Computer Vision 1

THEO GEVERS, SHAODI YOU, THOMAS MENSINK AND PASCAL METTES

MASTER AI

UNIVERSITY OF AMSTERDAM

Disclaimer

- Many of the slides used here are obtained from online resources (including many open lecture materials) without appropriate acknowledgement. They are used here for the sole purpose of classroom teaching. All the credit and all the copyrights belong to the original authors. You should not copy it, redistribute it, put it online, or use it for any other purposes than for this course.

Lectures/Theory

- 02-09-2019, 17:00-19:00, H0.08, **Introduction** (Szeliski 1) – **Theo Gevers**
- 09-09-2019, 17:00-19:00, H0.08, **Image Formation** (Szeliski: 2.1.1 + 2.1.2 + 2.1.5 +
- 2.2 + 2.3.2 + 2.3.3) – **Theo Gevers/Shaodi You**
- 16-09-2019, 17:00-19:00, H0.08, **Image Processing** (Szeliski: 3.1 + 3.2 + 3.3 + 4 +
- 8.4) – **Shaodi You**
- 23-09-2019, 17:00-19:00, H0.08, **Object Recognition** (Szeliski: 14; Bengio 5.7) – **Shaodi You**
- 30-09-2019, 17:00-19:00, H0.08, **Deep Learning** (Szeliski: 3.2; Bengio: 6 + 7.4 + 7.7 +
- 7.9 + 9) – **Thomas Mensink**
- 07-10-2019, 17:00-19:00, H0.08, **Deep Video** (Bengio: 10.1 + 10.2 (intro, .2, .3) + 10.3 + 10.4 + 10.5 + 10.7 + 10.10) – **Pascal Mettes**
- 14-10-2019, 17:00-19:00, H0.08, **Applications** (Szeliski: 12.6.2 + 12.6.3 + 12.2.4; Bengio: 14.1+ 14.2+14.3+14.6+15.1+ 15.2)
- 25-10-2019, 13:00-16:00, **Written Exam**

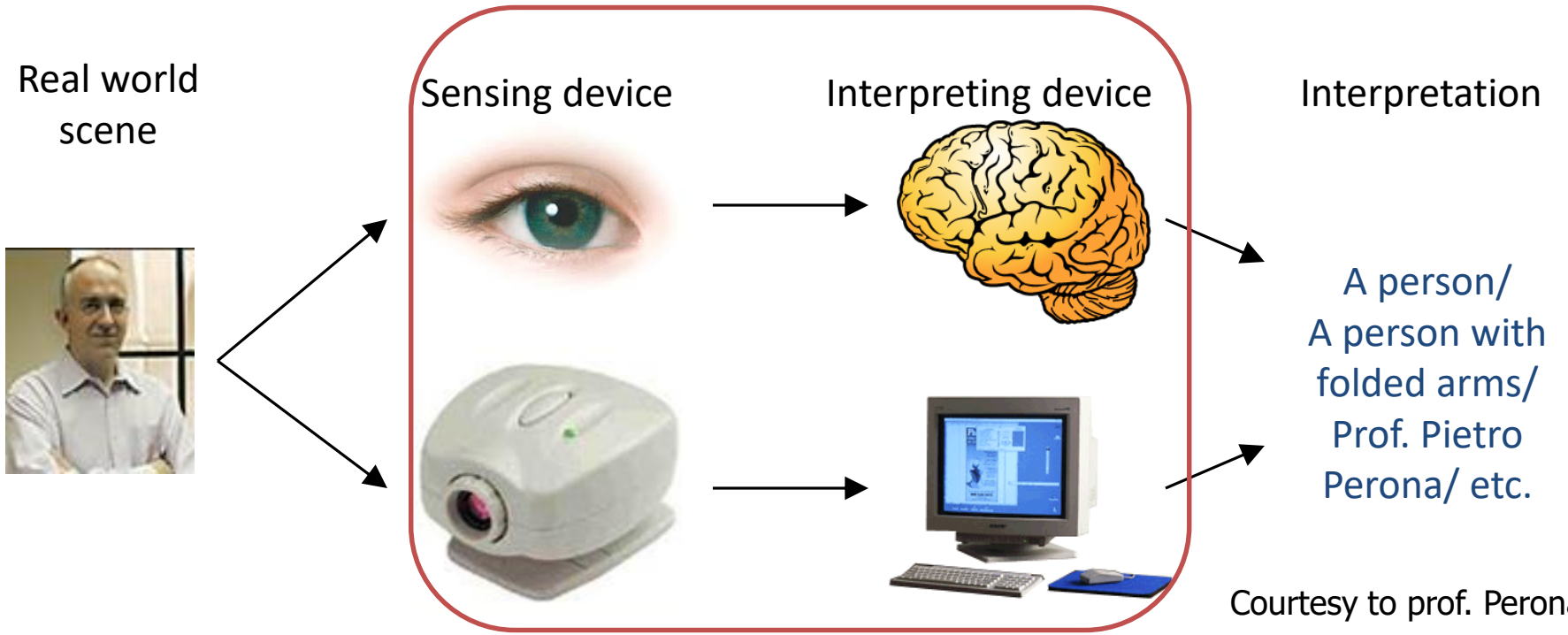
Basic Math For The Course

- Calculus / Mathematical Analysis
  - Differentiation and integration
  - Correlation and convolution
  - Surface gradient, normal, inner and outer product, principal curvature, quadratic form
  - Fourier analysis (optional)
- Algebra
  - Matrix operation
  - Linear function
  - Eigen value, eigen vector, determination, trace

The computer vision problem

- Make a computer to see and to understand images

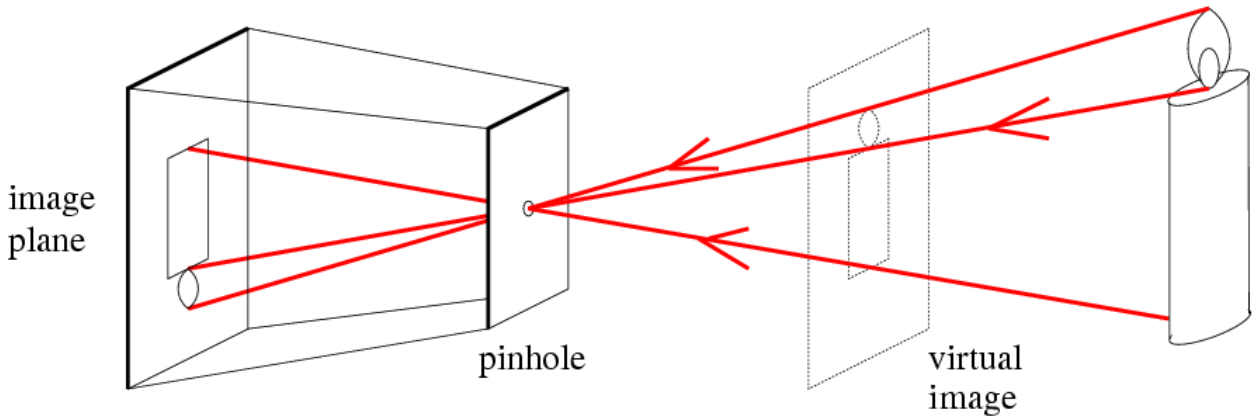
- We know it is physically possible – we do it every day and effortlessly!



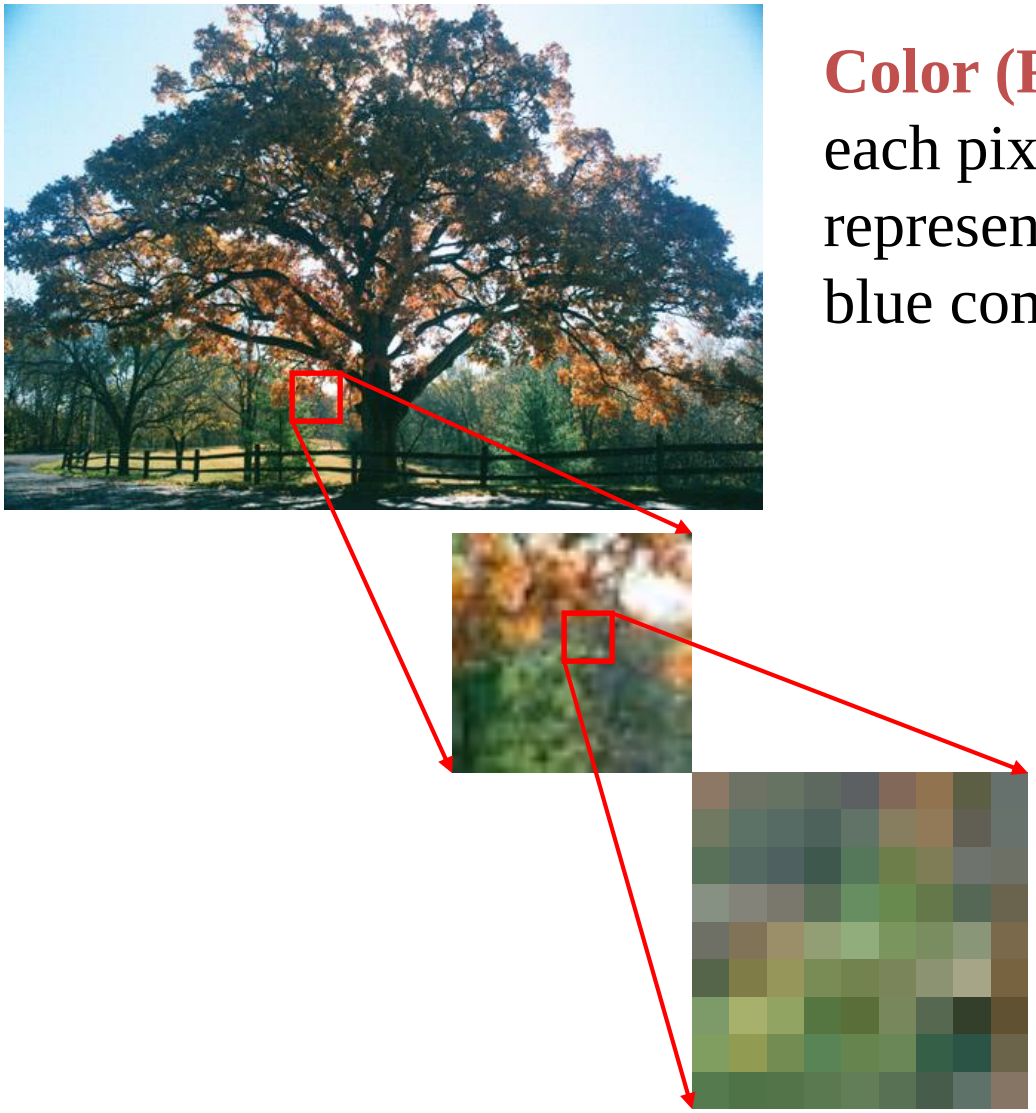
Courtesy to prof. Perona

Last week

- Pinhole camera model and camera matrix



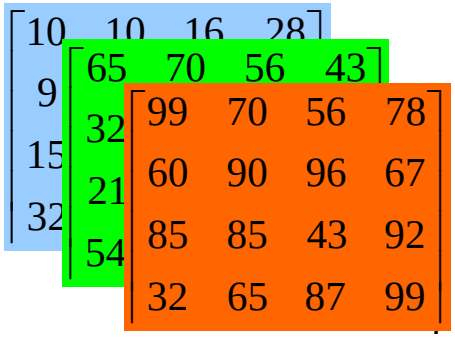
Colour Image



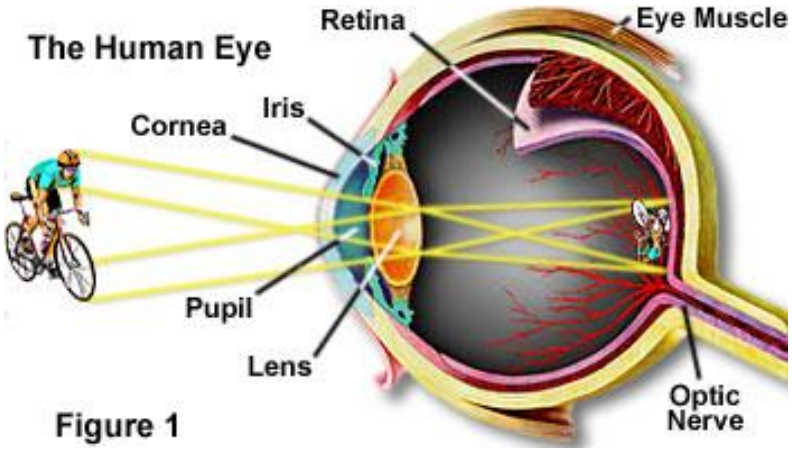
Color (RGB) image:
each pixel contains a vector representing red, green and blue components.

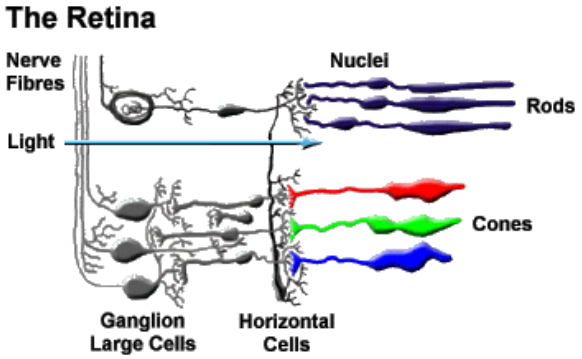
a multidimensional array of numbers (such as intensity image) or vectors (such as color image)

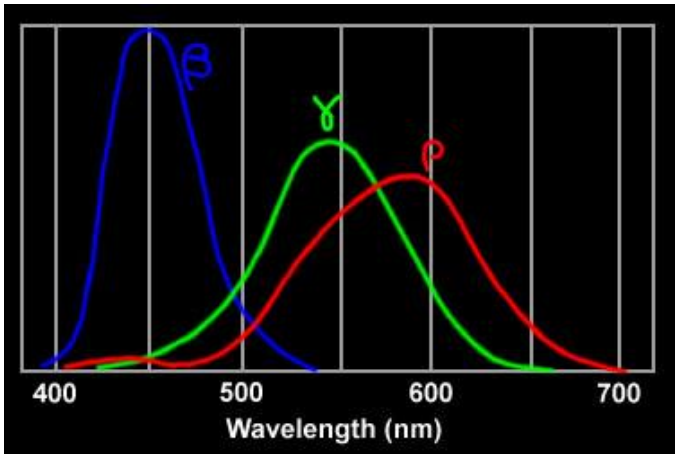
RGB components



Human Color Perception







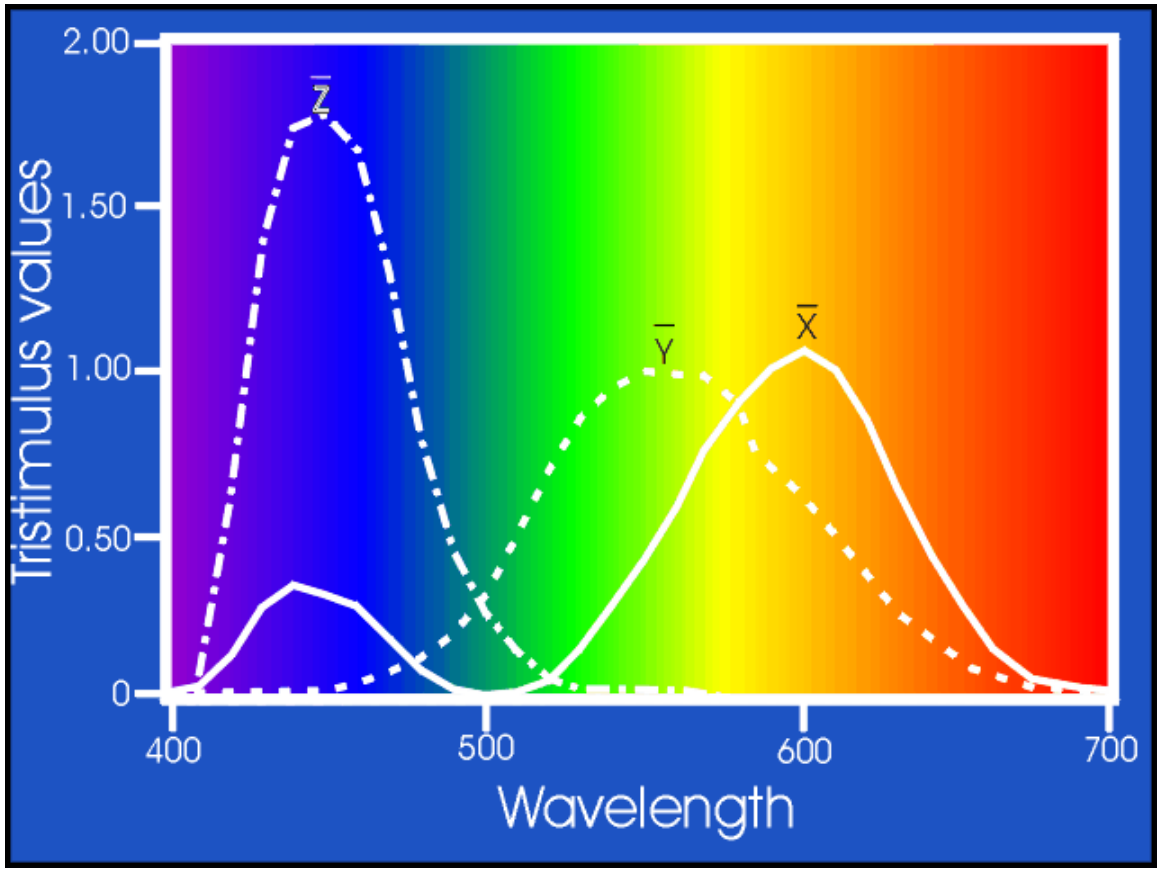
- A site about human color perception:
  - http://www.photo.net/photo/edscott/vis00010.htm

Colorimetry: CIE XYZ-system

$$X = \int_{\lambda} e(\lambda)\rho(\lambda)\bar{x}(\lambda)d\lambda$$

$$Y = \int_{\lambda} e(\lambda)\rho(\lambda)\bar{y}(\lambda)d\lambda$$

$$Z = \int_{\lambda} e(\lambda)\rho(\lambda)\bar{z}(\lambda)d\lambda$$

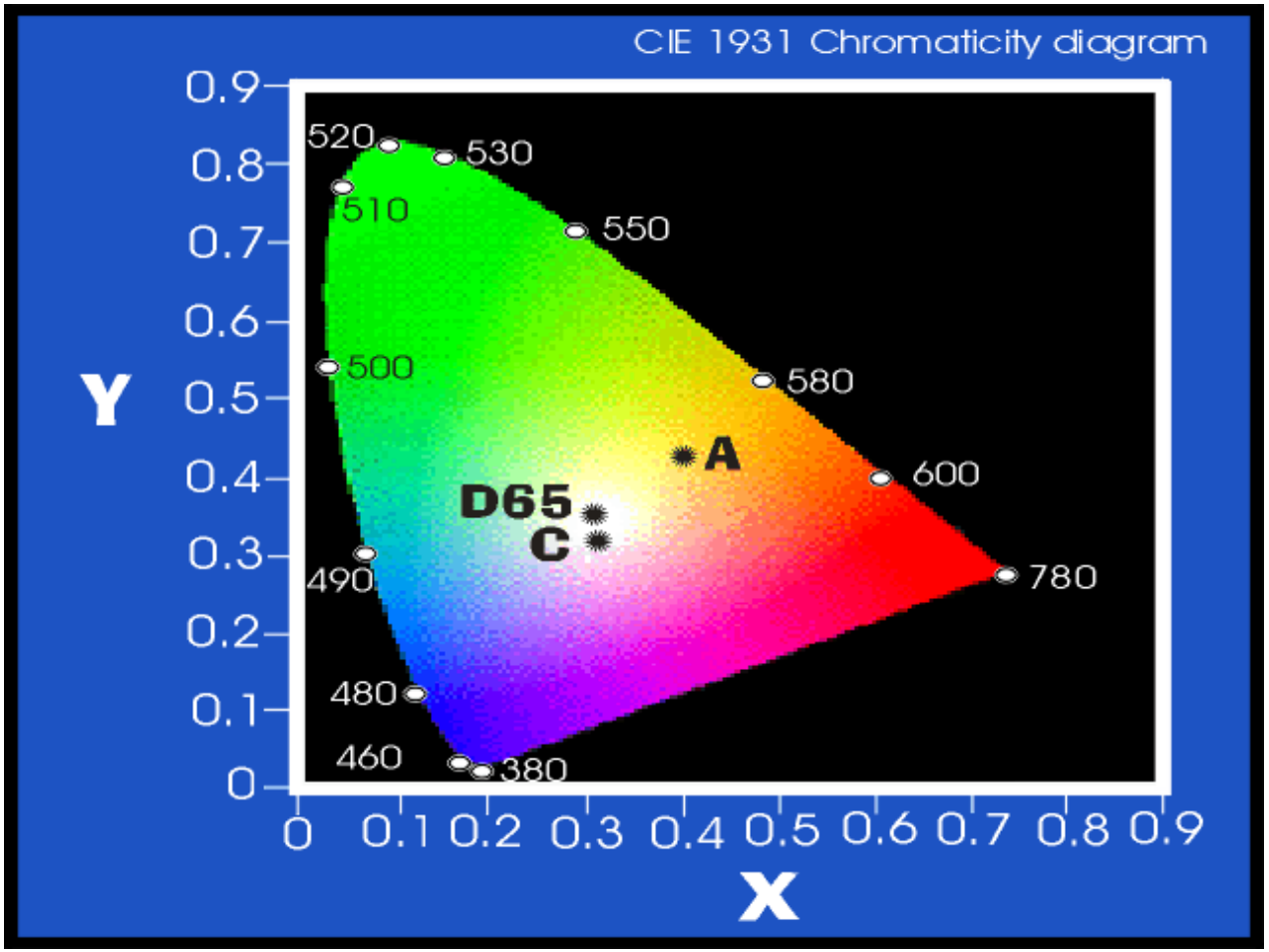


Recap: Colorimetry: Illuminants in the xy-plane

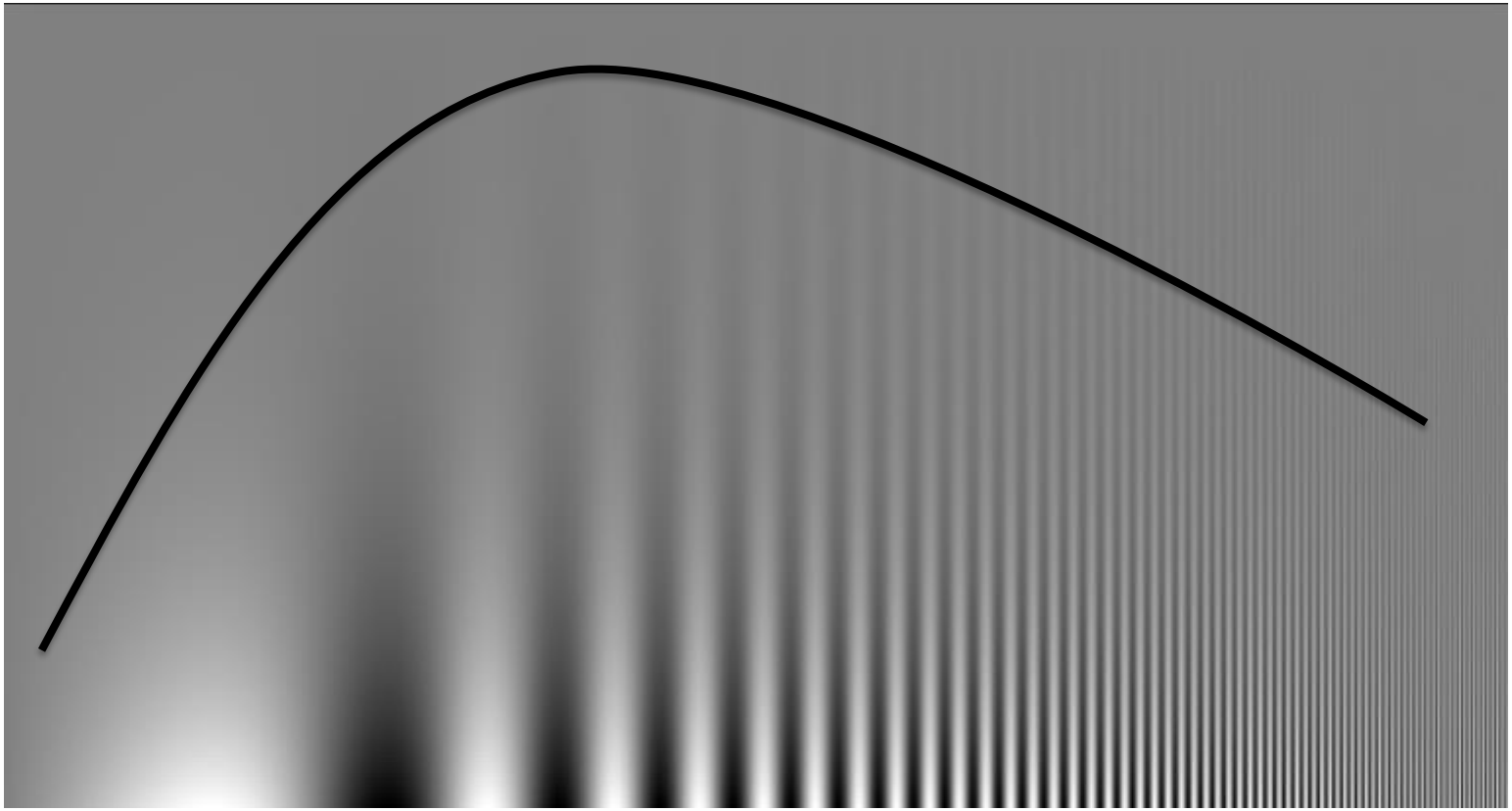
$$x = \frac{X}{X+Y+Z}$$

$$y = \frac{Y}{X+Y+Z}$$

$$z = \frac{Z}{X+Y+Z}$$



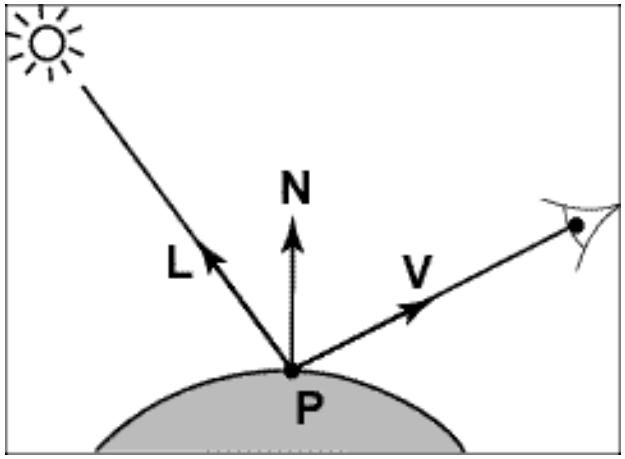
Robson-Campbell Curve



Today's Class

1. Reflection Models
2. Image Processing
   - Pixel processing
     - Grey image processing
     - Colour correction
   - Image Filtering
   - Edge Detection
   - Corner detection
     - Harris corner detector
     - SIFT
   - Applications

Modeling Image Formation

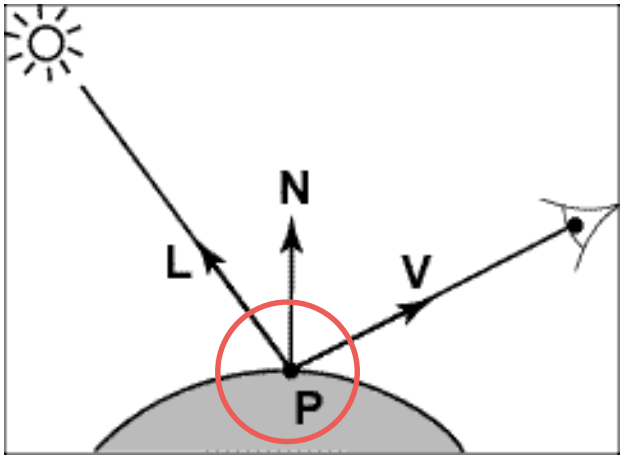


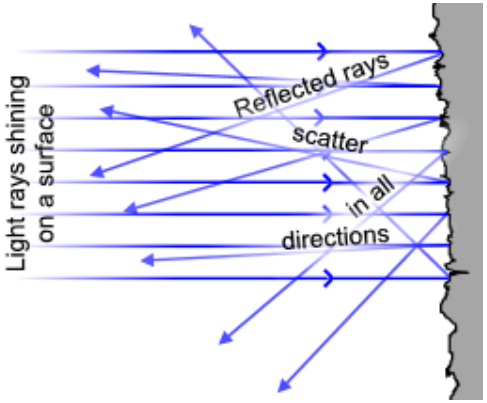
Now we need to reason about:

- How light interacts with the scene
- How a pixel value is related to light energy in the world

Track a "ray" of light all the way from light source to the sensor

Lambertian Reflectance



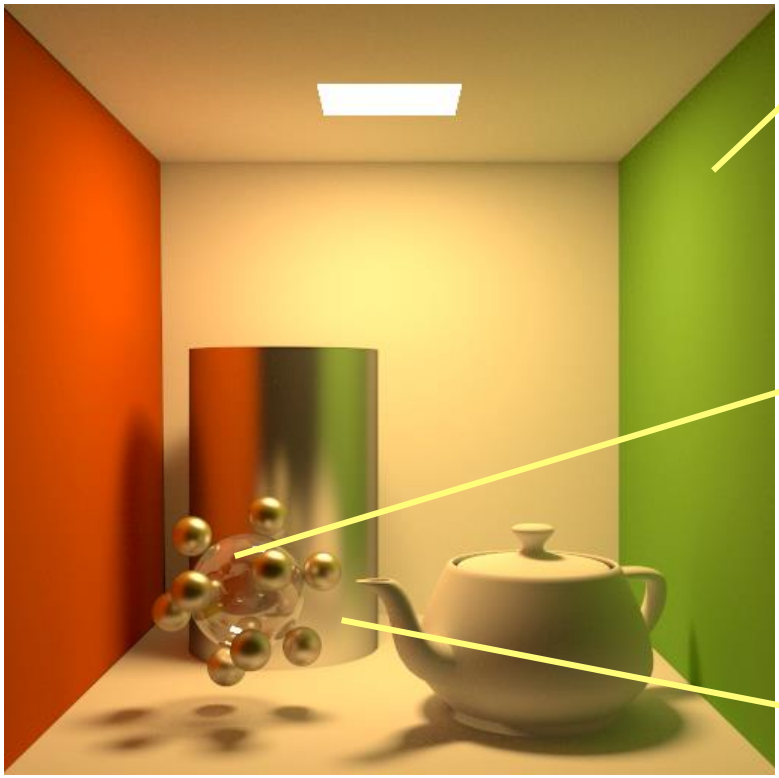


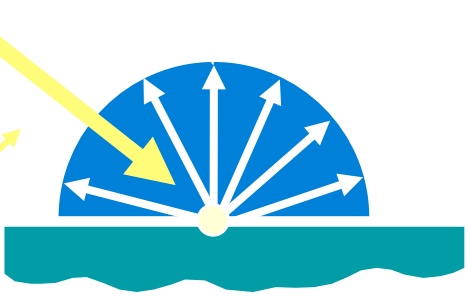
$$I = N \cdot L$$

Image intensity = Surface normal · Light direction

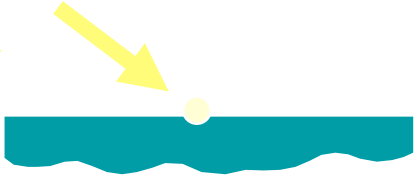
Image intensity $\propto$ cos(angle between N and L)

Materials - Three Forms



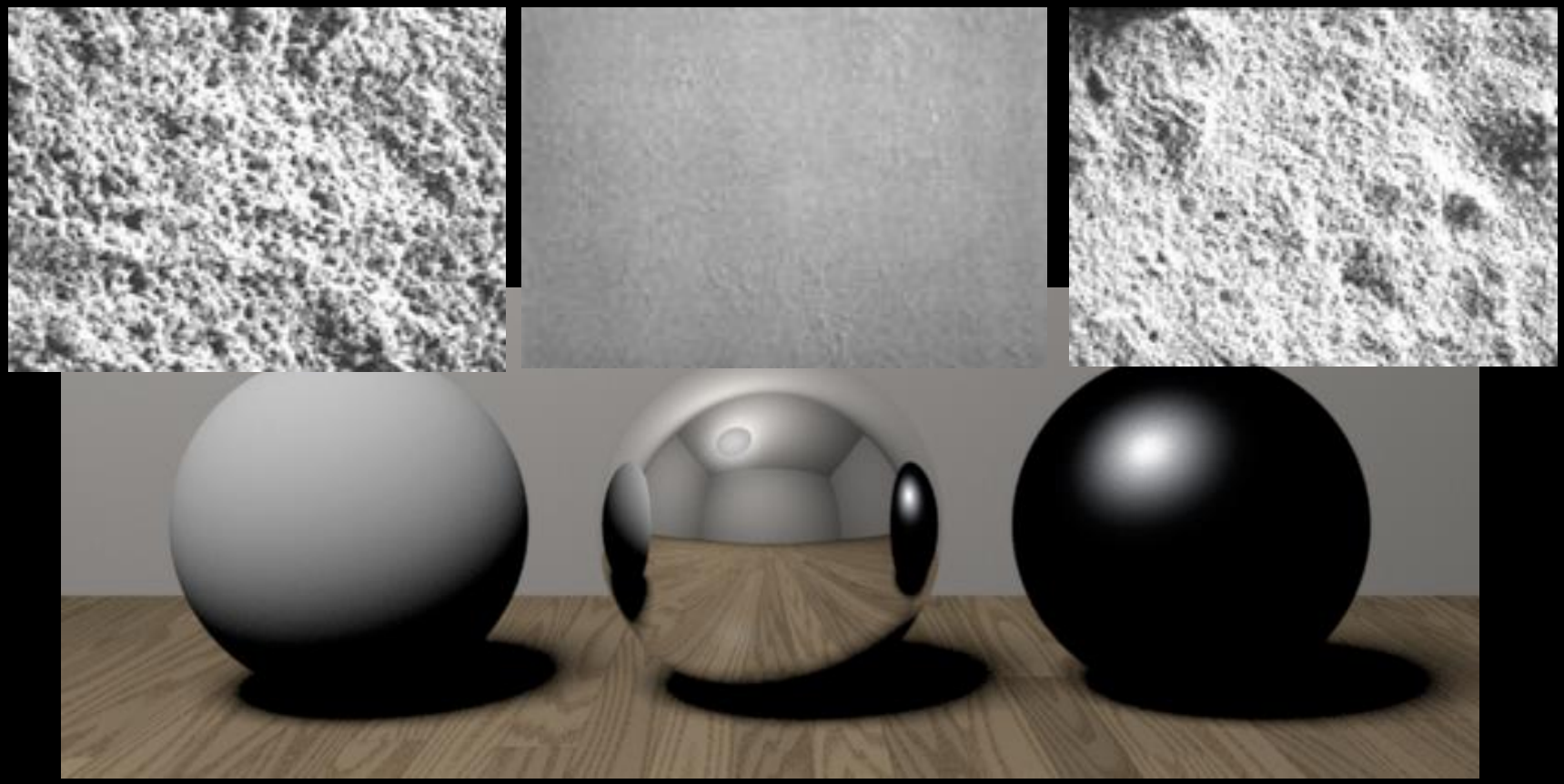


Ideal diffuse (Lambertian)

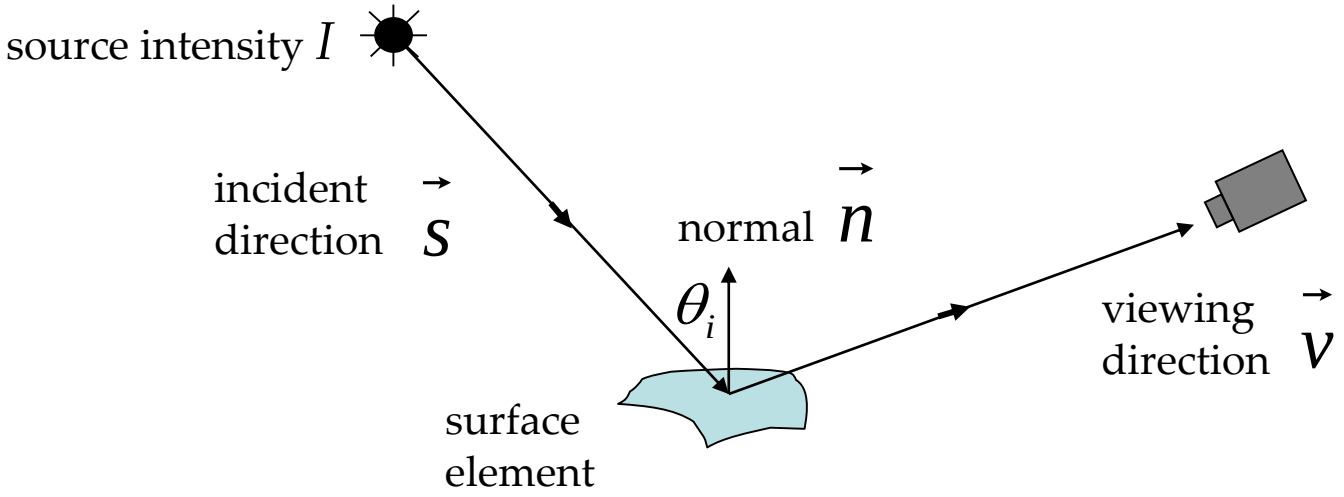


Ideal specular

Directional diffuse



BRDF: bidirectional reflectance distribution function



$$f(\theta_i, \phi_i; \theta_r, \phi_r)$$

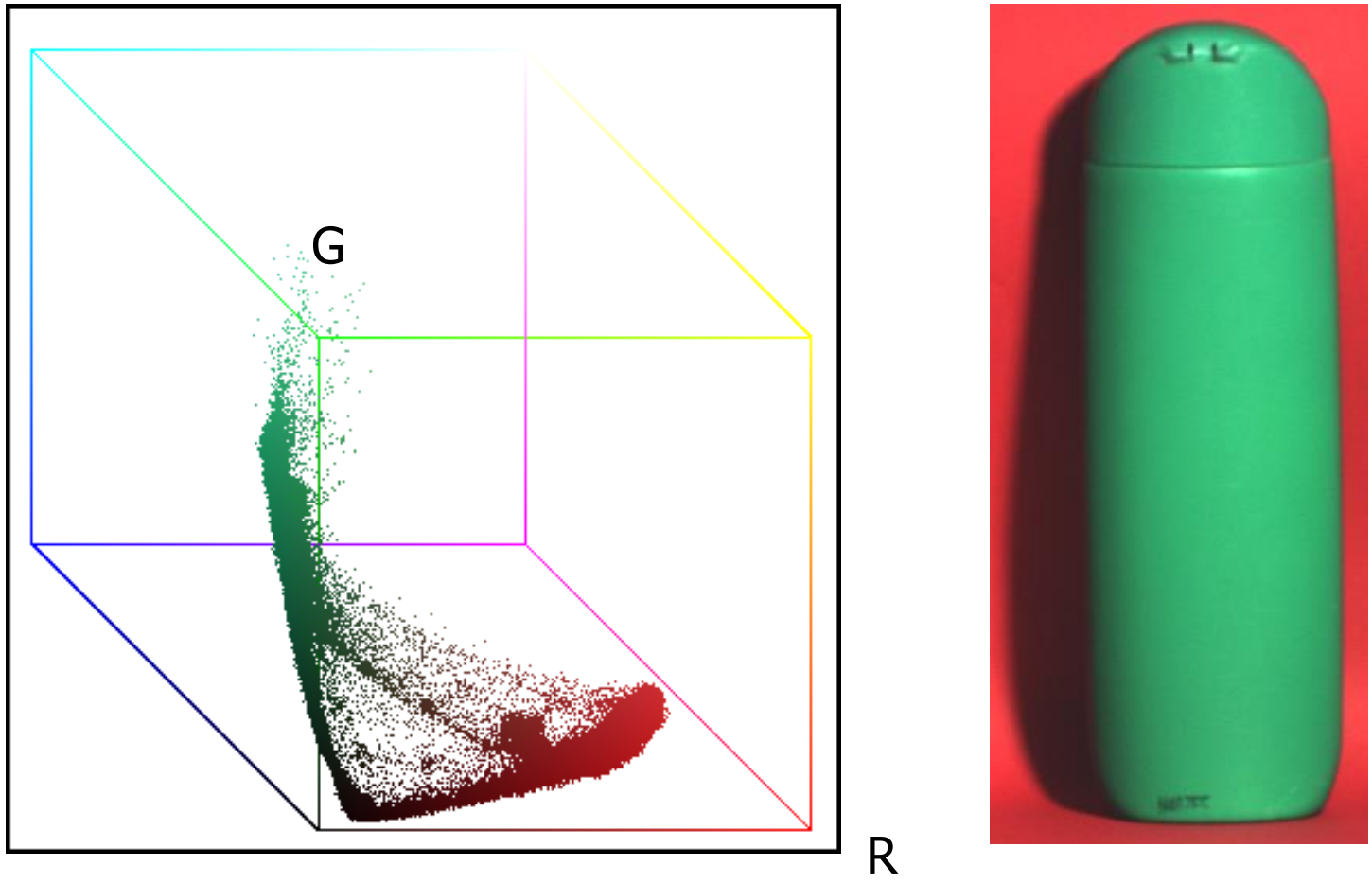
- Lambertian BRDF is simply a constant : $f(\theta_i, \phi_i; \theta_r, \phi_r) = \frac{\rho_d}{\pi}$ albedo

Reflection Model

$$C = m_b(\mathbf{n},\mathbf{s})\int_{\lambda} e(\lambda)\rho(\lambda)\, f_C(\lambda)d\lambda + m_s(\mathbf{n},\mathbf{s},\mathbf{v})\int_{\lambda} e(\lambda)c(\lambda)\, f_C(\lambda)d\lambda$$

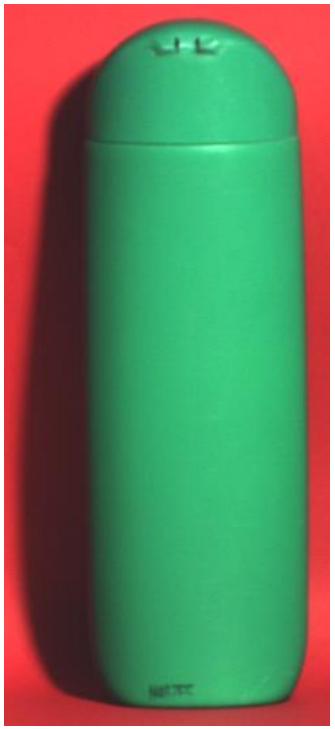
$\rho(\lambda)$	surface albedo	scene & viewpoint invariant
$e(\lambda)$	illumination	scene dependent
$\mathbf{n}$	object surface normal	object shape variant
$\mathbf{s}$	illumination direction	scene dependent
$\mathbf{v}$	viewer's direction	viewpoint variant
$f_C(\lambda)$	sensor sensitivity	camera dependent
$c(\lambda)$	specular	surface dependent

Body Reflectance in RGB - Matte Surfaces

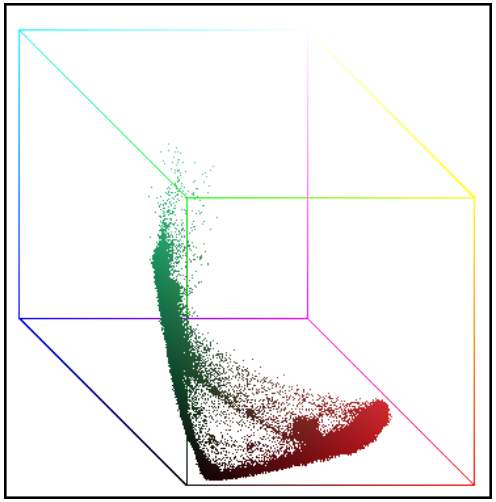


An object with its representation in RGB-colour space.

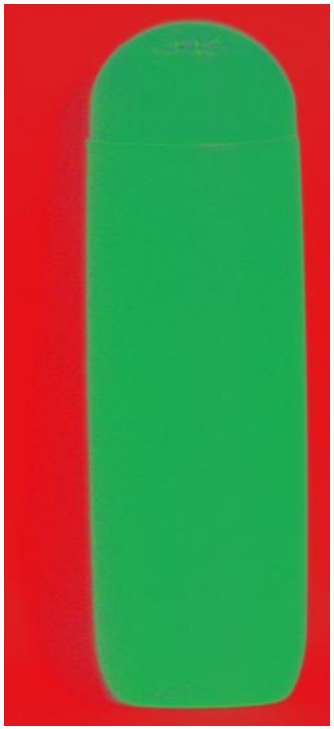
Colour Invariance / rgb space



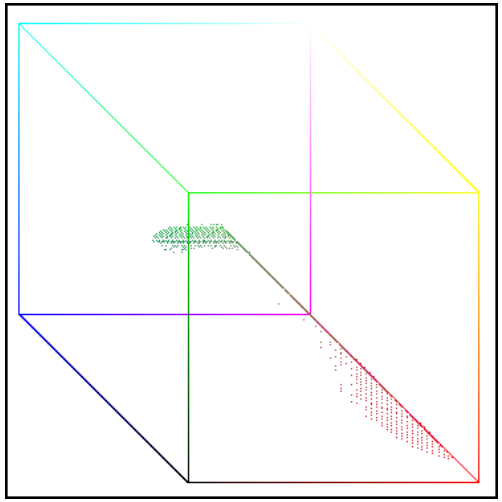
Original RGB image



3D plot of RGB image

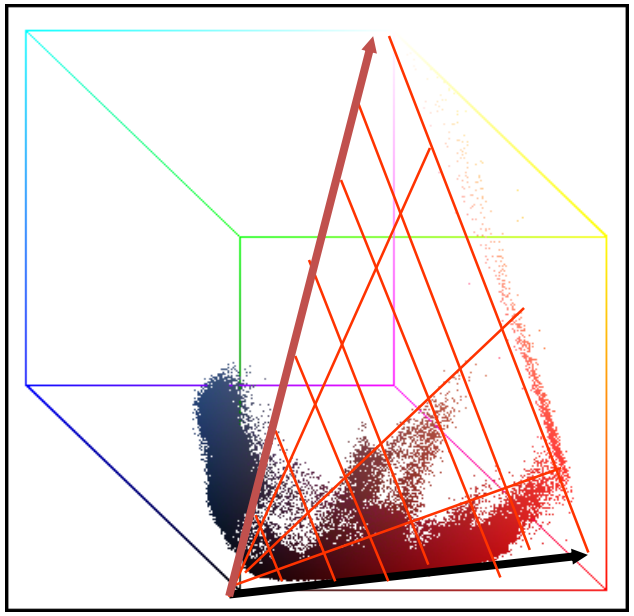


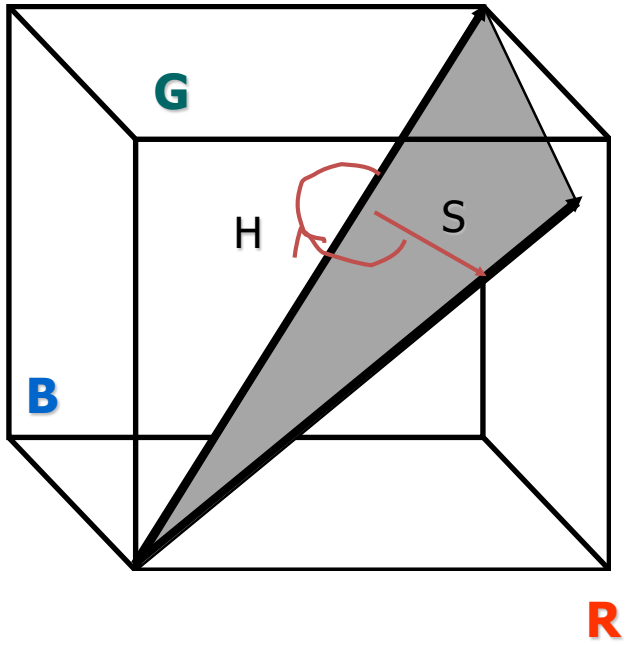
rgb image



3D plot of rgb image

Colour Invariance: Hue





Hue: $H(R,G,B) = \arctan\left(\frac{\sqrt{3}(G-B)}{(R-G)+(R-B)}\right)$

Colour invariance

$$l1(R,G,B) = \frac{(R-G)^2}{(R-G)^2 + (R-B)^2 + (G-B)^2}$$

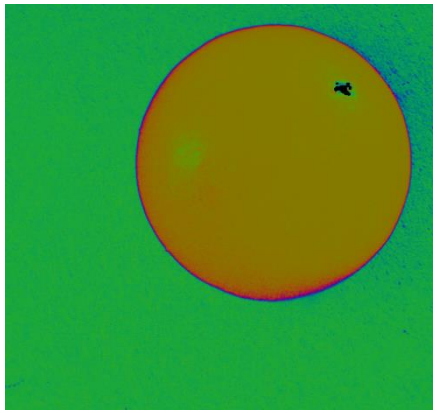
$$l2(R,G,B) = \frac{(R-B)^2}{(R-G)^2 + (R-B)^2 + (G-B)^2}$$

$$l3(R,G,B) = \frac{(G-B)^2}{(R-G)^2 + (R-B)^2 + (G-B)^2}$$

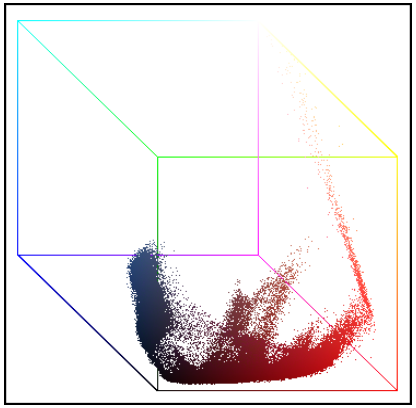
Colour Invariance: Hue

Original colour image

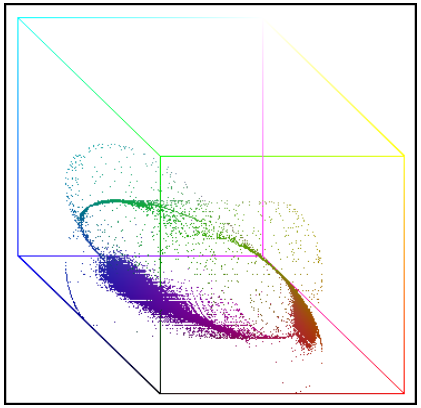
L1l2l3 transformation



L1l2l3 image



3D plot of colour image

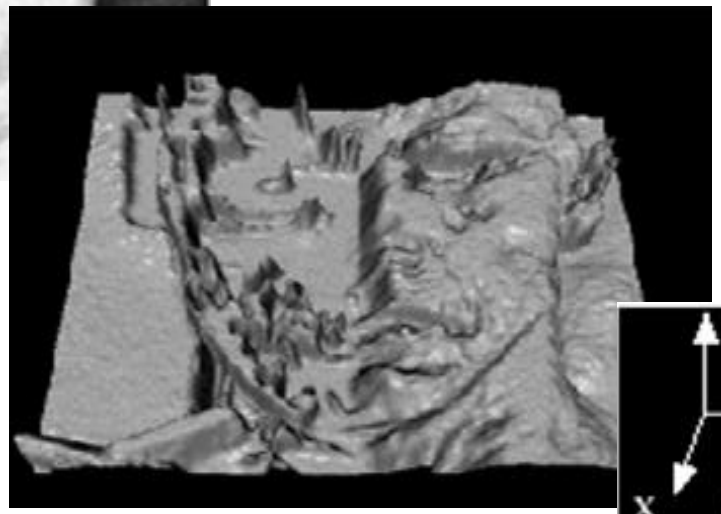


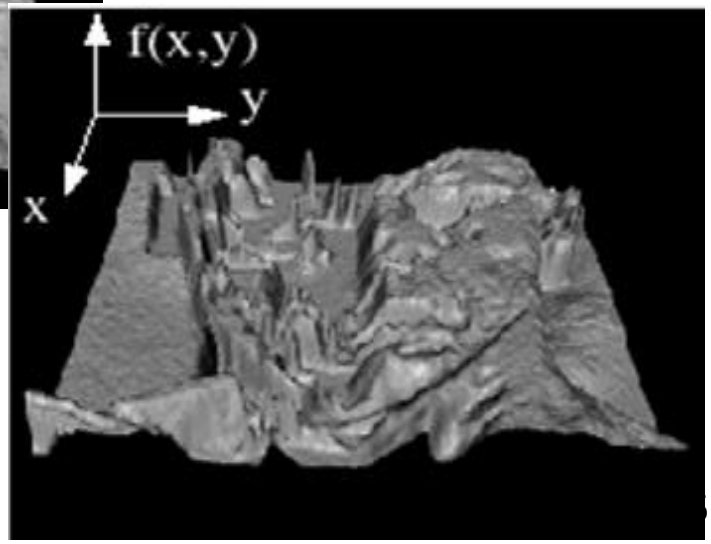
3D plot of l1l2l3 image

Today's Class

1. Reflection Models
2. Image Processing
   - Pixel processing
     - Grey image processing
     - Colour correction
   - Image Filtering
   - Key-point detection
     - Harris corner detector
     - SIFT
   - Applications

I = f(x,y)



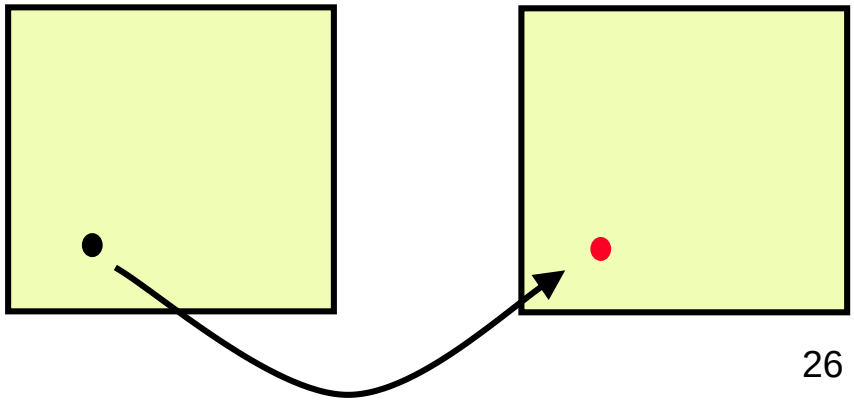


Point Operations

- Operation on ***Pixel's value***
- Context free
- Operation can be performed on the image histogram
- Example:

$$g(x,y) = M(f(x,y))$$

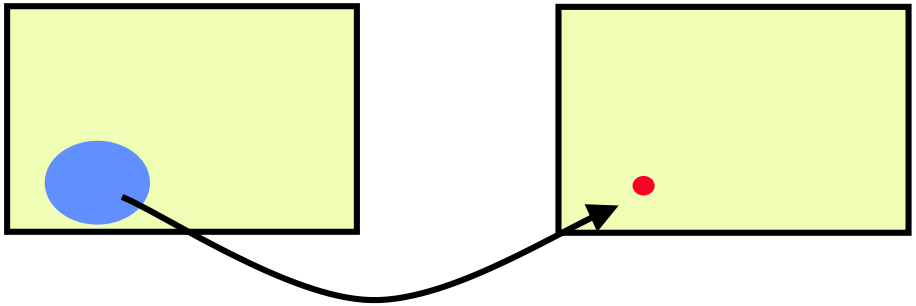
$$g(x,y) = \alpha \cdot f(x,y) + \beta$$



Neighborhood Operations

- Operation depends on ***Pixel's value*** and its spatial ***neighbour***s.
- Context dependant.
- Example:

$$g(x,y) = M\left(\{f(i,j) \mid (i,j) \in N(x,y)\}\right)$$



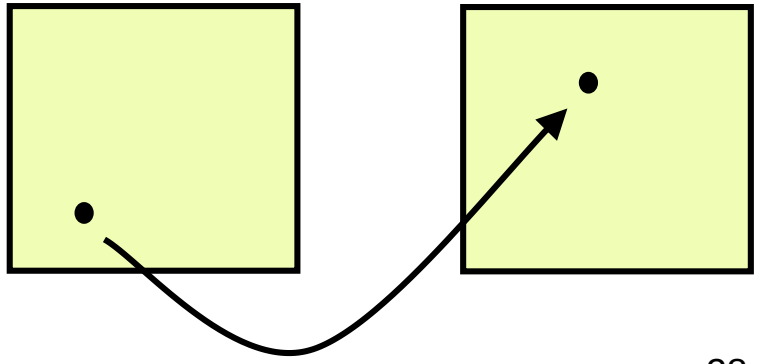
$$g(x,y) = \sum_{i,j \in N(x,y)} f(i,j)/n$$

Geometric Operations

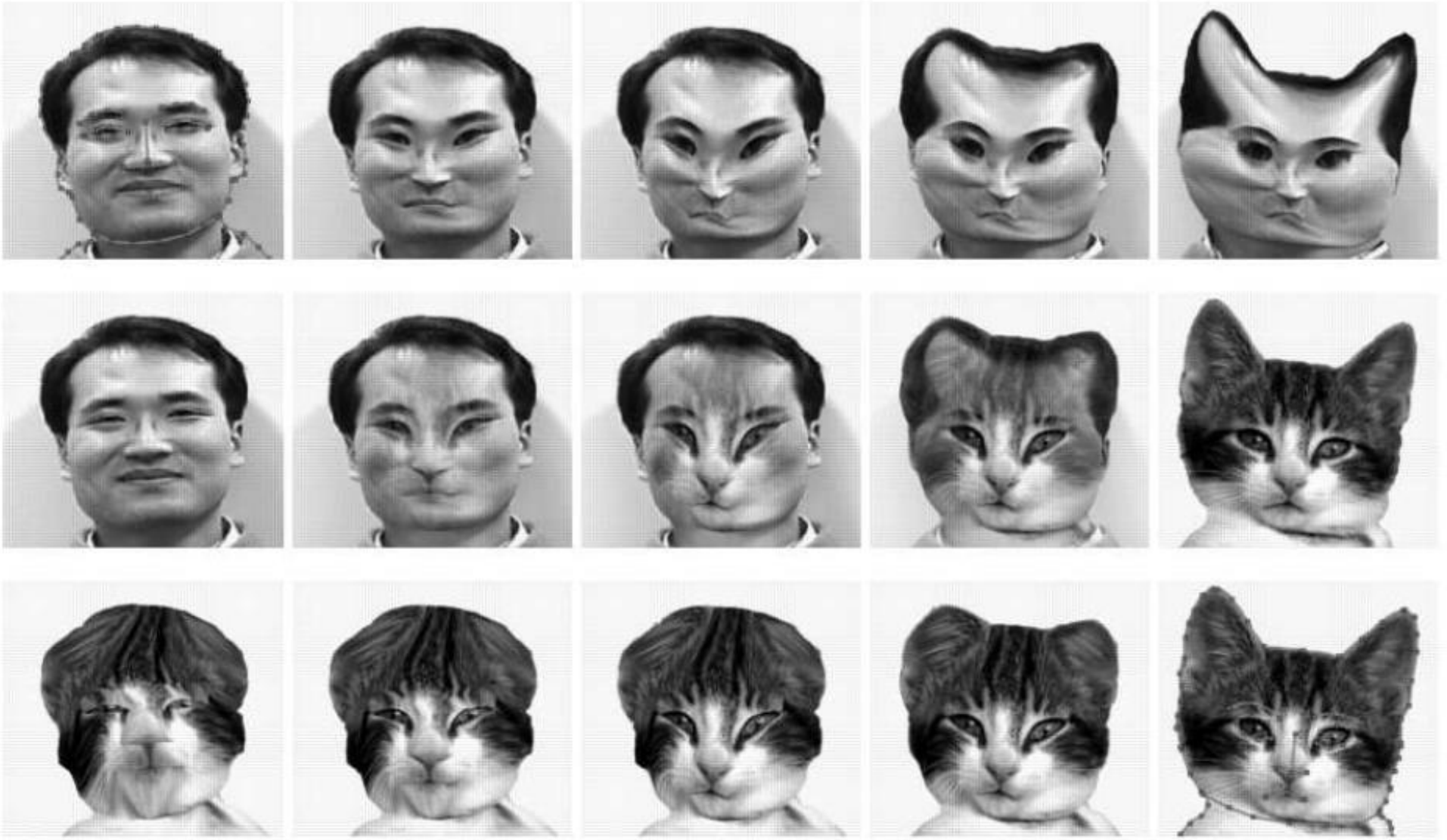
- Operation depend on ***pixel's coordinates***.
- Independent of pixels value.
- Context free.
- Example: translation

$$g(x,y) = f(G(x,y))$$

$$g(x,y) = f(x+a, y+b)$$



Geometric Operations : Image Morphing



Today's Class

1. Reflection Models
2. Image Processing
   - Pixel processing
     - Grey image processing
     - Colour correction
   - Image Filtering
   - Key-point detection
     - Harris corner detector
     - SIFT
   - Applications

Point operations:

(Histogram operation)

Point Operations

- A point operation can be defined as a mapping function: $v_{new} = M(v_{old})$
  where *v* stands for gray values.
- M(v) takes any value v in the source image into v_{new} in the destination image.
- Simplest case - Linear Mapping:

$$M(v) = \alpha v + \beta$$

- Often implemented on the image histogram.

Histogram

The histogram of a digital image with gray values

$$r_0, r_1, \cdots, r_{L-1}$$

is the discrete function

$$p(r_k) = \frac{n_k}{n}$$

n_k: Number of pixels with gray value r_k

n: total Number of pixels in the image

The function $p(r_k)$ represents the fraction of the total number of pixels with gray value r_k.

Histogram

- Histogram provides a global description of the **appearance** of the image.

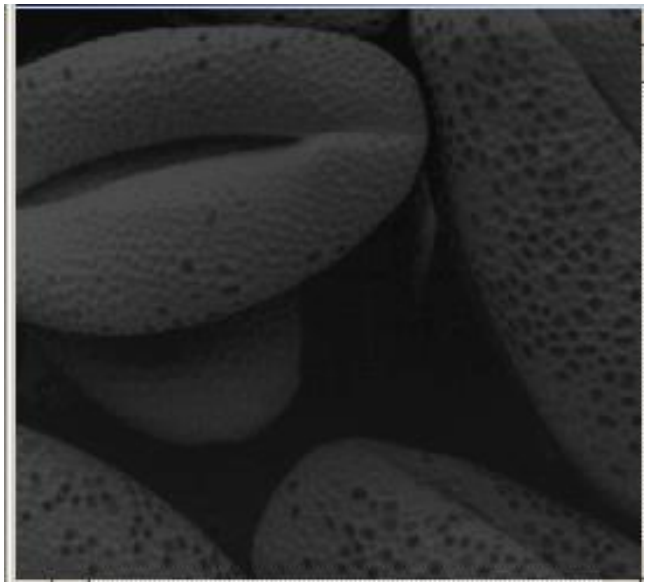
- If we consider the gray values in the image as realizations of a random variable R, with some probability density, histogram provides an approximation to this **probability density**.

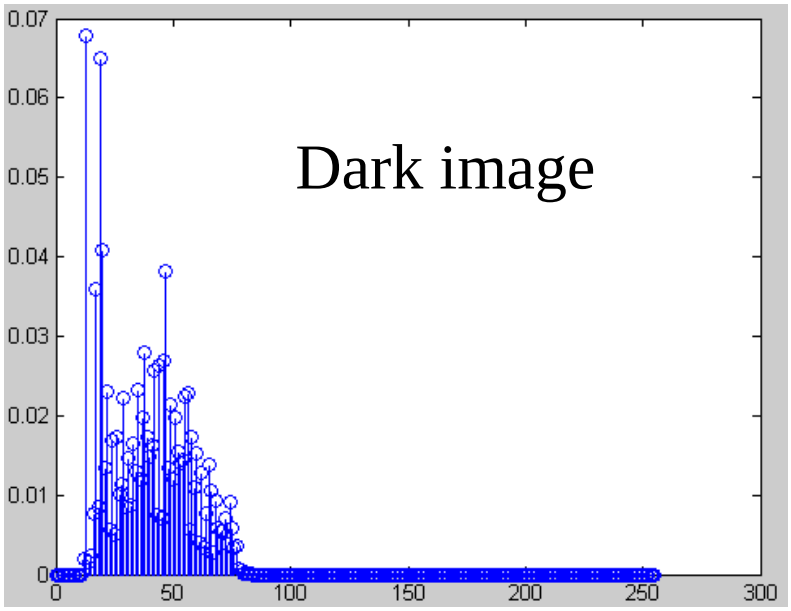
- In other words,

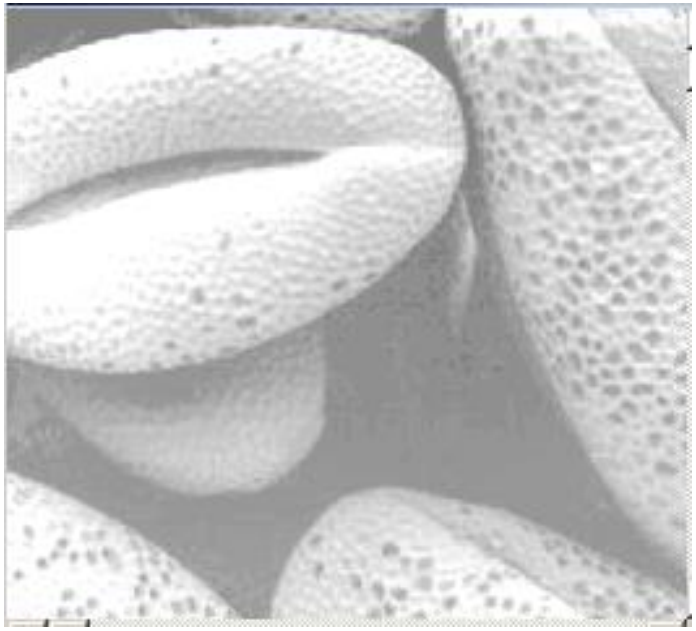
$$\Pr(R = r_k) \approx p(r_k)$$

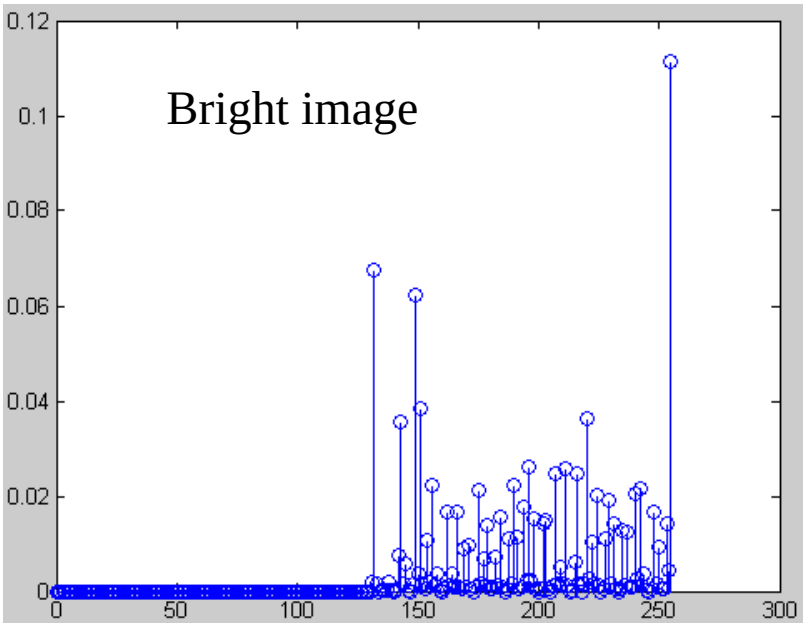
Some Typical Histograms

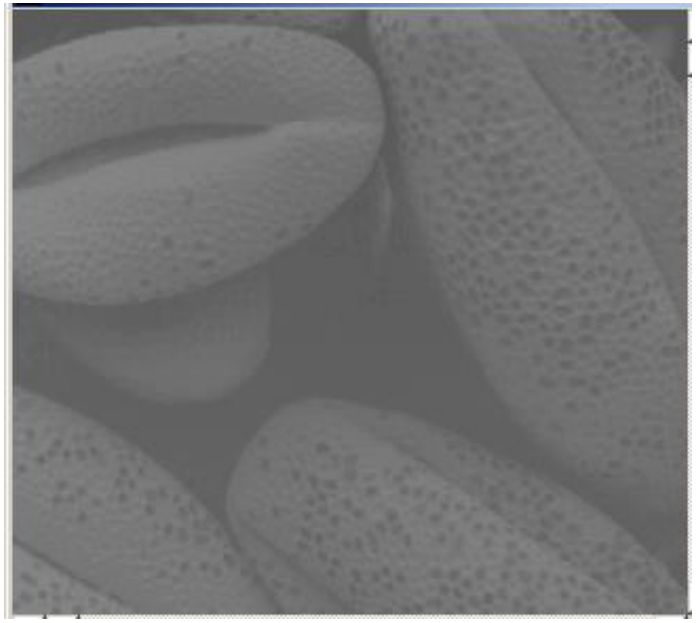
- The shape of a histogram provides useful information for image global appearance.

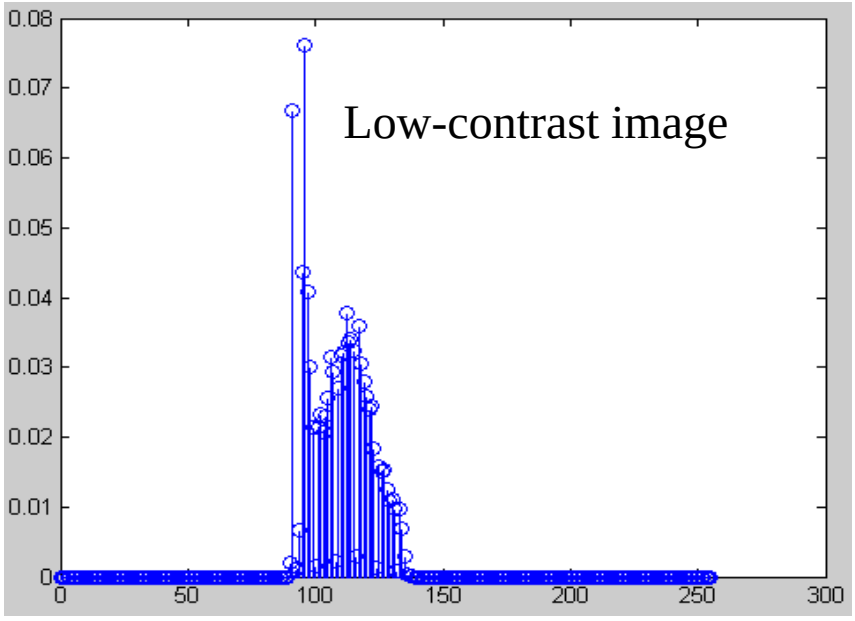


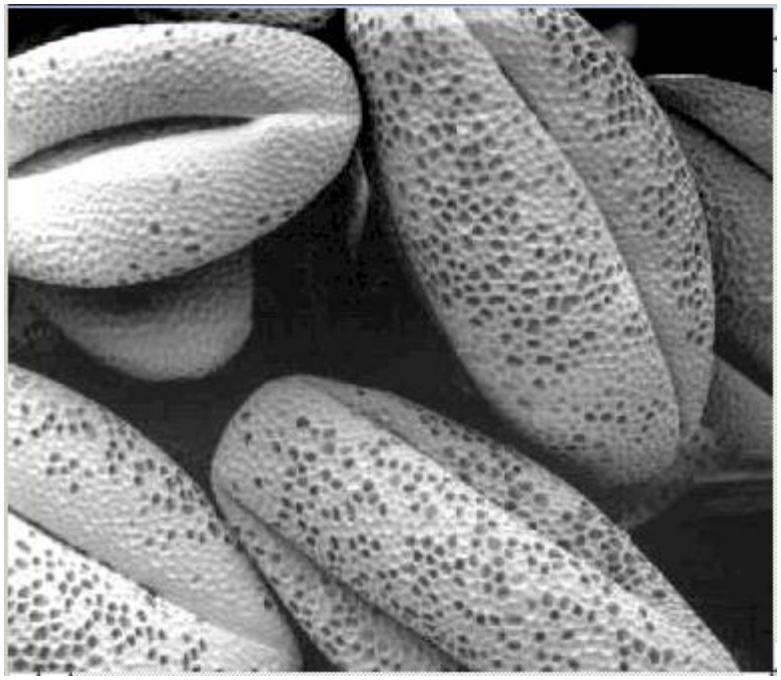


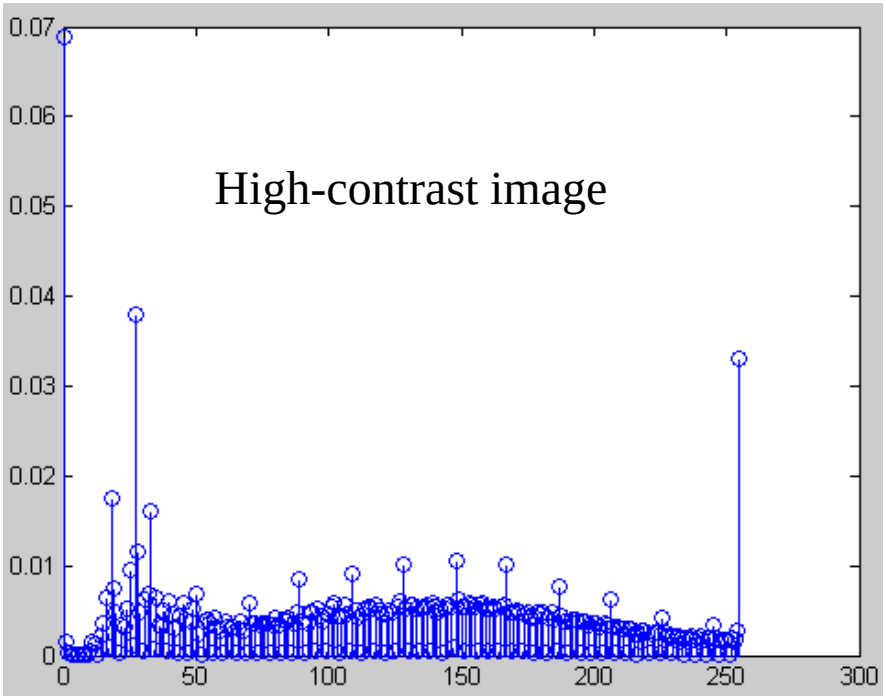



Bright image
0.12
0.1
0.08
0.06
0.04
0.02
0
0
50
100
150
200
250
300

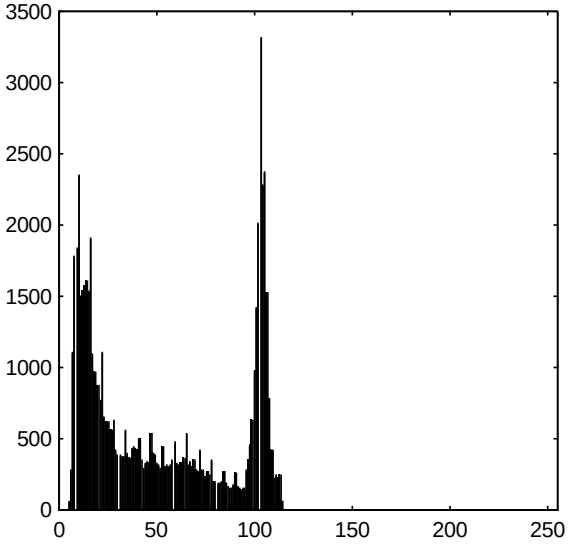



Low-contrast image
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
0
50
100
150
200
250
300

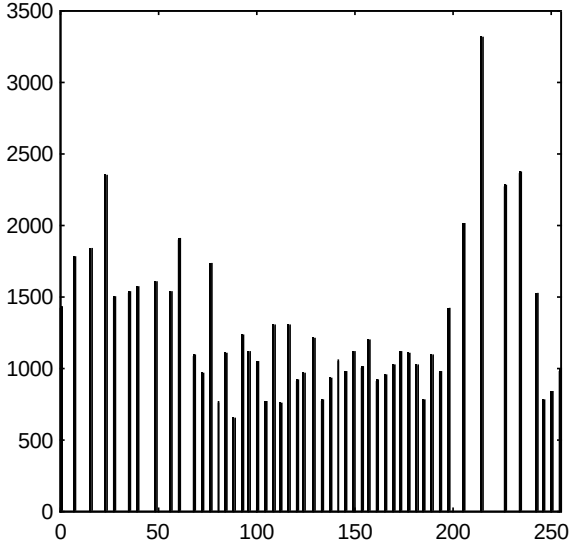



High-contrast image
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
0
50
100
150
200
250
300

Hist. Eq.: Example 1



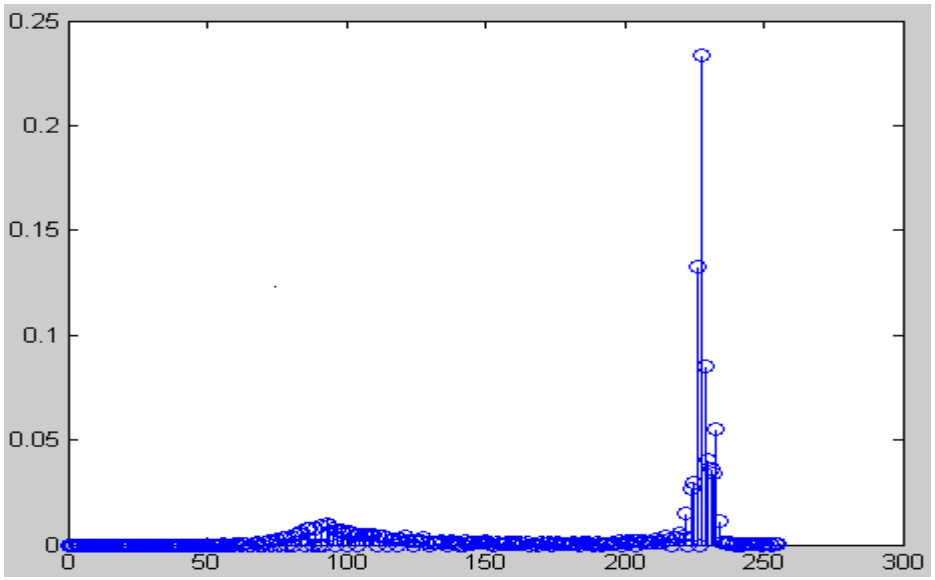
Original

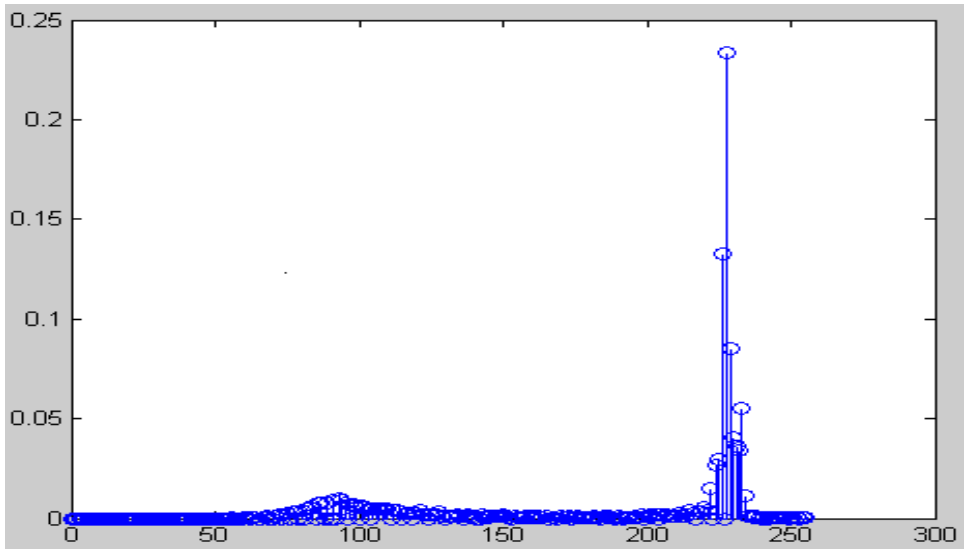


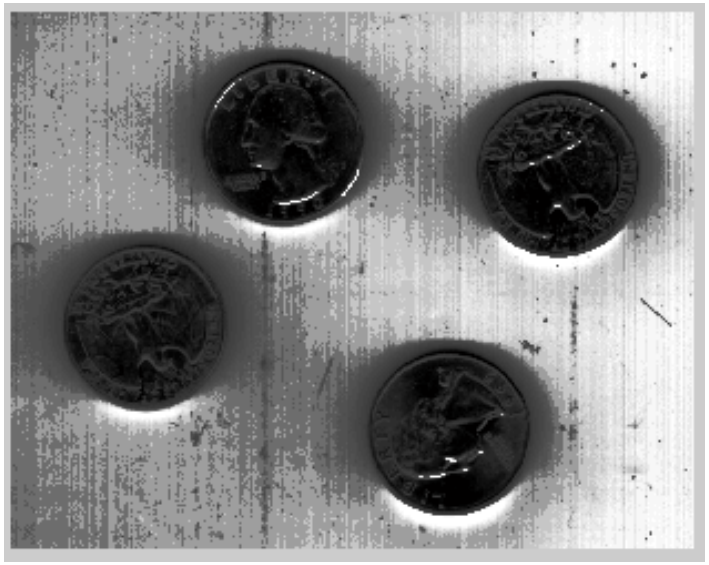
Equalized

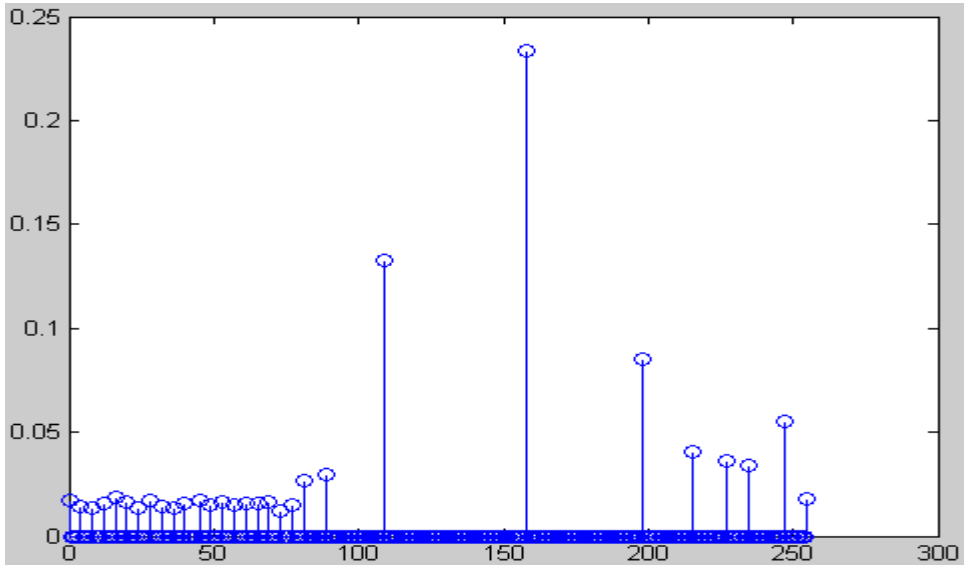
• Comments:

Histogram equalization may not always produce desirable results, particularly if the given histogram is very narrow. It can produce false edges and regions. It can also increase image “graininess” and “patchiness.”

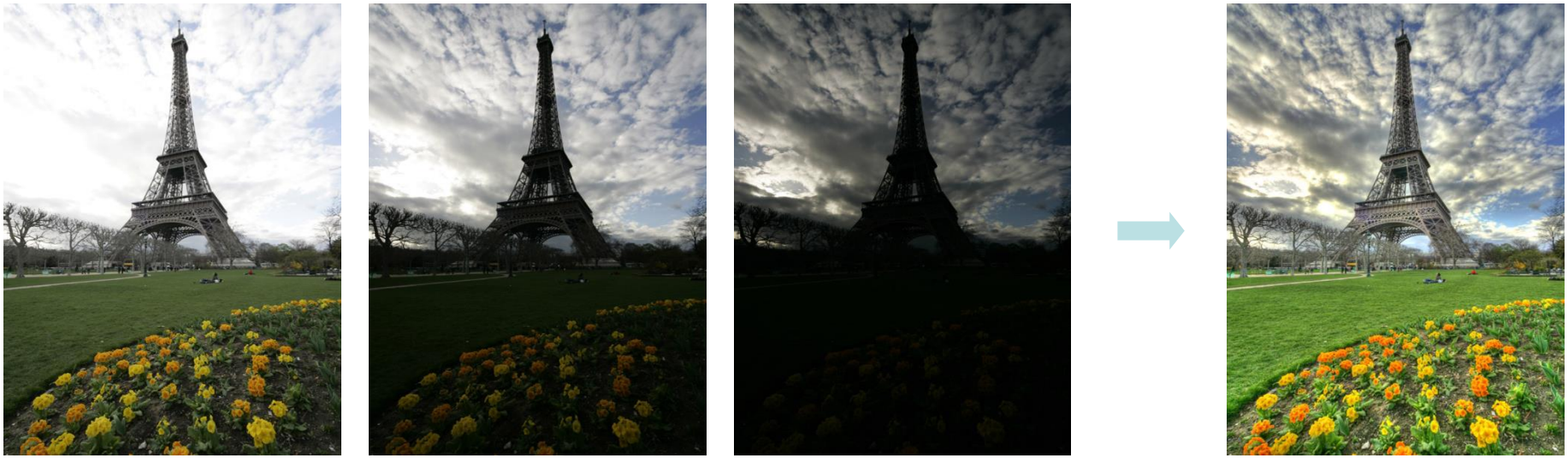



0.25
0.2
0.15
0.1
0.05
0
0
50
100
150
200
250
300




0.25
0.2
0.15
0.1
0.05
0
0
50
100
150
200
250
300

Application: HDR image enhancement



Today's Class

1. Reflection Models
2. Image Processing
   - Pixel processing
     - Grey image processing
     - Colour correction
   - Image Filtering
   - Key-point detection
     - Harris corner detector
     - SIFT
   - Applications

Illumination change

0 

1 

2 

3 

4 

5 

6 

7 

8 

9 

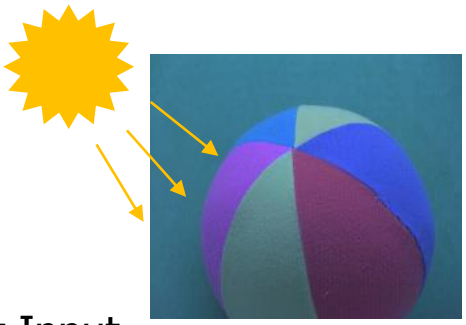
10 

11 illuminants

(0 is canonical)

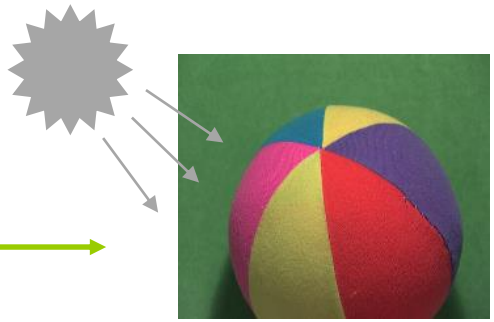
Color normalization

Unknown illuminant



Test Input

Canonical (reference) illuminant



Algorithm

(Map image as if it were taken under reference illuminant).

Color constancy

Gray-world assuption

Scale-by-max

(white patch assumption)

Unknown

Canonical

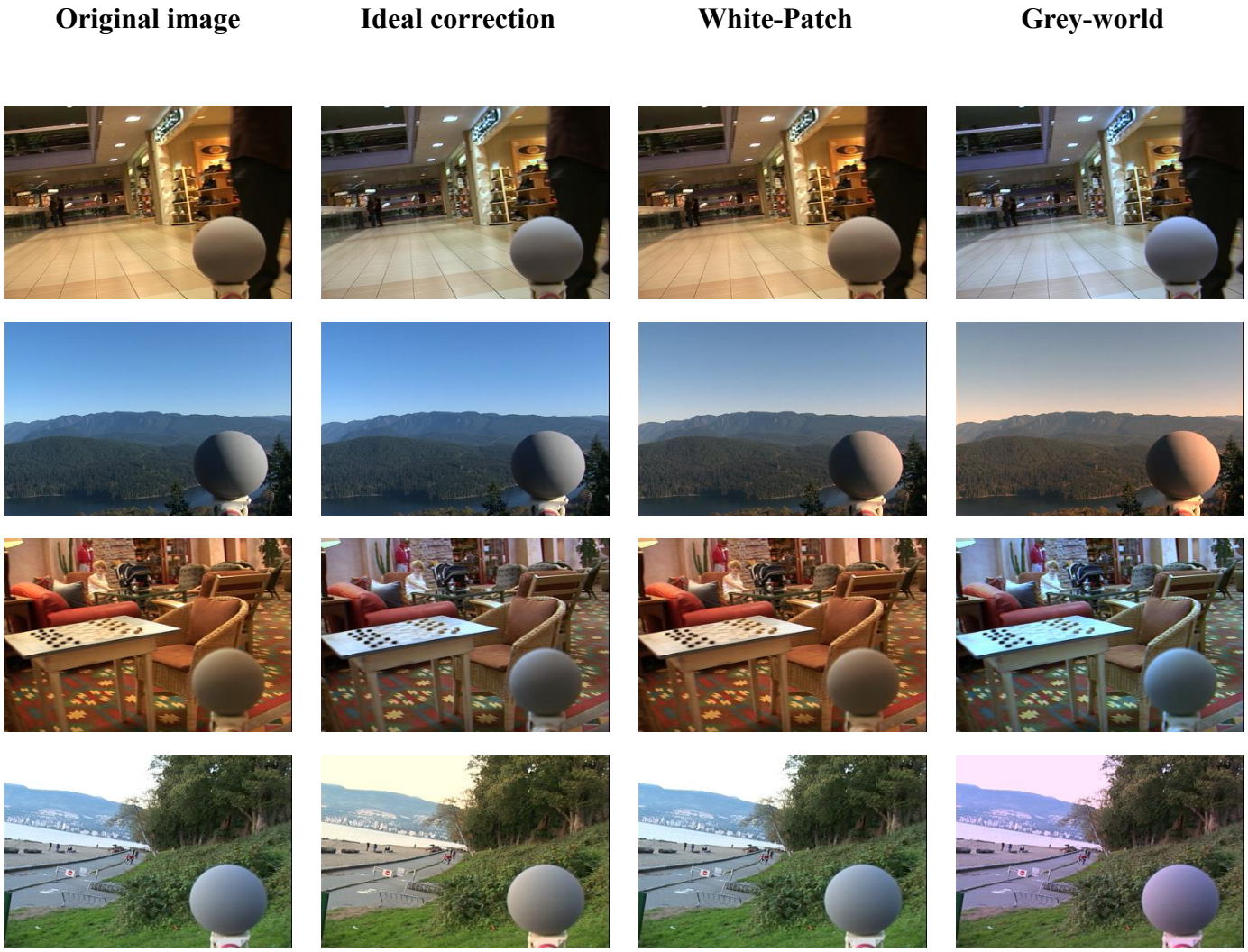
Mean color = constant

e.g. average value in R,G,B = 128

Max color = constant

e.g. max value in R,G,B = 255

White Patch and Grey-world Examples



Images from “A large image data set for color constancy research”, by F. Ciurea and B. Funt in CIC 2003.

Today's Class

1. Reflection Models
2. Image Processing
   - Pixel processing
     - Grey image processing
     - Colour correction
   - Neighborhood processing Image Filtering
   - Edge Detection
   - Corner detection
     - Harris corner detector
     - SIFT
   - Applications

Box Filter

What does it do?

- Replaces each pixel with an average of its neighborhood

$$g[\cdot,\cdot]$$

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Slide credit: David Lowe (UBC)

Image Filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

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

$h[.,.]$

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Image Filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

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

$h[.,.]$

	0	10							

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Image Filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

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

$h[.,.]$

	0	10	20						

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Image Filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

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

$h[.,.]$

	0	10	20	30					

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Image Filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

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

$h[.,.]$

	0	10	20	30	30				

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Image Filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

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

$h[.,.]$

	0	10	20	30	30				
				?					

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Image Filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

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

$h[.,.]$

	0	10	20	30	30				
						?			
				50					

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Image Filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

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

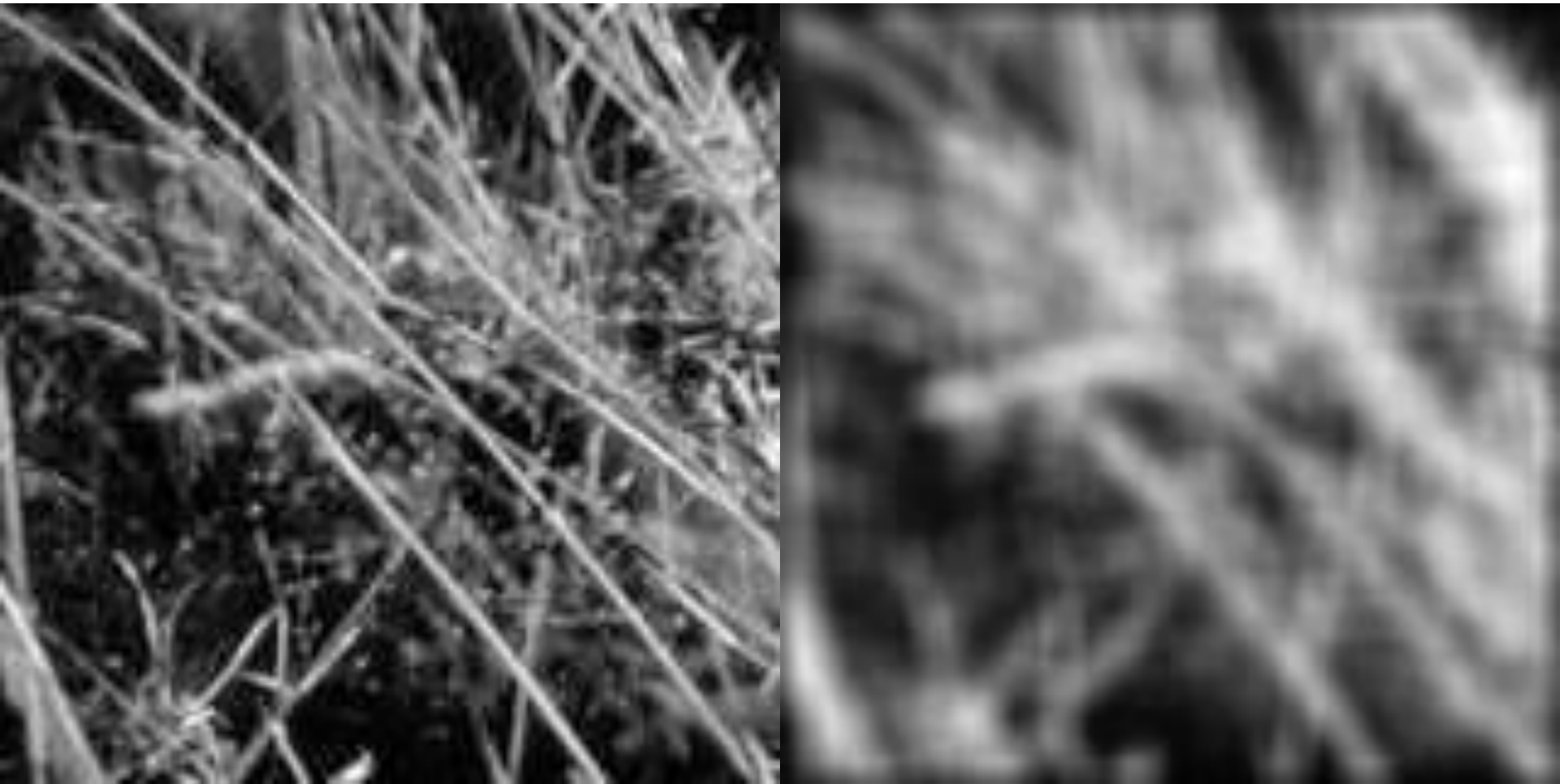
$h[.,.]$

	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	

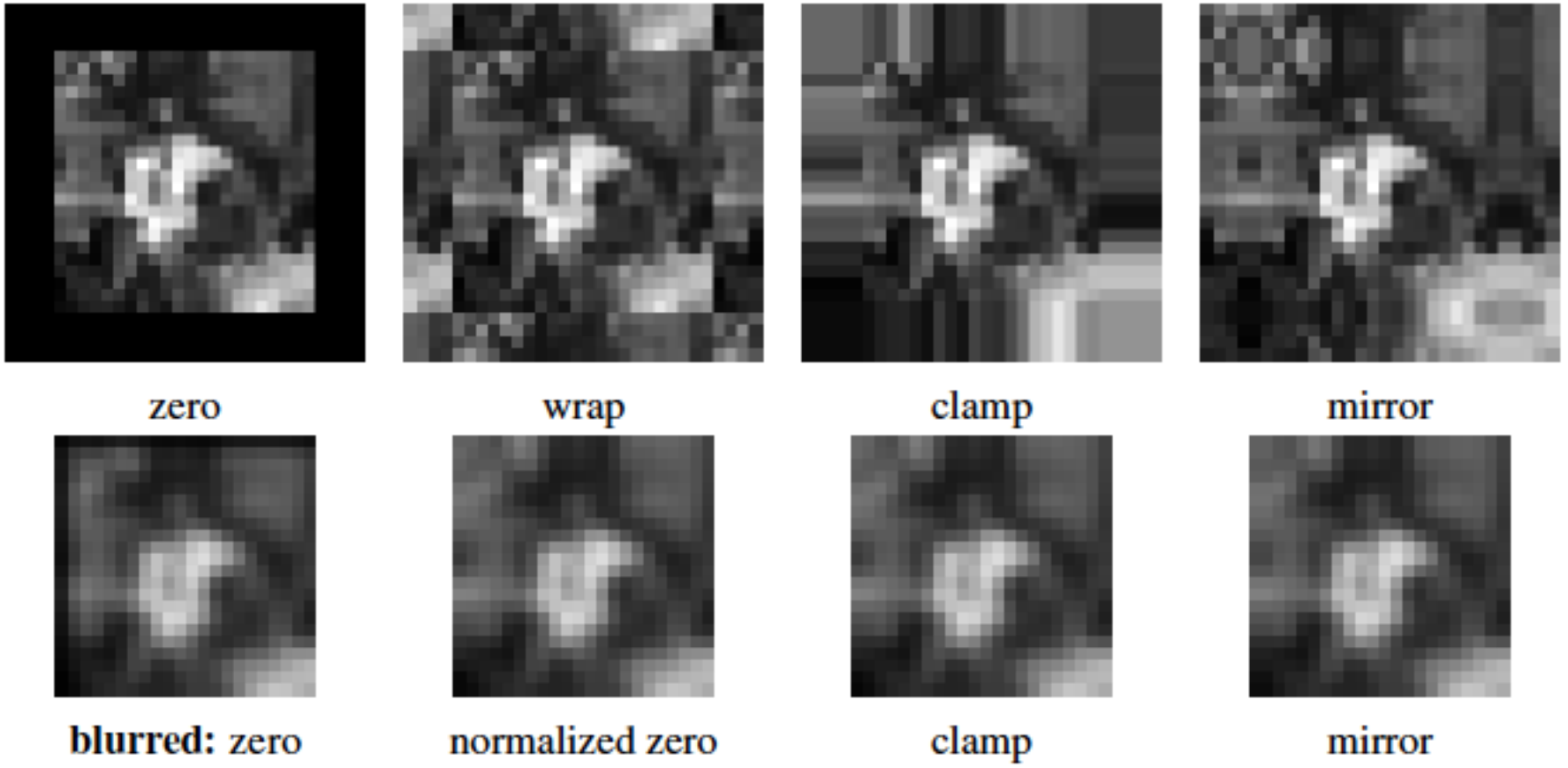
$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Smoothing with Box Filter



How to handle borders?



From Szeliski, Computer Vision, 2010

Mathematical definition of convolution

Convolution between image $f(x, y)$ and kernel $k(x, y)$ is

$$f(x, y) * k(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(u, v)\, k(x - u, y - v)\, du\, dv \tag{1}$$

In discrete form,

$$f(x, y) * k(x, y) = \sum_{i=0}^{W-1} \sum_{j=0}^{H-1} f(i, j)\, k(x - i, y - j) \tag{2}$$

where W and H are the the width and height of the image.

Convolution is commutative (Exercise):

$$f(x, y) * k(x, y) = k(x, y) * f(x, y). \tag{3}$$

Practice with Linear Filters

Original

0	0	0
0	1	0
0	0	0

?

Source: D. Lowe

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

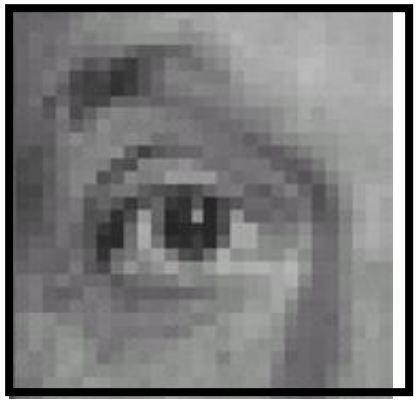
?

Source: D. Lowe

Practice with Linear Filters

Original

0	0	0
0	0	1
0	0	0



Shifted left
By 1 pixel

Source: D. Lowe

Practice with Linear Filters

Original

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \; ?$$

Source: D. Lowe

Practice with Linear Filters

Original

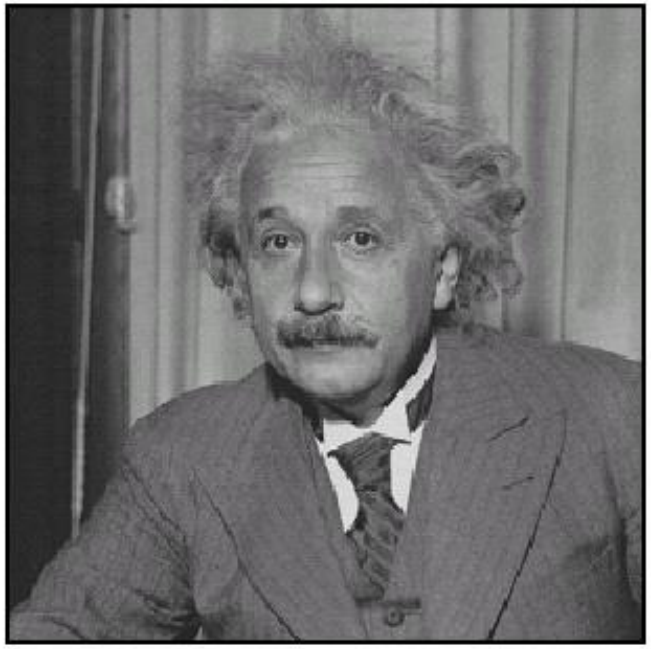
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Sharpening filter

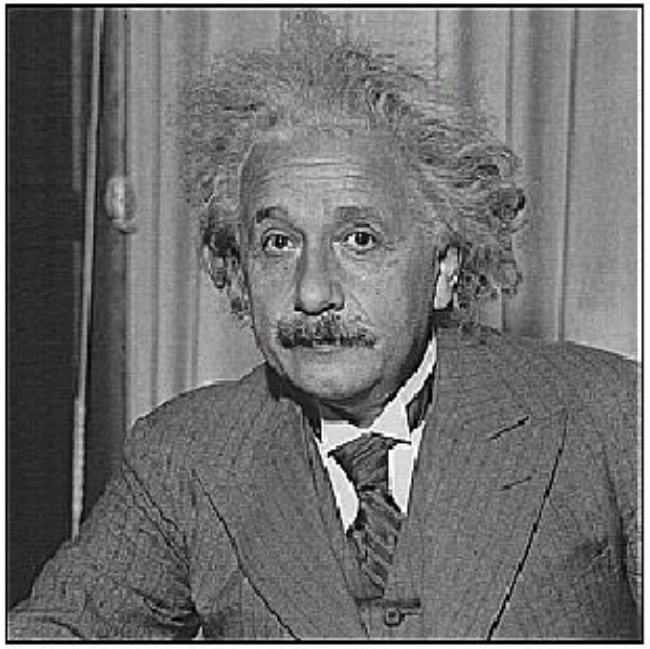
- Accentuates differences with local average

Source: D. Lowe

Sharpening



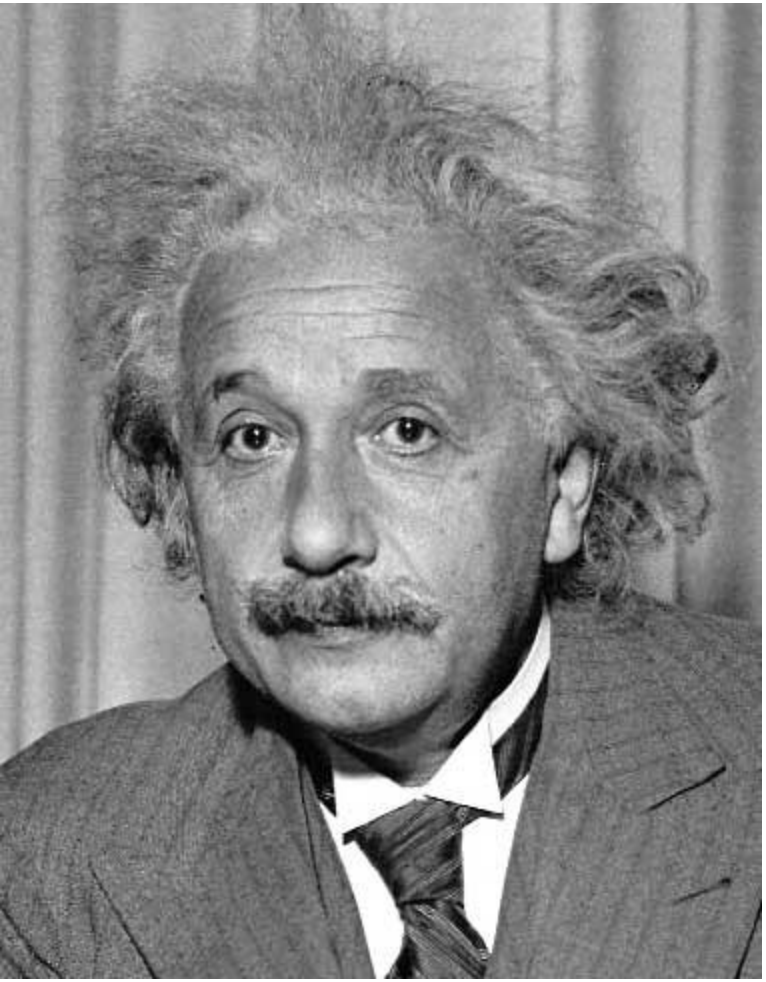
before



after

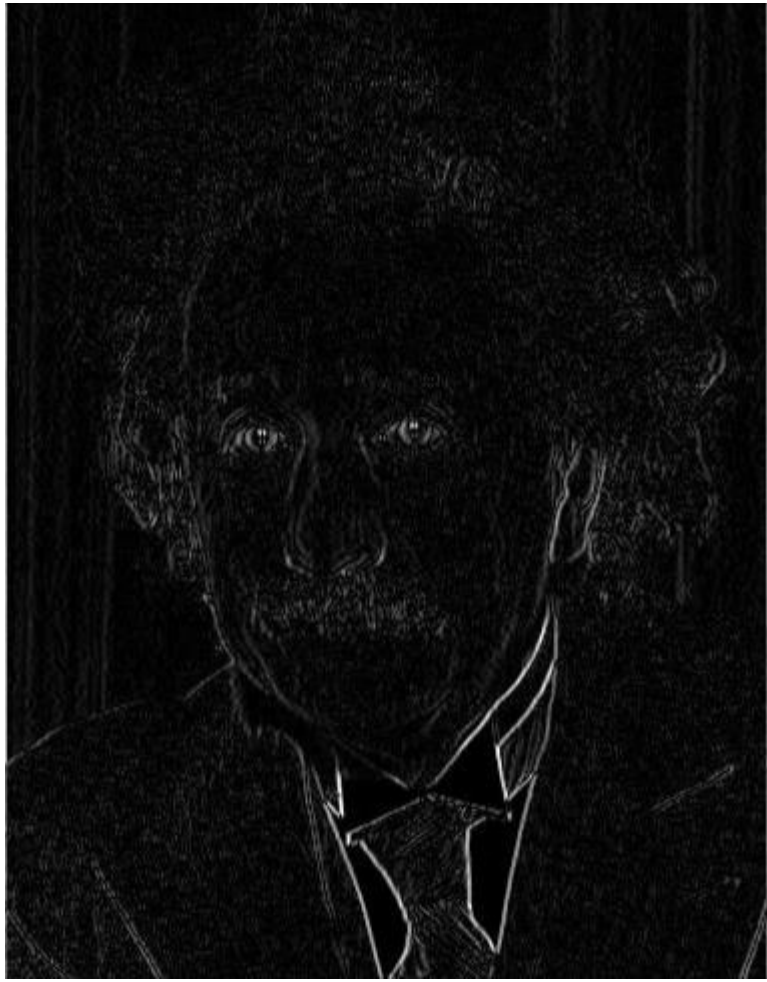
Source: D. Lowe

Other Filters



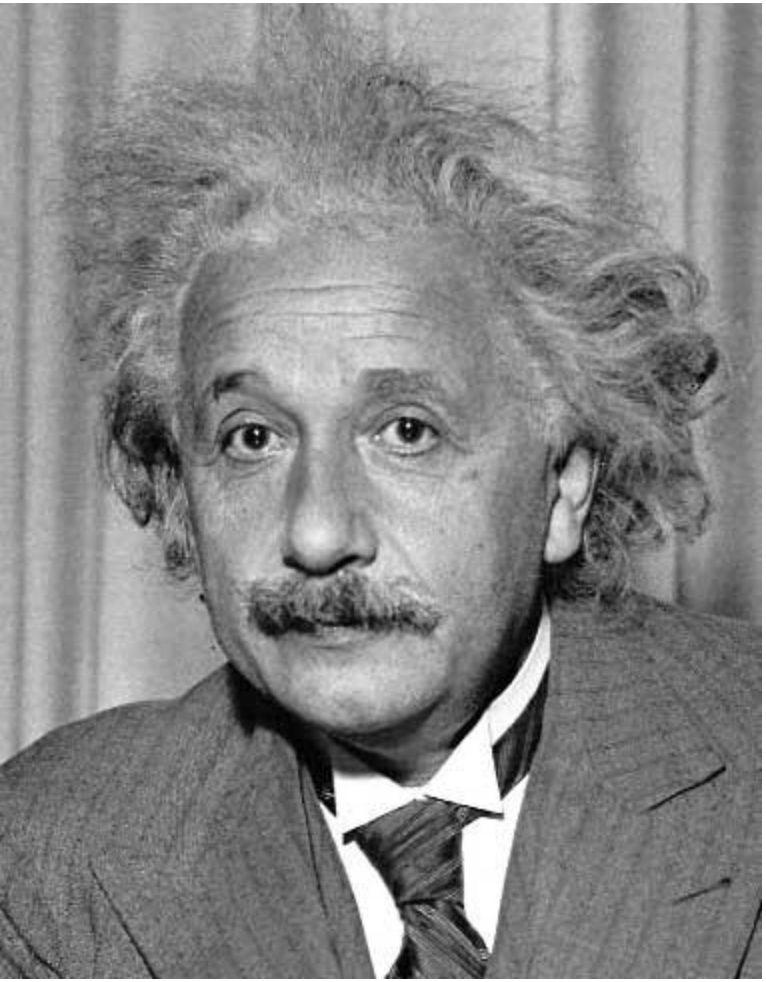
1	0	-1
2	0	-2
1	0	-1

Sobel



Vertical Edge
(absolute value)

Other Filters



1	2	1
0	0	0
-1	-2	-1

Sobel

Horizontal Edge (absolute value)

Edge filter: [-1 1]

$$\frac{\partial \mathbf{I}}{\partial x} \simeq \mathbf{I}(x,y) - \mathbf{I}(x-1,y)$$

$\otimes$ [-1, 1] =

h[m,n]

g[m,n]

f[m,n]

$[-1\ 1]^T$

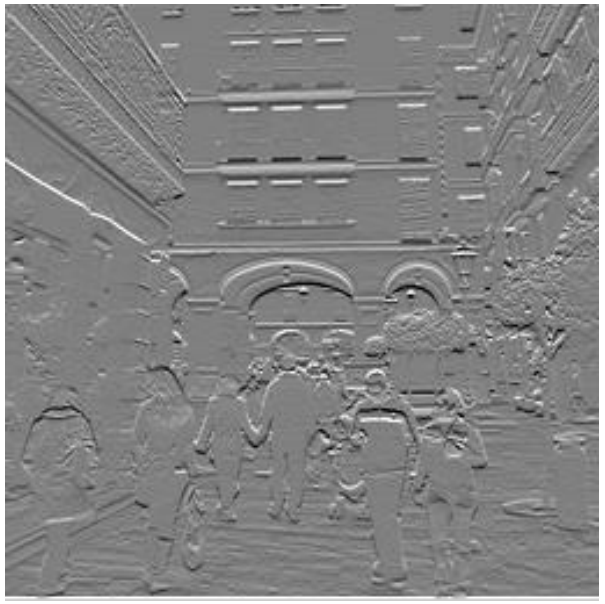
g[m,n]

$\otimes$

$[-1, 1]^T$

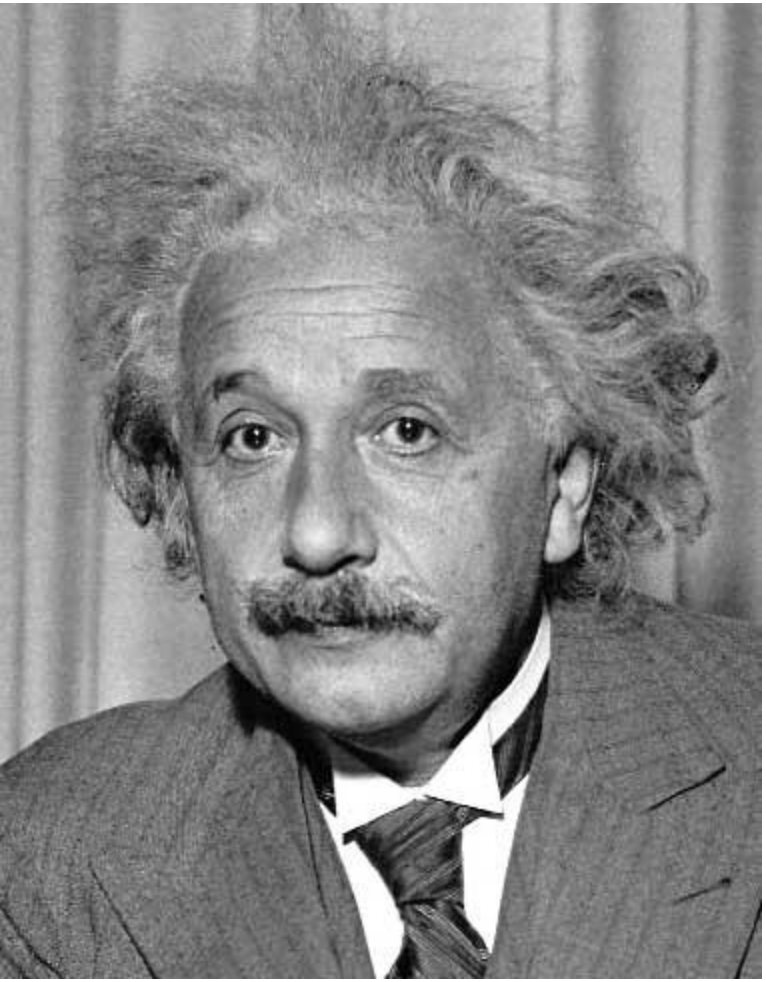
h[m,n]

=



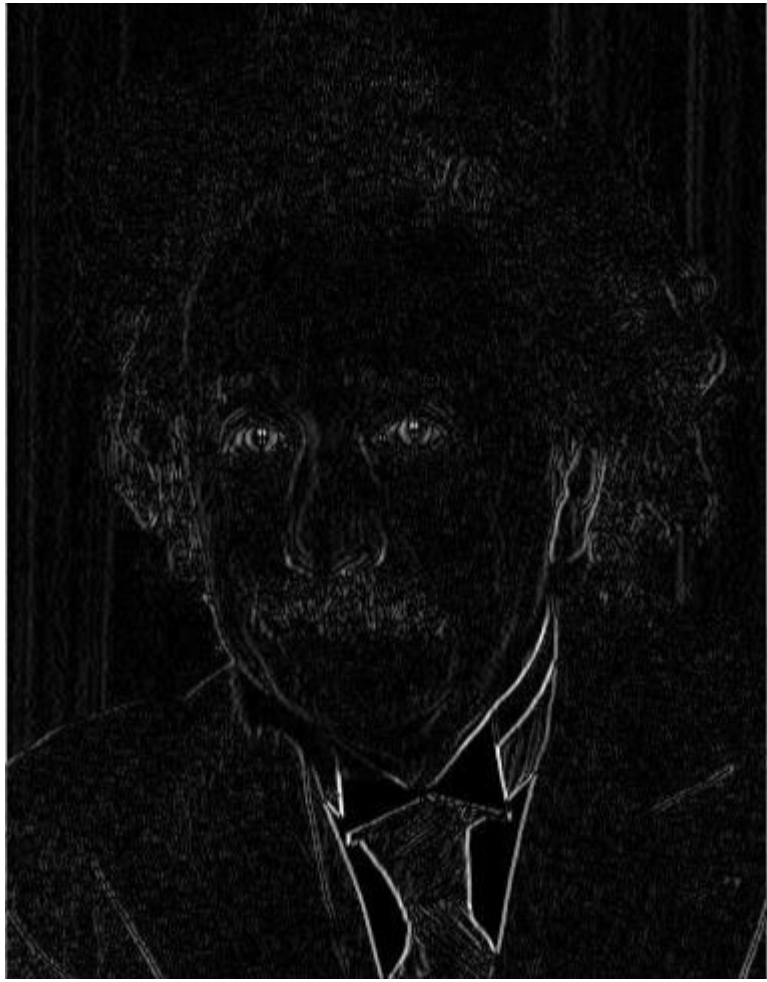
f[m,n]

Derivative filters



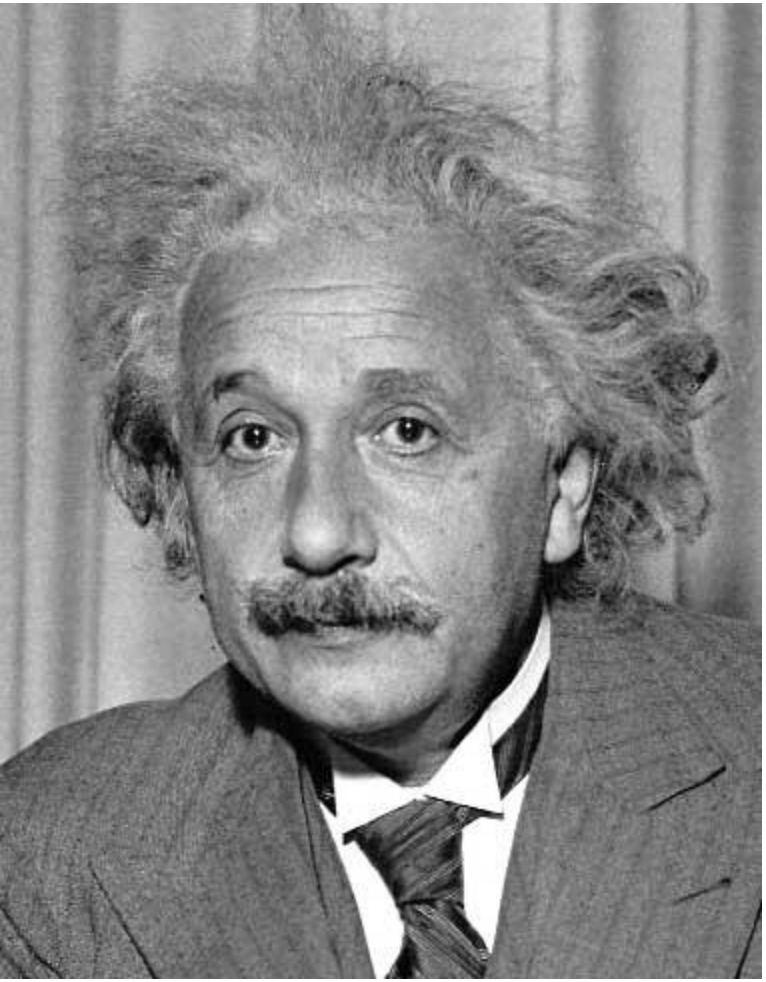
1	0	-1
2	0	-2
1	0	-1

Sobel



Vertical Edge
(absolute value)

Derivative filters



1	2	1
0	0	0
-1	-2	-1

Sobel

Horizontal Edge
(absolute value)

Motion blur filter

g[m,n]

A

h[m,n]

=

f[m,n]

Motion blur filter

g[m,n]

A

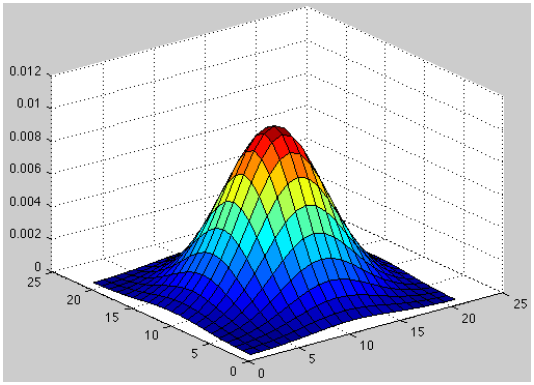
h[m,n]

=

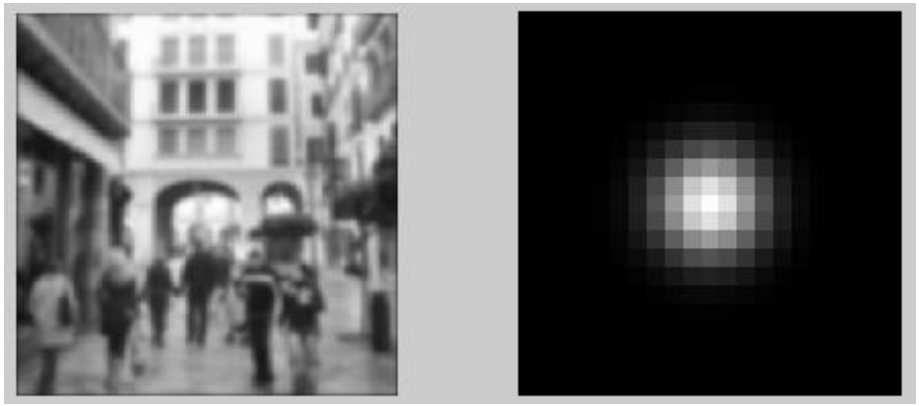
f[m,n]

Gaussian filter: change scales

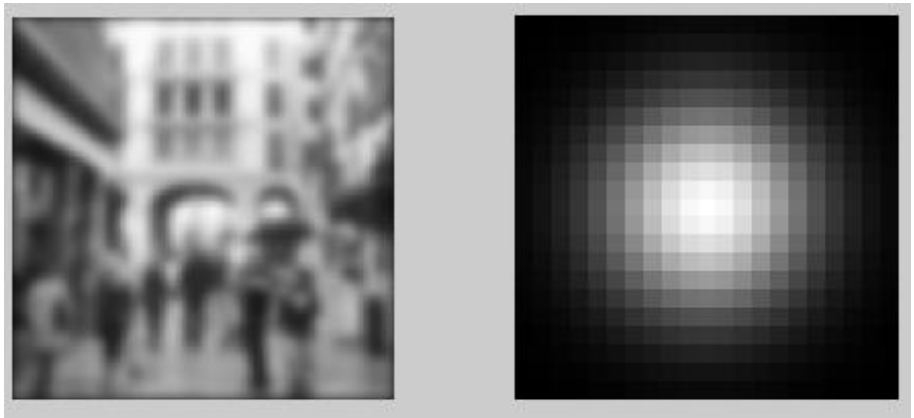
$$G(x, y; \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$



σ=1

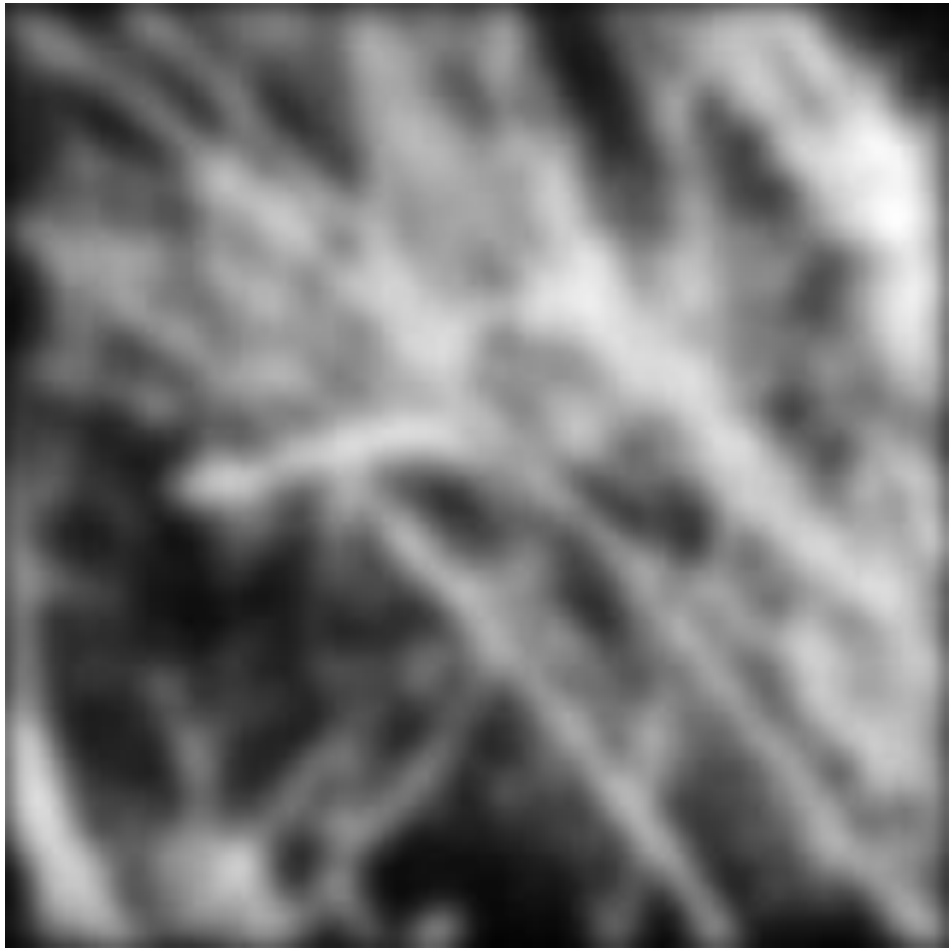


σ=2

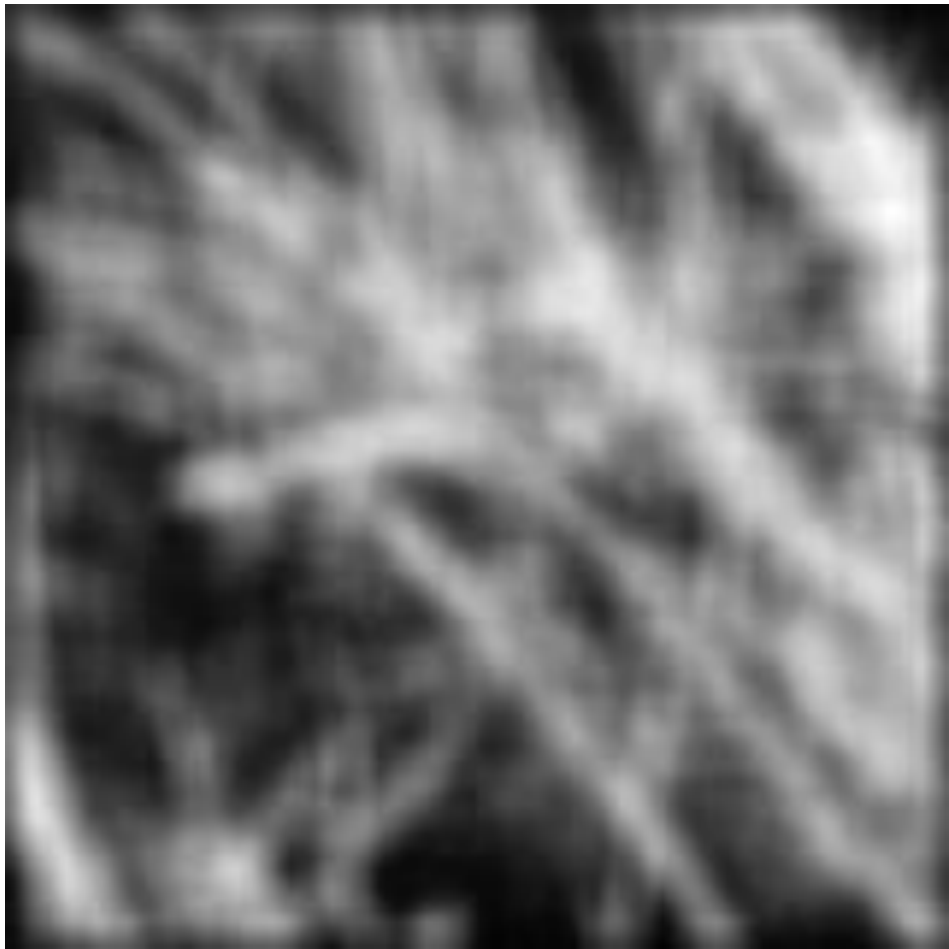


σ=4

Smoothing with Gaussian filter



Smoothing with Box Filter



Properties of Gaussian filter

- Rotational symmetry treats features of all orientations equally (isotropy).

- Efficient: Rule of thumb is kernel width $\geq 5\sigma$
  - Separable
  - Cascadable: Approach to large σ comes from identity

$$G_{\sigma_1} * G_{\sigma_2} = G_{\sqrt{\sigma_1^2+\sigma_2^2}}$$

Separability of the Gaussian filter

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$= \left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{x^2}{2\sigma^2}}\right)\left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{y^2}{2\sigma^2}}\right)$$

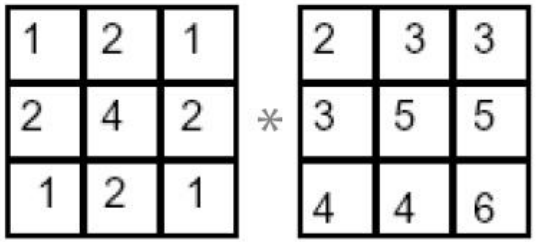
The 2D Gaussian can be expressed as the product of two functions, one a function of x and the other a function of y

In this case, the two functions are the (identical) 1D Gaussian

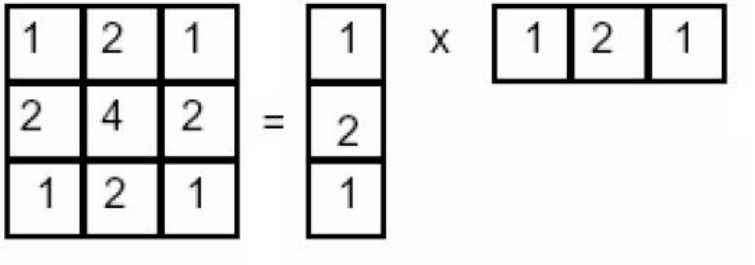
Source: D. Lowe

Separability example

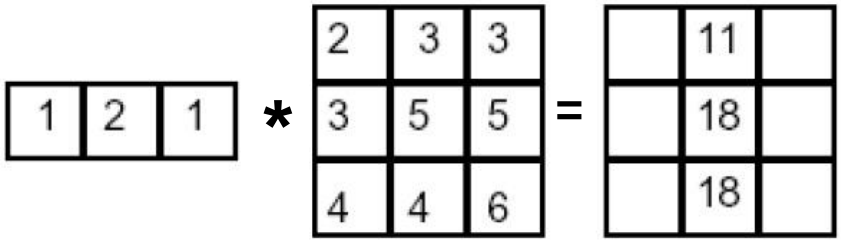
2D convolution
(center location only)



The filter factors
into a product of 1D
filters:



Perform convolution
along rows:



Followed by convolution
along the remaining column:

Source: K. Grauman

Salt-and-Pepper Noise: Gaussian

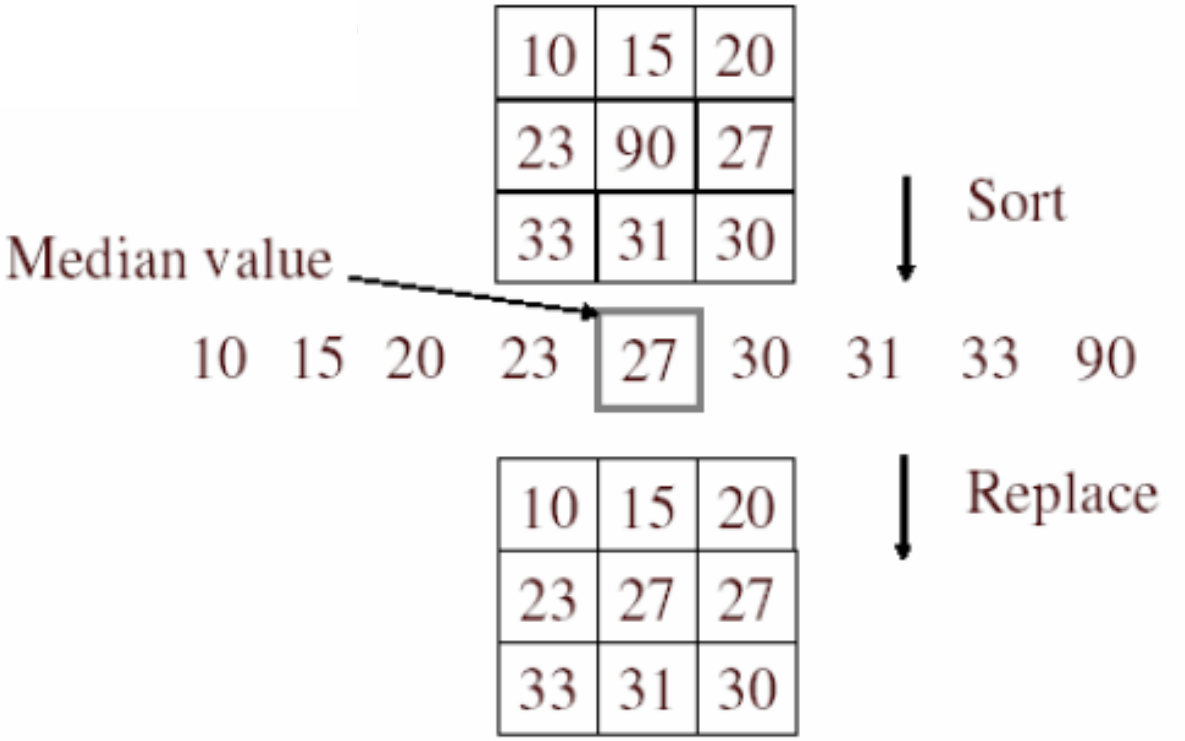
3x3

5x5

7x7

Alternative Idea: Median Filtering

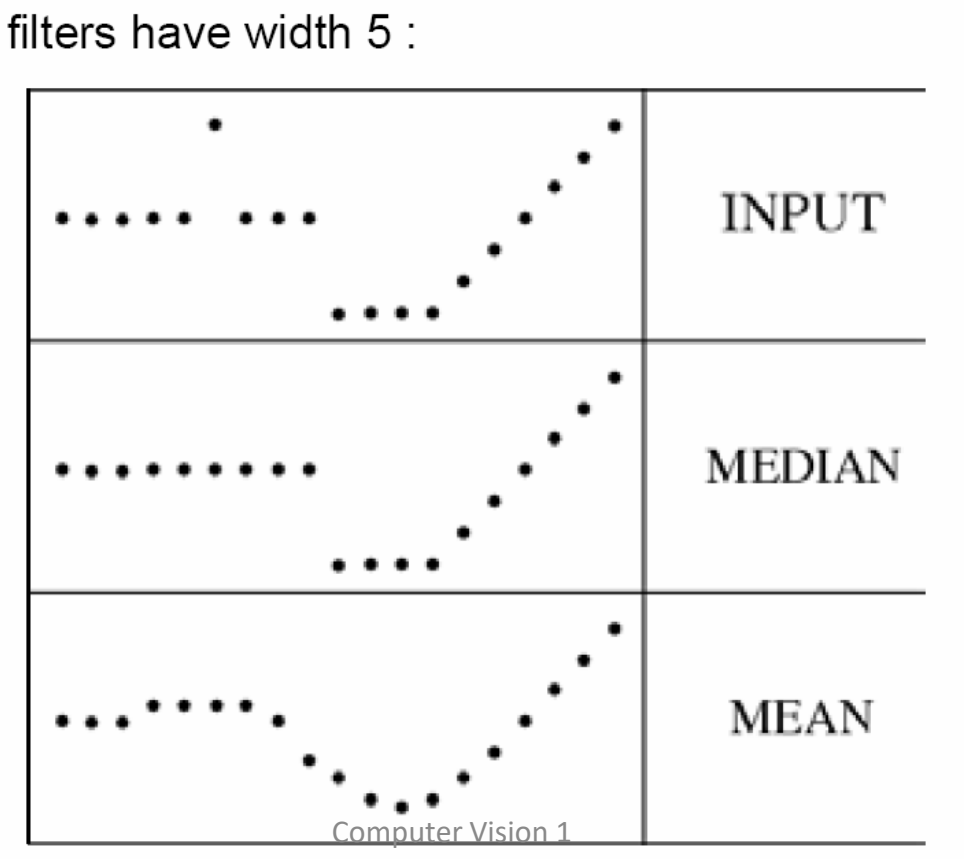
- A **median filter** operates over a window by selecting the median intensity in the window



Source: K. Grauman

Median Filter

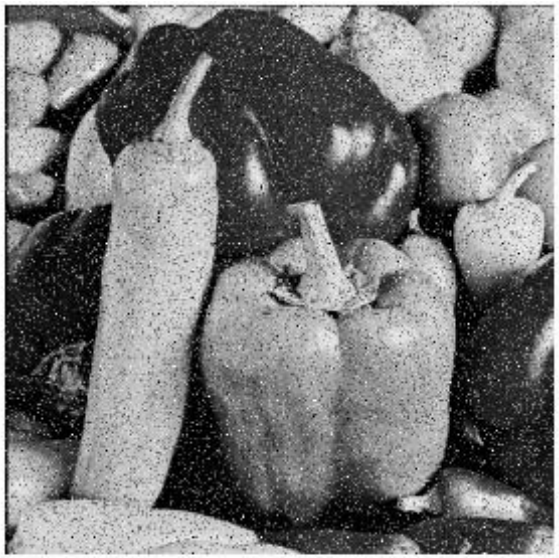
- What advantage does median filtering have over Gaussian filtering?
  - Robustness to outliers

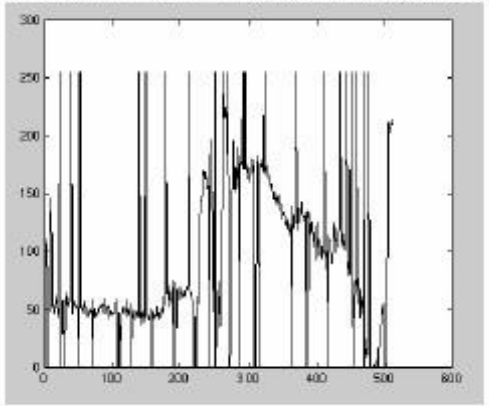


Source: K. Grauman

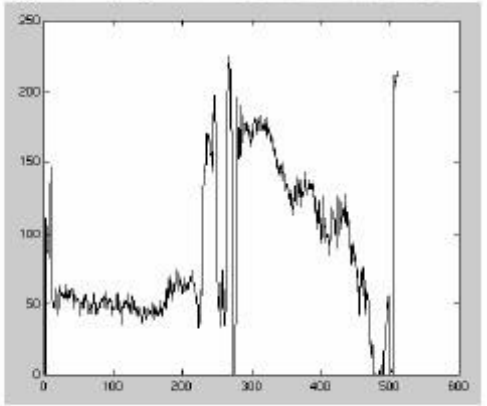
Median Filter

Salt-and-pepper noise



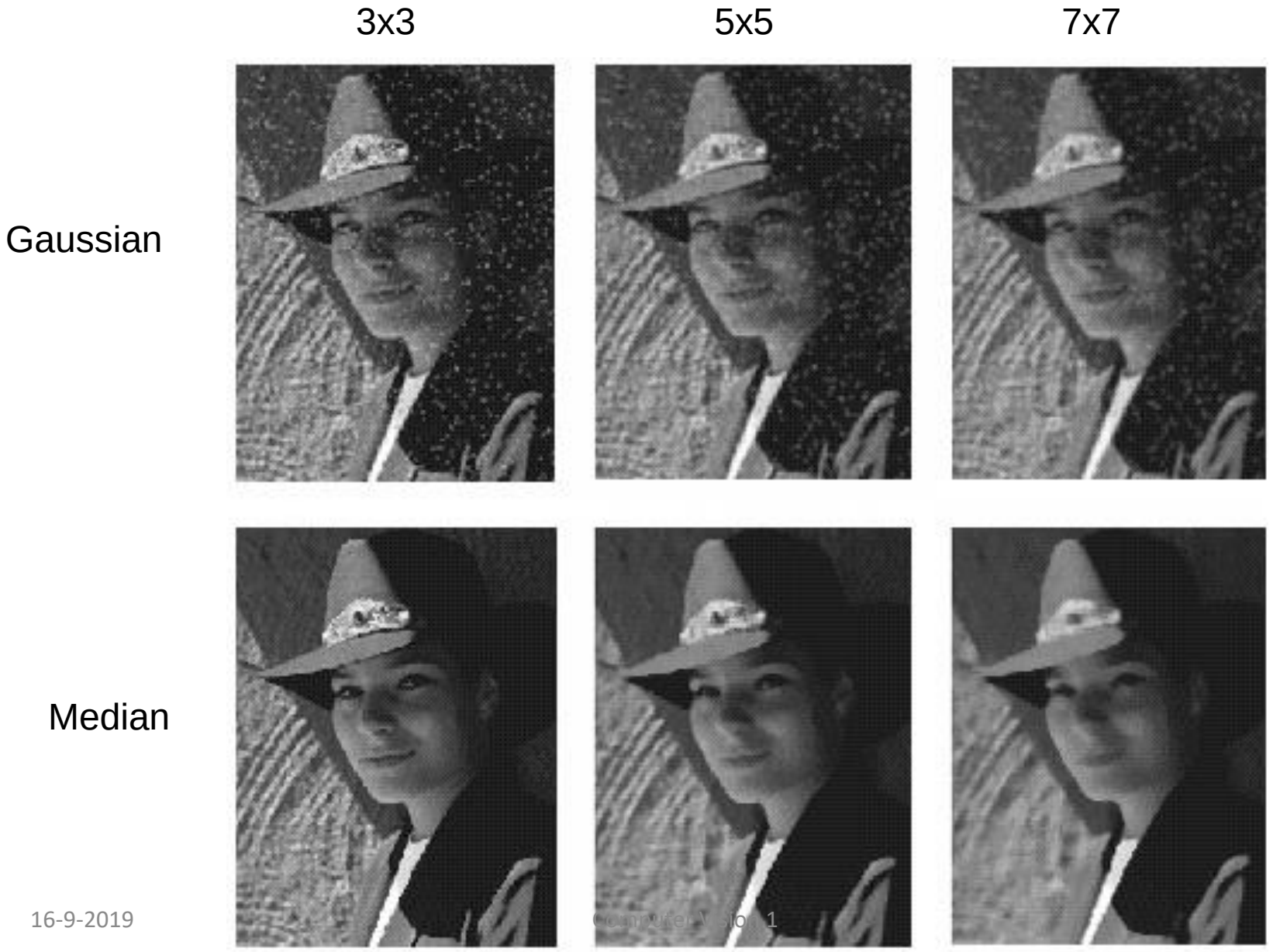


Median filtered



Source: M. Hebert

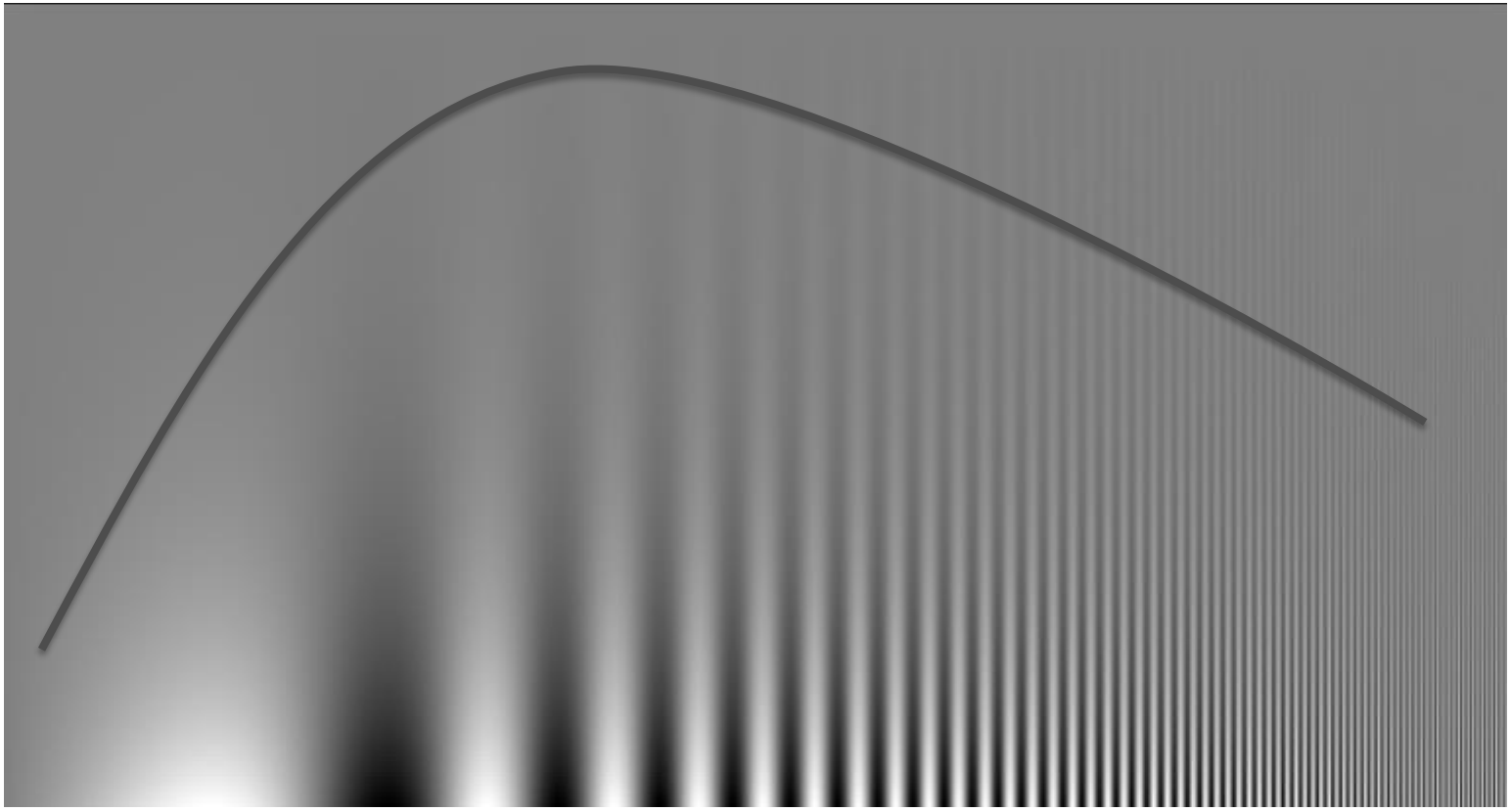
Median vs. Gaussian Filtering



Today's Class

1. Reflection Models
2. Image Processing
   - Pixel and Neighbourhood processing
     - Grey image processing
     - Colour correction
   - Image Filtering
   - Edge Detection
   - Corner detection
     - Harris corner detector
     - SIFT
   - Applications

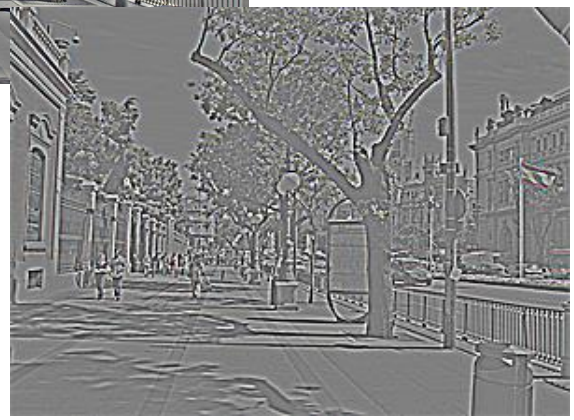
Robson-Campbell Curve



Human Visual Perception

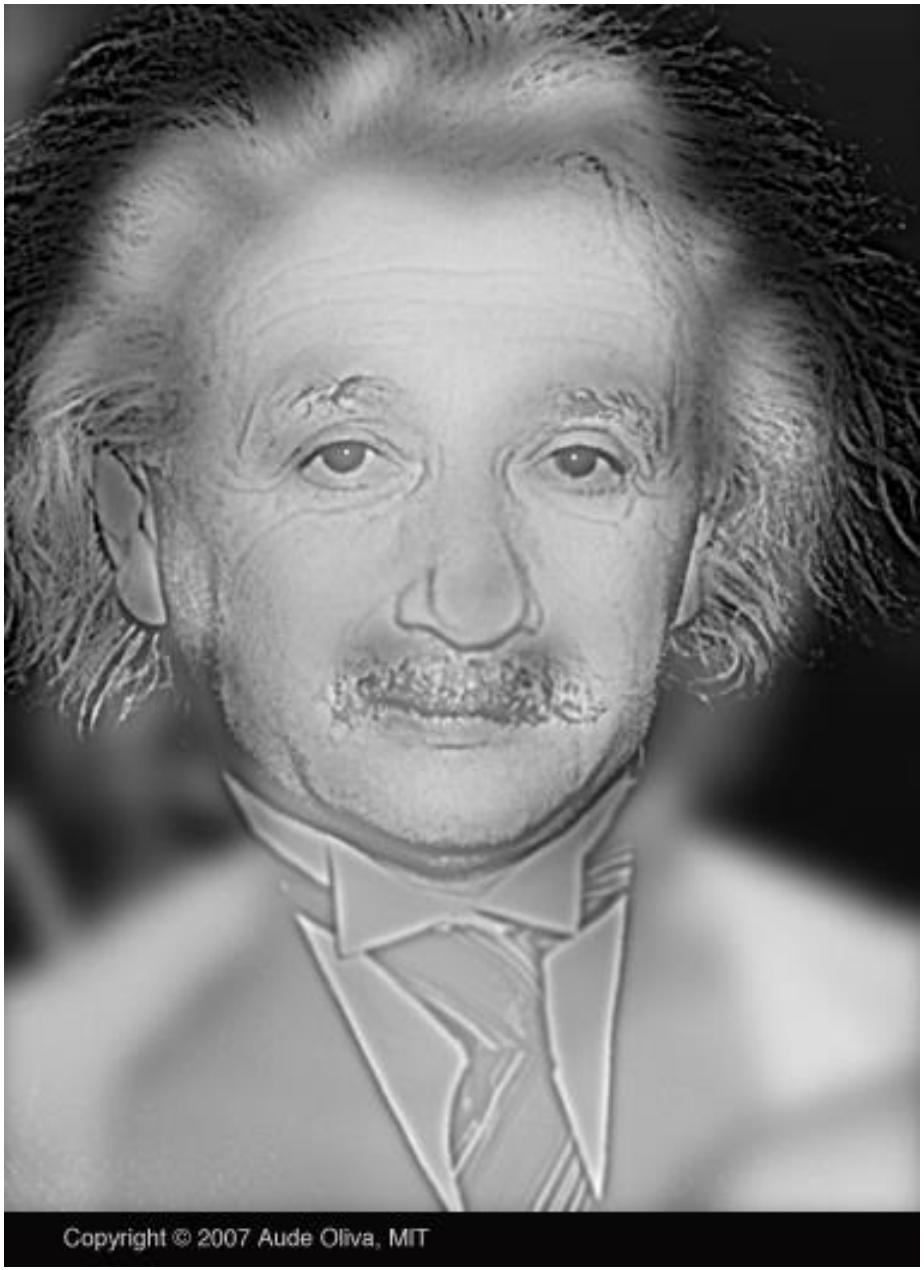
Low spatial frequency

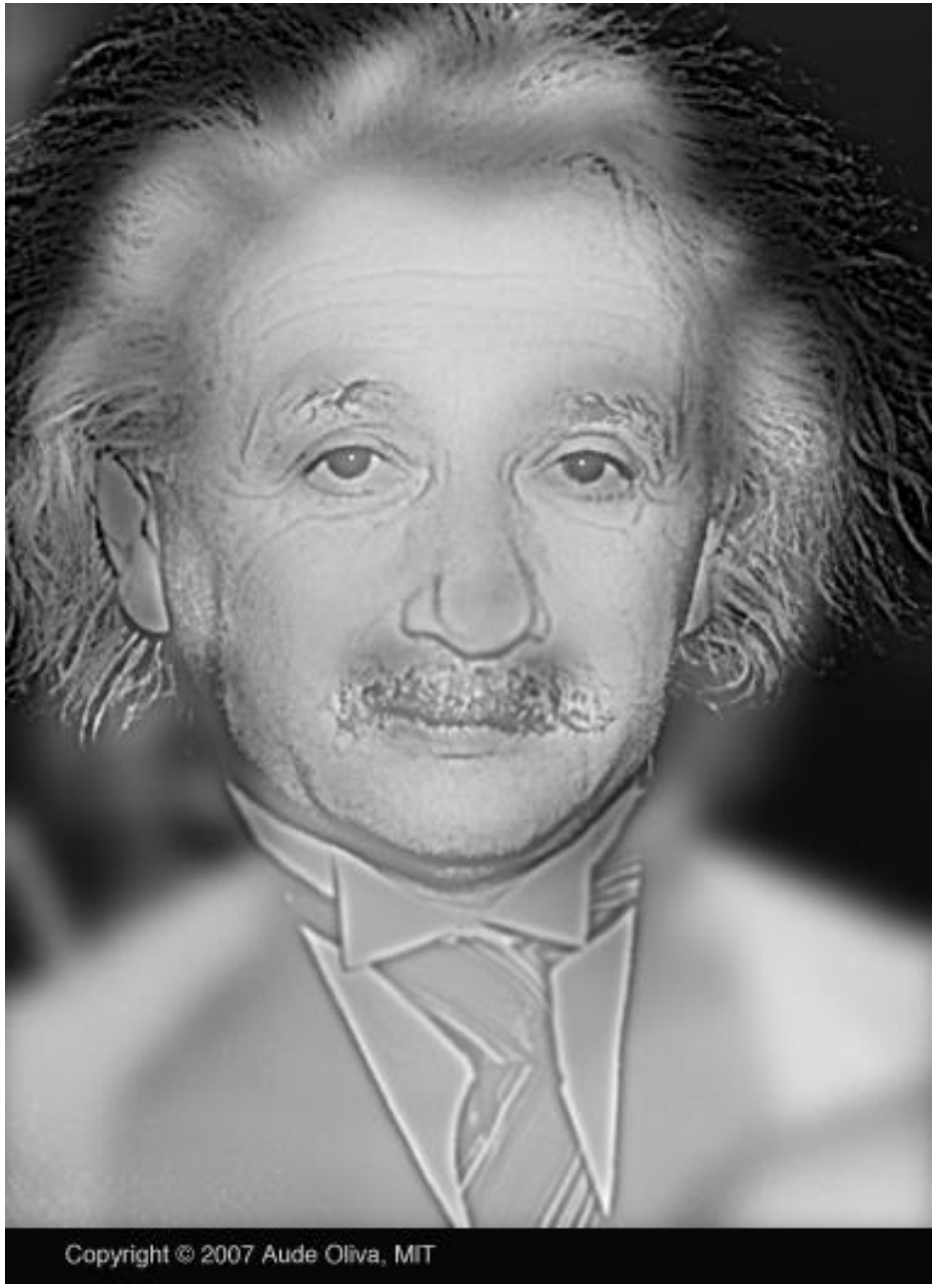
Medium spatial frequency

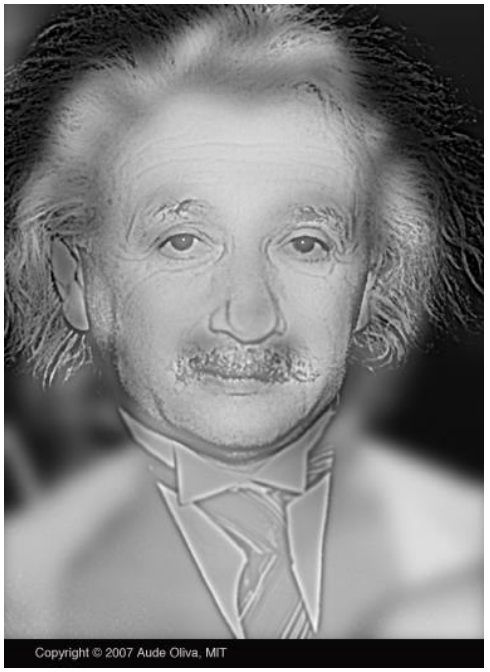


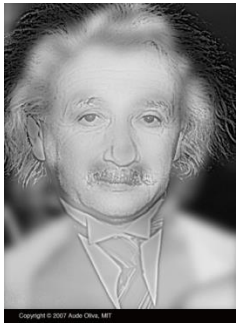
High spatial frequency

Why edge is important





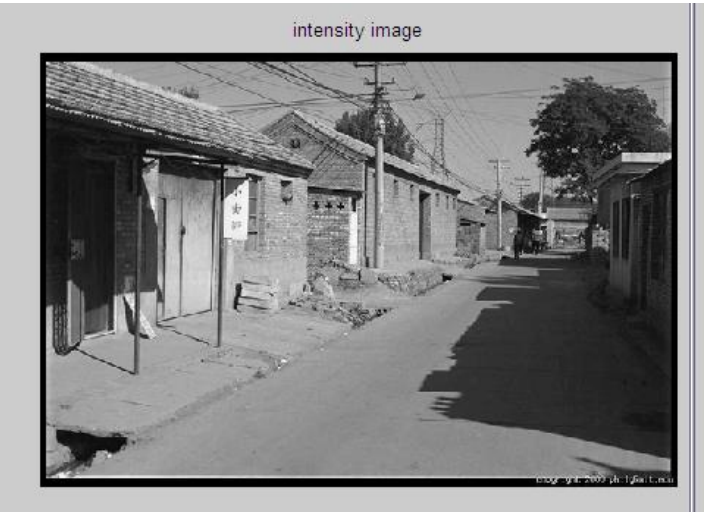


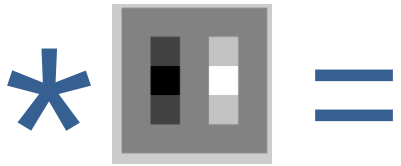


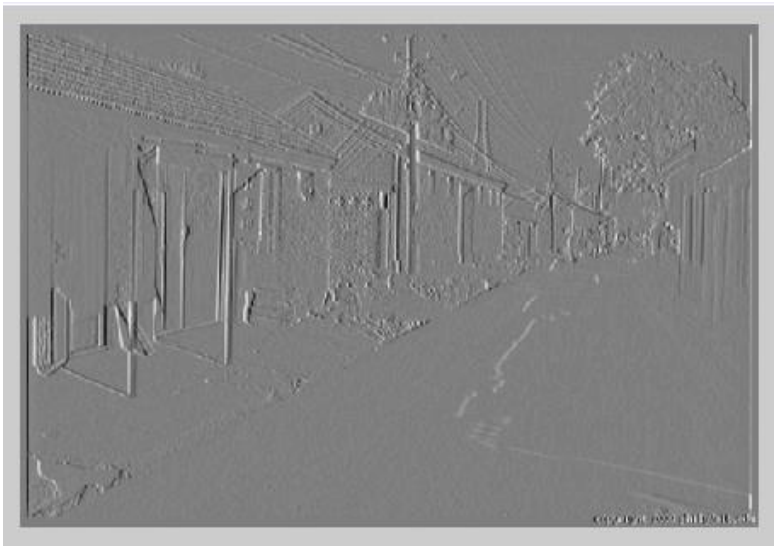
http://cvcl.mit.edu/hybrid_gallery/gallery.html

Review: Edge Detection

1	0	-1
2	0	-2
1	0	-1







1st and 2nd Derivative Operations

- For a 2D function, the first derivative and its kernel are

$$\frac{\partial f(x,y)}{\partial x} \cong f(x,y) - f(x-1,y)$$

-1	1

- The second derivative and its kernel become

$$\frac{\partial^2 f(x,y)}{\partial x^2} \cong [f(x+1,y) - f(x,y)] - [f(x,y) - f(x-1,y)]$$

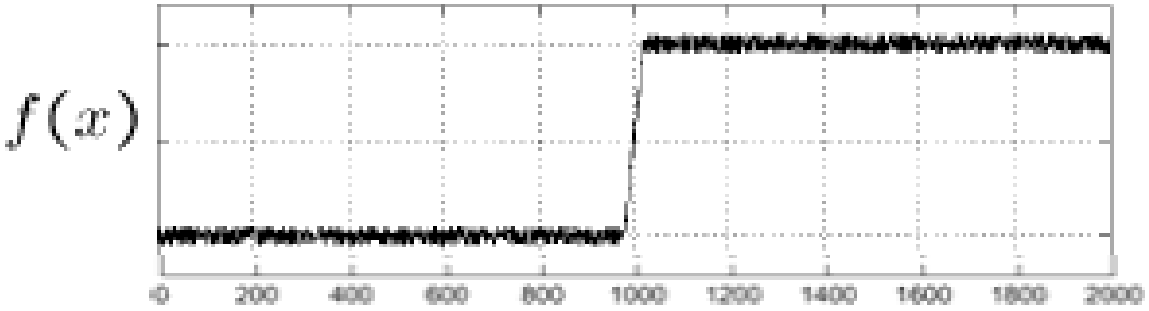
$$= f(x+1,y) - 2f(x,y) + f(x-1,y)$$

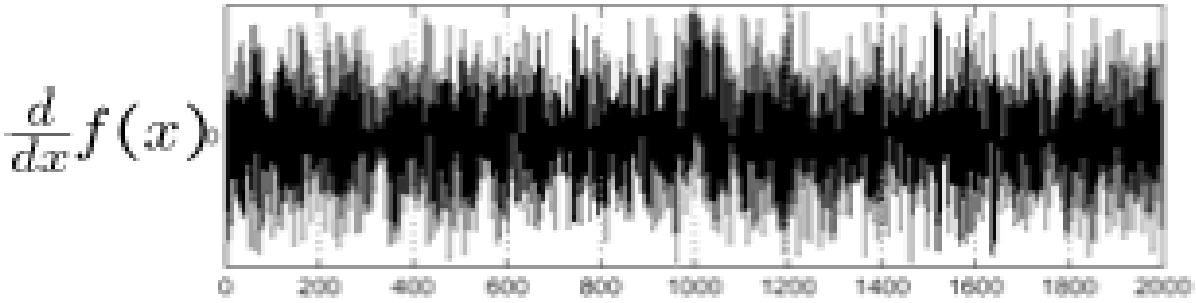
1	-2	1

Problem with noisy image

Consider a single row or column of the image

- Plotting intensity as a function of position gives a signal





Where is the edge?

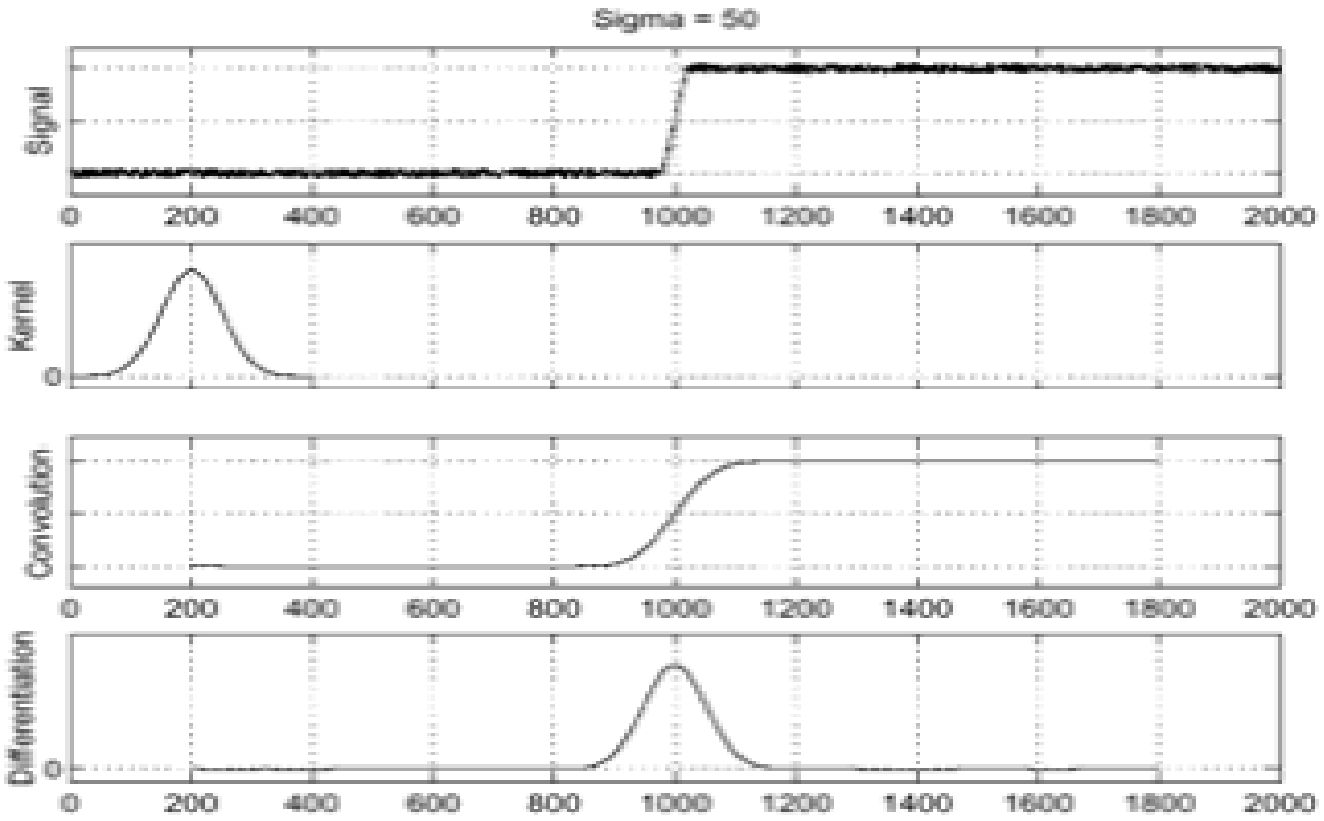
Solution: smooth the image first

f

h

$h \star f$

$\frac{\partial}{\partial x}(h \star f)$



Where is the edge?

Look for peaks in $\frac{\partial}{\partial x}(h \star f)$

Derivative of Gaussian filter

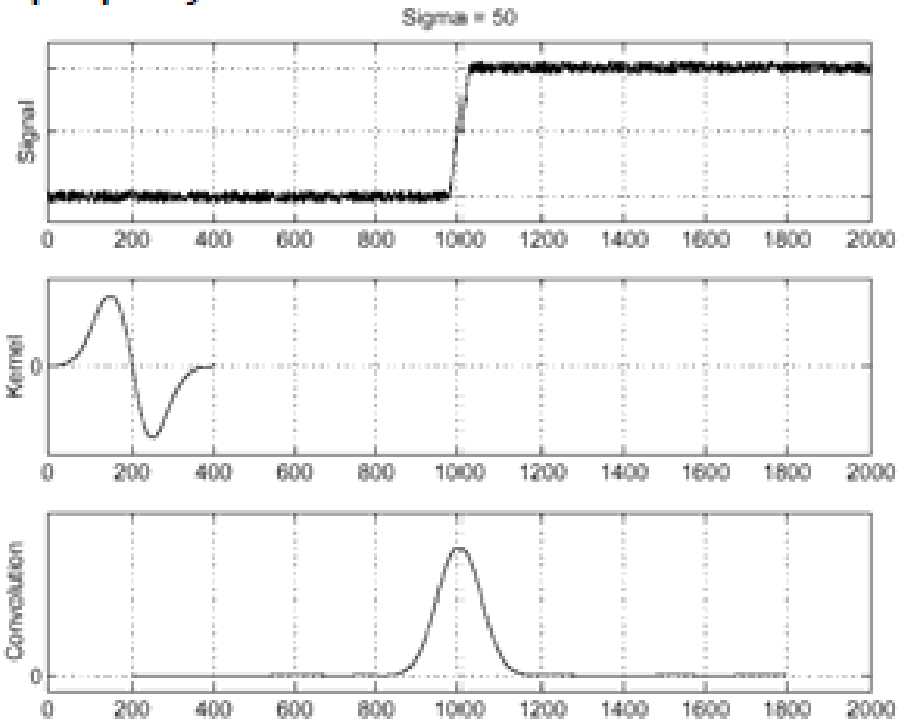
$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$$

Differentiation property of convolution.

f

$\frac{\partial}{\partial x}h$

$(\frac{\partial}{\partial x}h) \star f$



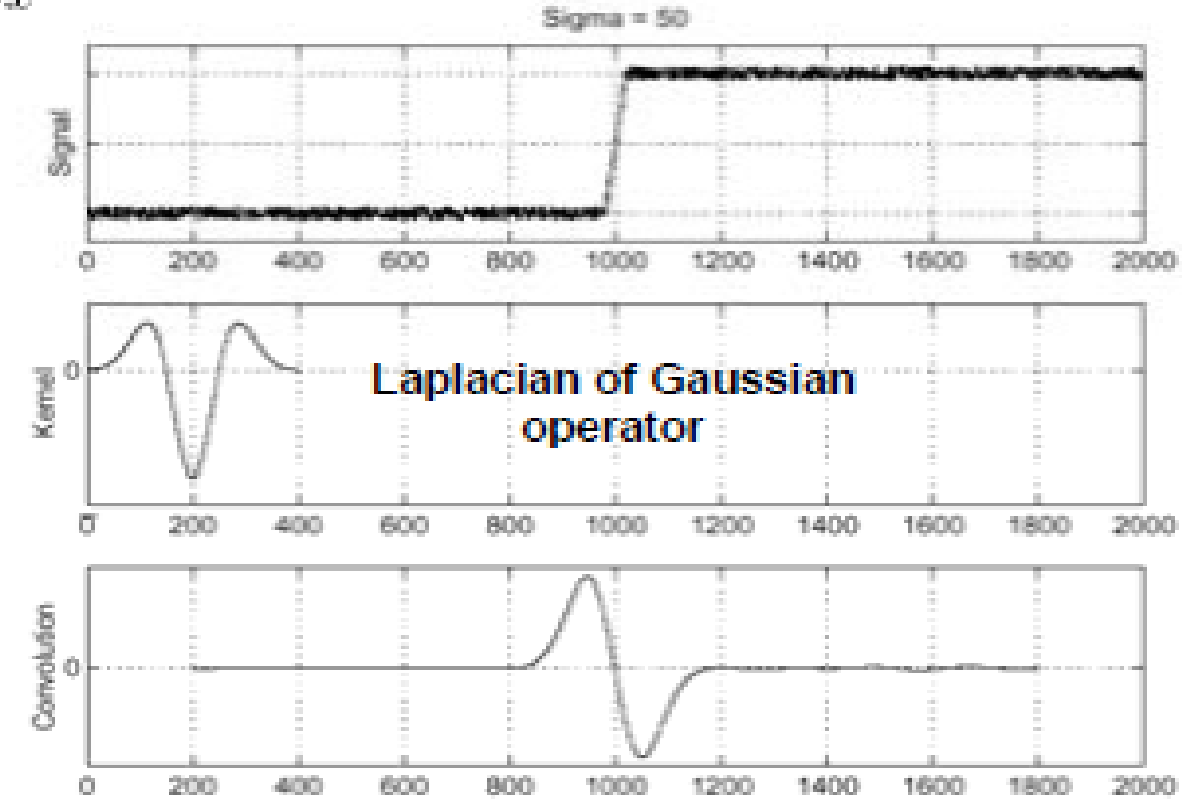
Edge as the zero-crossing of the second order derivative using Laplacian of Gaussian

Consider $\frac{\partial^2}{\partial x^2}(h \star f)$

f

$\frac{\partial^2}{\partial x^2}h$

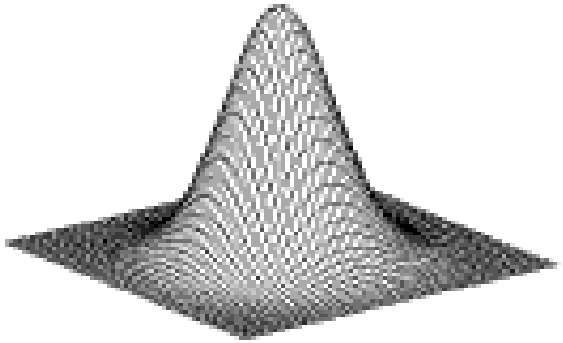
$(\frac{\partial^2}{\partial x^2}h) \star f$



Where is the edge?

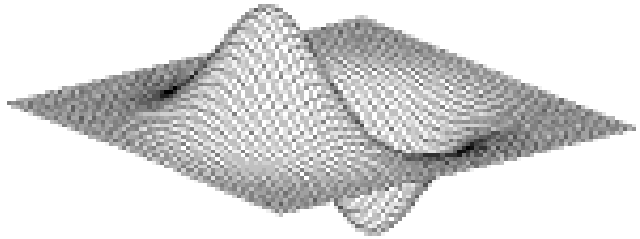
Zero-crossings of bottom graph

2D edge detection filters



Gaussian

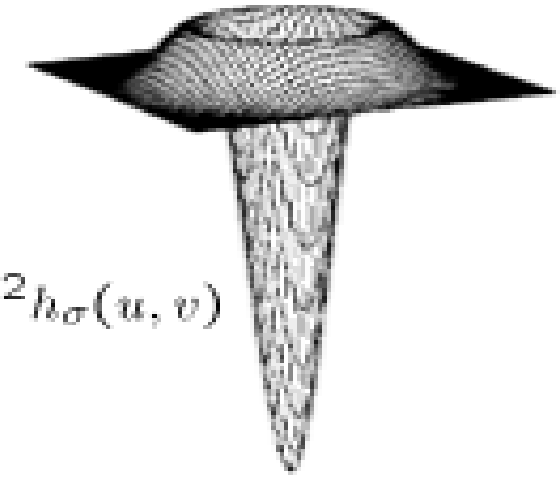
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



derivative of Gaussian

$$\frac{\partial}{\partial x} h_\sigma(u,v)$$

Laplacian of Gaussian



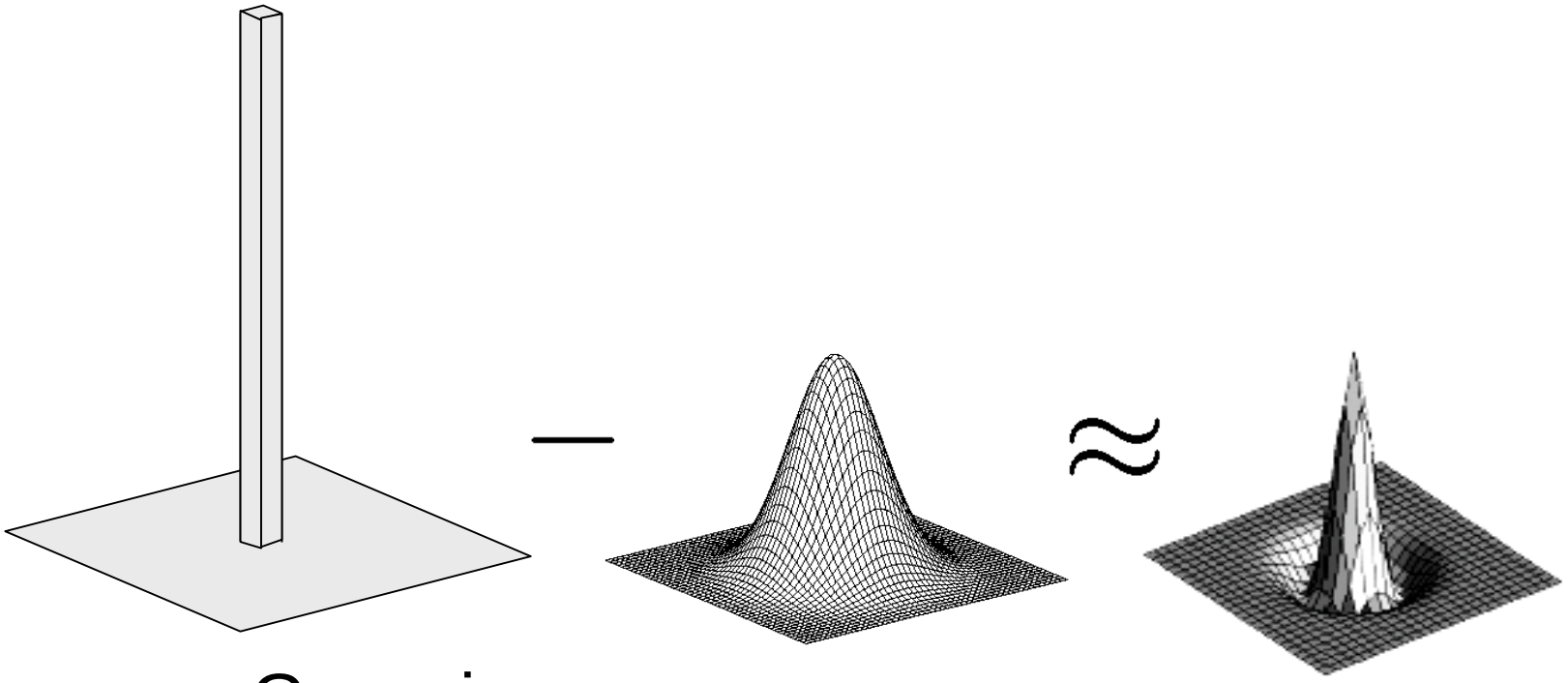
$$\nabla^2 h_\sigma(u,v)$$

- **∇^2 is the Laplacian operator:**

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Difference of Gaussians (DoG)

- Laplacian of Gaussian (LoG) can be approximately computed as Difference of Gaussians (DoG).



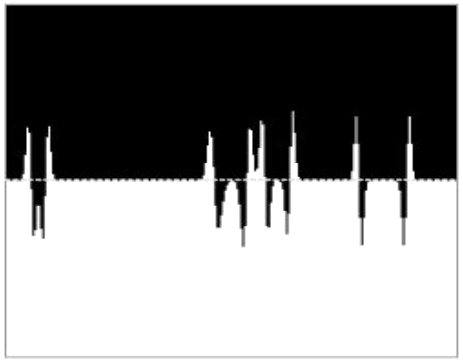
A narrow Gaussian (or an impulse function)

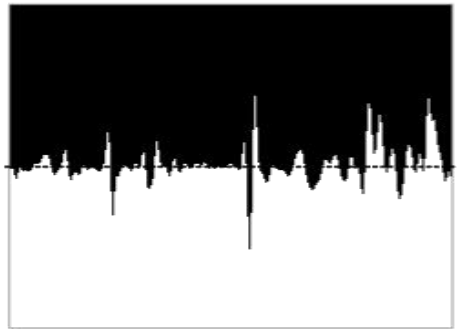
A wide Gaussian

Laplacian of Gaussian

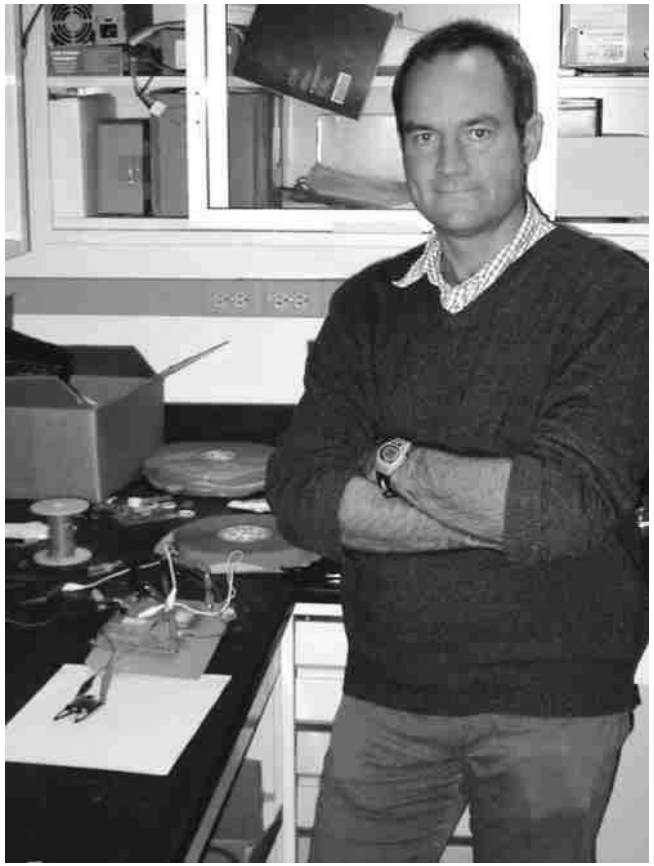
Example: LoG filtering

Note that LoG output has negative values which need offset (e.g. add 128) and scaling

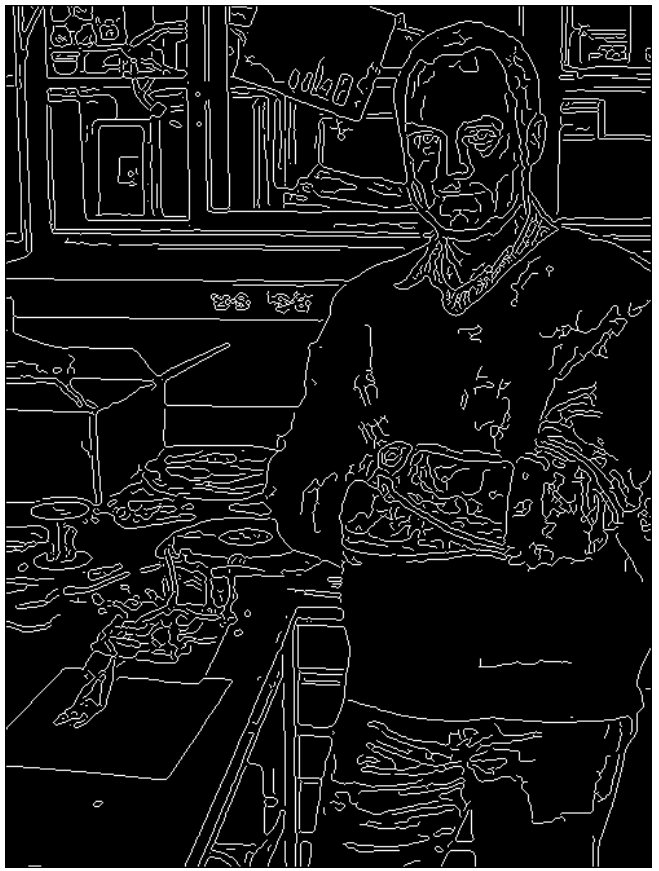




Canny Edge Detector



John Canny



Canny Edge Detector is one of the most successful edge detection method.

Ref: John F. Canny, "A computational approach to edge detection," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 8, No. 16, pp. 679-698, Nov. 1986.

Example: Canny edge detector

Original image (Lena)

Norm of the gradient

The Canny edge detector

Thresholding

Thinning (non-maximum suppression, edge following)

Today's Class

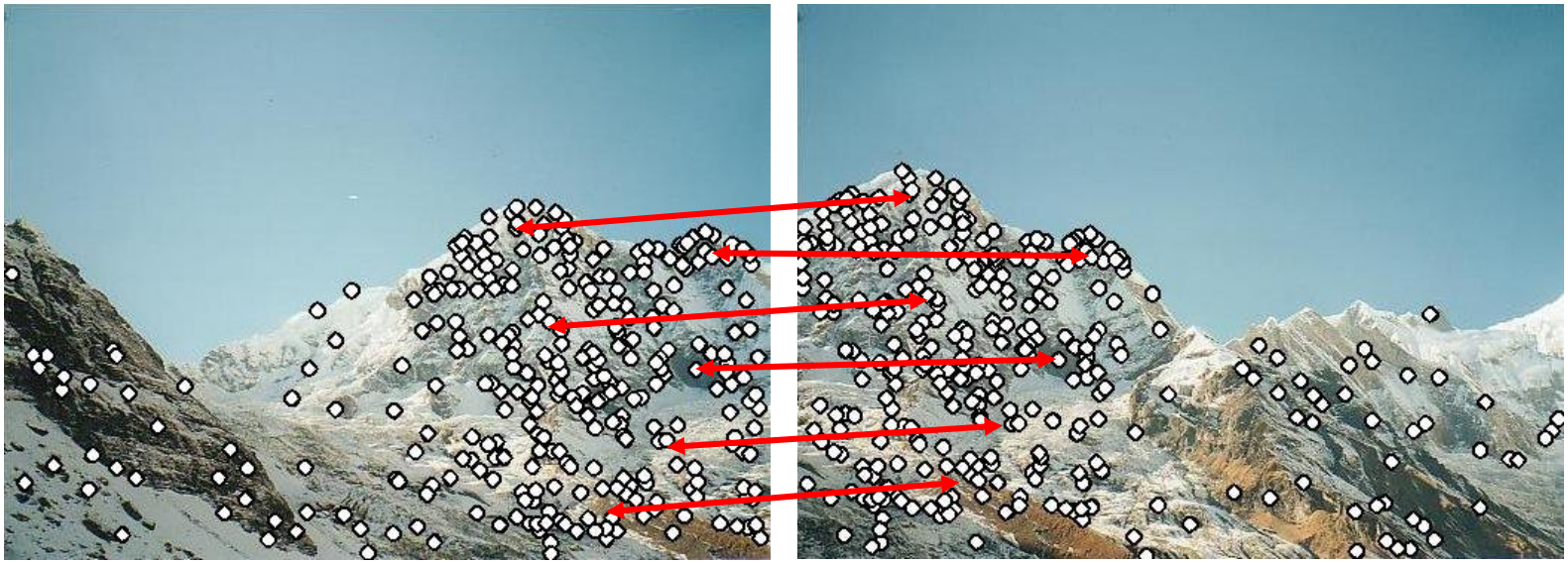
1. Reflection Models
2. Image Processing
   - Pixel and Neighbourhood processing
     - Grey image processing
     - Colour correction
   - Image Filtering
   - Edge Detection
   - Corner detection
     - Harris corner detector
     - SIFT
   - Applications

Why extract features?

- Motivation: panorama stitching
  - We have two images – how do we combine them?

Why extract corner features?

- Motivation: panorama stitching
  - We have two images – how do we combine them?



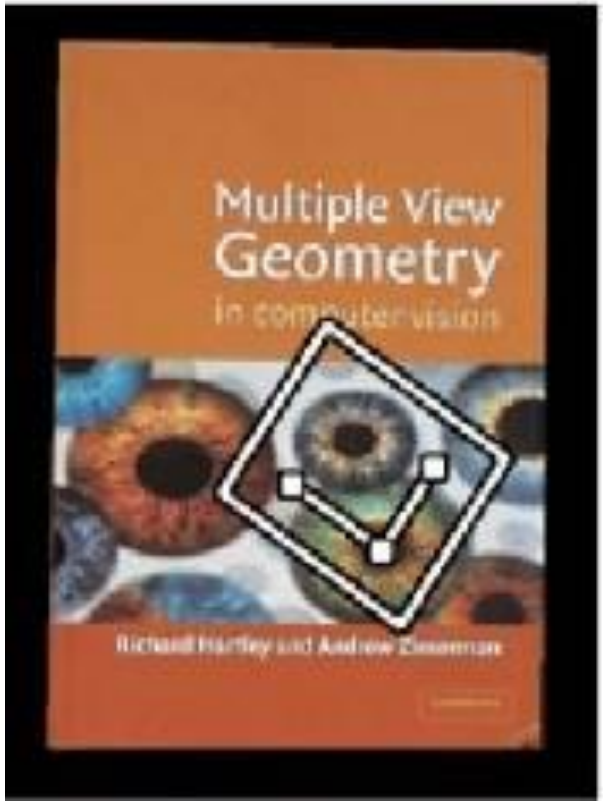
Step 1: extract features
Step 2: match features

Why extract features?

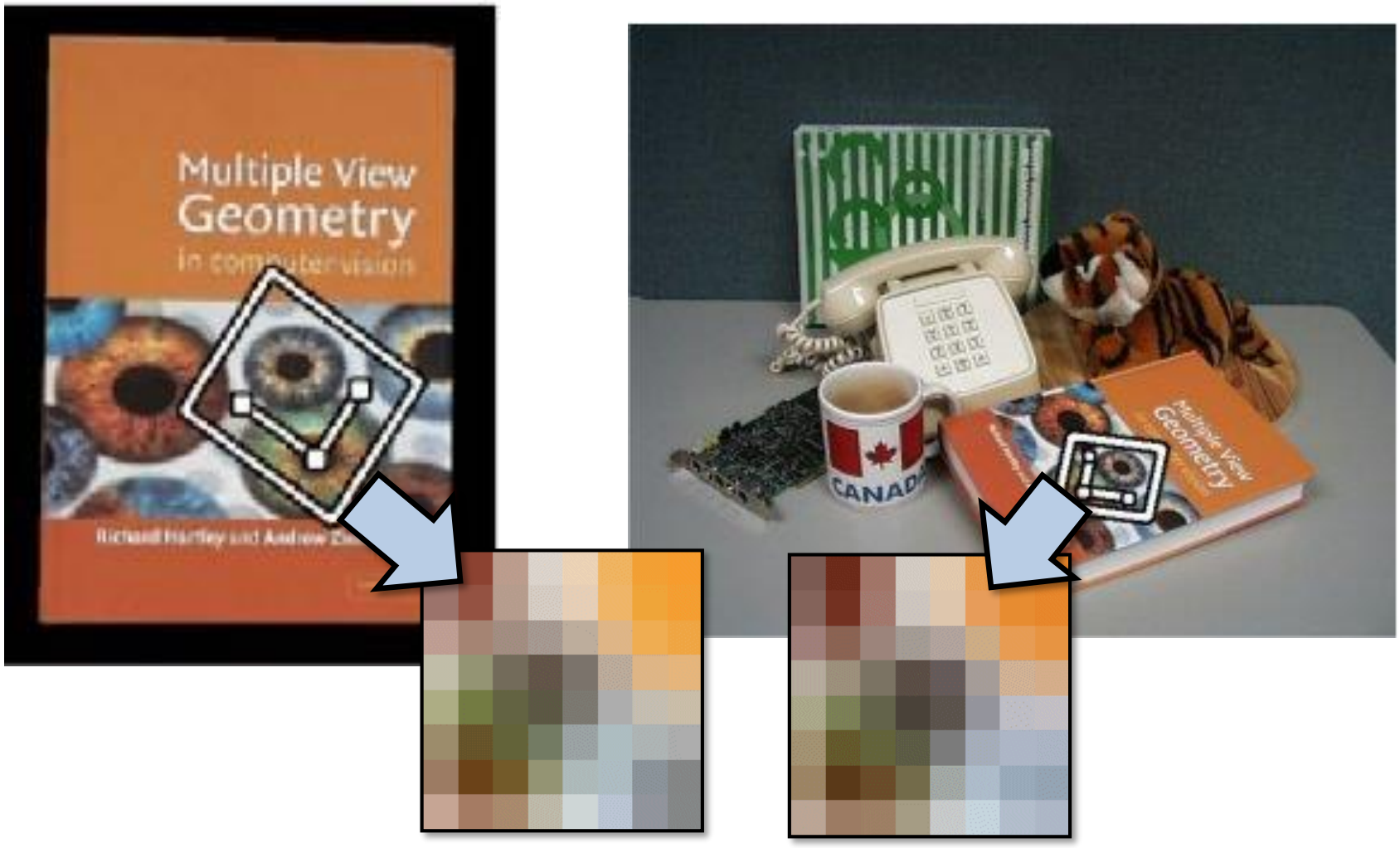
- Motivation: panorama stitching
  - We have two images – how do we combine them?

Step 1: extract features
Step 2: match features
Step 3: align images

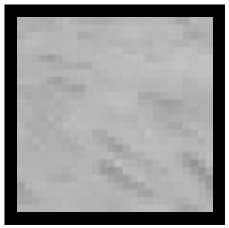
Feature Matching



Feature Matching

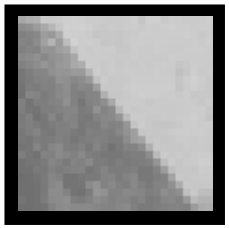


Interest Points



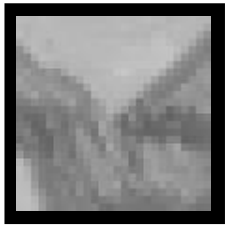
0D structure: single points

➡ not useful for matching



1D structure: lines

➡ edge, can be localised in 1D, subject to the aperture problem



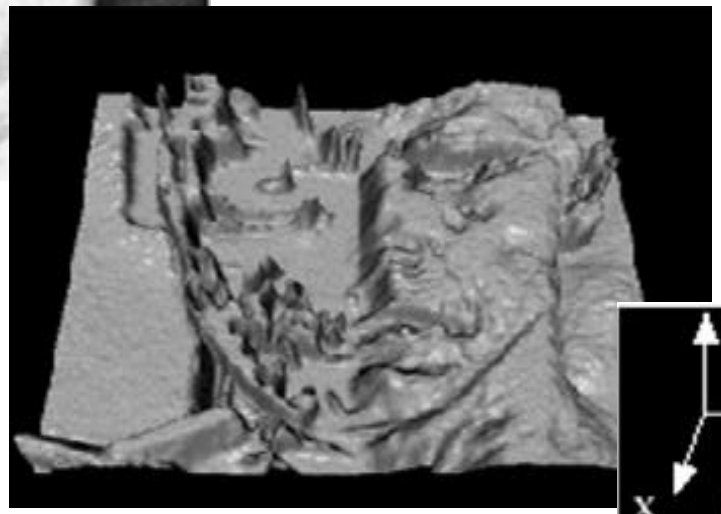
2D structure: corners

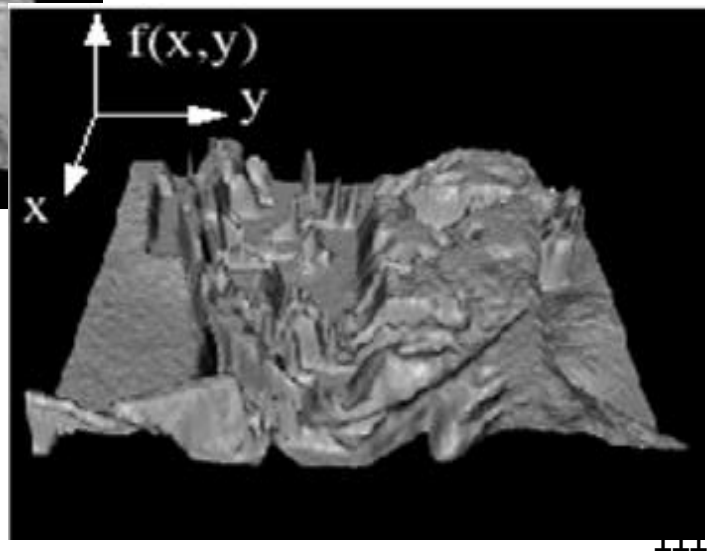
➡ corner, or **interest point**, can be localised in 2D, good for matching

Interest Points have **2D** structure.

Lets do some deferentaion on 2D surface:

I = f(x,y)





Harris Detector: Mathematics

Change of intensity for the shift [*u,v*]:

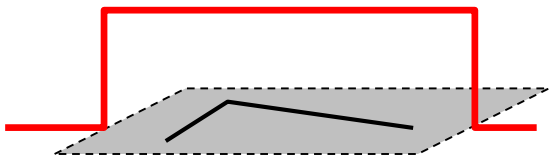
$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u, y+v) - I(x,y)\right]^2$$

Window function

Shifted intensity

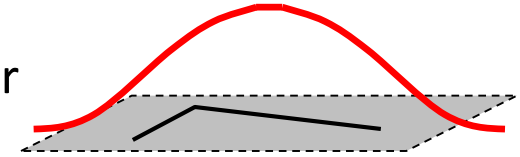
Intensity

Window function $w(x,y)$ =



1 in window, 0 outside

or



Gaussian

Measuring the "Properties" of E()

$$E(u,v) = \sum \left[I(x-u, y-v) - I(x,y) \right]^2 =$$

$$= \sum_{x,y} \left[I(x,y) - I_x u - I_y v + \varepsilon - I(x,y) \right]^2 \approx$$

$$\approx \sum_{x,y} \left[- I_x u - I_y v \right]^2 =$$

$$= \sum_{x,y} I_x{}^2 u^2 + I_x u I_y v + I_y v I_x u + I_y{}^2 v^2 =$$

$$= \sum_{x,y} \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} I_x{}^2 & I_x I_y \\ I_y I_x & I_y{}^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

$$E(u,v) \cong \begin{bmatrix} u, v \end{bmatrix} \quad M \quad \begin{bmatrix} u \\ v \end{bmatrix}$$

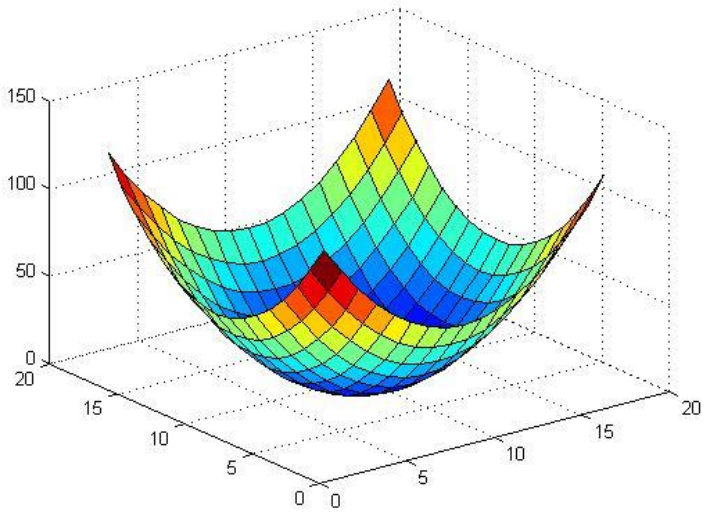
$$M = \sum_{x,y} \begin{bmatrix} I_x{}^2 & I_x I_y \\ I_y I_x & I_y{}^2 \end{bmatrix}$$

M depends on image properties

Quadratic Form and its Eigenvalue

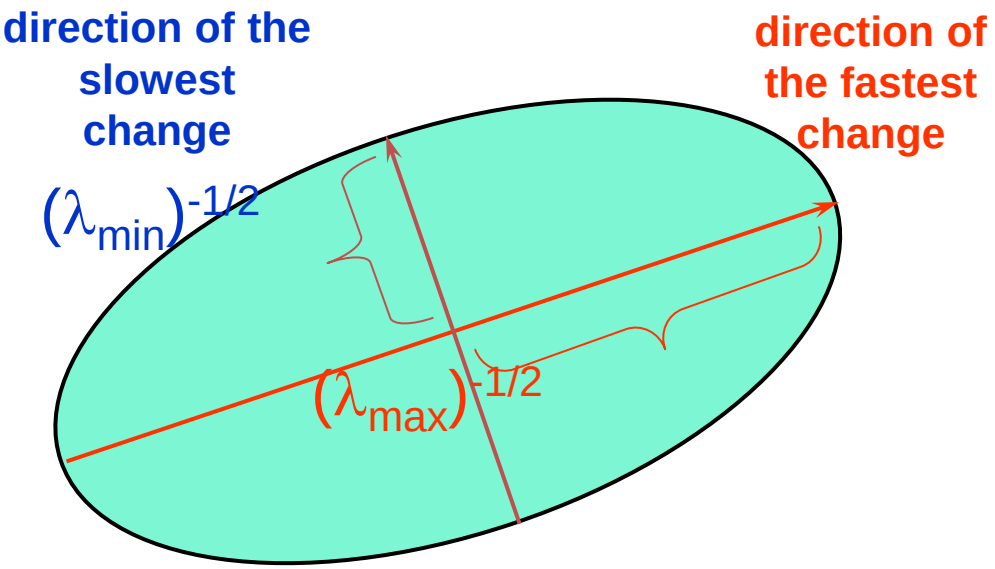
$$E(u,v) \cong \begin{bmatrix} u, v \end{bmatrix} \quad M \quad \begin{bmatrix} u \\ v \end{bmatrix}$$

$$M = U^T \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} U =$$

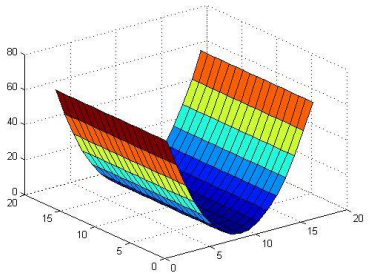


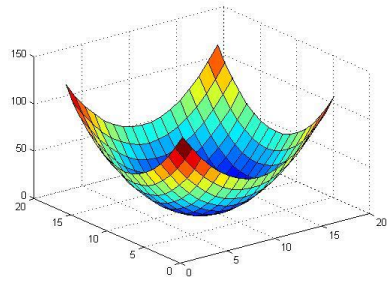
λ_1, λ_2 – eigenvalues of M

Ellipse $E(u,v)$ = const



Classification of Image Points using Eigenvalues of *M*:



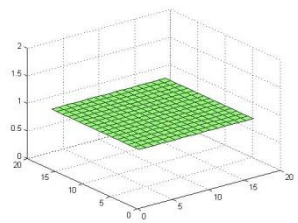


λ_2

"Edge"
$\lambda_2 >> \lambda_1$

"Corner"
λ_1 and λ_2 are large,
$\lambda_1 \sim \lambda_2$;
E increases in all directions

λ_1 and λ_2 are small;
E is almost constant in all directions



"Flat" region

"Edge"
$\lambda_1 >> \lambda_2$

λ_1

Harris Detector: Mathematics

Measure of corner response:

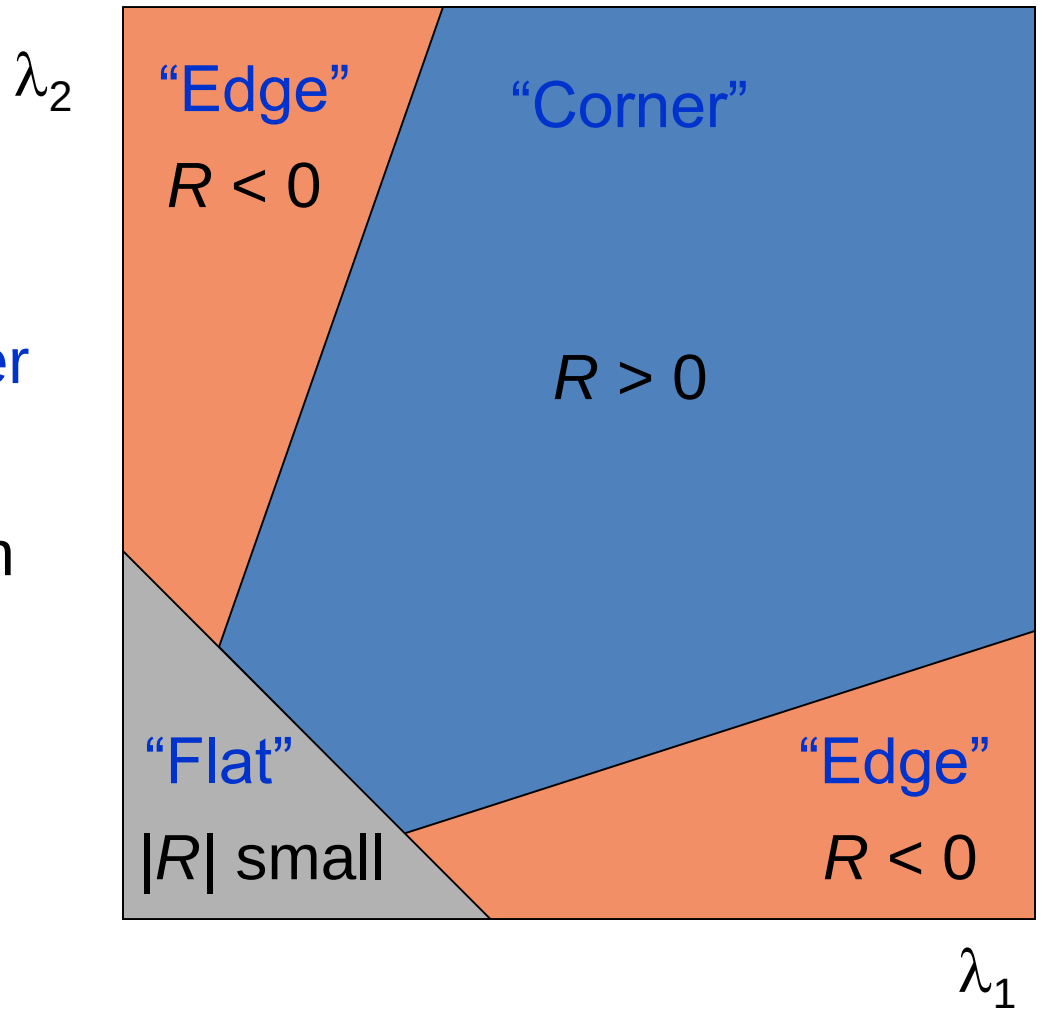
$$R = \det M - k\left(\operatorname{trace} M\right)^2$$

$$\det M = \lambda_1 \lambda_2$$
$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

(k – empirical constant, k = 0.04-0.06)

Harris Detector

- *R* depends only on eigenvalues of M
- *R* is large for a corner
- *R* is negative with large magnitude for an edge
- |*R*| is small for a flat region



$$R(x_0, y_0) = \det M - k\left(\operatorname{trace} M\right)^2$$

Summary of the Harris detector

1. Compute x and y derivatives of image

$$I_x = G_\sigma^x * I \quad I_y = G_\sigma^y * I$$

2. Compute products of derivatives at every pixel

$$I_{x2} = I_x.I_x \quad I_{y2} = I_y.I_y \quad I_{xy} = I_x.I_y$$

3. Compute the sums of the products of derivatives at each pixel

$$S_{x2} = G_{\sigma\prime} * I_{x2} \quad S_{y2} = G_{\sigma\prime} * I_{y2} \quad S_{xy} = G_{\sigma\prime} * I_{xy}$$

4. Define at each pixel (x, y) the matrix

$$H(x,y) = \begin{bmatrix} S_{x2}(x,y) & S_{xy}(x,y) \\ S_{xy}(x,y) & S_{y2}(x,y) \end{bmatrix}$$

5. Compute the response of the detector at each pixel

$$R = Det(H) - k(Trace(H))^2$$

6. Threshold on value of R. Compute nonmax suppression.

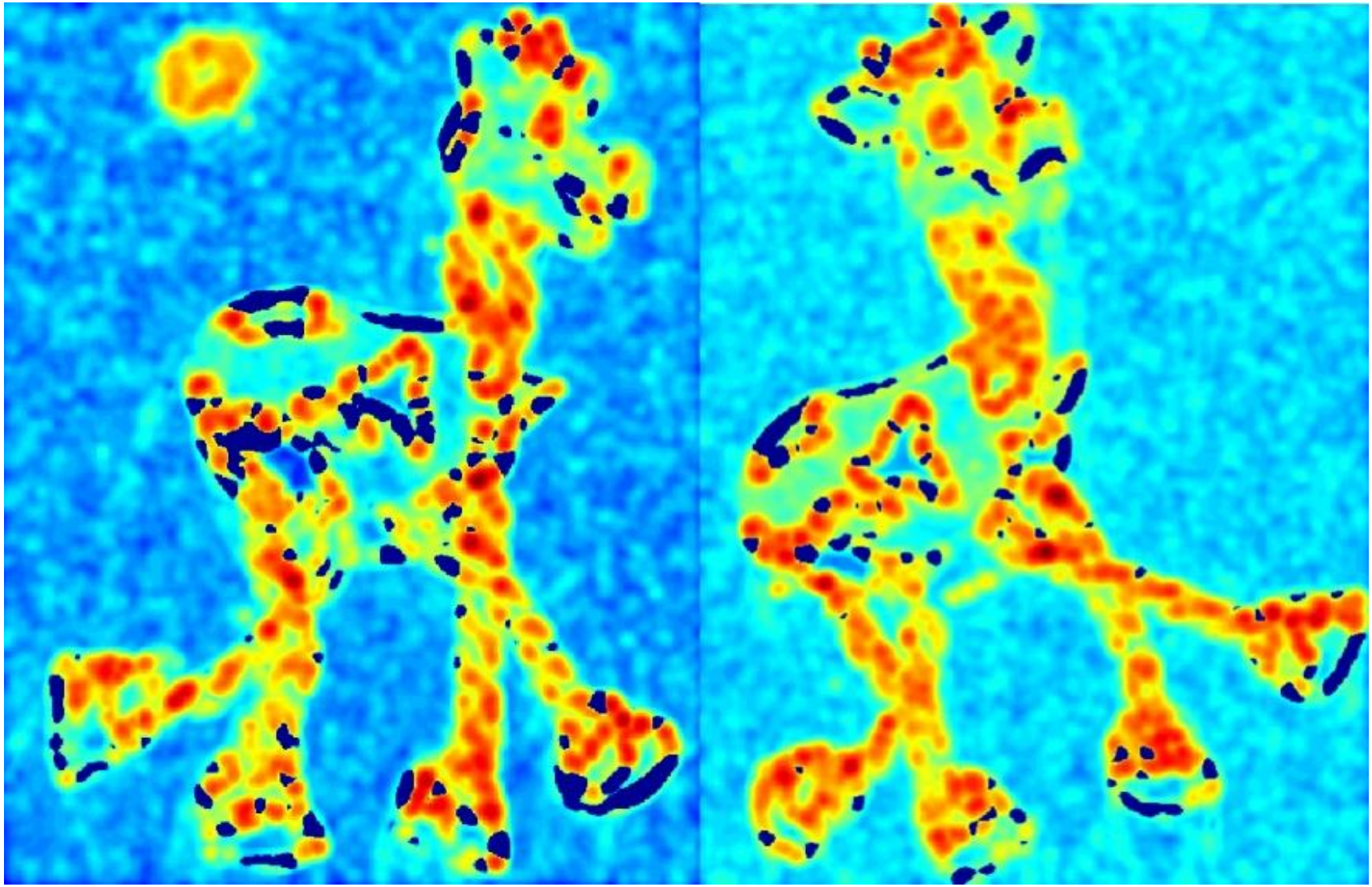
Harris Detector

- The Algorithm:
  - Find points with large corner response function R ($R >$ threshold)
  - Take the points of local maxima of R

Harris Detector: Workflow

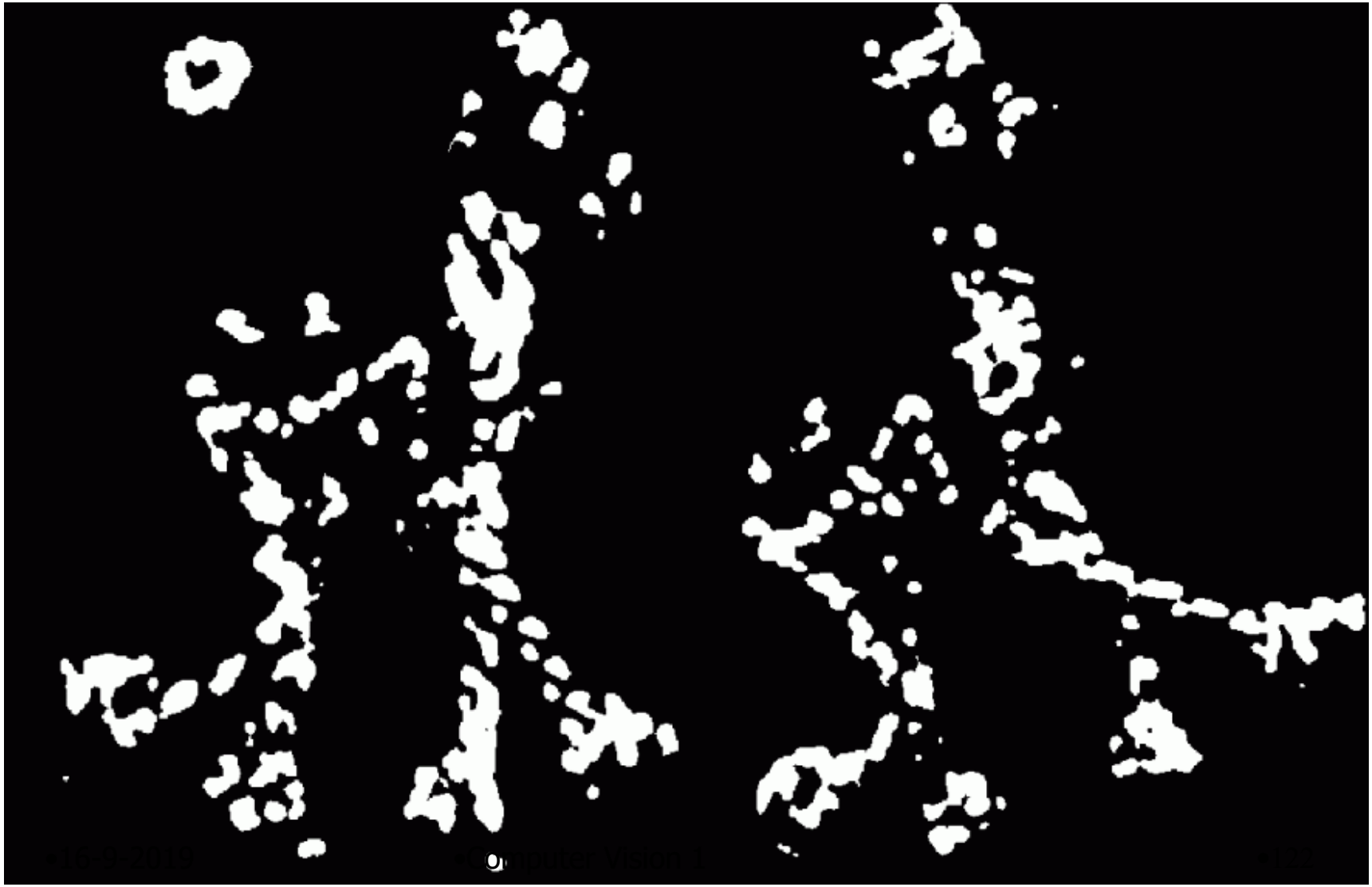
Harris Detector: Workflow

Compute corner response *R*



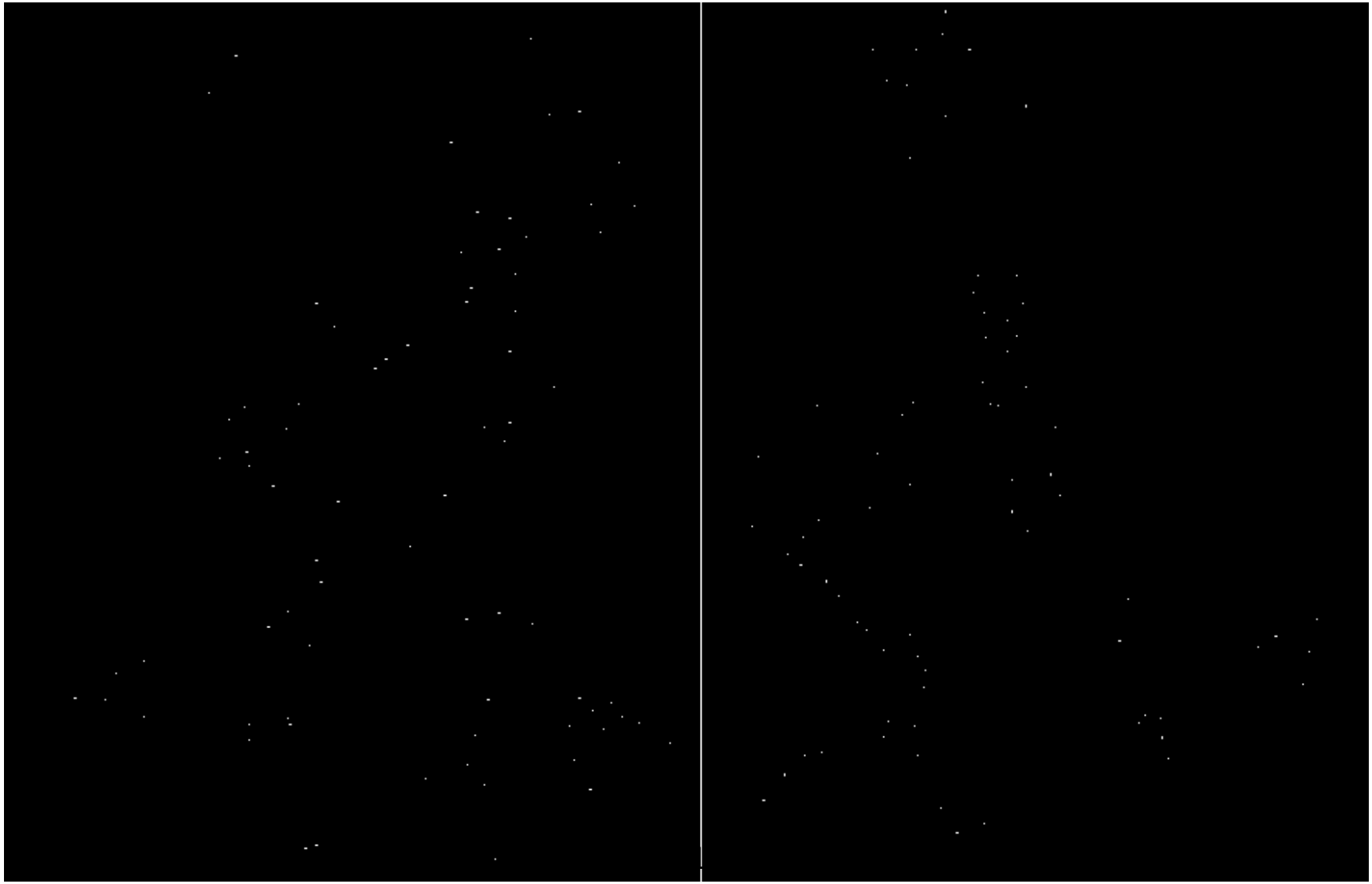
Harris Detector: Workflow

Find points with large corner response: R>threshold



Harris Detector: Workflow

Take only the points of local maxima of *R*

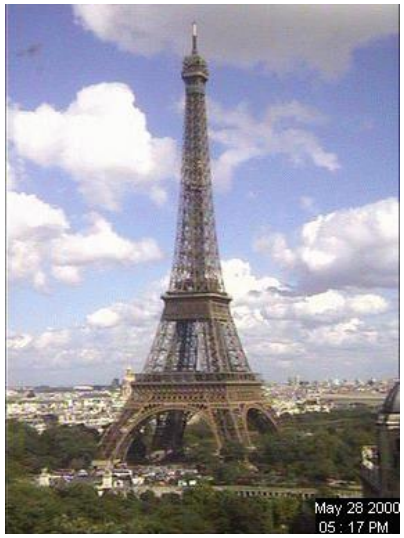


Harris Detector: Workflow

Today's Class

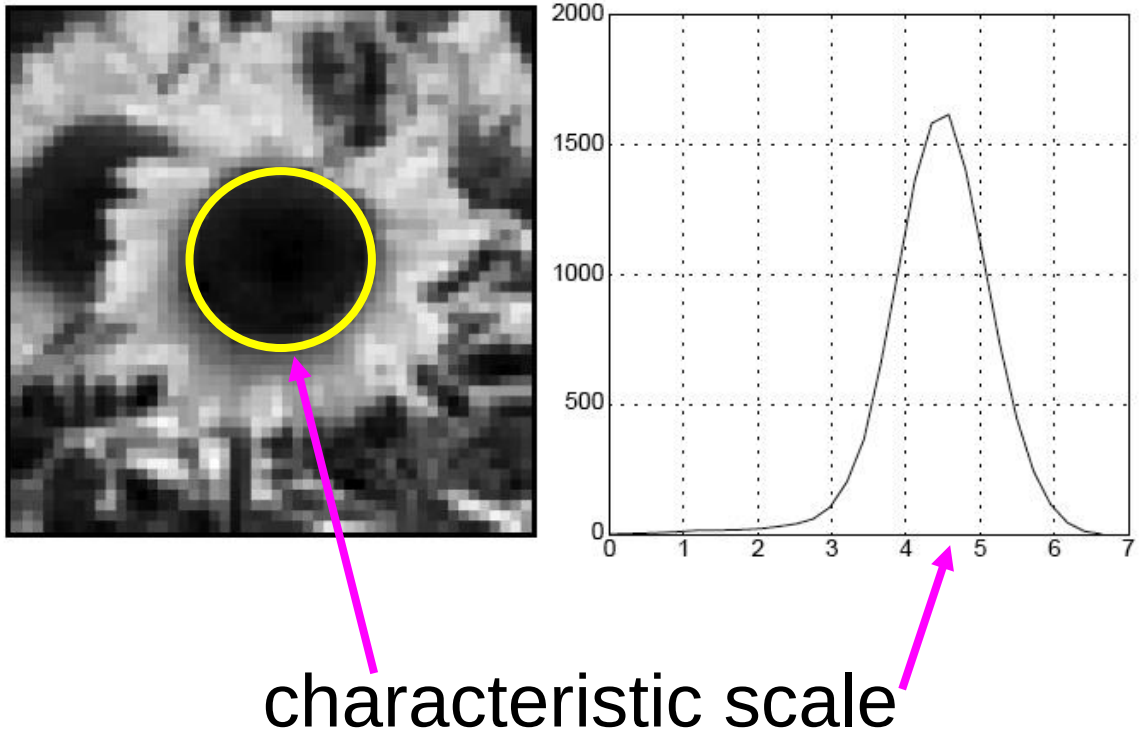
1. Reflection Models
2. Image Processing
   - Pixel and Neighbourhood processing
     - Grey image processing
     - Colour correction
   - Image Filtering
   - Edge Detection
   - Corner detection
     - Harris corner detector
     - SIFT
   - Applications

Can we detect *the same* interest points regardless of *image scale changes*?



Blob Detection in 2D

- We define the *characteristic scale* as the scale that produces peak of Laplacian response

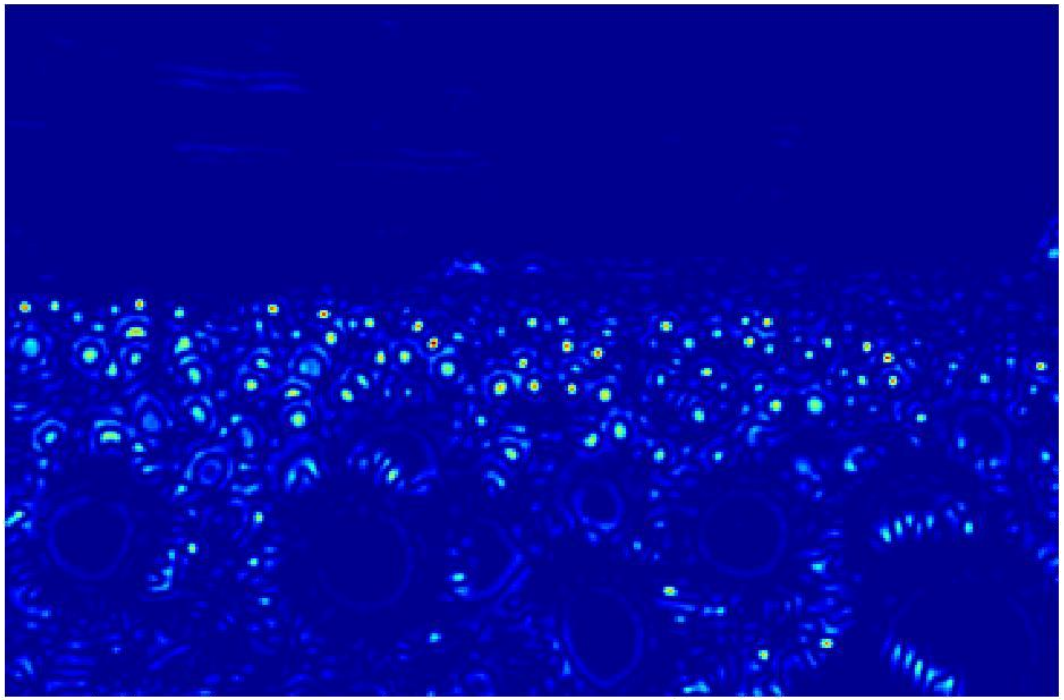


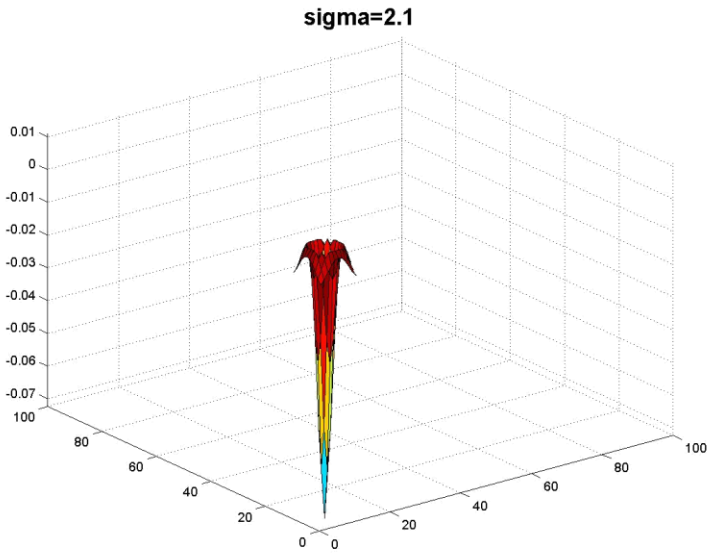
Slide credit: Lana Lazebnik

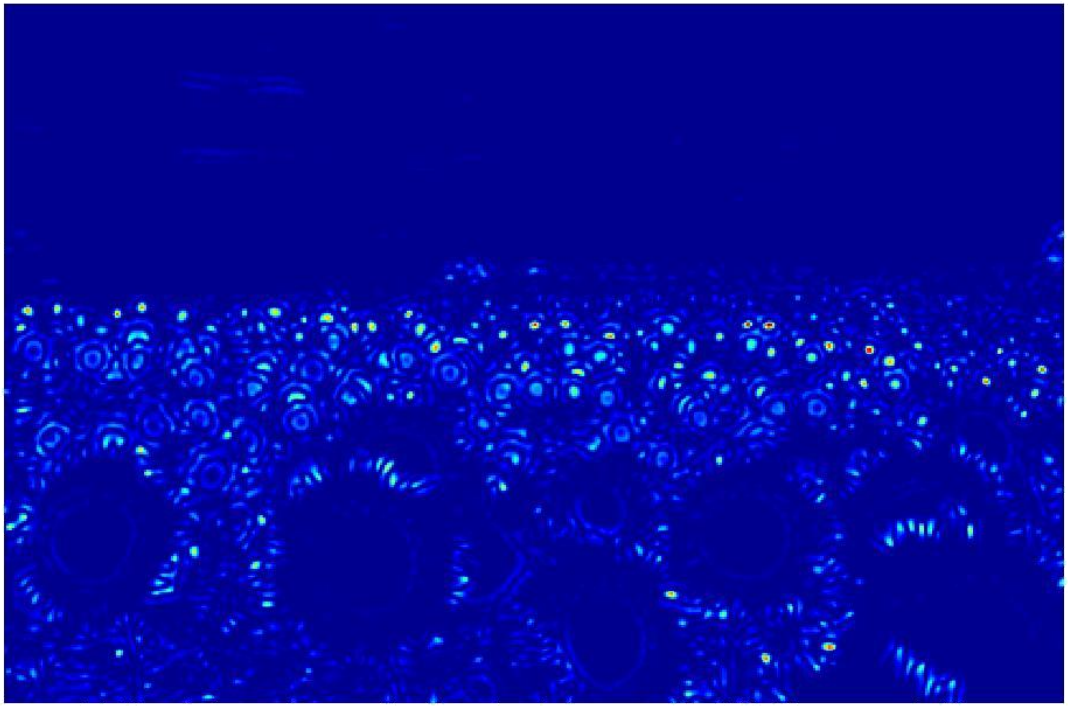
Exa

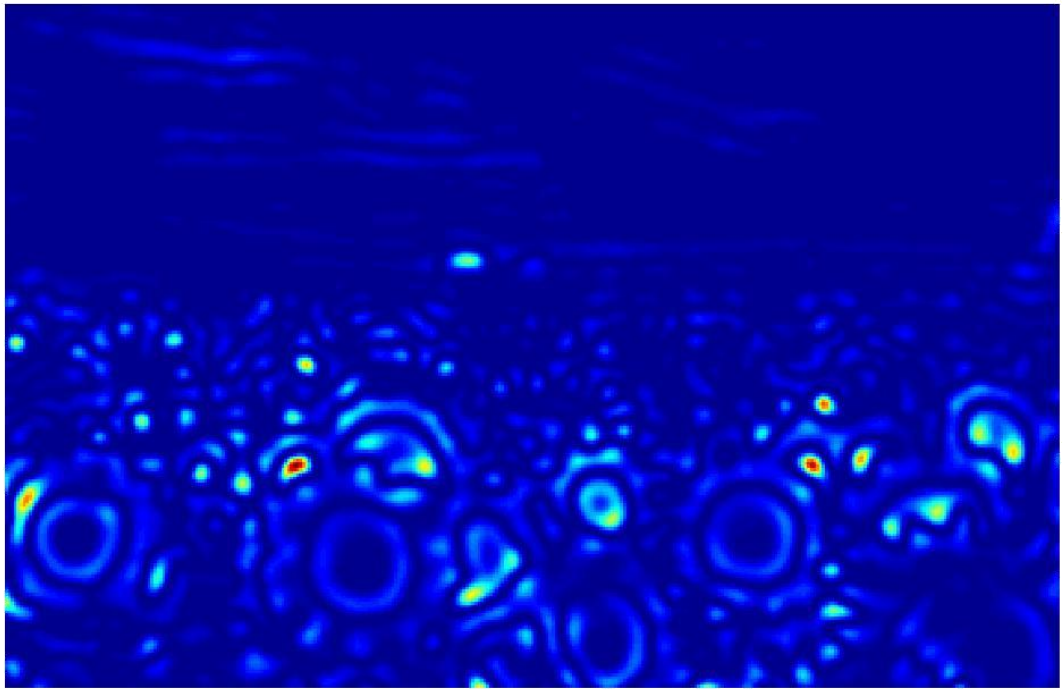
Original image
at ¾ the size

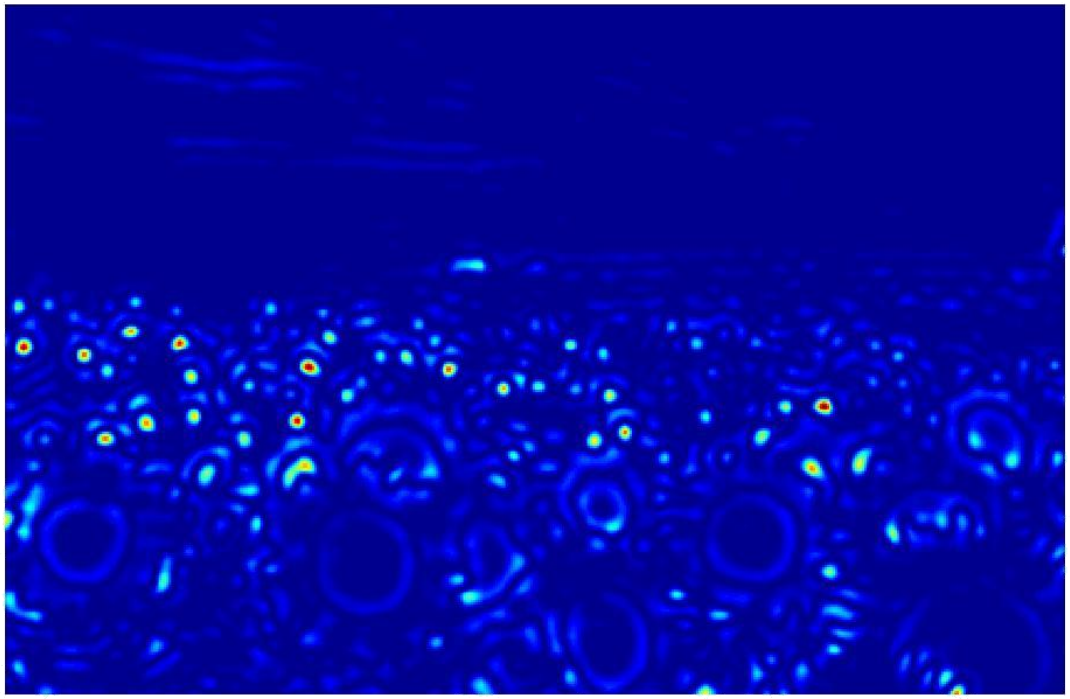
Original image
at ¾ the size

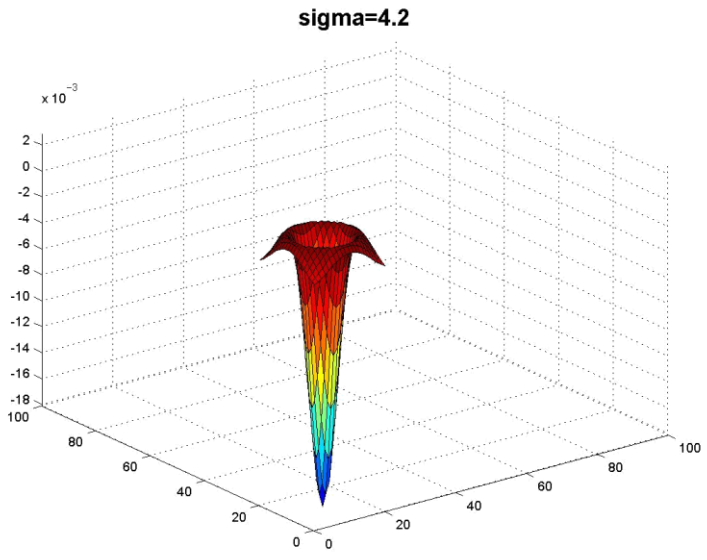


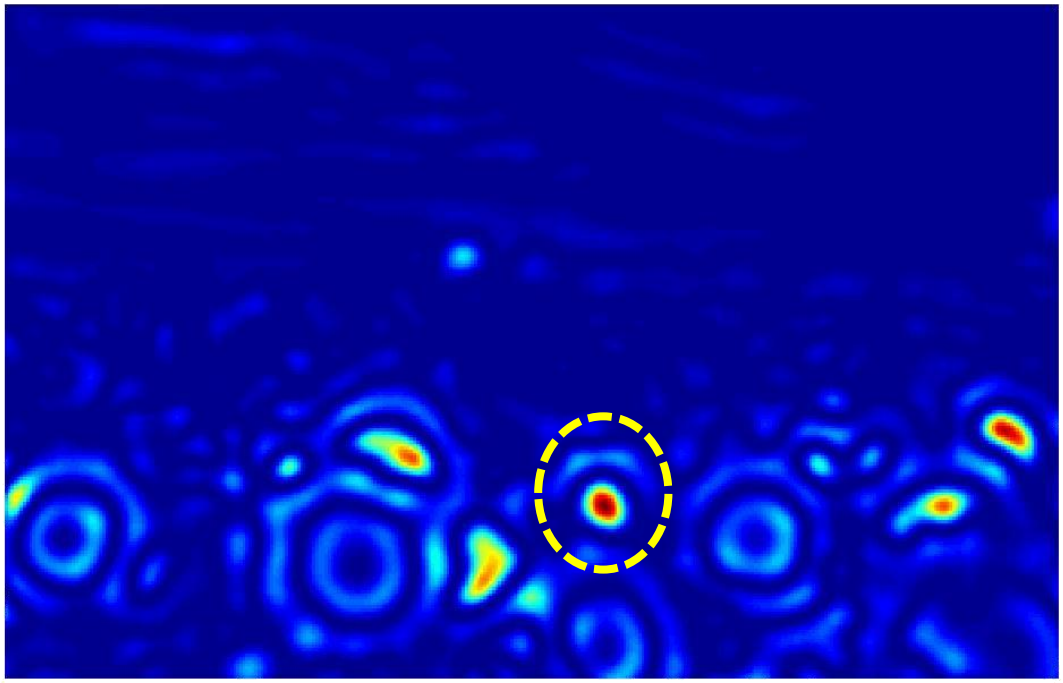


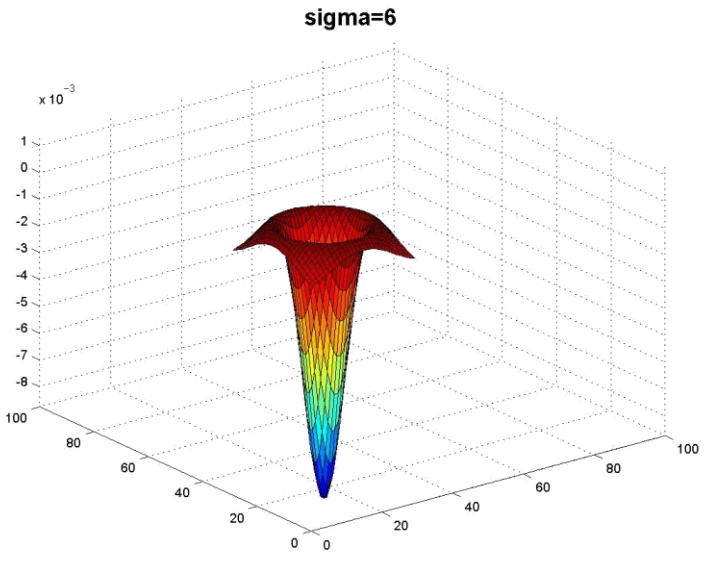


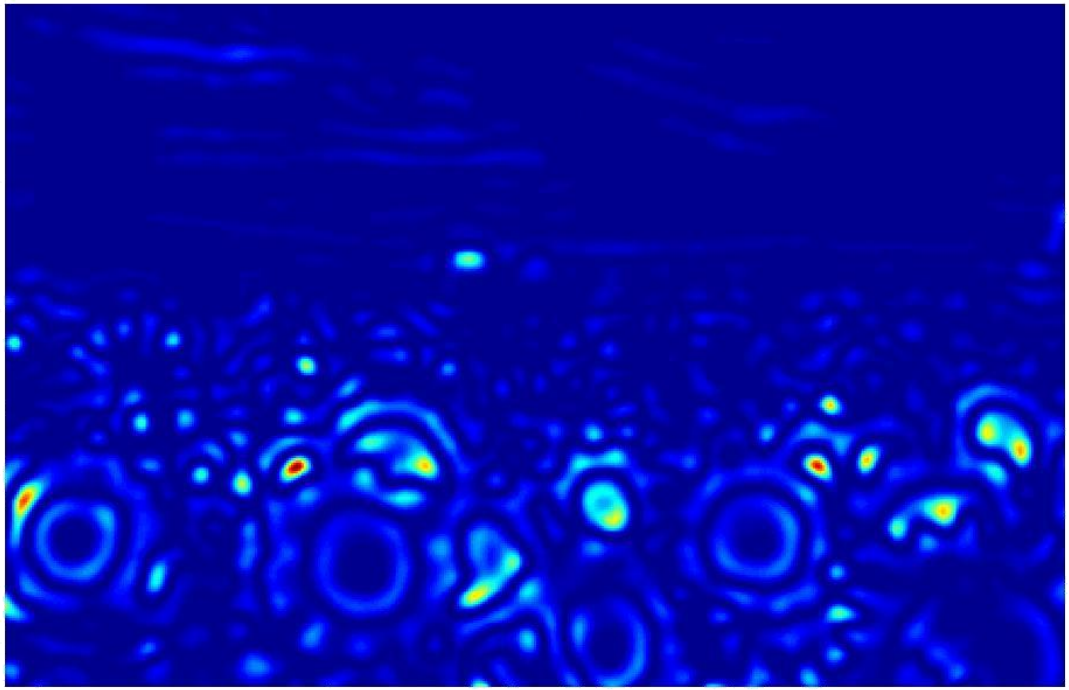


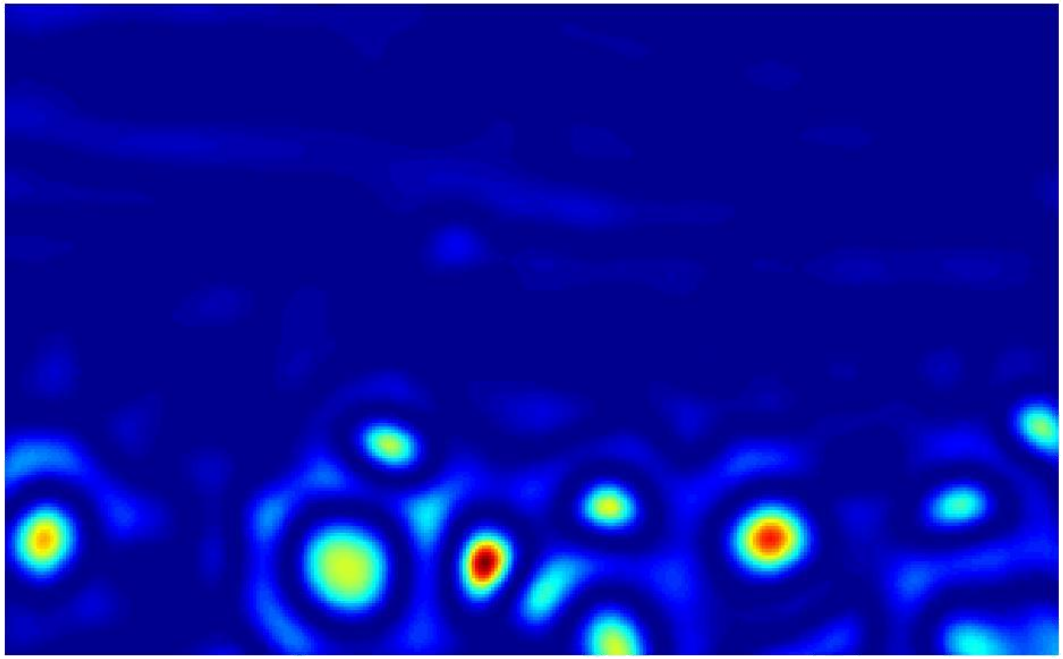


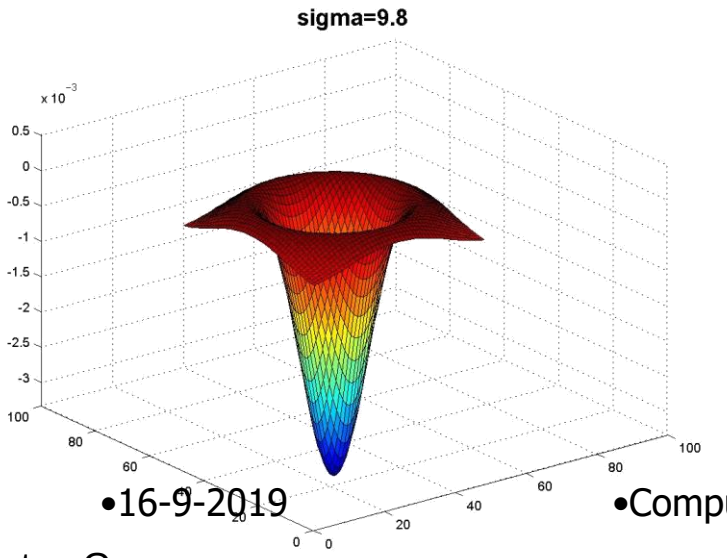

sigma=4.2
x 10^-3
2
0
-2
-4
-6
-8
-10
-12
-14
-16
-18
100
80
60
40
20
0
0
20
40
60
80
100

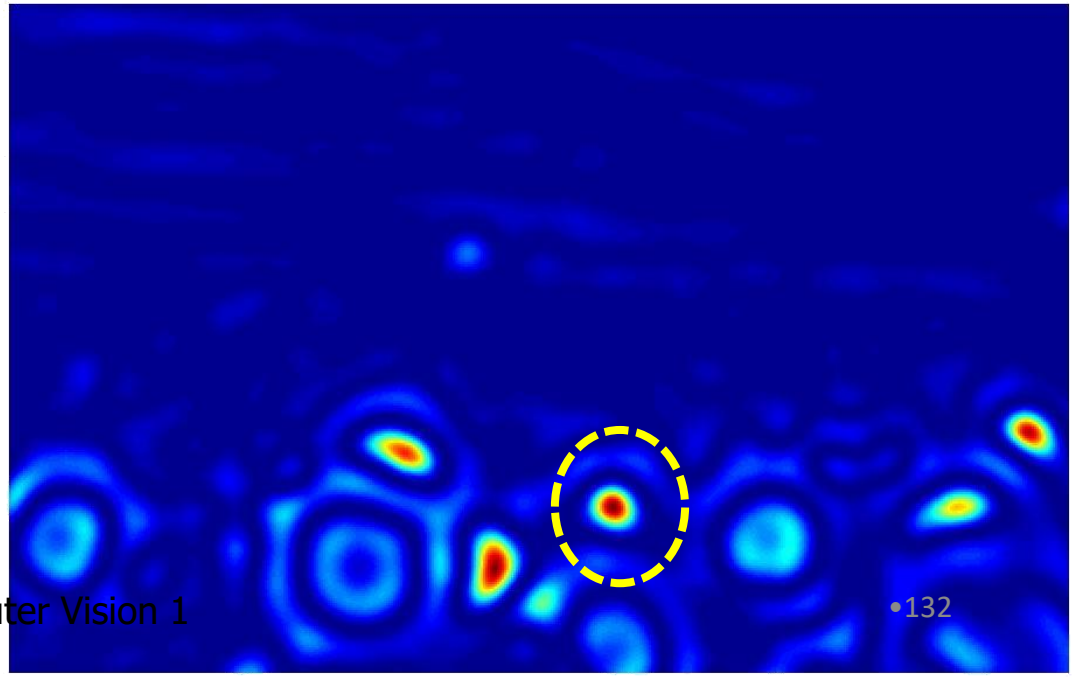



sigma=6
x 10^-3
1
0
-1
-2
-3
-4
-5
-6
-7
-8
100
80
60
40
20
0
0
20
40
60
80
100

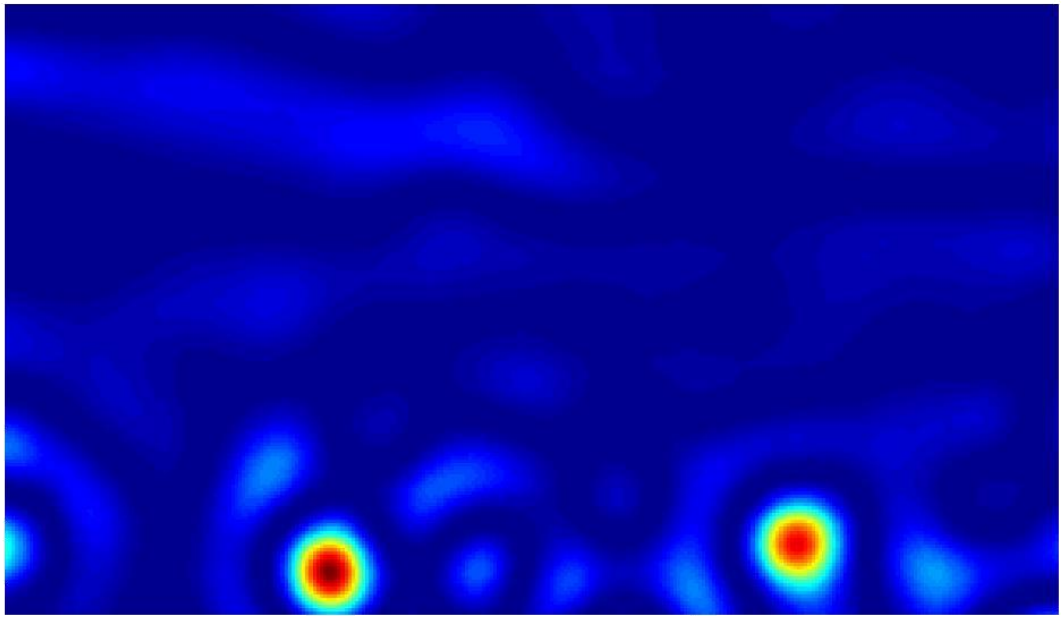


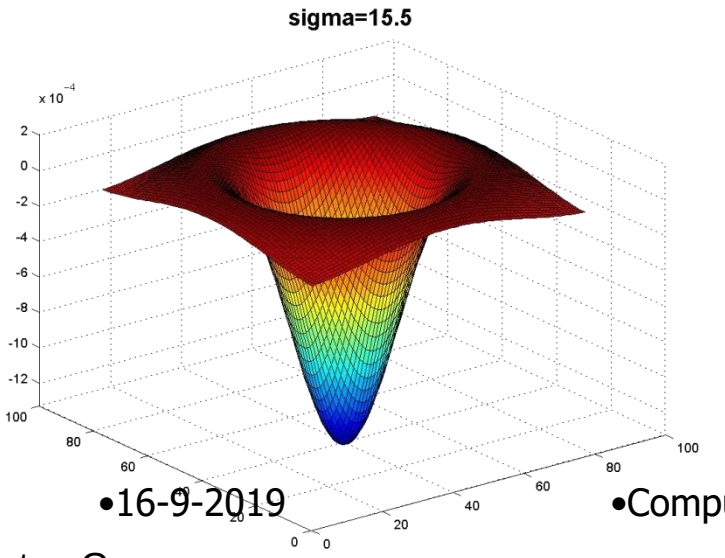


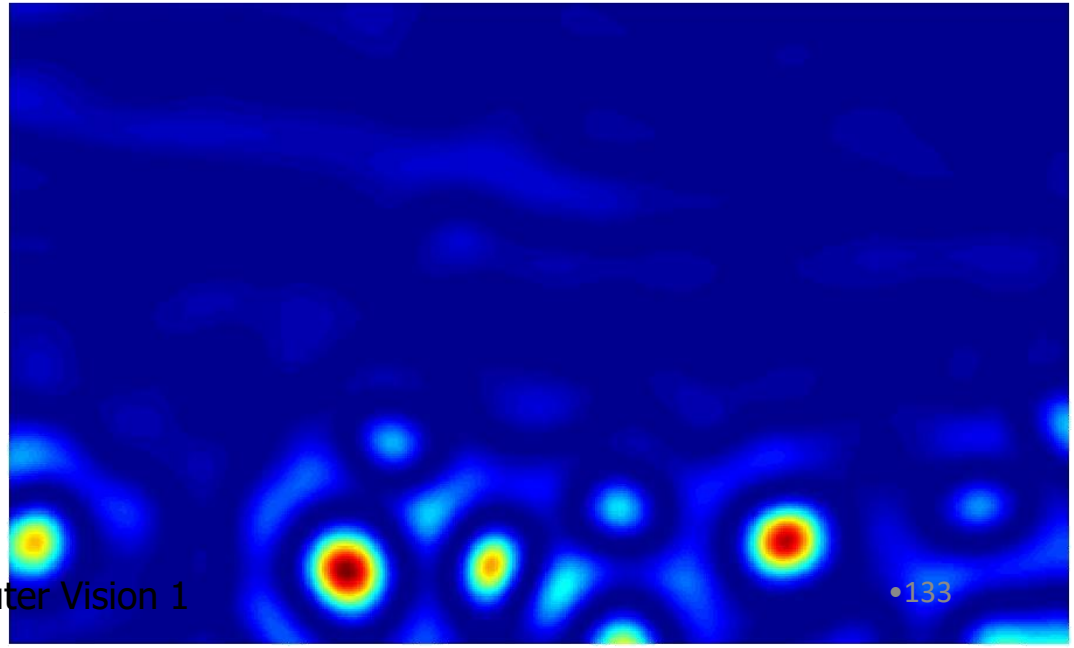




Kristen Grauman





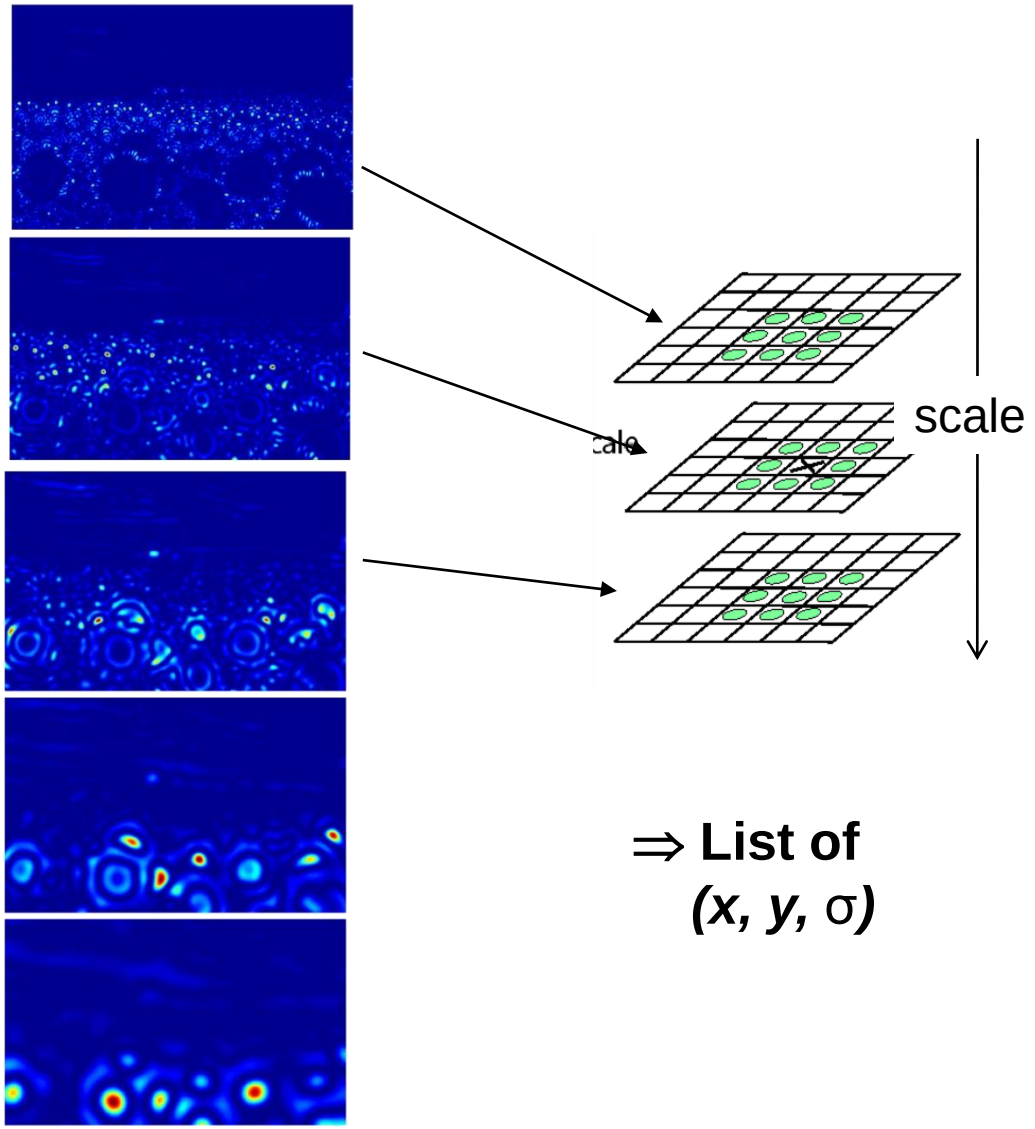


Kristen Grauman

Scale Invariant Interest Points

Interest points are local maxima in both position and scale.

$$L_{xx}(\sigma) + L_{yy}(\sigma)$$

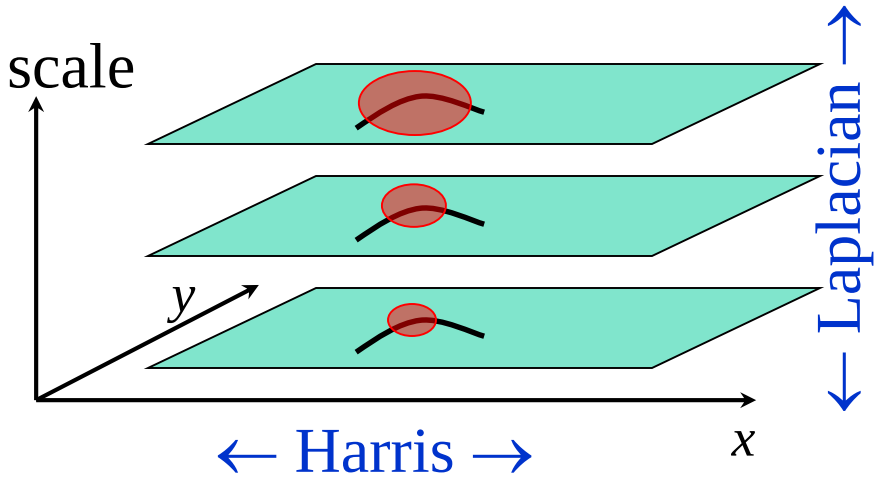


Squared filter response maps

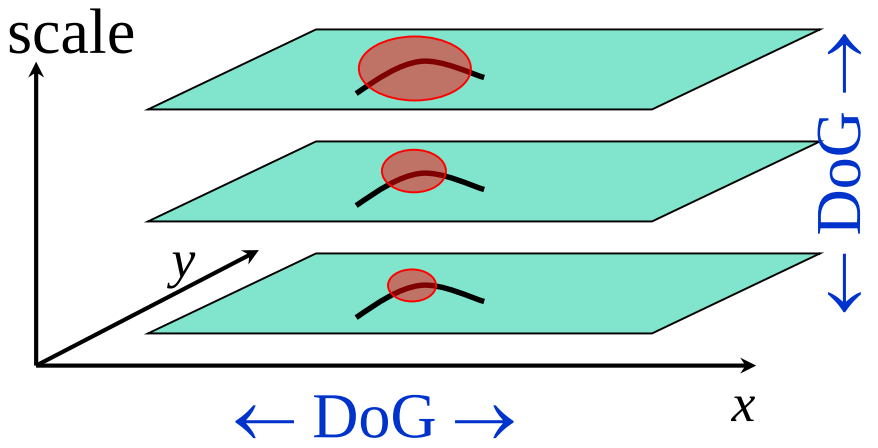
⇒ **List of** ***(x, y, σ)***

Two Popular Scale Invariant Detectors

- Harris-Laplacian[1]
  Find local maximum of:
  - Harris corner detector in space (image coordinates) + Laplacian in scale



- SIFT (Lowe)[2]
  Find local maximum of:
  - Difference of Gaussians both in space and scale

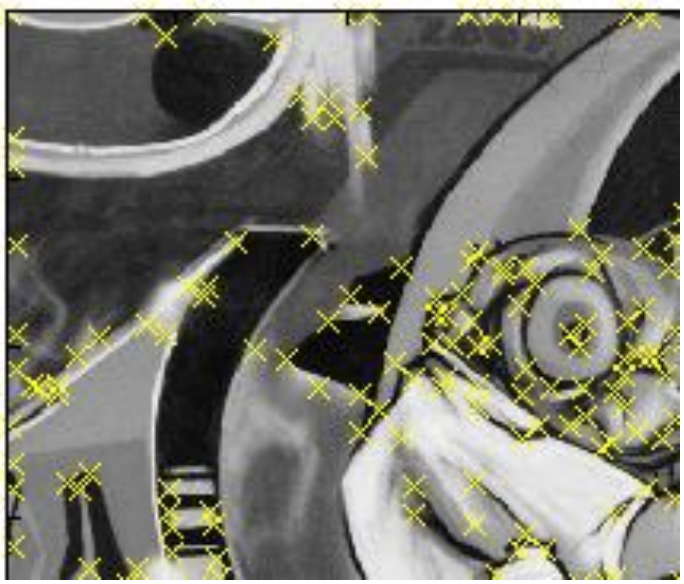


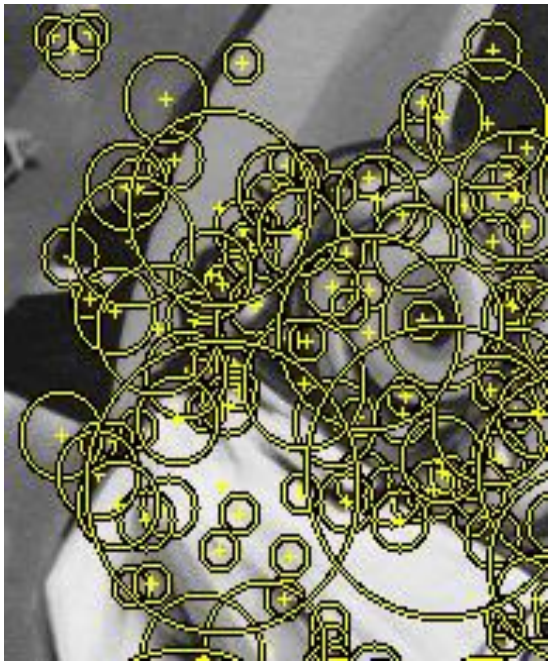
[1] K.Mikolajczyk, C.Schmid. "Indexing Based on Scale Invariant Interest Points". ICCV 2

[2] D.Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". Accepted to IJC

Comparison

Harris





LoG

Scale Invariant Feature Transform

Basic idea:

- Take 16x16 square window around detected feature (8x8 is shown below)
- Compute edge orientation (angle of the gradient - 90°) for each pixel
- Throw out weak edges (threshold gradient magnitude)
- (

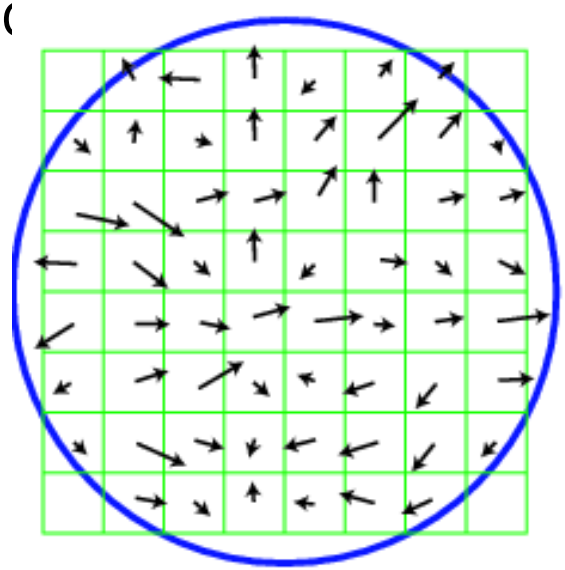
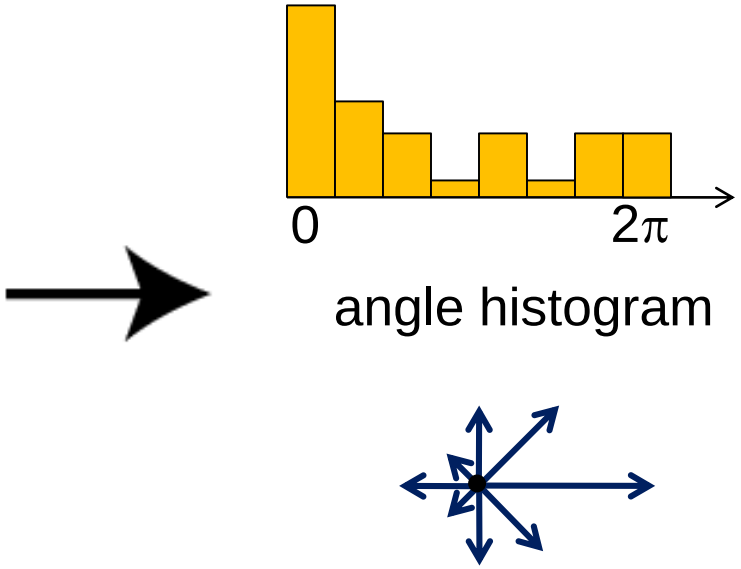


Image gradients



Adapted from slide by David Lowe

SIFT descriptor

Full version

- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Compute an orientation histogram for each cell (with 8 orientations)
- 16 cells (from 4x4 grids) * 8 orientations (per cell) = 128 dimensional descriptor

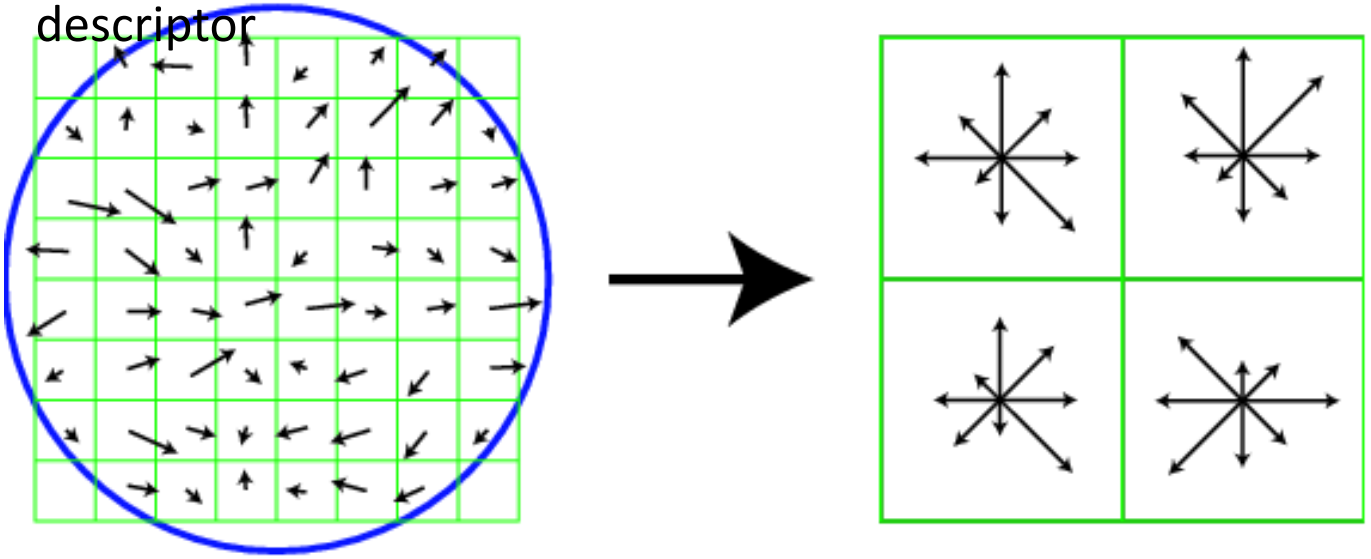


Image gradients

Keypoint descriptor

Adapted from slide by David Lowe

You can download SIFT code from Lowe's homepage

- For most local invariant feature detectors, executables are available online:
  - http://www.cs.ubc.ca/~lowe/keypoints/
  - http://www.vision.ee.ethz.ch/~surf
  - http://robots.ox.ac.uk/~vgg/research/affine

Today's Class

1. Reflection Models
2. Image Processing
   - Pixel and Neighbourhood processing
     - Grey image processing
     - Colour correction
   - Image Filtering
   - Edge Detection
   - Corner detection
     - Harris corner detector
     - SIFT
   - Applications

Applications

- Feature points are used for:
  - Image alignment
  - 3D reconstruction
  - Motion tracking
  - Robot navigation
  - Indexing and database retrieval
  - Object recognition

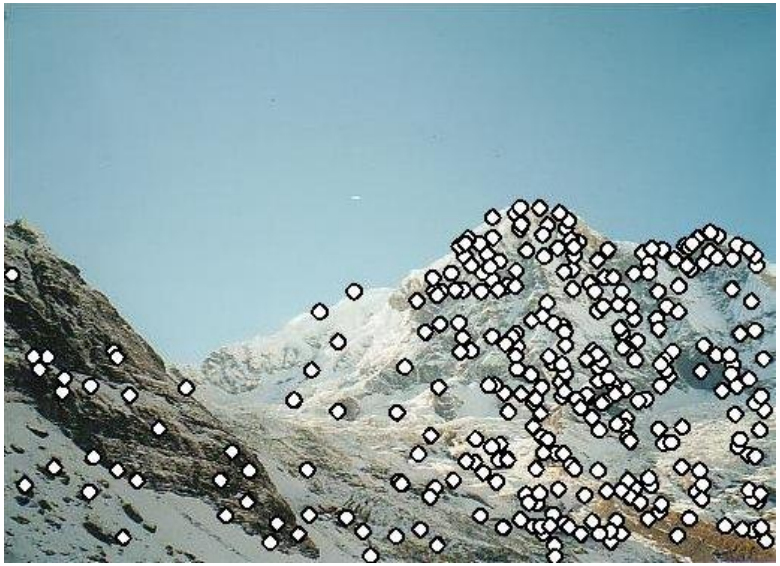
Application: Automatic Mosaicing

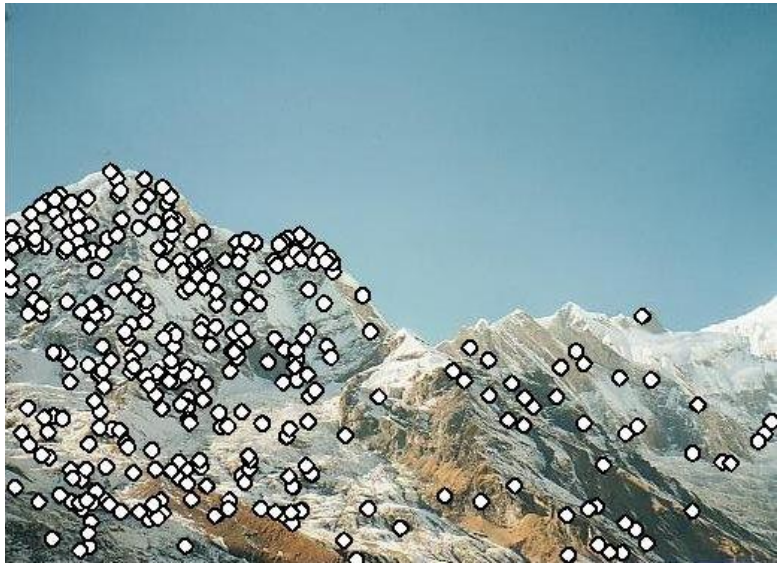
Building a Panorama

- We need to match/align/register images

Building a Panorama

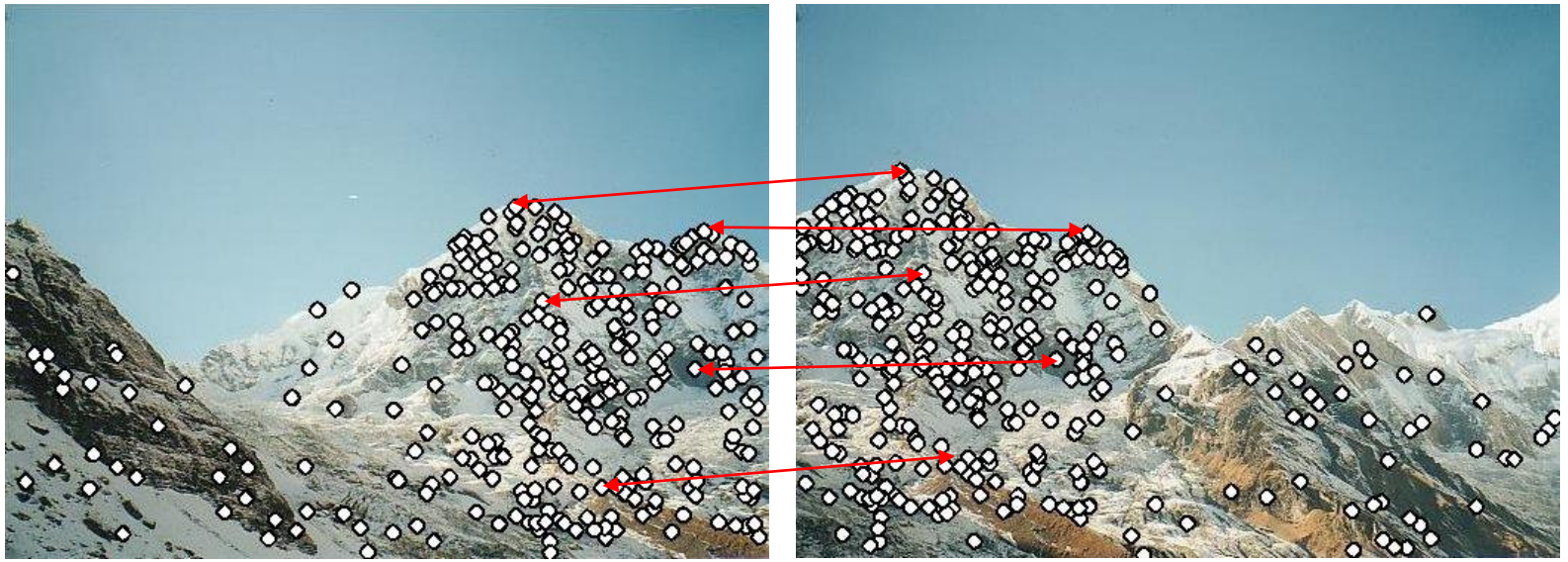
1) Detect feature points in both images





Building a Panorama

1. Detect feature points in both images
2. Find corresponding pairs



Slide: Derek Hoiem

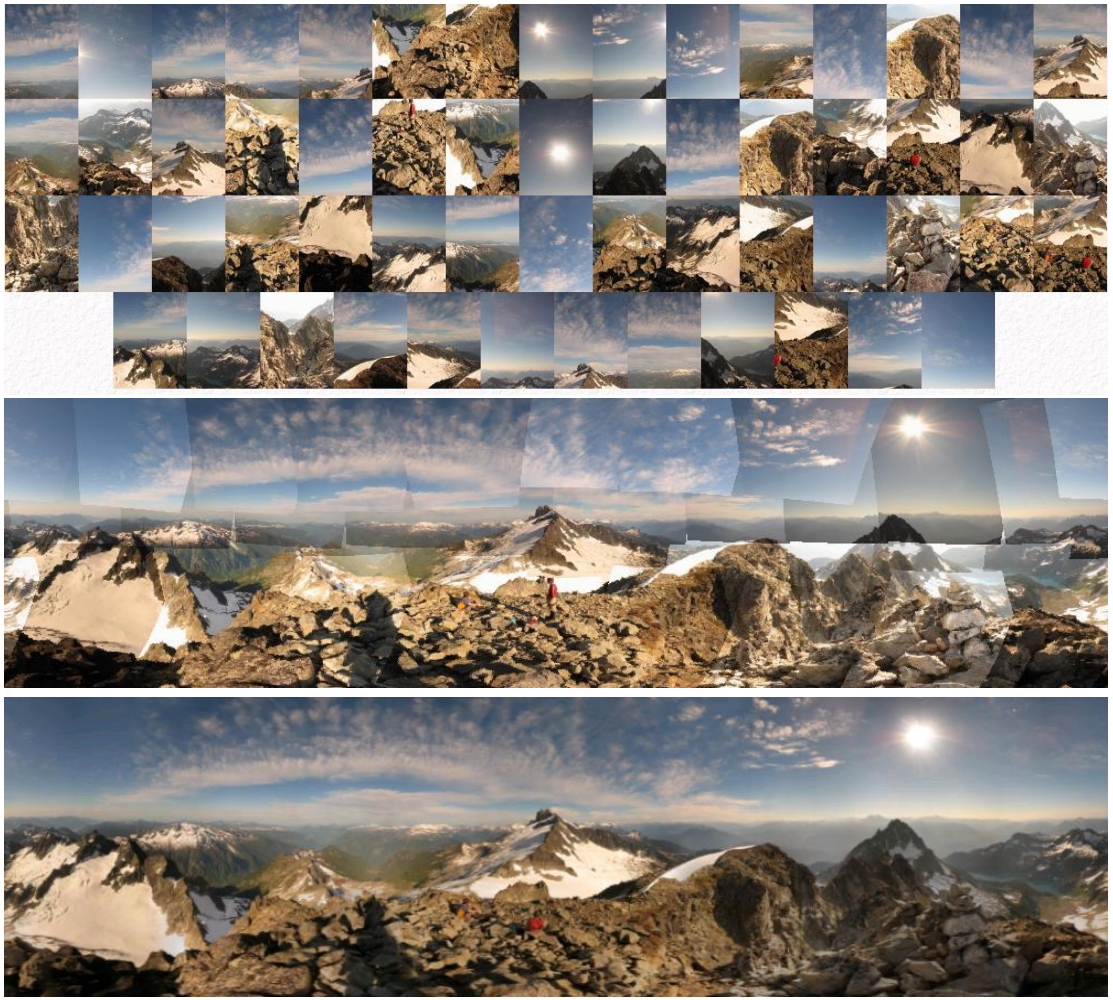
Slide: Derek Hoiem

Multi-band Blending

- Burt & Adelson 1983
  - Blend frequency bands over range $\propto$ l

Slide: Derek Hoiem

Automatic Mosaicing

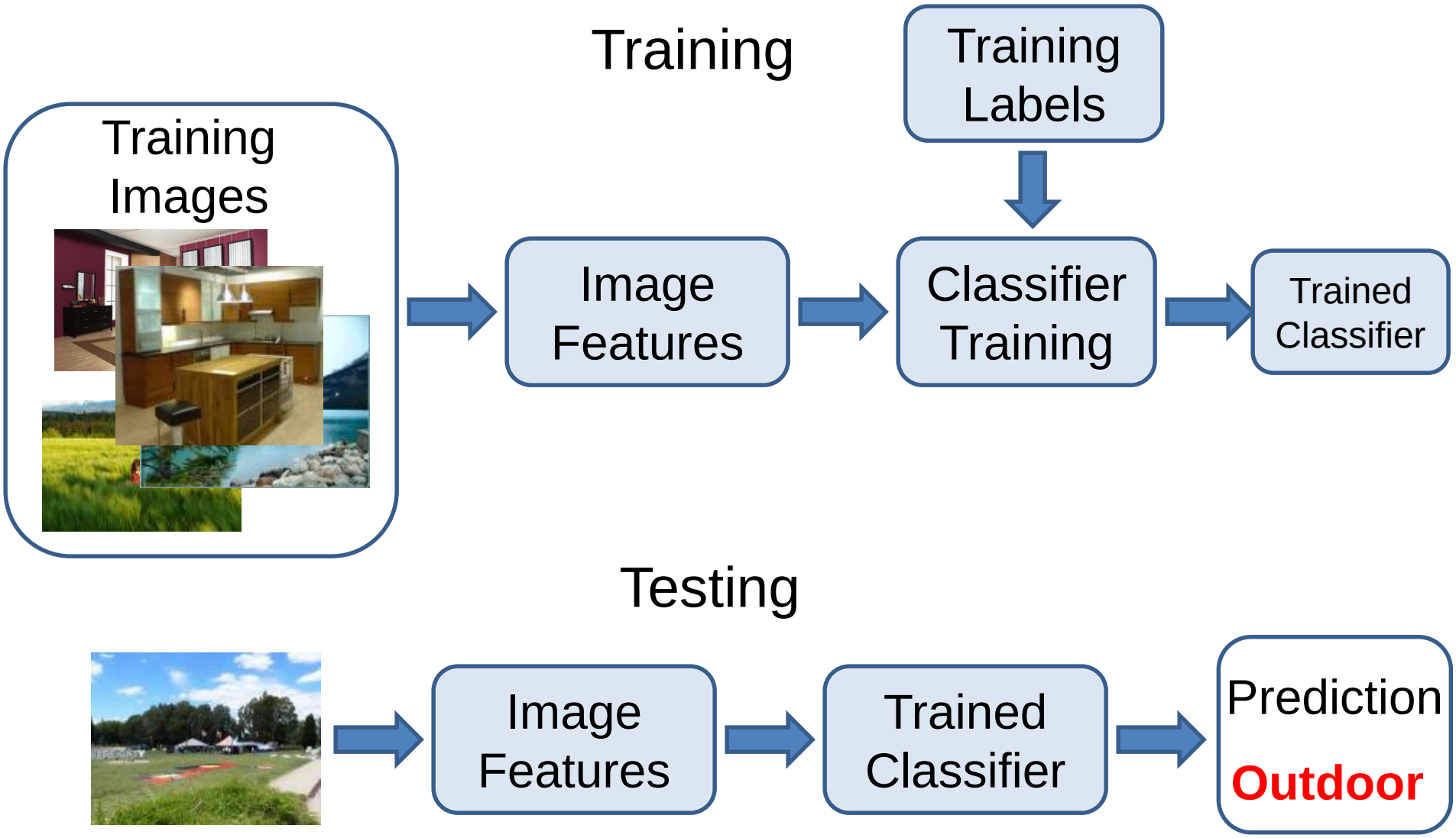


http://www.cs.ubc.ca/~mbrown/autostitch/autostitch.html

Example: Structure from Motion

https://www.youtube.com/watch?v=4cEQZreQ2zQ

Object recognition: next week



Today's Class

1. Reflection Models
2. Image Processing
   - Pixel processing
     - Grey image processing
     - Colour correction
   - Image Filtering
   - Edge Detection
   - Corner detection
     - Harris corner detector
     - SIFT
   - Applications