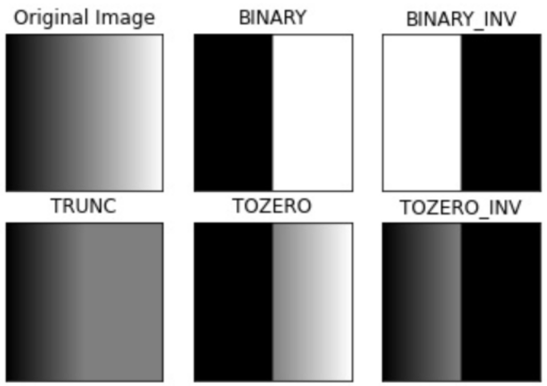
[https://docs.opencv.org/master/d7/d4d/tutorial_py_thresholding.html]

OpenCV

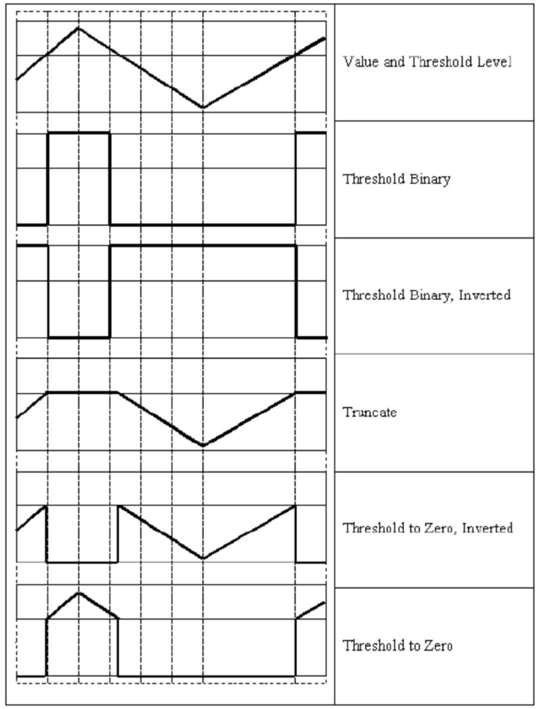
Simple (Global) Thresholding

“matplotlib is a very useful plot GUI in python!!”

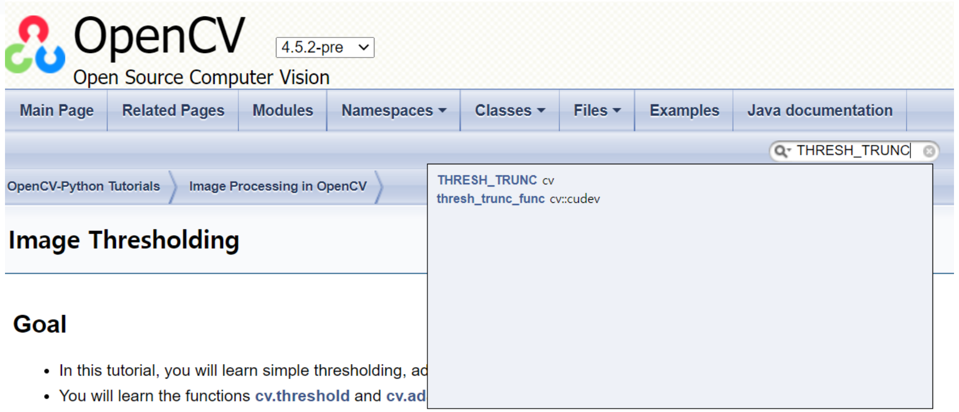
```python
import cv2 as cv
import numpy as np
from matplotlib import pyplot as plt

img = cv.imread('gradient.png',0)
ret,thresh1 = cv.threshold(img,127,255,cv.THRESH_BINARY)
ret,thresh2 = cv.threshold(img,127,255,cv.THRESH_BINARY_INV)
ret,thresh3 = cv.threshold(img,127,255,cv.THRESH_TRUNC)
ret,thresh4 = cv.threshold(img,127,255,cv.THRESH_TOZERO)
ret,thresh5 = cv.threshold(img,127,255,cv.THRESH_TOZERO_INV)

titles = ['Original Image','BINARY','BINARY_INV','TRUNC','TOZERO','TOZERO_INV']
images = [img, thresh1, thresh2, thresh3, thresh4, thresh5]

for i in range(6):
    plt.subplot(2,3,i+1),plt.imshow(images[i],'gray',vmin=0,vmax=255)
    plt.title(titles[i])
    plt.xticks([]),plt.yticks([])

plt.show()
```



image

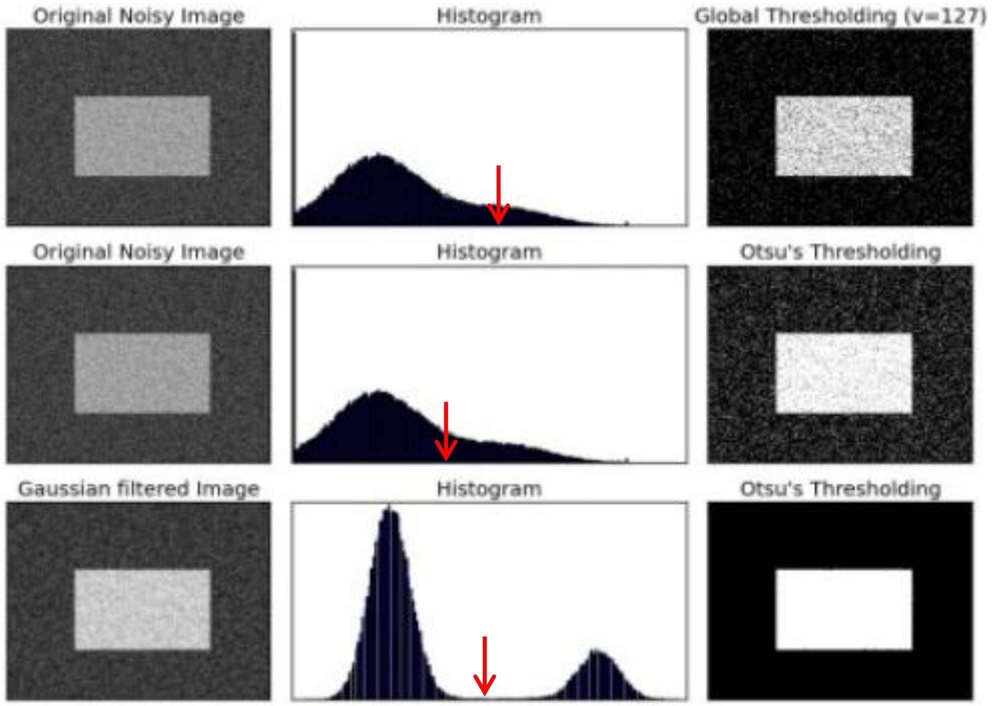


threshold types



Threshold Value

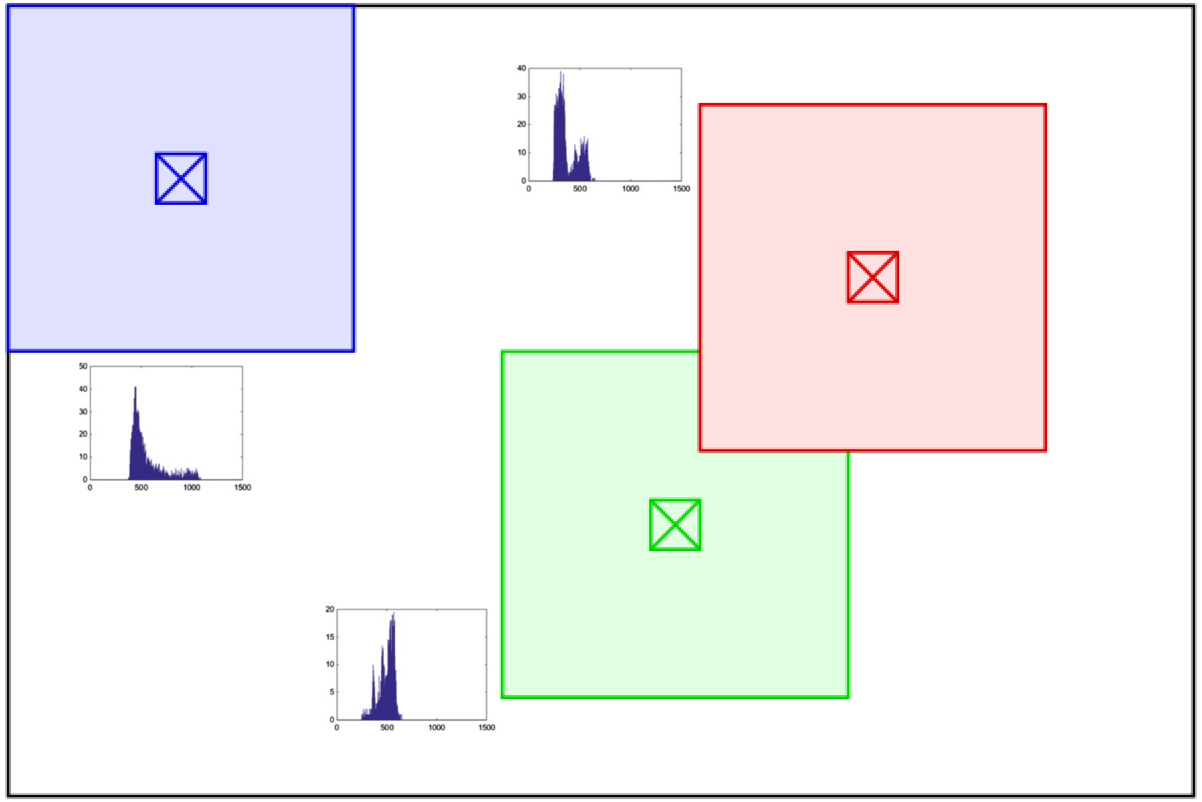
- It is important to determine the appropriate threshold value
- Otsu method (or, Otsu thresholding) determines the threshold value automatically
  - Otsu threshold value minimizes the within-cluster(or class) variance of the histogram
  - useful for a simple binary classification case

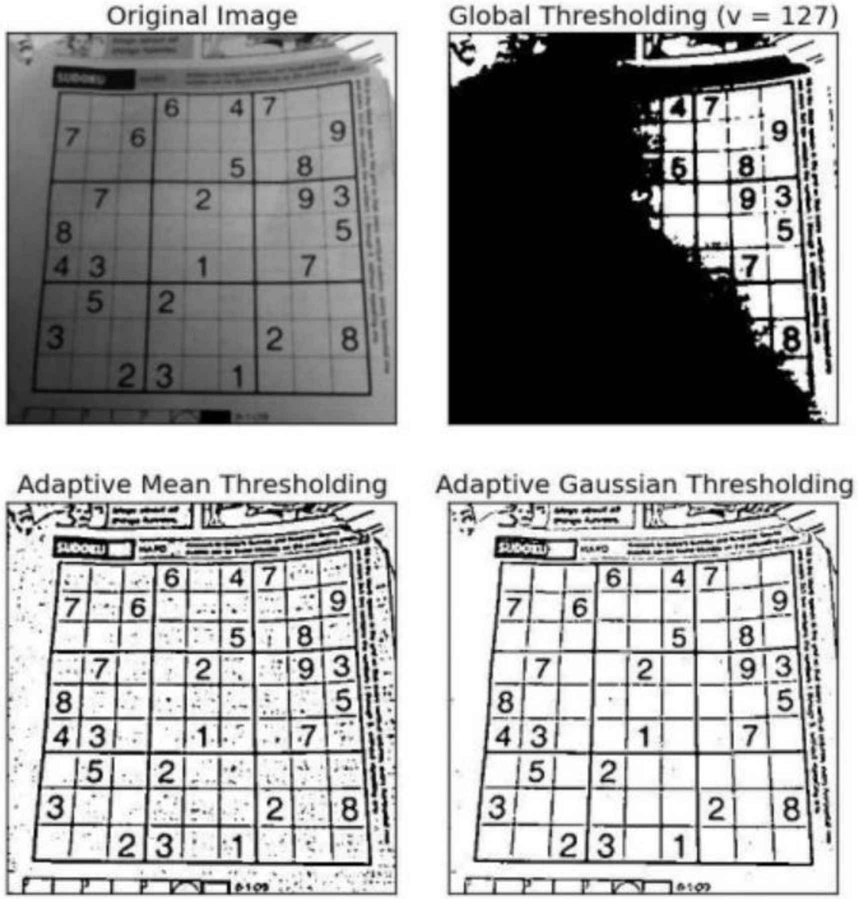


[https://docs.opencv.org/master/d7/d4d/tutorial_py_thresholding.html]

Adaptive Thresholding

- Locally-determined threshold for a pixel based on a small region around it
  - How do we decide the threshold value? ☺



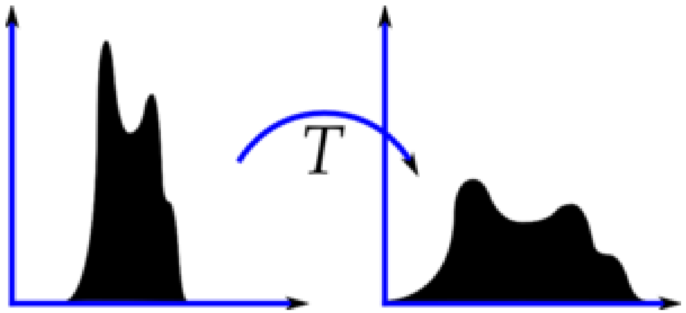


image

Which image looks better?

Histogram Equalization (HE)

- Generally, consider 1-channel histogram (e.g., gray image histogram)
- To enhance the image's contrast, we can modify the individual pixel values
  - Use the histogram processing, <u>**Histogram Equalization**</u>



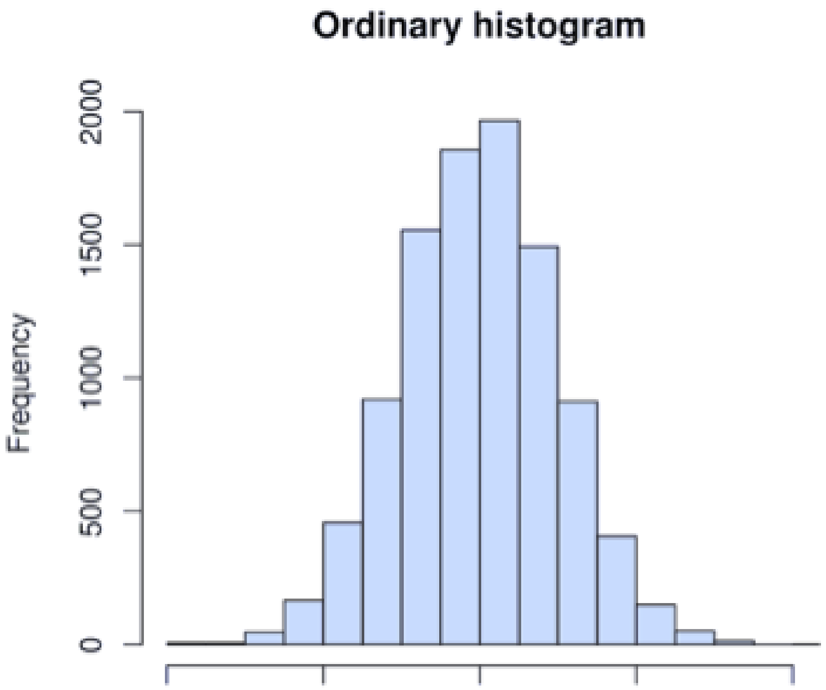
looks better!!

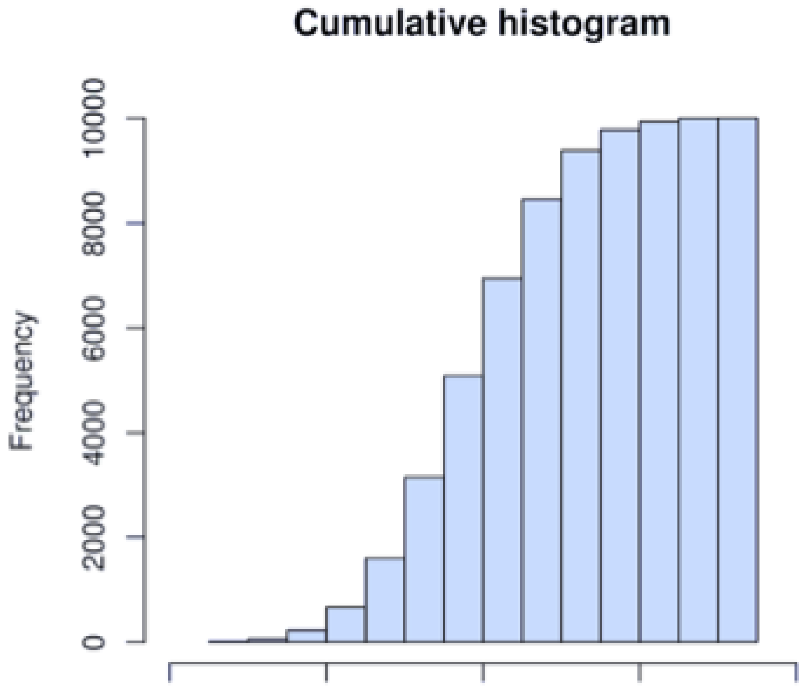
the image uses a small range of intensity values

the small range pixel values are stretched out along the valid pixel values

Cumulative Histogram

- The histogram can be though of a distribution function





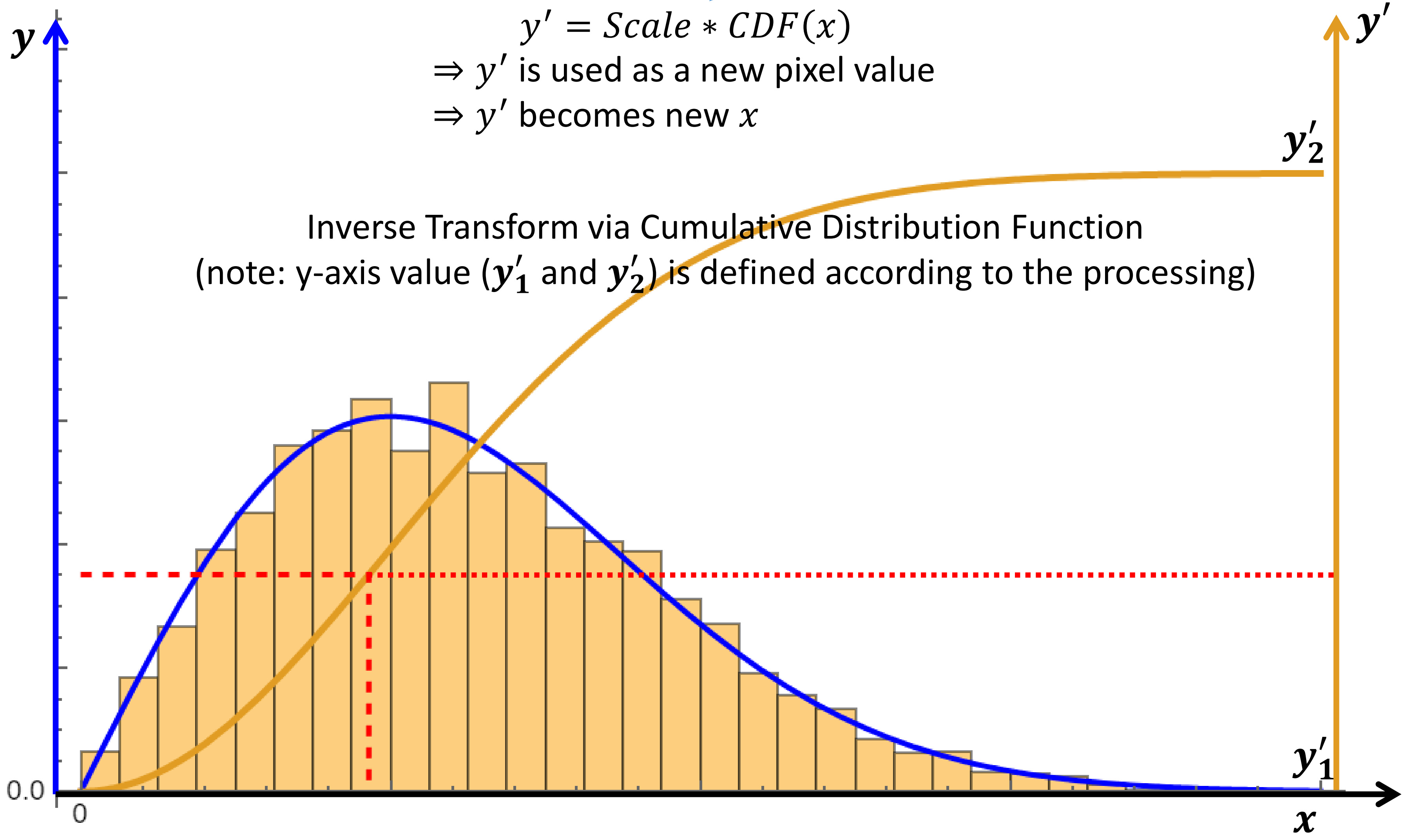
Inverse Transform of a Histogram

$y'_1 = Histo(x) \longrightarrow y'_2 = CDF(x)$

$y' = Scale * CDF(x)$

$\Rightarrow y'$ is used as a new pixel value

$\Rightarrow y'$ becomes new x



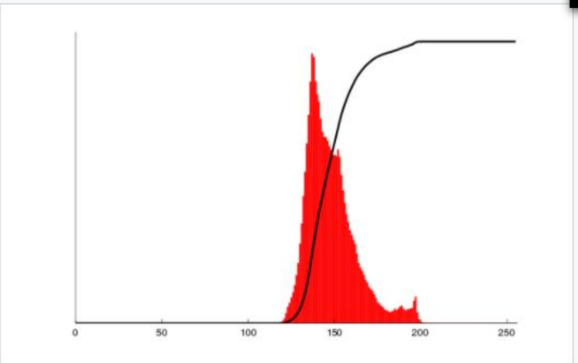
Inverse Transform via Cumulative Distribution Function

(note: y-axis value ($\boldsymbol{y'_1}$ and $\boldsymbol{y'_2}$) is defined according to the processing)

Histogram Equalization (HE)

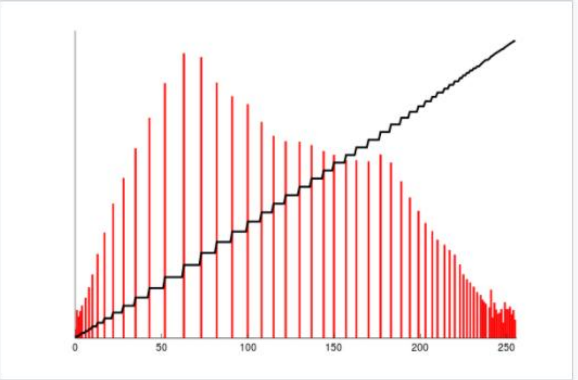
- HE is accomplished by the following steps
  1. Making a histogram of an image
  2. Making a cumulative histogram of the histogram
  3. Mapping min/max (user-defined) pixel values to 0/255 satisfying **linearized CDF**!!
- HE allows to **gain a higher contrast**!

Before Histogram Equalization



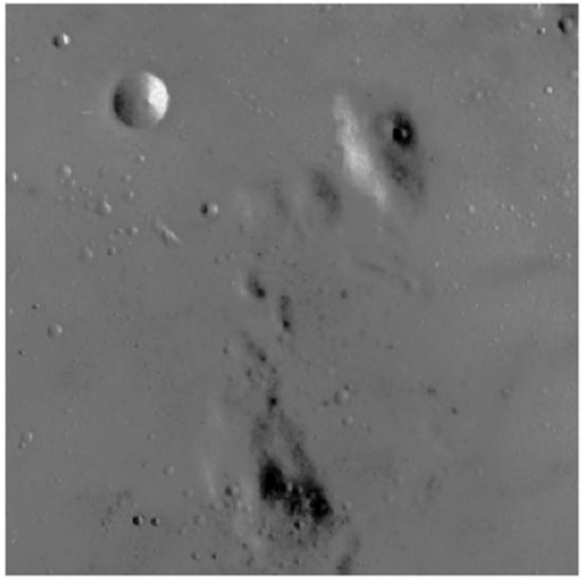
Corresponding histogram (red) and cumulative histogram (black)

After Histogram Equalization

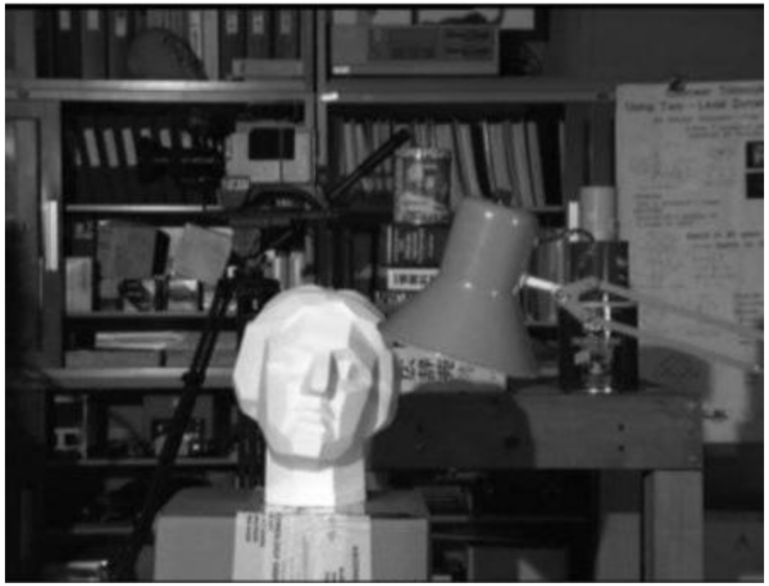


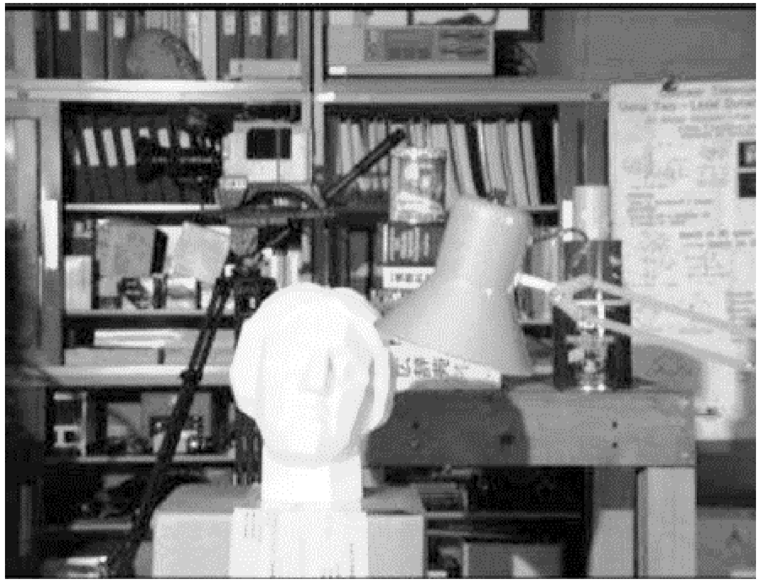
Corresponding histogram (red) and cumulative histogram (black)

Is only HE enough?



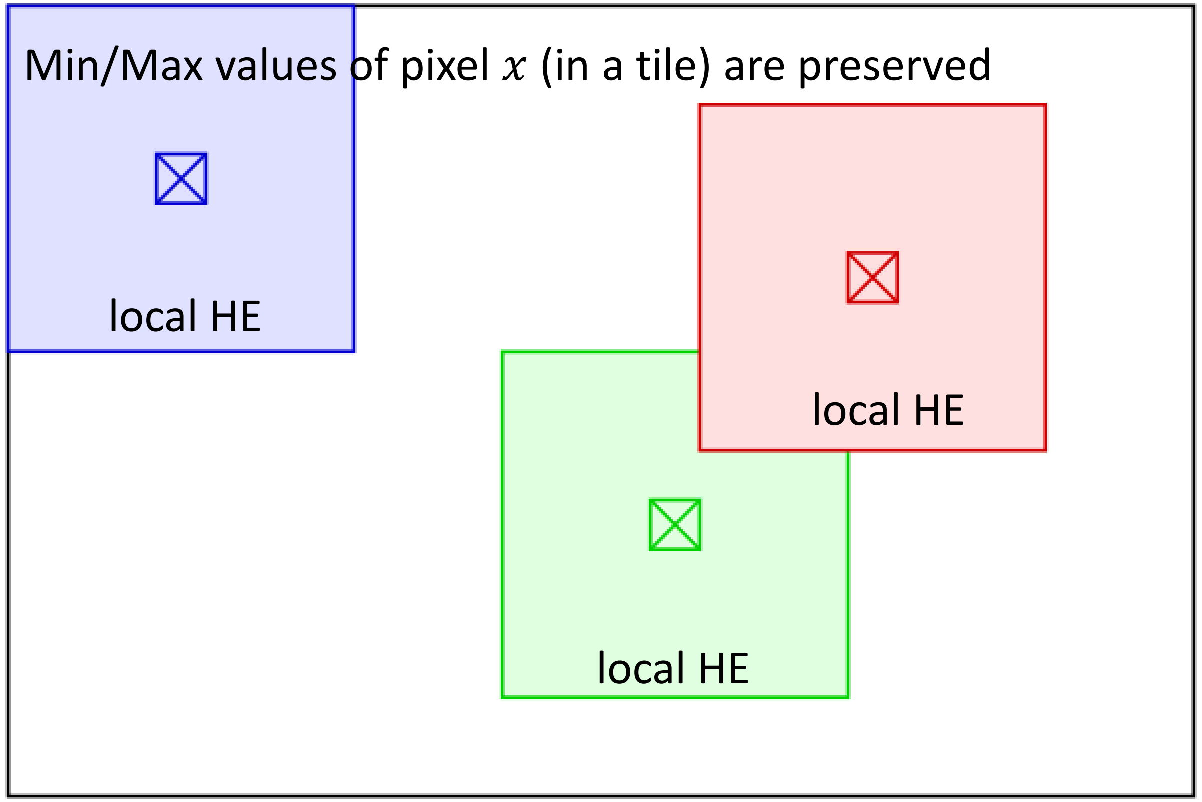
😐?!





Adaptive Histogram Equalization (AHE)

- In the previous, only considered the global contrast of the image
  - In many cases, over- or under- brightness occurs
- Obviously, **Adaptive** method exists based on **local analysis** of the image histogram
  - At a pixel, HE is performed on a **tile**, or **block**, centered at the pixel



[https://3months.tistory.com/407]

OpenCV uses 8x8 tile size by default

Adaptive Histogram Equalization (AHE)

- In the previous, only considered the global contrast of the image
  - In many cases, over- or under- brightness occurs
- Obviously, **Adaptive** method exists based on **local analysis** of the image histogram
  - At a pixel, HE is performed on a **tile**, or **block**, centered at the pixel

over- or under- brightness

better local details!!
with more consistent contrast enhancement!

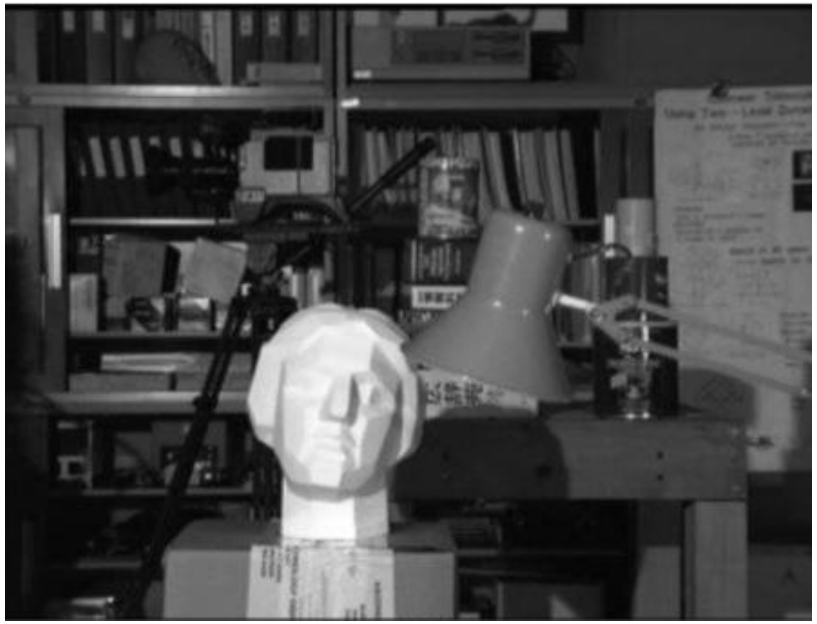
[https://stackoverflow.com/questions/20086032/local-histogram-equalization/28798628]

OpenCV

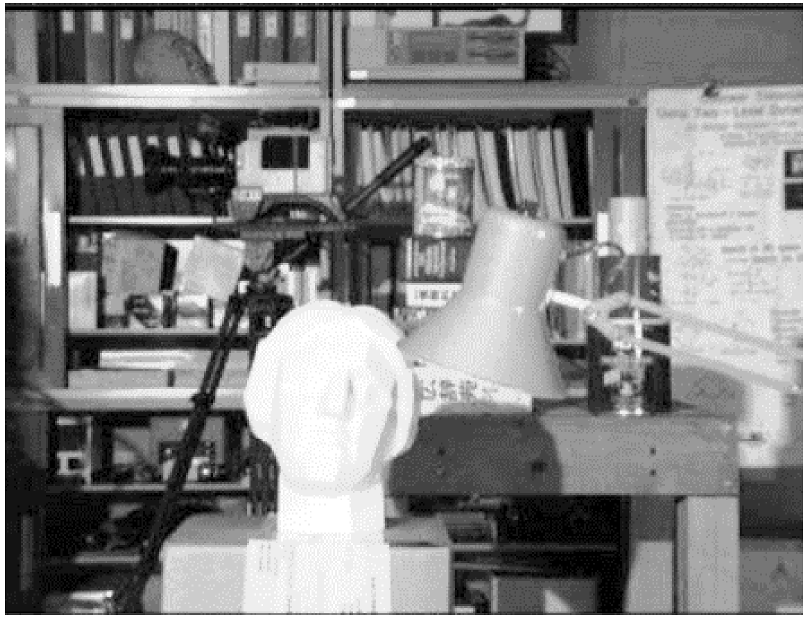
Contrast Limited AHE (CLAHE)

- AHE often suffers from overamplification problem
  - Due to the pixels whose values are biased at a narrow range of the local histogram

original

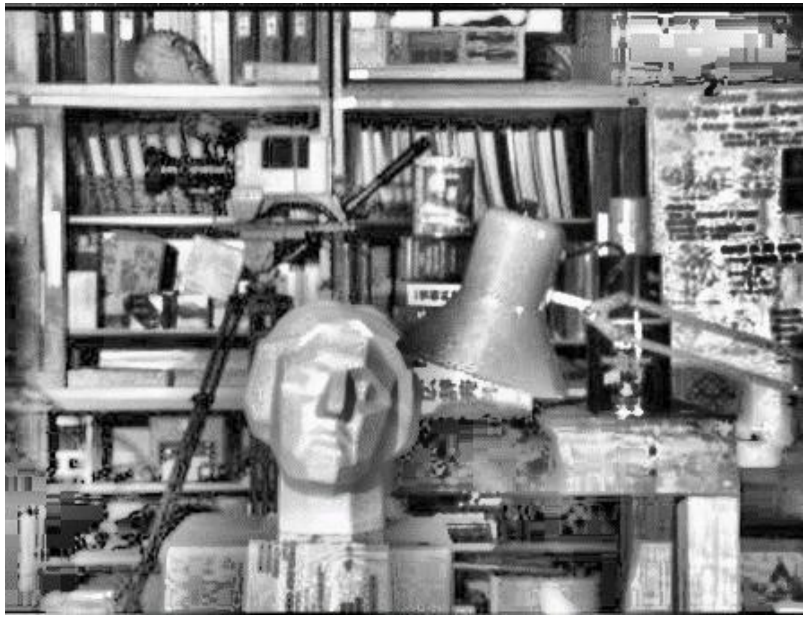


global HE



over-brightness

adaptive HE

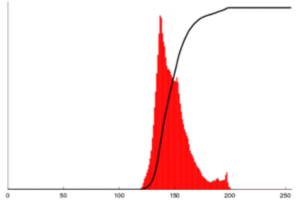


noise-like areas
(by overamplified)

cv.createCLAHE(clipLimit=∞,…).apply(…)

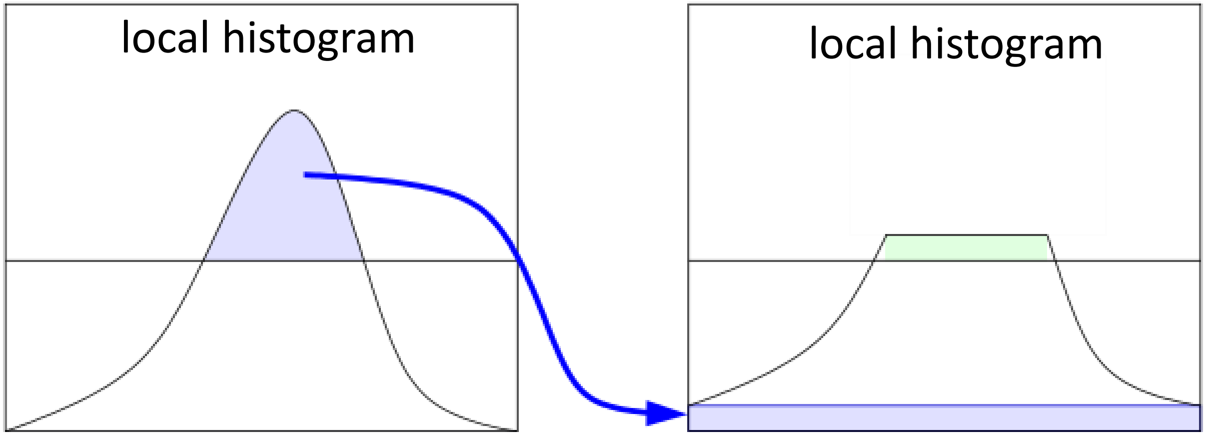
see “test_4_clahe.py”

Contrast Limited AHE (CLAHE)



- AHE often suffers from overamplification problem
  - Due to the pixels whose values are biased at a narrow range of the local histogram
- By limiting contrast (or, clipping contrast), avoid an **over-sloped CDF** of the local histogram

clip limit
normally, 2 ~ 4



Imagine the CDF of each histogram (before/after clipping)!

Note that the linear-like slope of CDF provides better contrast!

Min/Max values of pixel x (in a tile) are preserved

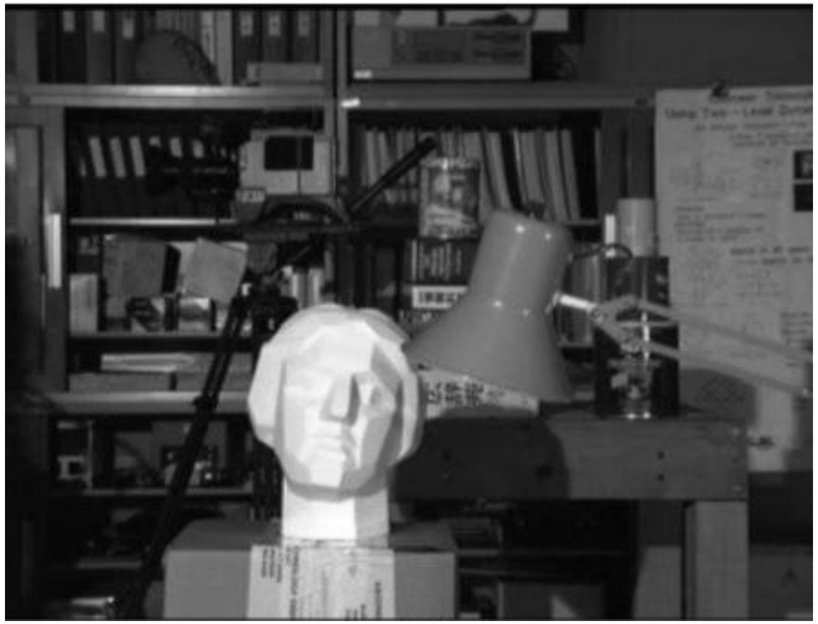
[https://3months.tistory.com/407]

OpenCV

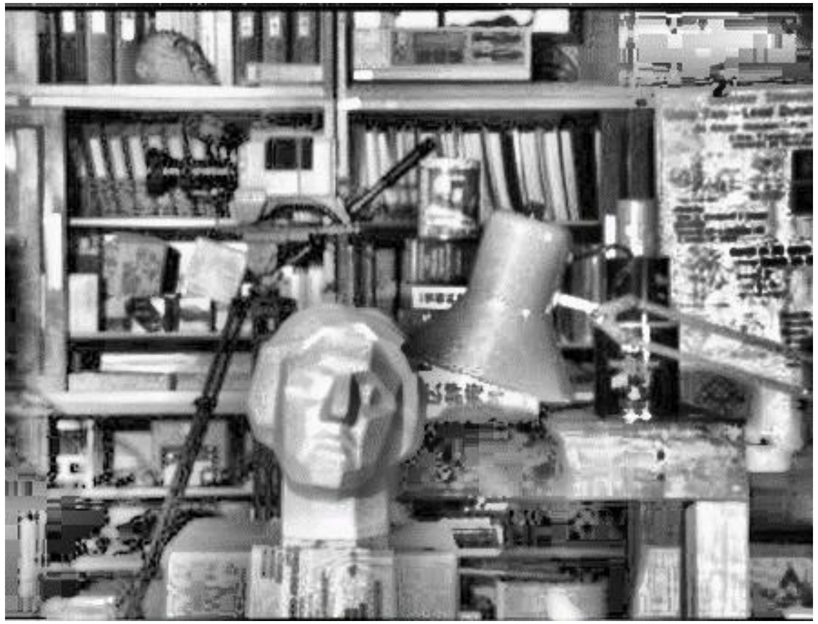
Contrast Limited AHE (CLAHE)

- AHE often suffers from overamplification problem
    - Due to the pixels whose values are biased at a narrow range of the local histogram
- By limiting contrast (or, clipping contrast), avoid an over-sloped CDF of the local histogram

original

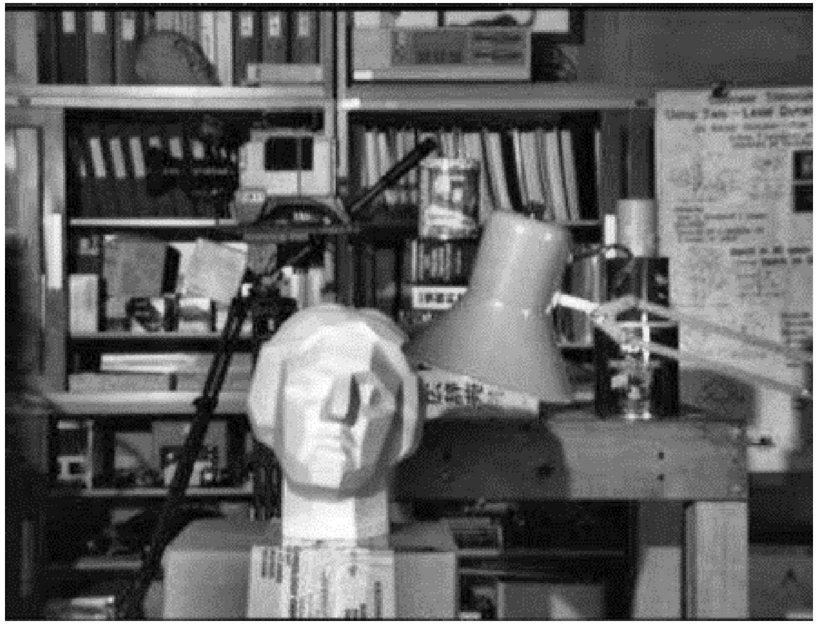


adaptive HE



cv.createCLAHE(clipLimit= ∞,…).apply(…)

CLAHE



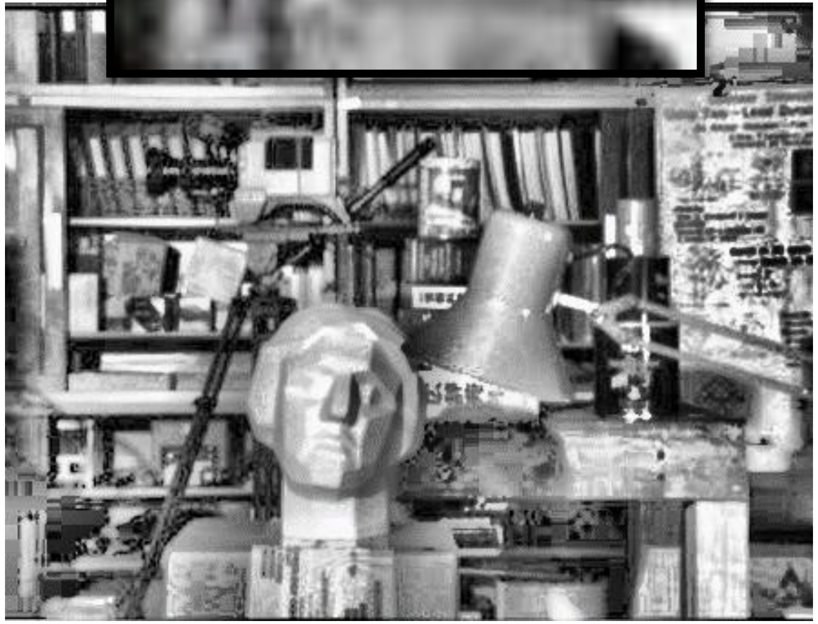
cv.createCLAHE(clipLimit=2,…).apply(…)

see “test_4_clahe.py”

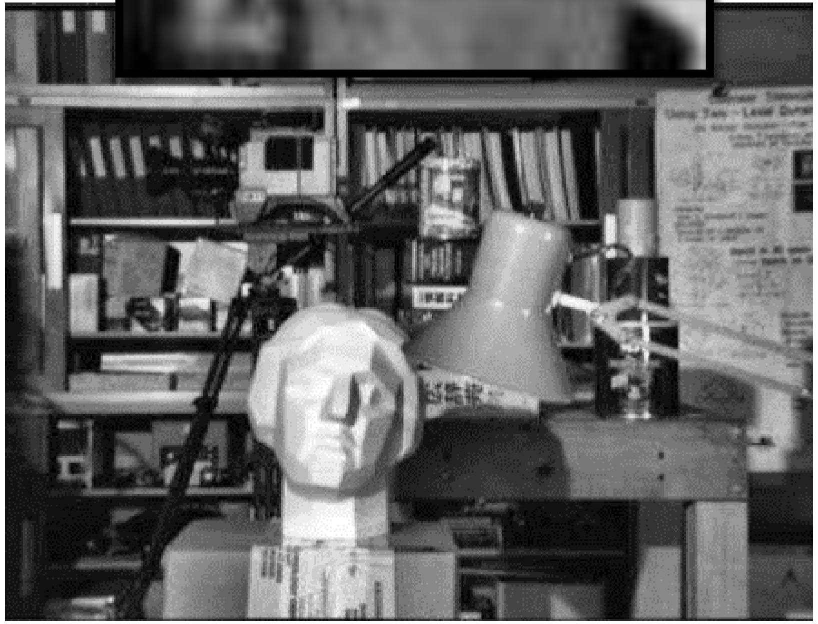
Op

trast E (CLA

- overam hose val row range
- clipping er-sloped am



cv.createCLAHE(clipLimit= ∞,…).apply(…)

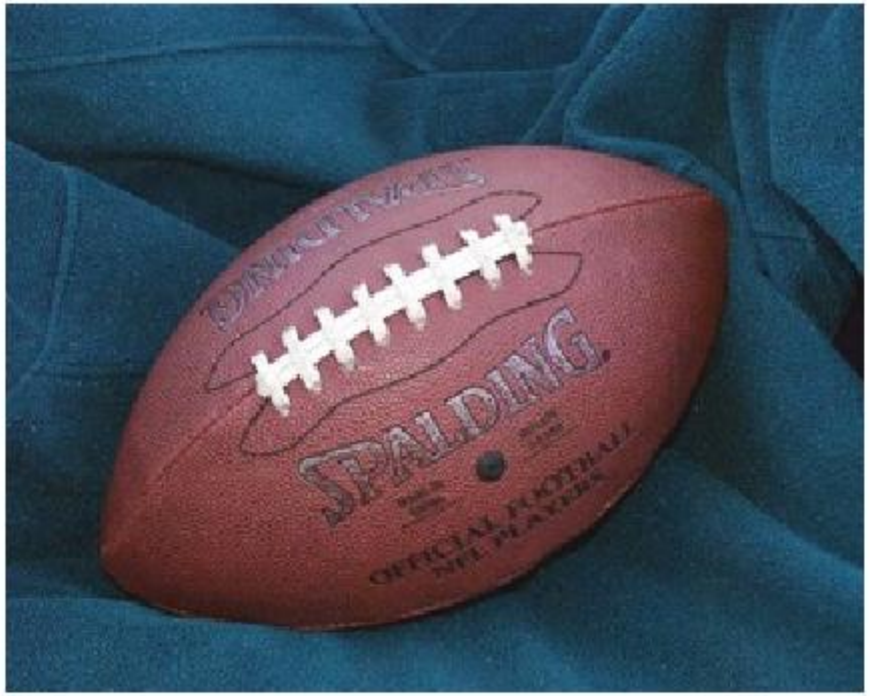


cv.createCLAHE(clipLimit=2,…).apply(…)

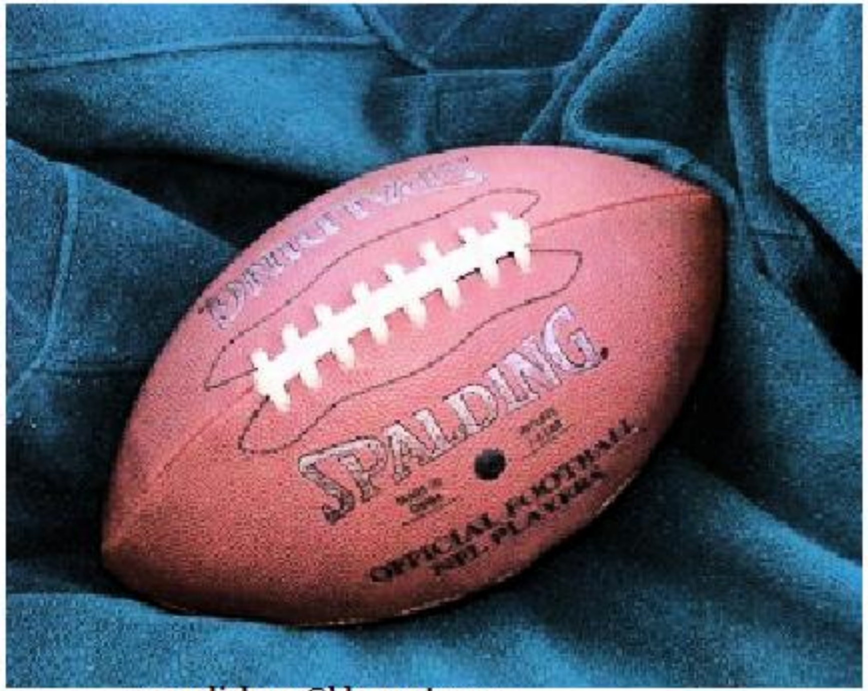
see “test_4_clahe.py”

Can you apply HE to Color Image?

Before Histogram Equalization



After Histogram Equalization



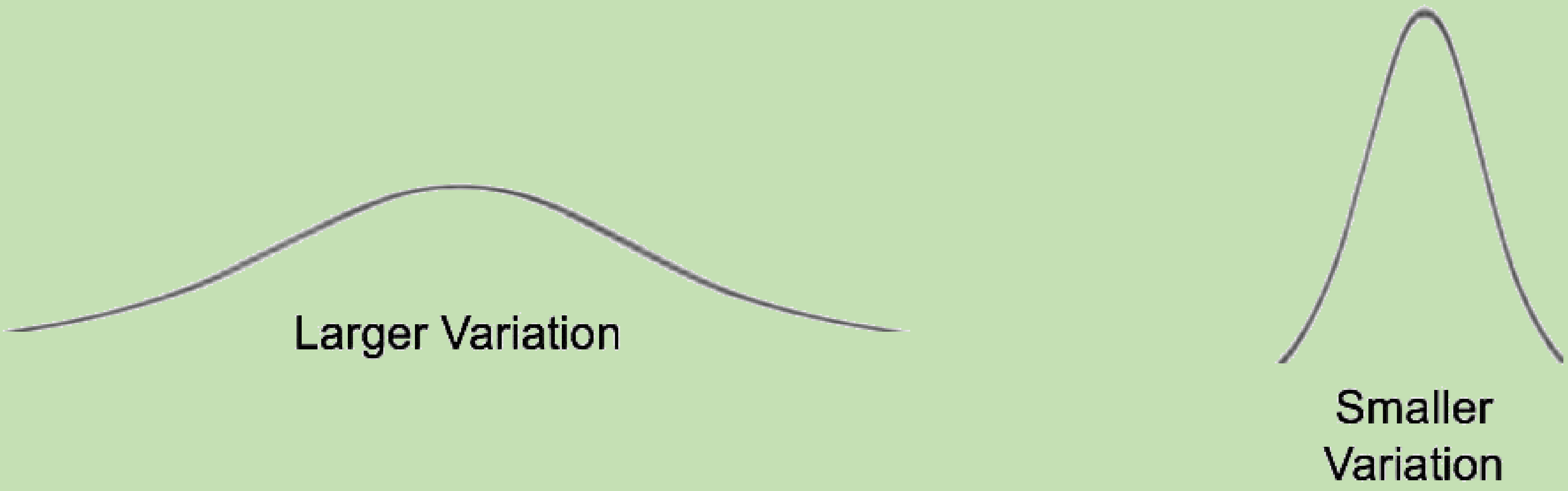
angeljohnsy@blogspot.com

Hint : remember color space you already learned!

☺

Mathematical Approaches for Image Processing

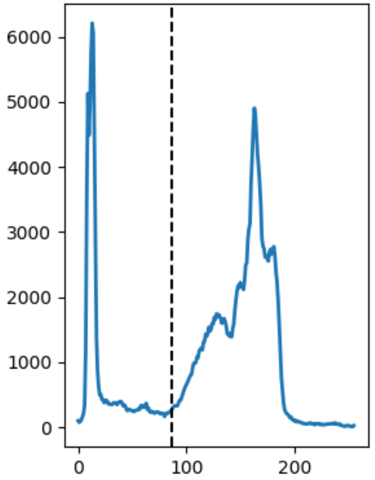
Variance and **Standard Deviation**
in Statistics

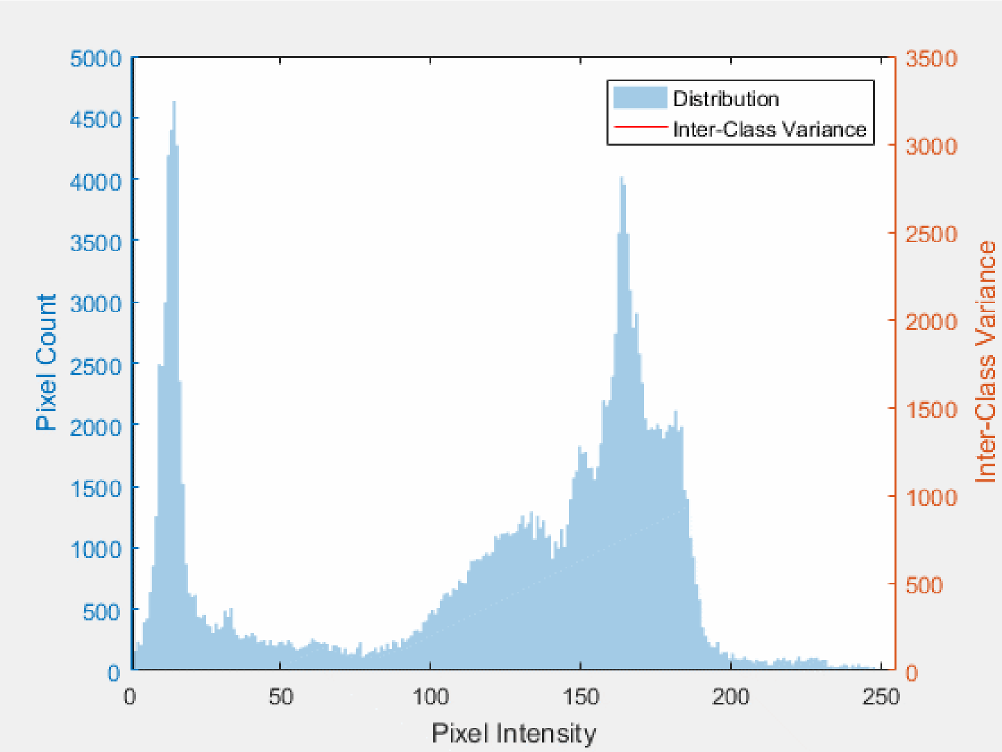


Although they are same concept,
Why Standard Deviation? instead of Variance?

Otsu Thresholding

- Considering the histogram variance enables to detect a global threshold value for binary classification automatically
- Otsu threshold value **maximizes the inter-class variance** of the histogram (or **minimizes the within-class variance**)

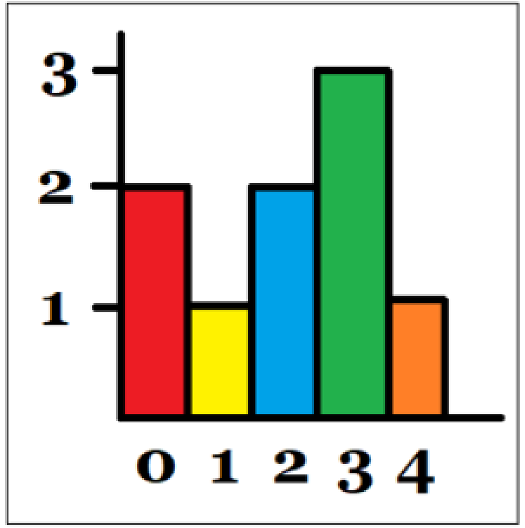




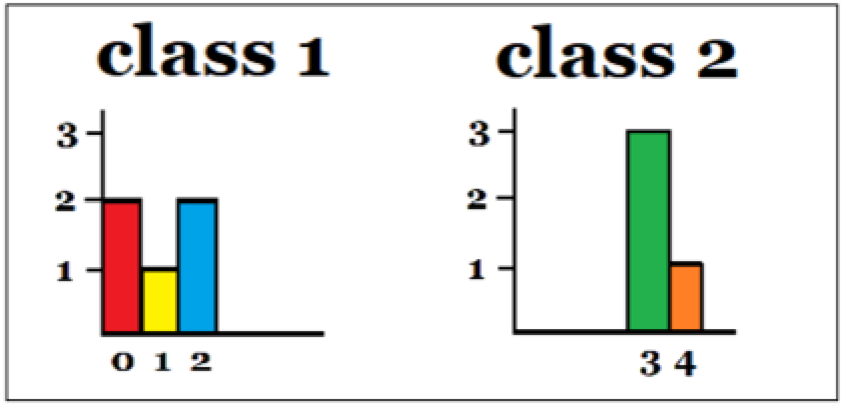
[https://en.wikipedia.org/wiki/Otsu%27s_method]

Otsu Thresholding

- We know ‘variance’ $\sigma^2 = \frac{\sum_{i=0}^{N}(X_i - \mu)^2}{N}$ X_i: i-th pixel value (histogram), μ : mean, N: # of pixels on an image
- Let’s assume that pixels are classified into **2 classes**
- The within-class variance $(V_w) = \sum_{i=0}^{N}(W_i * \sigma_i^2)$ W_i: # of pixels in class i / **N**



divide into $\longrightarrow$ two classes



$W_1 = 5/9$ $W_2 = 4/9$

$\sigma_1^2 = 4/5$ $\sigma_2^2 = 3/16$

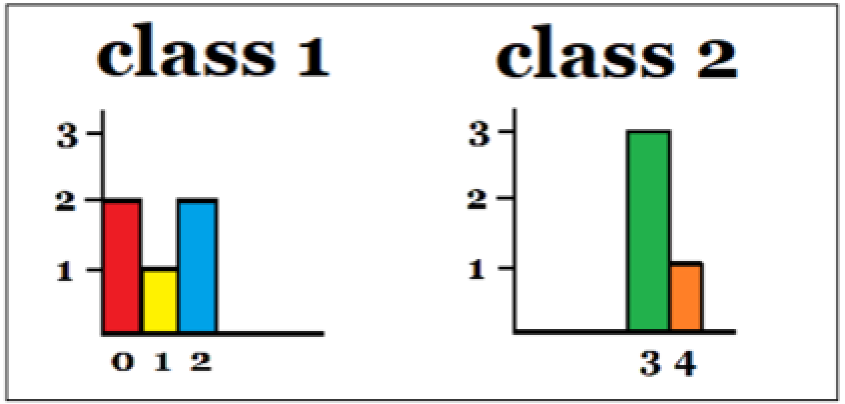
$V_w = W_1 * \sigma_1^2 + W_2 * \sigma_2^2 = 0.52777$

compute V_w along the pixel values in histogram, and find the value that minimizes the V_w

[https://medium.com/@hbyacademic/otsu-thresholding-4337710dc519]

Otsu Thresholding

- We know 'variance' $\sigma^2 = \frac{\sum_{i=0}^{N}(X_i - \mu)^2}{N}$ X_i: i-th pixel value (histogram), μ : mean, N: # of pixels on an image
- Let's assume that pixels are classified into **2 classes**
- The within-class variance $(V_w) = \sum_{i=0}^{N}(W_i * \sigma_i^2)$ W_i: # of pixels in class i / **N**
- The inter-class variance $(V_b) = V_T - V_w$ V_T: total variance



compute V_w along the pixel values in histogram, and find the value that **minimizes the V_w**

compute V_b along the pixel values in histogram, and find the value that **maximizes the V_b**

here, the value is the **Otsu threshold value**

[https://medium.com/@hbyacademic/otsu-thresholding-4337710dc519]

Adaptive Thresholding

- Locally-determined threshold for a pixel based on a small region around it
  - How do we decide the local threshold value?

How about Local-Otsu Thresholding? ☺