IMAGE PRE-PROCESSING

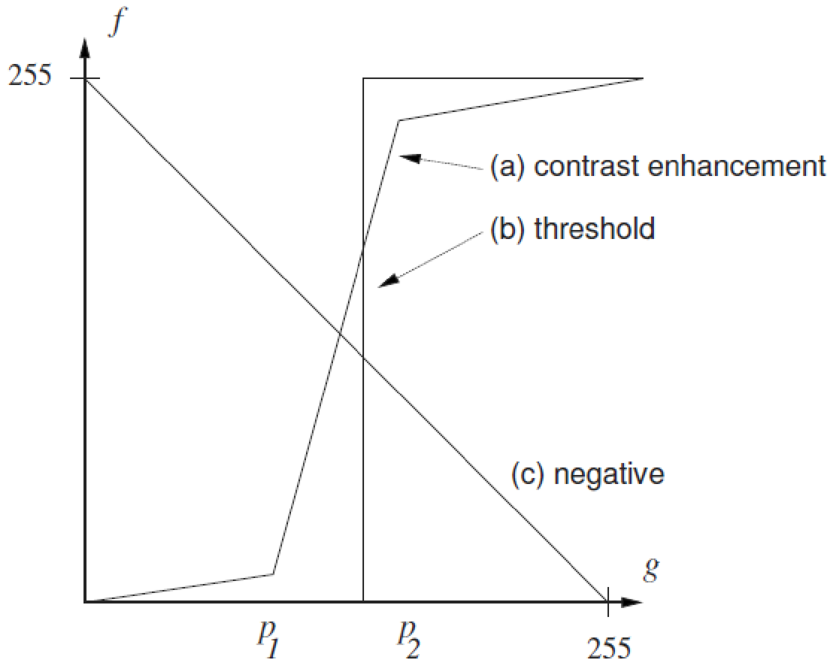
Duangpen jetpipattanapong

GRAY-SCALE TRANSFORMATION

- Gray-scale transformations do not depend on the pixel's position in the image.
- A transformation of the original brightness *p* from scale [*p*0, *pk*] into brightness *q* from a new scale [*q*0, *qk*]

$$q = \mathcal{T}(p)$$

- The most common gray-scale transformations



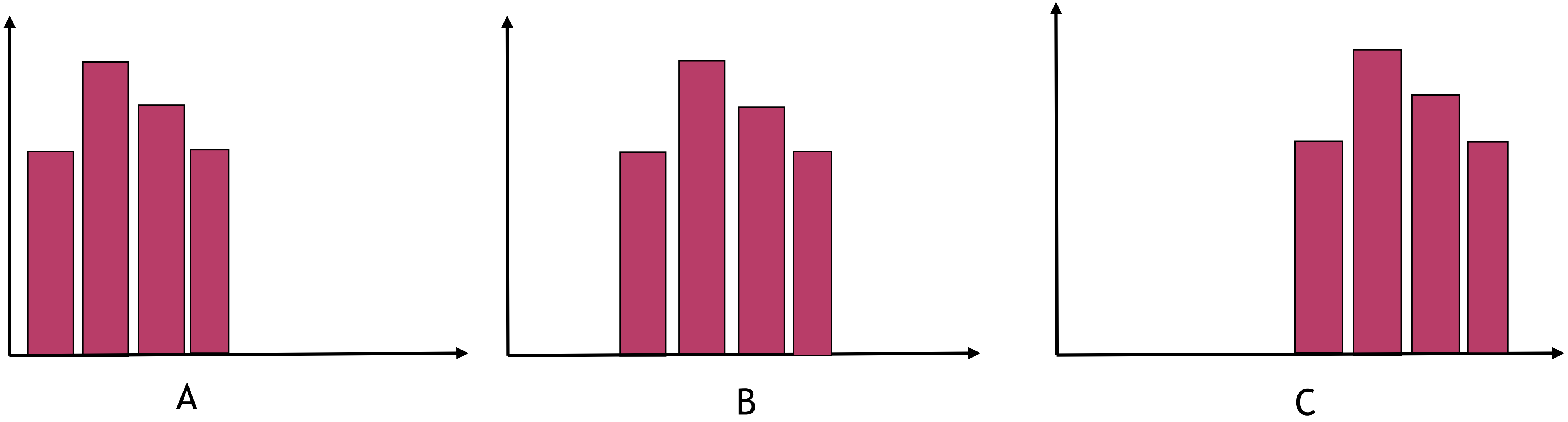
(a) **enhances the image contrast** between brightness values p_1 and p_2

(b) **brightness thresholding** and results in a black-and-white image

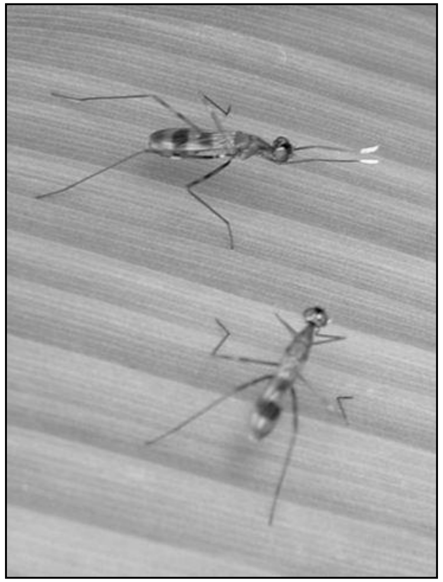
(c) the negative transformation

* The same principle can be used for color displays.

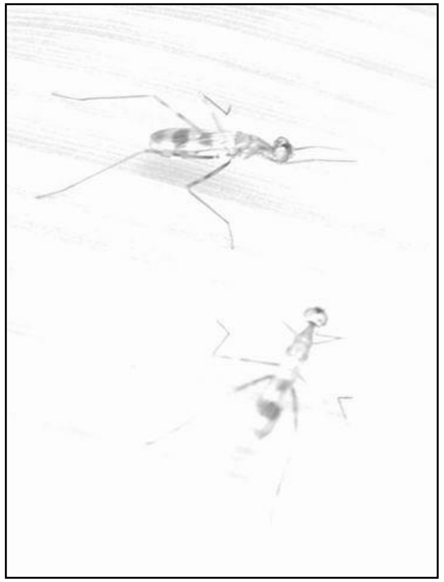
- What is the difference between the images that have these histogram characteristics?



Try to match the pictures and histograms

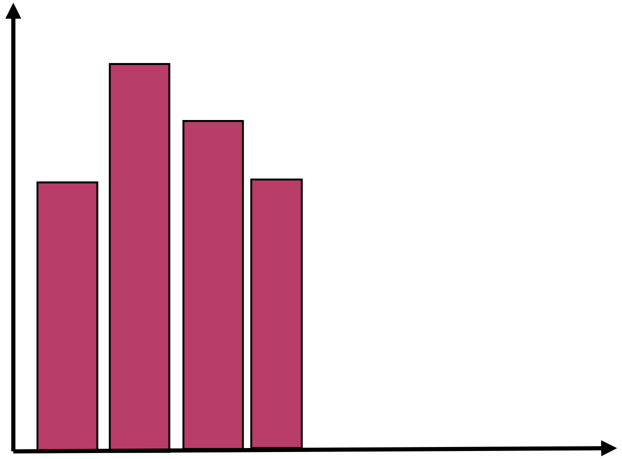


1

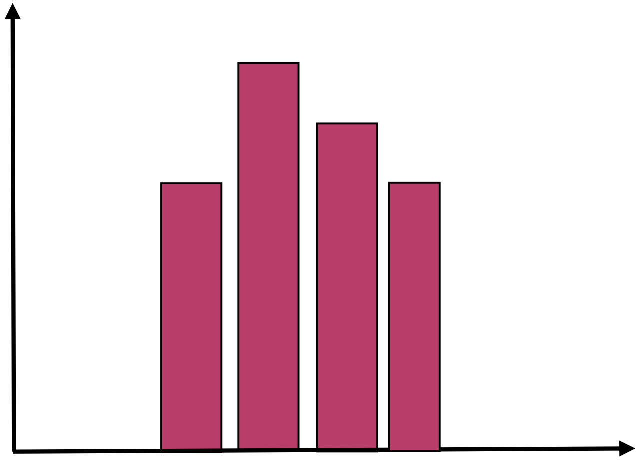


2

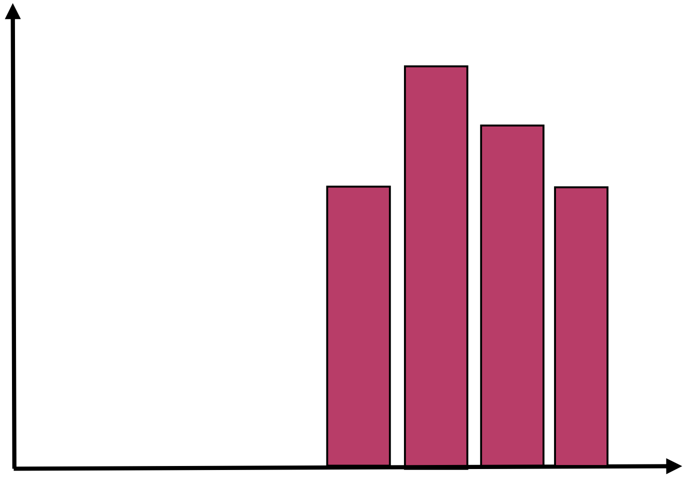
3



A

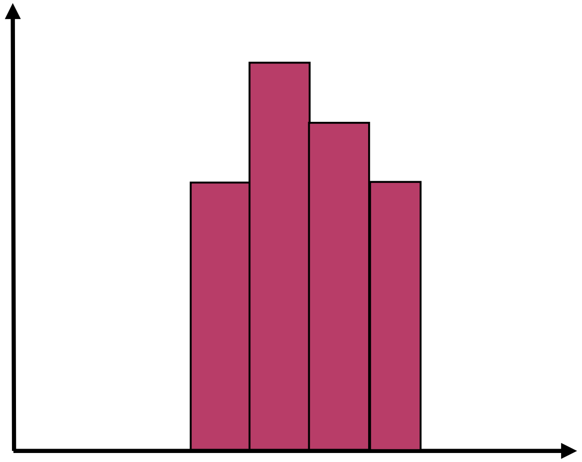


B

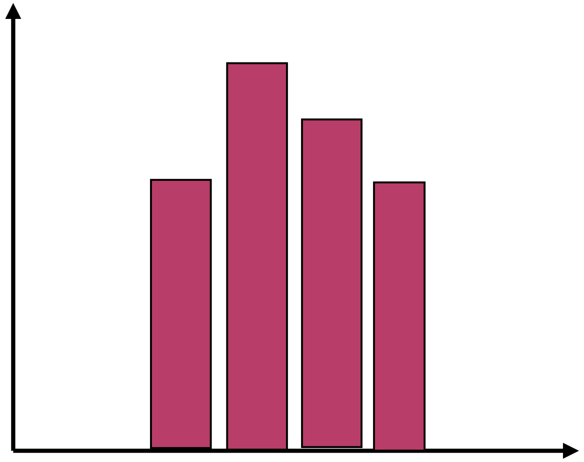


C

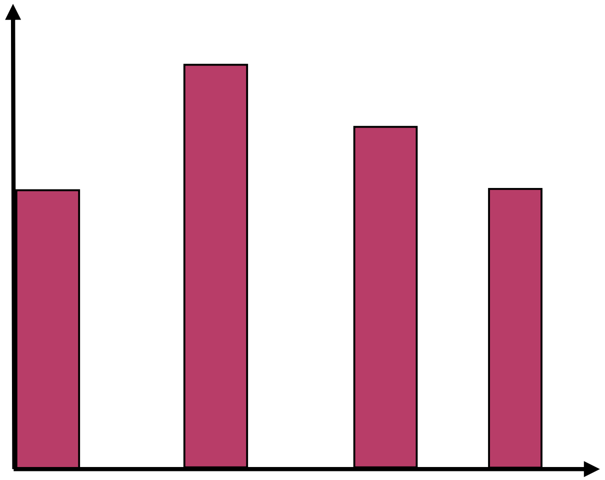
- What is the difference between the images that have these histogram characteristics?



A

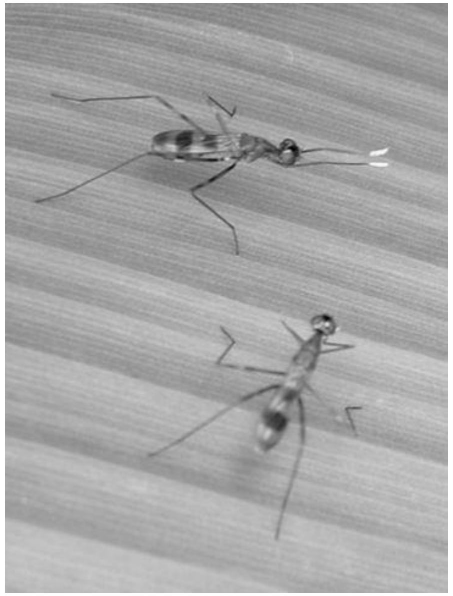


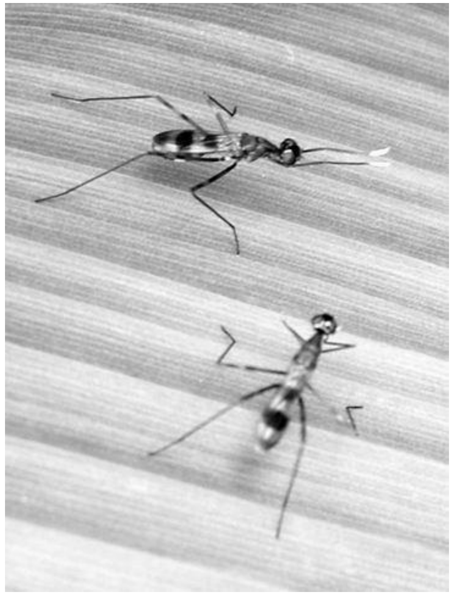
B

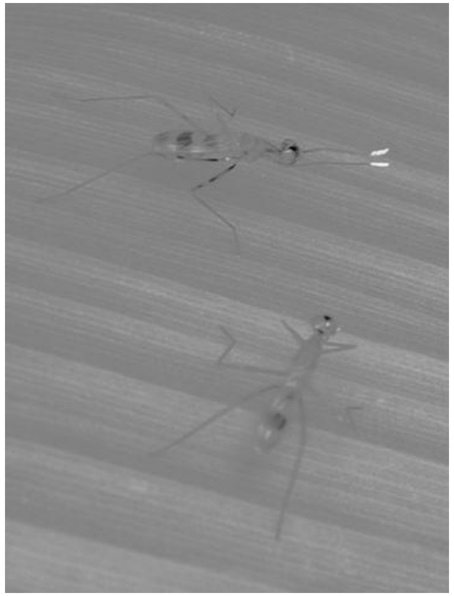


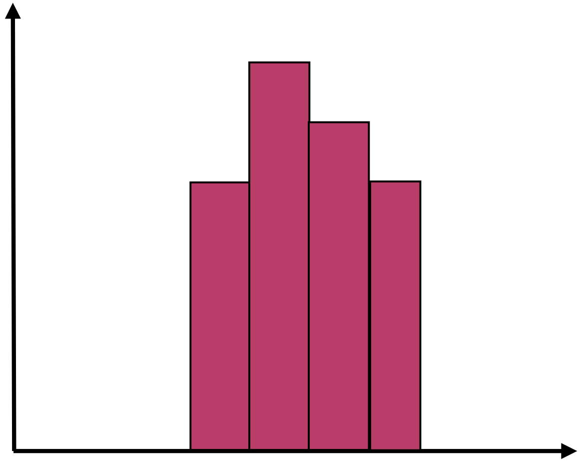
C

Try to match the pictures and histograms

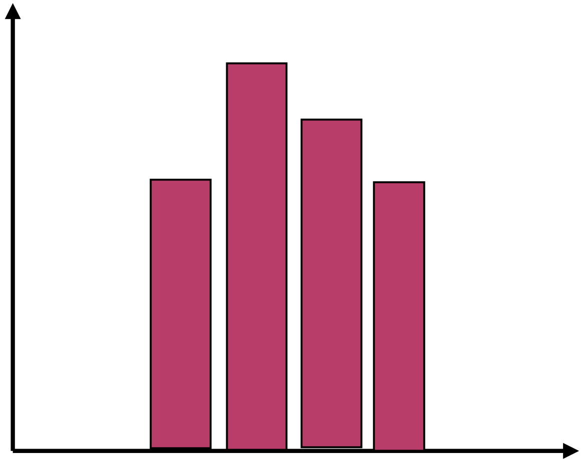




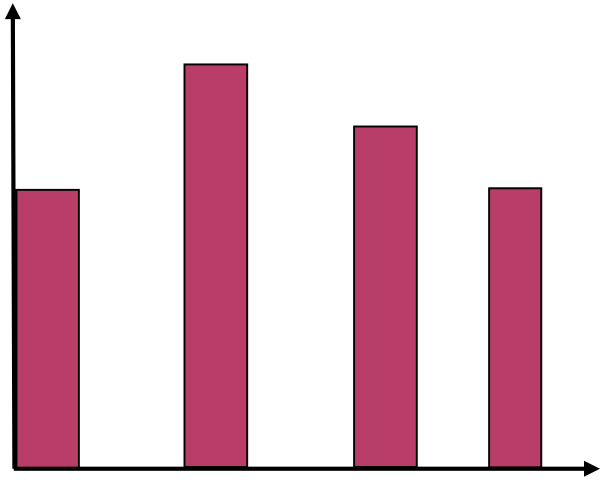




A



B



C

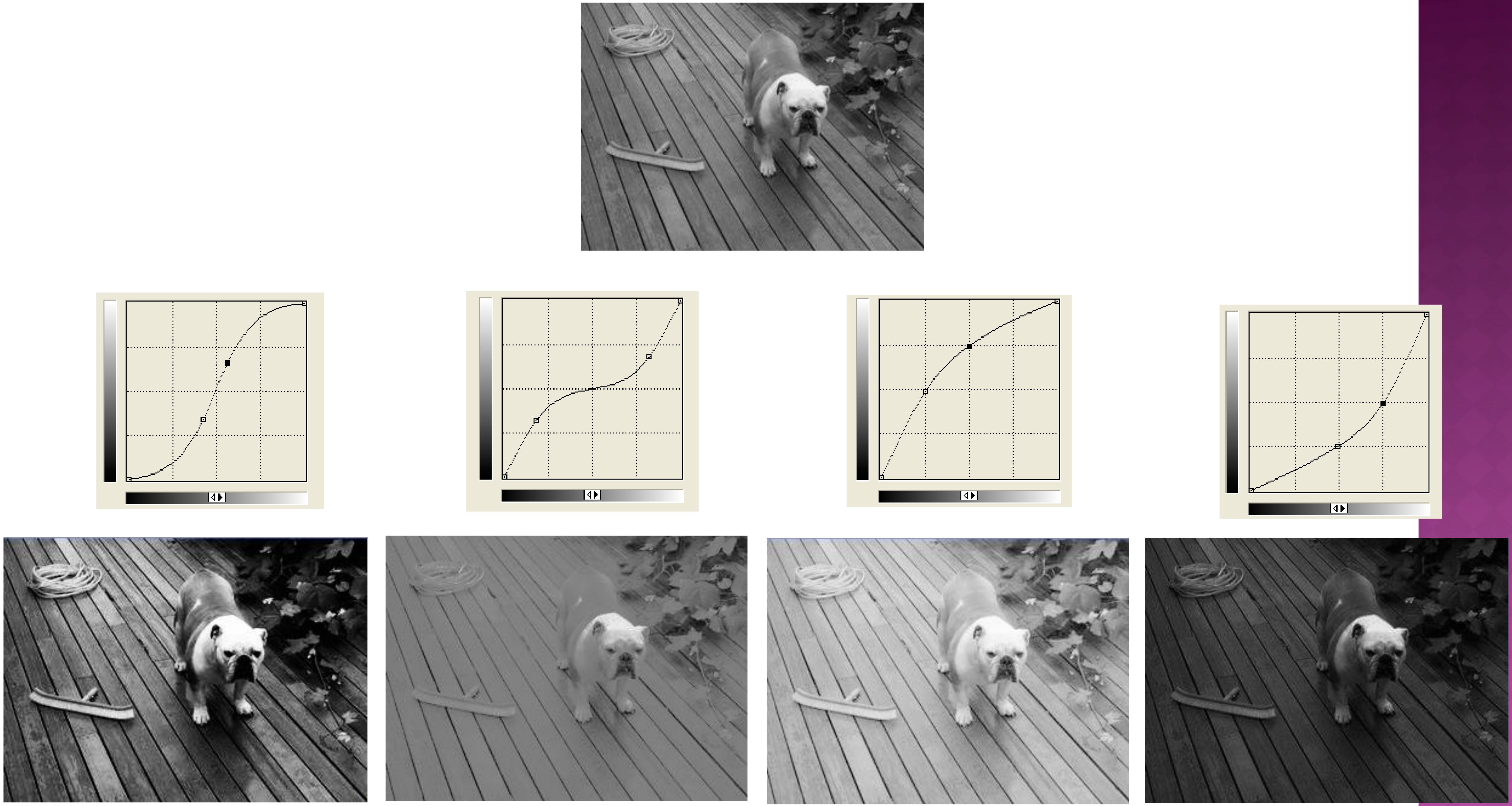
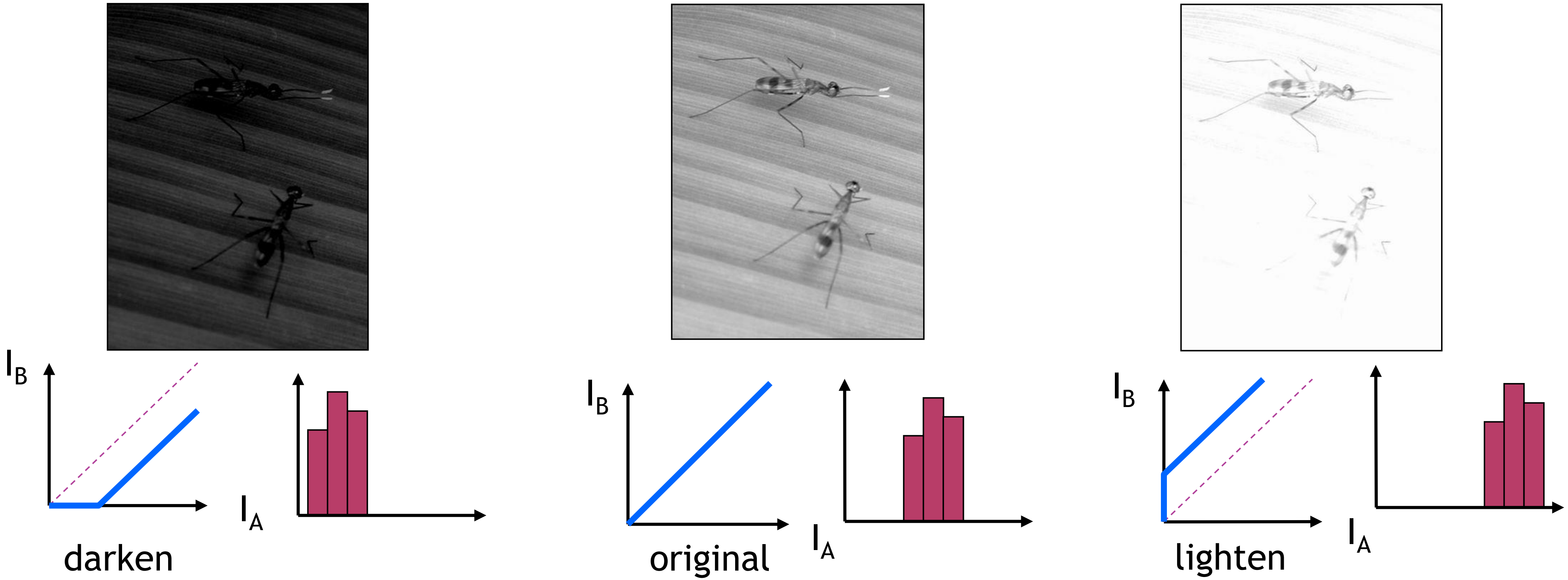


Image variation will be increased in the intensity range where the slope of the function f(x) is greater than 1

GLOBAL ALTERATION IN BRIGHTNESS

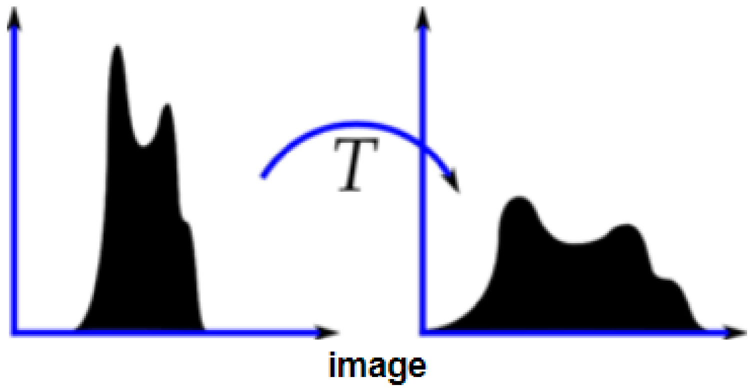
- Use to lighten or darken an image by add or subtract a constant to all the image pixels.

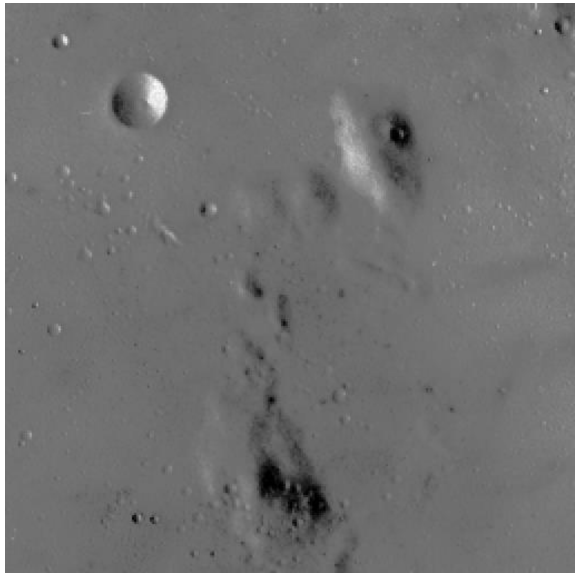
$$B[i,j] = A[i,j] + C$$

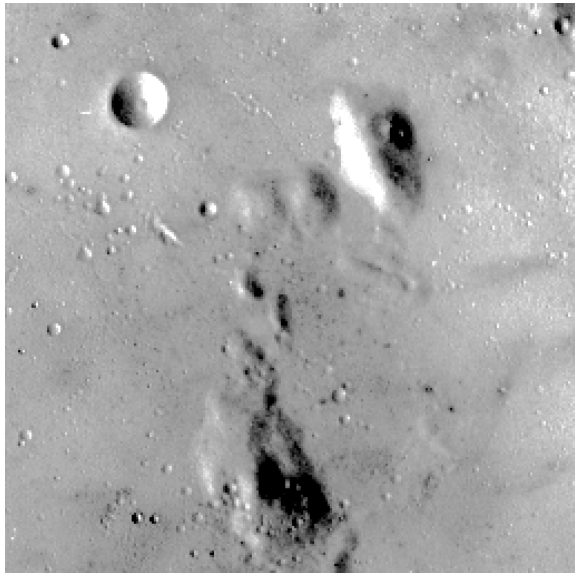


CONTRAST STRETCHING

- improve the contrast in an image by stretching the range of intensity values it contains to span a desired range of values
- can be divided into Linear and Non-Linear
  - The linear method uses Piecewise Linear functions
  - Non-linear method uses Non-Linear transformation functions :**Histogram Equalization**, **Gaussian Stretch** etc.

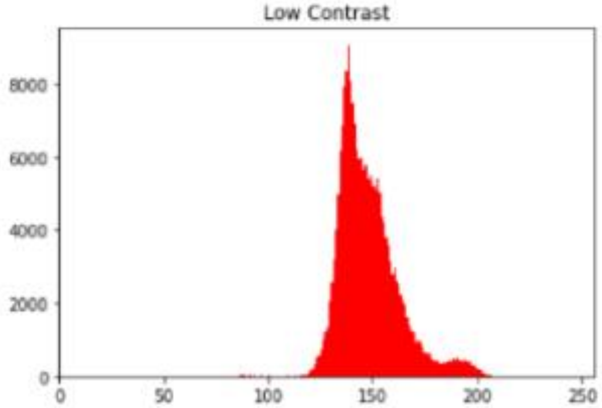


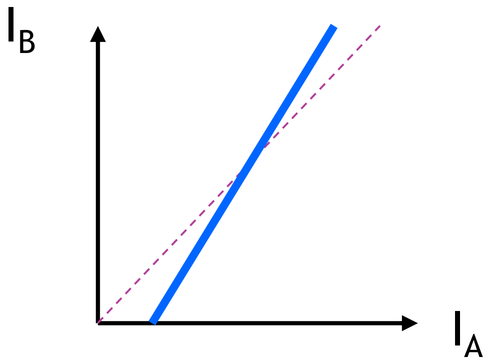


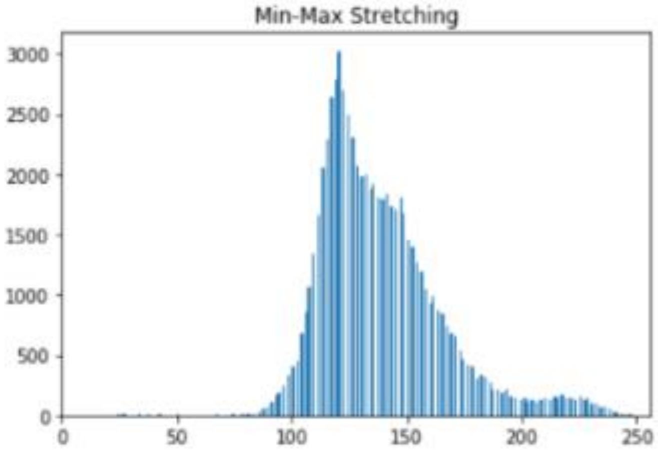


Min-Max Stretching

- Lower and upper values of the input image are made to span the full dynamic range



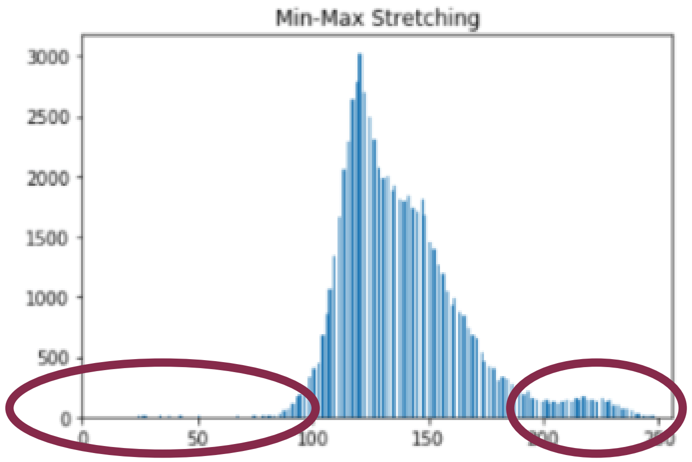


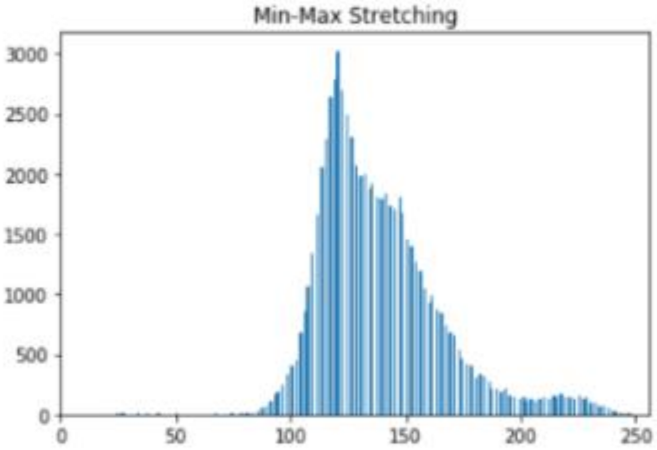


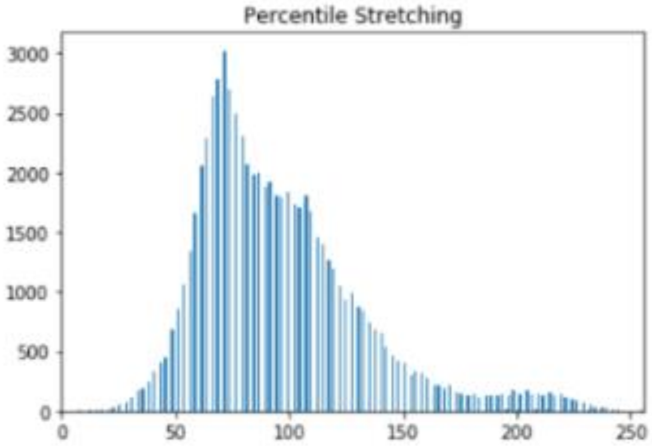
$$Xnew = \frac{Xinput - Xmin}{Xmax - Xmin} \times 255$$

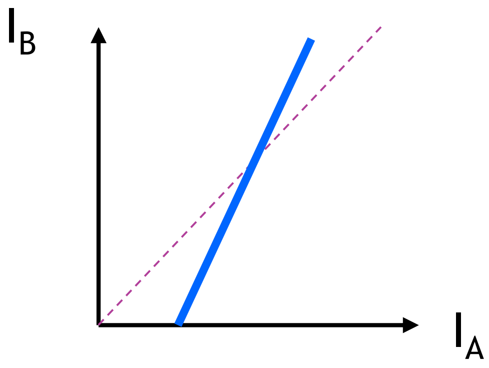
Percentile Stretching

- Sometimes, when Min-Max is performed, the tail ends of the histogram becomes long resulting in no improvement in the image quality.
- Clip a certain percentage like 1%, 2% of the data from the tail ends of the input image histogram.







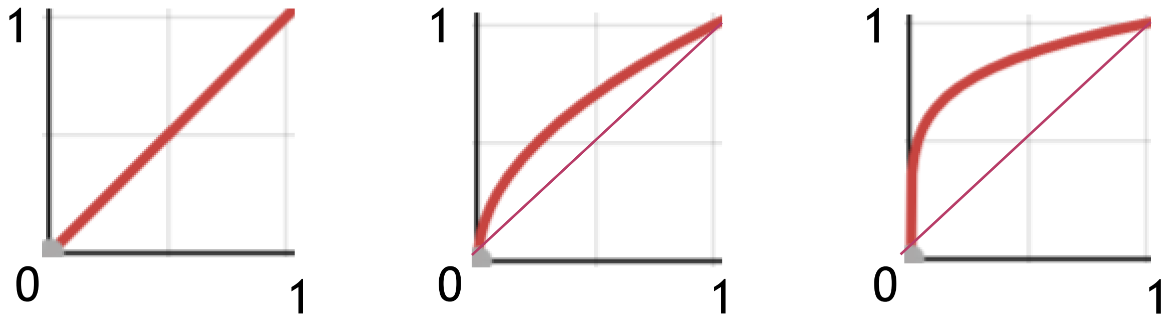


The formulae is same as Min-Max but now the X_{max} and X_{min} are the clipped values.

GAMMA CORRECTION

- human perceive *double the amount of light* as only a *fraction* brighter (a non-linear relationship)!
- Eyes are also much more sensitive to changes in dark tones than brighter tones
- Stretch the gray values of an image that is too dark on to the full set of gray values available.
- Gamma Correction

$$f(x) = x^{\frac{1}{\gamma}}$$



$\gamma =$ 1 2 5

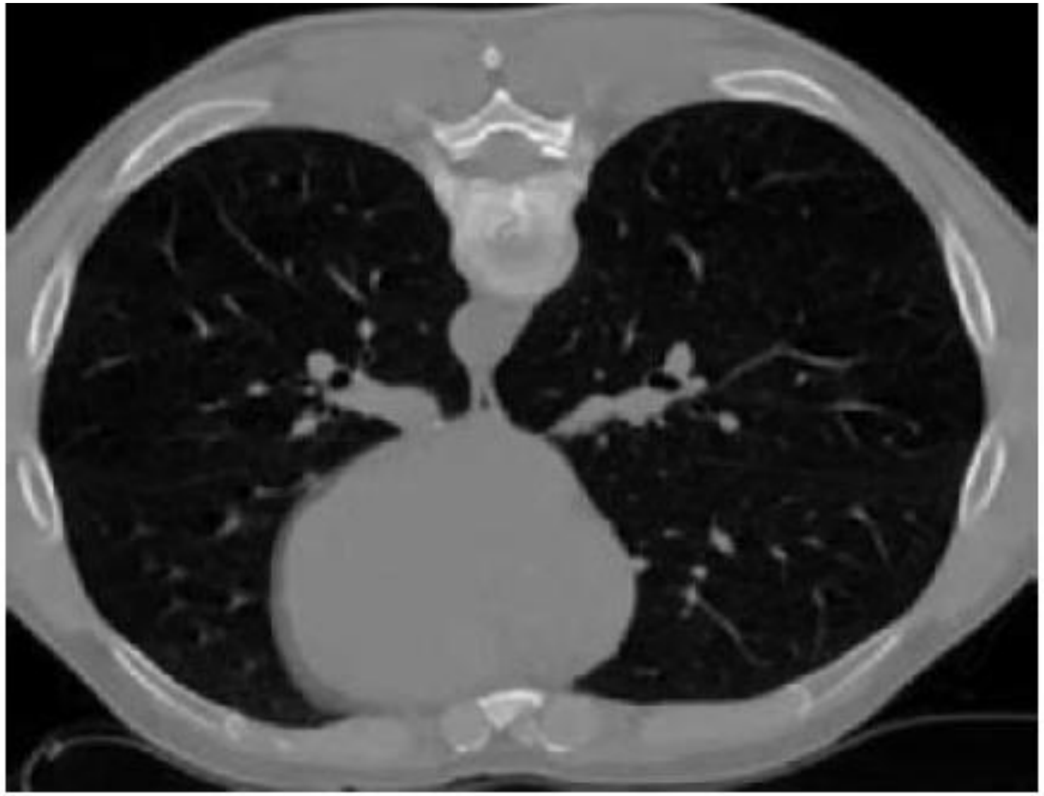
$\gamma =$ 1 1.5 2

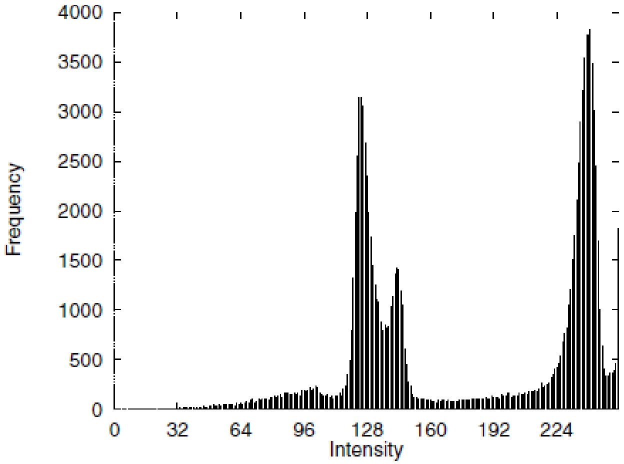
HISTOGRAM EQUALIZATION

- The output image should use all available gray level.
- The output image has approximately the same number of pixels of each gray level.
- Finding function F(g) that will enhance the general contrast in image, spreading the distribution of gray levels wider and more evenly, ideally to an equal number of pixel per gray level.

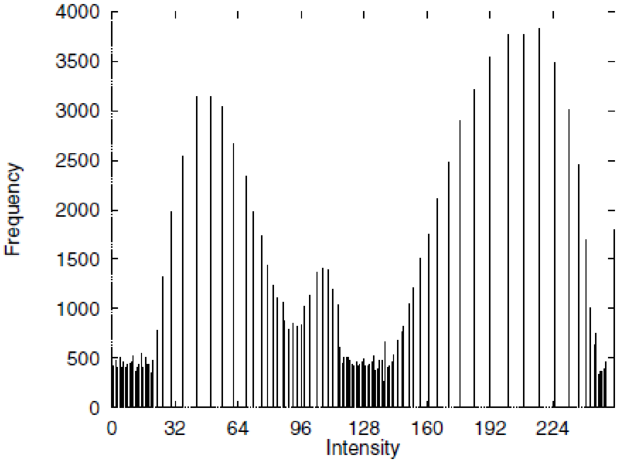
$$F(g) = \max\{0, round\left(\frac{g\_levels * \sum_{j=1}^{g} n_j}{N}\right) - 1\}$$

F(g) = new gray level
g-level = number of all gray levels
n_j = number of pixel at j level
N = number of all pixel





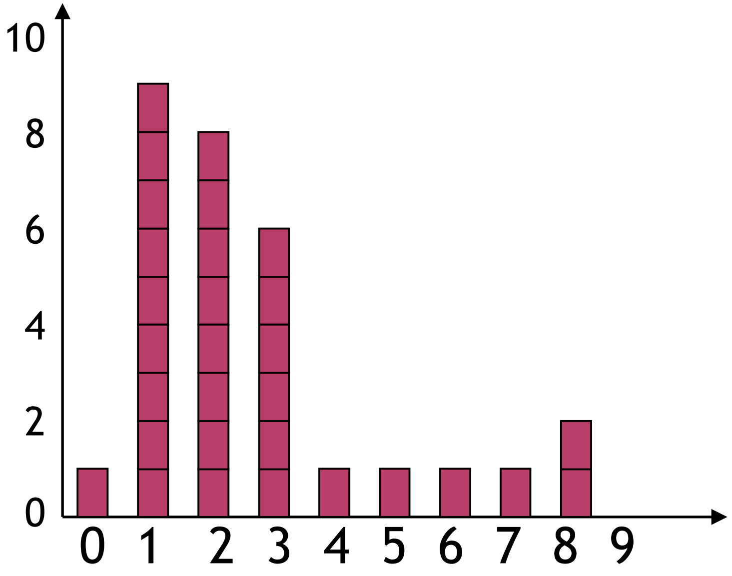
Frequency
Intensity
0
500
1000
1500
2000
2500
3000
3500
4000
0
32
64
96
128
160
192
224



Frequency
Intensity
0
500
1000
1500
2000
2500
3000
3500
4000
0
32
64
96
128
160
192
224

k	n_k	$\sum n_k$	$\frac{\sum n_k}{N}$	$K*\frac{\sum n_k}{N}$	$round\left(K*\frac{\sum n_k}{N}\right)-1$	$F(g)$
0	1					
1	9					
2	8					
3	6					
4	1					
5	1					
6	1					
7	1					
8	2					
9	0					

0	8	8	3	3	3
1	7	4	2	2	3
1	1	5	2	2	3
1	1	6	2	2	3
1	1	1	1	2	2



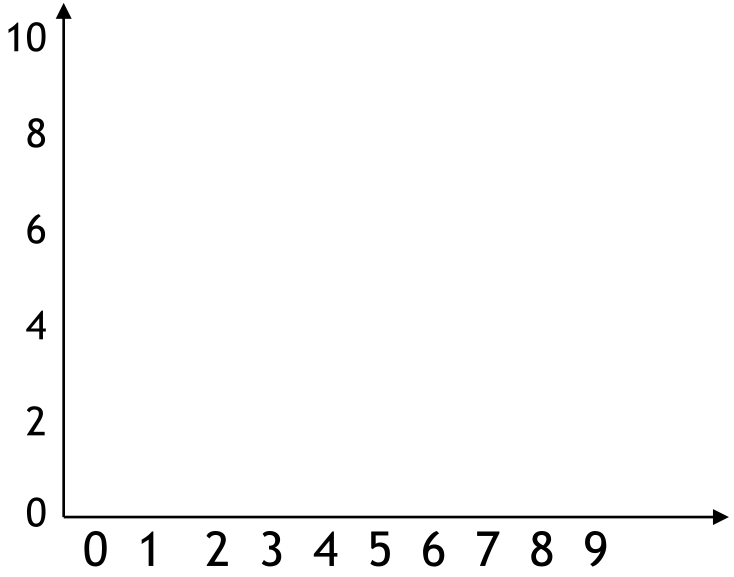


IMAGE SMOOTHING

IMAGE SMOOTHING

- Smoothing aim to reduce noise or the small fluctuation in image
- Noise is the high frequency in Fourier transform
- Smoothing also blurs the sharp edges which the important information in image

Before

After

MEAN FILTERING

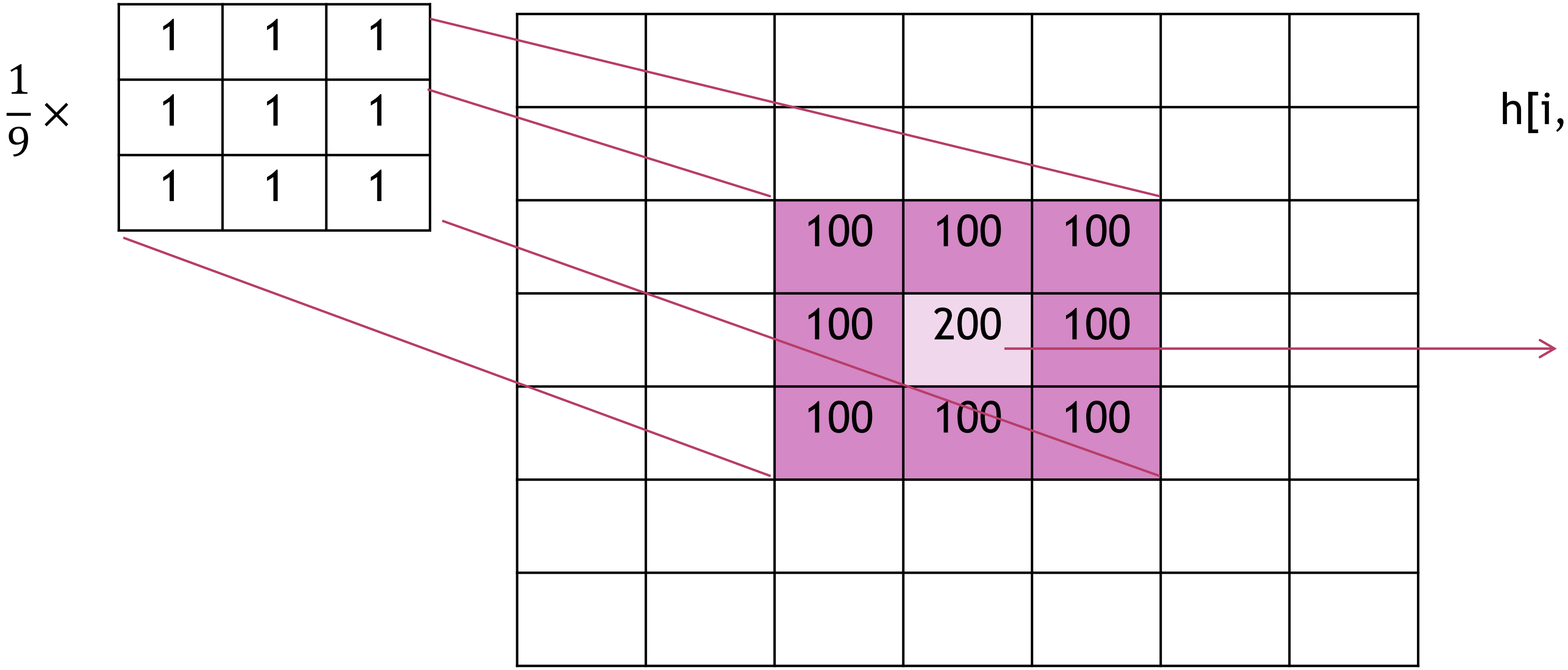
- Mean filtering is a local averaging operation and it is a one of the simplest linear filter
- The value of each pixel is replaced by the averaging a local neighborhood values of input image pixel

$$h[i,j] = \frac{1}{M} \sum_{(k,l) \in N} f[k,l]$$

N – neighborhood pixels if pixel f(i,j)
M – total number of pixels in the neighborhood N

For 3x3 neighborhood

$$h[i,j] = \frac{1}{9} \sum_{k=i-1}^{i+1} \sum_{l=i-1}^{j+1} f[k,l]$$



h[i,j]=(100+100+100
+ 100+200+100
+ 100+100+100)/9
= 111

Original Image

Mean 3X3

Mean 5X5

Mean 7x7

Original Image

Mean 3X3

Mean 5X5

Size of neighborhood N controls the amount of filtering
Larger neighborhood is larger mask will result a greater degree of filtering

0	0	0	0	0	0	0	0
0	200	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	200	200	200	200	0
0	0	0	200	200	200	200	0
0	0	100	200	200	200	200	0
0	0	0	200	200	200	200	0
0	0	0	0	0	0	0	0

What happen to the noise and edge of object?

Smoothing with mean 3x3 neighbor

Smoothing with mean 5x5 neighbor

MEDIAN FILTERING

- Replace a pixel value with the median of the gray value in the local neighborhood
- Effective in removing salt and pepper and impulse noise while retaining image detail

- Algorithm
  - Sort the pixels in to ascending order by gray level
  - Select the value of the middle pixel as the new value for pixel[i,j]

Note : if the number of pixels is even, the median is taken as the average of the middle two pixels after sorting h[i,j]=$median\ of\ gray\ level\ from\ f[i,j]\ neighbor$

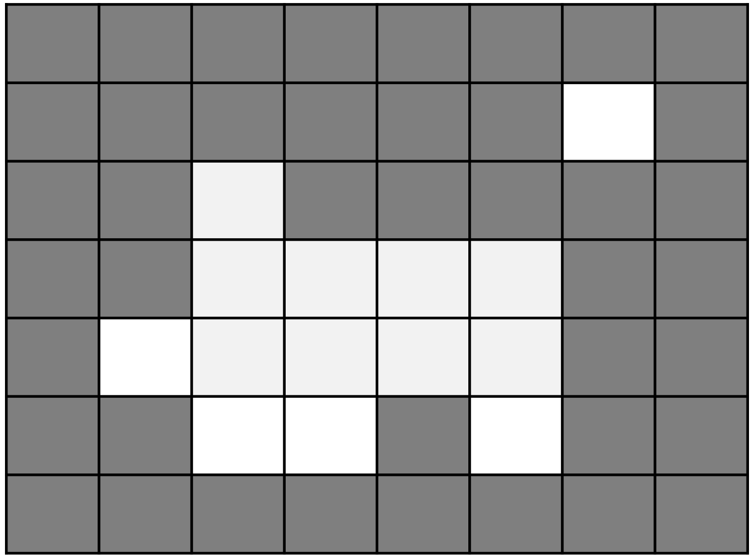
For 3x3 neighborhood

		100	100	100		
		110	200	110		
		120	120	120		

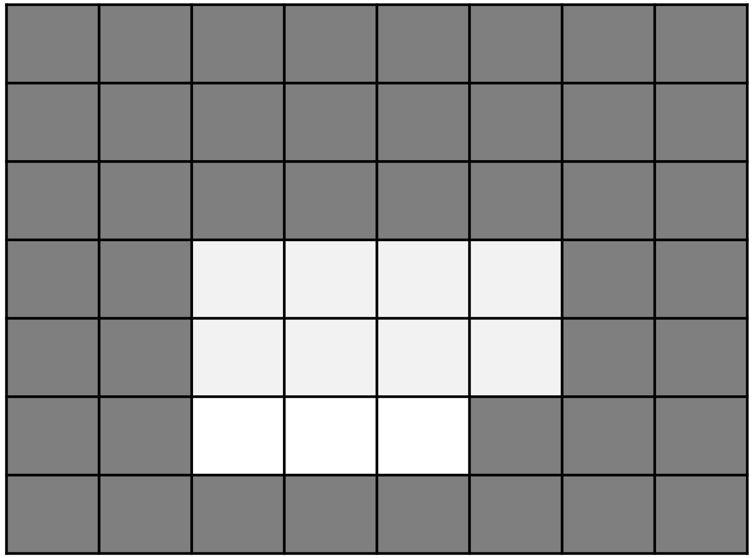
Median of
100 100 100 110 110 120 120 120 200

h[i,j]=110

Original Image

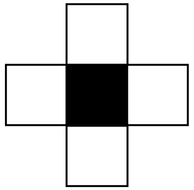


Median 3X3

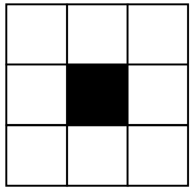


Median filtering can also remove salt and pepper noise and most other small artifacts

Pepper Noise

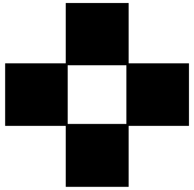


4-neighbor

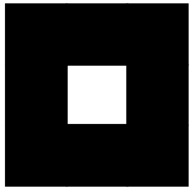


8-neighbor

Salt Noise



4-neighbor



8-neighbor

Original Image

Median 3X3

Median 5X5

Median 7x7

original

Median 3x3

original

Median 3x3

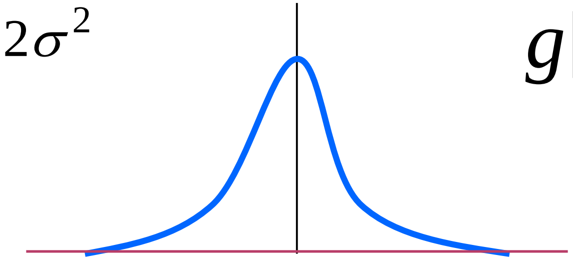
Median 5x5

GAUSSIAN FILTERING

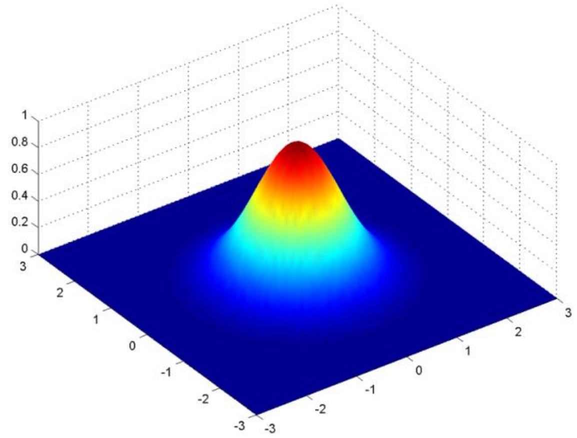
- Gaussian filter is filter with the weighs chosen according to the shape of a Gaussian function
- Good filter for removing noise draw from normal distribution
- 2D Gaussian functions are rotational symmetric, smoothing will be the same in all directions.
- weigh given to the neighbor decrease with distance from the central pixel

σ = Standard deviation

$$g(x) = ce^{\frac{-x^2}{2\sigma^2}}$$



$$g[i, j] = ce^{\frac{-(i^2+j^2)}{2\sigma^2}}$$



The mask weight of Gaussian filters can compute directly from the discrete Gaussian distribution

Calculate Gaussian Mask 7x7 with variance $\sigma^2 = 2$

$$g[i, j] = c e^{\frac{-(i^2+j^2)}{2\sigma^2}}$$

[i,j]	-3	-2	-1	0	1	2	3
-3	0.011	0.039	0.082	0.105	0.082	0.039	0.011
-2	0.039	0.135	0.287	0.368	0.287	0.135	0.039
-1	0.082	0.287	0.607	0.779	0.607	0.287	0.082
0	0.105	0.368	0.779	1.000	0.779	0.368	0.105
1	0.082	0.287	0.607	0.779	0.607	0.287	0.082
2	0.039	0.135	0.287	0.368	0.287	0.135	0.039
3	0.011	0.039	0.082	0.105	0.082	0.039	0.011

Filter weight become integer and corner array equal to 1

$$0.011 \times c = 1$$

$$c = \frac{1}{0.11} = 91$$

[i,j]	-3	-2	-1	0	1	2	3
-3	1	4	7	10	7	4	1
-2	4	12	26	33	26	12	4
-1	7	26	55	71	55	26	7
0	10	33	71	91	71	33	10
1	7	26	55	71	55	26	7
2	4	12	26	33	26	12	4
3	1	4	7	10	7	4	1

Gaussian Mask 7x7

$$\sum_{i=-3}^{3} \sum_{j=-3}^{3} g[i, j] = 1115$$

$$h[i, j] = \frac{1}{1115}\left(f[i, j] * g[i, j]\right)$$

$*$ is convolution operation

The mask weight of Gaussian filters can compute directly from the discrete Gaussian distribution

Calculate Gaussian Mask 5x5 with variance $\sigma^2 = 2$

$$g[i, j] = c e^{\frac{-(i^2 + j^2)}{2\sigma^2}}$$

$\sigma = 1$

1	12	55	90	55	12	1
12	148	665	1097	665	148	12
55	665	2981	4915	2981	665	55
90	1097	4915	8103	4915	1097	90
55	665	2981	4915	2981	665	55
12	148	665	1097	665	148	12
1	12	55	90	55	12	1

$\sigma = 2$

1	2	3	3	3	2	1
2	4	5	6	5	4	2
3	5	8	9	8	5	3
3	6	9	10	9	6	3
3	5	8	9	8	5	3
2	4	5	6	5	4	2
1	2	3	3	3	2	1

$\sigma = 3$

1	1	2	2	2	1	1
1	2	2	2	2	2	1
2	2	3	3	3	2	2
2	2	3	3	3	2	2
2	2	3	3	3	2	2
1	2	2	2	2	2	1
1	1	2	2	2	1	1

$\sigma = 4$

1	1	1	2	1	1	1
1	2	2	2	2	2	1
1	2	2	2	2	2	1
2	2	2	2	2	2	2
1	2	2	2	2	2	1
1	2	2	2	2	2	1
1	1	1	2	1	1	1

Original

Gaussian 3x3

Gaussian 5x5

Gaussian 7x7

Original Image

Mean 7x7

Median 7x7

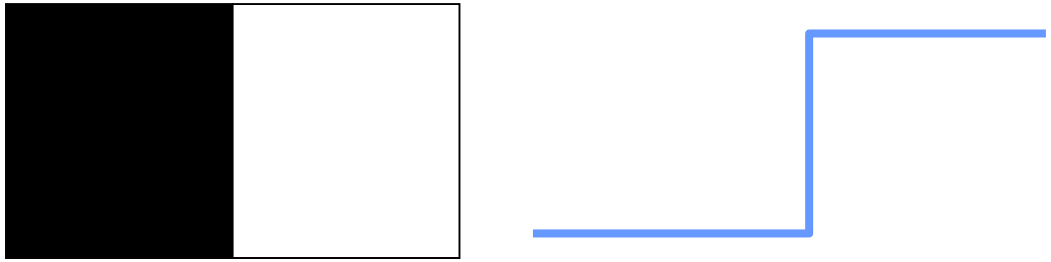
Gaussian 7x7

Compare Between Mean – Median – Gaussian filtering

EDGE DETECTION

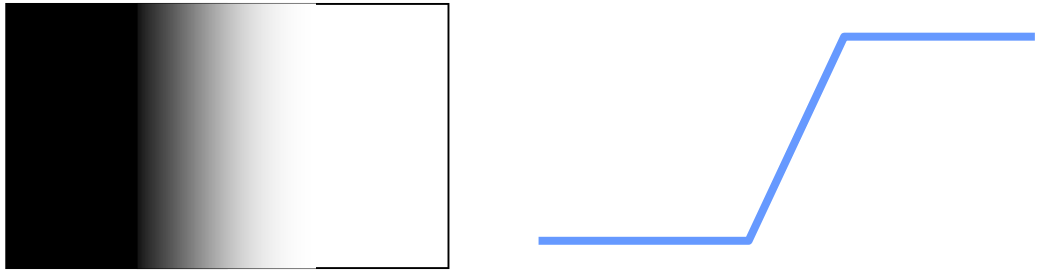
EDGE

- Edges are significant local intensity changes in the image.
- Edges typically occur on the boundary between two different regions in image.

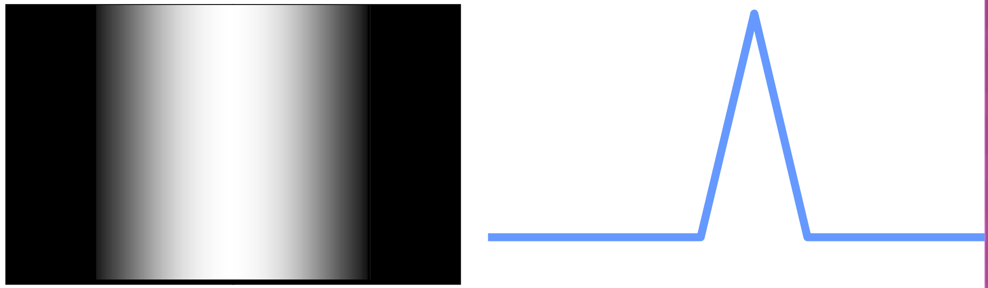


step

line



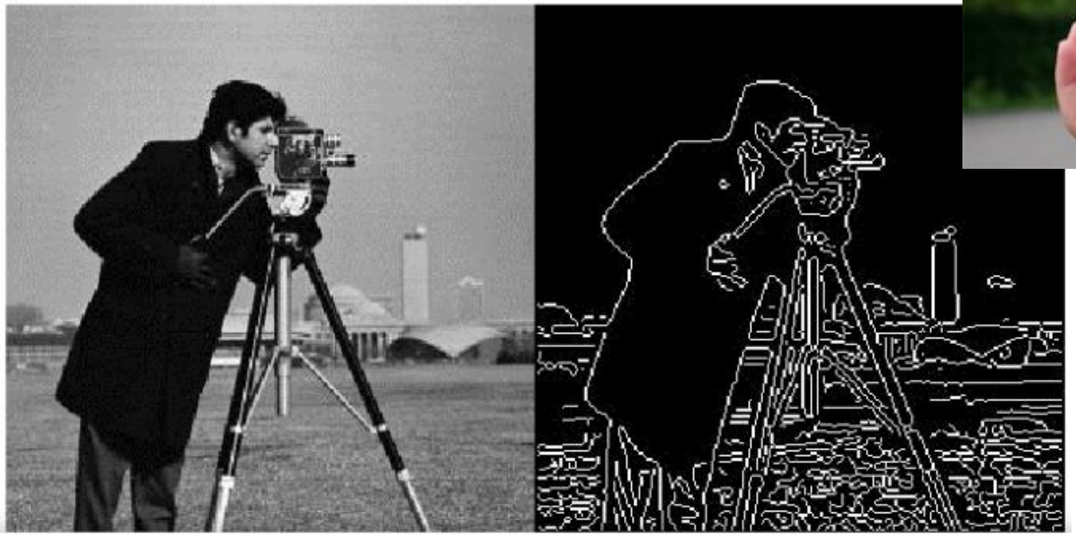
ramp



roof

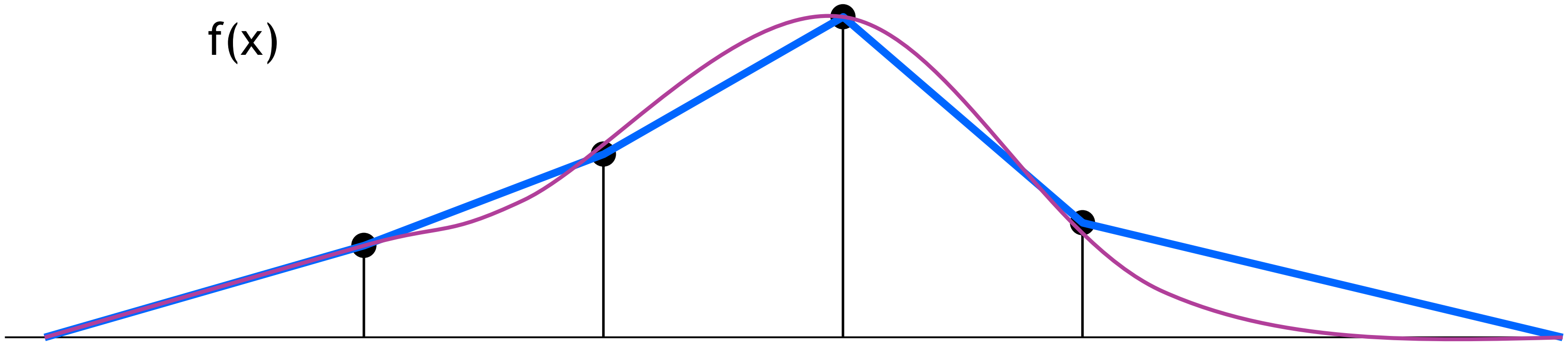
EDGE DETECTION

- Produce a set of edge positions from image
- Important features can be extracted from the edges



DIFFERENCING MASKS

- Derivative of 1D signal



Original $f =$

f[i-1]	f[i]	f[i+1]	f[i+2]

First derivative $f' =$

f[i]-f[i-1]	f[i+1]-f[i]	f[i+2]-f[i+1]

M'

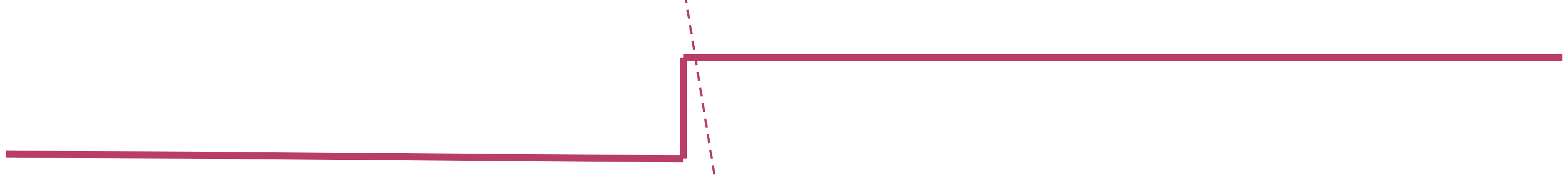
-1	1

Second derivative $f'' =$

f[i+1]-2f[i] +f[i-1]	f[i+2]-2f[i+1]+f[i]

M''

1	-2	1



Step edge

20	20	20	20	200	200	200	200	200

-1	**1**

First derivative

0	0	0	180	0	0	0	0	0

Step edge

200	200	200	200	20	20	20	20	20

-1	**1**

First derivative

-1	1

- This mask gives a response on perfect step edges.
- Use absolute value after applying the mask then the first derivative mask can be righter

-1	1

1	-1

- However, the difference occur at x+0.5

-1	1

- Apply M' to 3 coordinates and centered at signal point M[i] so that it computes the signal difference across the adjacent value

-1	0	1

The 1st derivative

Mask M’ = [-1, 0, 1]

Upward step

20	20	20	20	20	200	200	200	200	200

Downward step

200	200	200	200	200	20	20	20	20	20

ramp

20	20	20	20	80	140	200	200	200	200

line

20	20	20	20	200	200	20	20	20	20

The 2nd derivative

Mask M = [1, -2, 1]

Upward step

20	20	20	20	20	200	200	200	200	200

Downward step

200	200	200	200	200	20	20	20	20	20

ramp

20	20	20	20	80	140	200	200	200	200

line

20	20	20	20	200	200	20	20	20	20

GRADIENT

- The Gradient is the two-dimensional equivalent of the first derivative, defined as a vector

$$G[f(x,y)] = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

- Magnitude of Gradient

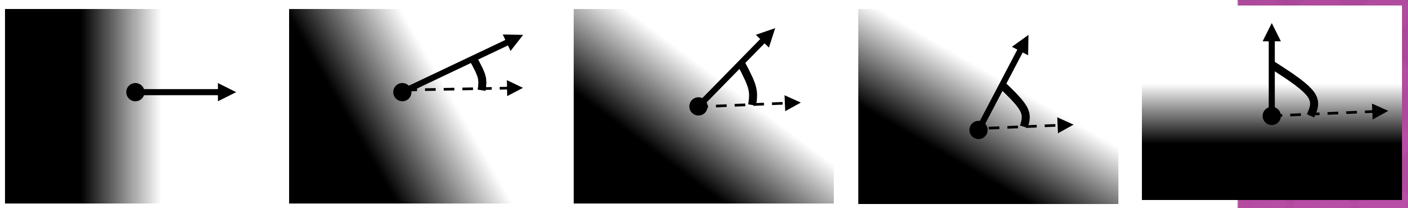
$$|G[f(x,y)]| = \sqrt{G_x^2 + G_y^2}$$

$$\approx |G_x| + |G_y|$$

$$\approx \max(|G_x|, |G_y|)$$

- Direction of Gradient

$$\theta_{(x,y)} = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$



- For digital images

$$G_x \cong f[i, j+1] - f[i, j]$$

$$G_y \cong f[i+1, j] - f[i, j]$$

$$G_x = \begin{array}{|c|c|}\hline -1 & 1 \\ \hline\end{array}$$

$$G_y = \begin{array}{|c|}\hline 1 \\ \hline -1 \\ \hline\end{array}$$

Expand to 3 coordinates and centered at the signal point

$$G_x = \frac{1}{2} \times \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline\end{array}$$

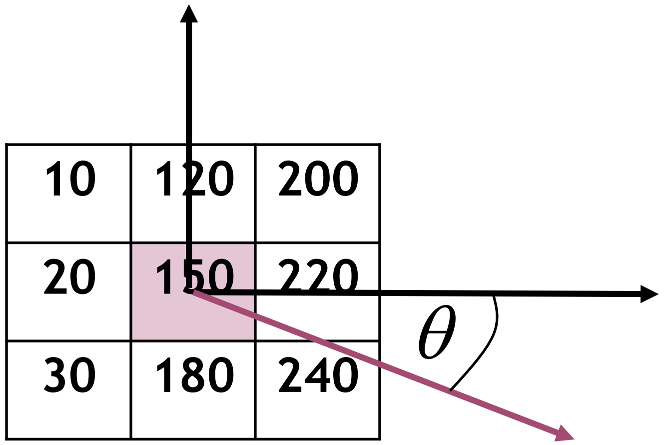
$$G_y = \frac{1}{2} \times \begin{array}{|c|}\hline 1 \\ \hline 0 \\ \hline -1 \\ \hline\end{array}$$

Reduce the noise of pixel edge by averaging 3 different estimates of the contrast in the neighborhood of [x,y]

$$G_x = \frac{1}{2} \times \frac{1}{3} \times \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline\end{array}$$

$$G_y = \frac{1}{2} \times \frac{1}{3} \times \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -1 & -1 \\ \hline\end{array}$$

often the division by 6 is ignored to save computation time



$$G_x = \frac{1}{2} \times \frac{1}{3} \times (-10 - 20 - 30 + 200 + 220 + 240)$$
$$= \frac{1}{2} \times \frac{1}{3} \times 600 = 100$$

$$G_y = \frac{1}{2} \times \frac{1}{3} \times (-30 - 180 - 240 + 10 + 120 + 200)$$
$$= \frac{1}{2} \times \frac{1}{3} \times -120 = -20$$

-1	0	1
-1	0	1
-1	0	1

$$\theta_{(x,y)} = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

$= \tan^{-1}\left(\frac{-20}{100}\right) = -0.197$ rad = -11.3 degree

1	1	1
0	0	0
-1	-1	-1

$$|G[f(x,y)]| = \sqrt{100^2 + (-20)^2}$$

=101.98

PREWITT EDGE DETECTION

$G_x =$

-1	0	1
-1	0	1
-1	0	1

$G_y =$

1	1	1
0	0	0
-1	-1	-1

$$|G[f(x,y)]| = \sqrt{G_x^2 + G_y^2}$$

Input image

G_x

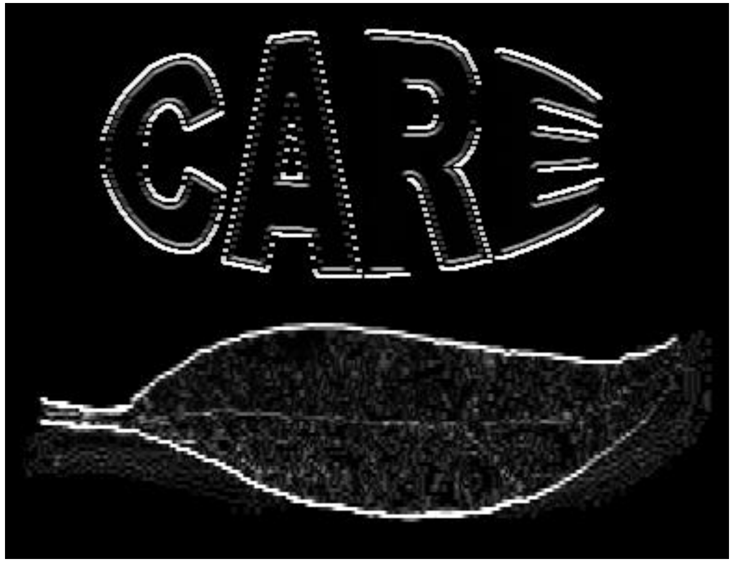
G_y

$\sqrt{G_x^2 + G_y^2}$

Input image

$\sqrt{G_x^2 + G_y^2}$

G_x



G_y

SOBEL EDGE DETECTION

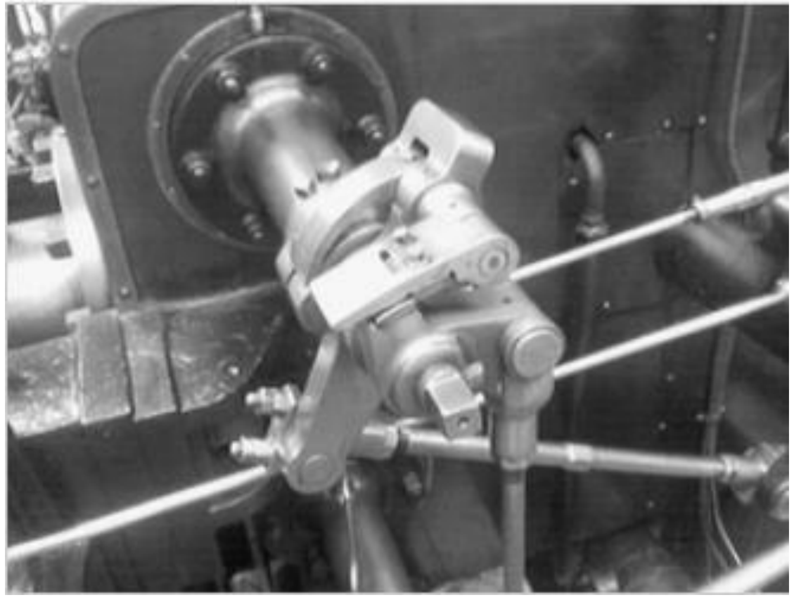
$G_x =$

-1	0	1
-2	0	2
-1	0	1

$G_y =$

1	2	1
0	0	0
-1	-2	-1

$$|G[f(x,y)]| = \sqrt{G_x^2 + G_y^2}$$



Input image

Gradient magnitude

ROBERTS CROSS EDGE DETECTION

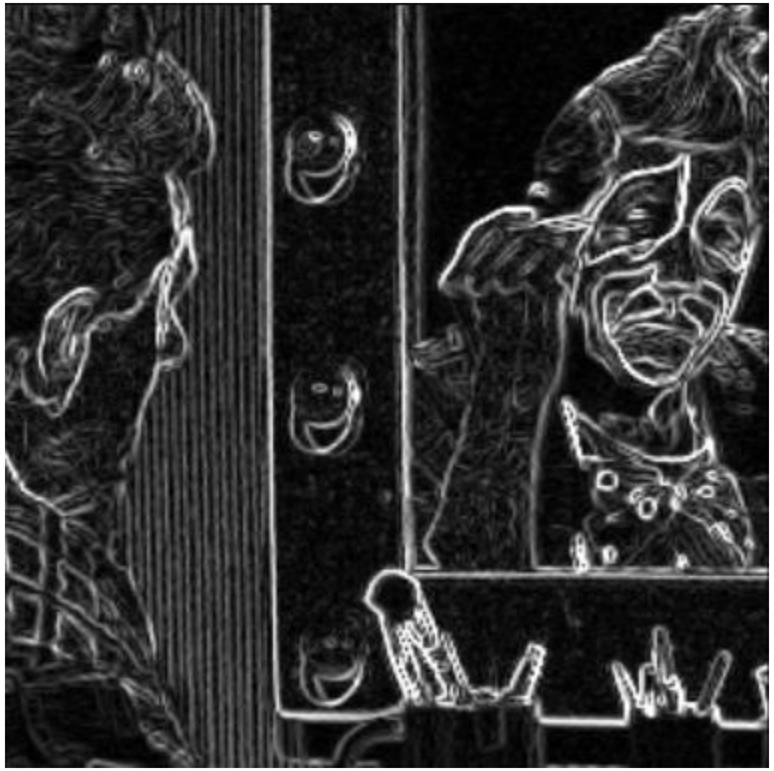
$G_x =$

1	0
0	-1

$G_y =$

0	1
-1	0

$$|G[f(x,y)]| = |G_x| + |G_y|$$



Compare the edge detection of the 3 methods

Original image

Robert operator

1	0
0	-1

0	1
-1	0

-1	0	1
-1	0	1
-1	0	1

1	1	1
0	0	0
-1	-1	-1

Prewitt operator

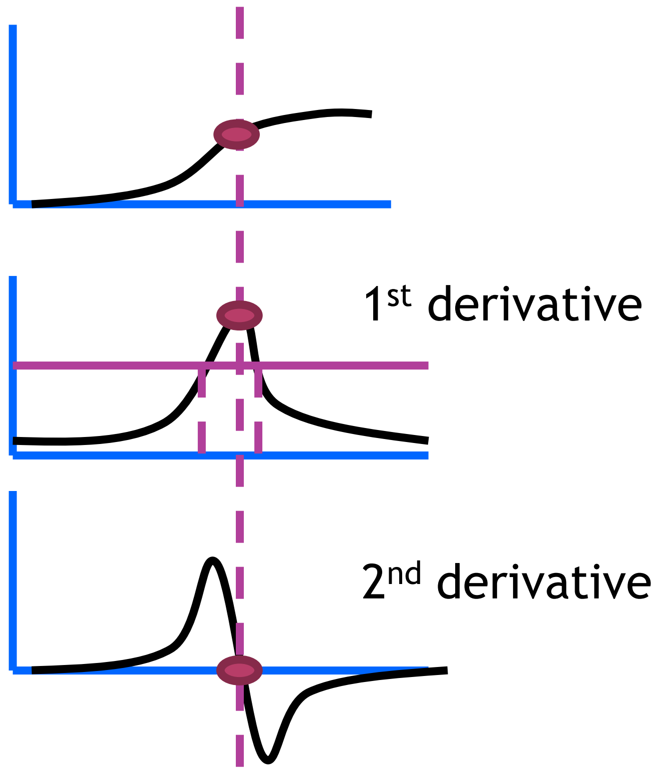
Sobel operator

-1	0	1
-2	0	2
-1	0	1

1	2	1
0	0	0
-1	-2	-1

LAPLACIAN OPERATOR

- The first derivative edge detection may detect too many edge points.
- A better approach would be to find only the points with local maxima in gradient values and consider them edge points.
- Edge is the zero crossing position in the second derivative



- An approximate to the Laplacian

$$\nabla^2 f = \frac{\partial^2 f}{\partial^2 x} + \frac{\partial^2 f}{\partial^2 y}$$

$$\frac{\partial^2 f}{\partial^2 x} = f[i,j+1]-2f[i,j]+f[i,j-1]$$

$$\frac{\partial^2 f}{\partial^2 y} = f[i+1,j]-2f[i,j]+f[i-1,j]$$

$$= \begin{bmatrix} 1 & -2 & 1 \end{bmatrix} + \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$$

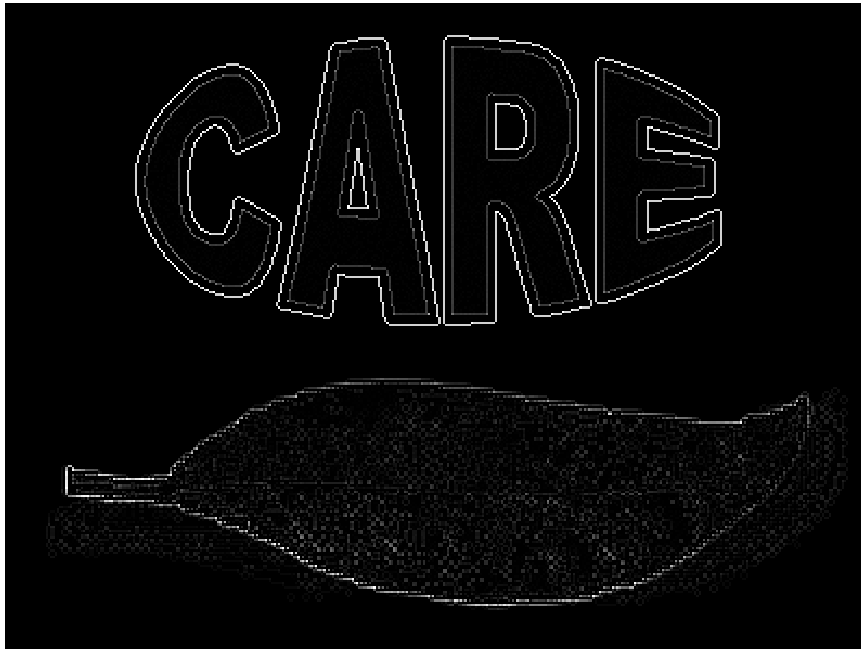
$$= \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

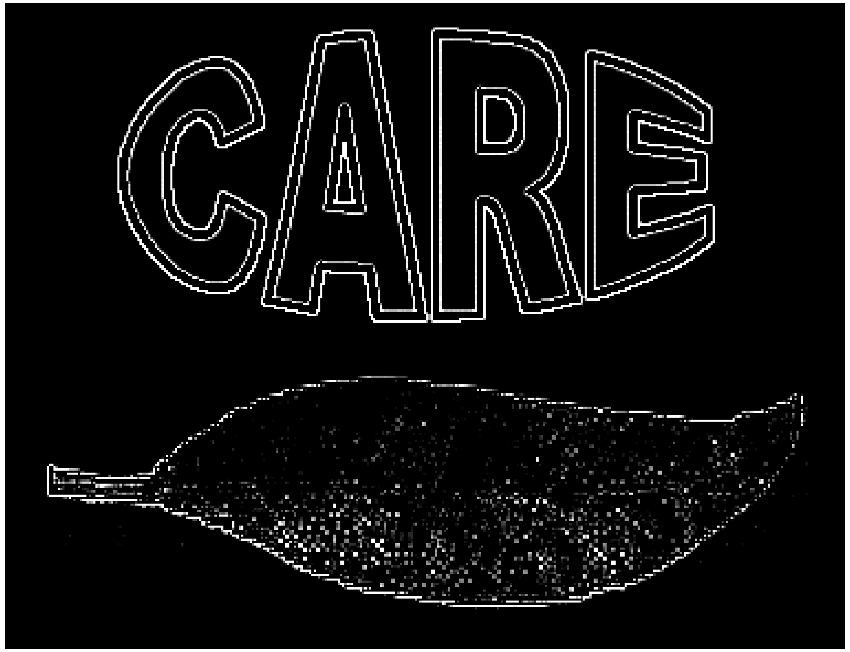
- Sometimes it is desired to give more weight to the center pixels

$$\approx \begin{bmatrix} 1 & 4 & 1 \\ 4 & -20 & 4 \\ 1 & 4 & 1 \end{bmatrix}$$

0	1	0
1	-4	1
0	1	0

1	4	1
4	-20	4
1	4	1

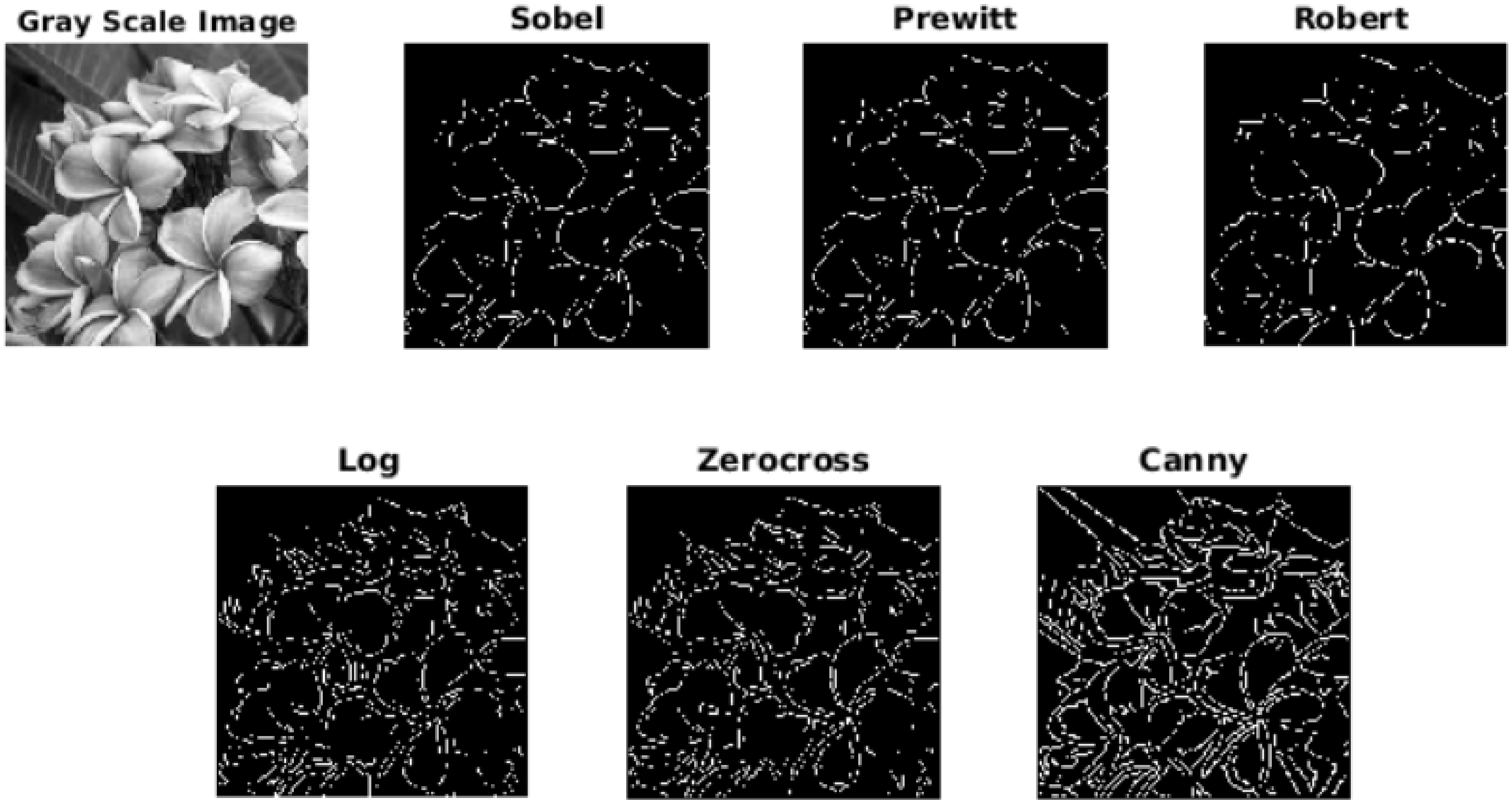




LAPLACIAN OF GAUSSIAN (LOG)

- Edge found by zero crossing of the second derivative are very sensitive to noise
- Laplacian of Gaussian combines the Gaussian filter with the Laplacian for edge detection to filter out the noise before edge enhancement

- Thresholding is to pass the large peak in the first derivative


Gray Scale Image
Sobel
Prewitt
Robert
Log
Zerocross
Canny

PROPERTY OF MASK

- smoothing mask
  - Coordinates are positive
  - **Sum to 1**
  - The amount of smoothing and noise reduction depends on the mask size.
  - Step edges are blurred up to mask size
- Derivative mask
  - Coordinate are positive and negative
  - **Sum to 0**
  - First derivative mask produces high absolute value at the point of high contrast

STEPS IN EDGE DETECTION

Smoothing

- suppress as much noise as possible, without destroying the true edges.

Enhancement:

- apply a filter to enhance the quality of the edges in the image

Edge detection:

- determine which edge pixels should be discarded as noise and which should be retained
- usually, thresholding provides the criterion used for the detection

Localization:

- determine the exact location of an edge (ex. sub-pixel resolution).
- Edge thinning and linking are usually required.

HIGH-PASS FILTER

- High-Pass Filter lets the high-frequency signal pass to output, making the edge of the picture more contrast.

Original Image

0	-1	0
-1	5	-1
0	-1	0

-1	-1	-1
-1	9	-1
-1	-1	-1

1	-2	1
-2	5	-2
1	-2	1