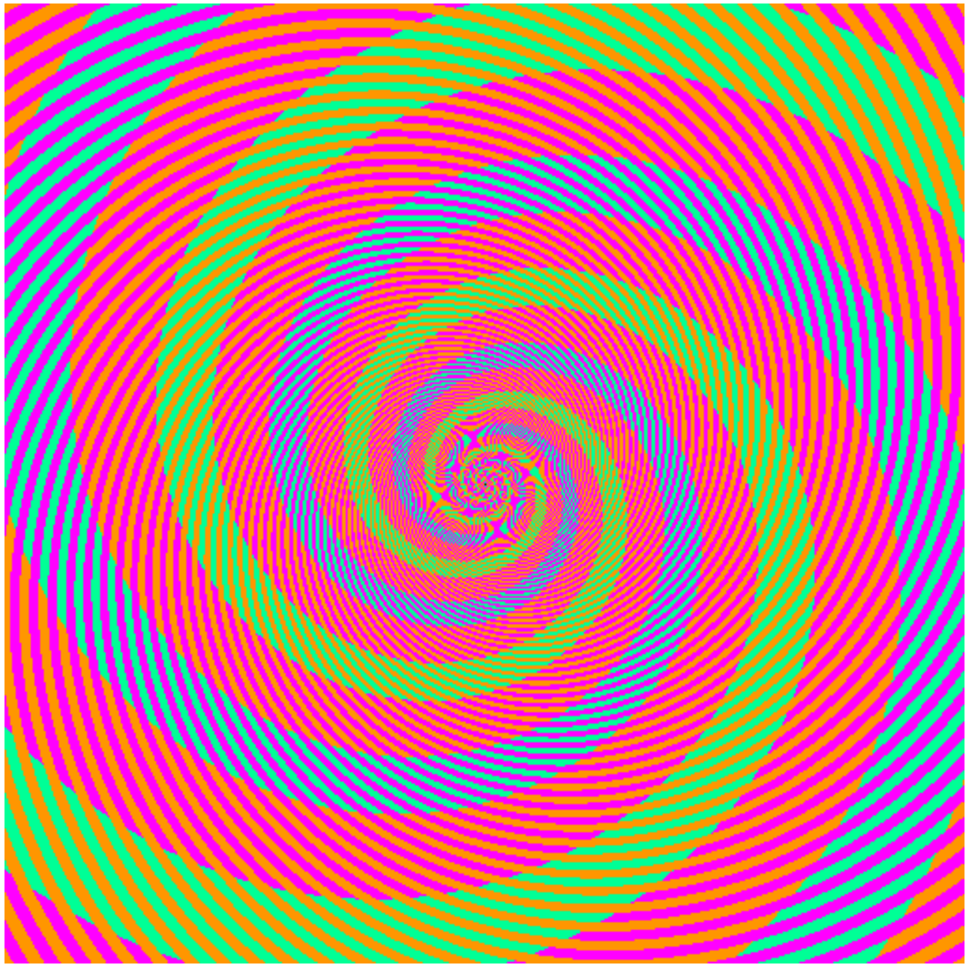
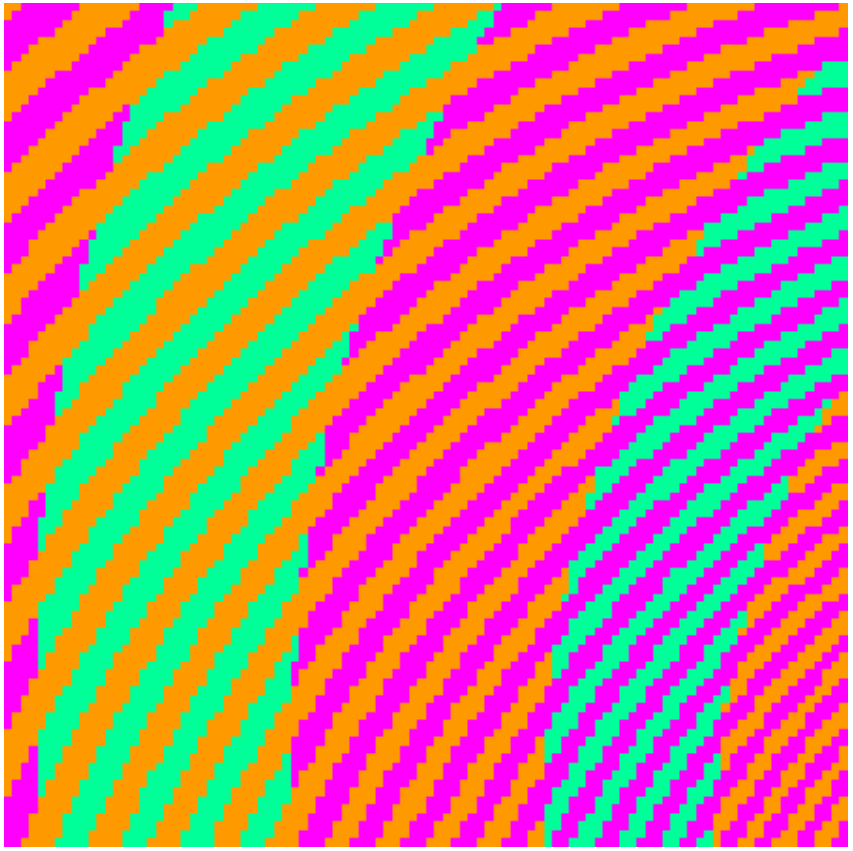


The blue and green colors are actually the same



Read about it here:
http://blogs.discovermagazine.com/badastronomy/2009/06/24/the-blue-and-the-green/

CSC589 Introduction to Computer Vision Lecture 12

Edges and Boundary Detection

Many slides from Lana Lazebnik, Steve Seitz, David Forsyth, David Lowe, Fei-Fei Li, and Derek Hoiem

Previous Lectures

- We’ve now touched on the first two chapters of Szeliski, roughly.
  - 1. Introduction
  - 3. Image Processing
- Now we’re moving on to
  - 4. Feature Detection and Matching
  - Camera and Image formation Chapter 2
  - Multiple views and motion (7, 8, 11)

Project 3 and mid-term exam

- Implement Pyramid image blending
- Will be out on Friday, due in a week.
- Mid-term exam is on **Monday, March 17th**.
- Time to review what we have learned so far.

Edge detection

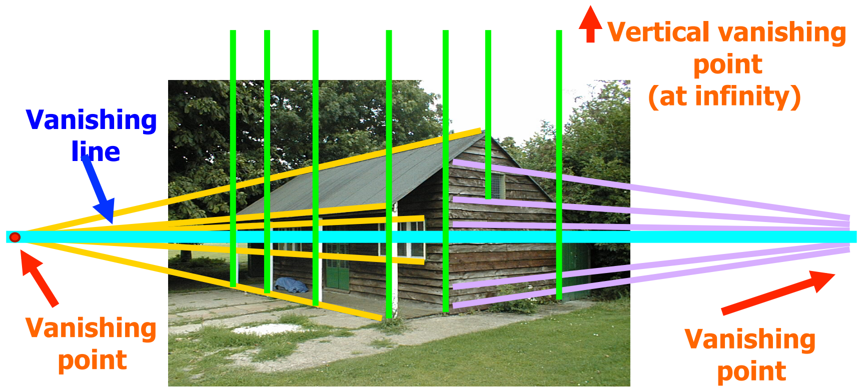
- **Goal:** Identify sudden changes (discontinuities) in an image
  - Intuitively, most semantic and shape information from the image can be encoded in the edges
  - More compact than pixels
- **Ideal:** artist's line drawing (but artist is also using object-level knowledge)

Source: D. Lowe

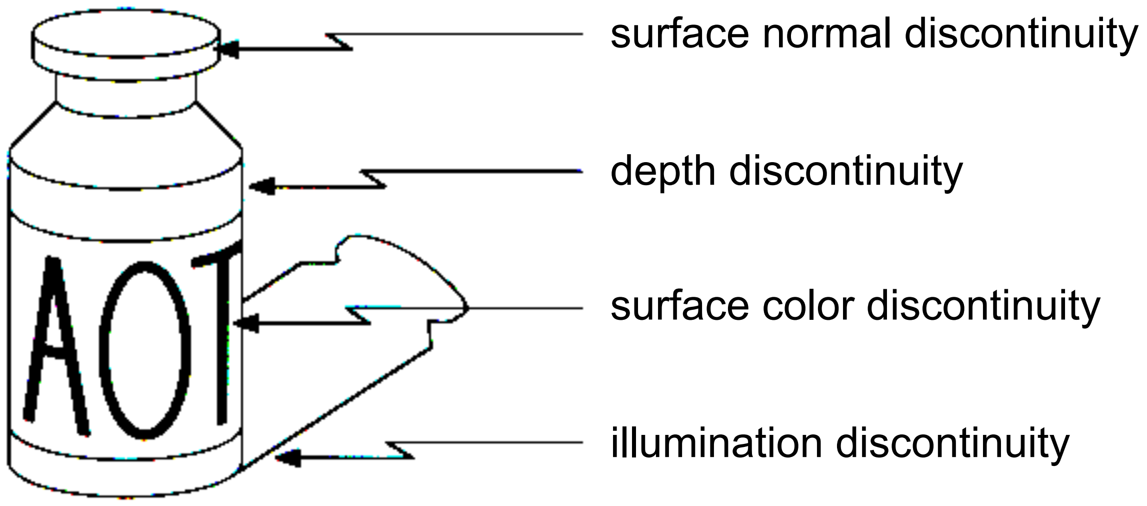
Why do we care about edges?

- Extract information, recognize objects

- Recover geometry and viewpoint



Origin of Edges


surface normal discontinuity
depth discontinuity
surface color discontinuity
illumination discontinuity
AOT

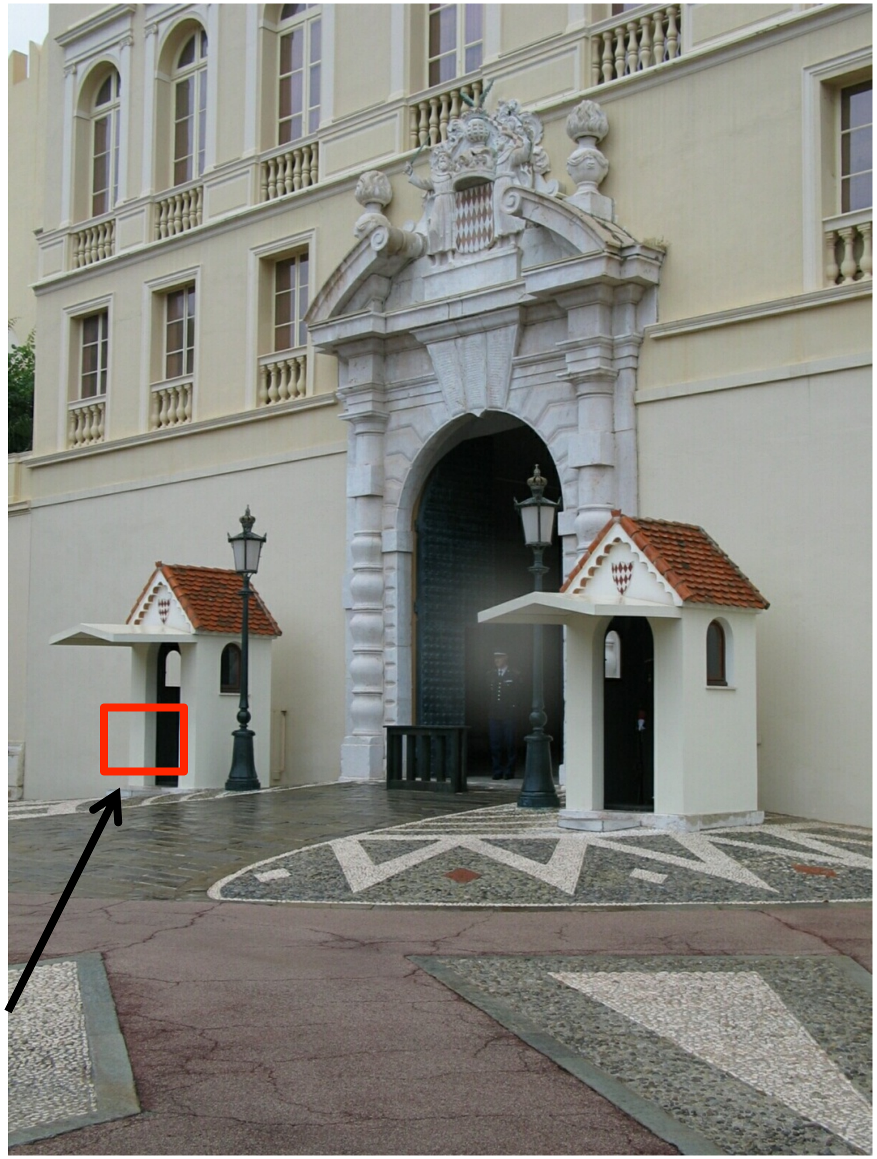
- Edges are caused by a variety of factors

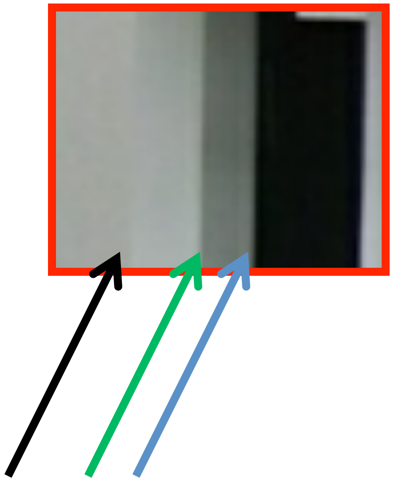
Source: Steve Seitz

Closeup of edges

Source: D. Hoiem

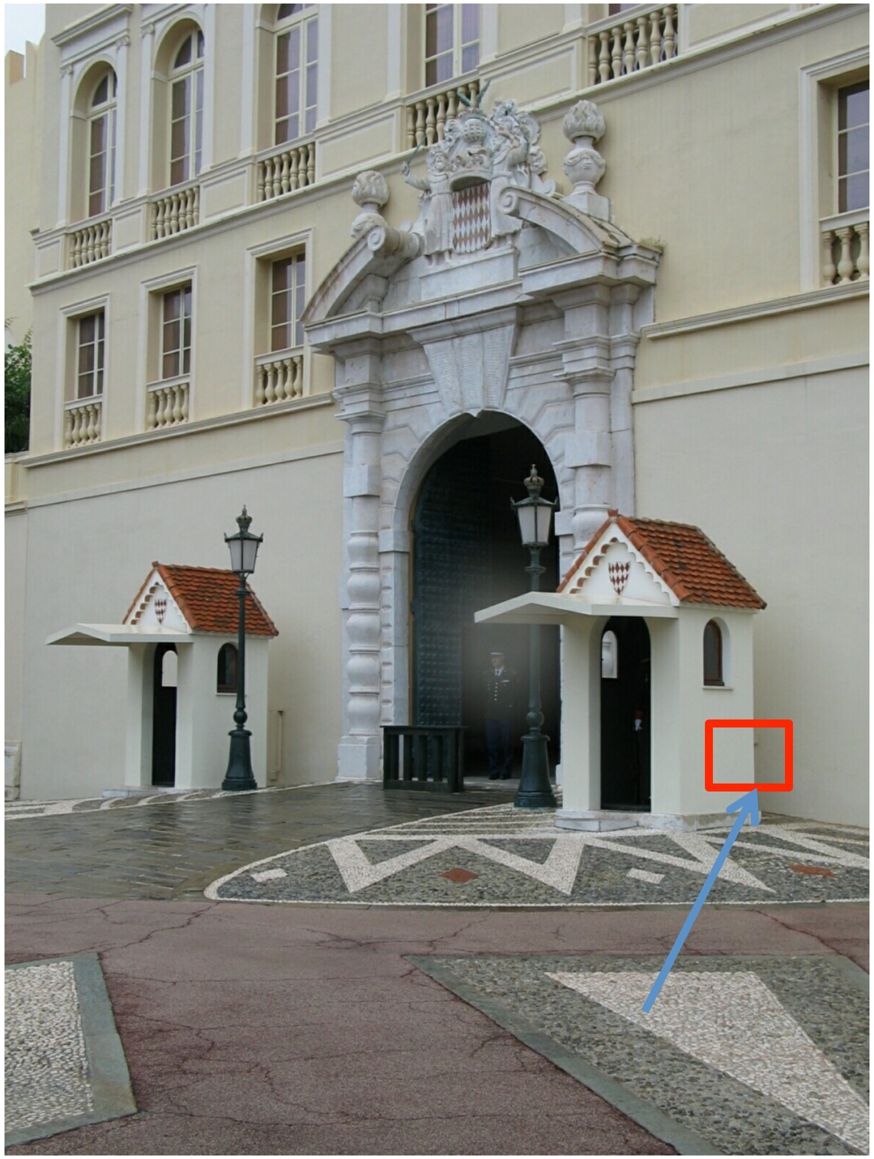
Closeup of edges

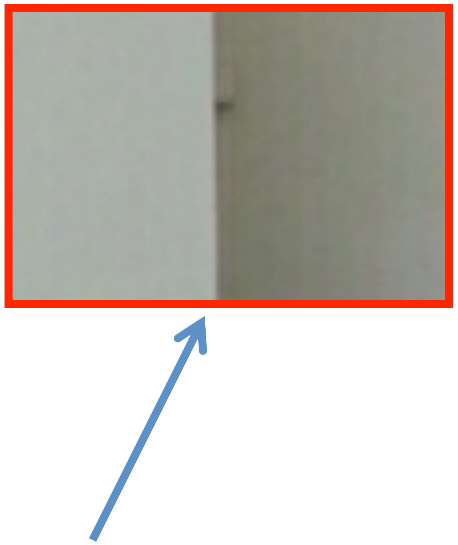




Source: D. Hoiem

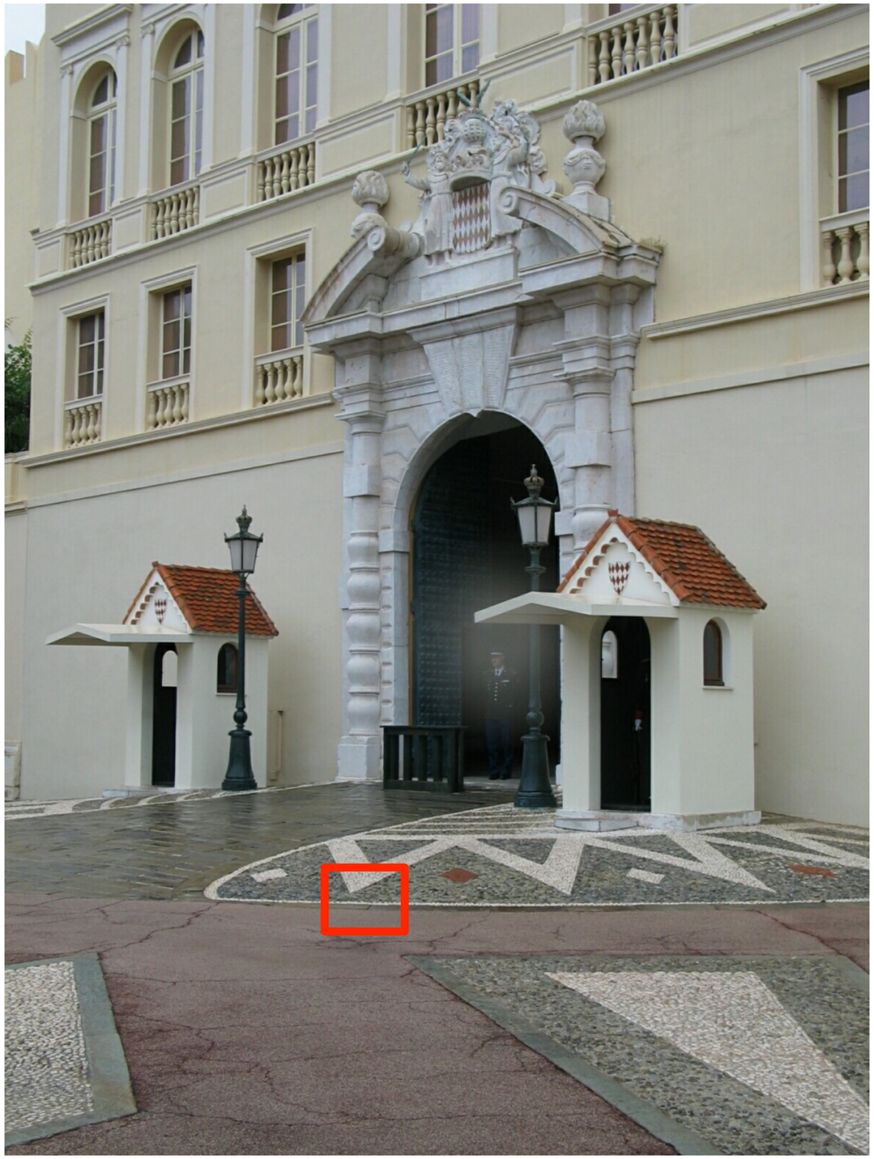
Closeup of edges

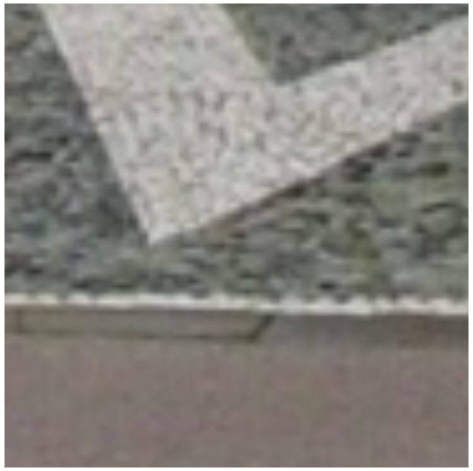




Source: D. Hoiem

Closeup of edges

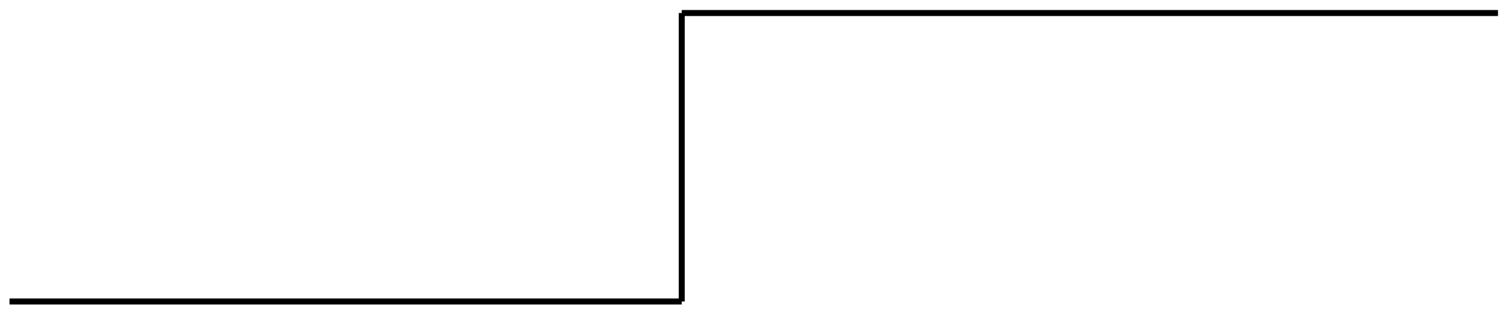




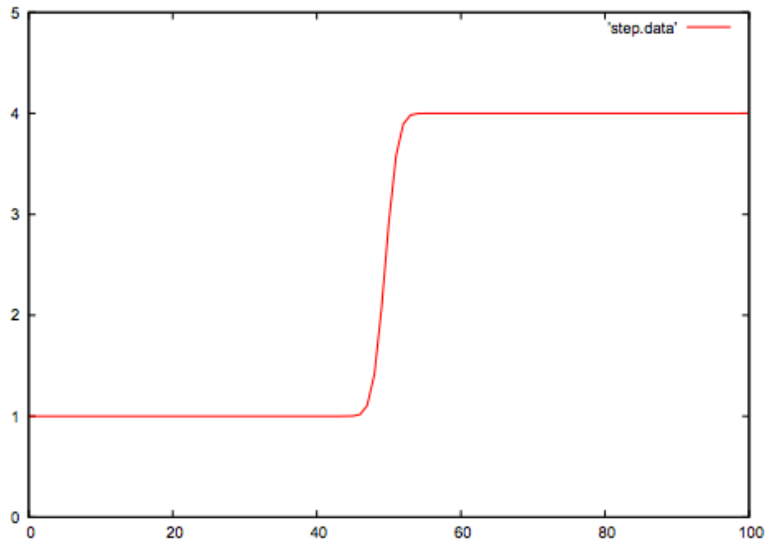
Source: D. Hoiem

Edge is Where Change Occurs

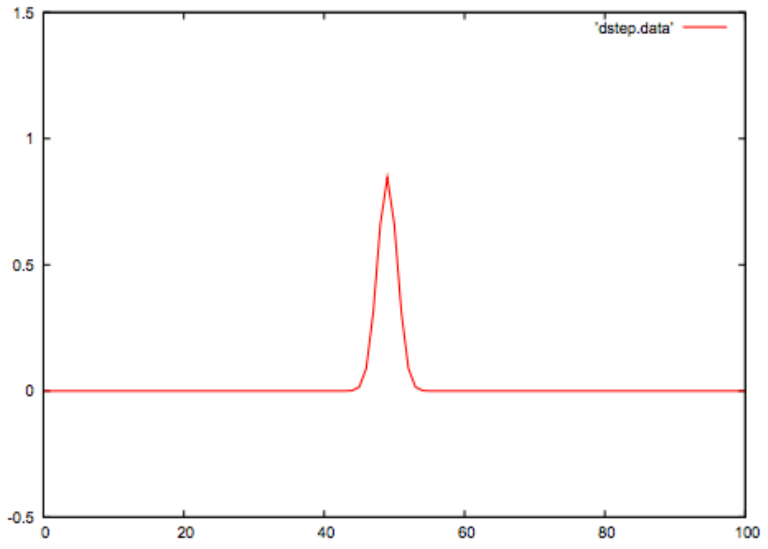
- Change is measured by derivative in 1D
- Biggest change, derivative has maximum magnitude
- Or 2nd derivative is zero.



Edge is Where Change Occurs



(a)



(b)

Characterizing edges

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

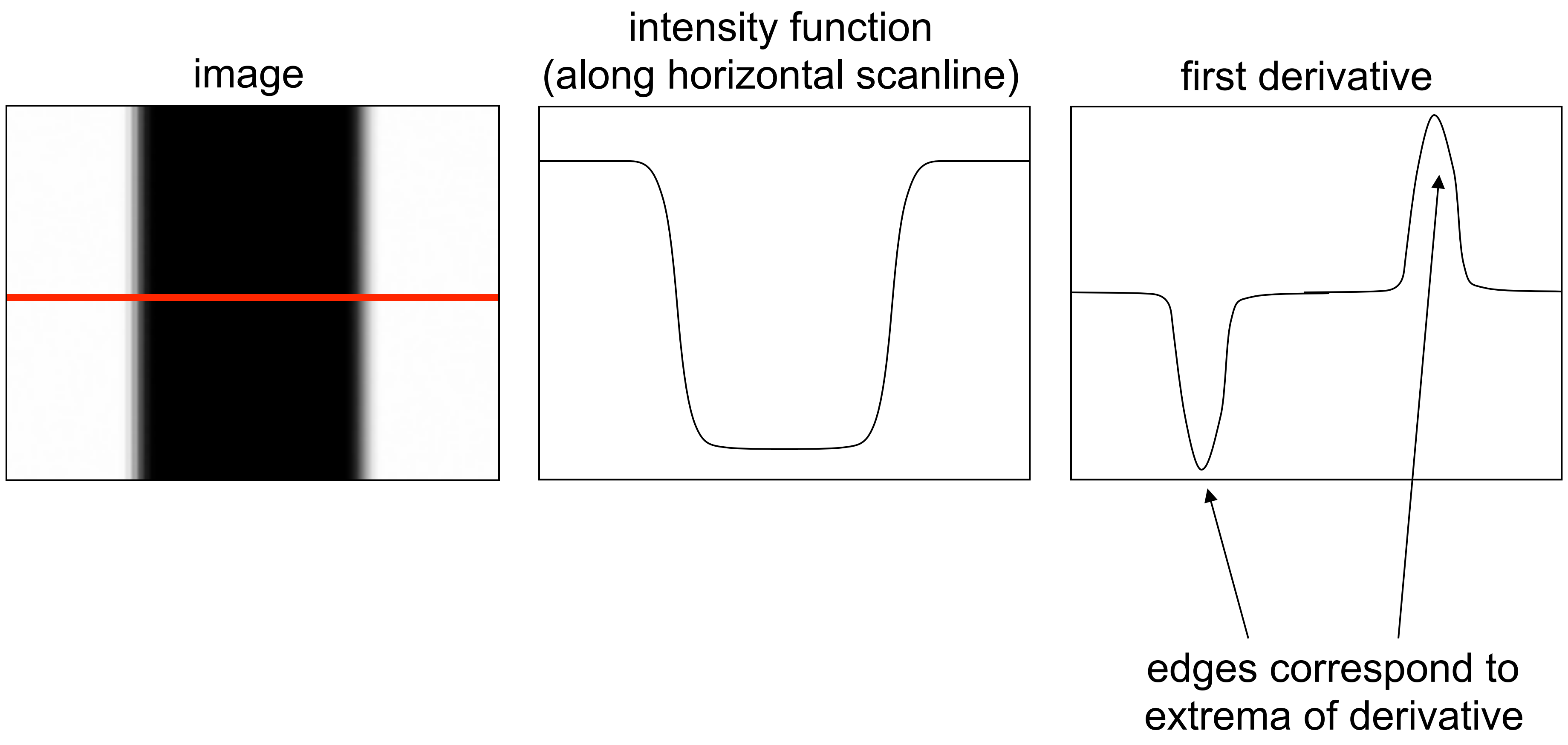
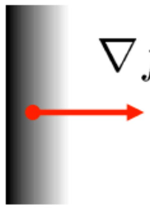


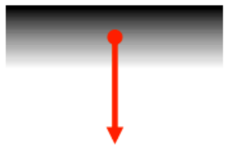
Image gradient

- The *gradient* of an image: $\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$

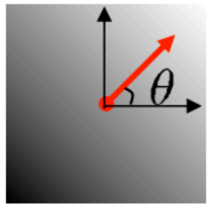
The gradient points in the direction of most rapid increase in intensity



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



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



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

The *edge strength* is given by the gradient magnitude:

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

The gradient direction is given by:

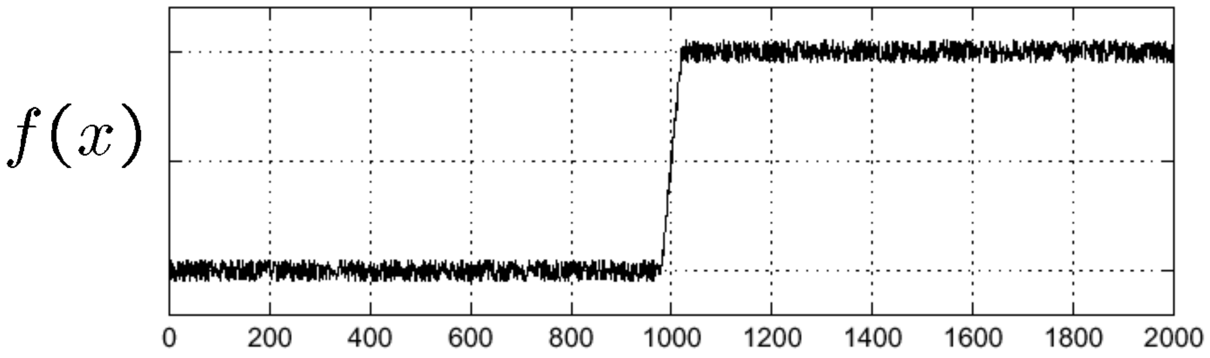
$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

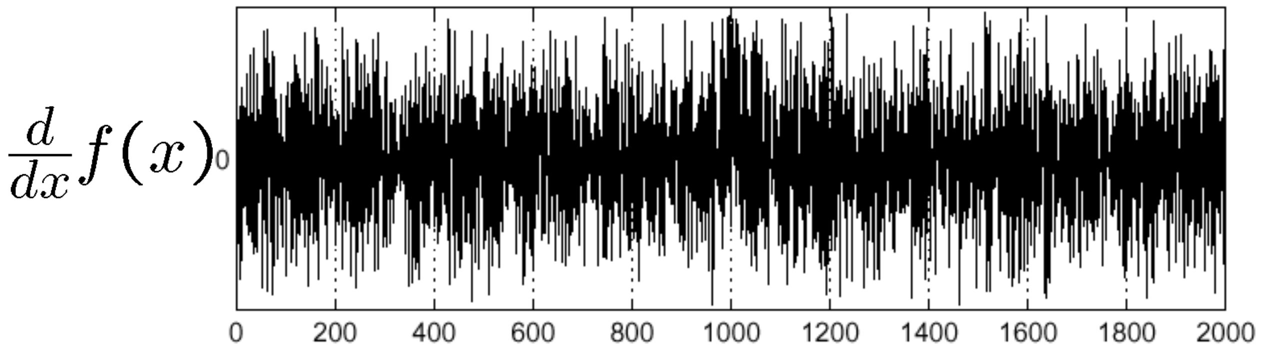
- how does this relate to the direction of the edge?

Source: Steve 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?

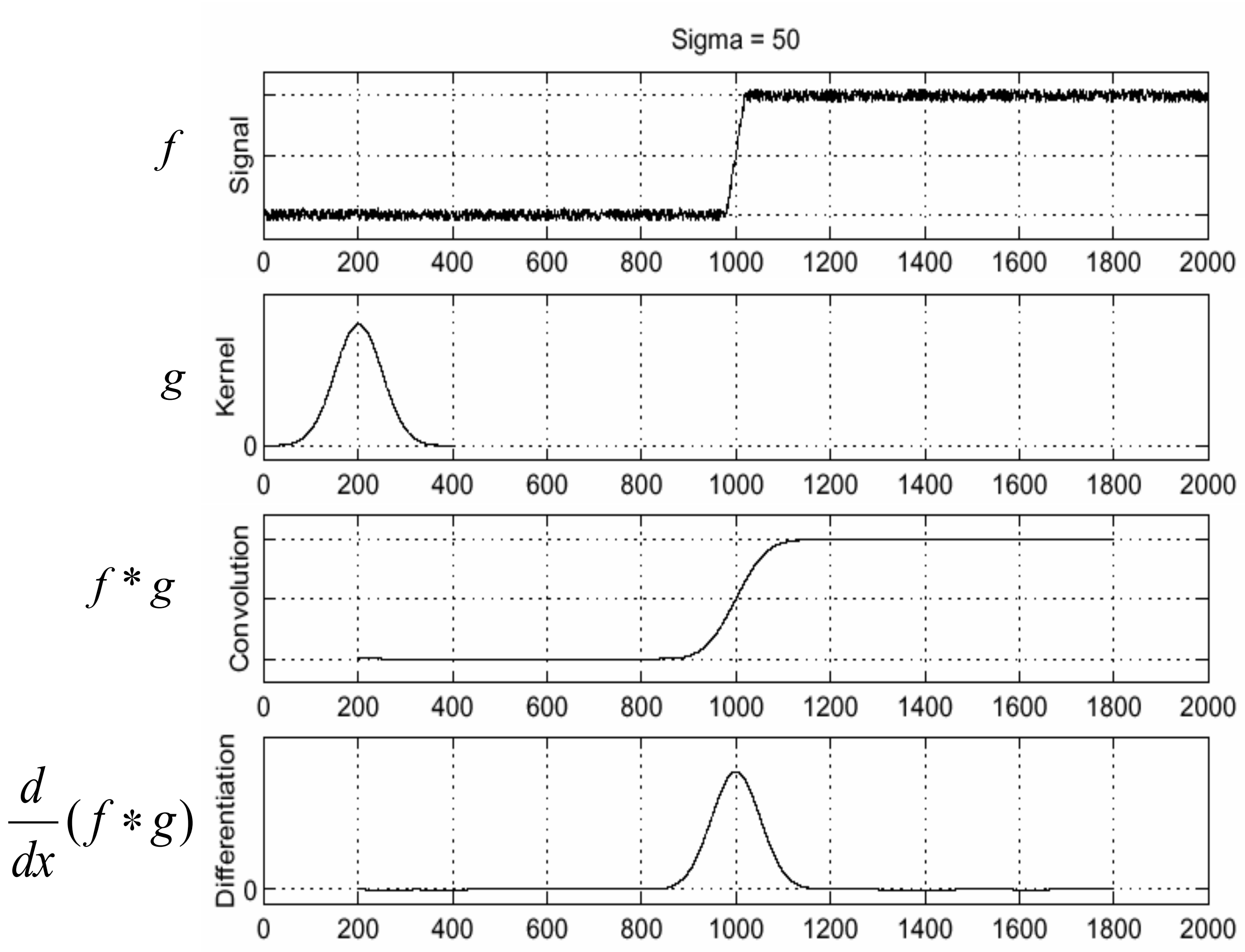
Source: S. Seitz

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

Solution: smooth first



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

Source: S. Seitz

Derivative theorem of convolution

- Differentiation is convolution, and convolution is associative:

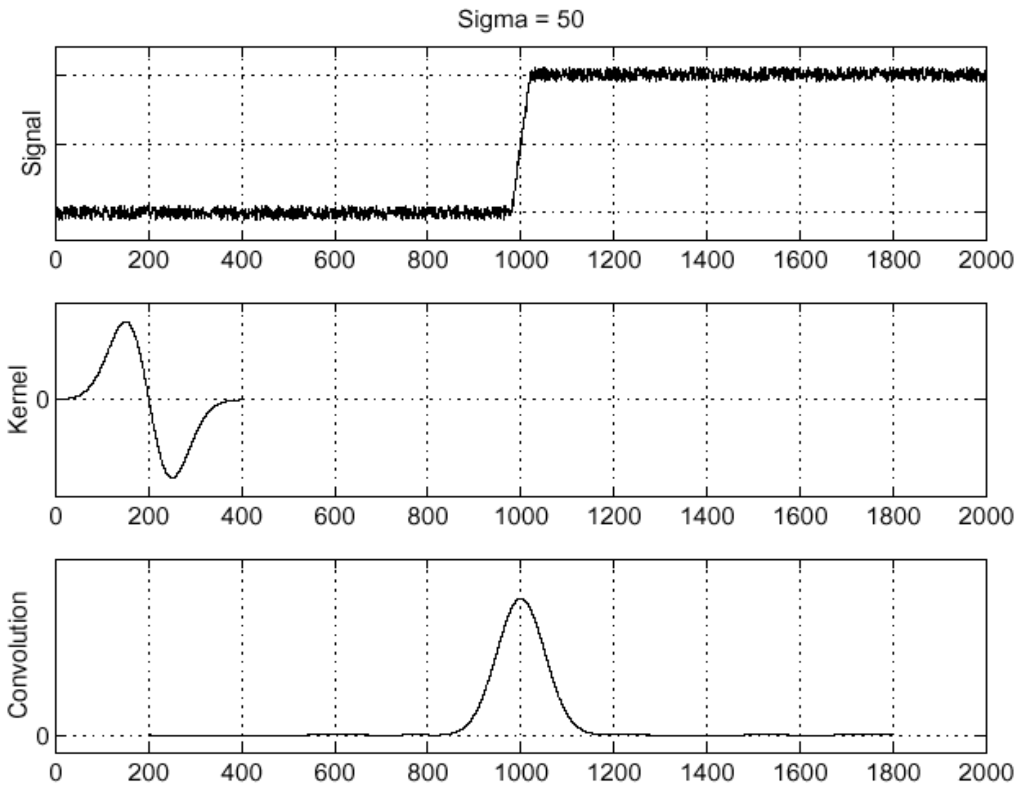
$$\frac{d}{dx}(f * g) = f * \frac{d}{dx} g$$

- This saves us one operation:

f

$\frac{d}{dx} g$

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



Source: S. Seitz

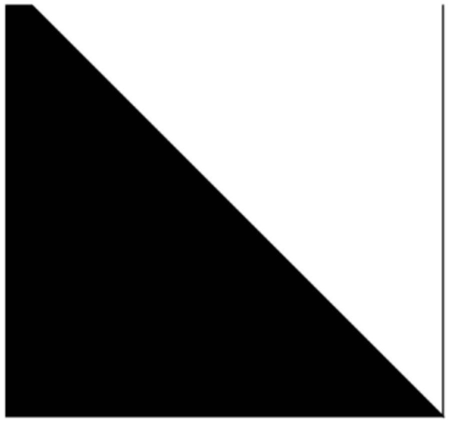
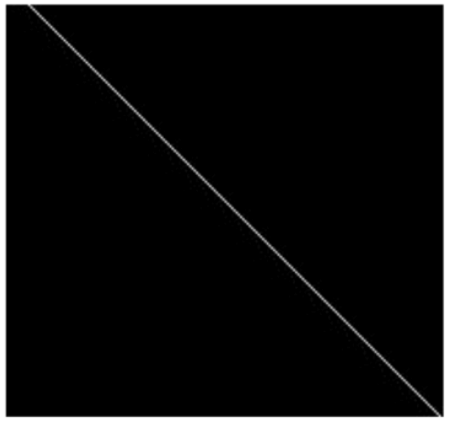


Image with Edge


Edge Location

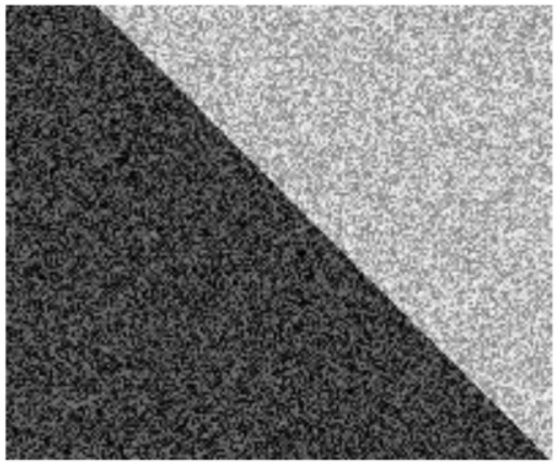
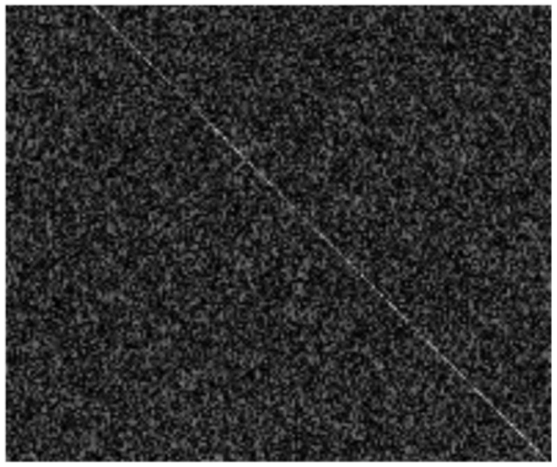
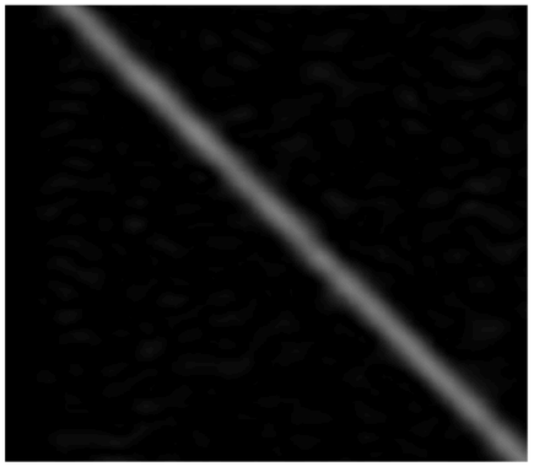
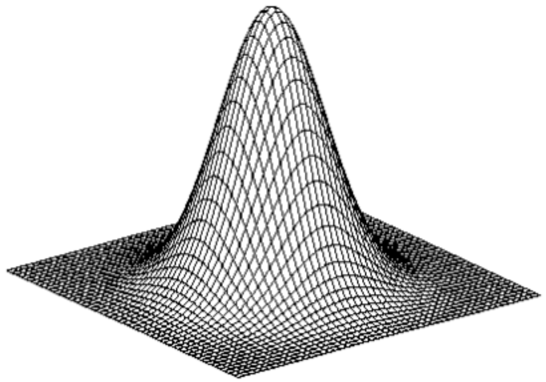


Image + Noise


Derivatives detect
edge *and* noise

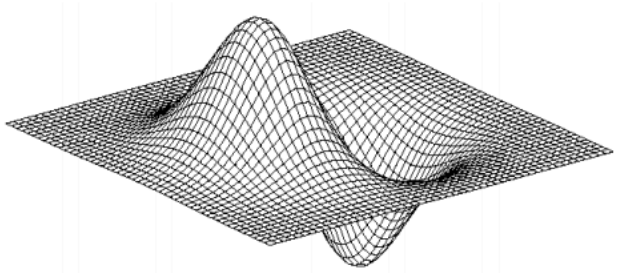

Smoothed derivative removes
noise, but blurs edge

2D edge detection filters



Gaussian

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



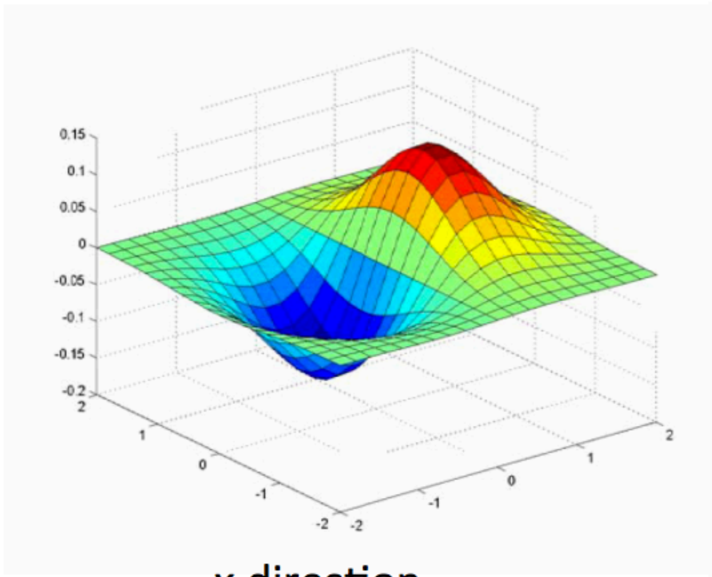
derivative of Gaussian (x)

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

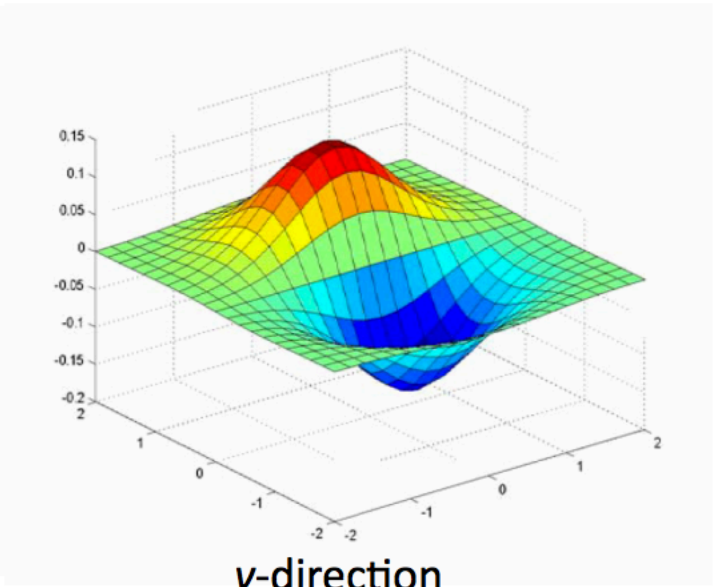
$$\nabla G_\sigma(\boldsymbol{x}) = (\frac{\partial G_\sigma}{\partial x}, \frac{\partial G_\sigma}{\partial y})(\boldsymbol{x}) = [-x \;\; -y] \quad \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

$$\nabla^2 G_\sigma(\boldsymbol{x}) = \frac{1}{\sigma^3}\left(2 - \frac{x^2+y^2}{2\sigma^2}\right)\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

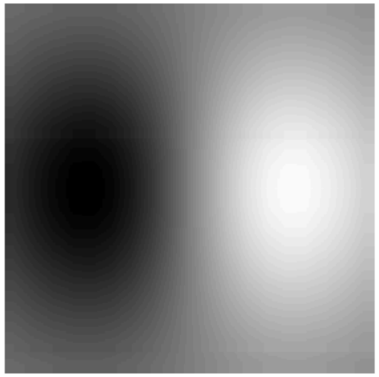
Derivative of Gaussian filter

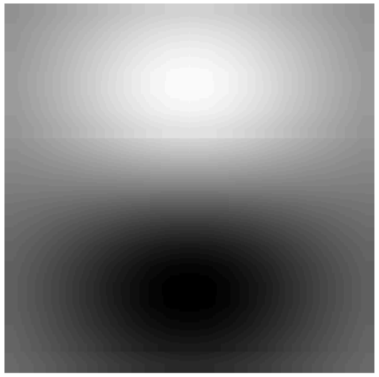


x-direction



y-direction





The Sobel operator

- Common approximation of derivative of Gaussian
  - A mask (not a convolution kernel)

$$\frac{1}{8}\begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array} \qquad \frac{1}{8}\begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array}$$

$$s_x \qquad\qquad\qquad s_y$$

- The standard defn. of the Sobel operator omits the 1/8 term
  - doesn't make a difference for edge detection
  - the 1/8 term **is** needed to get the right gradient magnitude

What kind of filters are those?

Sobel filter:

$$\mathbf{G}_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * \mathbf{A} \quad \text{and} \quad \mathbf{G}_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} * \mathbf{A}$$

Magnitude:

$$\mathbf{G} = \sqrt{\mathbf{G}_x{}^2 + \mathbf{G}_y{}^2}$$

Direction:

$$\mathbf{\Theta} = \operatorname{atan2}(\mathbf{G}_y, \mathbf{G}_x)$$

Tradeoff between smoothing and localization

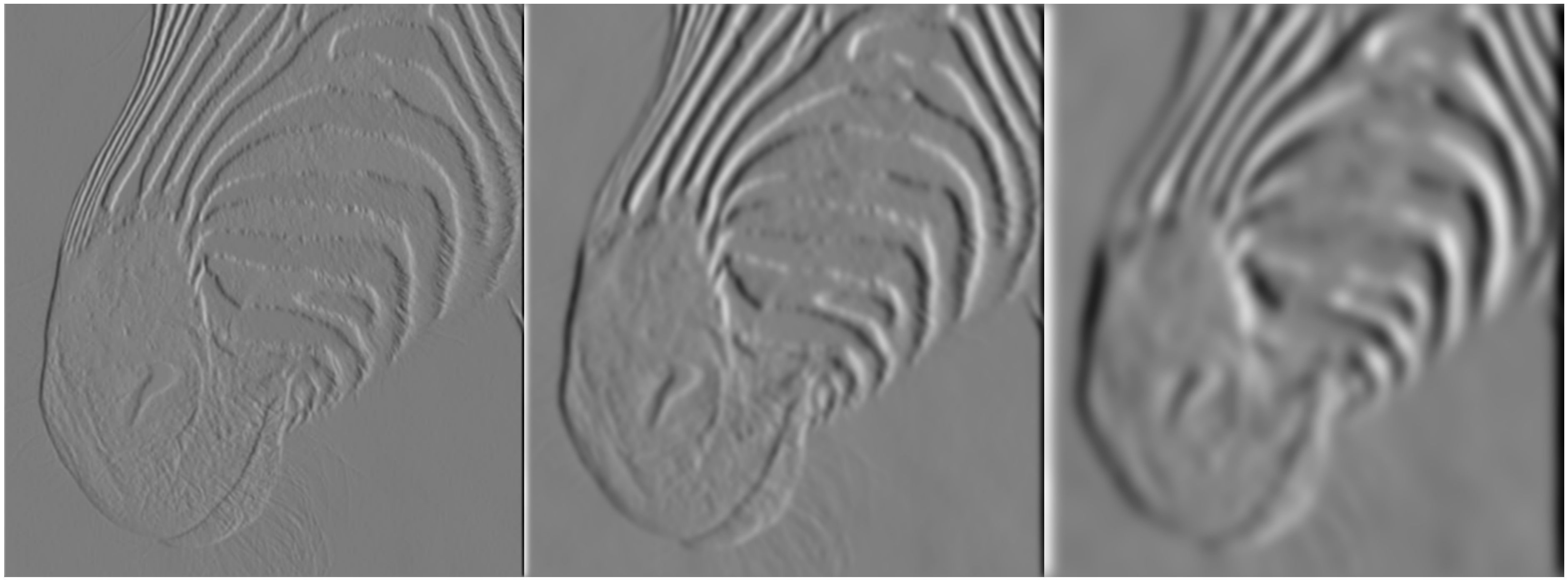


FIGURE 5.3: The scale (i.e., σ) of the Gaussian used in a derivative of Gaussian filter has significant effects on the results. The three images show estimates of the derivative in the x direction of an image of the head of a zebra obtained using a derivative of Gaussian filter with σ one pixel, three pixels, and seven pixels (**left** to **right**). Note how images at a finer scale show some hair, the animal's whiskers disappear at a medium scale, and the fine stripes at the top of the muzzle disappear at the coarser scale.

Source: D. Forsyth

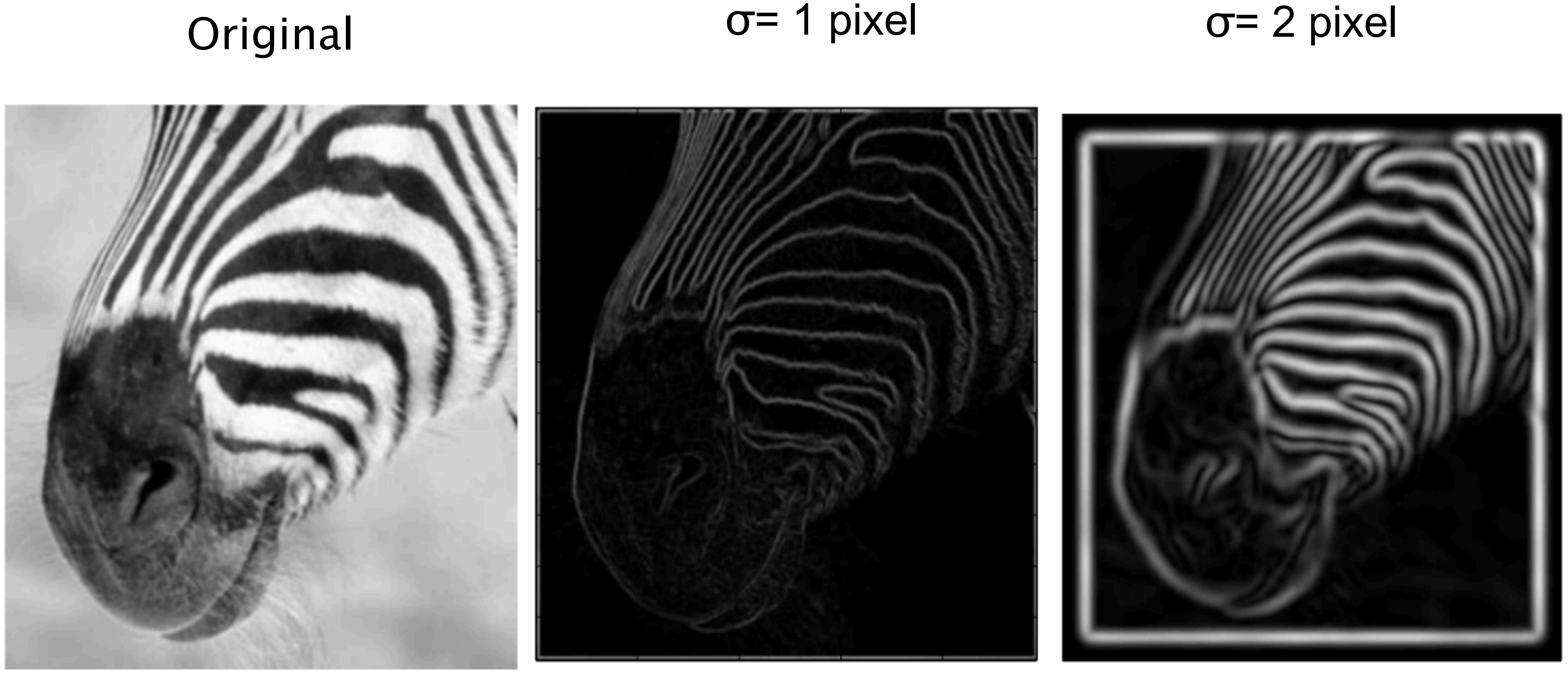
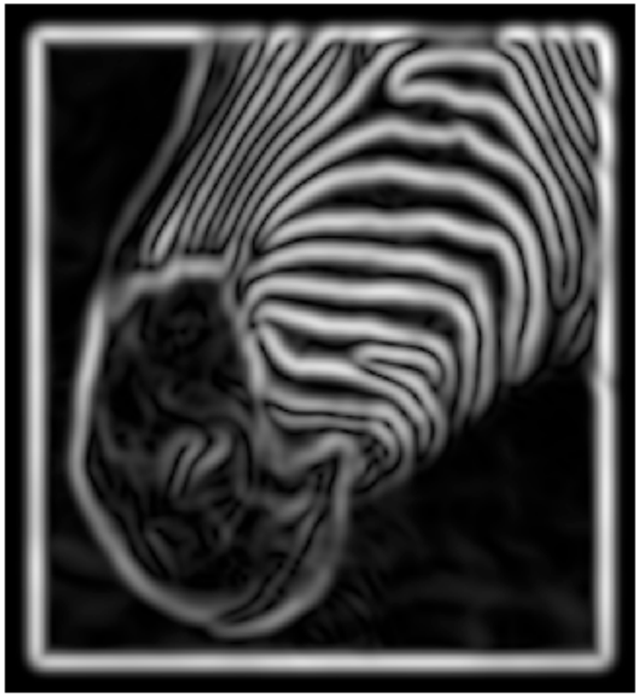


FIGURE 5.4: The gradient magnitude can be estimated by smoothing an image and then differentiating it. This is equivalent to convolving with the derivative of a smoothing kernel. The extent of the smoothing affects the gradient magnitude; in this figure, we show the gradient magnitude for the figure of a zebra at different scales. At the **center**, gradient magnitude estimated using the derivatives of a Gaussian with $\sigma = 1$ pixel; and on the **right**, gradient magnitude estimated using the derivatives of a Gaussian with $\sigma = 2$ pixel. Notice that large values of the gradient magnitude form thick trails.

Source: D. Forsyth

Implementation issues



- The gradient magnitude is large along a thick "trail" or "ridge," so how do we identify the actual edge points?
- How do we link the edge points to form curves?

Source: D. Forsyth

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: 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: L. Fei-Fei

Canny edge detector

- Minimizes the probability of multiply detecting an edge
- Minimizes the probability of failing to detect an edge
- Minimizes the distance of the reported edge to the true edge.

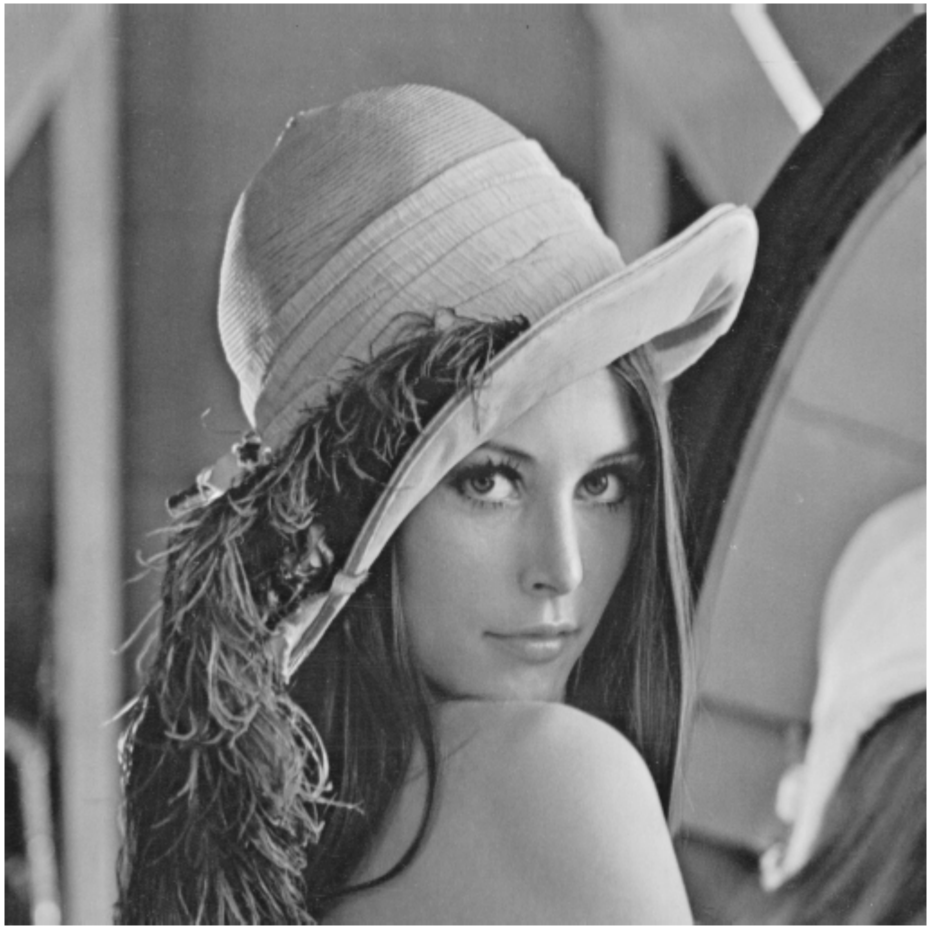
J. Canny, ***A Computational Approach To Edge Detection***, IEEE Trans. Pattern Analysis and Machine Intelligence, 8:679-714, 1986.

Source: L. Fei-Fei

Note about Matlab's Canny detector

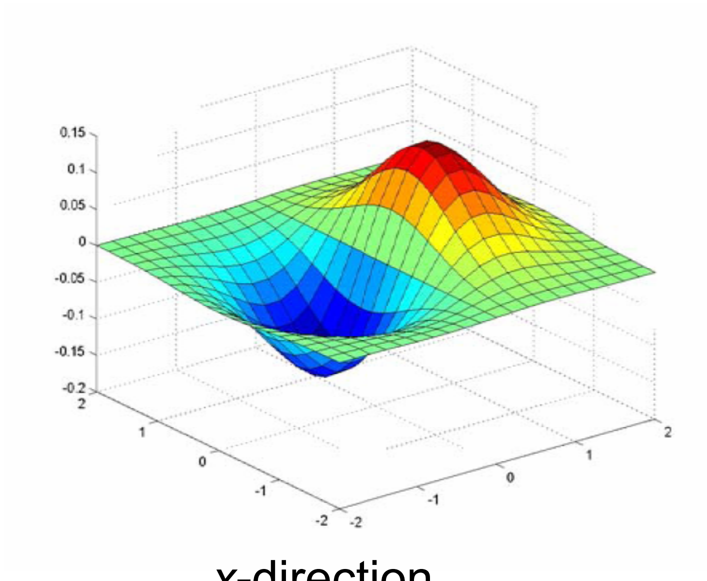
- Small errors in implementation
  - Gaussian function not properly normalized
  - First filters with a Gaussian, then a difference of Gaussian (equivalent to filtering with a larger Gaussian and taking difference)

Example

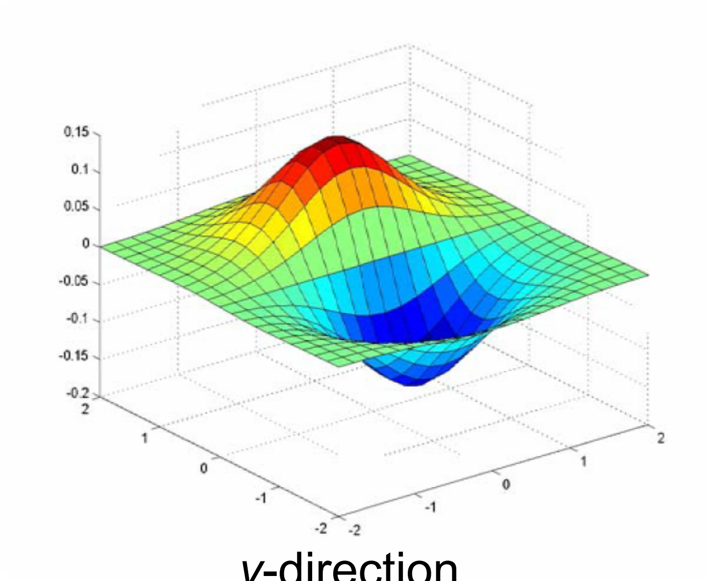


original image (Lena)

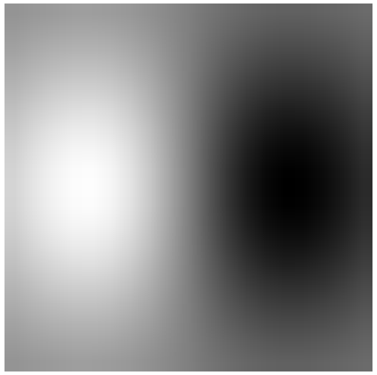
Derivative of Gaussian filter

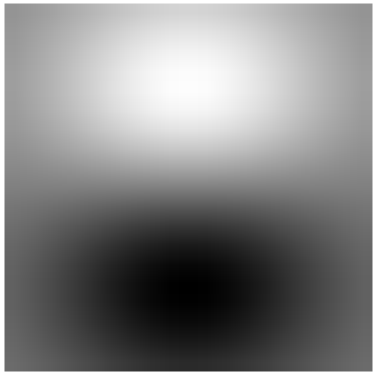


x-direction

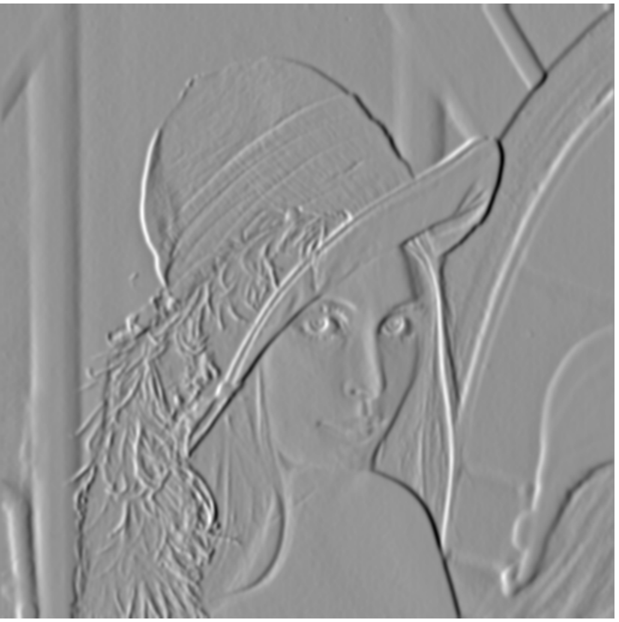


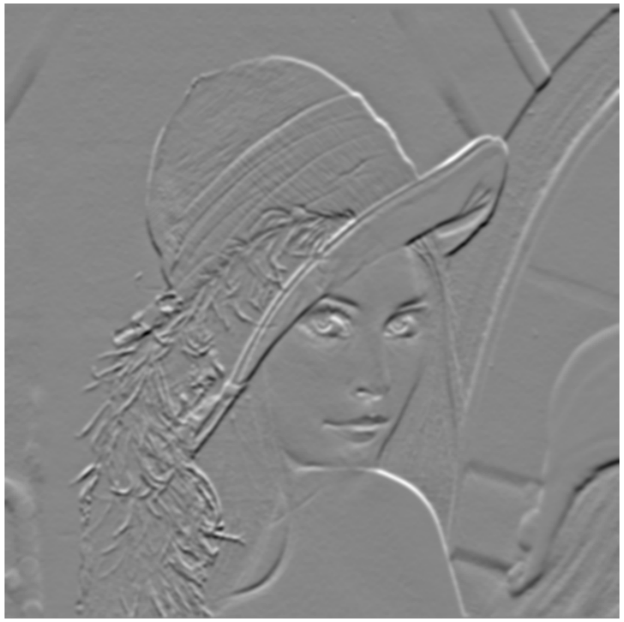
y-direction

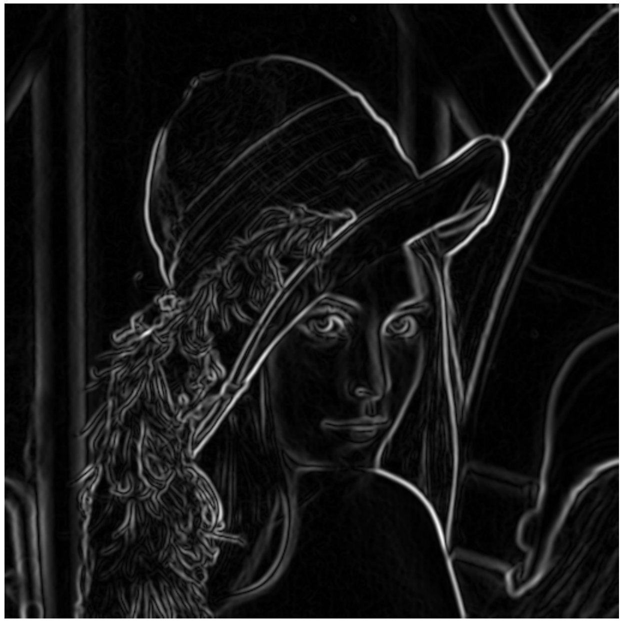




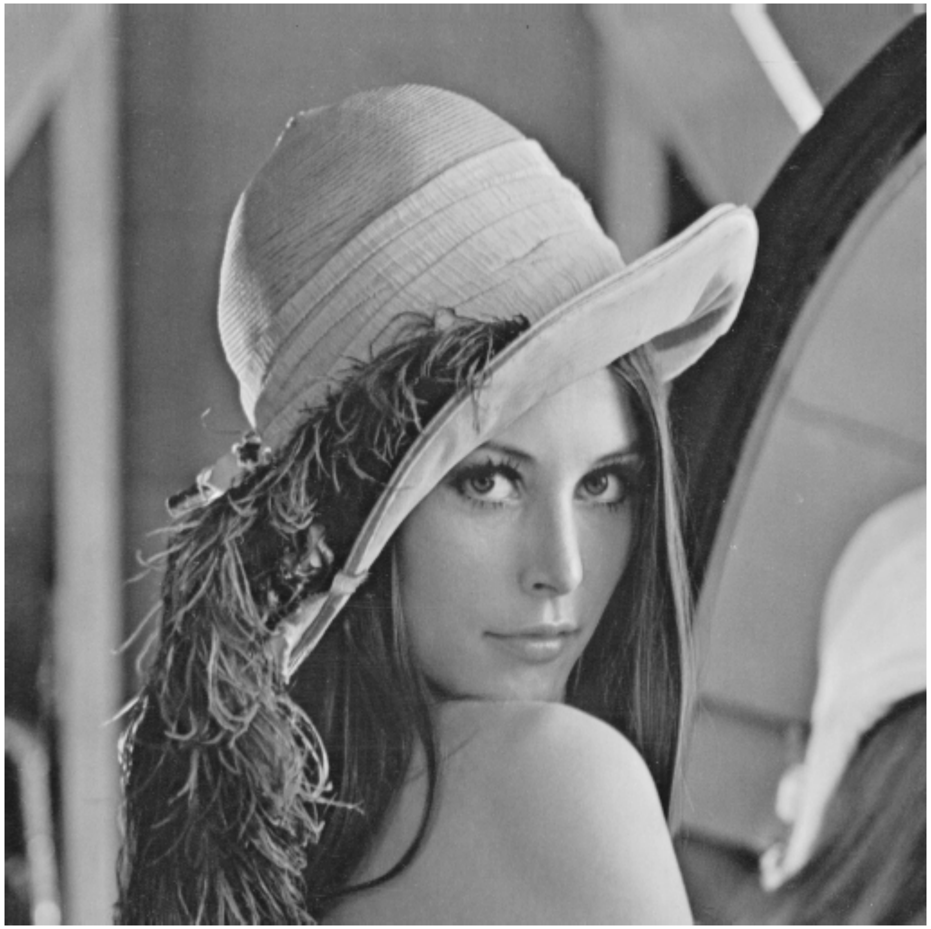
Compute Gradients (DoG)


X-Derivative of Gaussian


Y-Derivative of Gaussian

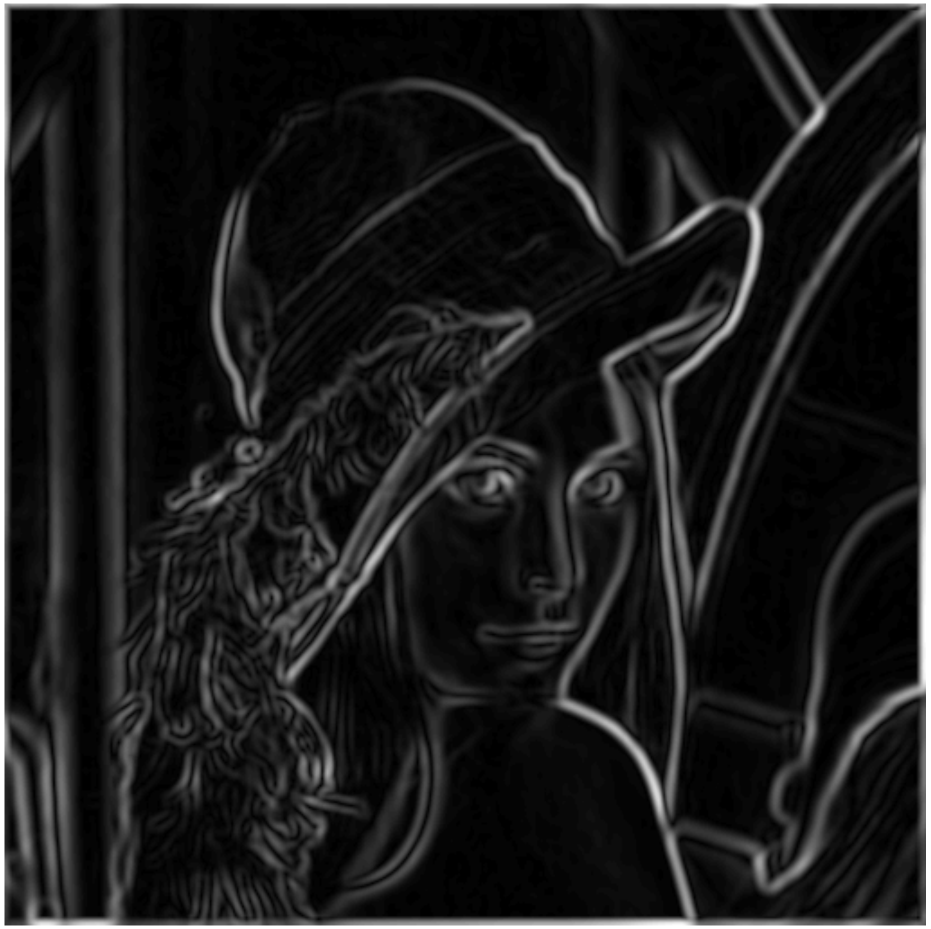

Gradient Magnitude

The Canny edge detector



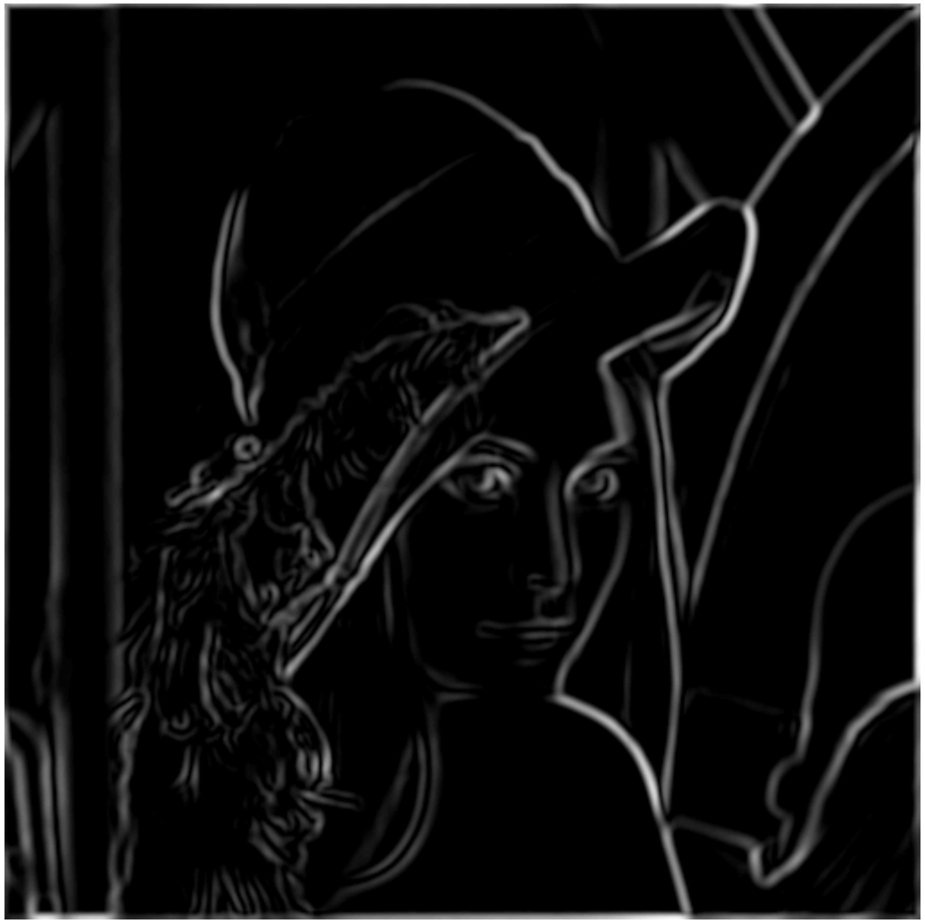
- original image (Lena)

The Canny edge detector



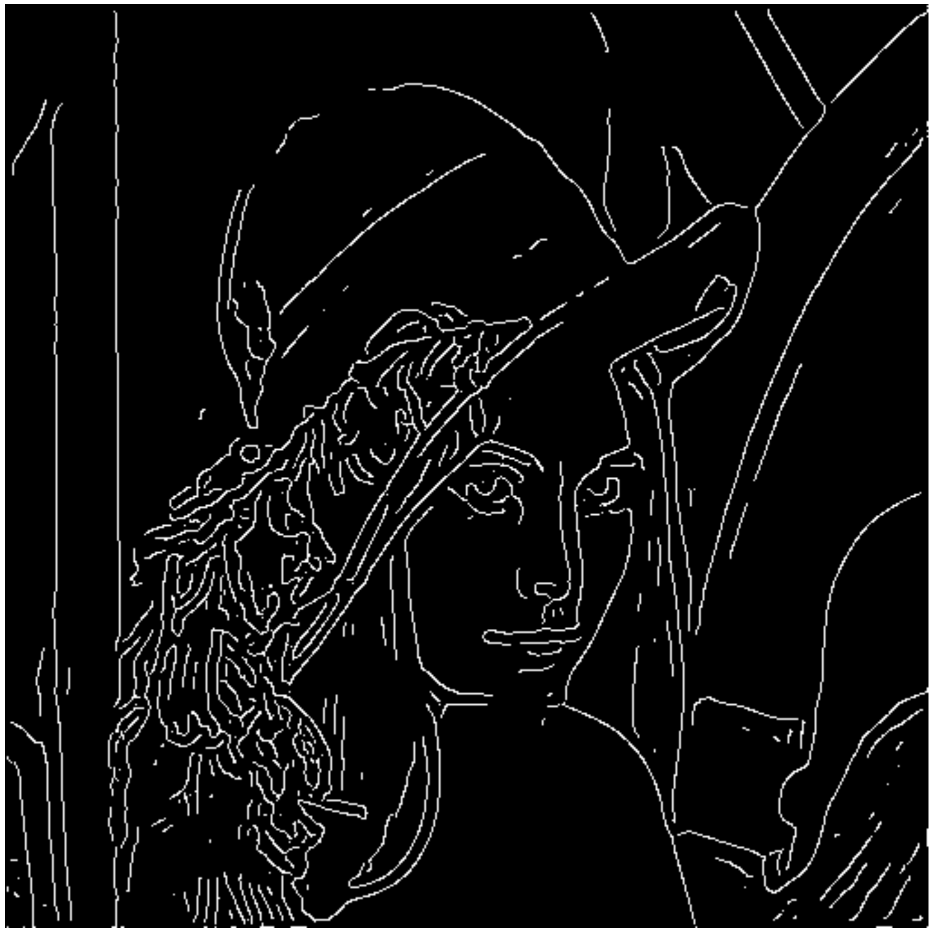
norm of the gradient

The Canny edge detector



thresholding

The Canny edge detector

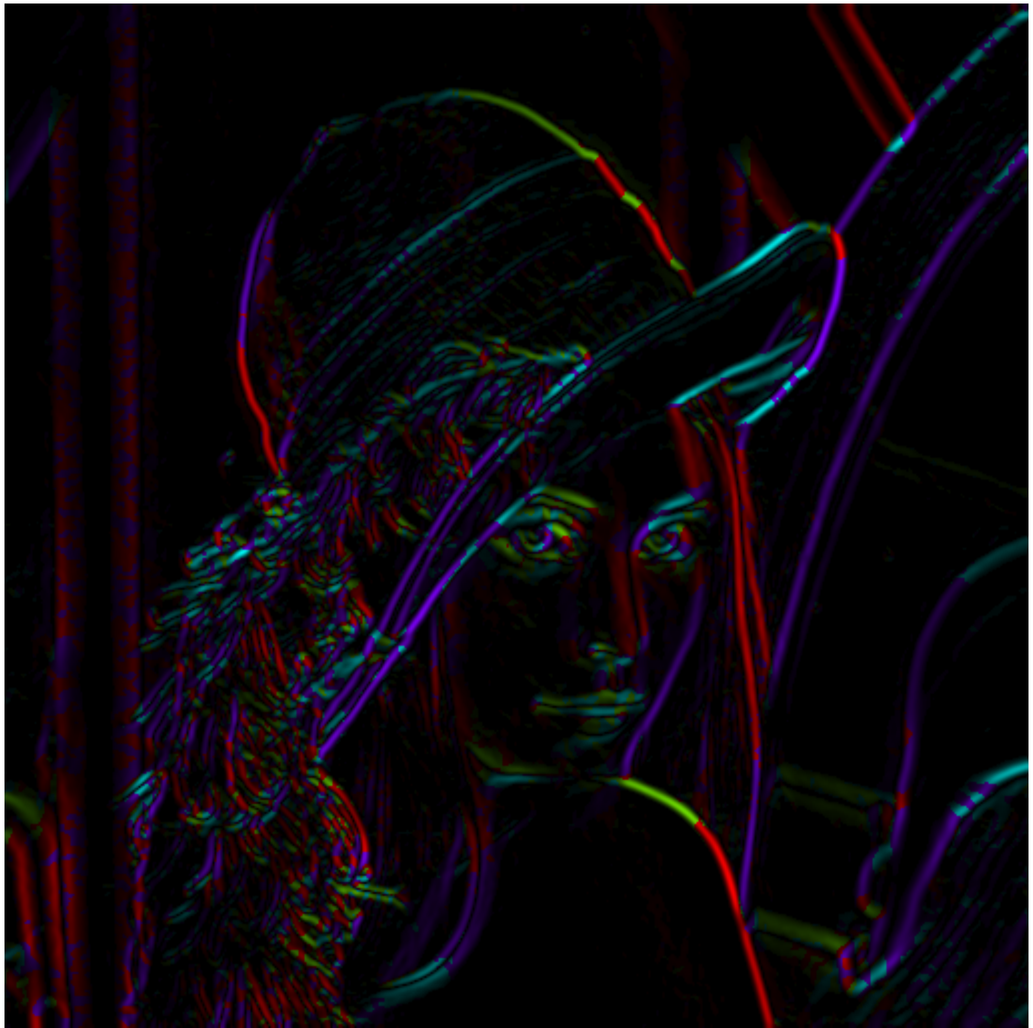


thinning

(non-maximum suppression)

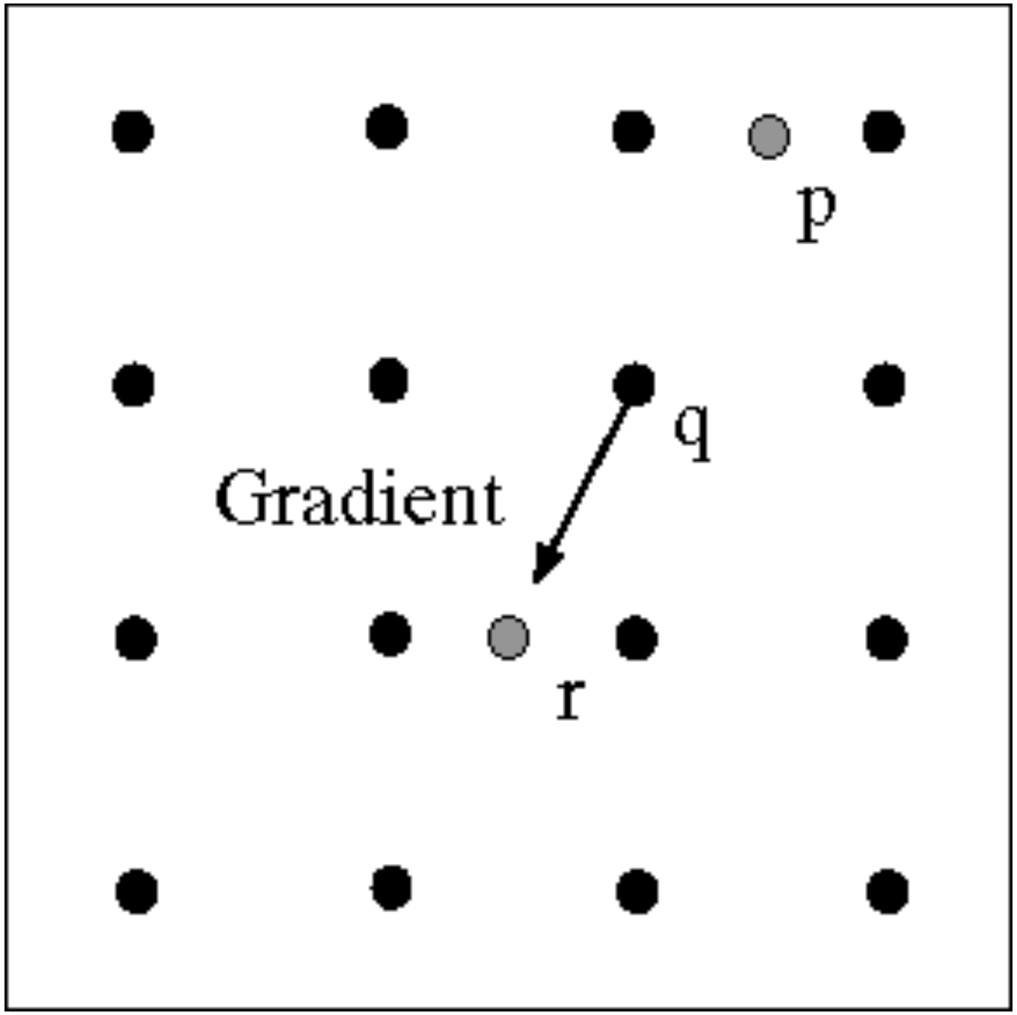
Get Orientation at Each Pixel

- Threshold at minimum level
- Get orientation

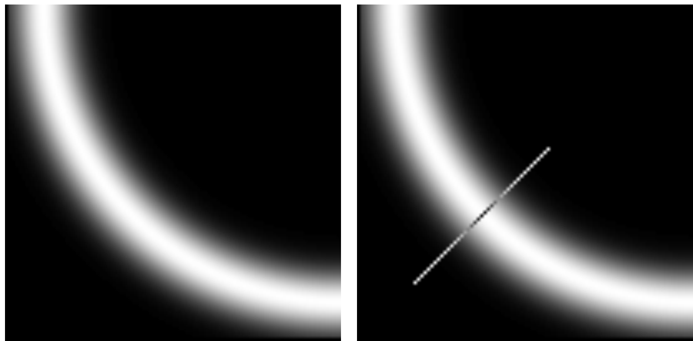


theta = atan2(gy, gx)

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

Non-maximum suppression

- m(x,y) is the local peak
- Canny calls local peak detection non-maximum suppression

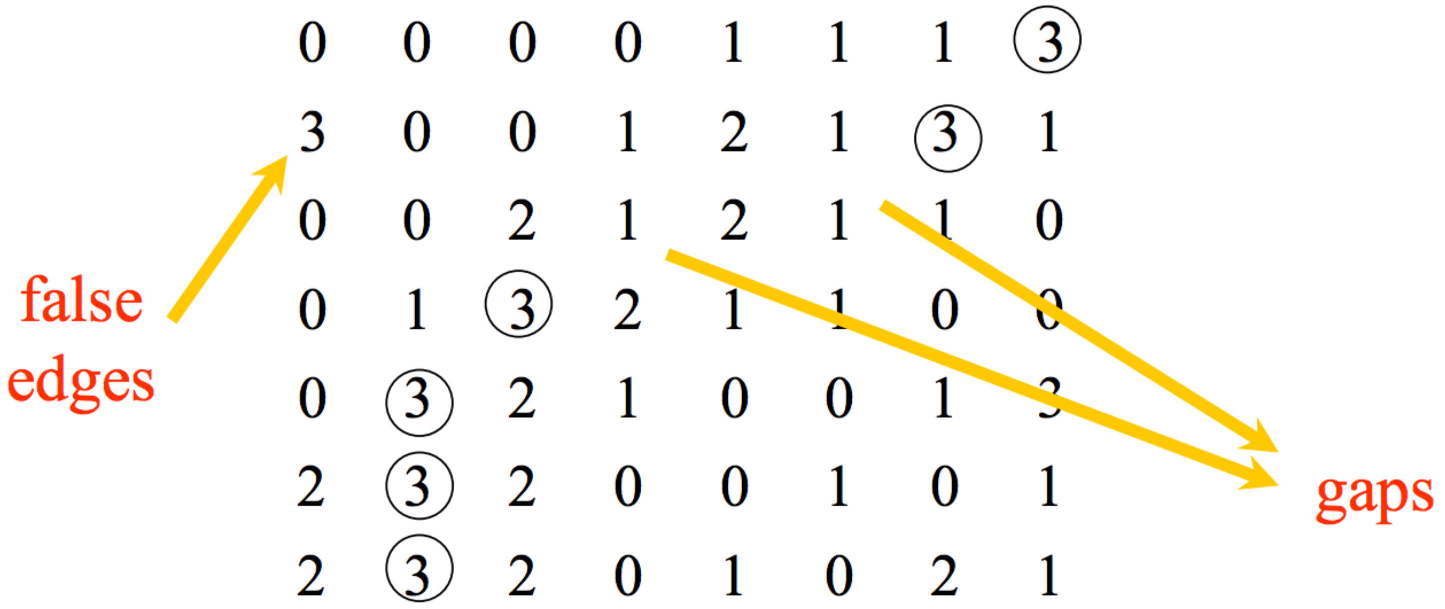
$$m(x, y) > m(x + \delta x, y + \delta y),$$

$$m(x, y) > m(x - \delta x, y - \delta y).$$

M(x,y) is a local peak whenever it is greater than the values in the gradient direction and the opposite of the gradient direction.

Non-maximum suppression

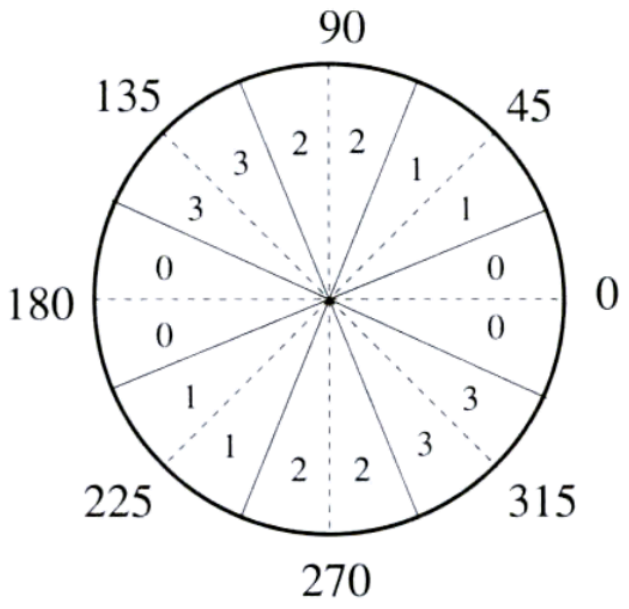
- Thin the broad ridges in M[i,j] into ridges that are only one pixel wide
- Find local maxima in M[i,j] by suppressing all values along the line of the Gradient that are not peak values of the ridge



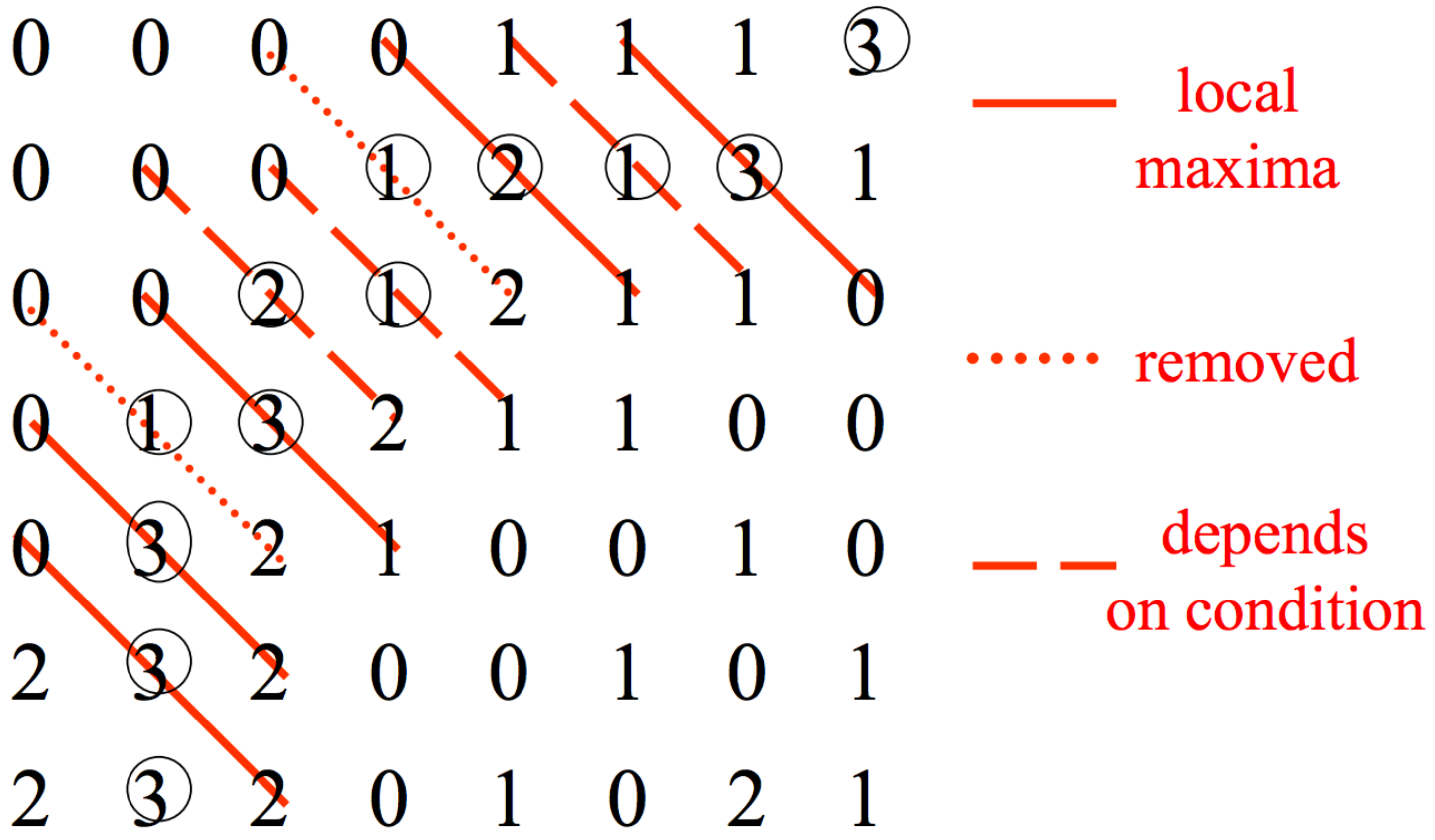
0	0	0	0	1	1	1	③
3	0	0	1	2	1	③	1
0	0	2	1	2	1	1	0
0	1	③	2	1	1	0	0
0	③	2	1	0	0	1	3
2	③	2	0	0	1	0	1
2	③	2	0	1	0	2	1

Non-maximum suppression

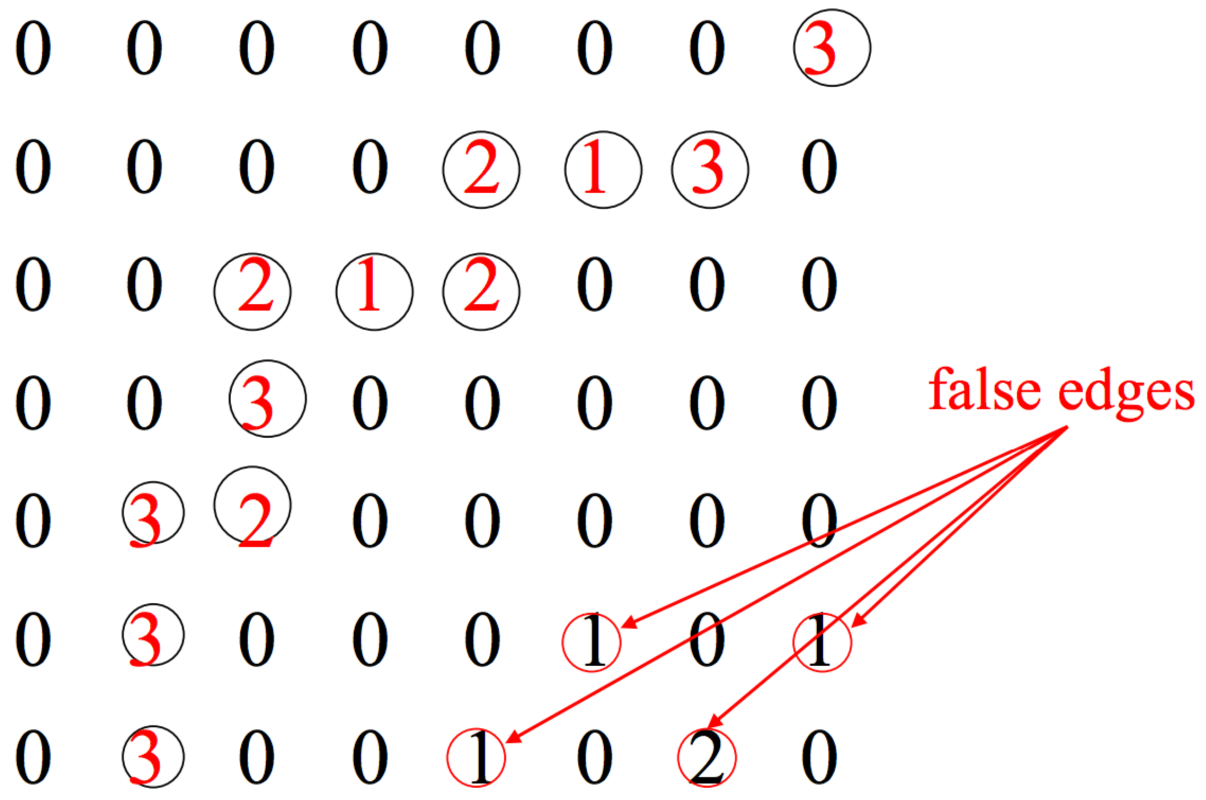
- Reduce angle of Gradient θ[i,j] to one of the 4 sectors
- Check the 3x3 region of each M[i,j]
- If the value at the center is not greater than the 2 values along the gradient, then M[i,j] is set to 0



Non-maximum suppression



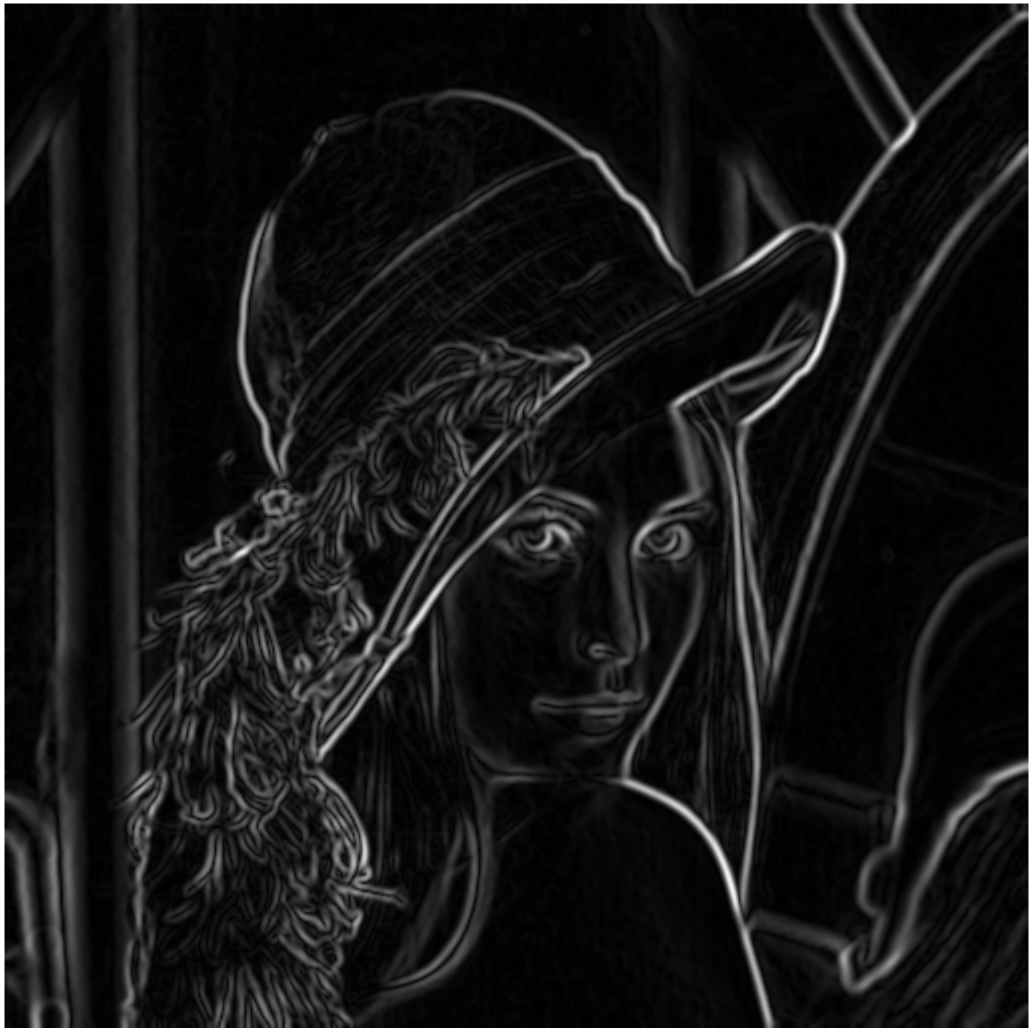
Non-maximum suppression



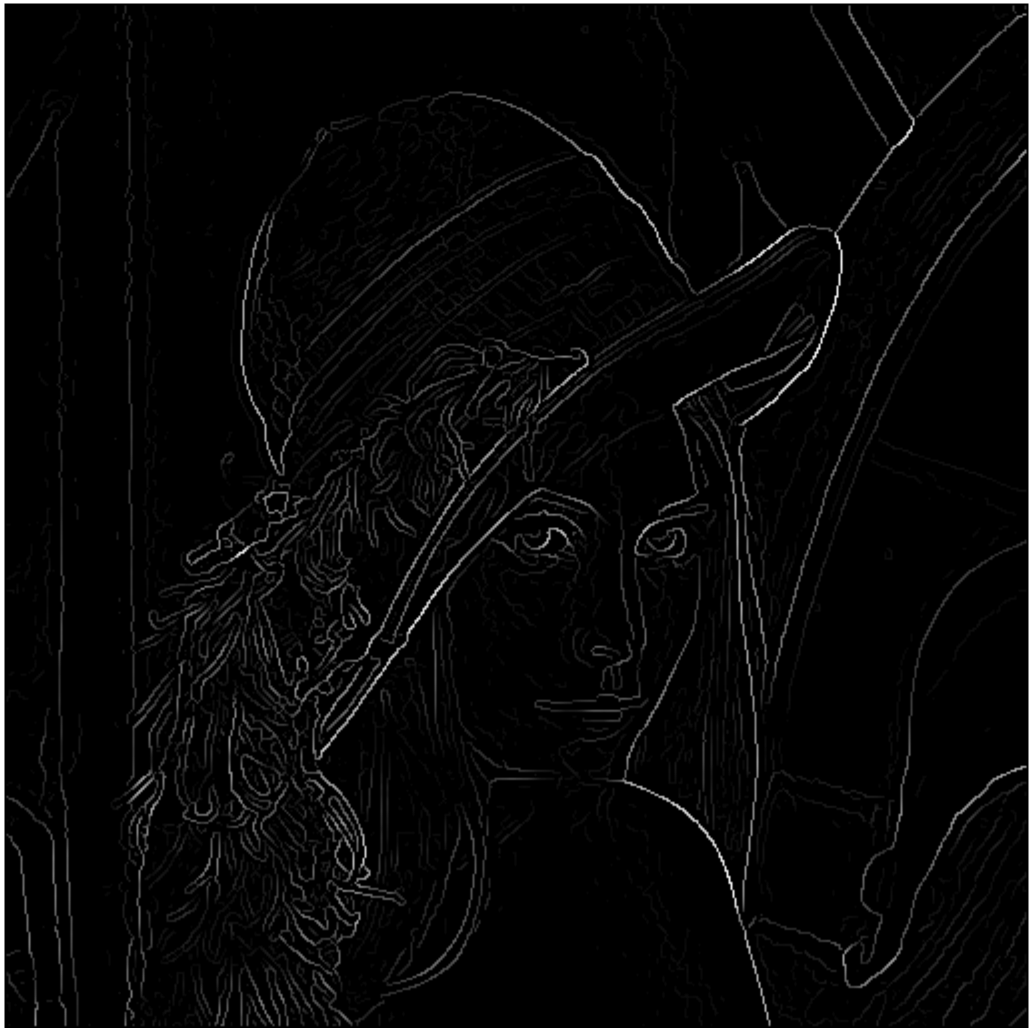
0	0	0	0	0	0	0	3
0	0	0	0	2	1	3	0
0	0	2	1	2	0	0	0
0	0	3	0	0	0	0	0
0	3	2	0	0	0	0	0
0	3	0	0	0	1	0	1
0	3	0	0	1	0	2	0

- The suppressed magnitude image will contain many false edges caused by noise or fine texture

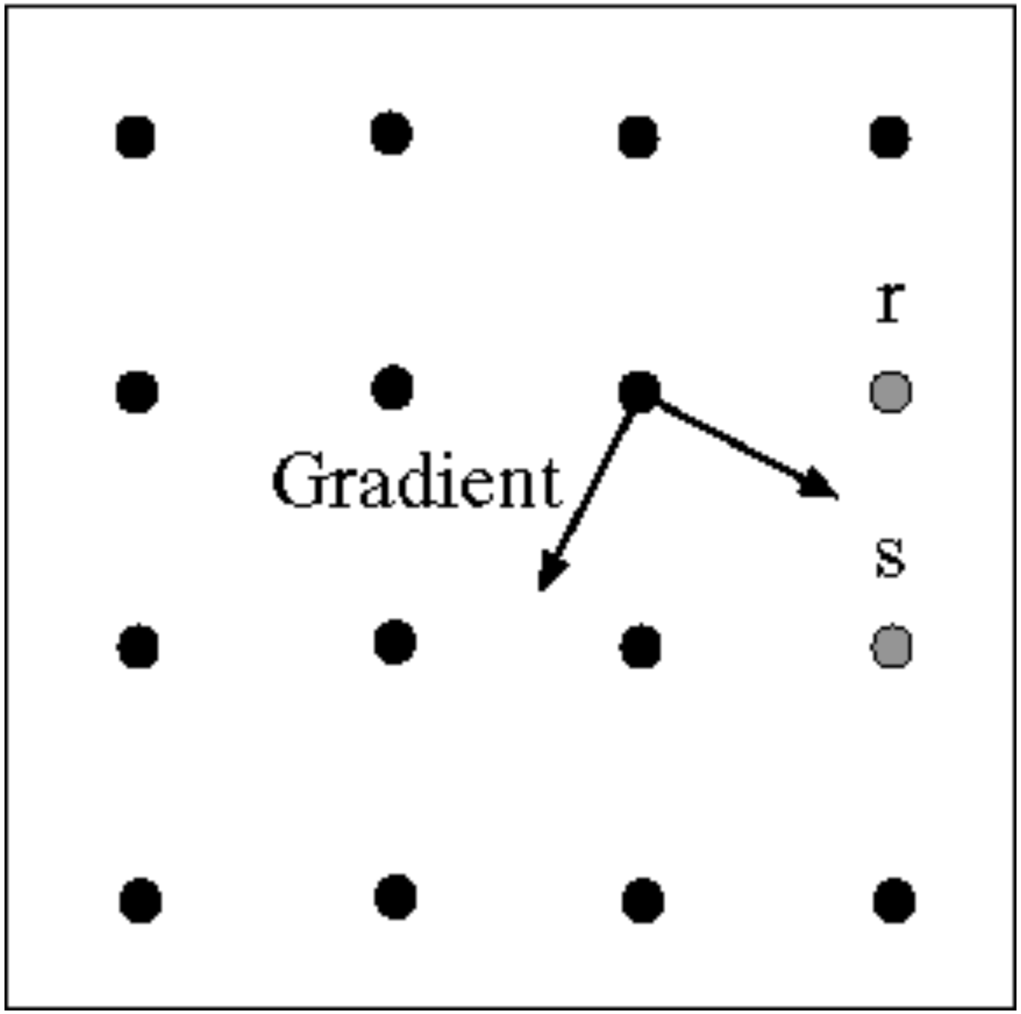
Before Non-max Suppression



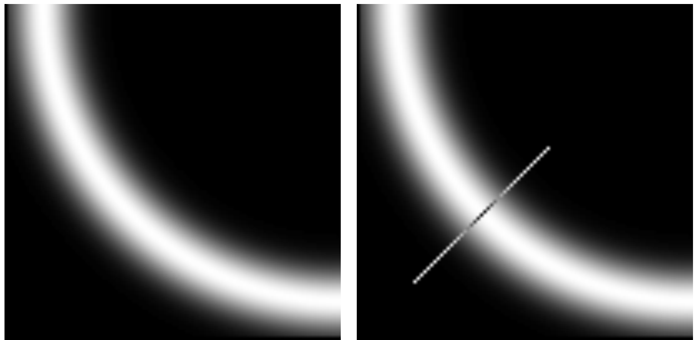
After non-max suppression



Edge linking



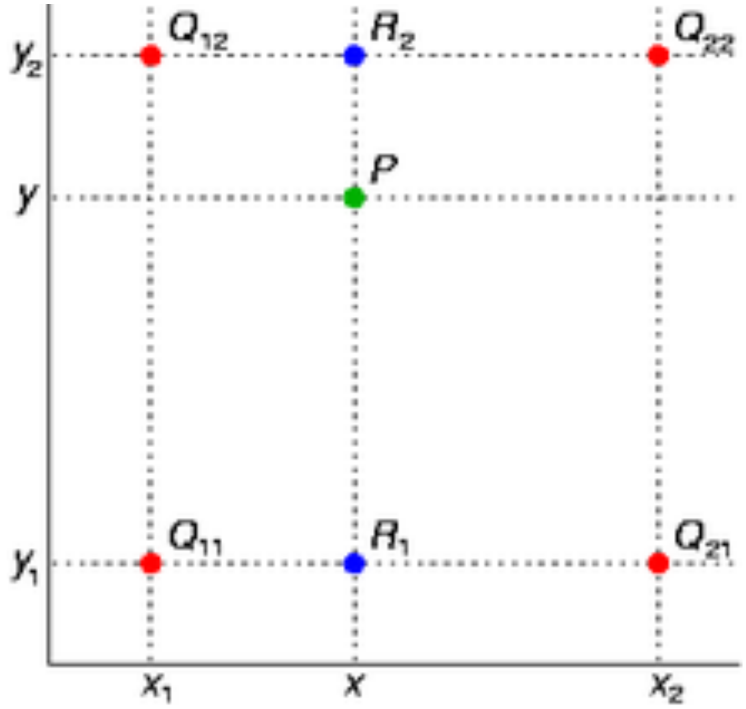
Assume the marked point is an edge point. Then we construct the tangent to the edge curve (which is normal to the gradient at that point) and use this to predict the next points (here either r or s).

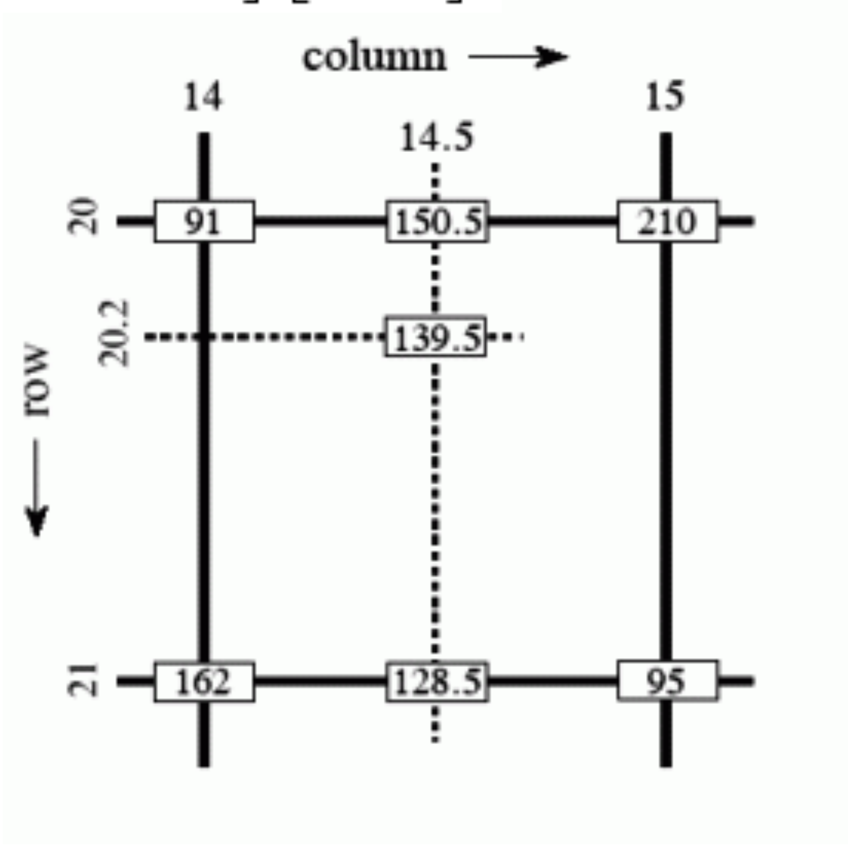


Source: D. Forsyth

Sidebar: Bilinear Interpolation

$$f(x,y) \approx \begin{bmatrix}1-x & x\end{bmatrix} \begin{bmatrix} f(0,0) & f(0,1) \\ f(1,0) & f(1,1) \end{bmatrix} \begin{bmatrix} 1-y \\ y \end{bmatrix}.$$



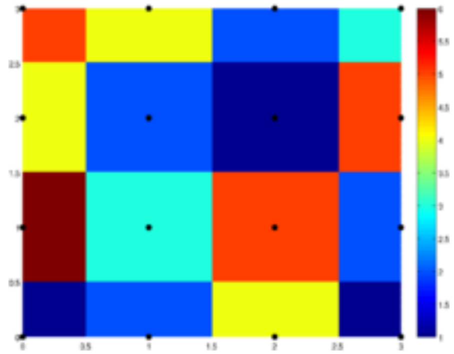


http://en.wikipedia.org/wiki/Bilinear_interpolation

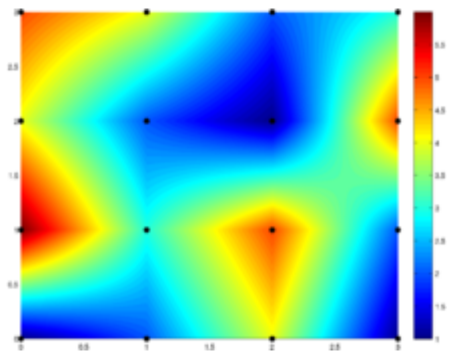
Sidebar: Interpolation options

- imx2 = imresize(im, 2, interpolation_type)

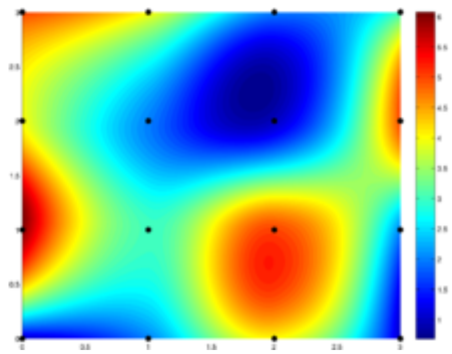
- 'nearest'
  - Copy value from nearest known
  - Very fast but creates blocky edges



- 'bilinear'
  - Weighted average from four nearest known pixels
  - Fast and reasonable results

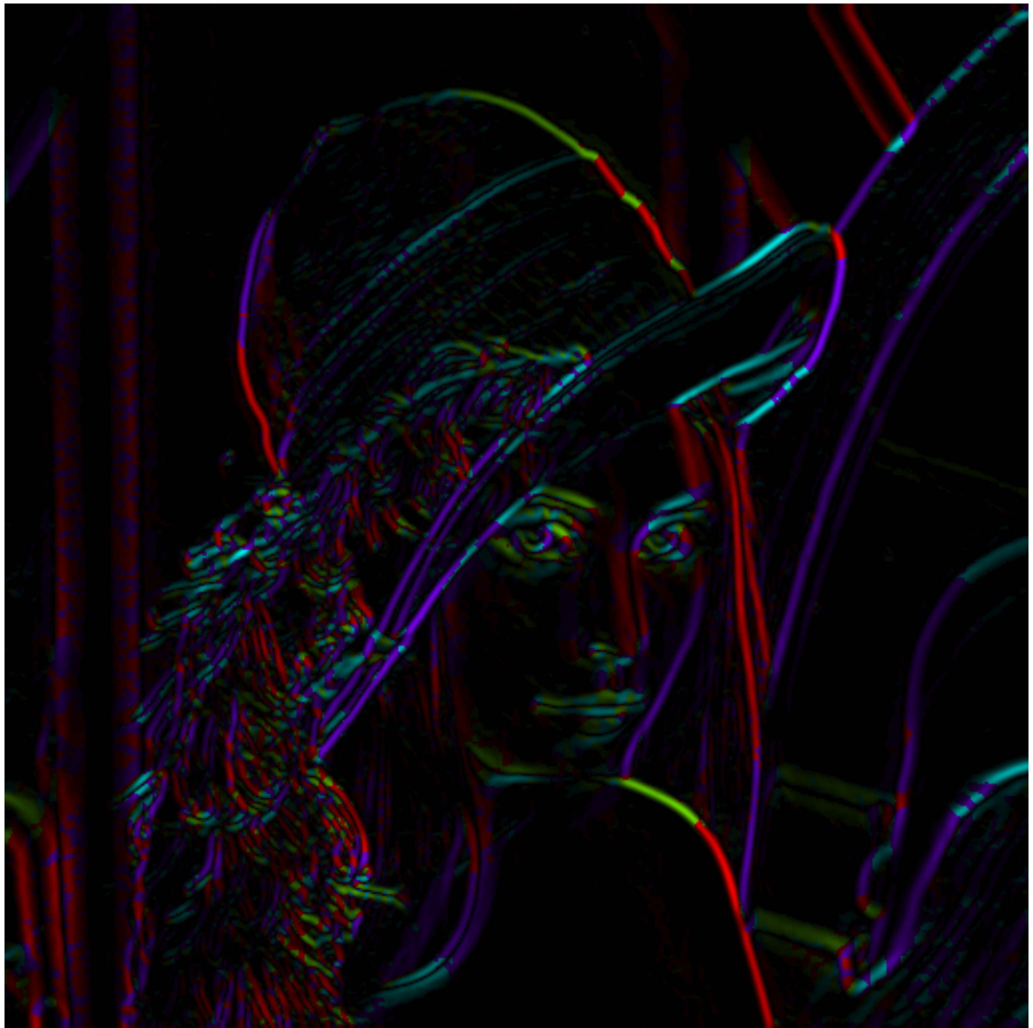


- 'bicubic' (default)
  - Non-linear smoothing over larger area (4x4)
  - Slower, visually appealing, may create negative pixel values

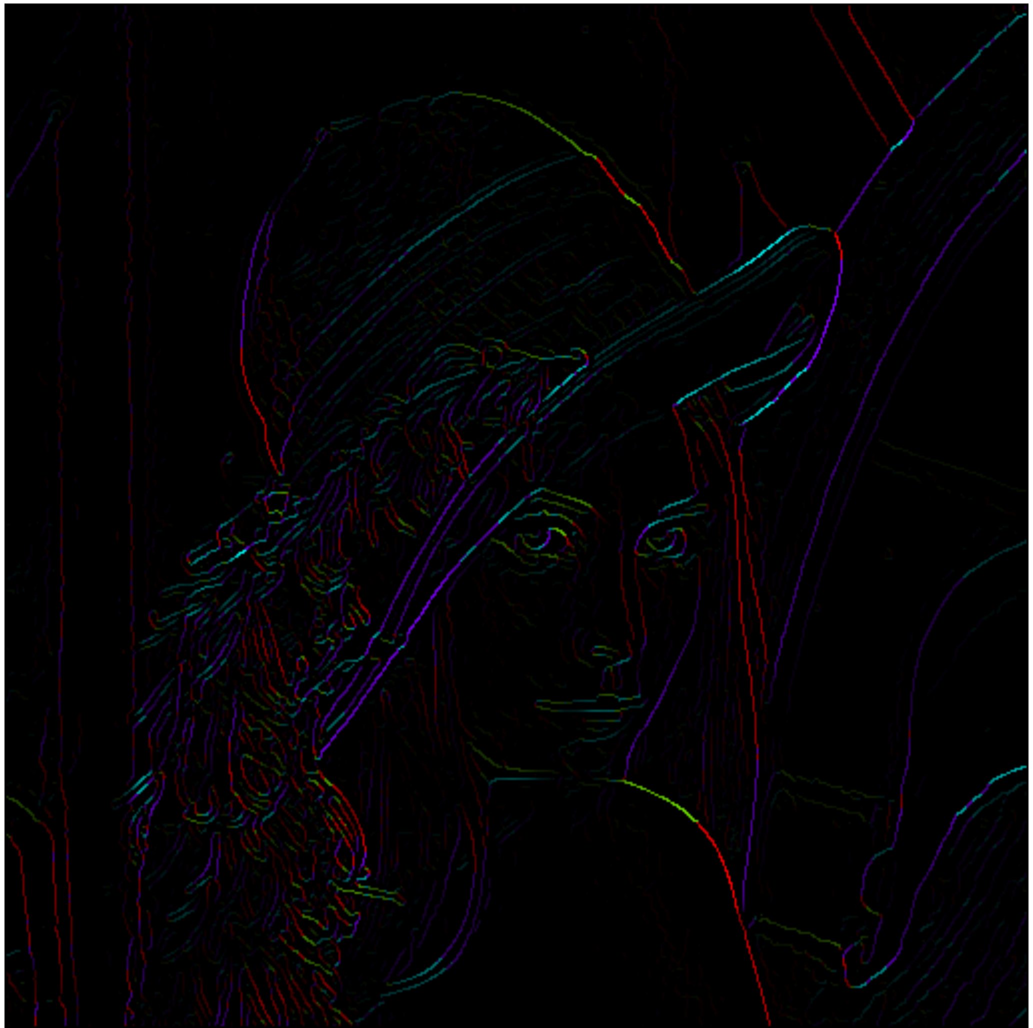


Examples from http://en.wikipedia.org/wiki/Bicubic_interpolation

Before Non-max Suppression



After non-max suppression

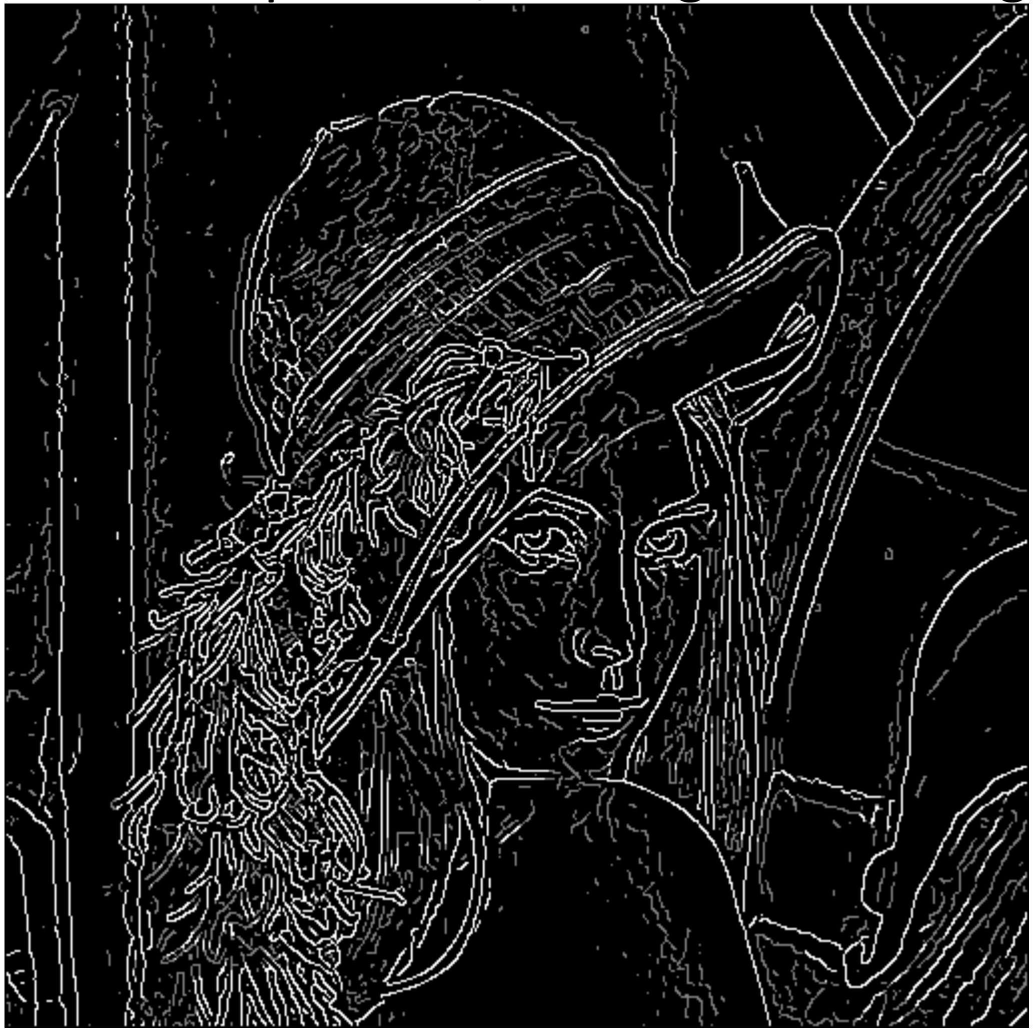


Double thresholding

- The remaining pixels are then categorized into strong, weak and non-edges.
- Above a high threshold, mark as strong.
- Between high and lo, mark as weak.
- Below lo threshold, mark as non-edges

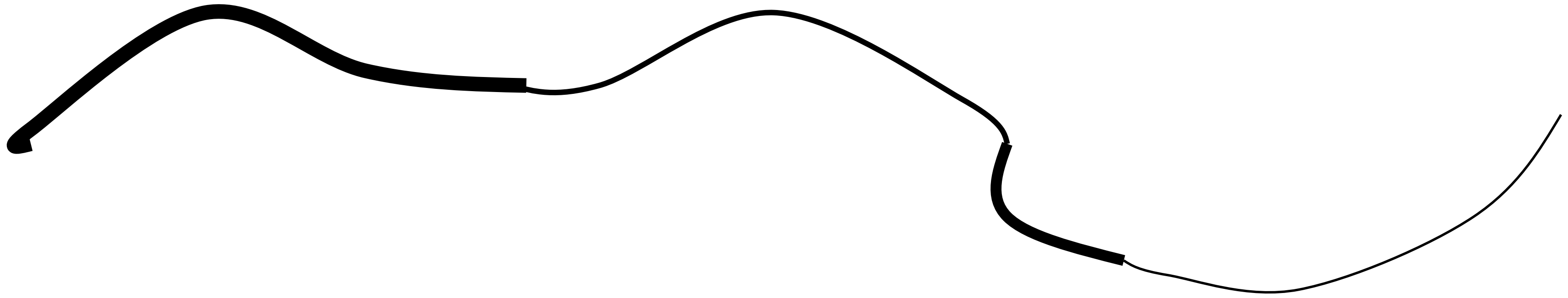
Hysteresis thresholding

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



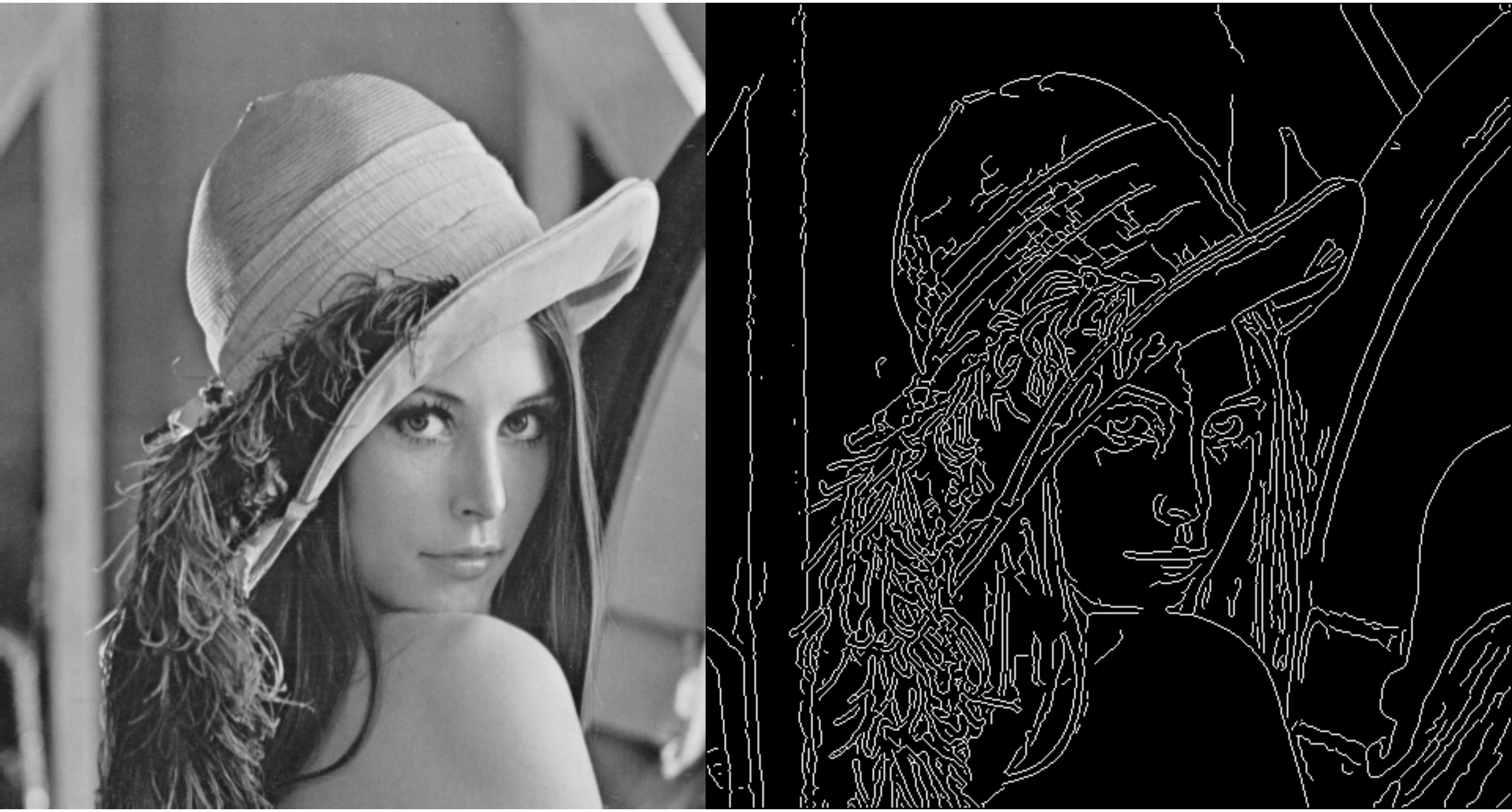
Hysteresis thresholding

- 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: S. Seitz

Final Canny Edges



Canny Edge Operator

- Smooth image I with 2D Gaussian: $G * I$
- Find local edge normal directions for each pixel

$$\bar{\mathbf{n}} = \frac{\nabla(G * I)}{|\nabla(G * I)|}$$

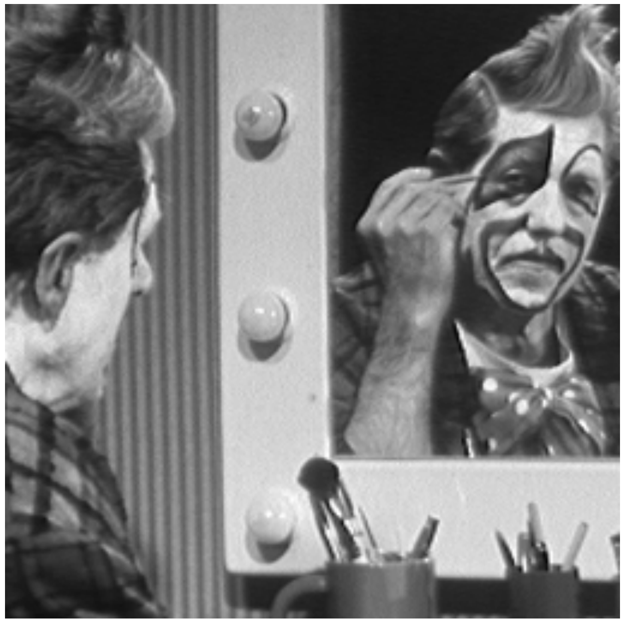
- Compute edge magnitudes $|\nabla(G * I)|$
- Find the location of the edges by finding zero-crossings along the edge normal directions (**non-maximum suppression**)

$$\frac{\partial^2 (G * I)}{\partial \bar{\mathbf{n}}^2} = 0$$

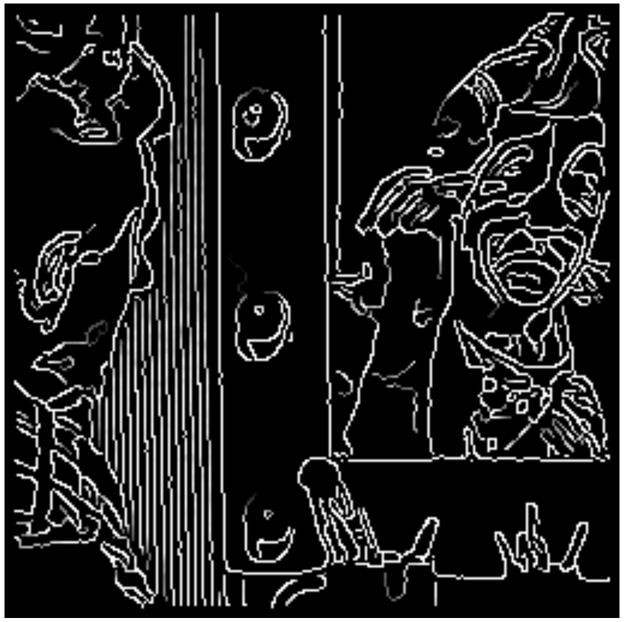
- Threshold edges in the image with **hysteresis** to eliminate spurious responses

Read Canny's original paper for further details

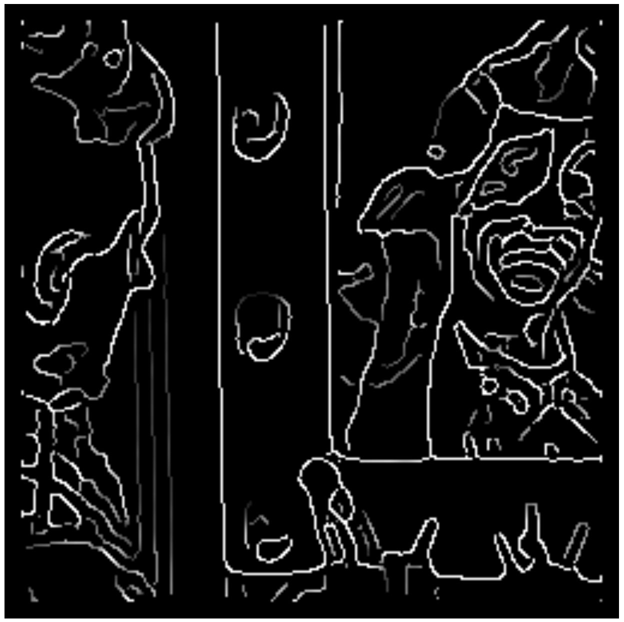
Effect of σ (Gaussian kernel spread/size)



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

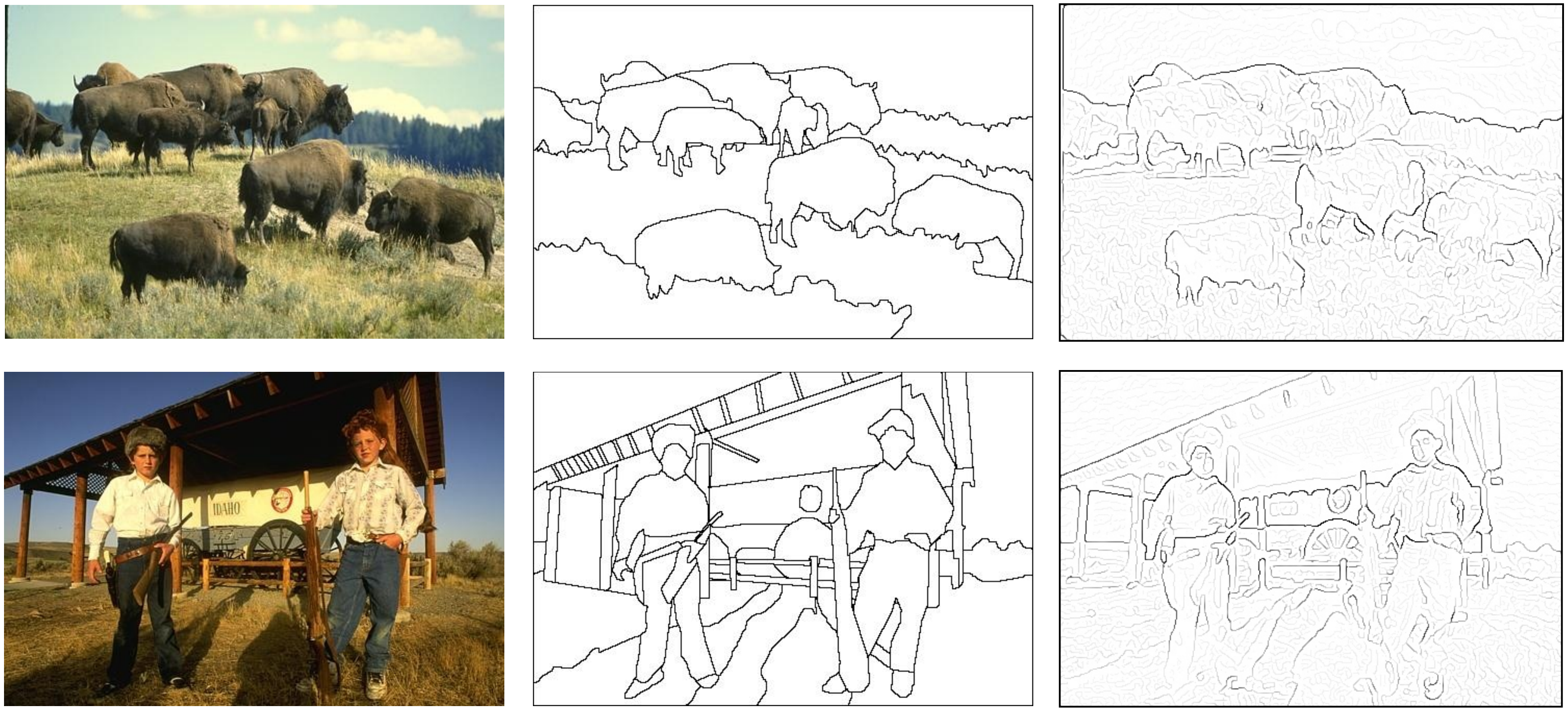
Source: S. Seitz

Take-home reading

- Szeliski Chapter 4.2 Edges
- A very good introduction of Canny edge detection:
- http://www.classes.cs.uchicago.edu/archive/2005/fall/35040-1/edges.pdf
- A youtube video is also helpful:
- https://www.youtube.com/watch?v=-Z3kr26Eci4

Where do humans see boundaries?

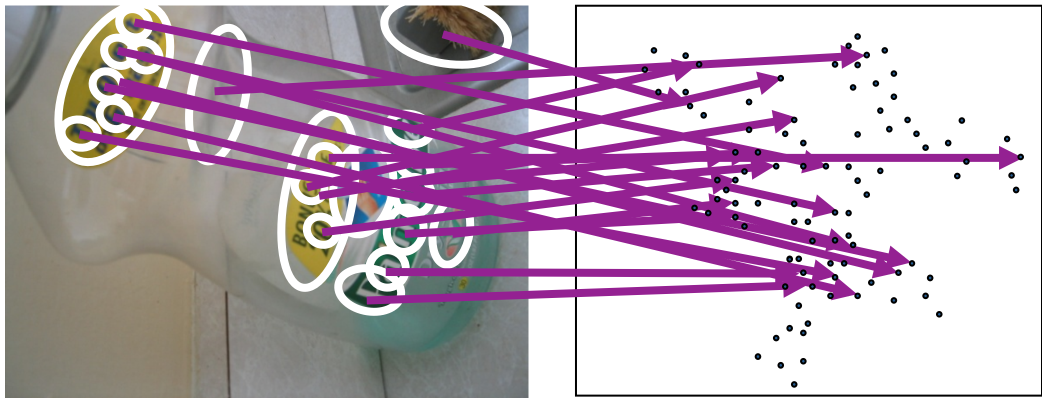
image | human segmentation | gradient magnitude



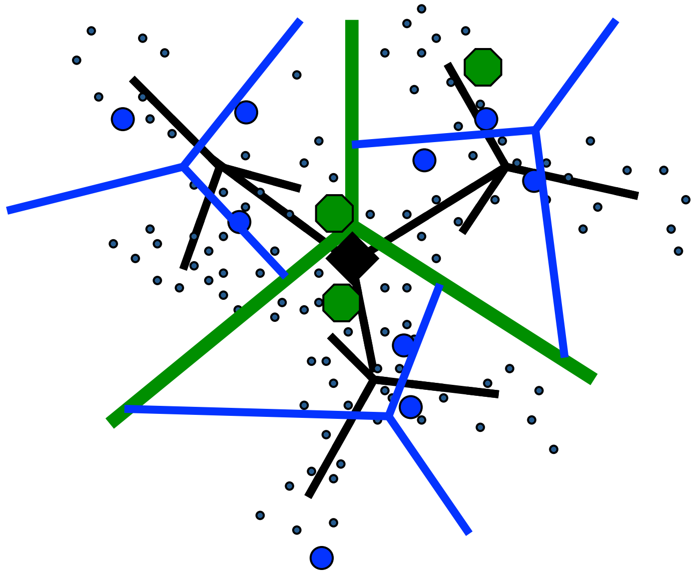
- Berkeley segmentation database:
  http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

Building Visual Dictionaries

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

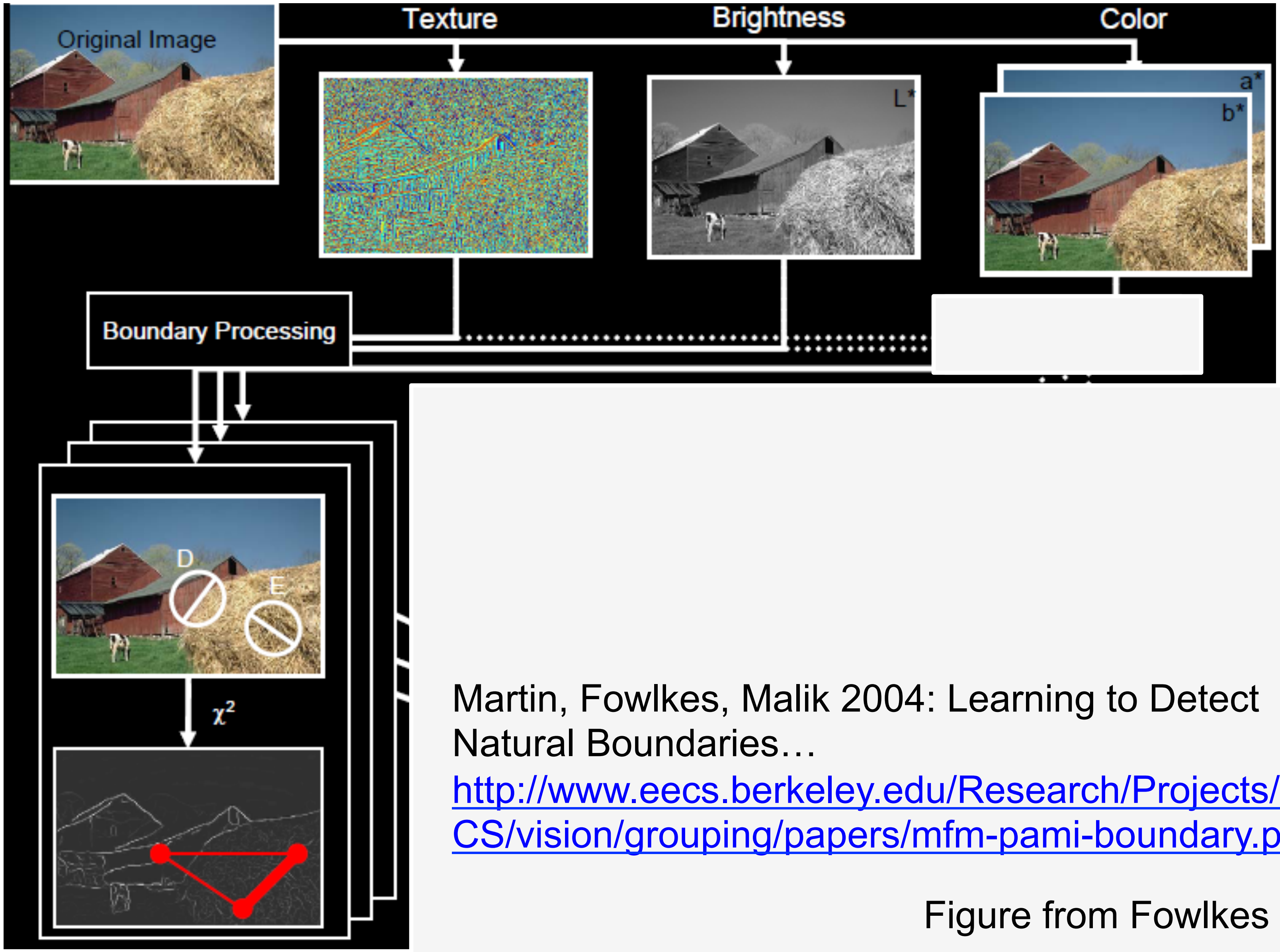


2. Cluster the patches
   – Cluster centers are the dictionary



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

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

pB Boundary Detector

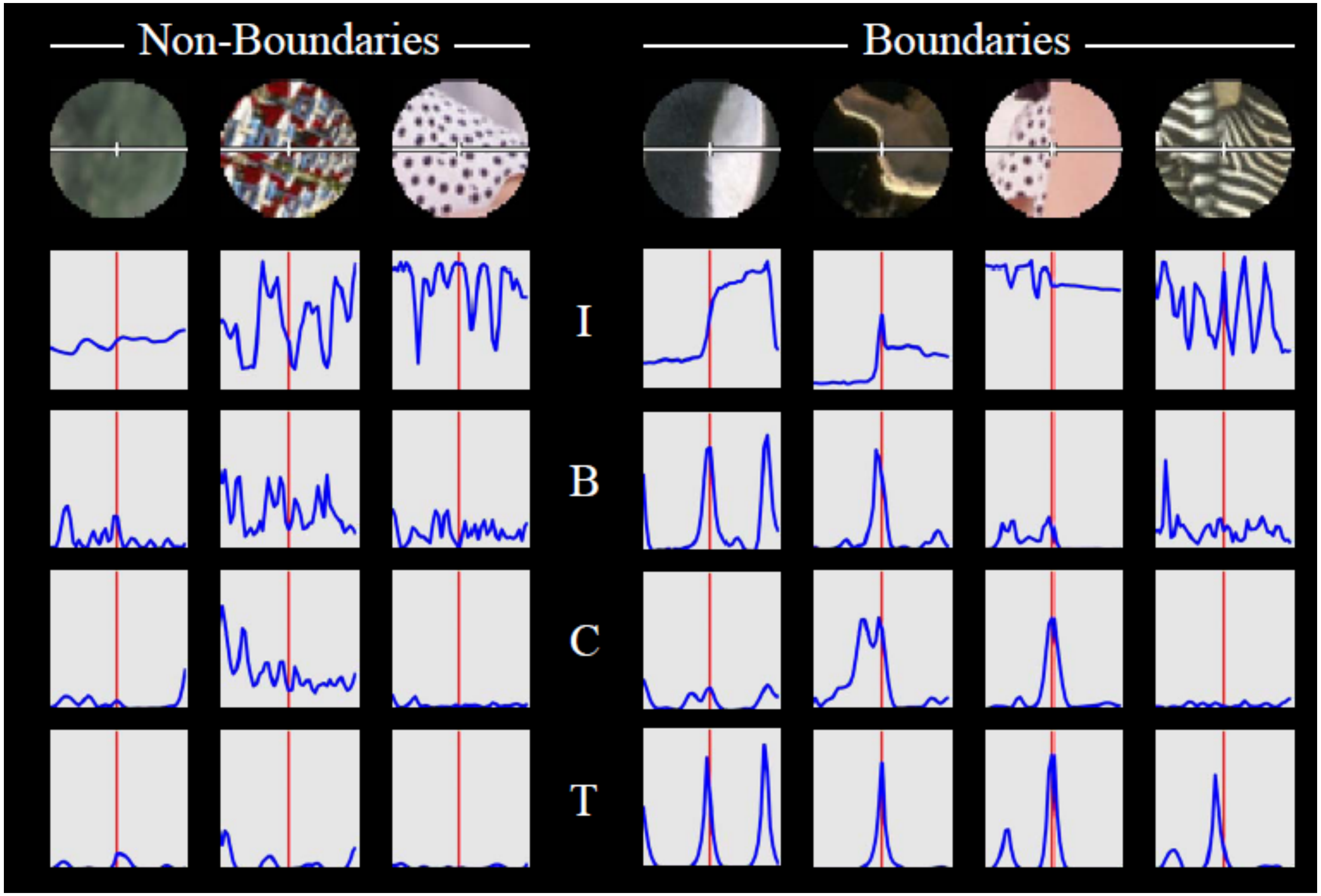


Figure from Fowlkes

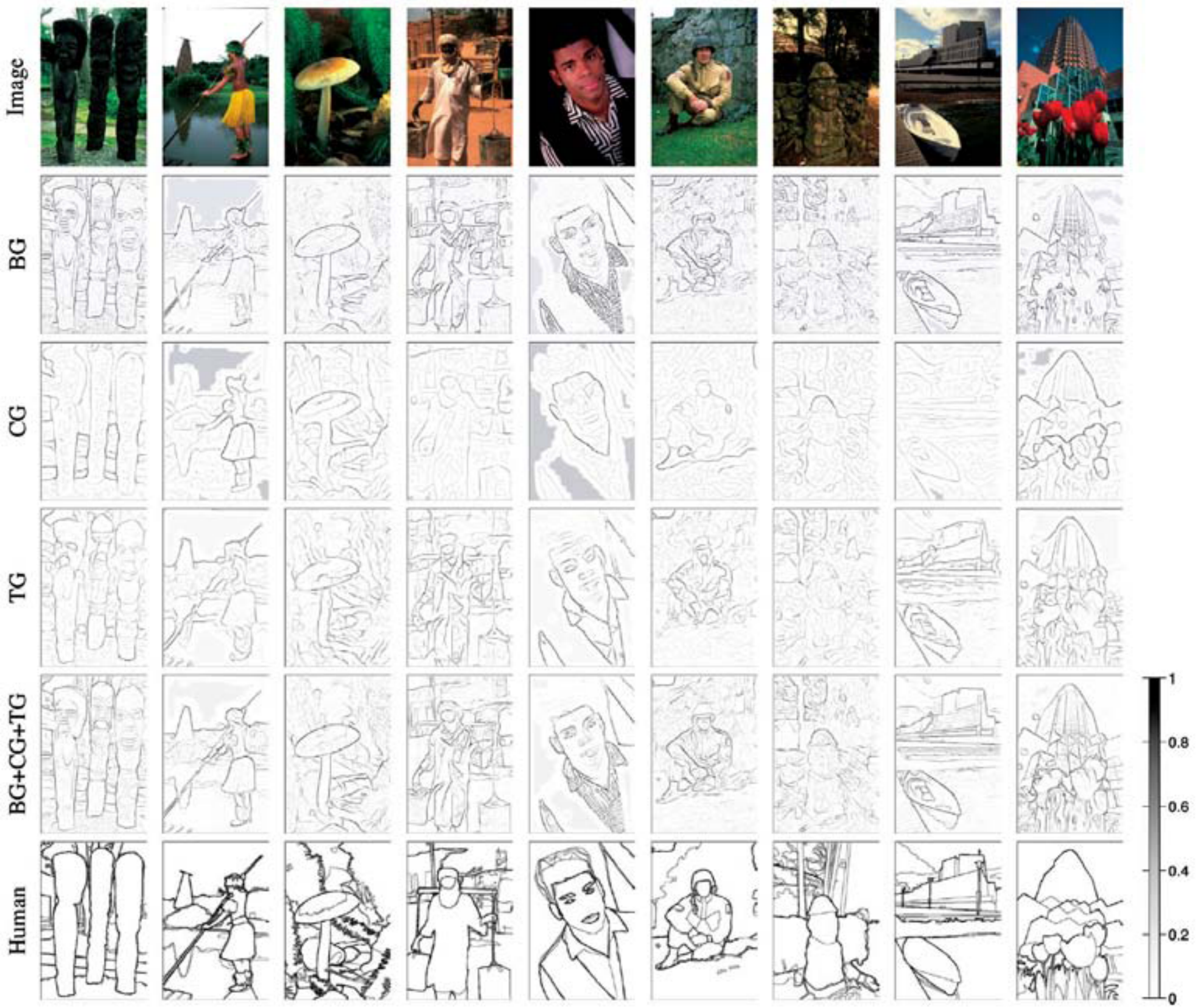
Brightness

Color

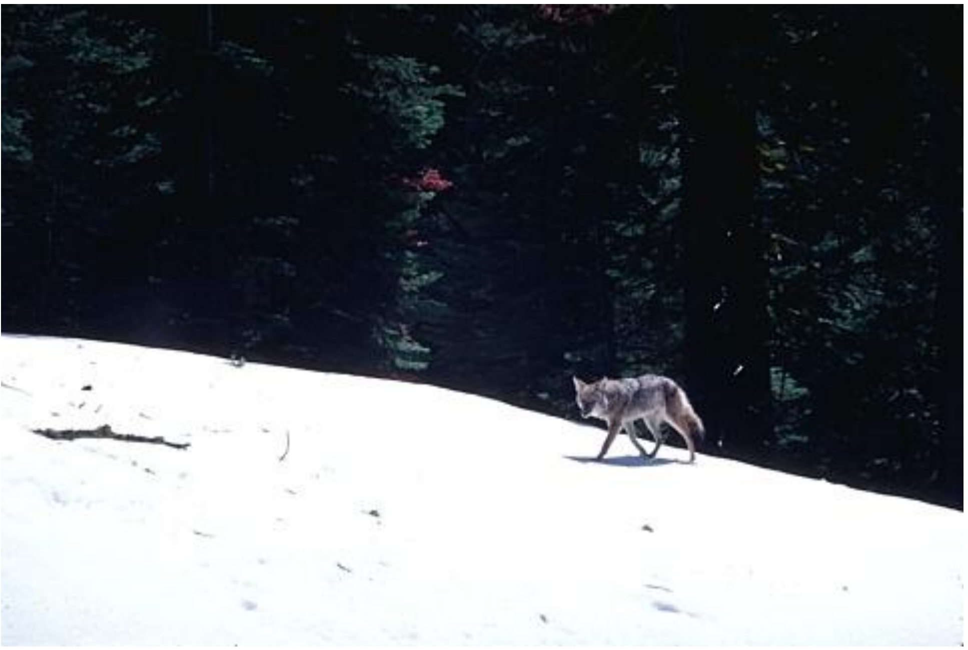
Texture

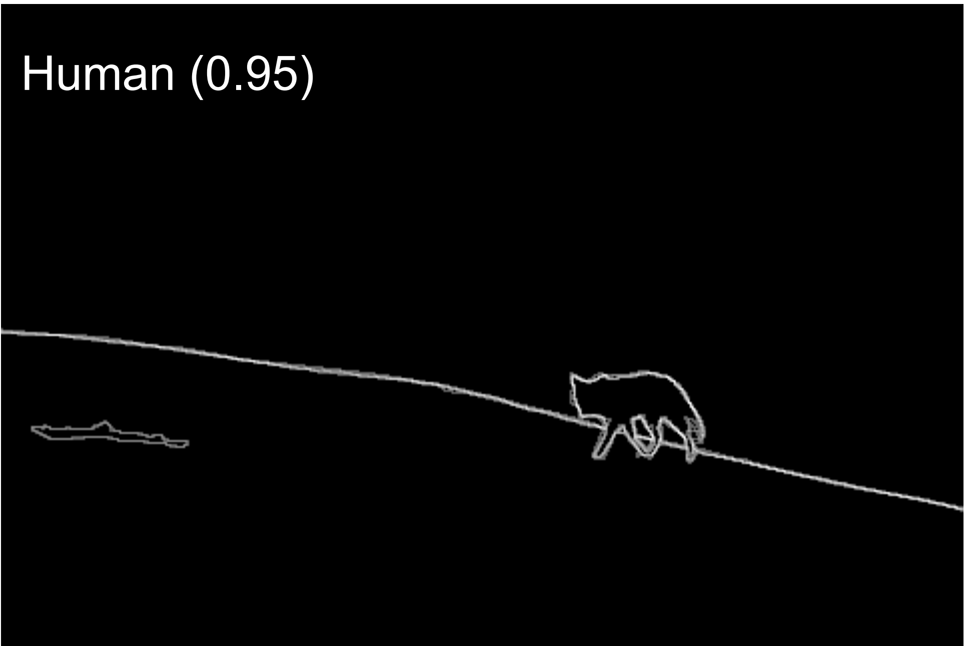
Combined

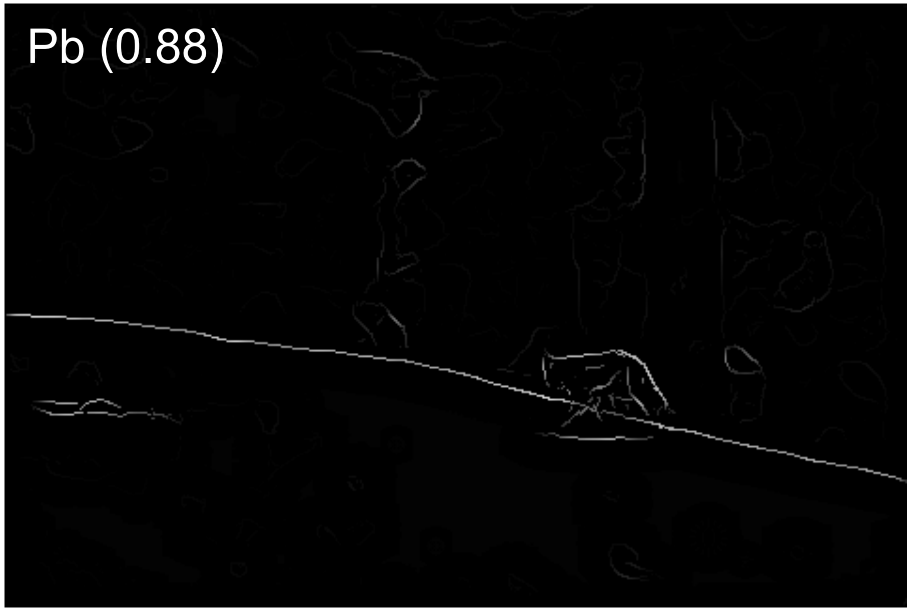
Human



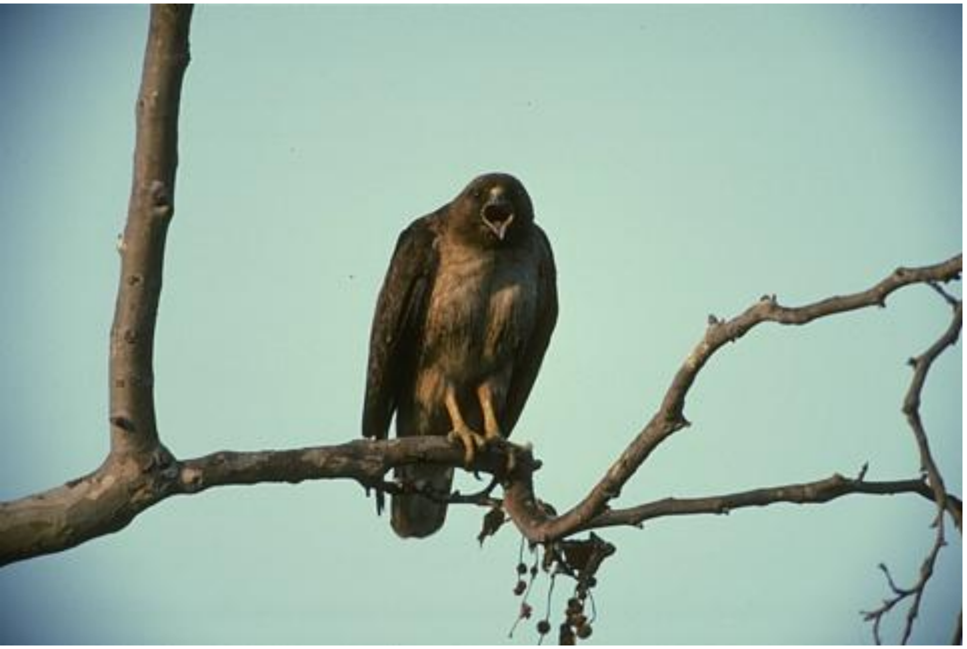
Results

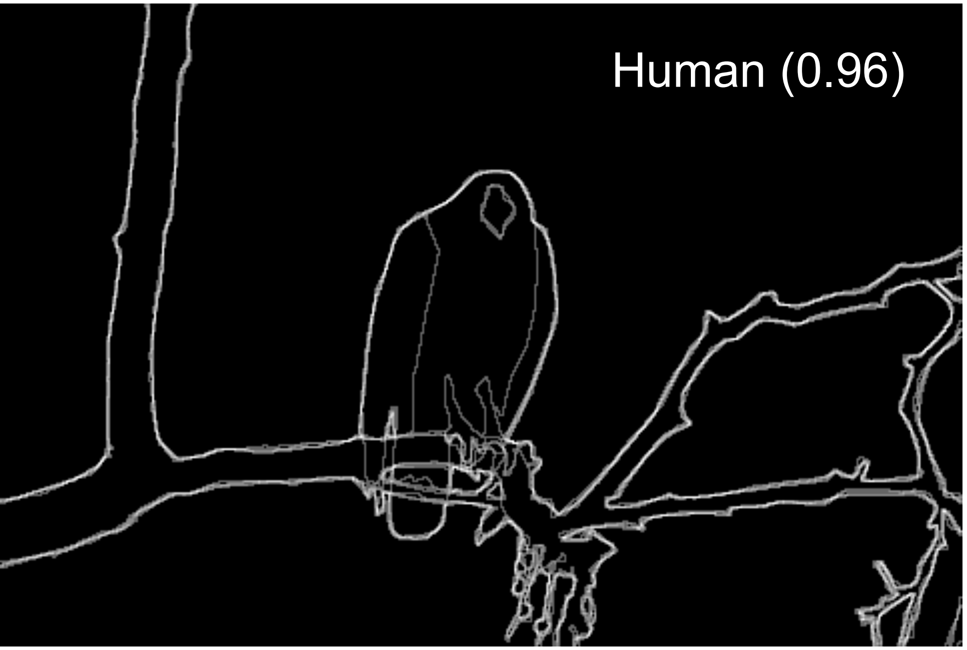


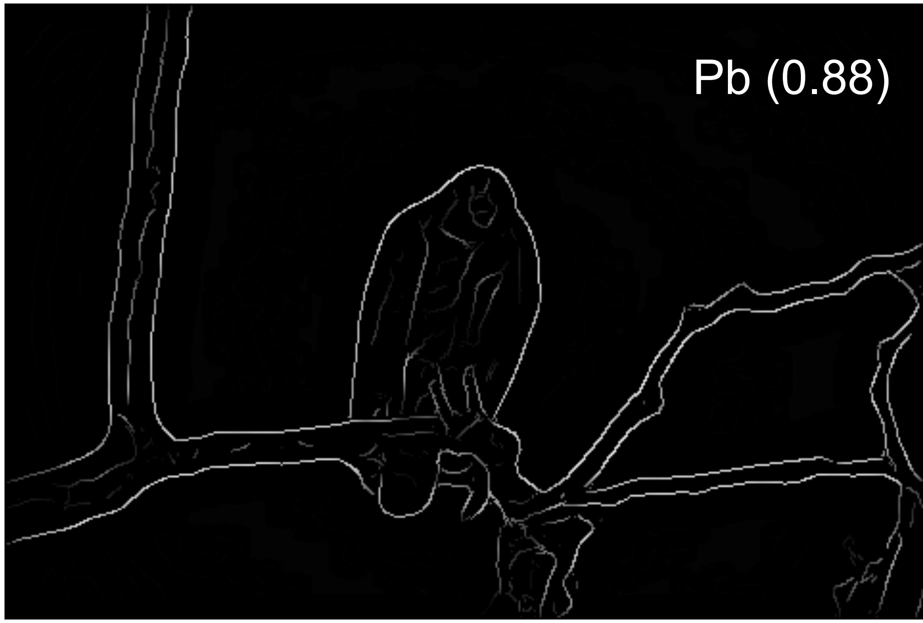


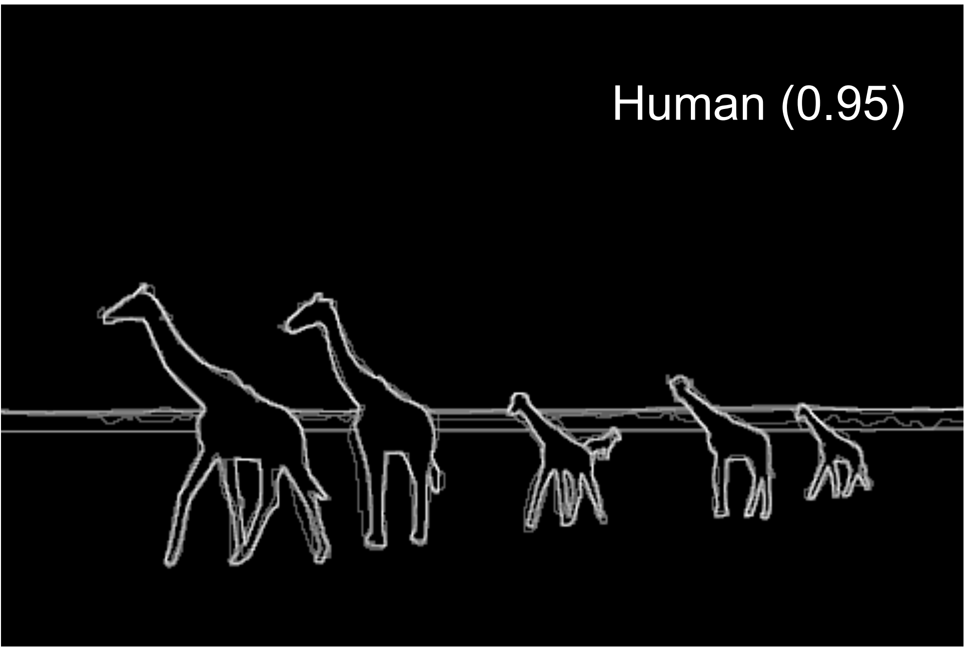


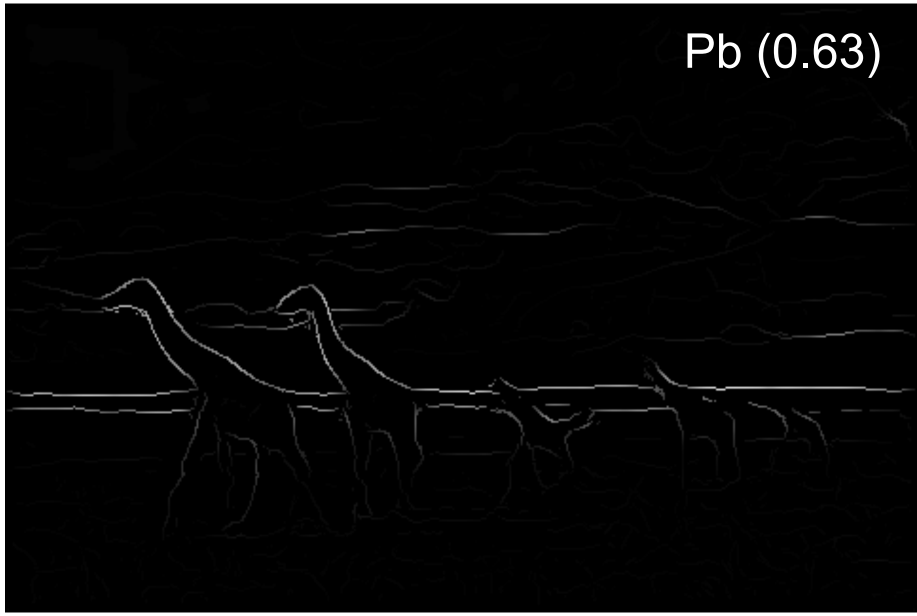
Results

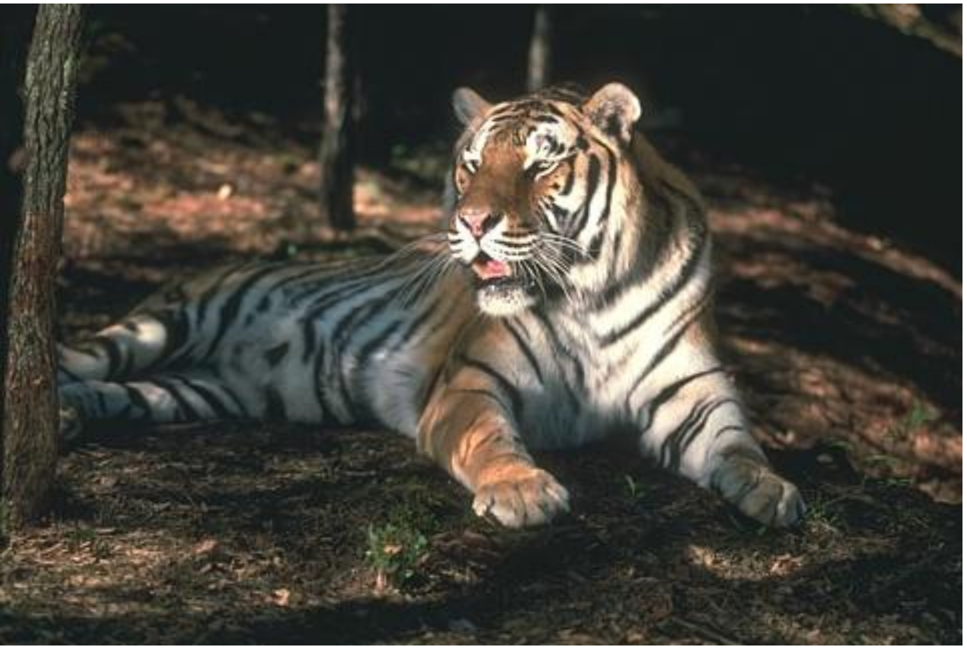


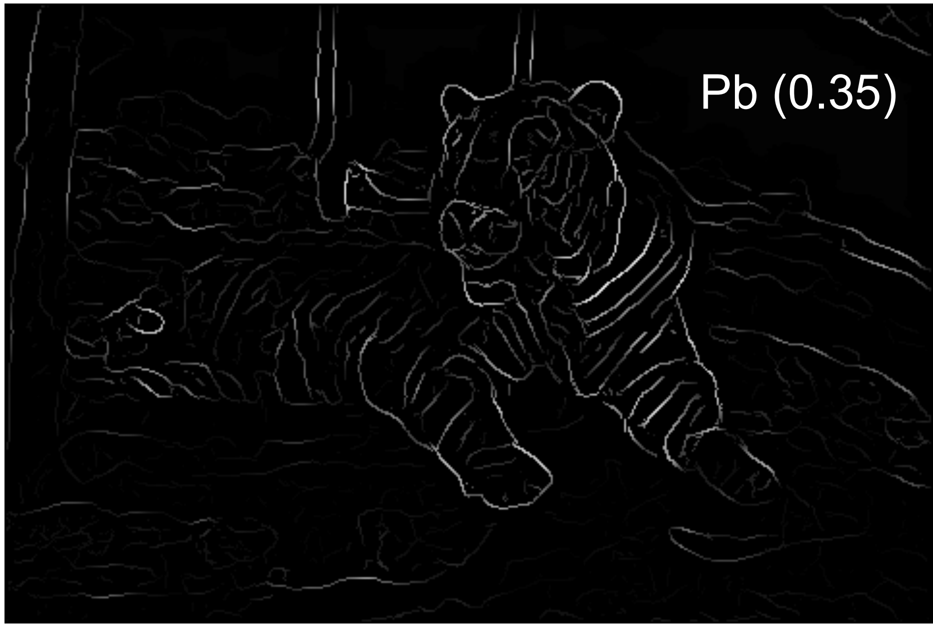


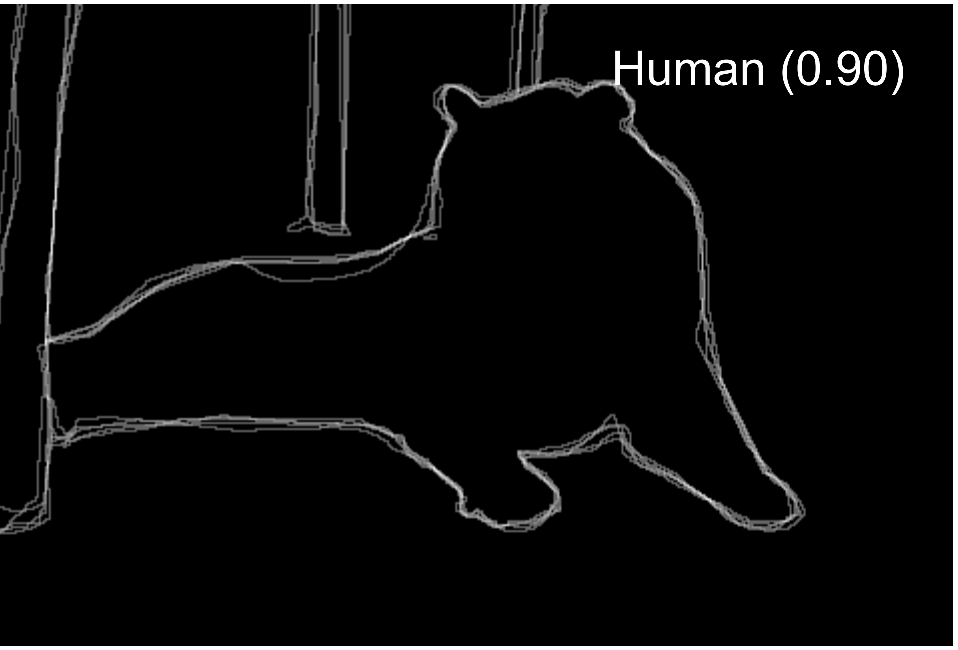



Human (0.95)


Pb (0.63)







For more:
http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/bench/html/108082-

Global pB boundary detector

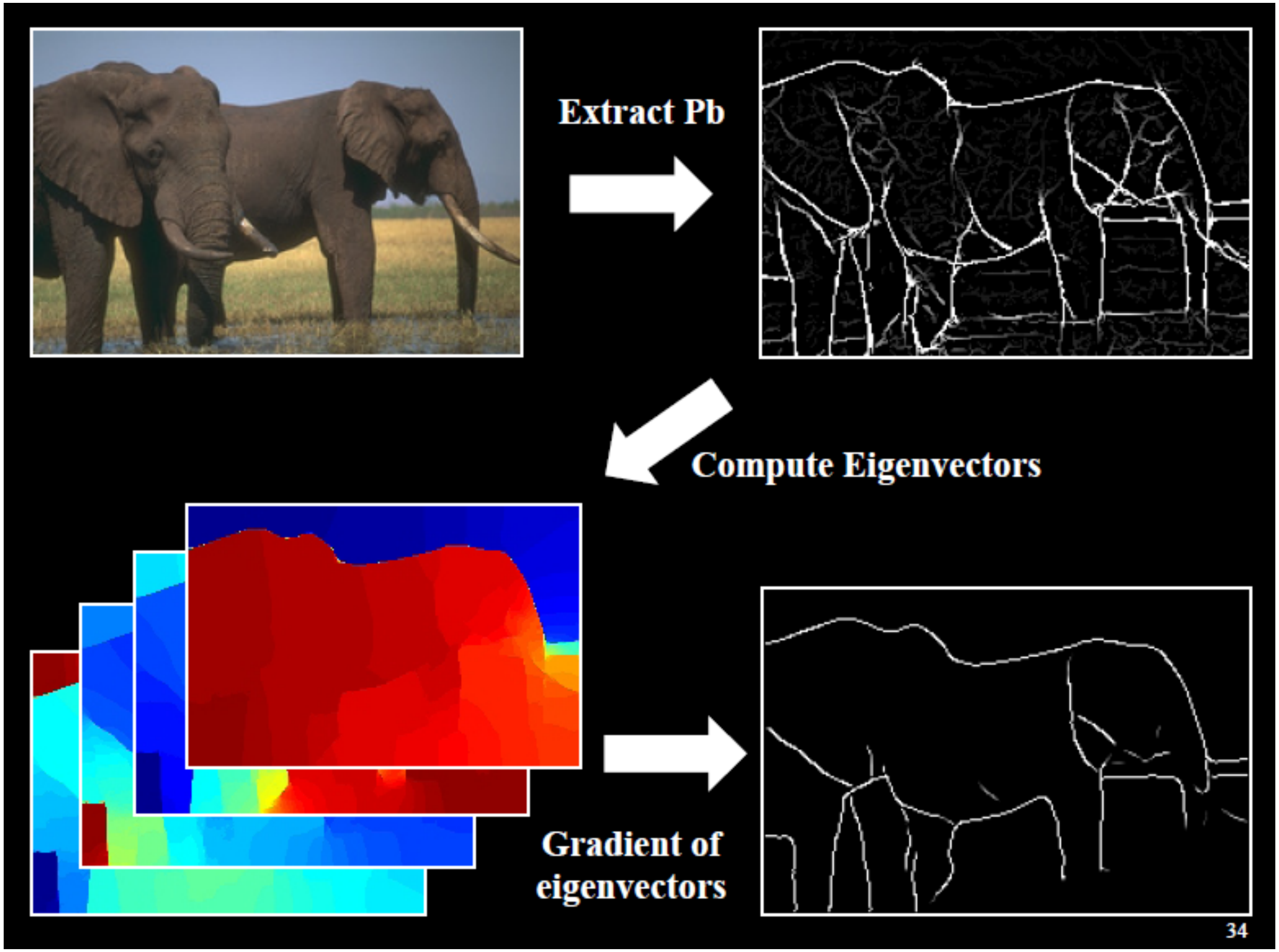
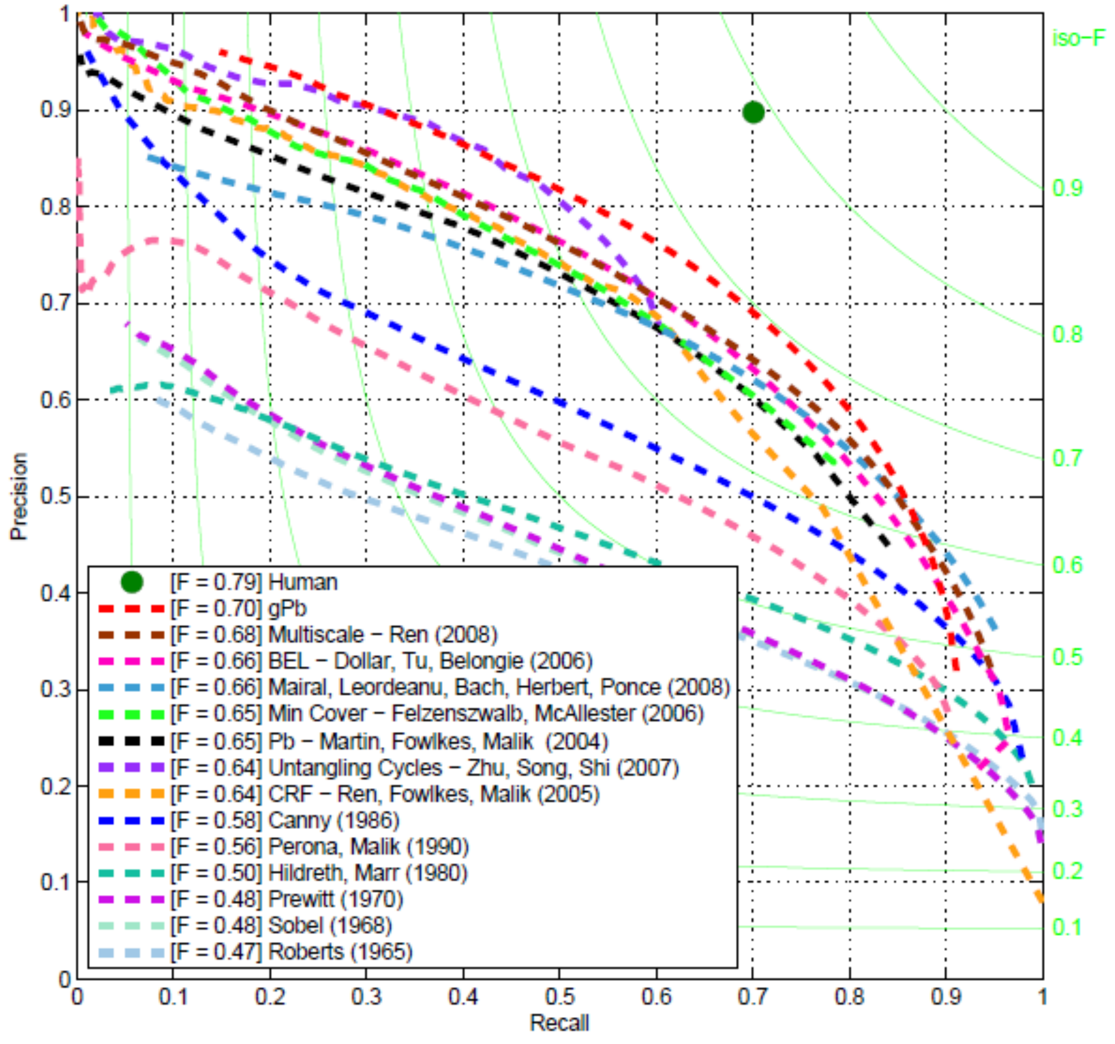


Figure from Fowlkes

45 years of boundary detection

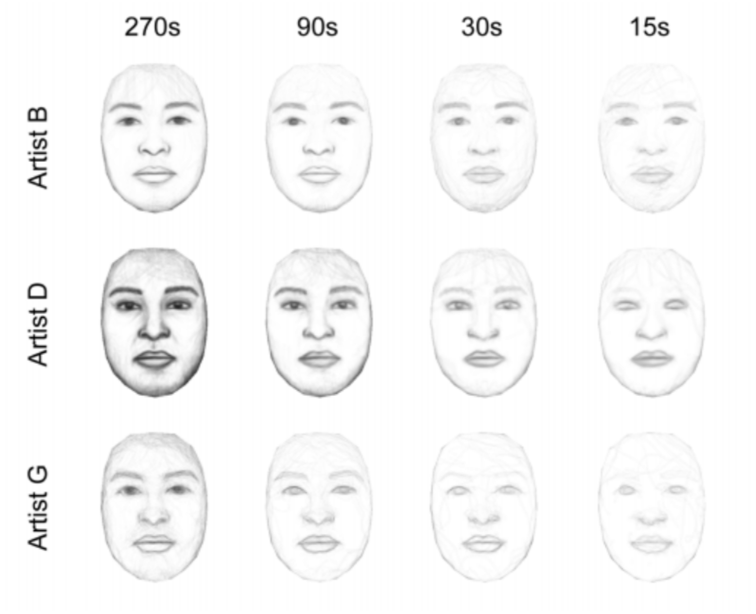


Source: Arbelaez, Maire, Fowlkes, and Malik. TPAMI 2011 (pdf)

State of edge detection

- Local edge detection works well
  - But many false positives from illumination and texture edges
- Some methods to take into account longer contours, but could probably do better
- Few methods that actually “learn” from data. Your project 5, Sketch Tokens, will do so.
- Poor use of object and high-level information

Style and abstraction in portrait sketching, Berger et al. SIGGRAPH 2013



- Learn from artist’s strokes so that edges are more likely in certain parts of the face.

Questions