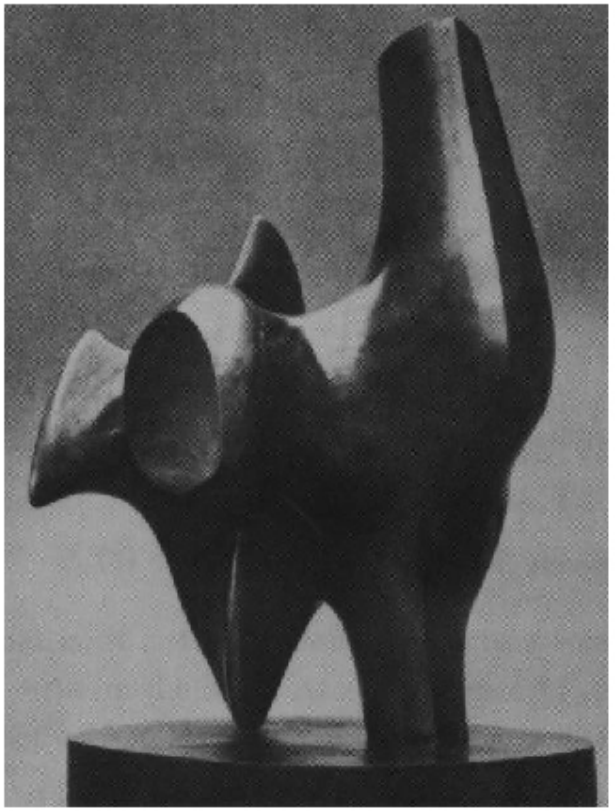
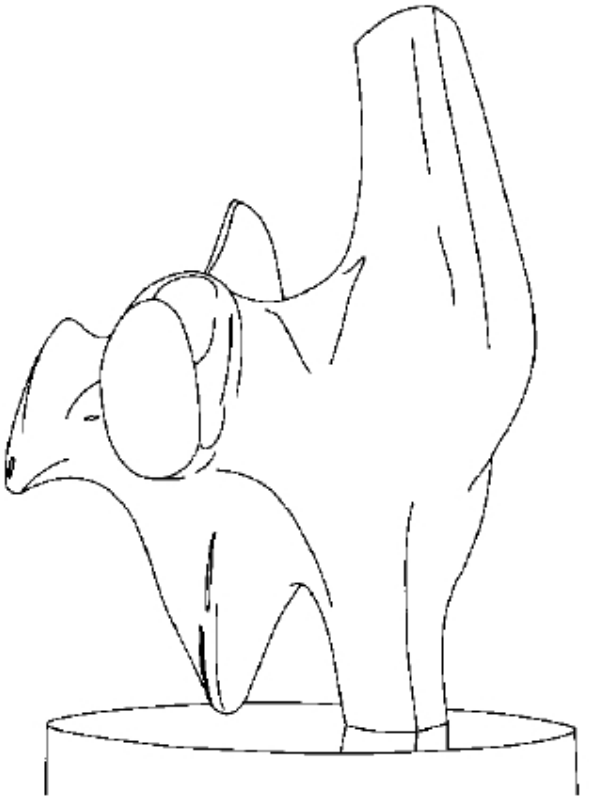
Edge Detection

- Slide content from Derek Hoiem, Cornelia Fermüller, Steve Seitz

Edge detection

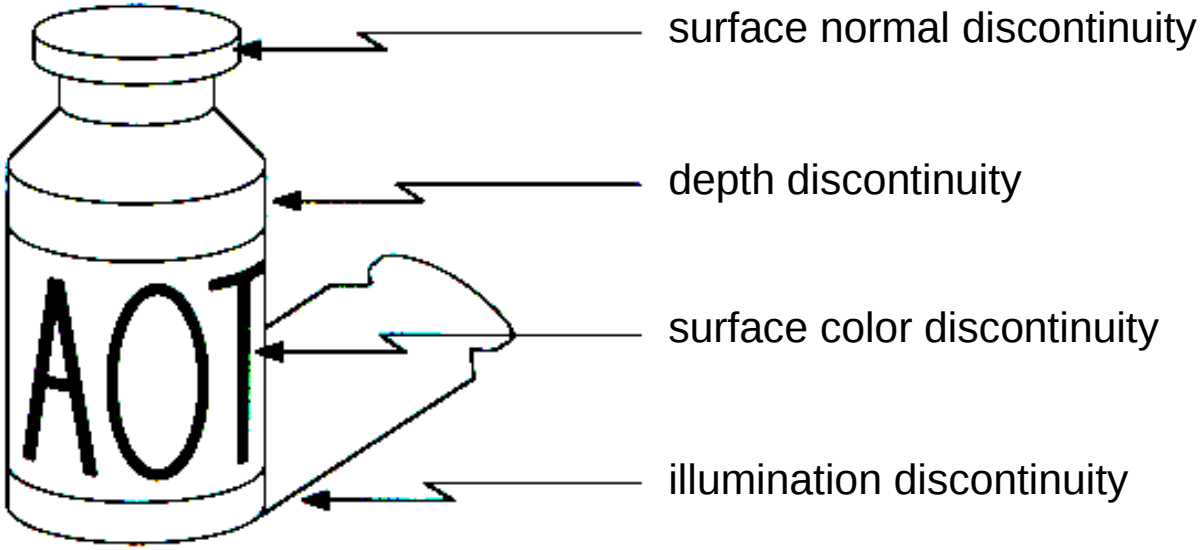




Convert a 2D image into a set of curves

- Extracts salient features of the scene
- More compact than pixels

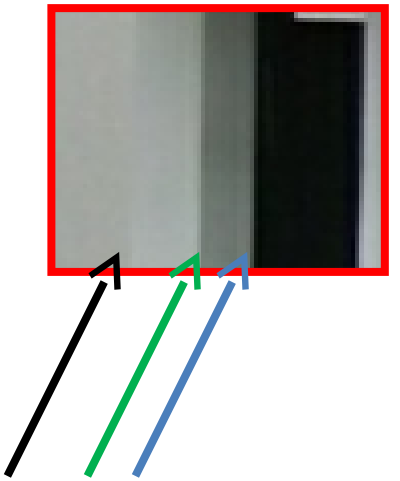
Origin of Edges



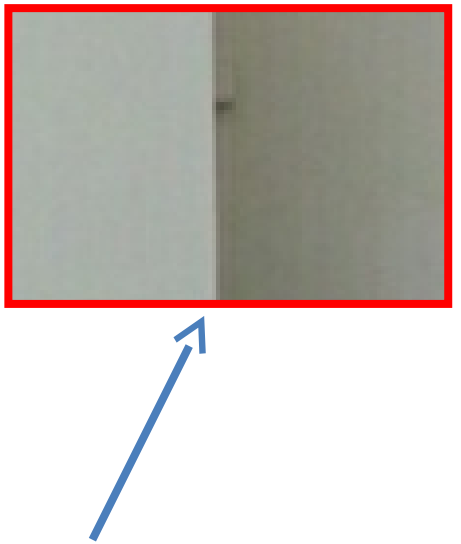
Edges are caused by a variety of factors

Closeup of edges

Closeup of edges

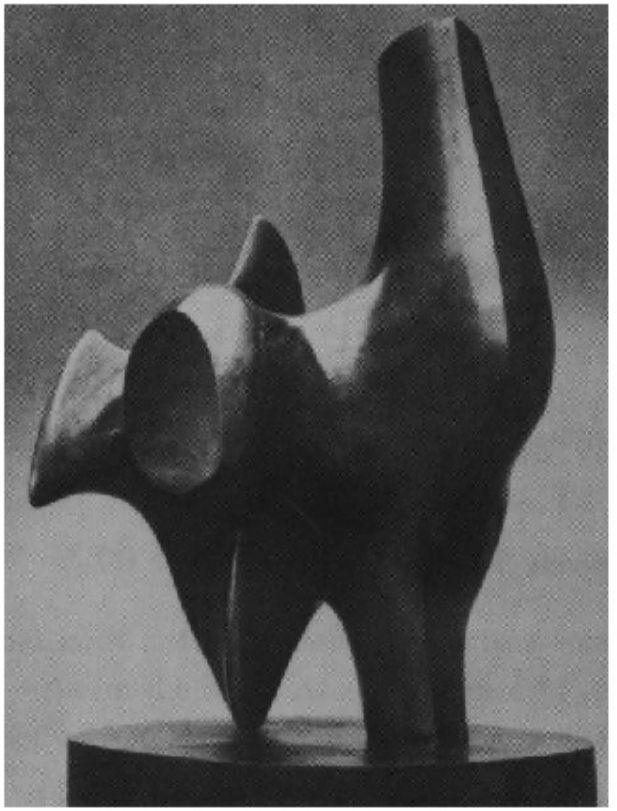


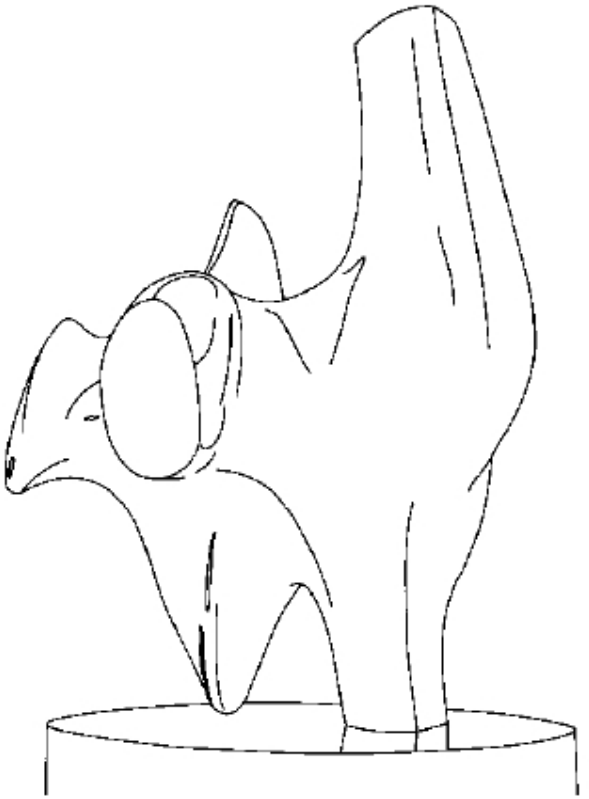
Closeup of edges



Closeup of edges

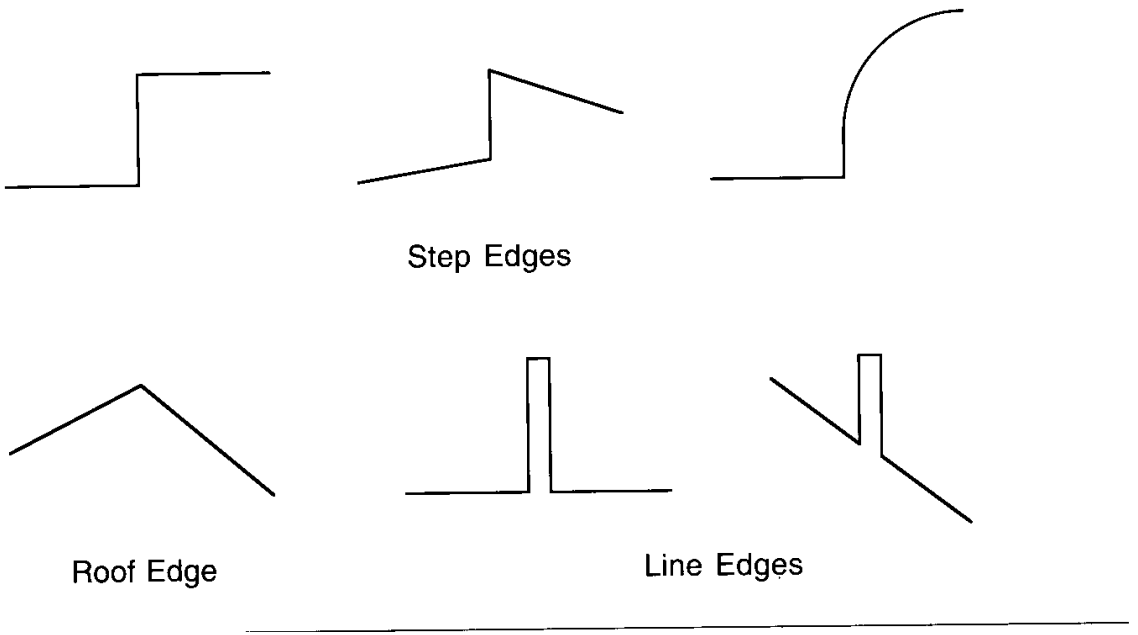
Edge detection





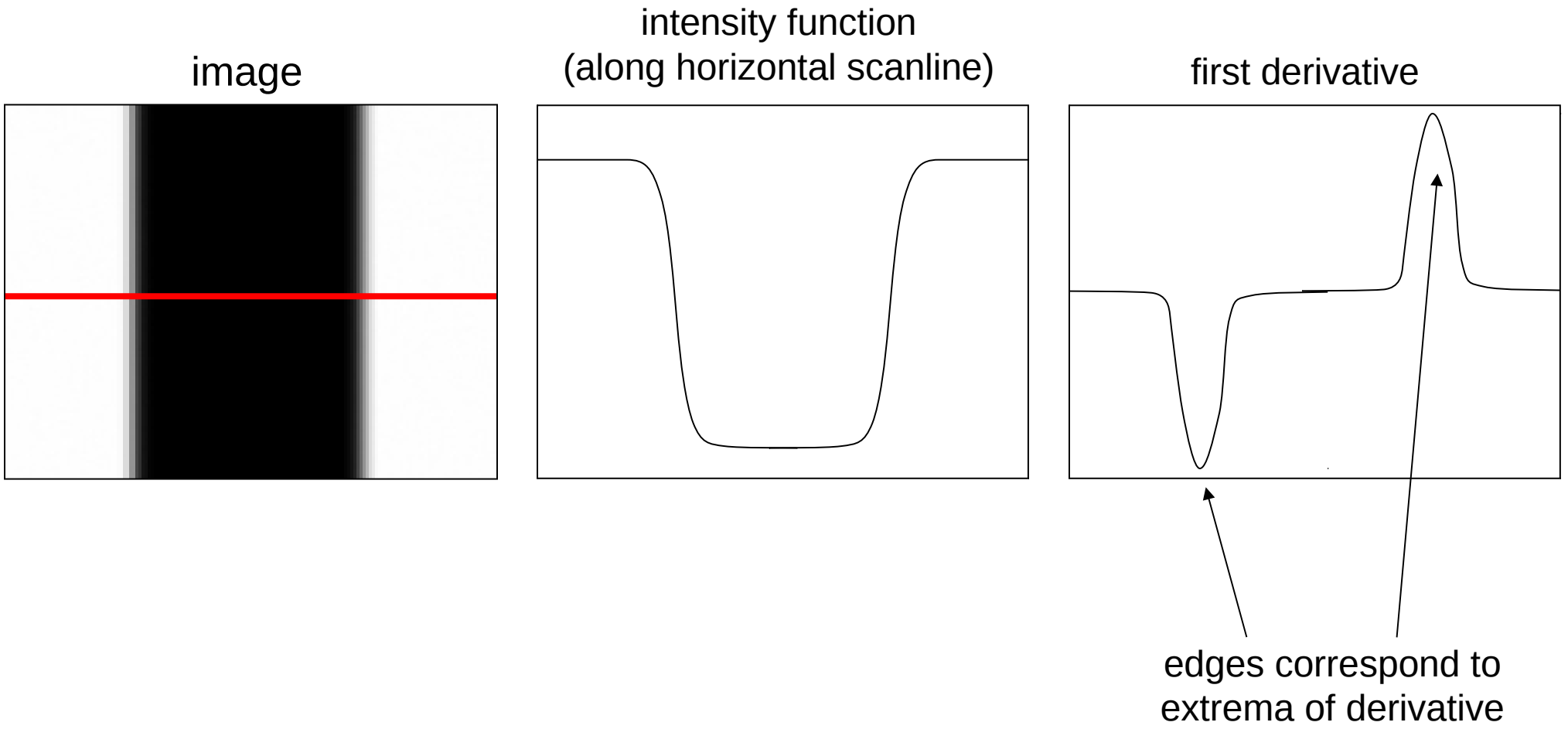
How can you tell that a pixel is on an edge?

Profiles of image intensity edges

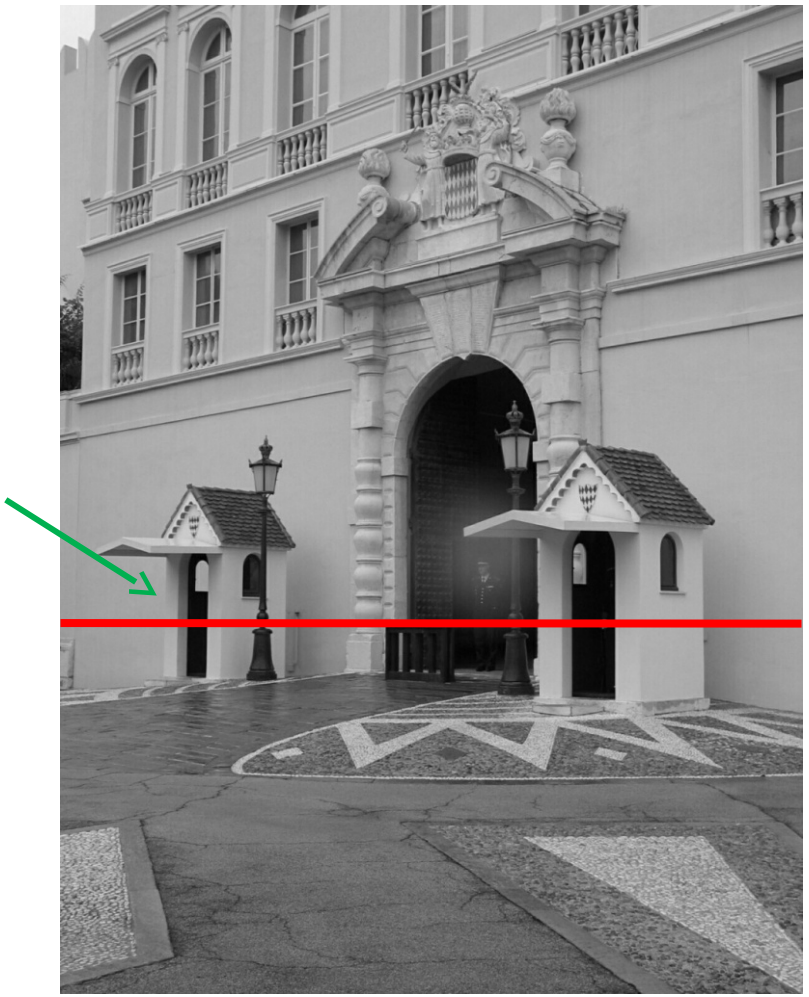


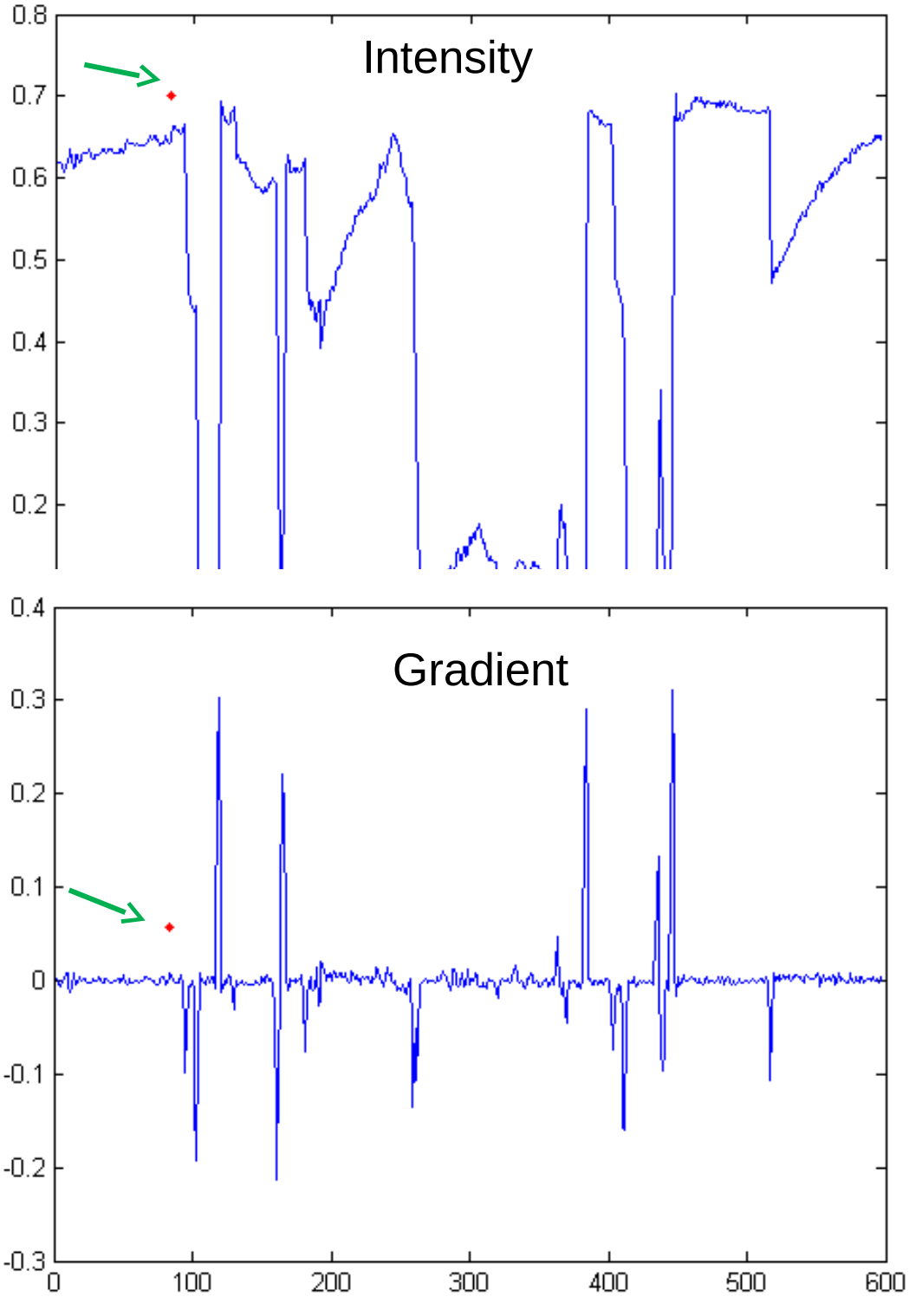
Characterizing edges

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

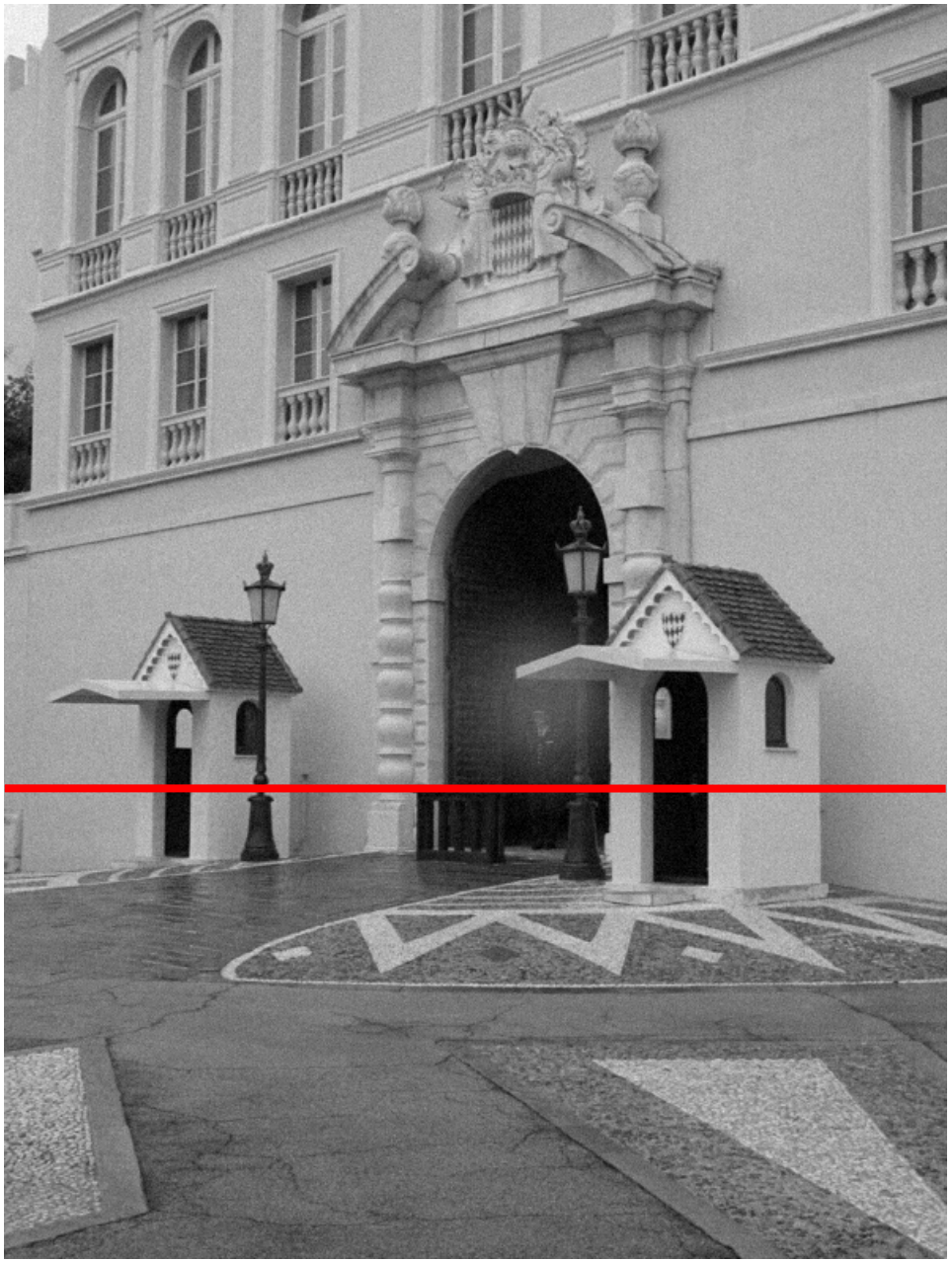


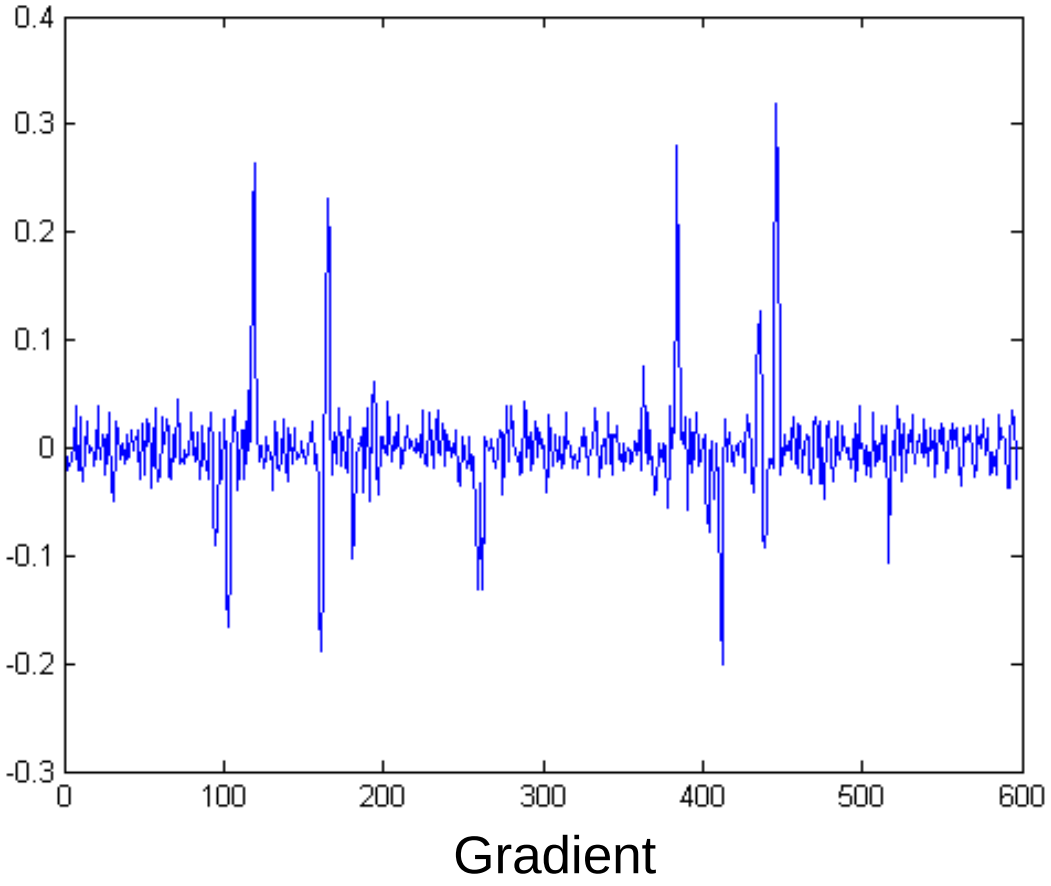
Intensity profile





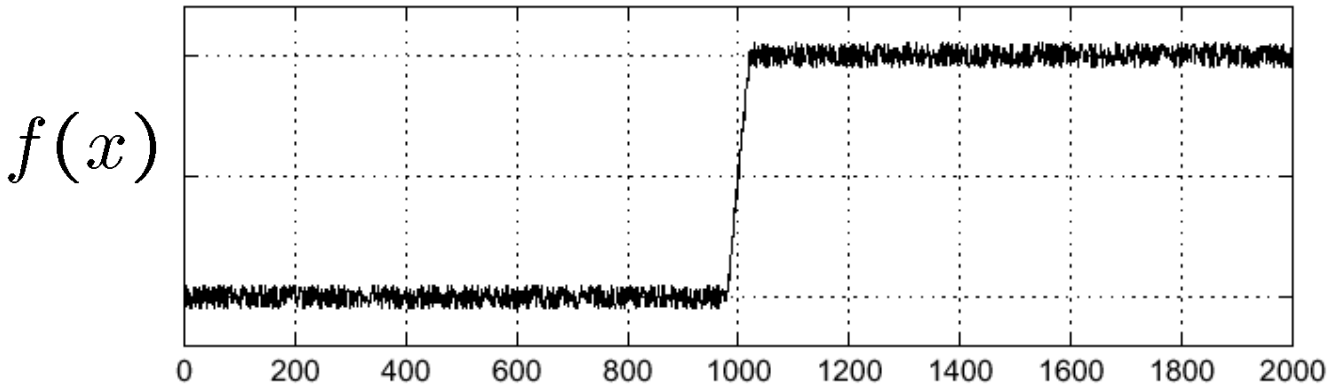
With a little Gaussian noise

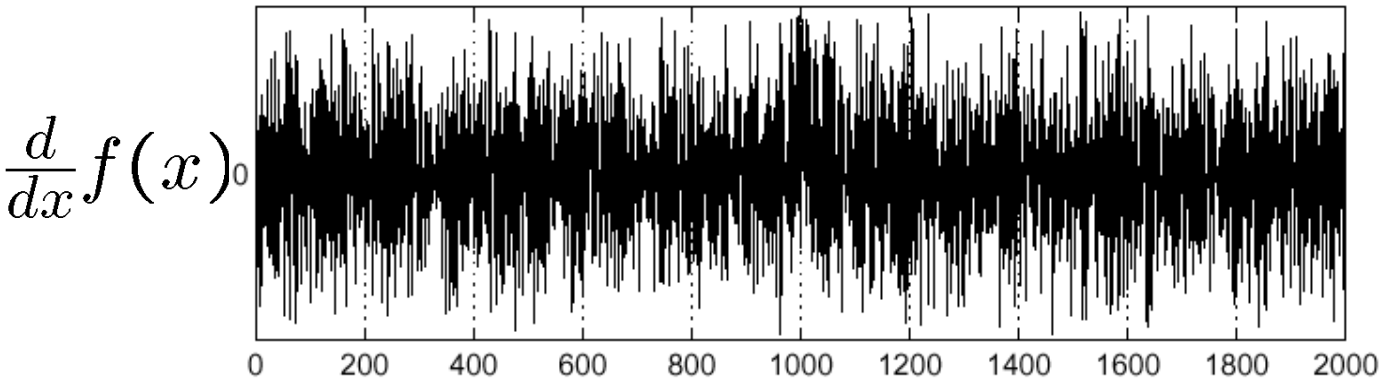




Effects of noise

- 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

Edge is Where Change Occurs

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

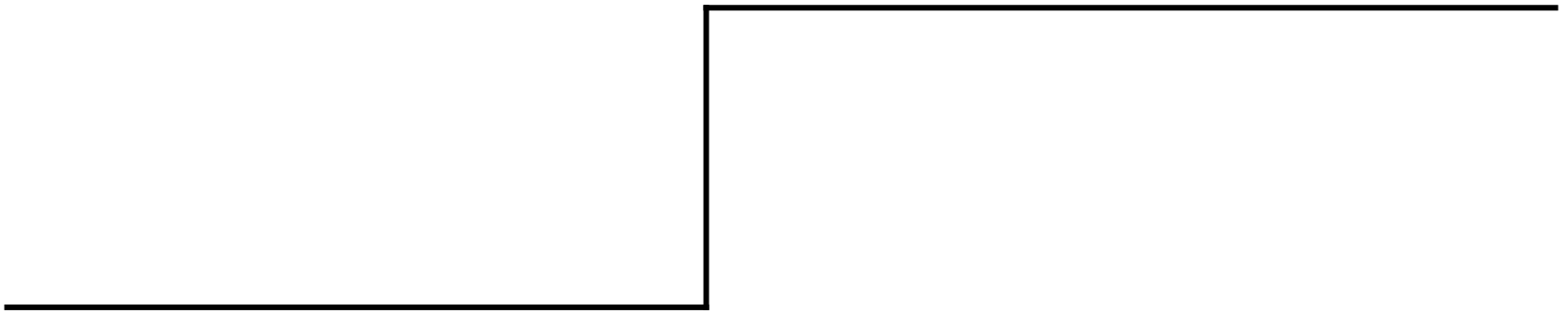
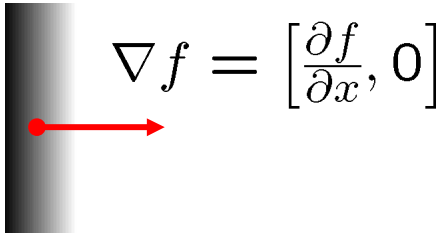


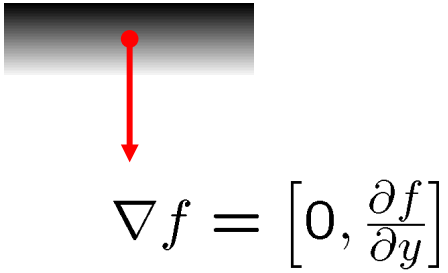
Image gradient

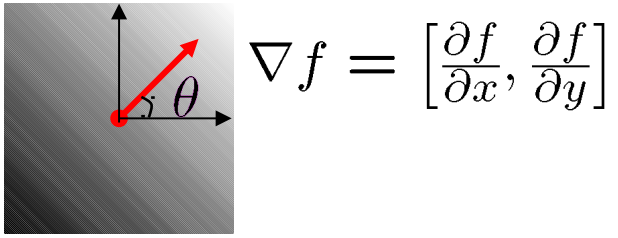
The gradient of an image:

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

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







The gradient direction 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

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

The discrete gradient

How can we differentiate a *digital* image f[x,y]?

- Option 1: reconstruct a continuous image, then take gradient
- Option 2: take discrete derivative (finite difference)

$$\frac{\partial f}{\partial x}[x, y] \approx f[x+1, y] - f[x, y]$$

The Sobel operator

Better approximations of the derivatives exist

- The *Sobel* operators below are very commonly used

$\frac{1}{8}$

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

s_x

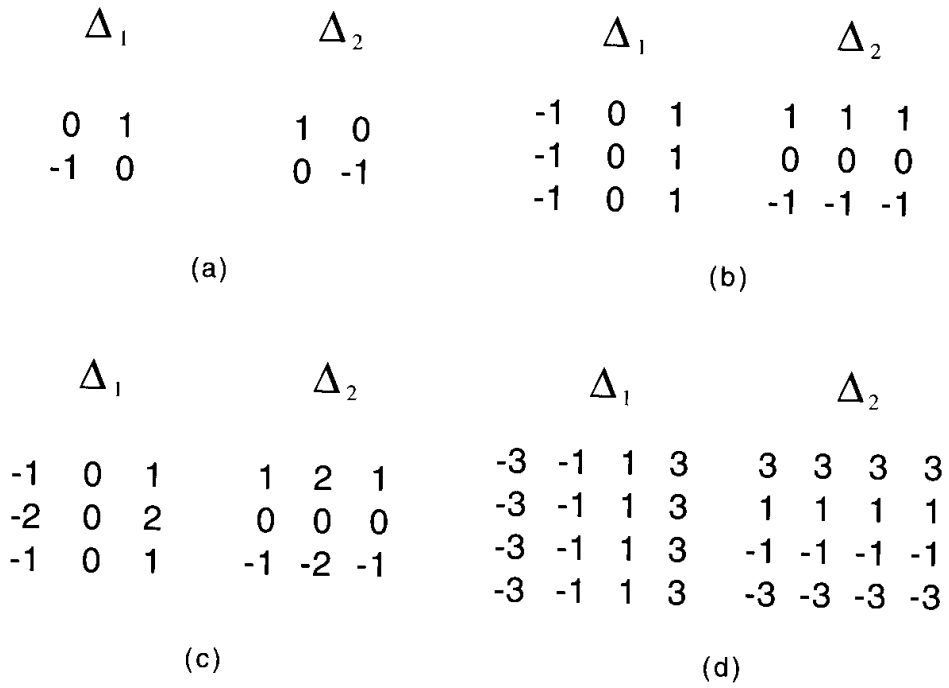
$\frac{1}{8}$

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

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 value, however

Gradient operators

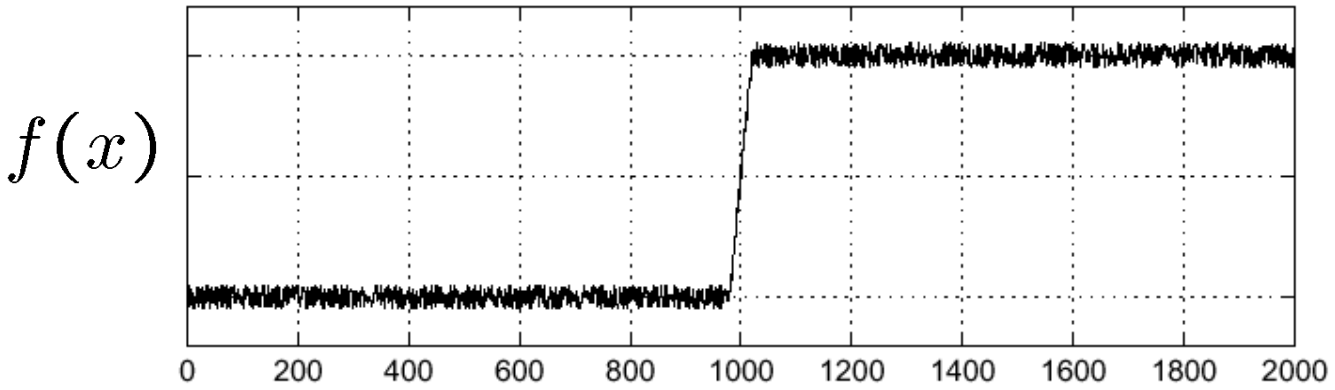


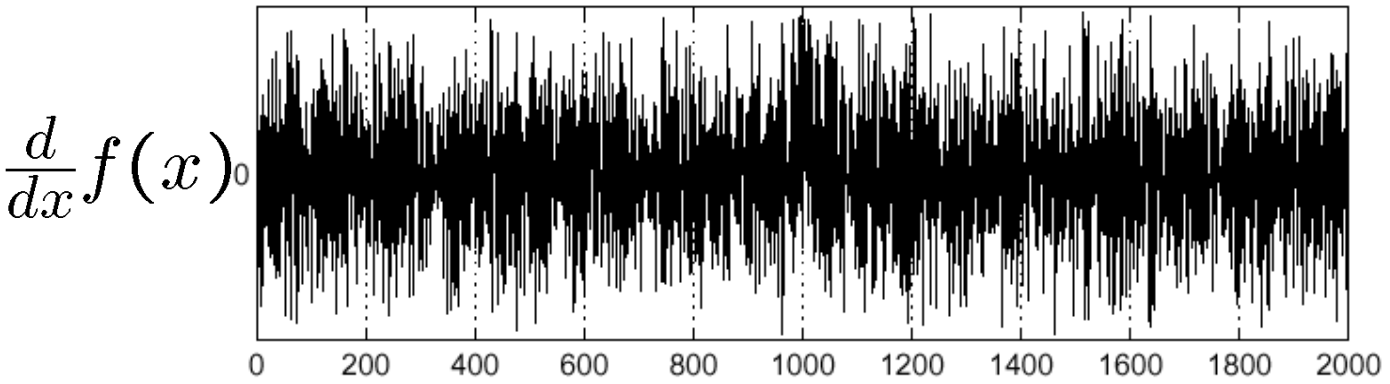
(a): Roberts' cross operator (b): 3x3 Prewitt operator
(c): Sobel operator (d) 4x4 Prewitt operator

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?

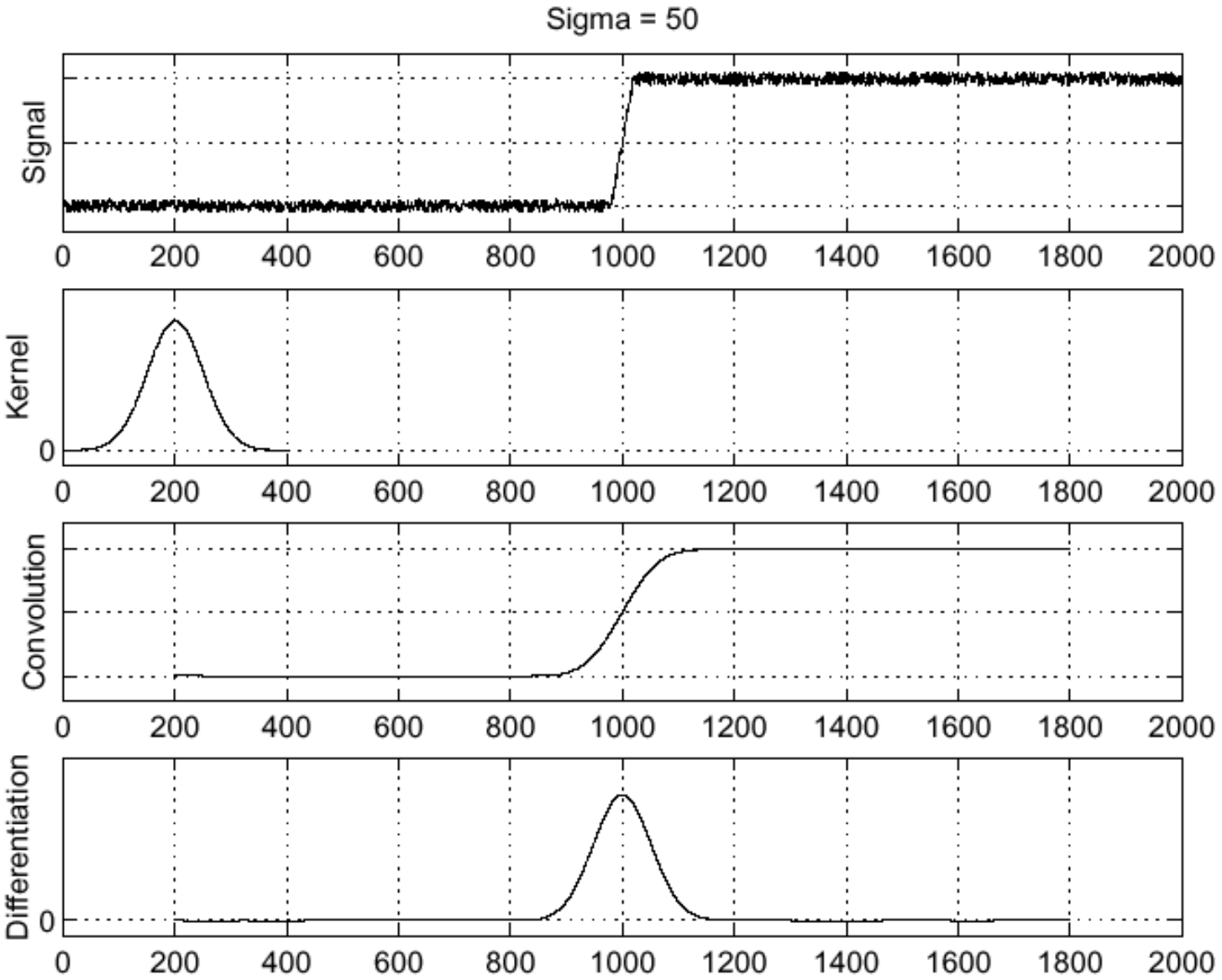
Solution: smooth first

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

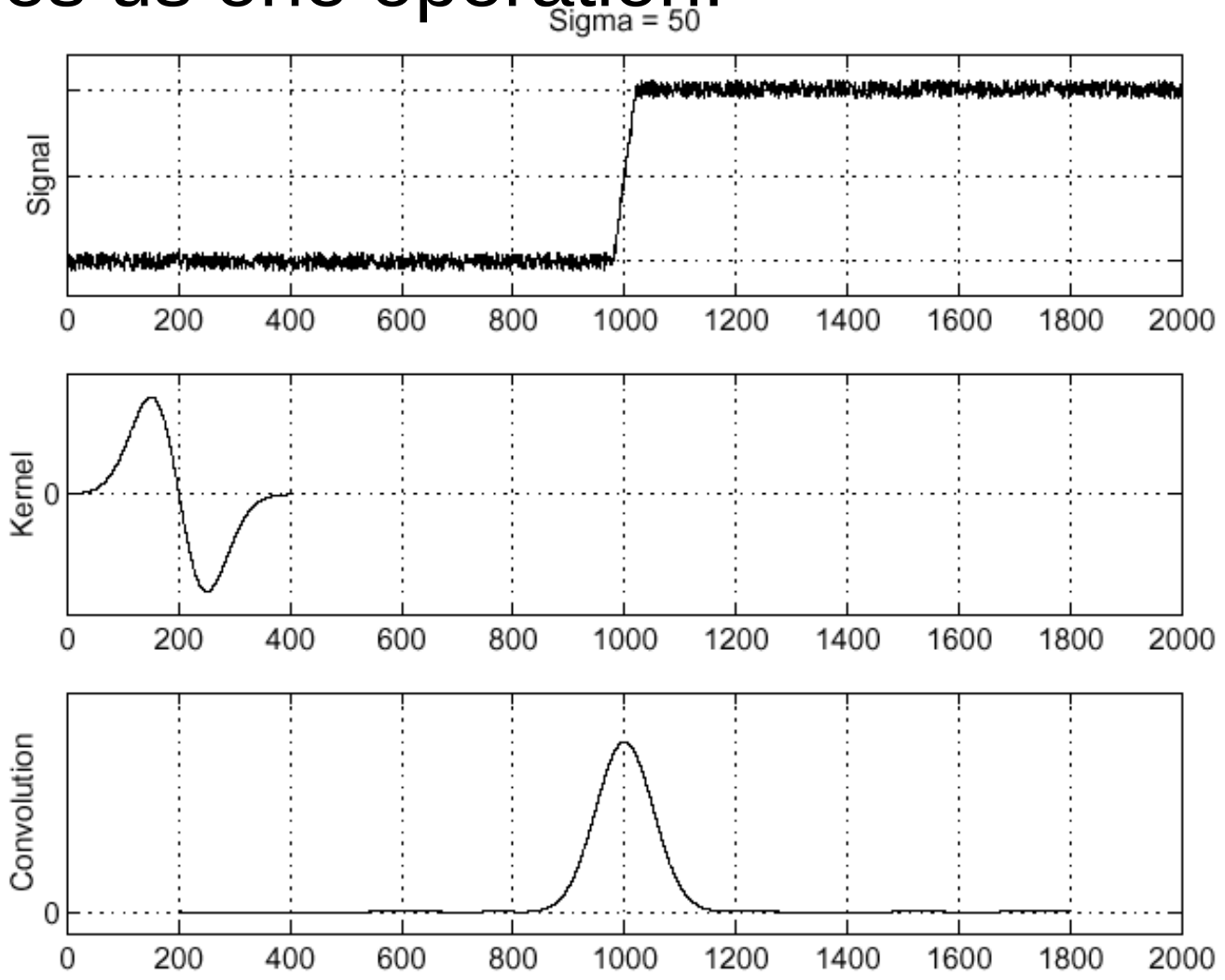
$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$$

This saves us one operation:

f

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

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



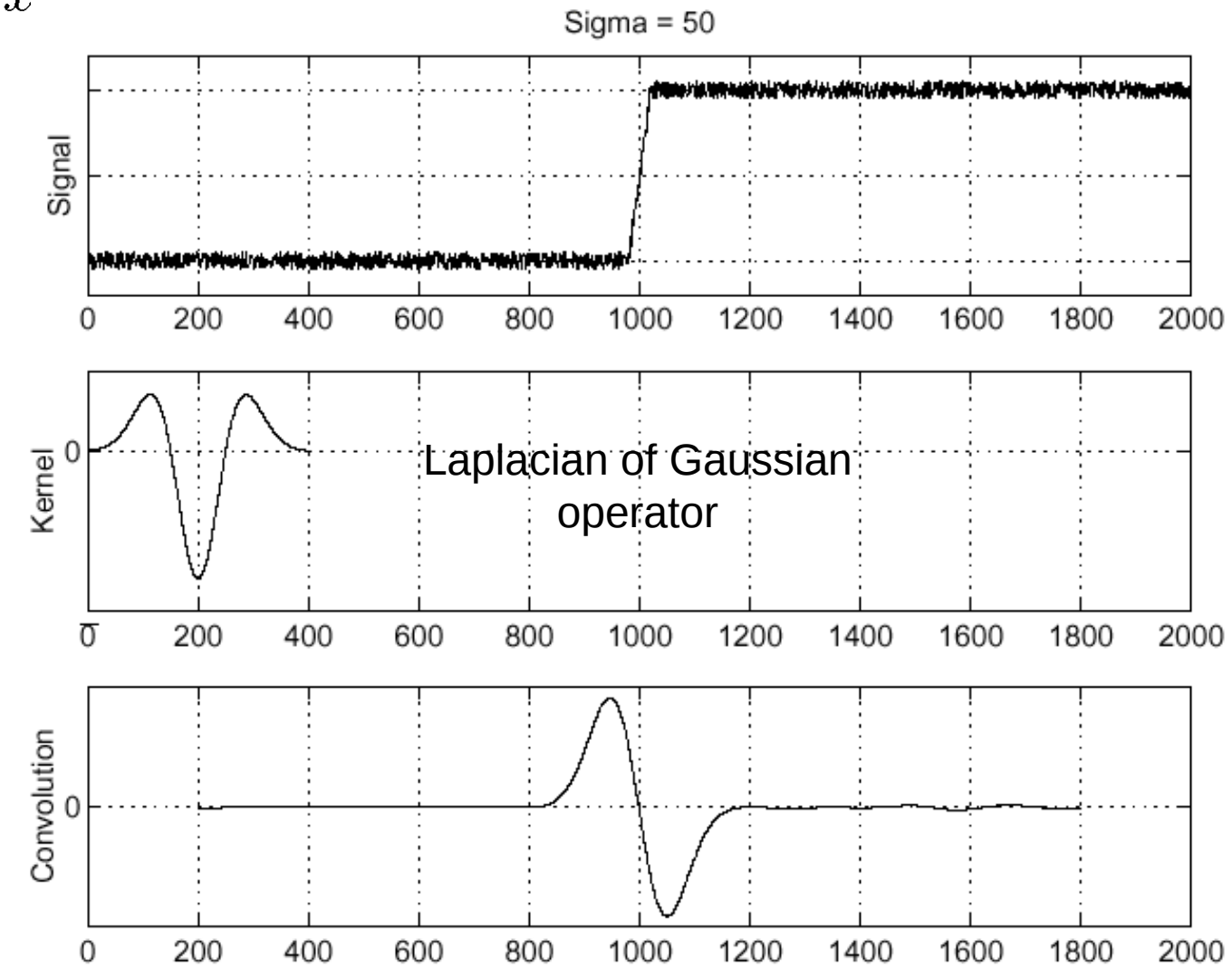
Laplacian of Gaussian

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

f

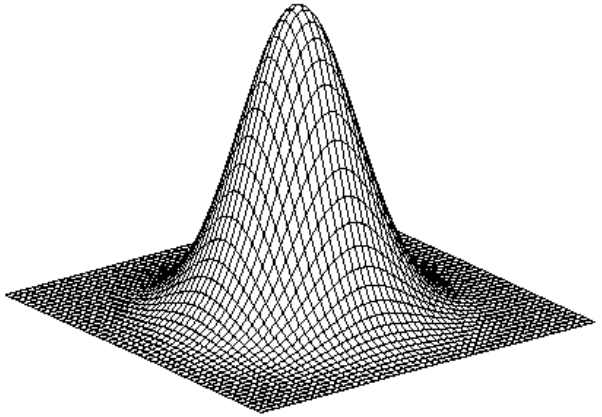
$\frac{\partial^2}{\partial x^2}h$

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



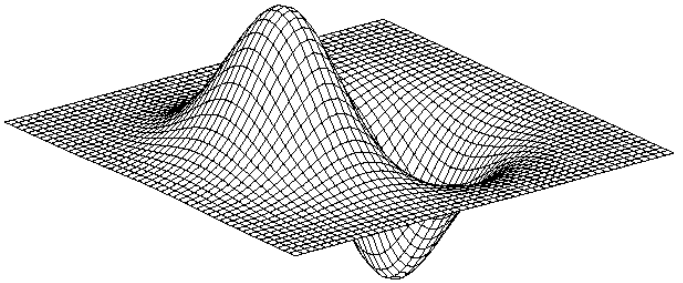
Where is the edge? Zero-crossings of bottom graph

2D edge detection filters



Gaussian

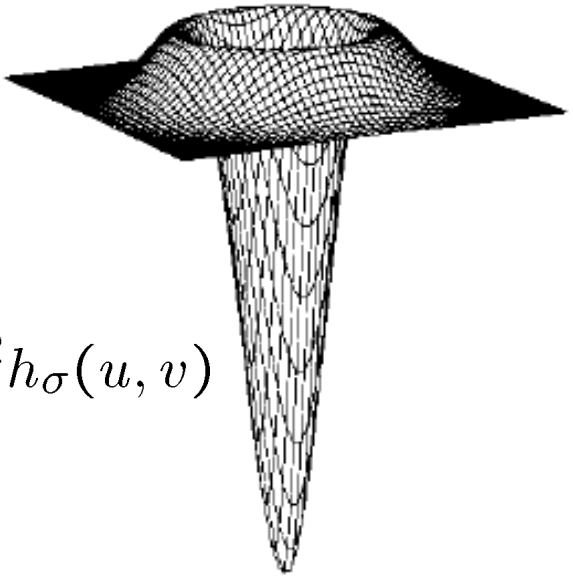
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



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:

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

Edge detection by subtraction

original

Edge detection by subtraction

smoothed (5x5 Gaussian)

Edge detection by subtraction

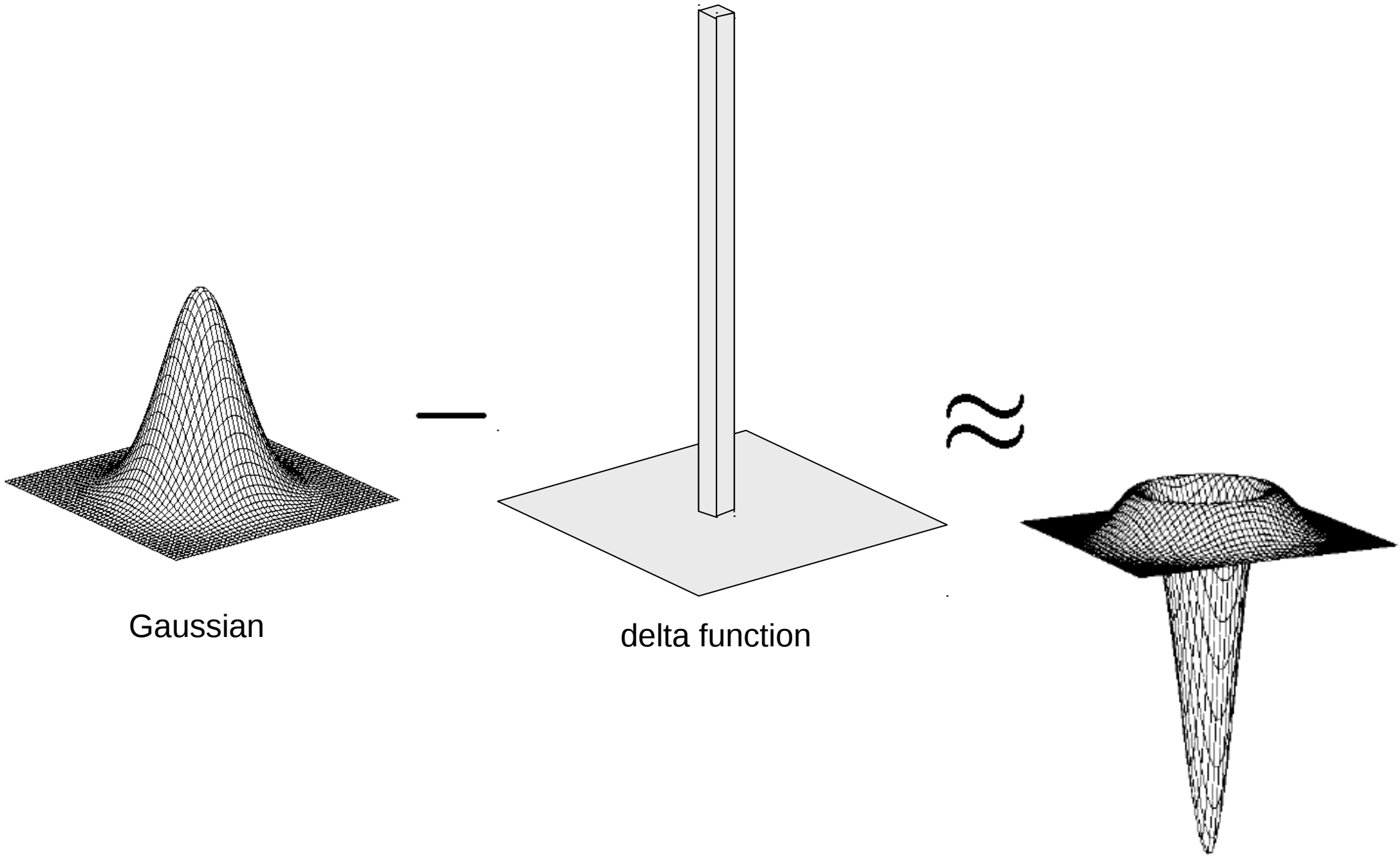
smoothed – original

(scaled by 4, offset +128)

Why does this work?

filter demo

Gaussian - image filter



Optimal Edge Detection: Canny

Assume:

- Linear filtering
- Additive iid Gaussian noise

Edge detector should have:

- Good Detection. Filter responds to edge, not noise.
- Good Localization: detected edge near true edge.
- Single Response: one per edge.

Optimal Edge Detection: Canny (continued)

- Optimal Detector is approximately Derivative of Gaussian.
- Detection/Localization trade-off
  - More smoothing improves detection
  - And hurts localization.
- This is what you might guess from (detect change) + (remove noise)

The Canny edge detector

original image (Lena)

The Canny edge detector

norm of the gradient

The Canny edge detector

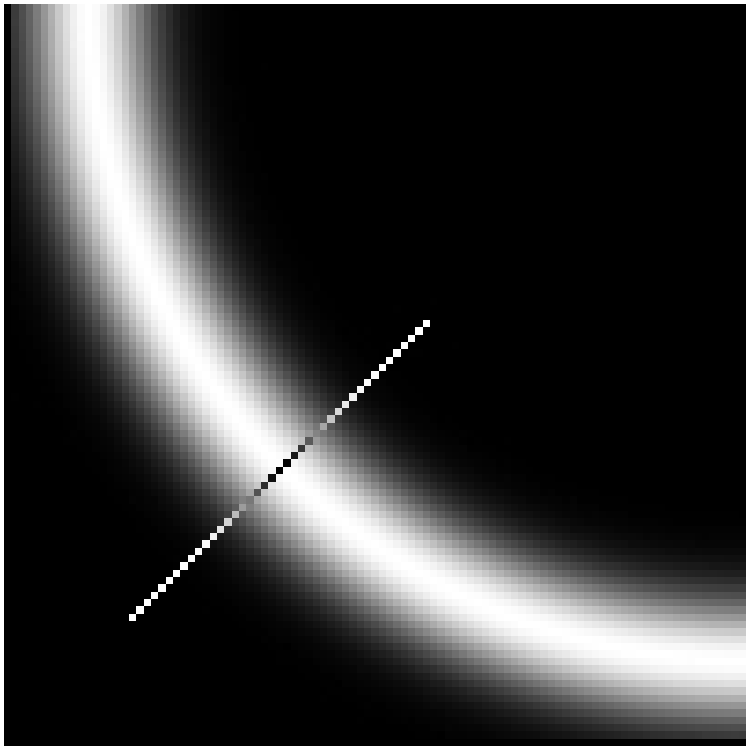
thresholding

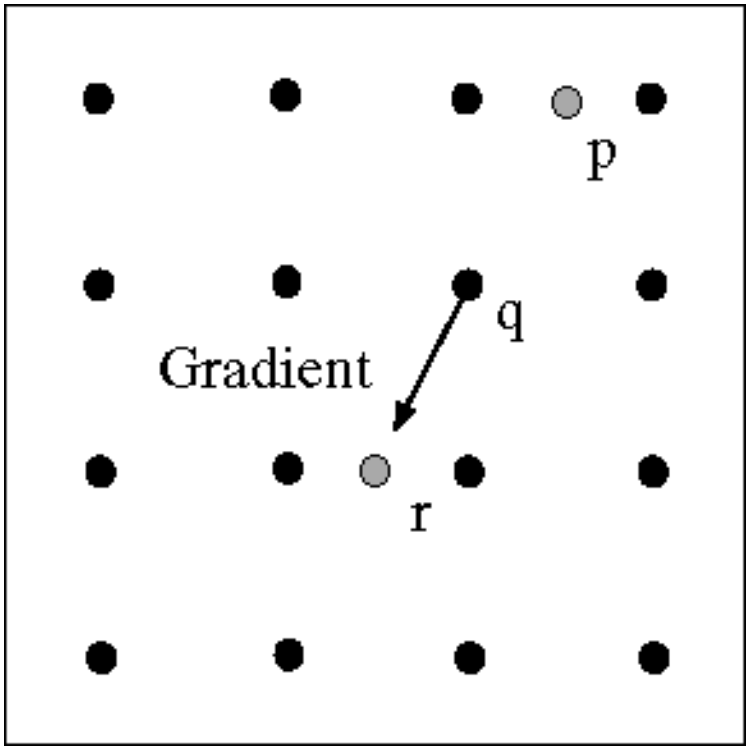
The Canny edge detector

thinning

(non-maximum suppression)

Non-maximum suppression

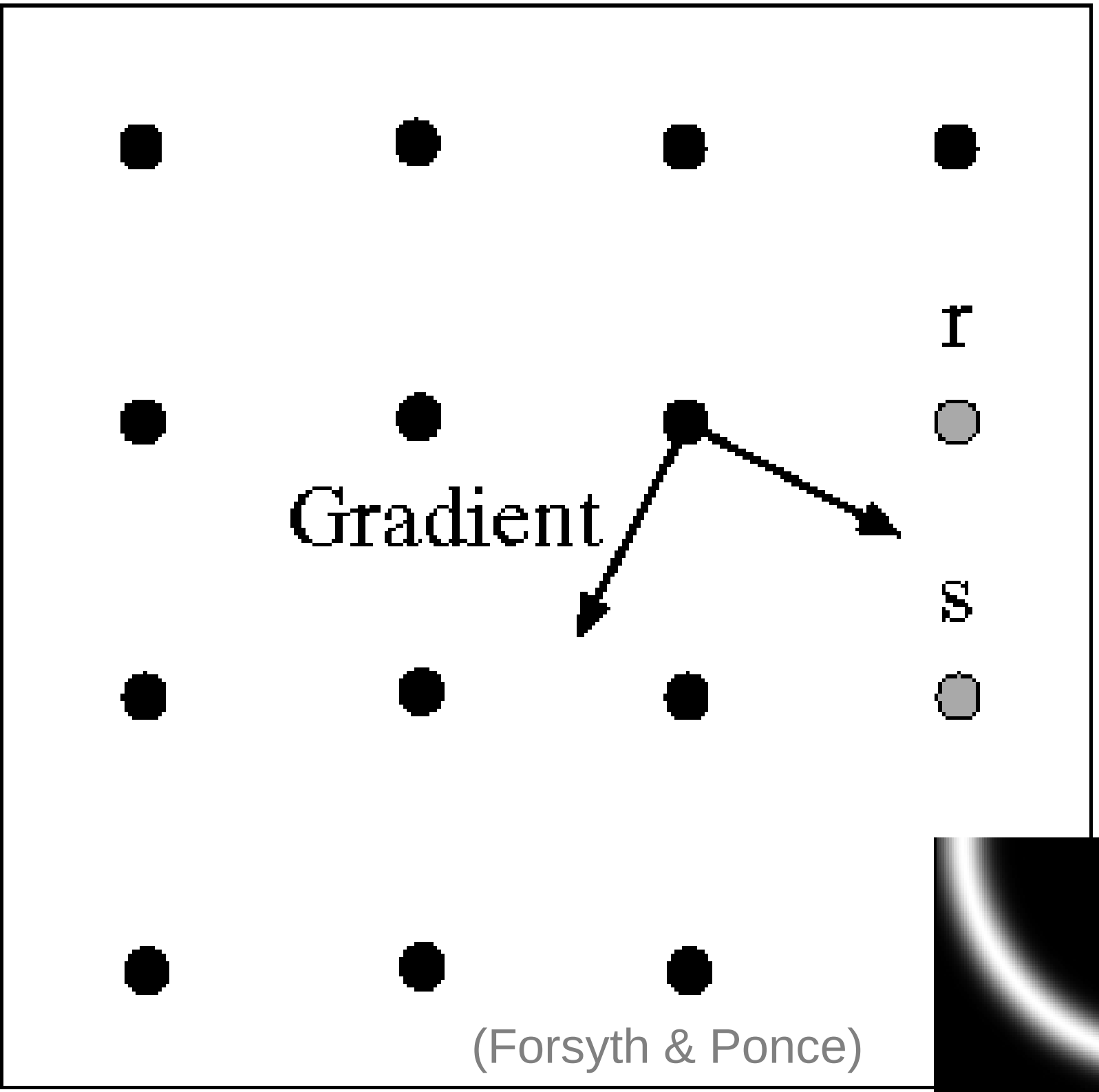




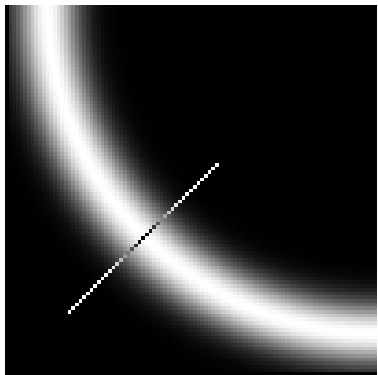
Check if pixel is local maximum along gradient direction

- requires checking interpolated pixels p and r

Predicting the next edge point



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).

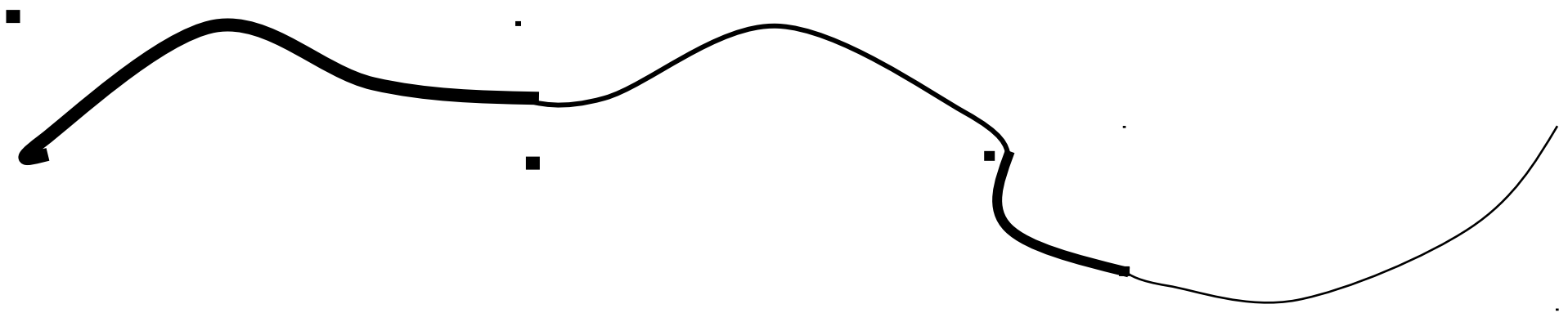


(Forsyth & Ponce)

Hysteresis

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.



Effect of σ (Gaussian kernel size)

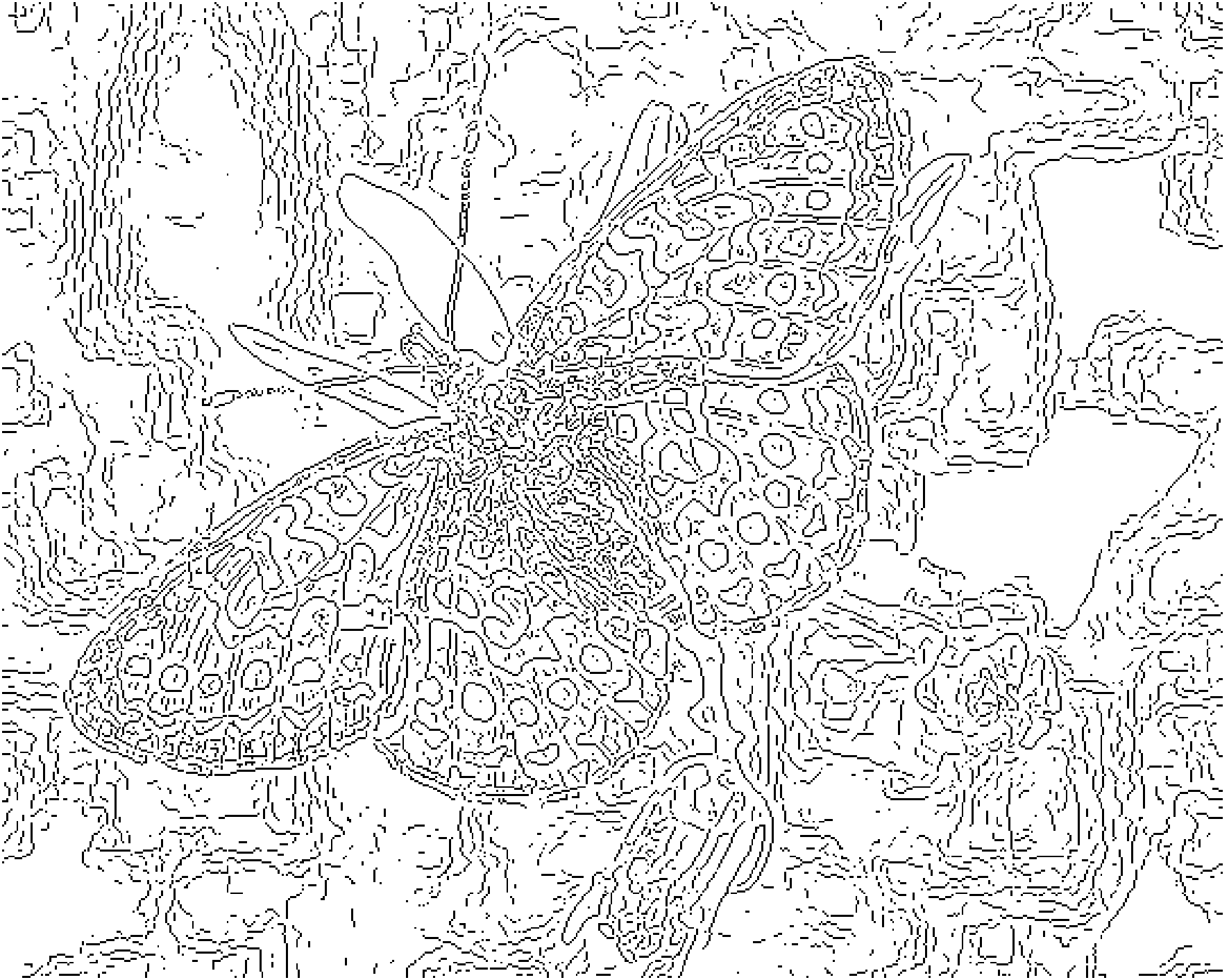
original

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



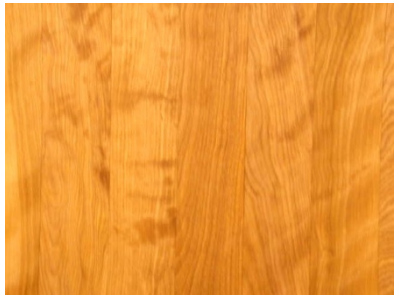
fine scale
high
threshold

coarse scale, high threshold

coarse
scale
low
threshold

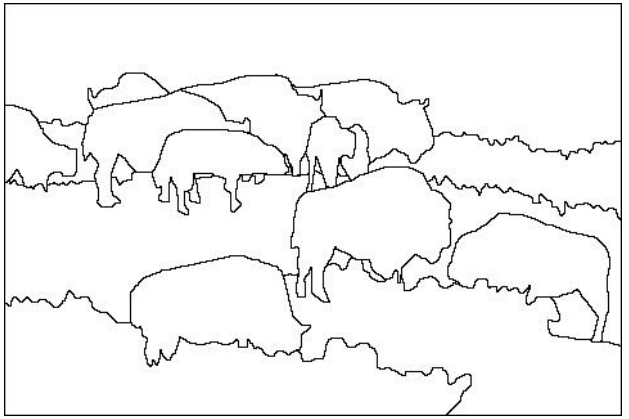
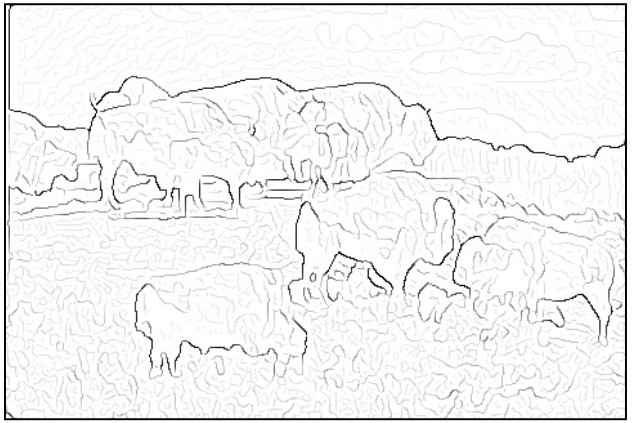
So, what scale to choose?

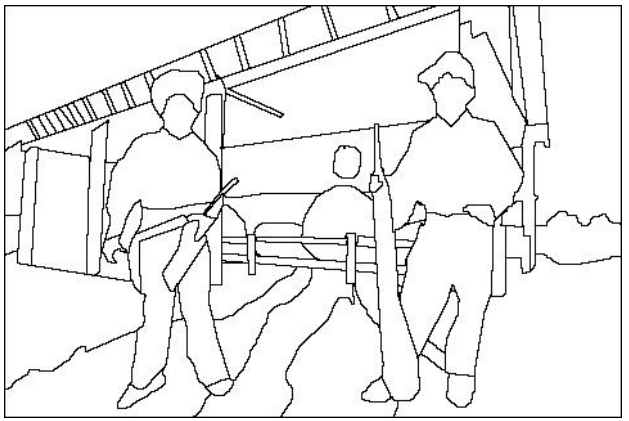
It depends what we're looking for.



Learning to detect boundaries

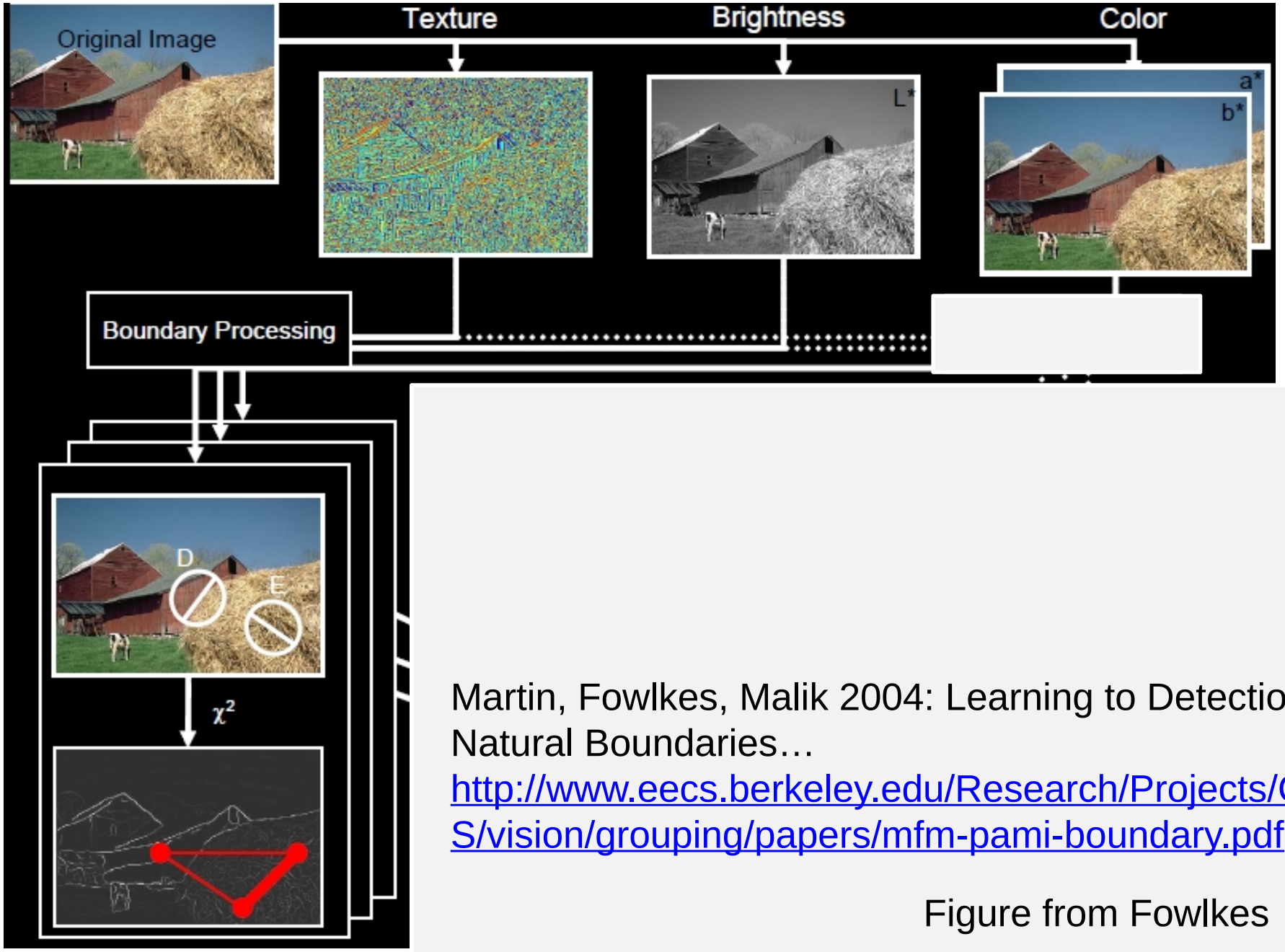
image | human segmentation | gradient magnitude


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

pB boundary detector



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

Figure from Fowlkes

pB Boundary Detector

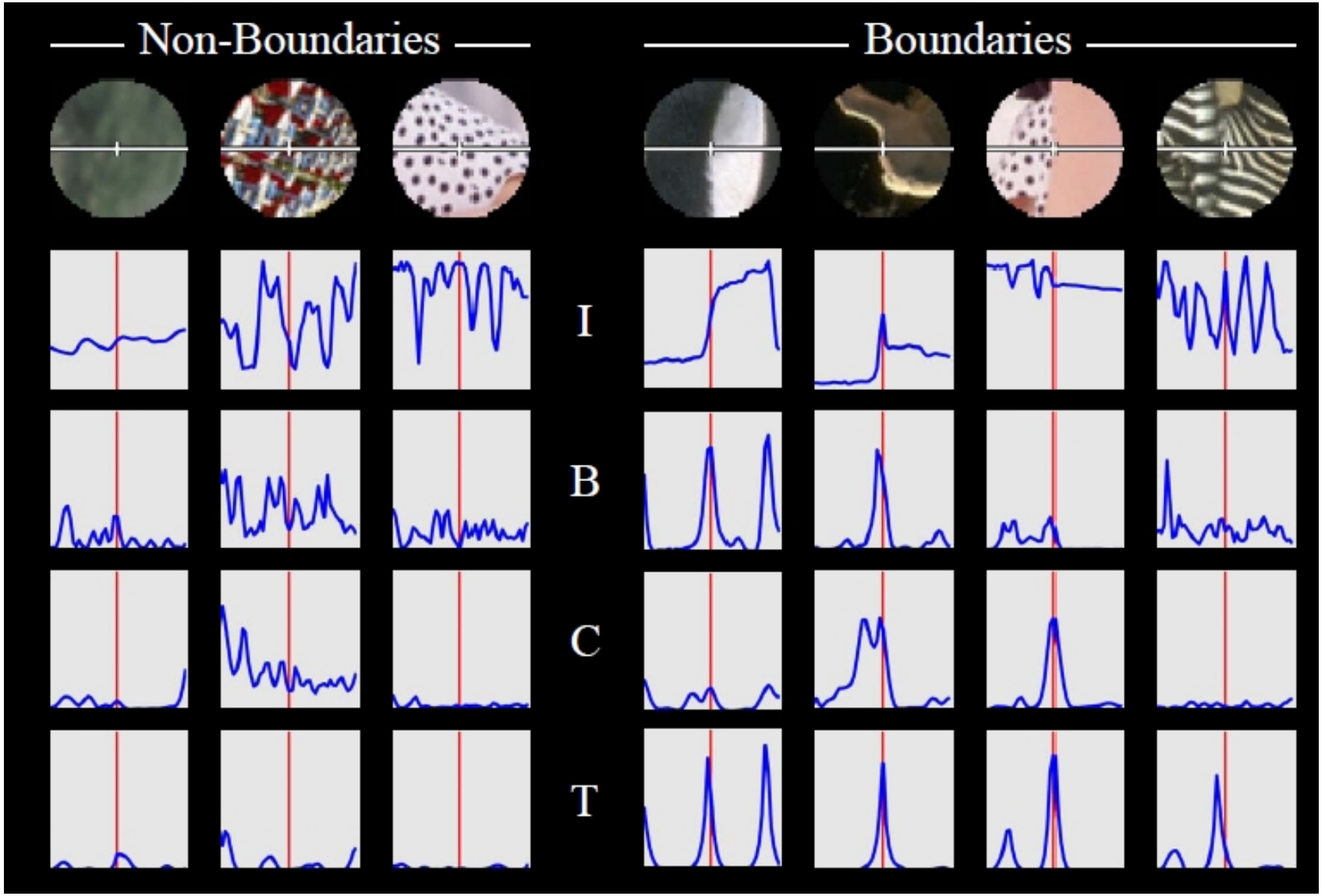


Figure from Fowlkes

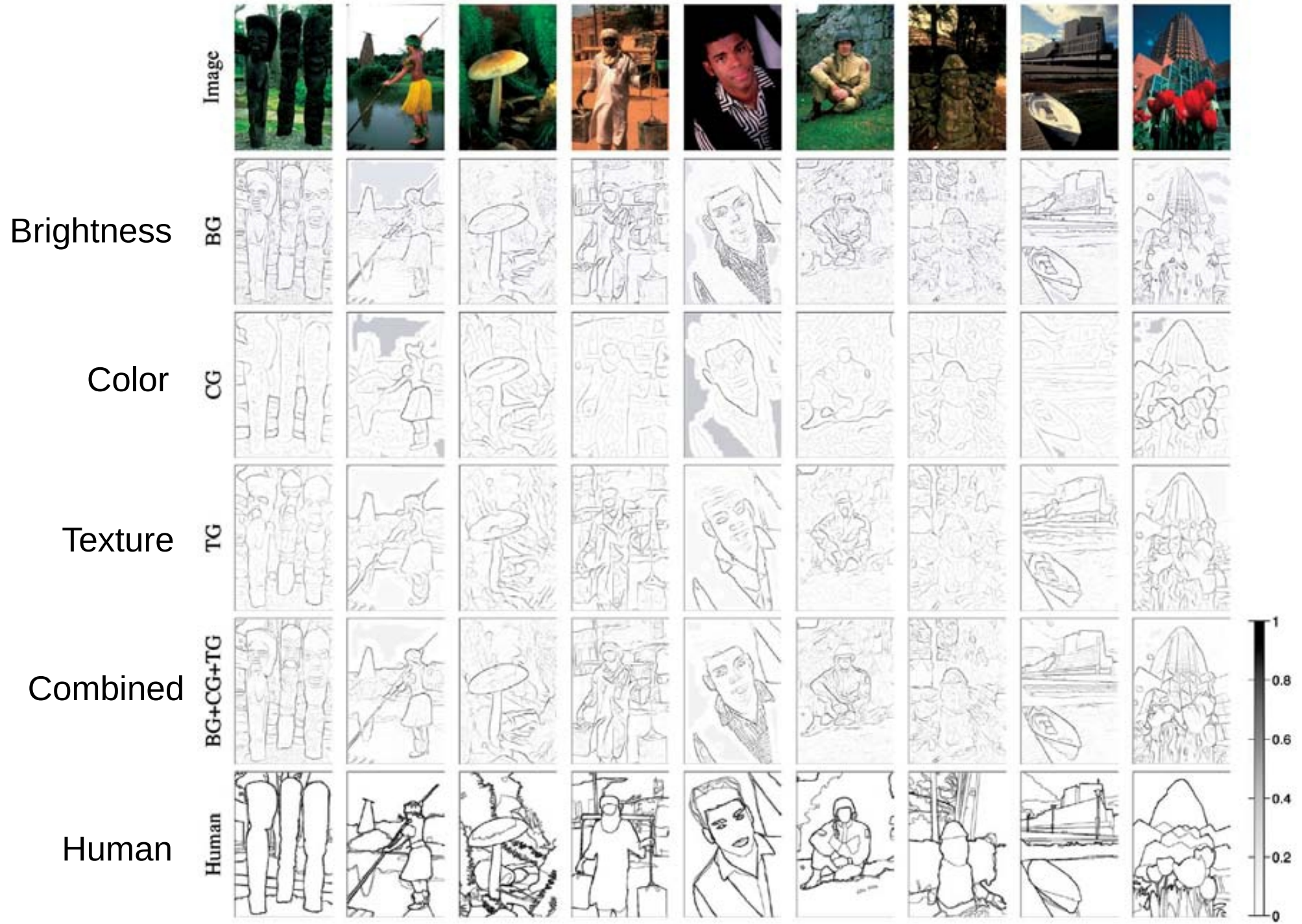
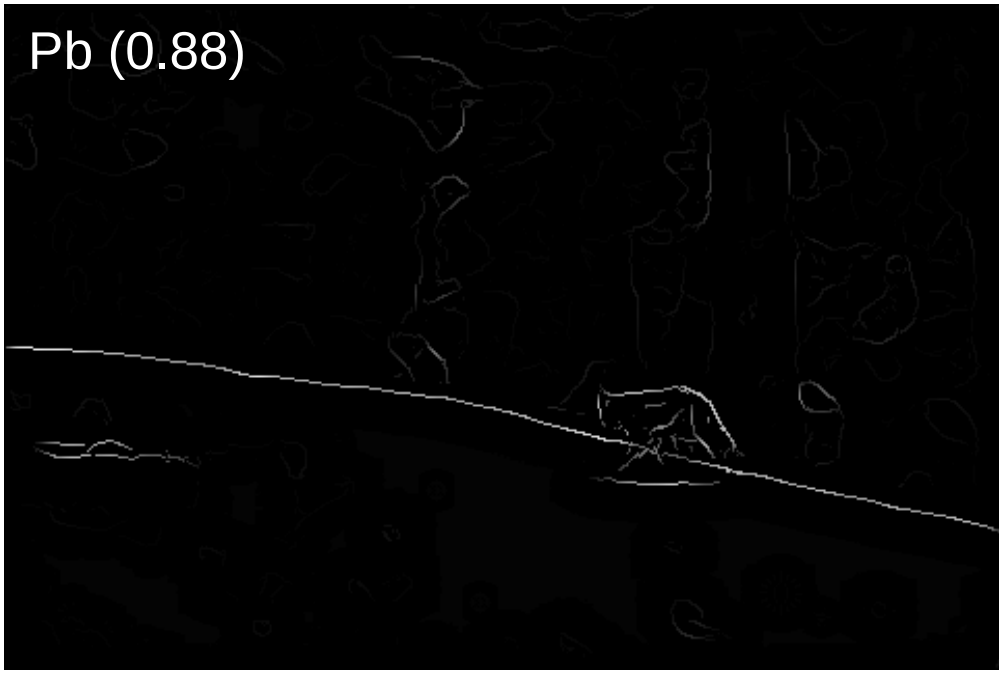
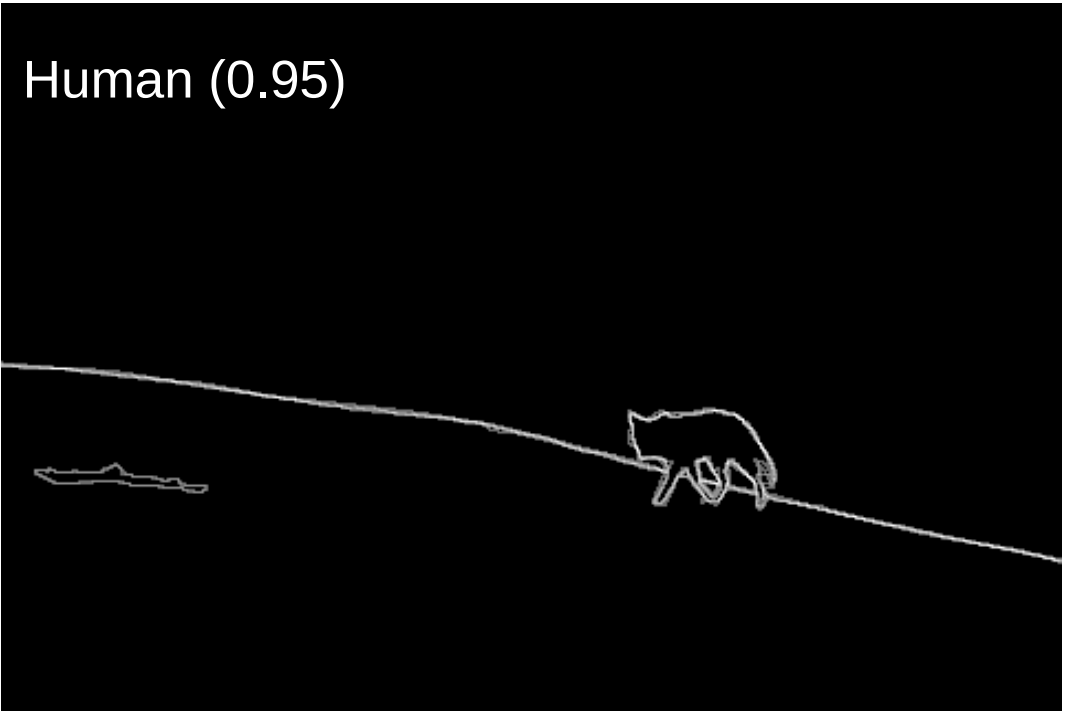
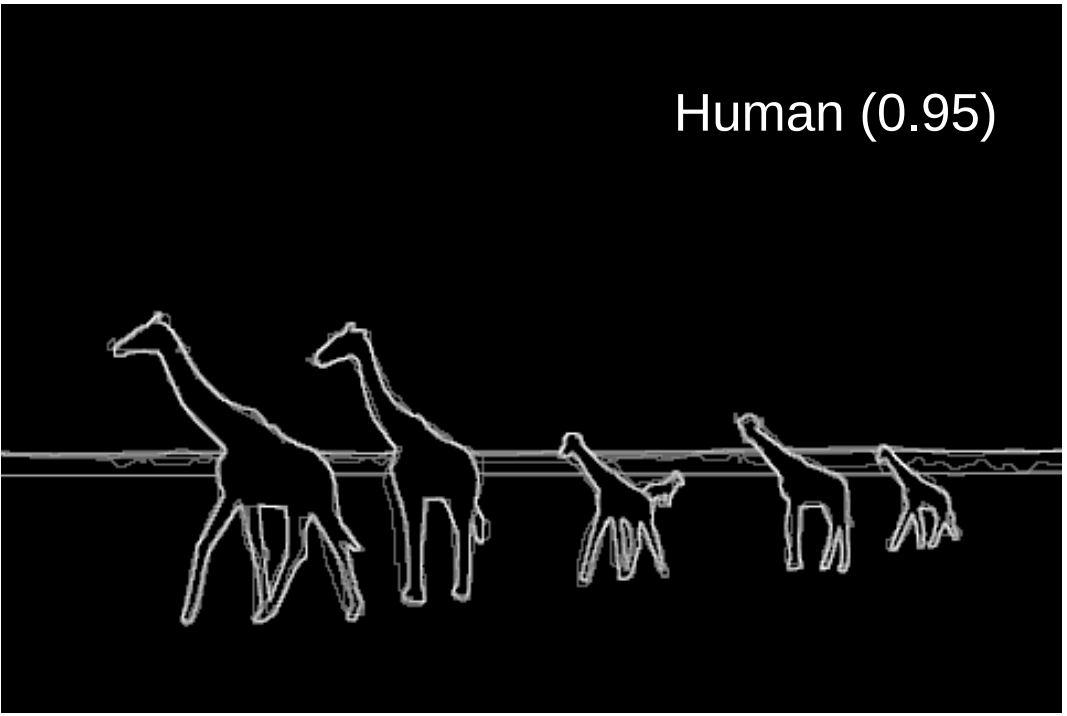


Image
BG
CG
TG
BG+CG+TG
Human
Brightness
Color
Texture
Combined
Human
1
0.8
0.6
0.4
0.2
0

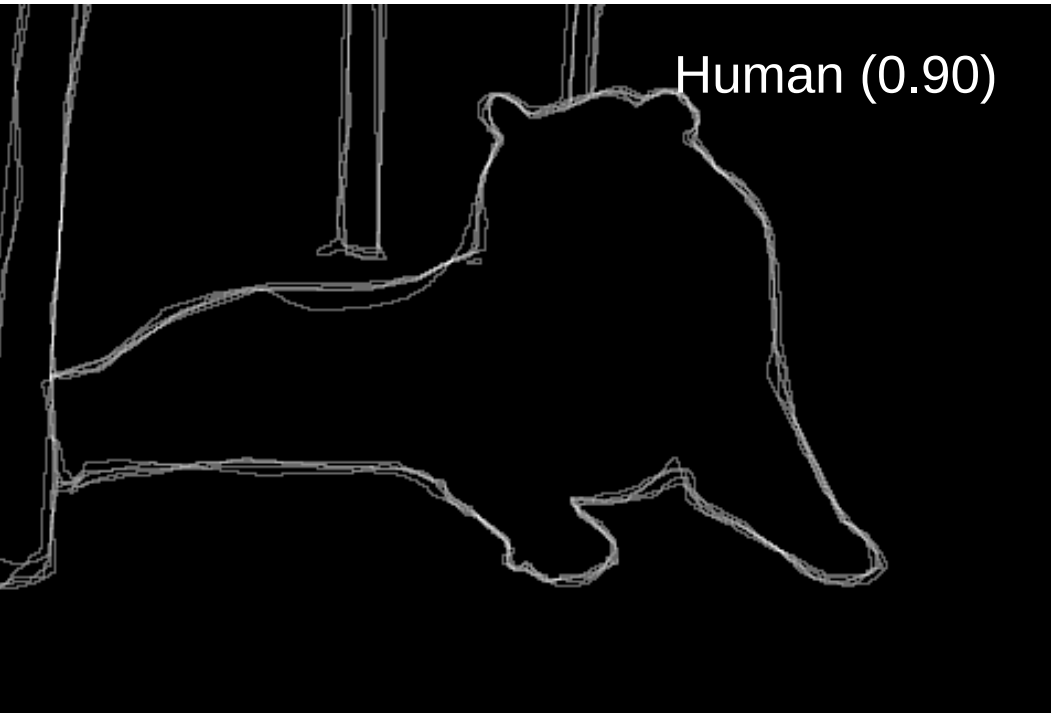
Results






Human (0.95)


Pb (0.63)



For more: [http ://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/bench/html/108082-color.html](http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/bench/html/108082-color.html)

State of edge detection

- Local edge detection is mostly solved
  - Intensity **gradient**, color, **texture**
- Some methods to take into account longer contours, but could probably do better
- Poor use of object and high-level information