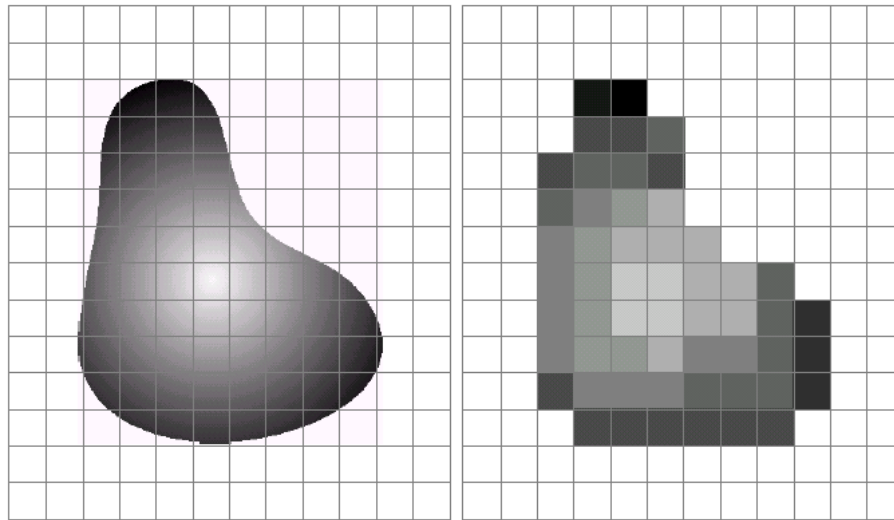# Advanced Multimedia

Image Content Analysis

Tamara Berg

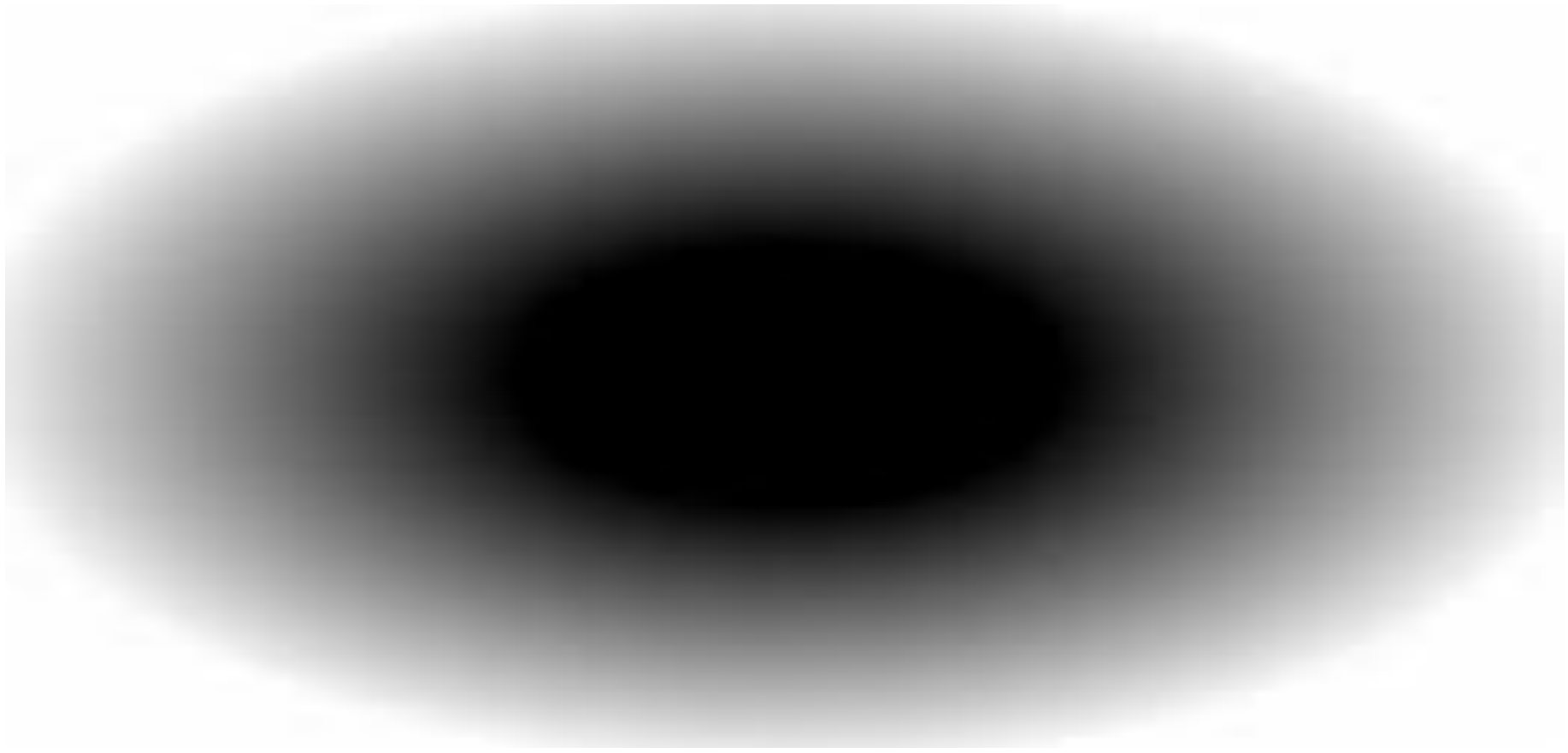# How are images stored?

Reminder: Images

a b

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

Images are sampled and quantized measurements of light hitting a sensor.
What do we mean by sampled?
What is being quantized?

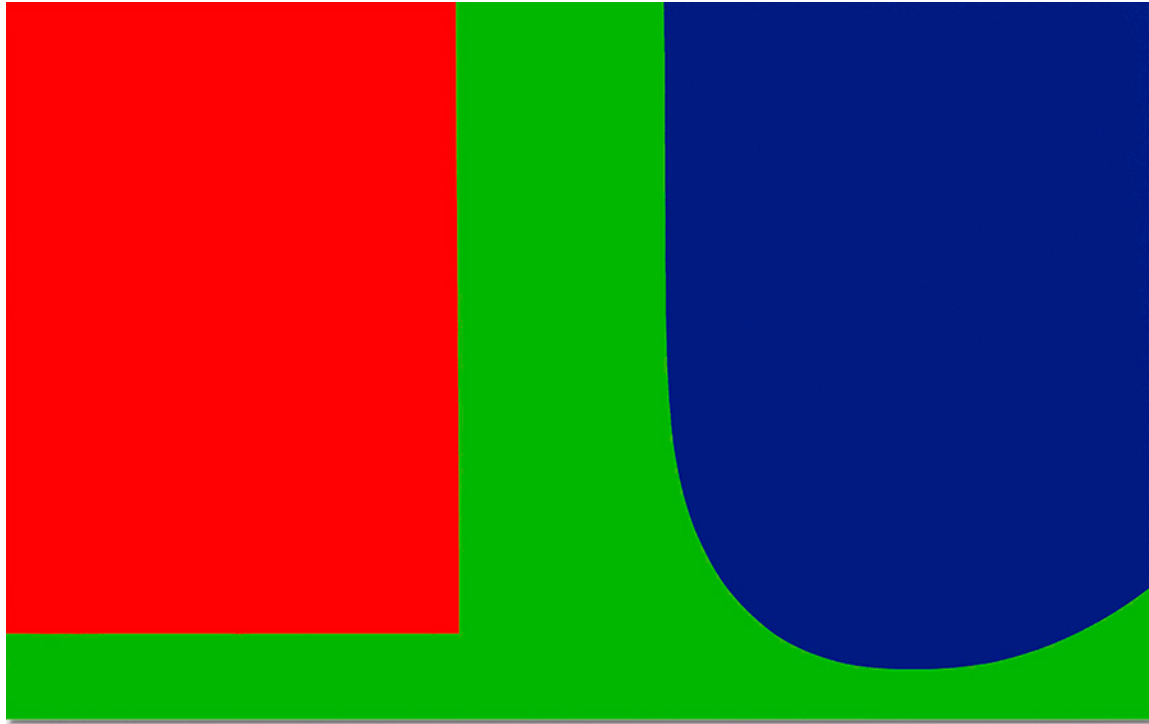# Images in the computer

Images in the computer

Color Images

img =

Ellsworth Kelly
Red Blue Green,
1963

Color Images

img =

Ellsworth Kelly
Red Blue Green,
1963

N

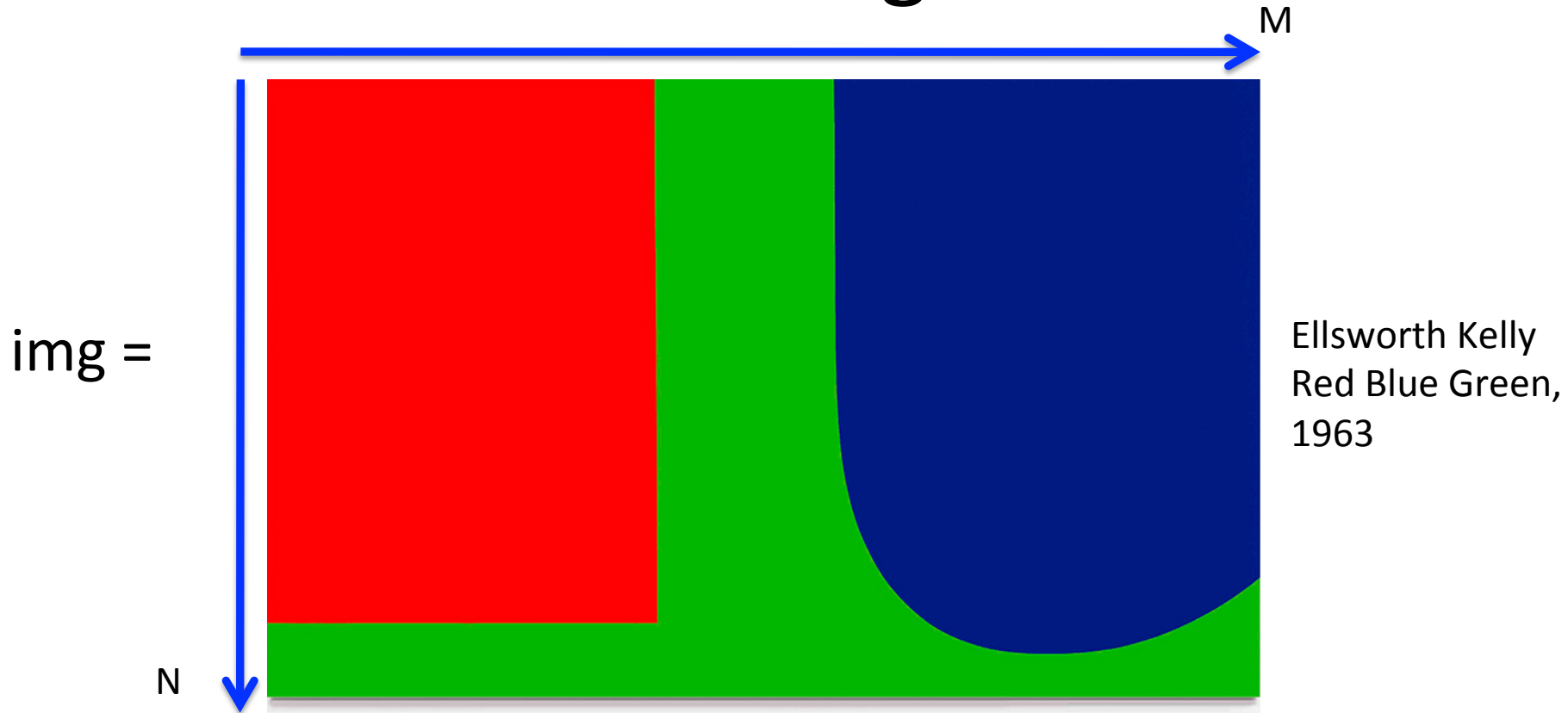# Color Images

img =

M

N

Ellsworth Kelly
Red Blue Green,
1963

Color Images

img =

Ellsworth Kelly
Red Blue Green,
1963

M

N
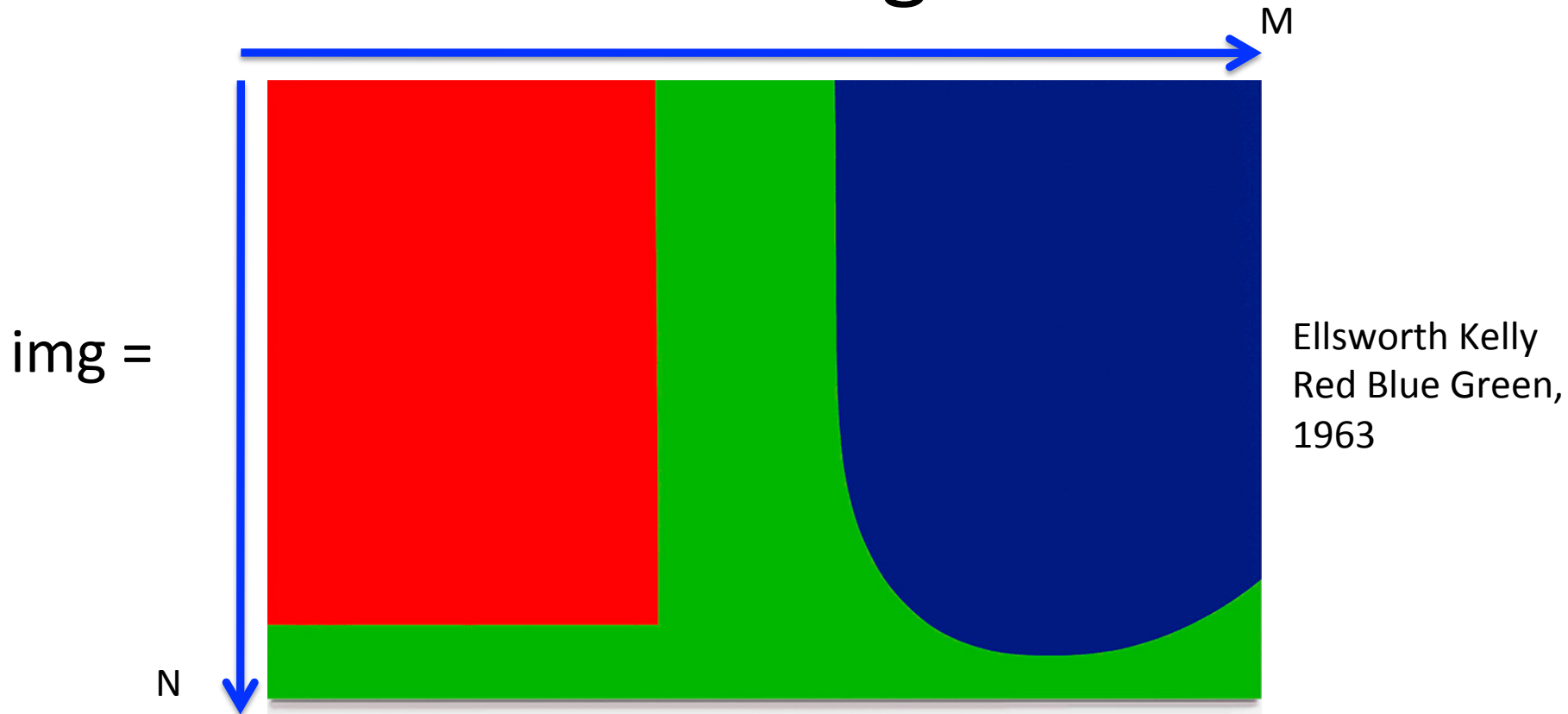
- Stored as 3d matrix of r, g, and b values at each pixel
  - Image matrix would be size NxMx3
  - R = img(:,:,1)   G = img(:,:,2) B = img(:,:,3)

Red component

img(:,:,1) =
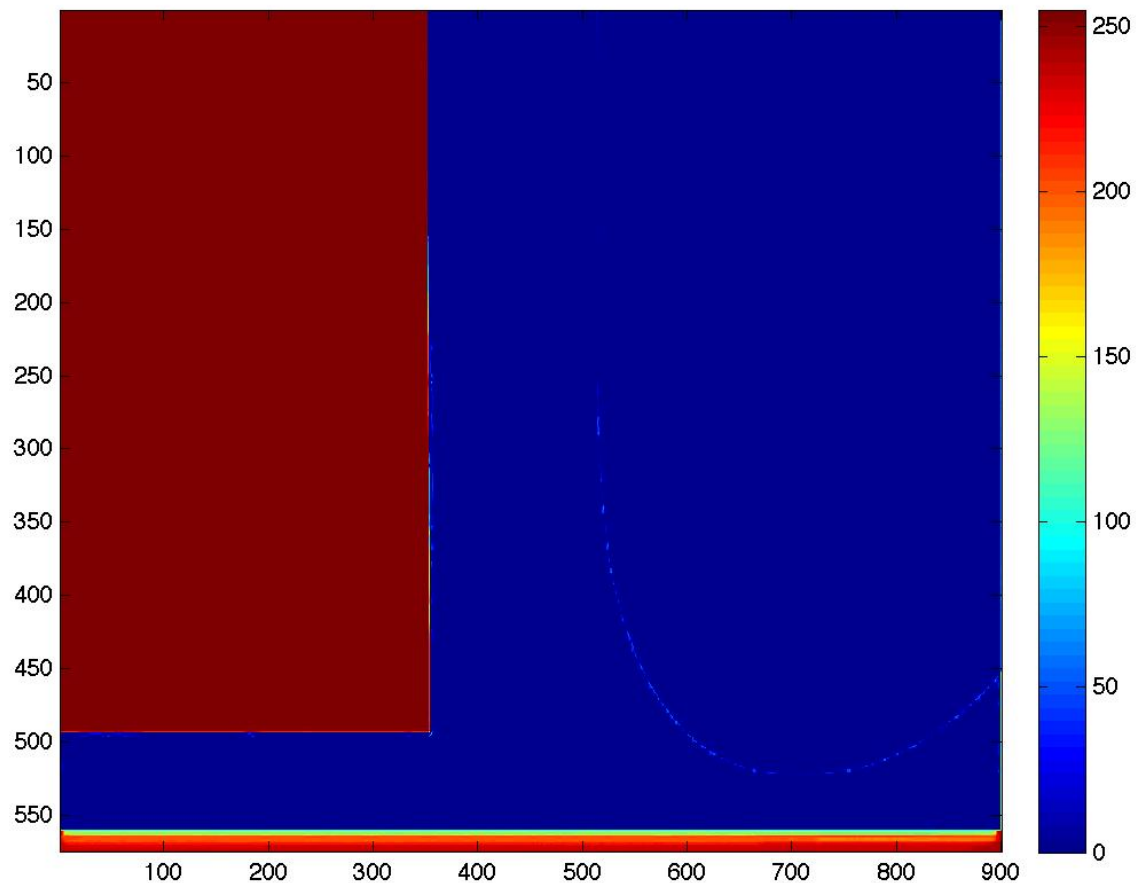
Green component

img(:,:,2) =
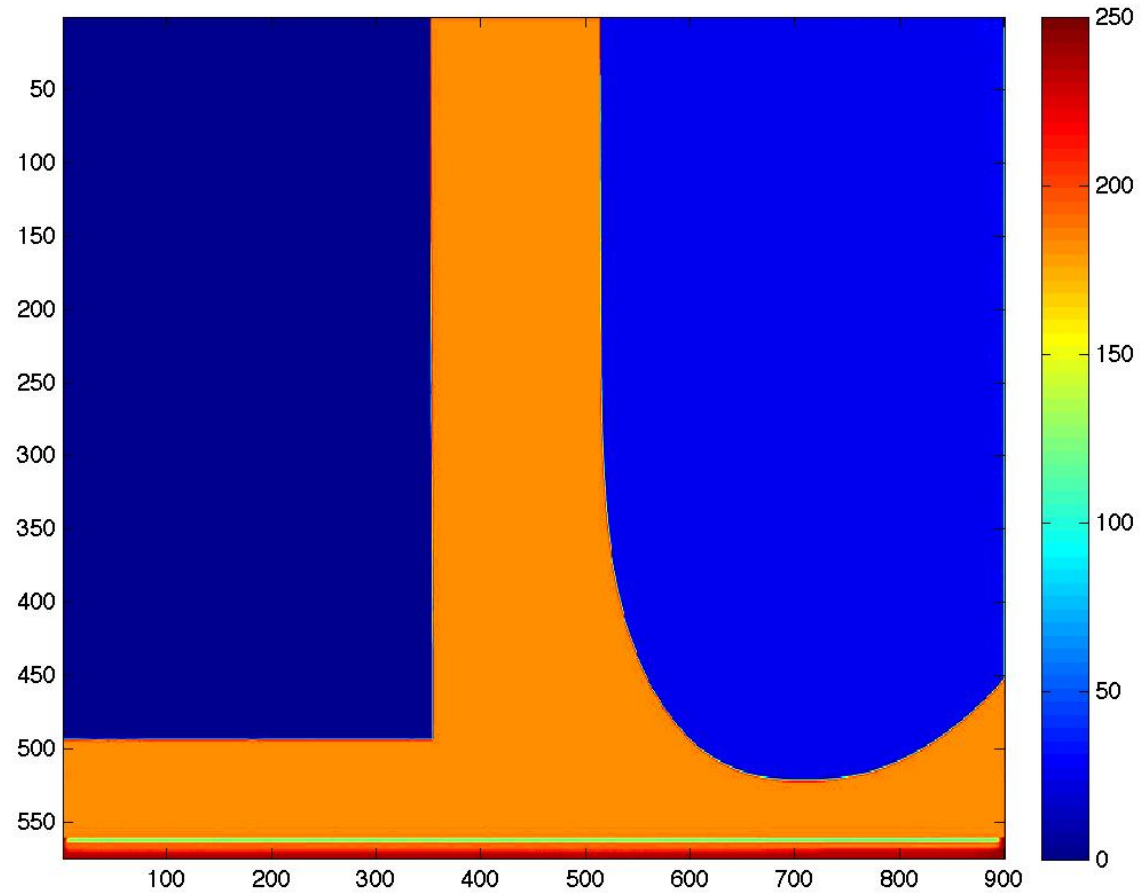
Blue component

img(:,:,3) =

Color Images

img =

M

N

Ellsworth Kelly
Red Blue Green,
1963
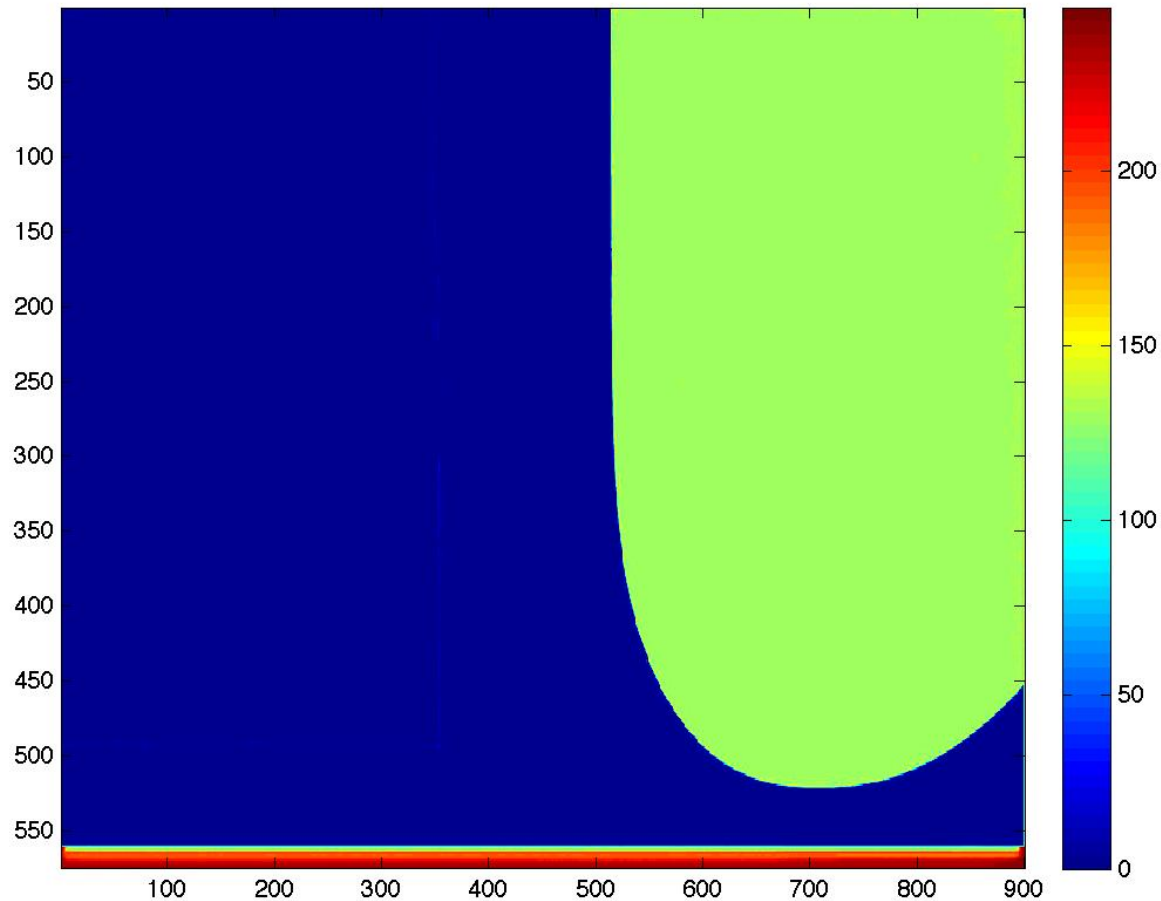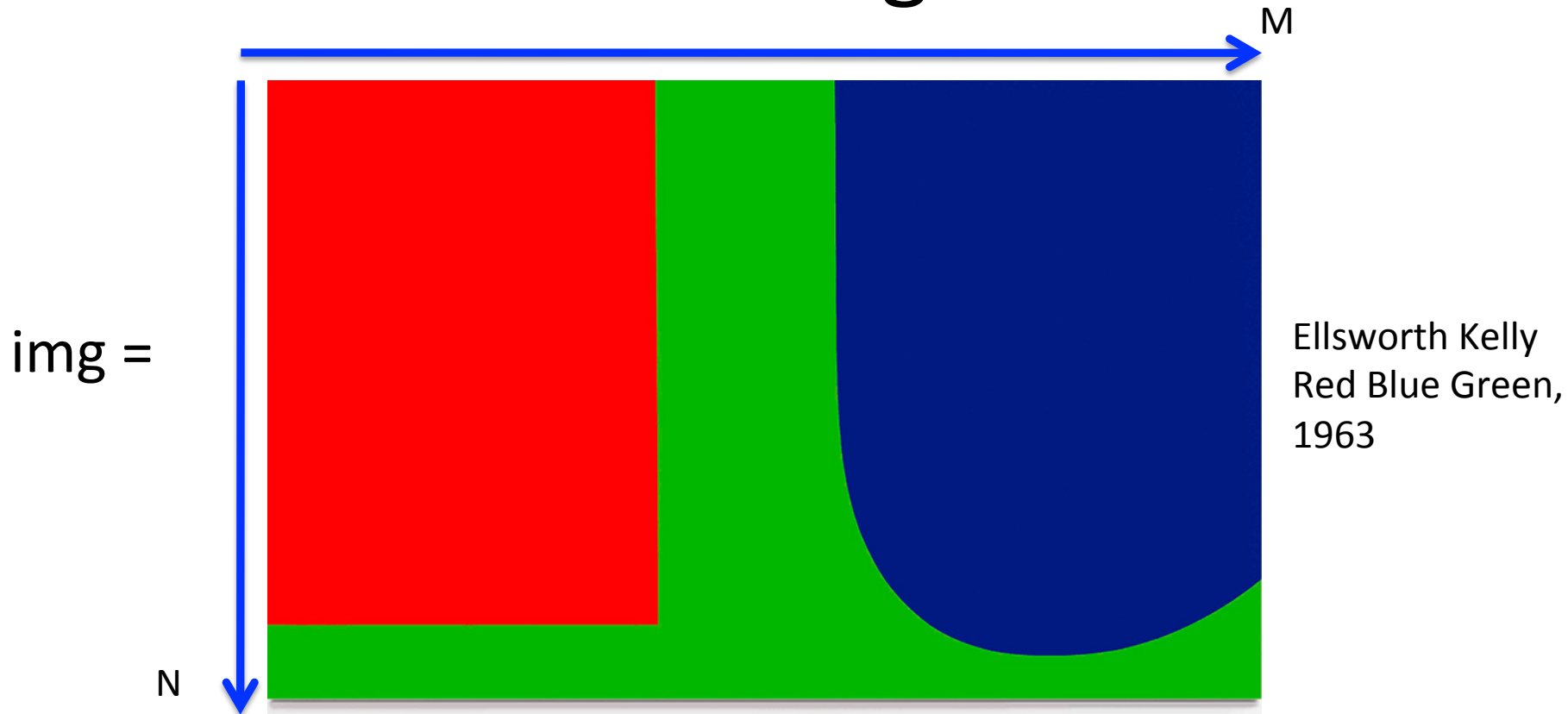
- Stored as 3d matrix of r, g, and b values at each pixel
  - So img(i,j,:) = [r g b]

Color Images

img =

Ellsworth Kelly
Red Blue Green,
1963

- Stored as 3d matrix of r, g, and b values at each pixel
  - So img(i,j,:) = [r g b].
  - In the case above this might be [255 0 0].

Useful Matlab image functions

- img = imread(filename); – read in an image
- imagesc(img); – display an image
- imwrite(img,outfilename); – write an image
- img(i,j,:) indexes into the ith row, jth column of the image.
- subimg = img(1:10,20:30,:) extracts part of img.

Matlab demo 1

This will be very useful for homework 3!

Motivation

- Image retrieval
  - We have a database of images
  - We have a query image
  - We want to find those images in our database that are most similar to the query

Motivation

- Image retrieval
  - We have a database of images
  - We have a query image
  - We want to find those images in our database that are most similar to the query

- Similarly to text retrieval, & music retrieval we first need a representation for our data.

How should we represent an image?

First try

Just represent the image by all its pixel values

First try

Img1 =

Img2 =

Say we measure similarity as:
sim = sum(abs(img1 − img2))

First try

Img1 =

Img2 =

Say we measure similarity as:
sim = average diff between
values in img1 and img2

How similar are these two images? Is this bad?

What do we want?

- Features should be robust to small changes in the image such as:
  - Translation
  - Rotation
  - Illumination changes

Second Try

Photo by: marielito

Represent the image as its average pixel color

Second Try

Photo by: marielito

Represent the image as its average pixel color
Pros?
Cons?

Third Try

Photo by: marielito
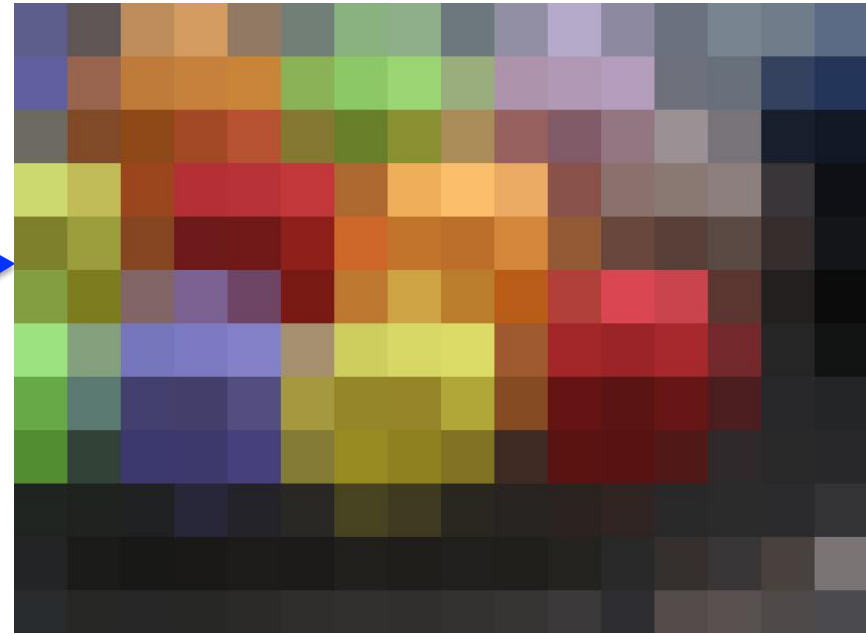
Represent the image as a spatial grid of average pixel colors
      Pros?
      Cons?

QBIC system

QBIC link

- First content based image retrieval system
  - Query by image content (QBIC)
  - IBM 1995
  - QBIC interprets the virtual canvas as a grid of coloured areas, then matches this grid to other images stored in the database.

Color is not always enough!

The representation of these two umbrella's should be similar....
Under a color based representation they look completely different!

What next?
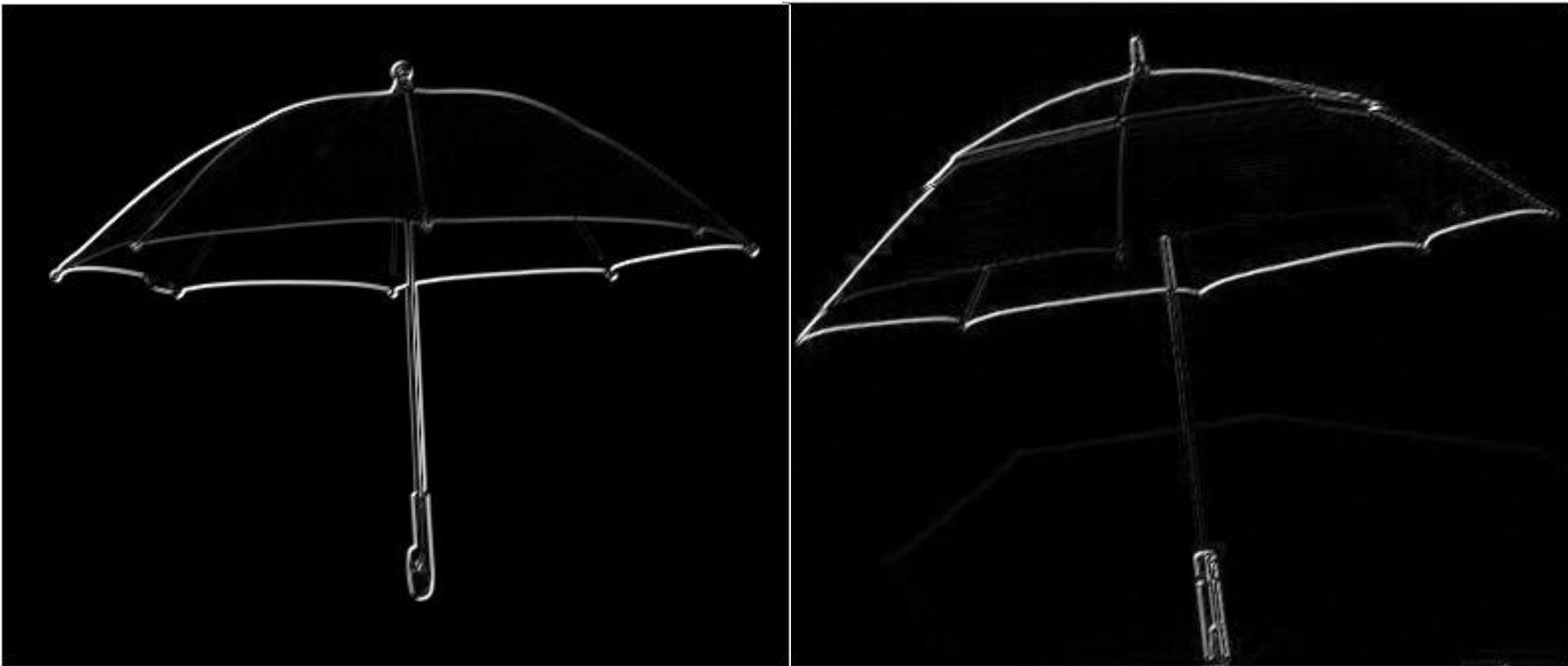
Edges! But what are they & how do we find them?

Reminder: Convolution

Filtering

Alternatively you can convolve the input signal with a filter to get frequency limited output signal.

Convolution:

$$(f * g)[n] \stackrel{\text{def}}{=} \sum_{m=-\infty}^{\infty} f[m] \cdot g[n - m]$$

(convolution [demo])

signal

f =	1	3	2	5	3	2	4	5

filter

g =	1/4	1/2	1/4

Filtering

Alternatively you can convolve the input signal with a filter to get frequency limited output signal.

Convolution:

$$(f * g)[n] \overset{\text{def}}{=} \sum_{m=-\infty}^{\infty} f[m] \cdot g[n-m]$$

(convolution [demo])

1	3	2	5	3	2	4	5
*	*	*					
1/4	1/2	1/4					

=	-	2.25						

Filtering

Alternatively you can convolve the input signal with a filter to get frequency limited output signal.

Convolution:

$$(f * g)[n] \stackrel{\text{def}}{=} \sum_{m=-\infty}^{\infty} f[m] \cdot g[n-m]$$

(convolution demo)

1	3	2	5	3	2	4	5

	*	*	*	
	1/4	1/2	1/4	

=	-	2.25	3					

Filtering

Alternatively you can convolve the input signal with a filter to get frequency limited output signal.

Convolution:

$$(f * g)[n] \stackrel{\text{def}}{=} \sum_{m=-\infty}^{\infty} f[m] \cdot g[n-m]$$

(convolution [demo])

1	3	2	5	3	2	4	5

	*	*	*				
	1/4	1/2	1/4				

=

-	2.25	3	3.75				

Filtering

Alternatively you can convolve the input signal with a filter to get frequency limited output signal.

Convolution:

$$(f * g)[n] \stackrel{\text{def}}{=} \sum_{m=-\infty}^{\infty} f[m] \cdot g[n-m]$$

(convolution [demo])

1	3	2	5	3	2	4	5
			*	*	*		
			1/4	1/2	1/4		

=	-	2.25	3	3.75	3.25			

Filtering

Alternatively you can convolve the input signal with a filter to get frequency limited output signal.

Convolution:

$$(f * g)[n] \overset{\text{def}}{=} \sum_{m=-\infty}^{\infty} f[m] \cdot g[n-m]$$

(convolution demo)

1	3	2	5	3	2	4	5
				*	*	*	
				1/4	1/2	1/4	

=	-	2.25	3	3.75	3.25	2.75		

Filtering

Alternatively you can convolve the input signal with a filter to get frequency limited output signal.

Convolution:

$$(f * g)[n] \overset{\text{def}}{=} \sum_{m=-\infty}^{\infty} f[m] \cdot g[n-m]$$

(convolution demo)

1	3	2	5	3	2	4	5

Convolution computes a weighted average.

					*	*	*
					1/4	1/2	1/4

=

-	2.25	3	3.75	3.25	2.75	3.75	-

Images -> 2d filtering
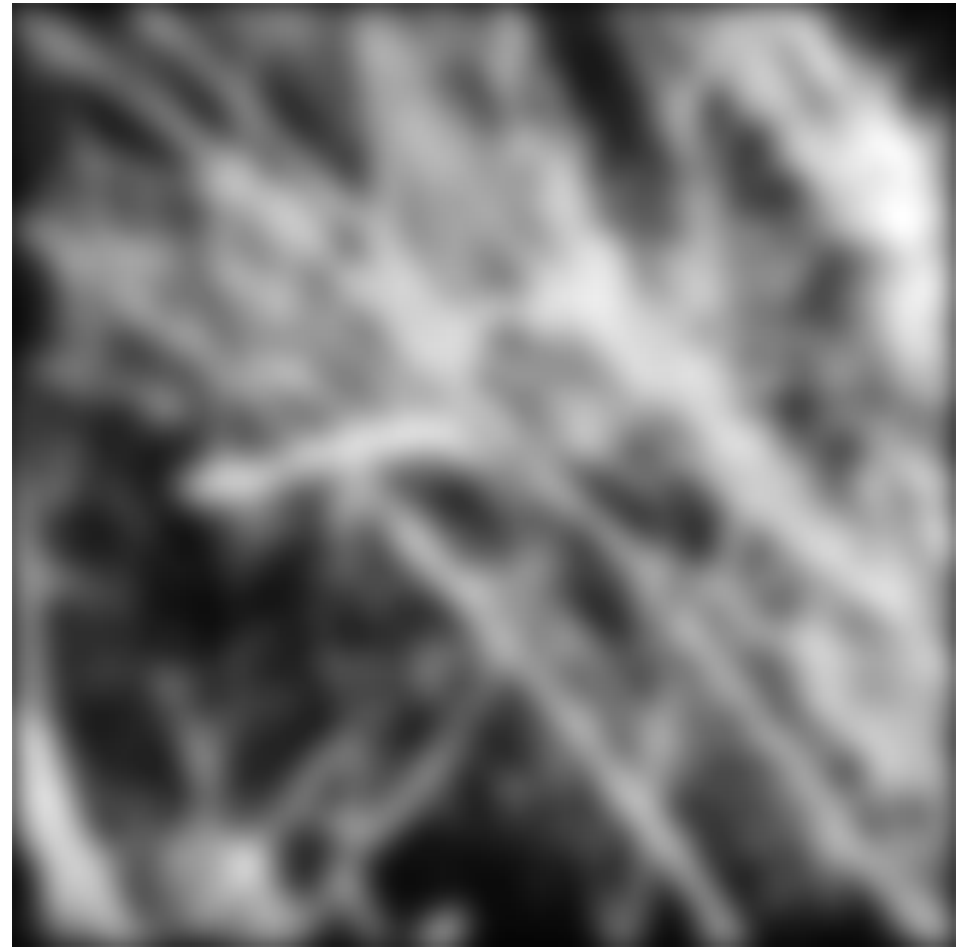
Moving average

- Let's replace each pixel with a *weighted* average of its neighborhood

- The weights are called the *filter kernel*

- What are the weights for a 3x3 moving average?

Moving average

- Let's replace each pixel with a *weighted* average of its neighborhood

- The weights are called the *filter kernel*

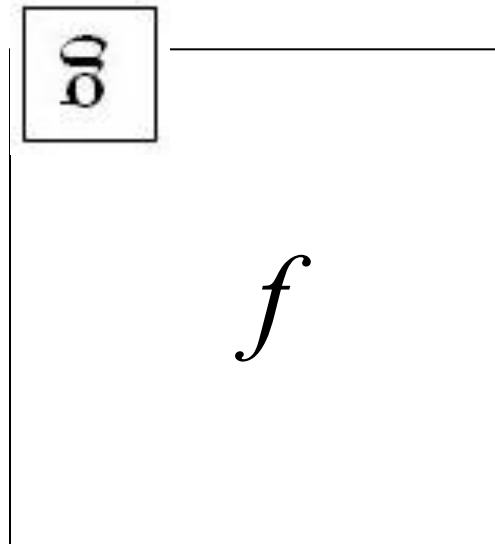- What are the weights for a 3x3 moving average?

$$\frac{1}{9}$$

1	1	1
1	1	1
1	1	1

"box filter"

Defining convolution in 2d

- Let *f* be the image and *g* be the kernel. The output of convolving *f* with *g* is denoted *f * g*.

$$(f * g)[m,n] = \sum_{k,l} f[m-k, n-l]\, g[k,l]$$

- Convention: kernel is "flipped"
- MATLAB: conv2 vs. filter2 (also imfilter)

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

g(x,y)

$$\frac{1}{9}$$

1	1	1
1	1	1
1	1	1

"box filter"

Moving Average In 2D

$$F[x, y]$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$$G[x, y]$$

	0								

Moving Average In 2D

$$F[x, y]$$

$$G[x, y]$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

G values: 0, 10

Moving Average In 2D

$$F[x, y]$$

$$G[x, y]$$

Moving Average In 2D

$$F[x, y]$$

$$G[x, y]$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

	0	10	20	30					

Moving Average In 2D

$$F[x, y]$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$$G[x, y]$$

	0	10	20	30	30				

Moving Average In 2D

$$F[x, y] \qquad G[x, y]$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

What is this filter doing?

Practice with linear filters
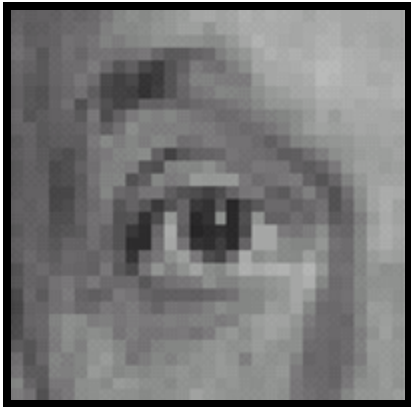
Original

0	0	0
0	1	0
0	0	0

?

Practice with linear filters
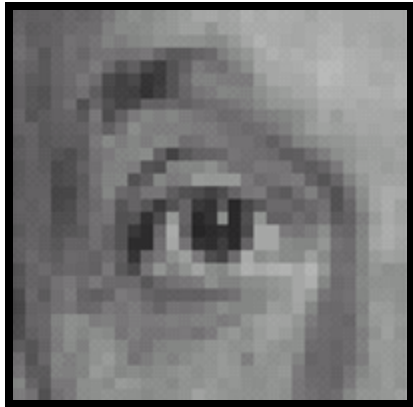
Original

0	0	0
0	1	0
0	0	0
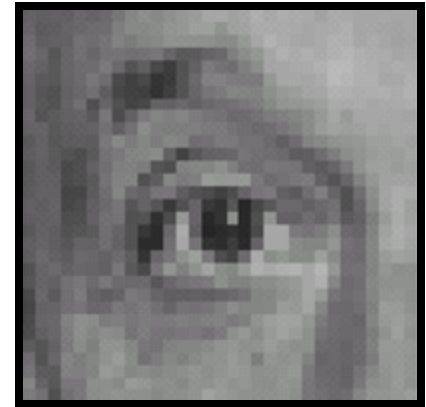
Filtered
(no change)

Source: D. Lowe

Practice with linear filters
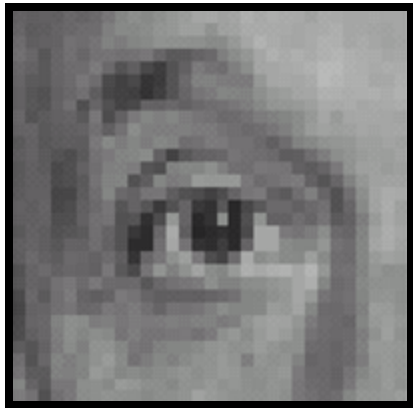
Original

0	0	0
0	0	1
0	0	0

?

Practice with linear filters
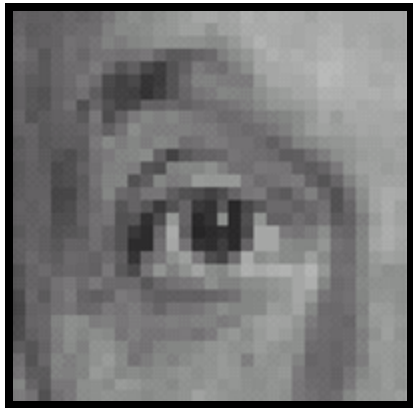
Original

0	0	0
0	0	1
0	0	0
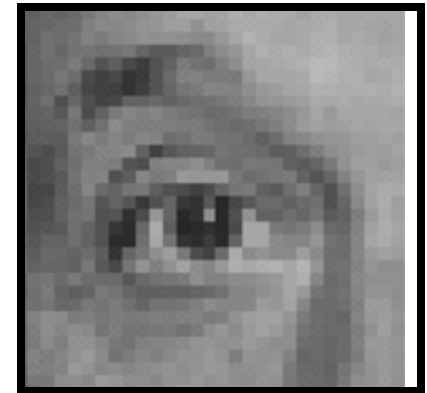
Shifted left
By 1 pixel

Practice with linear filters

Original

$$\frac{1}{9}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\\hline 1 & 1 & 1 \\\hline 1 & 1 & 1 \\\hline\end{array}$$

?
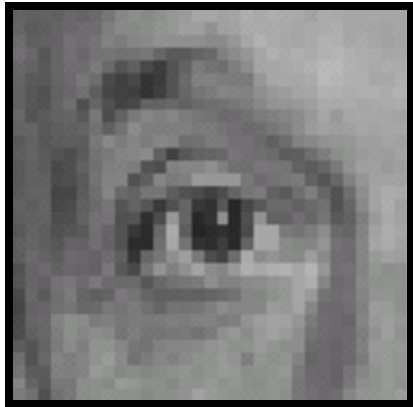
Practice with linear filters

Original

$$\frac{1}{9}$$

1	1	1
1	1	1
1	1	1
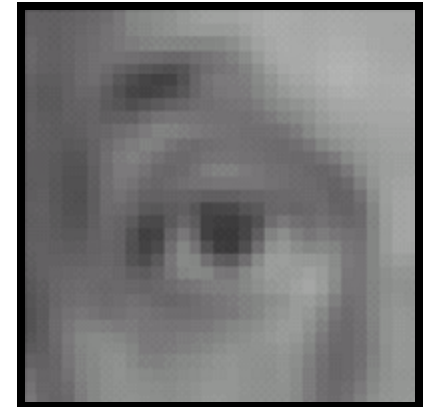
Blur (with a
box filter)

Gaussian Kernel

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$
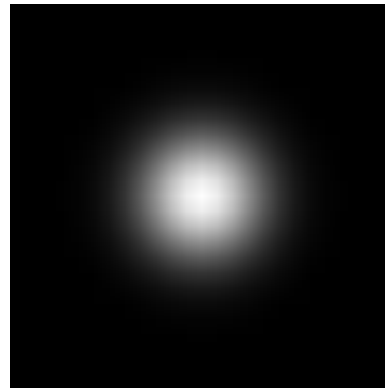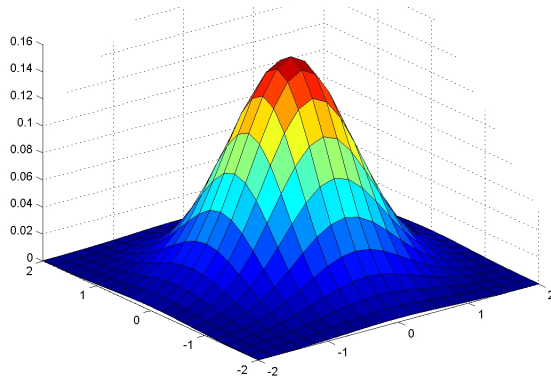
0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

5 x 5, σ = 1

- Constant factor at front makes volume sum to 1 (can be ignored, as we should re-normalize weights to sum to 1 in any case)

Example: Smoothing with a Gaussian
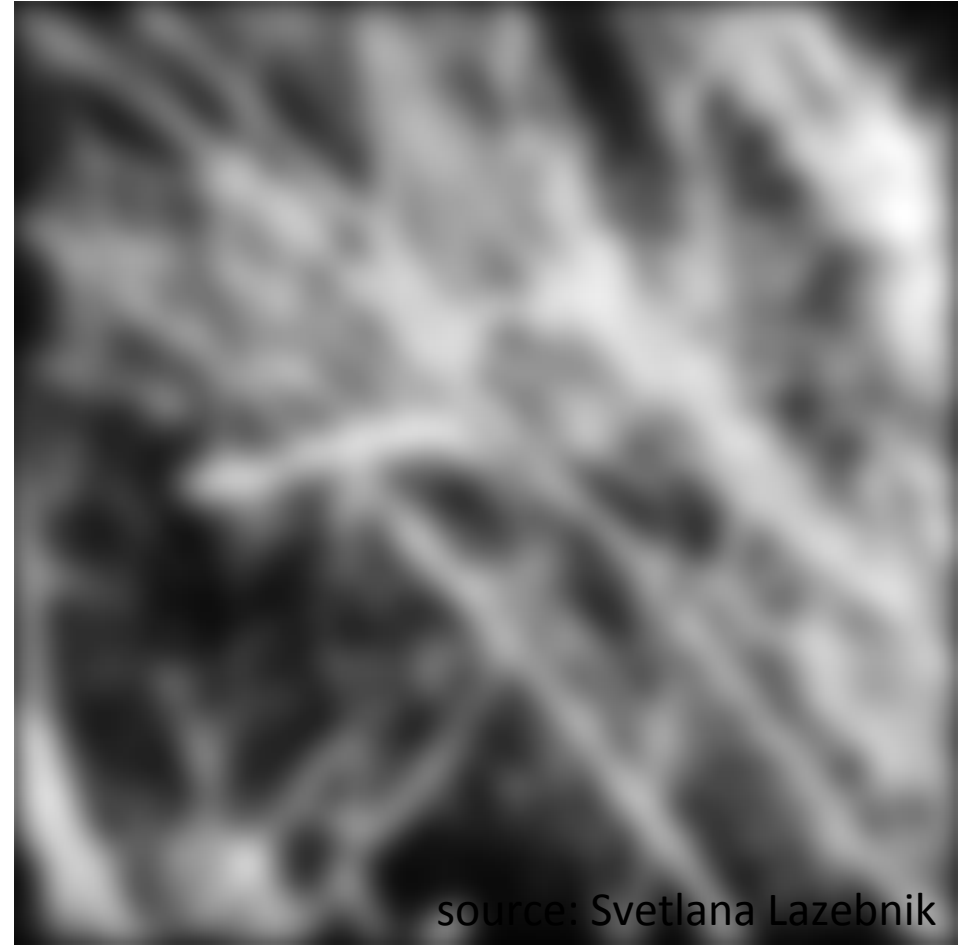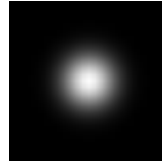
source: Svetlana Lazebnik

Edges

Edge detection

- **Goal:** Identify sudden changes (discontinuities) in an image
  - Intuitively, most semantic and shape information from the image can be encoded in the edges
  - More compact than pixels

- **Ideal:** artist's line drawing (but artist is also using object-level knowledge)

Source: D. Lowe

Origin of Edges

surface normal discontinuity

depth discontinuity

surface color discontinuity

illumination discontinuity
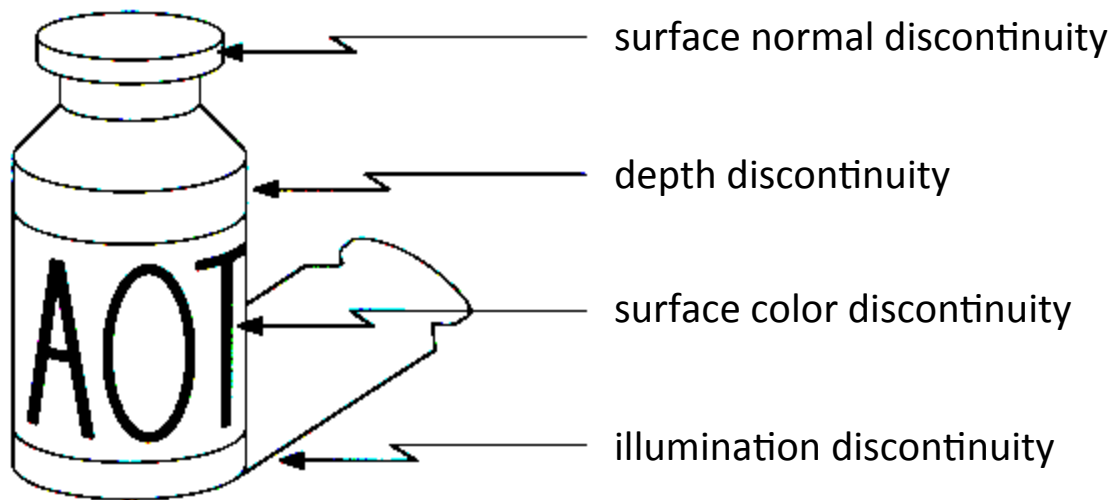
Edges are caused by a variety of factors

Characterizing edges

- An edge is a place of rapid change in the image intensity function

image

intensity function
(along horizontal scanline)

first derivative
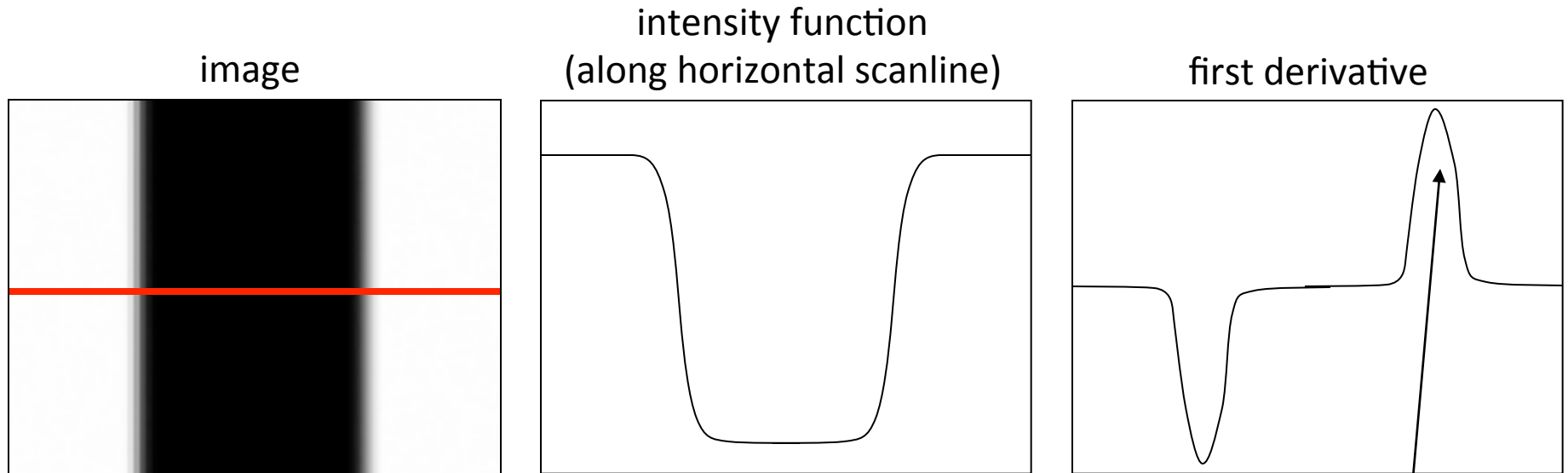
edges correspond to
extrema of derivative

Edge filters

Approximations of derivative filters:

$$\text{Prewitt:} \quad M_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad ; \quad M_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

$$\text{Sobel:} \quad M_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad ; \quad M_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

$$\text{Roberts:} \quad M_x = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \quad ; \quad M_y = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Convolve filter with image to get edge map

Edge filters

Approximations of derivative filters:

Prewitt:
$$M_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad ; \quad M_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

Sobel:
$$M_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad ; \quad M_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Roberts:
$$M_x = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \quad ; \quad M_y = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Respond highly to vertical edges

Source: K. Grauman

Edge filters

Approximations of derivative filters:

Prewitt: $M_x = \begin{array}{|c|c|c|} \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$; $M_y = \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -1 & -1 \\ \hline \end{array}$

Sobel: $M_x = \begin{array}{|c|c|c|} \hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$; $M_y = \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array}$

Roberts: $M_x = \begin{array}{|c|c|} \hline 0 & 1 \\ \hline -1 & 0 \\ \hline \end{array}$; $M_y = \begin{array}{|c|c|} \hline 1 & 0 \\ \hline 0 & -1 \\ \hline \end{array}$

Respond highly to horizontal edges

Edges: example
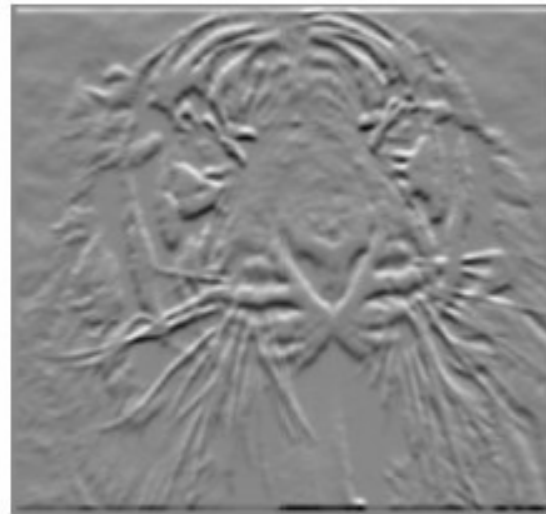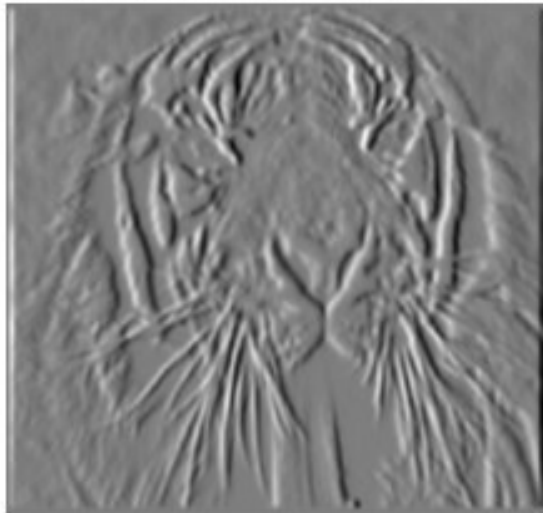
What about our umbrellas?

The representation of these two umbrella's should be similar....
Under a color based representation they look completely different!
How about using edges?

Edges

Red umbrella

Gray umbrella

Edges extracted using convolution with Prewitt filter

Edges

Edges overlayed from red and gray umbrellas.

How is this?

MatLab

- Imfilter
- filter

# Edge Energy in Spatial Grid
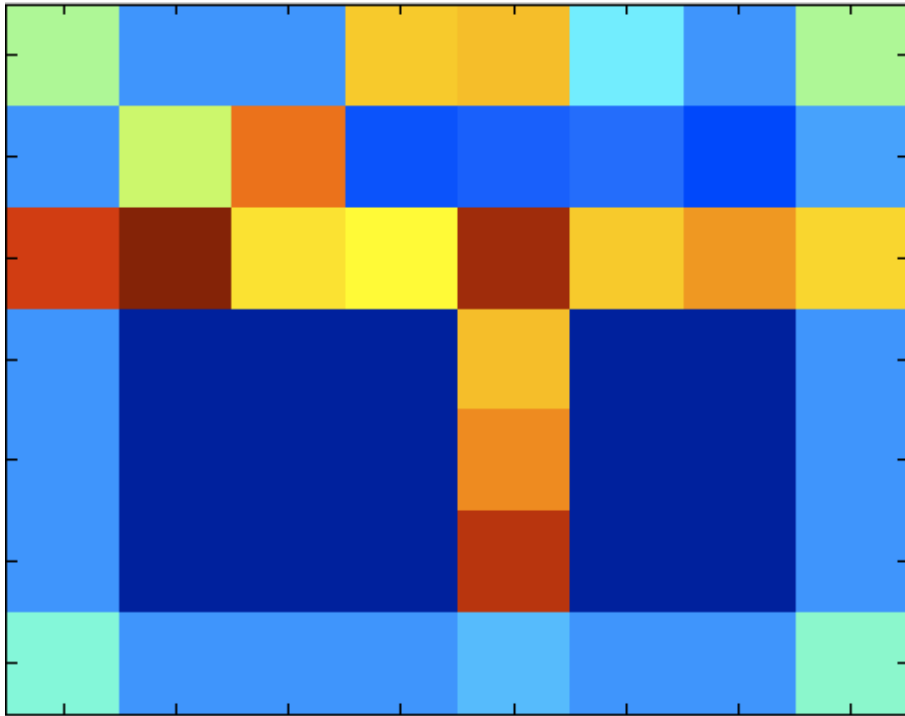
Red Umbrella
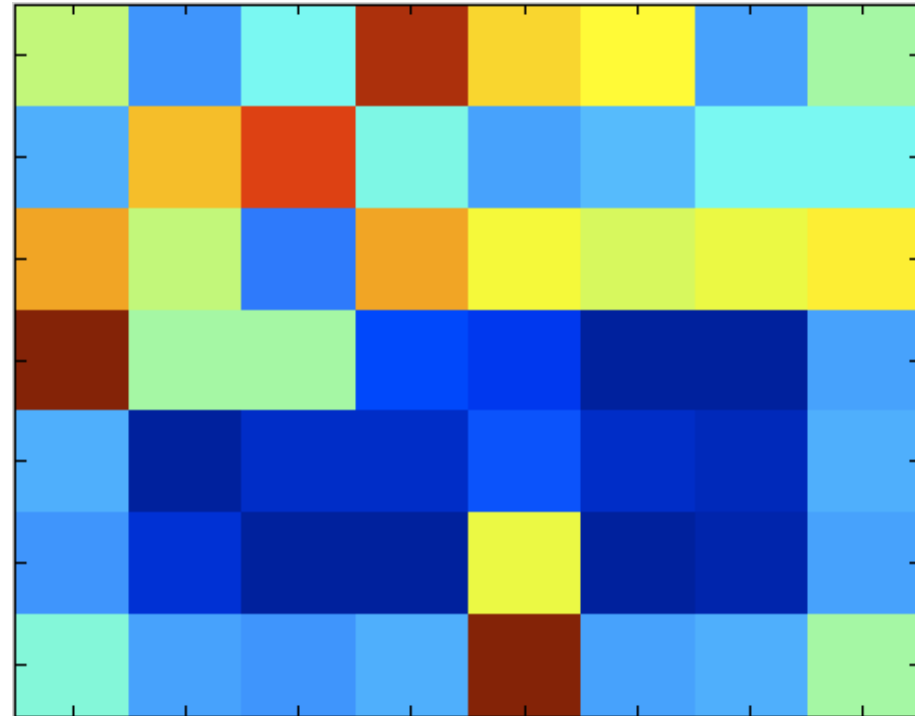
Gray Umbrella

How is this representation?

Quick overview of other common kinds of Features

Important concept: Histograms

Heights of Black Cherry Trees
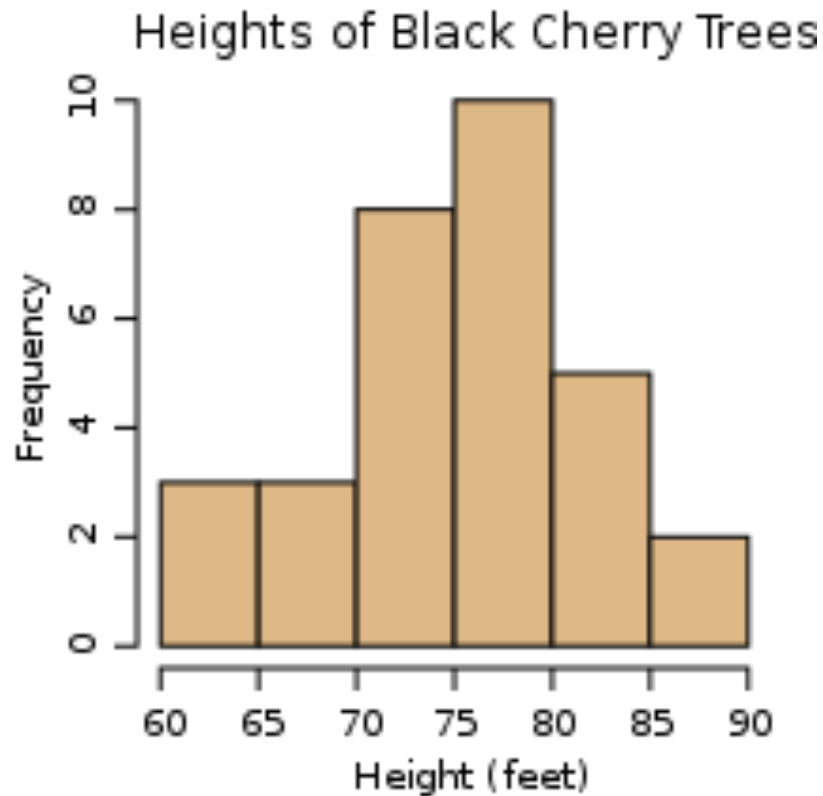
Graphical display of tabulated frequencies, shown as bars. It shows what proportion of cases fall into each of several categories. The categories are usually specified as non-overlapping intervals of some variable.

Color Histograms
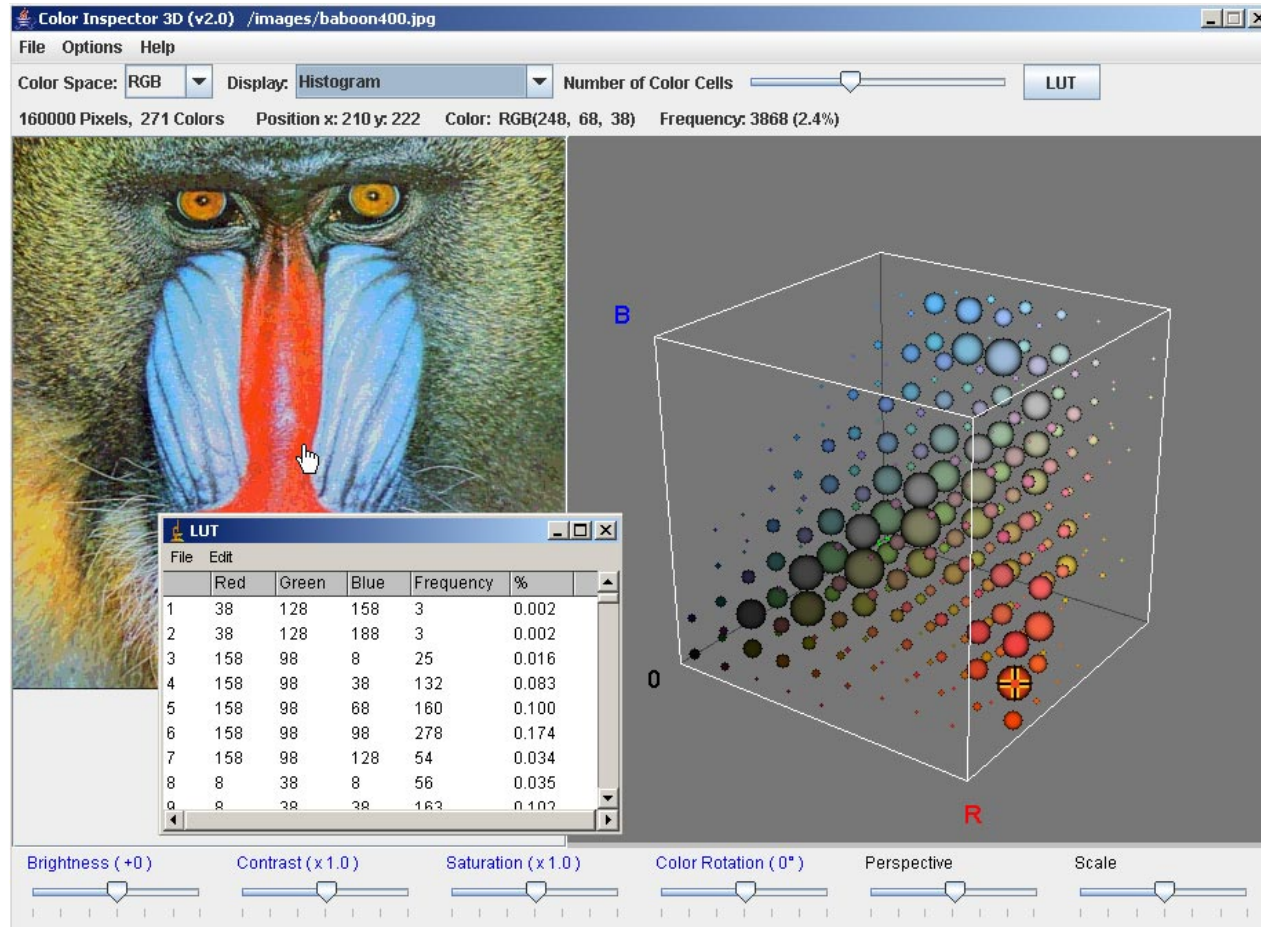
Representation of the distribution of colors in an image, derived by counting the number of pixels of each of given set of color ranges in a typically (3D) color space (RGB, HSV etc).
What are the bins in this histogram?

Image Shapes

- Determined only by outer boundary of an object

- Rigid Objects
  - Transformations: Translations, Rotations, uniform scaling (size changes)
  - Mirror image can be a different image
  - Congruent objects

- Non-rigid Objects (deformable shapes)
  - Ex: a person on different positions

Shape Descriptors: shape context
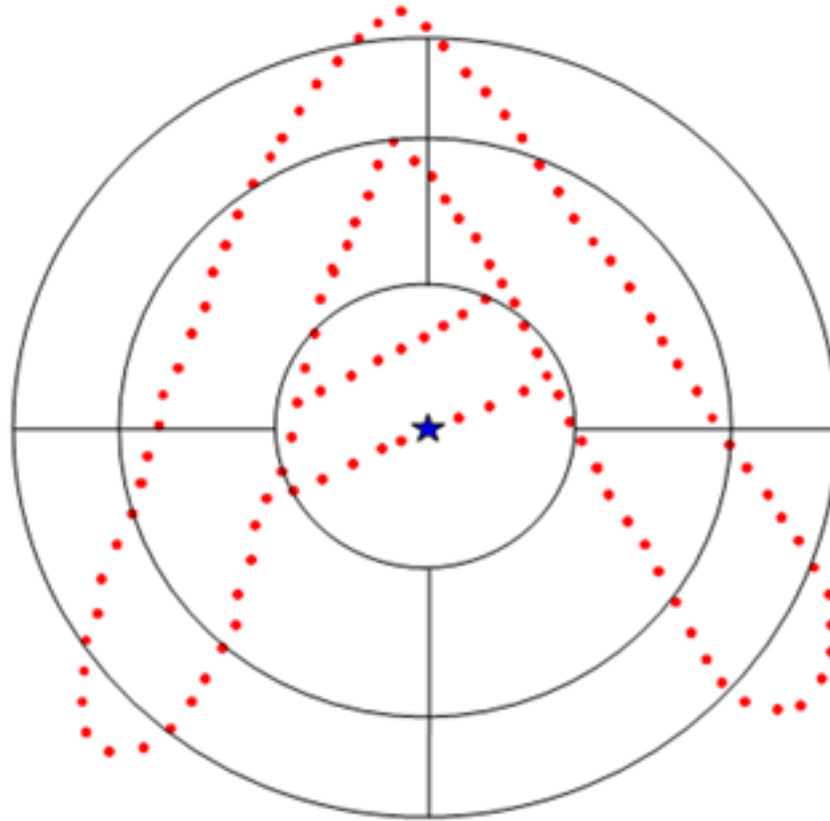
Representation of the local shape around a feature location (star) – as histogram of edge points in an image relative to that location. Computed by counting the edge points in a log polar space.

So what are the bins of this histogram?

Shape descriptors: SIFT

- Descriptor computation:
  - Divide patch into 4x4 sub-patches
  - Compute histogram of gradient orientations (convolve with filters that respond to edges in different directions) inside each subpatch
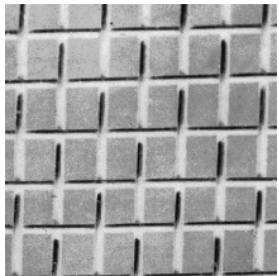 - Resulting descriptor: 4x4x8 = 128 dimensions
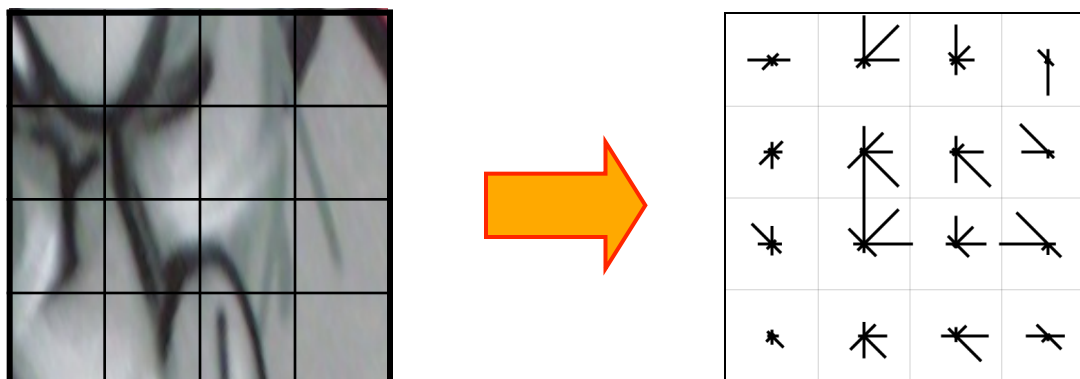
David G. Lowe. "Distinctive image features from scale-invariant keypoints." *IJCV* 60 (2), pp. 91-110, 2004.
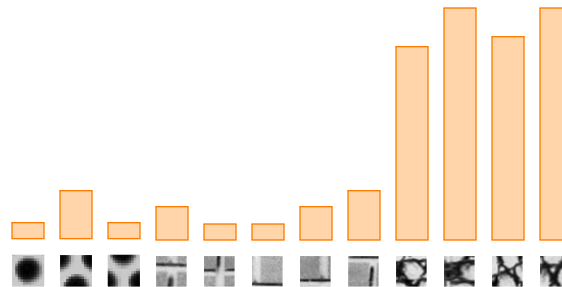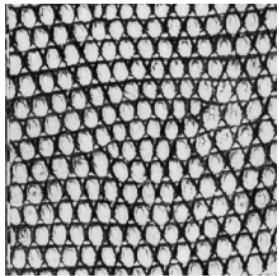
source: Svetlana Lazebnik

Texture Features
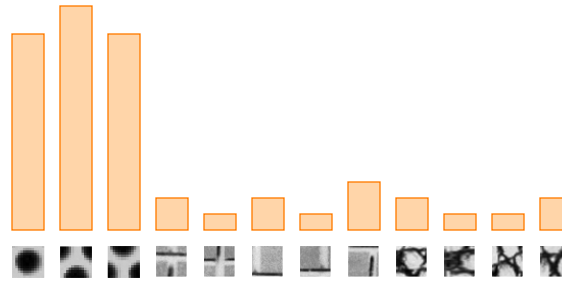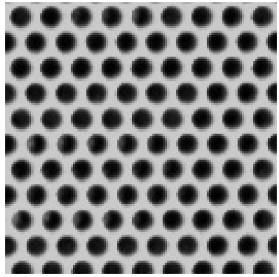
Convolve with various filters – spot, oriented bar.
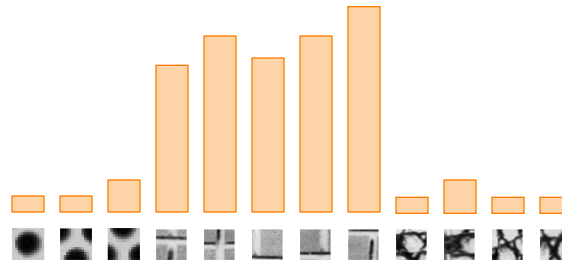
Compute histogram of responses.

Julesz, 1981; Cula & Dana, 2001; Leung & Malik 2001; Mori, Belongie & Malik, 2001; Schmid 2001; Varma & Zisserman, 2002, 2003; Lazebnik, Schmid & Ponce, 2003