Computer Vision Course
Lecture 05

Edge / Boundary Detection

Ceyhun Burak Akgül, PhD
cba-research.com

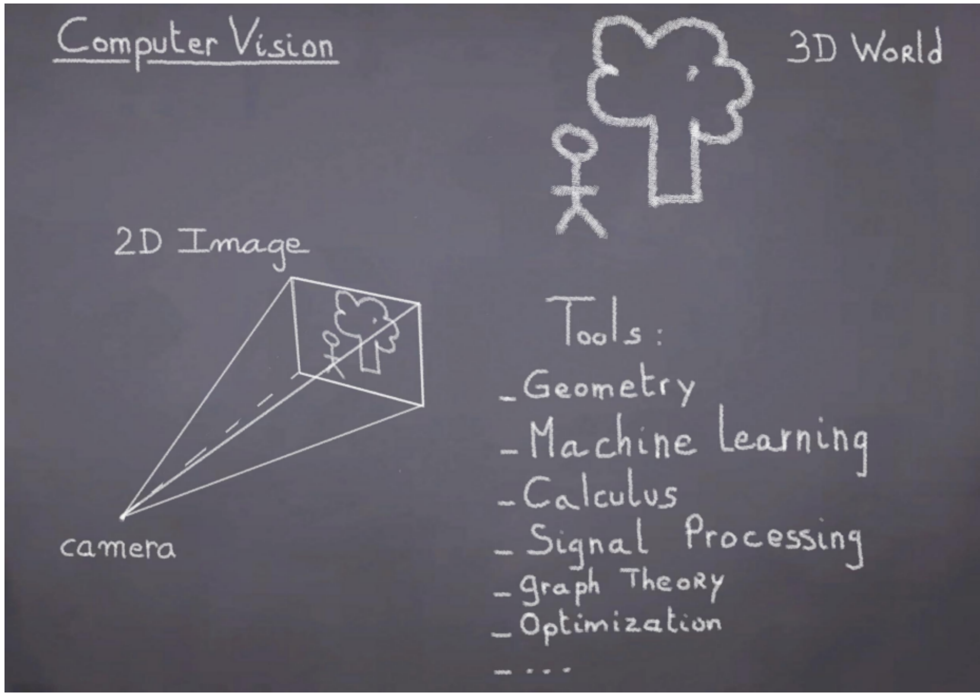
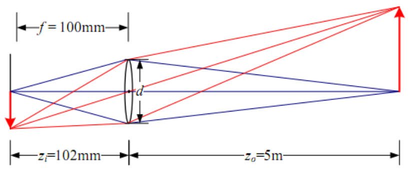


Photo credit: Olivier Teboul
vision.mas.ecp.fr/Personnel/teboul

Spring 2015
Last updated 25/03/2015

These slides have been adapted from James Hays's 2014 Computer Vision course slides at Brown University.

Course Outline



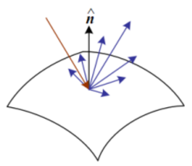
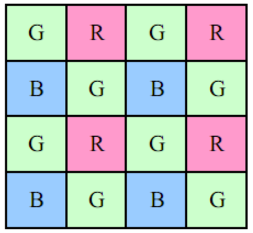


Image Formation and Processing

Light, Shape and Color

The Pin-hole Camera Model, The Digital Camera

Linear filtering, Template Matching, Image Pyramids



Feature Detection and Matching

Edge Detection, Interest Points: Corners and Blobs

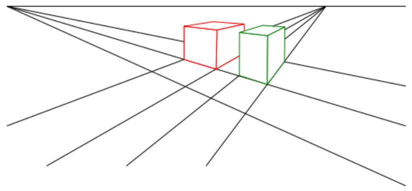
Local Image Descriptors

Feature Matching and Hough Transform

Multiple Views and Motion

Geometric Transformations, Camera Calibration

Feature Tracking , Stereo Vision



Segmentation and Grouping

Segmentation by Clustering, Region Merging and Growing

Advanced Methods Overview: Active Contours, Level-Sets, Graph-Theoretic Methods

Detection and Recognition

Problems and Architectures Overview

Statistical Classifiers, Bag-of-Words Model, Detection by Sliding Windows

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

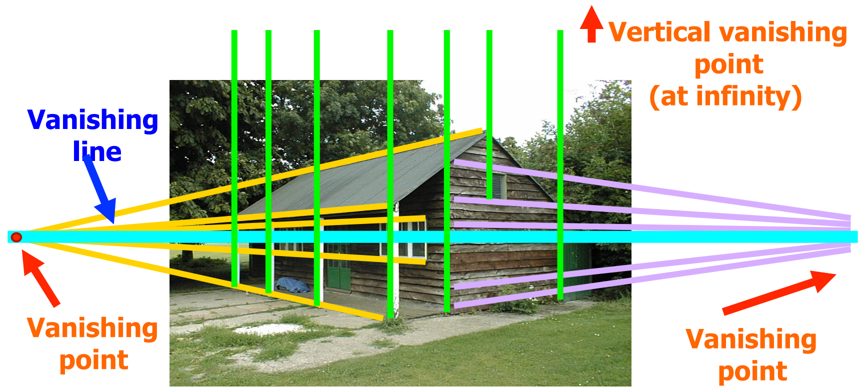
- **Ideal:** artist' s line drawing (but artist is also using object-level knowledge)

Source: D. Lowe

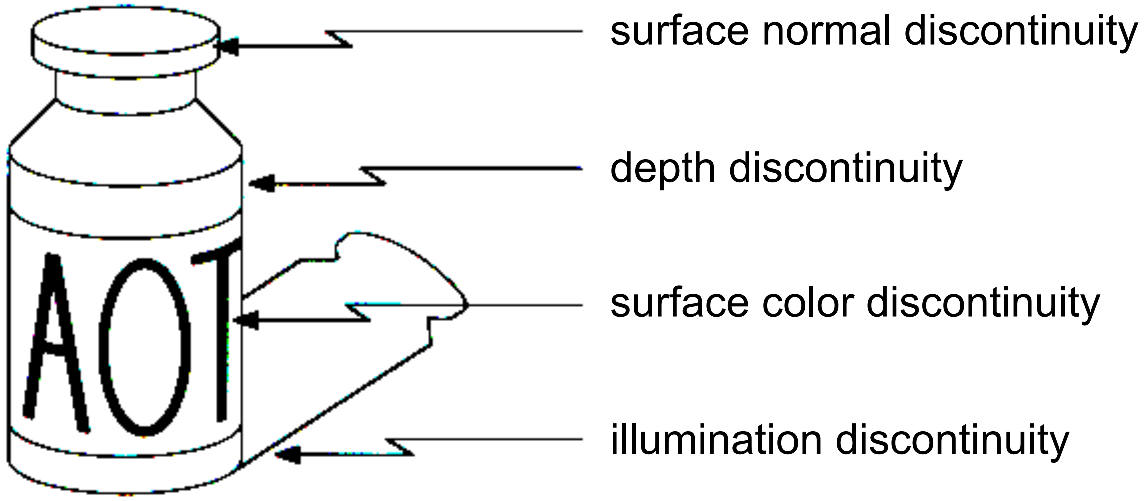
Why Are Edges Important?

- Extract information, recognize objects

- Recover geometry and viewpoint



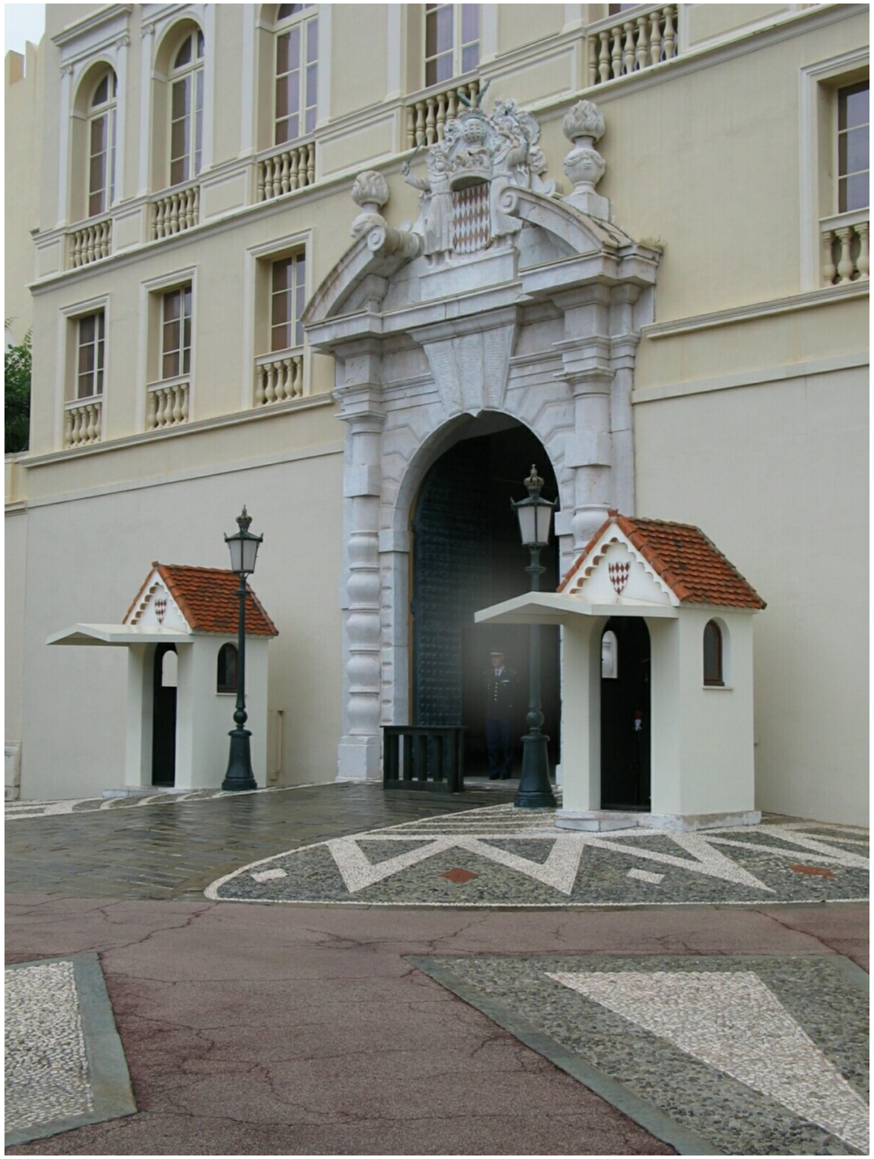
How Are Edges Formed?


surface normal discontinuity
depth discontinuity
surface color discontinuity
illumination discontinuity
AOT

- Edges are caused by a variety of factors

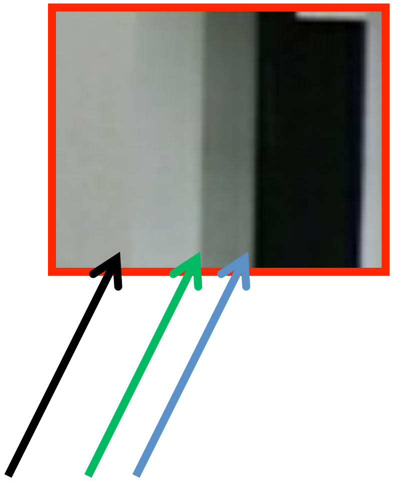
Source: Steve Seitz

Looking from Close



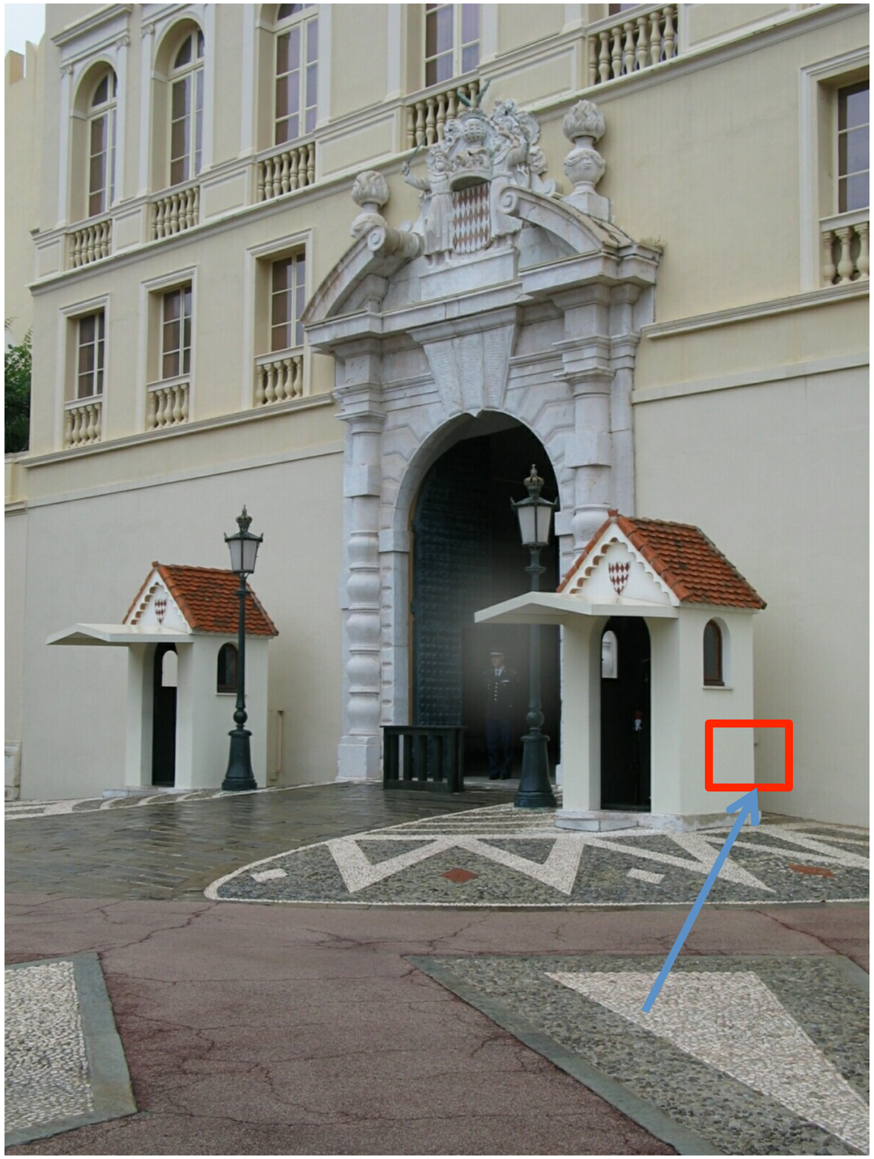
Source: D. Hoiem

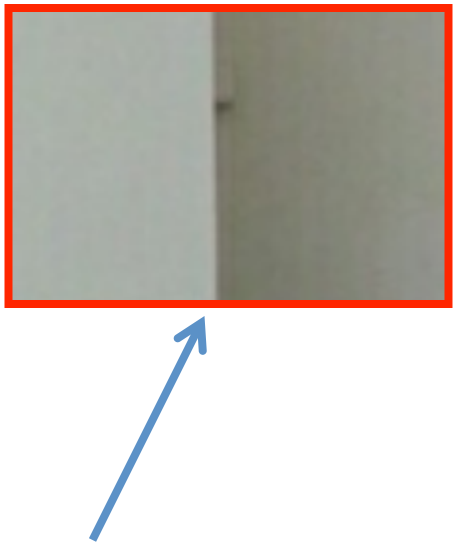
Looking from Close



Source: D. Hoiem

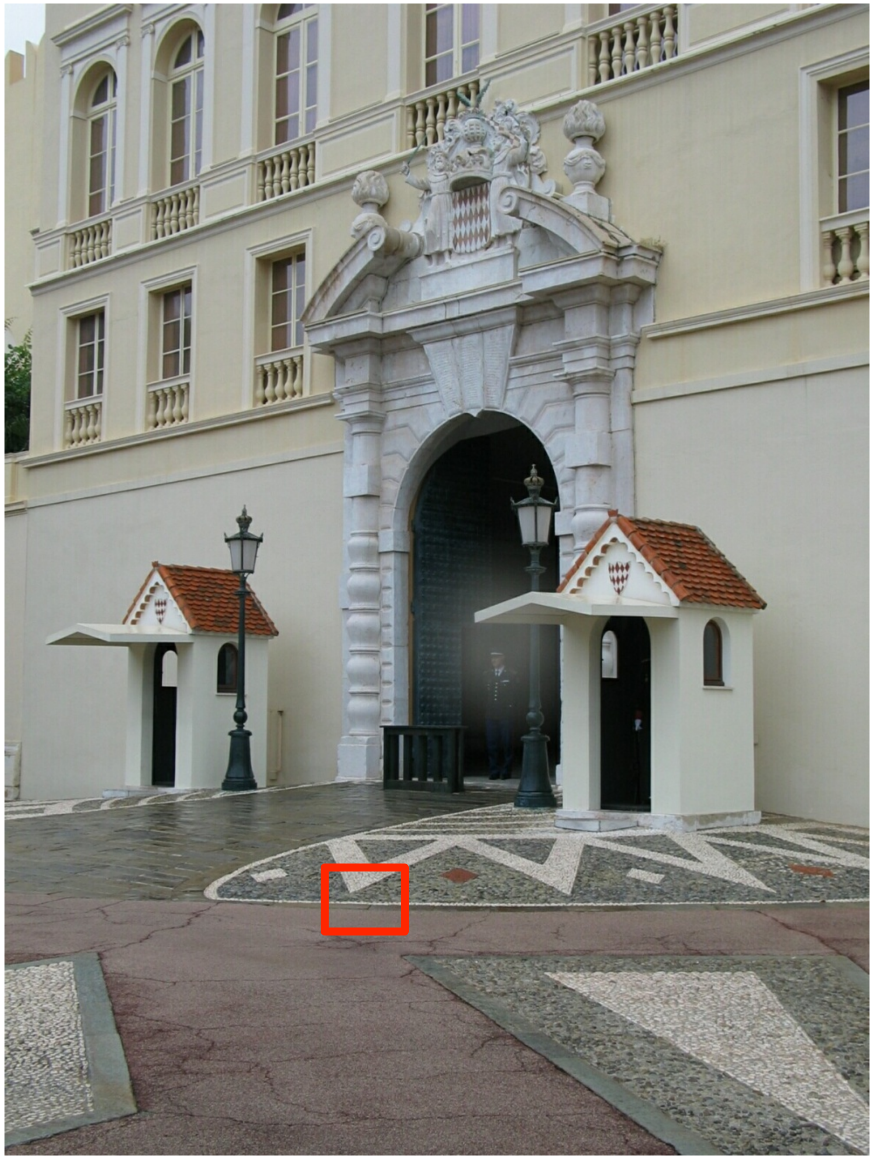
Looking from Close

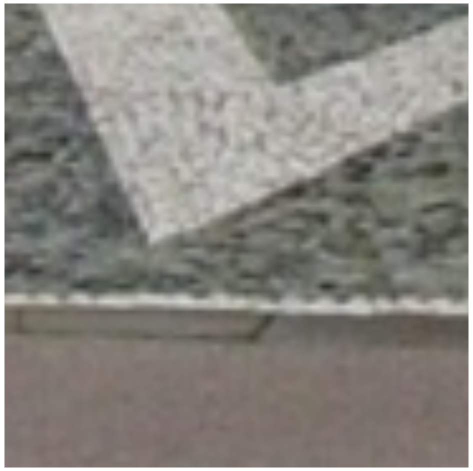




Source: D. Hoiem

Looking from Close

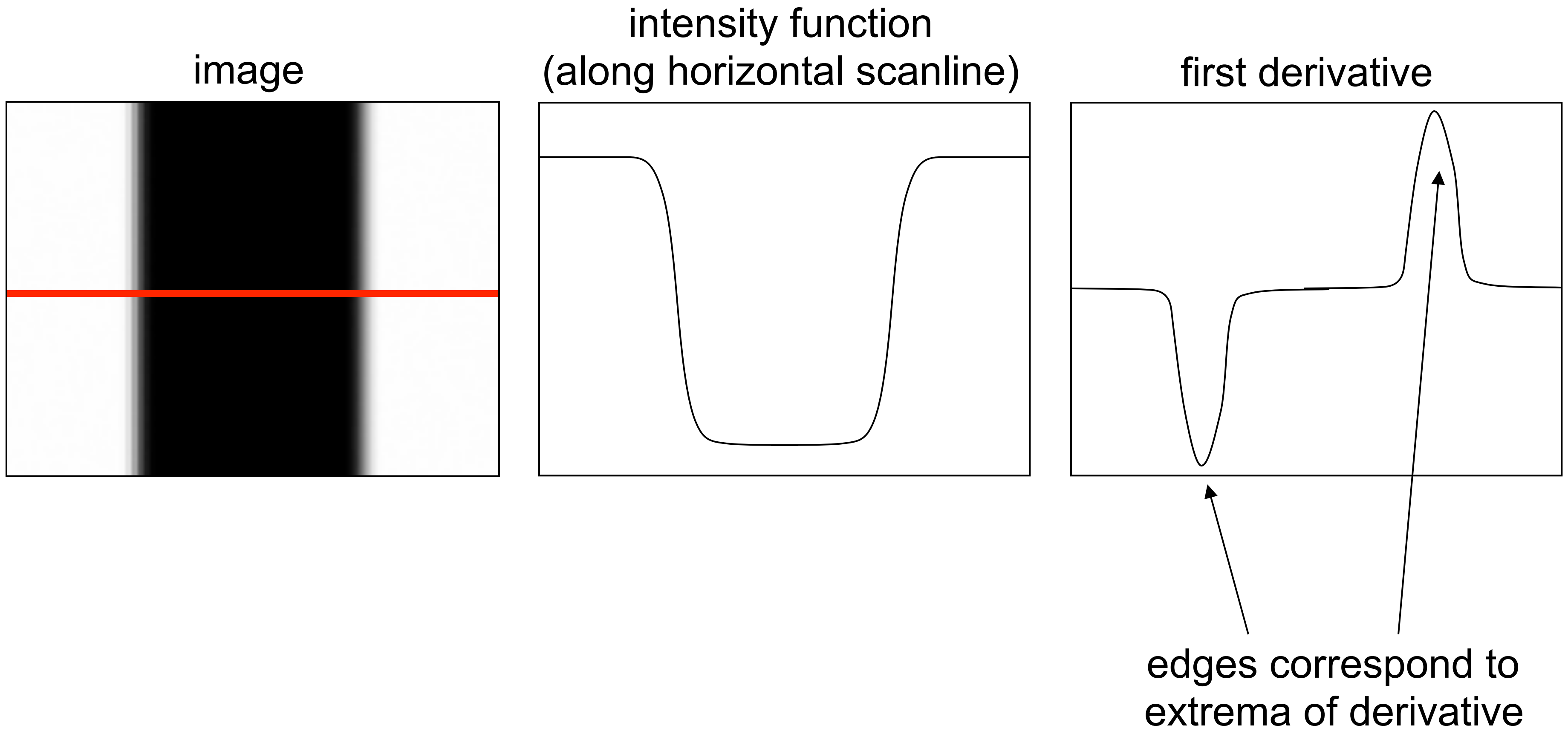




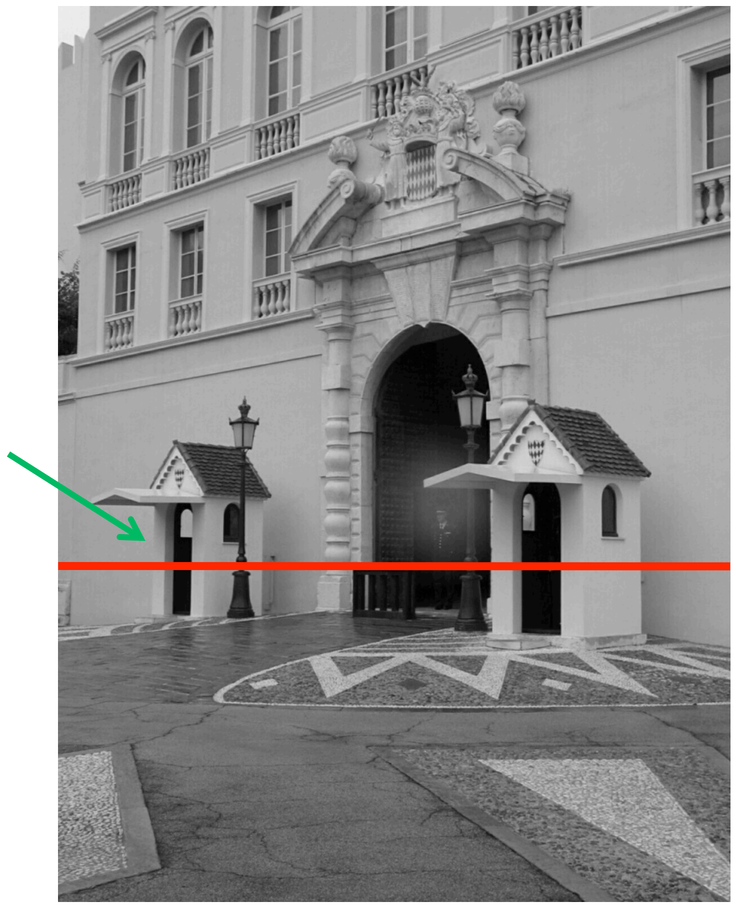
Source: D. Hoiem

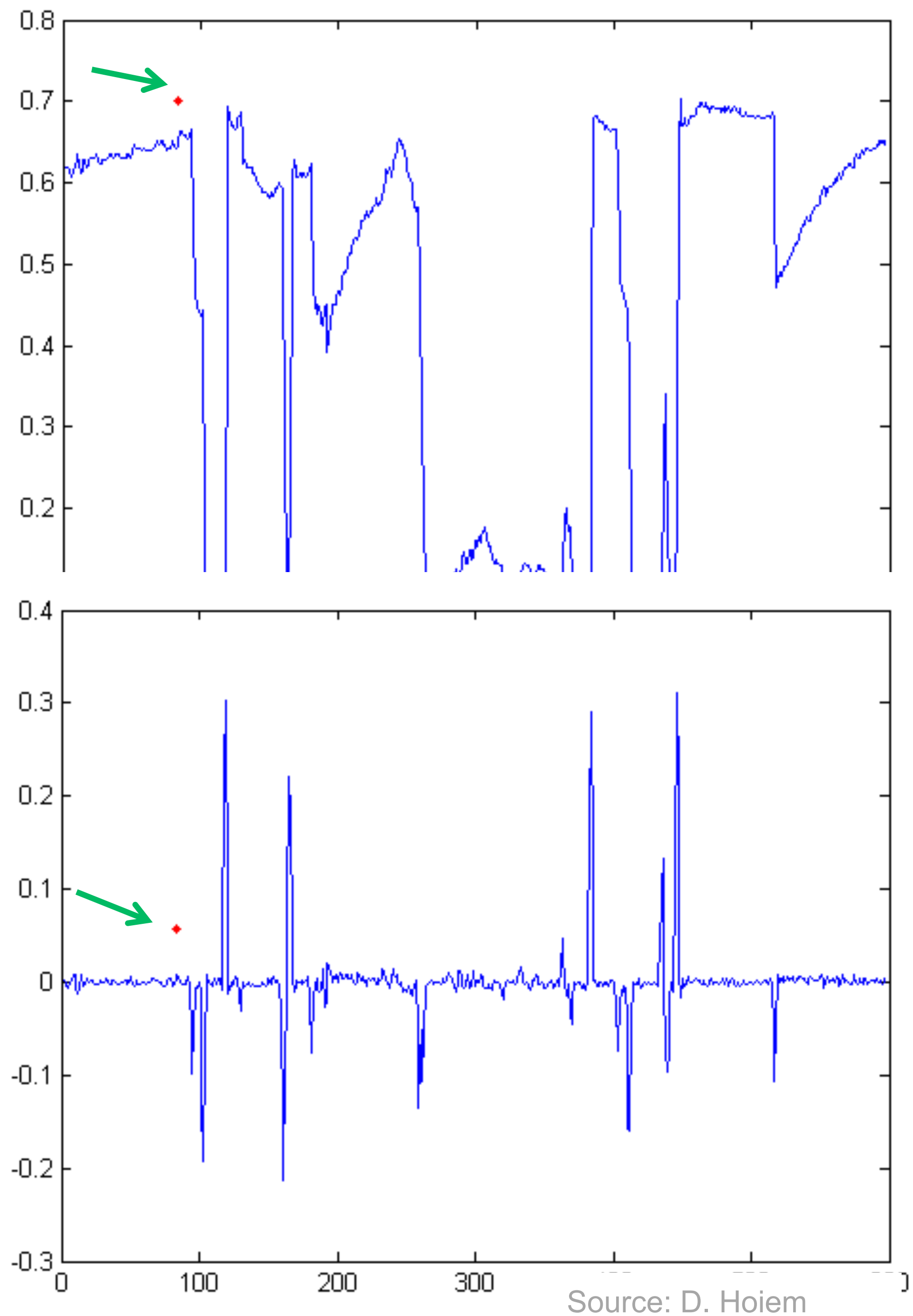
Characterizing Edges

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



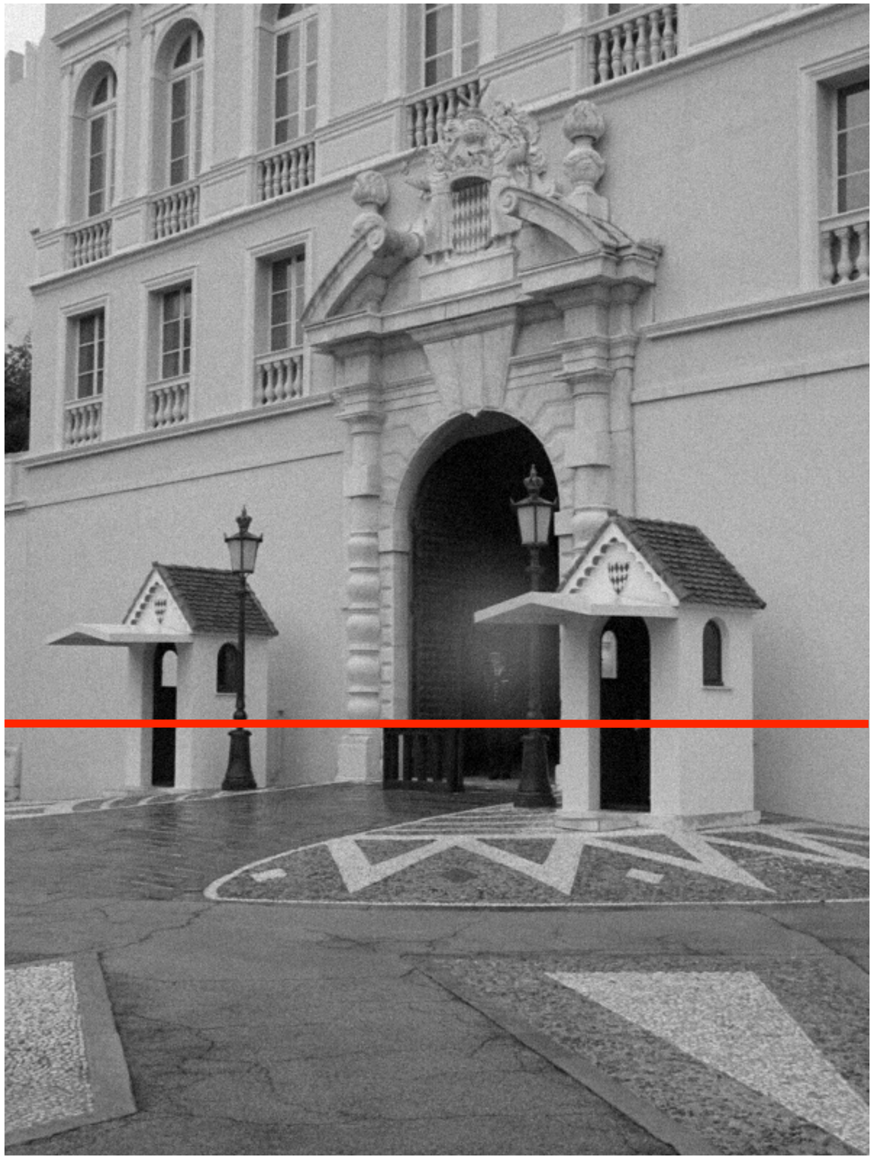
Intensity Profile

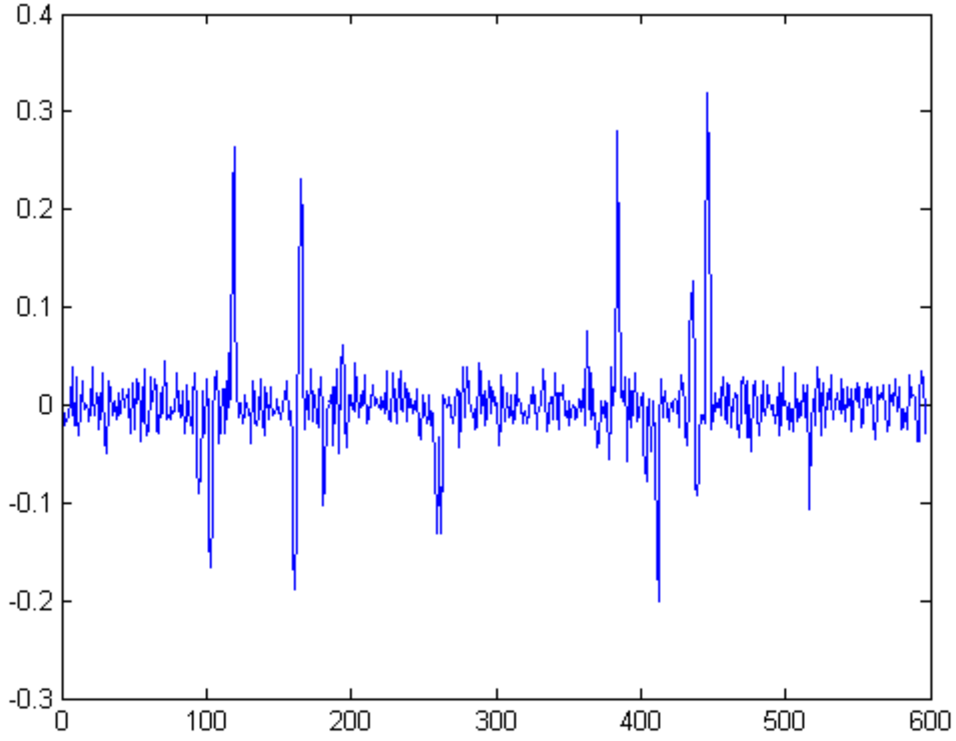




Source: D. Hoiem

Add Some Gaussian Noise



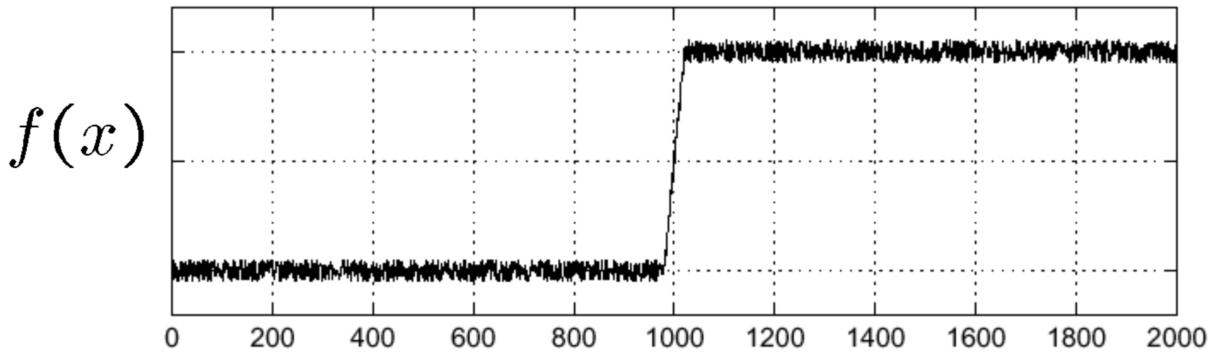


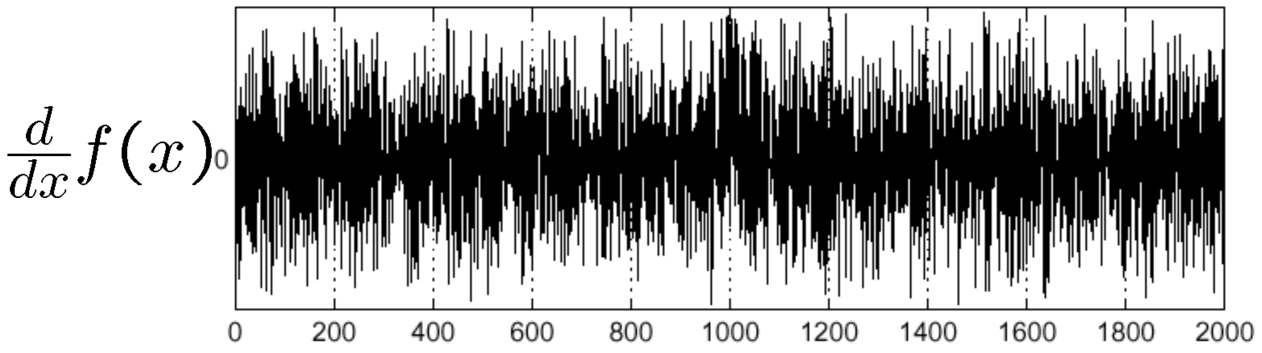
Gradient

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?

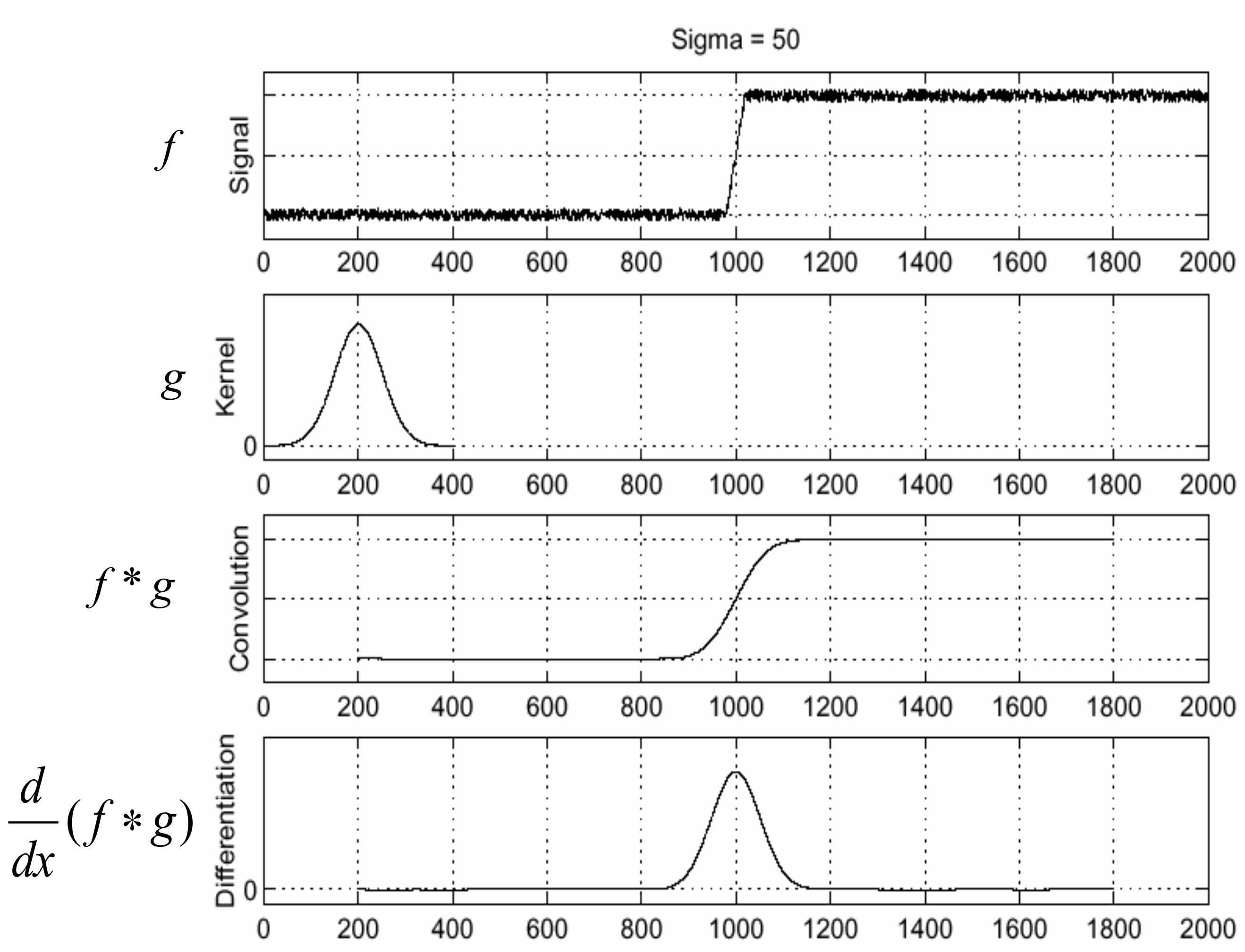
Source: S. Seitz

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

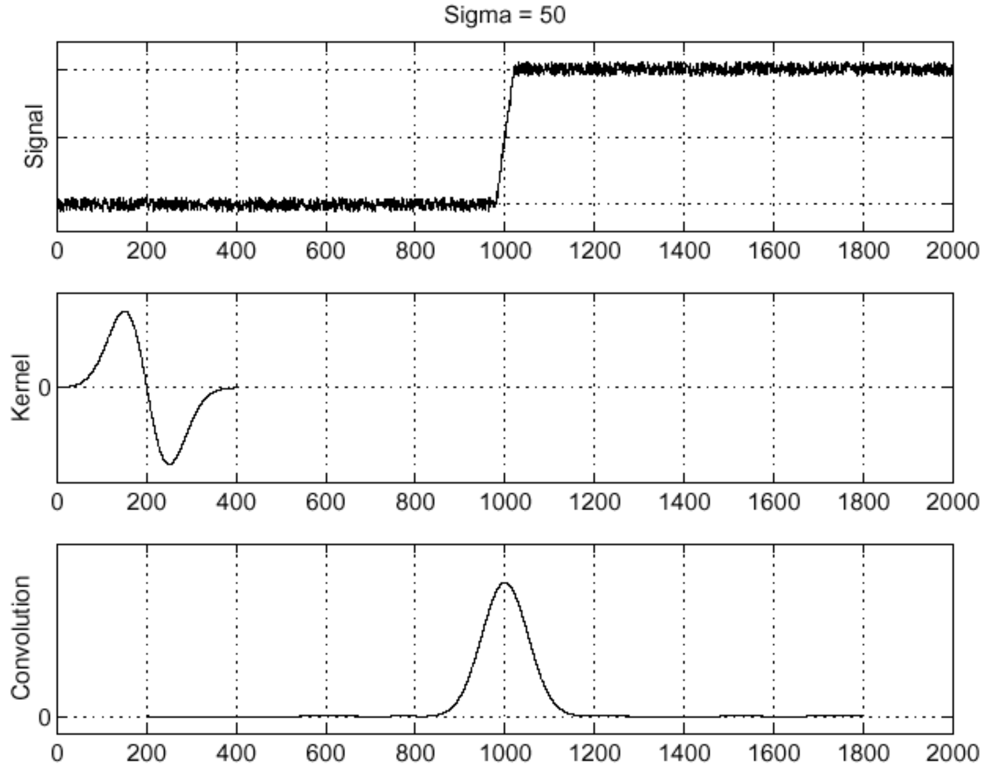
Derivative Theorem of Convolution

- Differentiation is convolution, and convolution is associative:
$$\frac{d}{dx}(f * g) = f * \frac{d}{dx} g$$
- This saves us one operation:

f

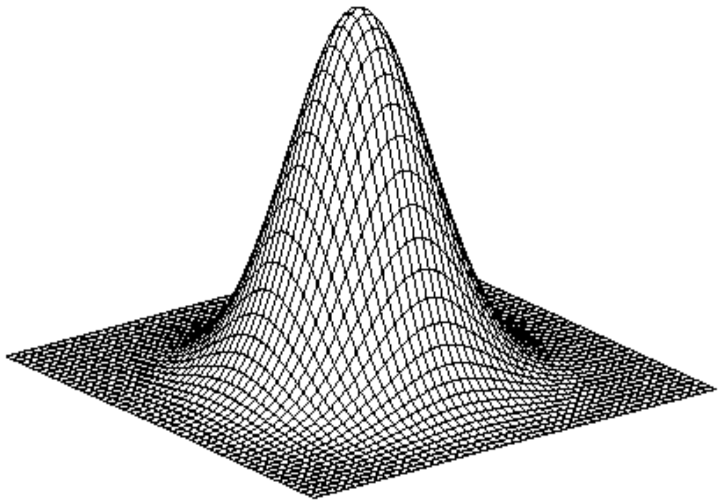
$\frac{d}{dx} g$

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

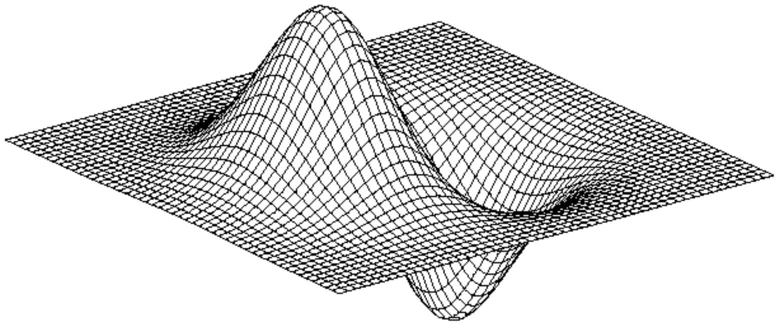


Source: S. Seitz

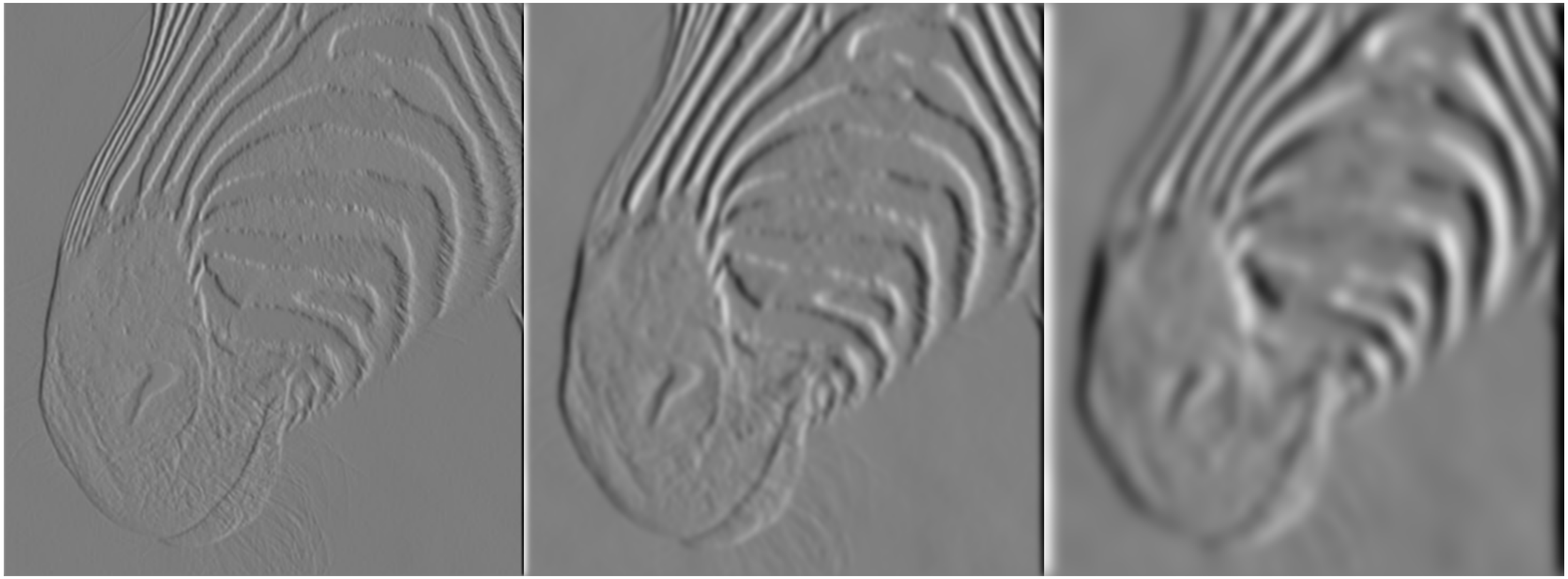
Derivative of Gaussian Filter



* [1 -1] =



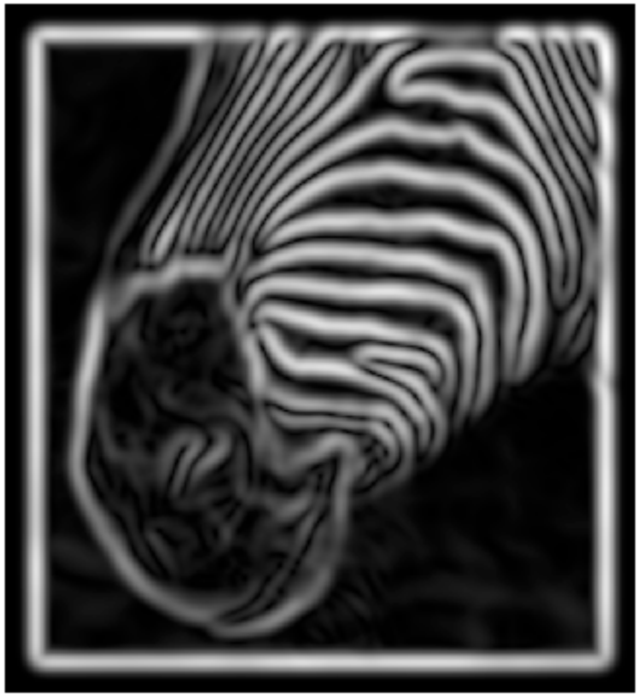
Smoothing vs Localization



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

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

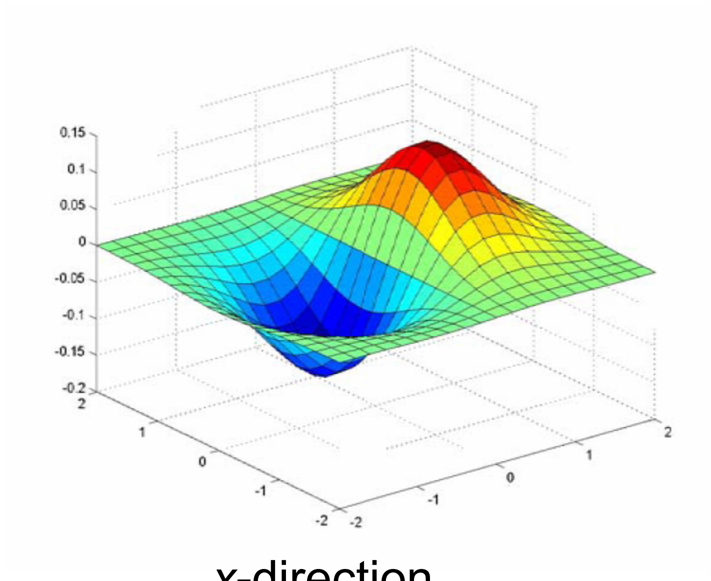
J. Canny, ***A Computational Approach To Edge Detection***, IEEE Trans. Pattern Analysis and Machine Intelligence, 8:679-714, 1986.

Source: L. Fei-Fei

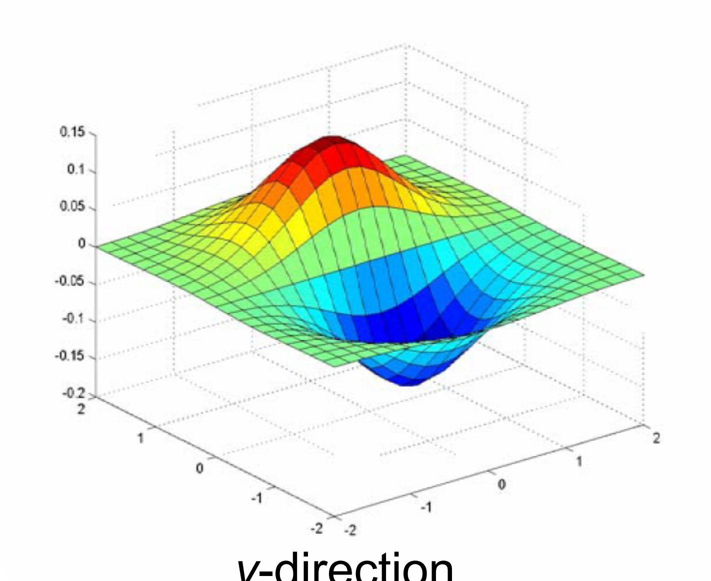
Example

original image (Lena)

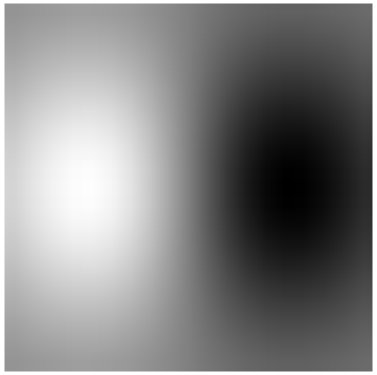
Derivative of Gaussian Filter

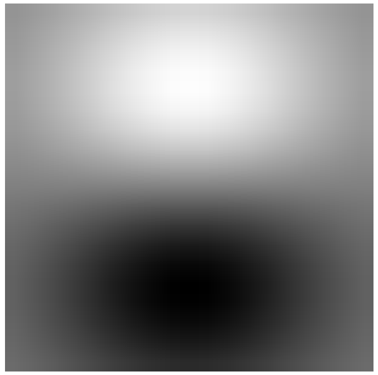


x-direction

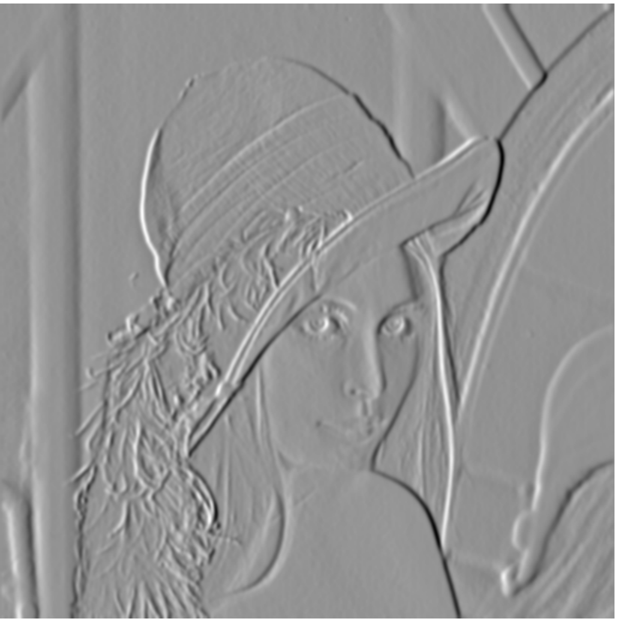


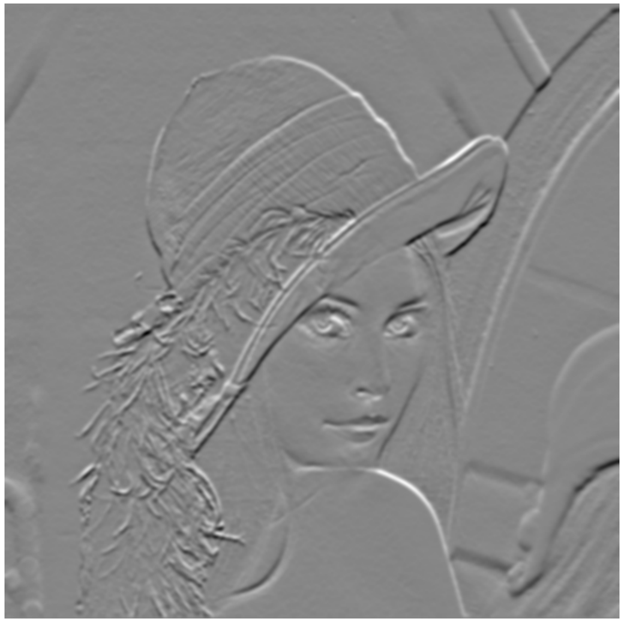
y-direction

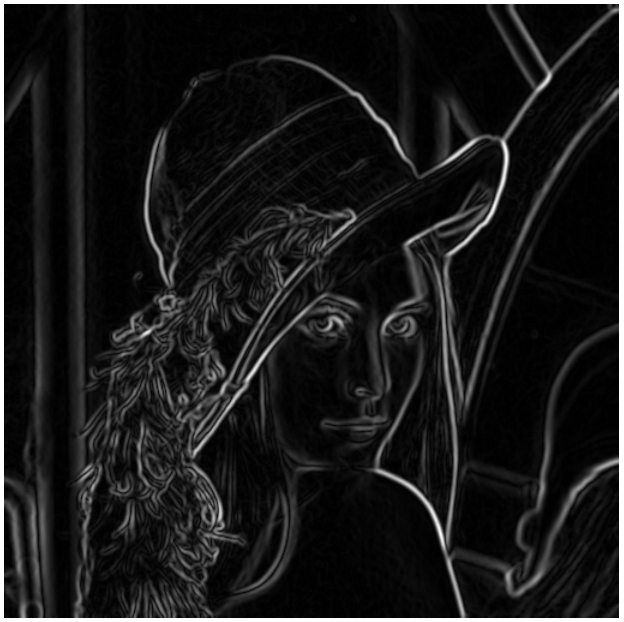




Compute Gradients (DoG)

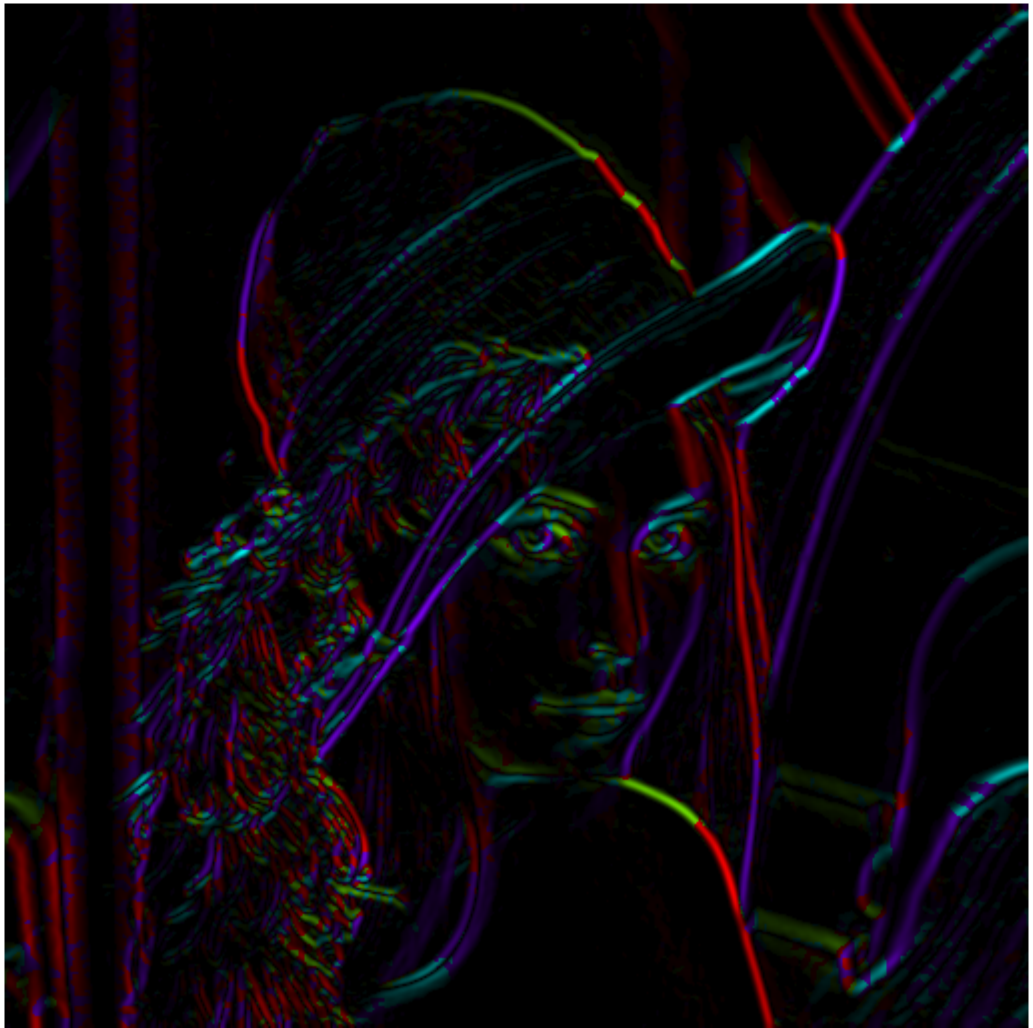

X-Derivative of Gaussian


Y-Derivative of Gaussian


Gradient Magnitude

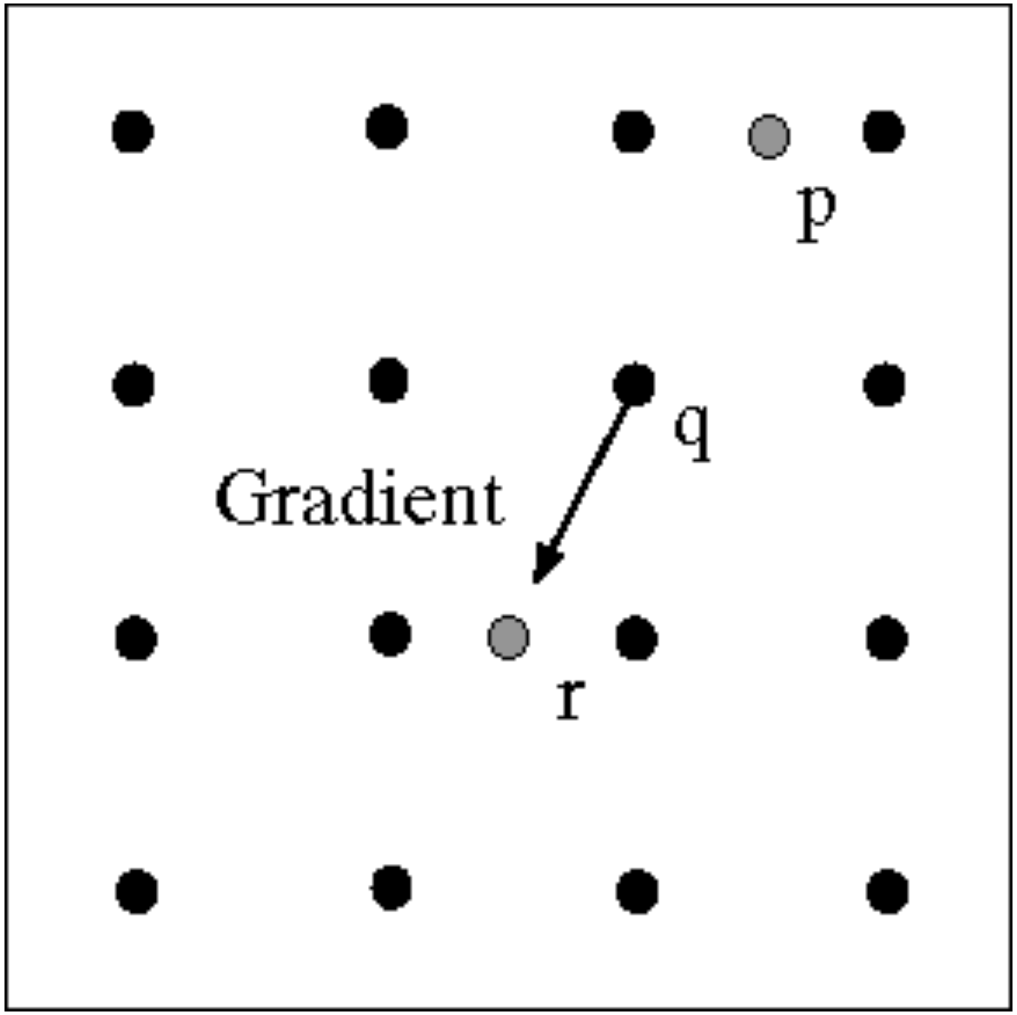
Get Orientation at Each Pixel

- Threshold at minimum level
- Get orientation

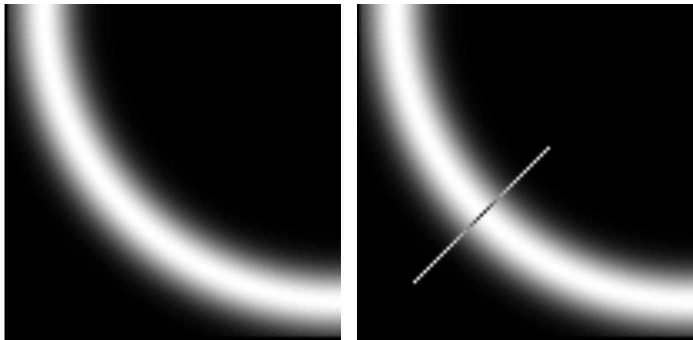


theta = atan2(gy, gx)

Non-Maximum Suppression

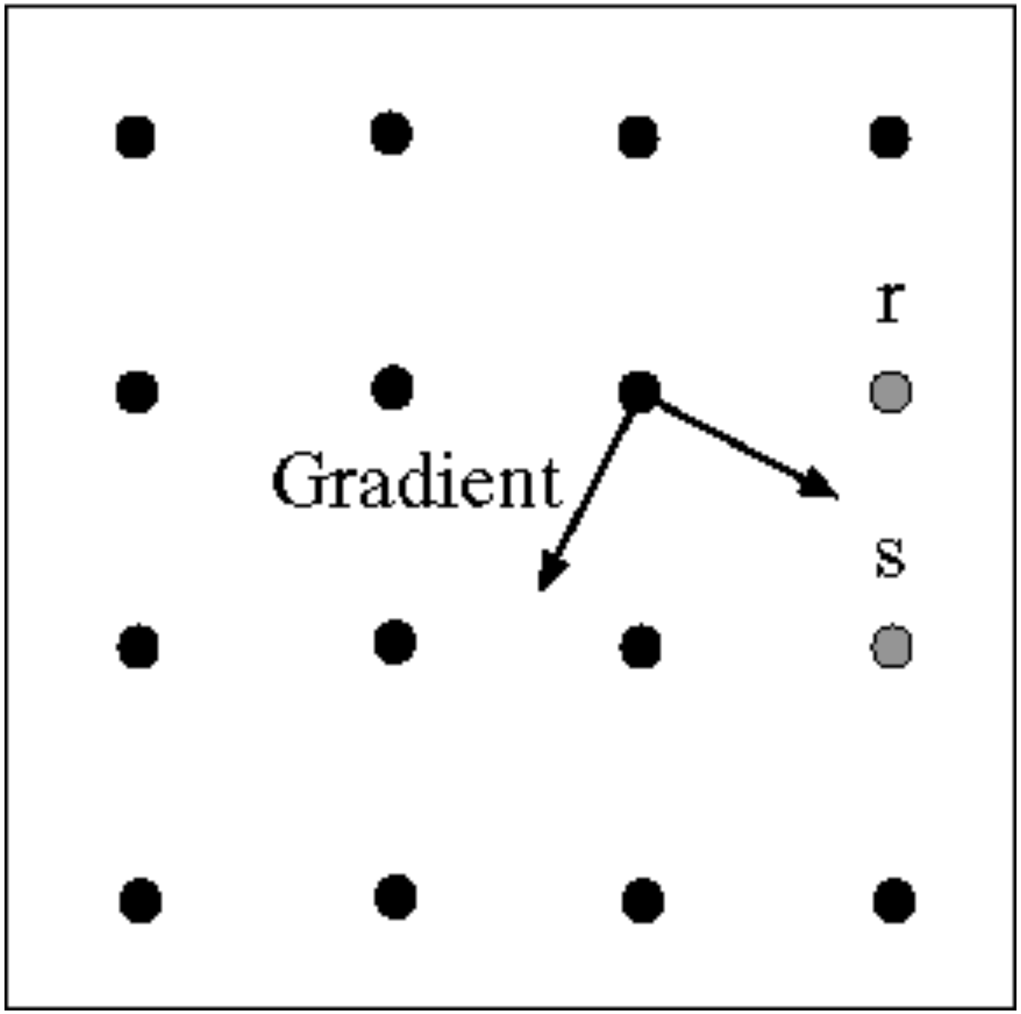


At q, we have a maximum if the value is larger than those at both p and at r. Interpolate to get these values.

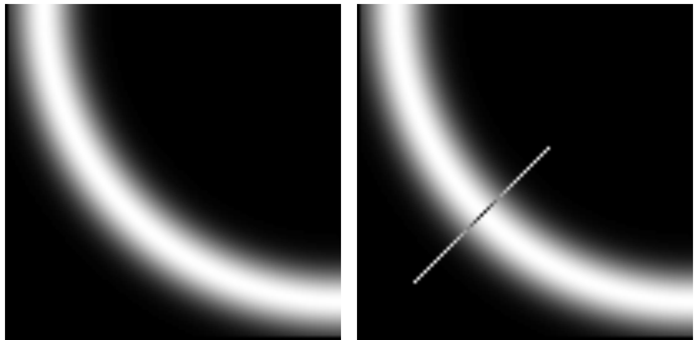


Source: D. Forsyth

Edge Linking



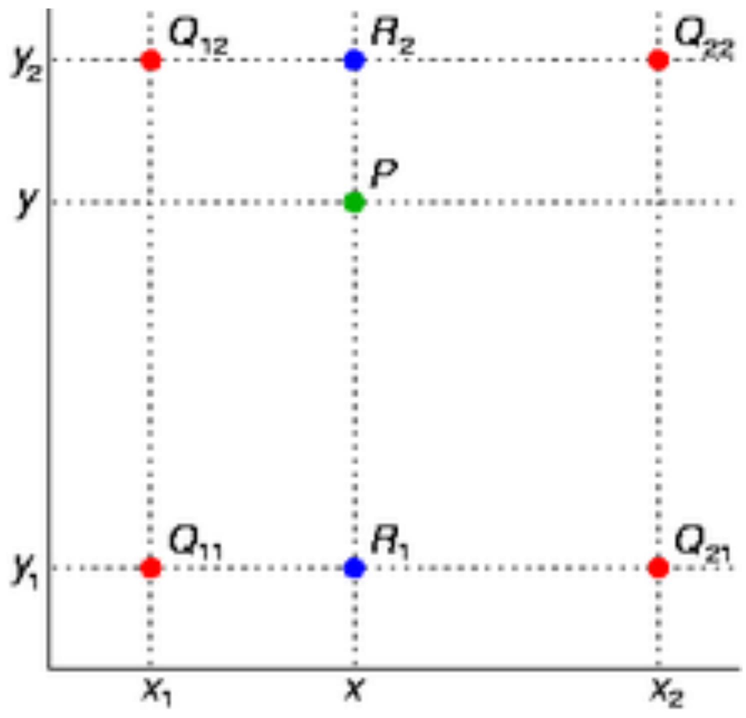
Assume the marked point is an edge point. Then we construct the tangent to the edge curve (which is normal to the gradient at that point) and use this to predict the next points (here either r or s).

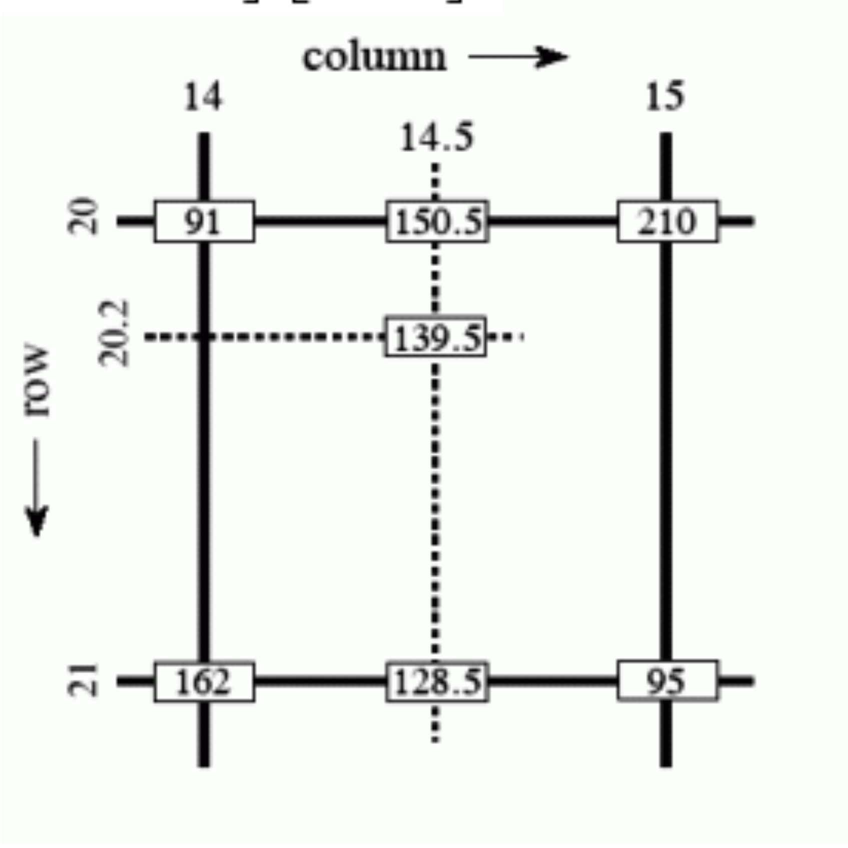


Source: D. Forsyth

Sidebar: Bilinear Interpolation

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



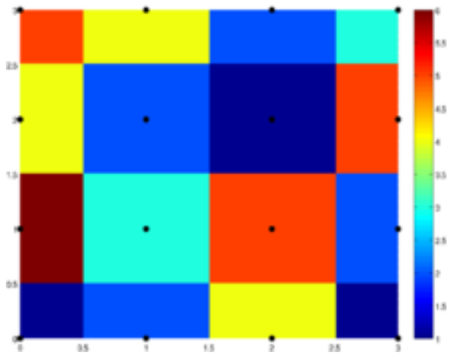


http://en.wikipedia.org/wiki/Bilinear_interpolation

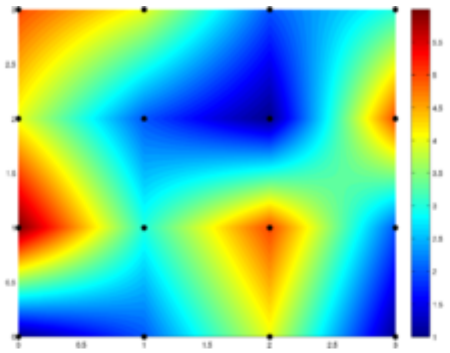
Sidebar: Interpolation Options

- imx2 = imresize(im, 2, interpolation_type)

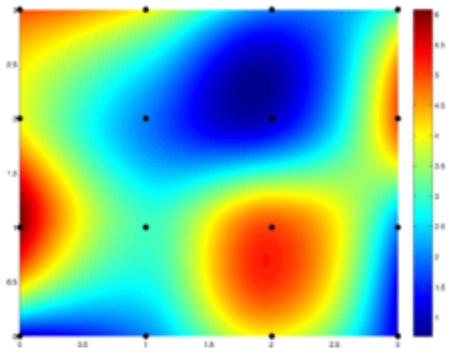
- 'nearest'
  - Copy value from nearest known
  - Very fast but creates blocky edges



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

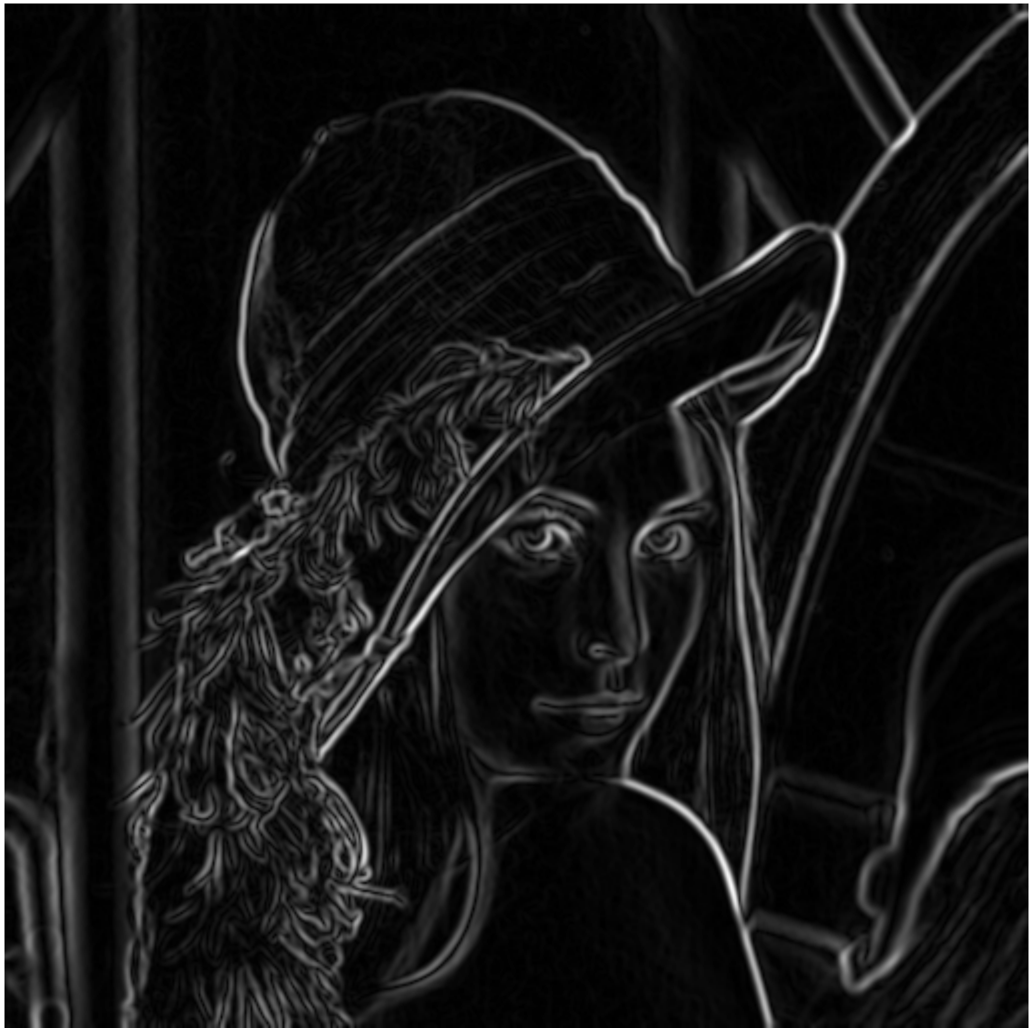


- 'bicubic' (default)
  - Non-linear smoothing over larger area (4x4)
  - Slower, visually appealing, may create negative pixel values

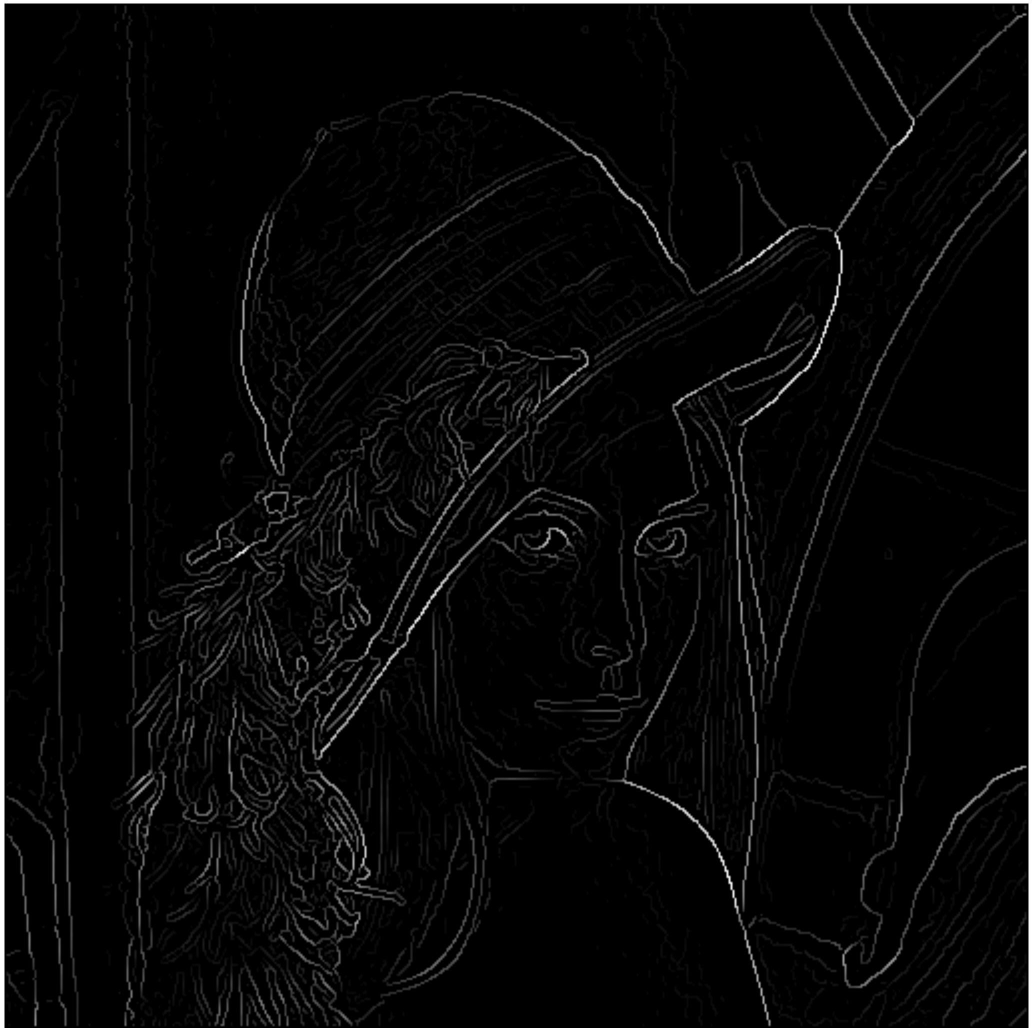


Examples from http://en.wikipedia.org/wiki/Bicubic_interpolation

Before Non-Max Suppression

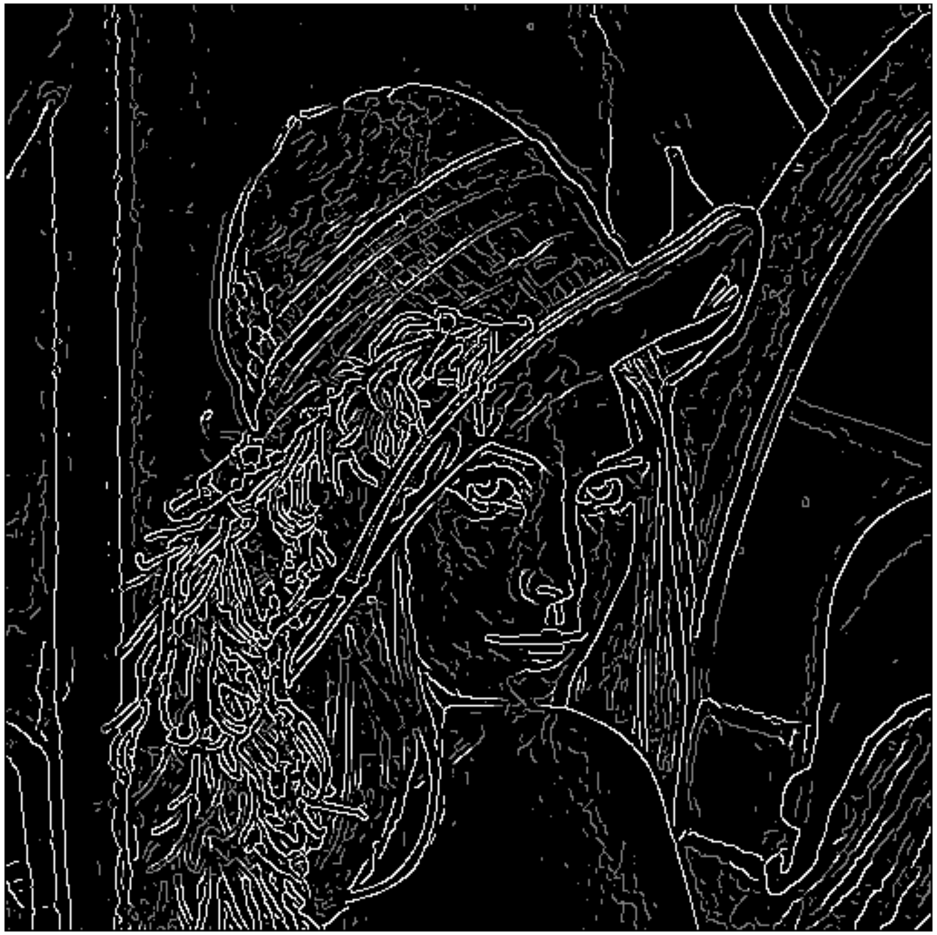


After Non-Max Suppression



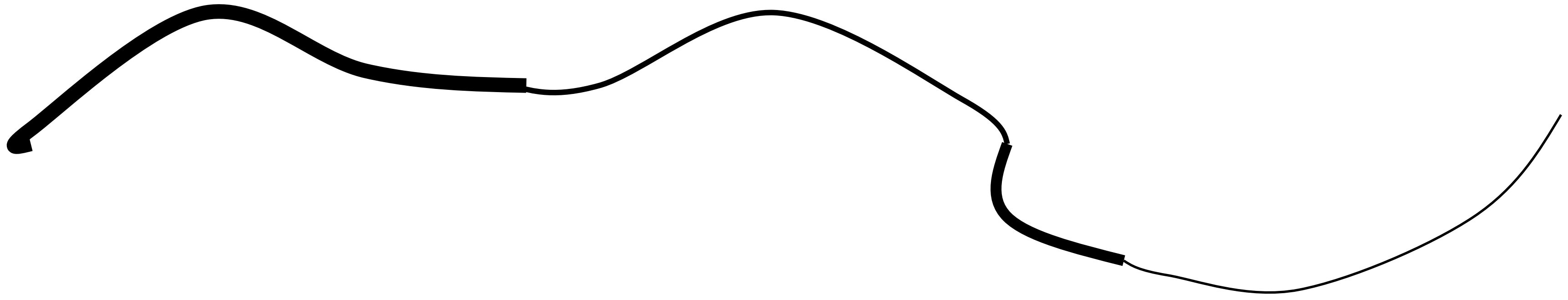
Hysteresis Thresholding

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



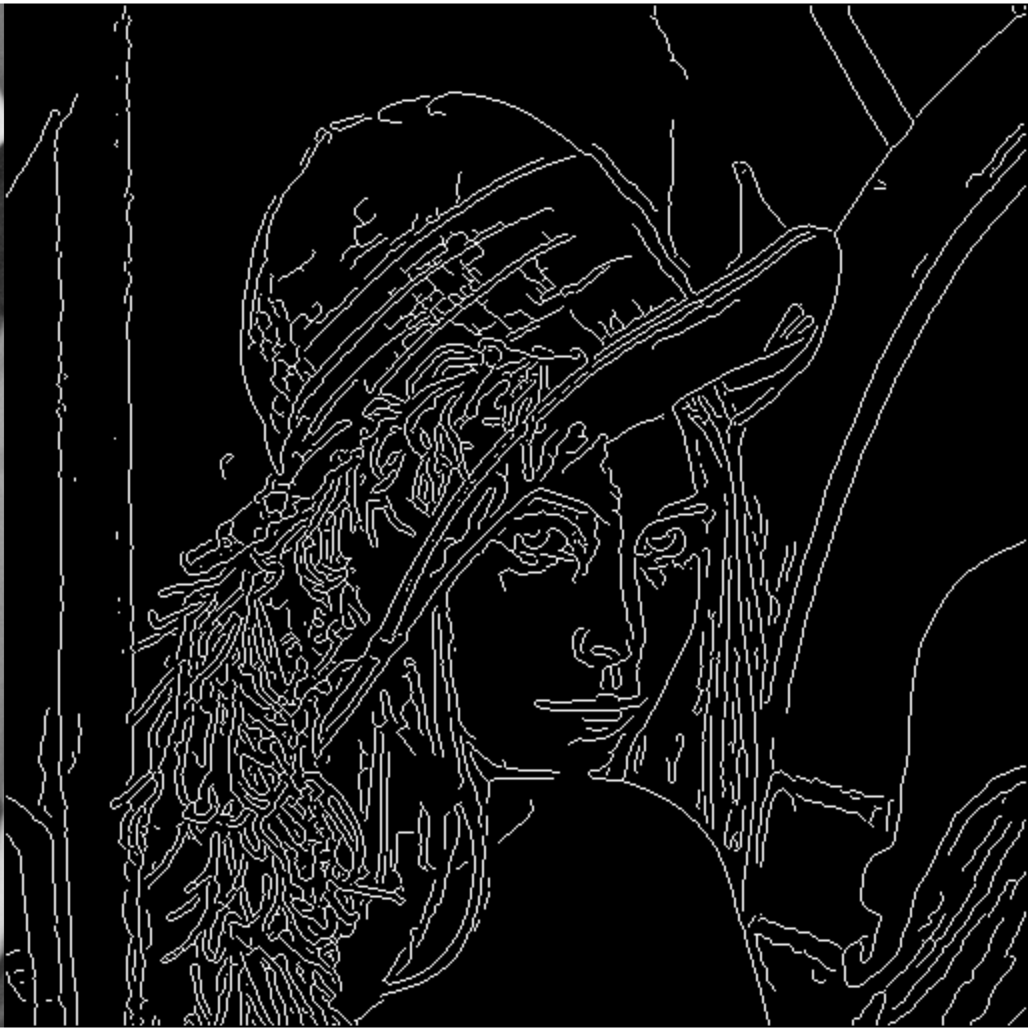
Hysteresis Thresholding

- 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



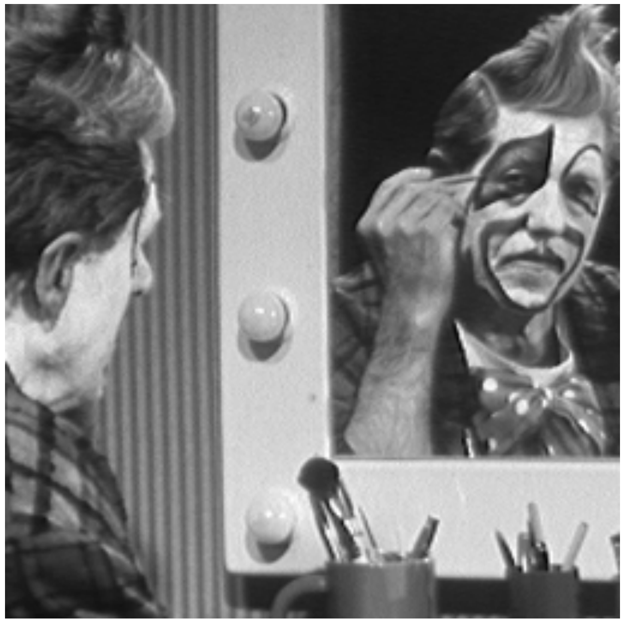
Canny Edge Detector

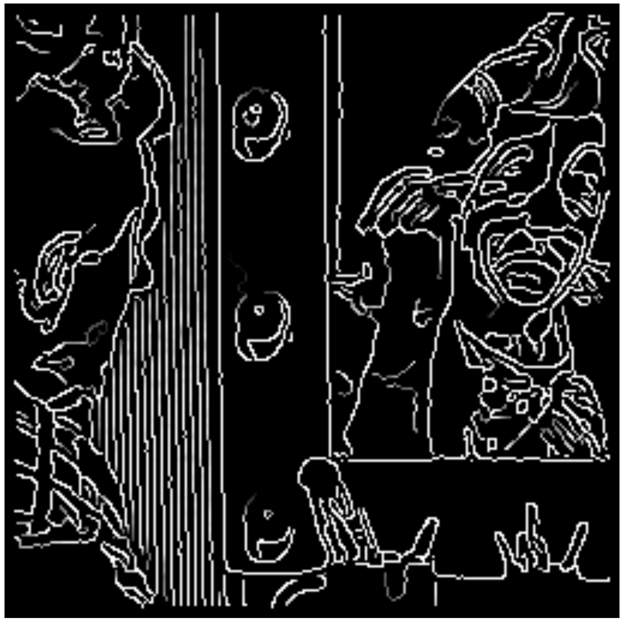
1. Filter image with x, y derivatives of Gaussian
2. Find magnitude and orientation of gradient
3. Non-maximum suppression:
   - Thin multi-pixel wide “ridges” down to single pixel width
4. Thresholding and linking (hysteresis):
   - Define two thresholds: low and high
   - Use the high threshold to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

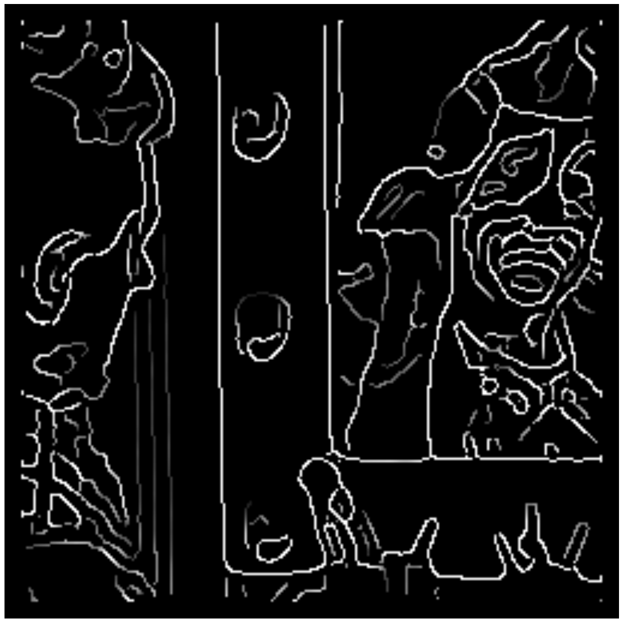
- MATLAB: edge(image, ‘canny’)

Source: D. Lowe, L. Fei-Fei

Effect of σ (Gaussian kernel size)


original


Canny with $\sigma = 1$

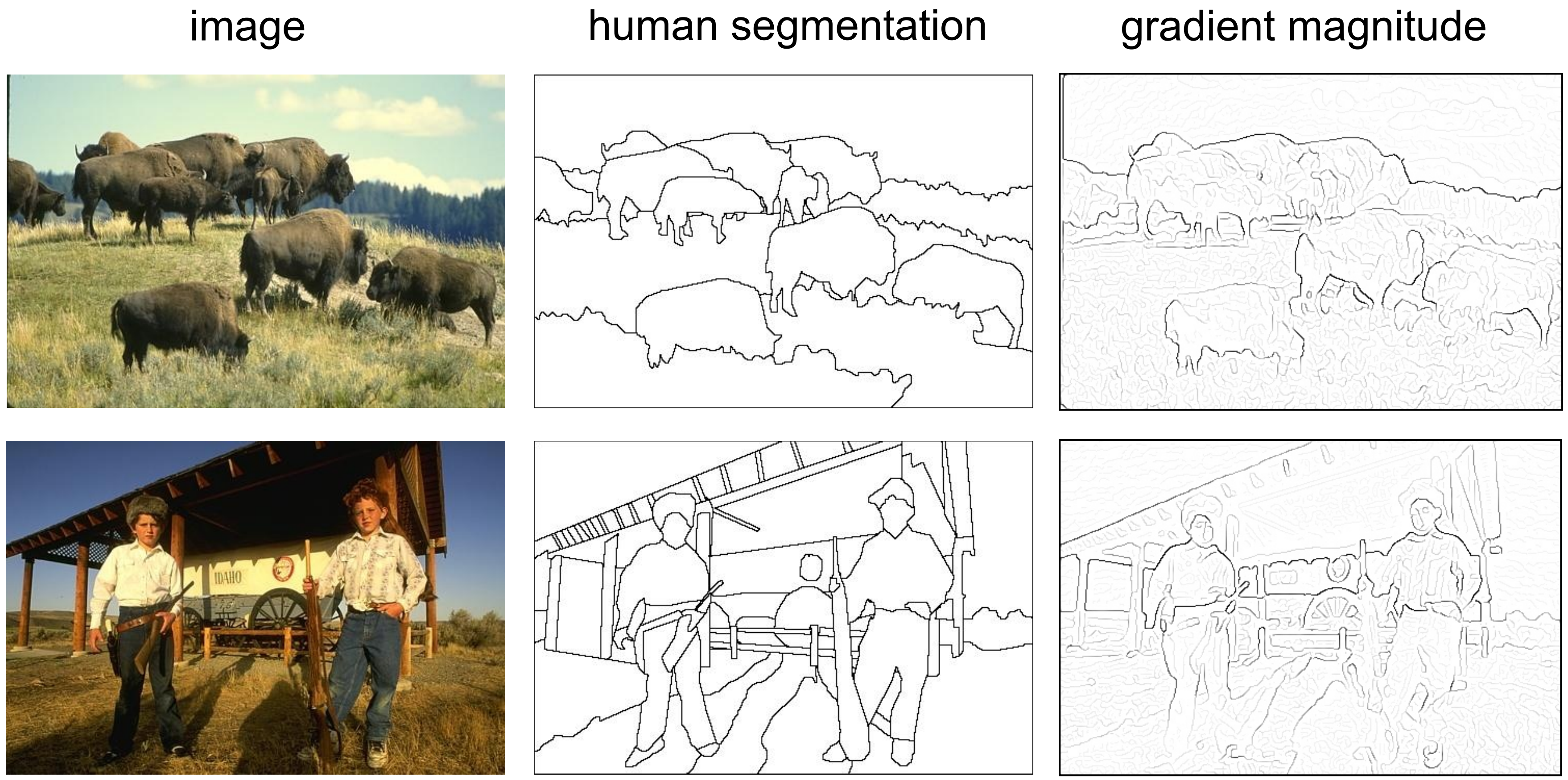

Canny with $\sigma = 2$

The choice of σ depends on desired behavior

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

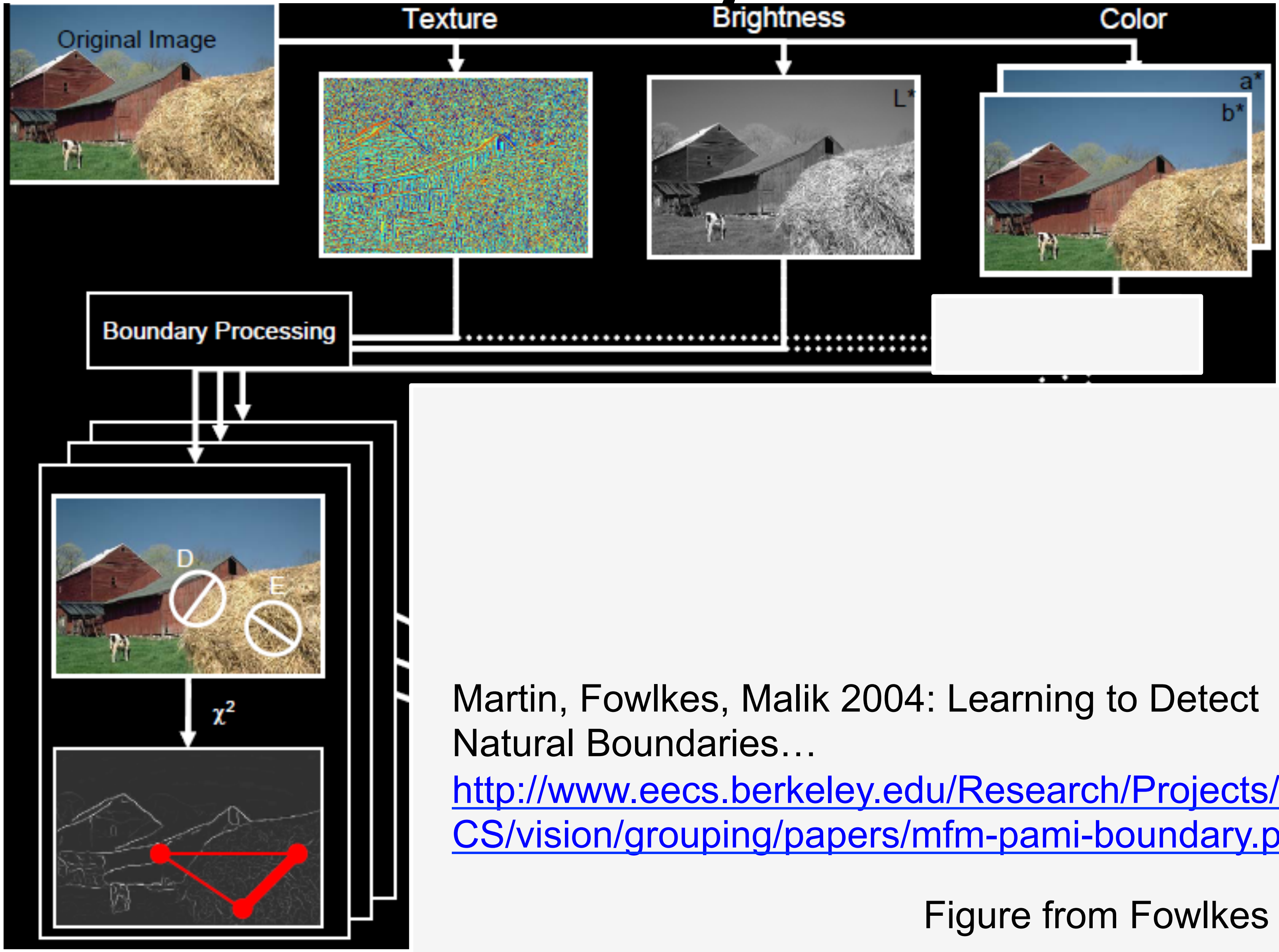
Source: S. Seitz

Where Do Humans See Boundaries?



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

Probabilistic 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

Probabilistic Boundary Detector

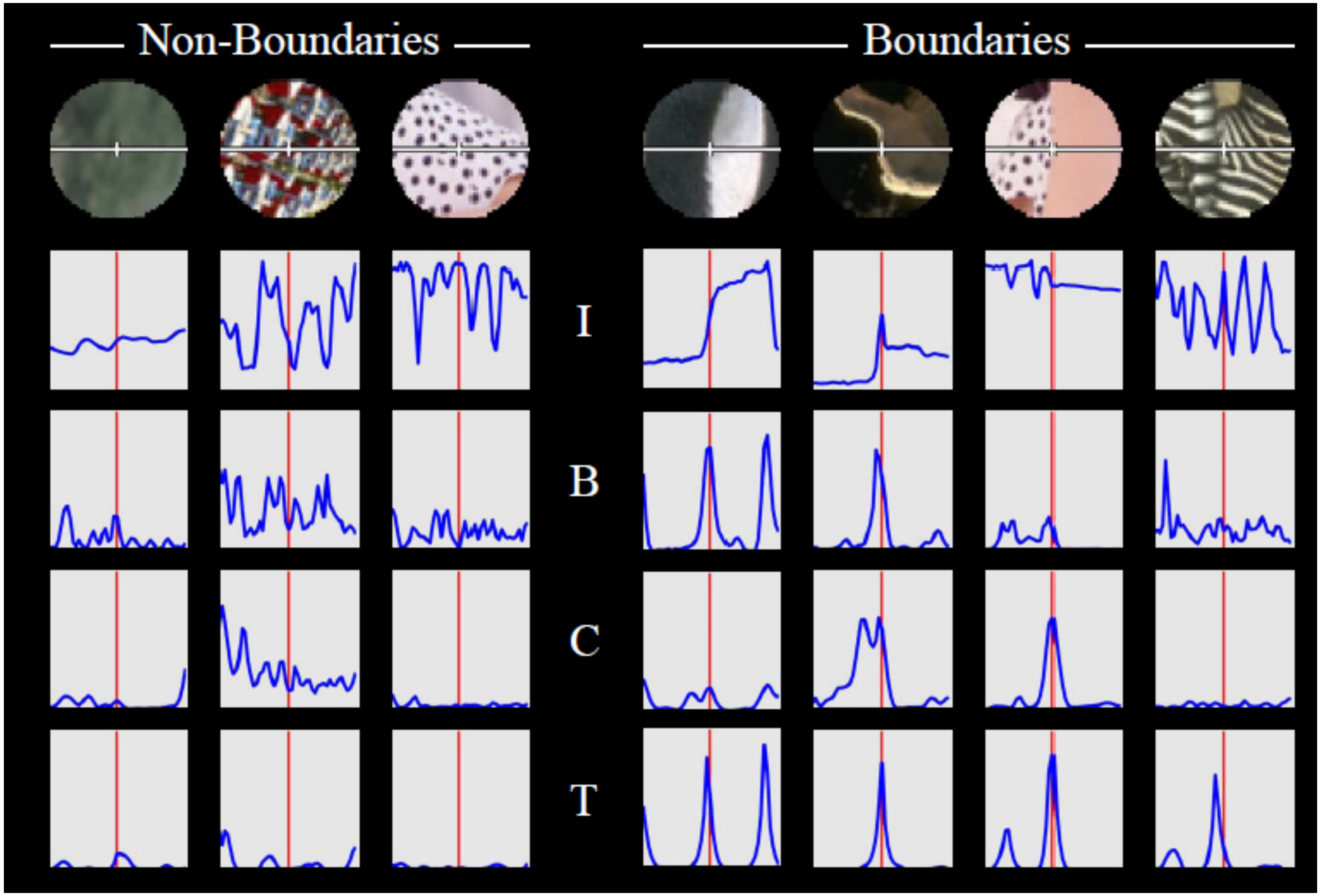


Figure from Fowlkes

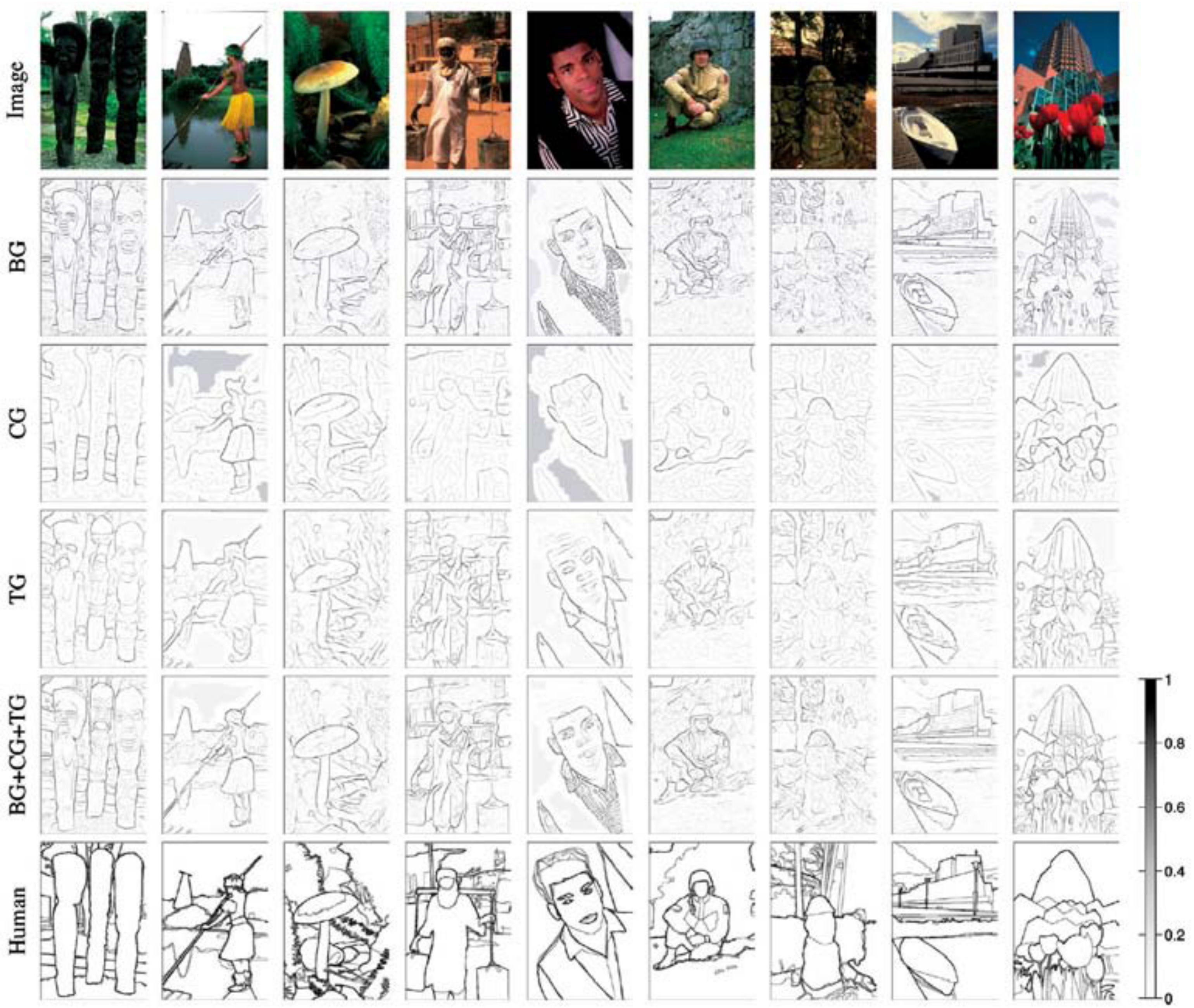
Brightness

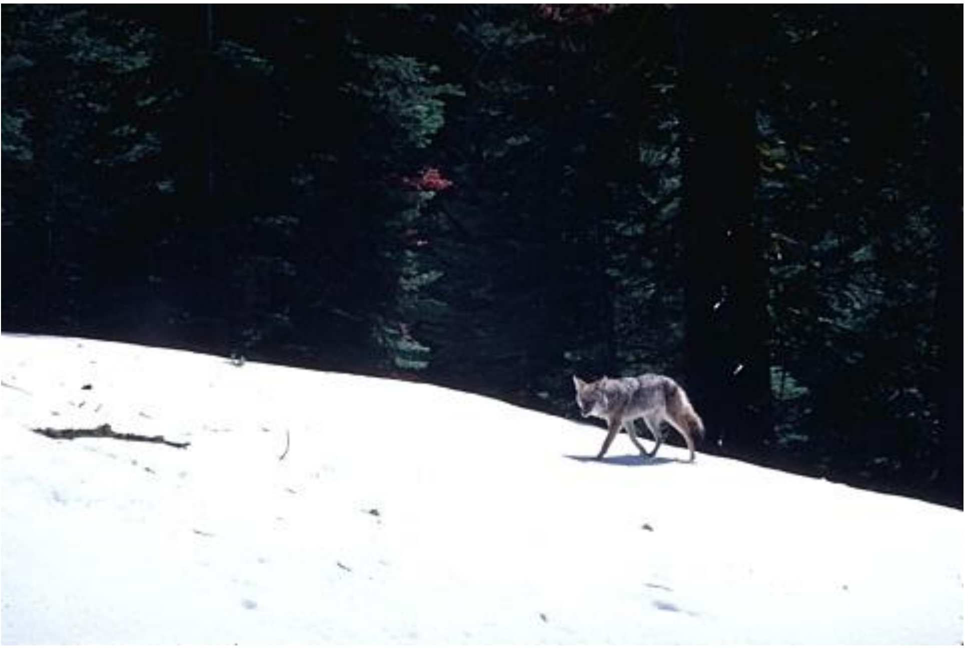
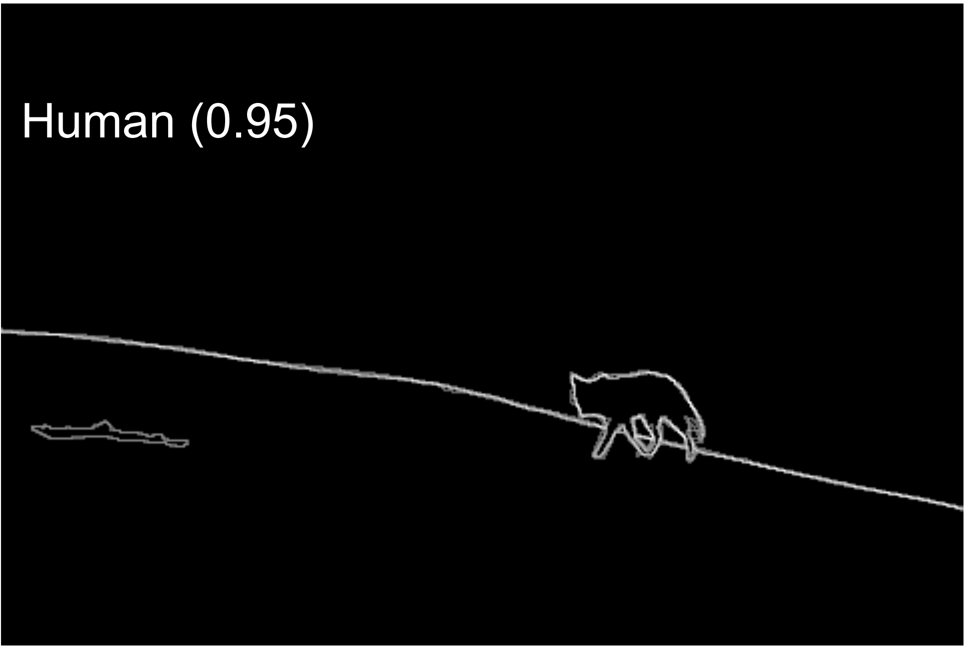
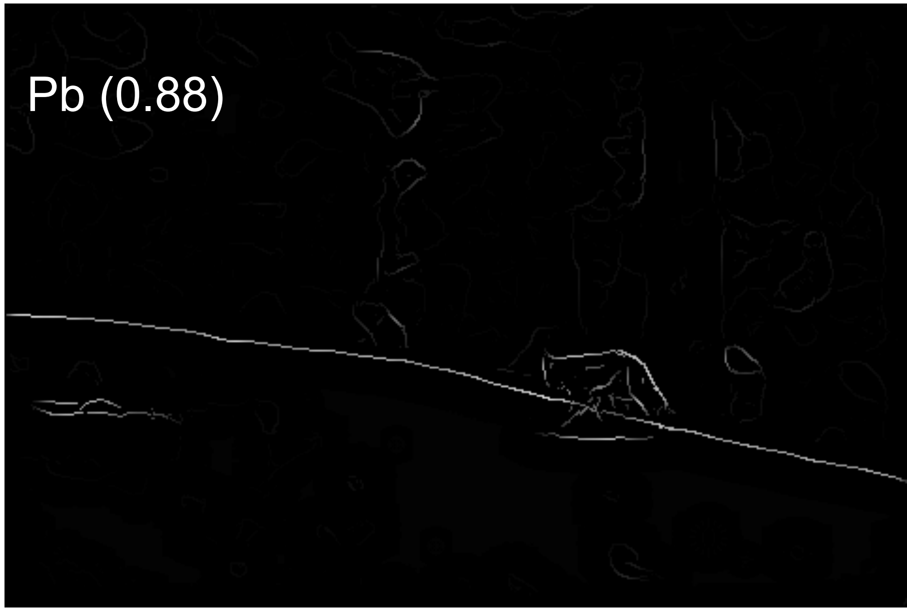
Color

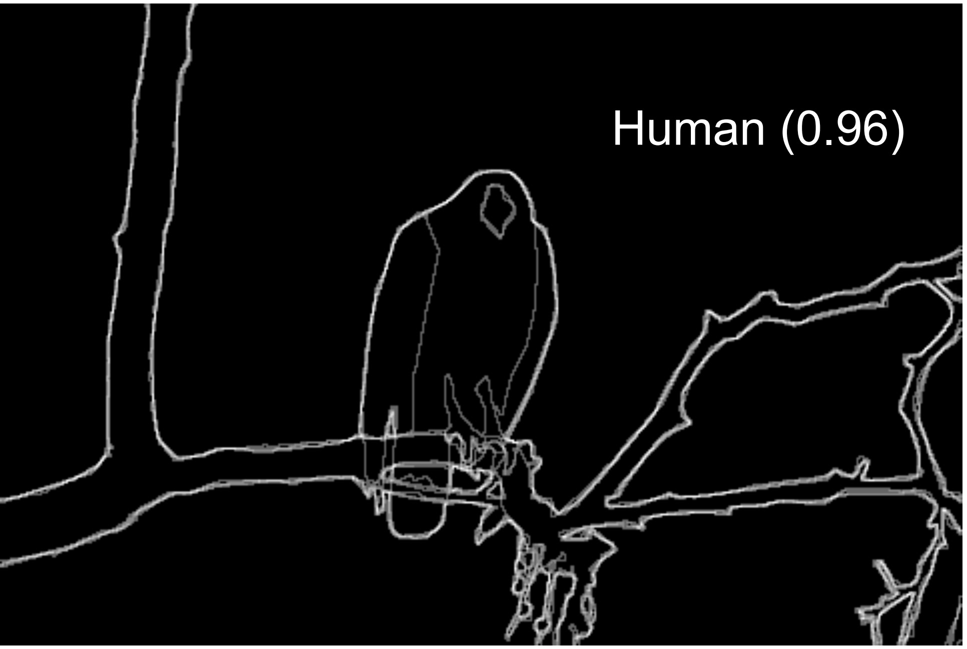
Texture

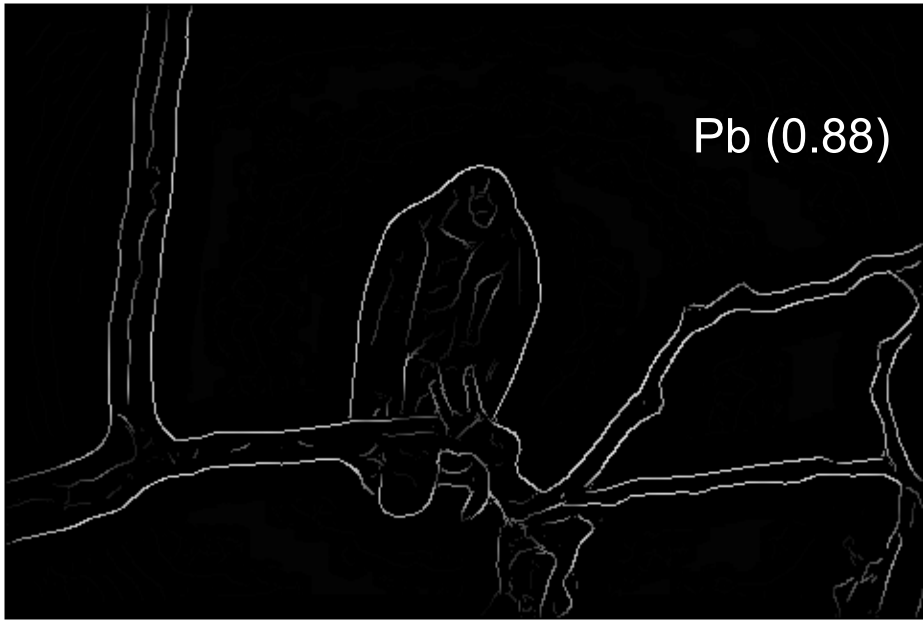
Combined

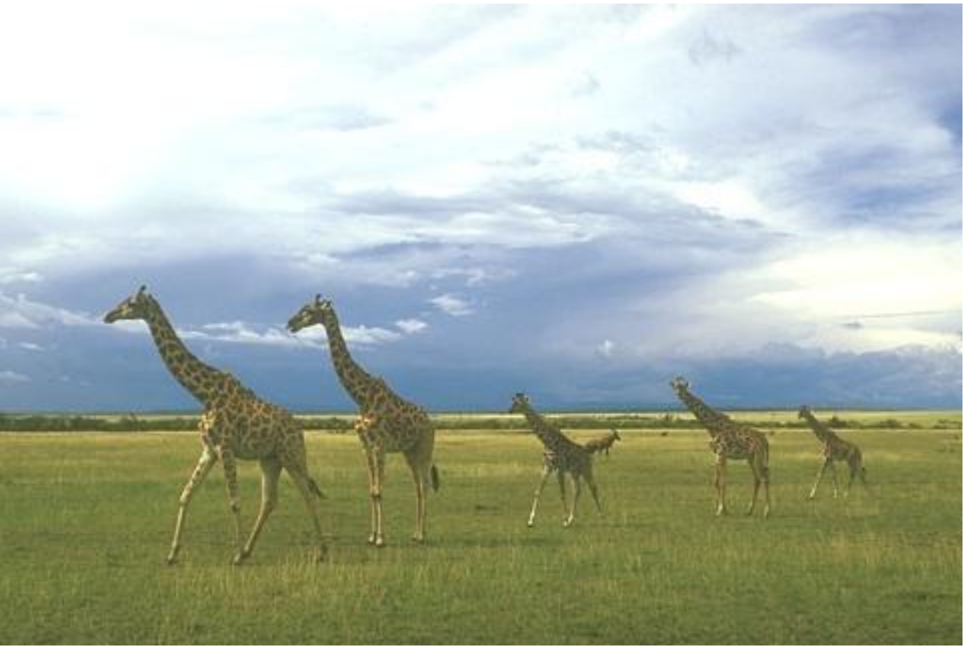
Human

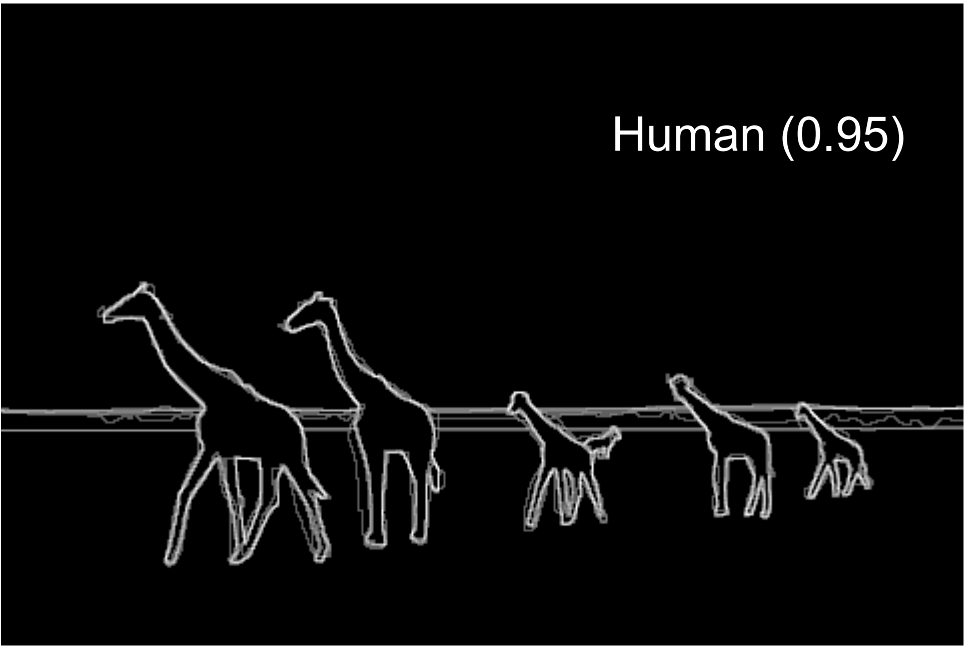


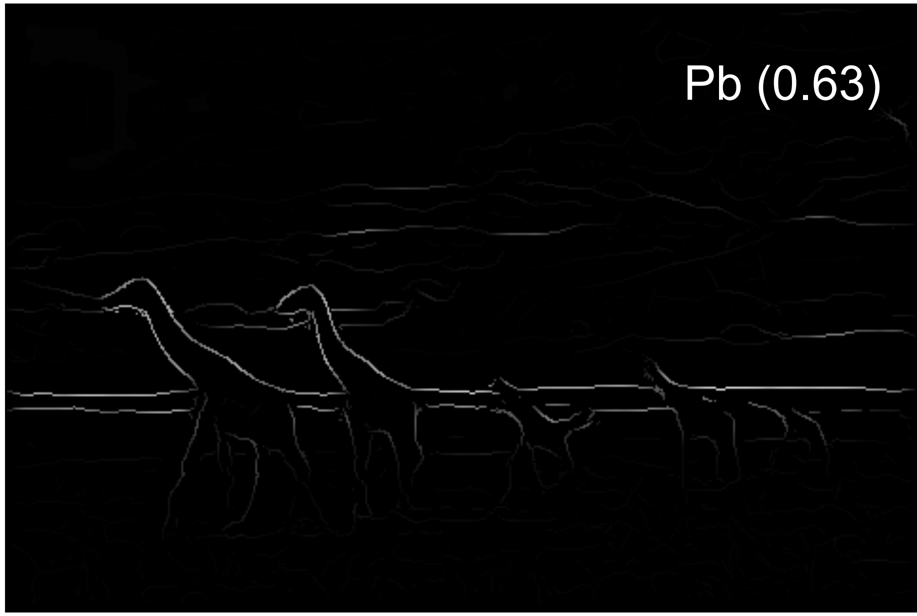


Human (0.95)

Pb (0.88)

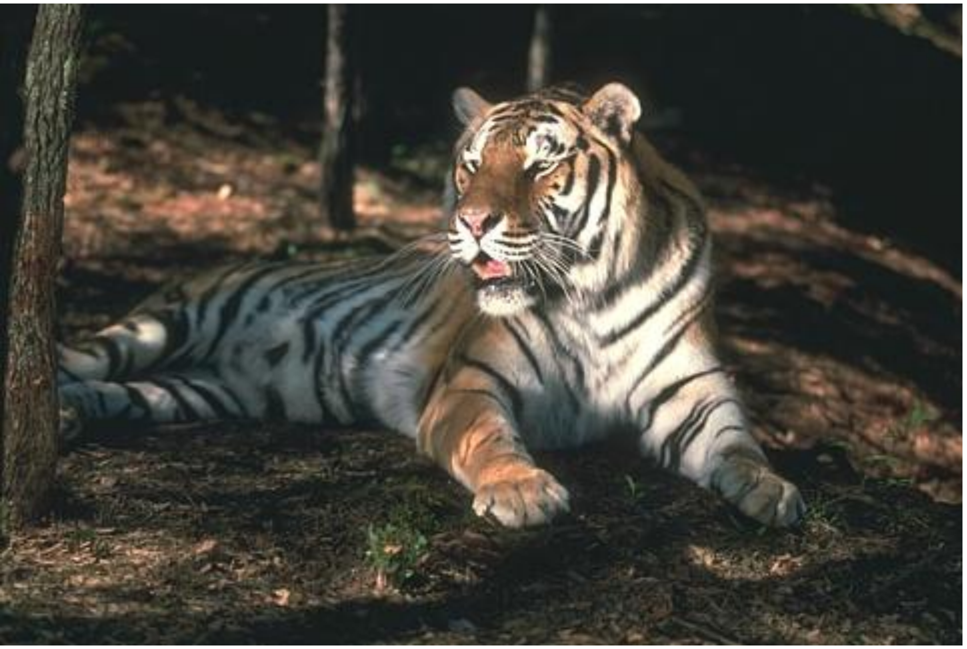

Human (0.96)

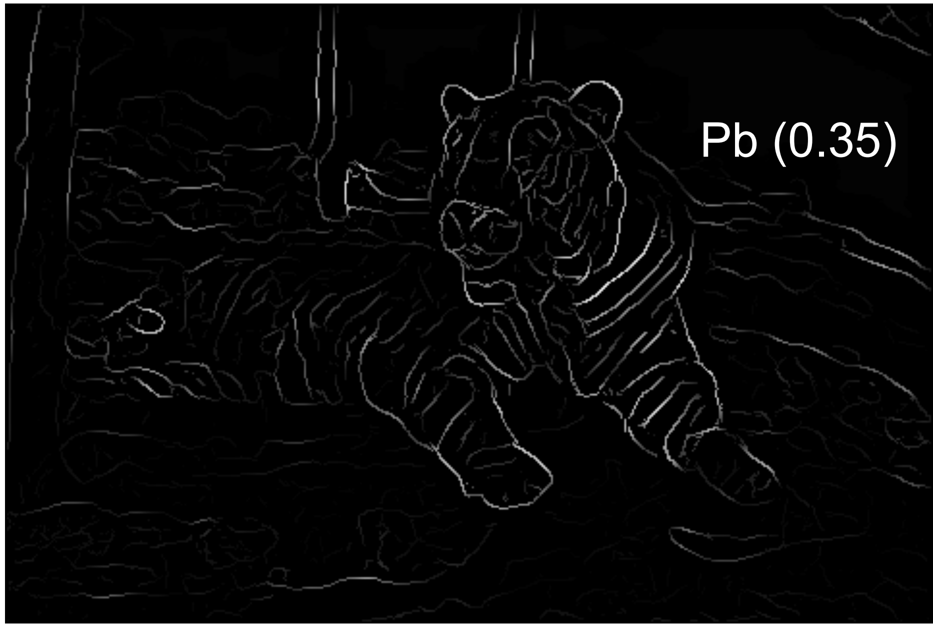

Pb (0.88)

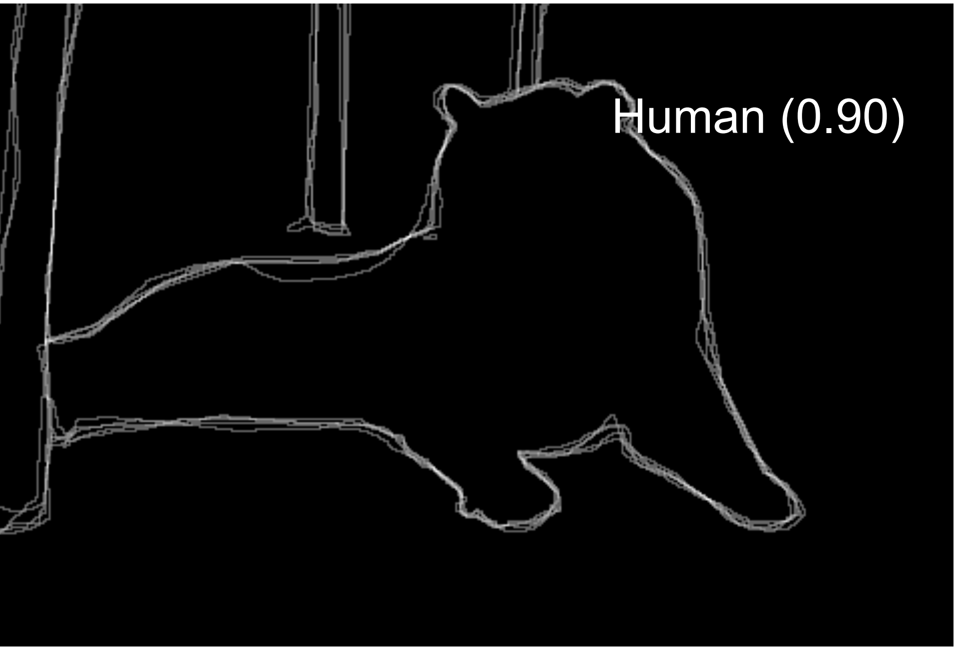



Human (0.95)


Pb (0.63)





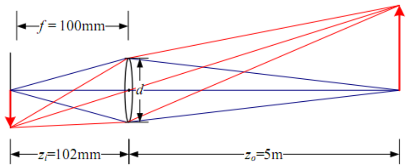


For more:
http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/bench/html/108082-

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.
- Poor use of object and high-level information

Course Outline



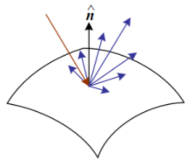
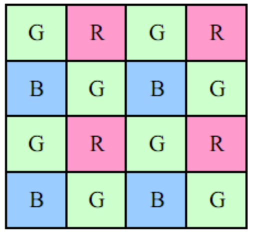


Image Formation and Processing

Light, Shape and Color

The Pin-hole Camera Model, The Digital Camera

Linear filtering, Template Matching, Image Pyramids



G	R	G	R
B	G	B	G
G	R	G	R
B	G	B	G

Feature Detection and Matching

Edge Detection, Interest Points: Corners and Blobs

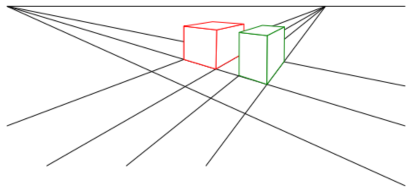
Local Image Descriptors

Feature Matching and Hough Transform

Multiple Views and Motion

Geometric Transformations, Camera Calibration

Feature Tracking , Stereo Vision



Segmentation and Grouping

Segmentation by Clustering, Region Merging and Growing

Advanced Methods Overview: Active Contours, Level-Sets, Graph-Theoretic Methods

Detection and Recognition

Problems and Architectures Overview

Statistical Classifiers, Bag-of-Words Model, Detection by Sliding Windows

Resources

Books

R. Szeliski, Computer Vision: Algorithms and Applications, 2010 – *available online*

D. A. Forsyth and J. Ponce, Computer Vision: A Modern Approach, 2003

L. G. Shapiro and G. C. Stockman, Computer Vision, 2001

Web

CVonline: The Evolving, Distributed, Non-Proprietary, On-Line Compendium of Computer Vision

http://homepages.inf.ed.ac.uk/rbf/CVonline/

Dictionary of Computer Vision and Image Processing

http://homepages.inf.ed.ac.uk/rbf/CVDICT/

Computer Vision Online

http://www.computervisiononline.com/

Programming

Development environments/languages: Matlab, Python and C/C++

Toolboxes and APIs: OpenCV, VLFeat Matlab Toolbox, Piotr's Computer Vision Matlab Toolbox, EasyCamCalib Software, FLANN, Point Cloud Library PCL, LibSVM, Camera Calibration Toolbox for Matlab