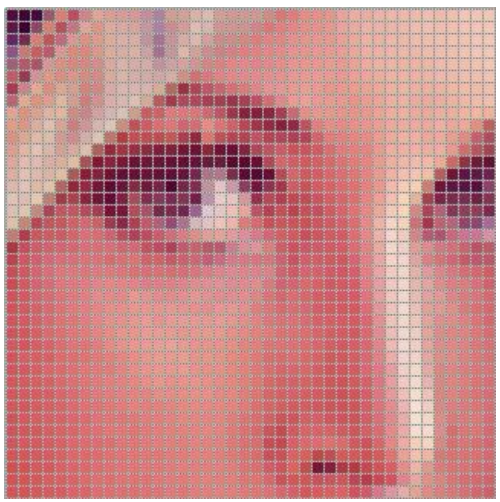
TDS3651
Visual Information Processing



LECTURE 2

Manipulating Pixels

Faculty of Computing and Informatics
Multimedia University
prepared by Lai-Kuan, Wong
modified by Yuen Peng, Loh

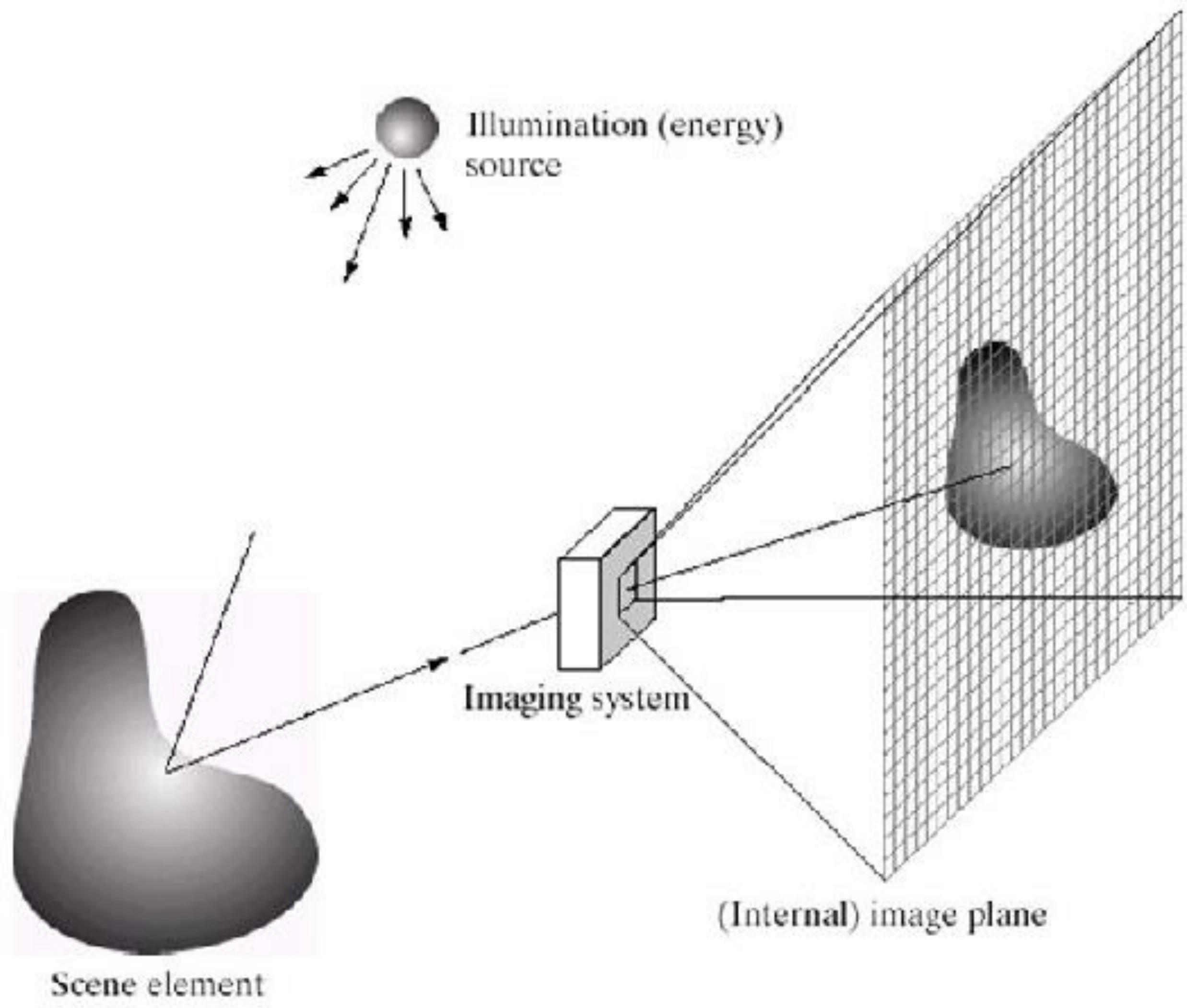
Some graphics and materials are taken from UTexas at Austin CS376 & CS381V notes, Stanford Uni's CS131 notes and Gonzalez & Wood's DIP textbook

Lecture Outline

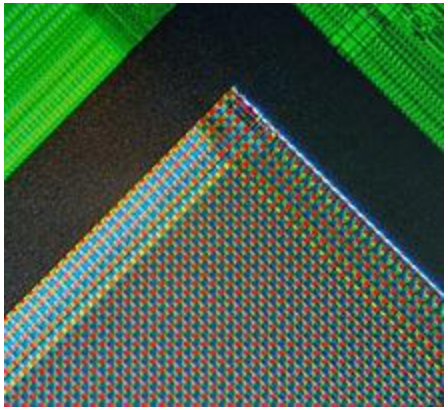
- Image Formation
- Point (Pixel)-based Processing
- Image Histograms
- Neighborhood Processing

Images and how they are represented

Image formation



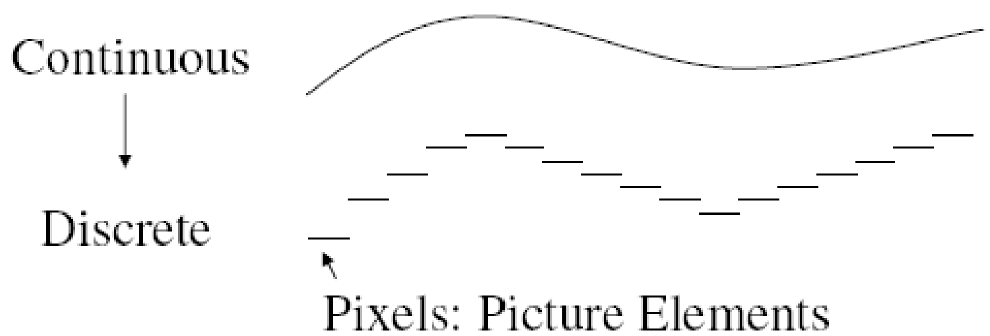
The digital camera



- A digital camera replaces film with a sensor array
  - Each cell in the array is a light-sensitive diode that converts photons to electrons

Digital images

- Computers work with discrete pieces of information
- How do we digitize a continuous-value image?



Generating a Digital Image

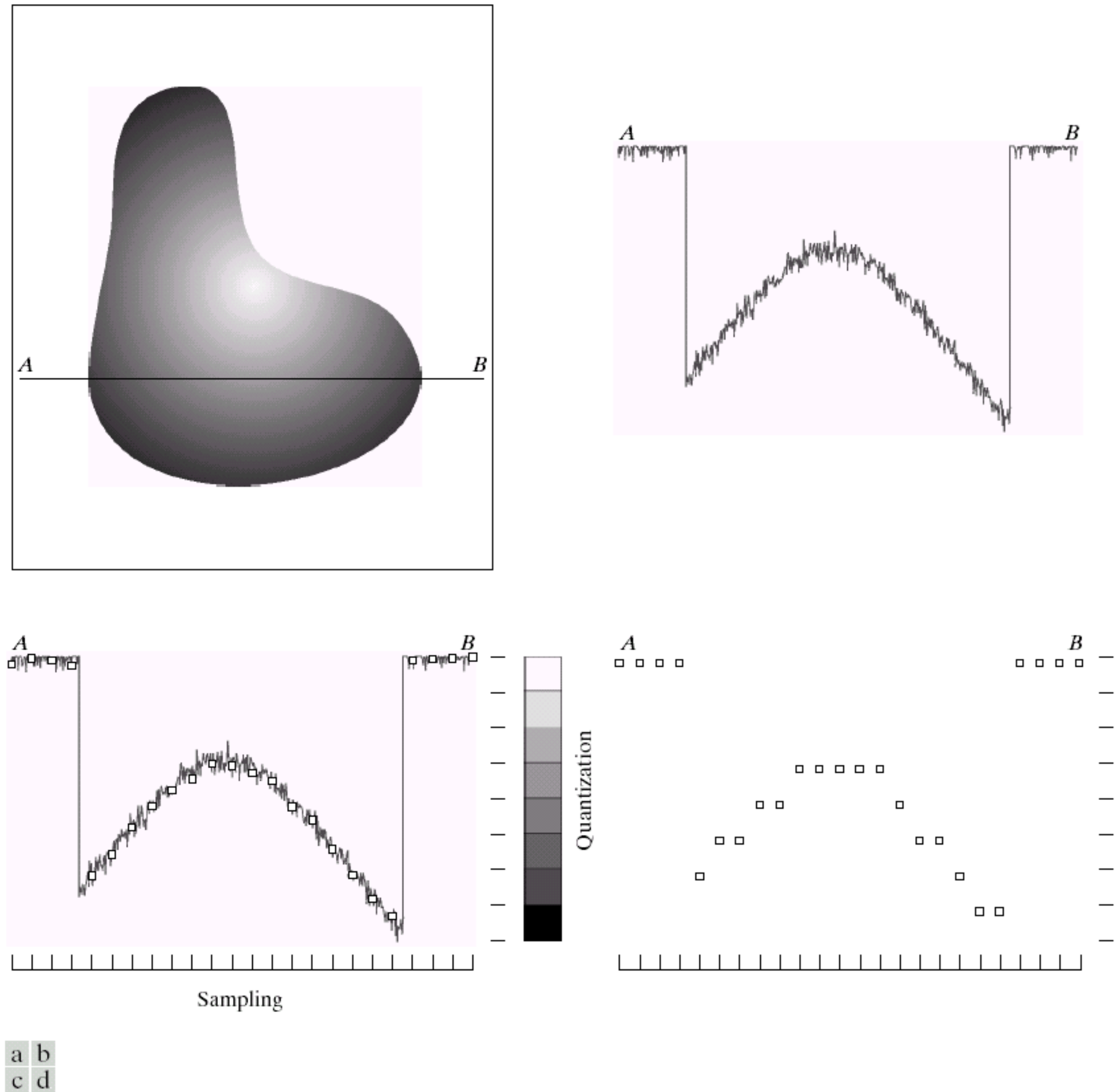


FIGURE 2.16 Generating a digital image. (a) Continuous image. (b) A scan line from A to B in the continuous image, used to illustrate the concepts of sampling and quantization. (c) Sampling and quantization. (d) Digital scan line.

Image Sampling & Quantization

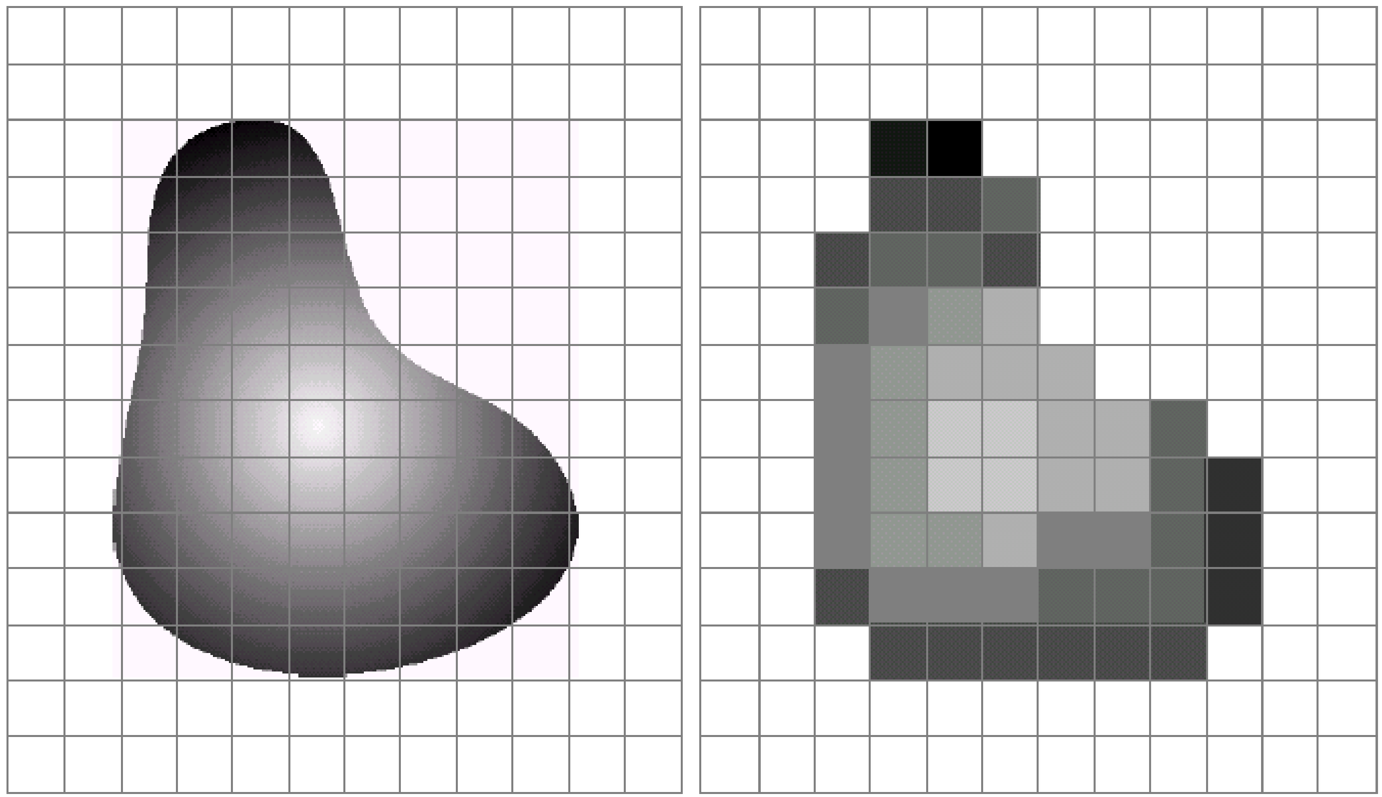
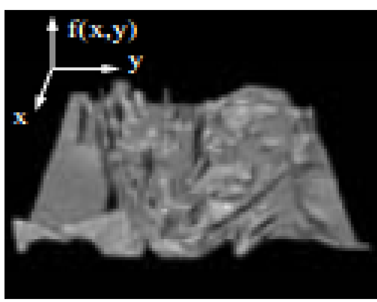


FIGURE 2.17 (a) Continuos image projected onto a sensor array. (b) Result of image sampling and quantization.

Image Sampling & Quantization

- **Sample** the 2D space on a regular grid
- **Quantize** each sample (round to nearest integer)
- Image thus represented as a matrix of integer values

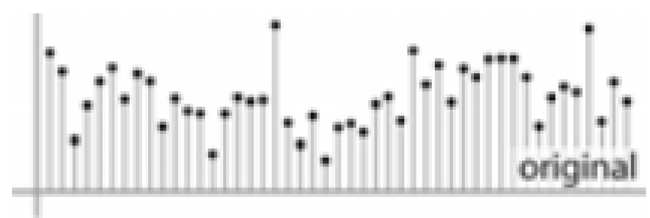


j

i

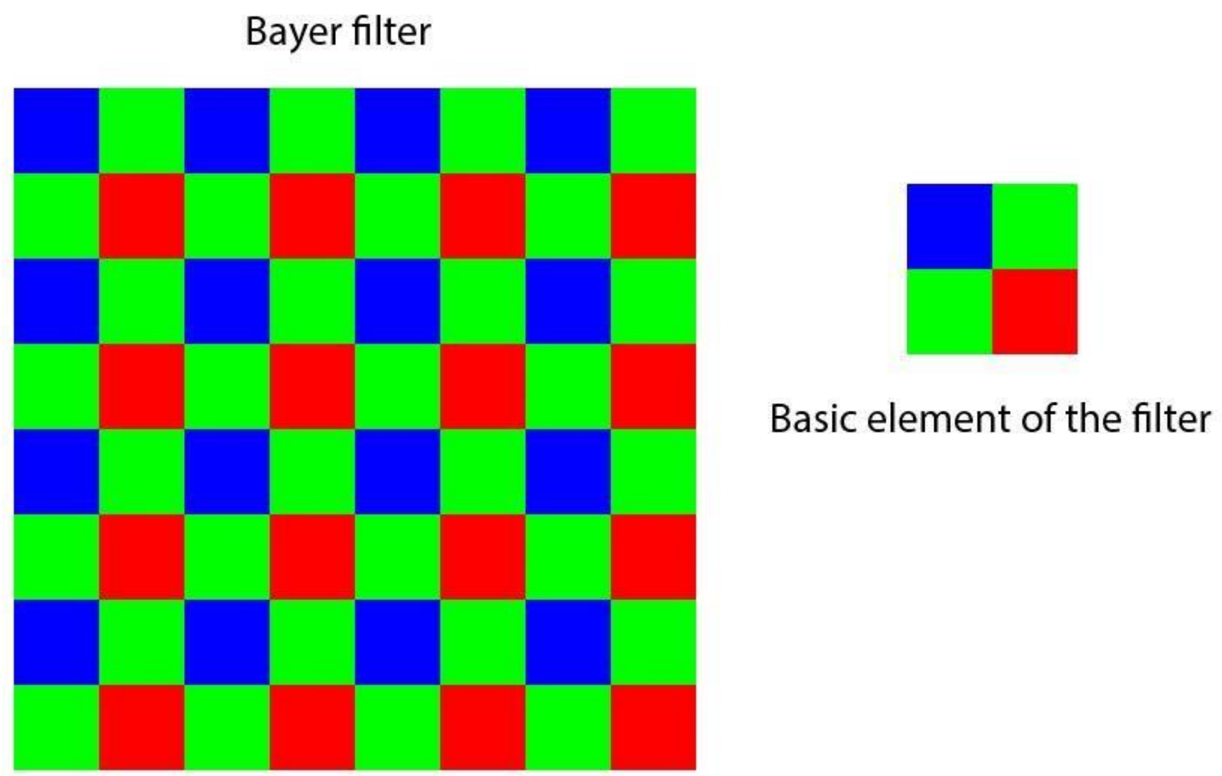
62	79	23	119	120	105	4	0
10	10	9	62	12	78	34	0
10	58	197	46	46	0	0	48
176	135	5	188	191	68	0	49
2	1	1	29	26	37	0	77
0	89	144	147	187	102	62	208
255	252	0	166	123	62	0	31
166	63	127	17	1	0	99	30

2D



1D

How does the image sensor capture colours?



Bayer filter
https://en.wikipedia.org/wiki/Bayer_filter

Capturing Digital Images (The Bayer Filter) – Computerphile
https://www.youtube.com/watch?v=LWxu4rkZBLw

Image as functions

- An image contains discrete number of pixels
  - A simple example
  - Pixel value:
    - “grayscale” (or “intensity”) : [0,255]

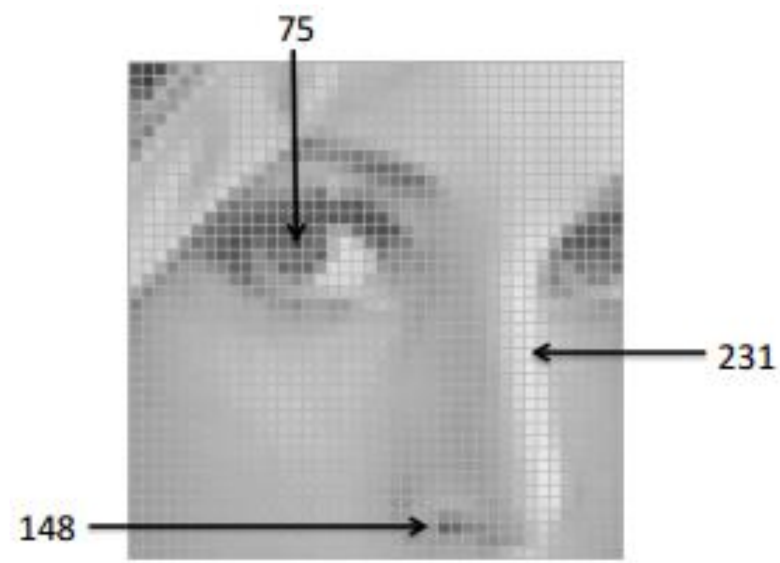
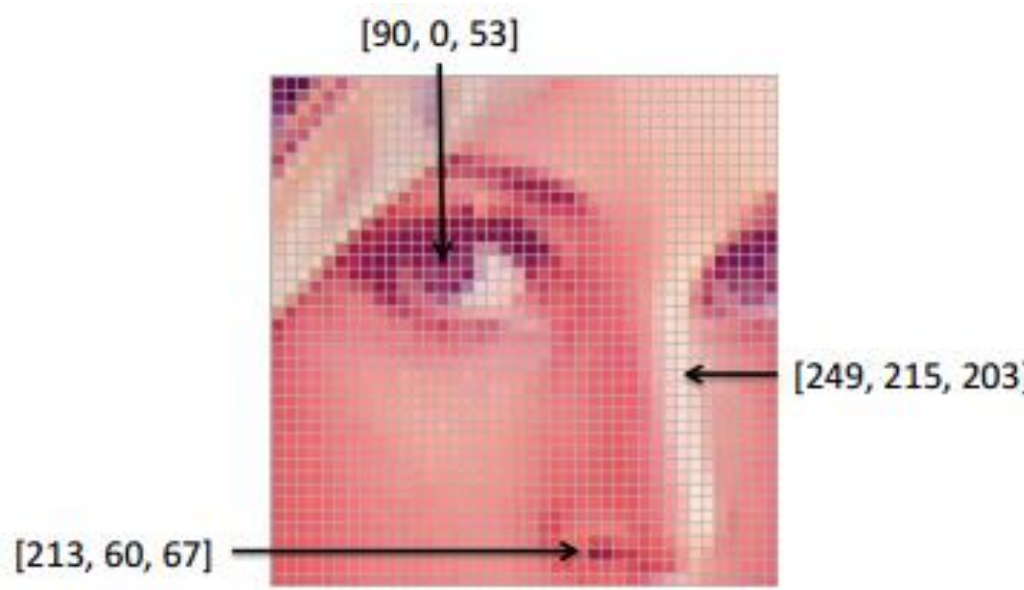


Image as functions

- An image contains discrete number of pixels
  - A simple example
  - Pixel value:
    - “grayscale” (or “intensity”) : [0,255]
    - “color”
      - RGB: [R, G, B]
      - Lab : [L, a, b]
      - HSV: [H, S, V]



Digital colour images

Color images, RGB color space

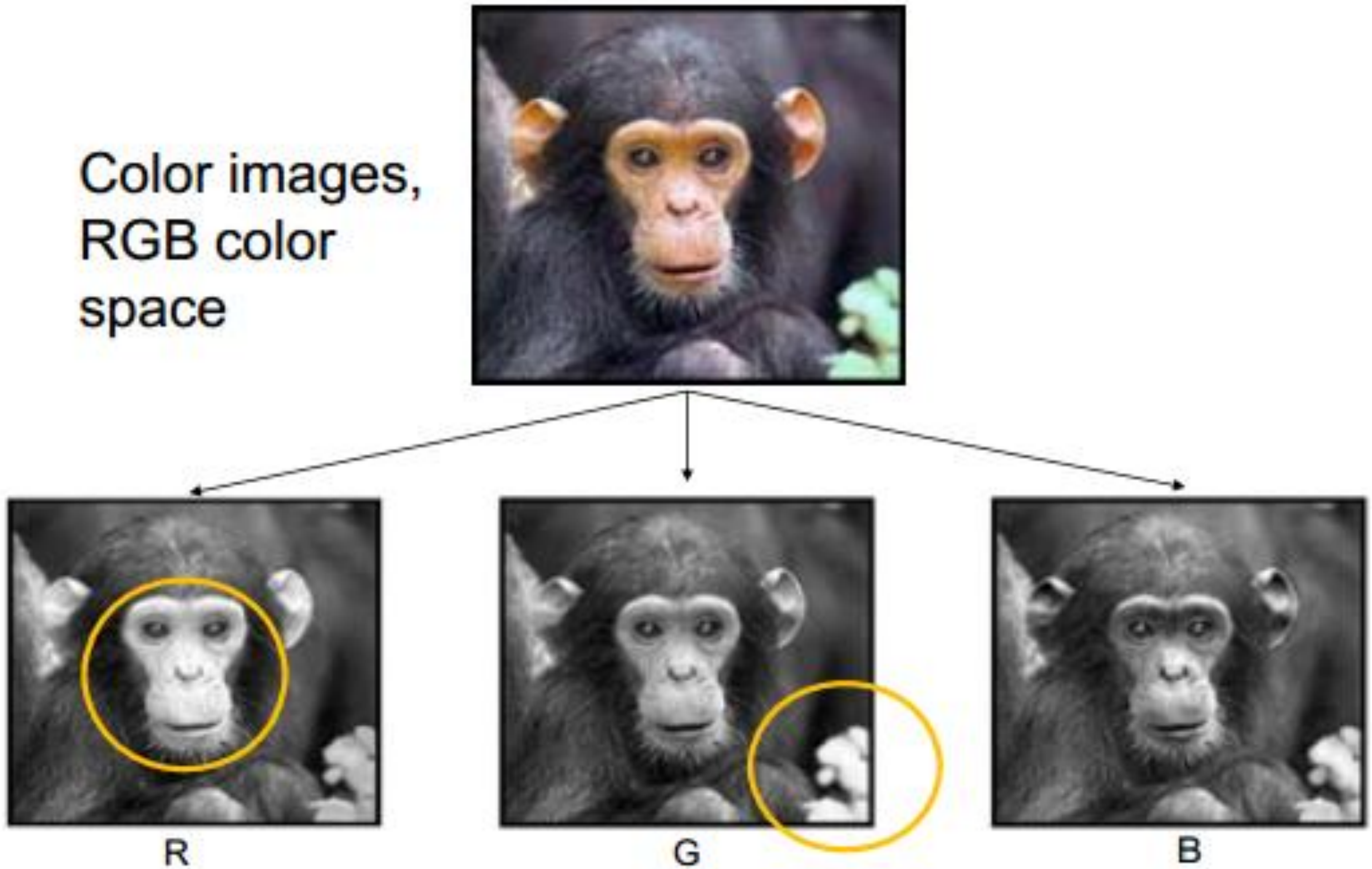


Image as functions

- **An image** as a function of f from R^2 to R^M
  - $f(x, y)$ gives the **intensity** at position (x, y)
  - Defined over a rectangle, with a finite range:

$$f\colon \underbrace{[a,b] \times [c,d]}_{\text{domain support}} \rightarrow \underbrace{[0,255]}_{\text{range}}$$

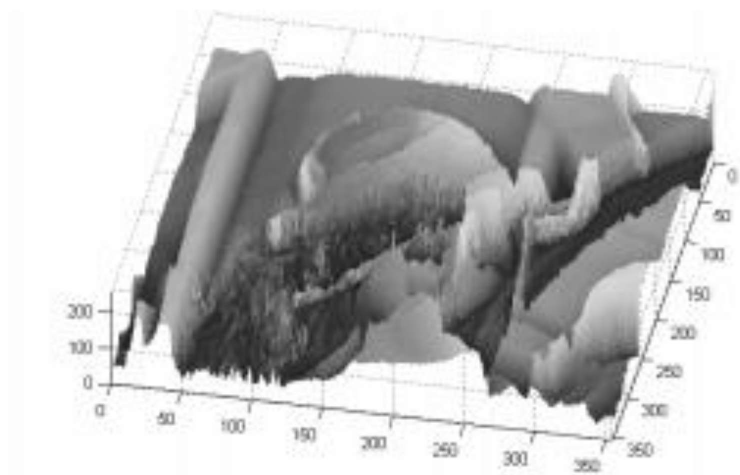
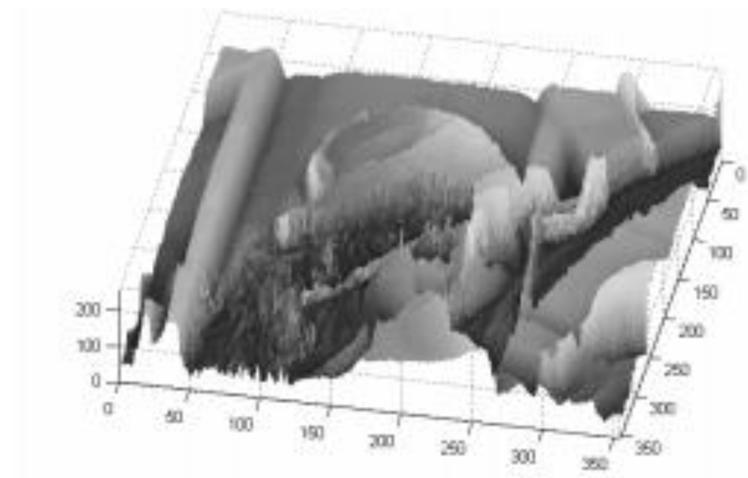


Image as functions

- **An image** as a function of f from R^2 to R^M
  - Color image (R^3)

$$f(x,y) = \begin{bmatrix} r(x,y) \\ g(x,y) \\ b(x,y) \end{bmatrix}$$



  - RGB-D image?

Image as functions

Q: RGB-D image?

A. Duration

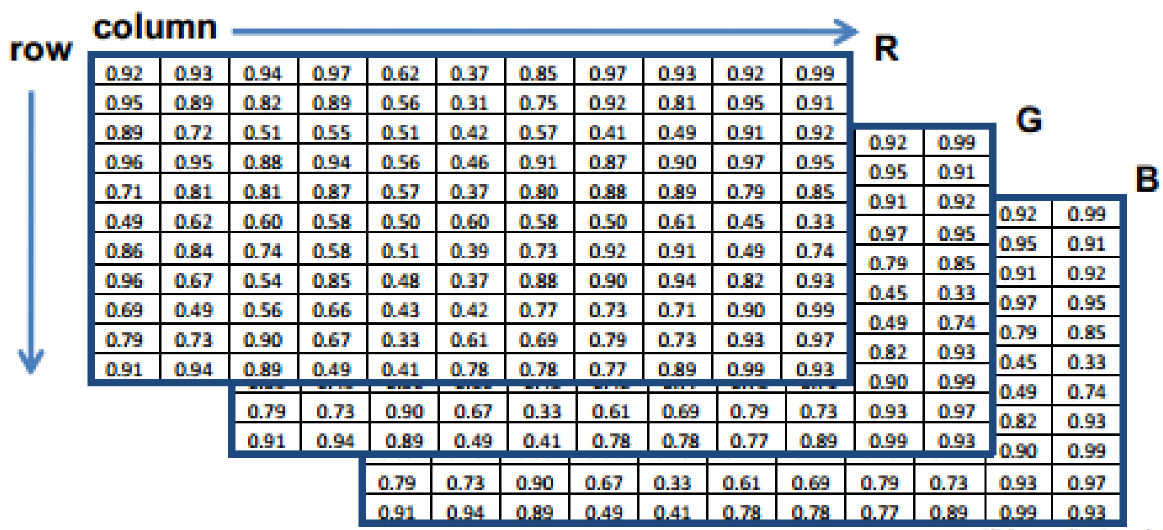
B. Depth

C. Direction

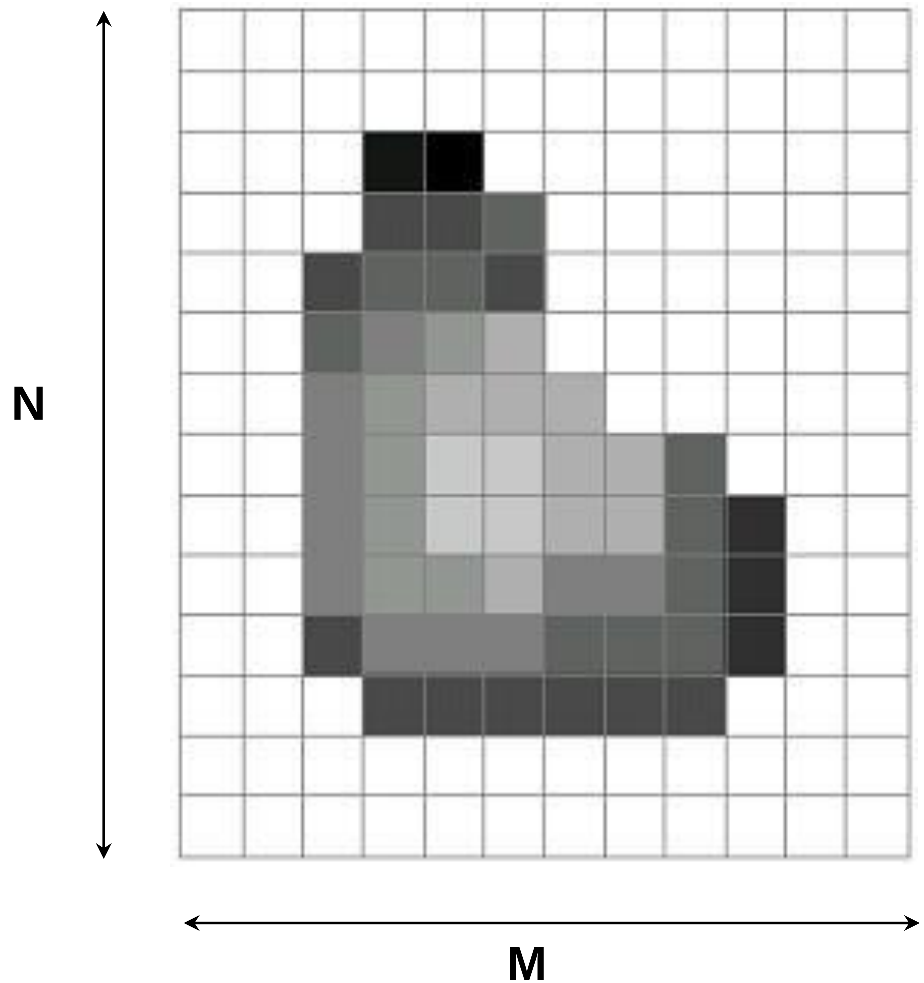
D. Don't know

How are images represented?

- Images represented as a 3-D array (or 'matrix')
  - Python: '`ndarray`' container in numpy. Image values are '`uint8`' type
  - Example: a $N \times M$ RGB image stored in array '`im`'
    - `im[0,0,0]` – top-left pixel value in R channel
    - `im[0,0]` – pixel values at location `(0,0)` for all 3 channels



Number of Bits



- *k*-bit image
- The number of gray levels typically is an integer power of 2

$$L = 2^k$$

- Number of bits required to store a digitized image

$$B = M \cdot N \cdot k$$

Resolution

- Resolution
  - How much details you can see in the image
  - Depends on sampling and gray levels
  - The higher the resolution of the image
    - The **better** the approximation of the digitized image from the original
    - The **larger** the size of the image

Subsampling

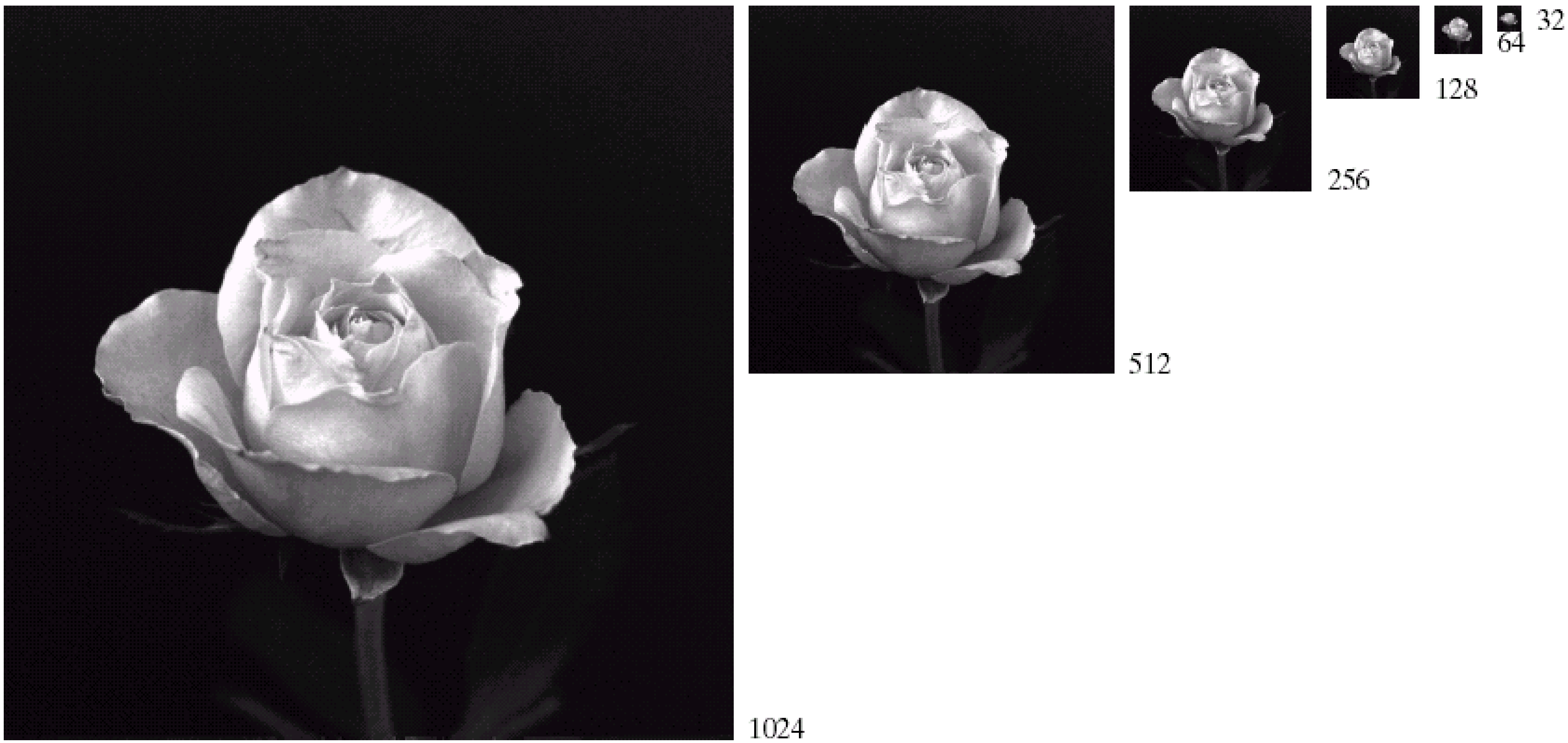
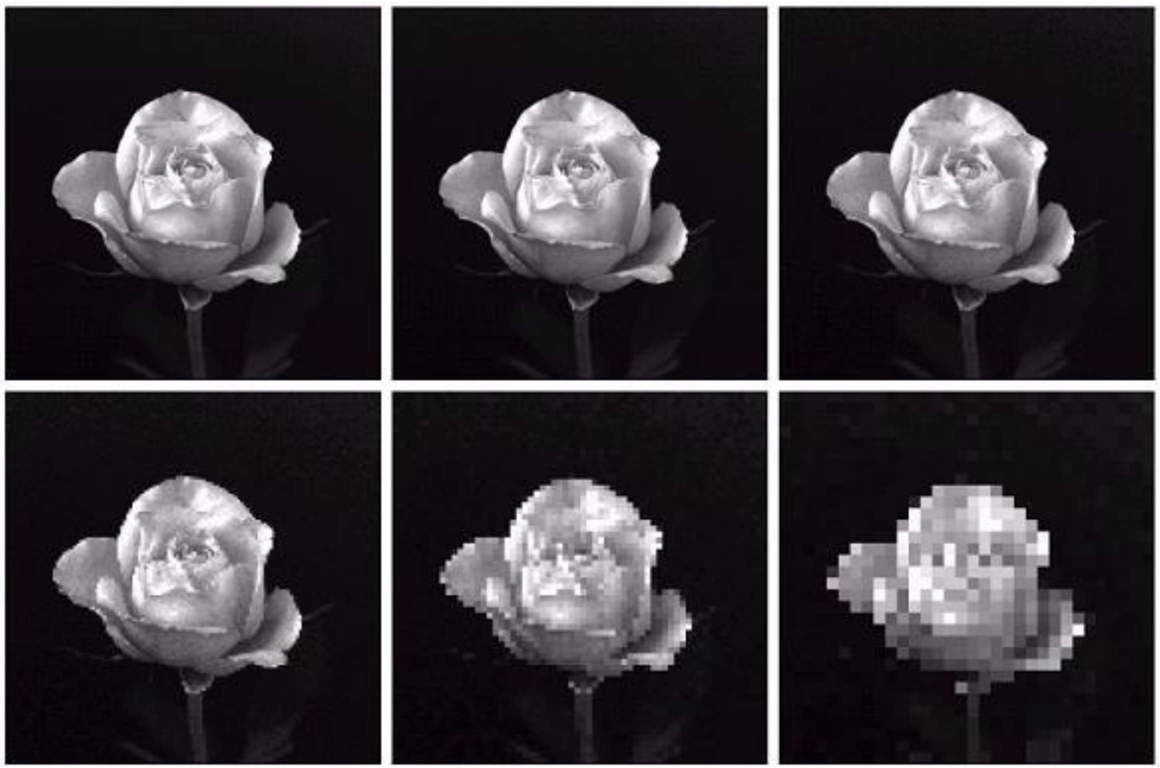


FIGURE 2.19 A 1024 × 1024, 8-bit image subsampled down to size 32 × 32 pixels. The number of allowable gray levels was kept at 256.

Subsampling Problem?

- Q: What are some problems that may occur if your images are sampled (or subsampled) to a lower resolution?
  A. Blurred
  B. Pixelated
  C. Lose color
  D. Distorted

Checkerboard Effect

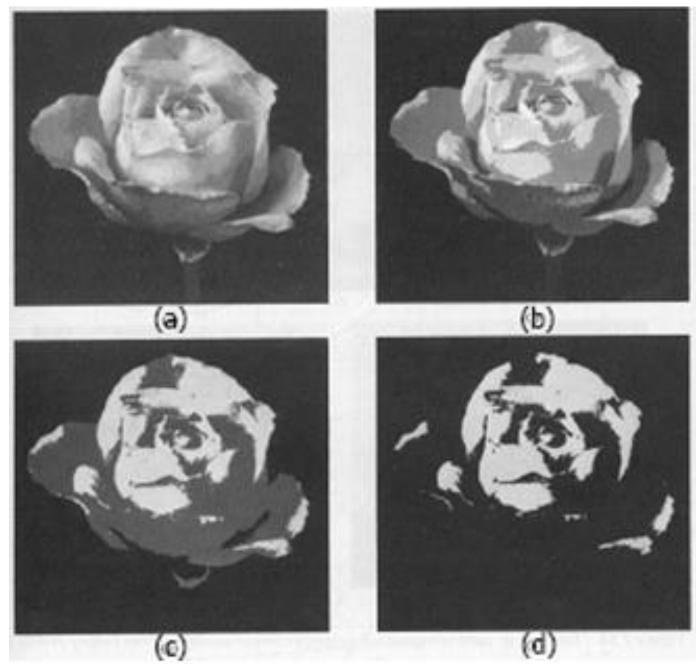


a	b	c
d	e	f

(a) 1024x1024
(b) 512x512
(c) 256x256
(d) 128x128
(e) 64x64
(f) 32x32

- If the resolution is decreased too much, the checkerboard effect can occur.

False Contouring



(a) Gray level = 16
(b) Gray level = 8
(c) Gray level = 4
(d) Gray level = 2

- If gray levels are insufficient, smooth areas will be affected
- False contouring occurs at smooth areas which has fine grayscale values

Question

- You have a 8-bit RGB image of resolution 1024x768 pixels. What is the size of the image file? (Give your answer in Mb)

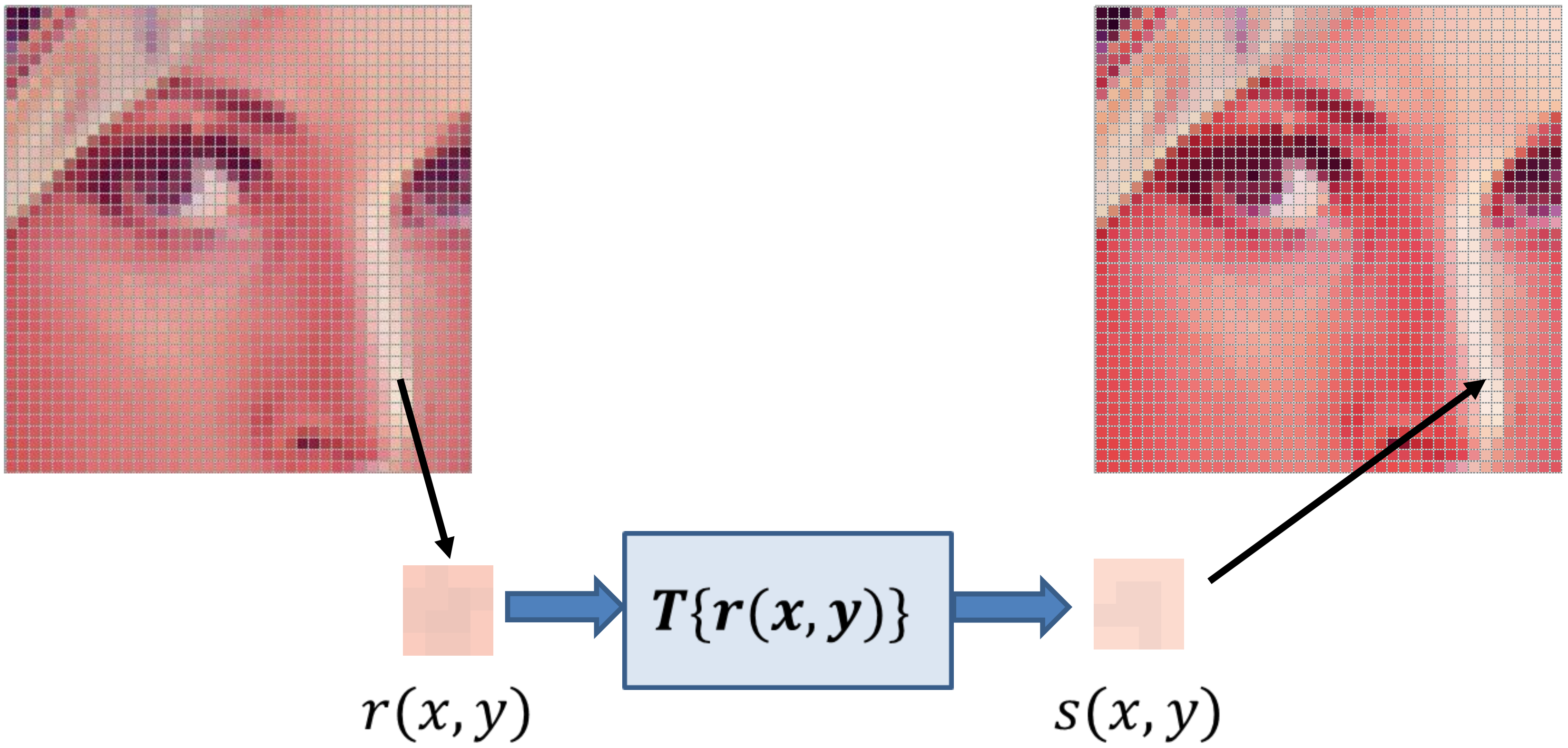
1Byte = 8bits
1kB = 1024B
1MB = 1024kB

Answer: ?

A. (1024x768)/(1024X1024) = 0.75 MB
B. (1024x768x3)/(1024X1024) = 2.25 MB
C. (1024x768x8)/(1024X1024) = 6.00 Mb
D. (1024x768x3x8)/(1024X1024) = 18.00 Mb

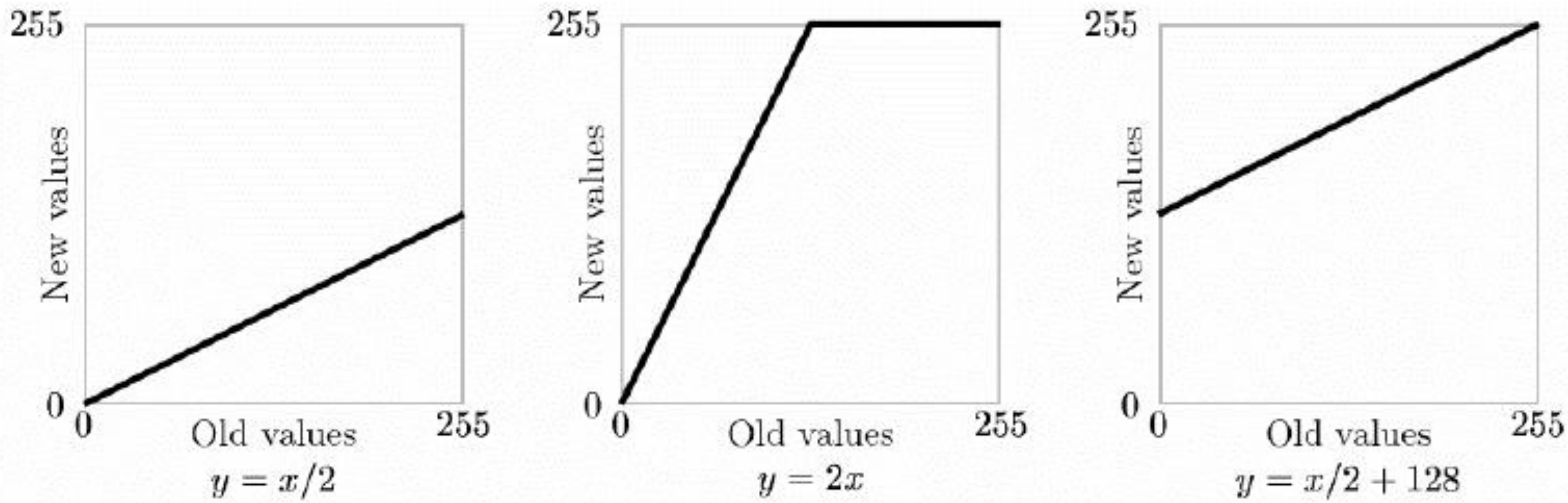
Pixel (Point)-based Processing

Pixel (or Point)-based Processing



With the transformation function T, do that for all pixels in the image!

Arithmetic Operations



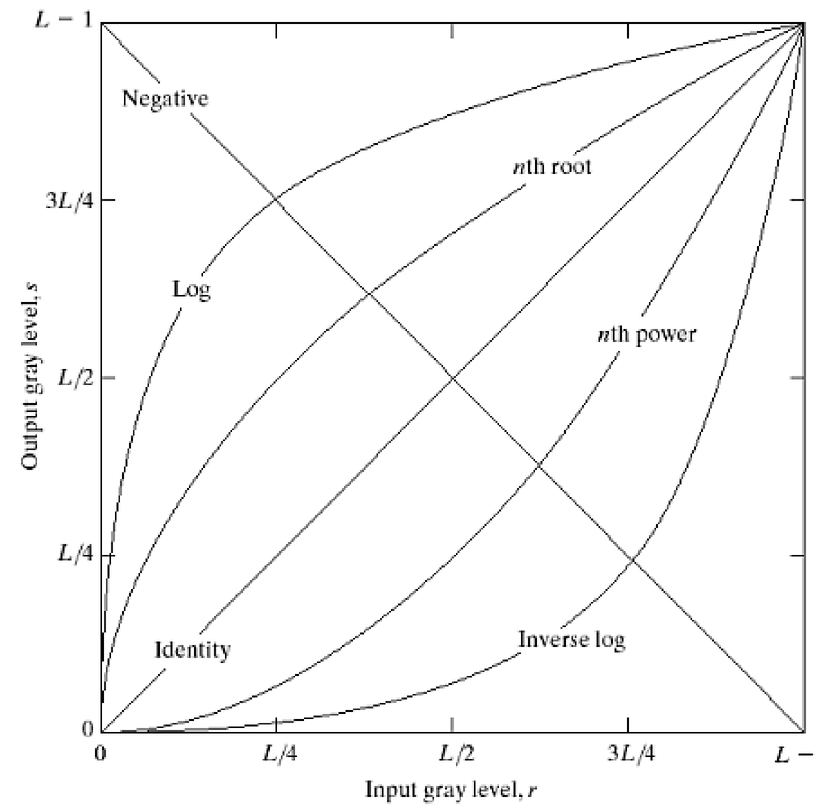
b3: $y = x/2$　　b4: $y = 2x$　　b5: $y = x/2 + 128$

What other operations?

- Whole family of possible functions that you can apply…

- **Question**: What can non-linear functions do?

E.g. irregular increase / decrease of pixel intensities (allows enhancement)



Linear Stretching

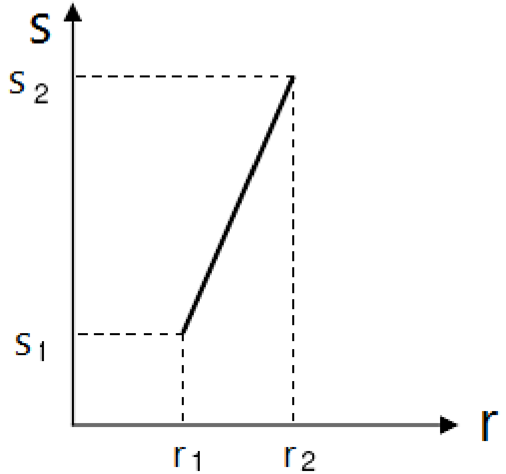
- Enhance the dynamic range by linear stretching the original gray levels to a new target range
- Example
  - Original range [100, 150]
  - Target range: [0, 255]
  - What's the general transformation function?

A. $S = \frac{(r - r_2)}{(r_1 - r_2)} * (S_2 - S_1) + S_1$

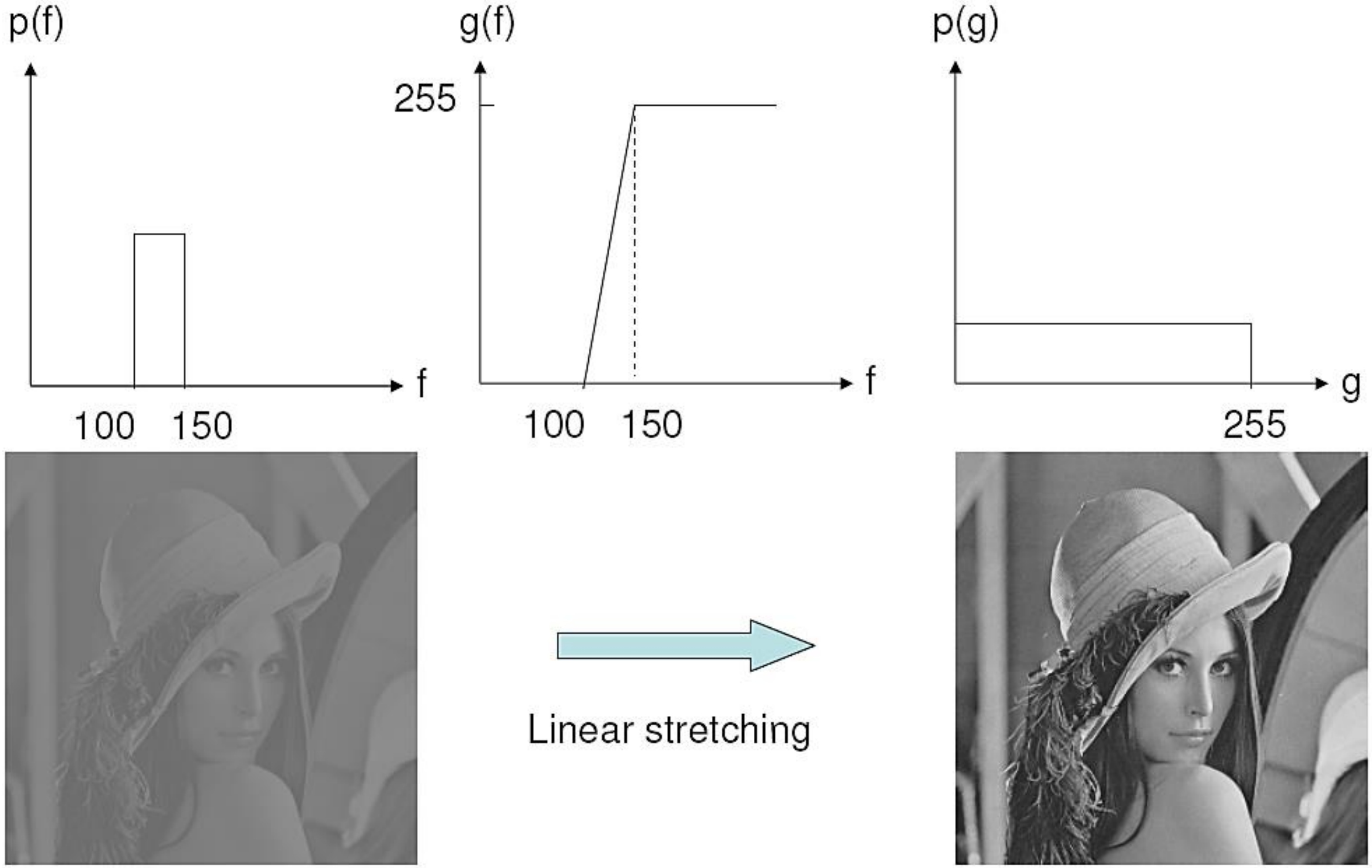
B. $S = \frac{(r - r_1)}{(r_2 - r_1)} * (S_2 - S_1) + S_1$

C. $S = \frac{(r - r_2)}{(r_1 - r_2)} * (S_1 - S_2) + S_1$

D. $S = \frac{(r - r_1)}{(r_2 - r_1)} * (S_1 - S_2) + S_1$

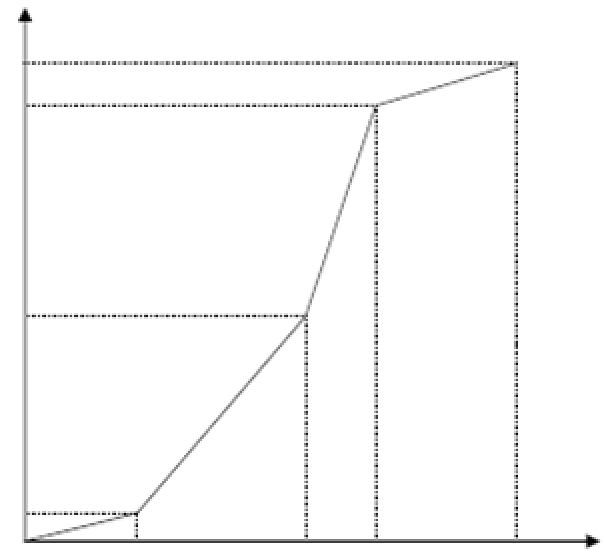


Linear Stretching

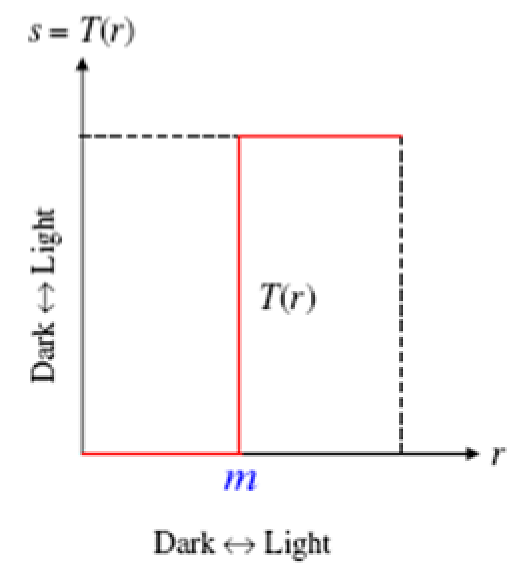


Piecewise Linear Stretching

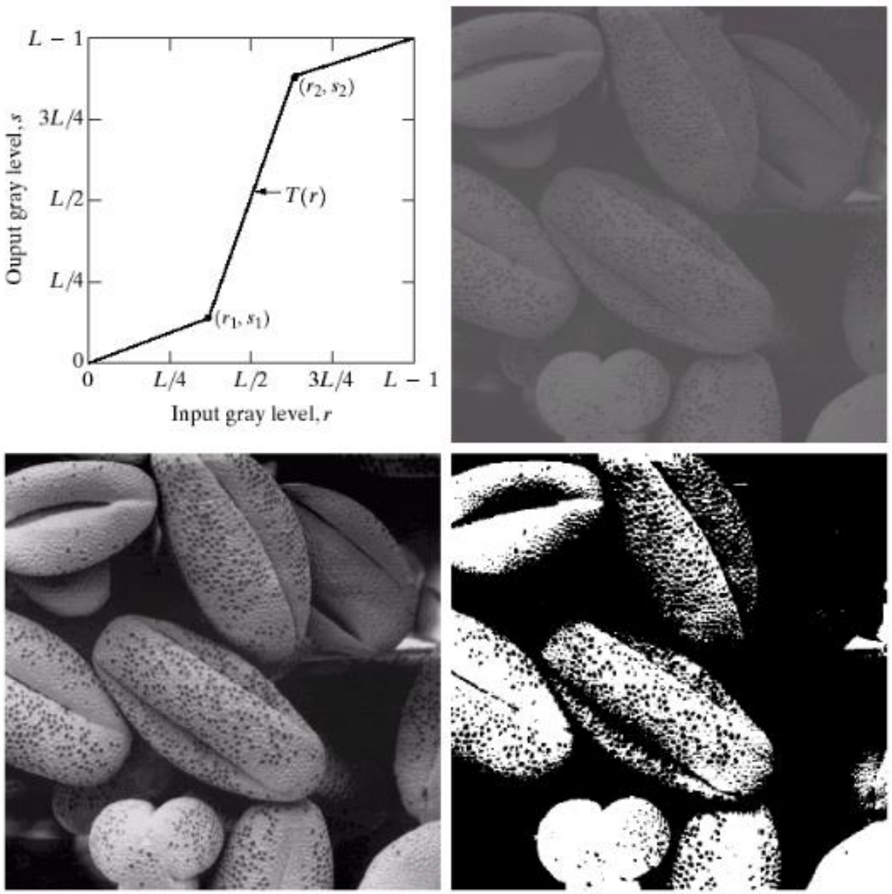
- Can have as many segments we want…



- Thresholding: Map to only two possible values, producing a binary image



Application: Contrast Stretching



- Problem: Low contrast image, result of poor illumination, lack of dynamic range
- Solution: Linear contrast stretching using the given transformation function *(bottom left)*
- Result after thresholding *(bottom right)*

We can use more than an image…

- **Image addition** – Pixel-wise addition of values from two images
  - Use to create double-exposures or composite images

+

=

$$g(x, y) = f_1(x, y) + f_2(x, y)$$

  - Or do a weighted blend:

$$g(x, y) = \alpha_1 f_1(x, y) + \alpha_2 f_2(x, y)$$

We can use more than an image…

- **Image subtraction** – Pixel-wise subtraction of one image from another image
  - Use to find changes between two images

$$g(x, y) = f_1(x, y) - f_2(x, y)$$

  - Absolute difference works better (Why?)

Histograms

Image Histogram

- Histogram: Diagram that shows distribution of data
- Histogram of a digital image with gray levels in the range $[0, L-1]$ is a discrete function with k bins

$$\mathrm{h}(r_k) = n_k$$

where

– r_k : the k-th gray level

– n_k : the number of pixels in the image having gray level r_k

– $\mathrm{h}(r_k)$: histogram of a digital image with gray levels r_k

Normalized Histogram

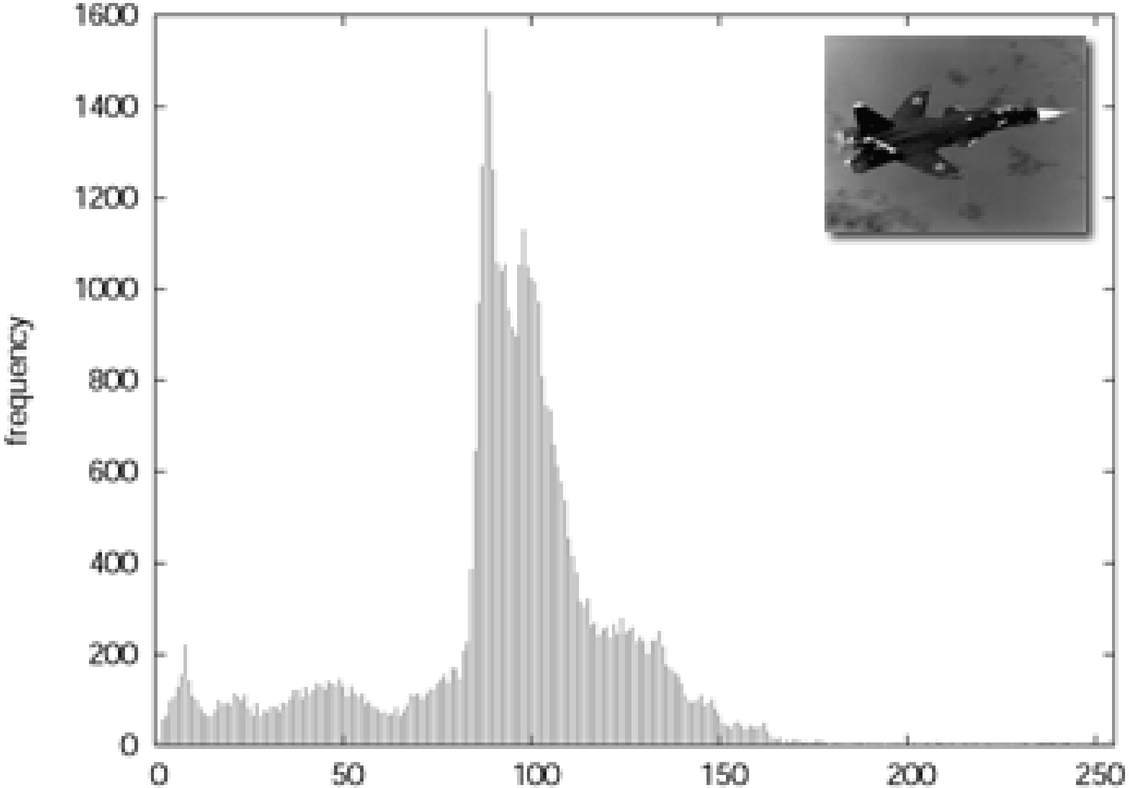
- Divide each bin count n_k of the histogram by the total number of pixels in the image, *n*

$$\mathrm{p}(r_k) = n_k/n$$

for $k = 0, 1, \ldots, \mathrm{L}-1$

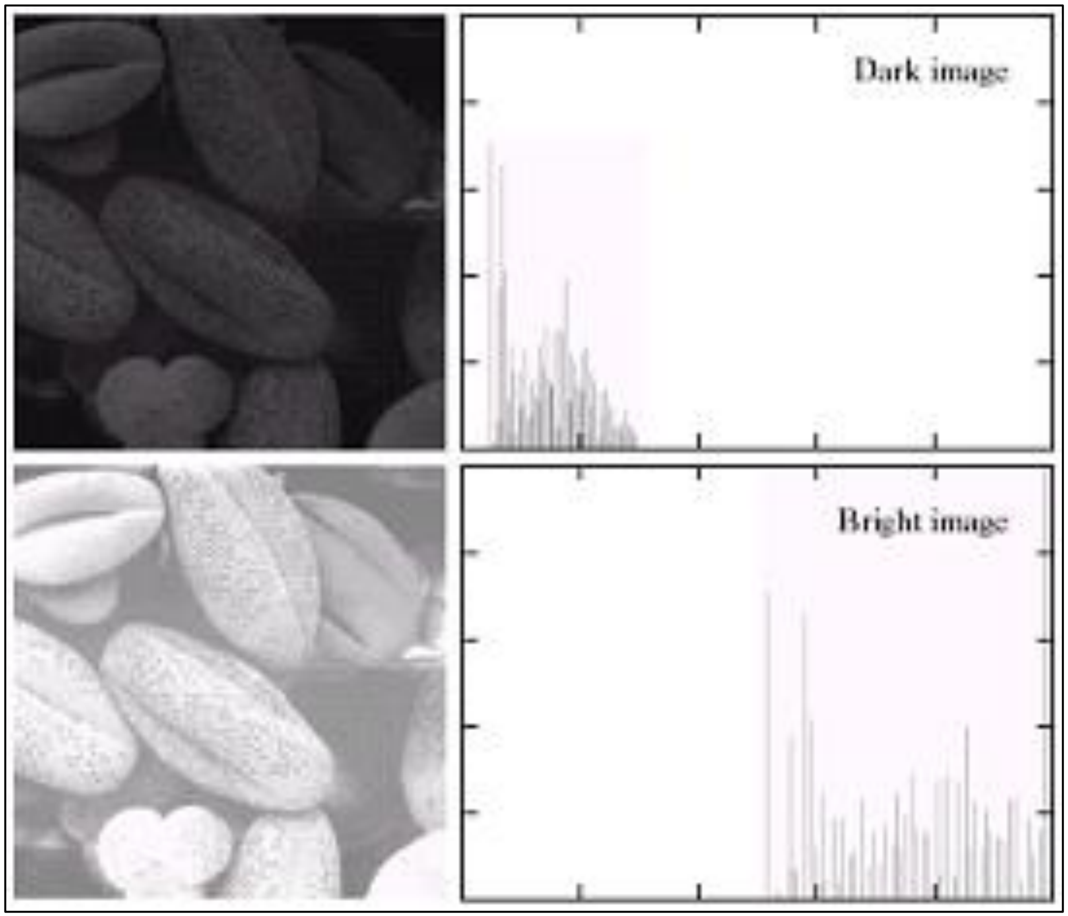
- $\mathrm{p}(r_k)$: probability of occurrence of gray level r_k
- The sum of all components of a normalized histogram is equal to 1

Image Histogram



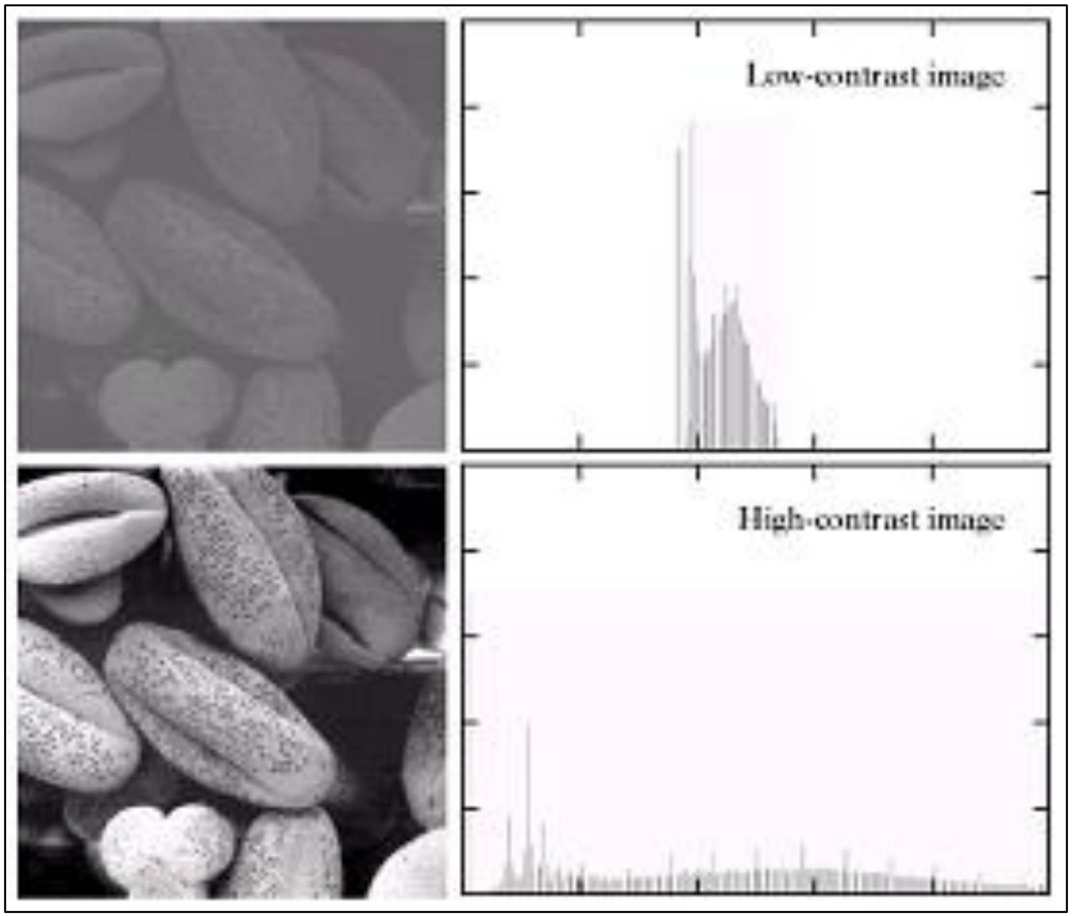
- What can the histogram of an image tell us?

Histogram & Image Contrast



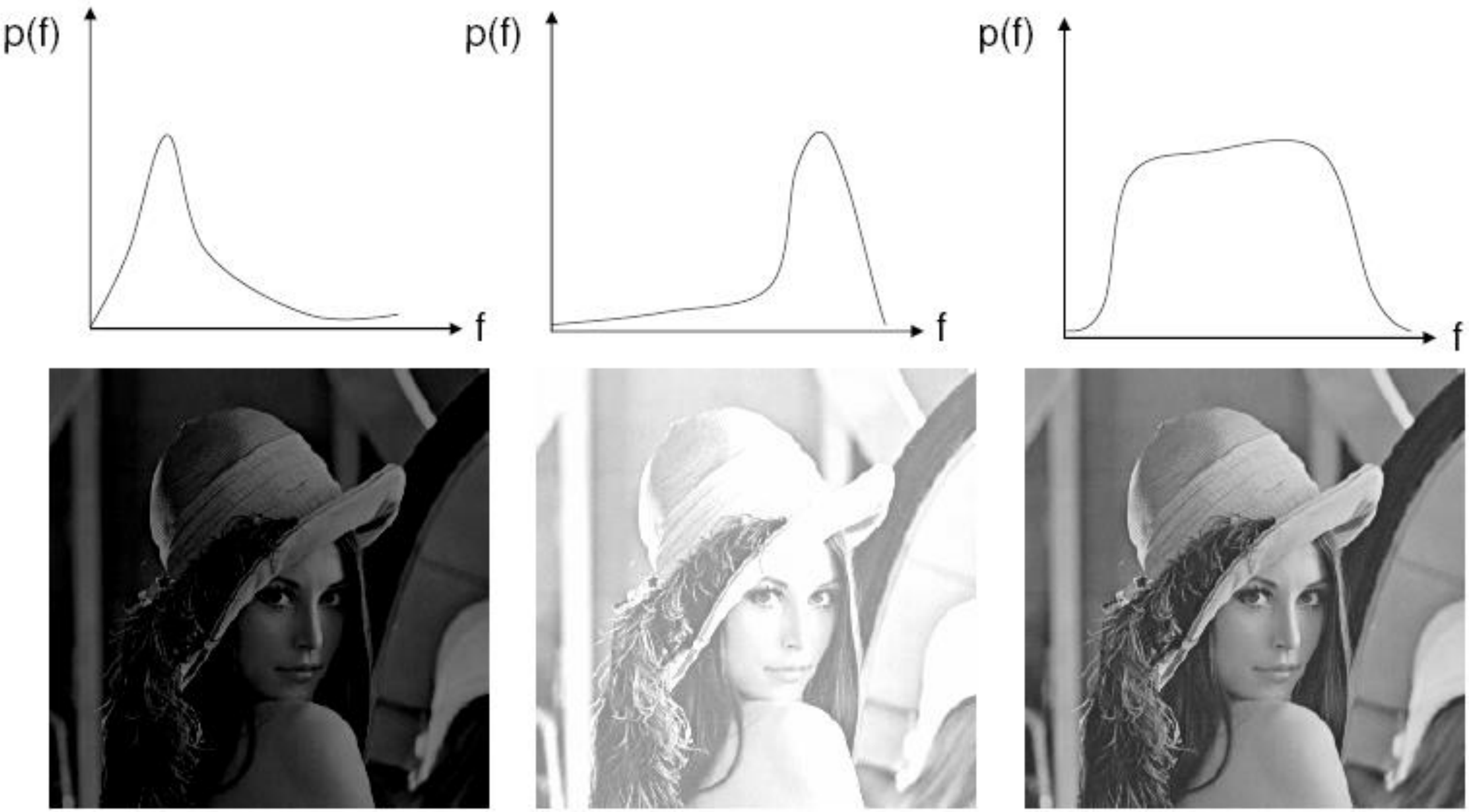
- **Dark Image**
  - Components of histogram are concentrated on the low side of the gray scale
- **Bright Image**
  - Components of histogram are concentrated on the high side of the gray scale

Histogram & Image Contrast



- **Low-contrast Image**
  - Histogram is narrow and centered towards the middle of the gray scale
- **High-contrast Image**
  - Histogram overs a 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 others

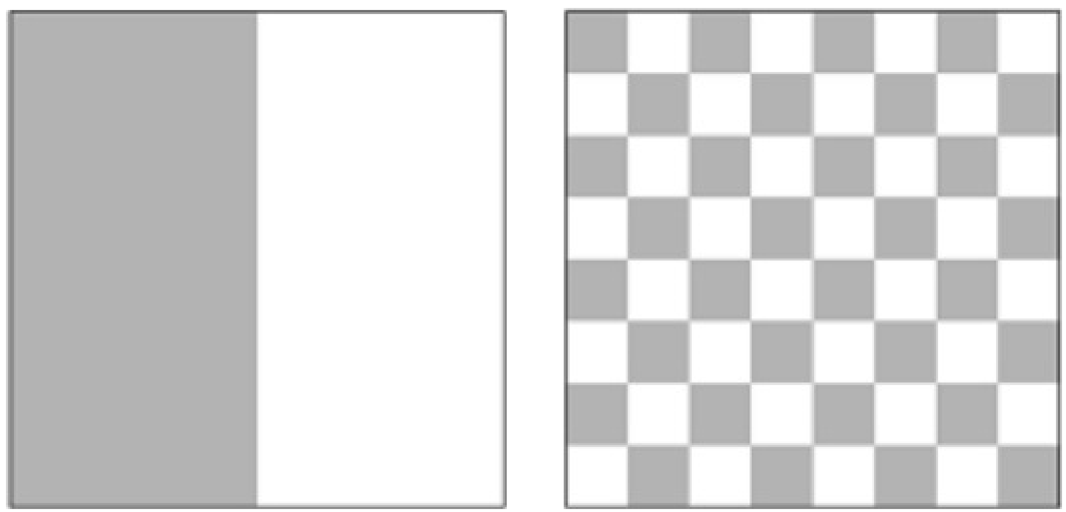
Histogram & Image Contrast



(a) Too dark (b) Too bright (c) Well balanced

Different images, same histogram!

- Visualize the histogram for the following two images of size 64x64.



- Do you think they have the same histogram? Why?

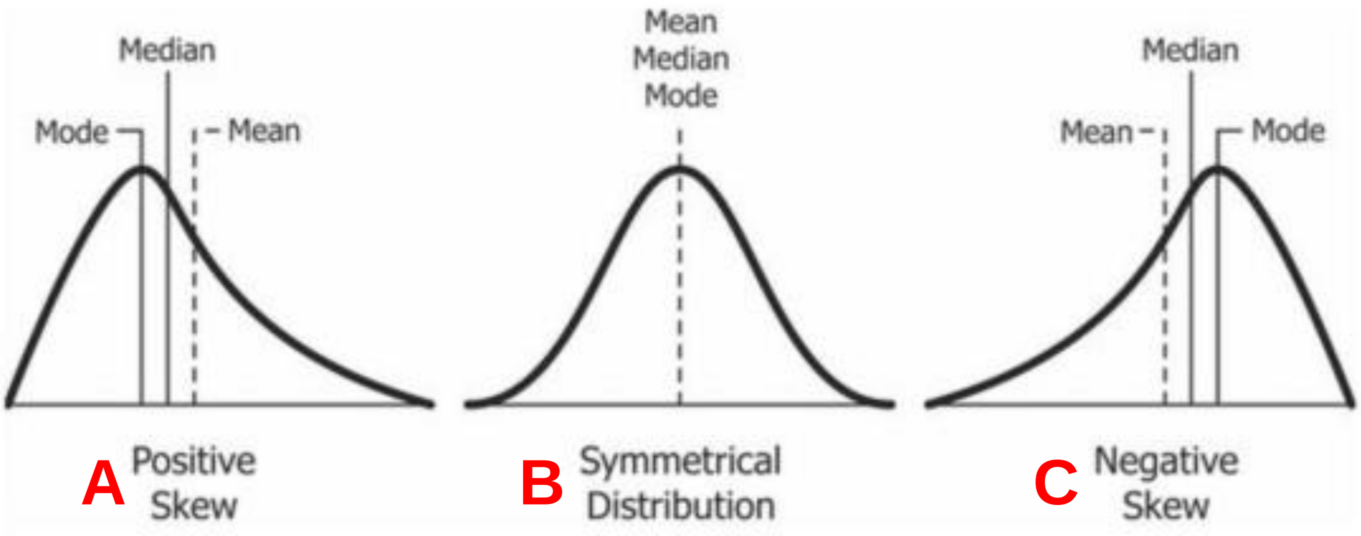
Different images, same histogram!

- The following images have the same histogram:

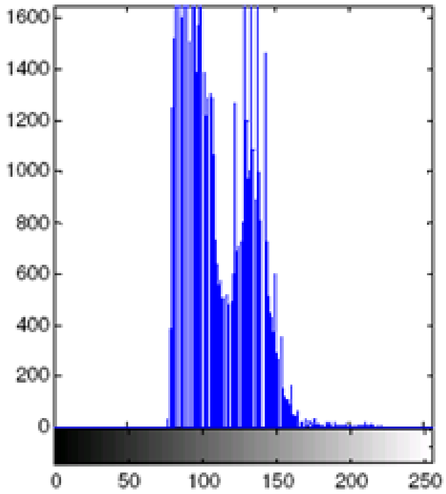
- Histogram reflects the **pixel intensity distribution**, not the spatial distribution!
- Can we reconstruct an image from a histogram?

Image histogram!

- What is the pattern of the histogram for the images below?



Problem with Contrast

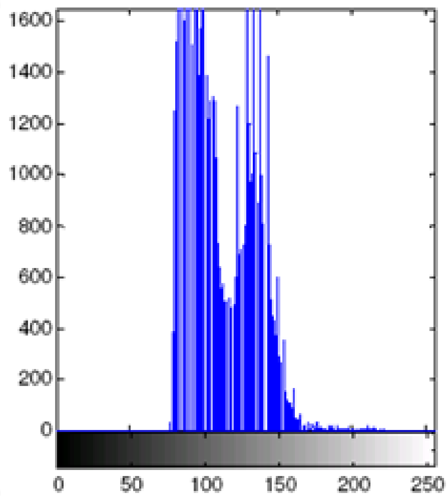


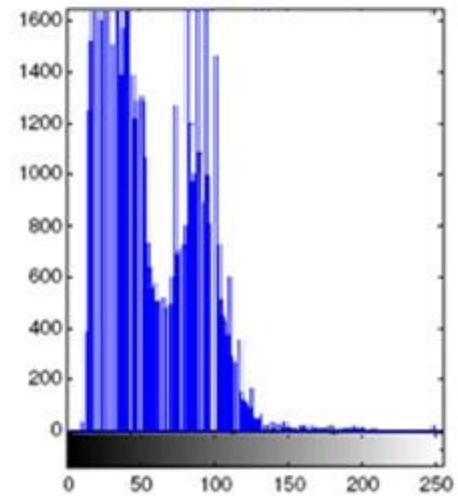
- How can we solve this problem of low/poor contrast in image?

You can attempt to find a point-based transformation function that can perform some kind of linear stretching…

Problem with Contrast

- Linear stretching

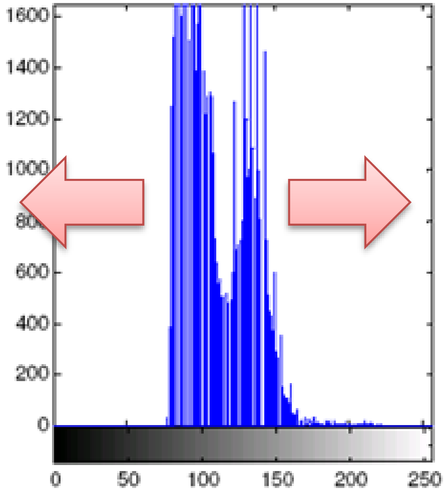




Is it good enough?

Histogram Equalization

- **Histogram EQUALization**
  - Aim: To **"equalize"** the histogram, to **"flatten"**, **"distribute as uniform as possible"** in an automatic way



  - **Idea: Adjust** probability density function of the original histogram so that the probabilities spread equally

Discrete Implementation

- Transforms an image with an arbitrary histogram to an image that has a somewhat flat histogram

1. Find the cumulative distribution (CD) of gray level l,

$$\tilde{g}(l) = \sum_{k=0}^{l} p_F(k), l = 0,1, \dots, K-1$$

2. Convert the CD values to the max possible range of values [0, L–1] by quantization

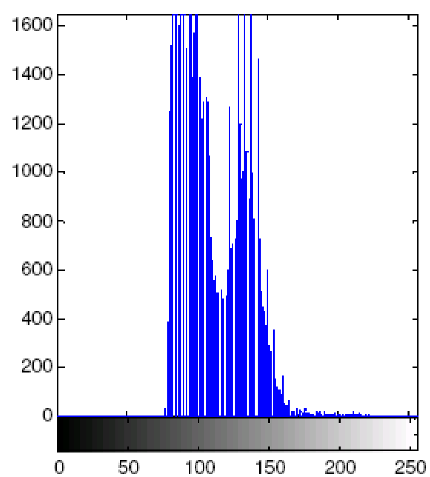
$$g(l) = \left\lfloor \left(\sum_{k=0}^{l} p_F(k) \right) * (L-1) \right\rfloor$$

3. Map the old pixel values to the new transformed values.

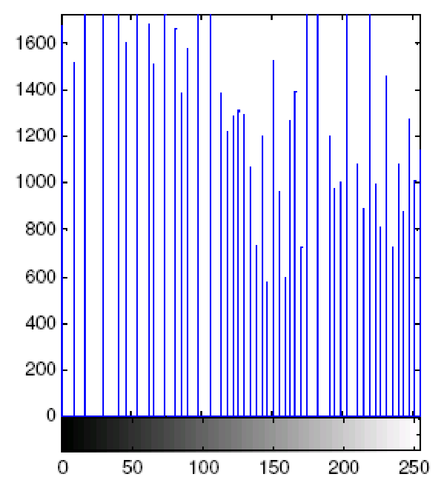
Correcting the Pouting Child

Original image with low contrast

Enhanced image



Original girl image with low contrast



Enhancement image with histogram equalization

Example: Discrete Implementation

0	76	0.19	0.19
1	100	0.25	0.44
2	84	0.21	0.65
3	64	0.16	0.81
4	32	0.08	0.89
5	24	0.06	0.95
6	12	0.03	0.98
7	8	0.02	1.00

Example: Discrete Implementation

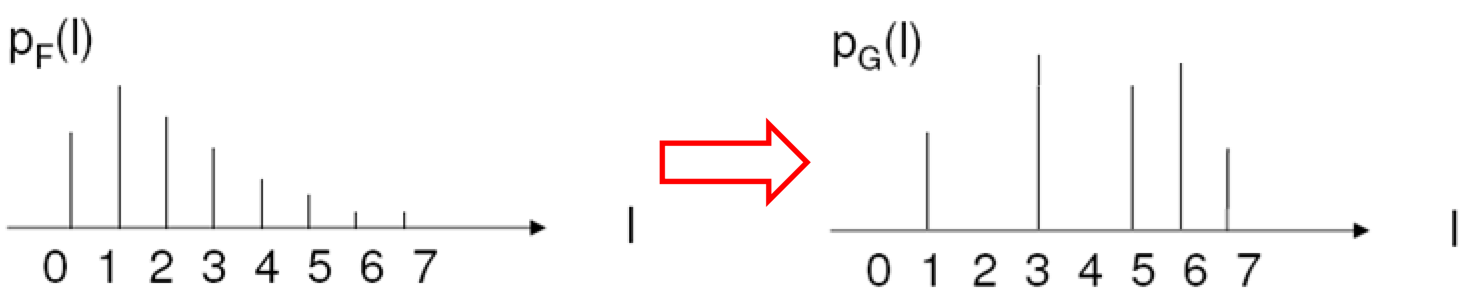
0	76	0.19	0.19	[1.33] = 1
1	100	0.25	0.44	[3.08] = 3
2	84	0.21	0.65	[4.55] = 5
3	64	0.16	0.81	[5.67] = 6
4	32	0.08	0.89	[6.03] = 6
5	24	0.06	0.95	[6.65] = 7
6	12	0.03	0.98	[6.86] = 7
7	8	0.02	1.00	[7] = 7

Example: Discrete Implementation

0	76	0.19	0.19	[1.33] = 1	0	0
1	100	0.25	0.44	[3.08] = 3	0.19	1
2	84	0.21	0.65	[4.55] = 5	0	2
3	64	0.16	0.81	[5.67] = 6	0.25	3
4	32	0.08	0.89	[6.03] = 6	0	4
5	24	0.06	0.95	[6.65] = 7	0.21	5
6	12	0.03	0.98	[6.86] = 7	0.16+0.08=0.24	6
7	8	0.02	1.00	[7] = 7	0.06+0.03+0.02=0.11	7

Example: Discrete Implementation

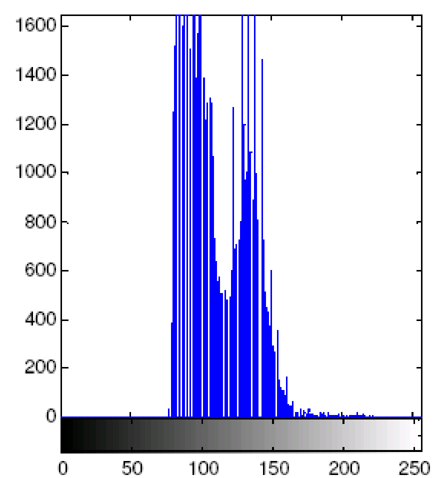
0	76	0.19	0.19	[1.33] = 1	0	0
1	100	0.25	0.44	[3.08] = 3	0.19	1
2	84	0.21	0.65	[4.55] = 5	0	2
3	64	0.16	0.81	[5.67] = 6	0.25	3
4	32	0.08	0.89	[6.03] = 6	0	4
5	24	0.06	0.95	[6.65] = 7	0.21	5
6	12	0.03	0.98	[6.86] = 7	0.16+0.08=0.24	6
7	8	0.02	1.00	[7] = 7	0.06+0.03+0.02=0.11	7



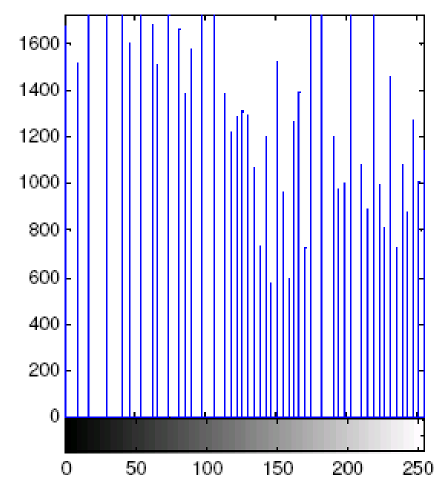
Correcting the Pouting Child

Original image with low contrast

Enhanced image



Original girl image with low contrast



Enhancement image with histogram equalization

Summary

- **Image Formation**
  - Pixels
  - Sampling and Quantization
- **Pixel (point)-based Processing**
- **Image Histograms**
  - Histogram Equalization

Recommended Reading

- [Gonzalez&Woods] Chapter 2 & 3