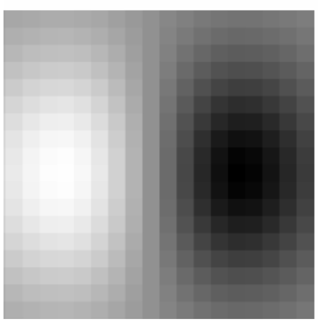
Lecture 5: Edges, Corners, Sampling, Pyramids

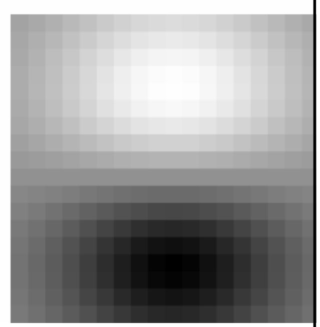
Thursday, Sept 13

Filters as templates

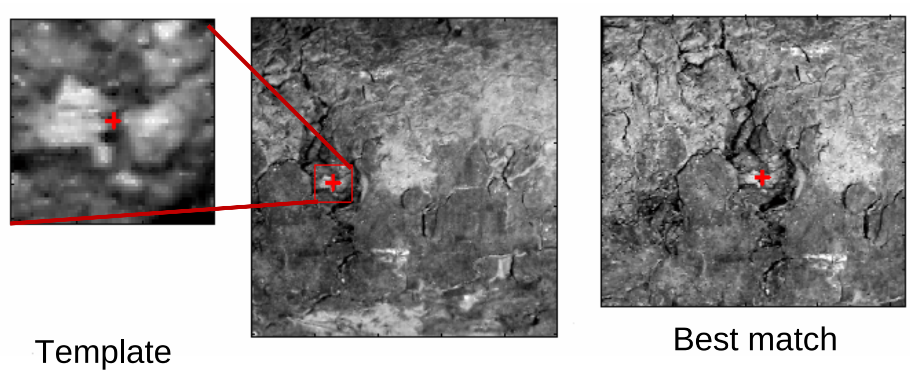
- Applying filter = taking a dot-product between image and some vector
- Filtering the image is a set of dot products

- Insight
  - filters look like the effects they are intended to find
  - filters find effects they look like





Normalized cross correlation



- Normalized correlation: normalize for image region brightness
- Windowed correlation search: inexpensive way to find a fixed scale pattern
- (Convolution = correlation if filter is symmetric)

Filters and scenes

Filters and scenes

- Scenes have holistic qualities
- Can represent scene categories with global texture
- Use *Steerable* filters, windowed for some limited spatial information
- Model likelihood of filter responses given scene category as mixture of Gaussians, (and incorporate some temporal info…)

[Torralba & Oliva, 2003]
[Torralba, Murphy, Freeman, and Rubin, ICCV 2003]

Steerable filters

- Convolution linear -- synthesize a filter of arbitrary orientation as a linear combination of "basis filters"

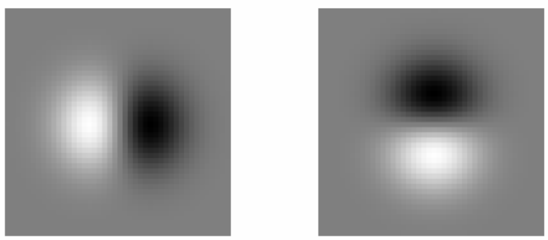
$$
\begin{aligned}
R_1^{0^\circ} &= G_1^0 * I \\
R_1^{90^\circ} &= G_1^{90} * I \\
\text{then} & \\
R_1^\theta &= \cos(\theta) R_1^{0^\circ} + \sin(\theta) R_1^{90^\circ}.
\end{aligned}
$$

- Interpolated filter responses more efficient than explicit filter at arbitrary orientation

[Freeman & Adelson, The Design and Use of Steerable Filters, PAMI 1991]

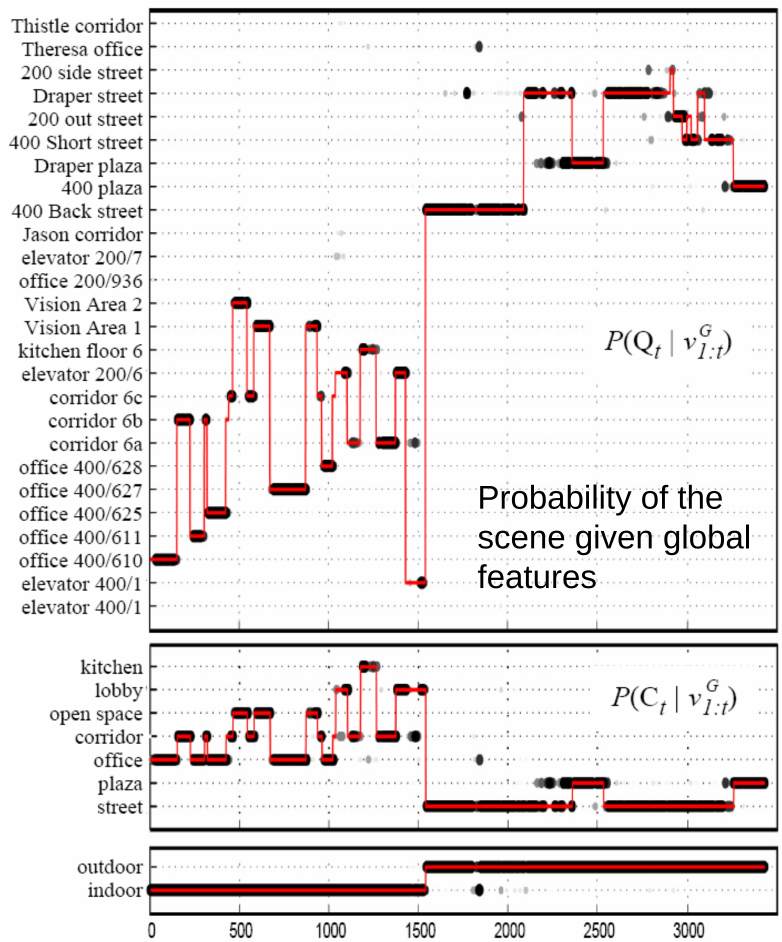
Steerable filters

Freeman &
Adelson, 1991



$G_1^{0°}$ $G_1^{90°}$

Basis filters for derivative of Gaussian



[Torralba, Murphy, Freeman, and Rubin, ICCV 2003]

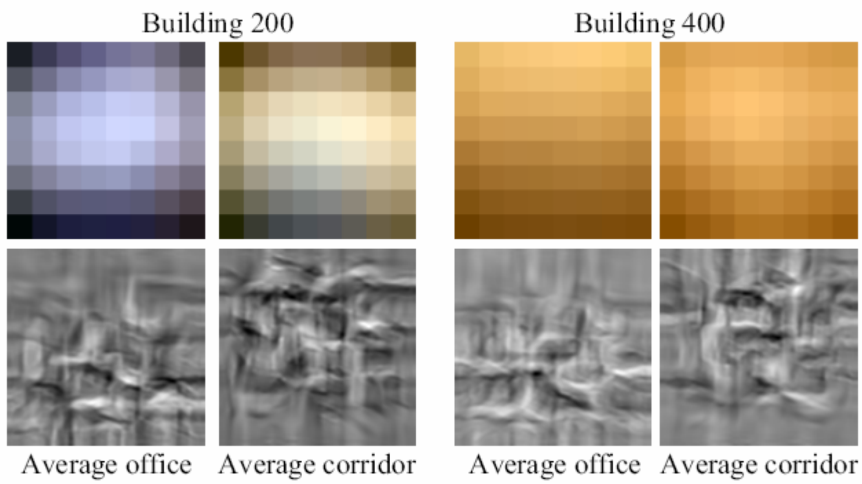
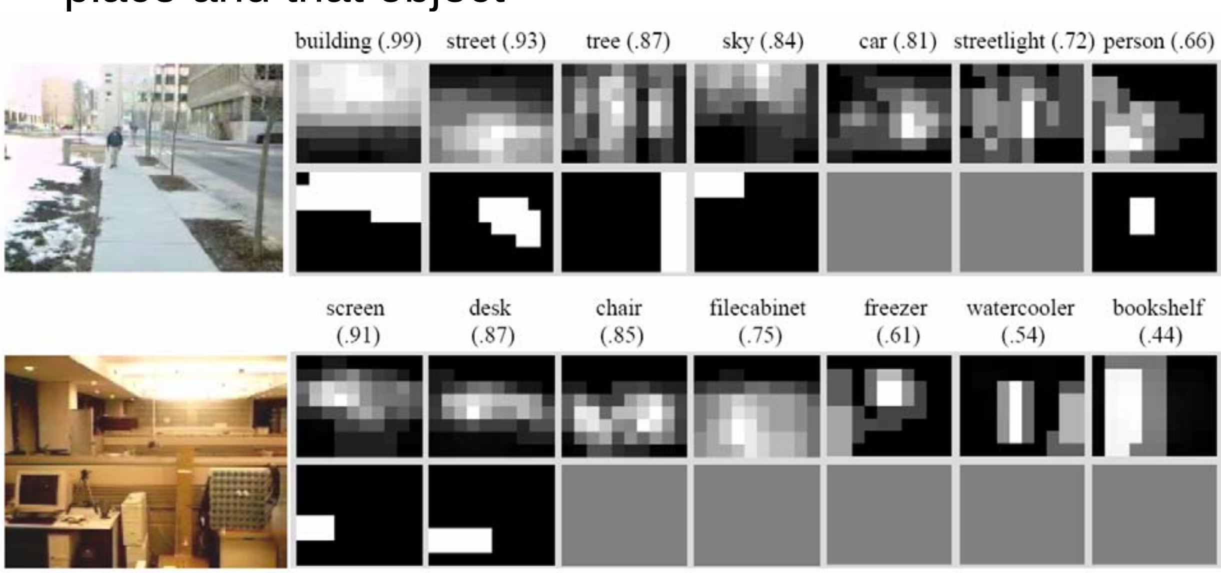


Figure 7. Average of color (top) and texture (bottom) signatures of offices and corridors for two different buildings. While the algorithm uses a richer representation than simply the mean images shows here, these averages show that the overall color of offices/corridors varies significantly between the two buildings, whereas the texture features are more stable.

[Torralba, Murphy, Freeman, and Rubin, ICCV 2003]

Contextual priors

- Use scene recognition → predict objects present
- For object(s) likely to be present, predict locations based on similarity to previous images with the same place and that object

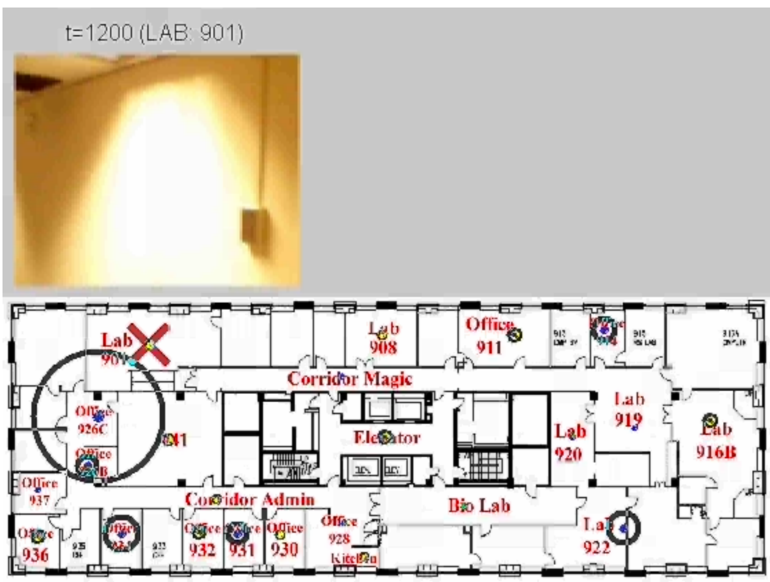


Scene category

Specific place

(black=right, red=wrong)

[Torralba, Murphy, Freeman, and Rubin, ICCV 2003]



Blue solid circle: recognition with temporal info

Black hollow circle: instantaneous recognition using global feature only

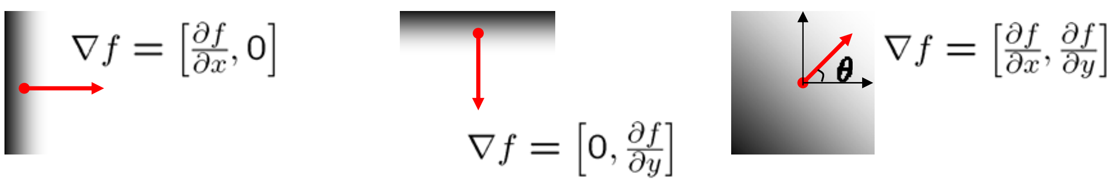
Cross: true location

Image gradient

The gradient of an image:

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

The gradient points in the direction of most rapid change in intensity



The gradient direction (orientation of edge normal) is given by:

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

The *edge strength* is given by the gradient magnitude

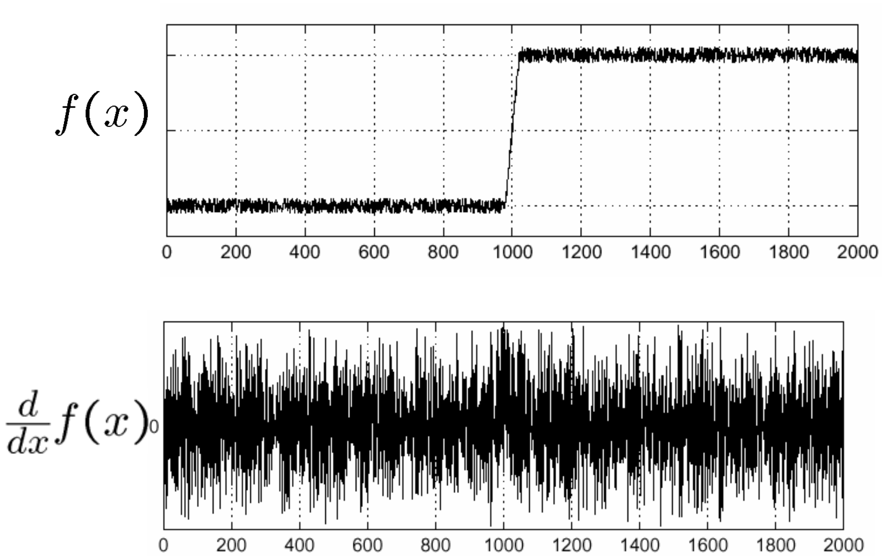
$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Slide credit S. Seitz

Effects of noise

Consider a single row or column of the image

- Plotting intensity as a function of position gives a signal



Where is the edge?

Slide credit S. Seitz

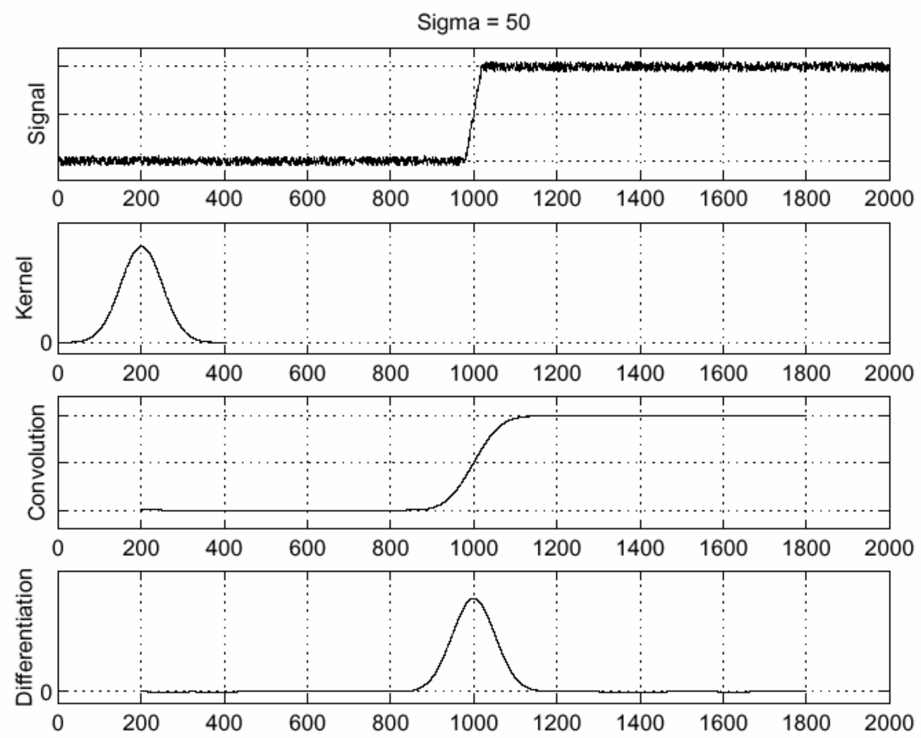
Solution: smooth first

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 theorem of convolution

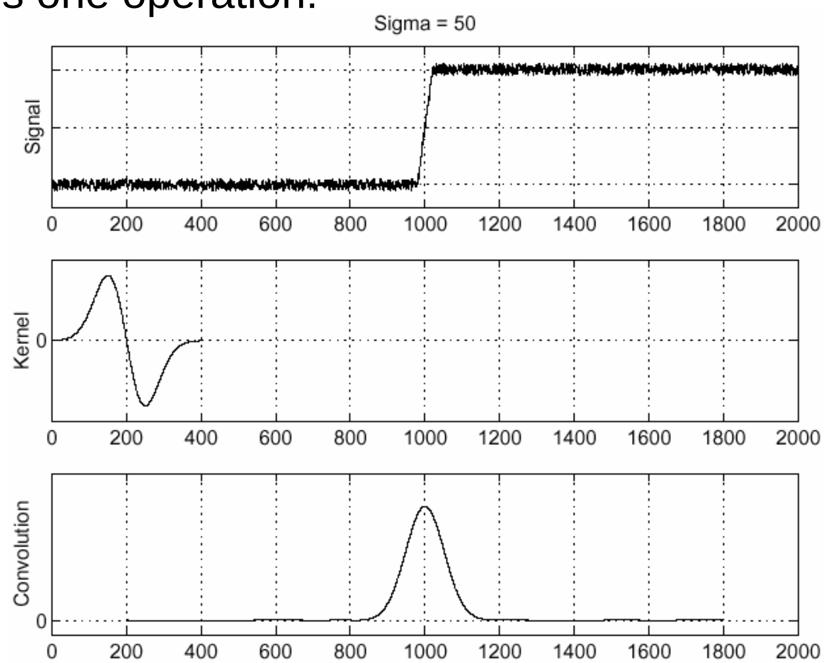
$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$$

This saves us one operation:

f

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

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



Slide credit S. Seitz

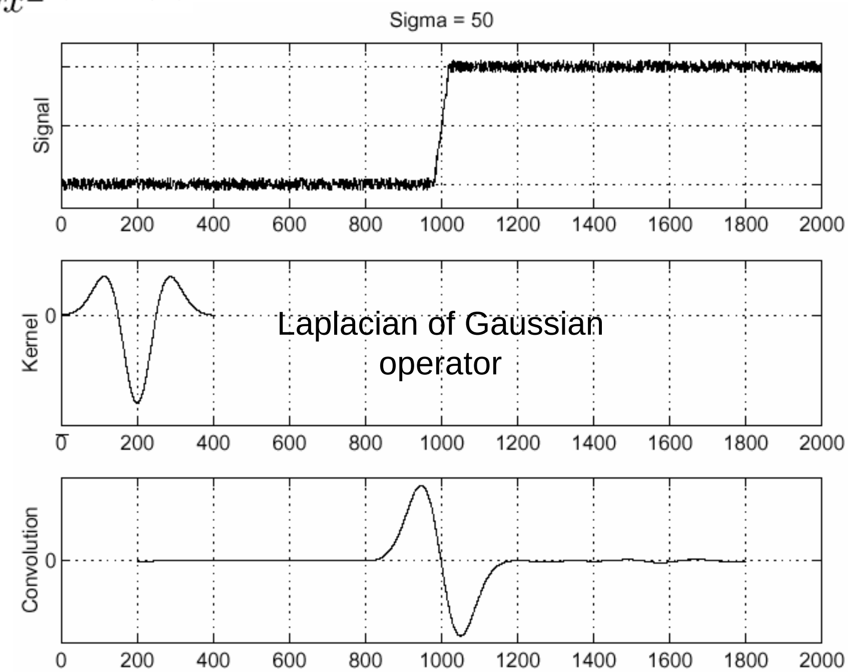
Laplacian of Gaussian

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

f

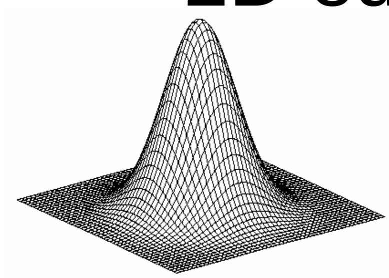
$\frac{\partial^2}{\partial x^2}h$

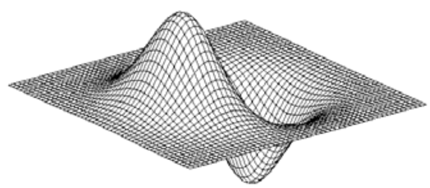
$(\frac{\partial^2}{\partial x^2}h) \star f$



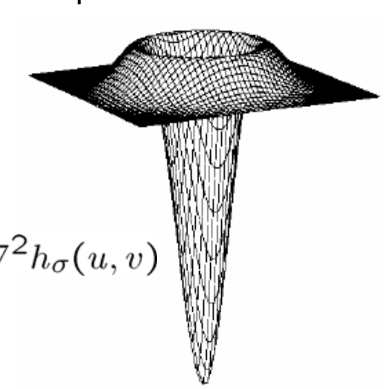
Where is the edge? Zero-crossings of bottom graph

2D edge detection filters





Laplacian of Gaussian



Gaussian

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

derivative of Gaussian

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

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

- ∇^2 is the **Laplacian** operator:

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

Slide credit S. Seitz

The Canny edge detector

original image (Lena)

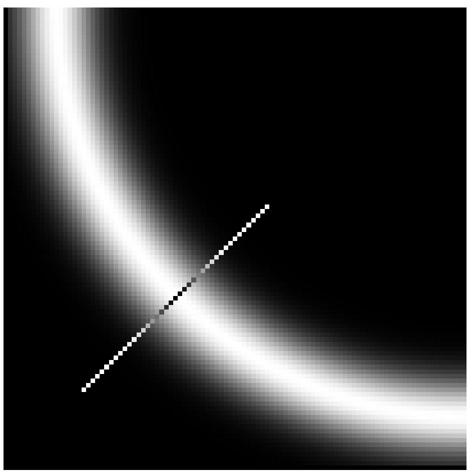
The Canny edge detector

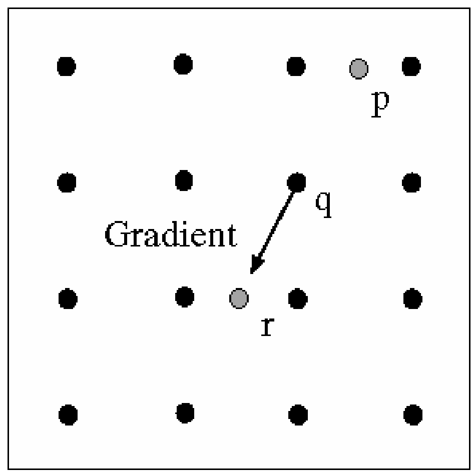
norm of the gradient

The Canny edge detector

thresholding

Non-maximum suppression





Check if pixel is local maximum along gradient direction, select single max across width of the edge

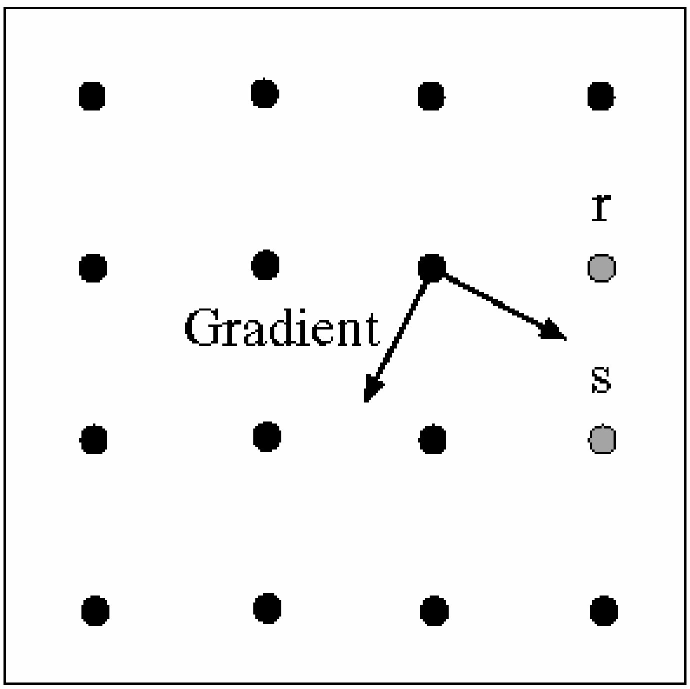
- requires checking interpolated pixels p and r

Slide credit S. Seitz

The Canny edge detector

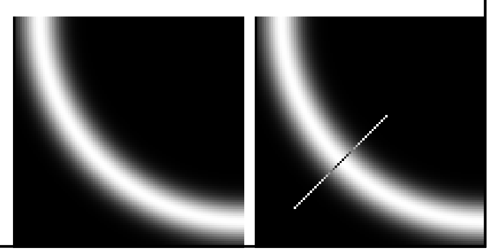
thinning
(non-maximum suppression)

Predicting the next edge point



(Forsyth & Ponce)

Assume the marked point is an edge point. Then we construct the tangent to the edge curve (which is normal to the gradient at that point) and use this to predict the next points (here either r or s).



Hysteresis Thresholding

Reduces the probability of false contours and fragmented edges

Given result of non-maximum suppression:

For all edge points that remain,

- locate next unvisited pixel where intensity > t_{high}
- start from that point, follow chains along edge and add points where intensity < t_{low}

Edge detection by subtraction

original

Edge detection by subtraction

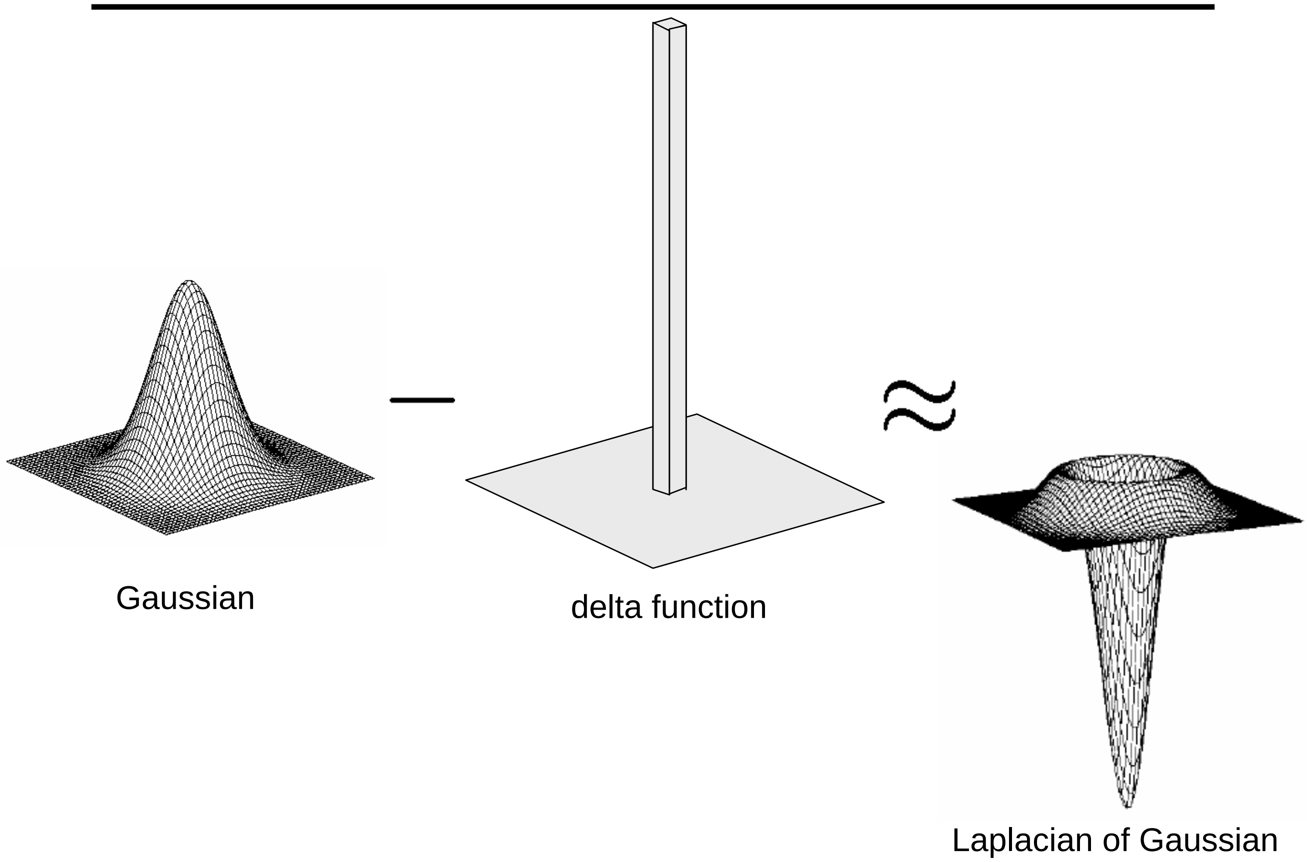
smoothed (5x5 Gaussian)

Edge detection by subtraction

smoothed – original
(scaled by 4, offset +128)

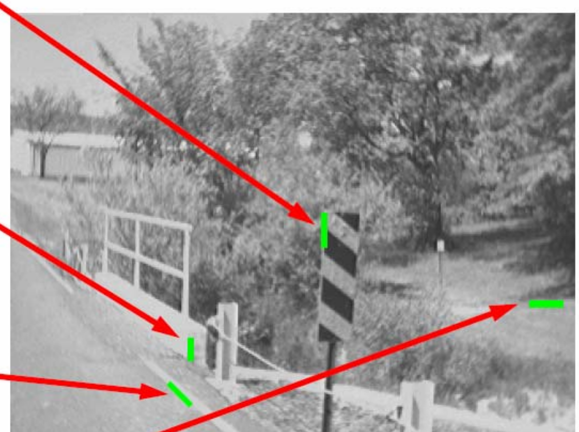
Why does this work?

Gaussian - image filter



Causes of edges

- Depth discontinuity
- Surface orientation discontinuity
- Reflectance discontinuity (i.e., change in surface material properties)
- Illumination discontinuity (e.g., shadow)



If the goal is image understanding, what do we want from an edge detector?

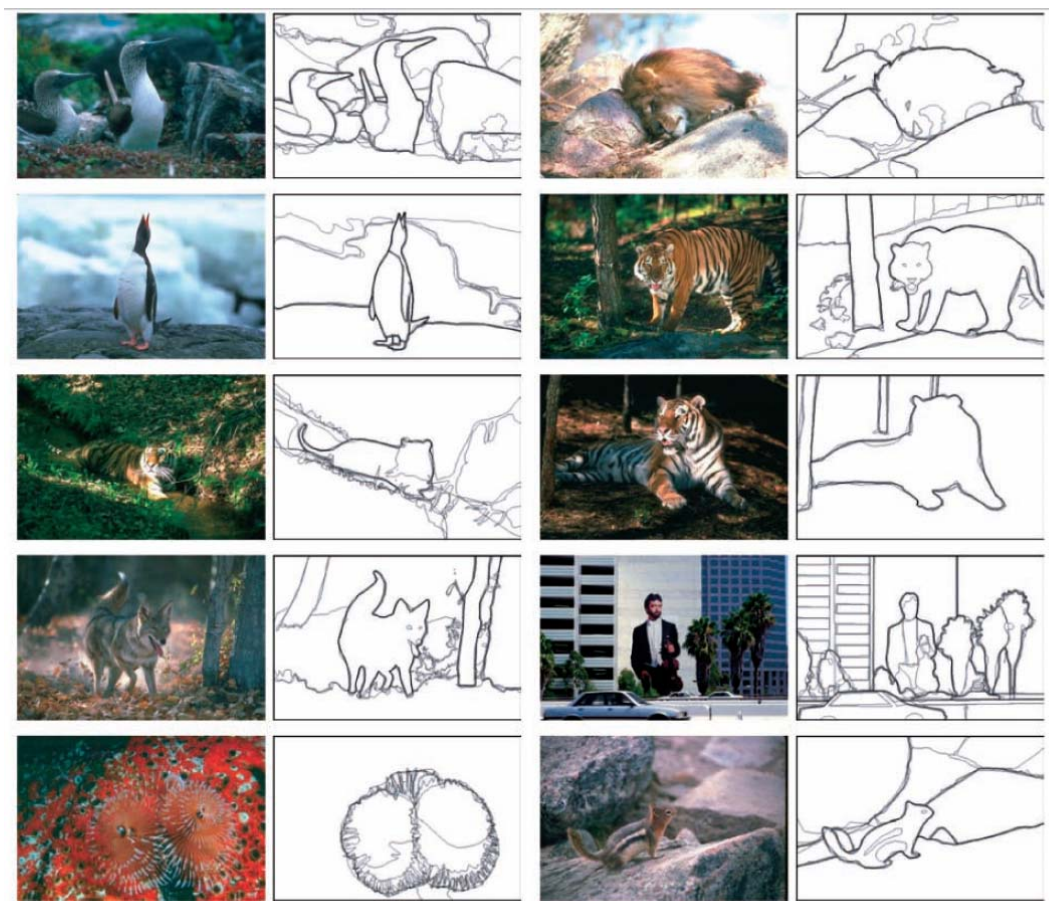
Adapted from C. Rasmussen

Learning good boundaries

- Use ground truth (human-labeled) boundaries in natural images to learn good features
- Supervised learning to optimize cue integration, filter scales, select feature types

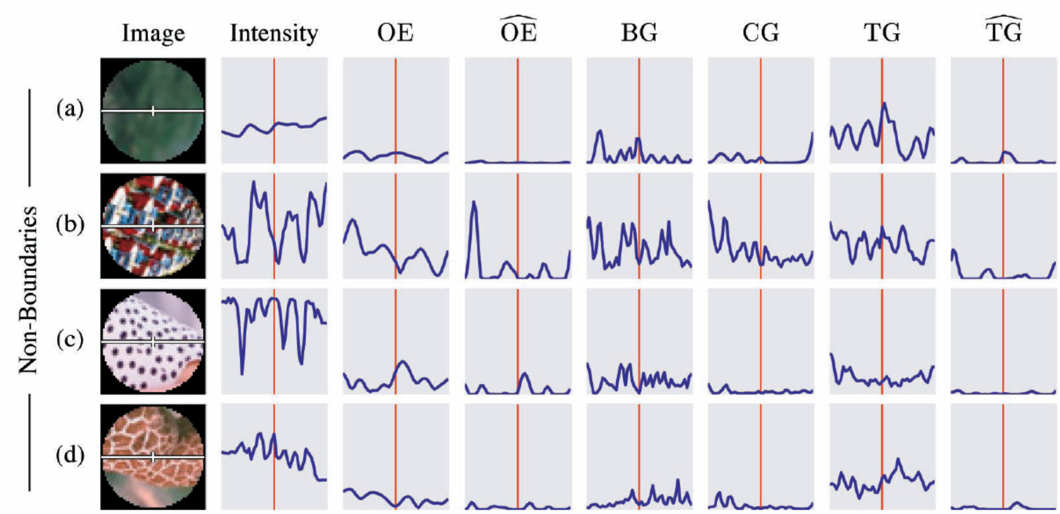
Work by D. Martin and C. Fowlkes and D. Tal and J. Malik, Berkeley Segmentation Benchmark, 2001

Human-
marked
segment
boundaries



[D. Martin et al. PAMI 2004]

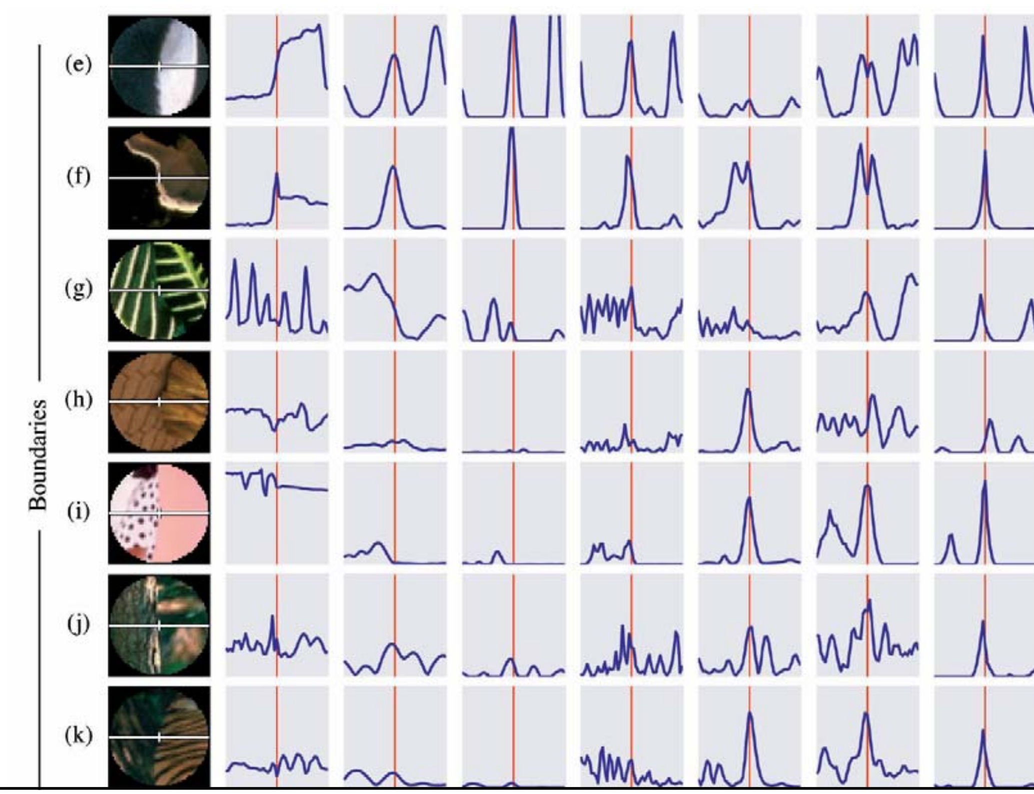
What features are responsible for perceived edges?



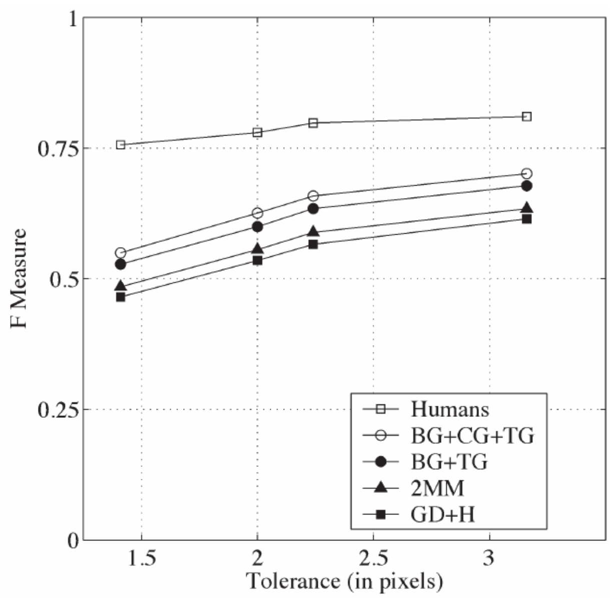
Feature profiles (oriented energy, brightness, color, and texture gradients) along the patch's horizontal diameter

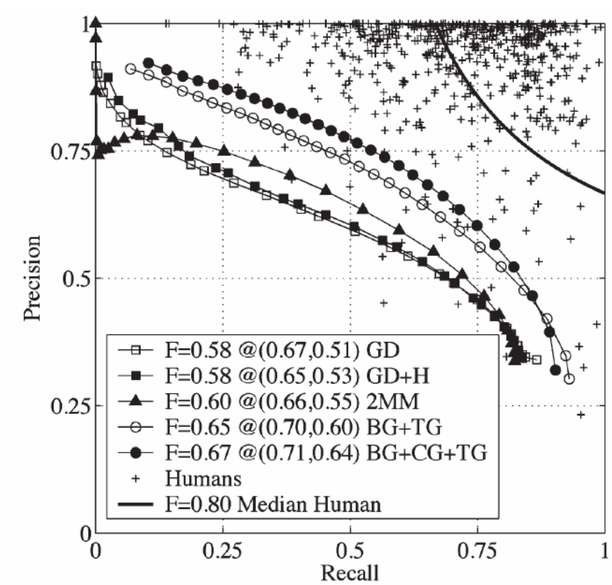
[D. Martin et al. PAMI 2004]

What features are responsible for perceived edges?



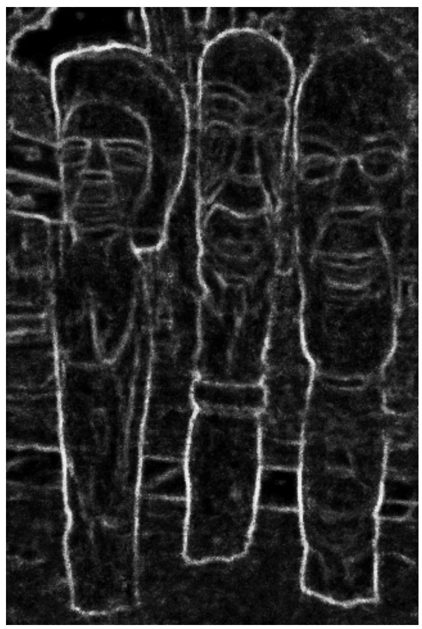
Learning good boundaries



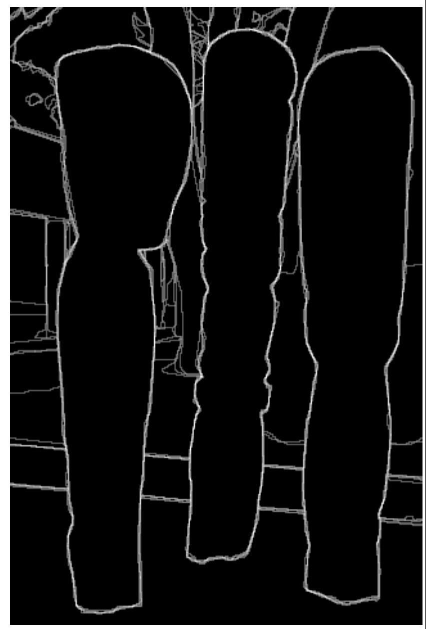


[D. Martin et al. PAMI 2004]

Original

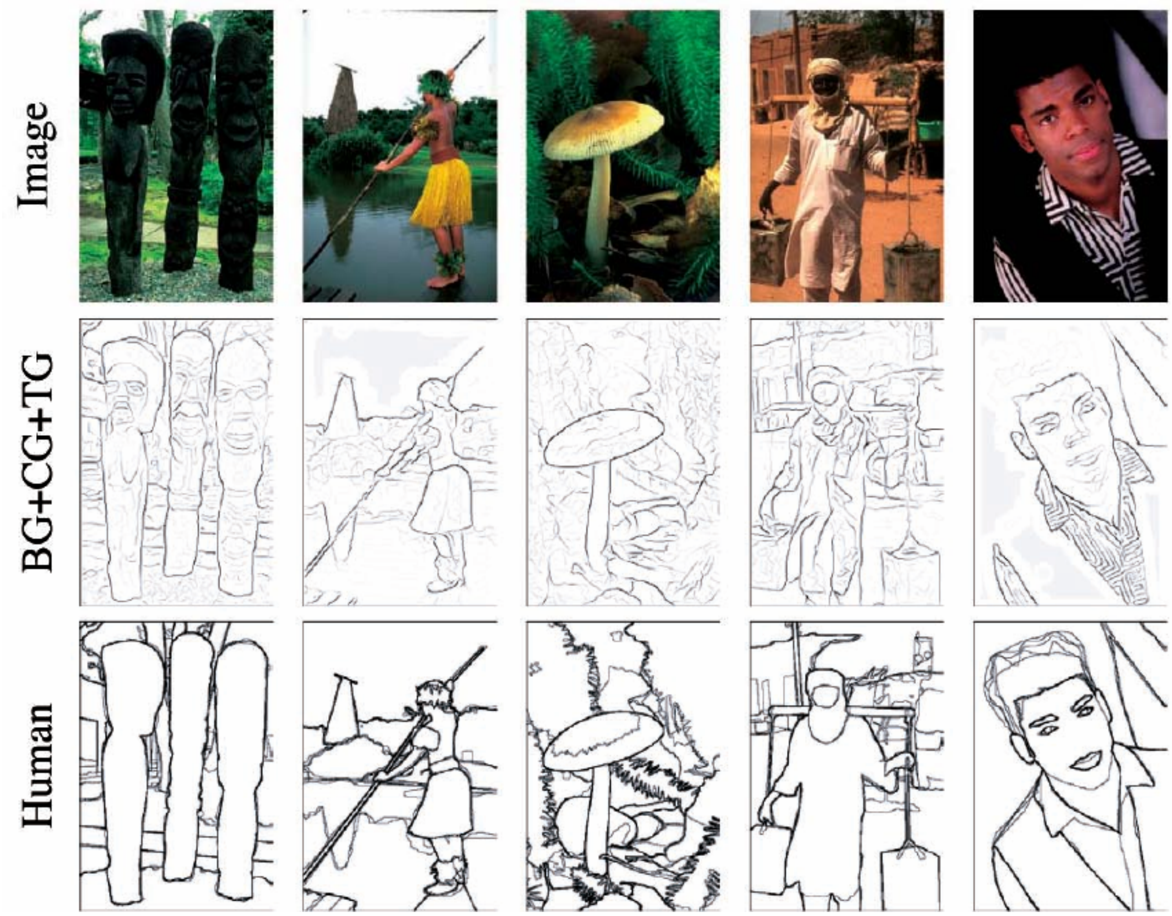


Boundary detection



Human-labeled

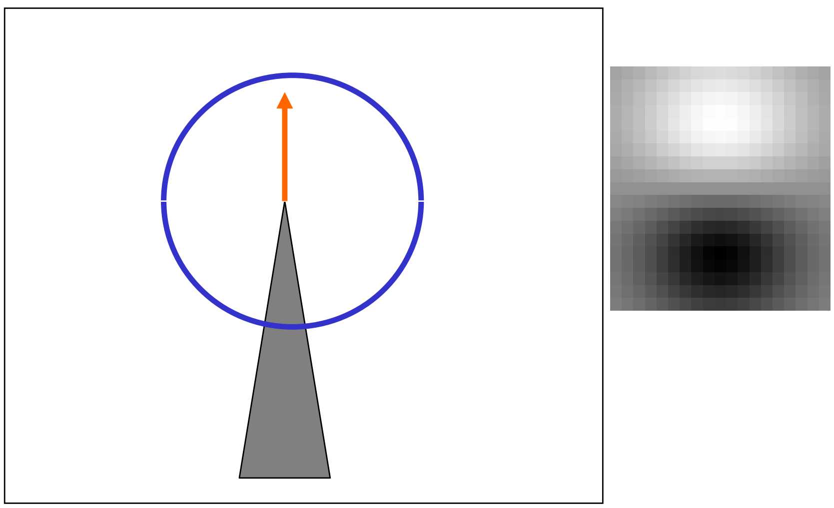
Berkeley Segmentation Database, D. Martin and C. Fowlkes and D. Tal and J. Malik



[D. Martin et al. PAMI 2004]

Edge detection and corners

- Partial derivative estimates in x and y fail to capture corners



Why do we care about corners?

Case study: panorama stitching

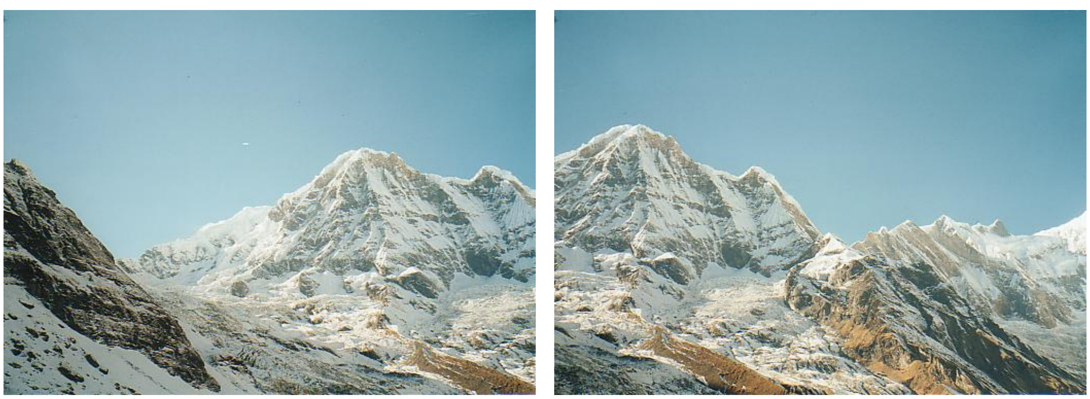
(a) Matier data set (7 images)

(b) Matier final stitch

[Brown, Szeliski, and Winder, CVPR 2005]

How do we build panorama?

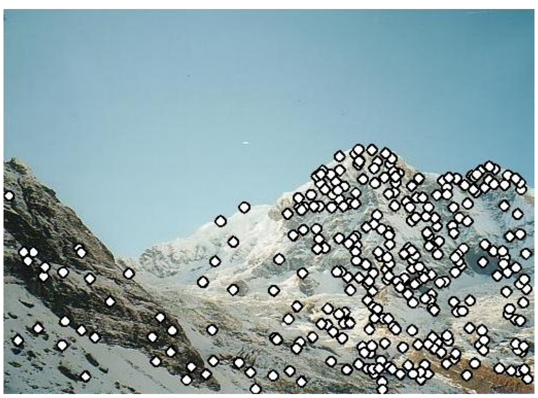
- We need to match (align) images

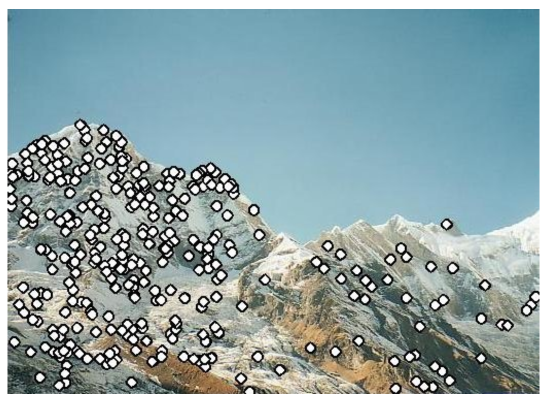


[Slide credit: Darya Frolova and Denis Simakov]

Matching with Features

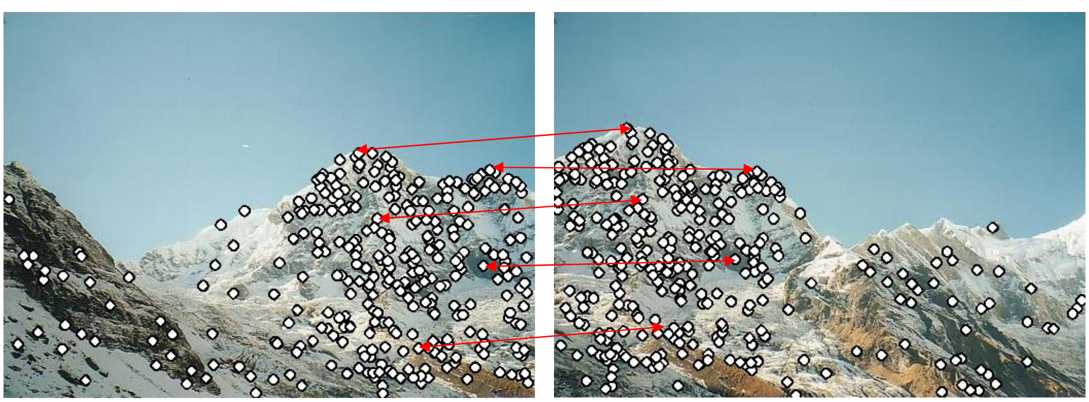
- Detect feature points in both images





Matching with Features

- Detect feature points in both images
- Find corresponding pairs



Matching with Features

- Detect feature points in both images
- Find corresponding pairs
- Use these pairs to align images

Matching with Features

- Problem 1:
  - Detect the *same* point *independently* in both images

no chance to match!

We need a repeatable detector

Matching with Features

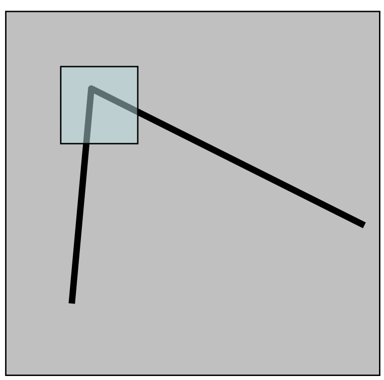
- (Problem 2:
  - For each point correctly recognize the corresponding one)

We need a reliable and distinctive descriptor

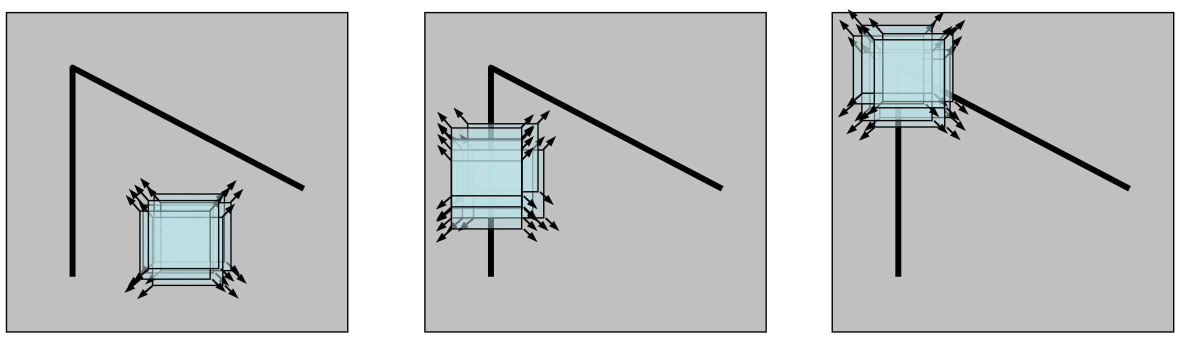
More on this aspect later!

Corner detection as an interest operator

- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



Corner detection as an interest operator



"flat" region:
no change in
all directions

"edge":
no change along
the edge direction

"corner":
significant change
in all directions

C.Harris, M.Stephens. "A Combined Corner and Edge Detector". 1988

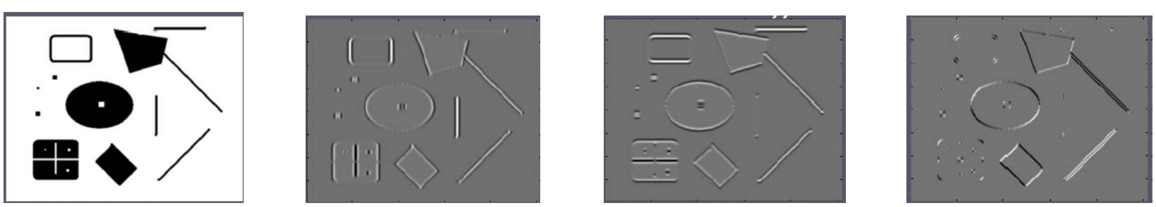
Corner Detection

M is a 2×2 matrix computed from image derivatives:

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

Gradient with respect to x, times gradient with respect to y

Sum over image region – area we are checking for corner

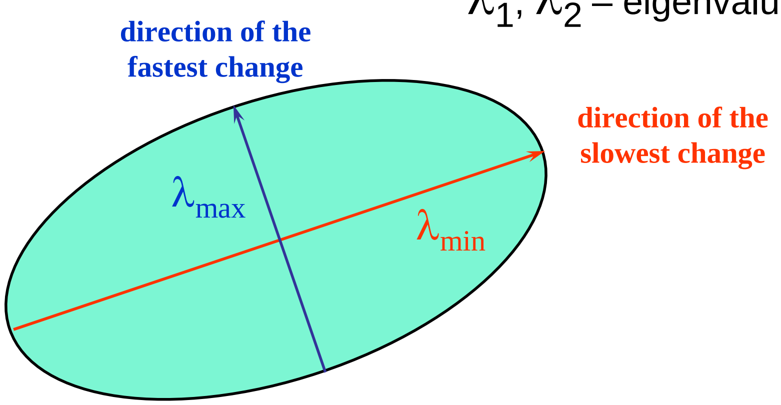


Corner Detection

Eigenvectors of M: encode edge directions

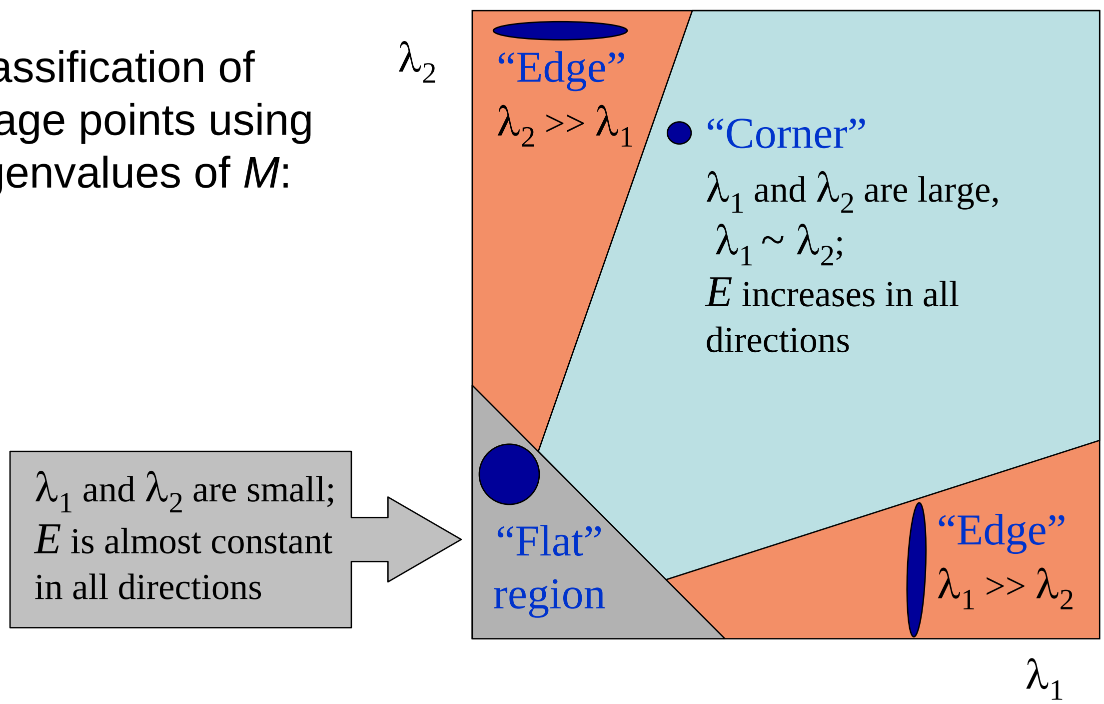
Eigenvalues of M: encode edge strength

λ_1, λ_2 – eigenvalues of M



Corner Detection

Classification of image points using eigenvalues of *M*:



Harris Corner Detector

Measure of corner response:

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

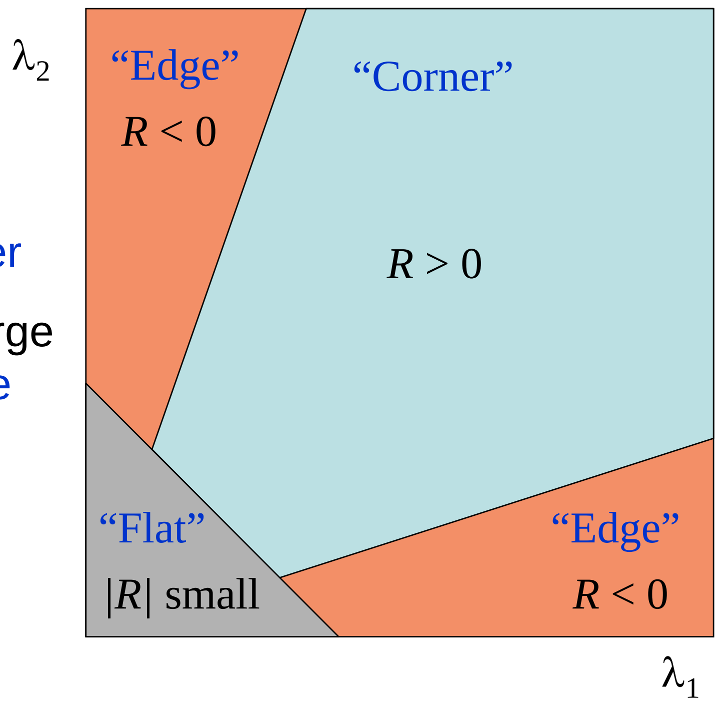
$$\det M = \lambda_1 \lambda_2$$
$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

Avoid computing eigenvalues themselves.

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

Harris Corner 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



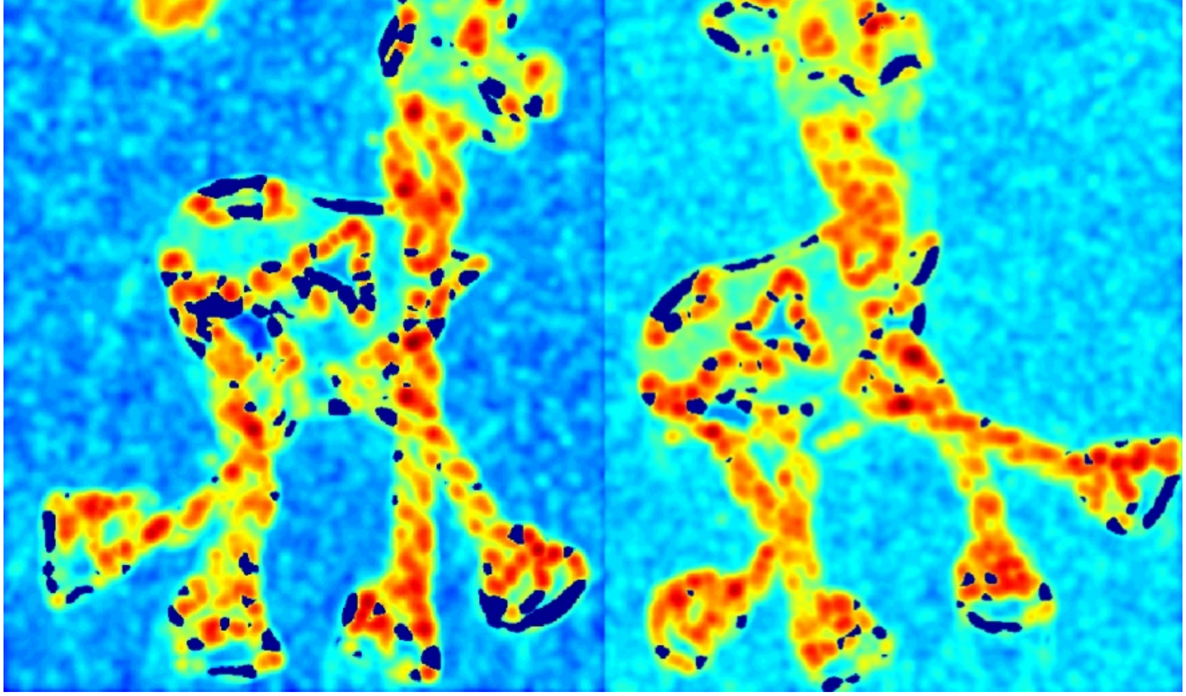
Harris Corner Detector

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

Harris Detector: Workflow

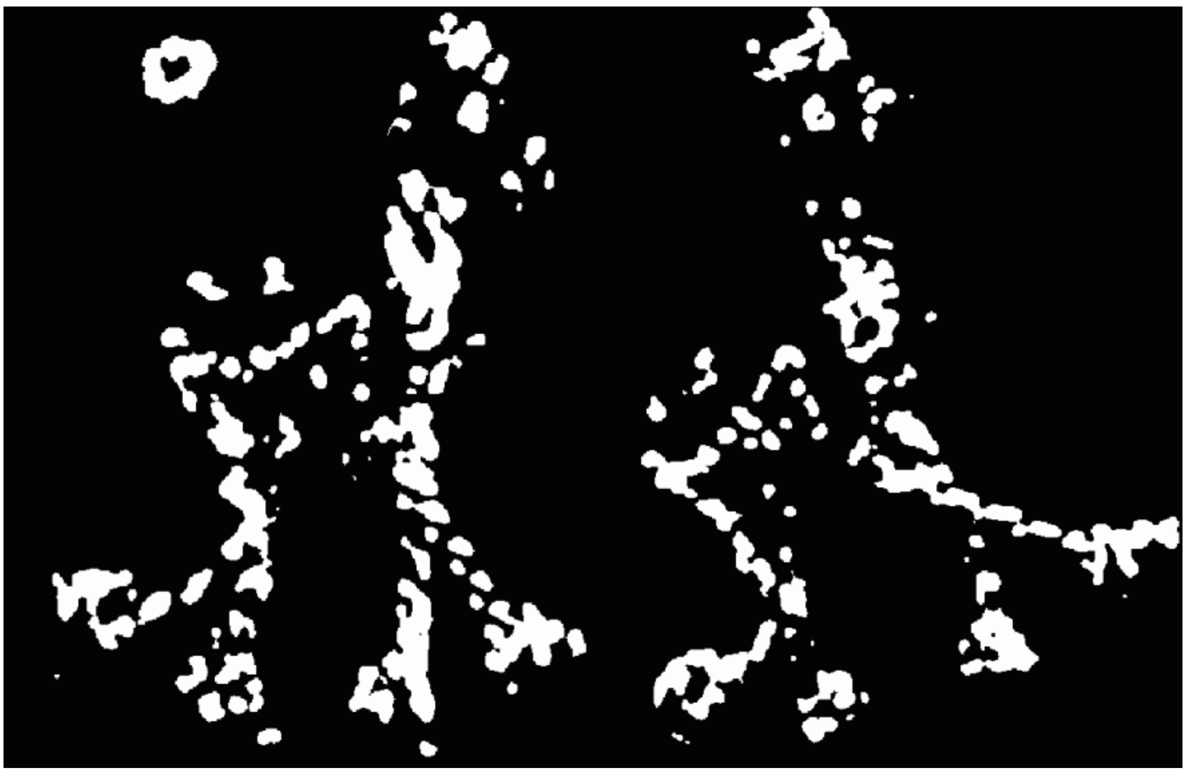
Harris Detector: Workflow

Compute corner response R



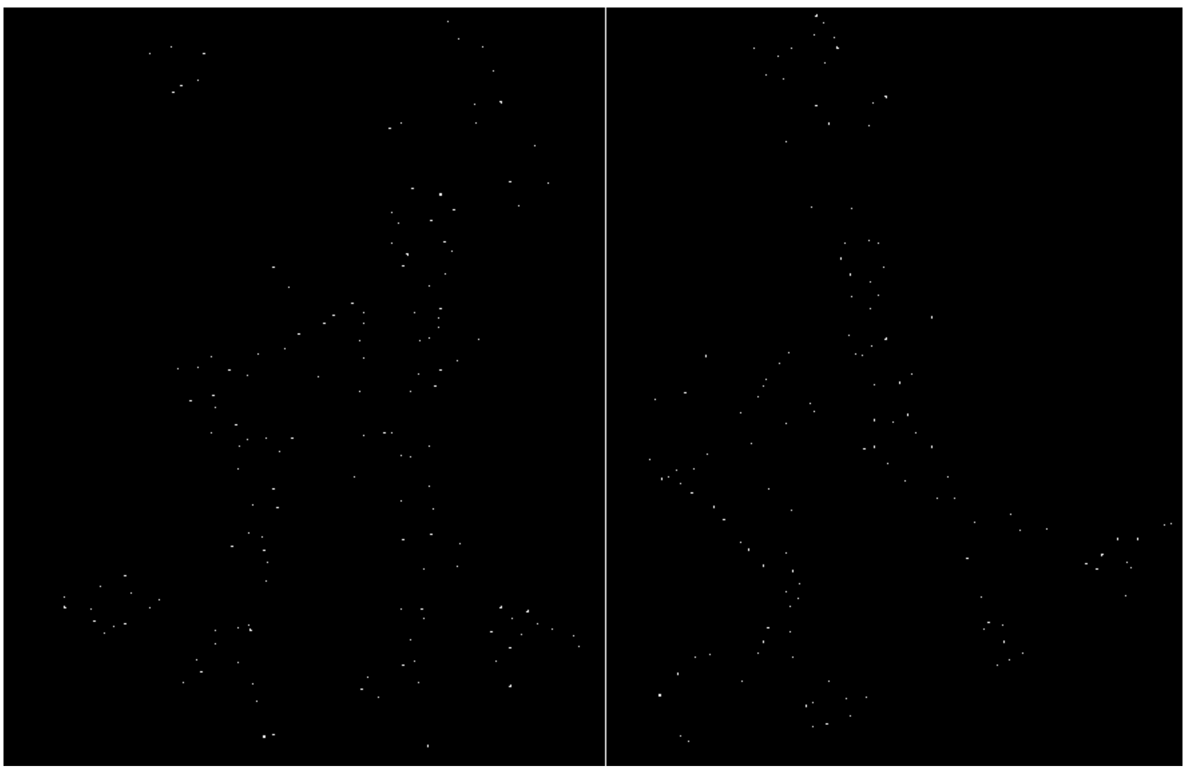
Harris Detector: Workflow

Find points with large corner response: R>threshold



Harris Detector: Workflow

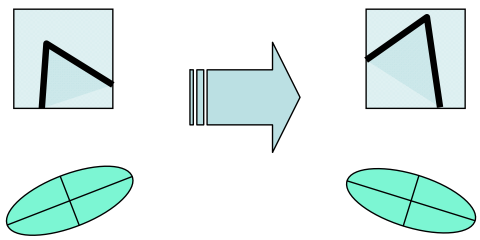
Take only the points of local maxima of R



Harris Detector: Workflow

Harris Detector: Some Properties

- Rotation invariance

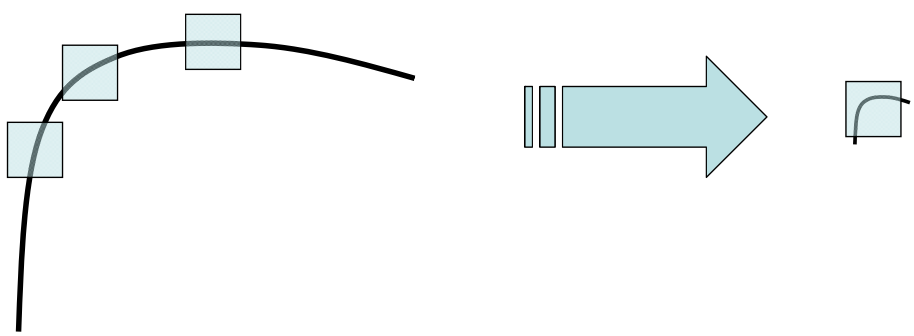


Ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner response R is invariant to image rotation

Harris Detector: Some Properties

- Not invariant to *image scale*!



All points will be
classified as edges

Corner !

More on interest operators/descriptors with invariance properties later.

This image is too big to fit on the screen. How can we reduce it?

How to generate a half-sized version?

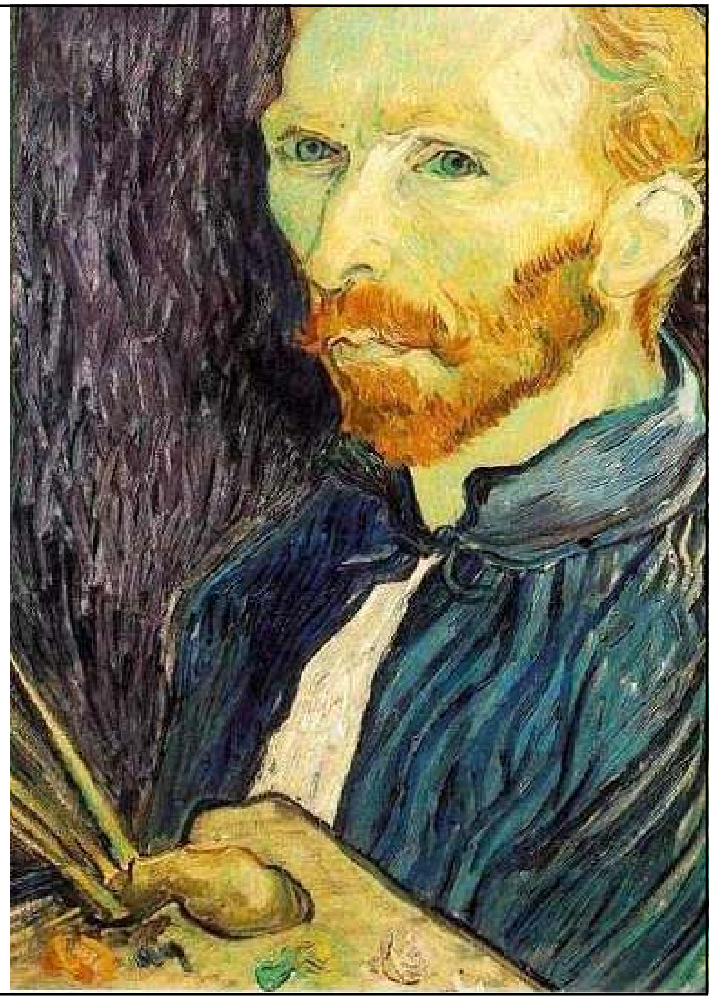
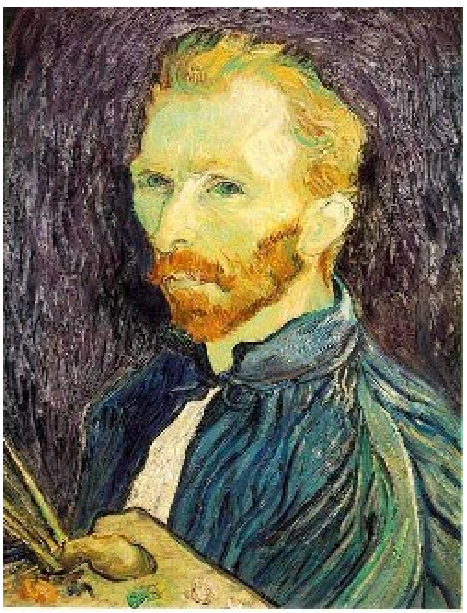


Image sub-sampling



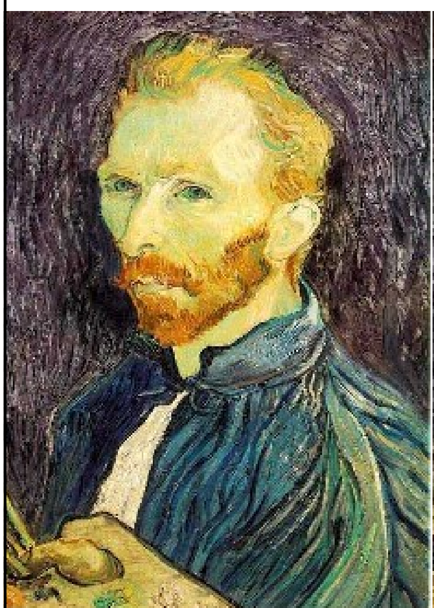
1/4

1/8

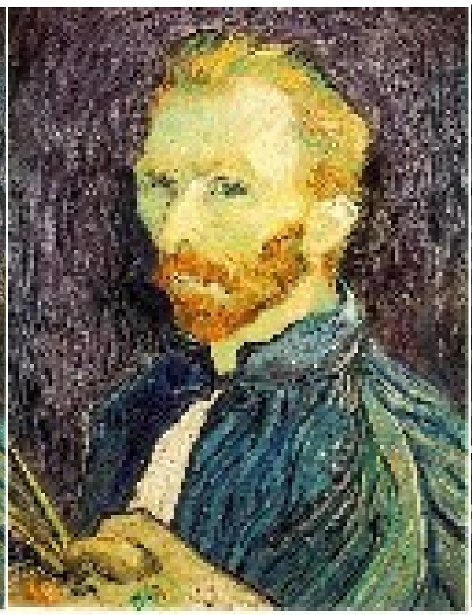
Throw away every other row and column to create a 1/2 size image
- called *image sub-sampling*

Slide credit: S. Seitz

Image sub-sampling



1/2

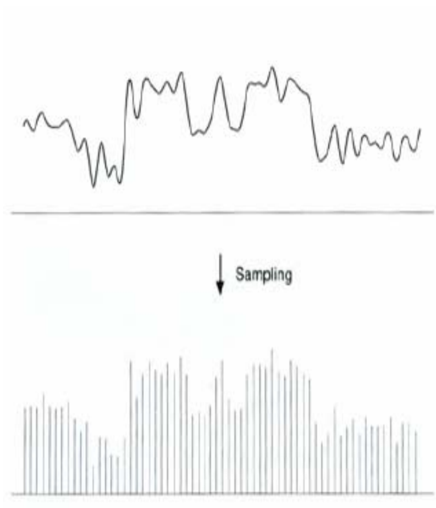


1/4 (2x zoom)

1/8 (4x zoom)

Sampling

- Continuous function → discrete set of values



Undersampling

- Information lost

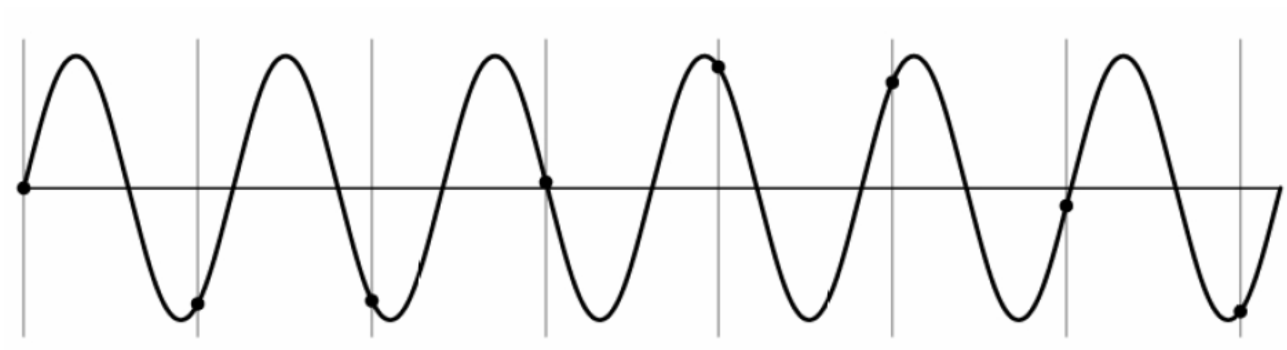
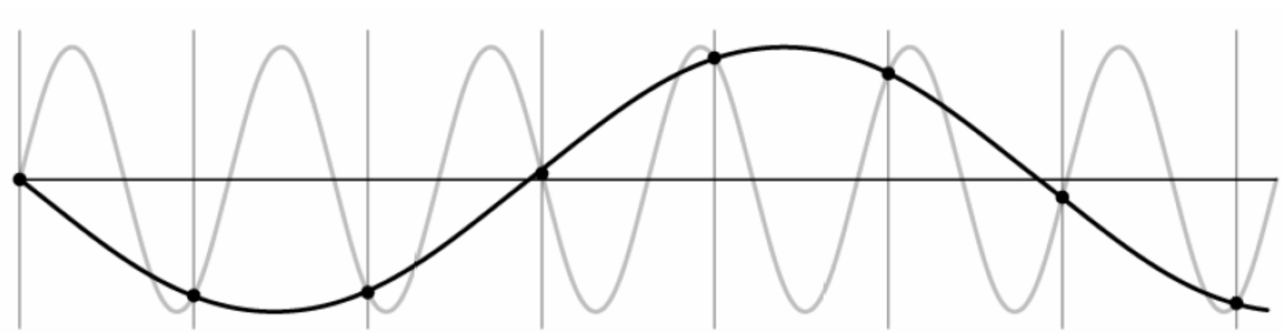


Figure credit: S. Marschner

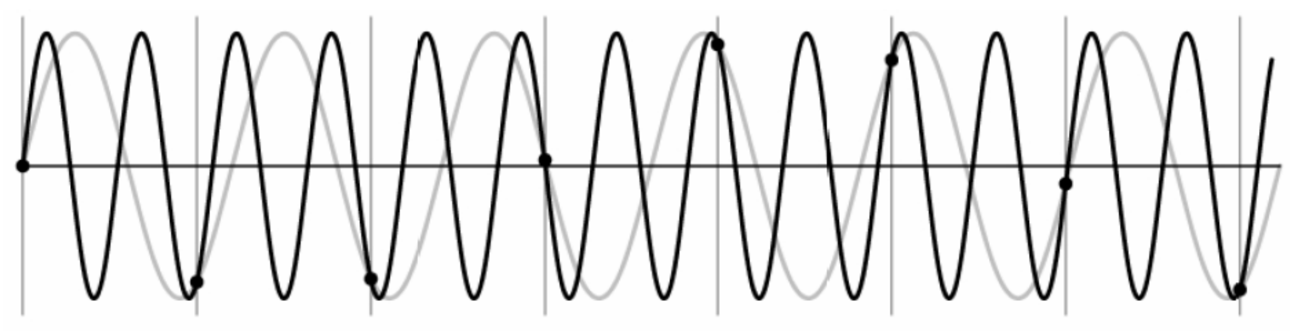
Undersampling

- Looks just like lower frequency signal!



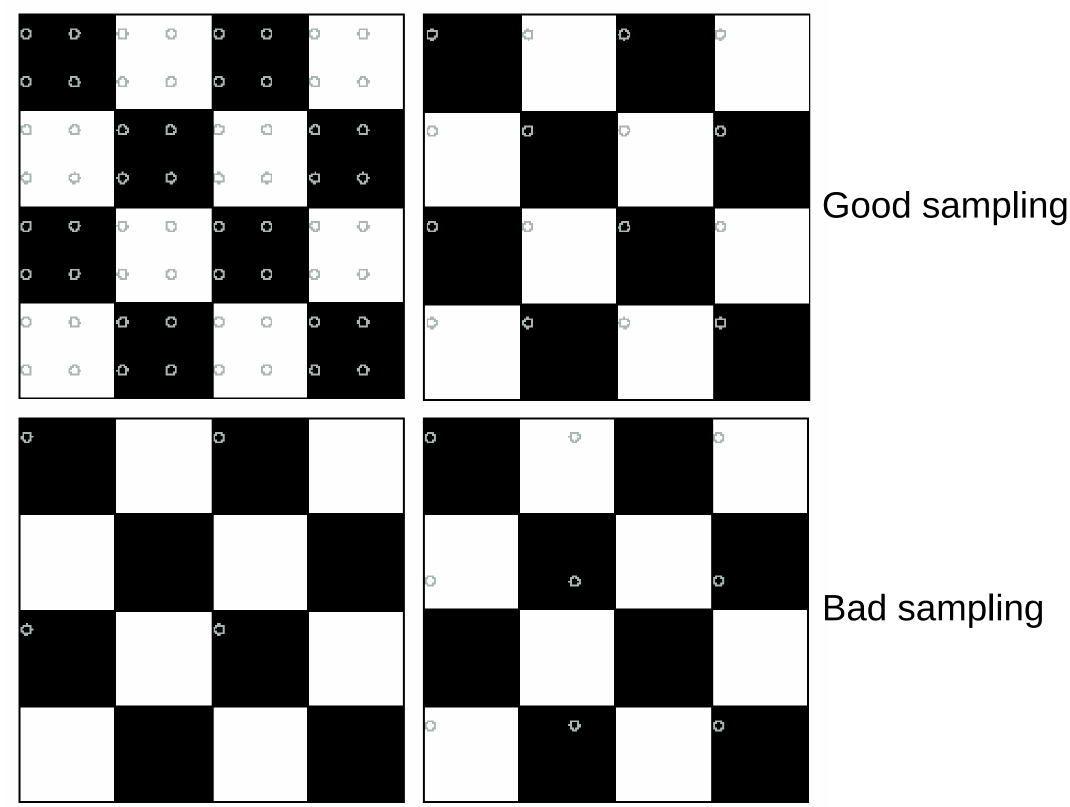
Undersampling

- Looks like higher frequency signal!

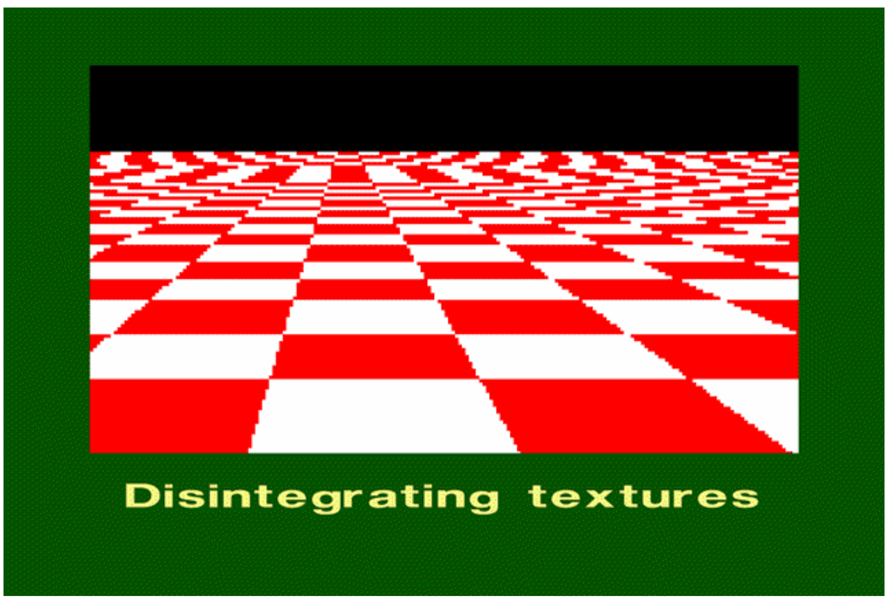


Aliasing: higher frequency information can appear as lower frequency information

Undersampling

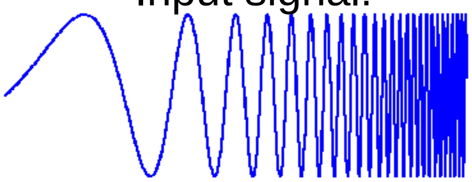


Aliasing

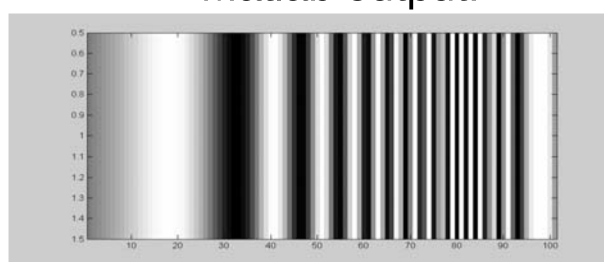


Aliasing

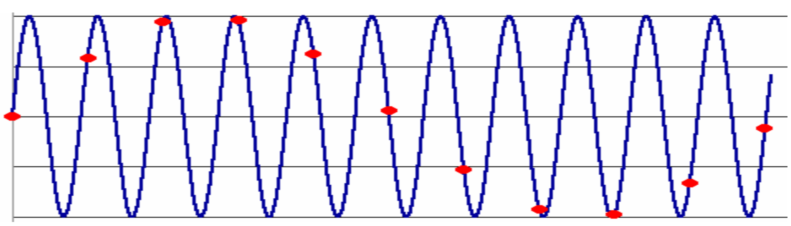
Input signal:



Matlab output:



x = 0:.05:5; imagesc(sin((2.^x).*x))

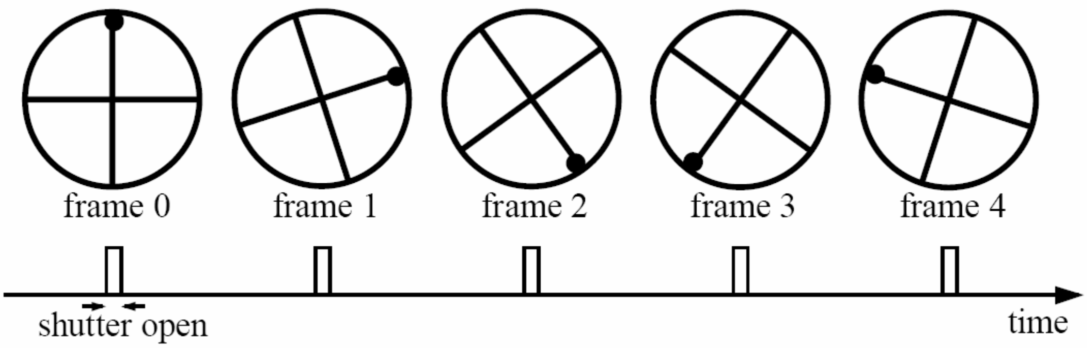


Not enough samples

Aliasing in video

Imagine a spoked wheel moving to the right (rotating clockwise). Mark wheel with dot so we can see what's happening.

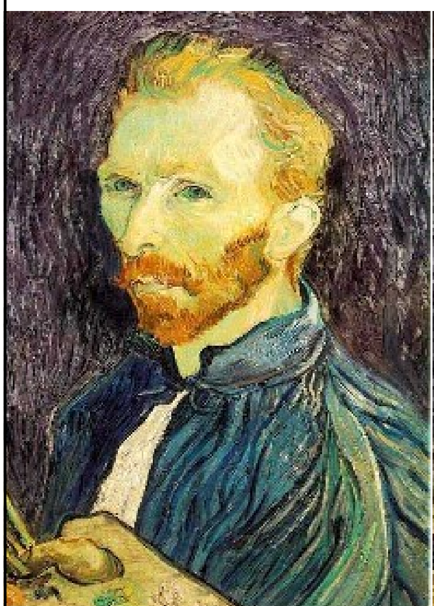
If camera shutter is only open for a fraction of a frame time (frame time = 1/30 sec. for video, 1/24 sec. for film):



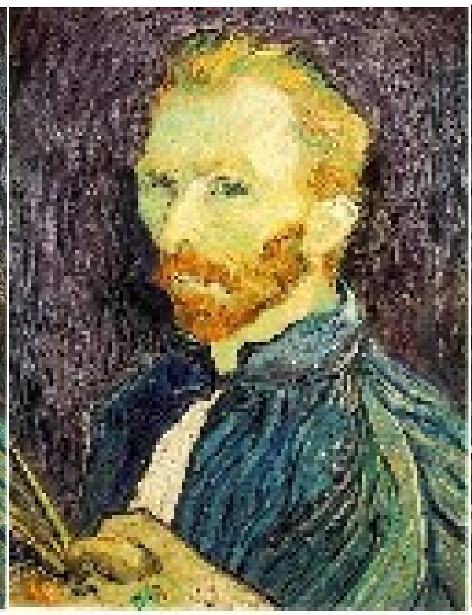
Without dot, wheel appears to be rotating slowly backwards! (counterclockwise)

Slide credit: S. Seitz

Image sub-sampling



1/2



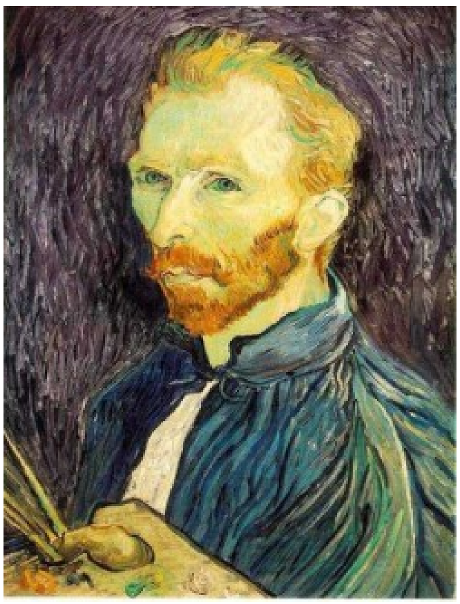
1/4 (2x zoom)

1/8 (4x zoom)

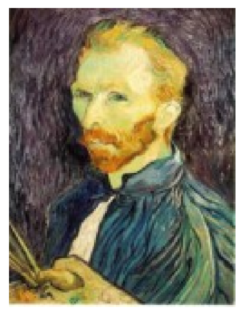
How to prevent aliasing?

- Sample more …
- Smooth – suppress high frequencies before sampling

Gaussian pre-filtering



Gaussian 1/2

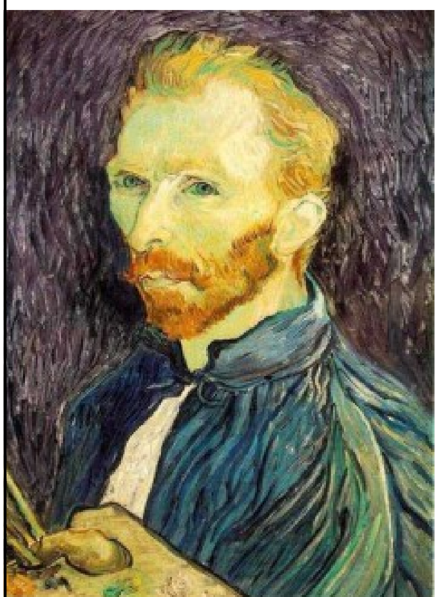


G 1/4

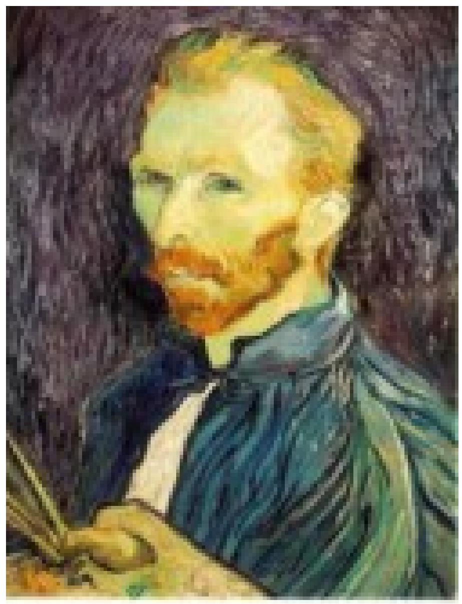
G 1/8

Solution: smooth the image, *then* subsample

Subsampling with Gaussian pre-filtering



Gaussian 1/2

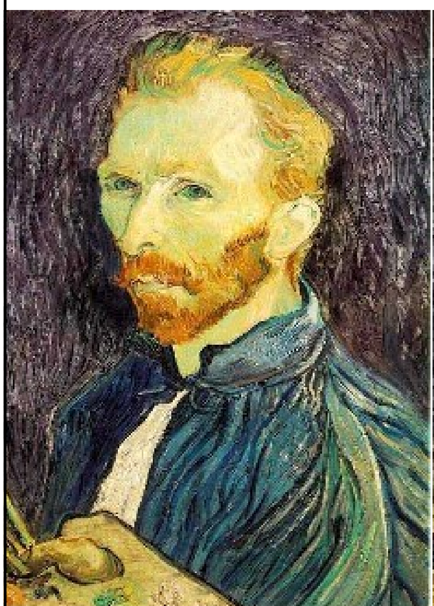


G 1/4

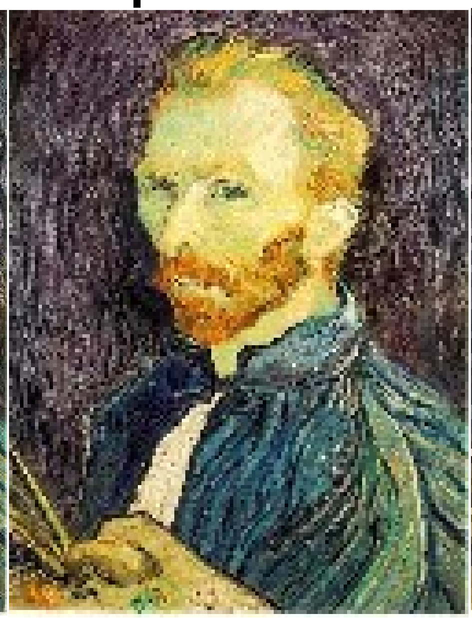
G 1/8

Solution: smooth the image, *then* subsample

Compare with...



1/2



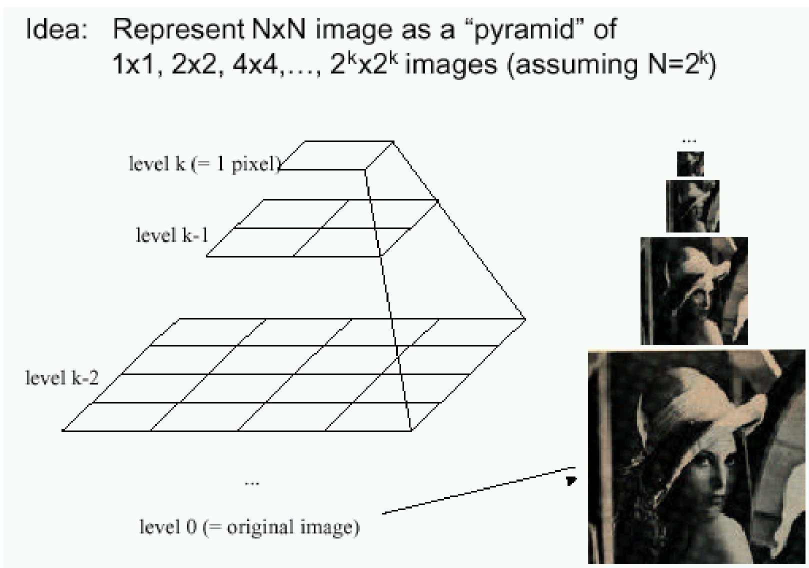
1/4 (2x zoom)

1/8 (4x zoom)

Image pyramids

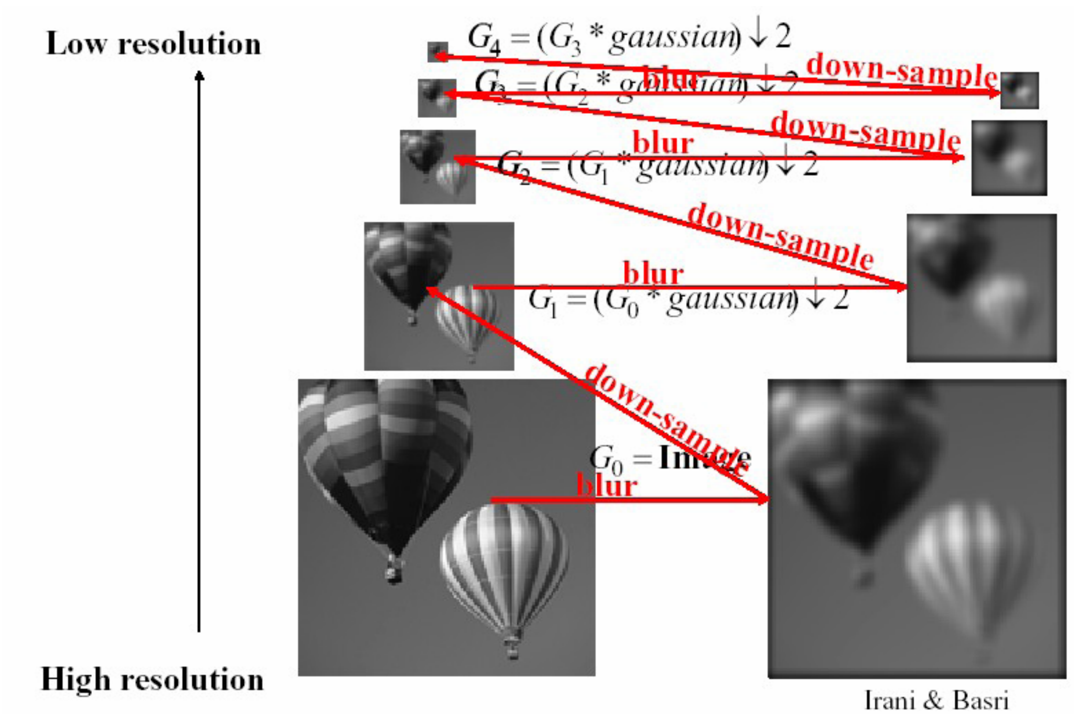
- Big bars (resp. spots, hands, etc.) and little bars are both interesting
- Inefficient to detect big bars with big filters
- Alternative:
  – Apply filters of fixed size to images of different sizes

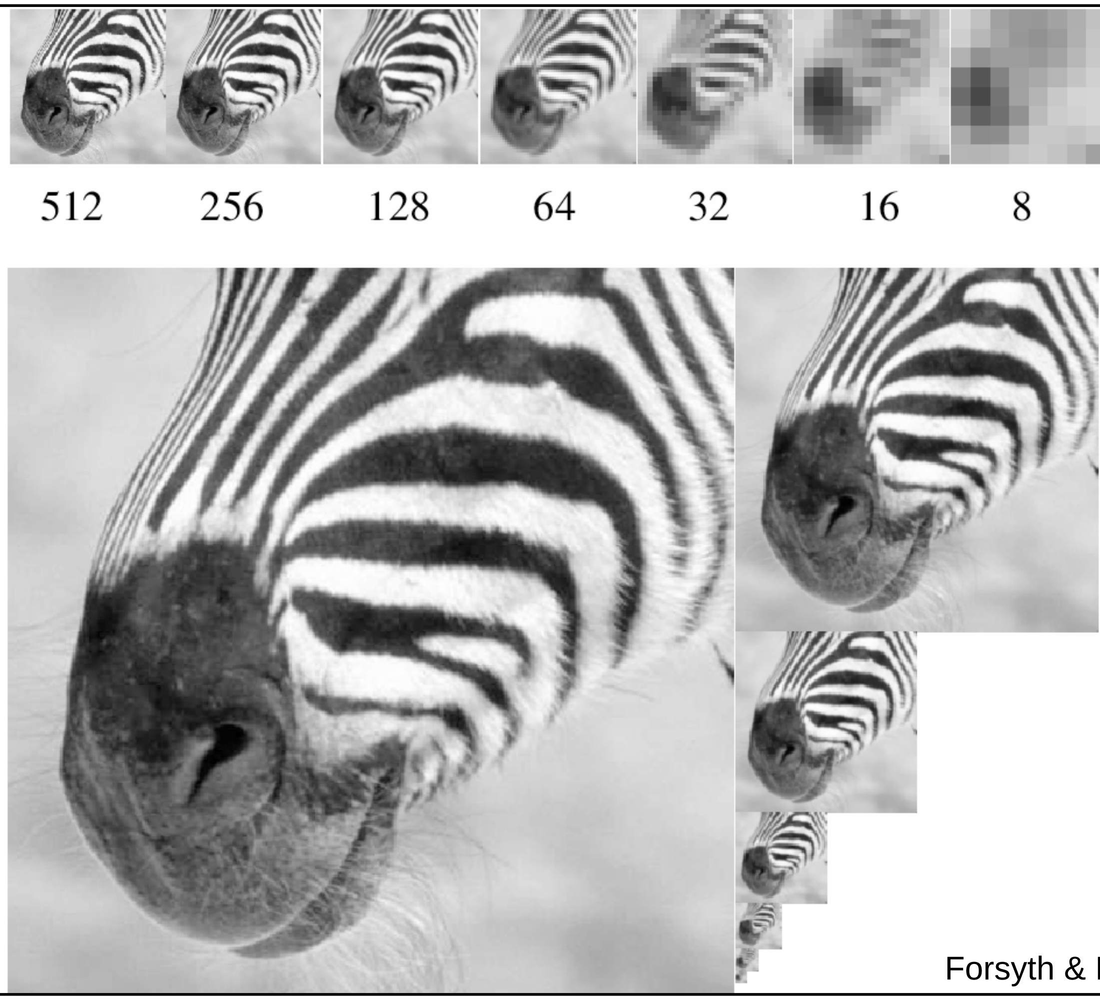
Image pyramids



- Known as a **Gaussian Pyramid** [Burt and Adelson, 1983]

Gaussian image pyramids



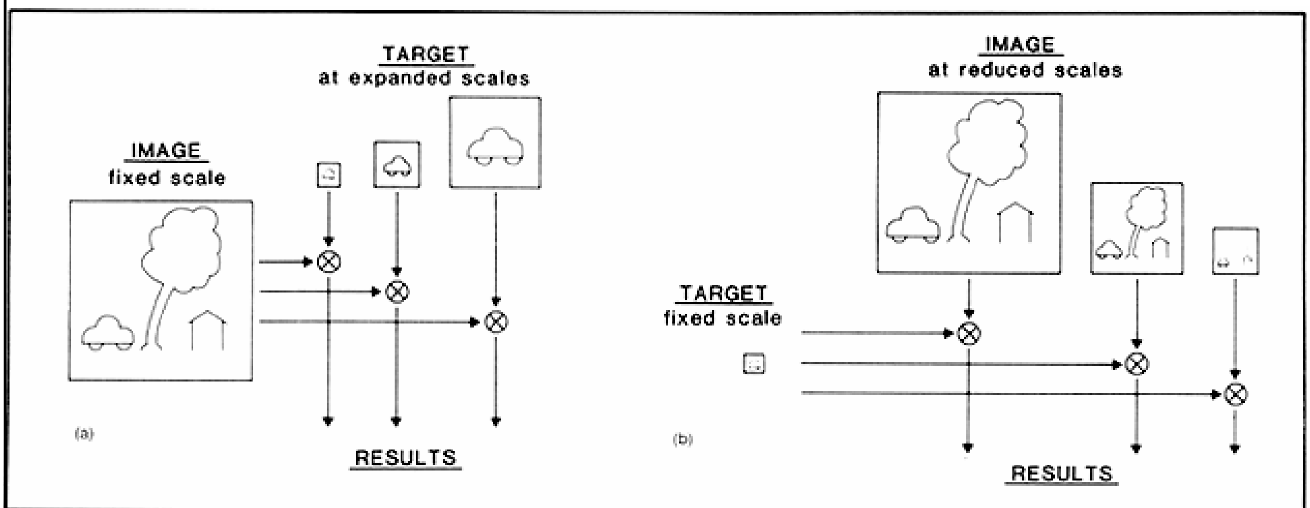


Forsyth & Ponce

Image pyramids

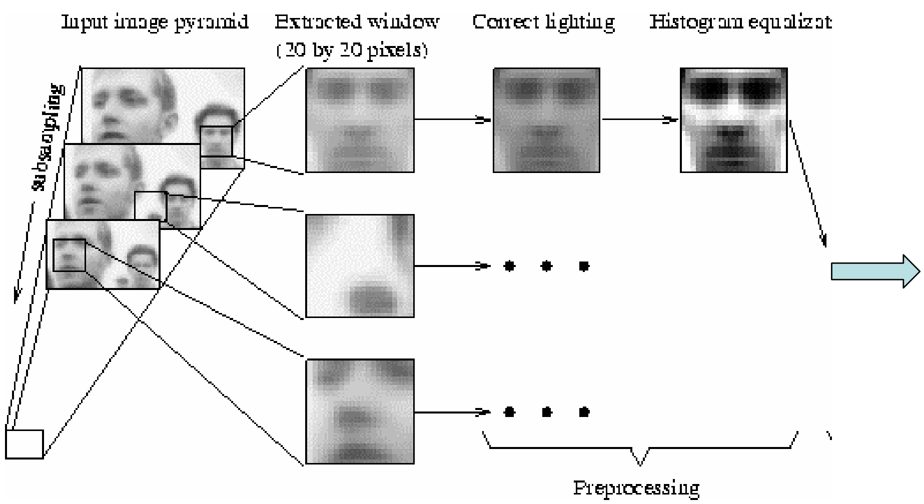
- Useful for
  - Coarse to fine matching, iterative computation; e.g. optical flow
  - Feature association across scales to find reliable features
  - Searching over scale

Image pyramids: multi-scale search



[Adelson et al., 1984]

Image pyramids: multi-scale search



Learning algorithm

Figure from Rowley et al. 1998