Digital Image Processing

Lecture 2
Intensity Transformation

Rahul JAIN

Spring 2022

Outline

- Sampling and quantization of images
- Some basic image processing methods
  - Negative transformation
  - Log transformations
  - Power-Law (Gamma) Transformations
  - Histogram equalization

Intensity of images

- The value or magnitude of f at spatial coordinates (x, y) is a positive scalar quantity.
- While its intensity values are propositional to energy radiated from a physical source (device).
- The **intensity** of a grayscale (monochrome) image f at (x, y) is the gray level $l(x, y)$ of the image at that point:

$$l(x, y) = i(x, y)r(x, y) \Leftrightarrow l = i \cdot r$$

  - i : amount of source illumiation incident on the scene being viewed (illumination)
  - r : amount of illumination reflected by the objects in the scene (reflection)

- The gray level l lies in the range:

$$L_{\min} \leq l \leq L_{\max}$$

  - In practice, $L_{min} = i_{min}r_{min}$ and $L_{max} = i_{max}r_{max}$
  - L_{min} should be positive and L_{max} should be finite

- This interval $[L_{min}, L_{max}]$ is called the **gray level (intensity scale)**

Intensity of images

- The range of interval $l = [L_{min}, L_{max}]$ is normally $[0, L-1]$:

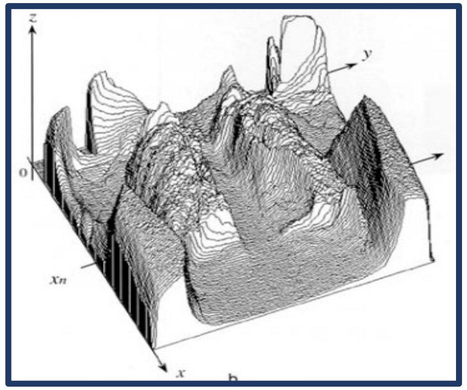
  where $l = L - 1$ is considered white and l=0 is considered black

- All intermediate values are gray and can be changed from black to white
- We can simply define intensity as the brightness per unit area of an image

brightness per unit area

Sampling and Quantization

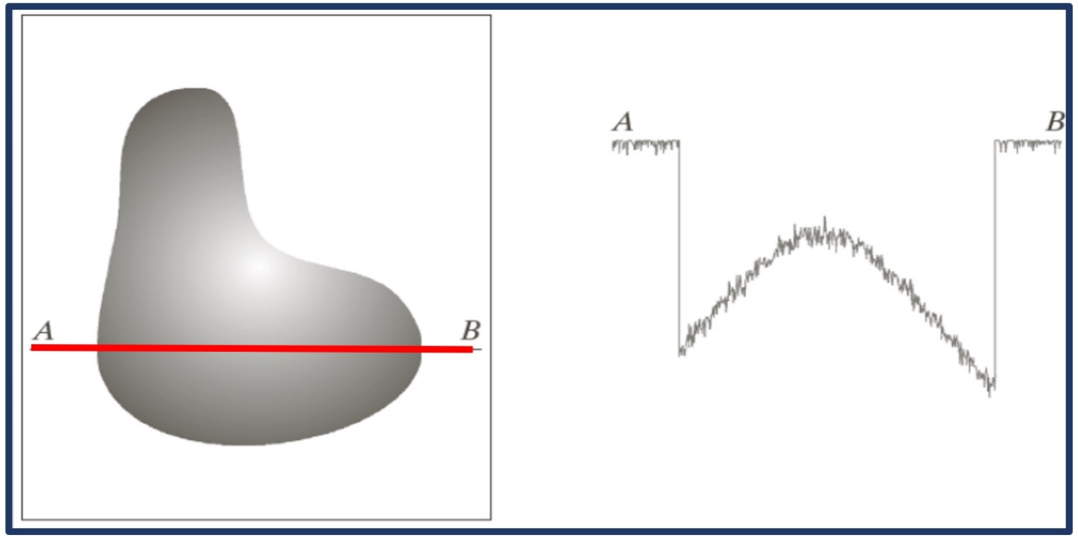
- An analog image is continuous with respect to the *X*- and *Y*-coordinates and also in amplitude
- To create a digital image, we need to convert the continuous sensed data into digital form



- We can sample an image in both coordinates and amplitude to make it relevant for computer processing
- The process of digitizing coordinate values is known as sampling
- The process of digitizing amplitude values is known as quantization

Sampling and Quantization

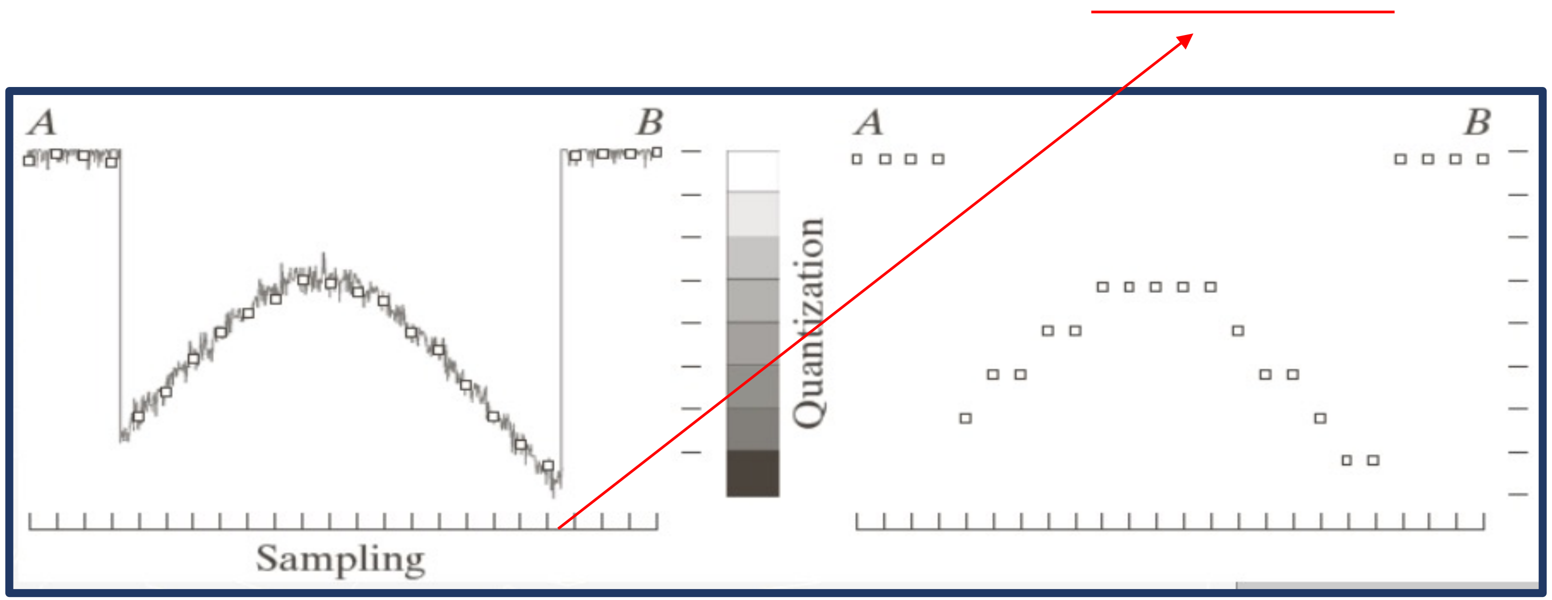
- Figure shows the sampling and quantization process



- **Left:** A continuous image
- **Right:** The one-dimensional function is a plot of the intensity levels (amplitude values) of the continuous image along the line segment *AB*

Sampling

- We take equally spaced samples along line *AB* to sample this function.
- The spatial location of each sample is indicated by a vertical mark



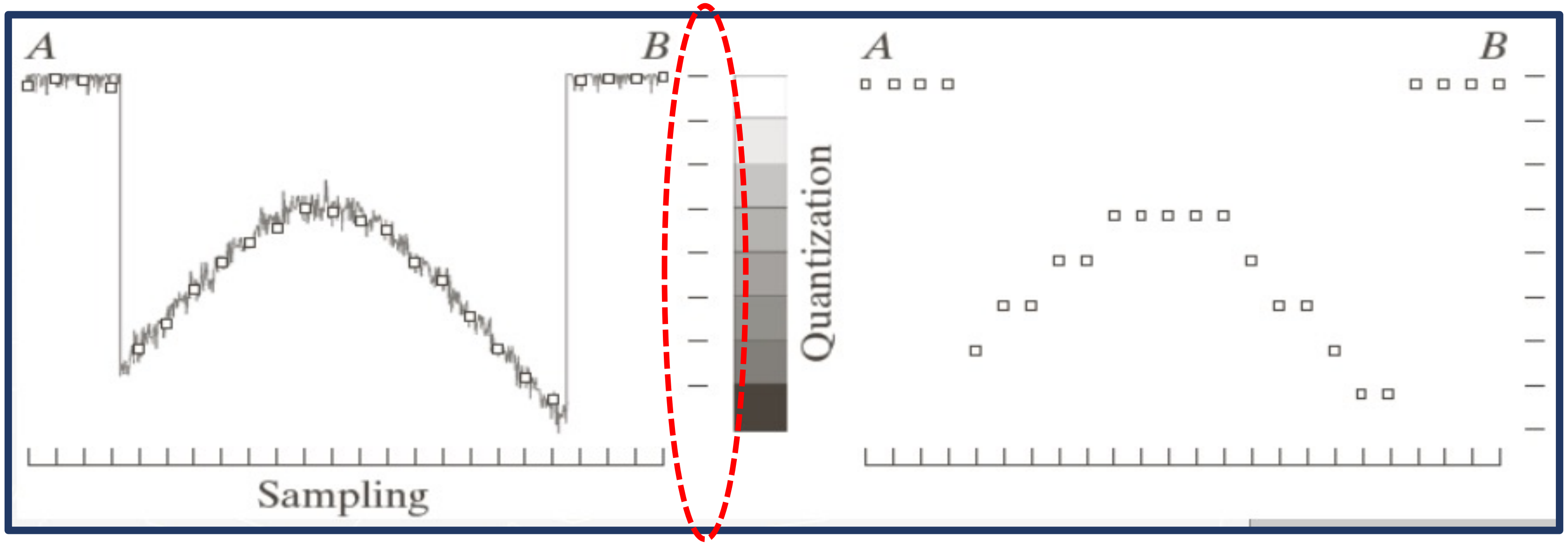
▪ Small squares are mapped on the function to represent the samples

▪ The set of discrete locations gives the sampled function

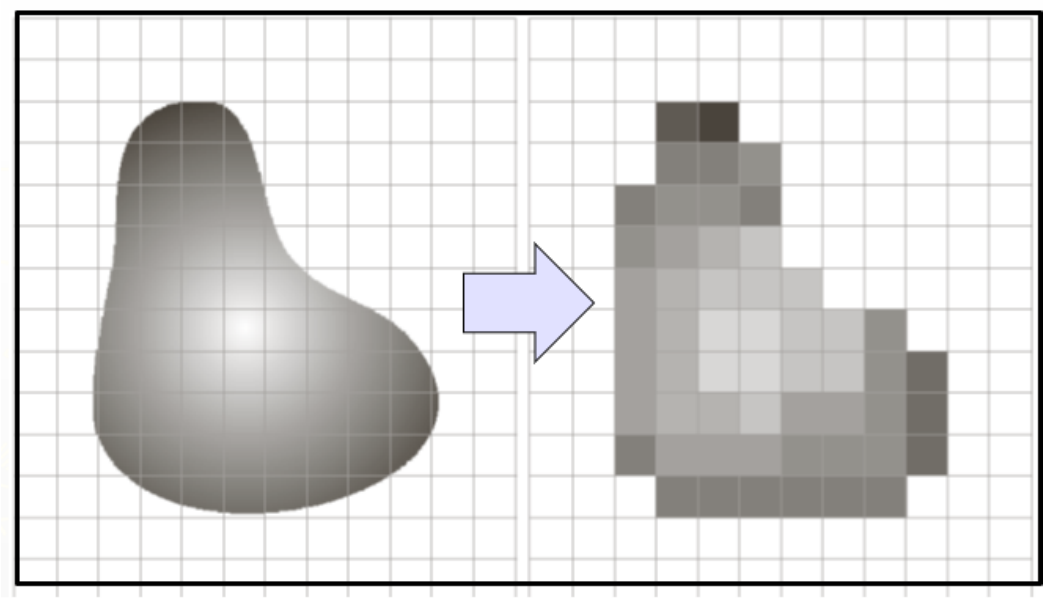
▪ These samples are discrete, but their intensity values are still continuous

Quantization

- Quantization is the process of converting sample intensity values into discrete quantities
- To form a digital function, we can divide the intensity scale into discrete intervals, (e.g., eight) ranging from black to white
- The continuous intensity values are quantized by assigning one of the eight values to each sample
  - i.e., assign a sample to the vertical mark that is closest to it



Sampling and Quantization



- In practice method of sampling is determined by the sensor arrangement used to generate the image
- The samples of digital data obtained through sampling and quantization

Representing Digital Image $f(x, y)$

- Suppose we sample the continuous image into a 2-D array $f(x, y)$
- Equally spaced samples in the form of an array are used to represent a digital image $f(x, y)$:

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix}$$

- There are $M \times N$ elements in this image (pixels), *M* is rows and *N* is rows
- Each element (pixel) has a discrete quantity/value

Sampling and Quantization

- The digitization process requires, M and N and L (the number of gray levels/intensity levels) to be powers of 2:

$$M = 2^k,\ N = 2^n,\ L = 2^m$$

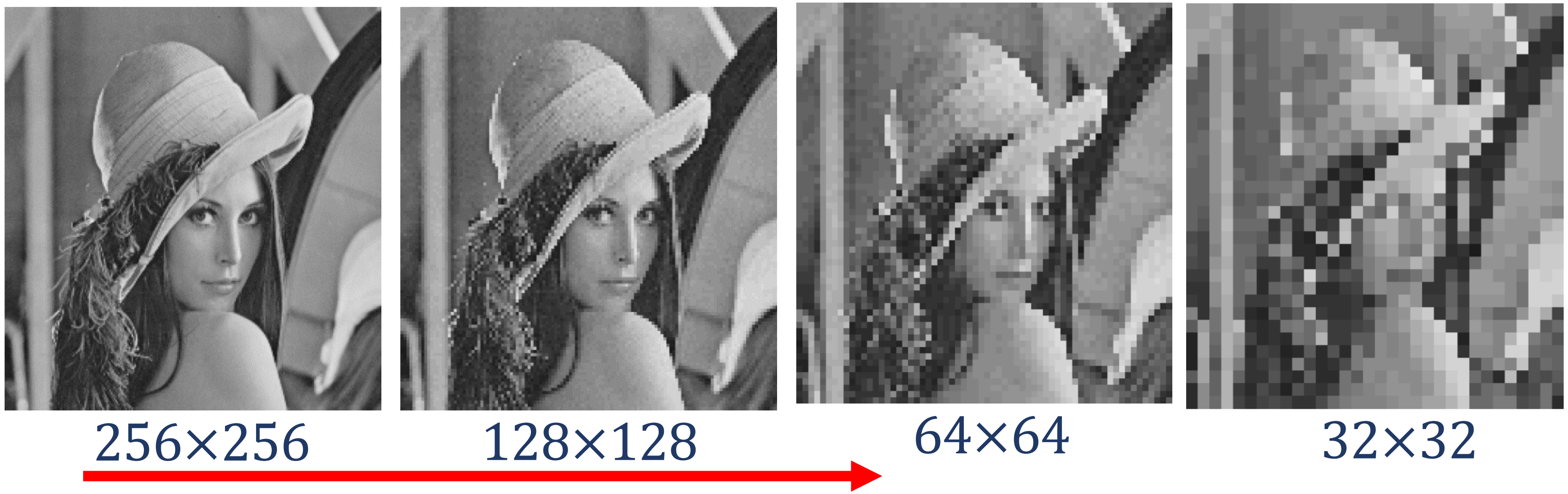
- The number of bits, b, necessary to store f will be

$$b = N \times M \times m.$$

- A 128 $\times$ 128 image with 64 (2^6) gray levels, for example, would take 98,304 (=128 $\times$ 128 $\times$ 6) bits of storage.

Effects of Reducing Spatial Resolution

- Here are some examples of digital images with different M and N (i.e., different spatial resolution)
  - The number of pixels used in the image's construction is referred to as spatial resolution



- Image quality decreases as the spatial resolution is reduced

Effects of Reducing Gray Levels

- What happened if you keep the spatial resolution the same but only reduced the gray level?

256

128

64

32

False contouring

- On the 32-level image, some ridge-like features appeared
- This effect is caused by the implementation of an insufficient number of intensity levels (known as false contouring)
- False contouring presences in parts of a digital image that are smooth

Effects of Reducing Gray Levels

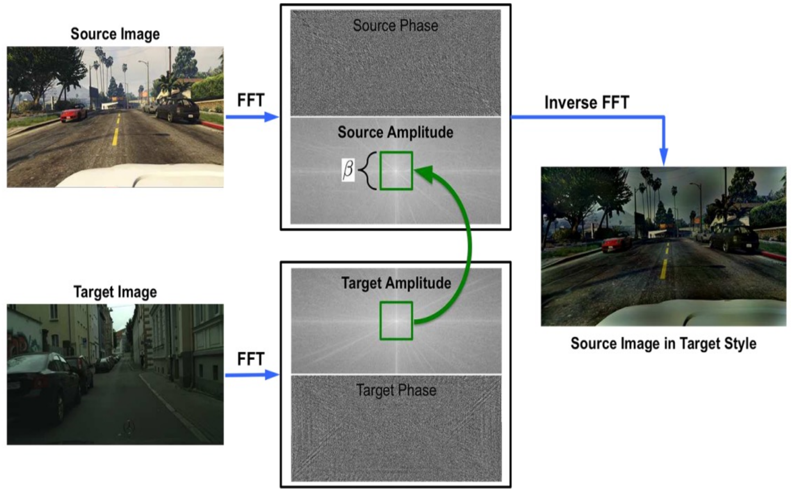
- How many samples (*M*,*N*) and gray levels (*L*) are required for a "good" image approximation?
- These parameters always depend on the image's content.

Spatial and Transform Domain

- There are two categories of digital image processing: Spatial and Transform domains

- Spatial domain: Image processing techniques process the intensity values of the image plane directly

- Transform domain:
  - In this domain, techniques do not directly process the intensity data of the image plane
  - They transform/map an image to a different domain, process it in that domain, and then reverse the transformation to return the results to the spatial domain



Y. Yang and S. Soatto, "FDA: **Fourier** Domain Adaptation for Semantic Segmentation," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020,

In this course, We will only cover Digital Image Processing in the spatial domain

Spatial Domain Processing

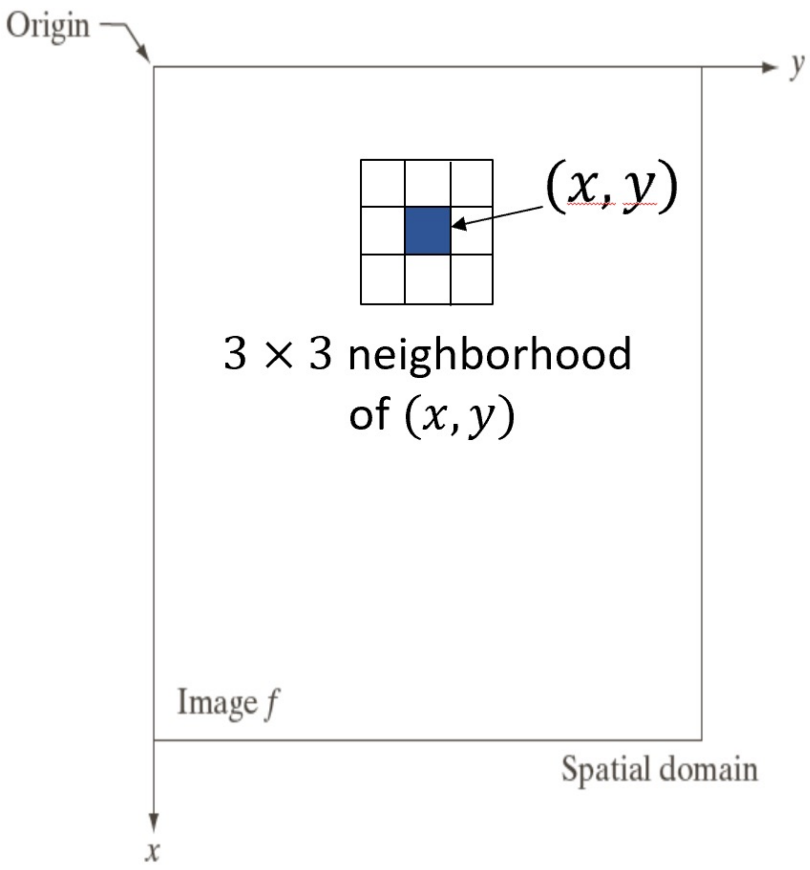
- The spatial domain methods are based on direct manipulation of pixels in an image.
- These methods are usually more computationally efficient and require less processing resources
- As a result, operations in the spatial domain are desirable
- These operations can be denoted in the following way:

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

where $f(x,y)$ is the input image, and $g(x,y)$ is the output image, T is an operator on f defined over a neighborhood of point (x,y)

Neighborhood of a Point

- Assume that (x, y) is single point in the image f
- The neighborhood of (x, y) refers to the region that contains this point
- Normally, the neighborhood is a rectangular region, centered on (x, y), and much smaller than picture f



Point Processing

- The smallest neighborhood of a pixel is 1×1 in size
- In this case, g depends only on the value of f at (x,y)

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

simplified as

$$s = T(r)$$

where, s denotes the output image's intensity and r is the intensity of the input image's intensity at any point (x,y)

- T is a function that transforms intensity

Basic Intensity Transformation Functions

- Intensity transformations are one of the most basic types of visual image processing
- We will go through three basic intensity transformations for image enhancement
  - Negative transformation
  - Log transformations
  - Power-Law (Gamma) Transformations

Negative Transformation

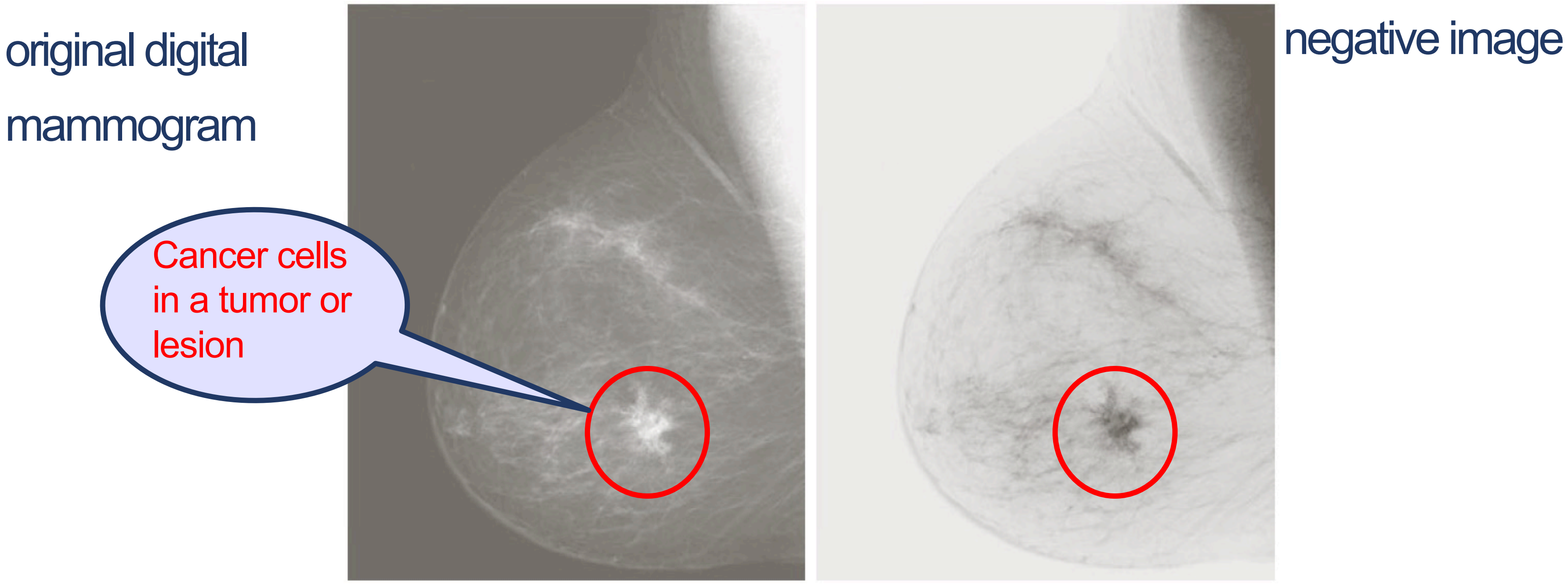
- The term "negative transformation" refers to the process of reversing the intensity of an image
- Assuming the original image is f and that the grey levels are in the range [0, L-1], the negative of f can be calculated using the formula:

$$s = L - 1 - r$$

- In case of $L = 256$, it is used to make the following transformation:

$$\begin{gathered} 0 \to 255,\ 1 \to 254 \\ \vdots \\ 254 \to 1,\ 255 \to 0 \end{gathered}$$

Example of Image Negatives



- In this scenario, analyzing the problem (tumor recognition) in the negative image is relatively easier
- This technique is useful for enhancing white or gray details/features in the focused region of an image

Log Transformations

- The general form of the log transformation is:

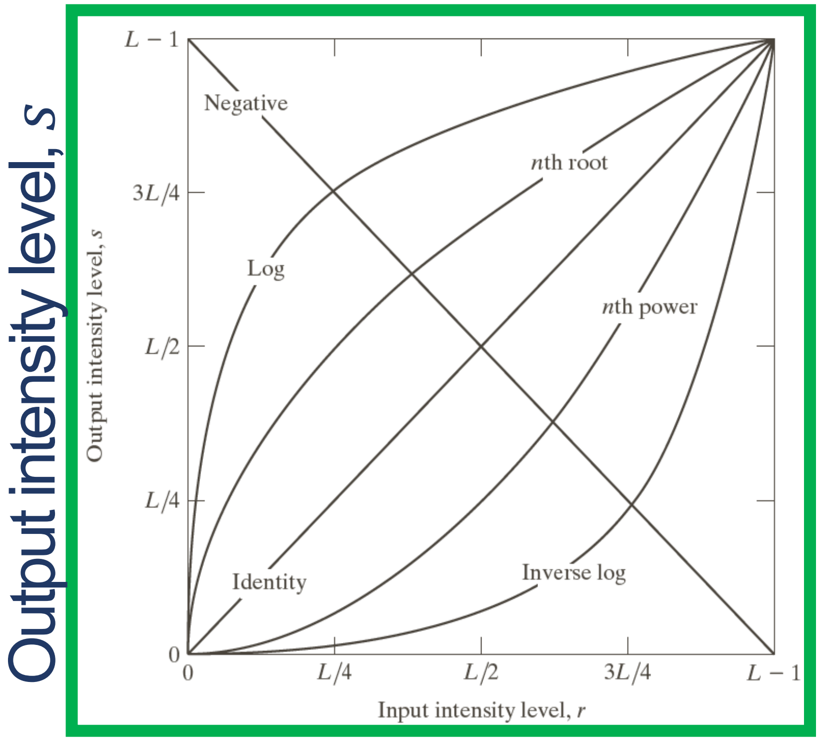
$$s = clog(1 + r)$$

where c is a constant and $r \geq 0$

The shape of the log curve shows that it maps a narrow range of low gray-level input values (intensity) to a wider range of output levels

The opposite is true of higher values of input levels

$s = c\log(1 + r)$



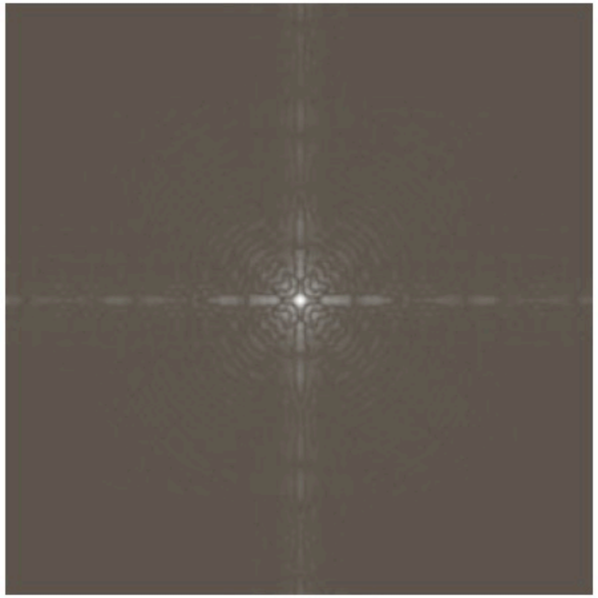
Log Transformations

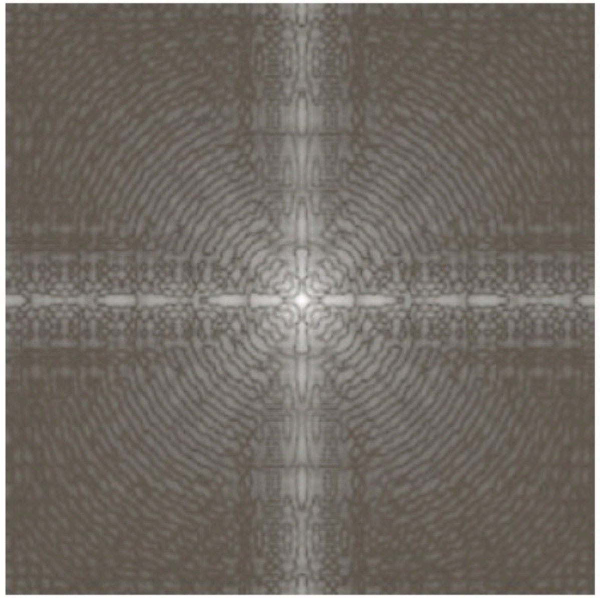
- For example: Fourier spectrum values have the range from 0 to 10^6 or higher, image display devices will not be able to display such a large range of intensity levels in general
- Log transformations are suitable in this situation

We use a log transformation of this type to expand the values of dark pixels in an image while compressing the higher-level values.

Example of Log Transformations

Linearly scaling





Log Transformation with c=1

- **Left:** the spectrum after linearly scaling the values for display in an 8-bit device, a significant degree of intensity information is lost
- **Right:** the same data can be displayed in the same display system after log transformations. It has a lot of information.

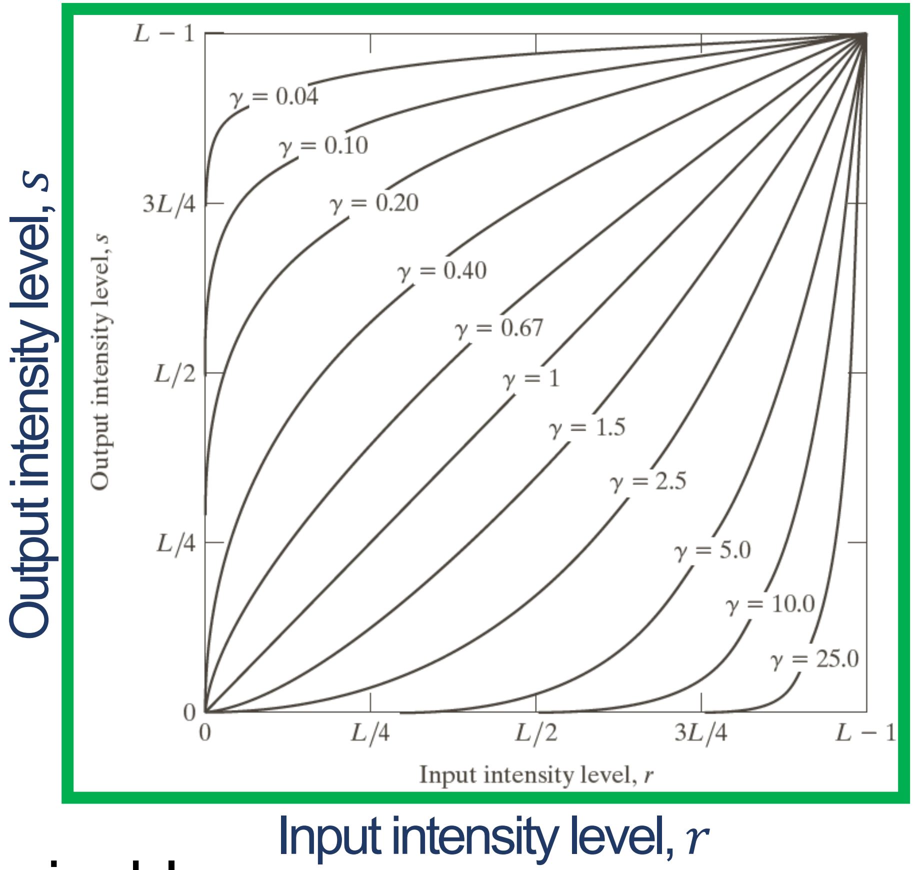
Power-Law (Gamma) Transformations

- The following is the basic form of Power-law transformation:

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

where c and γ are positive constants

- The figure depicts a number of transformation curves that can be generated by changing γ
- Power-Law transformations are similar to log transformations, but they are more customizable



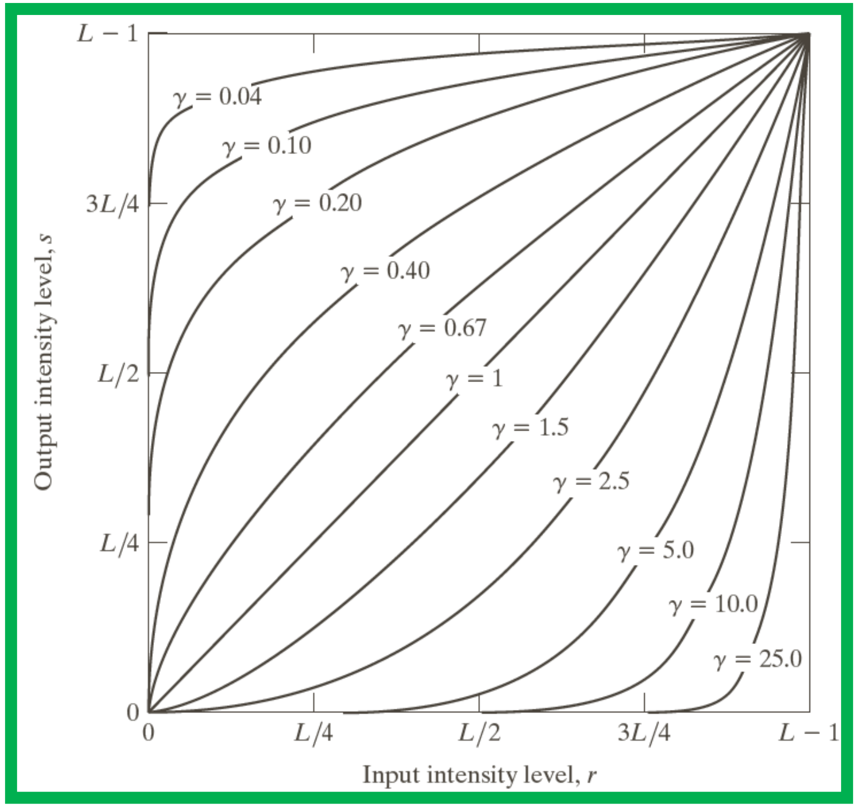
A plot of the equation for various values of γ

Power-Law (Gamma) Transformations

- The curves with the values of $\gamma < 1$ translate a narrow range of dark input values to a wide range of output values

- Curves with the values of $\gamma > 1$, have the exact opposite effect

Output intensity level, s



Input intensity level, r

Example of Contrast Manipulation

- Power-Law transformations are commonly used for contrast manipulation
- Example:
  - The gamma value must be adjusted to enhance the information, otherwise, the image would appear washed-out
  - It can be improved using a power-law transformation with $\gamma > 1$

original image | $\gamma = 3.0$ | $\gamma = 4.0$ | $\gamma = 5.0$

The best result ($\gamma = 4.0$)

Histogram Processing

- A **histogram** of a digital image f with intensities of [0,L-1], is a discrete function:

$$h(r_k) = n_k$$

where r_k is the k^{th} intensity value and n_k is the number of pixels in f with intensity r_k

- An example of histograms
  - x-axis indicates the gray levels [0, L-1]
  - y-axis means the number of pixels in the image with each intensity

4	4	3	3
4	4	3	3
4	1	2	3
0	1	2	3

Image

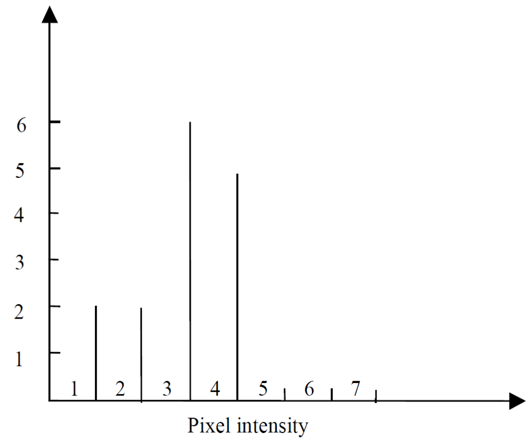


Image Histogram

- A histogram is a graph that displays the distribution of gray-level pixels.

Dark (black) pixels Bright (white) pixels

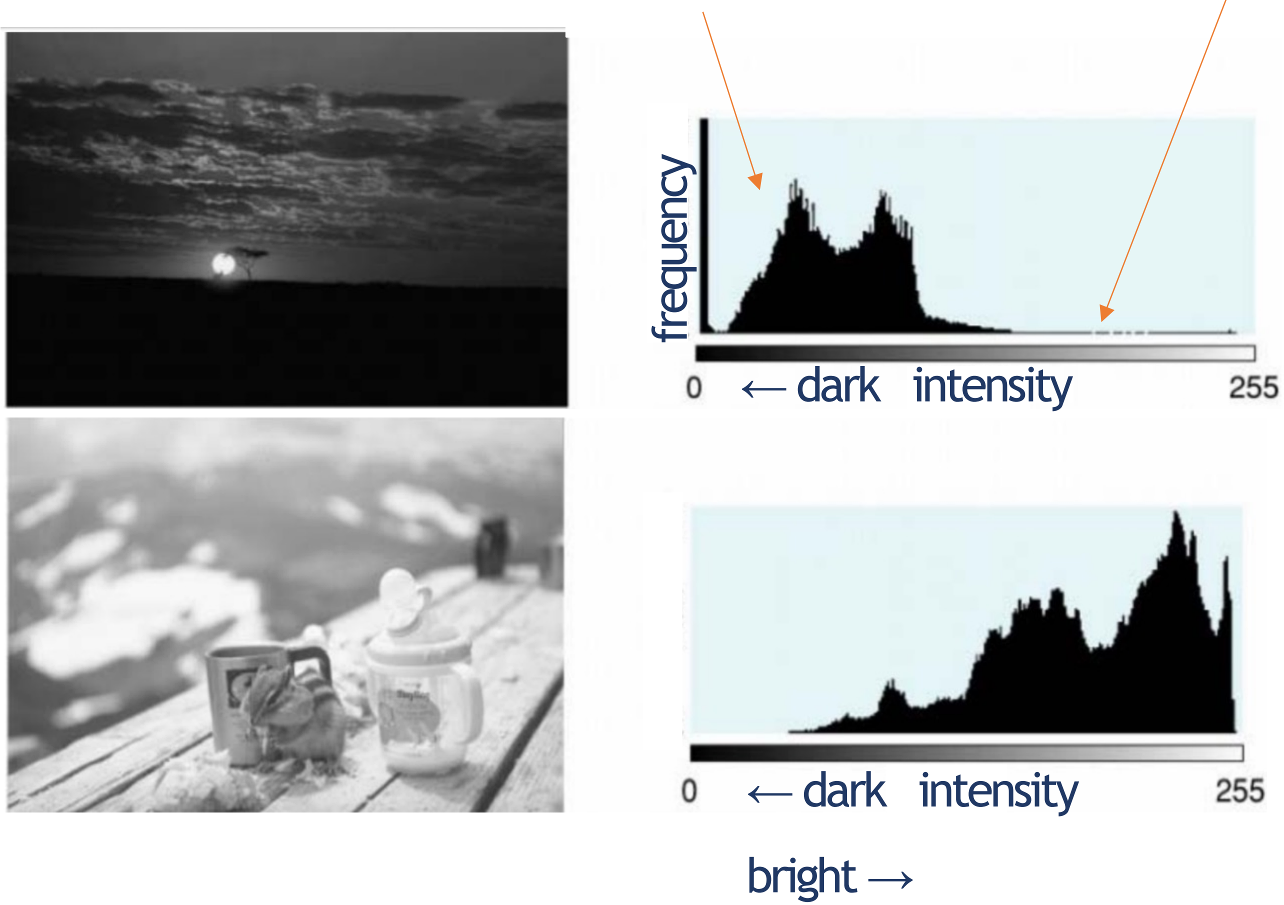


Image Histogram

- Image histograms are the basis for many spatial domain processing methods
- They give a general overview (global description) of how images are represented (what are the features)
- They can be used effectively for
  - Image enhancement, statistics, compression, and segmentation
- Histograms can be easily calculated at a very low cost
  - In real-time image processing, histograms are a common technique

Normalization of Histogram

- Normally, we need to normalize histograms in practice
- Normalization: Normalization is the process of dividing each histogram element by the total number of pixels in the image

$$p(r_k) = n_k / MN \quad \text{for } k = 0,1,2,\ldots,L-1$$

$p(r_k)$ is an estimate of probability of occurrence of intensity level r_k in an image

$$\sum p(r_k) = 1$$

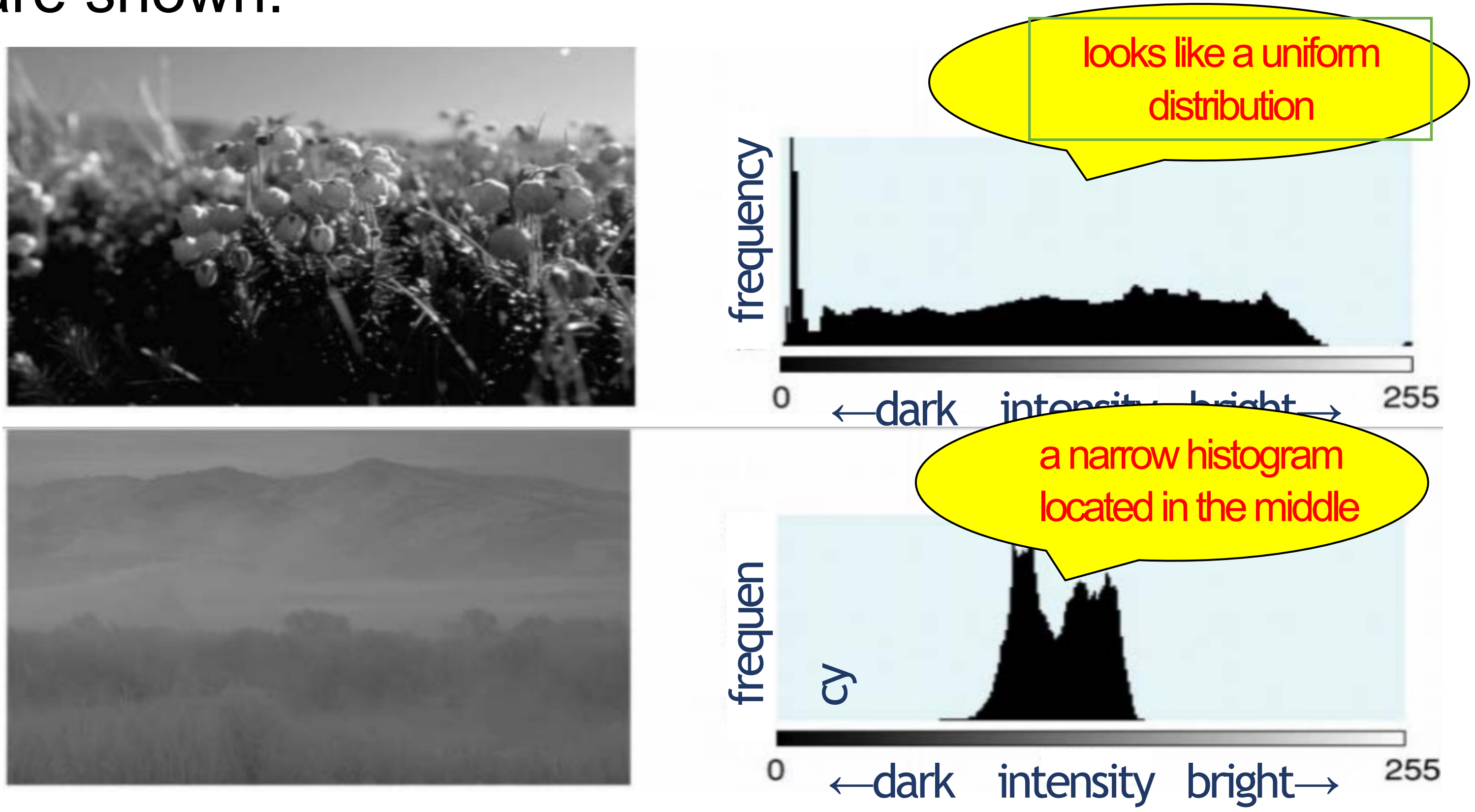
- $p(r_k)$ is probability distribution function *(PDF)*

Histogram Equalization

- Two images are shown.

high contrast

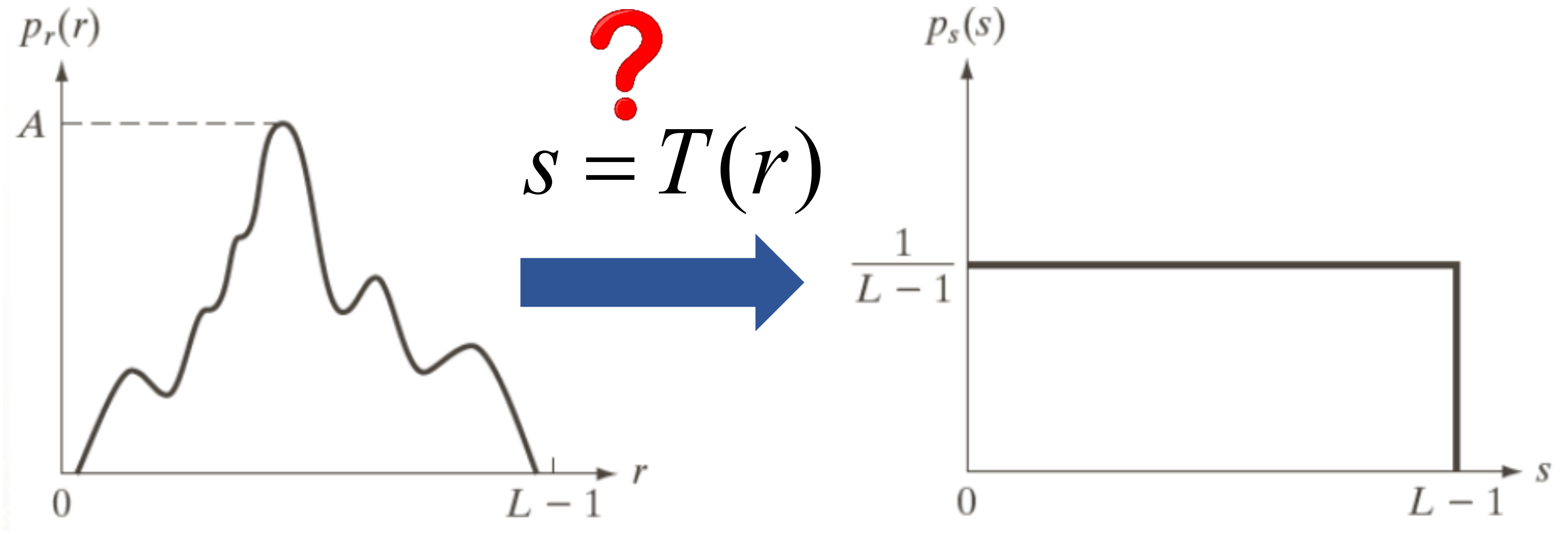
low contrast



- Histogram equalization is a contrast enhancement technique that involves spreading the histogram over a broad region until it resembles a uniform distribution

Histogram Equalization

- Finding an intensity transformation function $s = T(r)$ that can be used as a histogram equalization is a challenge.
- How can an arbitrary histogram be transformed into a uniform distribution, specifically?



Histogram Equalization

- The following formulae can be used for equalization:
  - In continuous case:

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

  - In discrete case:

$$\boxed{\begin{aligned} s_k &= T(r_k) = (L-1)\sum_{j=0}^{k} p_r(r_j) \\ &= (L-1)\sum_{j=0}^{k}\frac{n_j}{MN} = \frac{L-1}{MN}\sum_{j=0}^{k} n_j \qquad k = 0,1,\dots,L-1 \end{aligned}}$$

Because we deal with digital images, the second equation is important!

A processed image is obtained by mapping each pixel in the input image with intensity r_k into a corresponding pixel with level s_k

Example of Histogram Equalization

- Assume the following intensity distribution for a 3-bit image (L=8) of size 64×64 ($M \times N = 4096$):

r_k	n_k
$r_0 = 0$	790
$r_1 = 1$	1023
$r_2 = 2$	850
$r_3 = 3$	656
$r_4 = 4$	329
$r_5 = 5$	245
$r_6 = 6$	122
$r_7 = 7$	81

histogram

Intensity levels are integers
in the range $[0, L-1] = [0, 7]$

- We intend to use histogram equalization to create a new histogram (equalized histogram)

Example of Histogram Equalization

- First normalize the given histogram:

$$p(r_k) = n_k / MN \text{ for } k = 0,1,2,\ldots,L-1$$

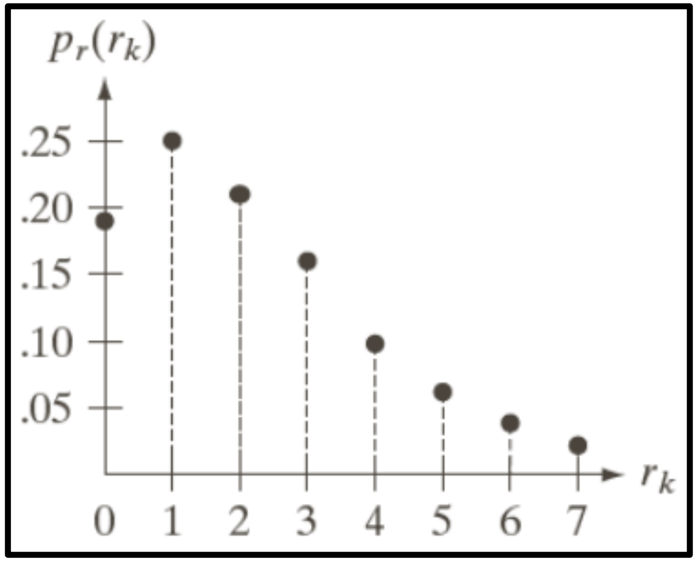
r_k	n_k	$p_r(r_k) = n_k/MN$
$r_0 = 0$	790	0.19
$r_1 = 1$	1023	0.25
$r_2 = 2$	850	0.21
$r_3 = 3$	656	0.16
$r_4 = 4$	329	0.08
$r_5 = 5$	245	0.06
$r_6 = 6$	122	0.03
$r_7 = 7$	81	0.02

histogram normalized

$\leftarrow 790/4096$

$\leftarrow 1023/4096$

$\vdots$



normalized histogram

Example of Histogram Equalization

Then, using the Equation as a transformation function, calculate transformed value s_k (i.e., the new gray level)

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

$$s_0 = T(r_0) = 7\sum_{j=0}^{0} p_r(r_j) = 7 \times 0.19 = 1.33 \rightarrow 1$$

it should be rounded to the nearest integer

$$s_1 = T(r_1) = 7\sum_{j=0}^{1} p_r(r_j) = 7(0.19 + 0.25) = 3.08 \rightarrow 3$$

Similarly,

$$s_2 = 4.55 \quad \rightarrow 5 \qquad s_3 = 5.67 \quad \rightarrow 6$$

$$s_4 = 6.23 \quad \rightarrow 6 \qquad s_5 = 6.65 \quad \rightarrow 7$$

$$s_6 = 6.86 \quad \rightarrow 7 \qquad s_7 = 7.00 \quad \rightarrow 7$$

Example of Histogram Equalization

- r_0=0 is mapped to s_0=1, in the equalized histogram, and gray level s_0=1 has 790 pixels and a normalized value of 0.19
- We can equalize the whole histogram using the same procedure:

s_0=1 ←	r_0=0	790	pixels	0.19	
s_1=3 ←	r_1=1	1023	pixels	0.25	
s_2=5 ←	r_2=2	850	pixels	0.21	
s_3=6 ←	r_3=3	656	pixels	0.16	0.24
s_4=6 ←	r_5=4	329	pixels	0.08	
s_5=7 ←	r_4=5	245	pixels	0.06	
s_6=7 ←	r_5=6	122	pixels	0.03	0.11
s_7=7 ←	r_7=7	81	pixels	0.02	

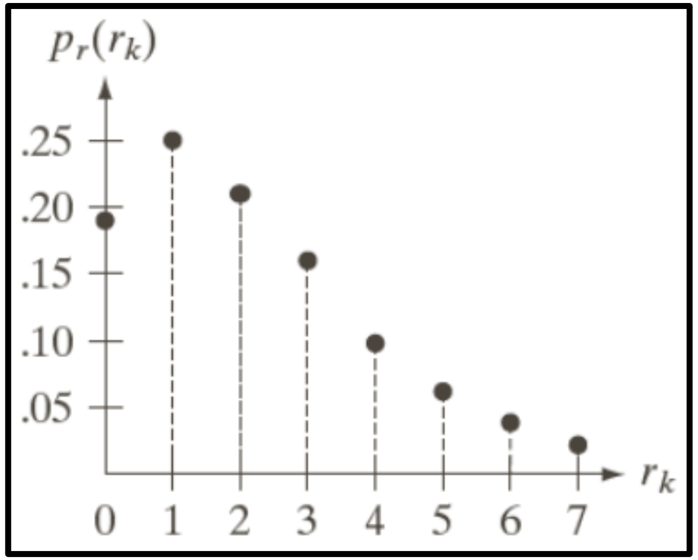
original histogram

s_k	n_k	$p_s(s_k)$
1	790	0.19
3	1023	0.25
5	850	0.21
6	656+329=985	0.24
7	245+122+81=448	0.11

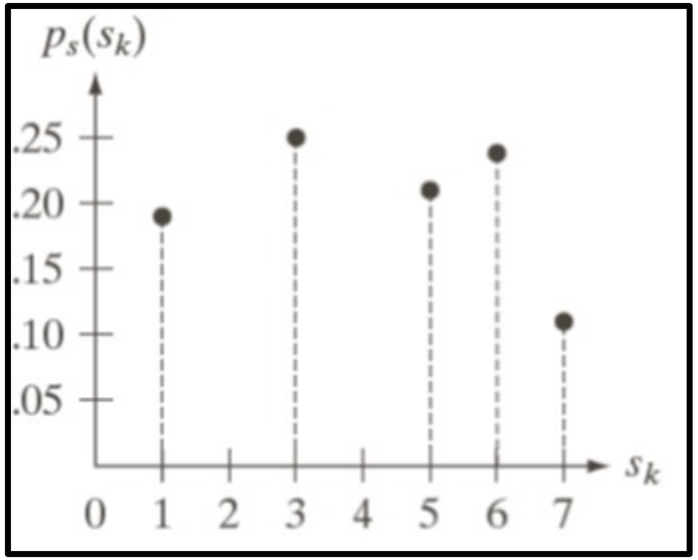
equalized histogram

Example of Histogram Equalization

- The equalized histogram is shown in this example
- The histogram after transformation is flatter than the original



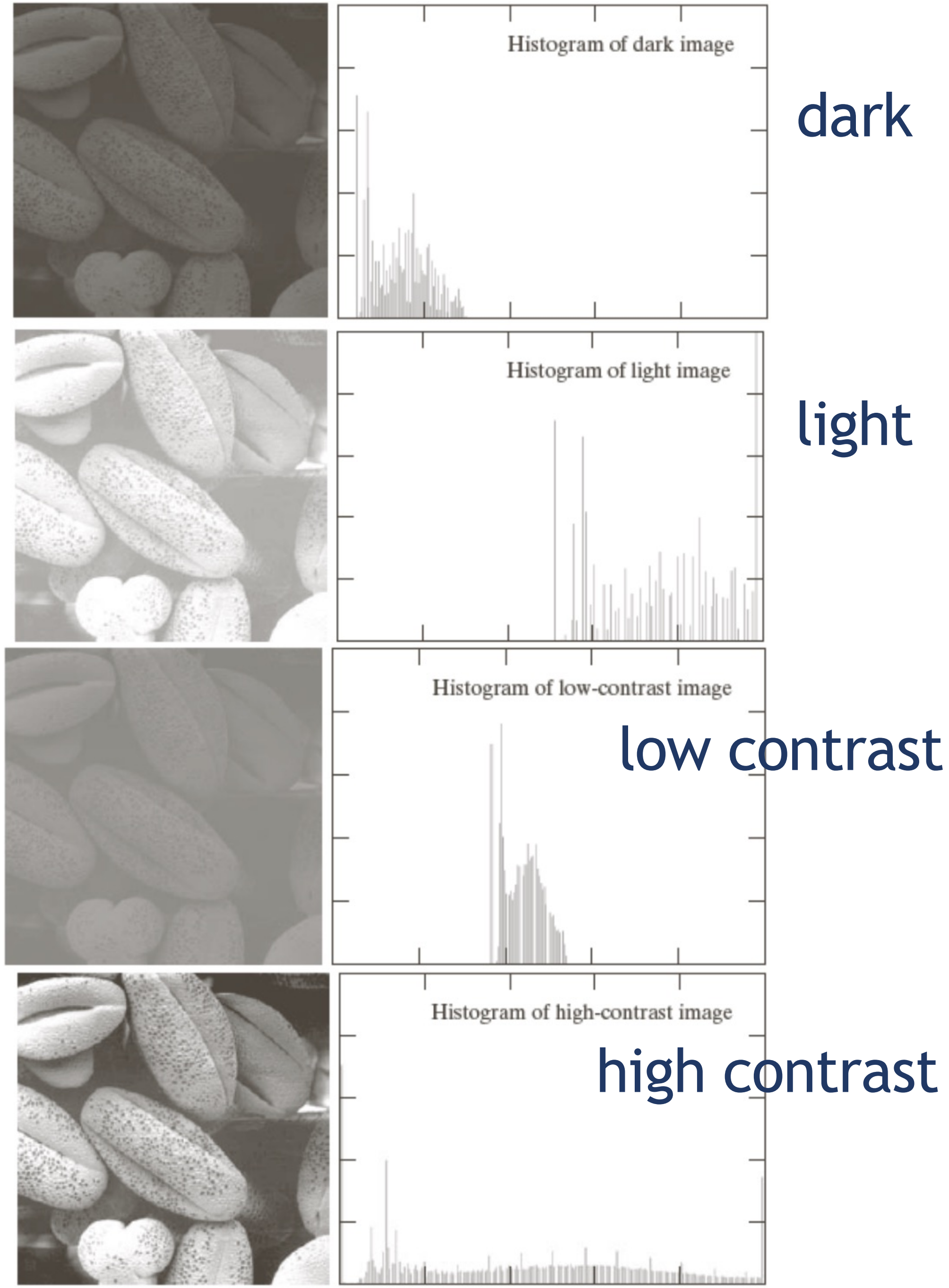
before equalization



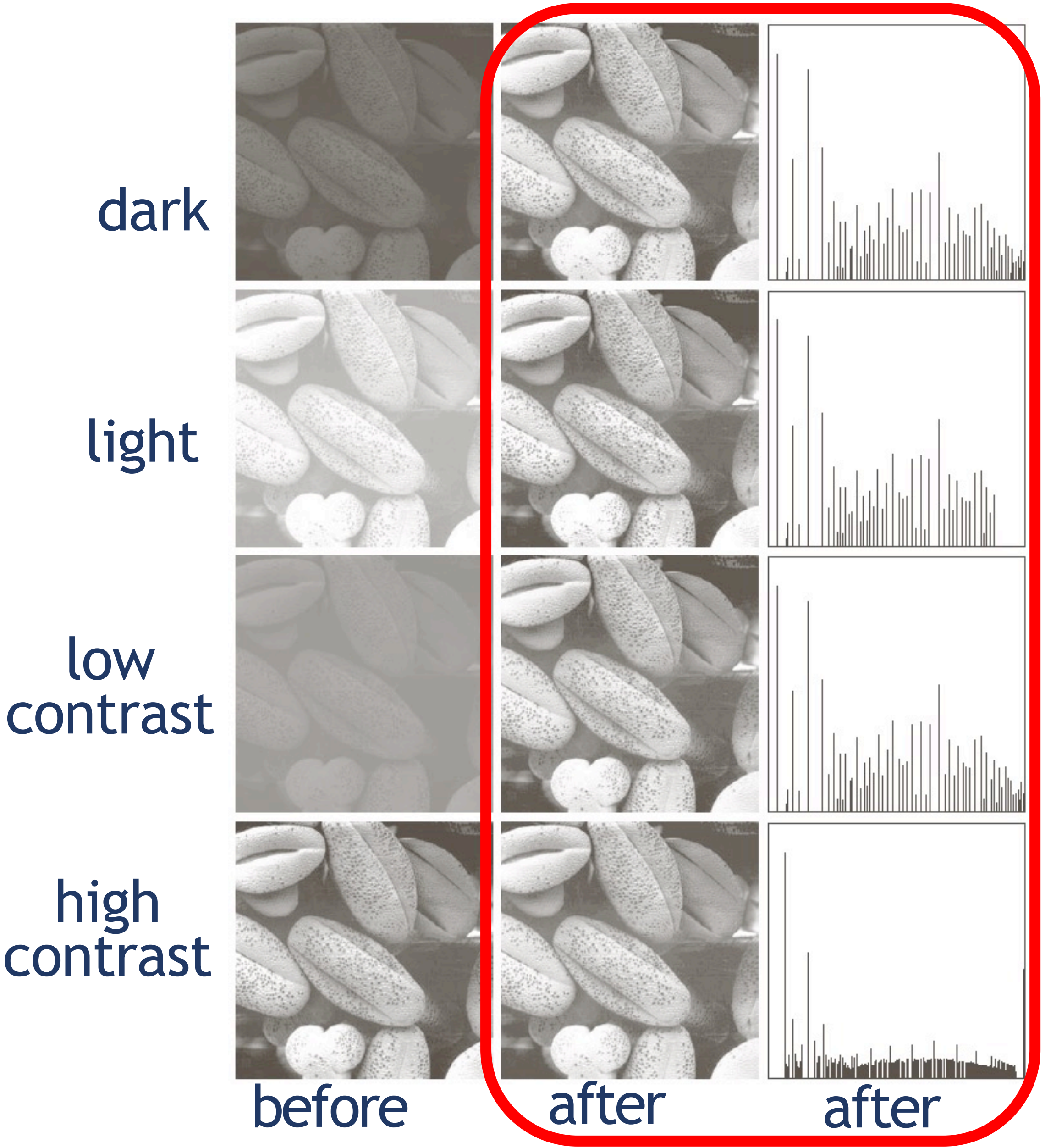
after equalization

Four Types of Images

- Four pollen photos were shown, each with four distinct intensity levels
  - **Dark Image:** The histogram is biased to the left
  - **Light Image:** The histogram is biased to the right
  - **Low-Contrast Image:** A narrow histogram located in the middle of the scale
  - **High-Contrast Image:** The histogram covers a wide range of the scale



Equalization on Each Image



- Histogram equalization was performed on each image
- The first three results showed that there was a significant improvement
- The result of the fourth image differs slightly because its histogram already covers a significant grayscale range

Explanation for exercise 2

Main Task

- Histogram computation and resolution reduction
- Histogram equalization

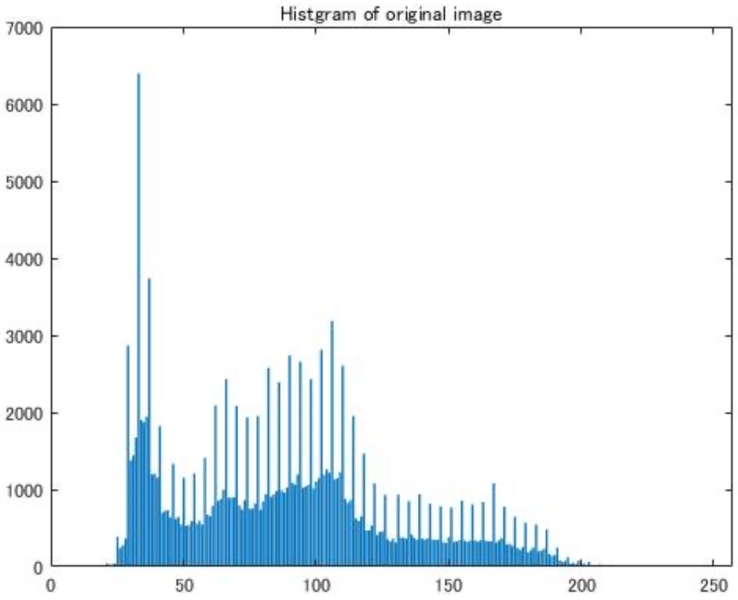
Histogram computation and resolution reduction

- Histogram computation

Test image

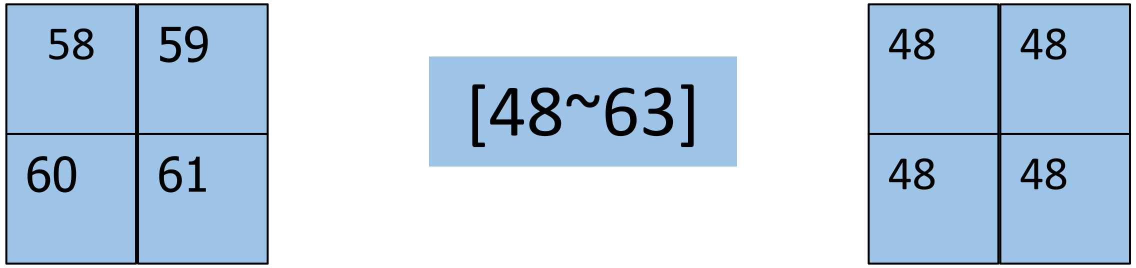
170	238	85	255	221	0
68	136	17	170	119	68
221	0	238	136	0	255
119	255	85	170	136	238
238	17	221	68	119	255
85	170	119	221	17	136

Pixel values
(from 0 to 255)



Histogram computation and resolution reduction

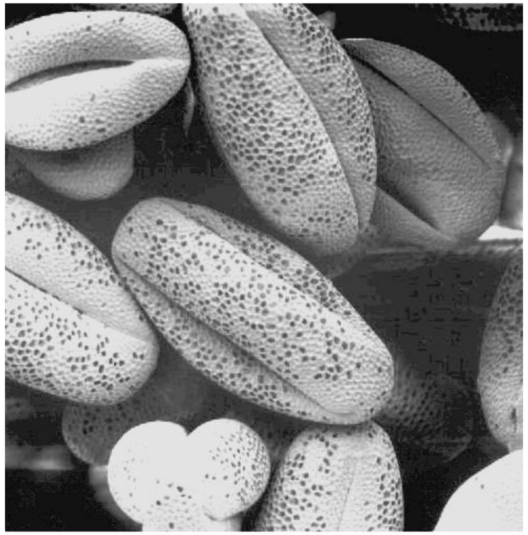
- Resolution reduction
  - From 8-bit to 4-bit
  - 8-bit: 0~255
  - 4-bit: 0~15 (However the image in MATLAB is default 8-bit, so you can divide 0~255 to 2^4 intervals:[0,15], [16,31],…,[240,255])



Histogram equalization

- Use the information you learned from the lesson
  - The function *cumsum* can be used to compute the cumulative sum

Test image



Output image

Thank you for your attention