中国科学技术大学
University of Science and Technology of China

Introduction

Juyong Zhang
School of Mathematics, USTC

The Course

- Learn how to solve math problems with tools
- Matlab, Mathematica, Python, Eigen, Ceres, etc…
- Grading policy:
  - Homework&Programming: 80%
  - Final Project: 20%

Covered Topics

- Image Processing, Image Filtering
- Face Detection, Face Recognition
- Image Stitching, Image Warping
- Tracking, Optical Flow,
- Stereo Matching, Epipolar Geometry
- Structure From Motion, 3D Surface Reconstruction
- Neural Network, etc…

TA & QQ Group

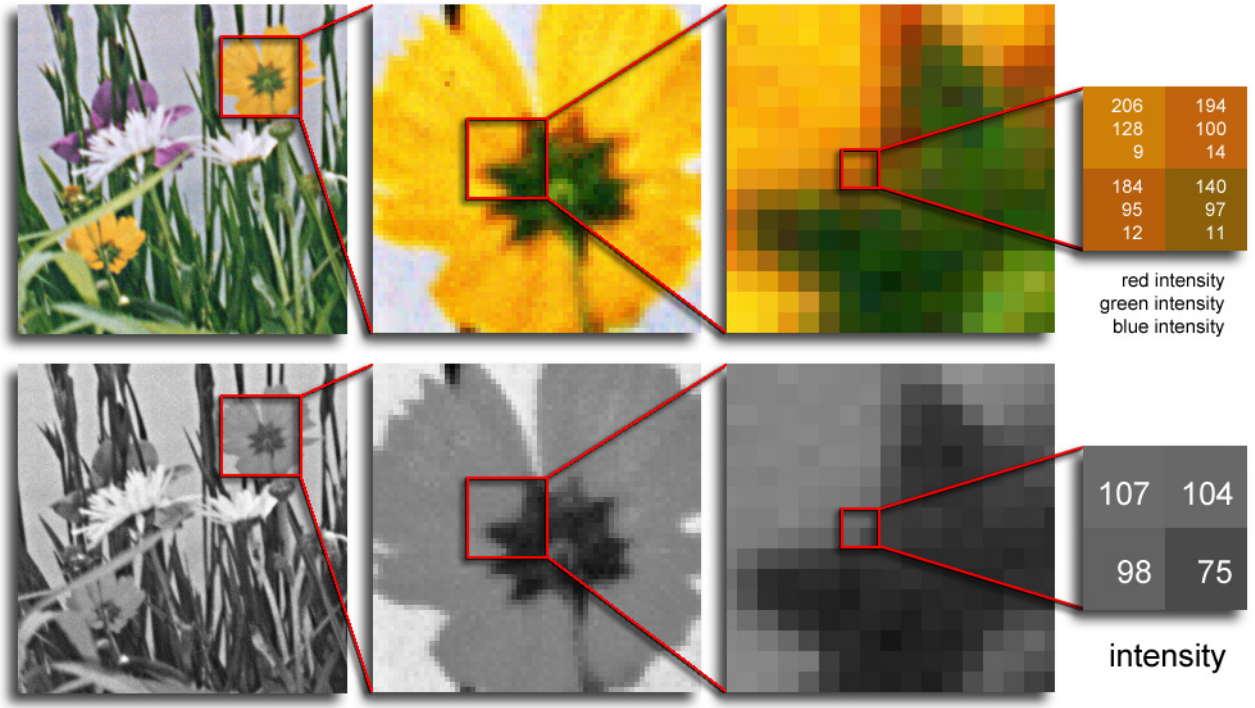
- 助教
  - 杨乐园（ ly_1207@mail.ustc.edu.cn ）
  - 许子航（ ad123456@mail.ustc.edu.cn ）
- 课程QQ群

Digital Image

Color images have 3 values per pixel; monochrome images have 1 value per pixel.

a grid of squares, each of which contains a single color

each square is called a pixel (for *picture element*)



Pixels

- A digital image, I, is a mapping from a 2D grid of uniformly spaced discrete points, $\{p = (r,c)\}$, into a set of positive integer values, $\{I(p)\}$, or a set of vector values, *e.g.*, $\{[R\ G\ B]^{T}(p)\}$.

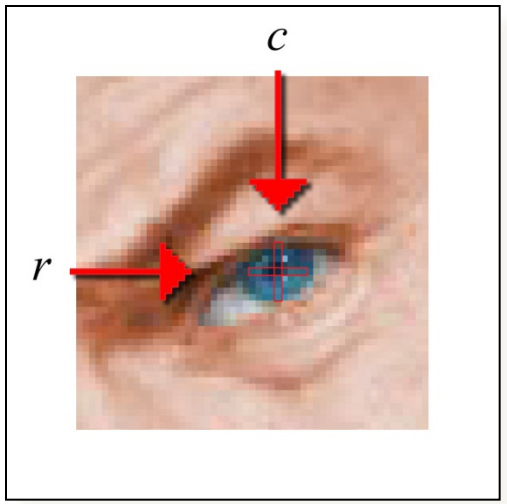
- At each column location in each row of I there is a value.

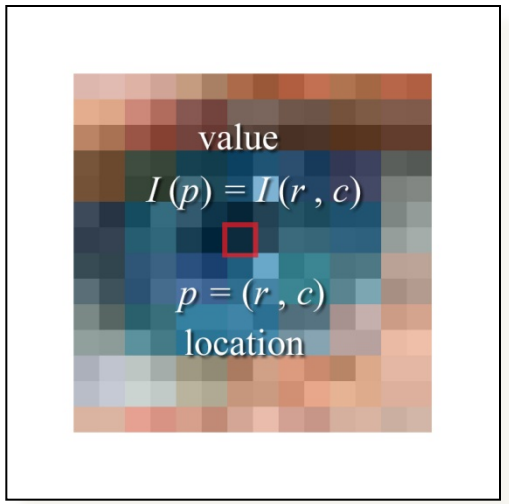
- The pair (p, $I(p)$) is called a "pixel" (for *picture element*).

Pixels

- p = (r,c) is the pixel location indexed by row, r, and column, c.
- I(p) = I(r,c) is the value of the pixel at location p.
- If I(p) is a single number then I is monochrome.
- If I(p) is a vector (ordered list of numbers) then I has multiple bands (e.g., a color image).

Pixels





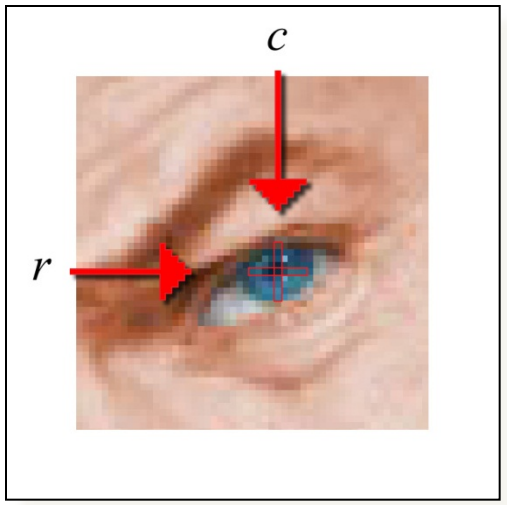
Pixel Location: $p = (r, c)$

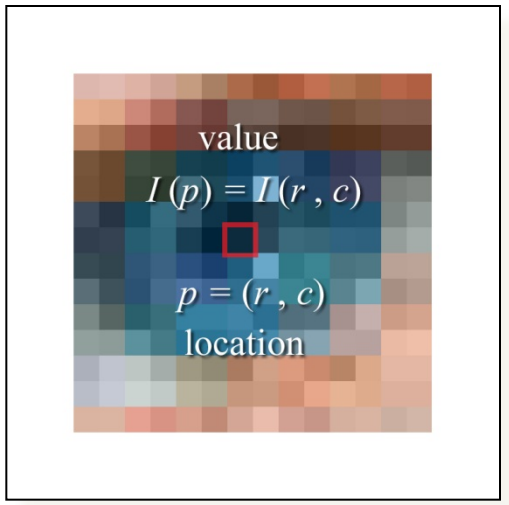
Pixel Value: $I(p) = I(r, c)$

Pixel : $[p, I(p)]$

Pixels

Pixel : $[p, I(p)]$

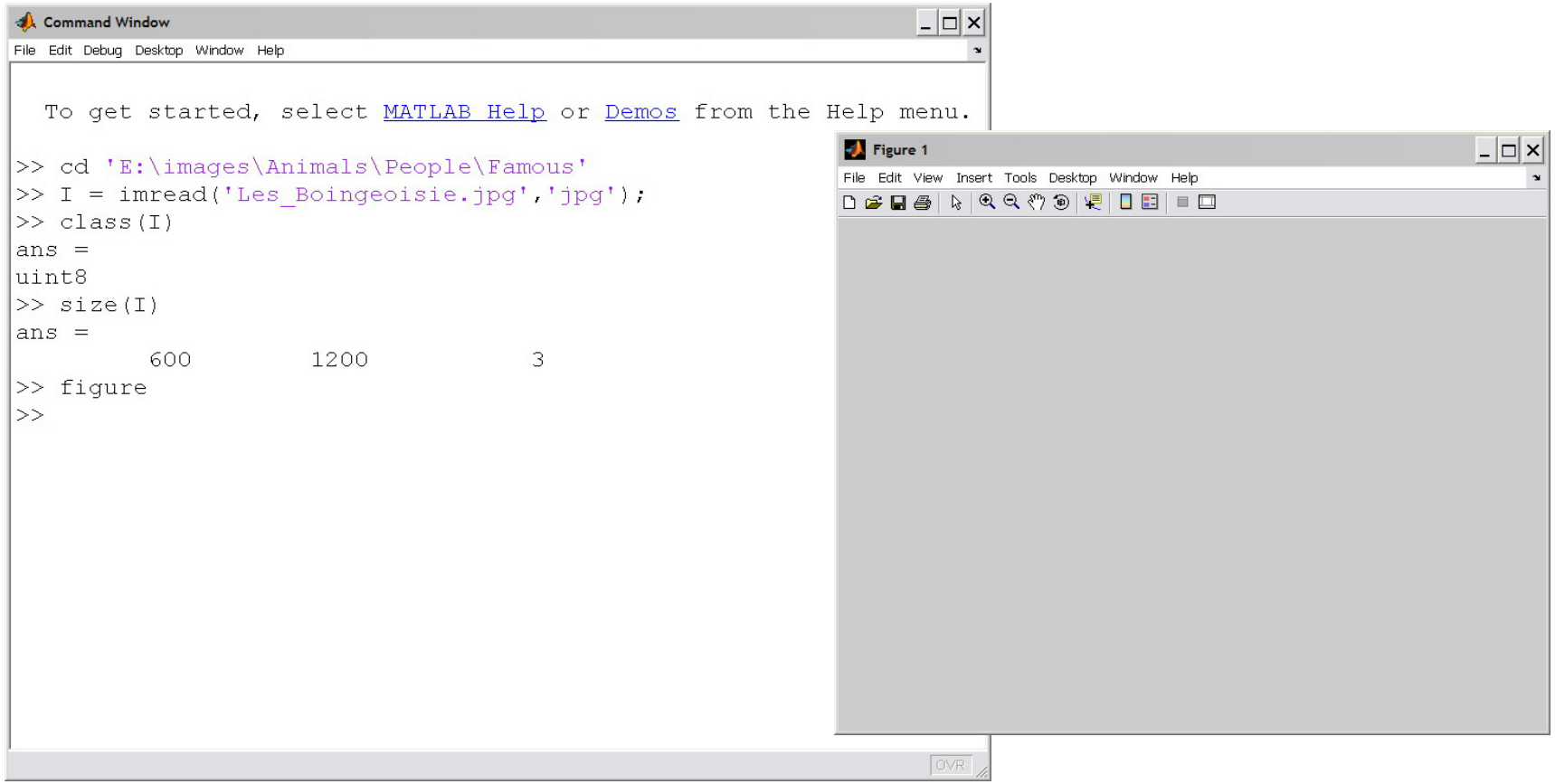




$$
\begin{aligned}
p &= (r, c) \\
&= (\text{row } \#,\ \text{col } \#) \\
&= (272,\ 277)
\end{aligned}
$$

$$
I(p) = \begin{bmatrix} \text{red} \\ \text{green} \\ \text{blue} \end{bmatrix} = \begin{bmatrix} 12 \\ 43 \\ 61 \end{bmatrix}
$$

Read an Image into Matlab



Read an Image into Matlab

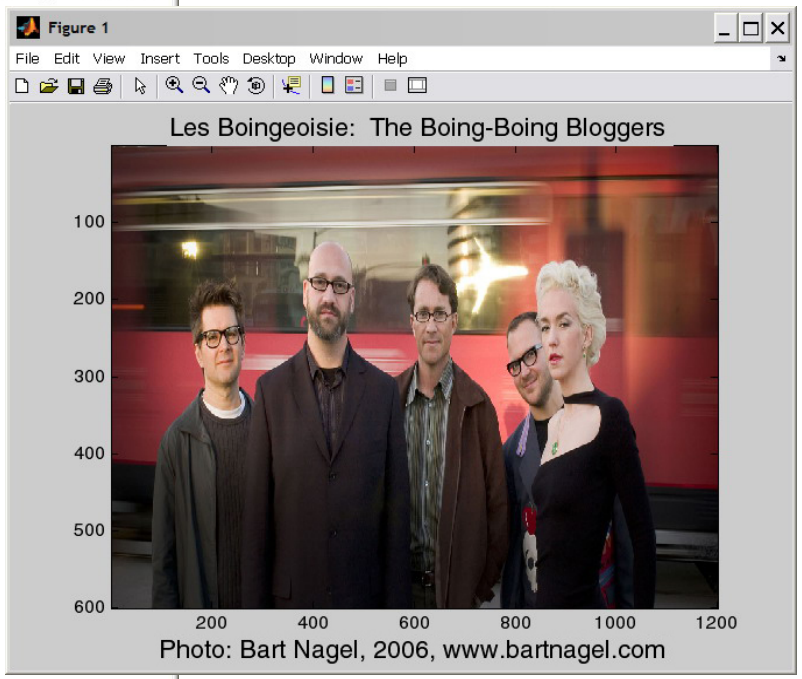
Command Window

File Edit Debug Desktop Window Help

To get started, select MATLAB Help or Demos from the Help menu.

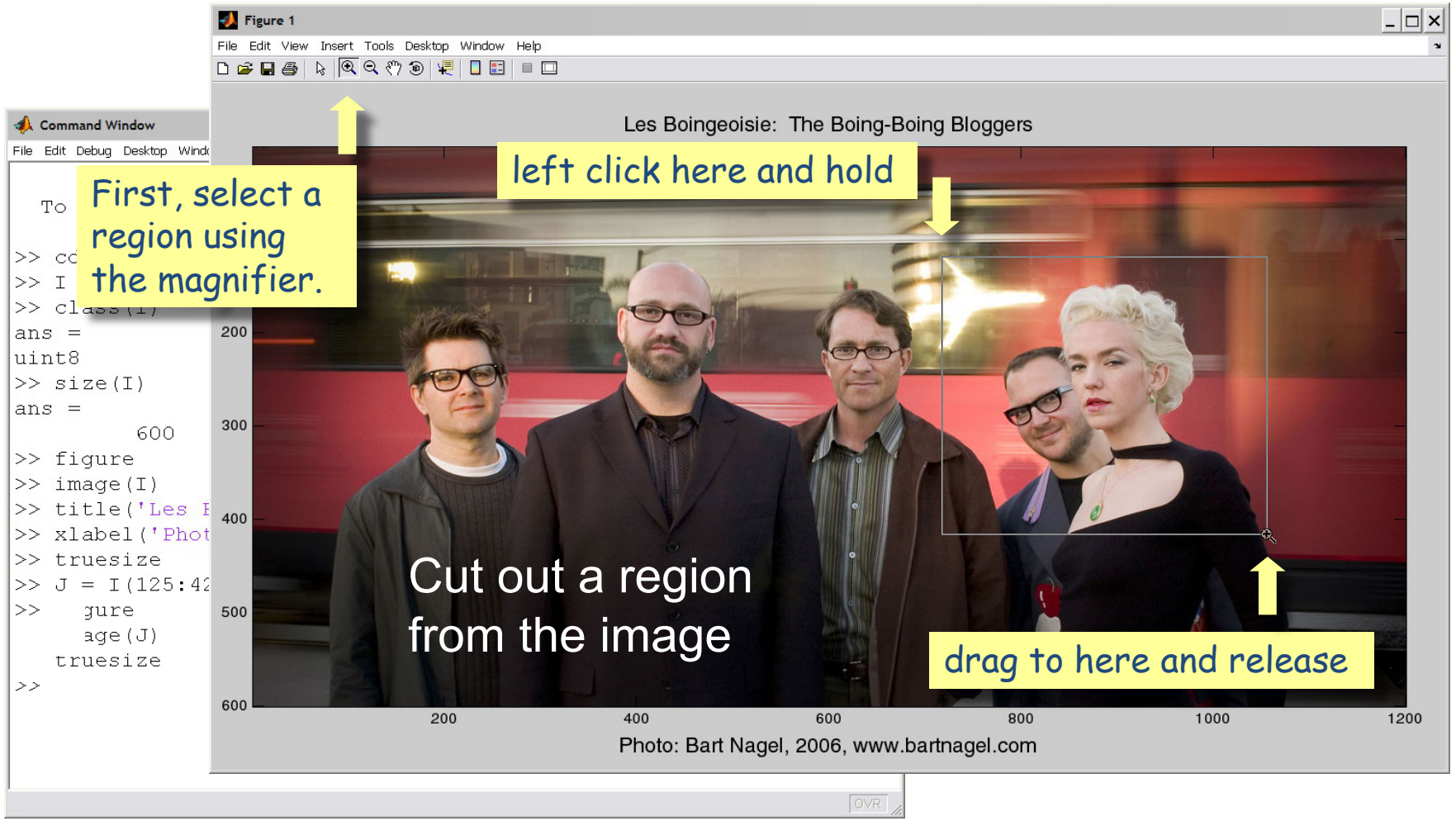
```
>> cd 'E:\images\Animals\People\Famous'
>> I = imread('Les_Boingeoisie.jpg','jpg');
>> class(I)
ans =
uint8
>> size(I)
ans =
         600        1200           3
>> figure
>> image(I)
>> title('Les Boingeoisie:  The Boing-Boing Bloggers')
>> xlabel('Photo: Bart Nagel, 2006, www.bartnagel.com')
>>
```

OVR

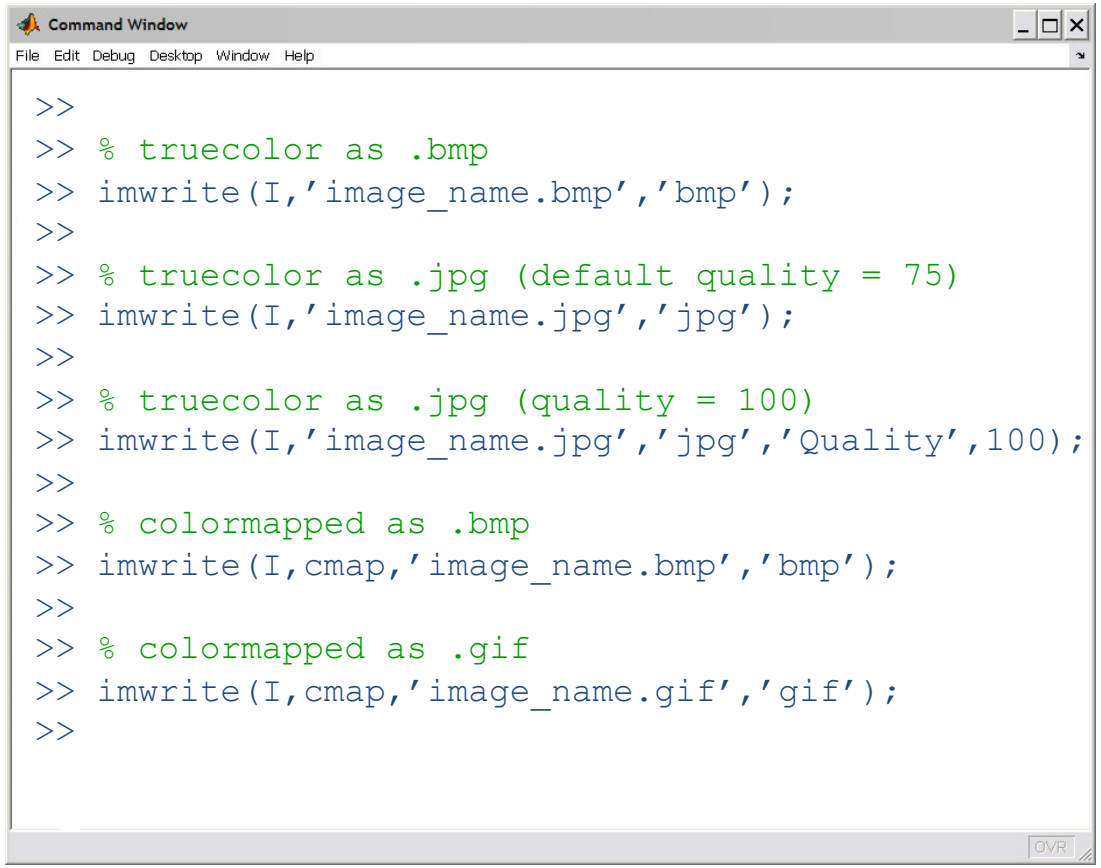


Read an Image into Matlab

Crop the Image



Saving Images as Files



Assuming that
'I' contains the image of the correct class,
that
'cmap' is a colormap,
and that
'image_name' is the file-name that you want.

Double Exposure: Adding Two Images

Jim Woodring - Bumperillo

Mark Rayden – The Ecstasy of Cecelia

Rayden Woodring – The Ecstasy of Bumperillo (?)

Double Exposure: Adding Two Images

```
>> JW = imread('Jim Woodring - Bumperillo.jpg','jpg');
>> figure
>> image(JW)
>> truesize
>> title('Bumperillo')
>> xlabel('Jim Woodring')
>> MR = imread('Mark Ryden - The Ecstasy of Cecelia.jpg','jpg');
>> figure
>> image(MR)
>> truesize
>> title('The Ecstasy of Cecelia')
>> xlabel('Mark Ryden')
>> [RMR,CMR,DMR] = size(MR);
>> [RJW,CJW,DJW] = size(JW);
>> rb = round((RJW-RMR)/2);
>> cb = round((CJW-CMR)/2);
>> JWplusMR = uint8((double(JW(rb:(rb+RMR-1),cb:(cb+CMR-1),:))+double(MR))/2);
>> figure
>> image(JWplusMR)
>> truesize
>> title('The Ecstasy of Bumperillo')
>> xlabel('Jim Woodring + Mark Ryden')
```

Example
Matlab Code

Double Exposure: Adding Two Images

```
>> JW = imread('Jim Woodring - Bumperillo.jpg','jpg');
>> figure
>> image(JW)
>> truesize
>> title('Bumperillo')
>> xlabel('Jim Woodring')
>> MR = imread('Mark Ryden - The Ecstasy of Cecelia.jpg','jpg');
>> figure
>> image(MR)
>> truesize
>> title('The Ecstasy of Cecelia')
>> xlabel('Mark Ryden')
>> [RMR,CMR,DMR] = size(MR);
>> [RJW,CJW,DJW] = size(JW);
>> rb = round((RJW-RMR)/2);
>> cb = round((CJW-CMR)/2);
>> JWplusMR = uint8((double(JW(rb:(rb+RMR-1),cb:(cb+CMR-1),:))+double(MR))/2);
>> figure
>> image(JWplusMR)
>> truesize
>> title('The Ecstasy of Bumperillo')
>> xlabel('Jim Woodring + Mark Ryden')
```

Example Matlab Code

Cut a section out of the middle of the larger image the same size as the smaller image.

Double Exposure: Adding Two Images

```
>> JW = imread('Jim Woodring - Bumperillo.jpg','jpg');
>> figure
>> image(JW)
>> truesize
>> title('Bumperillo')
>> xlabel('Jim Woodring')
>> MR = imread('Mark Ryden - The Ecstasy of Cecelia.jpg','jpg');
>> figure
>> image(MR)
>> truesize
>> title('The Ecstasy of Cecelia')
>> xlabel('Mark Ryden')
>> [RMR,CMR,DMR] = size(MR);
>> [RJW,CJW,DJW] = size(JW);
>> rb = round((RJW-RMR)/2);
>> cb = round((CJW-CMR)/2);
>> JWplusMR = uint8((double(JW(rb:(rb+RMR-1),cb:(cb+CMR-1),:))+double(MR))/2);
>> figure
>> image(JWplusMR)
>> truesize
>> title('The Ecstasy of Bumperillo')
>> xlabel('Jim Woodring + Mark Ryden')
```

Example Matlab Code

Note that the images are averaged, pixelwise.

Double Exposure: Adding Two Images

```
>> JW = imread('Jim Woodring - Bumperillo.jpg','jpg');
>> figure
>> image(JW)
>> truesize
>> title('Bumperillo')
>> xlabel('Jim Woodring')
>> MR = imread('Mark Ryden - The Ecstasy of Cecelia.jpg','jpg');
>> figure
>> image(MR)
>> truesize
>> title('The Ecstasy of Cecelia')
>> xlabel('Mark Ryden')
>> [RMR,CMR,DMR] = size(MR);
>> [RJW,CJW,DJW] = size(JW);
>> rb = round((RJW-RMR)/2);
>> cb = round((CJW-CMR)/2);
>> JWplusMR = uint8((double(JW(rb:(rb+RMR-1),cb:(cb+CMR-1),:))+double(MR))/2);
>> figure
>> image(JWplusMR)
>> truesize
>> title('The Ecstasy of Bumperillo')
>> xlabel('Jim Woodring + Mark Ryden')
```

Example Matlab Code

Note the data class conversions.

Intensity Masking: Multiplying Two Images

Jim Woodring - Bumperillo

Mark Rayden – The Ecstasy of Cecelia

Rayden Woodring – Bumperillo Ecstasy (?)

Intensity Masking: Multiplying Two Images

Example Matlab Code

```
>> JW = imread('Jim Woodring - Bumperillo.jpg','jpg');
>> MR = imread('Mark Ryden - The Ecstasy of Cecelia.jpg','jpg');
>> [RMR,CMR,DMR] = size(MR);
>> [RJW,CJW,DJW] = size(JW);
>> rb = round((RJW-RMR)/2);
>> cb = round((CJW-CMR)/2);
>> JWplusMR = uint8((double(JW(rb:(rb+RMR-1),cb:(cb+CMR-1),:))+double(MR))/2);
>> figure
>> image(JWplusMR)
>> truesize
>> title('The Extacsy of Bumperillo')
>> xlabel('Jim Woodring + Mark Ryden')
>> JWtimesMR = double(JW(rb:(rb+RMR-1),cb:(cb+CMR-1),:)).*double(MR);
>> M = max(JWtimesMR(:));
>> m = min(JWtimesMR(:));
>> JWtimesMR = uint8(255*(double(JWtimesMR)-m)/(M-m));
>> figure
>> image(JWtimesMR)
>> truesize
>> title('EcstasyBumperillo')
```

Intensity Masking: Multiplying Two Images

```
>> JW = imread('Jim Woodring - Bumperillo.jpg','jpg');
>> MR = imread('Mark Ryden - The Ecstasy of Cecelia.jpg','jpg');
>> [RMR,CMR,DMR] = size(MR);
>> [RJW,CJW,DJW] = size(JW);
>> rb = round((RJW-RMR)/2);
>> cb = round((CJW-CMR)/2);
>> JWplusMR = uint8((double(JW(rb:(rb+RMR-1),cb:(cb+CMR-1),:))+double(MR))/2);
>> figure
>> image(JWplusMR)
>> truesize
>> title('The Extacsy of Bumperillo')
>> xlabel('Jim Woodring + Mark Ryden')
>> JWtimesMR = double(JW(rb:(rb+RMR-1),cb:(cb+CMR-1),:)).*double(MR);
>> M = max(JWtimesMR(:));
>> m = min(JWtimesMR(:));
>> JWtimesMR = uint8(255*(double(JWtimesMR)-m)/(M-m));
>> figure
>> image(JWtimesMR)
>> truesize
>> title('EcstasyBumperillo')
```

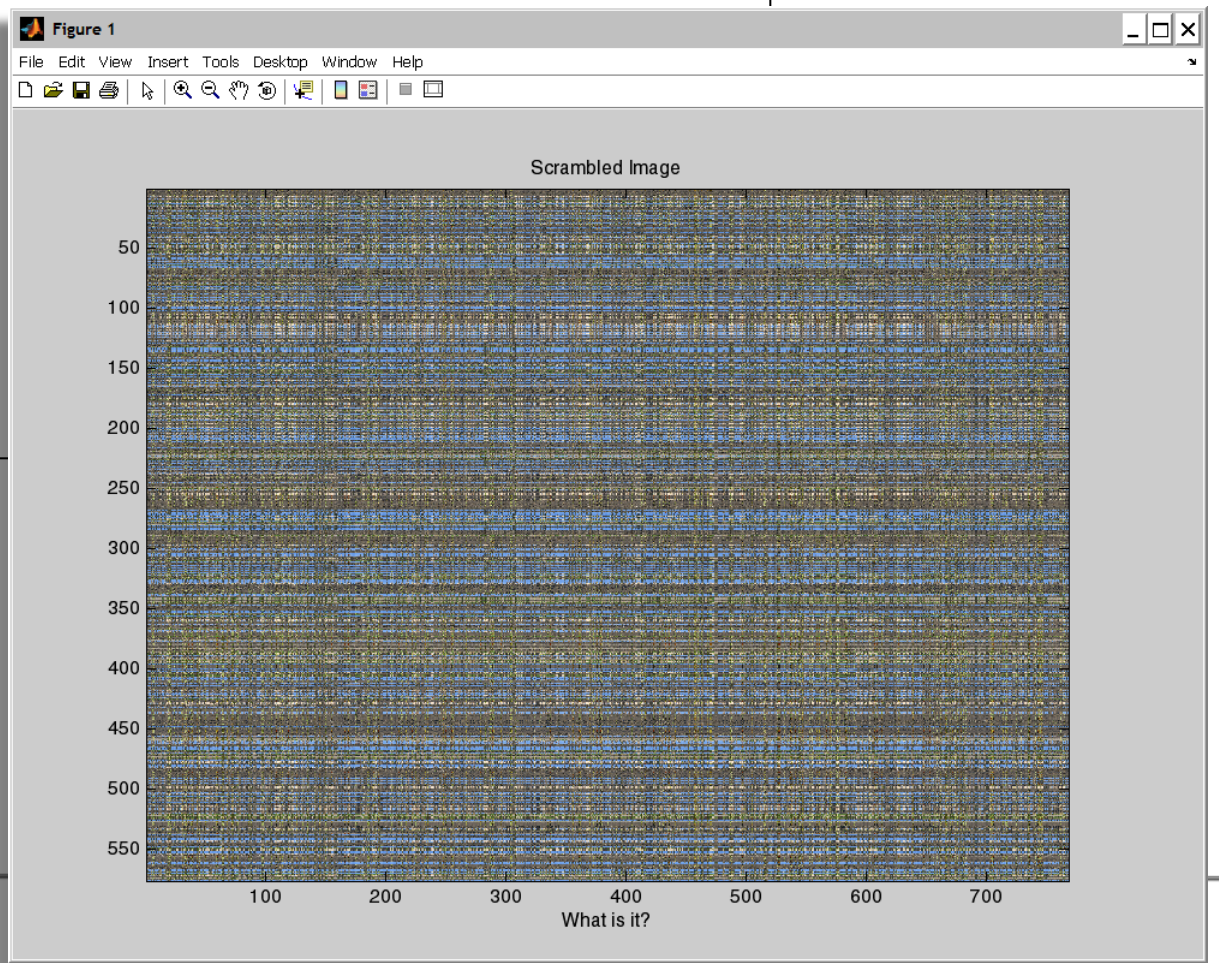
Example Matlab Code

Note that the images are multiplied, pixelwise.

Note how the image intensities are scaled back into the range 0-255.

Pixel Indexing in Matlab

```
>> I = imread('Lawraa - Flickr - 278635073_883bd891ec_o.jpg','jpg');
>> size(I)
ans =
   576   768     3
>> r = randperm(576);
>> c = randperm(768);
>> J = I(r,c,:);
>> figure
>> image(J)
>> truesize
>> title('Scrambled Image')
>> xlabel('What is it?')
```



Point Processing of Images

- In a digital image, point = pixel.
- Point processing transforms a pixel's value as function of its value alone;
- it does not depend on the values of the pixel's neighbors.

Point Processing of Images

- Brightness and contrast adjustment
- Gamma correction
- Histogram equalization
- Histogram matching
- Color correction.

Point Processing

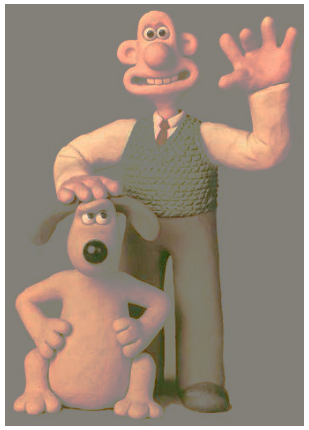
- gamma

- brightness

original

+ brightness

+ gamma



histogram mod

- contrast

original

+ contrast

histogram EQ

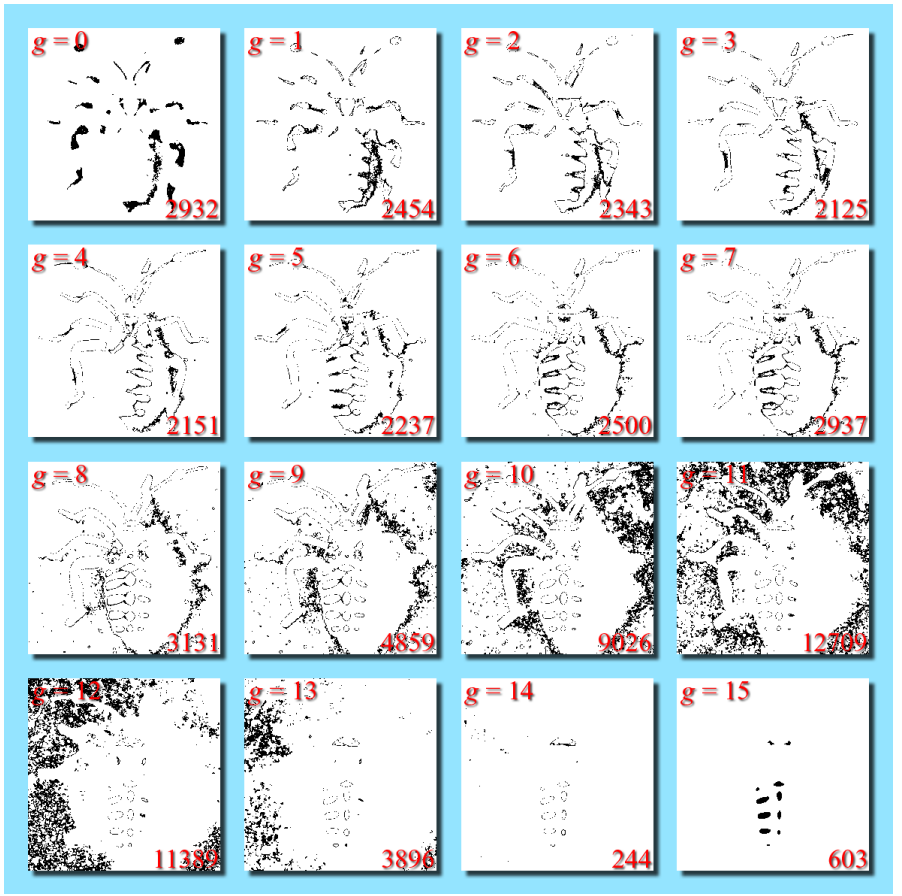
The Histogram of a Grayscale Image

- Let **I** be a 1-band (grayscale) image.
- **I**(r,c) is an 8-bit integer between 0 and 255.
- Histogram, $h_{\mathbf{I}}$, of **I**:
  - a 256-element array, $h_{\mathbf{I}}$
  - $h_{\mathbf{I}}(g)$, for $g = 1, 2, 3, \ldots, 256$, is an integer
  - $h_{\mathbf{I}}(g)$ = number of pixels in **I** that have value g-1.

The Histogram of a Grayscale Image

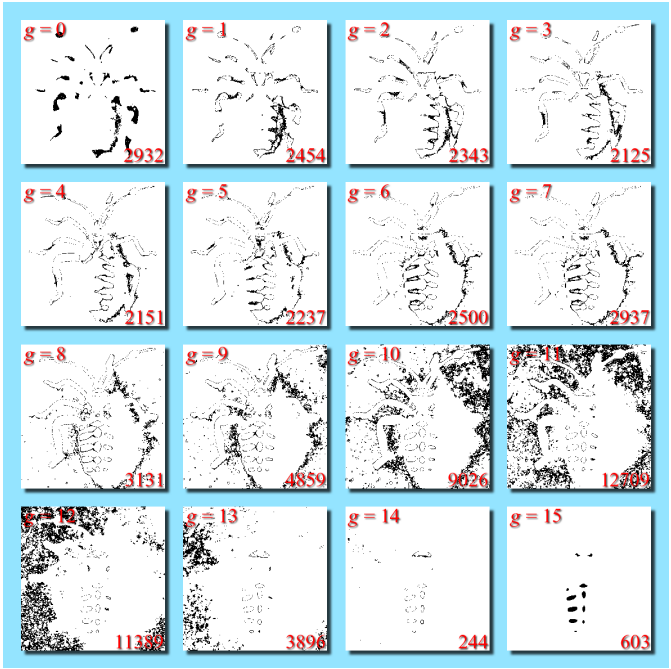
16-level (4-bit) image

lower RHC: number of pixels with intensity g



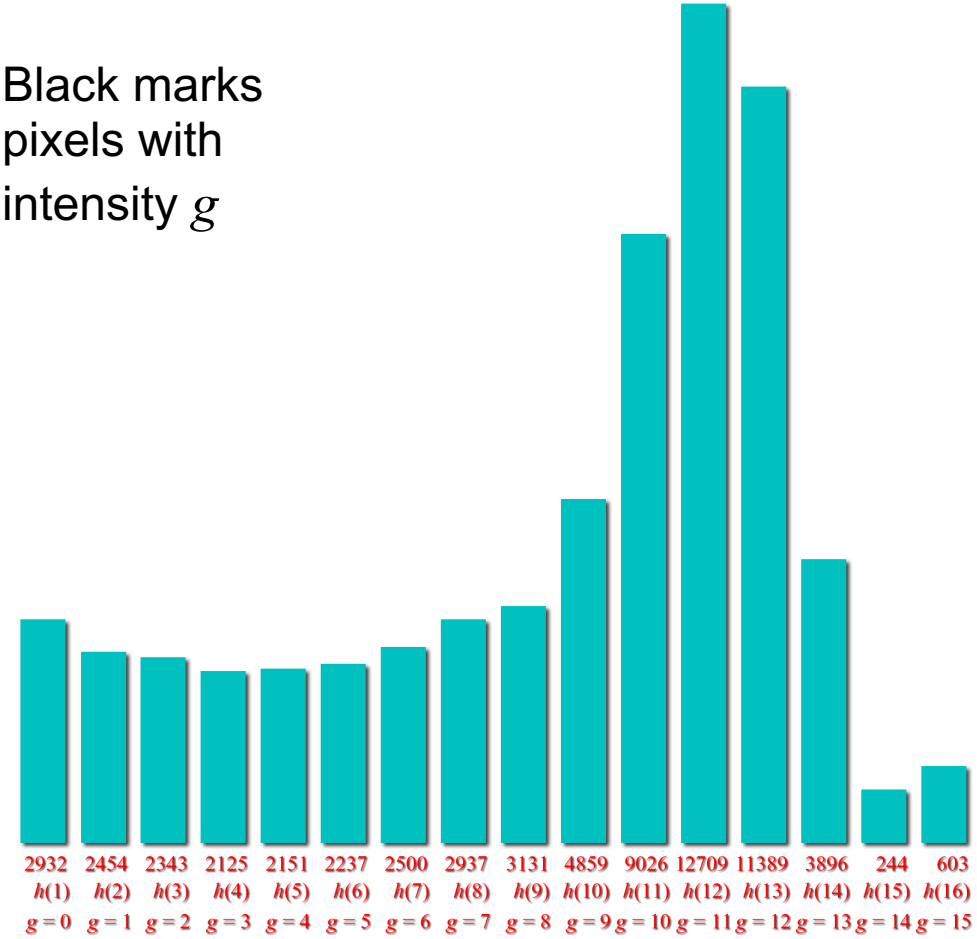
black marks pixels with intensity g

The Histogram of a Grayscale Image

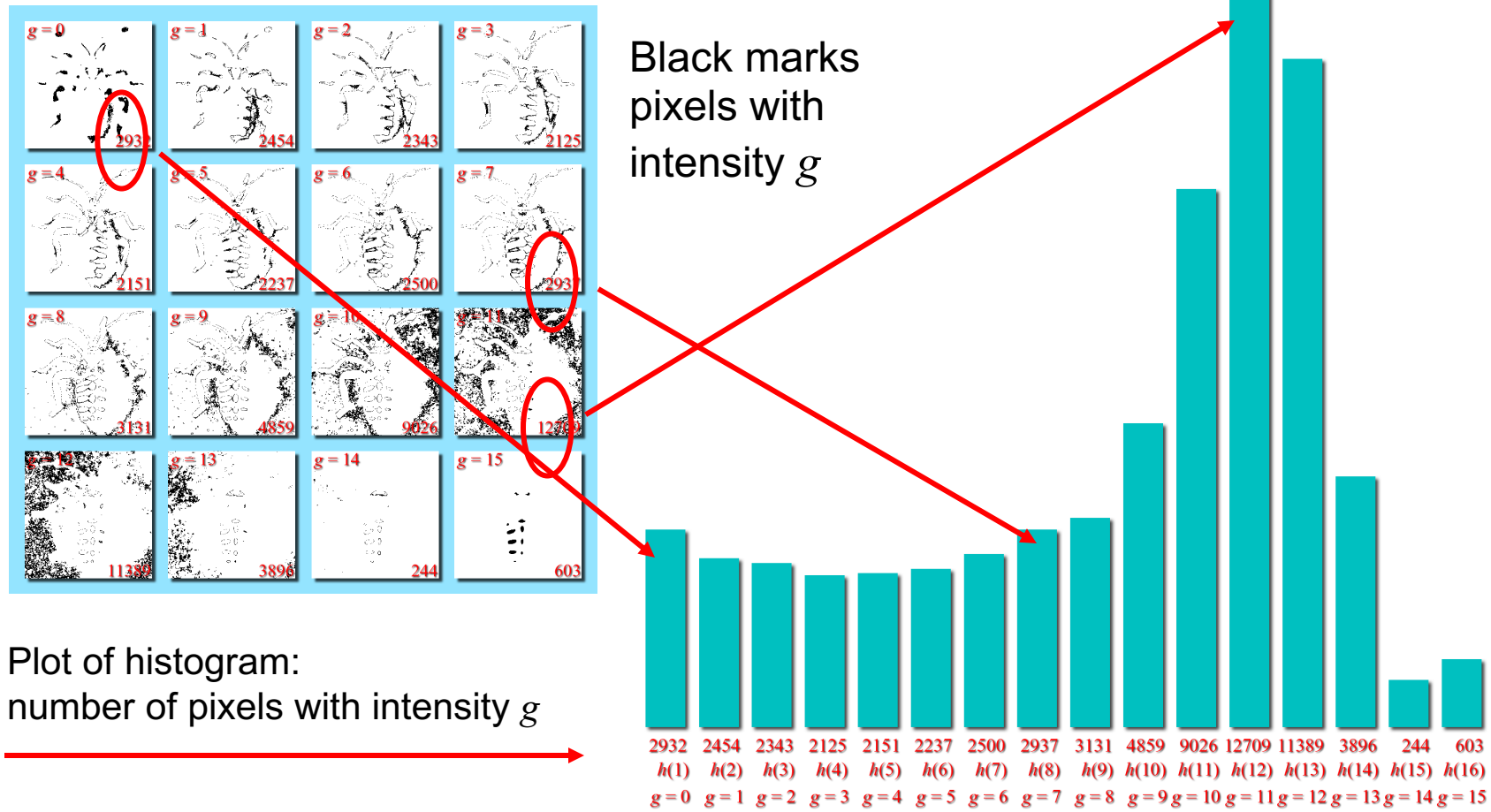


Plot of histogram:
number of pixels with intensity g

Black marks pixels with intensity g

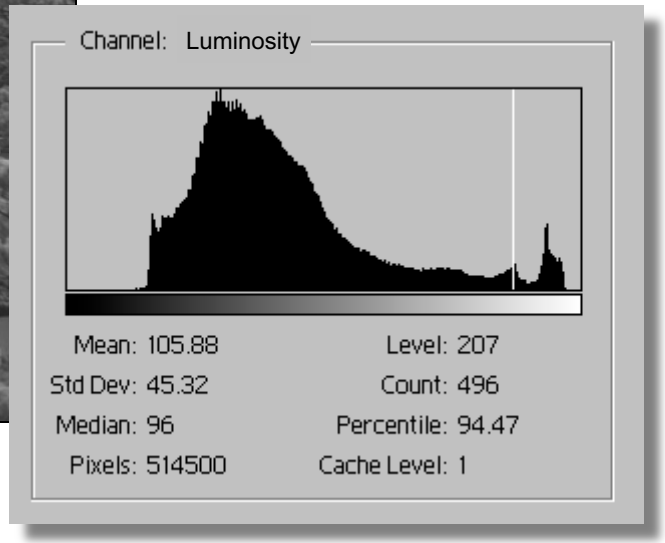


The Histogram of a Grayscale Image



The Histogram of a Grayscale Image

$h_{\mathbf{I}}(g+1) =$ the number of pixels in $\mathbf{I}$ with graylevel g.

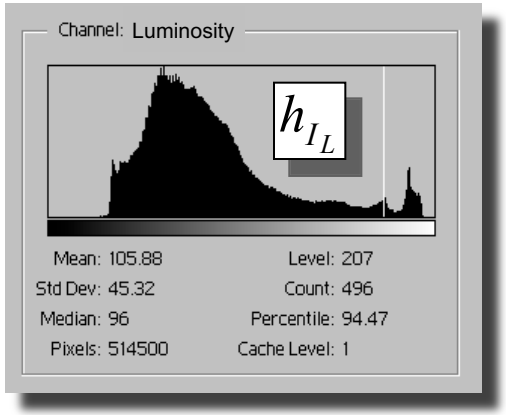


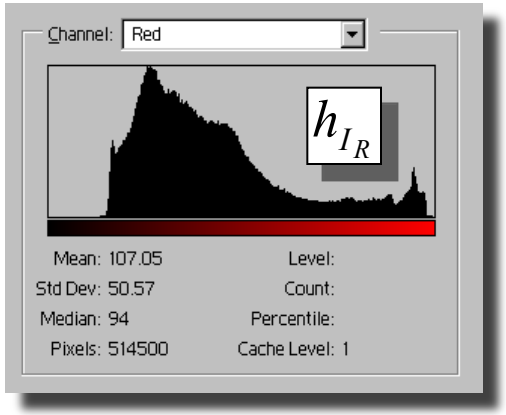
The Histogram of a Color Image

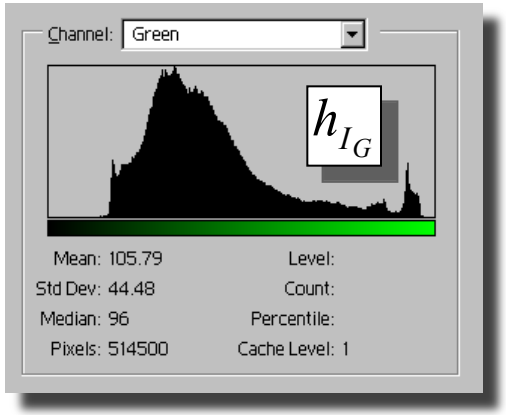
- If **I** is a 3-band image (truecolor, 24-bit)
- then **I**(r,c,b) is an integer between 0 and 255.
- Either **I** has 3 histograms:
  - $h_{\mathbf{R}}(g+1)$ = # of pixels in **I**(:,:,1) with intensity value g
  - $h_{\mathbf{G}}(g+1)$ = # of pixels in **I**(:,:,2) with intensity value g
  - $h_{\mathbf{B}}(g+1)$ = # of pixels in **I**(:,:,3) with intensity value g
- or 1 vector-valued histogram, $h(g,1,b)$ where
  - $h(g+1,1,1)$ = # of pixels in **I** with red intensity value g
  - $h(g+1,1,2)$ = # of pixels in **I** with green intensity value g
  - $h(g+1,1,3)$ = # of pixels in **I** with blue intensity value g

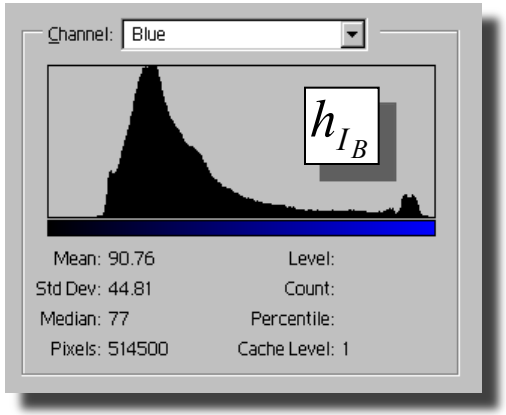
The Histogram of a Color Image

There is one histo-gram per color band R, G, & B. Value histogram is from 1 band = (R+G+B)/3









Value or Luminance Histograms

The Value histogram of a 3-band (truecolor) image, **I**, is the histogram of the value image,

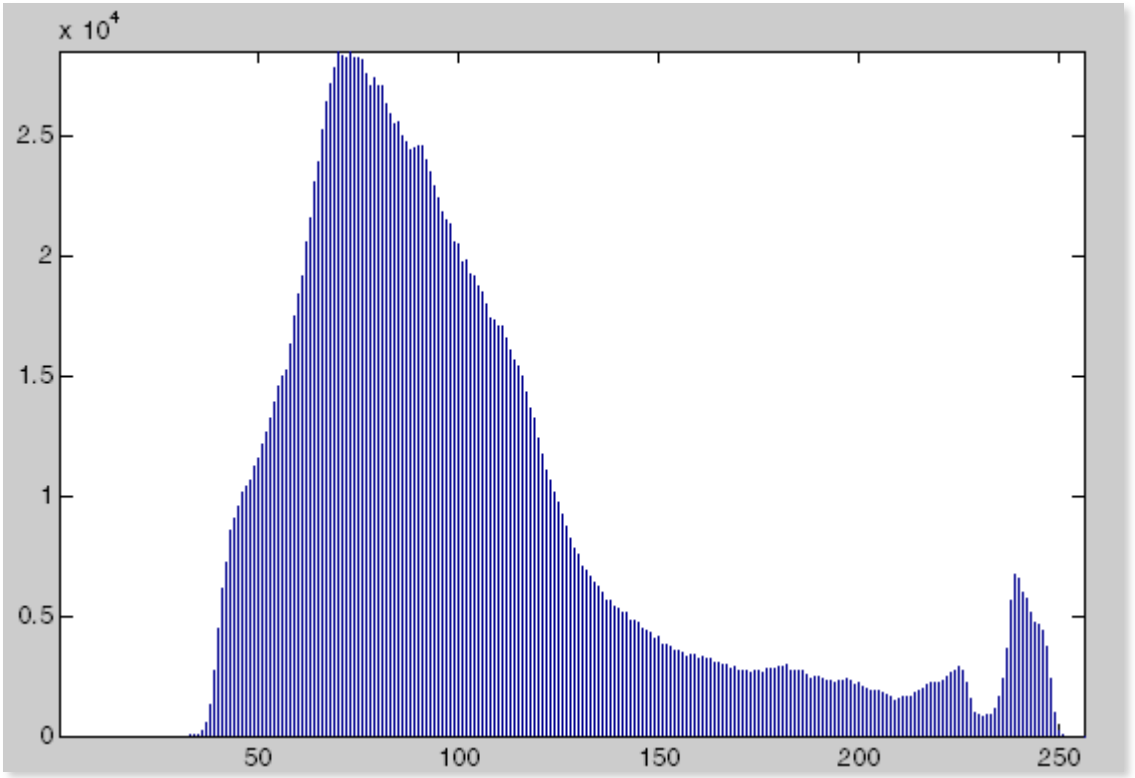
$$\mathbf{V}(r,c) = \frac{1}{3}[\mathbf{R}(r,c) + \mathbf{G}(r,c) + \mathbf{B}(r,c)]$$

Where **R**, **G**, and **B** are the red, green, and blue bands of **I**.
The luminance histogram of **I** is the histogram of the luminance image,

$$\mathbf{L}(r,c) = 0.299 \cdot \mathbf{R}(r,c) + 0.587 \cdot \mathbf{G}(r,c) + 0.114 \cdot \mathbf{B}(r,c)$$

Value Histogram

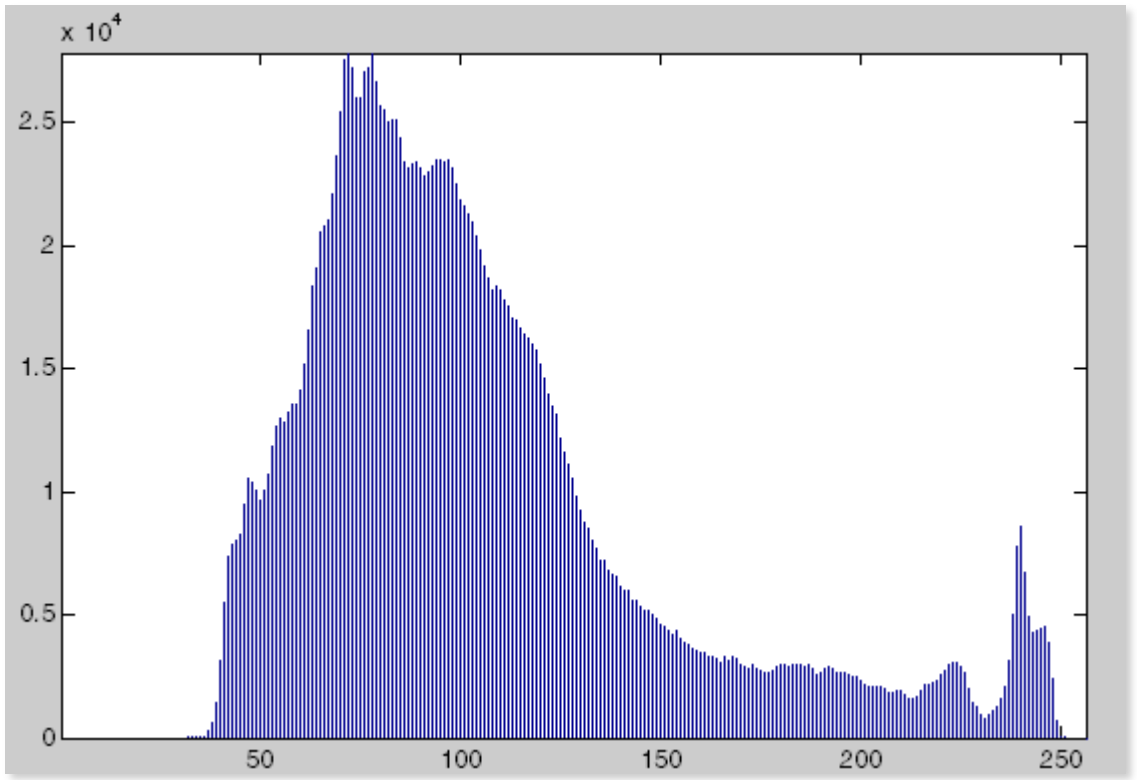
Value image, **V**.



Histogram of the value image.

Luminance Histogram

Luminance image, **L**.



Histogram of the luminance image.

Multi-Band Histogram Calculator in Matlab

```
% Multi-band histogram calculator
function h=histogram(I)

[R C B]=size(I);

% allocate the histogram
h=zeros(256,1,B);

% range through the intensity values
for g=0:255
   h(g+1,1,:) = sum(sum(I==g)); % accumulate
end

return;
```

Multi-Band Histogram Calculator in Matlab

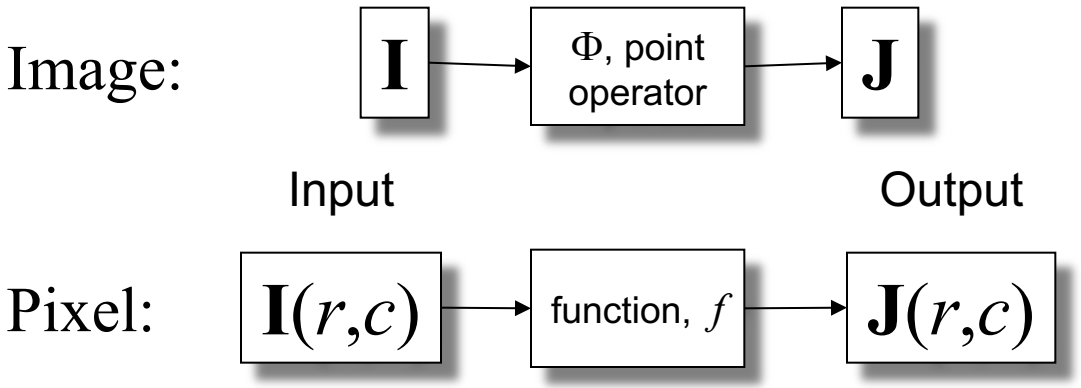
```
% Multi-band histogram calculator
function h=histogram(I)

[R C B]=size(I);

% allocate the histogram
h=zeros(256,1,B);

% range through the intensity
for g=0:255
   h(g+1,1,:) = sum(sum(I==g)); % accumulate
end
```

Loop through all intensity levels (0-255)
Tag the elements that have value g.
The result is an RxCxB *logical* array that has a 1 wherever $\mathbf{I}(r,c,b) = g$ and 0's everywhere else.
Compute the number of ones in each band of the image for intensity g.
Store that value in the 256x1xB histogram at $h(g+1,1,b)$.

If `B==3`, then `h(g+1,1,:)` contains 3 numbers: the number of pixels in bands 1, 2, & 3 that have intensity `g`.

`sum(sum(I==g))` computes one number for each band in the image.

Point Ops via Functional Mappings



$$\mathbf{J} = \mathsf{F}[\mathbf{I}]$$

If $\mathbf{I}(r,c) = g$
and $f(g) = k$
then $\mathbf{J}(r,c) = k$.

The transformation of image **I** into image **J** is accomplished by replacing each input intensity, g, with a specific output intensity, k, at every location (r,c) where $\mathbf{I}(r,c) = g$.

The rule that associates k with g is usually specified with a function, f, so that $f(g) = k$.

Point Ops via Functional Mappings

One-band Image

$$\mathbf{J}(r,c) = f\left(\mathbf{I}(r,c)\right),$$
for all pixels locations (r,c).

Three-band Image

$$\mathbf{J}(r,c,b) = f\left(\mathbf{I}(r,c,b)\right), \text{ or}$$
$$\mathbf{J}(r,c,b) = f_b\left(\mathbf{I}(r,c,b)\right),$$
for $b = 1,2,3$ and all (r,c).

Point Ops via Functional Mappings

One-band Image

$$\mathbf{J}(r,c) = f\left(\mathbf{I}(r,c)\right),$$
$$\text{for all pixels locations } (r,c).$$

Either all 3 bands are mapped through the same function, f, or …

Three-band Image

$$\mathbf{J}(r,c,b) = f\left(\mathbf{I}(r,c,b)\right), \text{ or}$$
$$\mathbf{J}(r,c,b) = f_b\left(\mathbf{I}(r,c,b)\right),$$
$$\text{for } b = 1,2,3 \text{ and all } (r,c$$

… each band is mapped through a separate function, f_b.

Point Operations using Look-up Tables

A look-up table (LUT) implements a functional mapping.

If $k = f(g)$,
for $g = 0,\ldots,255$,
and if k takes on values in $\{0,\ldots,255\},\ldots$

… then the LUT that implements f is a $256\text{x}1$ array whose $(g+1)^{\text{th}}$ value is $k = f(g)$.

To remap a 1-band image, $\mathbf{I}$, to $\mathbf{J}$:

$$\mathbf{J} = \text{LUT}(\mathbf{I}+1)$$

Point Operations using Look-up Tables

If **I** is 3-band, then

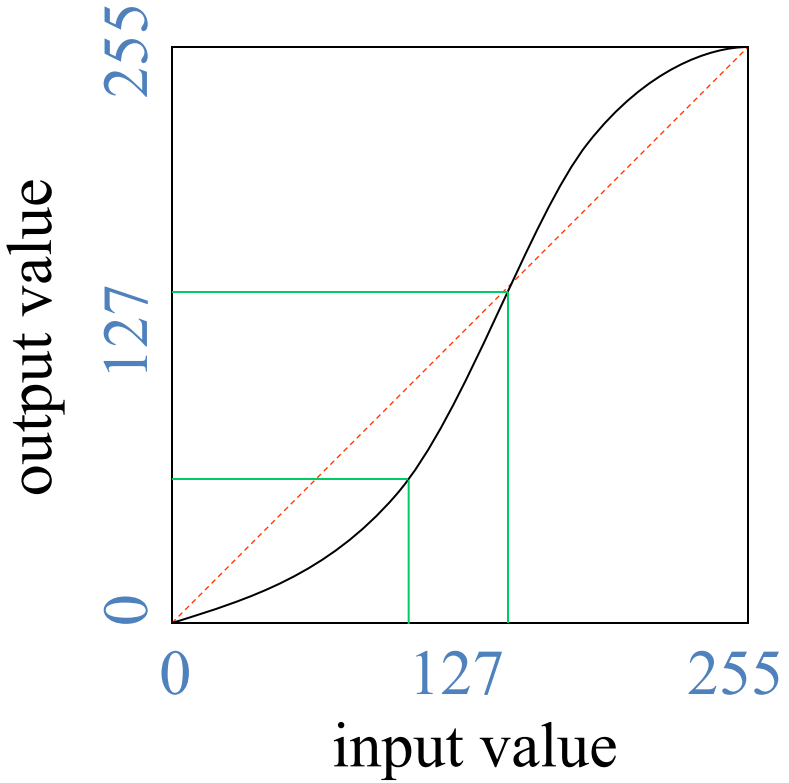
a) each band is mapped separately using the same LUT for each band *or*

b) each band is mapped using different LUTs – one for each band.

a) $\mathbf{J} = \mathrm{LUT}(\mathbf{I}+1)$, *or*

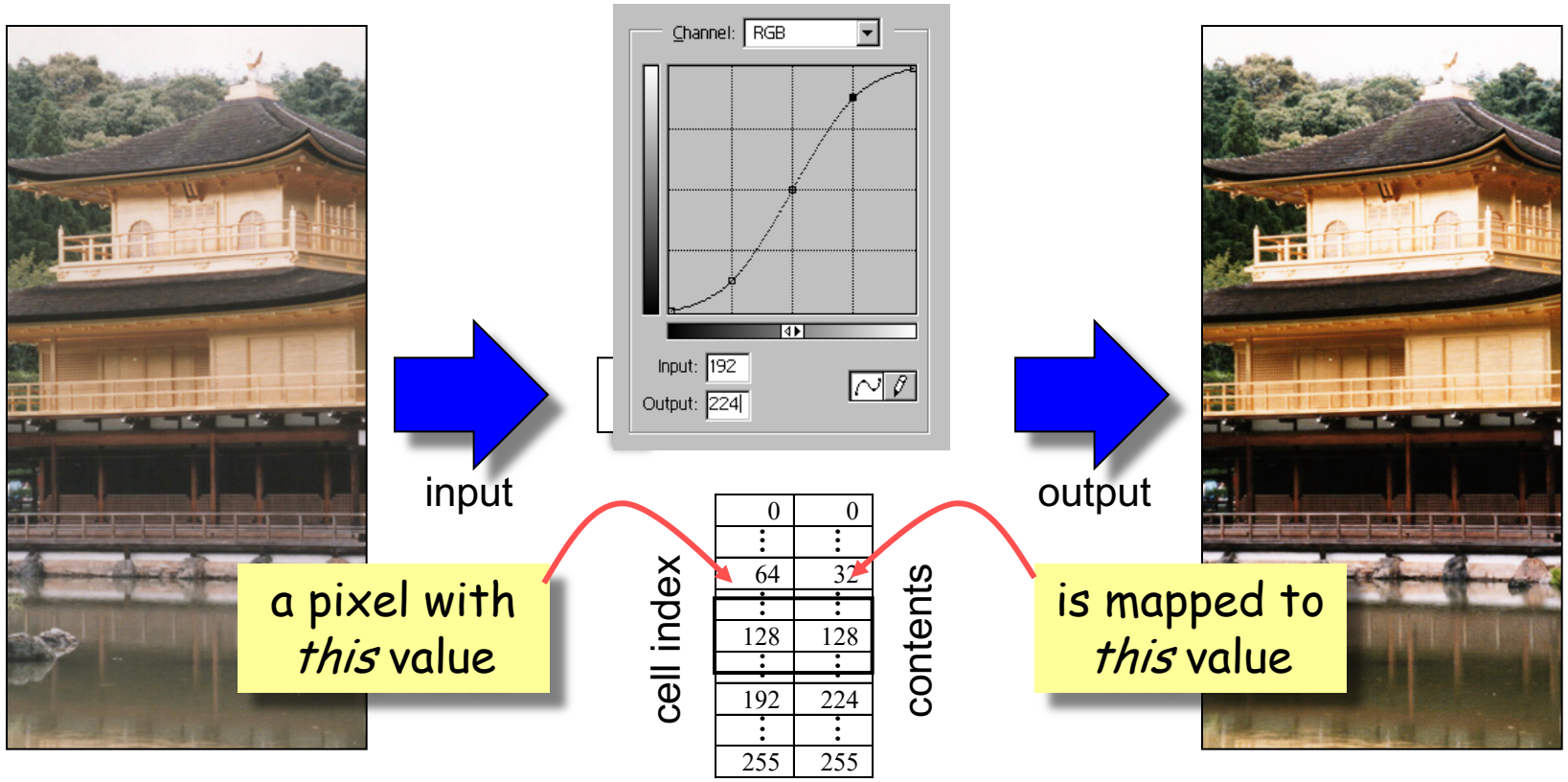
b) $\mathbf{J}(:,:,b) = \mathrm{LUT}_b(\mathbf{I}(:,:,b)+1)$ for $b = 1,2,3.$

Point Operations = Look-up Table Ops



E.g.:	index	value
	...	...
	101	64
	102	68
	103	69
	104	70
	105	70
	106	71
	...	...
	input	output

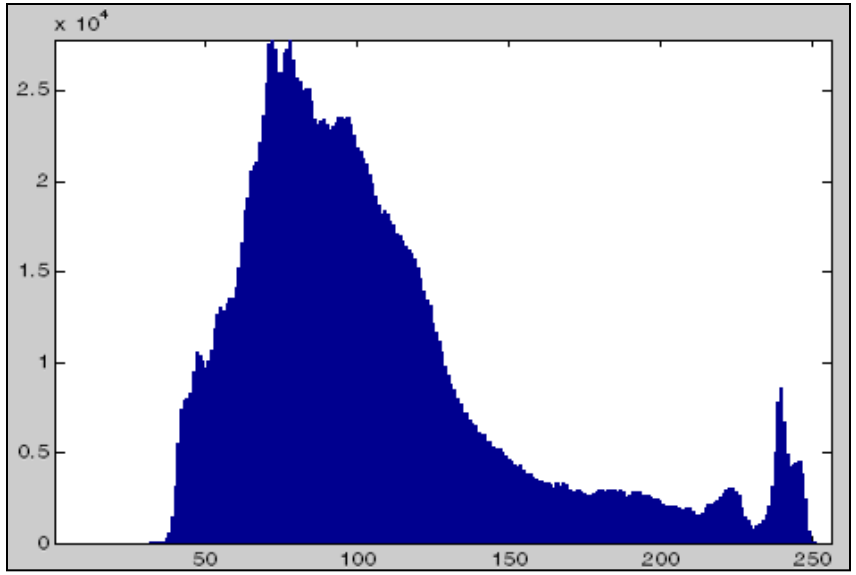
Look-Up Tables



Point Processes: Original Image

Kinkaku-ji (金閣寺 Temple of the Golden Pavilion), also known as Rokuon-ji (鹿苑寺 Deer Garden Temple), is a Zen Buddhist temple in Kyoto, Japan.

Photo by Richard Alan Peters II, August 1993.

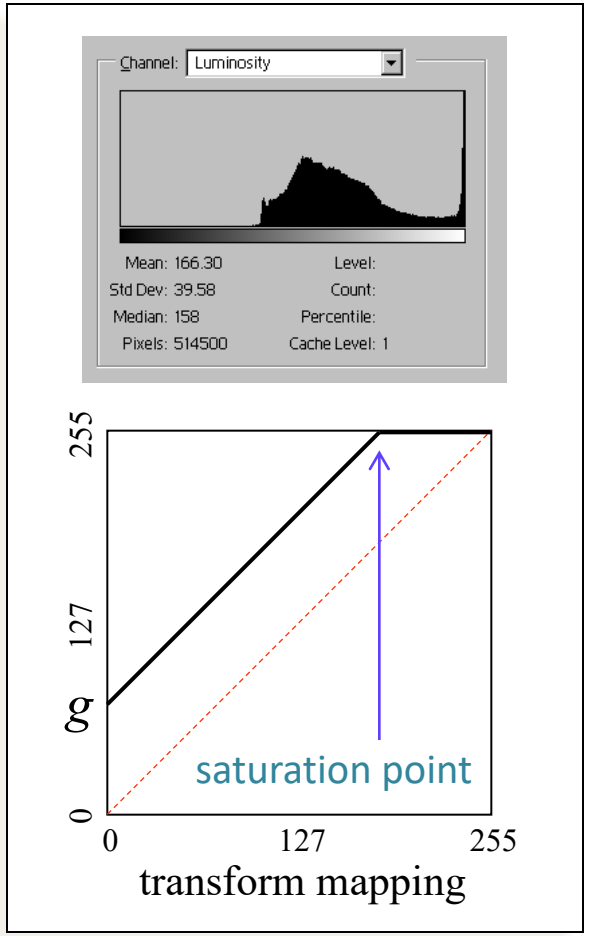


Luminance Histogram

Point Processes: Increase Brightness

$$\mathbf{J}(r,c,b) = \begin{cases} \mathbf{I}(r,c,b) + g, & \text{if } \mathbf{I}(r,c,b) + g < 256 \\ 255, & \text{if } \mathbf{I}(r,c,b) + g > 255 \end{cases}$$

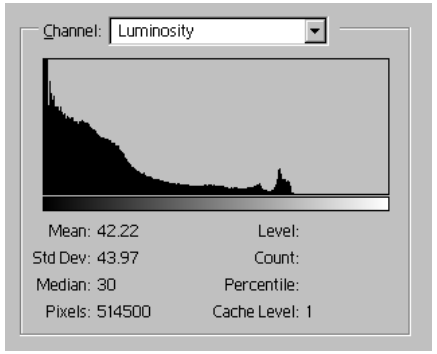
$g \geq 0$ and $b \in \{1,2,3\}$ is the band index.

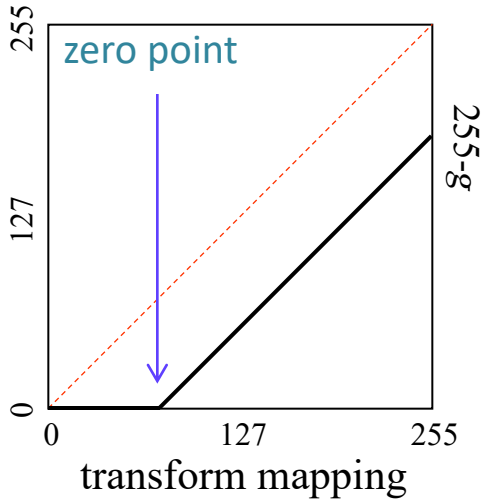


Point Processes: Decrease Brightness

$$\mathbf{J}(r,c,b) = \begin{cases} 0, & \text{if } \mathbf{I}(r,c,b) - g < 0 \\ \mathbf{I}(r,c,b) - g, & \text{if } \mathbf{I}(r,c,b) - g > 0 \end{cases}$$

$g \geq 0$ and $b \in \{1,2,3\}$ is the band index.





Point Processes: Decrease Contrast

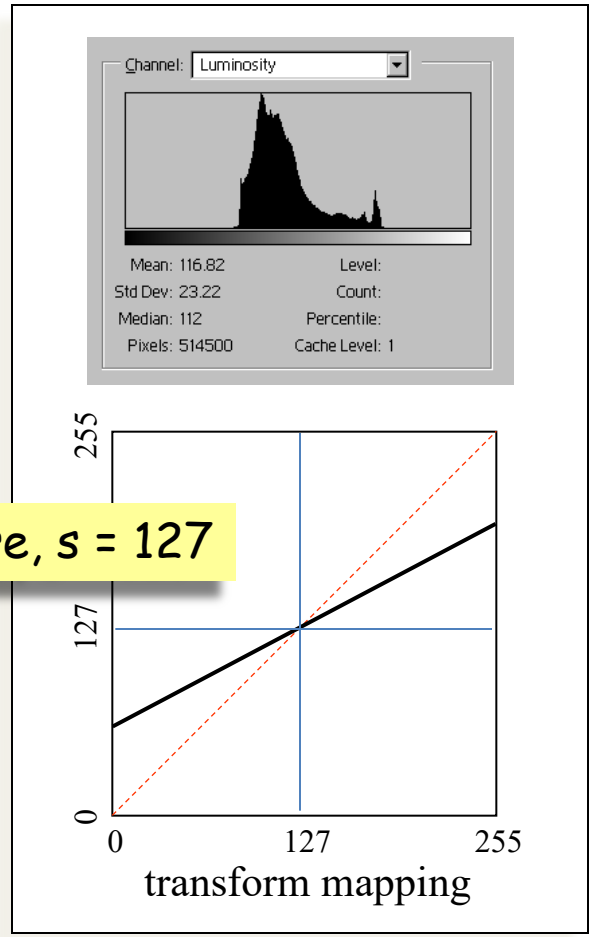
$$\mathbf{T}(r,c,b) = a[\mathbf{I}(r,c,b) - s] + s,$$

where $0 \le a < 1.0$,

$s \in \{0,1,2,\ldots,255\}$, and

$b \in \{1,2,3\}$.

s is the center of the contrast function.



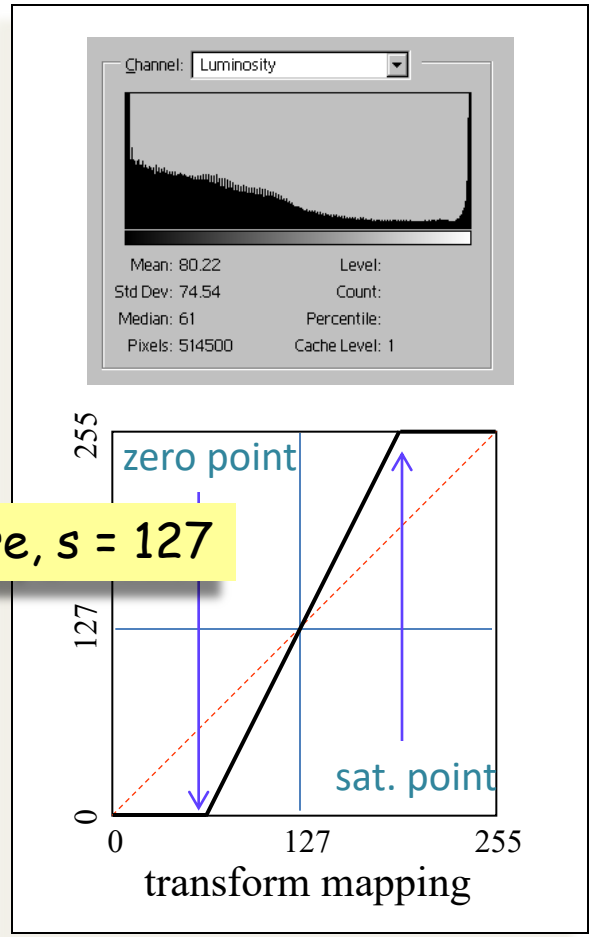
Here, s = 127

Point Processes: Increase Contrast

$$\mathbf{T}(r,c,b) = a\left[\mathbf{I}(r,c,b) - s\right] + s$$

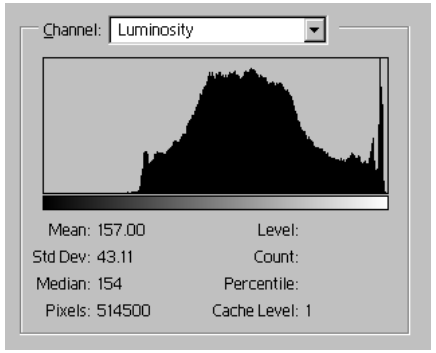
$$\mathbf{J}(r,c,b) = \begin{cases} 0, & \text{if } \mathbf{T}(r,c,b) < 0, \\ \mathbf{T}(r,c,b), & \text{if } 0 \le \mathbf{T}(r,c,b) \le 255, \\ 255, & \text{if } \mathbf{T}(r,c,b) > 255. \end{cases}$$

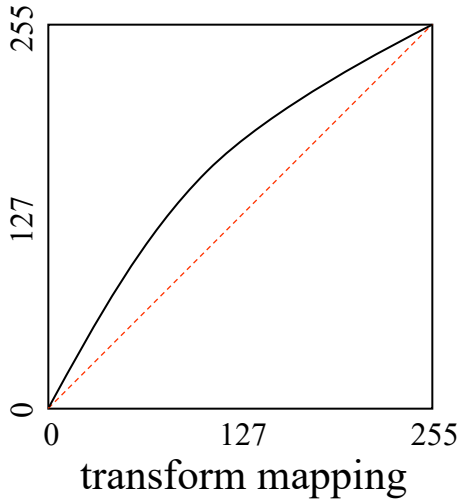
$$a > 1, \quad s \in \{0,\ldots,255\}, \quad b \in \{1,2,3\}$$



Point Processes: Increased Gamma

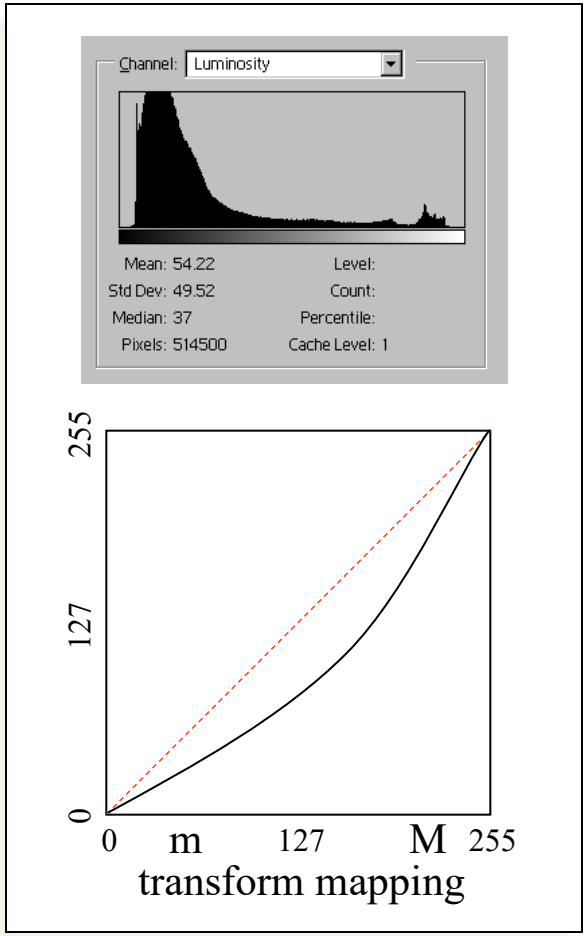
$$\mathbf{J}(r,c) = 255 \cdot \left[\frac{\mathbf{I}(r,c)}{255} \right]^{\frac{1}{\gamma}} \quad \text{for} \quad \gamma > 1.0$$





Point Processes: Decreased Gamma

$$\mathbf{J}(r,c) = 255 \cdot \left[\frac{\mathbf{I}(r,c)}{255}\right]^{\frac{1}{\gamma}} \quad \text{for} \quad \gamma < 1.0$$



The Probability Density Function of an Image

Let $A = \sum_{g=0}^{255} h_{\mathbf{I}_k}(g+1)$.

pdf
[lower case]

Note that since $h_{\mathbf{I}_k}(g+1)$ is the number of pixels in $\mathbf{I}_k$ (the kth color band of image $\mathbf{I}$) with value g, A is the number of pixels in $\mathbf{I}$. That is if $\mathbf{I}$ is R rows by C columns then $A = R \times C$.

Then,

$$p_{\mathbf{I}_k}(g+1) = \frac{1}{A} h_{\mathbf{I}_k}(g+1)$$

This is the probability that an arbitrary pixel from $\mathbf{I}_k$ has value g.

is the graylevel probability density function of $\mathbf{I}_k$.

The Probability Density Function of an Image

- $p_{band}(g+1)$ is the fraction of pixels in (a specific band of) an image that have intensity value g.
- $p_{band}(g+1)$ is the probability that a pixel randomly selected from the given band has intensity value g.
- Whereas the sum of the histogram $h_{band}(g+1)$ over all g from 1 to 256 is equal to the number of pixels in the image, the sum of $p_{band}(g+1)$ over all g is 1.
- p_{band} is the normalized histogram of the band.

The Probability Distribution Function of an Image

Let $\mathbf{q} = [q_1\ q_2\ q_3] = \mathbf{I}(r,c)$ be the value of a randomly selected pixel from $\mathbf{I}$. Let g be a specific graylevel. The probability that $q_k \leq g$ is given by

$$P_{I_k}(g+1) = \sum_{\gamma=0}^{g} p_{\mathbf{I}_k}(\gamma+1) = \frac{1}{A}\sum_{\gamma=0}^{g} h_{\mathbf{I}_k}(\gamma+1) = \frac{\sum_{\gamma=0}^{g} h_{\mathbf{I}_k}(\gamma+1)}{\sum_{\gamma=0}^{255} h_{\mathbf{I}_k}(\gamma+1)},$$

where $h_{Ik}(\gamma+1)$ is the histogram of the kth band of $\mathbf{I}$.

PDF [upper case]

This is the probability that any given pixel from $\mathbf{I}_k$ has value less than or equal to g.

Point Processes: Histogram Equalization

Task: remap image **I** so that its histogram is as close to constant as possible

Let $P_{\mathbf{I}}(\mathrm{g}+1)$ be the probability distribution function of **I**.

Then **J** has, as closely as possible, the correct histogram if

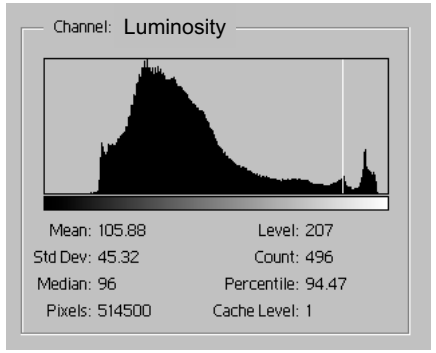
$$\mathbf{J}(r,c,b) = 255 \cdot P_{\mathbf{I}}\left[\mathbf{I}(r,c,b)+1\right].$$

The PDF itself is used as the LUT.

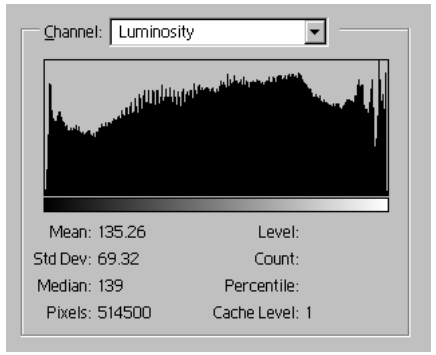
all bands processed similarly

Point Processes: Histogram Equalization

$$\mathbf{J}(r,c,b) = 255 \cdot P_{\mathbf{I}}(g+1),$$
$$g = \mathbf{I}(r,c,b), \quad \mathrm{b} \in \{1,2,3\}.$$



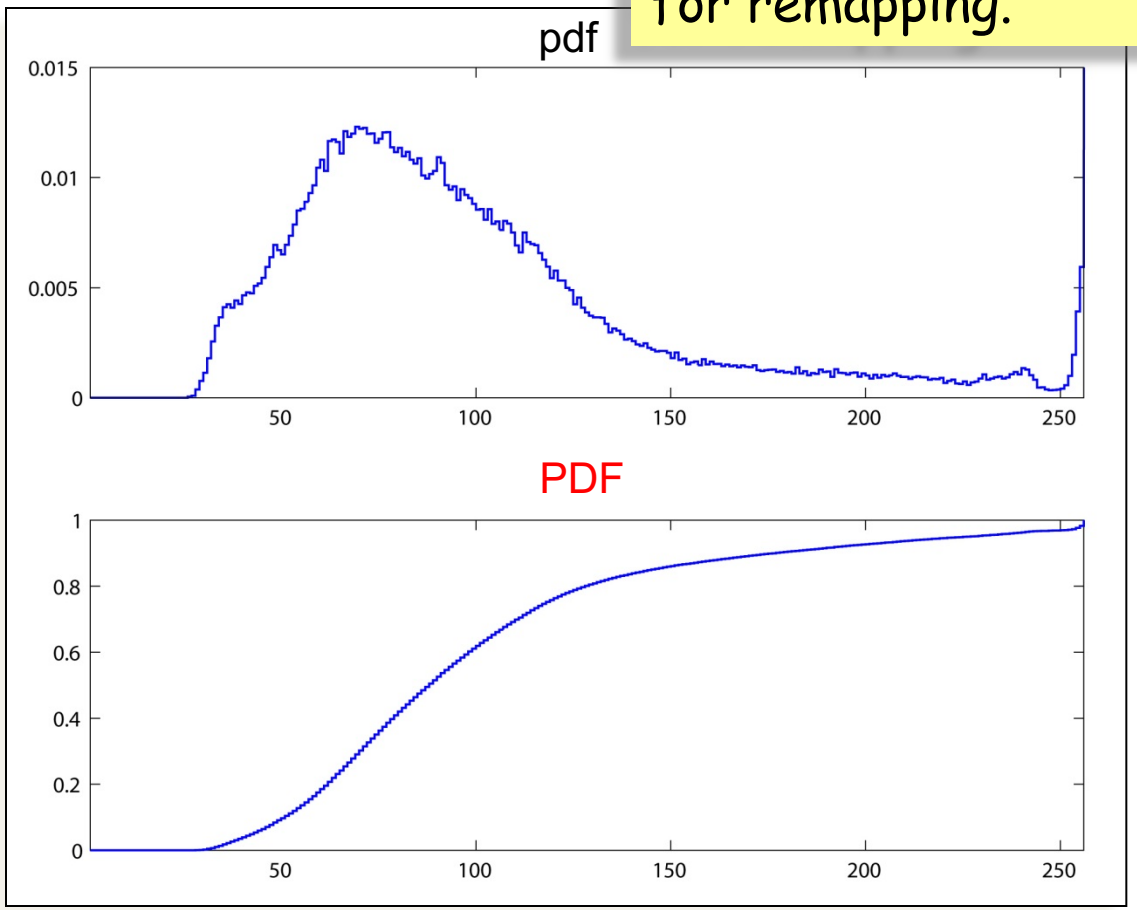
before



after

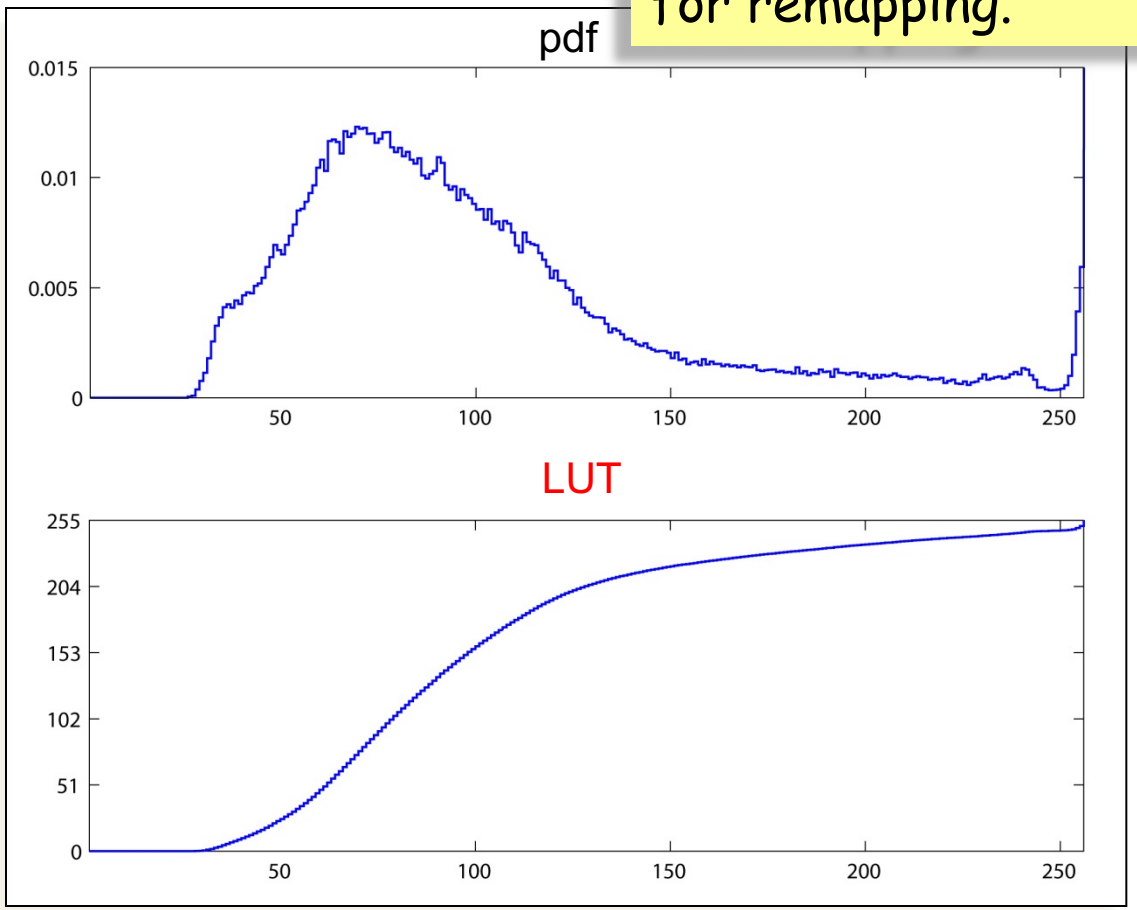
Histogram EQ

The PDF is the LUT for remapping.



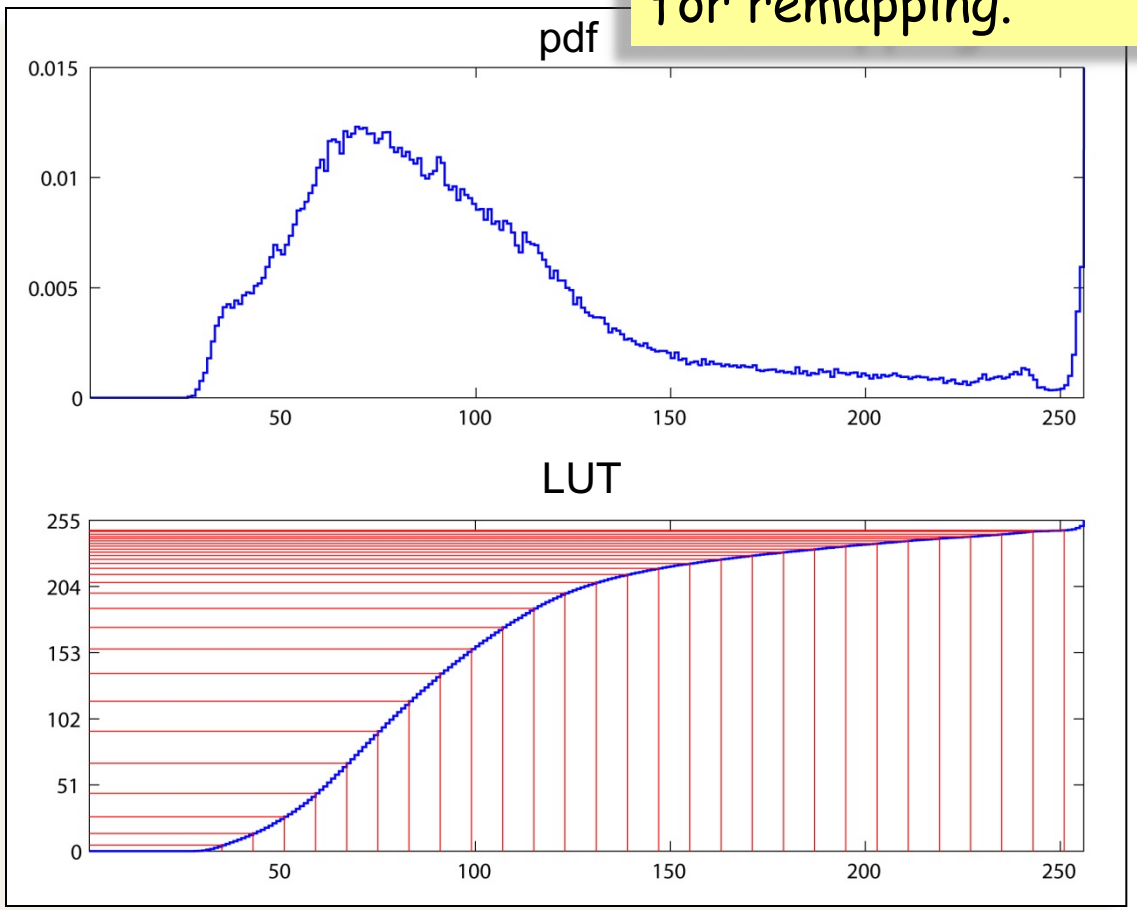
Histogram EQ

The PDF is the LUT for remapping.



Histogram EQ

The PDF is the LUT for remapping.



Point Processes: Histogram Equalization

Task: remap image $\mathbf{I}$ with min = $m_{\mathbf{I}}$ and max = $M_{\mathbf{I}}$ so that its histogram is as close to constant as possible and has min = $m_{\mathbf{J}}$ and max = $M_{\mathbf{J}}$.

Let $P_{\mathbf{I}}(\mathrm{g}+1)$ be the probability distribution function of $\mathbf{I}$.

Then $\mathbf{J}$ has, as closely as possible, the correct histogram if

Using intensity extrema

$$\mathbf{J}(r,c) = (M_{\mathbf{J}} - m_{\mathbf{J}}) \frac{P_{\mathbf{I}}[\mathbf{I}(r,c)+1] - P_{\mathbf{I}}(m_{\mathbf{I}}+1)}{P_{\mathbf{I}}(M_{\mathbf{I}}+1) - P_{\mathbf{I}}(m_{\mathbf{I}}+1)} + m_{\mathbf{J}}.$$

Point Processes: Histogram Matching

Task: remap image **I** so that it has, as closely as possible, the same histogram as image **J**.

Because the images are digital it is not, in general, possible to make $h_{\mathbf{I}} \equiv h_{\mathbf{J}}$. Therefore, $p_{\mathbf{I}} \not\equiv p_{\mathbf{J}}$.

Q: How, then, can the matching be done?
A: By matching percentiles.

Matching Percentiles

… assuming a 1-band image or a single band of a color image.

Recall:

- The PDF of image **I** is such that $0 \leq P_{\mathbf{I}}(g_{\mathbf{I}}) \leq 1$.
- $P_{\mathbf{I}}(g_{\mathbf{I}}+1) = c$ means that c is the fraction of pixels in **I** that have a value less than or equal to $g_{\mathbf{I}}$.
- $100c$ is the *percentile* of pixels in **I** that are less than or equal to $g_{\mathbf{I}}$.

To match percentiles, replace all occurrences of value $g_{\mathbf{I}}$ in image **I** with the value, $g_{\mathbf{J}}$, from image **J** whose percentile in **J** most closely matches the percentile of $g_{\mathbf{I}}$ in image **I**.

Matching Percentiles

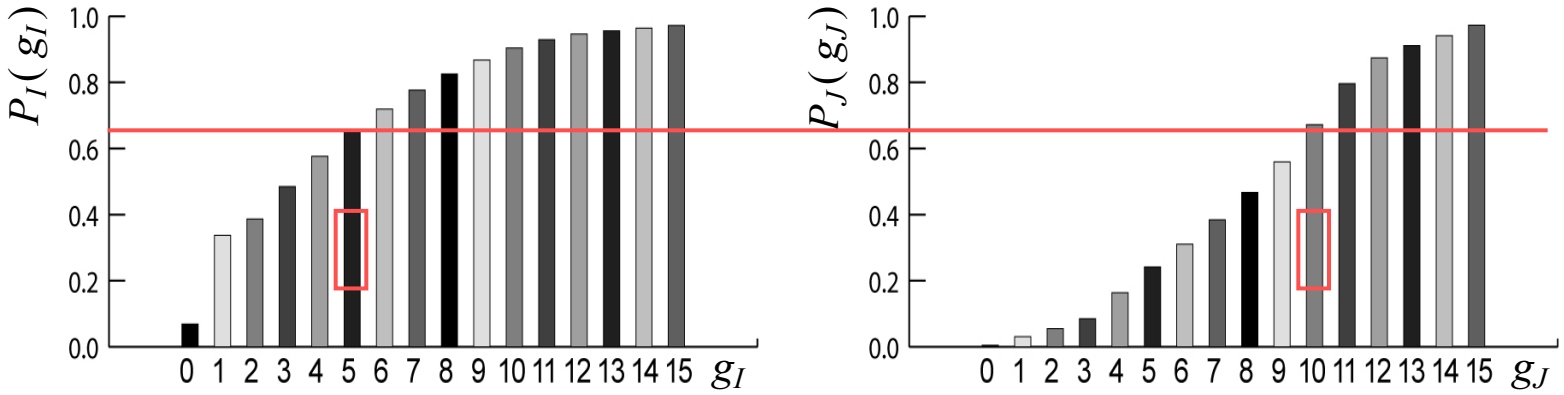
... assuming a 1-band image or a single band of a color image.

So, to create an image, **K**, from image **I** such that **K** has nearly the same PDF as image **J** do the following:

If $\mathbf{I}(r,c) = g_{\mathbf{I}}$ then let $K(r,c) = g_{\mathbf{J}}$ where $g_{\mathbf{J}}$ is such that

$P_{\mathbf{I}}(g_{\mathbf{I}}) > P_{\mathbf{J}}(g_{\mathbf{J}}-1)$ AND $P_{\mathbf{I}}(g_{\mathbf{I}}) \leq P_{\mathbf{J}}(g_{\mathbf{J}})$.

Example:
$\mathbf{I}(r,c) = 5$
$P_{\mathbf{I}}(5) = 0.65$
$P_{\mathbf{J}}(9) = 0.56$
$P_{\mathbf{J}}(10) = 0.67$
$\mathbf{K}(r,c) = 10$



Histogram Matching Algorithm

Assuming a 1-band image or a single band of a color image.

$[R,C] = \texttt{size}(\mathbf{I});$

$\mathbf{K} = \texttt{zeros}(R, C);$

$g_{\mathbf{J}} = m_{\mathbf{J}};$

for $g_{\mathbf{I}} = m_{\mathbf{I}}$ to $M_{\mathbf{I}}$

  while $g_{\mathbf{J}} < 255$ AND $P_{\mathbf{I}}(g_{\mathbf{I}}+1) < 1$ AND

    $P_{\mathbf{J}}(g_{\mathbf{J}}+1) < P_{\mathbf{I}}(g_{\mathbf{I}}+1)$

    $g_{\mathbf{J}} = g_{\mathbf{J}} + 1;$

  end

  $\mathbf{K} = \mathbf{K} + \left[g_{\mathbf{J}} \cdot \left(\mathbf{I} == g_{\mathbf{I}} \right) \right]$

end

This directly matches image $\mathbf{I}$ to image $\mathbf{J}$.

$P_{\mathbf{I}}(g_{\mathbf{I}}+1)$: PDF of $\mathbf{I}$

$P_{\mathbf{J}}(g_{\mathbf{J}}+1)$: PDF of $\mathbf{J}$.

$m_{\mathbf{J}} = \min \mathbf{J},$

$M_{\mathbf{J}} = \max \mathbf{J},$

$m_{\mathbf{I}} = \min \mathbf{I},$

$M_{\mathbf{I}} = \max \mathbf{I}.$

Better to use a LUT.

Example: Histogram Matching

Image pdf

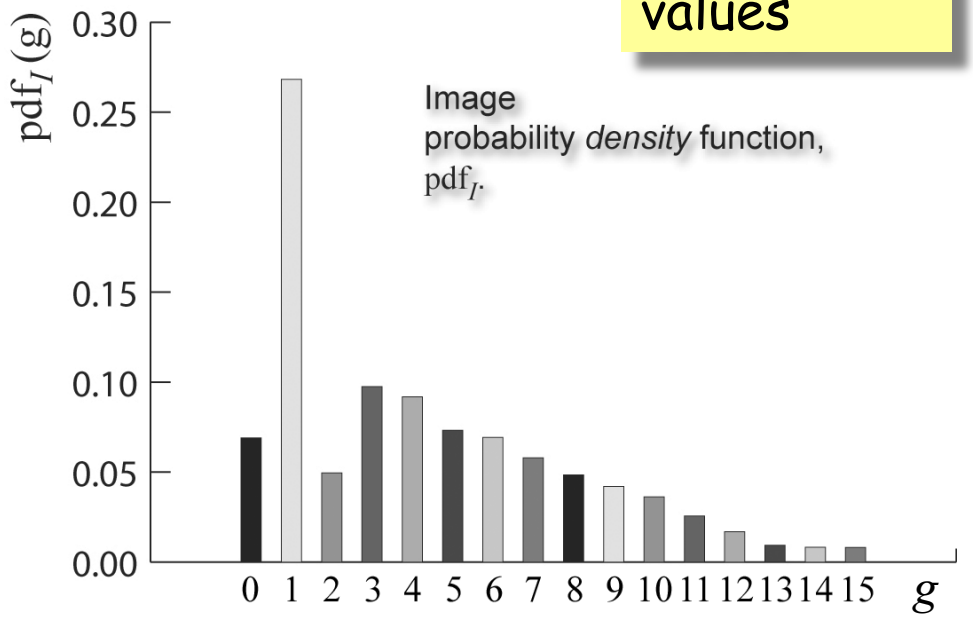


Image with 16 intensity values

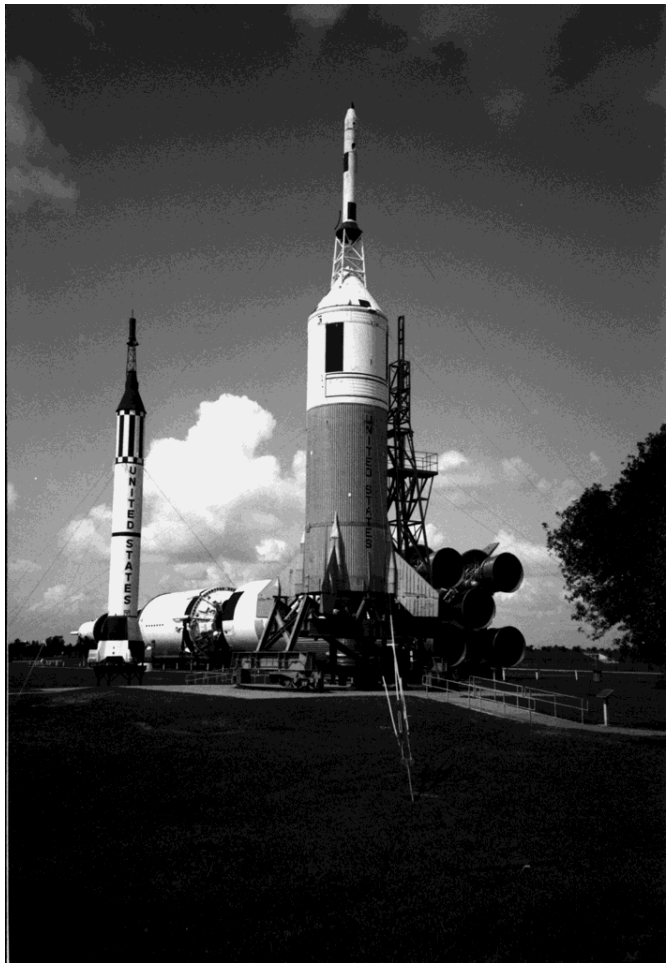
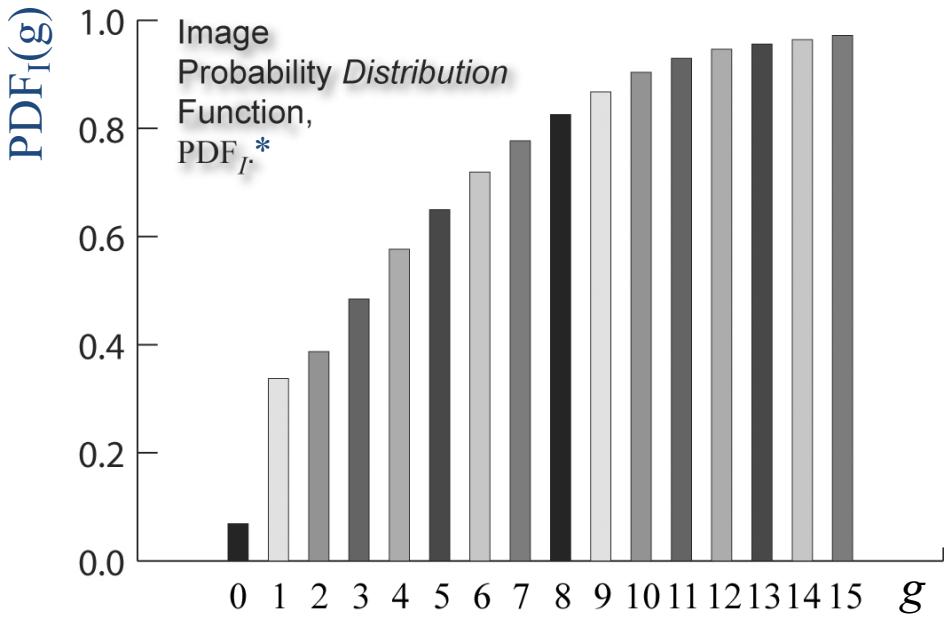
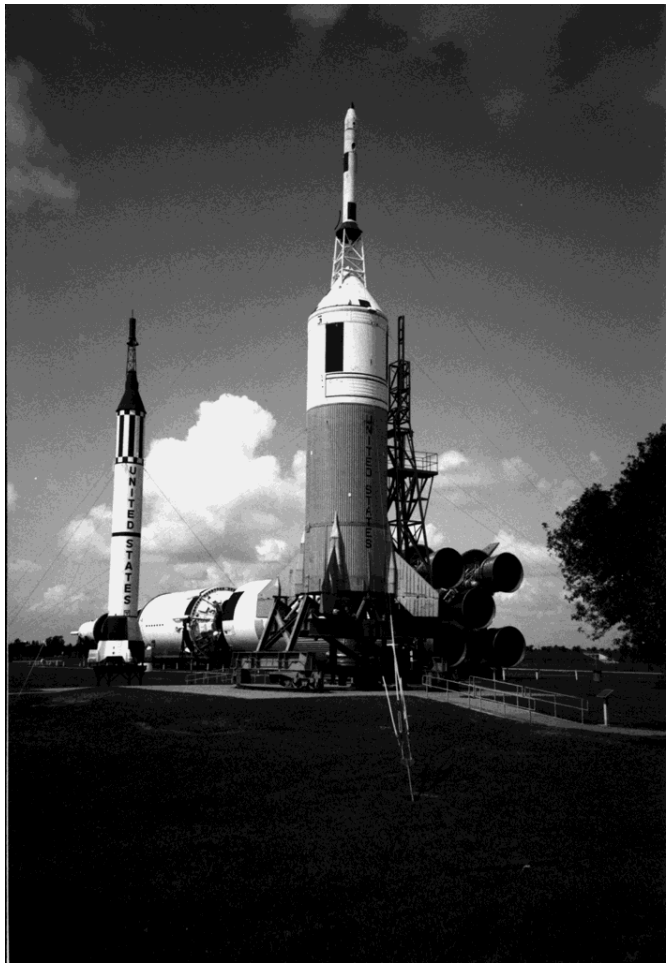


Image PDF

Example: Histogram Matching

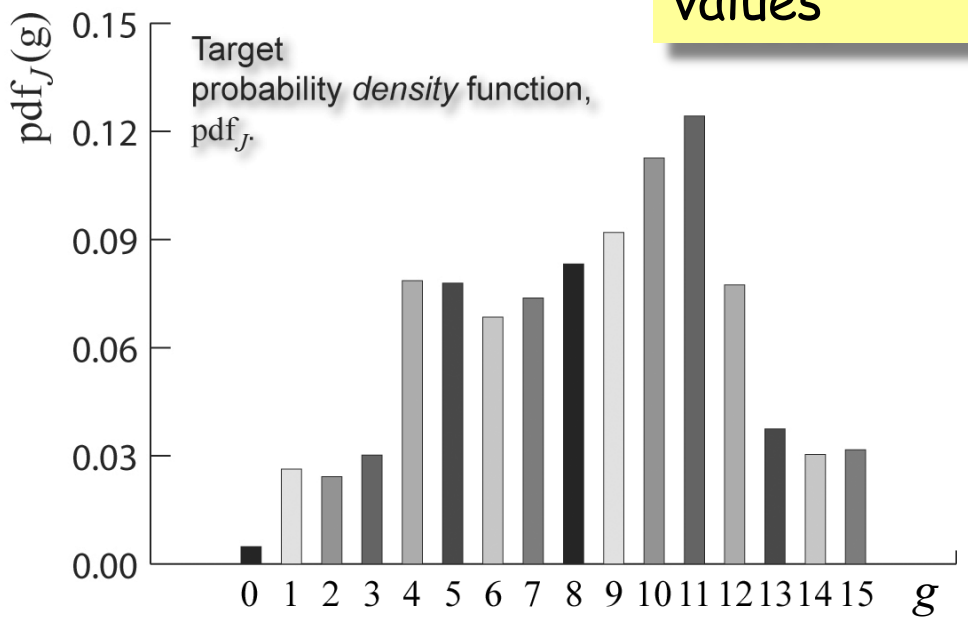




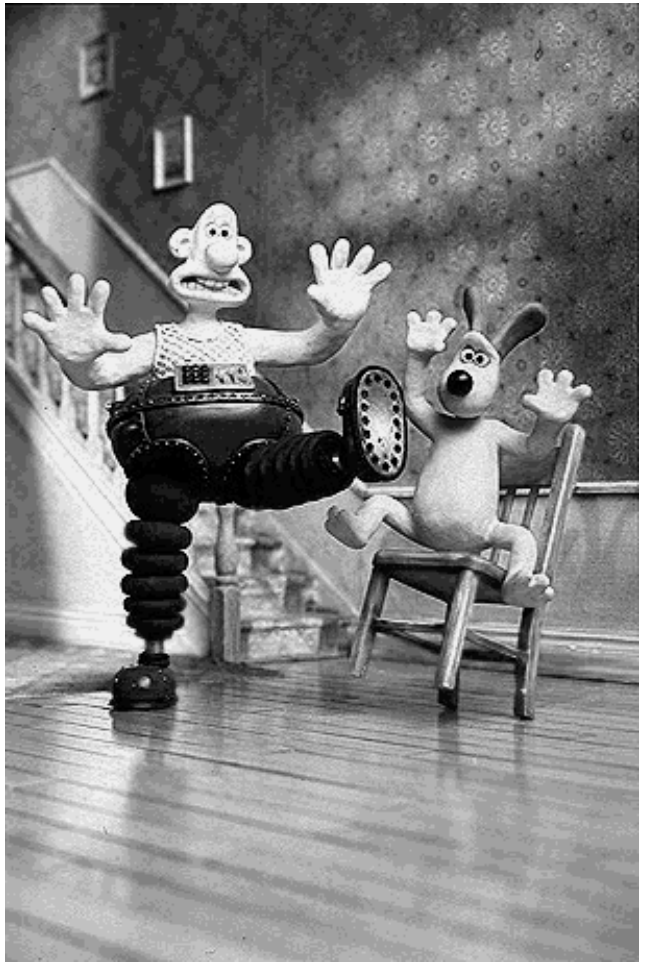
*a.k.a Cumulative Distribution Function, CDF$_I$.

Example: Histogram Matching

Target pdf

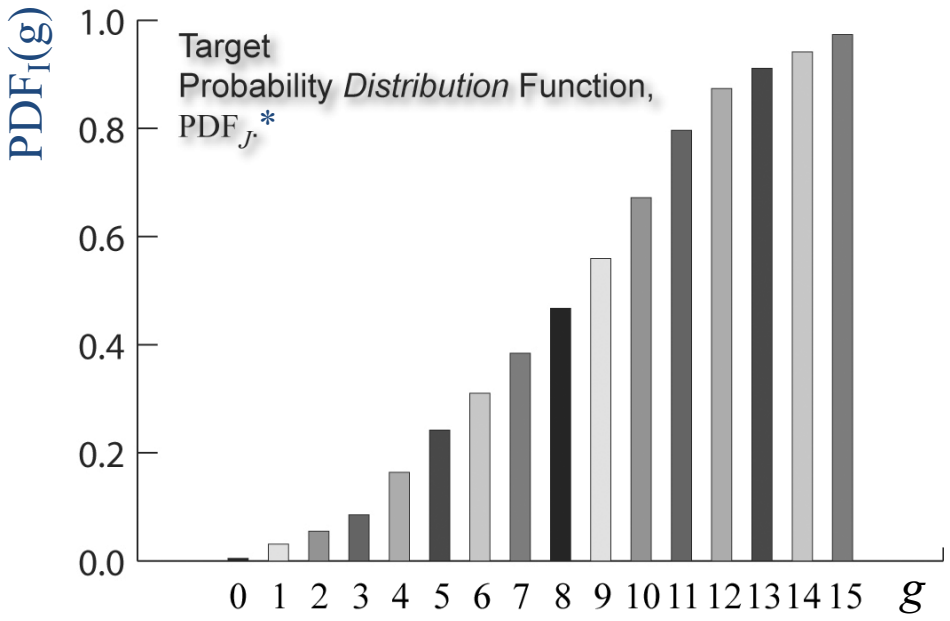


Target with 16 intensity values

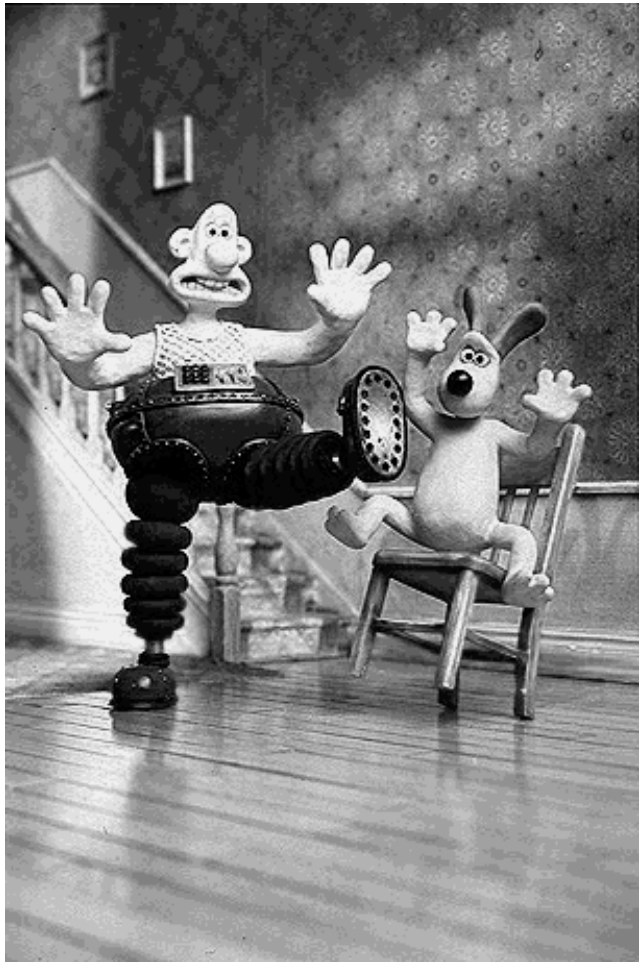


Target PDF

Example: Histogram Matching



*a.k.a Cumulative Distribution Function, CDF$_J$.



Histogram Matching with a Lookup Table

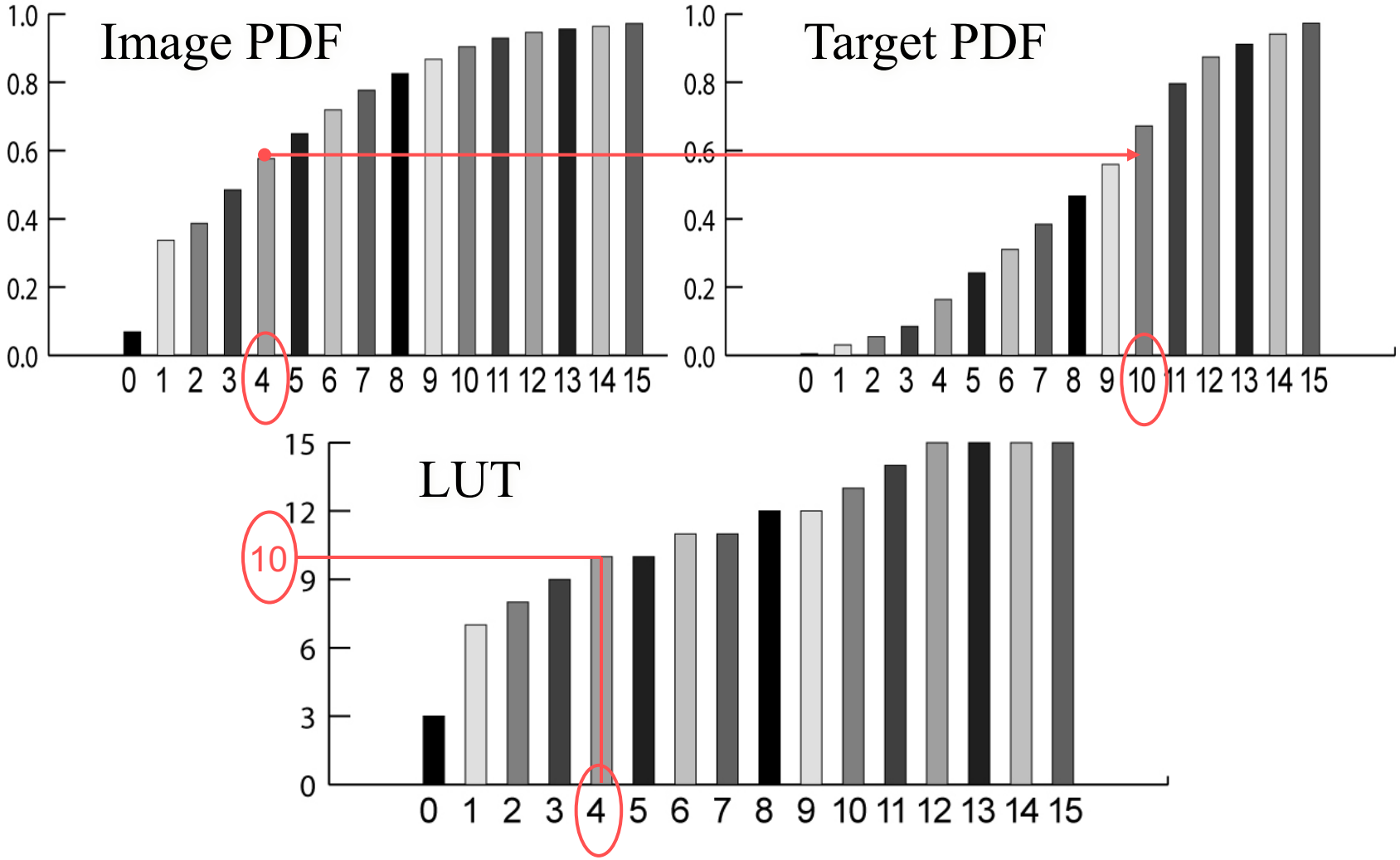
Often it is faster or more versatile to use a lookup table (LUT). Rather than remapping each pixel in the image separately, one can create a table that indicates to which target value each input value should be mapped. Then

$$\mathbf{K} = \mathrm{LUT}[\mathbf{I}+1]$$

In *Matlab* if the LUT is a 256 × 1 matrix with values from 0 to 255 and if image **I** is of type **`uint8`**, it can be remapped with the following code:

```
K = uint8(LUT(I+1));
```

LUT Creation



Look Up Table for Histogram Matching

```
LUT = zeros(256,1);
g_J = 0;
for g_I = 0 to 255
    while P_J(g_J+1) < P_I(g_I+1) AND g_J < 255
        g_J = g_J + 1;
    end
    LUT(g_I+1) = g_J;
end
```

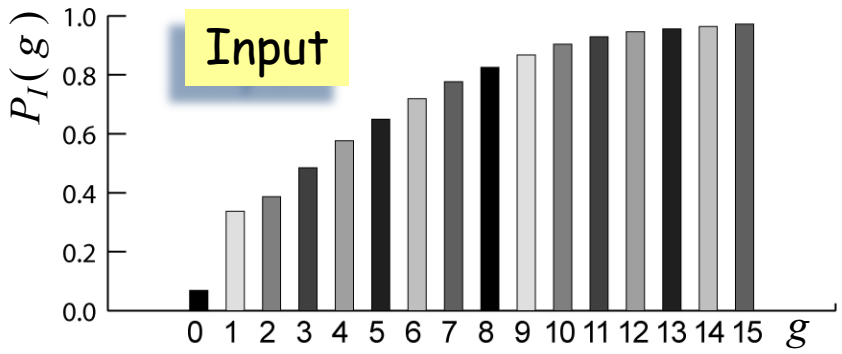
This creates a look-up table which can then be used to remap the image.

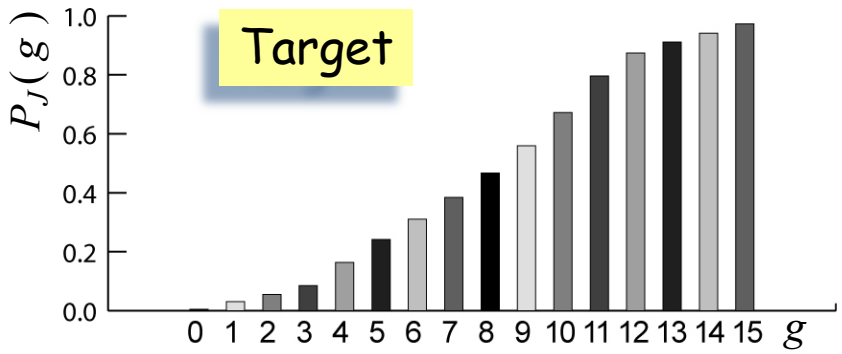
$P_{\mathbf{I}}(g_{\mathbf{I}}+1)$: PDF of **I**,

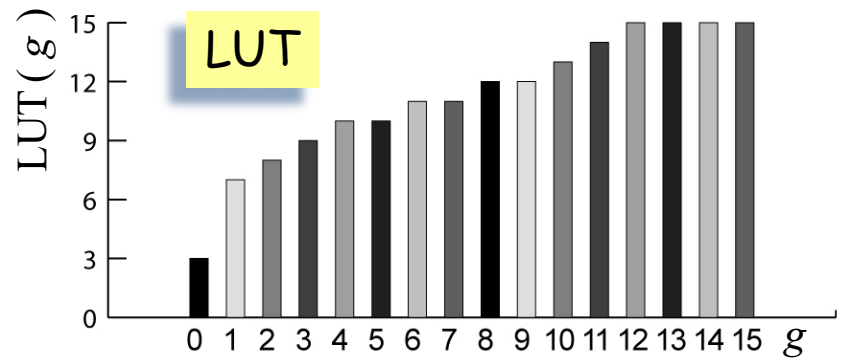
$P_{\mathbf{J}}(g_{\mathbf{J}}+1)$: PDF of **J**,

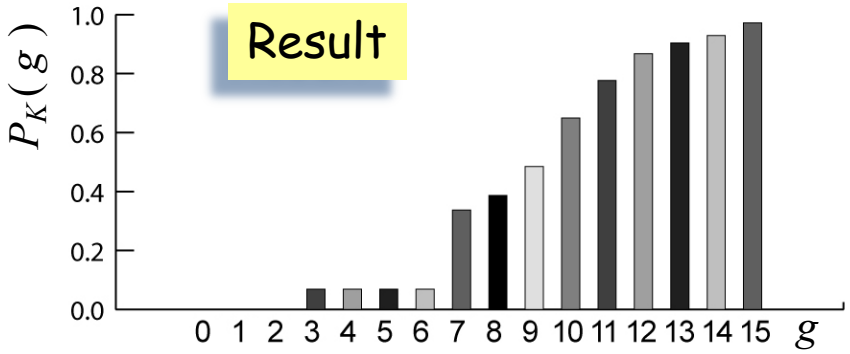
$\mathrm{LUT}(g_{\mathbf{I}}+1)$: Look- Up Table

Input & Target PDFs, LUT and Resultant PDF

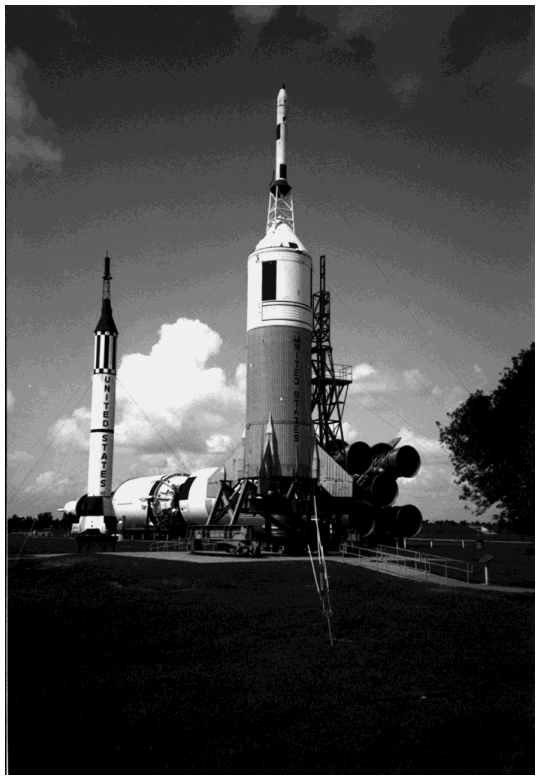




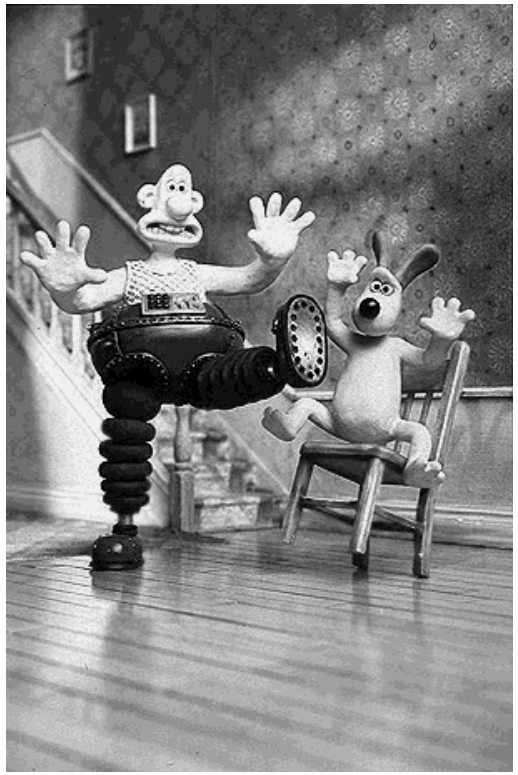




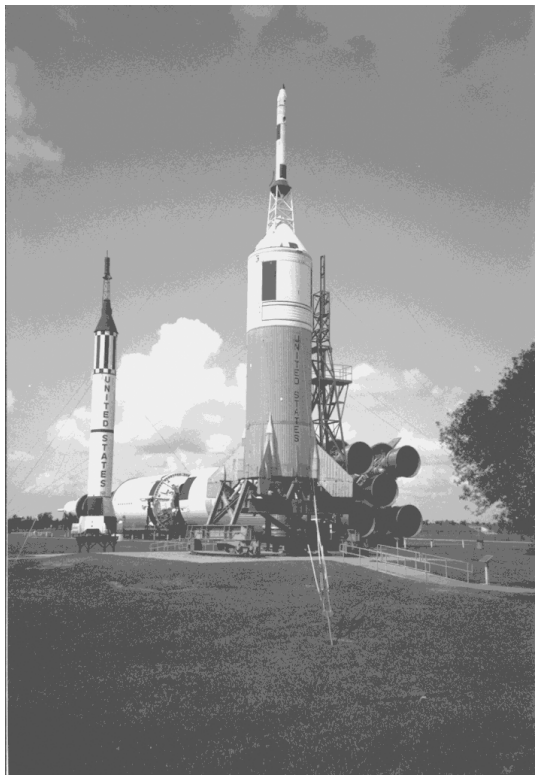
Example: Histogram Matching



original



target



remapped