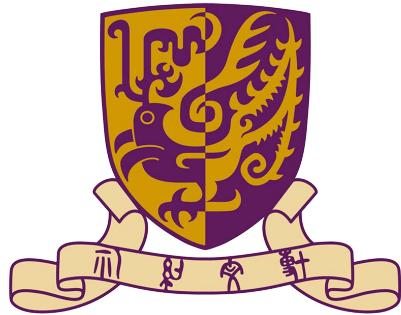


EIE4512 - Digital Image Processing

Intensity Transformation



香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Zhen Li

lizhen@cuhk.edu.cn

School of Science and Engineering

The Chinese University of Hong Kong, Shen Zhen

Januray 15 - 17, 2019

Spatial Domain Process

- Spatial domain representation
 - Image as a function of spatial coordinates $f(x, y)$
- Spatial domain process
 - $g(x, y) = T[f(x, y)]$
- Spatial domain processing is the foundation of many image enhancement/restoration technologies.

Outline

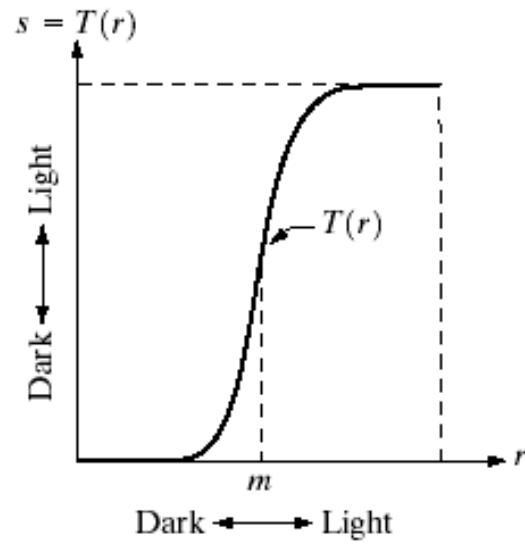
- Intensity Transformation (3.2)
- Histogram Processing (3.3)
- Spatial Filtering (3.4 – 3.6)
 - Linear filters
 - Order-statistics filters

Intensity Transformation

- Definition
 - Pixels are transformed independently.
 - The pixel value $g(x, y)$ is a function of the source pixel value $f(x, y)$.
- The intensity transformation is defined by an intensity transformation function T

$$s = T(r)$$

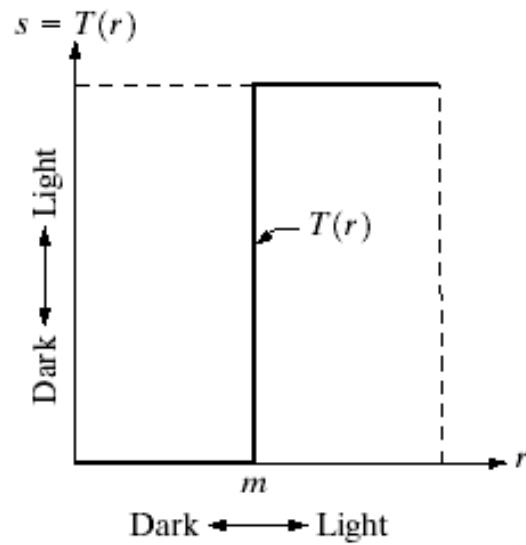
Intensity Transformation (Example)



Contrast stretching



Intensity Transformation (Example)



Thresholding



Intensity Transformation Functions

- Image Negatives
- Power-Law transformation
- Piecewise-Linear transformation



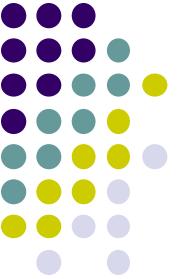
Point Operations

- Point operations changes a pixel's intensity value according to some function (don't care about pixel's neighbor)

$$a' \leftarrow f(a)$$

$$I'(u, v) \leftarrow f(I(u, v))$$

- Also called a **homogeneous operation**
- New pixel intensity **depends on**
 - Pixel's previous intensity $I(u, v)$
 - Mapping function $f()$
- **Does not depend on**
 - Pixel's location (u, v)
 - Intensities of neighboring pixels



Some Homogeneous Point Operations

- Addition (Changes brightness)

$$f(p) = p + k \quad \text{E.g. } f_{\text{bright}}(p) = p + 10$$

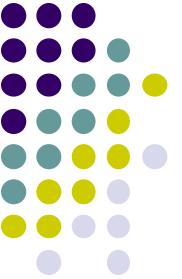
- Multiplication (Stretches/shrinks image contrast range)

$$f(p) = k \times p \quad \text{E.g. } f_{\text{contrast}}(p) = p \times 1.5$$

- Real-valued functions

$$\exp(x), \log(x), (1/x), x^k, \text{etc.}$$

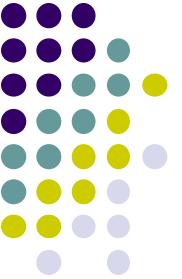
- Quantizing pixel values
- Global thresholding
- Gamma correction



Point Operation Pseudocode

- **Input:** Image with pixel intensities $I(u,v)$ defined on $[1 .. w] \times [1 .. H]$
- **Output:** Image with pixel intensities $I'(u,v)$

```
for v = 1 .. h
    for u = 1 .. w
        set  $I'(u, v) = f(I(u,v))$ 
```



Non-Homogeneous Point Operation

- New pixel value depends on:
 - Old value + **pixel's location (u,v)**

$$a' \leftarrow g(a, u, v)$$

$$I'(u, v) \leftarrow g(I(u, v), u, v)$$



Clamping

- Deals with pixel values outside displayable range
 - If ($a > 255$) $a = 255$;
 - If ($a < 0$) $a = 0$;
- Function below will **clamp** (force) all values to fall within range $[a,b]$

$$f(p) = \begin{cases} a & \text{if } p < a \\ p & \text{if } a \leq p \leq b \\ b & \text{if } p > b \end{cases}$$



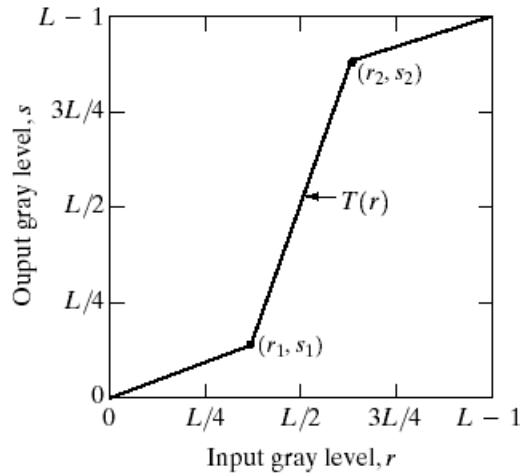
Example: Modify Intensity and Clamp

- Point operation: increase image contrast by 50% then clamp values above 255

```
1  public void run(ImageProcessor ip) {  
2      int w = ip.getWidth();  
3      int h = ip.getHeight();  
4  
5      for (int v = 0; v < h; v++) {  
6          for (int u = 0; u < w; u++) {  
7              int a = (int) (ip.get(u, v) * 1.5 + 0.5); ←  
8              if (a > 255)  
9                  a = 255;    // clamp to maximum value  
10             ip.set(u, v, a);  
11         }  
12     }  
13 }
```

Increase contrast
by 50%

Piecewise-Linear Contrast Stretching



a
b
c
d

FIGURE 3.10
Contrast stretching.
(a) Form of transformation function. (b) A low-contrast image. (c) Result of contrast stretching. (d) Result of thresholding. (Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra, Australia.)

Contrast Stretching: Math

Equations

$$\text{Let } T(r) = a \cdot r + b$$

$$a \cdot r_1 + b = s_1$$

$$a \cdot r_2 + b = s_2$$

Solutions

$$a = \frac{s_2 - s_1}{r_2 - r_1}$$

$$b = \frac{s_1 r_2 - s_2 r_1}{r_2 - r_1}$$

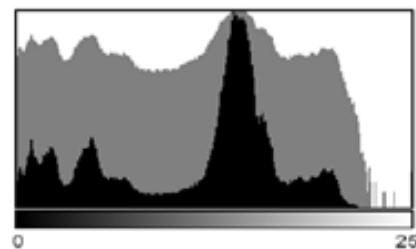


Inverting Images

$$f_{\text{invert}}(a) = -a + a_{\max} = a_{\max} - a$$

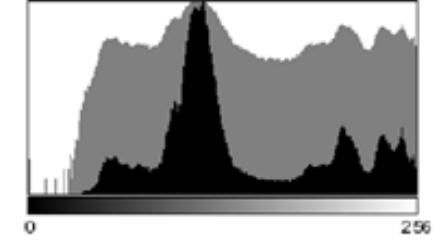
- 2 steps

1. Multiple intensity by -1
2. Add constant (e.g. a_{\max})
to put result in range
 $[0, a_{\max}]$



(a)

Original



(c)

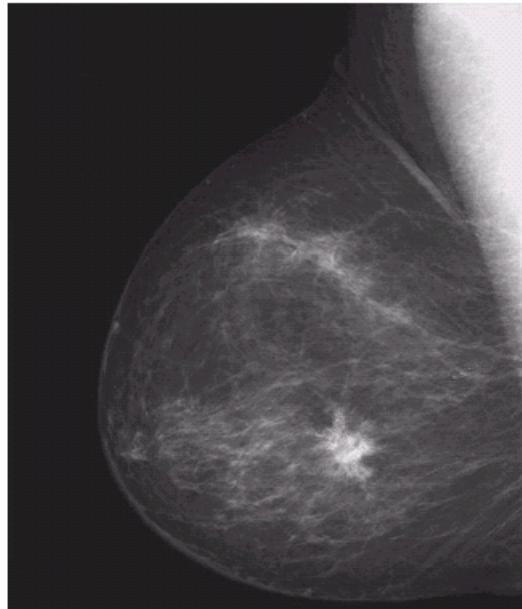
Inverted Image



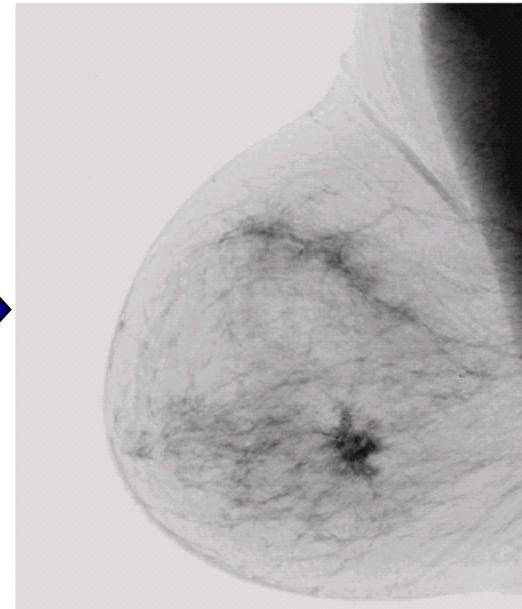
Image Negatives (Inverted Images)

- Image negatives useful for enhancing white or grey detail embedded in dark regions of an image
 - Note how much clearer the tissue is in the negative image of the mammogram below

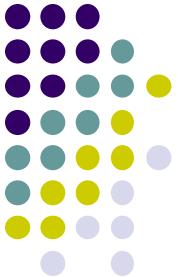
Original Image



$$s = 1.0 - r$$



Negative Image



Thresholding

- Input values below **threshold** a_{th} set to a_0
- Input values above **threshold** a_{th} set to a_1

$$f_{\text{threshold}}(a) = \begin{cases} a_0 & \text{for } a < a_{\text{th}} \\ a_1 & \text{for } a \geq a_{\text{th}} \end{cases}$$

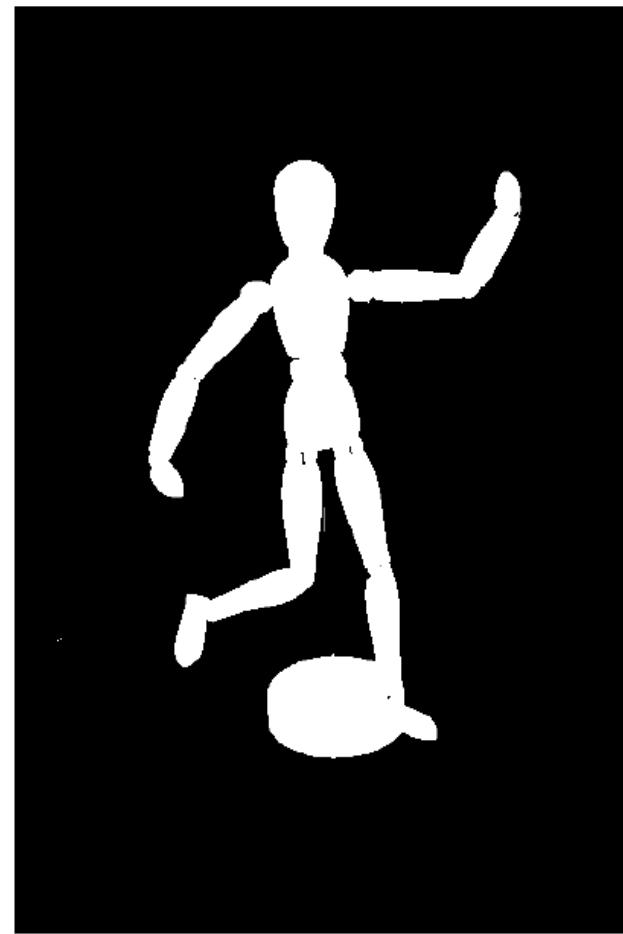
- Converts grayscale image to binary image (binarization) if
 - $a_0 = 0$
 - $a_1 = 1$
- Implemented as **im2bw()** and **multithresh()**
<https://goo.gl/fpWRvZ>



Thresholding Example



Original Image

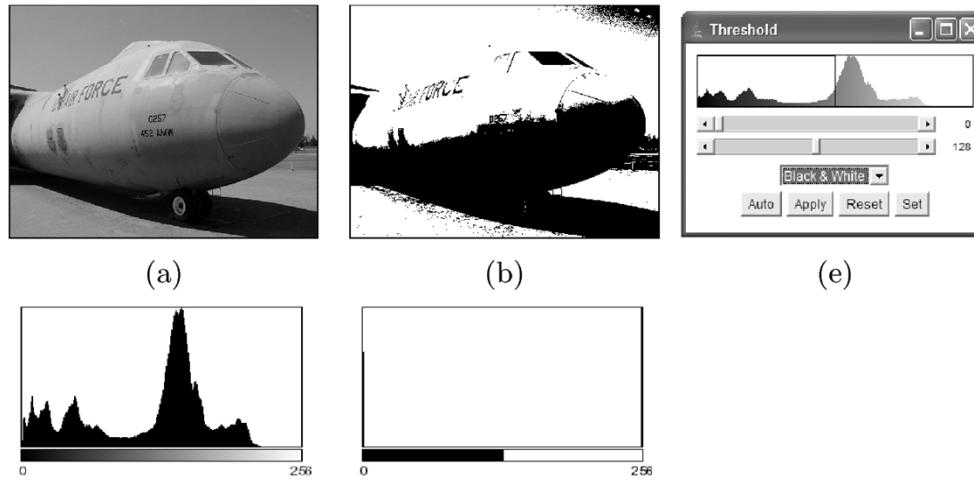


Thresholded Image

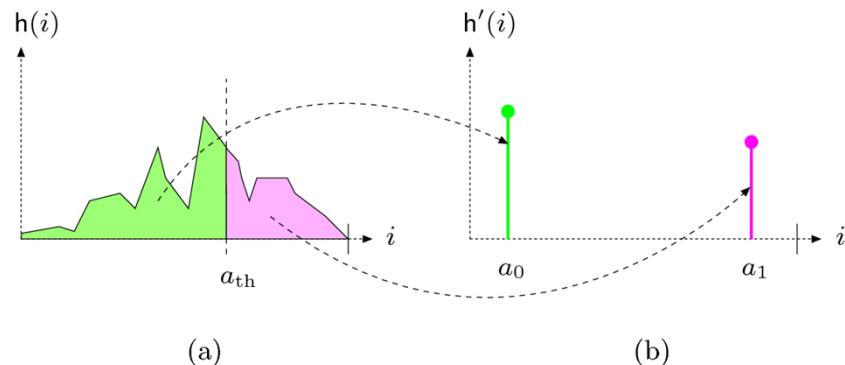


Thresholding and Histograms

- Example with $a_{th} = 128$



- Thresholding splits histogram, merges halves into a_0 a_1

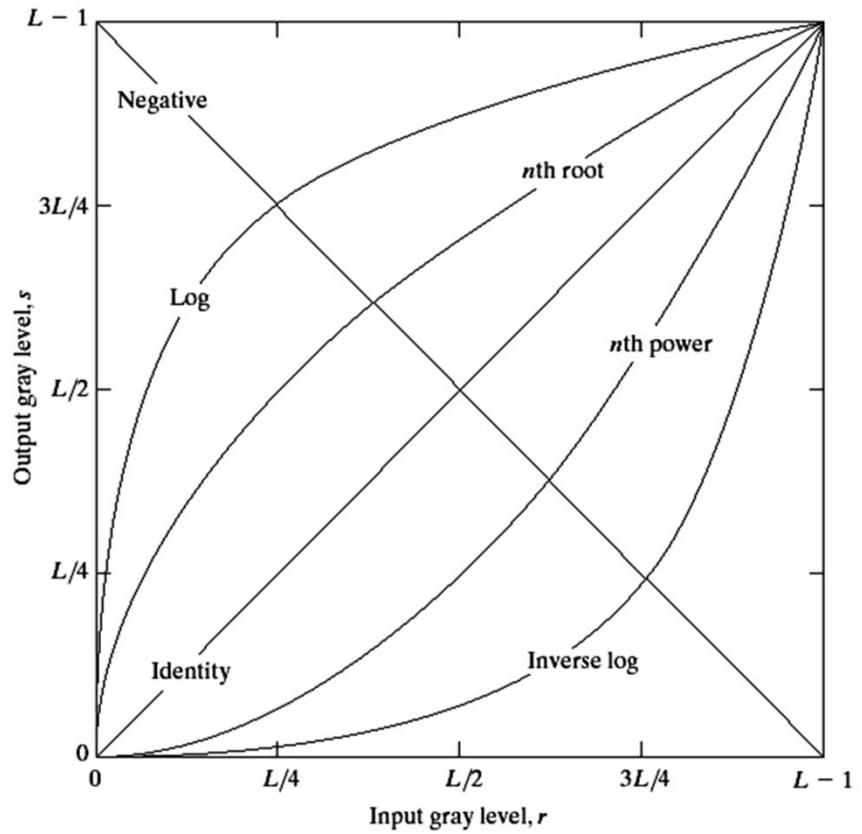


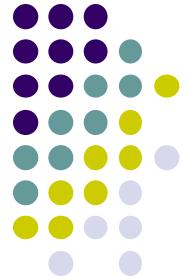


Basic Grey Level Transformations

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

- 3 most common gray level transformation:
 - Linear
 - Negative/Identity
 - Logarithmic
 - Log/Inverse log
 - Power law
 - n^{th} power/ n^{th} root



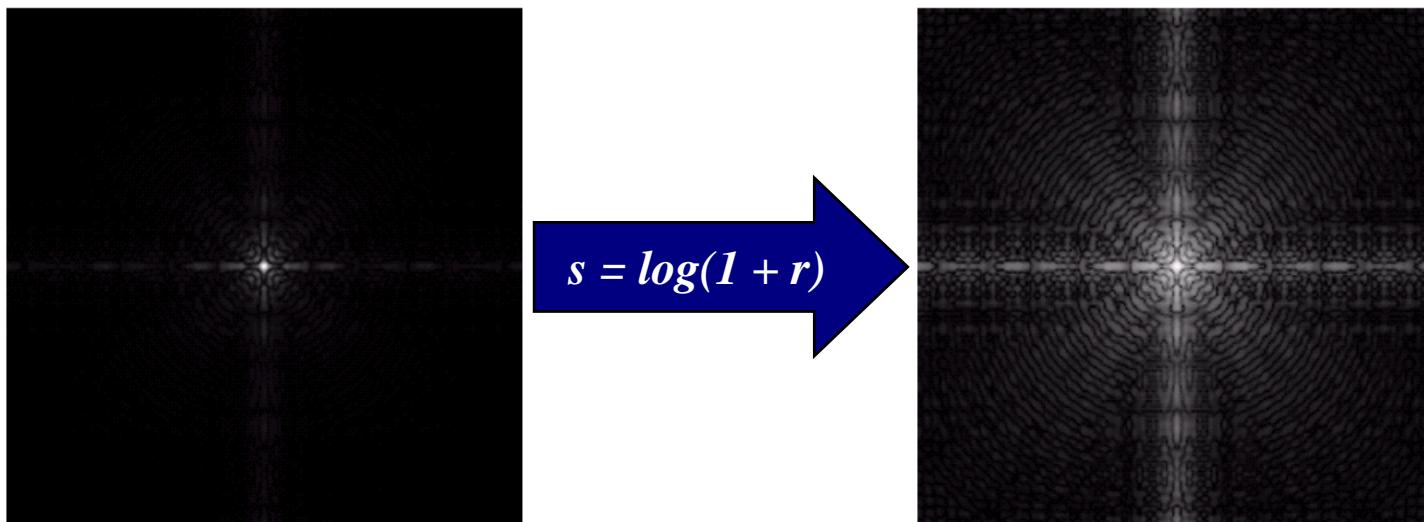


Logarithmic Transformations

- Maps narrow range of input levels => wider range of output values
- Inverse log transformation does opposite transformation
- The general form of the log transformation is

$$\text{New pixel value} \longrightarrow s = c * \log(1 + r) \longleftarrow \text{Old pixel value}$$

- Log transformation of Fourier transform shows more detail



Power Law Transformations

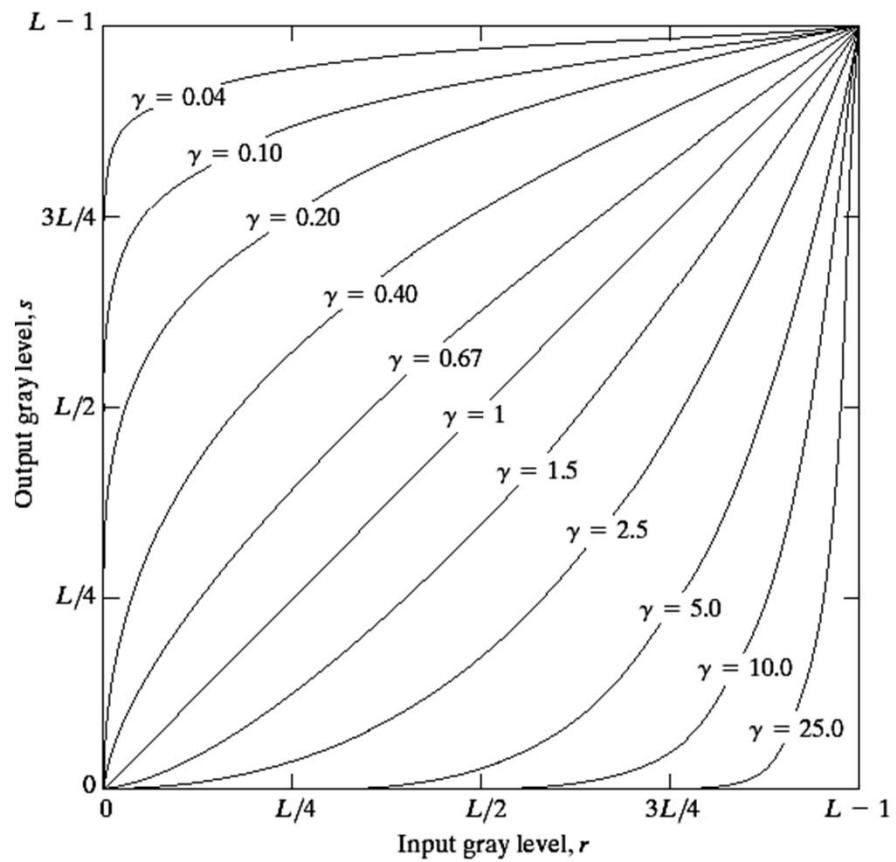


- Power law transformations have the form

$$s = c * r^\gamma$$

New pixel value Constant Old pixel value Power

- Map narrow range of dark input values into wider range of output values or vice versa
- Varying γ gives a whole family of curves



Power Law Example



- Magnetic Resonance (MR) image of fractured human spine
- Different power values highlight different details

Original



$$s = r^{-0.6}$$



$$r^{-0.4} = s$$

$$s = r^{-0.2}$$



Power Law



When $\gamma < 1$, stretch the darker range

Power Law

Original image



When $\gamma > 1$, stretch the lighter range

$\gamma = 3.0$

$\gamma = 4.0$



$\gamma = 5.0$



Intensity Windowing

- A clamp operation, then linearly stretching image intensities to fill possible range
- To window an image in $[a,b]$ with max intensity M

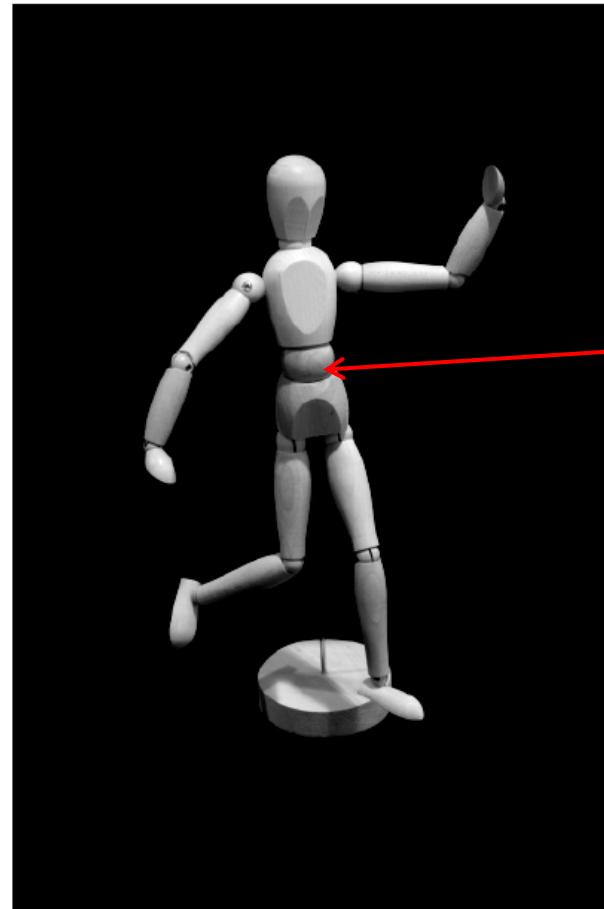
$$f(p) = \begin{cases} 0 & \text{if } p < a \\ M \times \frac{p-a}{b-a} & \text{if } a \leq p \leq b \\ M & \text{if } p > b \end{cases}$$



Intensity Windowing Example



Original Image

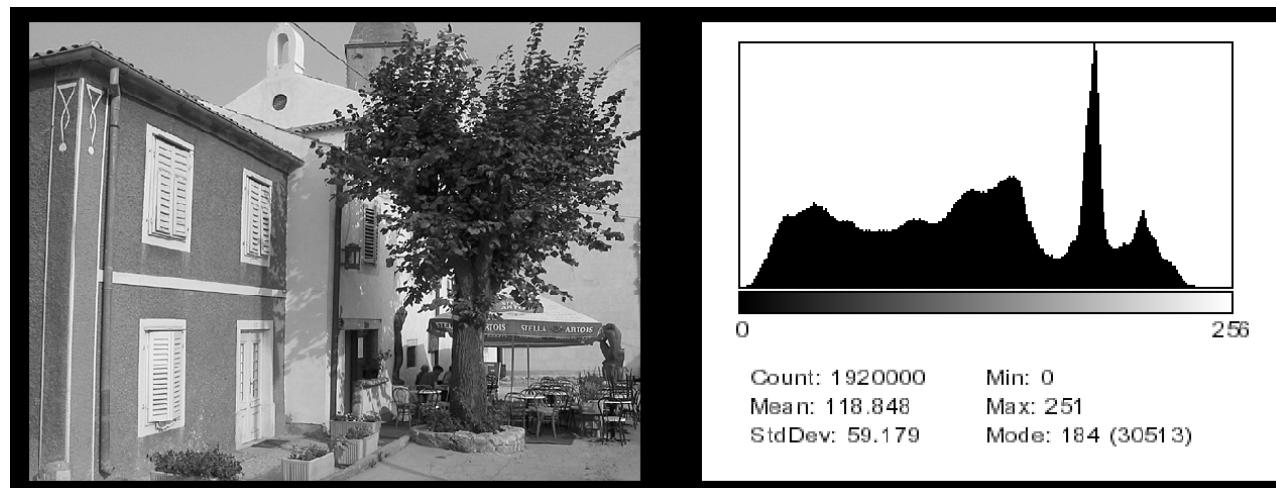


Windowed Image



Histograms

- Histograms plots how many times (frequency) each intensity value in image occurs
- Example:
 - Image (left) has 256 distinct gray levels (8 bits)
 - Histogram (right) shows frequency (how many times) each gray level occurs





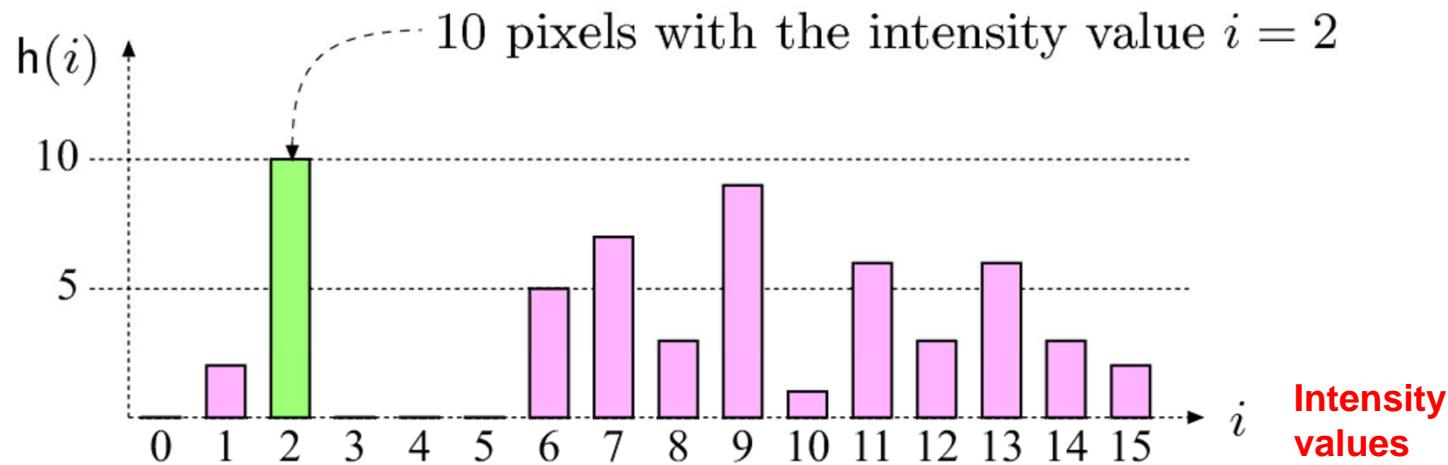
Histograms

- Many cameras display real time histograms of scene
- Helps avoid taking over-exposed pictures
- Also easier to detect types of processing previously applied to image





Histograms



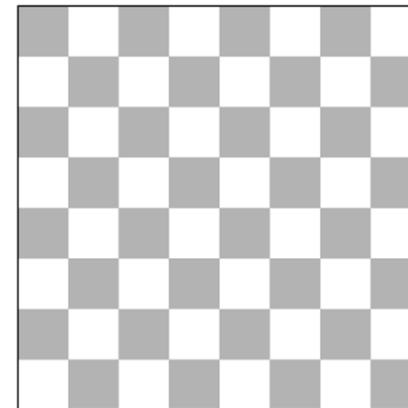
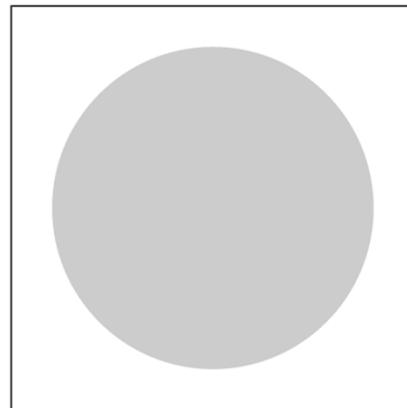
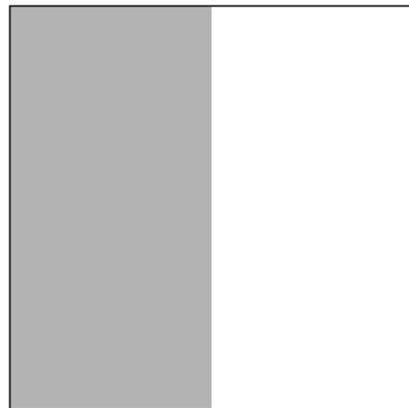
$h(i)$	0	2	10	0	0	0	5	7	3	9	1	6	3	6	3	2
i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

- E.g. $K = 16$, 10 pixels have intensity value = 2
- Histograms: only statistical information
- No indication of **location** of pixels



Histograms

- Different images can have **same** histogram
- 3 images below have same histogram



- Half of pixels are gray, half are white
 - Same histogram = same statistics
 - Distribution of intensities could be different
- Can we reconstruct image from histogram? No!



Histograms

- So, a histogram for a grayscale image with intensity values in range

$$I(u, v) \in [0, K - 1]$$

would contain exactly K entries

- E.g. 8-bit grayscale image, $K = 2^8 = 256$
- Each histogram entry is defined as:
$$h(i) = \text{number of pixels with intensity } i \text{ for all } 0 < i < K.$$
- E.g: $h(255) = \text{number of pixels with intensity } = 255$
- Formal definition

$$h(i) = \text{card}\{(u, v) \mid I(u, v) = i\}$$

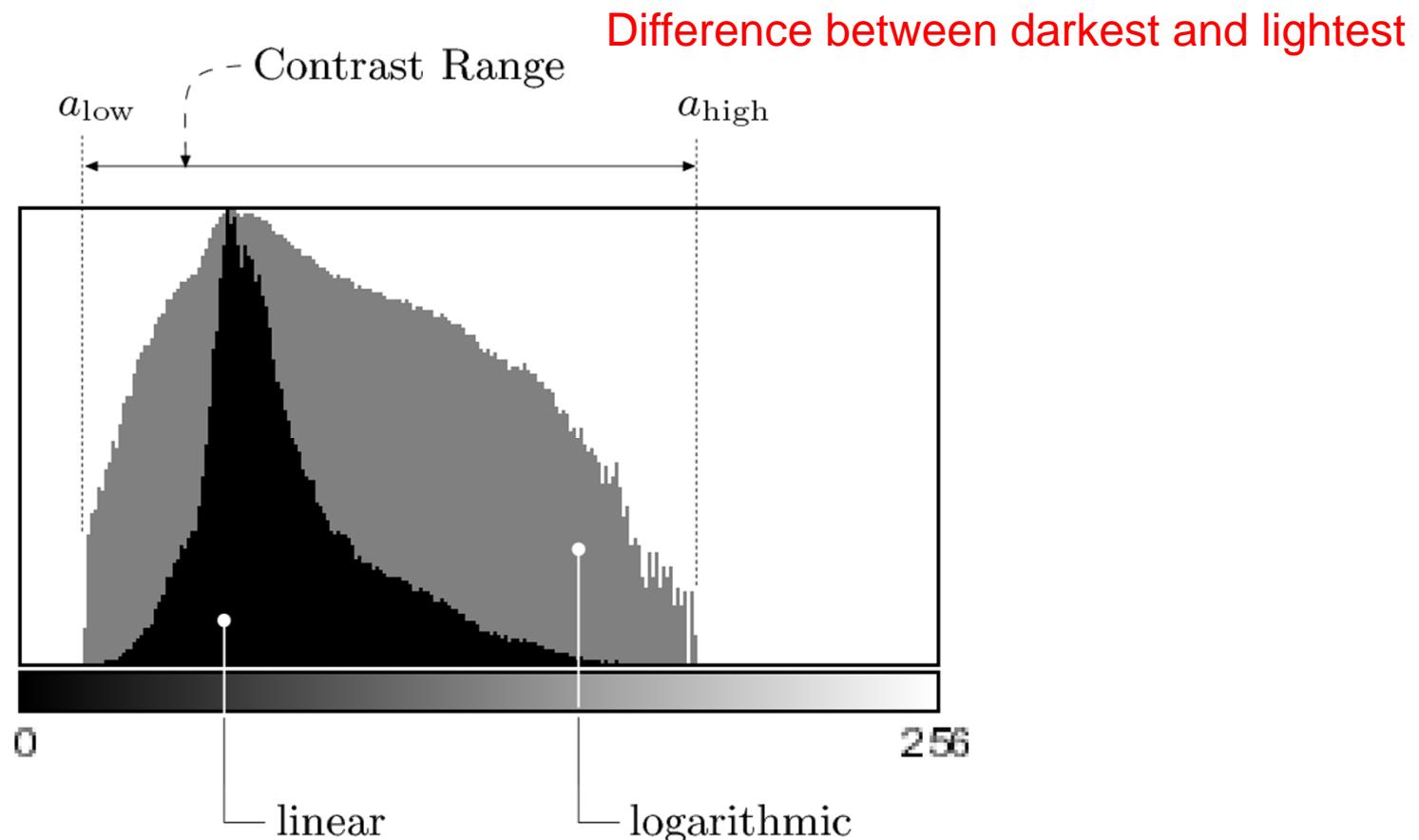
Number (size of set) of pixels

such that



Interpreting Histograms

- Log scale makes low values more visible





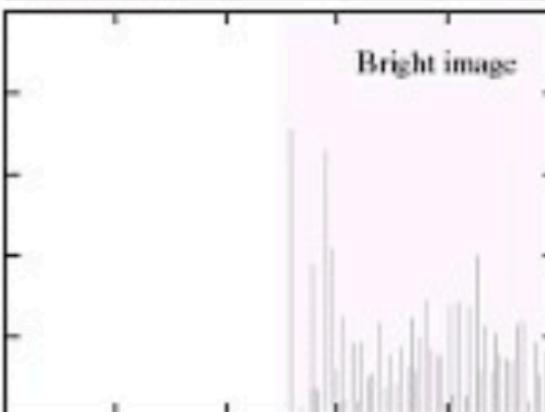
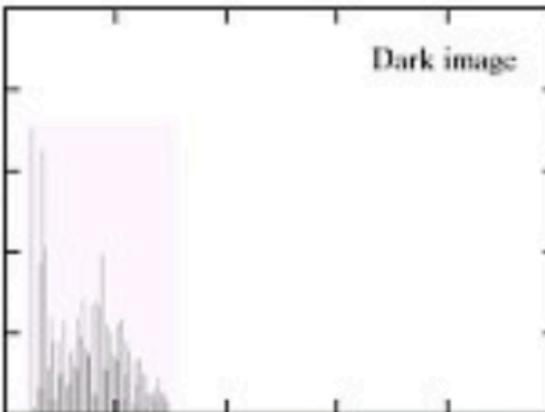
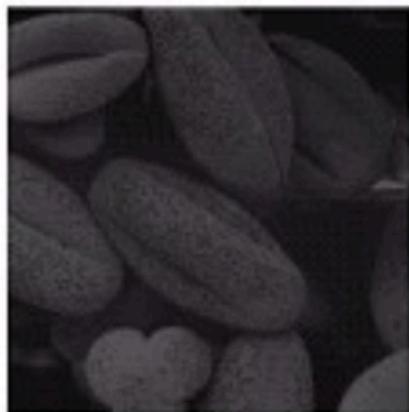
Histograms

- Histograms help detect image acquisition issues
- Problems with image can be identified on histogram
 - Over and under exposure
 - Brightness
 - Contrast
 - Dynamic Range
- Point operations can be used to alter histogram. E.g
 - Addition
 - Multiplication
 - Exp and Log
 - Intensity Windowing (Contrast Modification)

Significance of Histogram

- The basics for many spatial domain processing techniques.
- Often used for image enhancement.
- Information contained in the histograms is also useful for image compression.

Histogram Examples



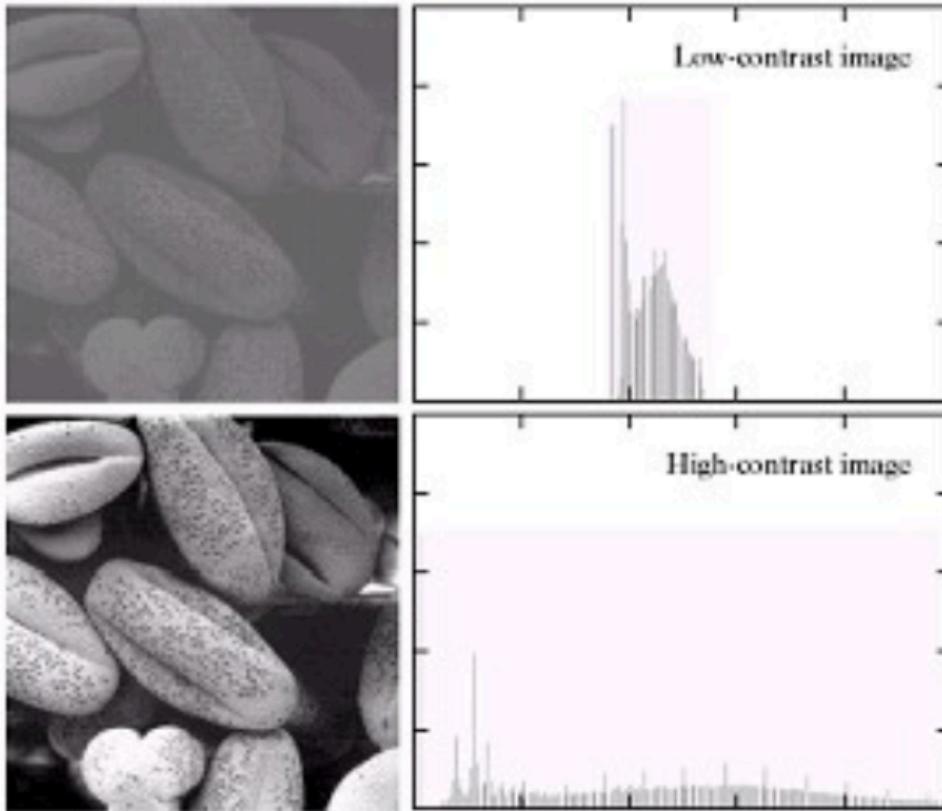
Dark image

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 Examples



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



Image Brightness

- Brightness of a grayscale image is the **average intensity** of all pixels in image

$$B(I) = \frac{1}{wh} \sum_{v=1}^h \sum_{u=1}^w I(u, v)$$

2. Divide by total number of pixels

1. Sum up all pixel intensities

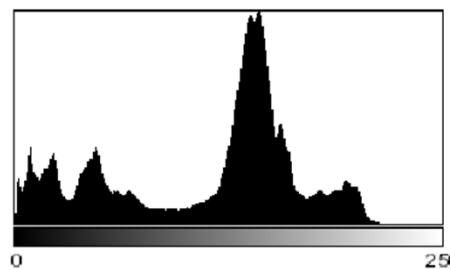


Detecting Bad Exposure using Histograms

Exposure? Are intensity values spread **(good)** out or bunched up **(bad)**

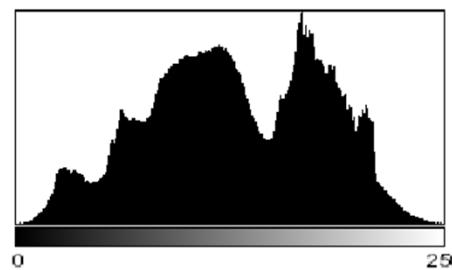


Image



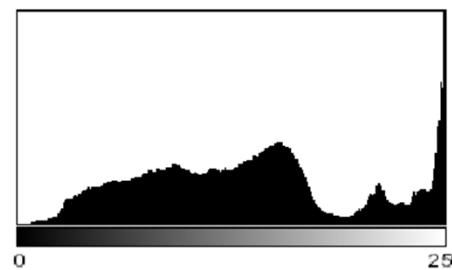
(a)

Underexposed



(b)

Properly Exposed



(c)

Overexposed

Histogram



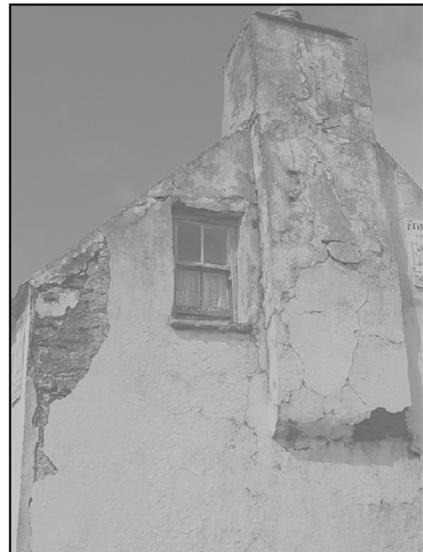
Image Contrast

- The contrast of a grayscale image indicates how easily objects in the image can be distinguished
- **High contrast image:** many distinct intensity values
- **Low contrast:** image uses few intensity values

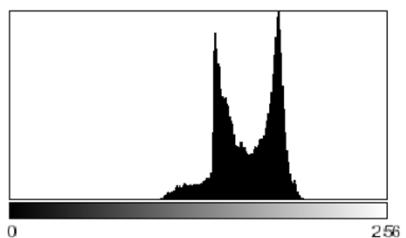


Histograms and Contrast

Good Contrast? Widely spread intensity values
+ large difference between min and max intensity values

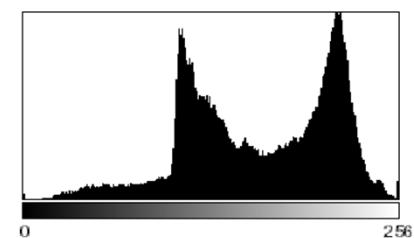


Image



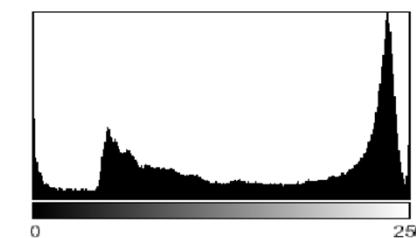
(a)

Low contrast



(b)

Normal contrast



(c)

High contrast

Histogram



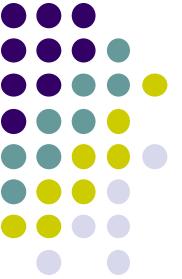
Contrast Equation?

- Many different equations for contrast exist
- Examples:

$$\text{Contrast} = \frac{\text{Change in Luminance}}{\text{Average Luminance}}$$

- Michalson's equation for contrast

$$C_M(I) = \frac{\max(I) - \min(I)}{\max(I) + \min(I)}$$



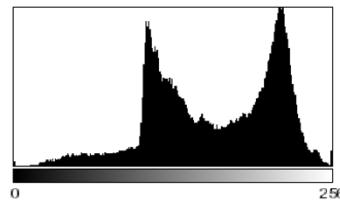
Contrast Equation?

- These equations work well for simple images with 2 luminances (i.e. uniform foreground and background)
- Does not work well for complex scenes with many luminances or if min and max intensities are small



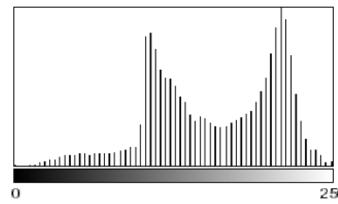
Histograms and Dynamic Range

- **Dynamic Range:** Number of distinct pixels in image



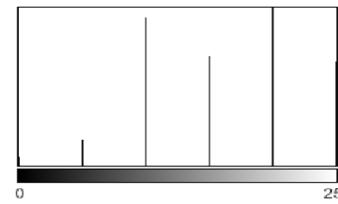
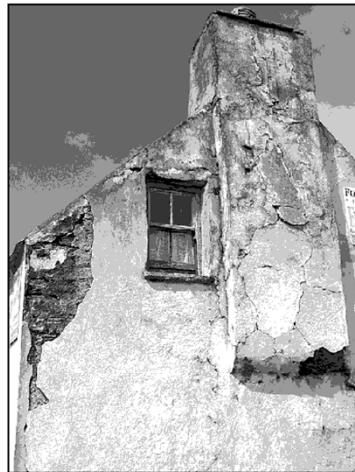
(a)

High Dynamic Range



(b)

Low Dynamic Range
(64 intensities)



(c)

Extremely low
Dynamic Range
(6 intensity values)

- Difficult to increase image dynamic range (e.g. interpolation)
- HDR (12-14 bits) capture typical, then down-sample



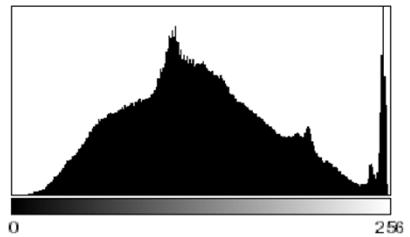
High Dynamic Range Imaging

- **High dynamic range** means very bright and very dark parts in a single image (many distinct values)
- Dynamic range in photographed scene may exceed number of available bits to represent pixels
- Solution:
 - Capture multiple images at different exposures
 - Combine them using image processing

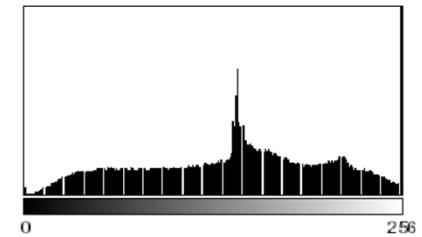


Detecting Image Defects using Histograms

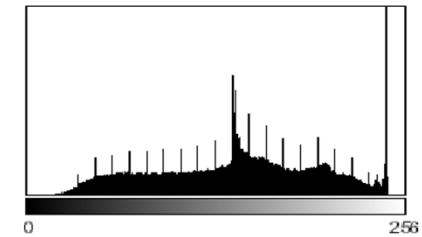
- No “best” histogram shape, depends on application
- Image defects
 - **Saturation:** scene illumination values outside the sensor’s range are set to its min or max values => results in spike at ends of histogram
 - **Spikes and Gaps in manipulated images** (not original).



(a)



(b)



(c)

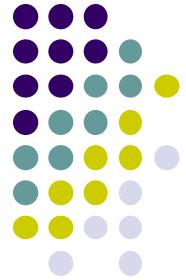
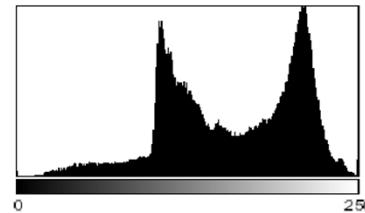


Image Defects: Effect of Image Compression

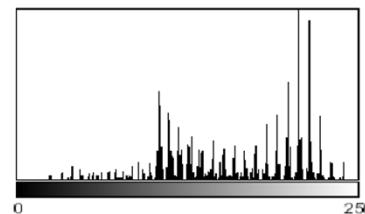
- Histograms show impact of image compression
- Example: in GIF compression (GIFs compress by removing horizontal redundancy. <https://goo.gl/4FKE3L>), dynamic range is reduced to only few intensities (quantization)

Original Image



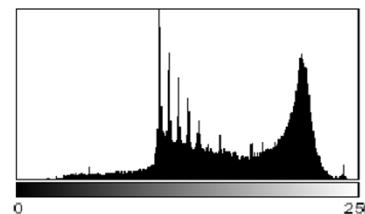
(a)

Original Histogram



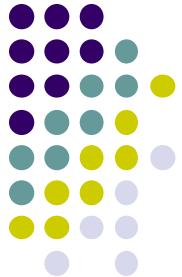
(b)

Histogram after GIF conversion



(c)

Fix? Scaling image by 50% and
Interpolating values recreates
some lost colors
But GIF artifacts still visible

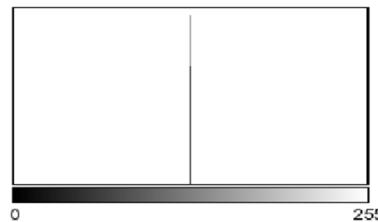


Effect of Image Compression

- Example: Effect of JPEG compression on line graphics
(PNG behaves like GIF in that it compresses horizontal patterns, but it also handles vertical patterns and dithering.)
- JPEG compression designed for color images



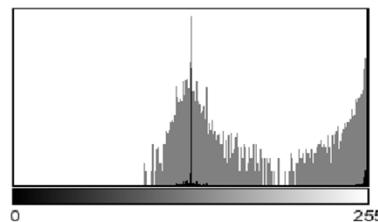
(a)



(b)



(c)



(d)

Original histogram has only 2 intensities (gray and white)

JPEG image appears dirty, fuzzy and blurred

Its Histogram contains gray values not in original



Computing Histograms

```
1 public class Compute_Histogram implements PlugInFilter {  
2  
3     public int setup(String arg, ImagePlus img) {  
4         return DOES_8G + NO_CHANGES; ← Receives 8-bit image,  
5     }                                         Will not change it  
6  
7     public void run(ImageProcessor ip) {  
8         int[] H = new int[256]; // histogram array ← Create array to store  
9         int w = ip.getWidth();                                histogram computed  
10        int h = ip.getHeight(); ← Get width and height of  
11  
12        for (int v = 0; v < h; v++) {  
13            for (int u = 0; u < w; u++) {  
14                int i = ip.getPixel(u,v); ← Iterate through image  
15                H[i] = H[i] + 1;                                pixels, add each  
16            }                                              intensity to appropriate  
17        }                                              histogram bin  
18        ... //histogram H[] can now be used  
19    }  
20  
21 } // end of class Compute_Histogram
```



Large Histograms: Binning

- High resolution image can yield very large histogram
- Example: 32-bit image = $2^{32} = 4,294,967,296$ columns
- Such a large histogram impractical to display
- Solution? Binning!
 - Combine **ranges of intensity values** into histogram columns

So, given the image $I : \Omega \rightarrow [0, K - 1]$, the binned histogram for I is the function

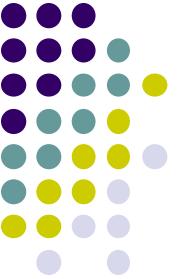
$$h(i) = \text{card}\{(u, v) \mid a_i \leq I(u, v) < a_{i+1}\},$$

$$\text{where } 0 = a_0 < a_1 < \dots < a_B = K.$$

Number (size of set) of pixels

such that

Pixel's intensity is
between a_i and a_{i+1}



Calculating Bin Size

- Typically use equal sized bins
- Bin size?
$$\frac{\text{Number of distinct values in image}}{\text{Number of bins}}$$
- Example: To create 256 bins from 14-bit image

$$\text{Bin size} = \frac{2^{14}}{256} = 64$$

$$\begin{aligned} h(0) &\leftarrow 0 \leq I(u, v) < 64 \\ h(1) &\leftarrow 64 \leq I(u, v) < 128 \\ h(2) &\leftarrow 128 \leq I(u, v) < 192 \\ &\vdots &&\vdots &&\vdots \\ h(j) &\leftarrow a_j \leq I(u, v) < a_{j+1} \\ &\vdots &&\vdots &&\vdots \\ h(255) &\leftarrow 16320 \leq I(u, v) < 16384 \end{aligned}$$



Binned Histogram

- To calculate which bin a pixel's intensity belongs to

$$\frac{I(u, v)}{k_B} = \frac{I(u, v)}{K/B} = I(u, v) \cdot \frac{B}{K}$$

- Previous example, $B = 256$, $K = 2^{14} = 16384$

```
1 int[] binnedHistogram(ImageProcessor ip) {  
2     int K = 256; // number of intensity values  
3     int B = 32; // size of histogram, must be defined  
4     int[] H = new int[B]; // histogram array  
5     int w = ip.getWidth();  
6     int h = ip.getHeight();  
7  
8     for (int v = 0; v < h; v++) {  
9         for (int u = 0; u < w; u++) {  
10             int a = ip.getPixel(u, v);  
11             int i = a * B / K; // integer operations only!  
12             H[i] = H[i] + 1;  
13         }  
14     }  
15     // return binned histogram  
16     return H;  
17 }
```

Create array to store histogram computed

Calculate which bin to add pixel's intensity

Increment corresponding histogram



Color Image Histograms

Two types:

1. Intensity histogram:

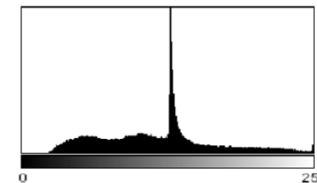
- Convert color image to gray scale
- Display histogram of gray scale

2. Individual Color Channel Histograms:

3 histograms (R,G,B)



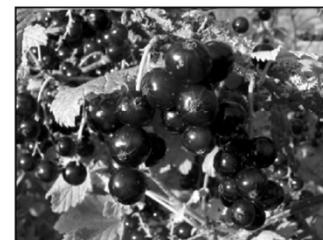
(a)



(b) h_{Lum}



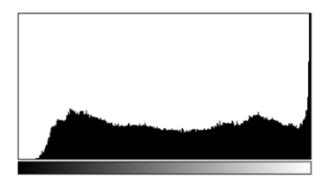
(c) R



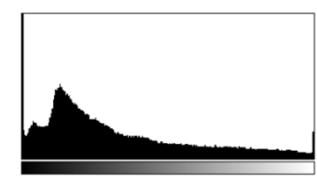
(d) G



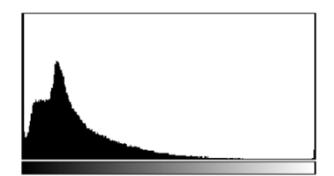
(e) B



(f) h_R



(g) h_G



(h) h_B



Color Image Histograms

- Both types of histograms provide useful information about lighting, contrast, dynamic range and saturation effects
- No information about the actual color distribution!
- Images with totally different RGB colors can have same R, G and B histograms
- Solution to this ambiguity is the **Combined Color Histogram**.
 - More on this later



Cumulative Histogram

- Useful for certain operations (e.g. histogram equalization) later
- Analogous to the **Cumulative Density Function (CDF)**
- Definition:

$$H(i) = \sum_{j=0}^i h(j) \quad \text{for } 0 \leq i < K$$

- Recursive definition

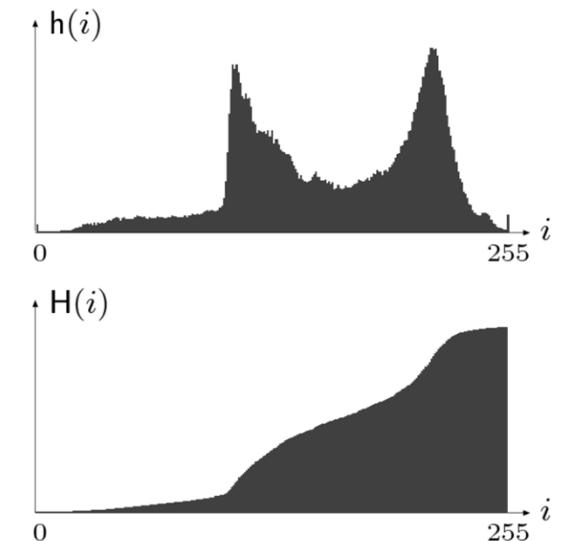
$$H(i) = \begin{cases} h(0) & \text{for } i = 0 \\ H(i-1) + h(i) & \text{for } 0 < i < K \end{cases}$$

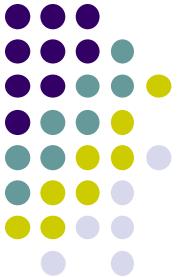
- Monotonically increasing

$$H(K-1) = \sum_{j=0}^{K-1} h(j) = M \cdot N$$

Last entry of
Cum. histogram

Total number of
pixels in image





Automatic Contrast Adjustment

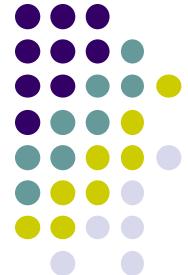
- Point operation that modifies pixel intensities such that available range of values is fully covered
- Algorithm:
 - Find high and lowest pixel intensities $a_{\text{low}}, a_{\text{high}}$
 - Linear stretching of intensity range



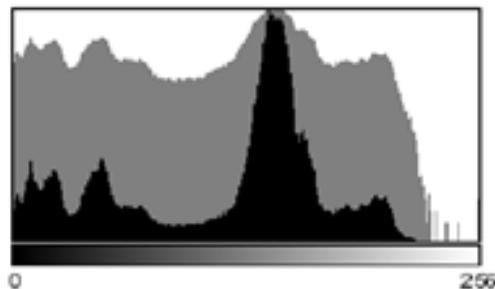
$$f_{\text{ac}}(a) = a_{\text{min}} + (a - a_{\text{low}}) \cdot \frac{a_{\text{max}} - a_{\text{min}}}{a_{\text{high}} - a_{\text{low}}}$$

If $a_{\text{min}} = 0$ and $a_{\text{max}} = 255$

$$f_{\text{ac}}(a) = (a - a_{\text{low}}) \cdot \frac{255}{a_{\text{high}} - a_{\text{low}}}$$

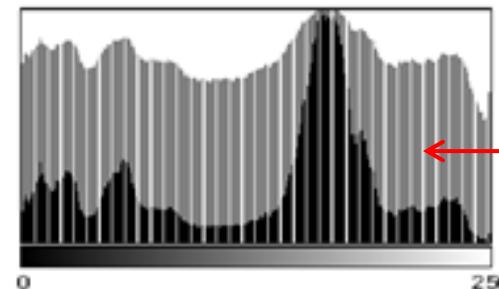


Effects of Automatic Contrast Adjustment



(a)

Original



(b)

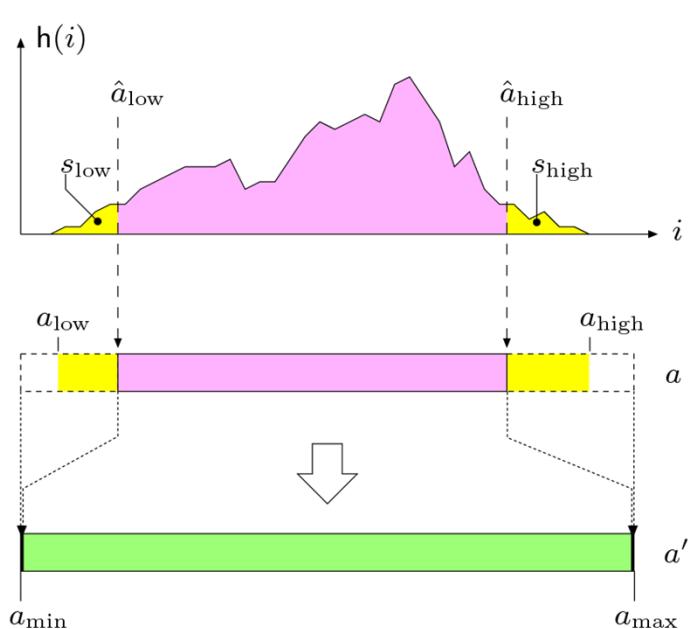
Result of automatic
Contrast Adjustment

Linearly stretching
range causes gaps
in histogram



Modified Contrast Adjustment

- Better to map only certain range of values
- Get rid of tails (usually noise) based on predefined percentiles ($s_{\text{low}}, s_{\text{high}}$)



$$\hat{a}_{\text{low}} = \min\{ i \mid H(i) \geq M \cdot N \cdot s_{\text{low}} \}$$

$$\hat{a}_{\text{high}} = \max\{ i \mid H(i) \leq M \cdot N \cdot (1 - s_{\text{high}}) \}$$

$$f_{\text{mac}}(a) = \begin{cases} a_{\text{min}} & \text{for } a \leq \hat{a}_{\text{low}} \\ a_{\text{min}} + (a - \hat{a}_{\text{low}}) \cdot \frac{a_{\text{max}} - a_{\text{min}}}{\hat{a}_{\text{high}} - \hat{a}_{\text{low}}} & \text{for } \hat{a}_{\text{low}} < a < \hat{a}_{\text{high}} \\ a_{\text{max}} & \text{for } a \geq \hat{a}_{\text{high}} \end{cases}$$

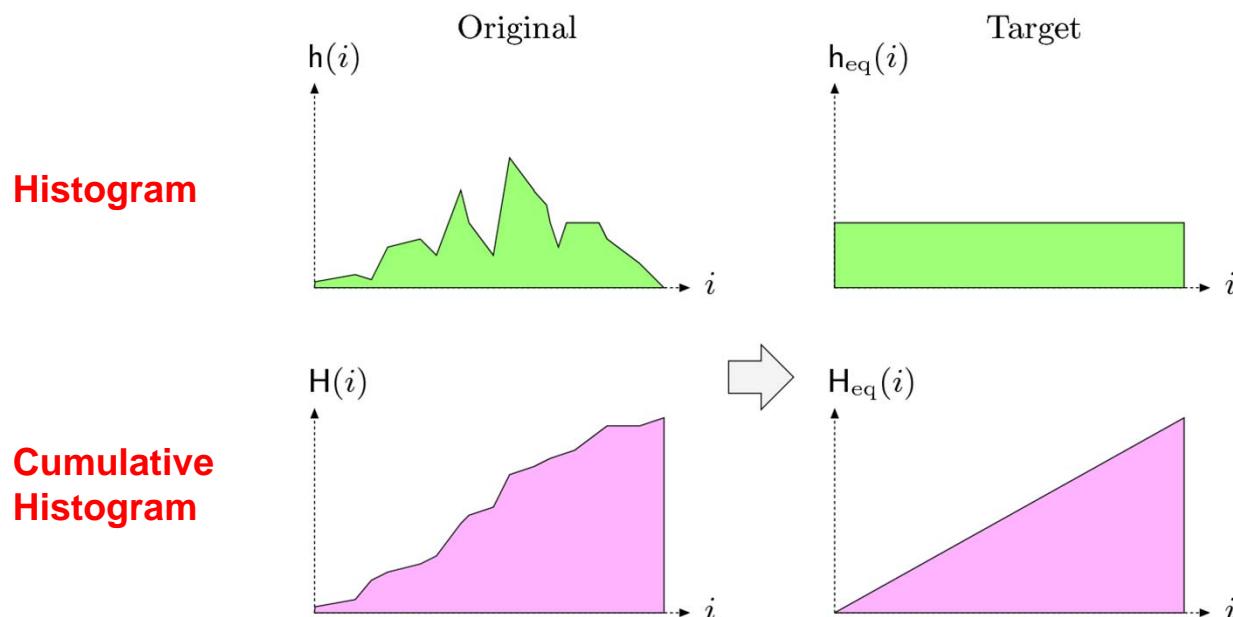
Histogram Equalization

- As the low-contrast image's histogram is narrow, if we distribute the histogram to wider range, the quality of the image will be improved.
- We can do it by adjusting the probability density function of the original histogram so that the probability spreads equally.
- The histogram equalization is an approach to enhance an image. Specifically, it designs a transformation $T(\cdot)$, such that the output values are uniformly distributed in $[0, L - 1]$.



Histogram Equalization

- Adjust 2 different images to make their histograms (intensity distributions) similar
- Apply a point operation that changes histogram of modified image into **uniform distribution**



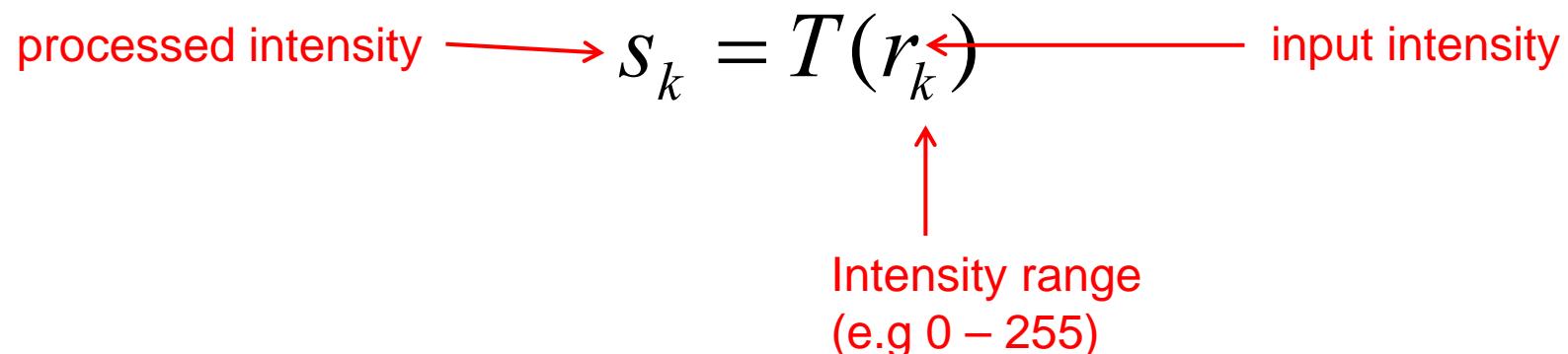


Histogram Equalization

Spreading out the frequencies in an image (or equalizing the image) is a simple way to improve dark or washed out images

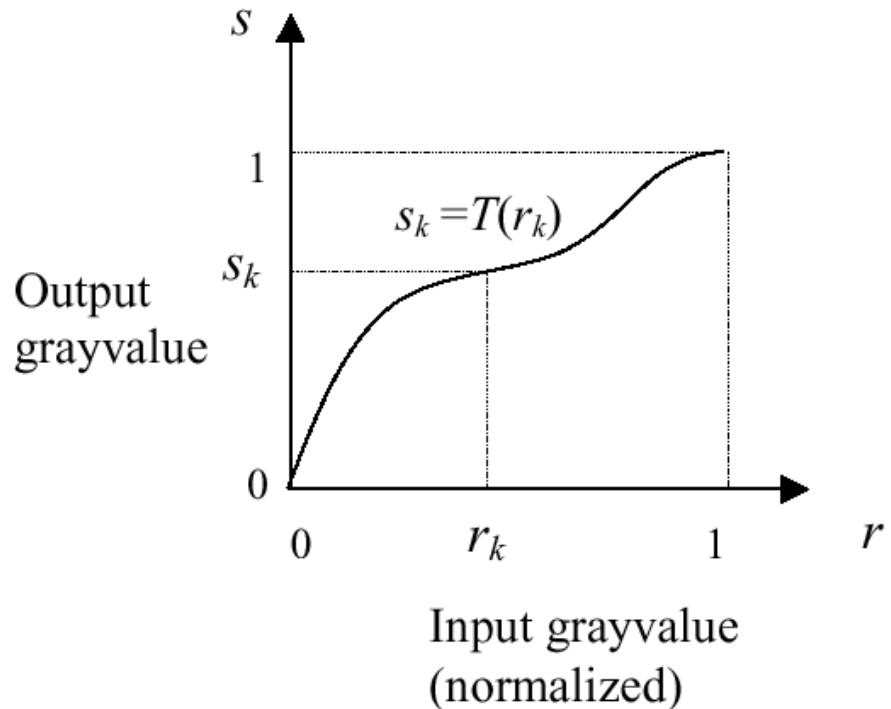
Can be expressed as a transformation of histogram

- r_k : input intensity
- s_k : processed intensity
- k : the intensity range
(e.g 0.0 – 1.0)



Assumptions

- $T(r)$ is monotonically increasing over $[0, L - 1]$.
- $T(r)$ maps $[0, L - 1]$ into $[0, L - 1]$, so as to preserve the range of allowed gray values.



Transformation Function for Histogram Equalization

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

- It can be seen that it satisfies the two conditions:
 - Monotonically increasing
 - The outputs are in the range [0, L-1].
- But our goal is to “equalize” the histogram. Is it achieved?

Probability Density Function

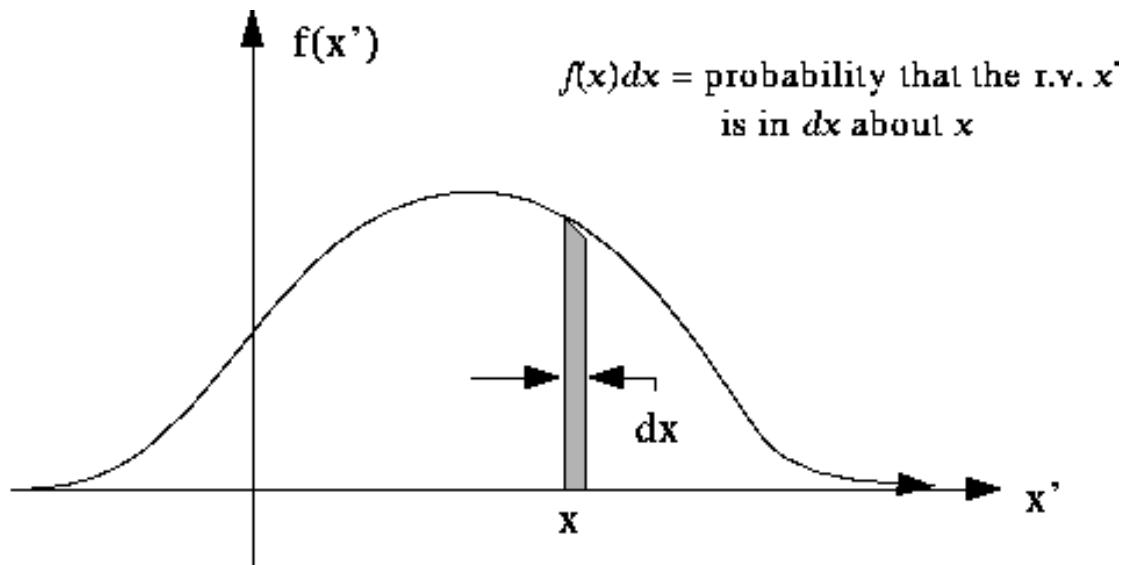


Figure 4. Typical Probability Distribution Function (*pdf*)

$$p(x) = \lim_{\Delta x \rightarrow 0} \frac{\Pr(x' \leq x + \Delta x) - \Pr(x' \leq x)}{\Delta x}$$

Transformed Density

This indicates that
 p_s is uniform

$$p_s(s) = p_r(r) \left| \frac{dr}{ds} \right| = p_r(r) \frac{1}{(L-1)p_r(r)} = \frac{1}{L-1}$$

$$\begin{aligned}\frac{ds}{dr} &= \frac{dT(r)}{dr} \\ &= (L-1) \frac{d}{dr} \left(\int_0^r p_r(w) dw \right) \\ &= (L-1)p_r(r)\end{aligned}$$

Discretize the Transform Rule

Continuous Version:

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

Discrete Version:

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

Histogram Equalization: Example

Suppose that a 3-bit image ($L=8$) of size 64×64 pixels ($MN = 4096$) has the intensity distribution shown in following table.

Get the histogram equalization transformation function and give the $p_s(s_k)$ for each s_k .

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 Equalization: Example

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

$$s_k = \frac{L-1}{MN} \sum_{j=0}^k n_j$$

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

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

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

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

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

Histogram Equalization: Example

final transform:

$$r_0 \rightarrow s_0 = 1 \Rightarrow 790 \text{ pixels map to } 1$$

$$r_1 \rightarrow s_1 = 3 \Rightarrow 1023 \text{ pixels map to } 3$$

$$r_2 \rightarrow s_2 = 5 \Rightarrow 850 \text{ pixels map to } 5$$

$$r_3 \rightarrow s_3 = 6 \Rightarrow 656 + 329 = 985 \text{ pixels map to } 6$$

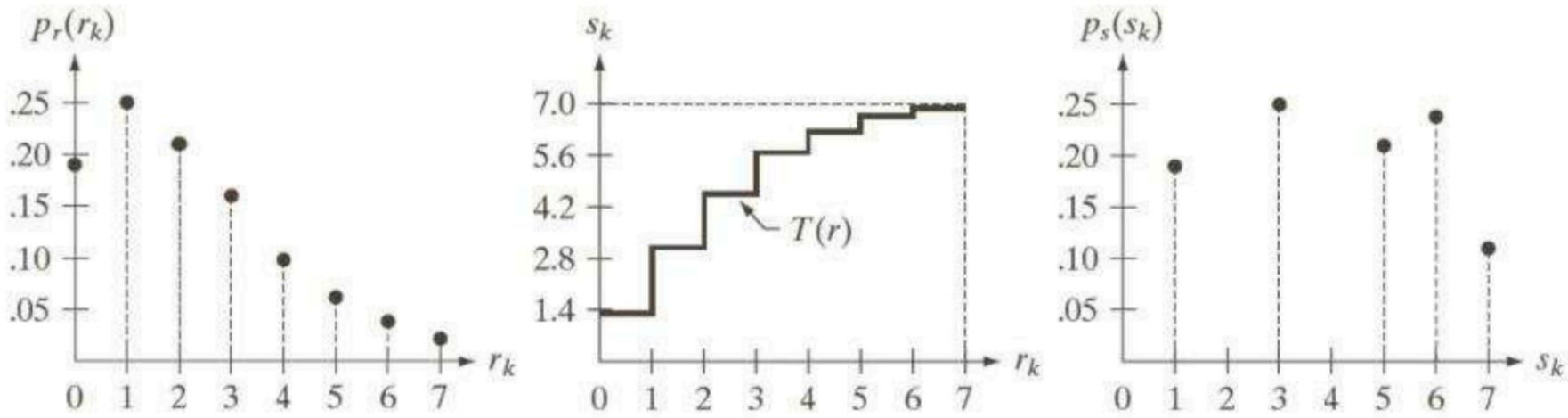
$$r_4 \rightarrow s_4 = 6 \Rightarrow 656 + 329 = 985 \text{ pixels map to } 6$$

$$r_5 \rightarrow s_5 = 7 \Rightarrow 245 + 122 + 81 = 458 \text{ pixels map to } 7$$

$$r_6 \rightarrow s_6 = 7 \Rightarrow 245 + 122 + 81 = 458 \text{ pixels map to } 7$$

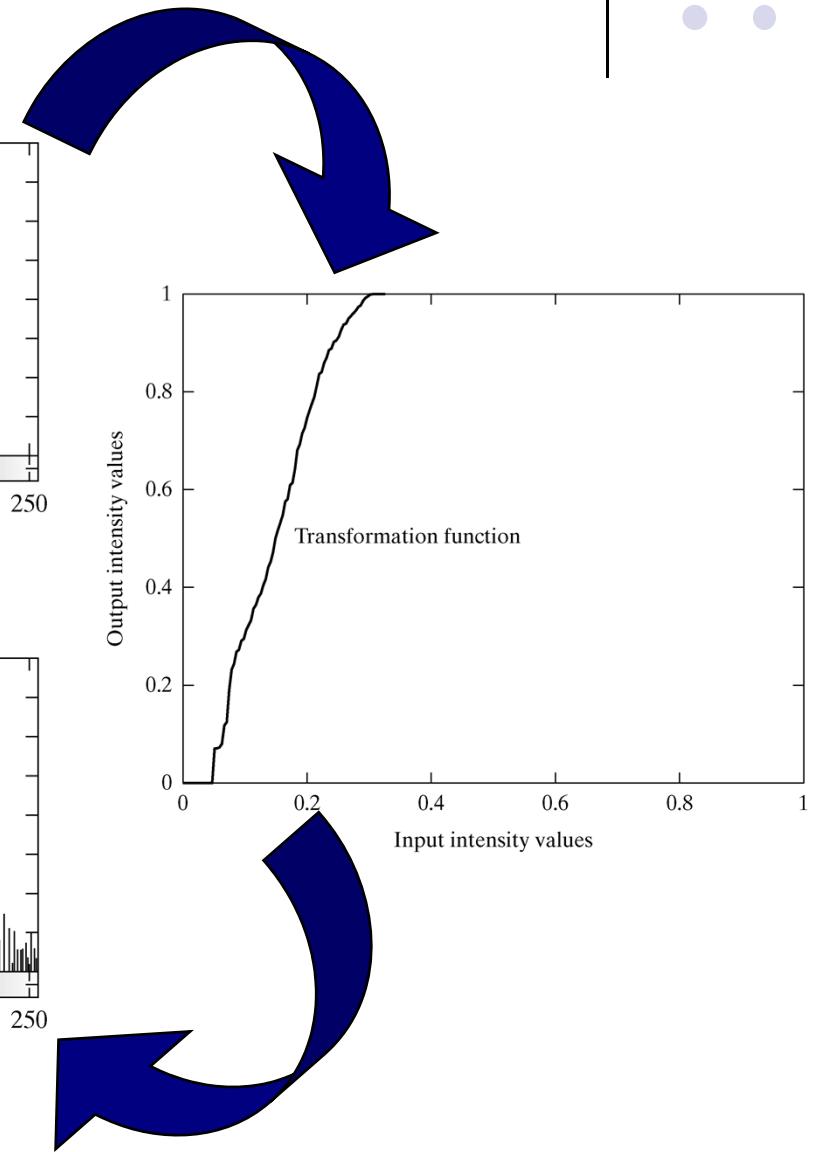
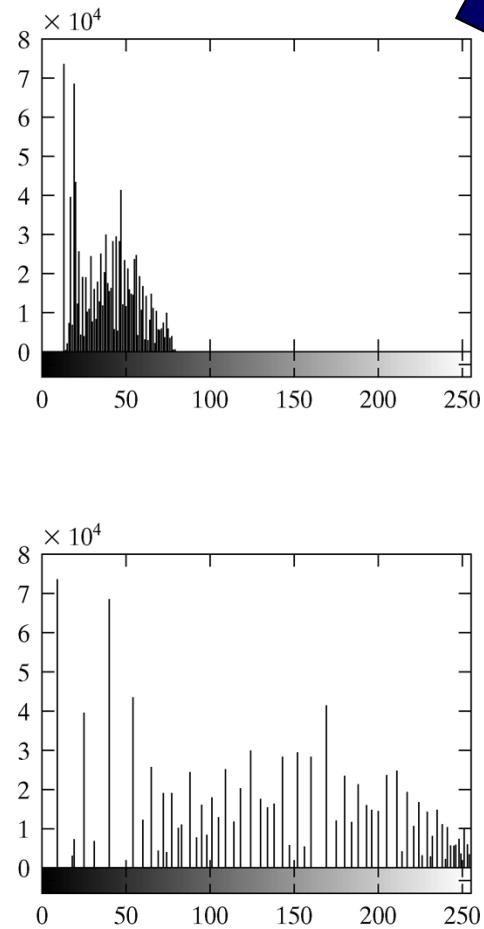
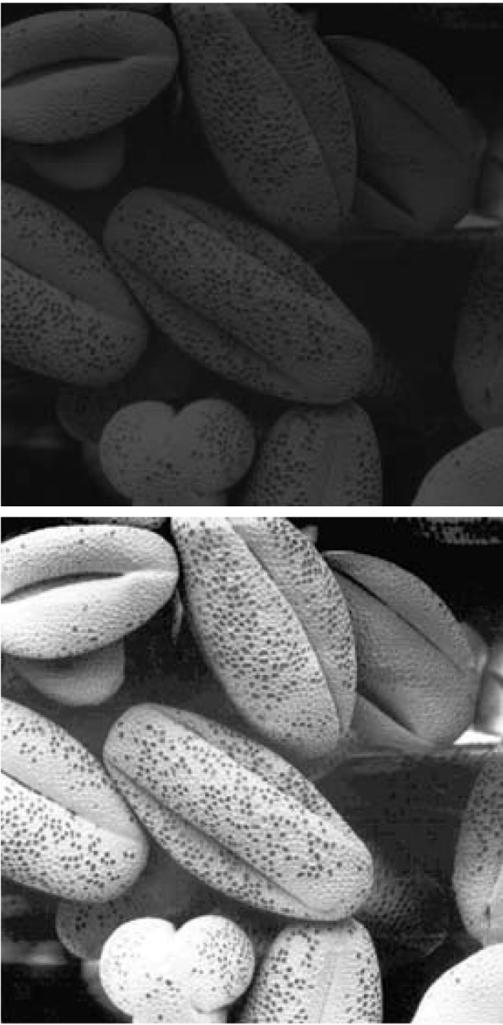
$$r_7 \rightarrow s_7 = 7 \Rightarrow 245 + 122 + 81 = 458 \text{ pixels map to } 7$$

Histogram Equalization: Example



a b c

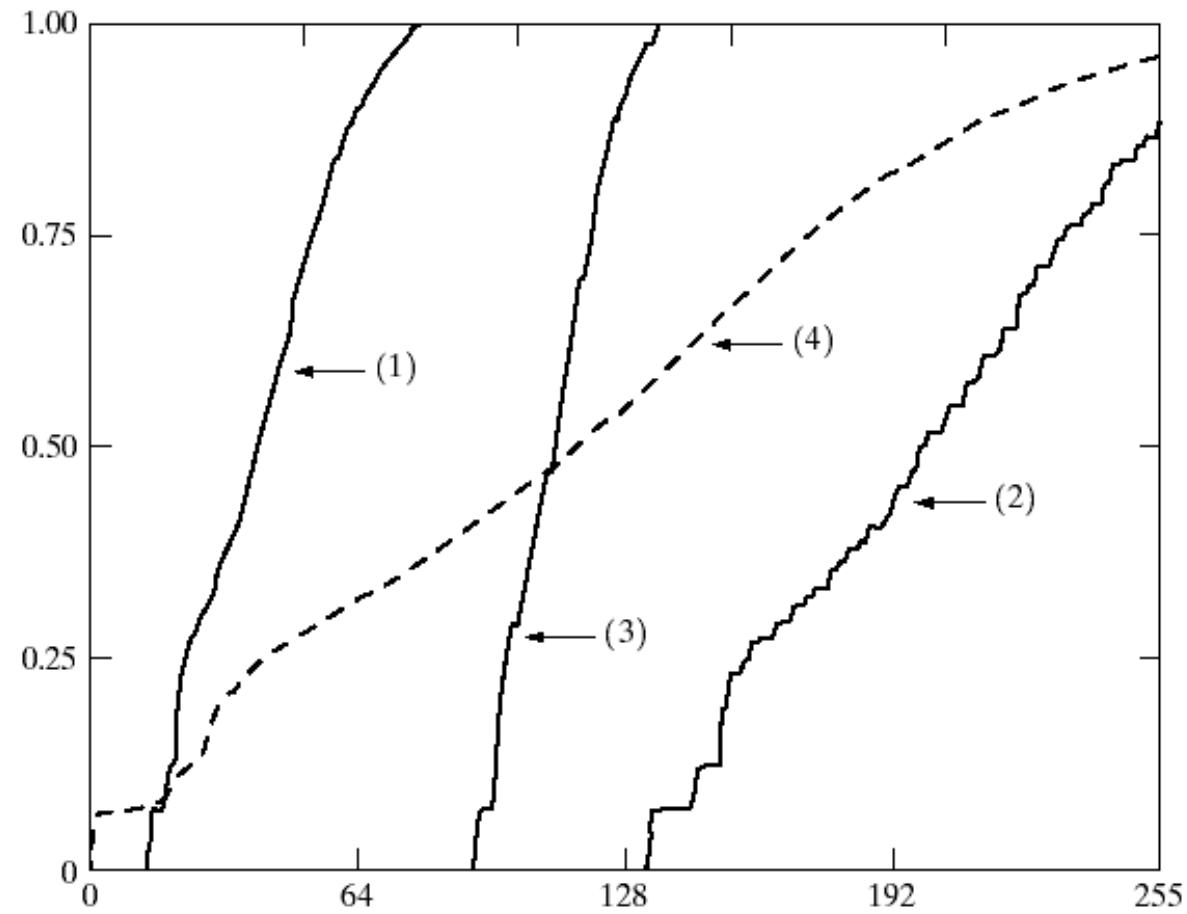
FIGURE 3.19 Illustration of histogram equalization of a 3-bit (8 intensity levels) image. (a) Original histogram. (b) Transformation function. (c) Equalized histogram.



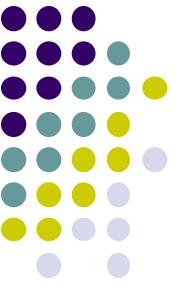


Equalization Transformation Functions

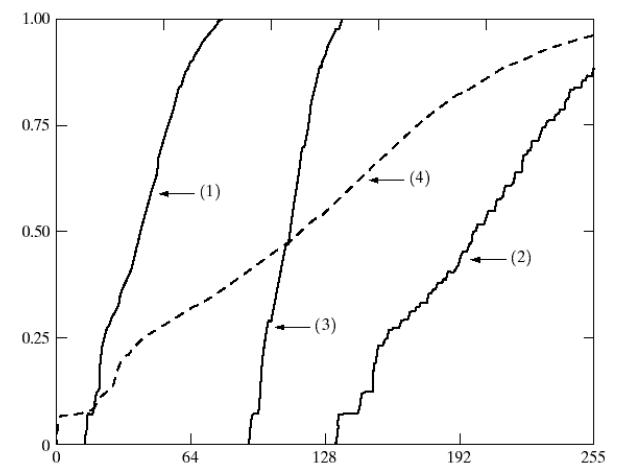
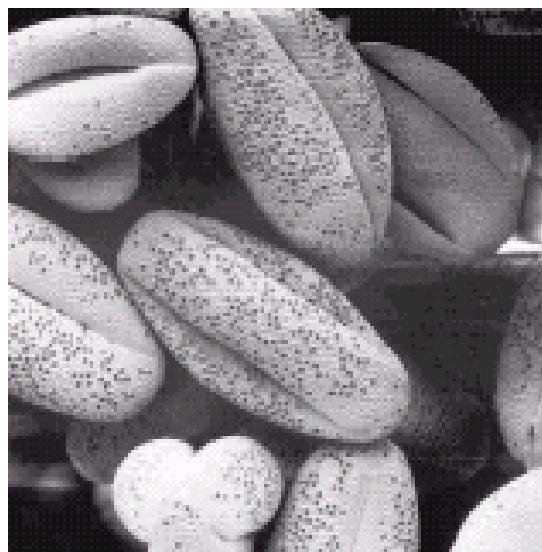
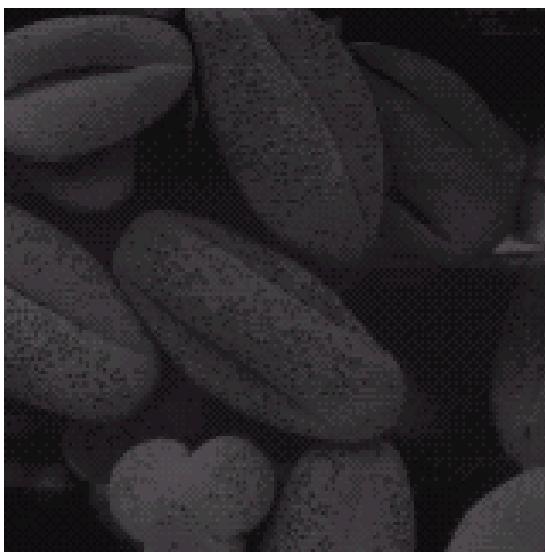
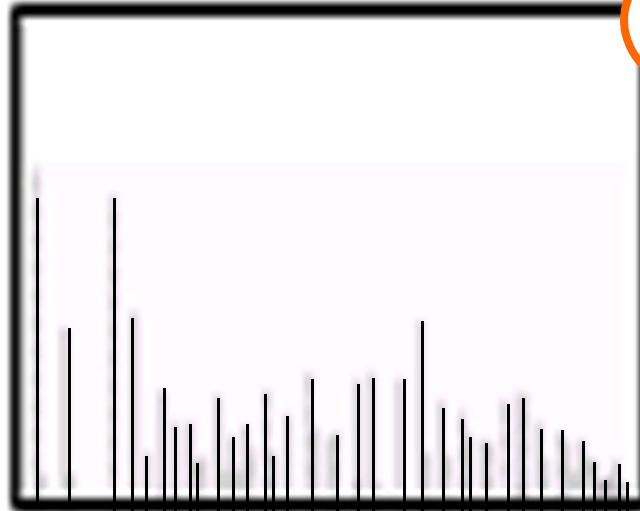
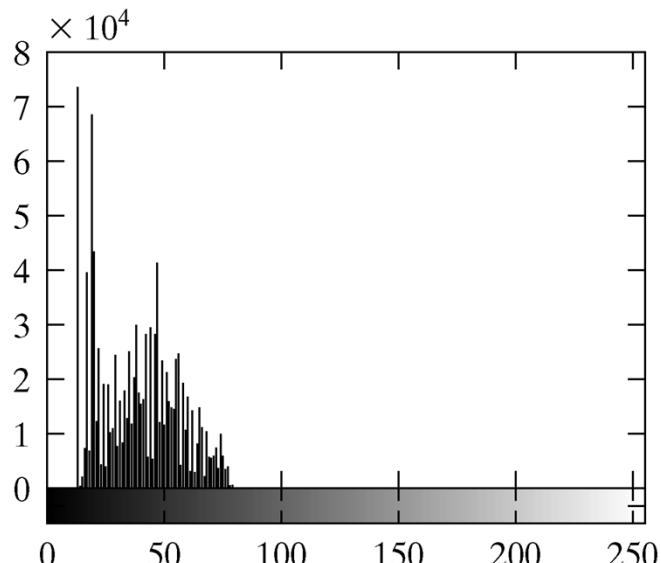
Different equalization function (1-4) may be used



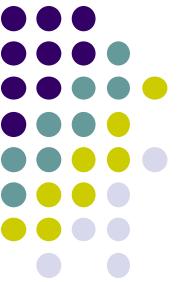
Equalization Examples



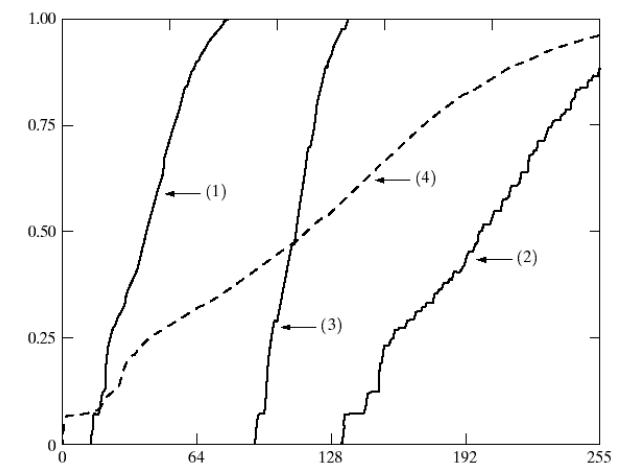
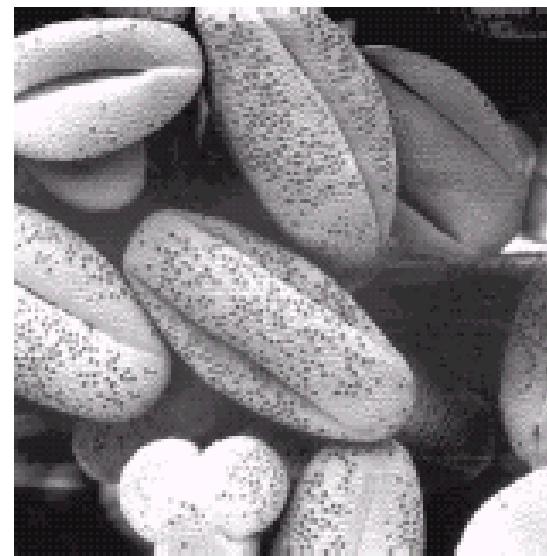
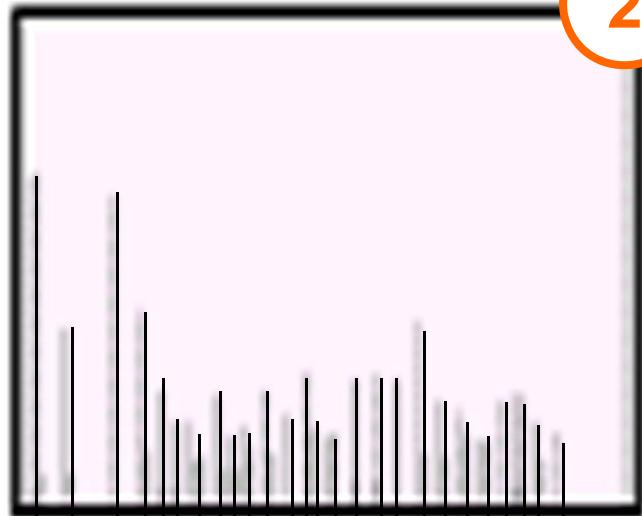
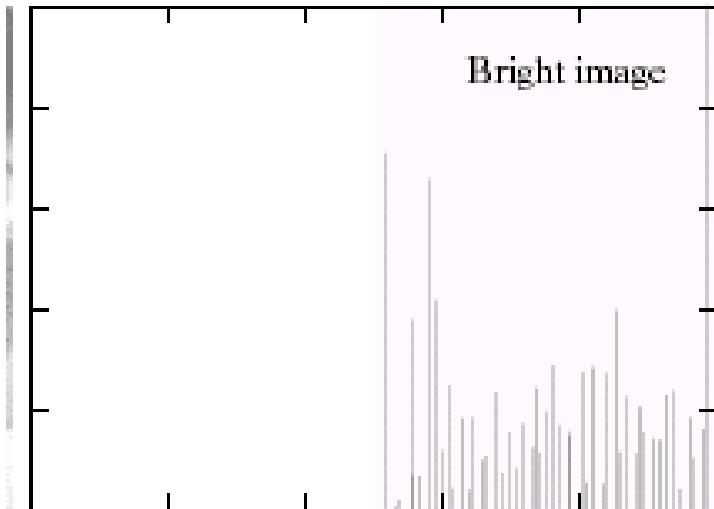
1



Equalization Examples



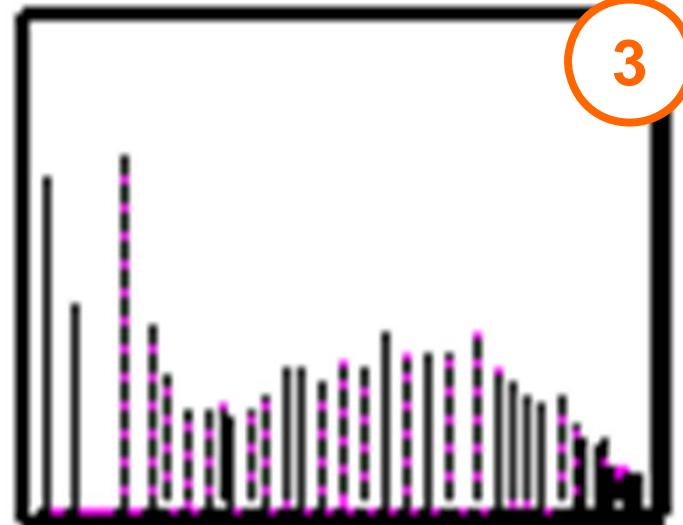
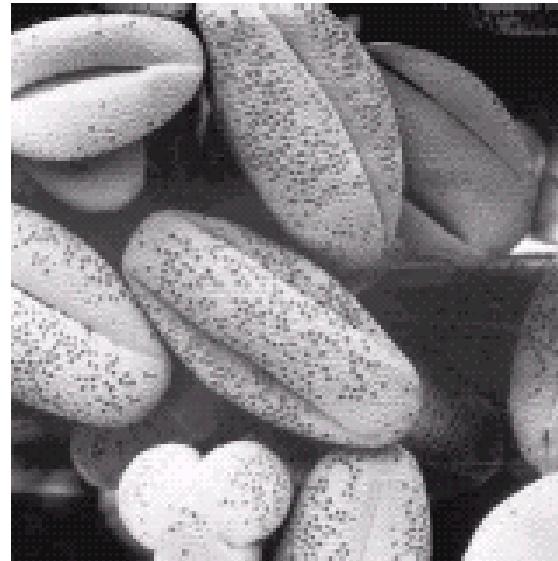
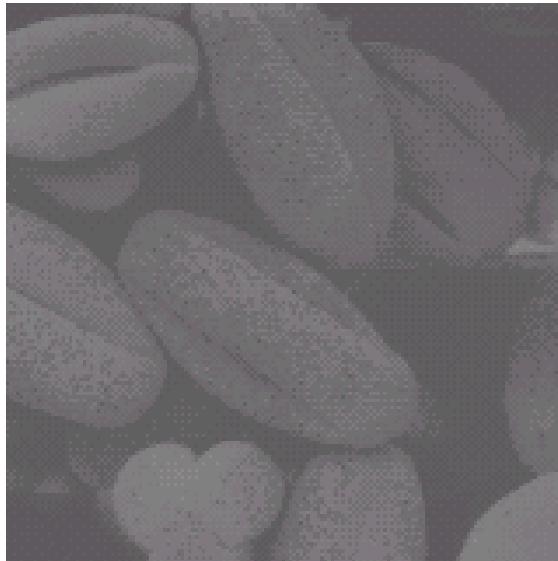
2



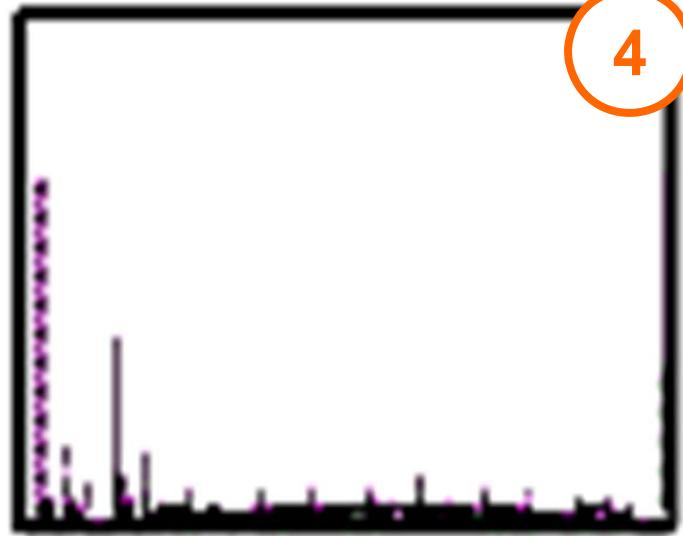
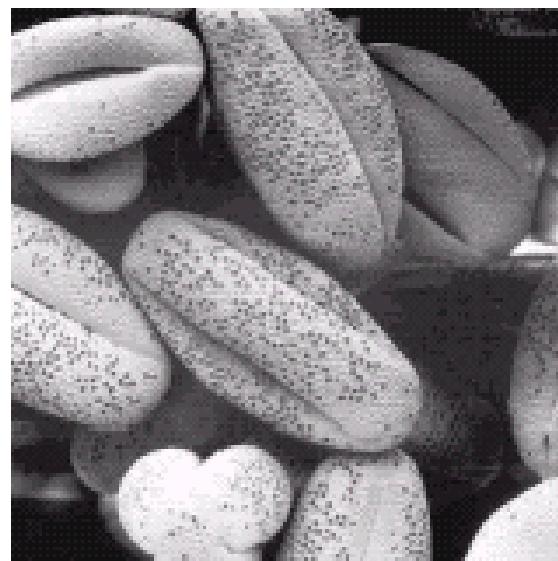
Equalization Examples



Images taken from Gonzalez & Woods, Digital Image Processing (2002)



3



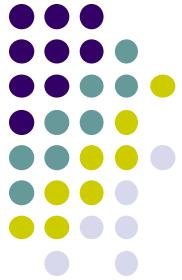
4

Histogram Equalization Algorithm

- Given an image, compute the histogram.
- Normalize the histogram to obtain pixel-wise probabilities
- Compute the transformation table.
- For each pixel, replace its value with a new value from the transformation table.

Histogram Equalization: Summary

- Histogram equalization yields an approximately uniform distribution in the output.
- The equalized image spreads a wider range of the intensity scale and enhances the contrast and clarity of details.
- The algorithm is automatic and can be easily implemented by a computer program.



References

- Wilhelm Burger and Mark J. Burge, Digital Image Processing, Springer, 2008
 - Histograms (Ch 4)
 - Point operations (Ch 5)
- University of Utah, CS 4640: Image Processing Basics, Spring 2012
- Rutgers University, CS 334, Introduction to Imaging and Multimedia, Fall 2012
- Gonzales and Woods, Digital Image Processing (3rd edition), Prentice Hall