Edge and Texture

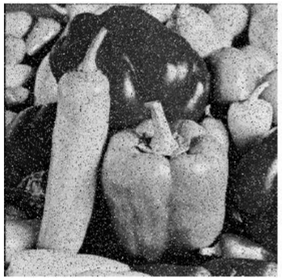
CMP19– Computer Vision

Pinar Duygulu

Hacettepe University

Filters for features

- Previously, thinking of filtering as a way to remove or reduce **noise**
- Now, consider how filters will allow us to abstract higher-level "**features**".
  - Map raw pixels to an intermediate representation that will be used for subsequent processing
  - Goal: reduce amount of data, discard redundancy, preserve what's useful



Source: Darrell, Berkeley

Edge detection

- **Goal:** Identify sudden changes (discontinuities) in an image
  - Intuitively, most semantic and shape information from the image can be encoded in the edges
  - More compact than pixels
- **Ideal:** artist's line drawing (but artist is also using object-level knowledge)

Source: D. Lowe

Source: Hays, Brown

Edge detection

- **Goal**: map image from 2d array of pixels to a set of curves or line segments or contours.
- **Why?**

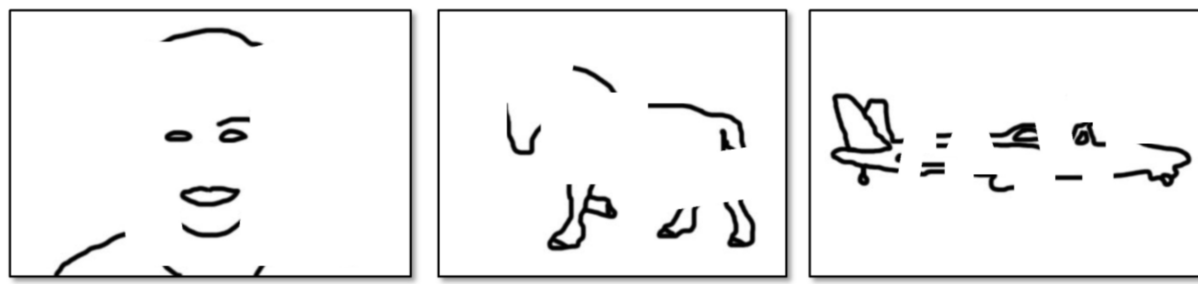


Figure from J. Shotton et al., PAMI 2007

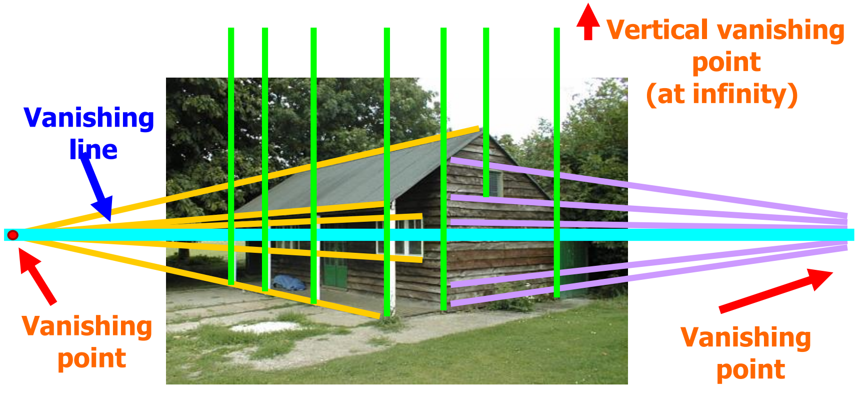
- **Main idea**: look for strong gradients, post-process

Source: Darrell, Berkeley

Why do we care about edges?

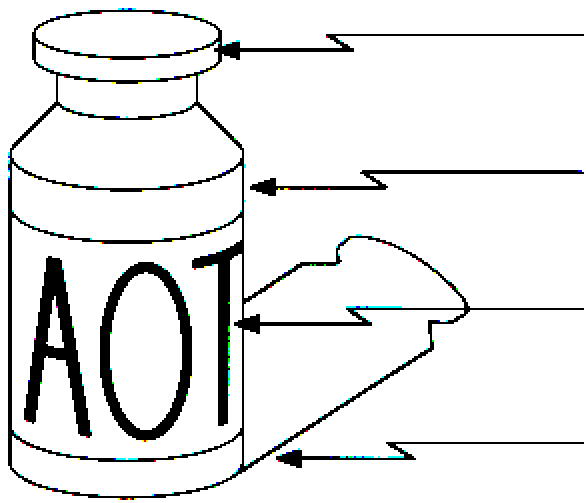
- Extract information, recognize objects

- Recover geometry and viewpoint



Source: Hays, Brown

Origin of Edges



- Edges are caused by a variety of factors

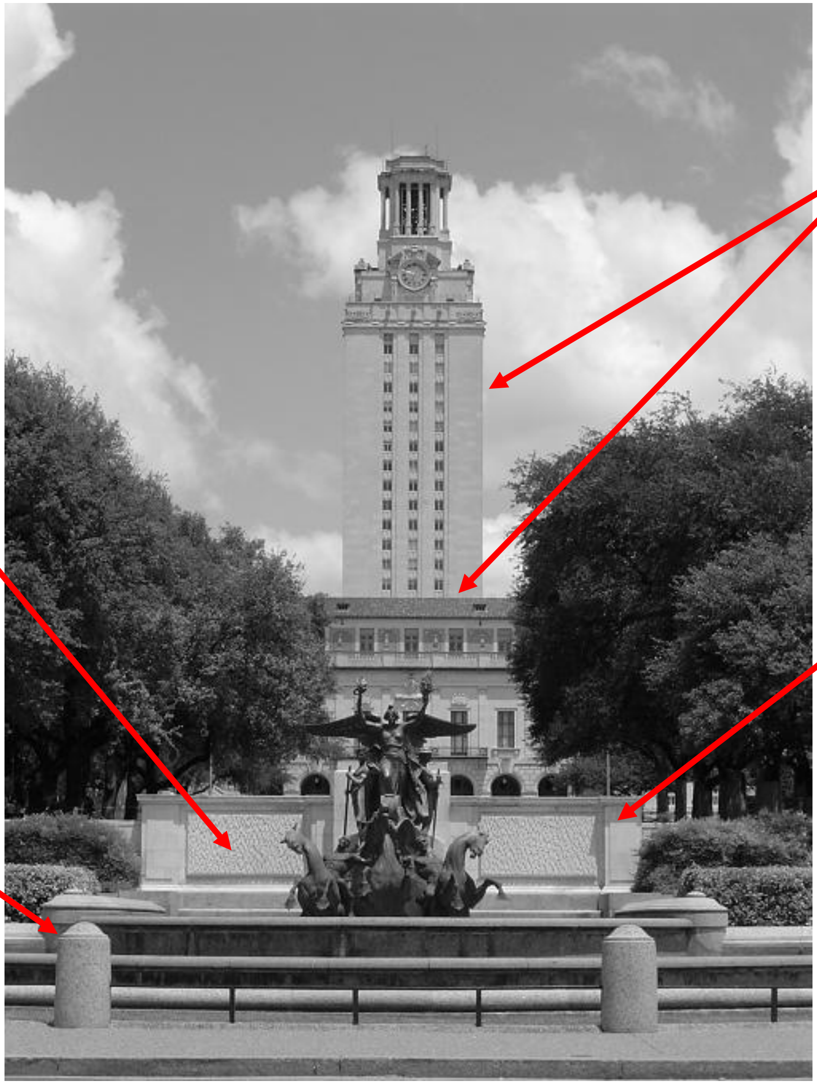
Source: Steve Seitz

Source: Hays, Brown

What can cause an edge?

Reflectance change: appearance information, texture

Change in surface orientation: shape

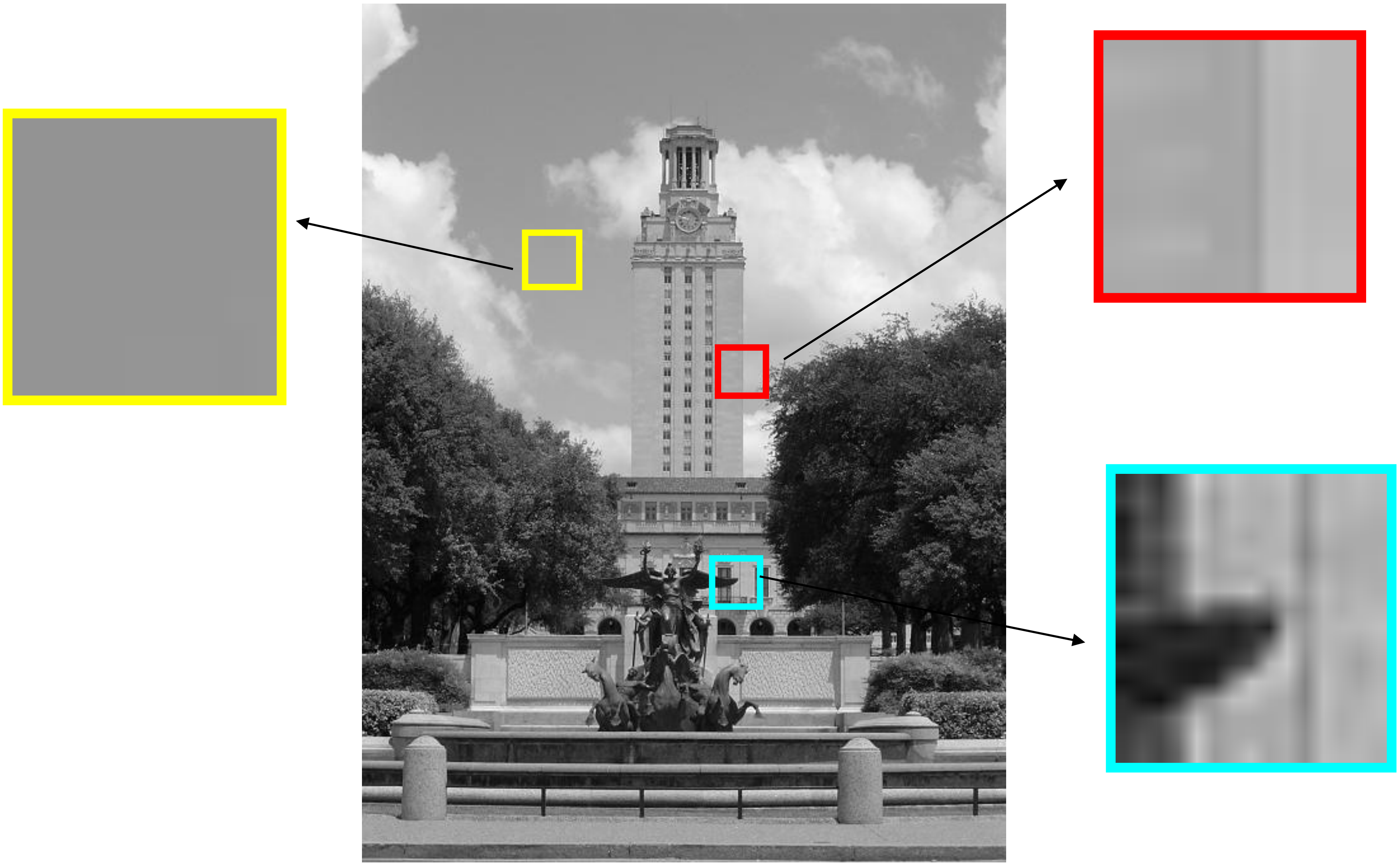


Depth discontinuity: object boundary

Cast shadows

Source: Darrell, Berkeley

Contrast and invariance



Source: Darrell, Berkeley

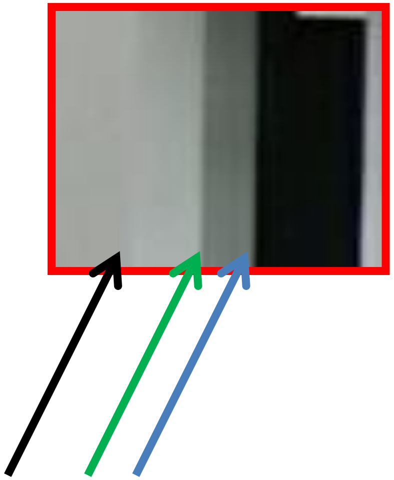
Closeup of edges

Source: Hays, Brown

Source: D. Hoiem

Closeup of edges

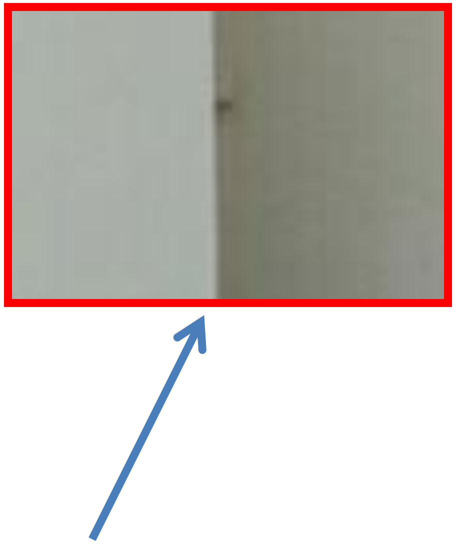
Source: Hays, Brown



Source: D. Hoiem

Closeup of edges

Source: Hays, Brown



Source: D. Hoiem

Closeup of edges

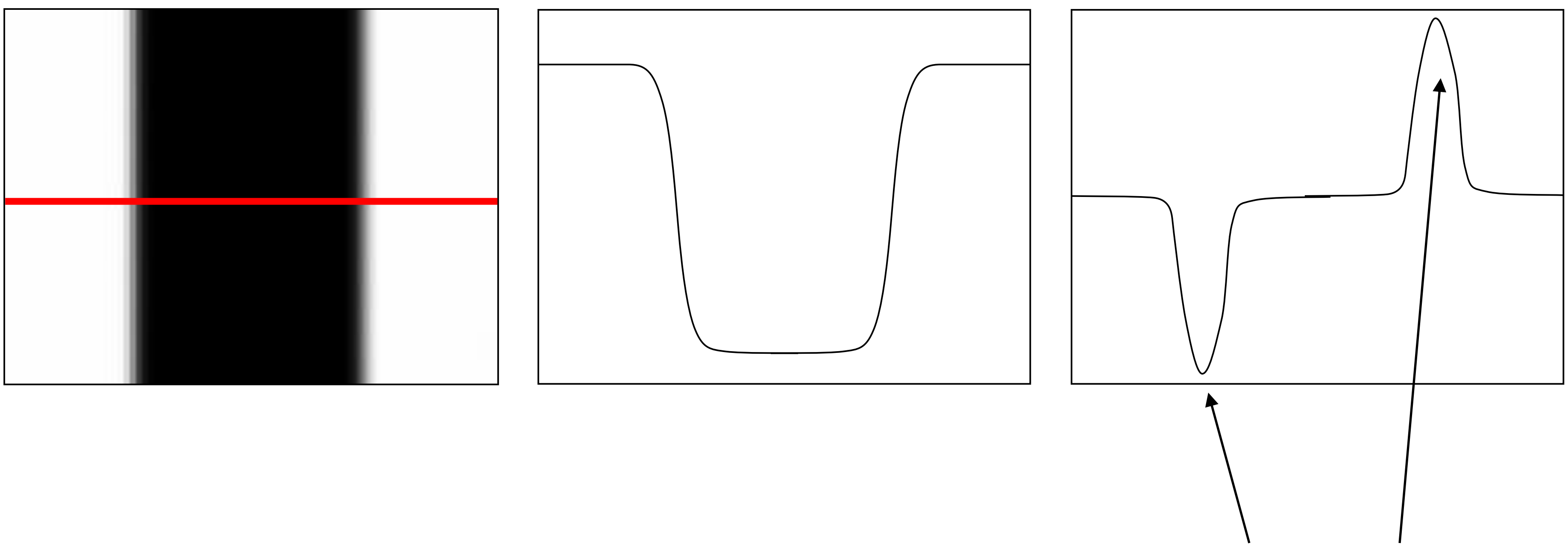
Source: Hays, Brown

Source: D. Hoiem

Characterizing edges

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

image | intensity function (along horizontal scanline) | first derivative



edges correspond to extrema of derivative

Source: Hays, Brown

Differentiation and convolution

For 2D function, f(x,y), the partial derivative is:

$$\frac{\partial f(x,y)}{\partial x} = \lim_{\varepsilon \to 0} \frac{f(x+\varepsilon, y) - f(x,y)}{\varepsilon}$$

For discrete data, we can approximate using finite differences:

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

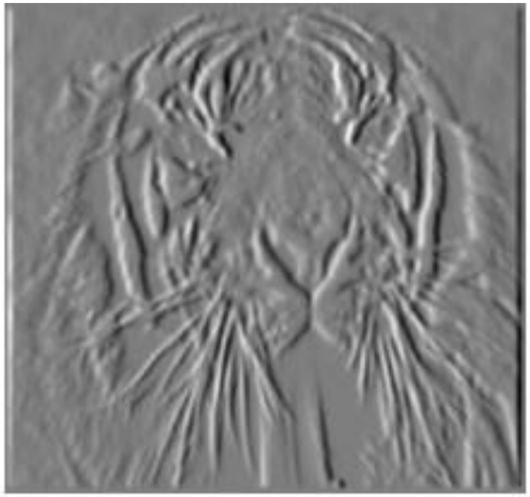
To implement above as convolution, what would be the associated filter?

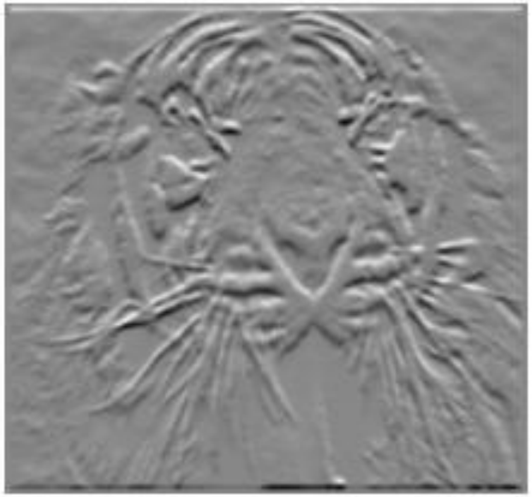
Source: Darrell, Berkeley

Partial derivatives of an image

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

-1	1





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

-1
1

? or

1
-1

Which shows changes with respect to x?

(showing flipped filters)

Source: Darrell, Berkeley

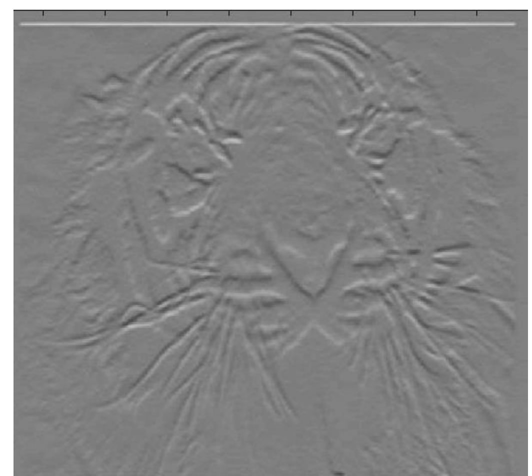
Assorted finite difference filters

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

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

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

```
>> My = fspecial('sobel');
>> outim = imfilter(double(im), My);
>> imagesc(outim);
>> colormap gray;
```



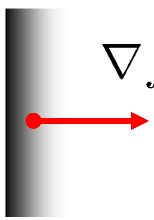
Source: Darrell, Berkeley

Image gradient

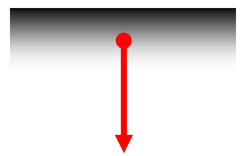
The gradient of an image:

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

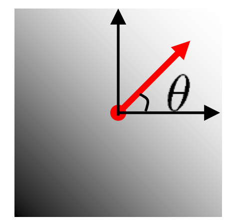
The gradient points in the direction of most rapid change in intensity



$$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$$



$$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$$



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

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

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

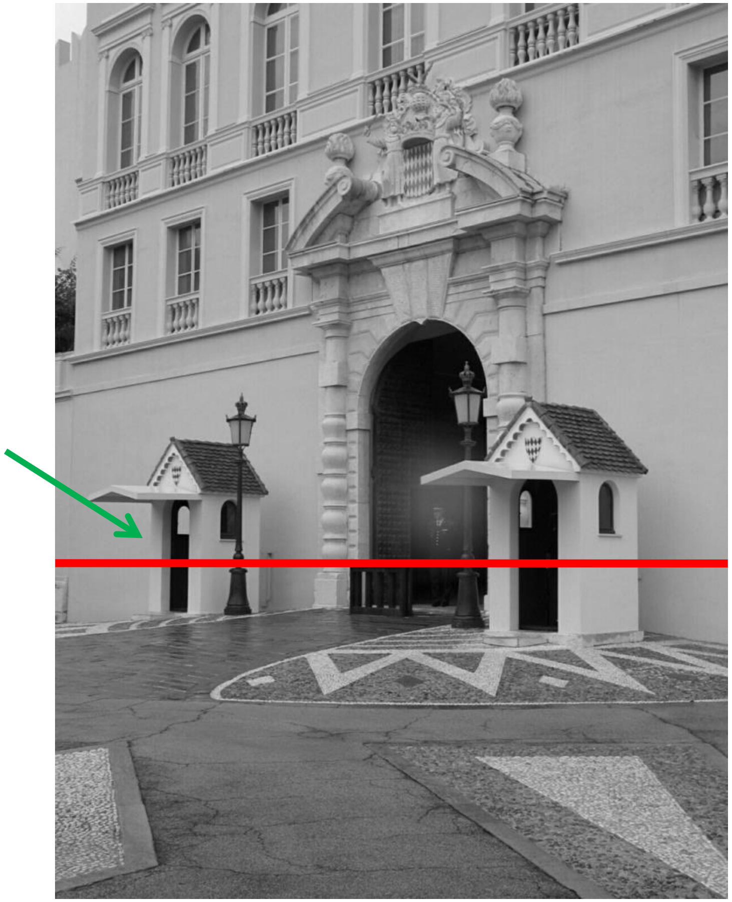
The *edge strength* is given by the gradient magnitude

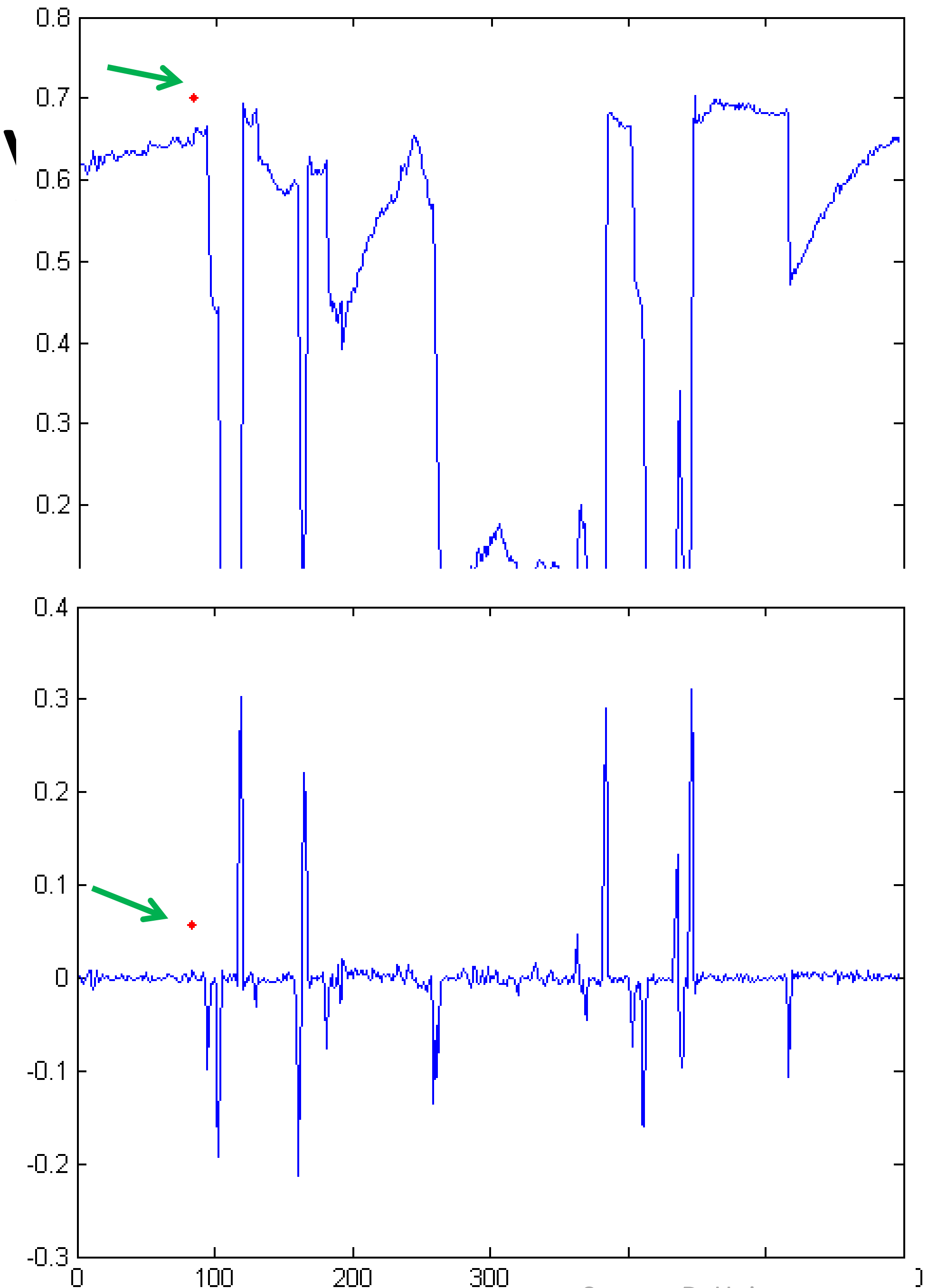
$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Source: Darrell, Berkeley

Slide credit S. Seitz

Intensit



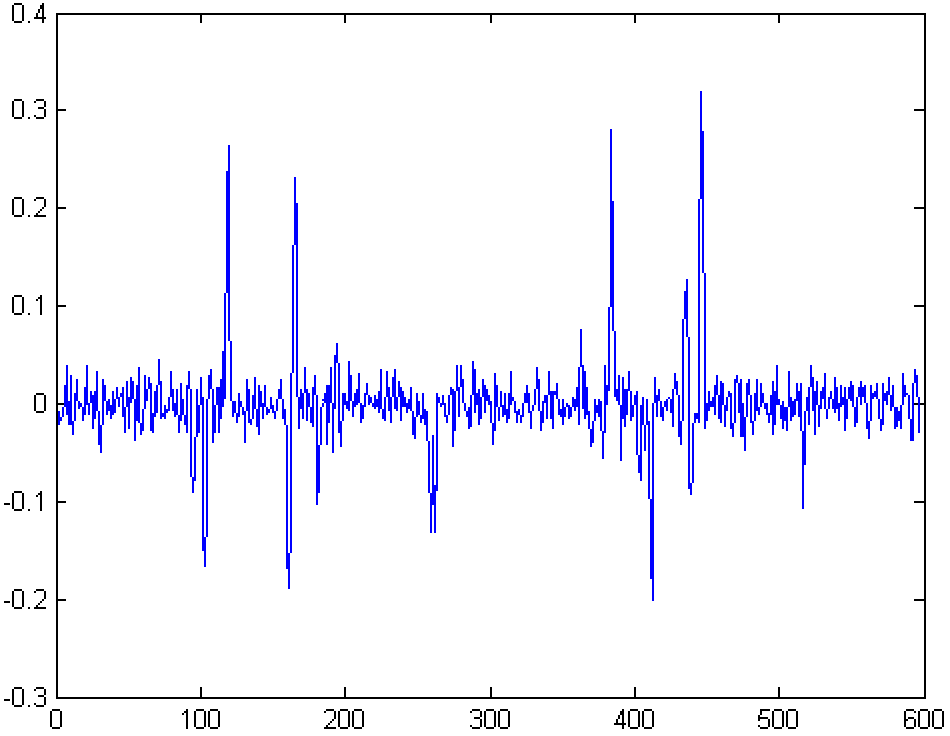


Source: D. Hoiem

Source: Hays, Brown

With a little Gaussian noise

Source: Hays, Brown

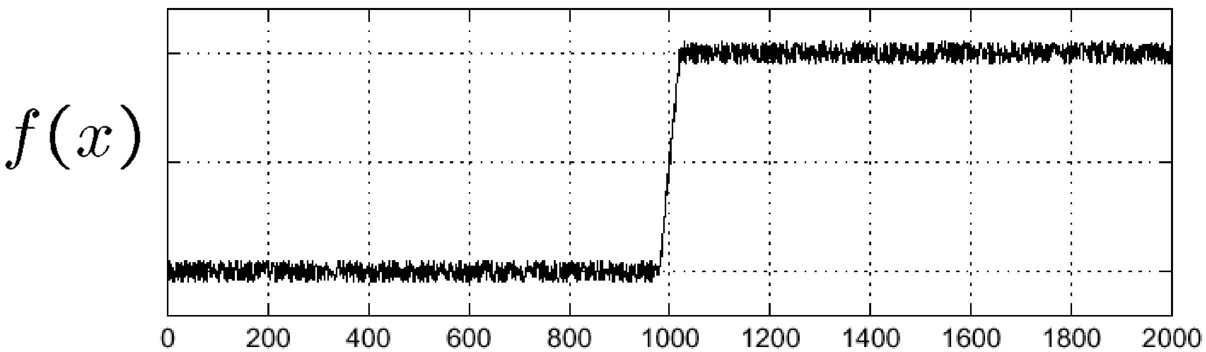


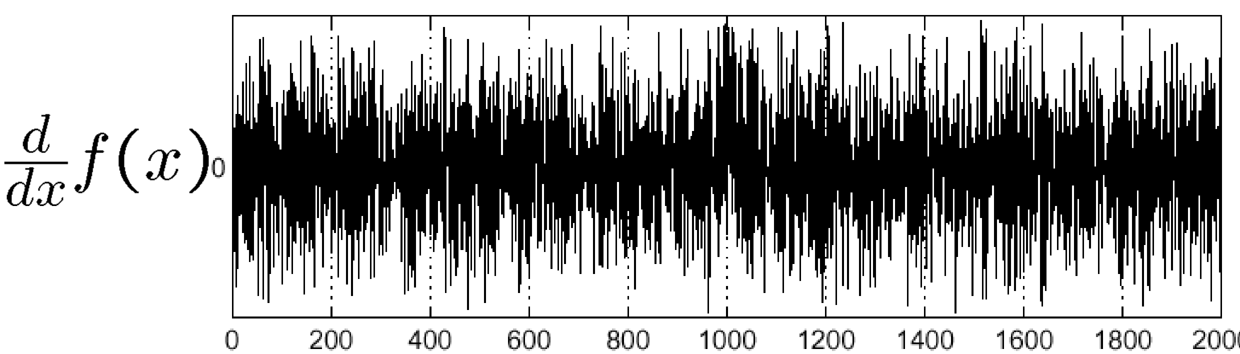
Gradient

Source: D. Hoiem

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?

Source: S. Seitz

Source: Hays, Brown

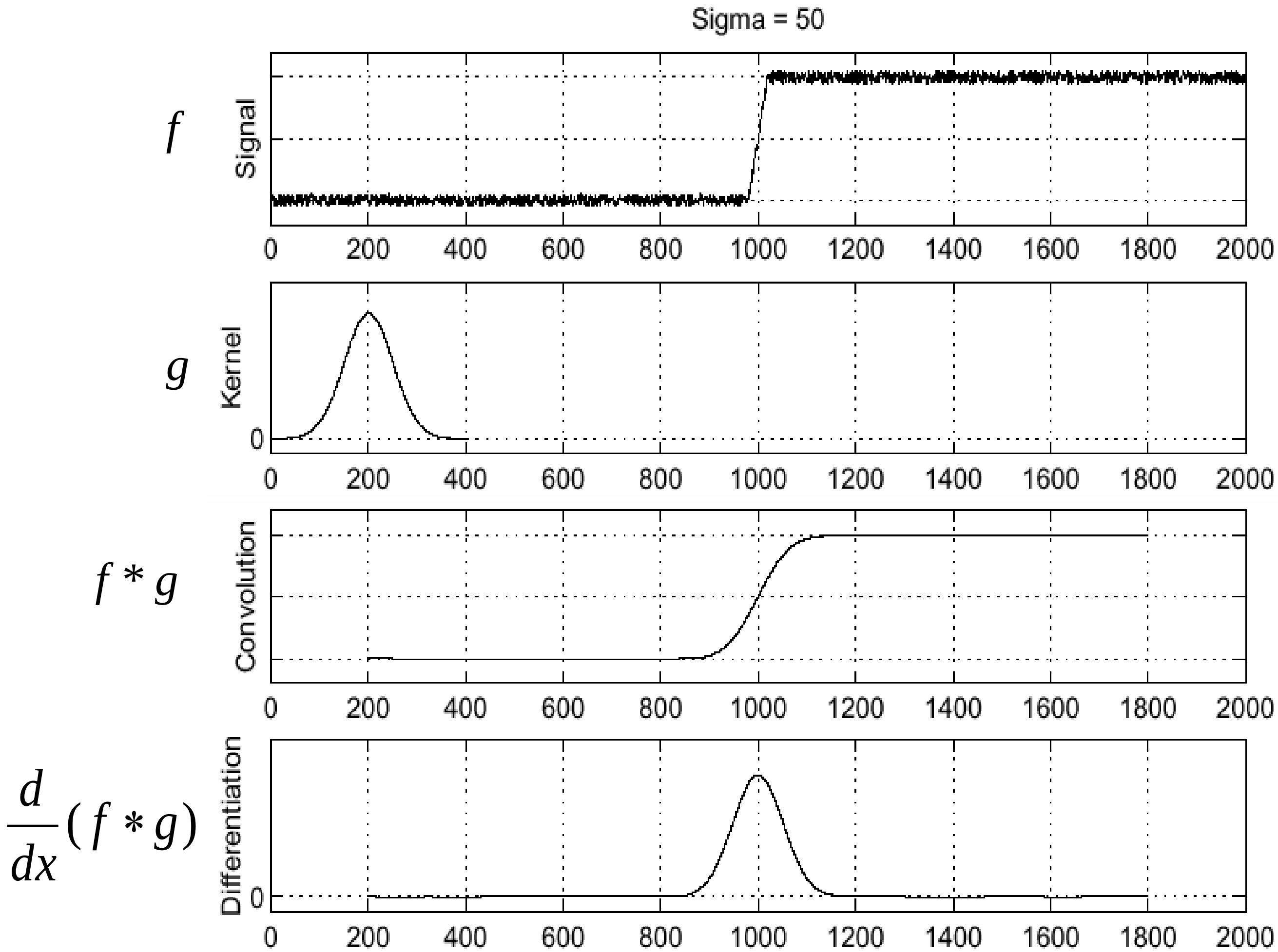
Effects of noise

- Difference filters respond strongly to noise
  - Image noise results in pixels that look very different from their neighbors
  - Generally, the larger the noise the stronger the response
- What can we do about it?

Source: D. Forsyth

Source: Hays, Brown

Solution: smooth first



- To find edges, look for peaks in $\frac{d}{dx}(f * g)$

Source: S. Seitz

Source: Hays, Brown

Derivative theorem of convolution

- Differentiation is convolution, and convolution is associative:

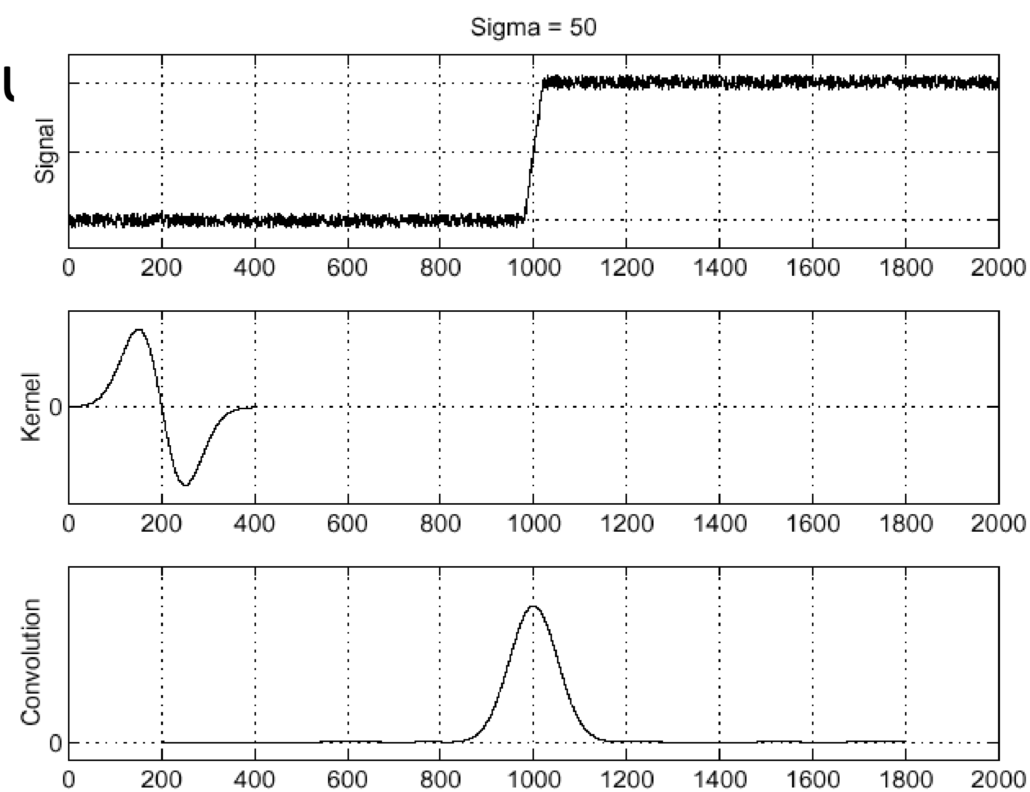
$$\frac{d}{dx}(f * g) = f * \frac{d}{dx} g$$

- This saves u

f

$\frac{d}{dx} g$

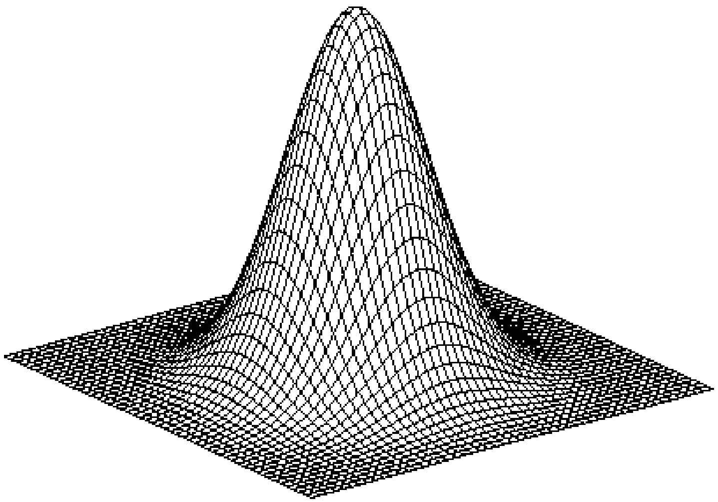
$f * \frac{d}{dx} g$



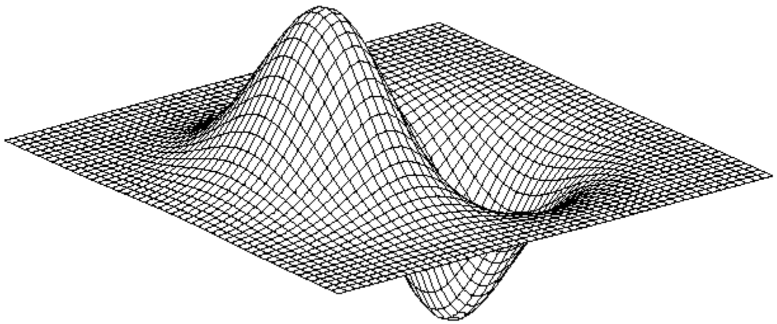
Source: S. Seitz

Source: Hays, Brown

Derivative of Gaussian filter

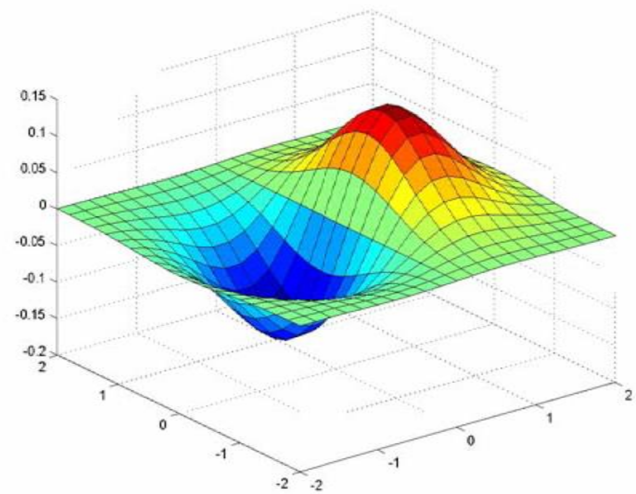


* [1 -1] =

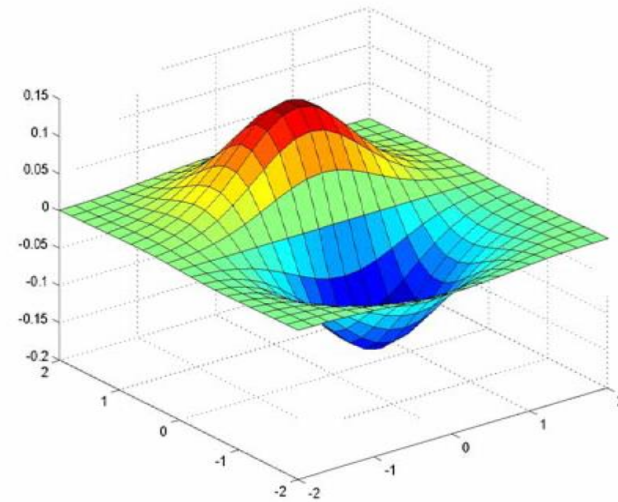


Source: Hays, Brown

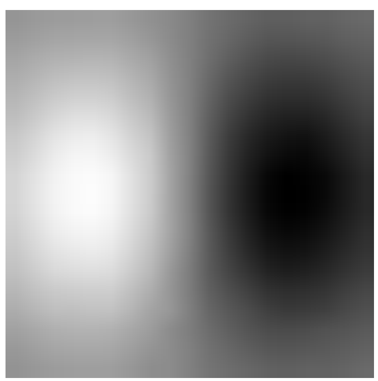
Derivative of Gaussian filters

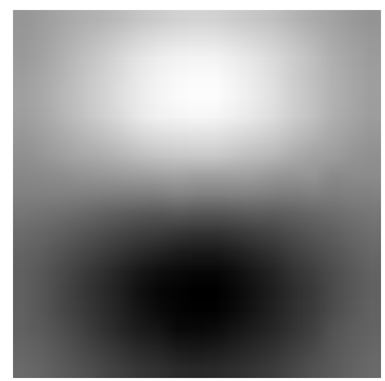


x-direction



y-direction





Source: Darrell, Berkeley

Source: L. Lazebnik

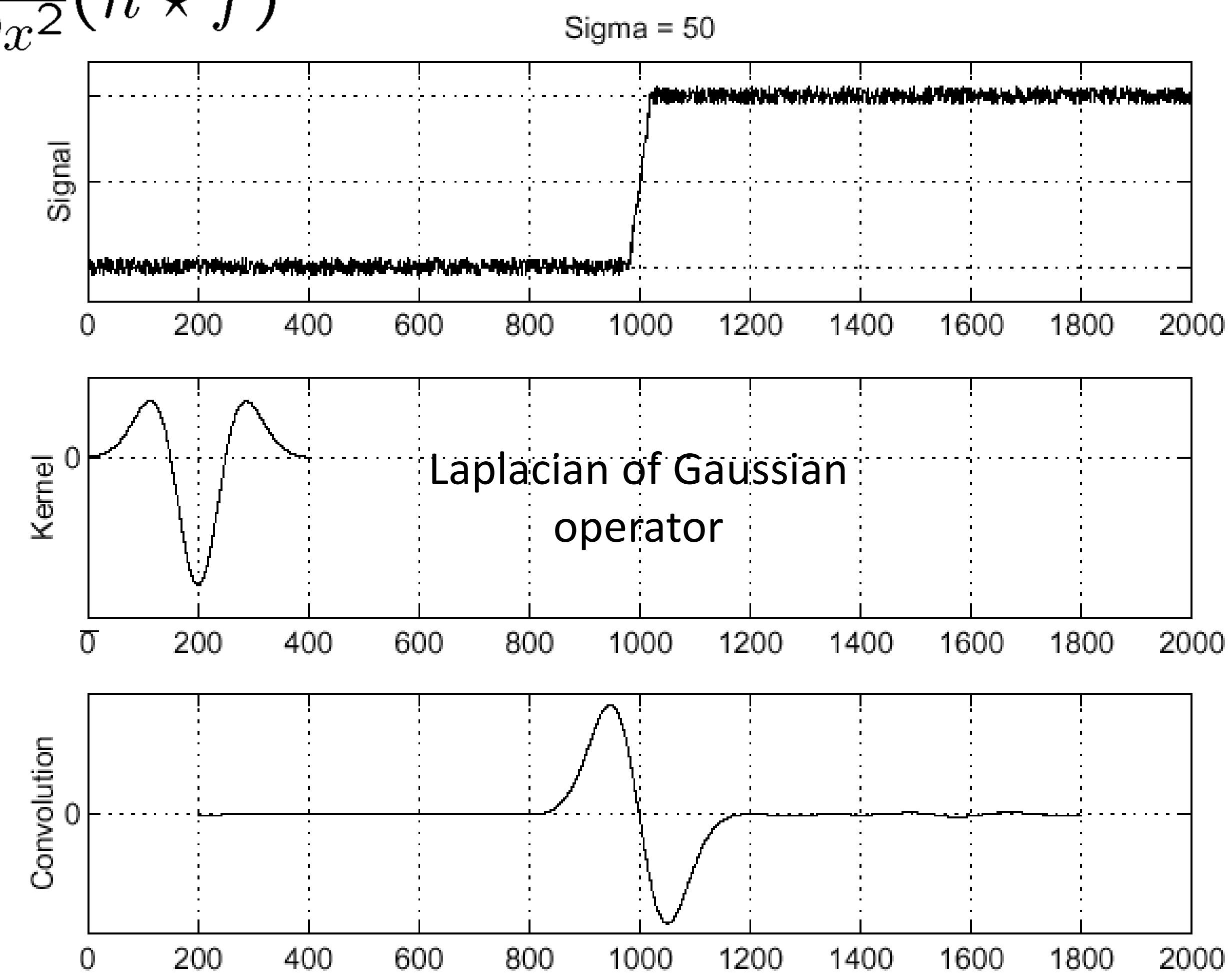
Laplacian of Gaussian

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

f

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

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

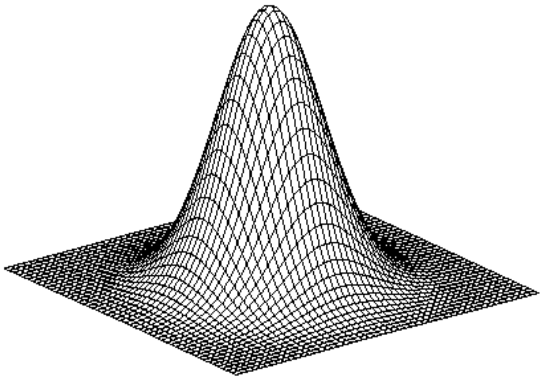


Where is the edge?

Zero-crossings of bottom graph

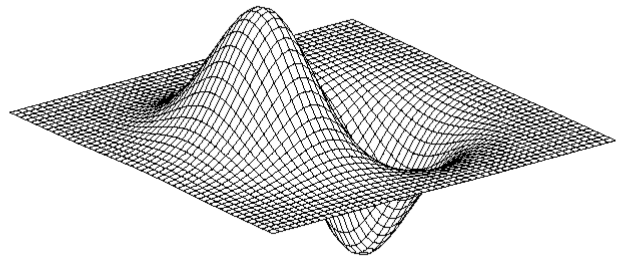
Source: Darrell, Berkeley

2D edge detection filters



Gaussian

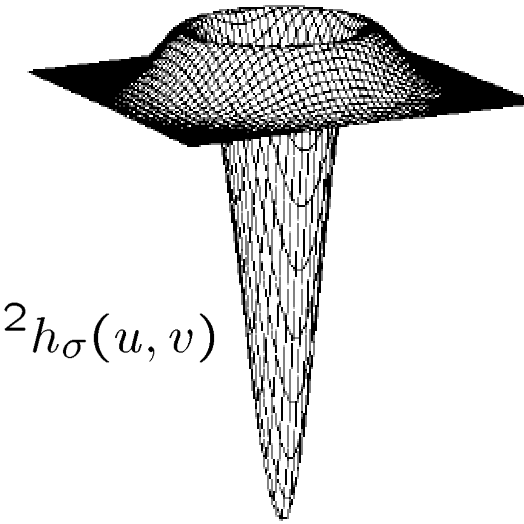
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



derivative of Gaussian

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

Laplacian of Gaussian



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

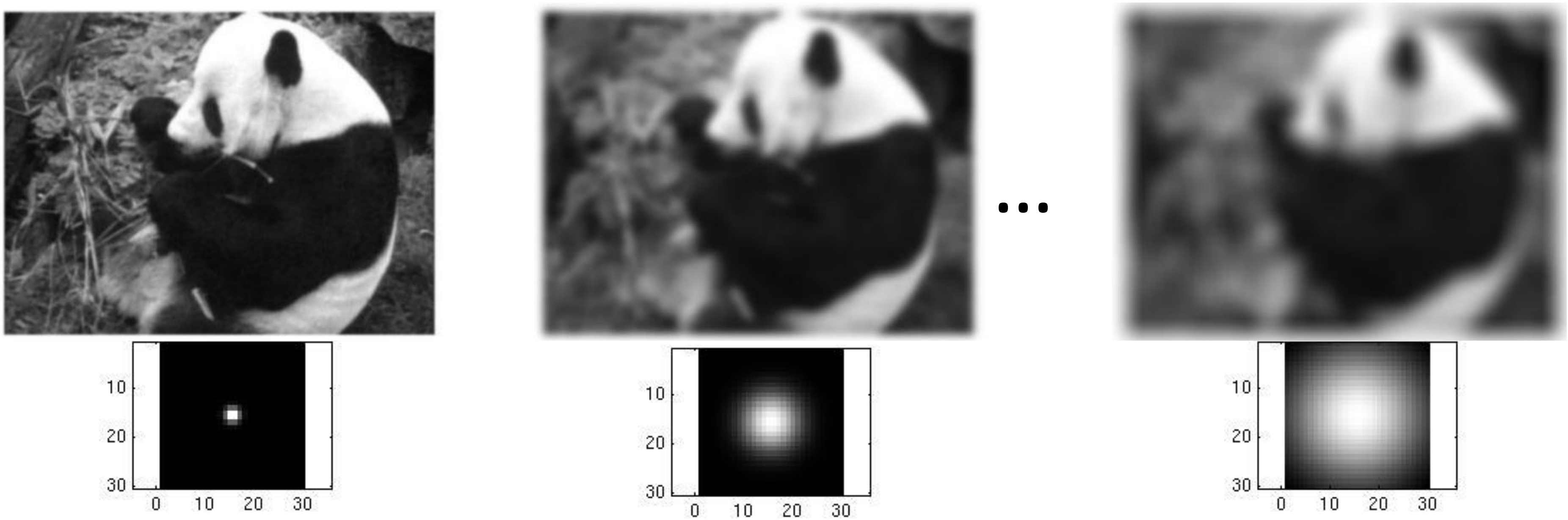
- ∇^2 is the Laplacian operator:

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

Source: Darrell, Berkeley

Smoothing with a Gaussian

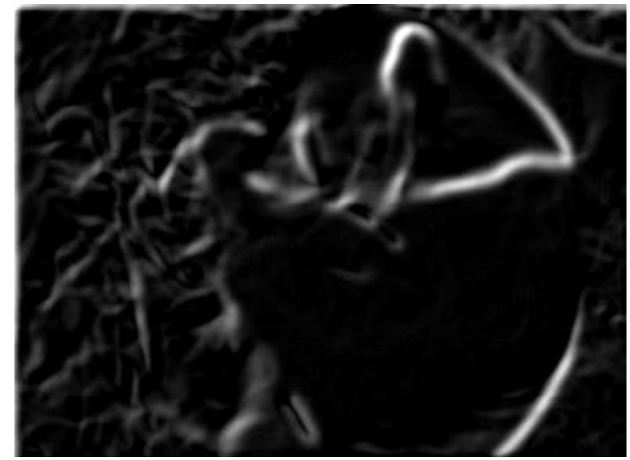
Recall: parameter σ is the “scale” / “width” / “spread” of the Gaussian kernel, and controls the amount of smoothing.



Source: Darrell, Berkeley

Effect of σ on derivatives

σ = 1 pixel



σ = 3 pixels

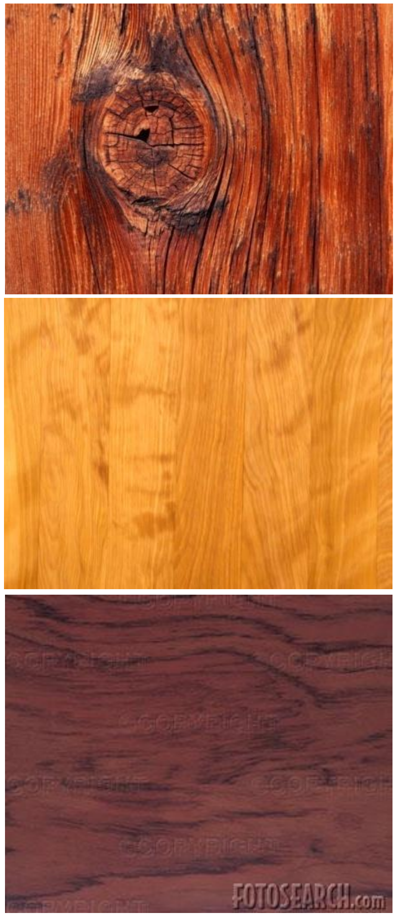
The apparent structures differ depending on Gaussian's scale parameter.

Larger values: larger scale edges detected
Smaller values: finer features detected

Source: Darrell, Berkeley

So, what scale to choose?

It depends what we're looking for.



Too fine of a scale…can't see the forest for the trees.

Too coarse of a scale…can't tell the maple grain from the cherry.

Source: Darrell, Berkeley

Thresholding

- Choose a threshold value t
- Set any pixels less than t to zero (off)
- Set any pixels greater than or equal to t to one (on)

Source: Darrell, Berkeley

Original image

Source: Darrell, Berkeley

Gradient magnitude image

Source: Darrell, Berkeley

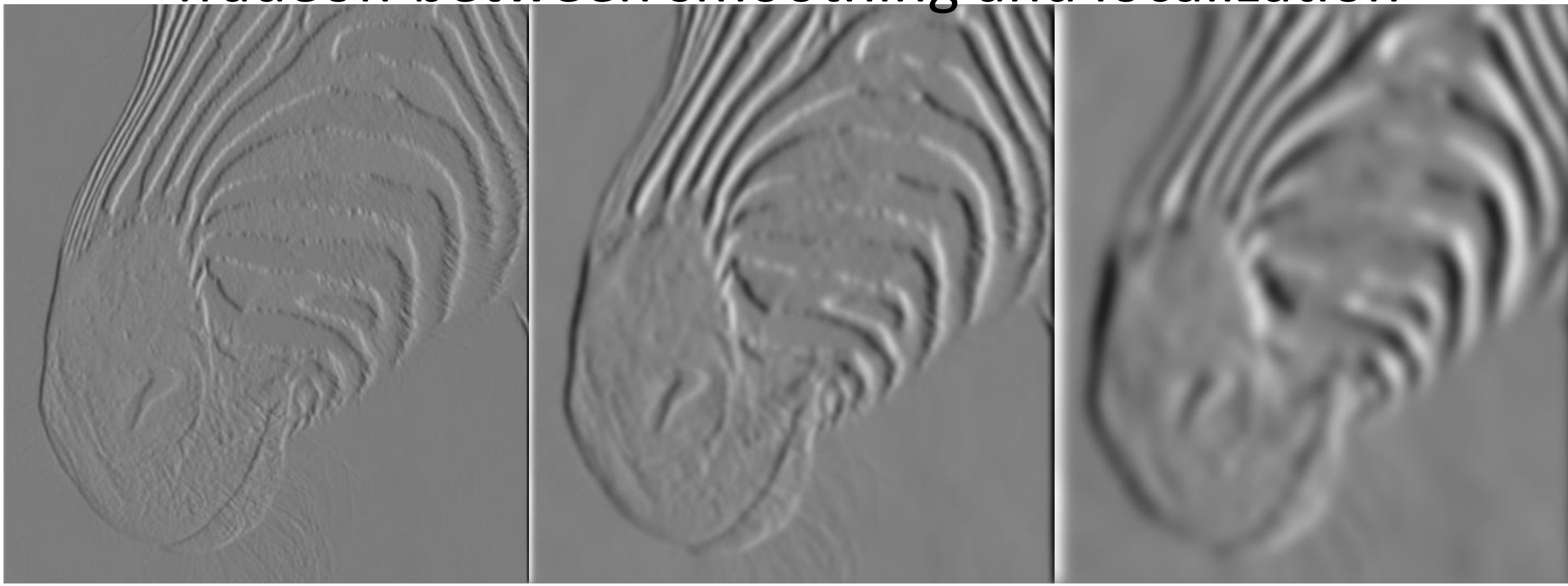
Thresholding gradient with a lower threshold

Source: Darrell, Berkeley

Thresholding gradient with a higher threshold

Source: Darrell, Berkeley

Tradeoff between smoothing and localization



1 pixel 3 pixels 7 pixels

- Smoothed derivative removes noise, but blurs edge. Also finds edges at different “scales”.

Source: D. Forsyth

Source: Hays, Brown

Designing an edge detector

- Criteria for a good edge detector:
  - **Good detection:** the optimal detector should find all real edges, ignoring noise or other artifacts
  - **Good localization**
    - the edges detected must be as close as possible to the true edges
    - the detector must return one point only for each true edge point
- Cues of edge detection
  - Differences in color, intensity, or texture across the boundary
  - Continuity and closure
  - High-level knowledge

Source: Hays, Brown

Source: L. Fei-Fei

Canny edge detector

- This is probably the most widely used edge detector in computer vision
- Theoretical model: step-edges corrupted by additive Gaussian noise
- Canny has shown that the first derivative of the Gaussian closely approximates the operator that optimizes the product of *signal-to-noise ratio* and localization

J. Canny, ***A Computational Approach To Edge Detection***, IEEE Trans. Pattern Analysis and Machine Intelligence, 8:679-714, 1986.

Source: Hays, Brown

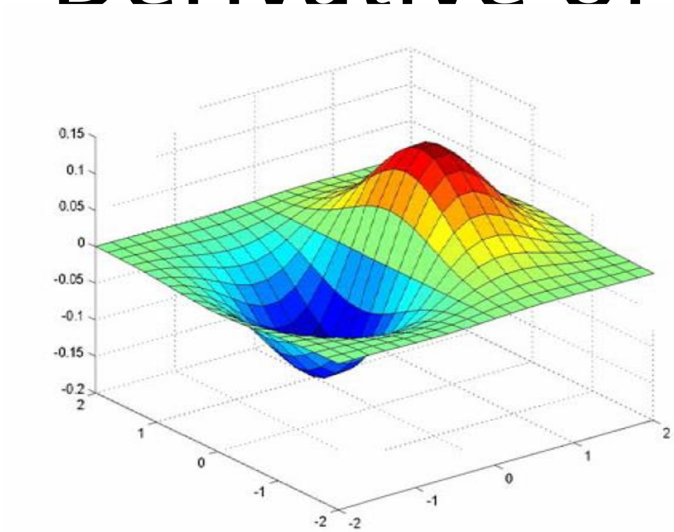
Source: L. Fei-Fei

Example

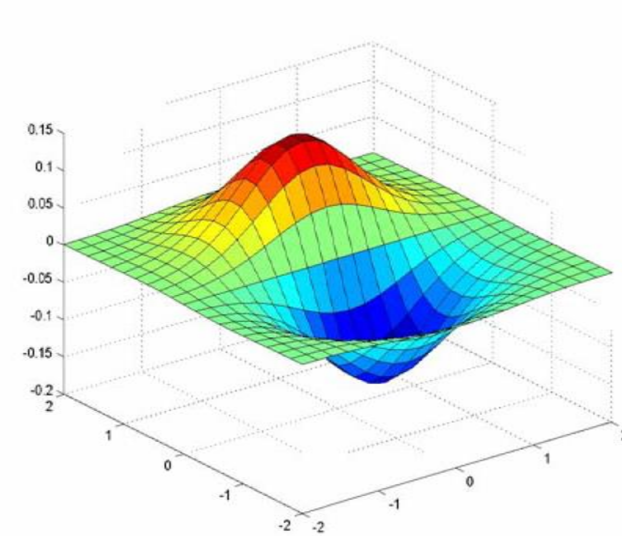
original image (Lena)

Source: Hays, Brown

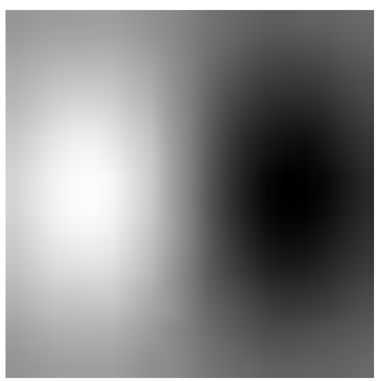
Derivative of Gaussian filter

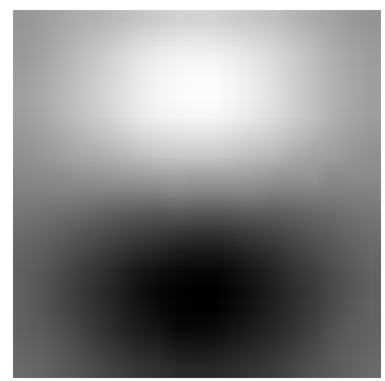


x-direction



y-direction





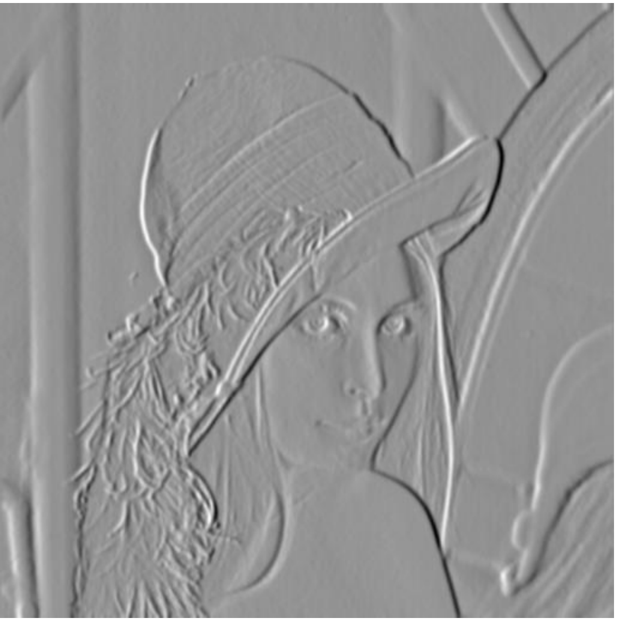
Source: Hays, Brown

The Canny edge detector

original image (Lena)

Source: Darrell, Berkeley

Compute Gradients (DoG)



X-Derivative of Gaussian

Y-Derivative of Gaussian

Gradient Magnitude

Source: Hays, Brown

The Canny edge detector

norm of the gradient

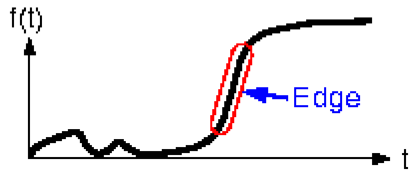
Source: Darrell, Berkeley

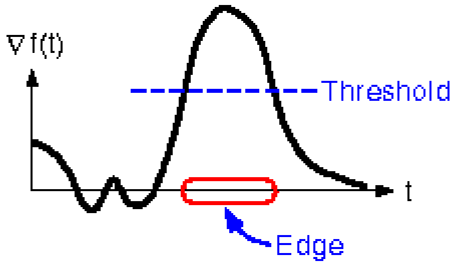
The Canny edge detector

thresholding

Source: Darrell, Berkeley

The Canny edge detector

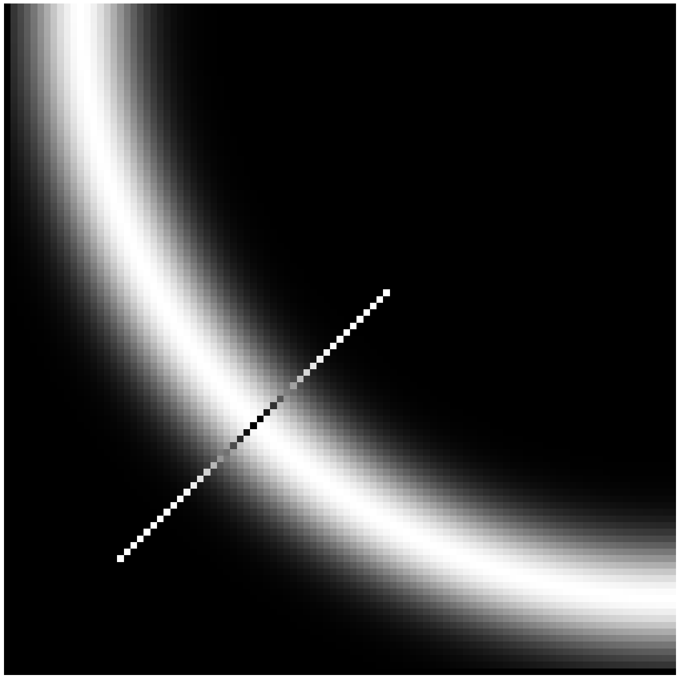


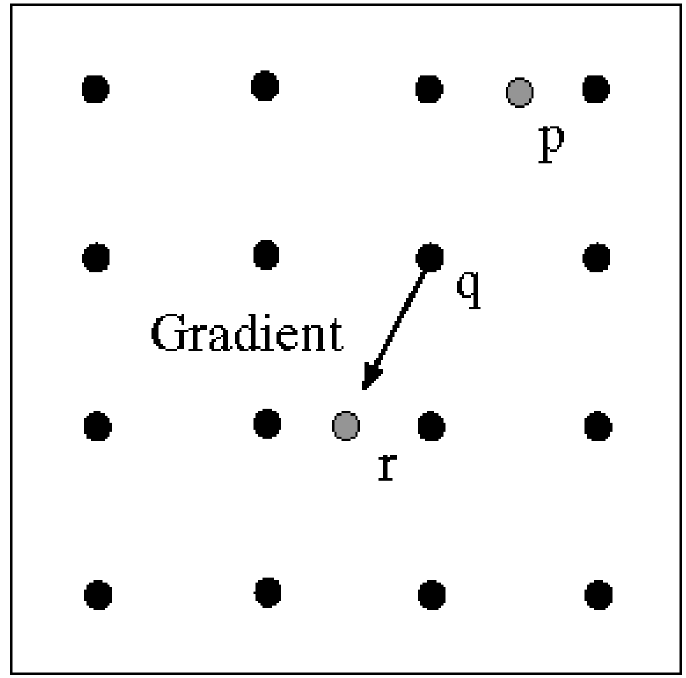


How to turn these thick regions of the gradient into curves?

Source: Darrell, Berkeley

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

Source: Darrell, Berkeley

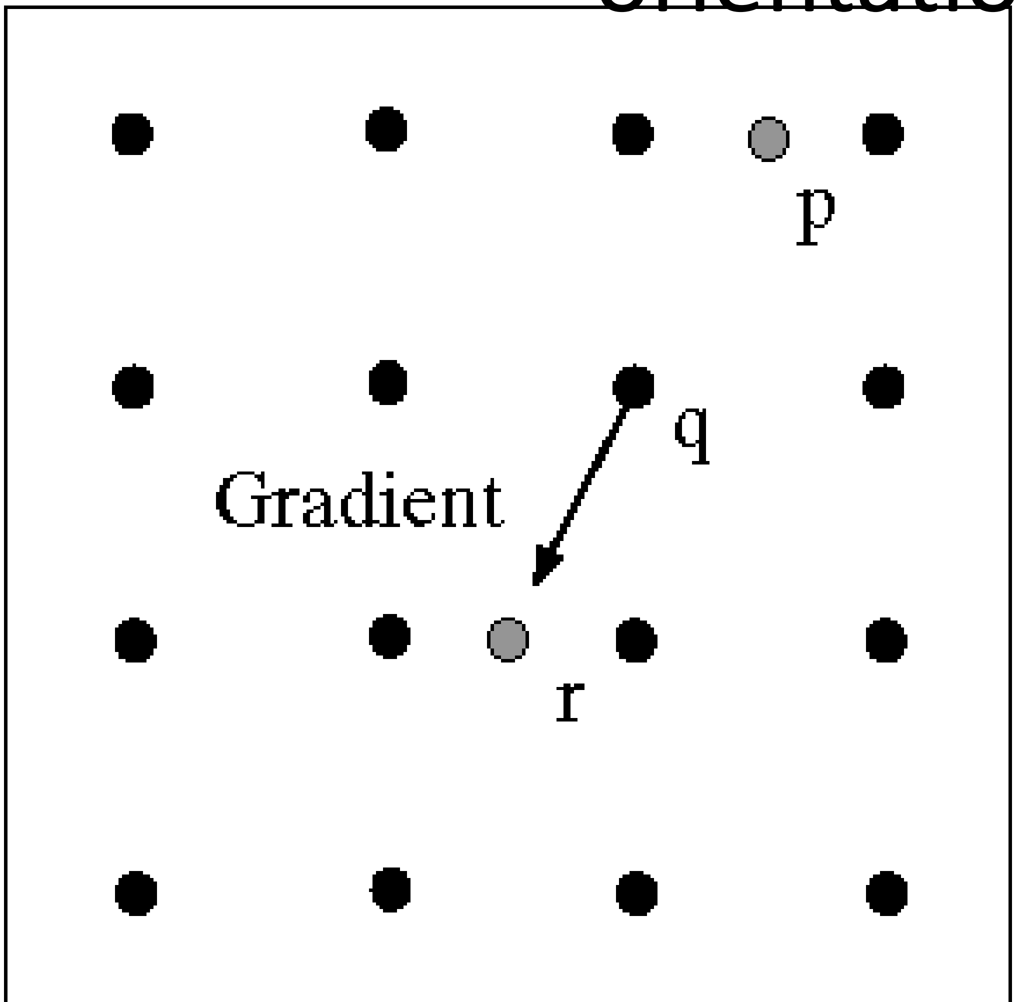
Get Orientation at Each Pixel

- Threshold at minimum level

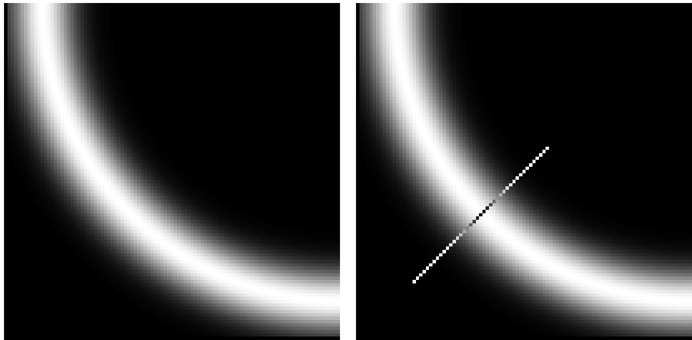
theta = atan2(gy, gx)

Source: Hays,

Non-maximum suppression for each orientation



At q, we have a maximum if the value is larger than those at both p and at r. Interpolate to get these values.



Source: D. Forsyth

Source: Hays, Brown

Before Non-max Suppression

Source: Hays, Brown

After non-max suppression

Source: Hays, Brown

The Canny edge detector

Problem: pixels along this edge didn't survive the thresholding

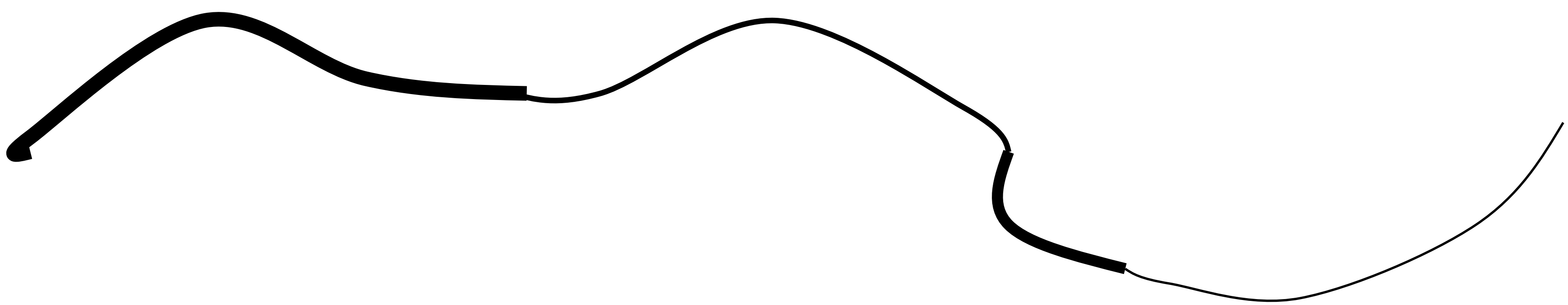
thinning

(non-maximum suppression)

Source: Darrell, Berkeley

Hysteresis thresholding

- Check that maximum value of gradient value is sufficiently large
  - drop-outs? use **hysteresis**
    - use a high threshold to start edge curves and a low threshold to continue them.



Source: Darrell, Berkeley

Source: S. Seitz

Hysteresis thresholding

- Threshold at low/high levels to get weak/strong edge pixels
- Do connected components, starting from strong edge pixels

Source: Hays, Brown

Hysteresis thresholding

original image

high threshold
(strong edges)

low threshold
(weak edges)

hysteresis threshold

Source: Darrell, Berkeley

Source: L. Fei-Fei

Final Canny Edges

Source: Hays, Brown

Canny edge detector

1. Filter image with x, y derivatives of Gaussian
2. Find magnitude and orientation of gradient
3. Non-maximum suppression:
   – Thin multi-pixel wide “ridges” down to single pixel width
4. Thresholding and linking (hysteresis):
   – Define two thresholds: low and high
   – Use the high threshold to start edge curves and the low threshold to continue them

• MATLAB: edge(image, ‘canny’)

Source: Hays, Brown

Source: D. Lowe, L. Fei-Fei

Effect of σ (Gaussian kernel spread/size)

original

Canny with $\sigma = 1$

Canny with $\sigma = 2$

The choice of σ depends on desired behavior

- large σ detects large scale edges
- small σ detects fine features

Source: Hays, Brown

Source: S. Seitz

Object boundaries vs. edges

Background

Texture

Shadows

Source: Darrell, Berkeley

Edge detection is just the beginning…

image | human segmentation | gradient magnitude

Berkeley segmentation database:
http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

Much more on segmentation later in term…

Source: Darrell, Berkeley

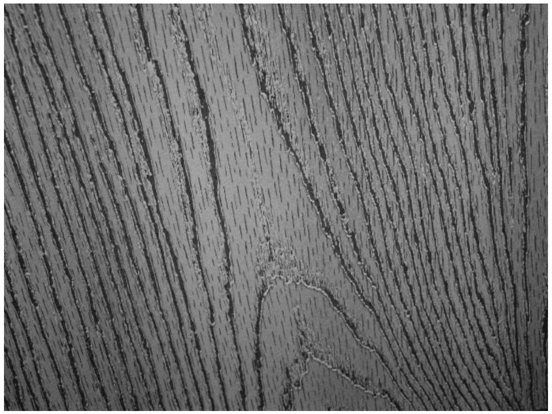
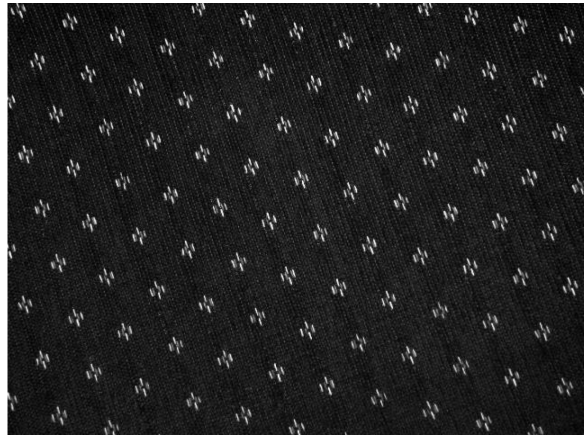
Source: L. Lazebnik

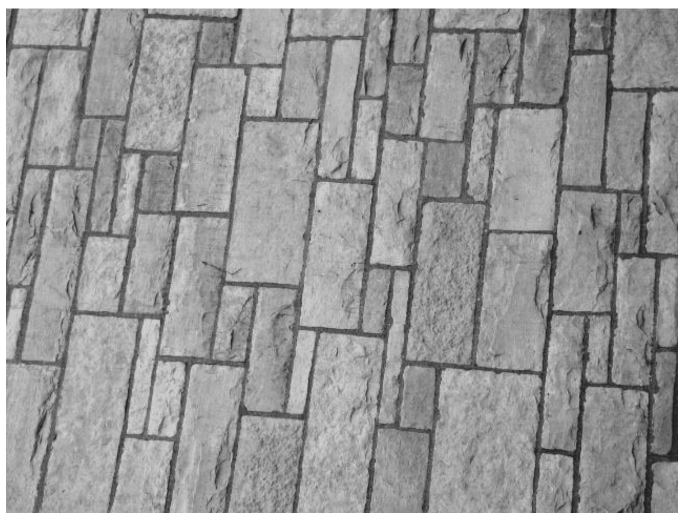
Representing Texture

Source: Hays, Brown

Source: Forsyth

Texture and Material

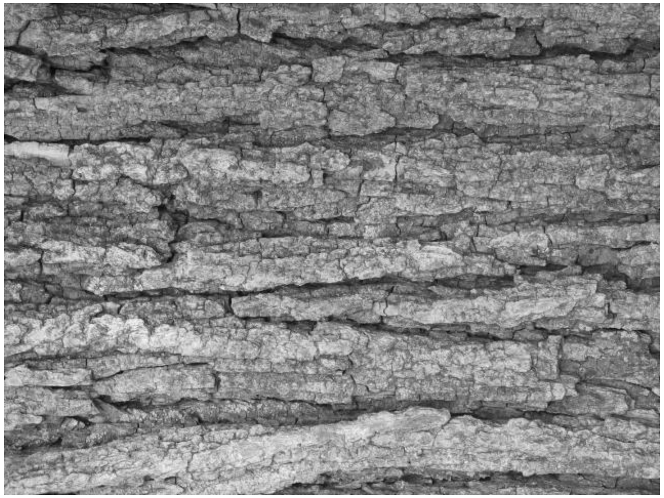





Source: Hays, Brown

http://www-cvr.ai.uiuc.edu/ponce_grp/data/texture_database/samples/

Texture and Orientation



Source: Hays, Brown

http://www-cvr.ai.uiuc.edu/ponce_grp/data/texture_database/samples/

Texture and Scale

Source: Hays, Brown

http://www-cvr.ai.uiuc.edu/ponce_grp/data/texture_database/samples/

What is texture?

Regular or stochastic patterns caused by bumps, grooves, and/or markings

Source: Hays, Brown

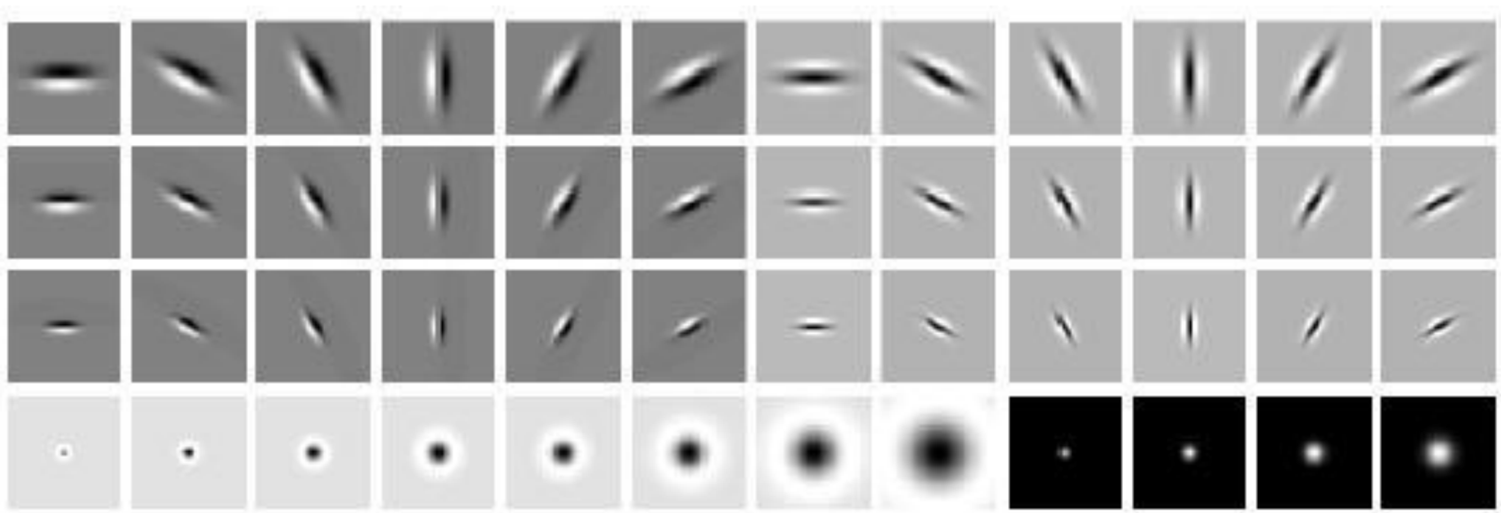
How can we represent texture?

- Compute responses of blobs and edges at various orientations and scales

Source: Hays, Brown

Overcomplete representation: filter banks

LM Filter Bank

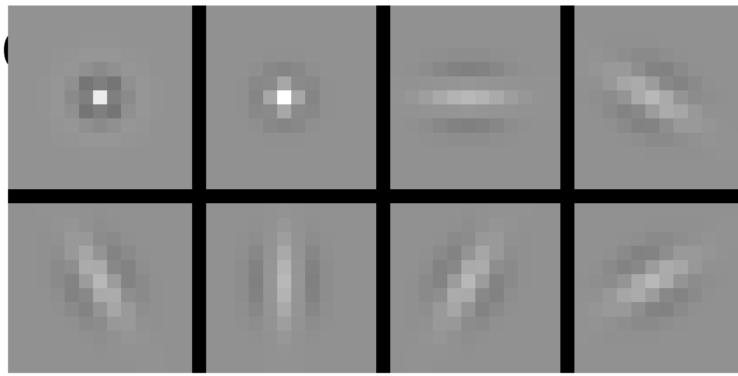


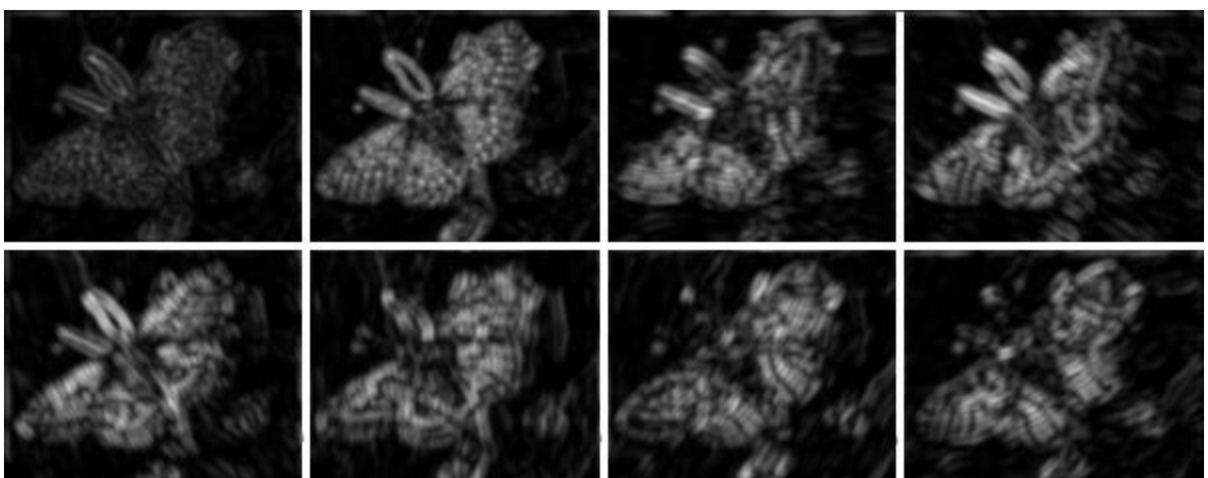
Code for filter banks: *www.robots.ox.ac.uk/~vgg/research/texclass/filters.html*

Source: Hays, Brown

Filter banks

- Process image with each filter and keep r /abs r





Source: Hays, Brown

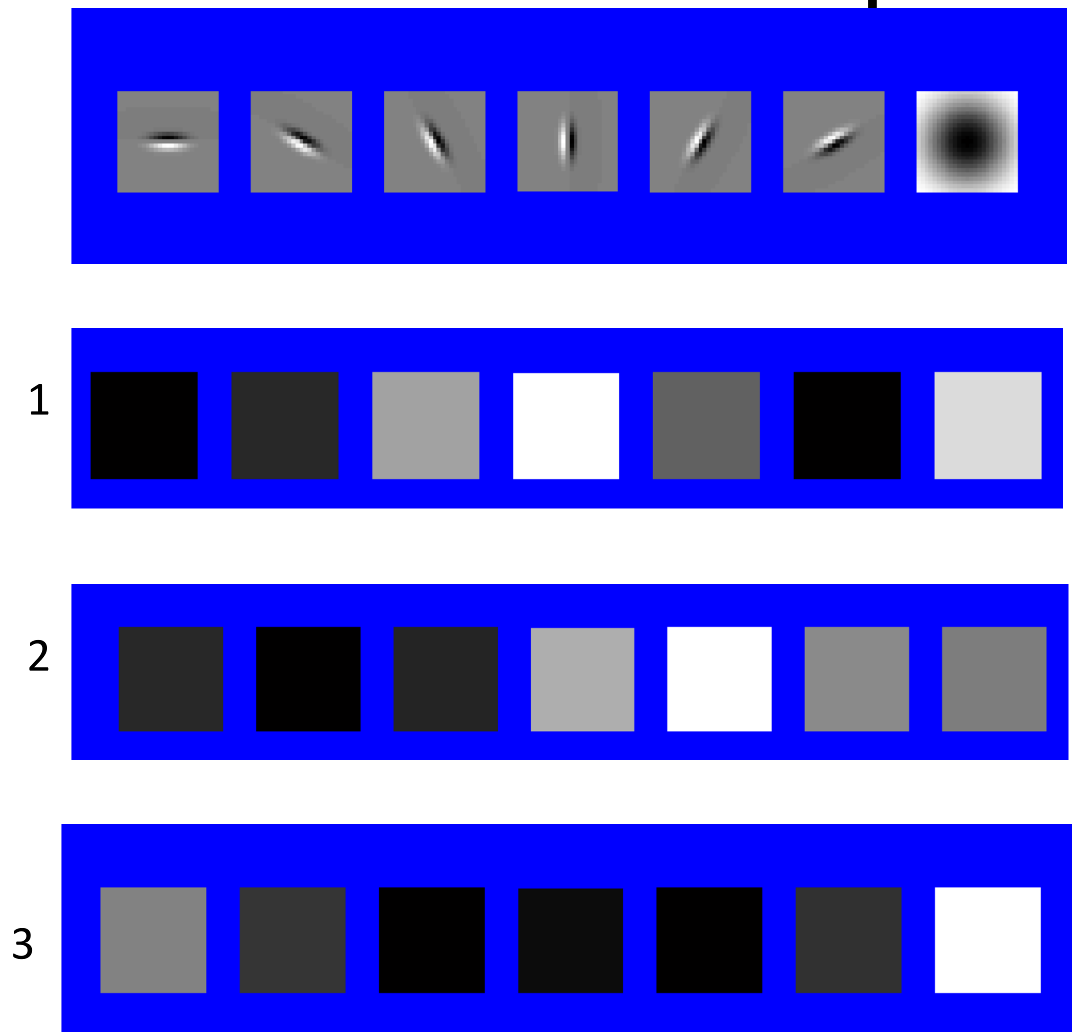
How can we represent texture?

- Measure responses of blobs and edges at various orientations and scales
- Idea 1: Record simple statistics (e.g., mean, std.) of absolute filter responses

Source: Hays, Brown

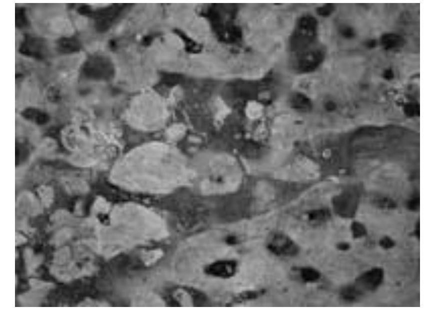
Can you match the texture to the response?

Filters



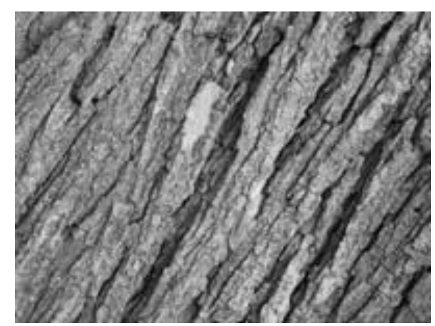
Mean abs responses

A



B

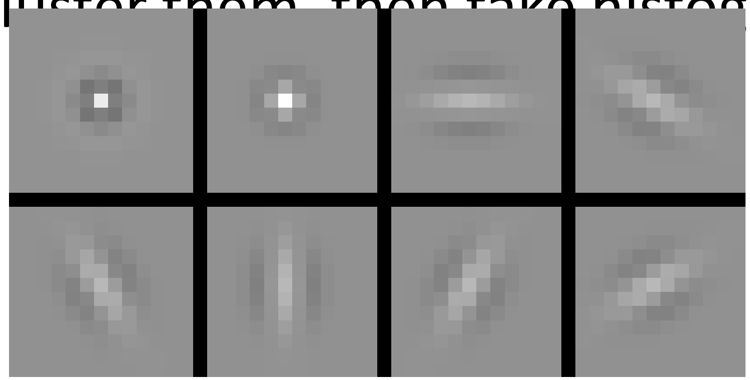
C

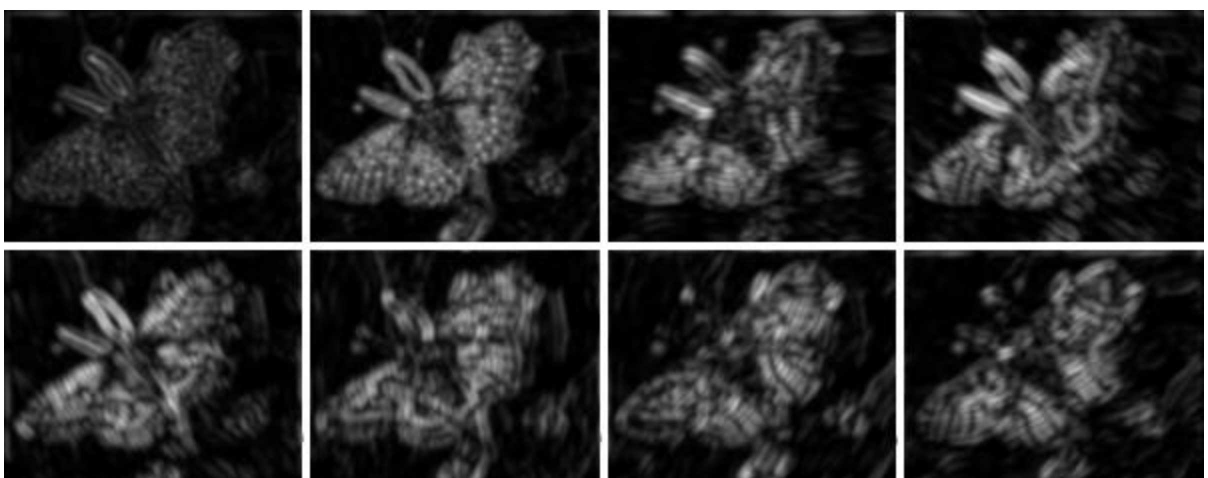


Source: Hays, Brown

Representing texture

- Idea 2: take vectors of filter responses at each pixel and cluster them, then take histograms.





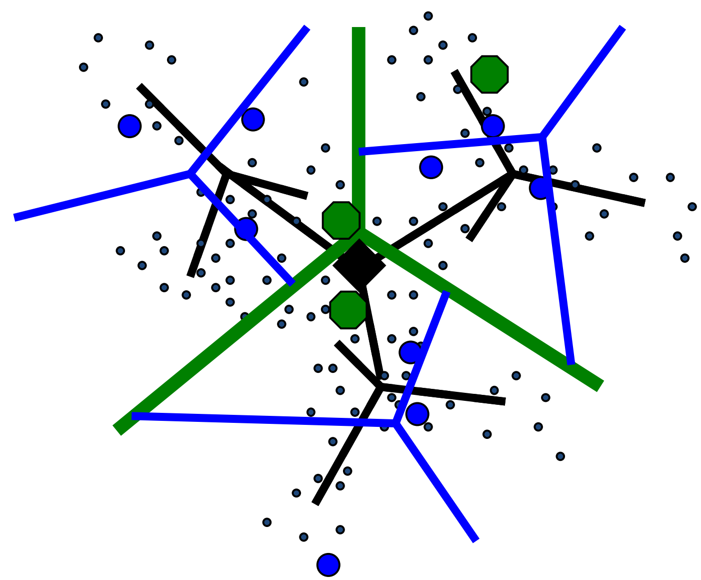
Source: Hays, Brown

Building Visual Dictionaries

1. Sample patches from a database
   – E.g., 128 dimensional SIFT vectors

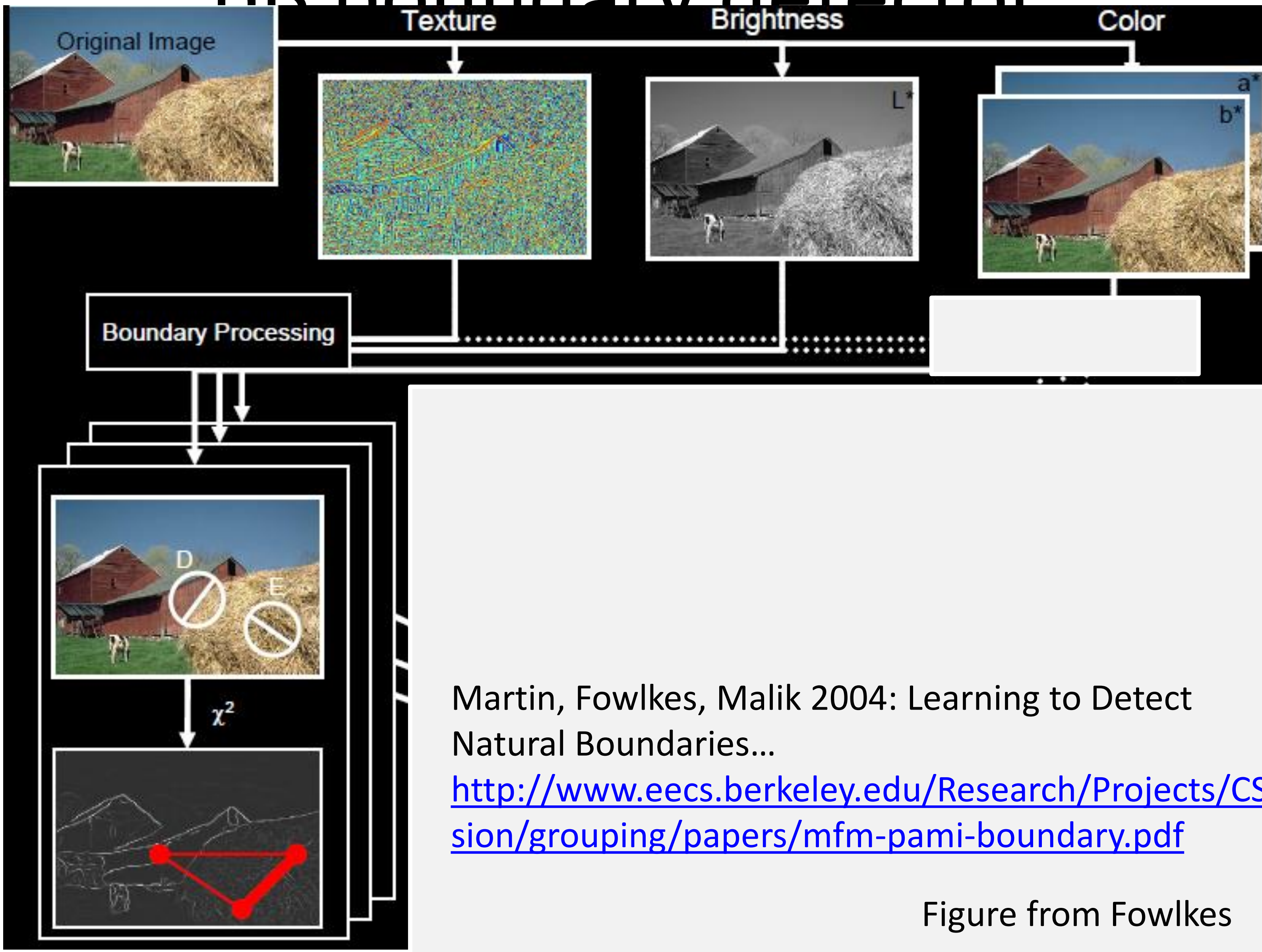
2. Cluster the patches
   – Cluster centers are the dictionary

3. Assign a codeword (number) to each new patch, according to the nearest cluster



Source: Hays, Brown

pB boundary detector

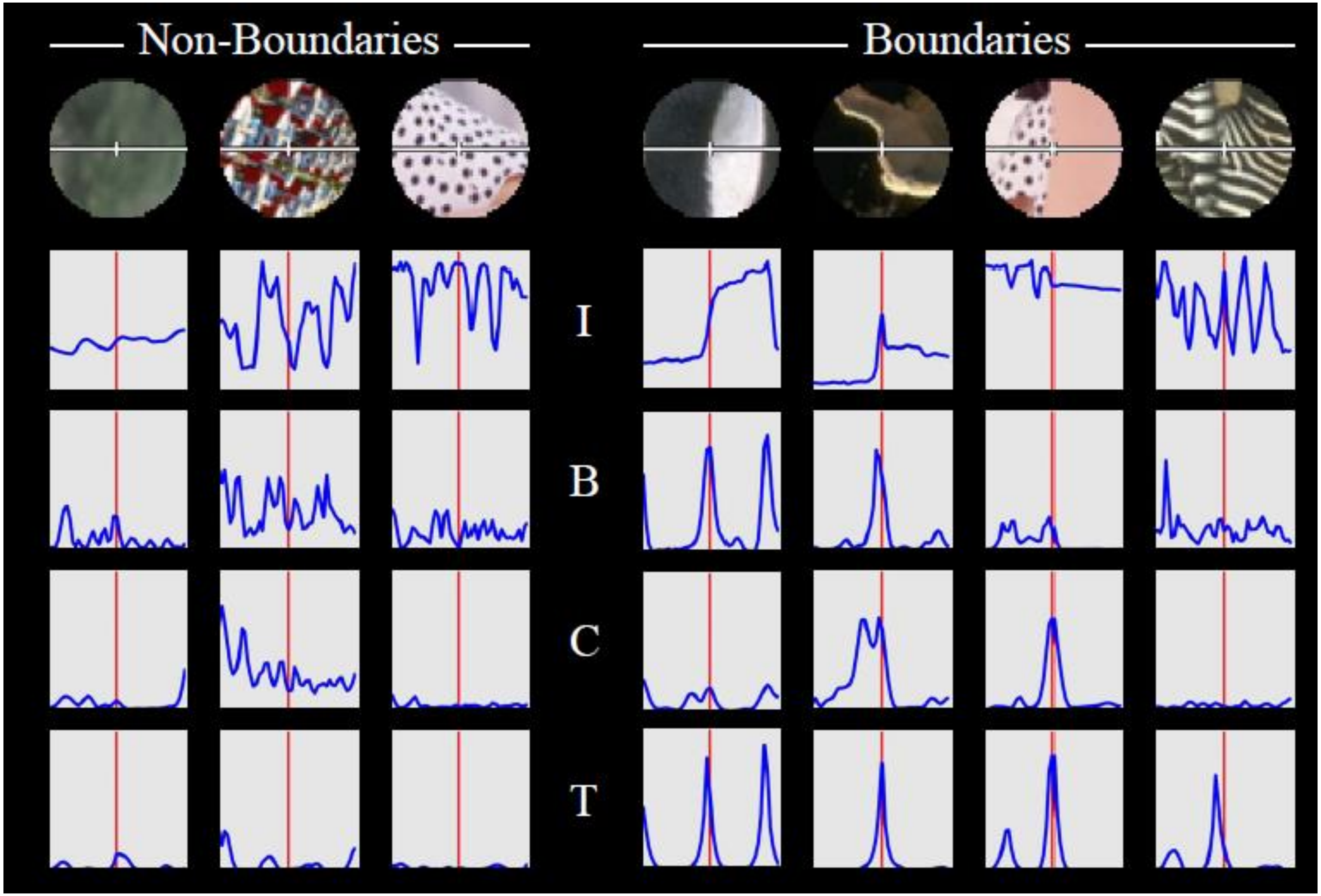


Martin, Fowlkes, Malik 2004: Learning to Detect Natural Boundaries…
http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/papers/mfm-pami-boundary.pdf

Figure from Fowlkes

Source: Hays, Brown

pB Boundary Detector

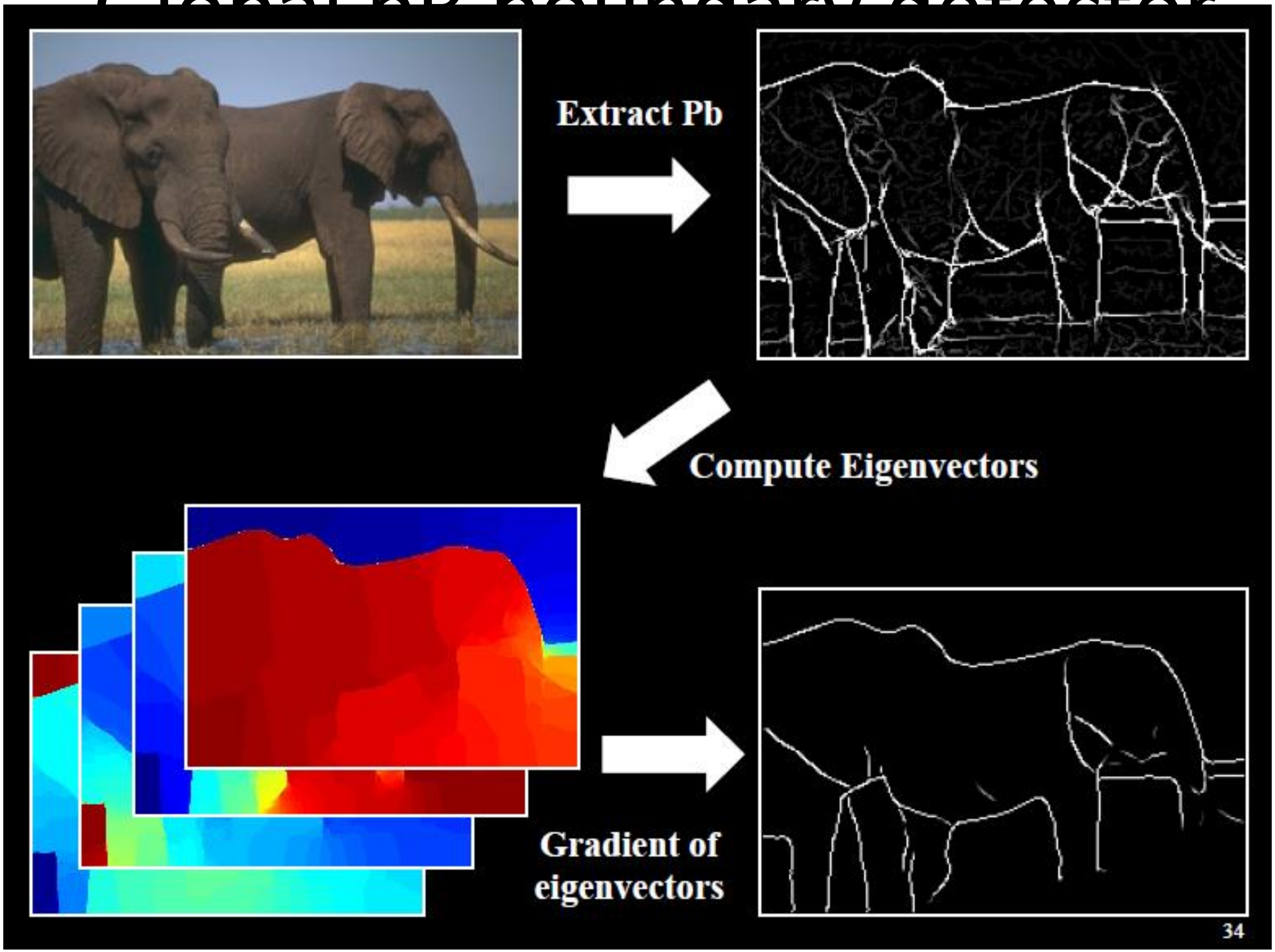


Source: Hays, Brown

Figure from Fowlkes

Source: Hays, Brown

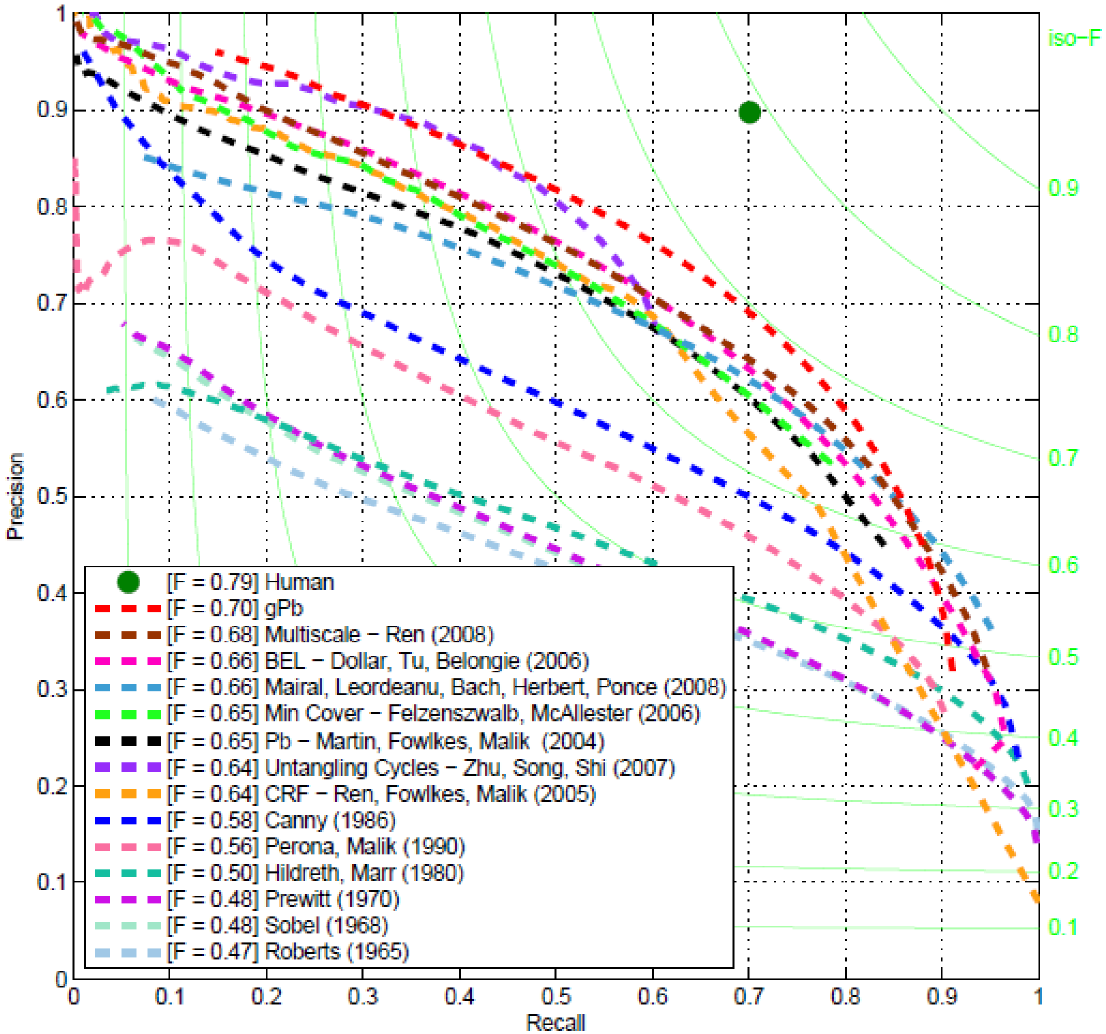
Global pB boundary detector



Source: Hays, Brown

Figure from Fowlkes

45 years of boundary detection



Source: Arbelaez, Maire, Fowlkes, and Malik. TPAMI 2011 (pdf)

Source: Hays, Brown