Image Enhancement in the Spatial Domain

Dr. Aruna Kumar S V
Assistant Professor
FET- Computer Science and Engineering
Ramaiah University of Applied Sciences, Bengaluru

Image Enhancement in the Spatial Domain

- At the end of this session, students will be able to:
  - Describe various Gray Level Transformation techniques.
  - Understand the importance of Histogram Processing.
  - Identify different Arithmetic and Logical operations for Image Enhancement.
  - Understand the importance First and Second Order Derivatives in Image Enhancement.

Principle Objective of Enhancement

- Process an image so that the result will be more suitable than the original image for a specific application.
- The suitableness is up to each application.
- A method which is quite useful for enhancing an image may not necessarily be the best approach for enhancing another images

2 Domains

- Spatial Domain : (image plane)
  - Techniques are based on direct manipulation of pixels in an image
- Frequency Domain :
  - Techniques are based on modifying the Fourier transform of an image
- There are some enhancement techniques based on various combinations of methods from these two categories.

Good images

- For human visual
  - The visual evaluation of image quality is a highly subjective process.
  - It is hard to standardize the definition of a good image.
- For machine perception
  - The evaluation task is easier.
  - A good image is one which gives the best machine recognition results.
- A certain amount of trial and error usually is required before a particular image enhancement approach is selected.

Spatial Domain

- Procedures that operate directly on pixels.

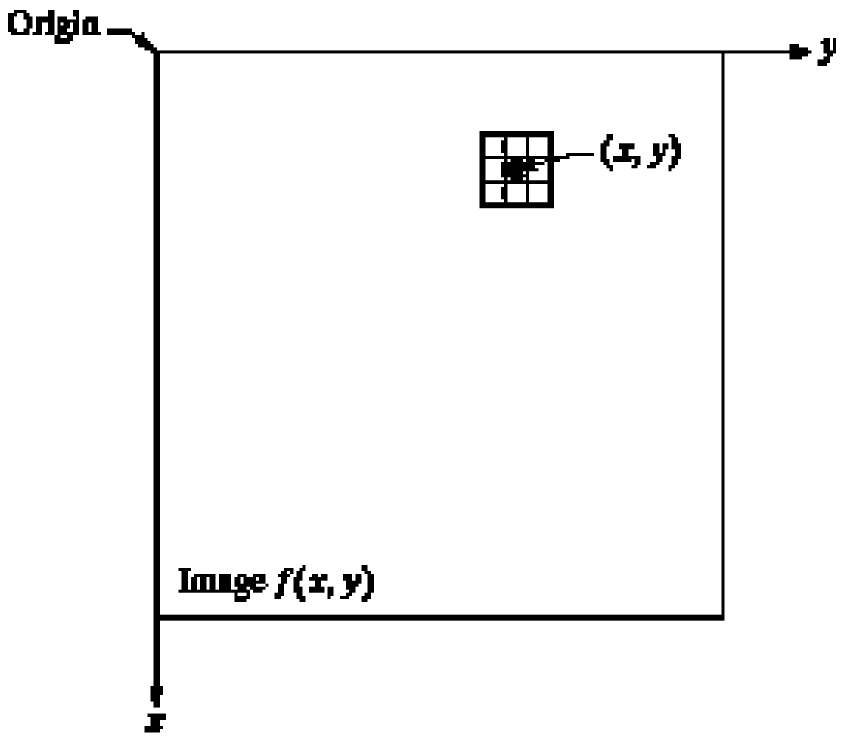
  $$g(x,y) = T[f(x,y)]$$

  where

  – $f(x,y)$ is the input image

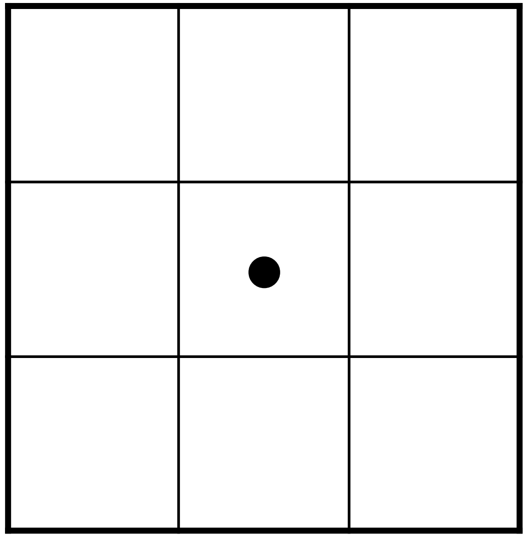
  – $g(x,y)$ is the processed image

  – T is an operator on f defined over some neighborhood of (x,y)



Mask/Filter

- Neighborhood of a point (x,y) can be defined by using a square/rectangular (common used) or circular subimage area centered at (x,y)
- The center of the subimage is moved from pixel to pixel starting at the top of the corner



Point Processing

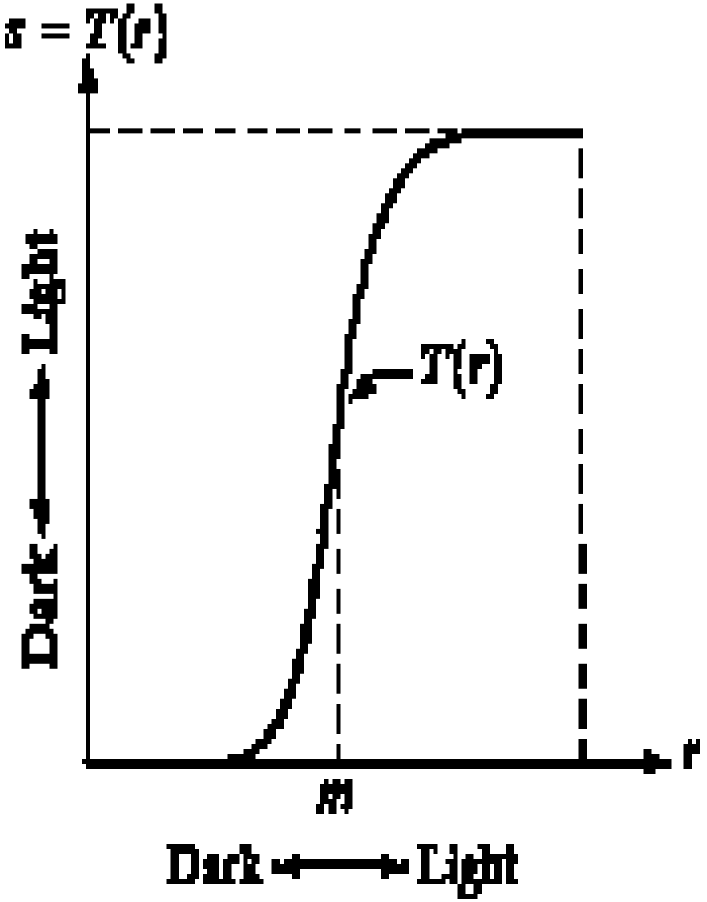
- Neighborhood = 1x1 pixel
- ***g*** depends on only the value of ***f*** at ***(x,y)***
- ***T*** = gray level (or intensity or mapping) transformation function

$$s = T(r)$$

- Where
  - ***r*** = gray level of ***f(x,y)***
  - ***s*** = gray level of ***g(x,y)***

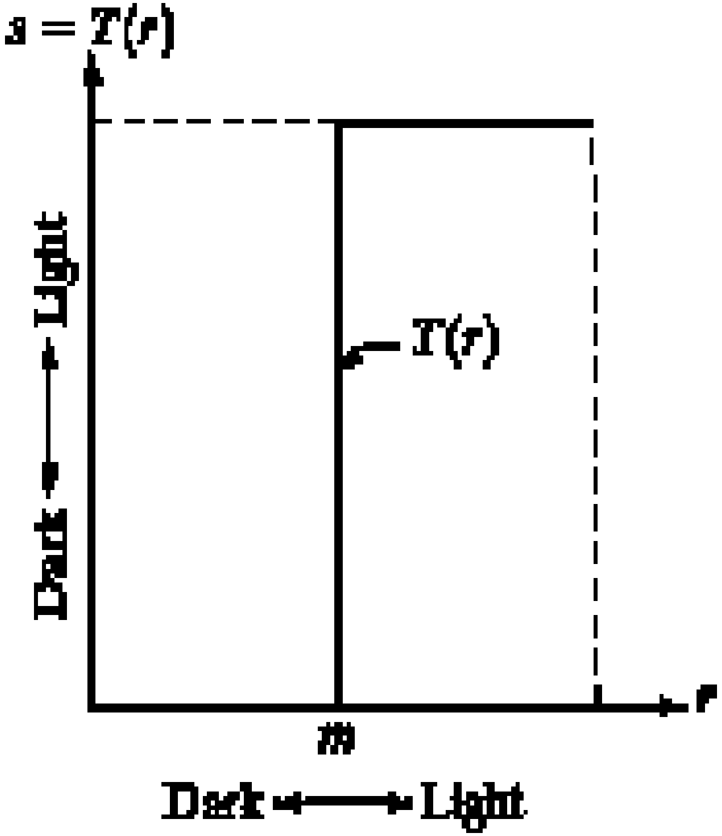
Contrast Stretching

- Produce higher contrast than the original by
  - darkening the levels below m in the original image
  - Brightening the levels above m in the original image



Thresholding

- Produce a two-level (binary) image

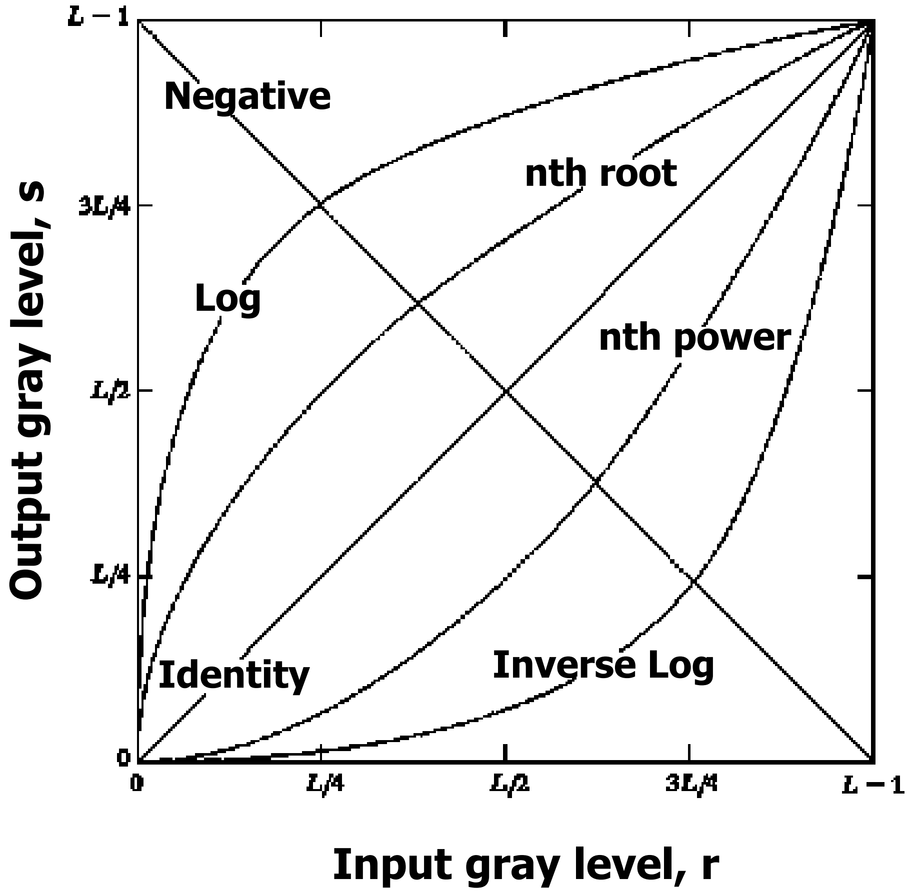


Mask Processing or Filter

- Neighborhood is bigger than 1x1 pixel
- Use a function of the values of f in a predefined neighborhood of (x,y) to determine the value of g at (x,y)
- The value of the mask coefficients determine the nature of the process
- Used in techniques
  - Image Sharpening and Image Smoothing

3 basic gray-level transformation functions

- Linear function
  - Negative and identity transformations
- Logarithm function
  - Log and inverse-log transformation
- Power-law function
  - n^{th} power and n^{th} root transformations



Identity function

- Output intensities are identical to input intensities.
- Is included in the graph only for completeness.

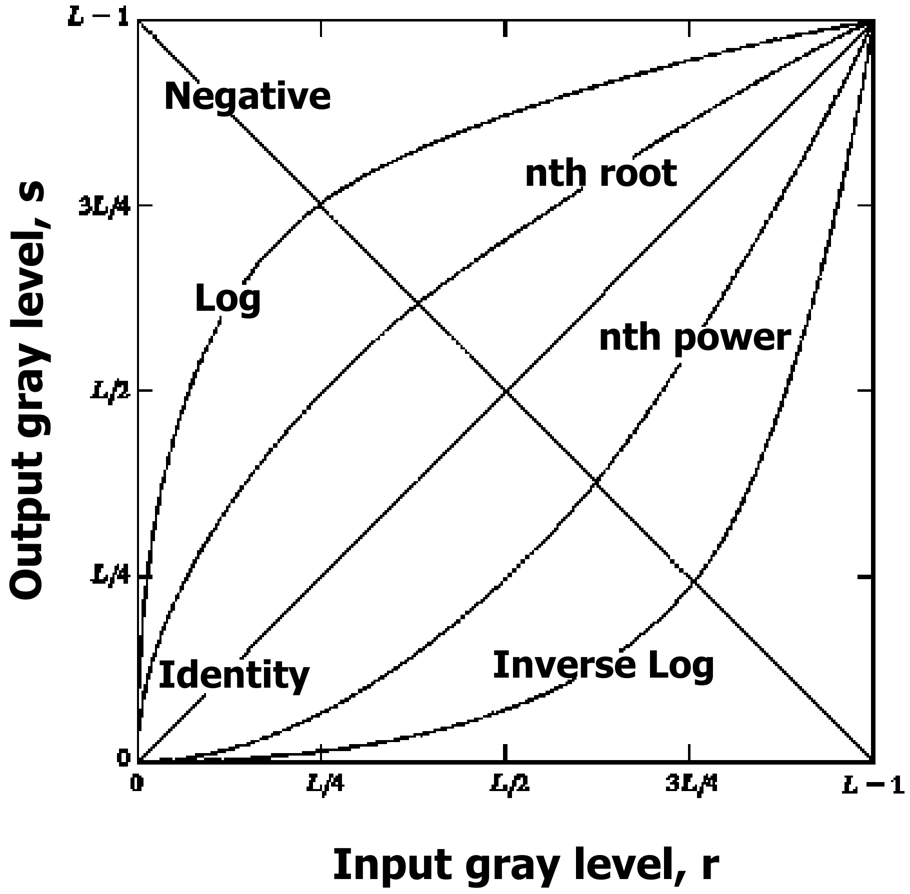
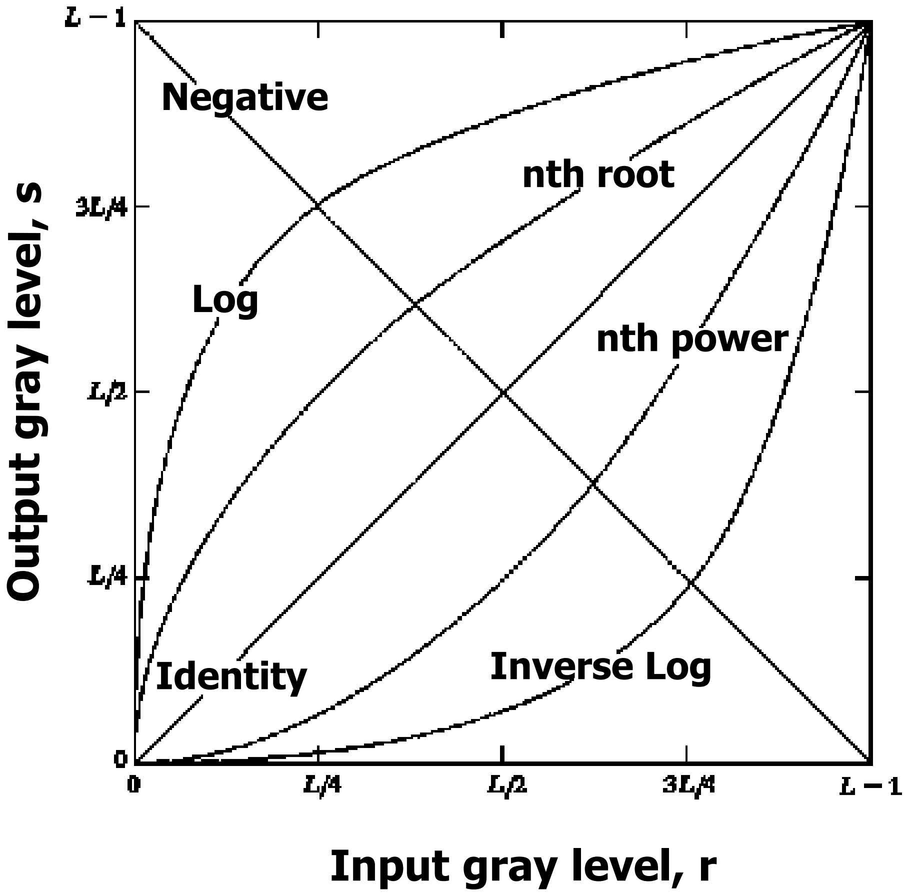


Image Negatives

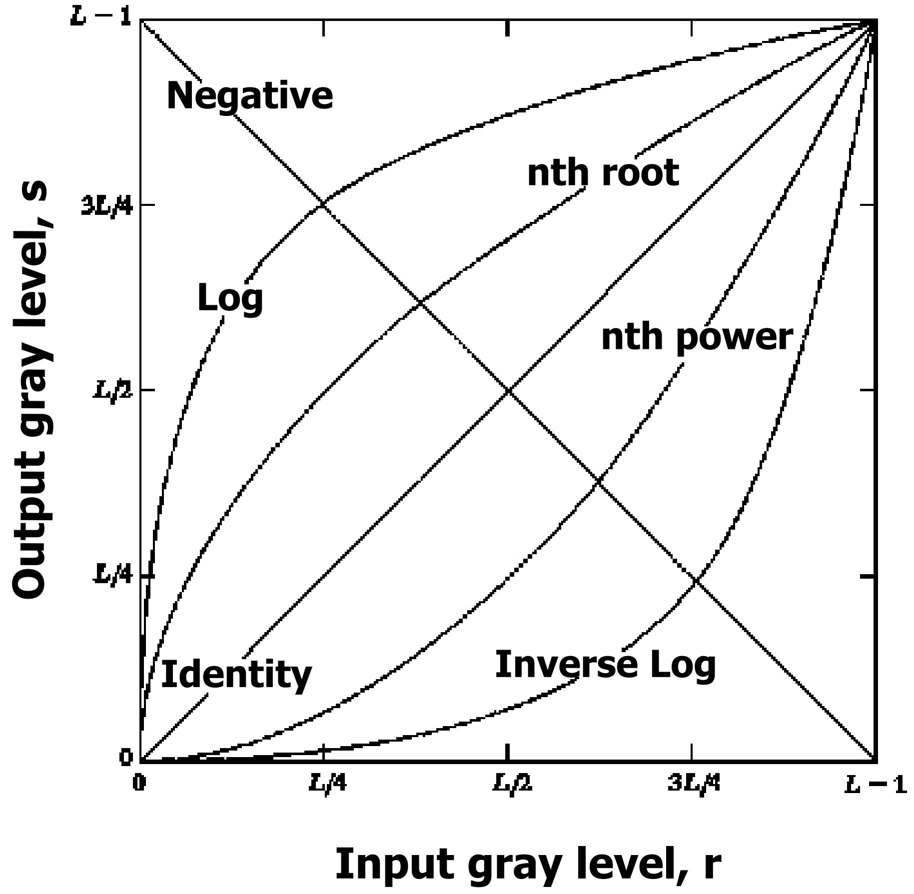
- An image with gray level in the range ***[0, L-1]*** where $L = 2^n$; n = 1, 2…
- Negative transformation :

$$s = L - 1 - r$$

- Reversing the intensity levels of an image.
- Suitable for enhancing white or gray detail embedded in dark regions of an image, especially when the black area dominant in size.



Log Transformations



s = c log (1+r)

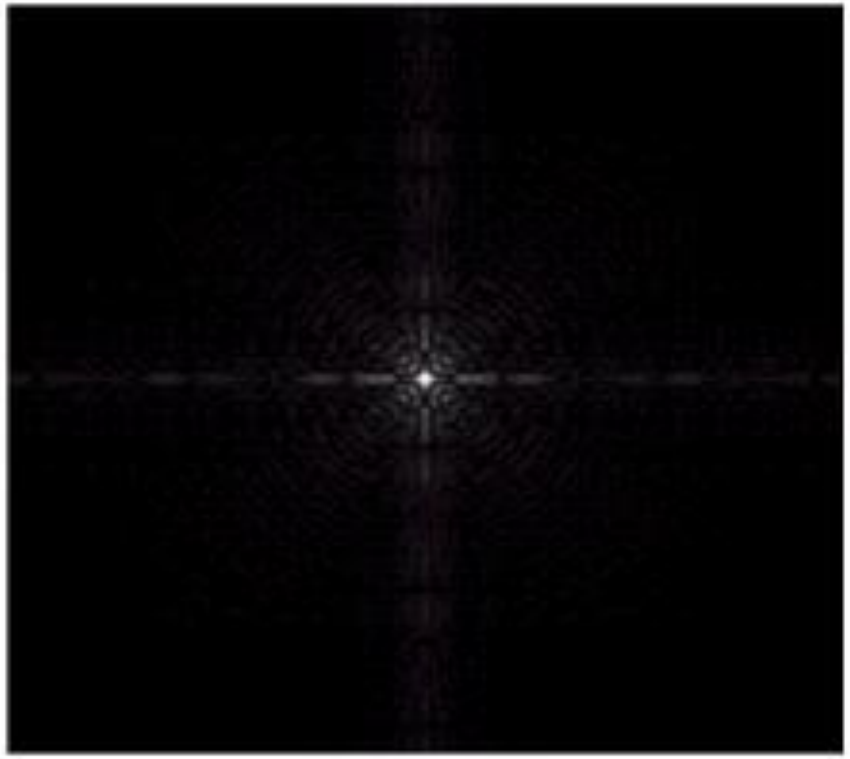
c is a constant and $r \geq 0$

- Log curve maps a narrow range of low gray-level values in the input image into a wider range of output levels.
- Used to expand the values of dark pixels in an image while compressing the higher-level values.

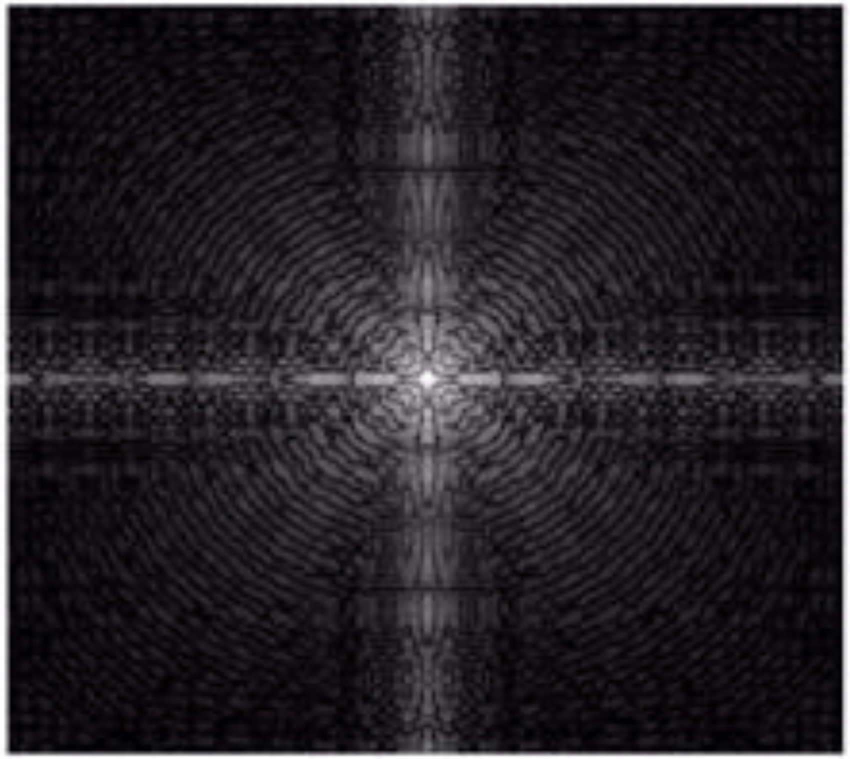
Log Transformations

- It compresses the dynamic range of images with large variations in pixel values
- Example of image with dynamic range: Fourier spectrum image
- It can have intensity range from 0 to 10^6 or higher.
- We can't see the significant degree of detail as it will be lost in the display.

Example of Logarithm Image



Fourier Spectrum with range = 0 to 1.5 x 10^6



Result after apply the log transformation with c = 1, range = 0 to 6.2

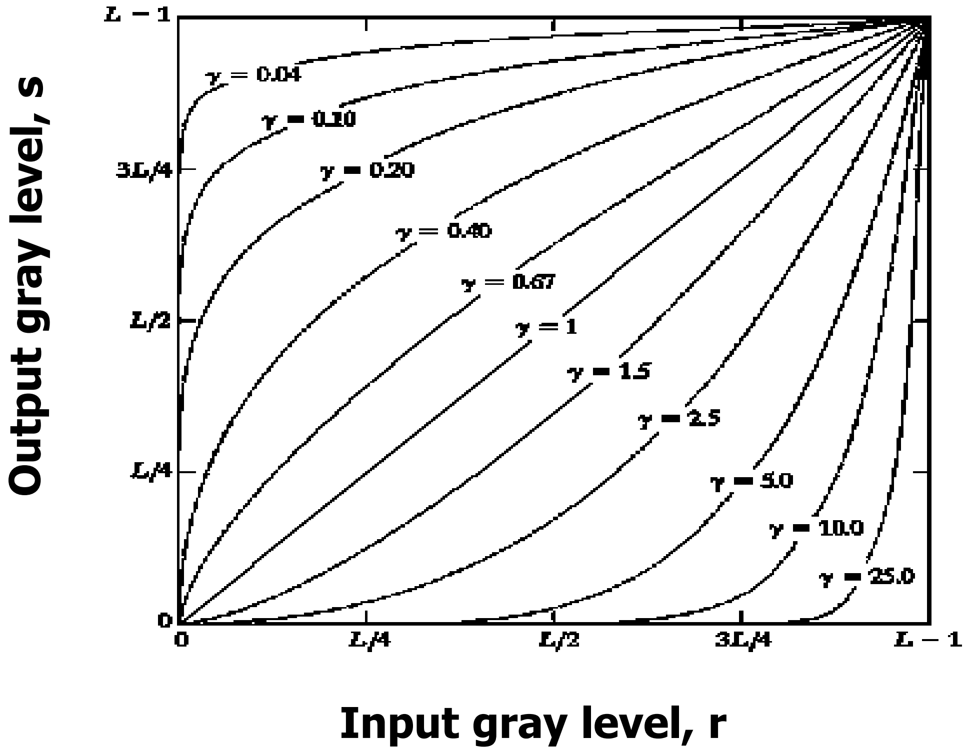
Inverse Logarithm Transformations

- Do opposite to the Log Transformations
- Used to expand the values of high pixels in an image while compressing the darker-level values.

Power-Law Transformations

$$s = cr^{\gamma}$$

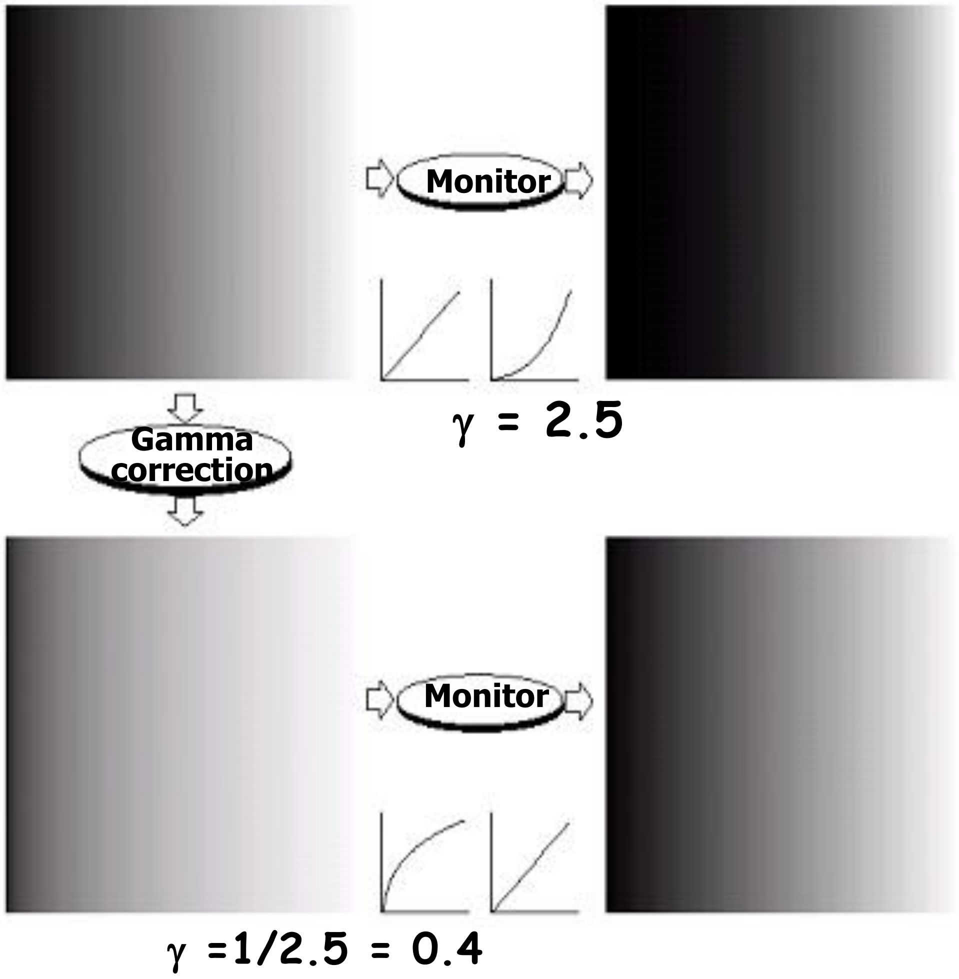
- c and γ are positive constants
- Power-law curves with fractional values of γ map a narrow range of dark input values into a wider range of output values, with the opposite being true for higher values of input levels.
- $c = \gamma = 1$ ⇨ Identity function



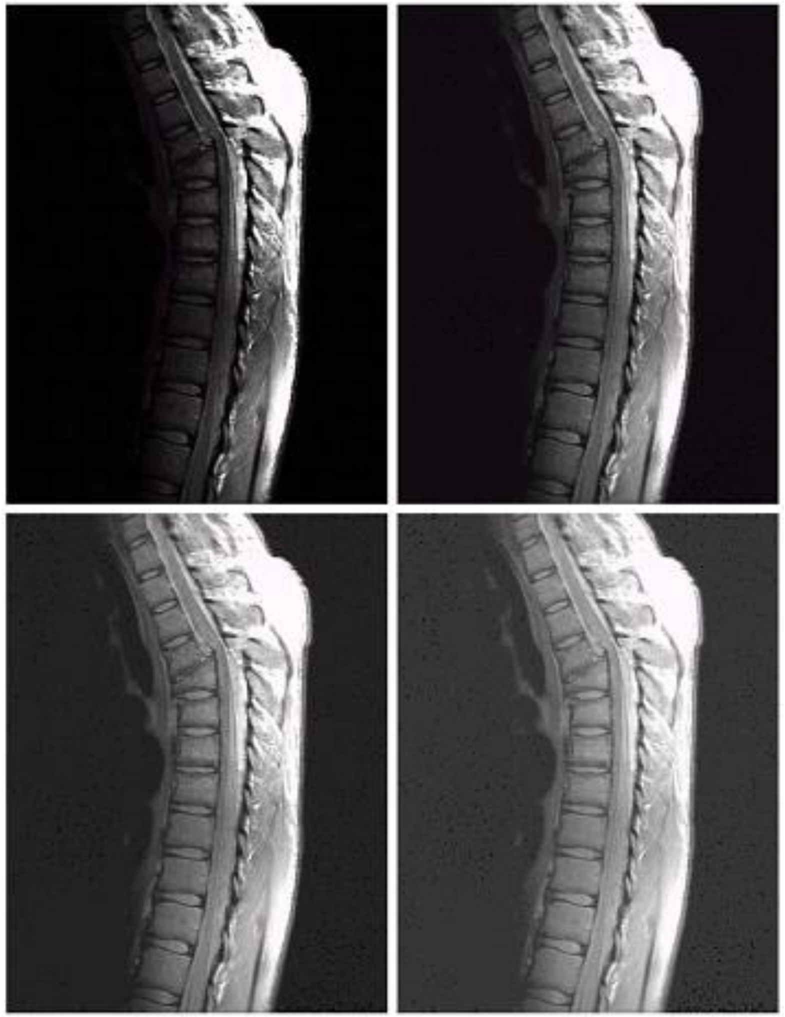
Plots of $s = cr^{\gamma}$ for various values of γ (c = 1 in all cases)

Gamma correction

- Cathode ray tube (CRT) devices have an intensity-to-voltage response that is a power function, with γ varying from 1.8 to 2.5
- The picture will become darker.
- Gamma correction is done by preprocessing the image before inputting it to the monitor with $s = cr^{1/\gamma}$



Another example : MRI



a	b
c	d

(a) a magnetic resonance image of an upper thoracic human spine with a fracture dislocation and spinal cord impingement

- The picture is predominately dark
- An expansion of gray levels are desirable ⇨ needs $\gamma < 1$

(b) result after power-law transformation with $\gamma = 0.6$, c=1

(c) transformation with $\gamma = 0.4$ (best result)

(d) transformation with $\gamma = 0.3$ (under acceptable level)

Effect of decreasing gamma

- When the γ is reduced too much, the image begins to reduce contrast to the point where the image started to have very slight “wash-out” look, especially in the background

Another example

a	b
c	d

(a) image has a washed-out appearance, it needs a compression of gray levels ⇨ needs $\gamma > 1$

(b) result after power-law transformation with $\gamma = 3.0$ (suitable)

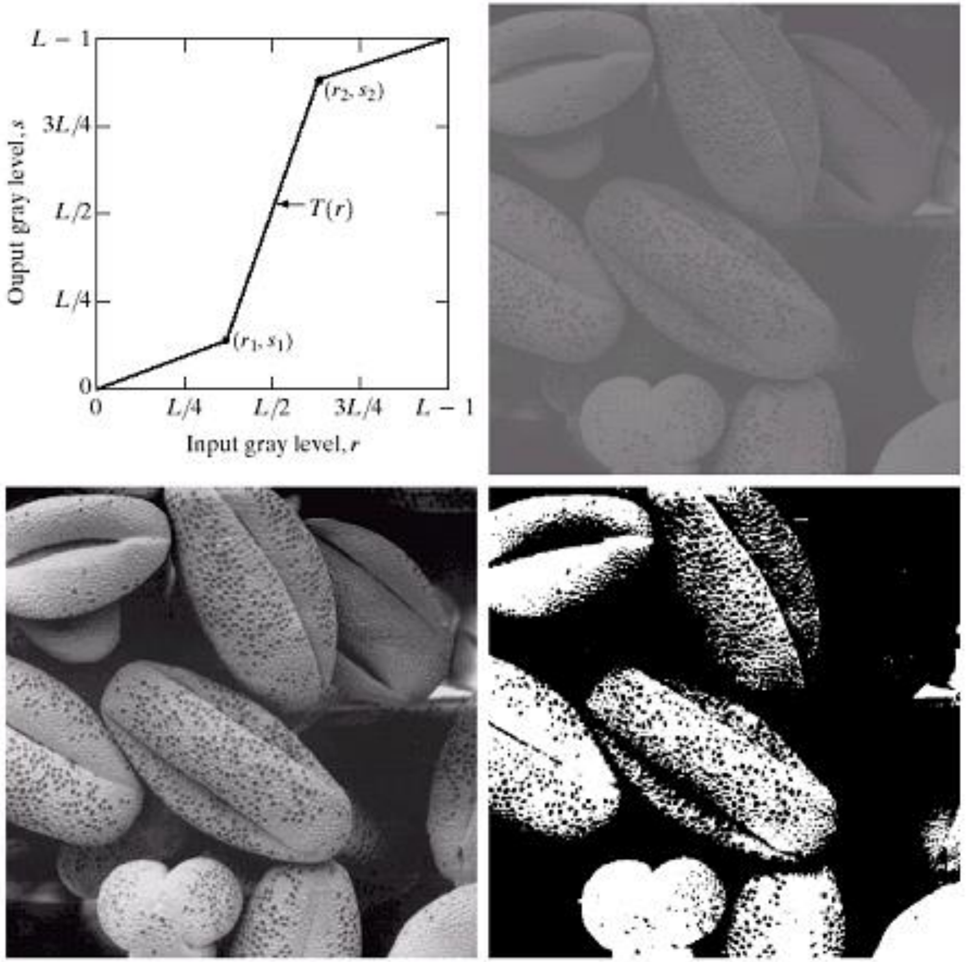
(c) transformation with $\gamma = 4.0$ (suitable)

(d) transformation with $\gamma = 5.0$ (high contrast, the image has areas that are too dark, some detail is lost)

Piecewise-Linear Transformation Functions

- Advantage:
  - The form of piecewise functions can be arbitrarily complex
- Disadvantage:
  - Their specification requires considerably more user input

Contrast Stretching



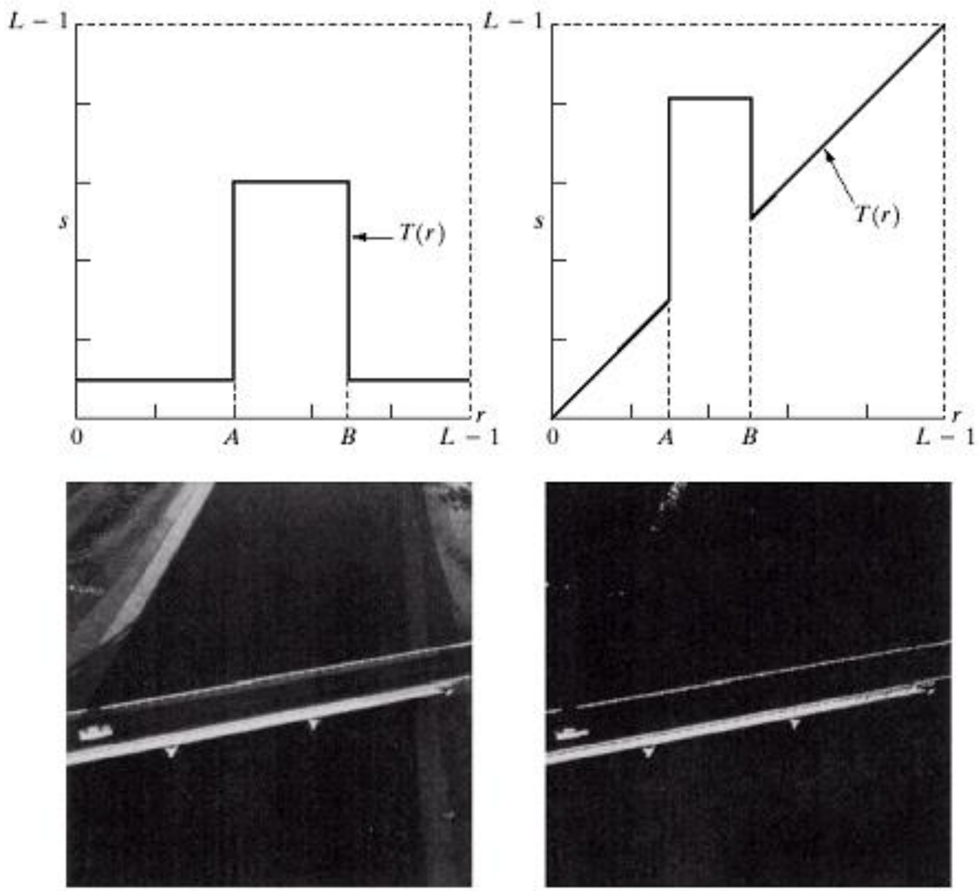
(a) increase the dynamic range of the gray levels in the image

(b) a low-contrast image : result from poor illumination, lack of dynamic range in the imaging sensor, or even wrong setting of a lens aperture of image acquisition

(c) result of contrast stretching: $(r_1,s_1) = (r_{min},0)$ and $(r_2,s_2) = (r_{max},L-1)$

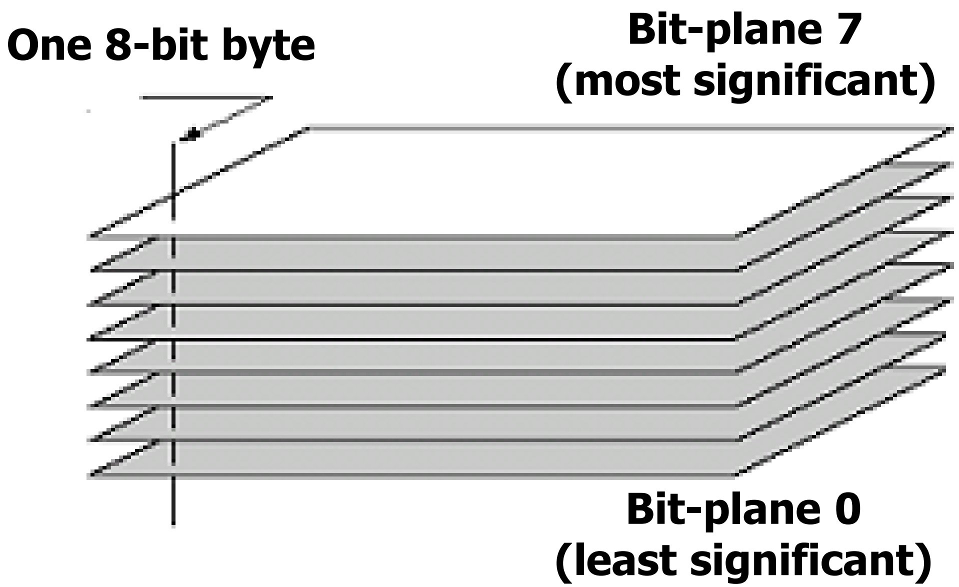
(d) result of thresholding

Gray-level slicing



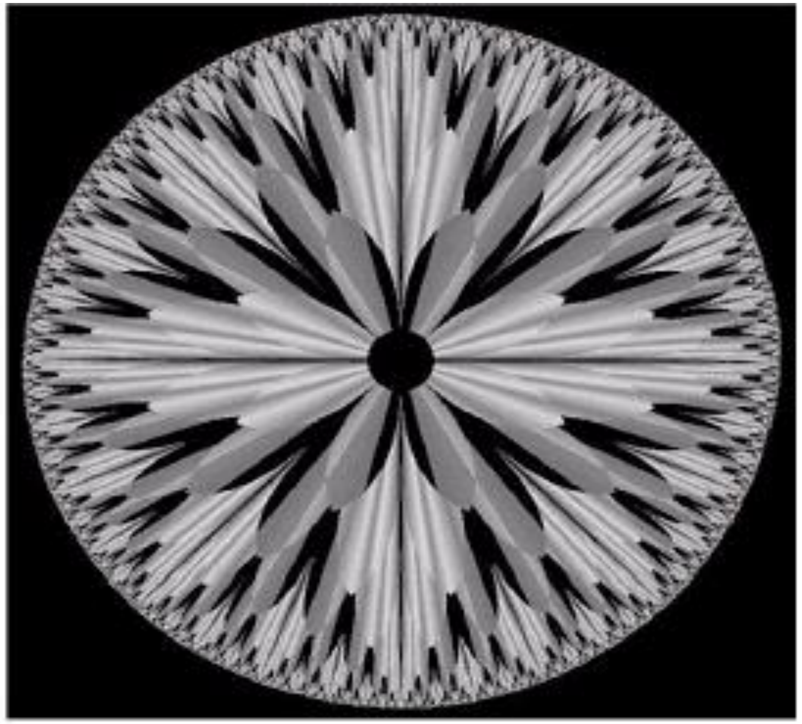
- Highlighting a specific range of gray levels in an image
  - Display a high value of all gray levels in the range of interest and a low value for all other gray levels
- (a) transformation highlights range [A,B] of gray level and reduces all others to a constant level
- (b) transformation highlights range [A,B] but preserves all other levels

Bit-plane slicing



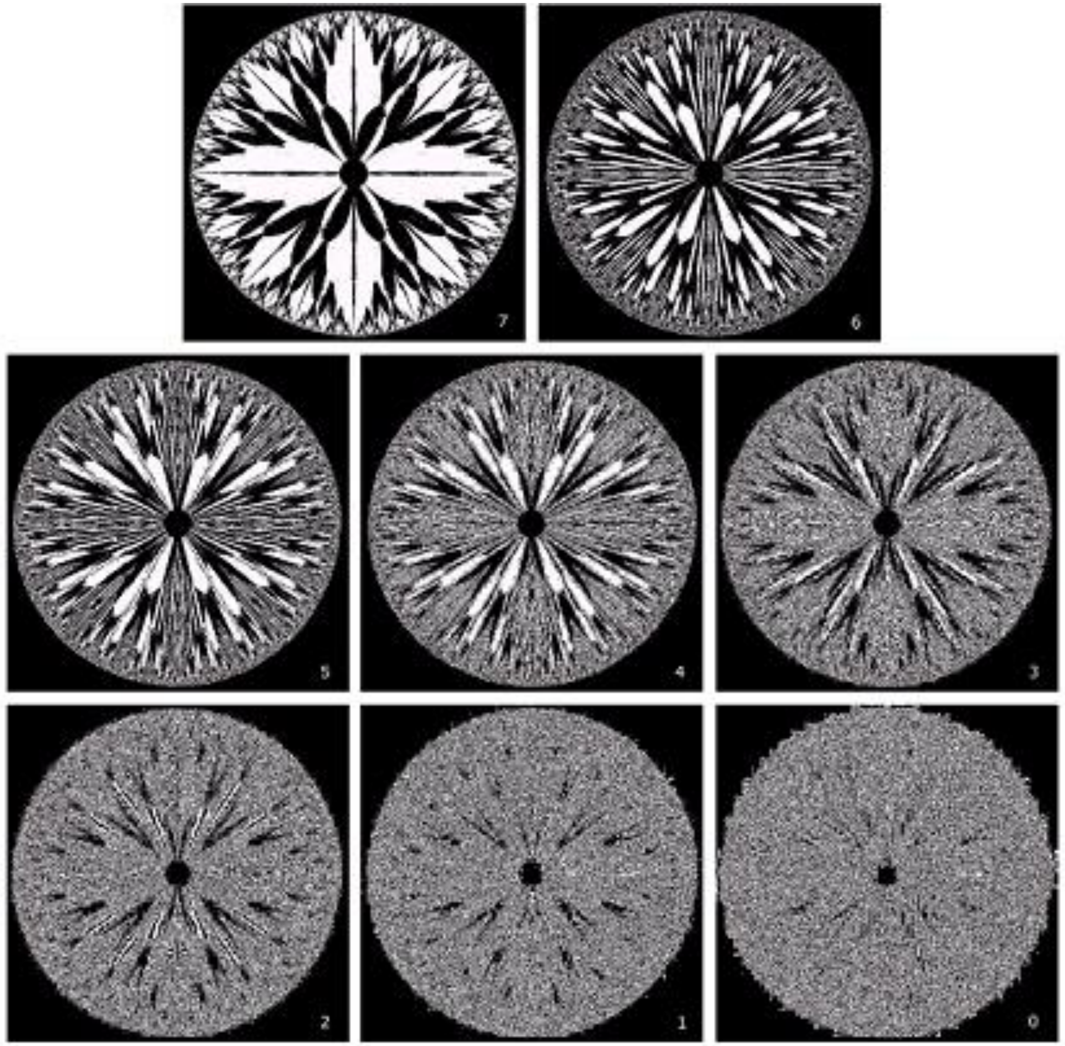
- Highlighting the contribution made to total image appearance by specific bits
- Suppose each pixel is represented by 8 bits
- Higher-order bits contain the majority of the visually significant data
- Useful for analyzing the relative importance played by each bit of the image

Example



An 8-bit fractal image

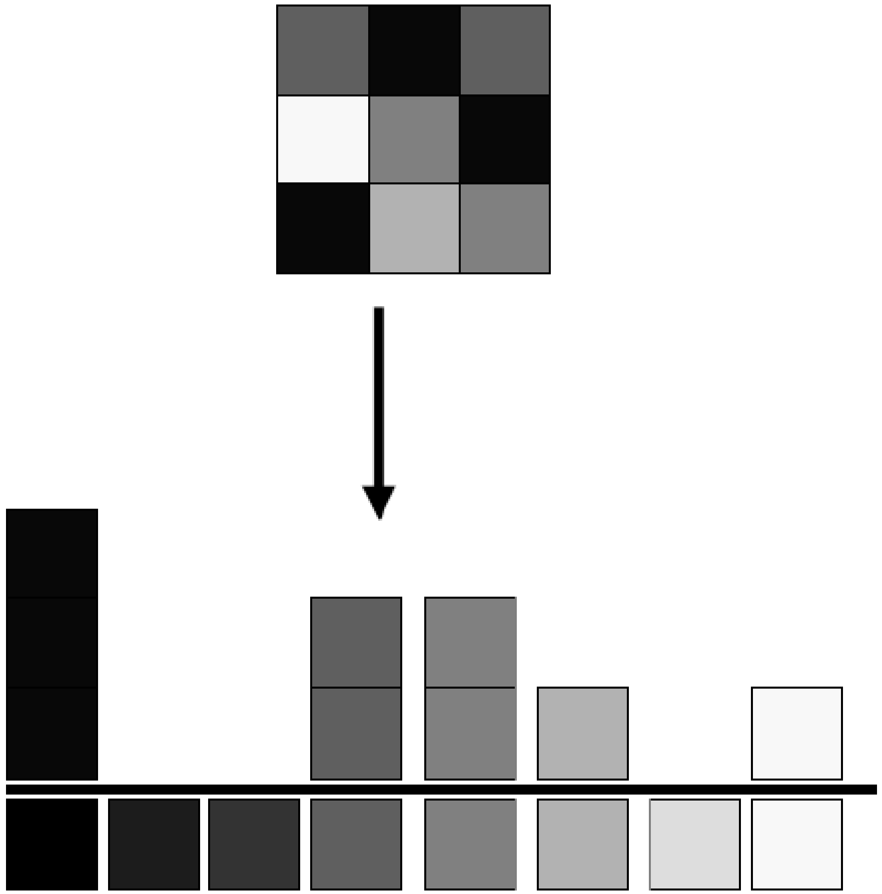
8 bit planes



<table>
<tr><td colspan="3">Bit-plane 7</td><td colspan="3">Bit-plane 6</td></tr>
<tr><td colspan="2">Bit-plane 5</td><td colspan="2">Bit-plane 4</td><td colspan="2">Bit-plane 3</td></tr>
<tr><td colspan="2">Bit-plane 2</td><td colspan="2">Bit-plane 1</td><td colspan="2">Bit-plane 0</td></tr>
</table>

Histogram Processing

Histogram



Histogram Processing

- Histogram of a digital image with gray levels in the range [0,L-1] is a discrete function

$$h(r_k) = n_k$$

- Where
  - r_k : the kth gray level
  - n_k : the number of pixels in the image having gray level r_k
  - $h(r_k)$: histogram of a digital image with gray levels r_k

Normalized Histogram

- dividing each of histogram at gray level r_k by the total number of pixels in the image, n

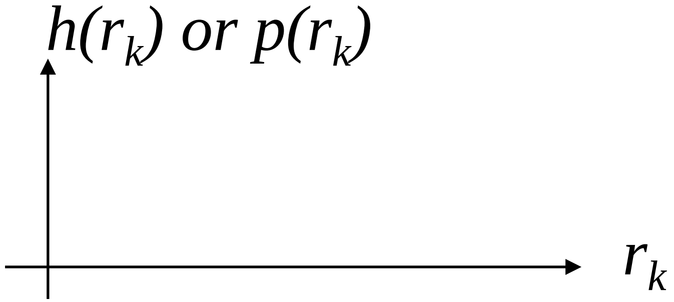
$$p(r_k) = n_k / n$$

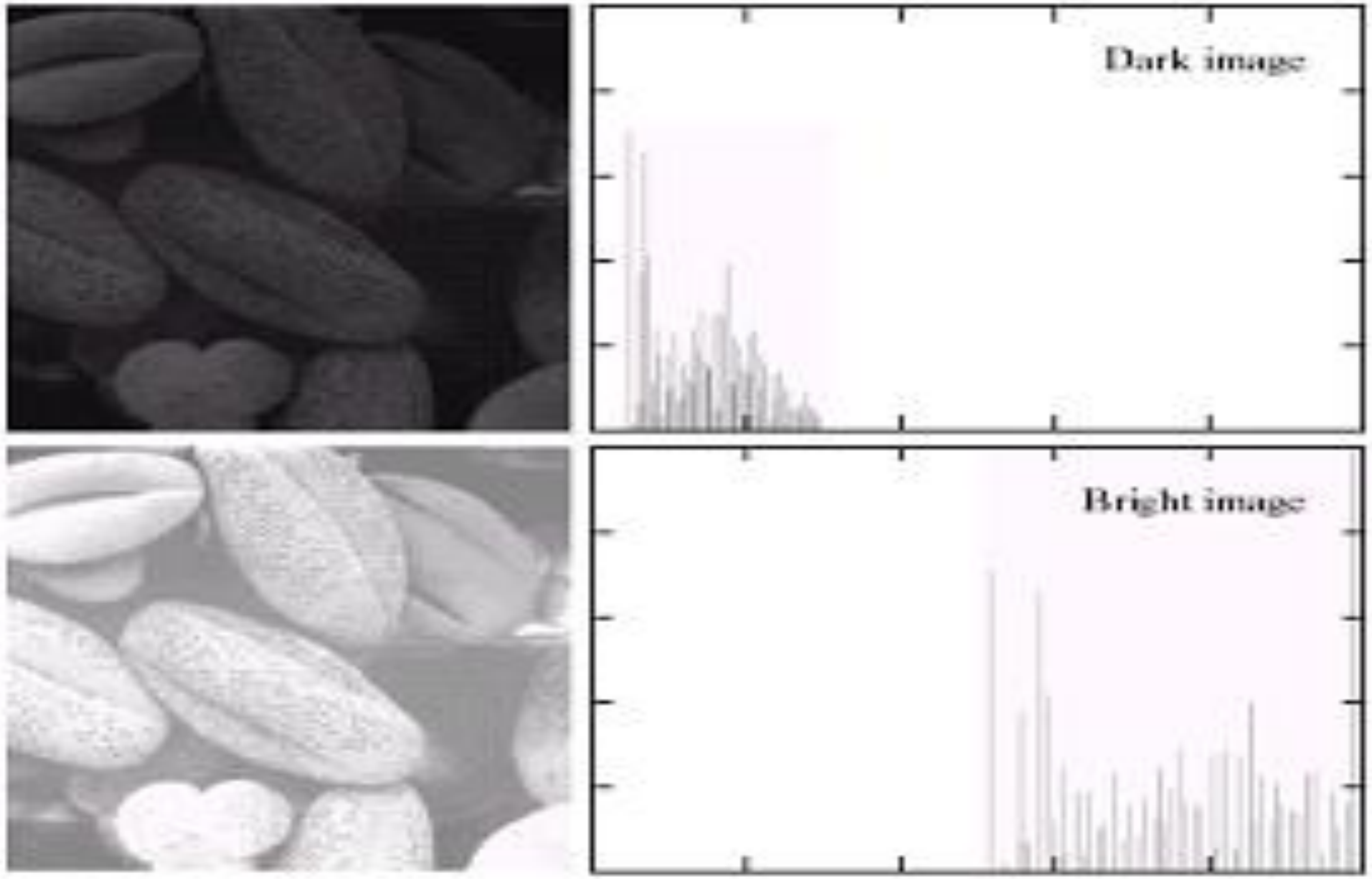
- For k = 0,1,…,L-1
- $p(r_k)$ gives an estimate of the probability of occurrence of gray level r_k
- The sum of all components of a normalized histogram is equal to 1

Histogram Processing

- Basic for numerous spatial domain processing techniques
- Used effectively for image enhancement
- Information inherent in histograms also is useful in image compression and segmentation

Example





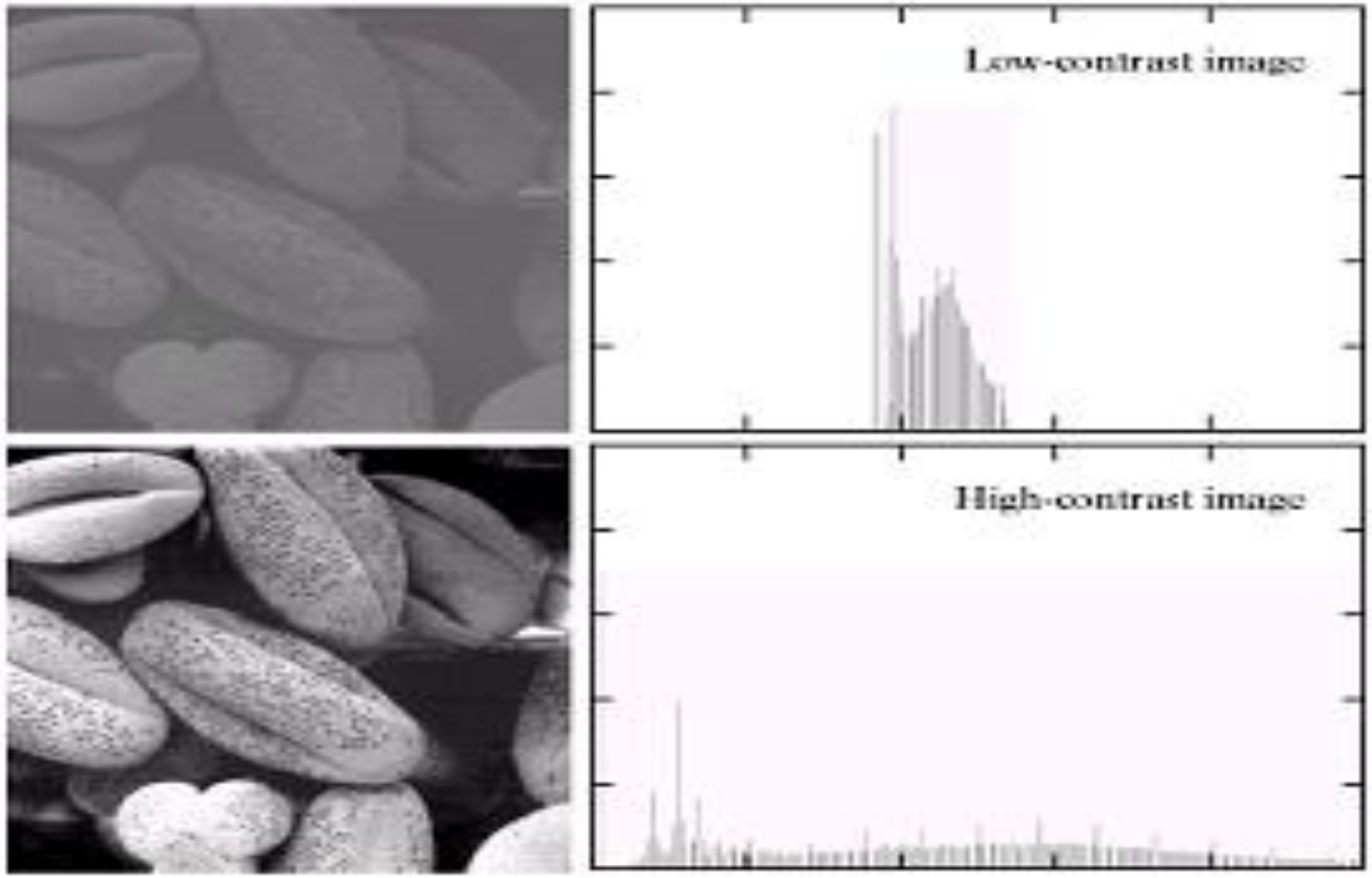
Dark image

Components of histogram are concentrated on the low side of the gray scale.

Bright image

Components of histogram are concentrated on the high side of the gray scale.

Example



Low-contrast image

histogram is narrow and centered toward the middle of the gray scale

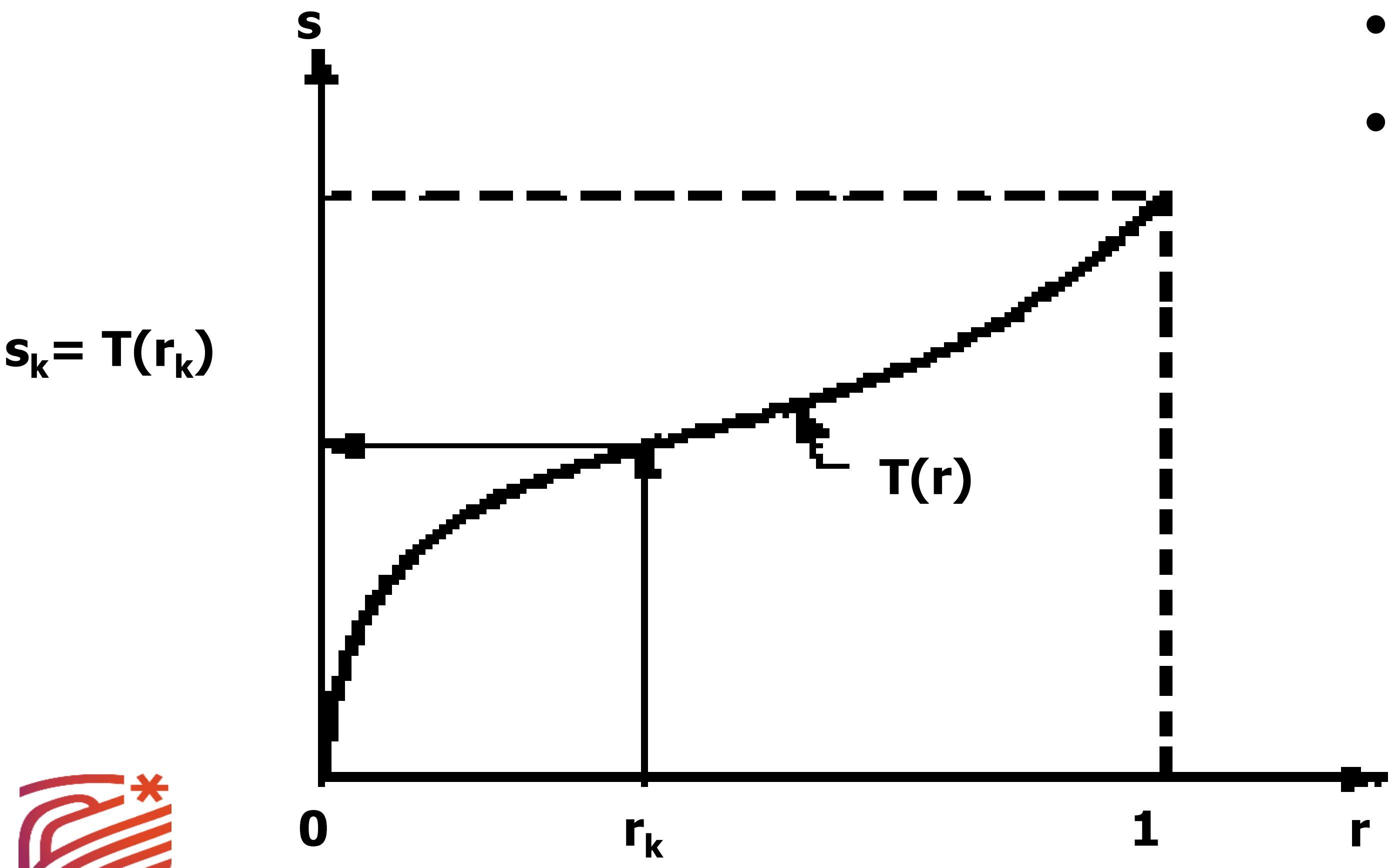
High-contrast image

histogram covers broad range of the gray scale and the distribution of pixels is not too far from uniform, with very few vertical lines being much higher than the others

Histogram Equalization

- As the low-contrast image's histogram is narrow and centered toward the middle of the gray scale, if we distribute the histogram to a wider range the quality of the image will be improved.
- We can do it by adjusting the probability density function of the original histogram of the image so that the probability spread equally

Histogram transformation



$$s = T(r)$$

- Where $0 \leq r \leq 1$
- T(r) satisfies

  (a). T(r) is single-valued and monotonically increasingly in the interval $0 \leq r \leq 1$

  (b). $0 \leq T(r) \leq 1$ for $0 \leq r \leq 1$

2 Conditions of T(r)

- Single-valued (one-to-one relationship) guarantees that the inverse transformation will exist
- Monotonicity condition preserves the increasing order from black to white in the output image thus it won't cause a negative image
- $0 \le T(r) \le 1$ for $0 \le r \le 1$ guarantees that the output gray levels will be in the same range as the input levels.
- The inverse transformation from s back to r is

$$r = T^{-1}(s) \quad ; 0 \le s \le 1$$

Probability Density Function

- The gray levels in an image may be viewed as random variables in the interval [0,1]
- PDF is one of the fundamental descriptors of a random variable

Histogram Equalization

Continues Values

$$s = T(r) = (L-1)\int_0^r p_r(w)\,dw$$

Discrete Values

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

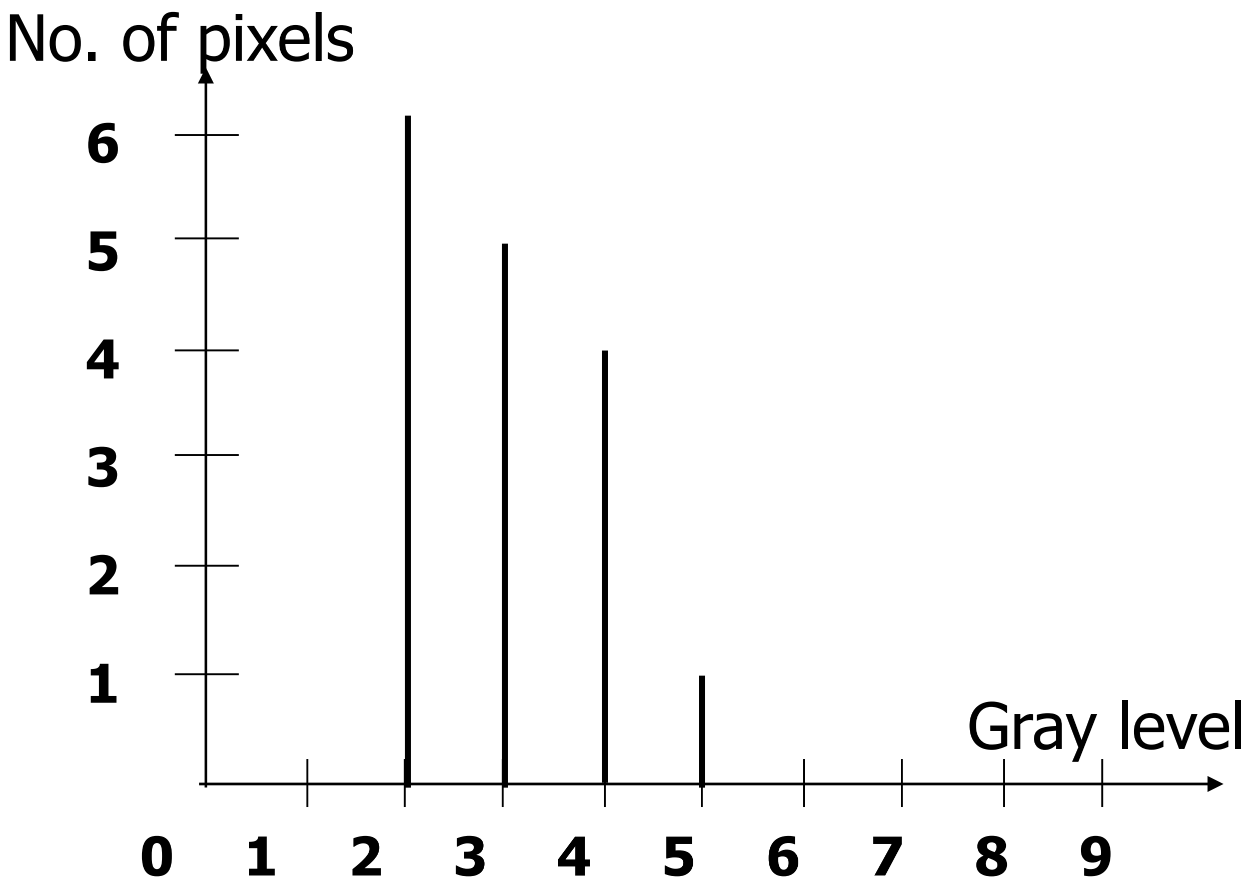
$$= (L-1)\sum_{j=0}^{k}\frac{n_j}{n} = \frac{L-1}{n}\sum_{j=0}^{k} n_j \qquad \text{k=0,1,..., L-1}$$

Example

2	3	3	2
4	2	4	3
3	2	3	5
2	4	2	4

4x4 image

Gray scale = [0,9]



histogram

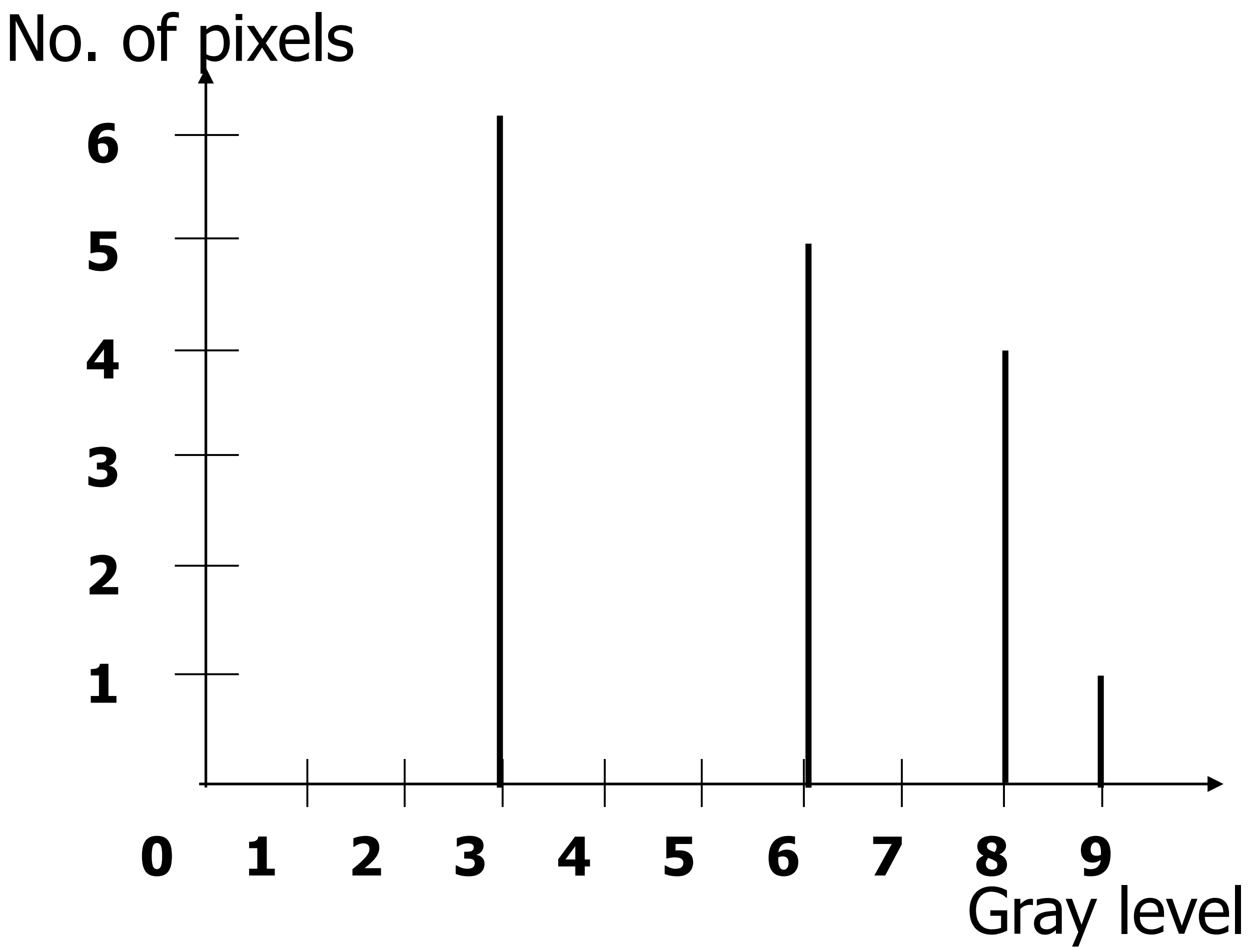
Gray Level(j)	0	1	2	3	4	5	6	7	8	9
No. of pixels	0	0	6	5	4	1	0	0	0	0
$\sum_{j=0}^{k} n_j$	0	0	6	11	15	16	16	16	16	16
$s = \sum_{j=0}^{k} \frac{n_j}{n}$	0	0	6 / 16	11 / 16	15 / 16	16 / 16	16 / 16	16 / 16	16 / 16	16 / 16
s x 9	0	0	3.3 ≈3	6.1 ≈6	8.4 ≈8	9	9	9	9	9

Example

3	6	6	3
8	3	8	6
6	3	6	9
3	8	3	8

Output image

Gray scale = [0,9]



Histogram equalization

Histogram Equalization

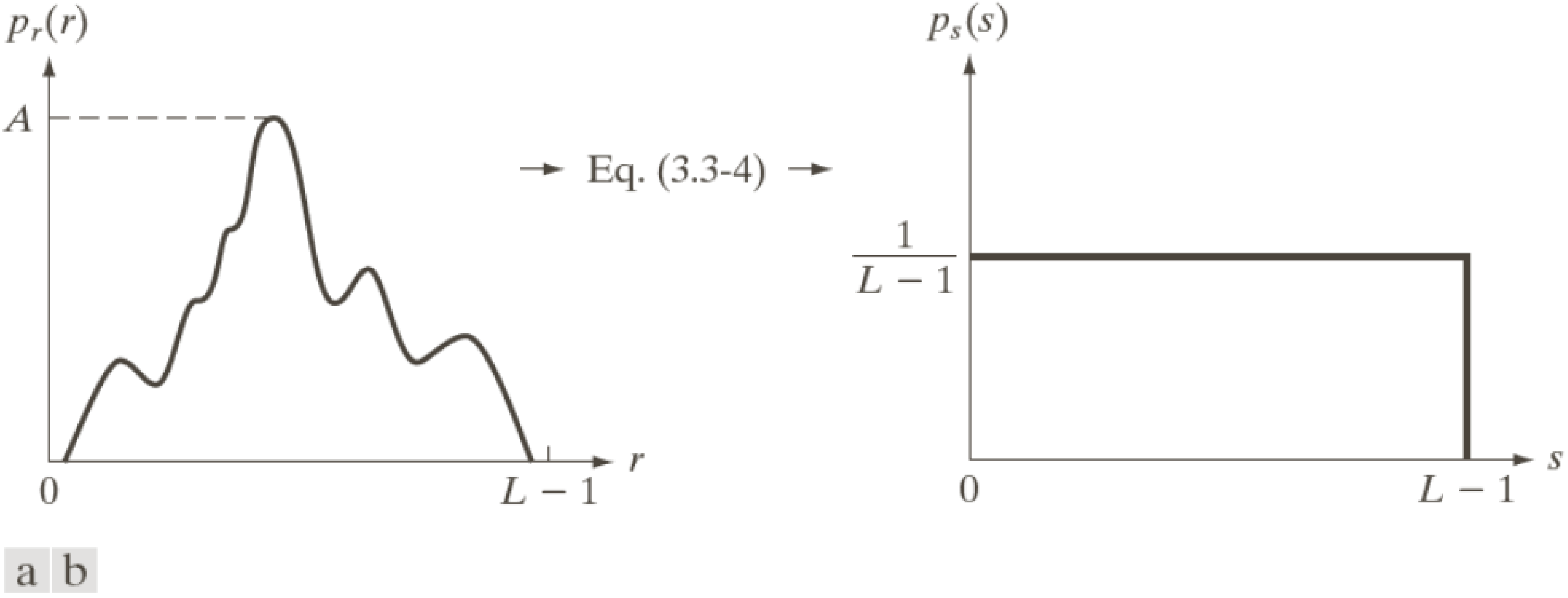
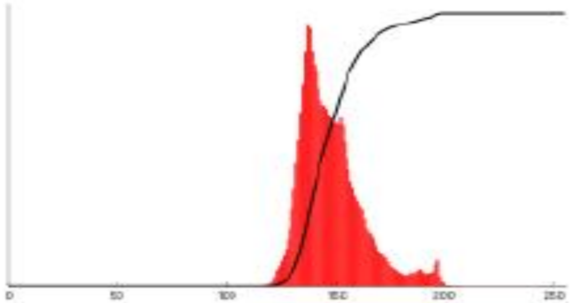
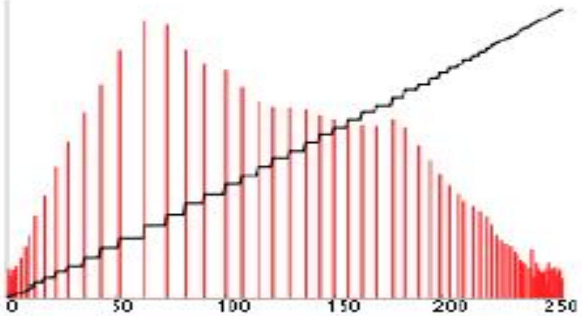


FIGURE 3.18 (a) An arbitrary PDF. (b) Result of applying the transformation in Eq. (3.3-4) to all intensity levels, r. The resulting intensities, s, have a uniform PDF, independently of the form of the PDF of the r's.

Histogram Equalization

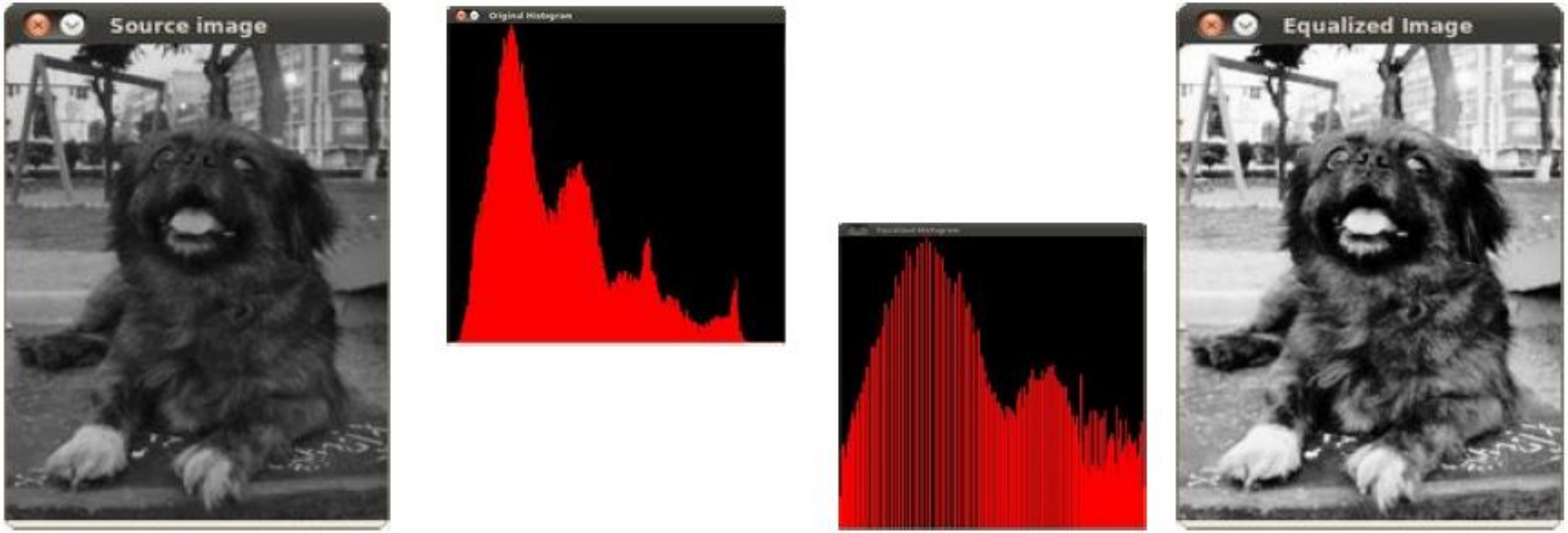


Histogram & Cumulative Histogram



Before & After Histogram Equalization

Source: https://en.wikipedia.org/wiki/Histogram_equalization

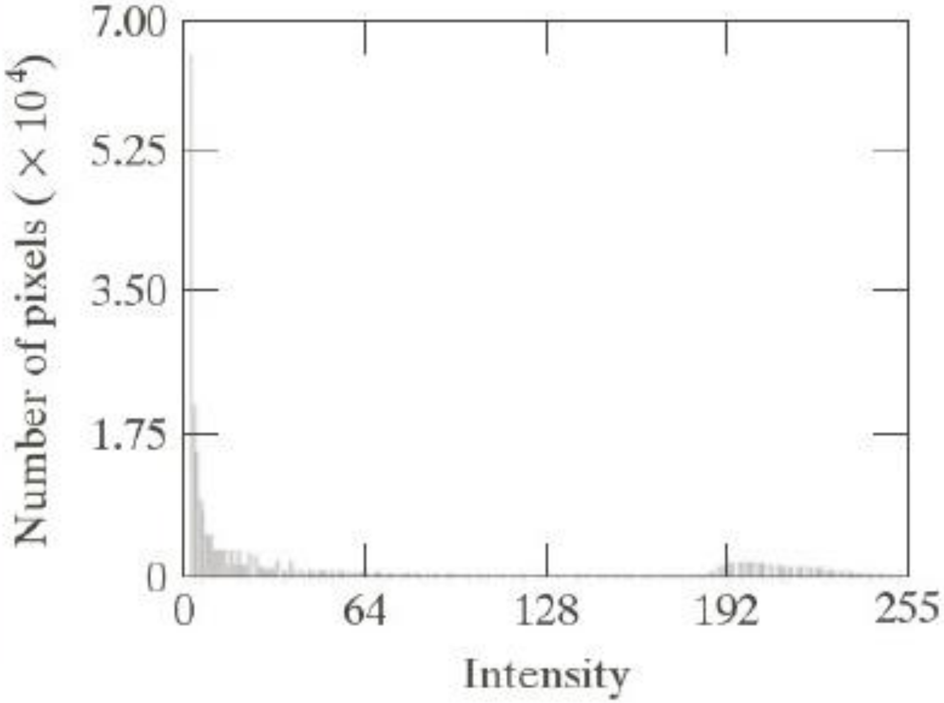


Before & After Histogram Equalization

Source: http://docs.opencv.org/2.4/doc/tutorials/imgproc/histograms/histogram_equalization/histogram_equalization.html

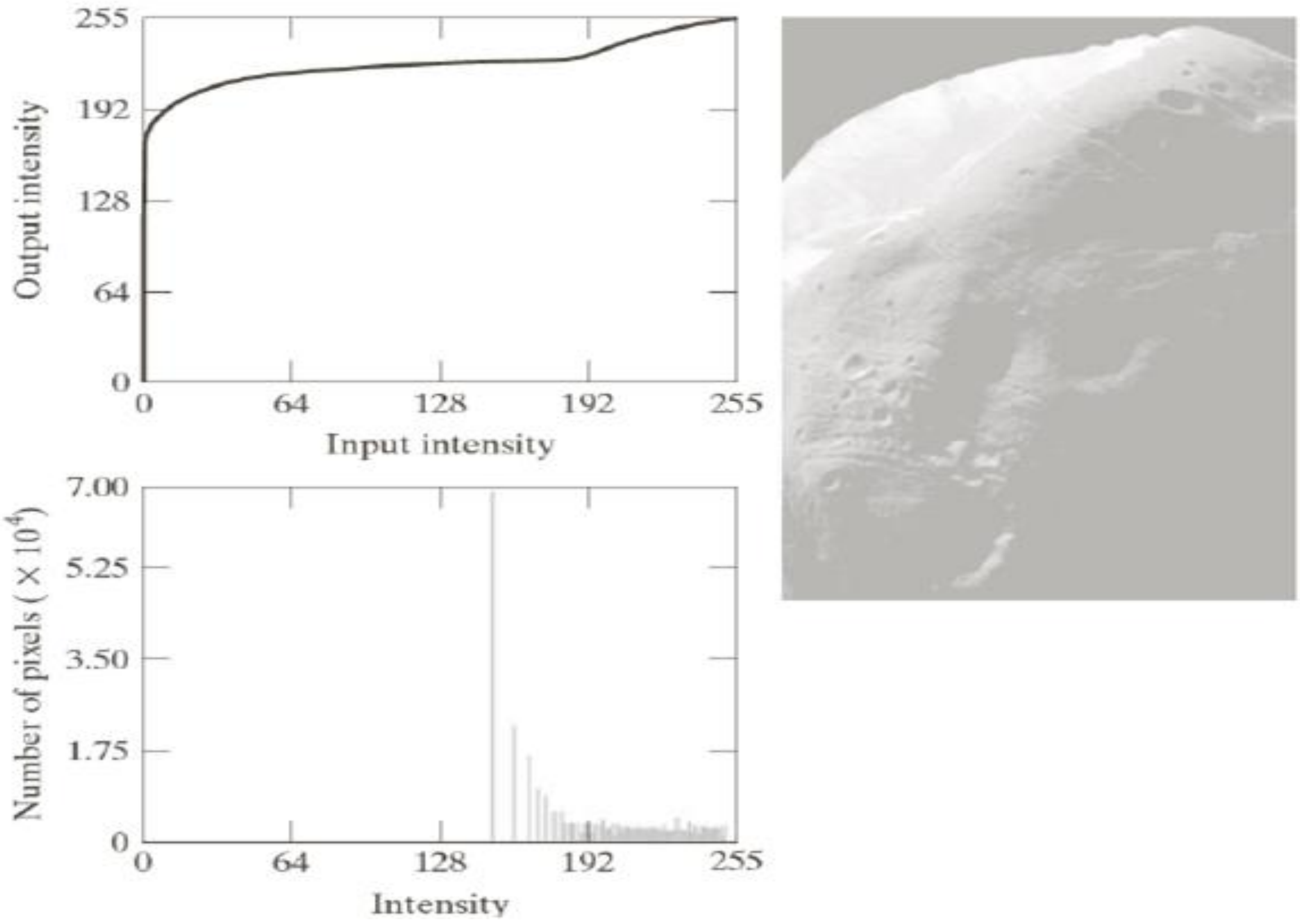
Note

- It is clearly seen that
  - Histogram equalization distributes the gray level to reach the maximum gray level (white) because the cumulative distribution function equals 1 when $0 \le r \le L-1$
  - If the cumulative numbers of gray levels are slightly different, they will be mapped to little different or same gray levels as we may have to approximate the processed gray level of the output image to integer number
  - Thus the discrete transformation function can't guarantee the one to one mapping relationship



a b

FIGURE 3.23
(a) Image of the Mars moon Phobos taken by NASA's *Mars Global Surveyor.* (b) Histogram. (Original image courtesy of NASA.)



a b
c

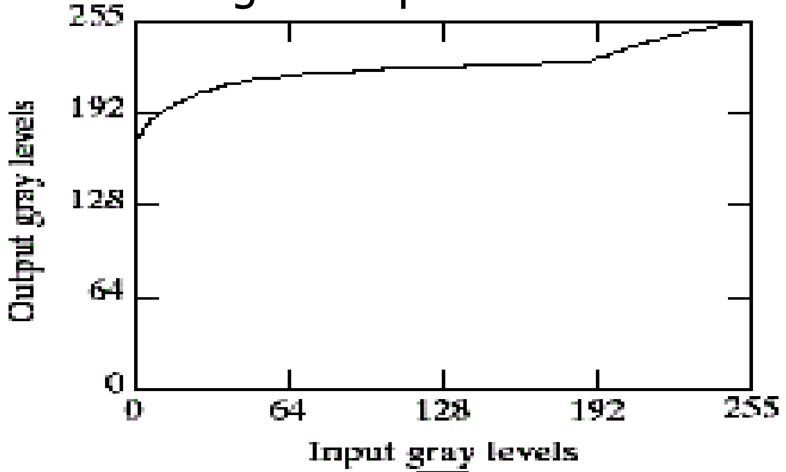
FIGURE 3.24
(a) Transformation function for histogram equalization.
(b) Histogram-equalized image (note the washed-out appearance).
(c) Histogram of (b).

- Histogram Equalization – Failure Example
  - If the histogram is heavily skewed, equalization may not produce good result
  - Then we need to find transformation to a ‘desired’ histogram

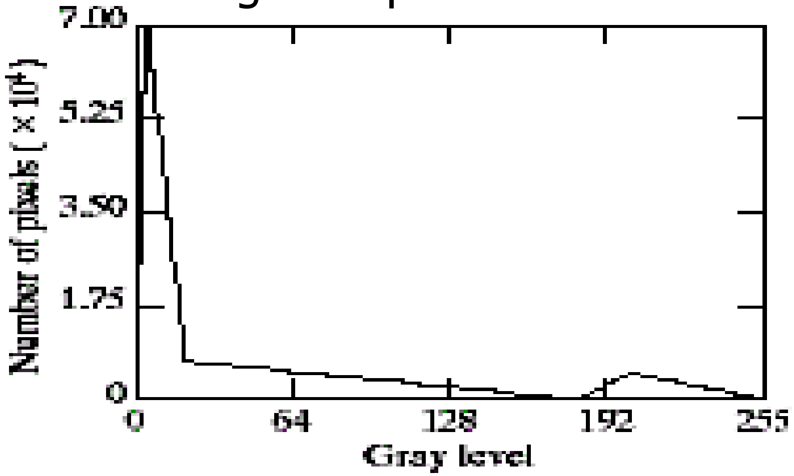
Solve the problem

- Since the problem with the transformation function of the histogram equalization was caused by a large concentration of pixels in the original image with levels near 0
- A reasonable approach is to modify the histogram of that image so that it does not have this property

Histogram Equalization



Histogram Specification



Histogram Matching/ Specification

Histogram Matching/ Specification

- Goal: Specify the shape of the histogram:

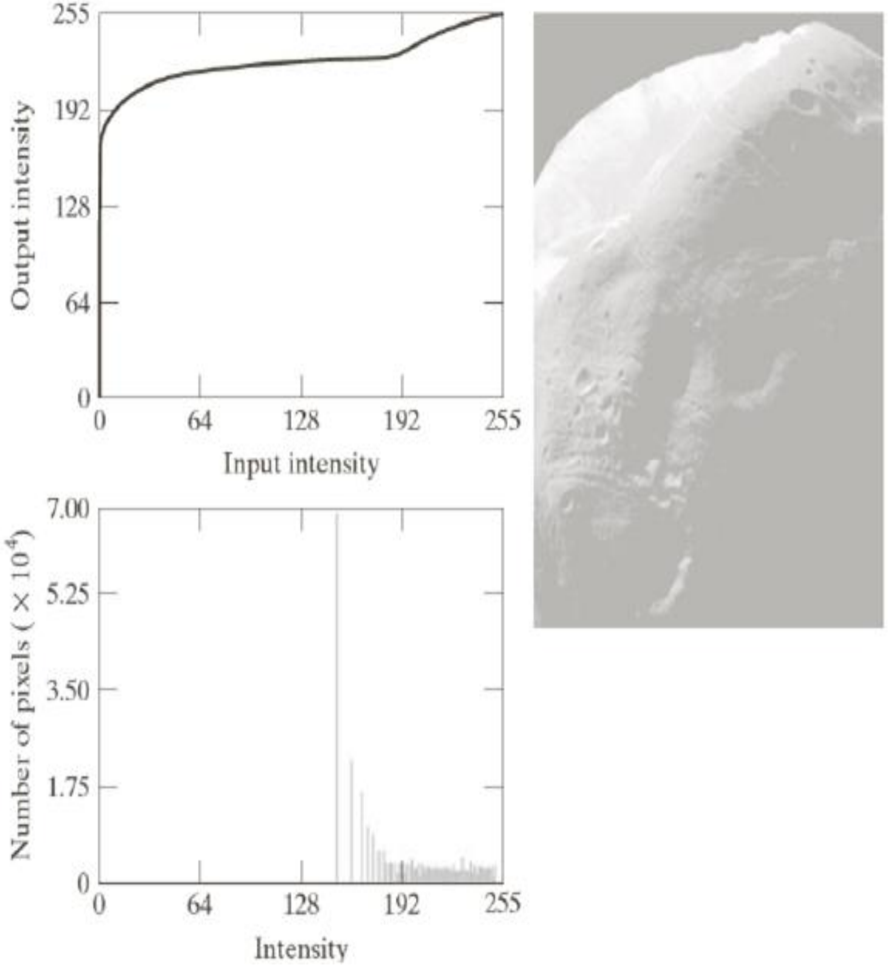
$$p_r(r) \xrightarrow{?} p_z(z)$$

$$s = T(r) = (L-1)\int_0^{r} p_r(w)\, dw$$

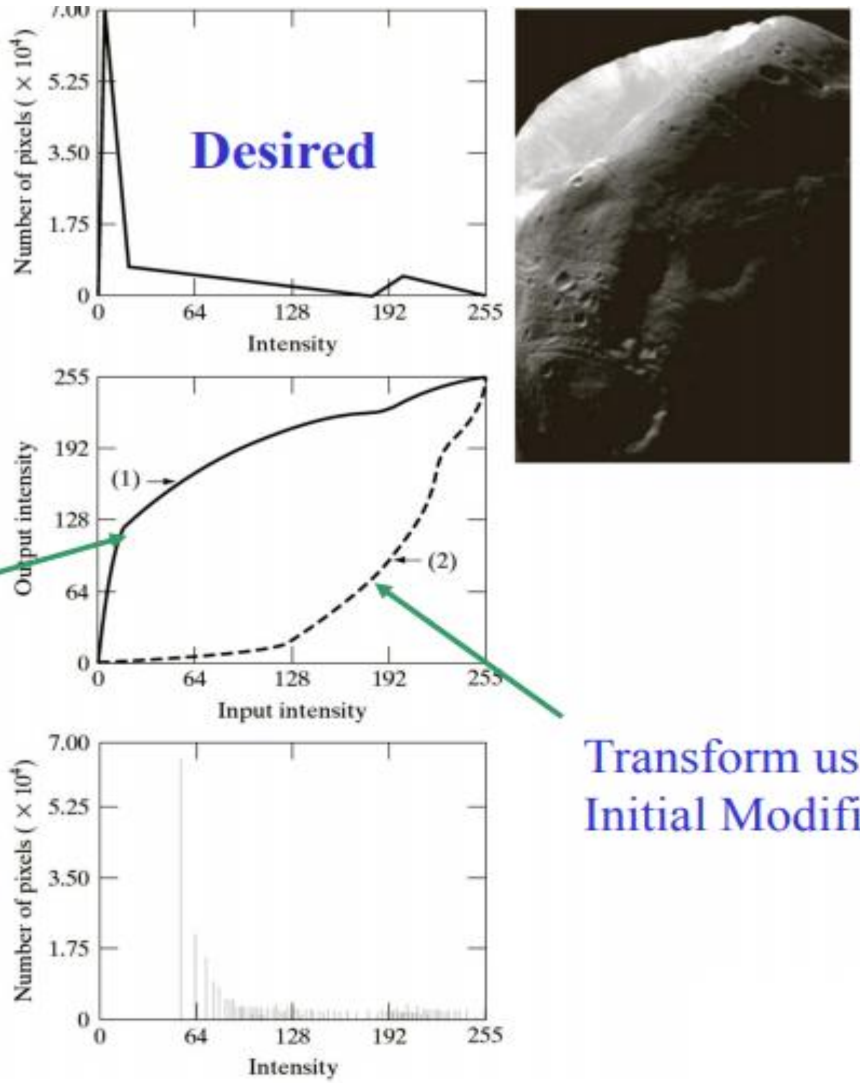
$$G(z) = (L-1)\int_0^{z} p_z(t)\, dt = s$$

$$z = G^{-1}[T(r)] = G^{-1}(s)$$

where $p_r(r)$ and $p_z(z)$ are i/p & o/p PDFs



Transform using
Initial CDF



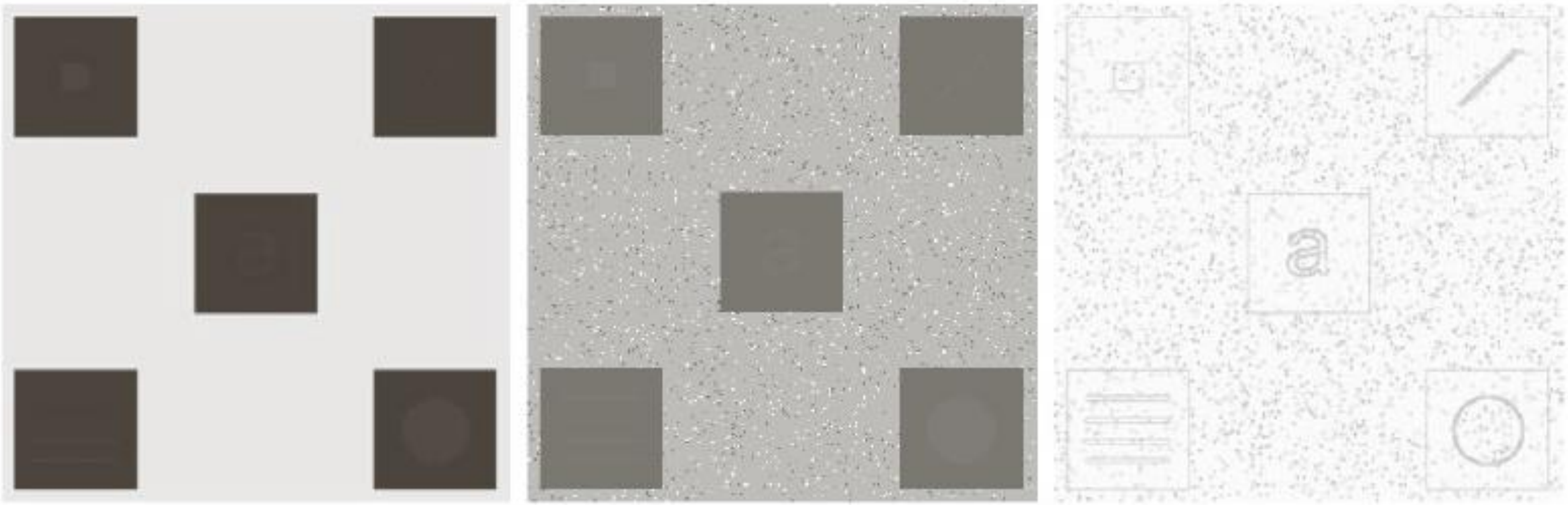
Transform using
Initial Modified CDF

- Histogram processing methods are global processing, in the sense that pixels are modified by a transformation function based on the gray-level content of an entire image.
- Sometimes, we may need to enhance details over small areas in an image, which is called a local enhancement.

Local Histogram Processing

Local Processing Steps

- Define a Neighborhood
- Move its center from pixel to pixel
- Apply histogram equalization / matching @ center
- Non-overlapping computation is fast but blocky



a b c

FIGURE 3.26 (a) Original image. (b) Result of global histogram equalization. (c) Result of local histogram equalization applied to (a), using a neighborhood of size 3 × 3.

Histogram Statistics For Image Enhancement

Use of Global Statistical Measures

$$\mu_n(r) = \sum_{i=0}^{L-1} (r_i - m)^n \, p(r_i) \approx \frac{1}{MN} \sum_{x=1}^{M} \sum_{y=1}^{N} \left[f(x,y) - m \right]^n$$

$$m = \sum_{i=0}^{L-1} r_i p(r_i) \approx \frac{1}{MN} \sum_{x=1}^{M} \sum_{y=1}^{N} f(x,y)$$

Gross adjustments in overall intensity (m) and contrast (μ_2)

Histogram Statistics For Image Enhancement

Local mean and local variance:

$$m_{S_{xy}}(x,y)=\sum_{i=0}^{L-1} r_i p_{S_{xy}}(r_i) \approx \frac{1}{|S_{xy}|}\sum_{(s,t)\in S_{xy}} f(s,t)$$

$$\sigma^2_{S_{xy}}(x,y)=\sum_{i=0}^{L-1}\left(r_i - m_{S_{xy}}(x,y)\right)^2 p_{S_{xy}}(r_i) \approx \frac{1}{|S_{xy}|}\sum_{(s,t)\in S_{xy}} \left[f(s,t)-m_{S_{xy}}(x,y)\right]^2$$

S_{xy} : Neighborhood centered on (x,y)

Local information intensity and contrast (edges)

Original

Local Histogram

Local Statistics

Enhancement using Arithmetic/Logic Operations

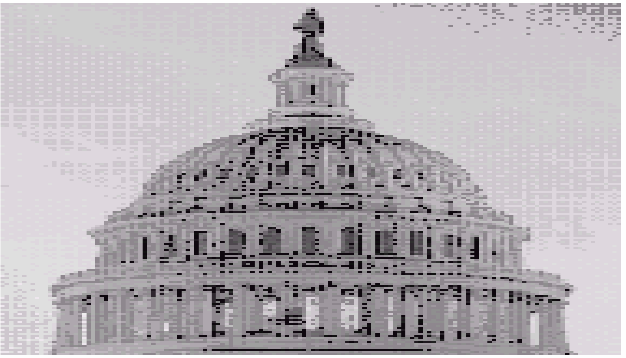
Enhancement using Arithmetic/Logic Operations

- Arithmetic/Logic operations perform on pixel by pixel basis between two or more images
- except NOT operation which perform only on a single image

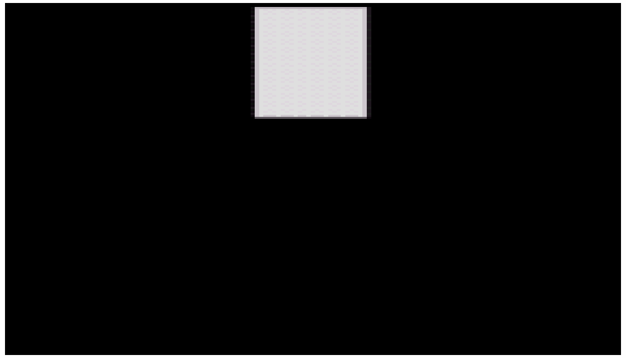
Logic Operations

- Logic operation performs on gray-level images, the pixel values are processed as binary numbers
- light represents a binary 1, and dark represents a binary 0
- NOT operation = negative transformation

Example of AND Operation



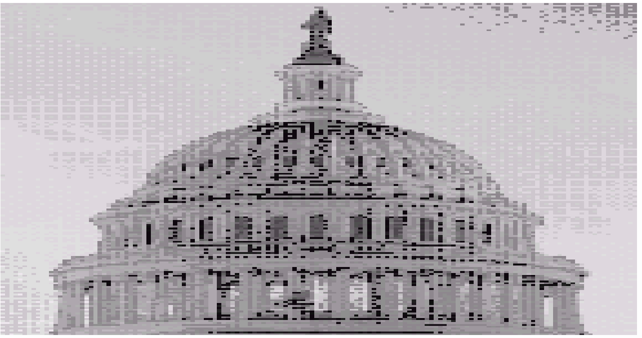
original image



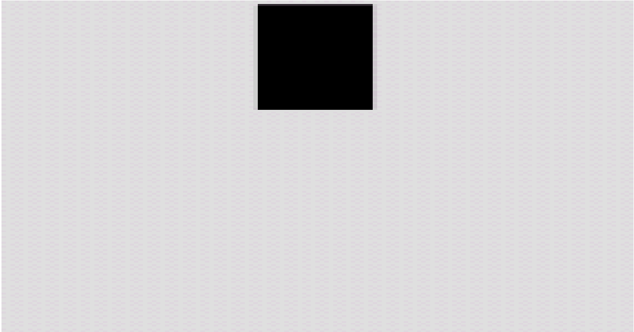
AND image mask

result of AND operation

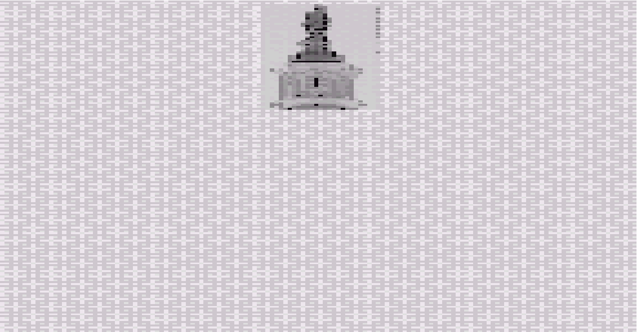
Example of OR Operation



original image



OR image mask



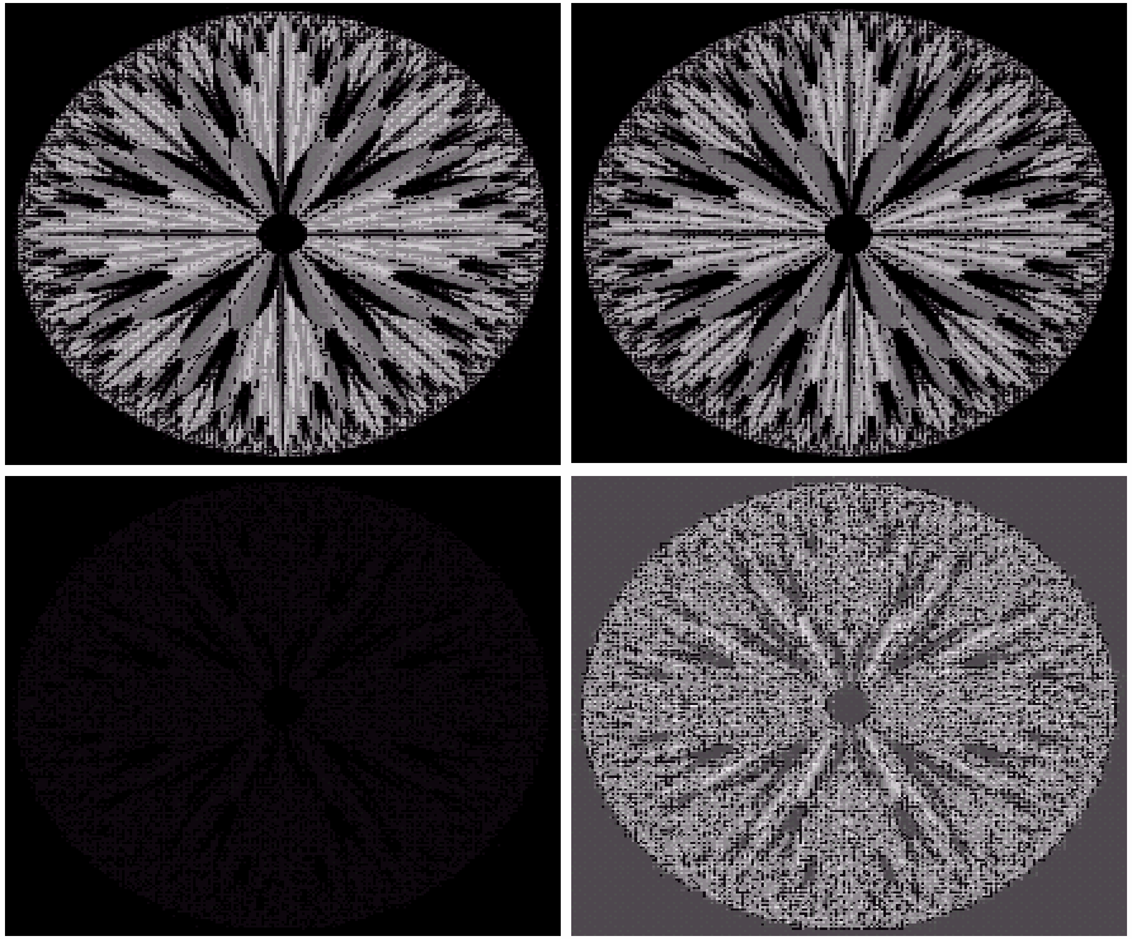
result of OR operation

Image Subtraction

$$g(x,y) = f(x,y) - h(x,y)$$

Image Subtraction

a	b
c	d



a). original fractal image

b). result of setting the four lower-order bit planes to zero
- refer to the bit-plane slicing
- the higher planes contribute significant detail
- the lower planes contribute more to fine detail
- image b). is nearly identical visually to image a), with a very slightly drop in overall contrast due to less variability of the gray-level values in the image.

c). difference between a). and b). (nearly black)

d). histogram equalization of c). (perform contrast stretching transformation)

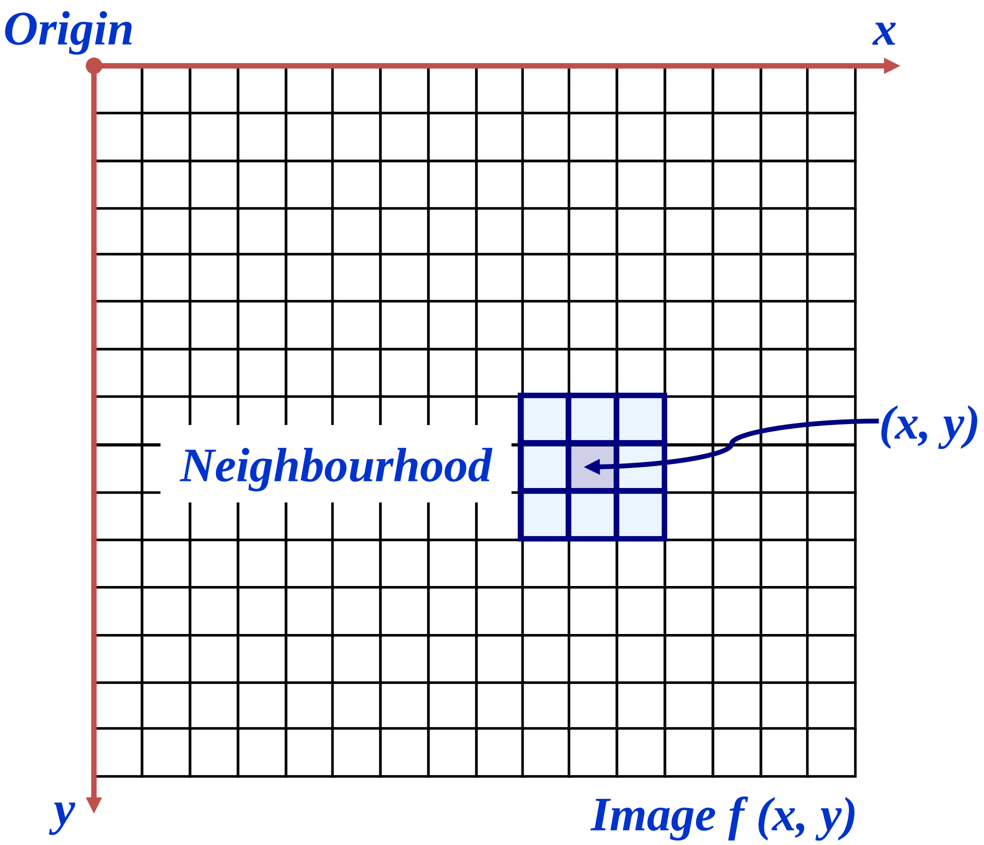
Spatial Filtering

Neighbourhood Operations

- Neighbourhood operations simply operate on a larger neighbourhood of pixels than point operations

- Neighbourhoods are mostly a rectangle around a central pixel

- Any size rectangle and any shape filter are possible



Simple Neighbourhood Operations

- Some simple neighbourhood operations include:
  - **Min:** Set the pixel value to the minimum in the neighbourhood
  - **Max:** Set the pixel value to the maximum in the neighbourhood
  - **Average:** Average of the pixel values
  - **Median:** The median value of a set of numbers is the midpoint value in that set (e.g. from the set [1, 7, 15, 18, 24] 15 is the median).

Simple Neighbourhood Operations Example

Original Image

x

123	127	128	119	115	130
140	145	148	153	167	172
133	154	183	192	194	191
194	199	207	210	198	195
164	170	175	162	173	151

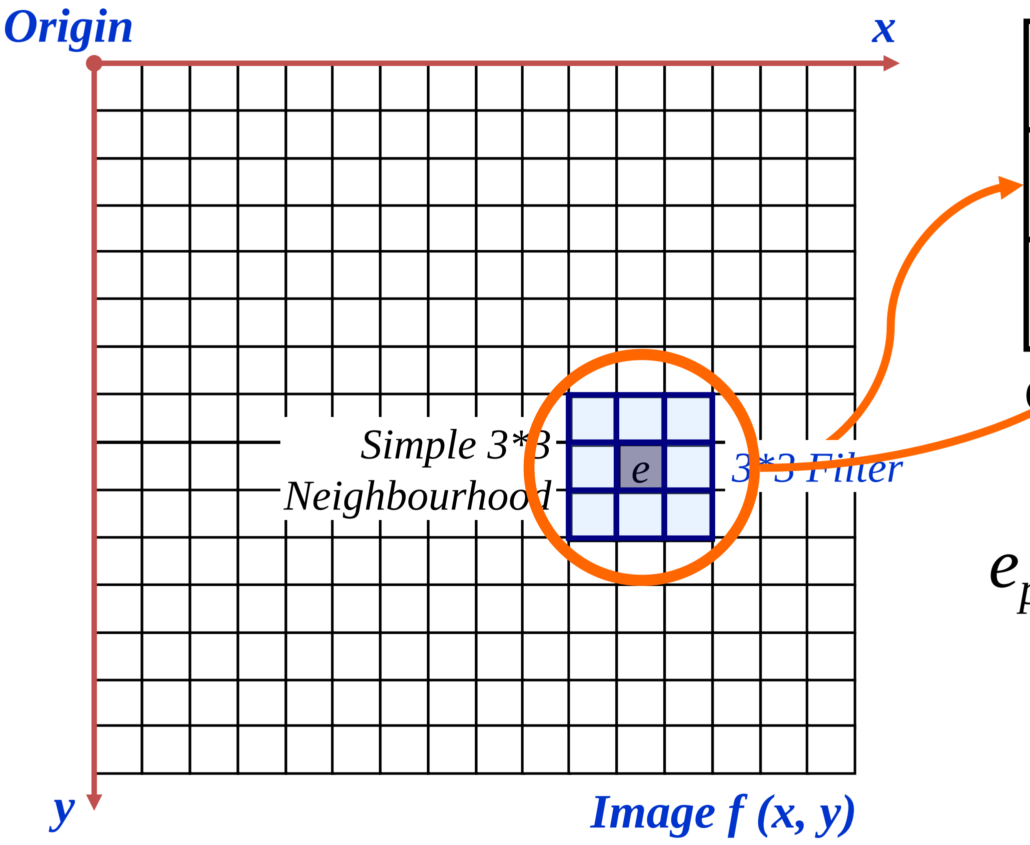
y

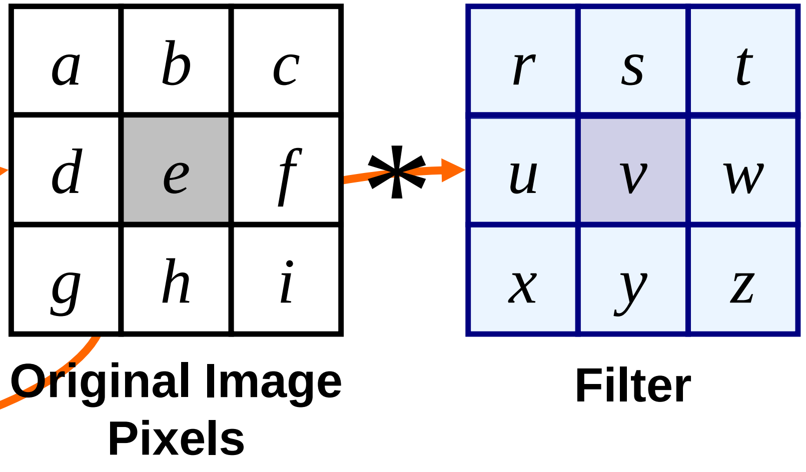
Enhanced Image

x

y

The Spatial Filtering Process

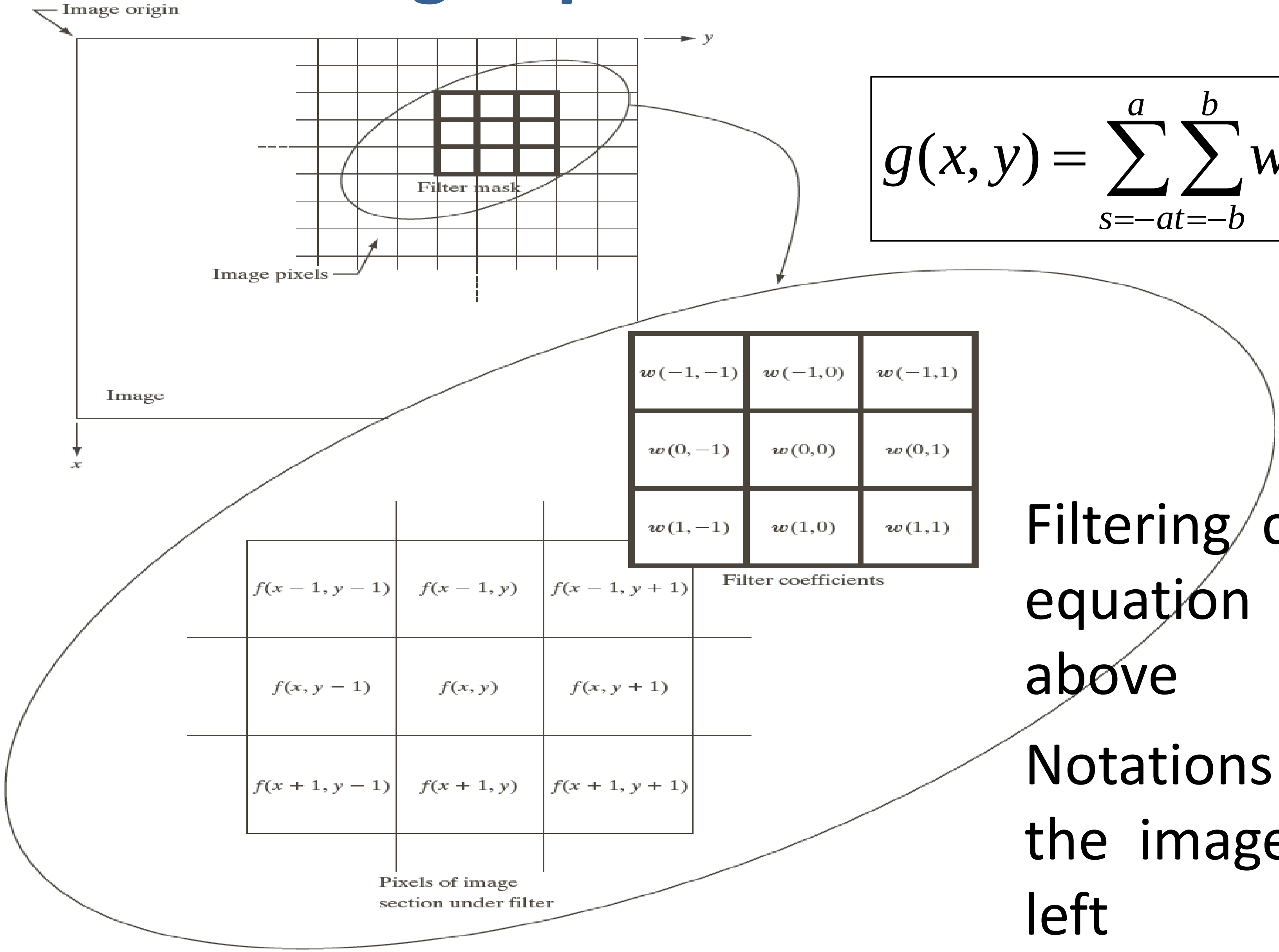




$$
\begin{aligned}
e_{processed} = {} & v*e + \\
& r*a + s*b + t*c + \\
& u*d + w*f + \\
& x*g + y*h + z*i
\end{aligned}
$$

The above is repeated for every pixel in the original image to generate the filtered image

Spatial Filtering: Equation Form



$$\boxed{g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t)}$$

Filtering can be given in equation form as shown above

Notations are based on the image shown to the left

Smoothing Spatial Filters

- Smoothing filters are used for blurring and for noise reduction
- Blurring is used in removal of small details and bridging of small gaps in lines or curves
- Smoothing spatial filters
  - linear filters
  - nonlinear filters

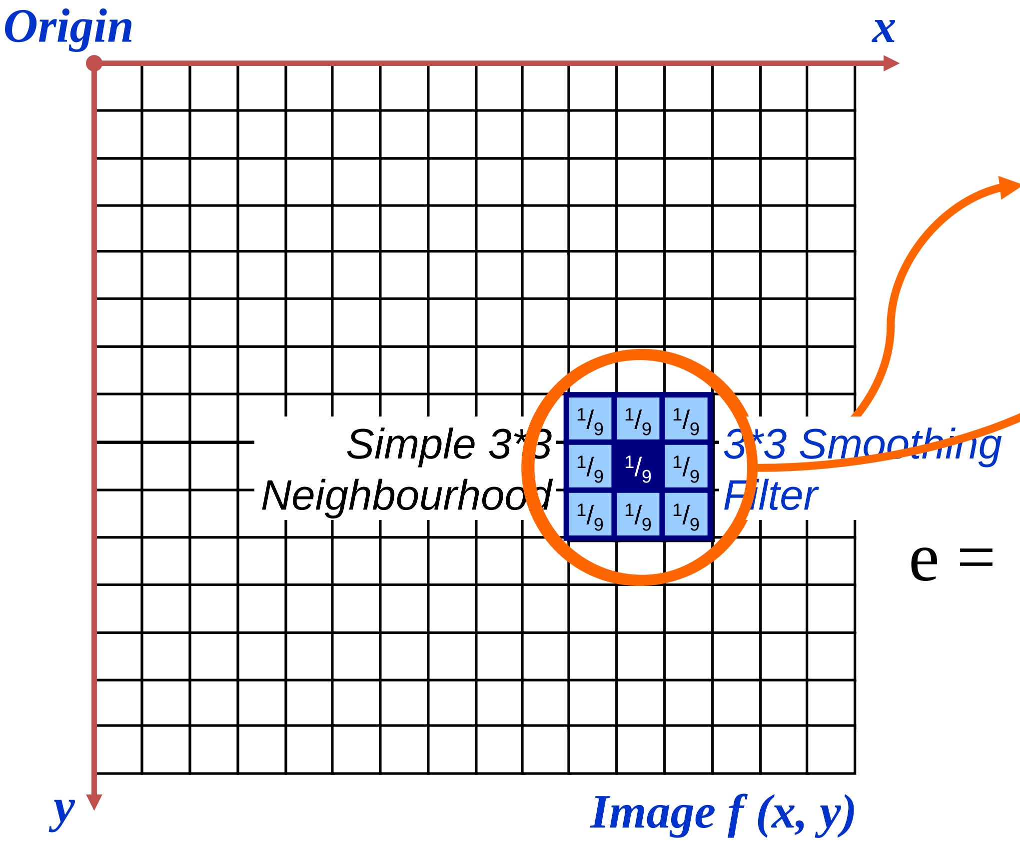
Spatial Smoothing Linear Filters (Average)

- One of the simplest spatial filtering operations we can perform is a smoothing operation
  - Simply average all of the pixels in a neighbourhood around a central value
  - Especially useful in removing noise from images
  - Also useful for highlighting gross detail

$^1/_9$	$^1/_9$	$^1/_9$
$^1/_9$	$^1/_9$	$^1/_9$
$^1/_9$	$^1/_9$	$^1/_9$

Simple averaging filter

Spatial Smoothing Linear Filters



104	*100*	*108*
99	*106*	*98*
95	*90*	*85*

Original Image Pixels

$*$

$^1/_9$	$^1/_9$	$^1/_9$
$^1/_9$	$^1/_9$	$^1/_9$
$^1/_9$	$^1/_9$	$^1/_9$

Filter

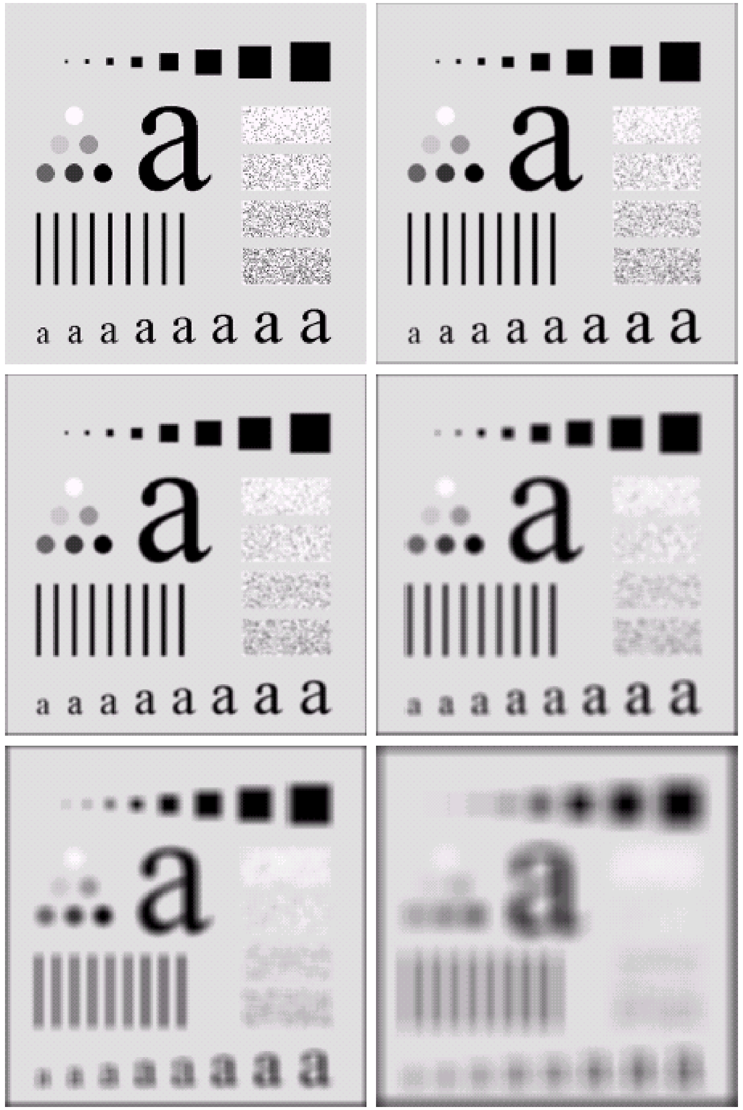
$$
\begin{aligned}
e = {} & ^1/_9*106 + \\
& ^1/_9*104 + {}^1/_9*100 + {}^1/_9*108 + \\
& ^1/_9*99 + {}^1/_9*98 + \\
& ^1/_9*95 + {}^1/_9*90 + {}^1/_9*85 \\
= {} & 98.3333
\end{aligned}
$$

The above is repeated for every pixel in the original image to generate the smoothed image.

Image Smoothing Example

- The image at the top left is an original image of size 500*500 pixels
- The subsequent images show the image after filtering with an averaging filter of increasing sizes

  3, 5, 9, 15 and 35
- Notice how detail begins to disappear



Spatial Smoothing Linear Filters (Weighted Average)

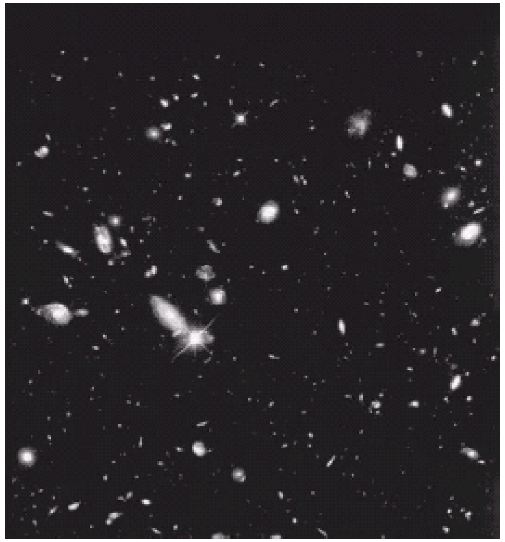
- More effective smoothing filters can be generated by allowing different pixels in the neighbourhood different weights in the averaging function
  - Pixels closer to the central pixel are more important
  - Often referred to as a *weighted averaging*

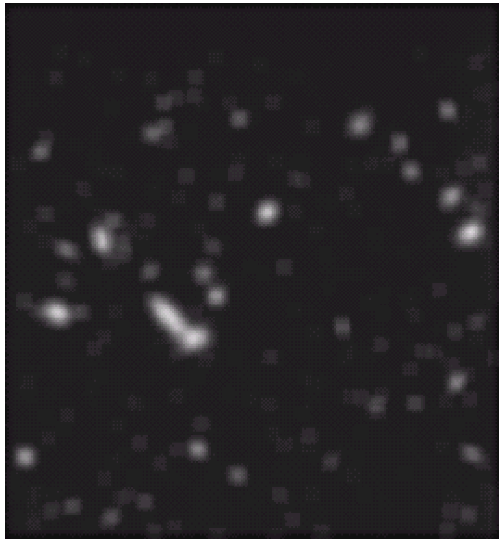
$^1/_{16}$	$^2/_{16}$	$^1/_{16}$
$^2/_{16}$	$^4/_{16}$	$^2/_{16}$
$^1/_{16}$	$^2/_{16}$	$^1/_{16}$

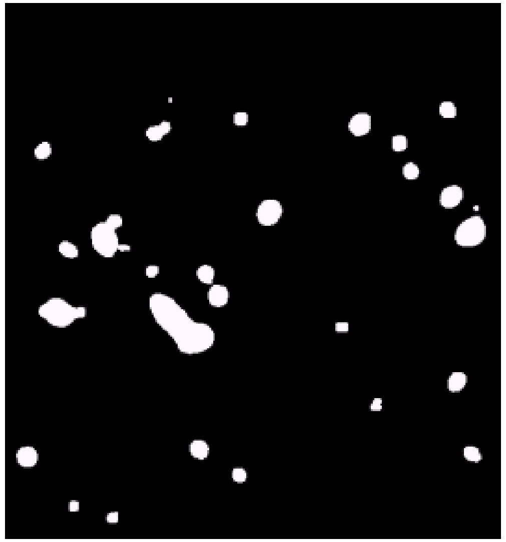
Weighted averaging filter

Another Smoothing Example

By smoothing the original image we get rid of lots of the finer detail which leaves only the gross features for thresholding


Original Image


Smoothed Image


Thresholded Image

Order-statistic (Nonlinear) Filters

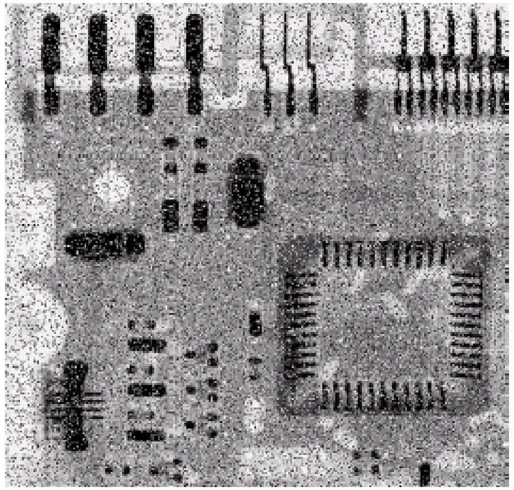
- Nonlinear
- Based on ordering (ranking) the pixels contained in the filter mask
- Replacing the value of the center pixel with the value determined by the ranking result

E.g., median filter, max filter, min filter

Median Filtering

- Median filtering is particularly effective in the presence of impulse noise (salt and pepper noise).
- Unlike average filtering, median filtering does not blur too much image details.

Averaging Filter Vs. Median Filter Example



Original Image
With Noise

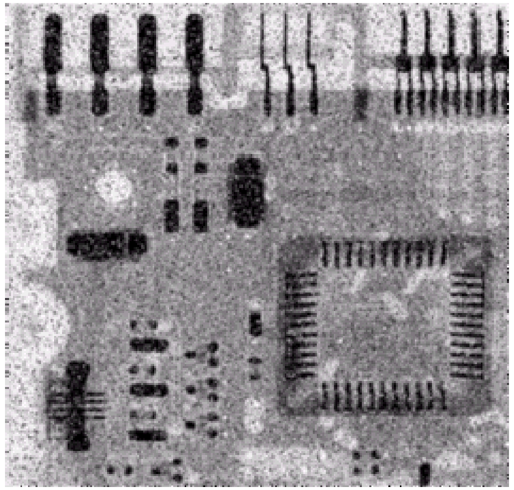


Image After
Averaging Filter

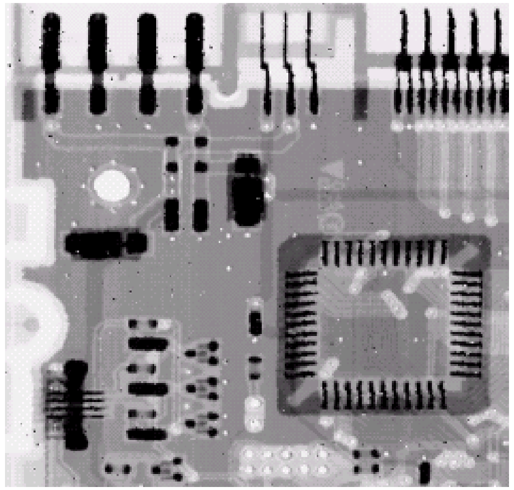


Image After
Median Filter

Filtering is often used to remove noise from images

Sometimes a median filter works better than an averaging filter

Median Filtering

- Advantages:
  - Removes impulsive noise
  - Preserves edges
- Disadvantages:
  - poor performance when # of noise pixels in the window is greater than 1/2 # in the window
  - poor performance with Gaussian noise

Correlation & Convolution

- The filtering we have been talking about so far is referred to as *correlation* with the filter itself referred to as the *correlation kernel*
- *Convolution* is a similar operation, with just one subtle difference

a	b	c
d	e	e
f	g	h

Original Image Pixels

$*$

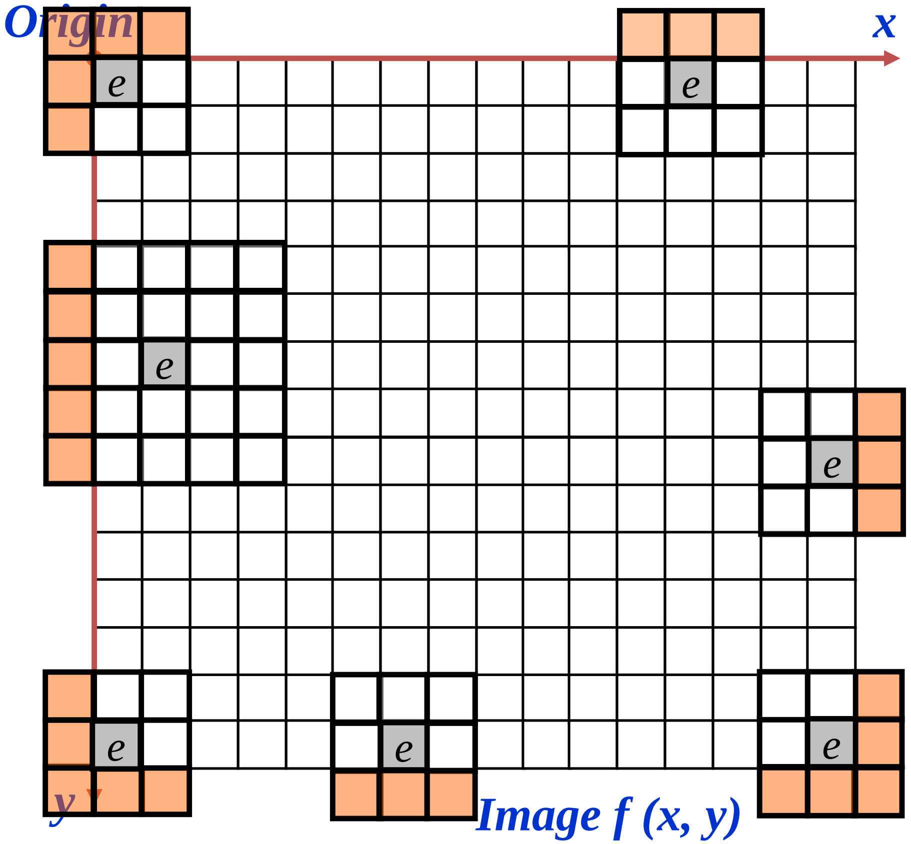
r	s	t
u	v	w
x	y	z

Filter

$$
\begin{aligned}
e_{processed} = {} & v*e + \\
& z*a + y*b + x*c + \\
& w*d + u*e + \\
& t*f + s*g + r*h
\end{aligned}
$$

Strange Things Happen At The Edges!

At the edges of an image we are missing pixels to form a neighbourhood

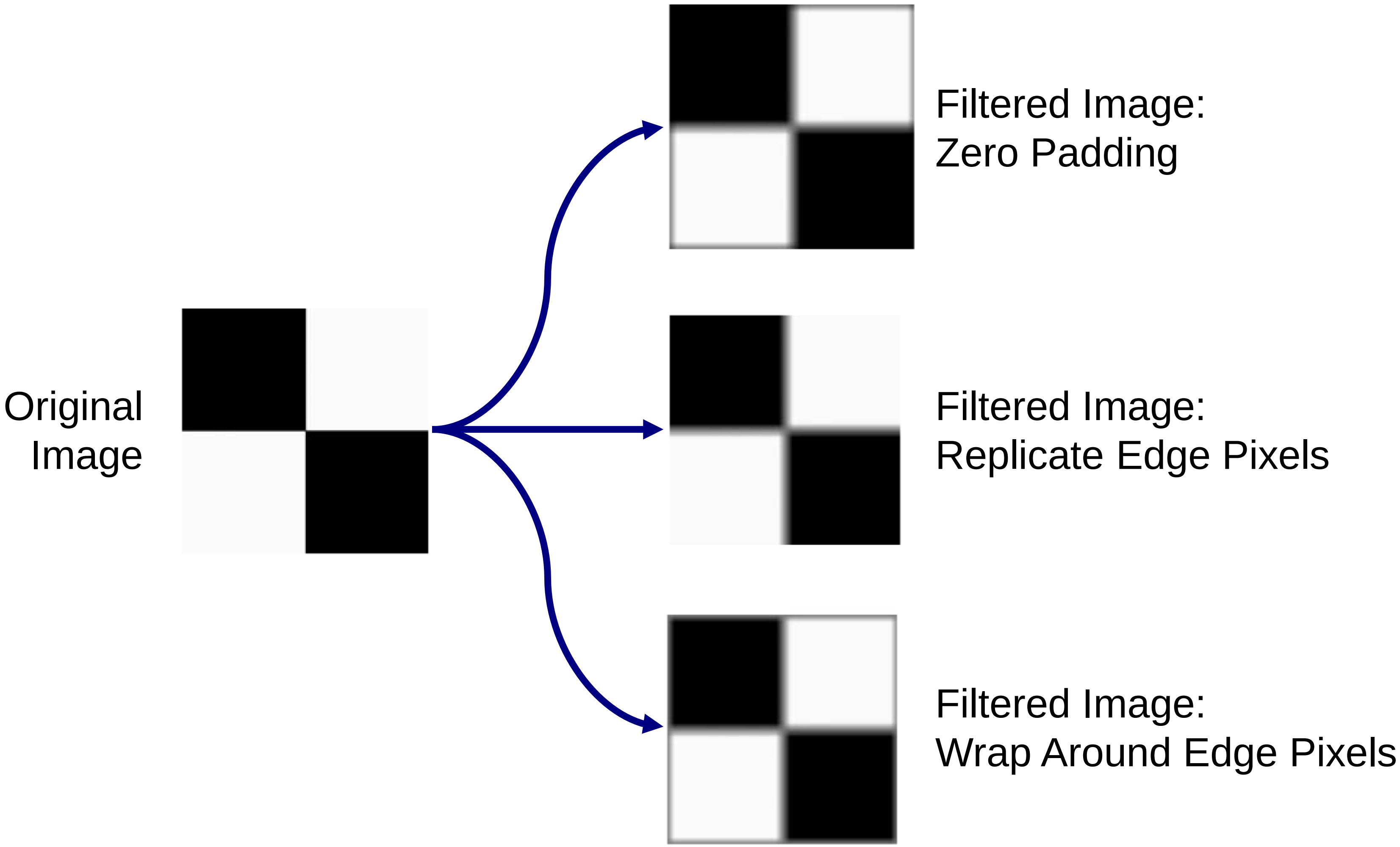


Strange Things Happen At The Edges! (cont…)

There are a few approaches to dealing with missing edge pixels:

- Omit missing pixels
  - Only works with some filters
  - Can add extra code and slow down processing
- Pad the image
  - Typically with either all white or all black pixels
- Replicate border pixels
- Truncate the image
- Allow pixels *wrap around* the image
  - Can cause some strange image artefacts

Strange Things Happen At The Edges! (cont…)



Spatial Filtering for Image Sharpening

Spatial Filtering for Image Sharpening

Background: To highlight fine detail in an image or to enhance blurred detail

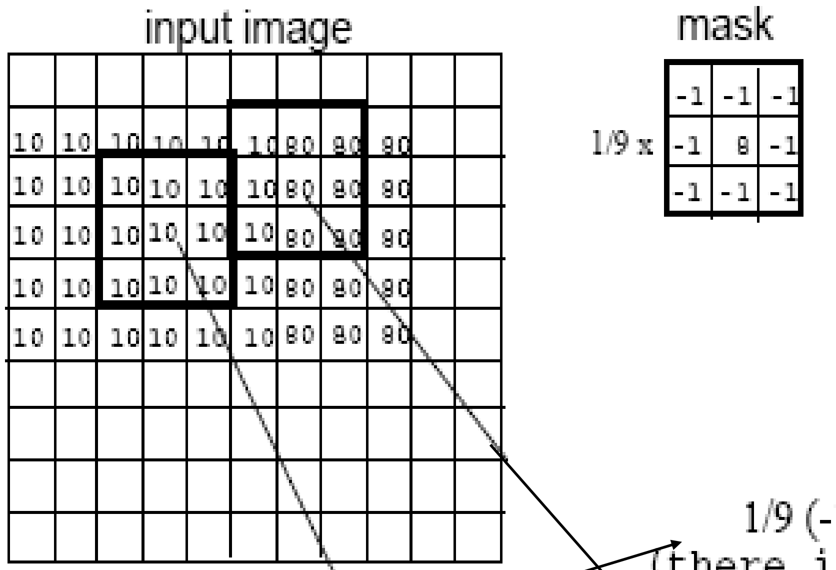
Applications: electronic printing, medical imaging, industrial inspection, autonomous target detection (smart weapons) etc.,

Foundation (Blurring vs Sharpening):

- Blurring/smoothing is performed by spatial averaging (equivalent to integration)
- Sharpening is performed by noting only the gray level changes in the image that is the differentiation

Spatial Filtering for Image Sharpening

- Operation of Image Differentiation
  - Enhance edges and discontinuities (magnitude of output gray level >>0)
  - De-emphasize areas with slowly varying gray-level values (output gray level: 0)



1/9 (-10 - 10 - 10 -10 + 80 -10 -10 -10 -10) = 0
(there is no variation in the gray-levels)

1/9 (-10 - 80 - 80 -10 + 640 -80 -10 -80 -80) = 210/9 > 0
(there is variation in the gray-levels)

Common Sharpening Filters

- Unsharp masking
- High Boost filter
- Derivative

Sharpening Filters: Unsharp Masking

- Obtain a sharp image by subtracting a lowpass filtered (i.e., smoothed) image from the original image:

$$Highpass = Original - Lowpass$$

(with some contrast enhancement)

Sharpening Filters: High Boost

- Image sharpening emphasizes edges but details are lost.
- **Idea**: amplify input image, then subtract a lowpass image.

$$\begin{aligned} Highboost &= A\ Original - Lowpass \\ &= (A-1)\ Original + Original - Lowpass \\ &= (A-1)\ Original + Highpass \end{aligned}$$

(A-1) 

+ 

= 

Sharpening Filters: High Boost

- If **A=1**, the result is unsharp masking.
- If **A>1**, part of the original image is added back to the high pass filtered image.

$$\textbf{High boost} = (A-1)\ Original + Highpass$$

One way to implement high boost filtering is using these masks:

A>=1
w = 9A-1

-1	-1	-1
-1	w	-1
-1	-1	-1

A=2
w = 17

-1	-1	-1
-1	17	-1
-1	-1	-1

Derivatives

- First Order Derivative
  - A basic definition of the first-order derivative of a one-dimensional function f(x) is the difference

$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

- Second Order Derivative
  - Similarly, we define the second-order derivative of a one-dimensional function f(x) is the difference

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x)$$

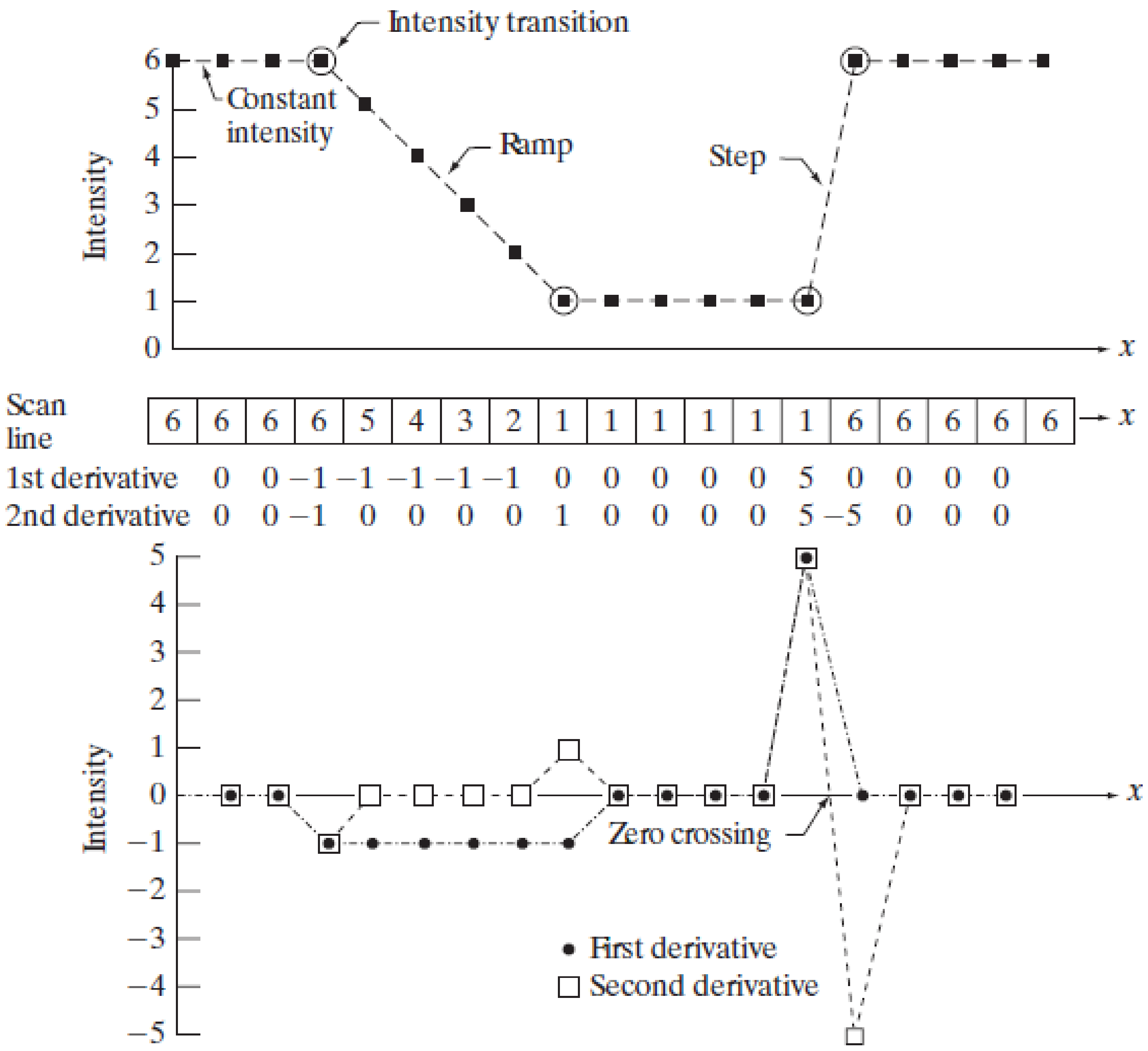
Properties of Derivatives

First Derivative

- Must be zero in the areas of constant intensity
- Must be non-zero at the onset of an intensity ramp or step
- Must be *non-zero* along ramps

Second Derivative

- Must be zero in the areas of constant intensity
- Must be non-zero at the onset and end of a ramp or step
- Must be zero along ramps of constant slope



a
b
c

FIGURE 3.36
Illustration of the first and second derivatives of a 1-D digital function representing a section of a horizontal intensity profile from an image. In (a) and (c) data points are joined by dashed lines as a visualization aid.

Sharpening Filters: Gradient (1st derivative)

- The derivative of an image results in a sharpened image.
- Image derivatives can be computed using the gradient.
- For a function $f(x, y)$ the gradient of f at coordinates (x, y) is given as the column vector:

$$\nabla \mathrm{f} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f}{\partial x} \\ \dfrac{\partial f}{\partial y} \end{bmatrix}$$

1st Derivative Filtering

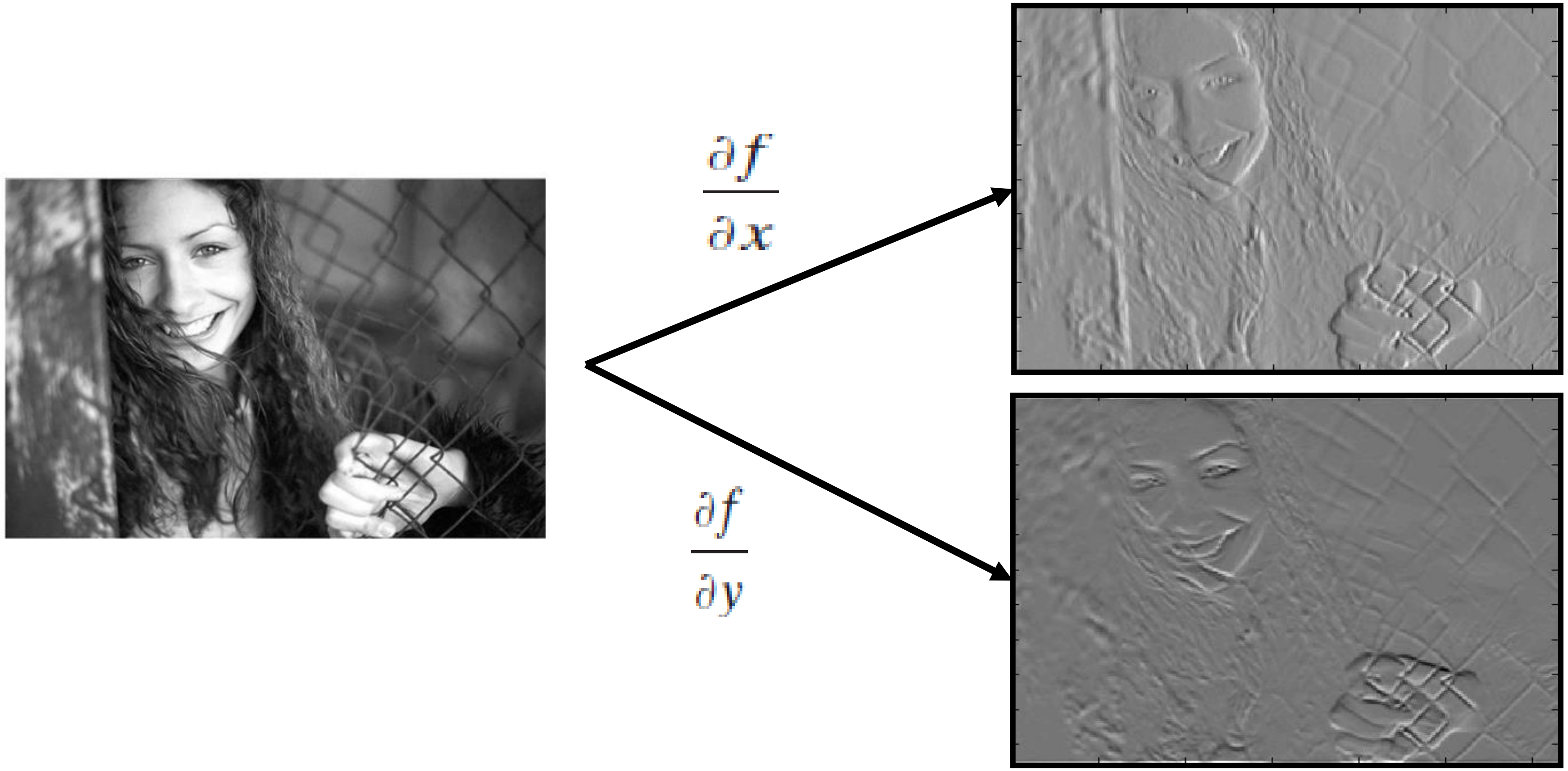
- The magnitude and direction of this vector is given by:

$$\nabla f = mag(\nabla \mathrm{f})$$
$$= \left[G_x^2 + G_y^2\right]^{1/2}$$
$$= \left[\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2\right]^{1/2}$$

$$direction(grad(f)) = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

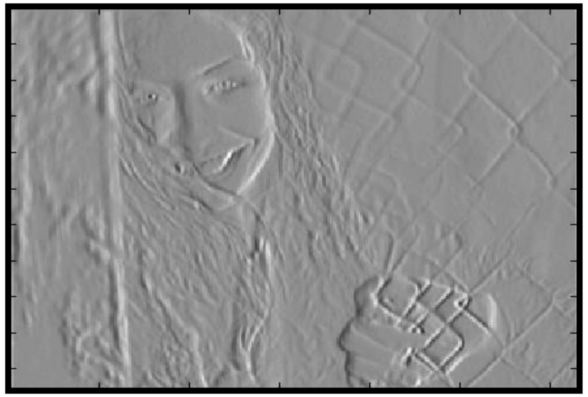
Example: visualize partial derivatives

- The gradient can be visualized by mapping the values to [0, 255]

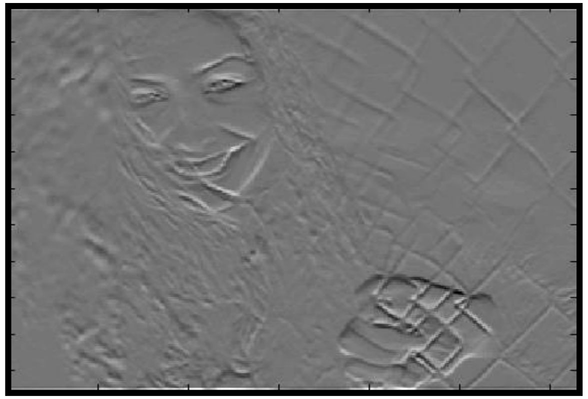


Example: visualize partial derivatives

$$\frac{\partial f}{\partial x}$$



$$\frac{\partial f}{\partial y}$$



Gradient Magnitude

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

(isotropic, i.e.,
edges in all directions)

Implement Gradient Using Masks

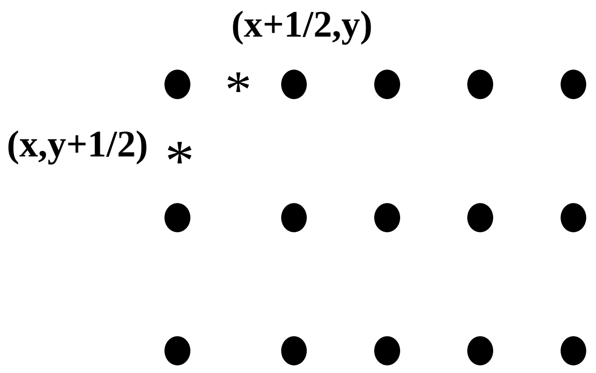
- We can implement $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial y}$ using masks:

-1	1

1
-1

$\frac{\partial f}{\partial x}$ good approximation at (x+1/2,y)

$\frac{\partial f}{\partial y}$ good approximation at (x,y+1/2)



$$\frac{\partial f}{\partial x} \quad f(x+1,y)-f(x,y)$$

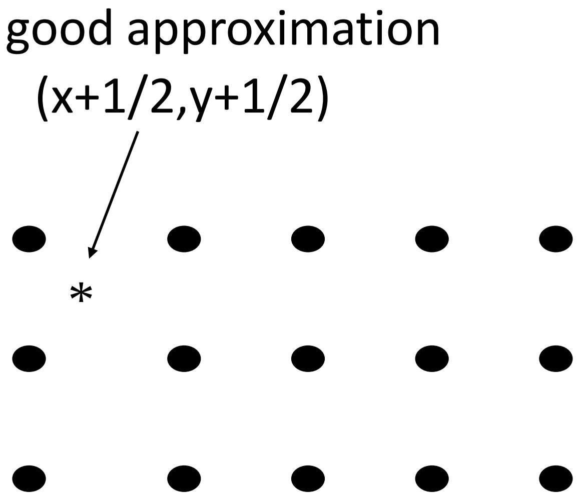
$$\frac{\partial f}{\partial y} \quad f(x,y)-f(x,y+1)$$

Implement Gradient Using Masks

- A different approximation of the gradient:

$$\frac{\partial f}{\partial x}(x,y) = f(x,y) - f(x+1,y+1)$$

$$\frac{\partial f}{\partial y}(x,y) = f(x+1,y) - f(x,y+1),$$



1st Derivative Filtering

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

Roberts Cross Gradient

$$G_x = (z_9 - z_5) \quad G_y = (z_8 - z_6)$$

−1	0
0	1

0	−1
1	0

Sobel Gradient

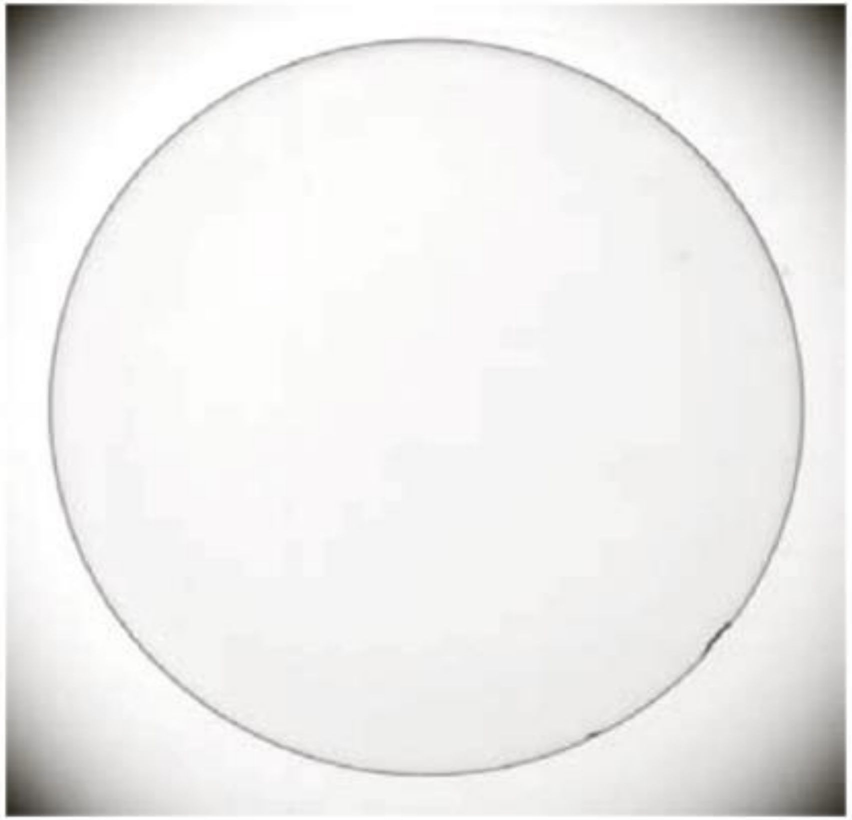
$$G_x = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

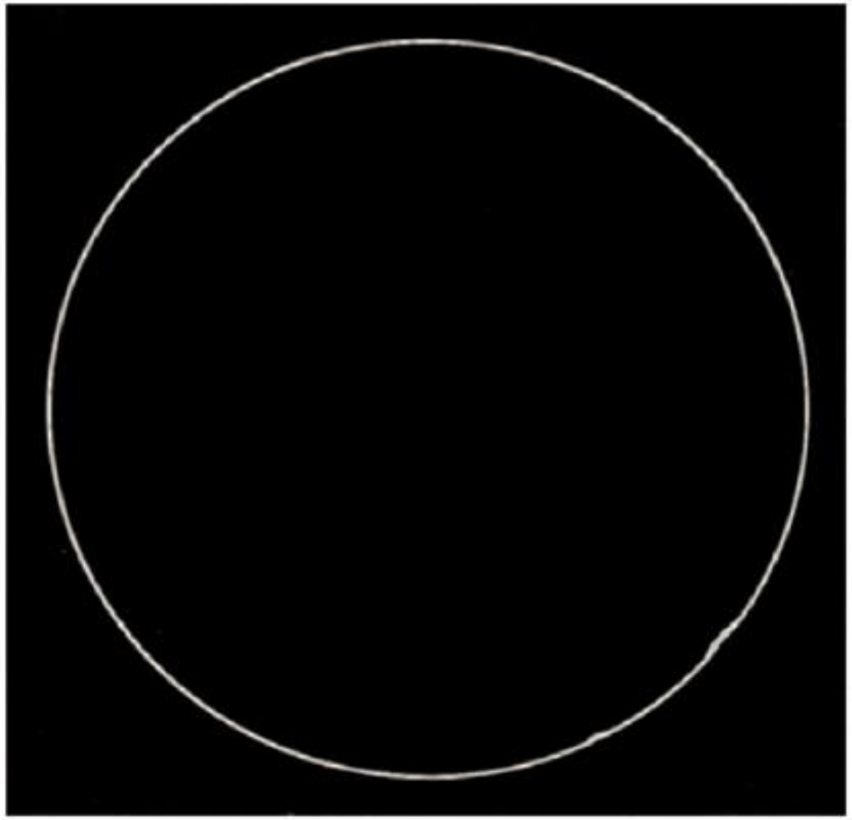
$$G_y = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

Example





Second Order Derivative

Laplacian

- Laplacian as an isotropic Enhancer:

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

$$\frac{\partial^2 f}{\partial x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$

$$\frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y)$$

$$\nabla^2 f = \left[f(x+1,y) + f(x,y+1) + f(x-1,y) + f(x,y-1) - 4f(x,y) \right]$$

Laplacian

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \; 90^\circ \textit{ isotropic} \qquad \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \; 45^\circ \textit{ isotropic}$$

Laplacian

0	1	0
1	−4	1
0	1	0

1	1	1
1	−8	1
1	1	1

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

a	b
c	d

FIGURE 3.37 (a) Filter mask used to implement Eq. (3.6-6). (b) Mask used to implement an extension of this equation that includes the diagonal terms. (c) and (d) Two other implementations of the Laplacian found frequently in practice.


Thank you!