Grouping and Edges

Computer Vision
Fall 2018
Columbia University

Homework 2

- Posted online Monday
- **Due October 8** before class starts — no exceptions!
- Get started early — covers material up to today

Image Gradients
Review

First Derivative

$$* [-1,1] =$$

$$\frac{\partial I}{\partial x}$$

$$* [-1,1]^T =$$

$$\frac{\partial I}{\partial y}$$

Second Derivative

$$* [-1, 1] =$$

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

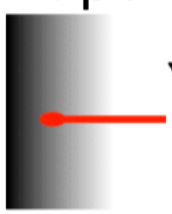
$$* [-1, 1]^T =$$

$$\frac{\partial^2 I}{\partial y^2}$$

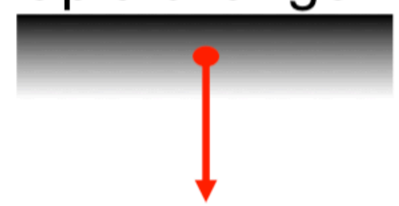
Image Gradients

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

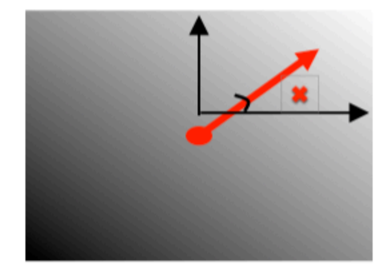
It points in the direction of most rapid change in intensity



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



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



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

The gradient direction is given by:

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

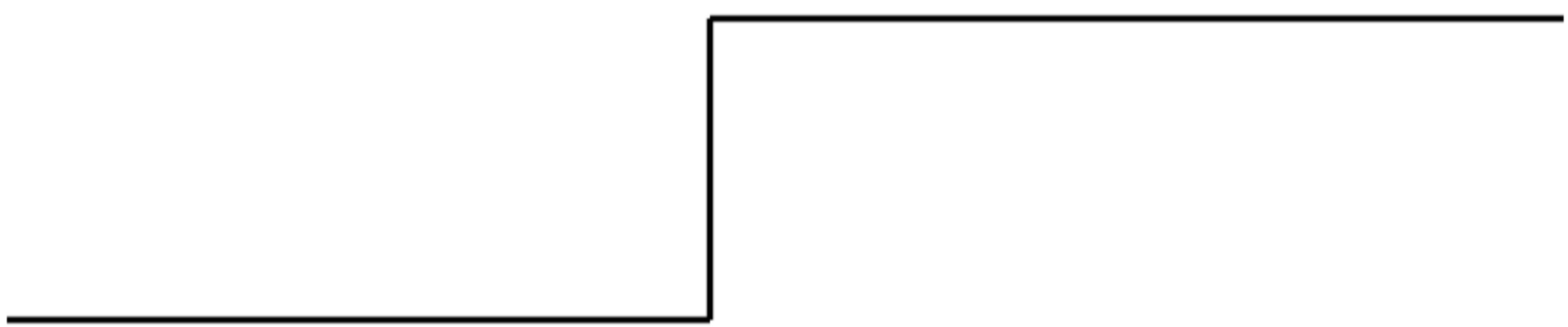
- how does this relate to the direction of the edge?

Source: Seitz and Szeliski

What is an edge?

Change is measured by derivative in 1D

- Biggest change, derivative has maximum magnitude
- Or 2^{nd} derivative is zero.

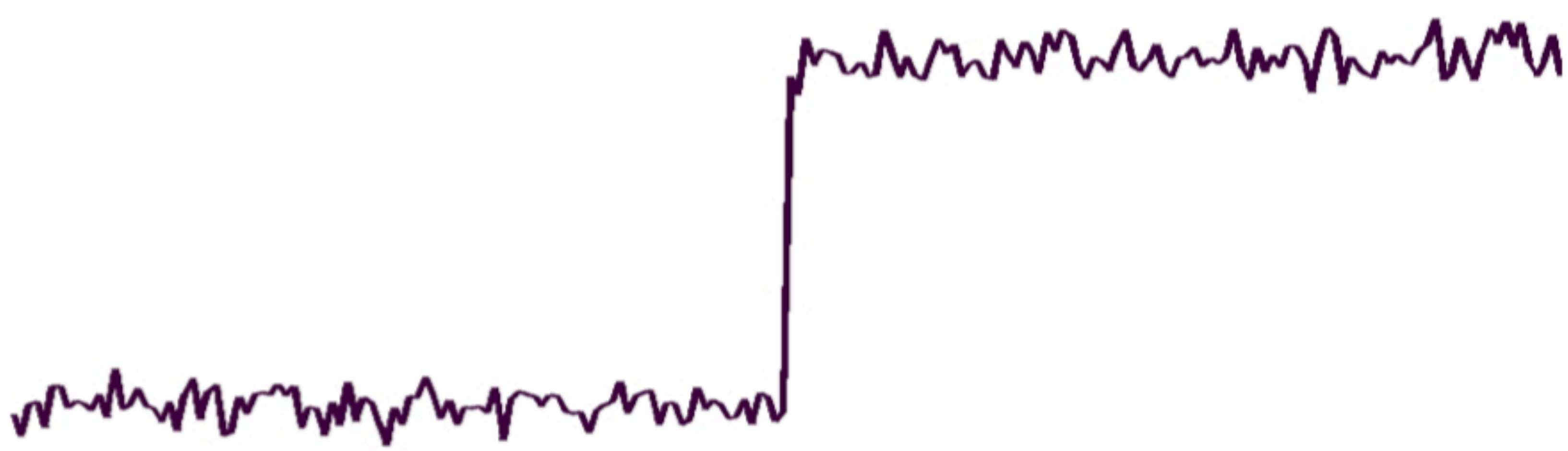


Source: G Hager

What about noise?

Derivative is high everywhere.

Must smooth before taking gradient.



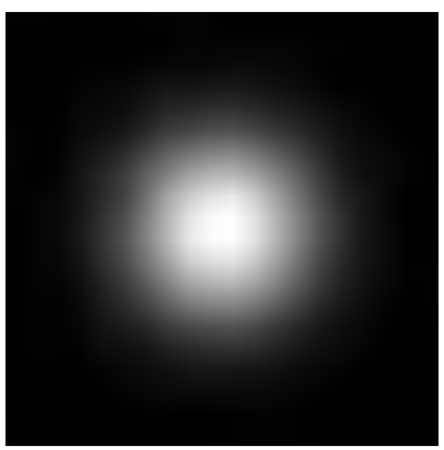
Source: G Hager

Handling Noise

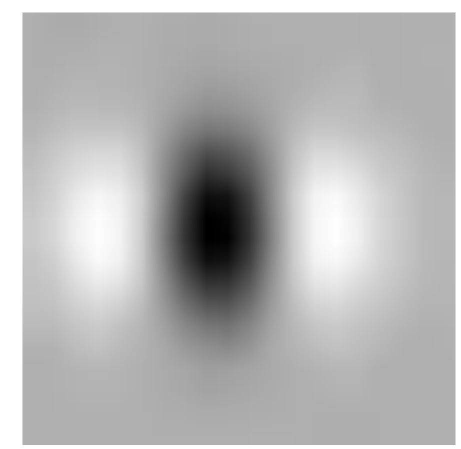
- Filter with a Gaussian to smooth, then take gradients
- But, convolution is linear

Gaussian Filter

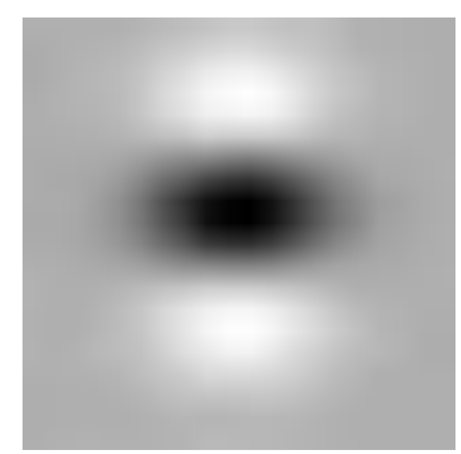
Laplacian Filter



$$* [-1,1] * [-1,1] =$$



$$* [-1,1]^T * [-1,1]^T =$$

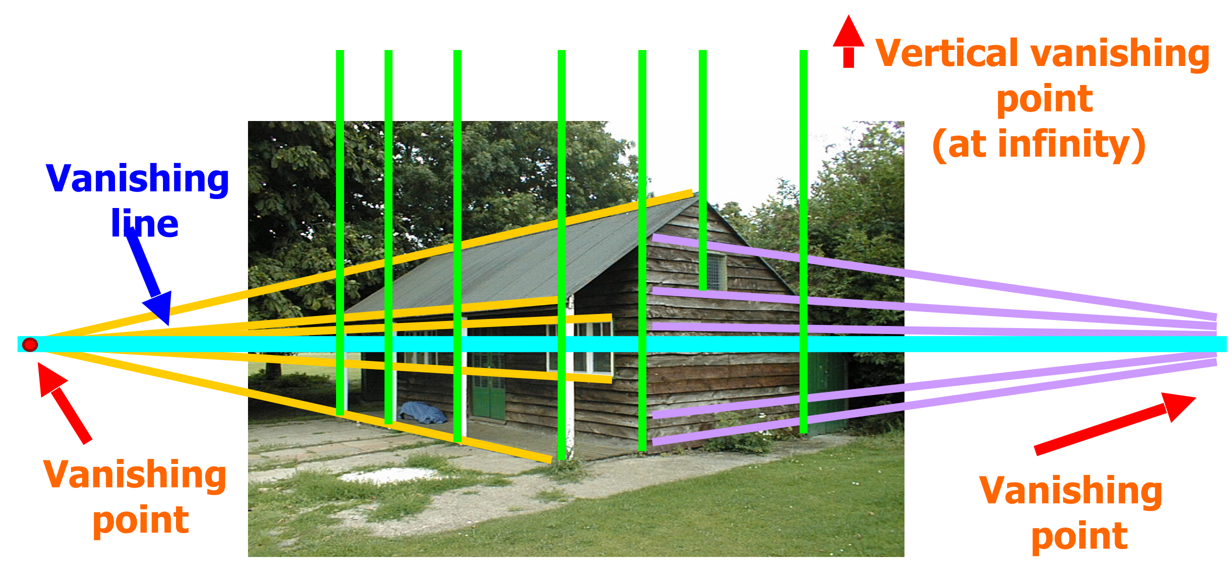



Edges

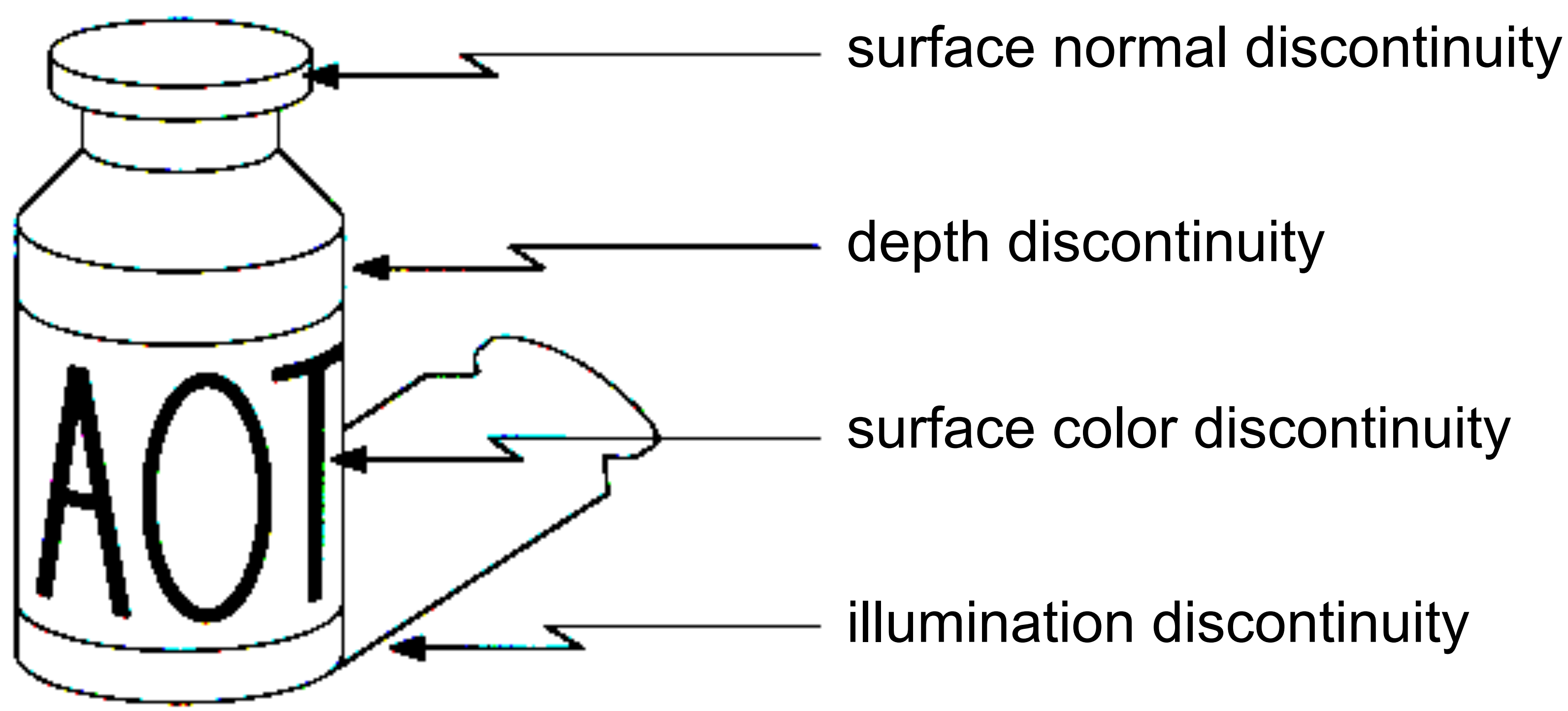
Why do we care about edges?

- Extract information
  - Recognize objects

- Help recover geometry and viewpoint



Origin of Edges



- Edges are caused by a variety of factors

Source: Steve Seitz

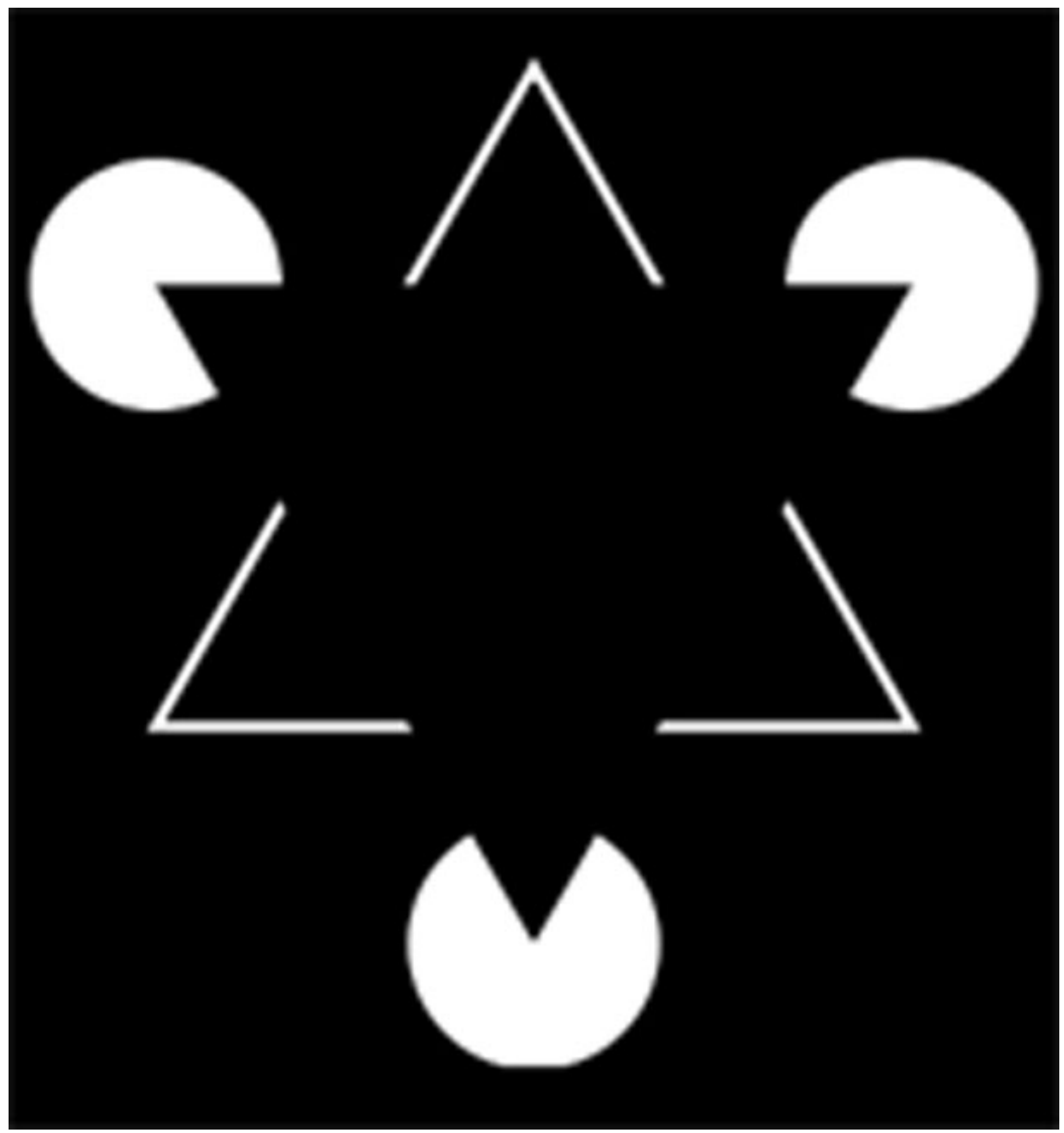
Low-level edges vs. perceived contours

Background

Texture

Shadows

Kanizsa Triangle



Low-level edges vs. perceived contours

image **human segmentation** **gradient magnitude**

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

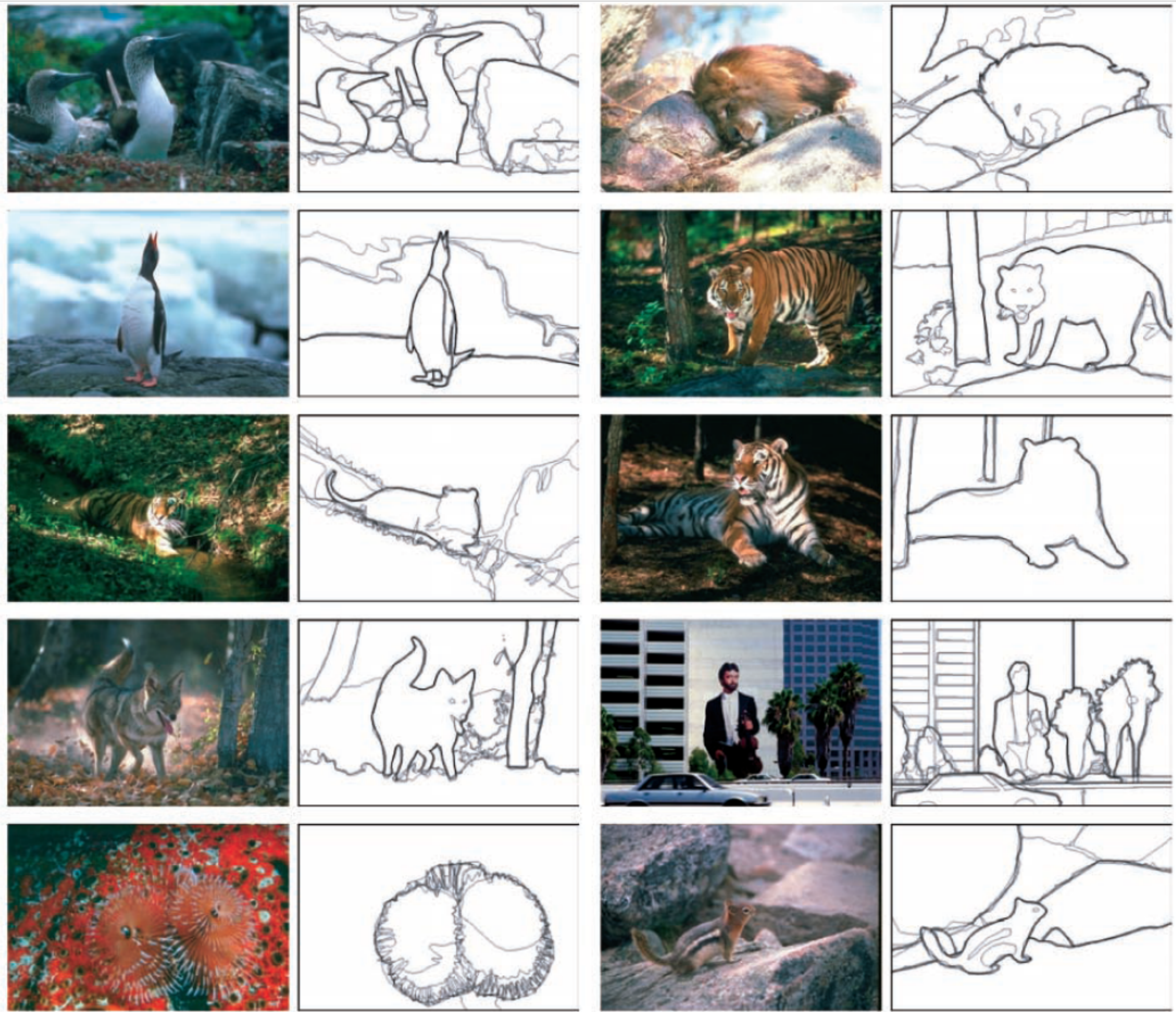
Source: L. Lazebnik

Protocol

You will be presented a photographic image. Divide the image into some number of segments, where the segments represent "things" or "parts of things" in the scene. The number of segments is up to you, as it depends on the image. Something between 2 and 30 is likely to be appropriate. It is important that all of the segments have approximately equal importance.

- Custom segmentation tool
- Subjects obtained from work-study program (UC Berkeley undergraduates)

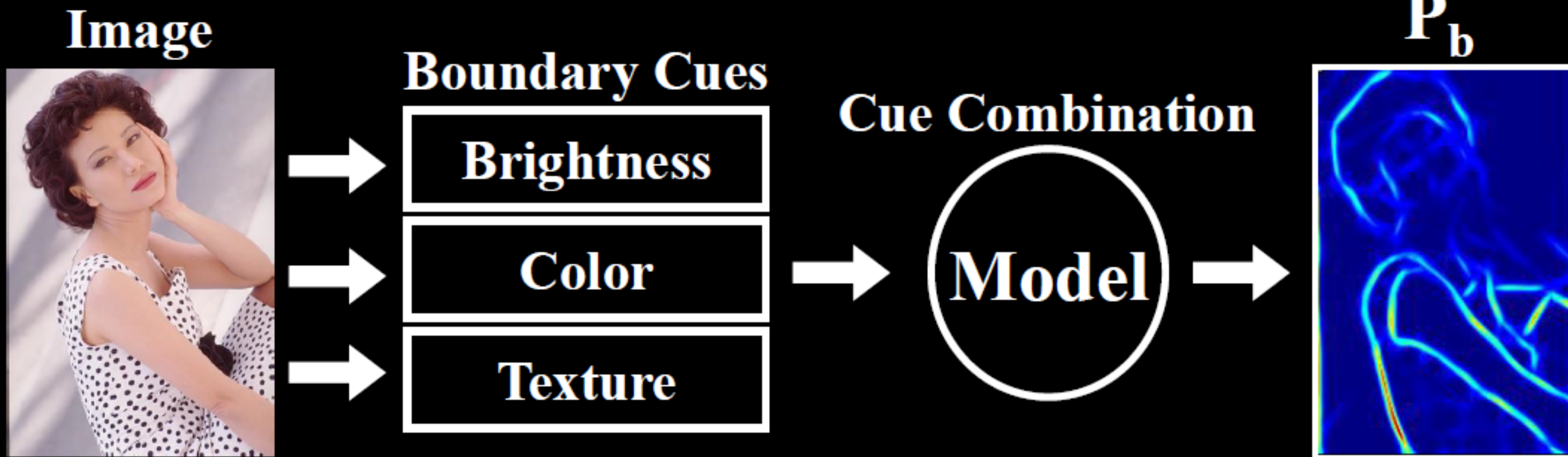
Learn from humans which combination of features is most indicative of a "good" contour?



[D. Martin et al. PAMI 2004]

Human-marked segment boundaries

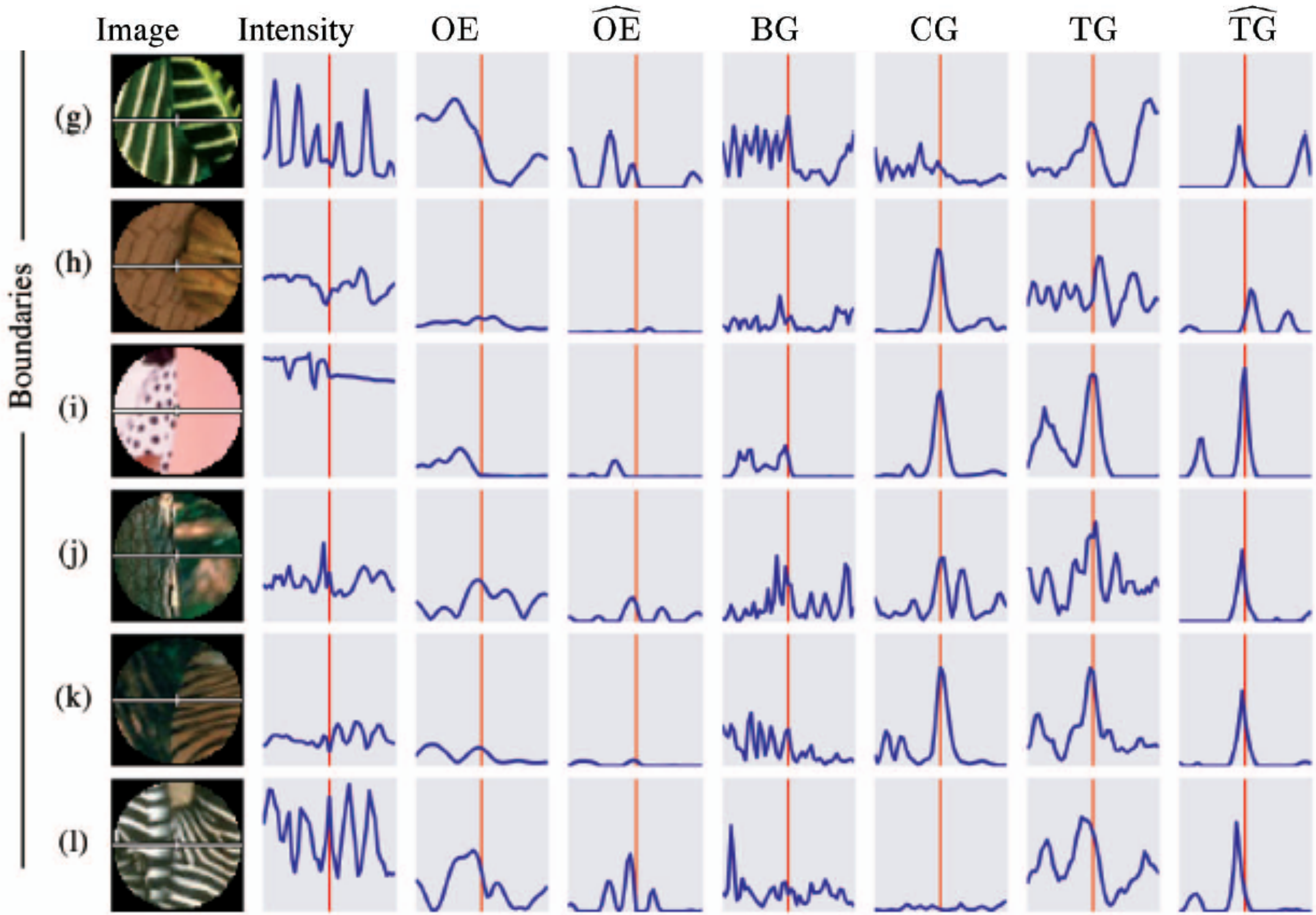
Dataflow



Challenges: texture cue, cue combination

Goal: learn the posterior probability of a boundary $P_b(x,y,\theta)$ from local information only

What features are responsible for perceived edges?

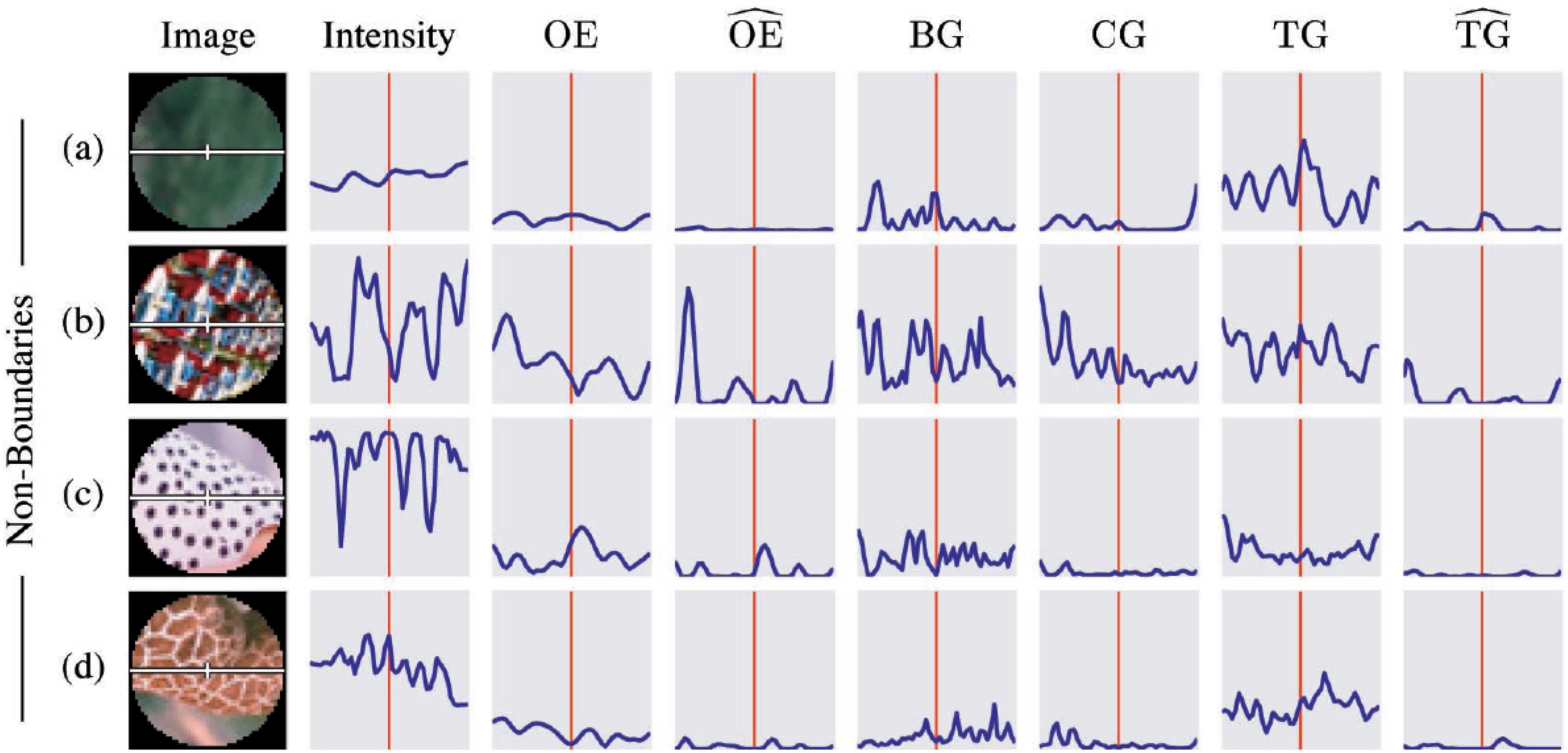


Feature profiles (oriented energy, brightness, color, and texture gradients) along the patch's horizontal diameter

[D. Martin et al. PAMI 2004]

Kristen Grauman, UT-Austin

What features are responsible for perceived edges?

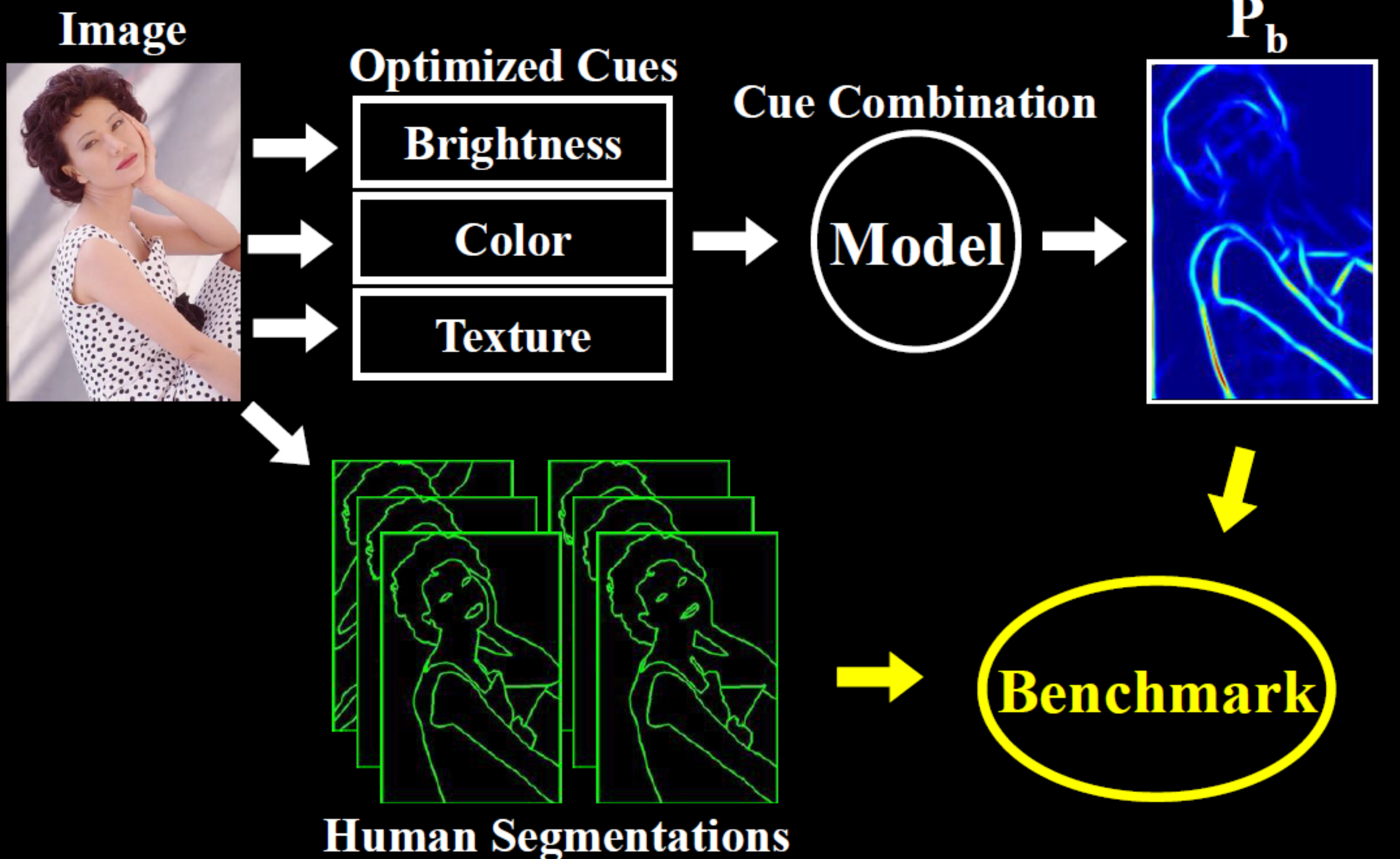


Feature profiles (oriented energy, brightness, color, and texture gradients) along the patch's horizontal diameter

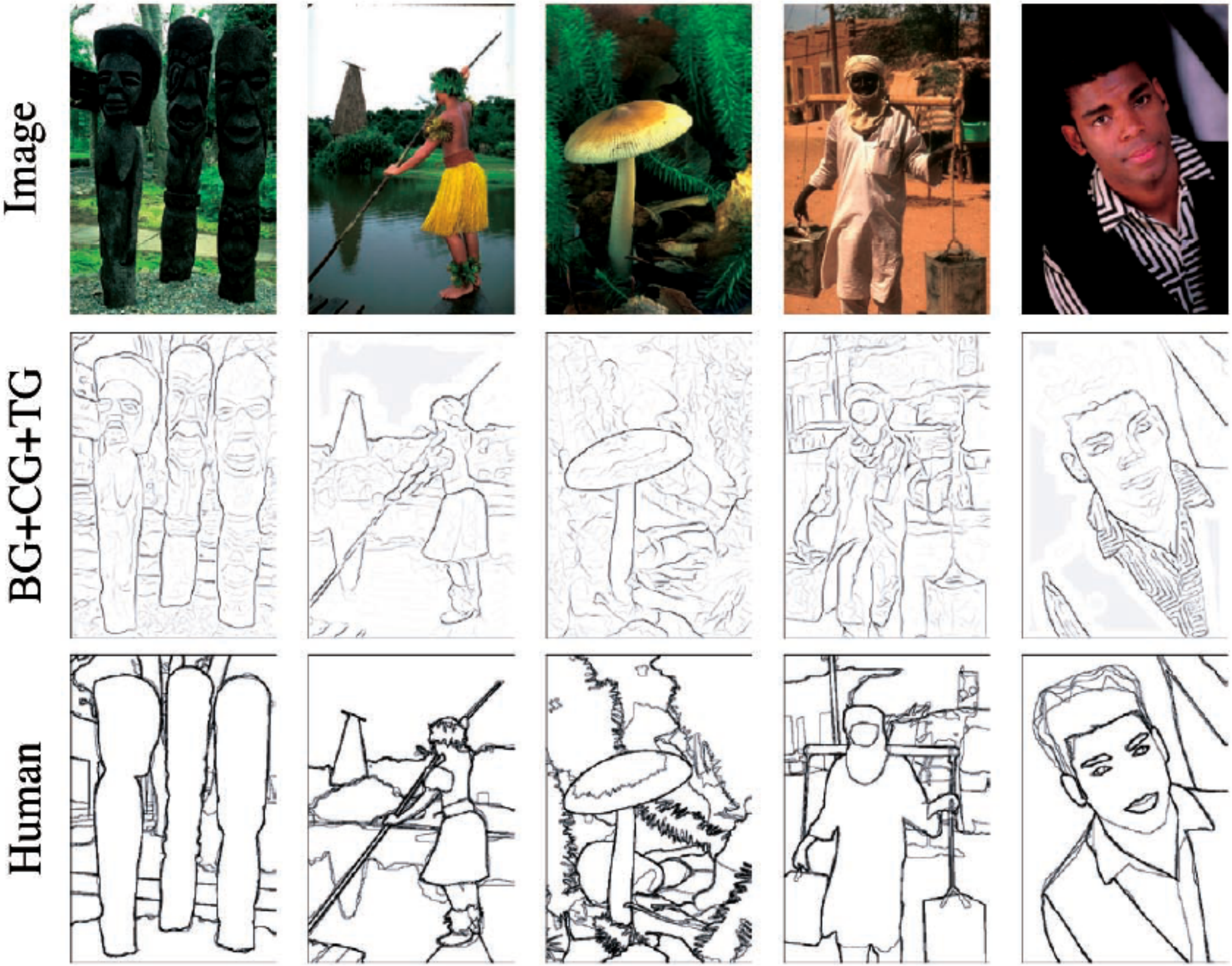
[D. Martin et al. PAMI 2004]

Kristen Grauman, UT-Austin

Dataflow



Credit: David Martin



[D. Martin et al. PAMI 2004]

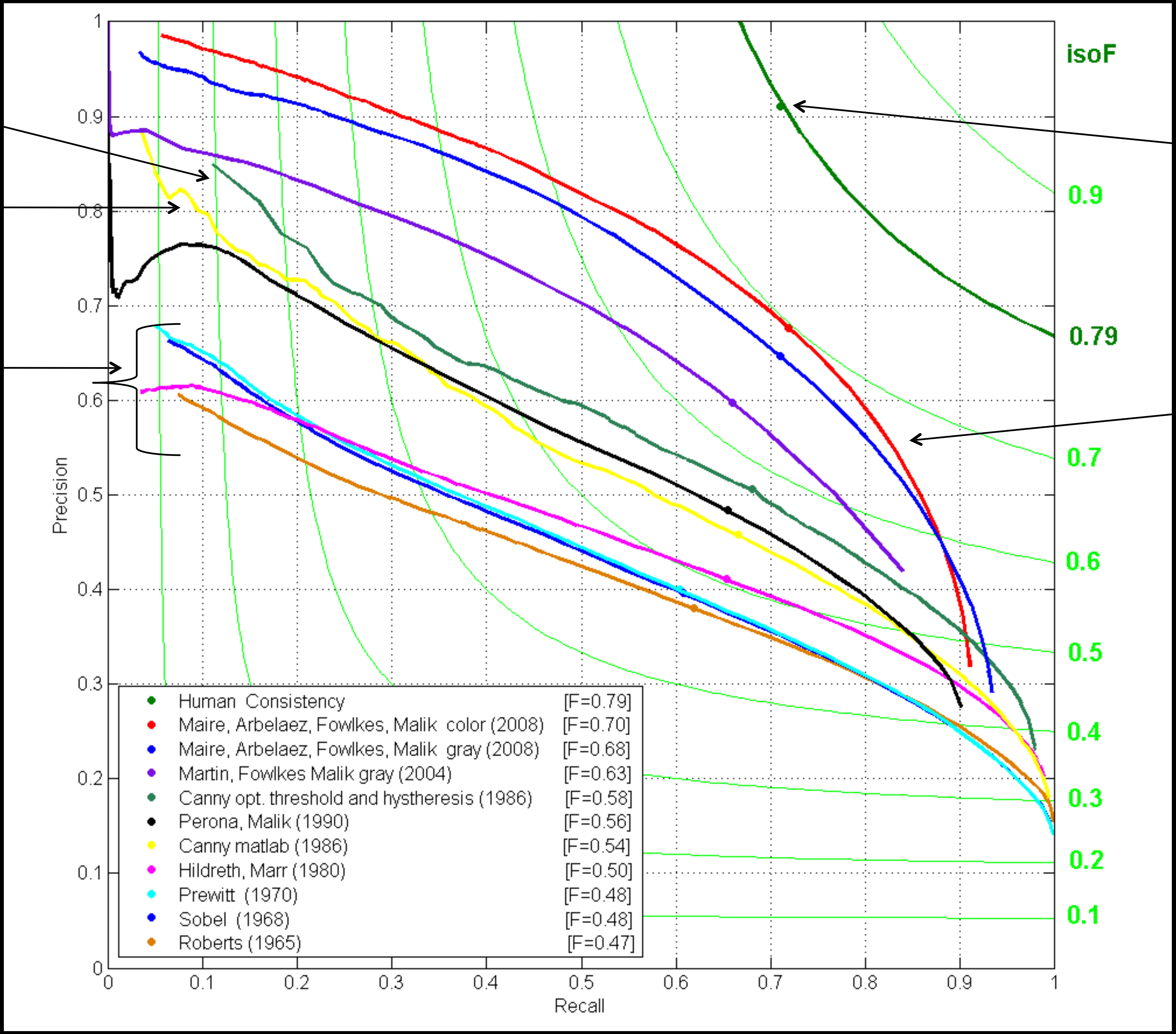
Kristen Grauman, UT-Austin

Contour Detection

Canny+opt thresholds

Canny

Prewitt, Sobel, Roberts



Human agreement

Learned with combined features

Source: Jitendra Malik: http://www.cs.berkeley.edu/~malik/malik-talks-ptrs.html

Canny Edge Detector

Widely used edge detector

John Canny’s masters thesis

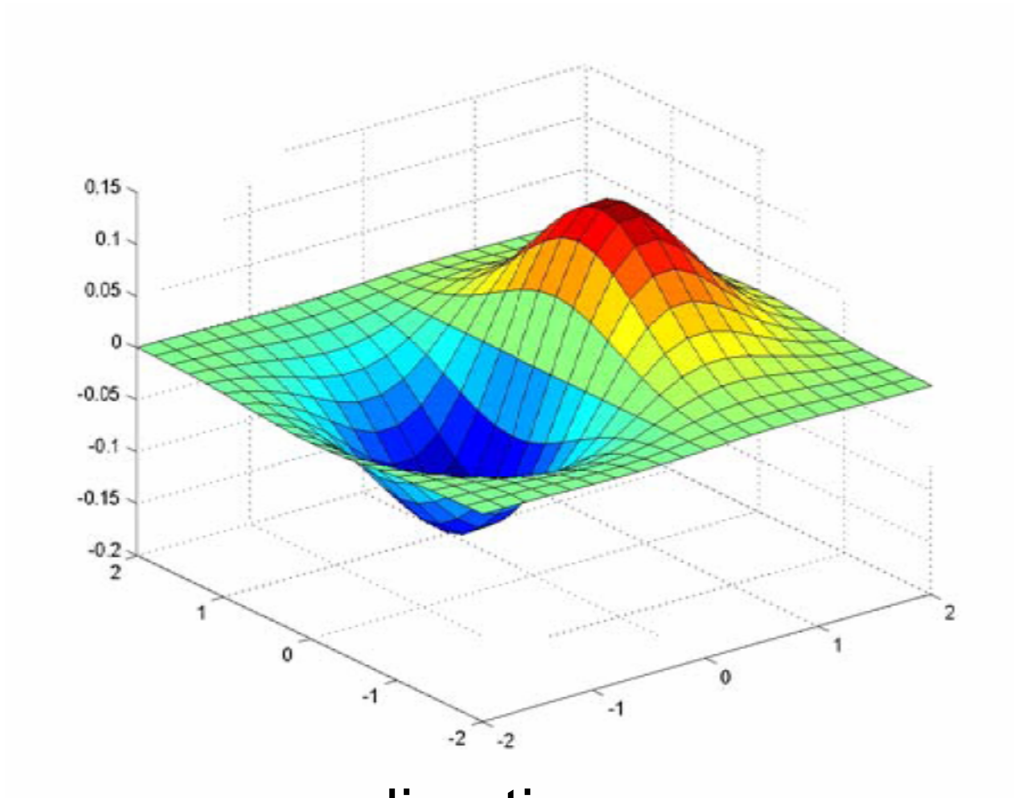
Demonstrator Image

Canny edge detector

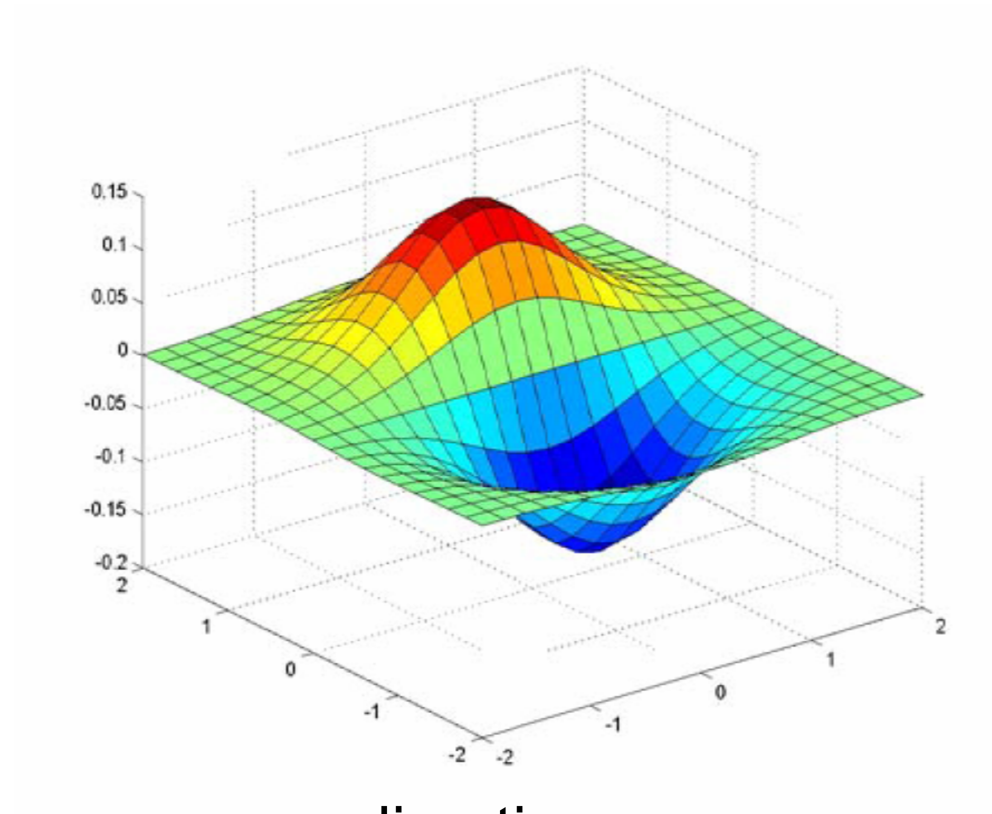
1. Filter image with x, y derivatives of Gaussian

Source: D. Lowe, L. Fei-Fei

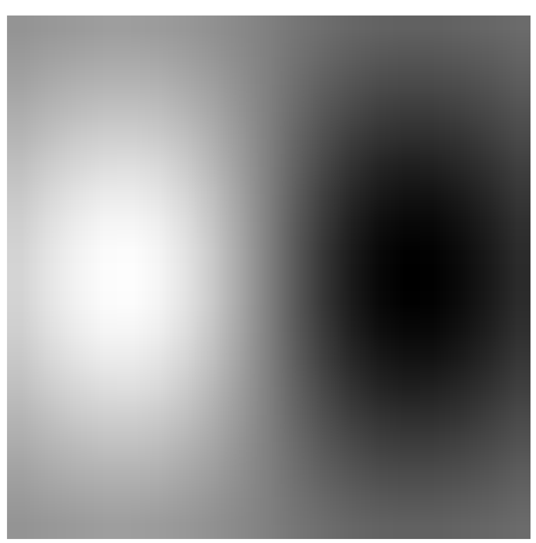
Derivative of Gaussian filter

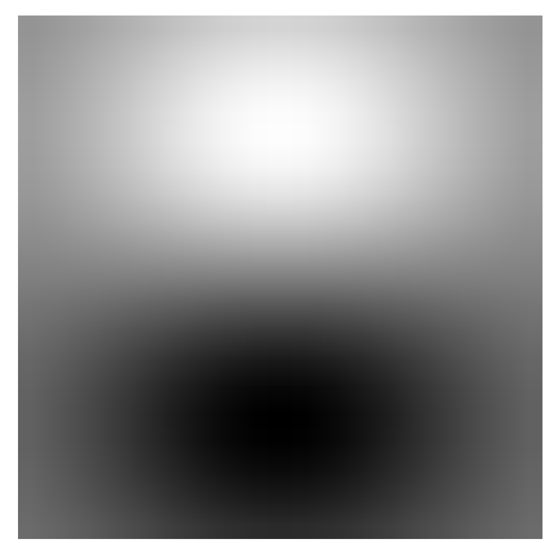


x-direction

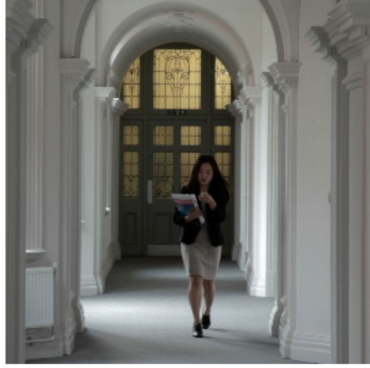


y-direction



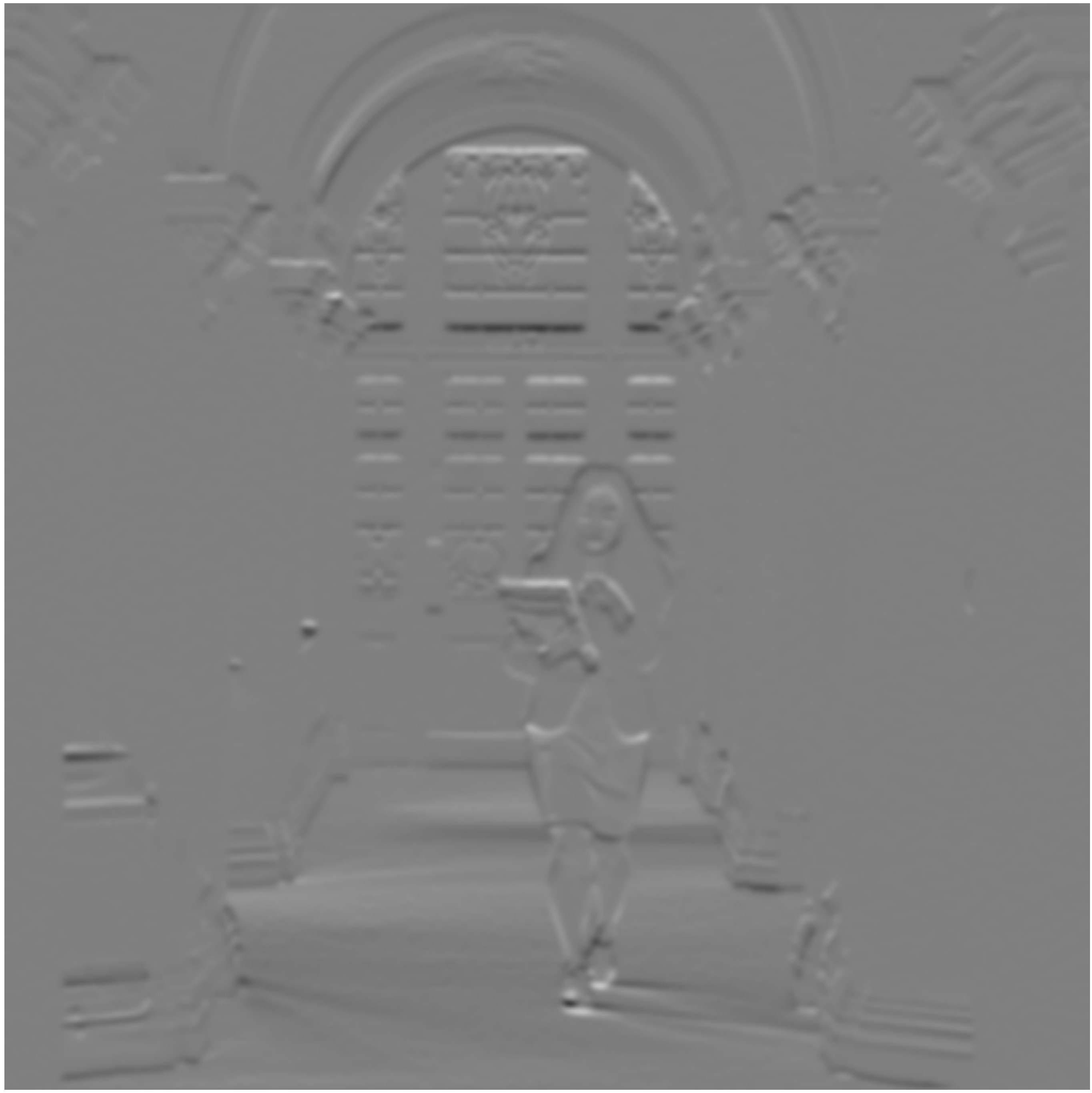


Compute Gradients



X Derivative of Gaussian

Y Derivative of Gaussian



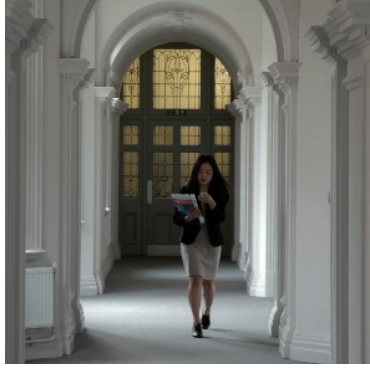
(x2 + 0.5 for visualization)

Canny edge detector

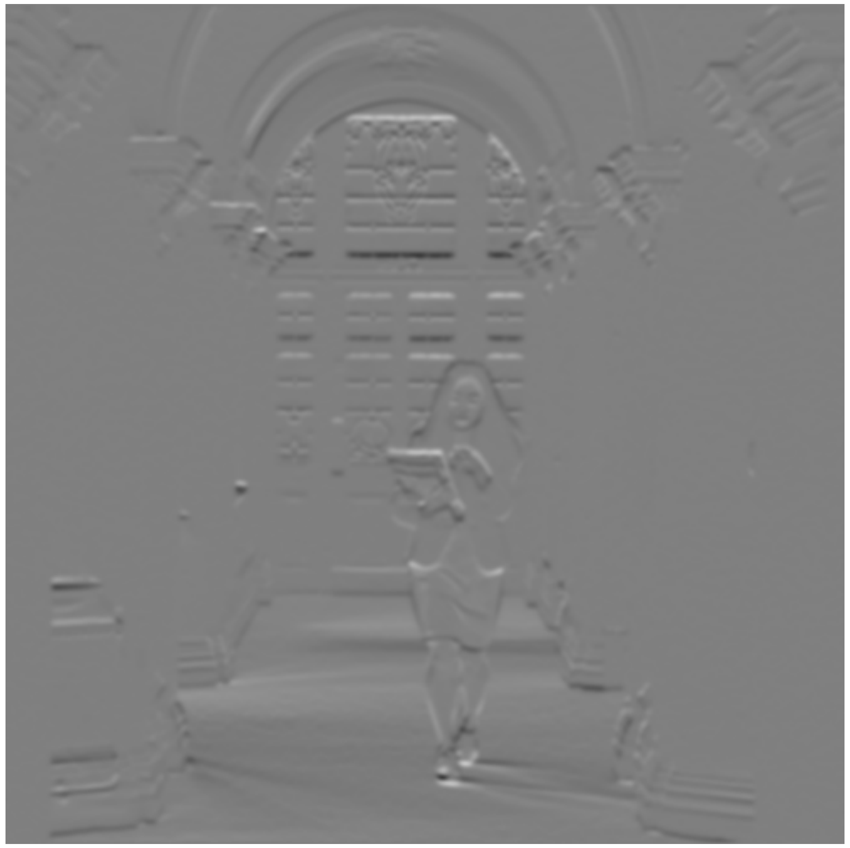
1. Filter image with x, y derivatives of Gaussian
2. Find magnitude and orientation of gradient

Source: D. Lowe, L. Fei-Fei

Compute Gradient Magnitude

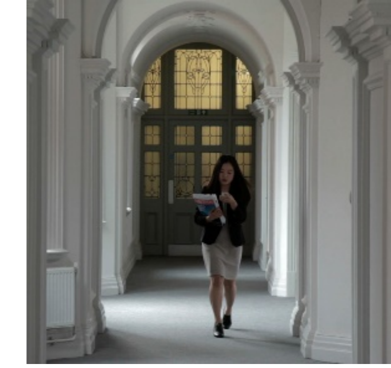


sqrt(XDerivOfGaussian .^2 + YDerivOfGaussian .^2) = gradient magnitude

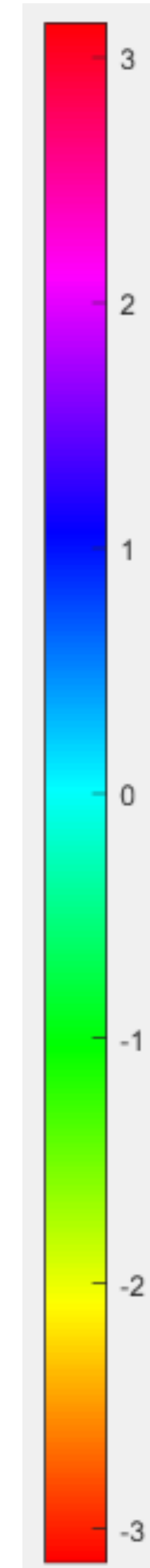


(x4 for visualization)

Compute Gradient Orientation



- Threshold magnitude at minimum level
- Get orientation via theta = atan2(gy, gx)

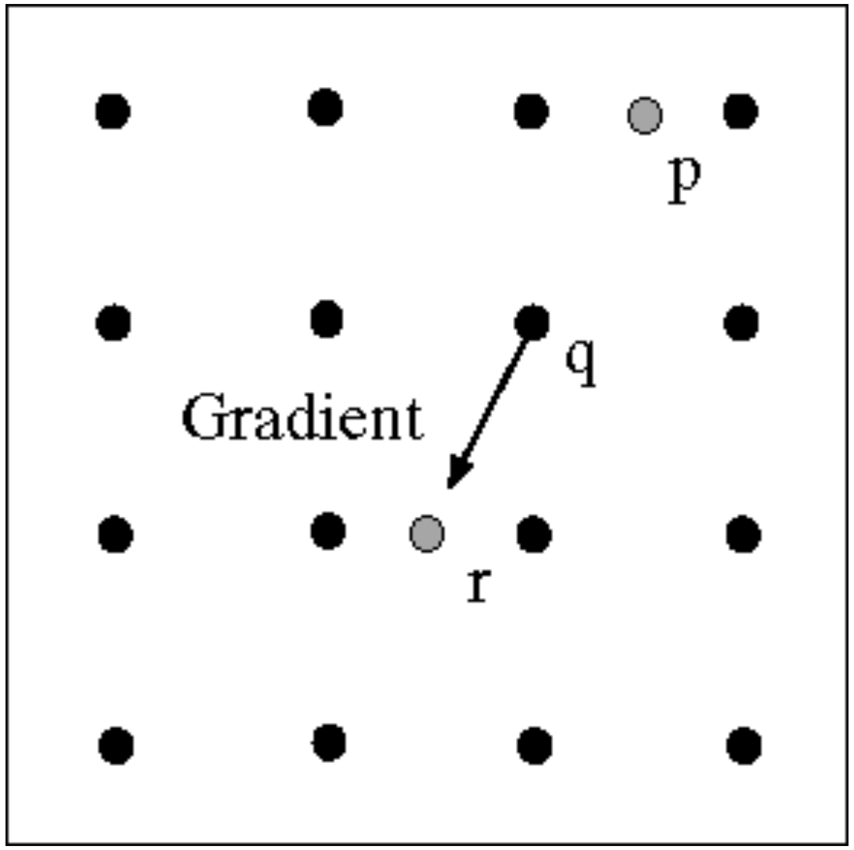


Canny edge detector

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” to single pixel width

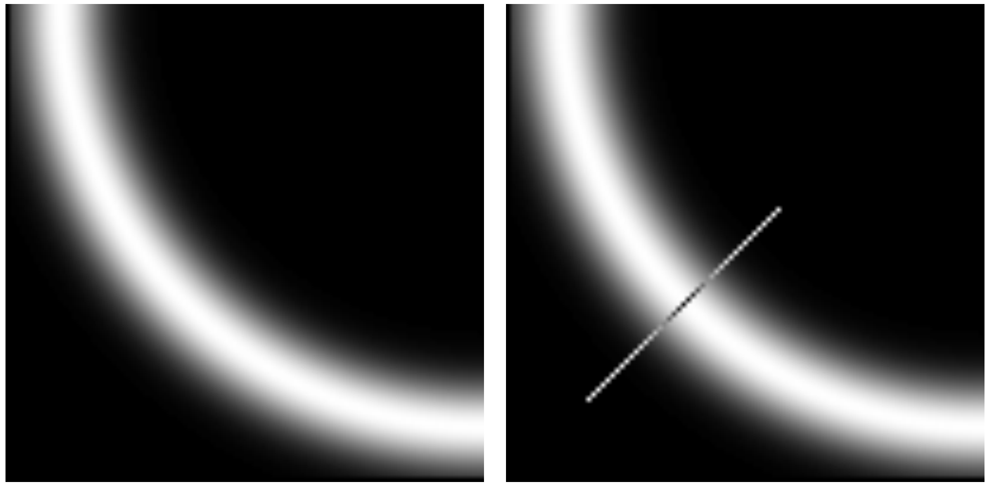
Source: D. Lowe, L. Fei-Fei

Non-maximum suppression for each orientation



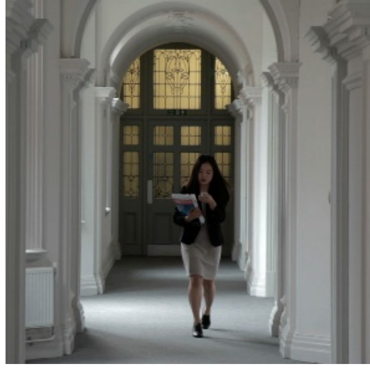
At pixel q:
We have a maximum if the value is larger than those at both p and at r.

Interpolate along gradient direction to get these values.



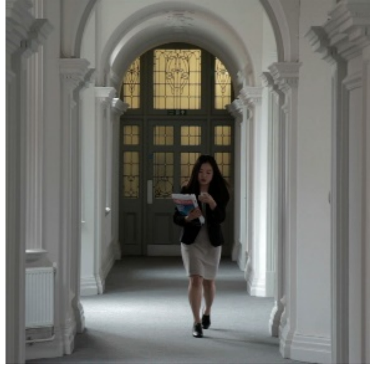
Source: D. Forsyth

Before Non-max Suppression



Gradient magnitude (x4 for visualization)

After non-max suppression



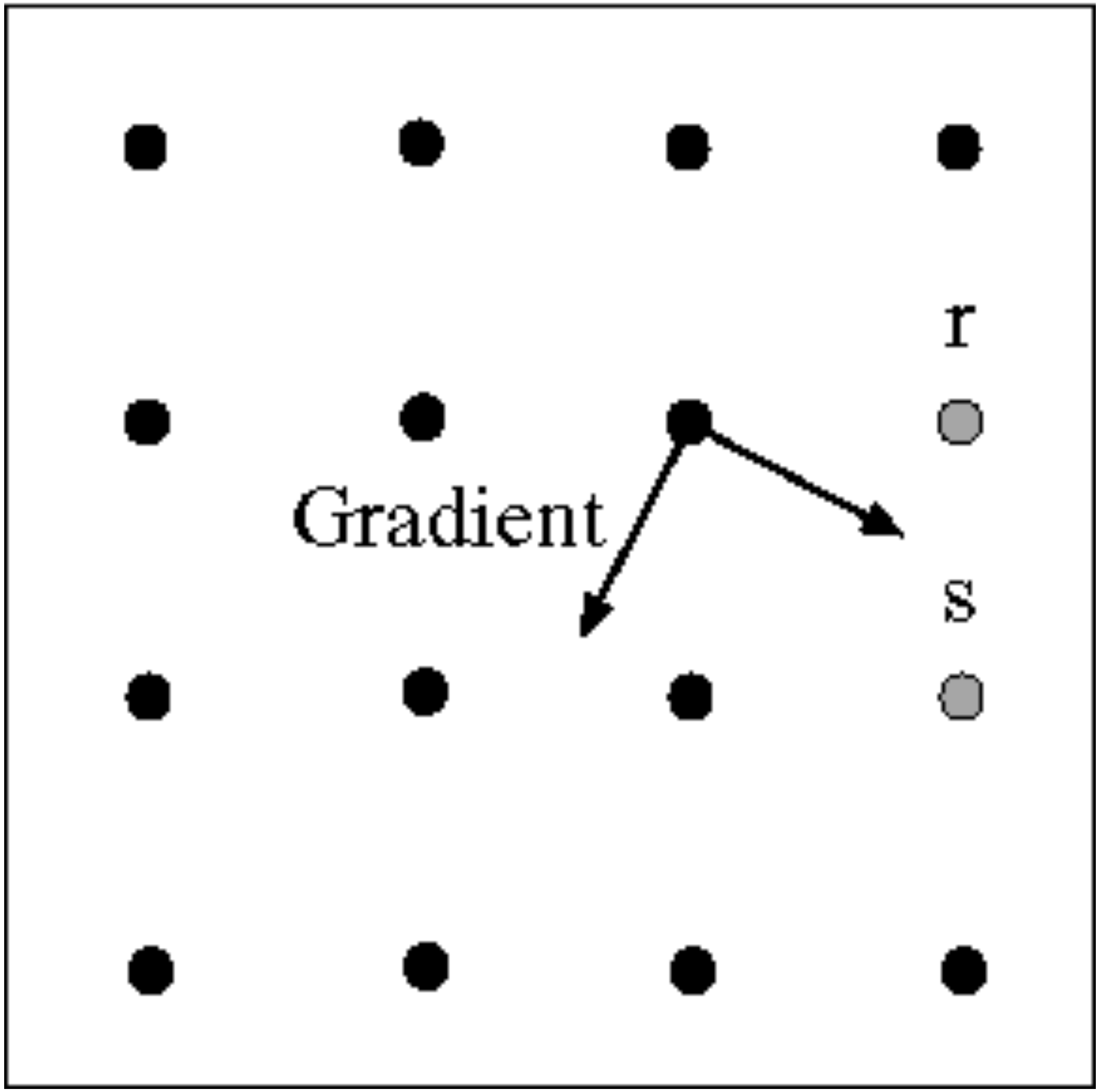
Gradient magnitude (x4 for visualization)

Canny edge detector

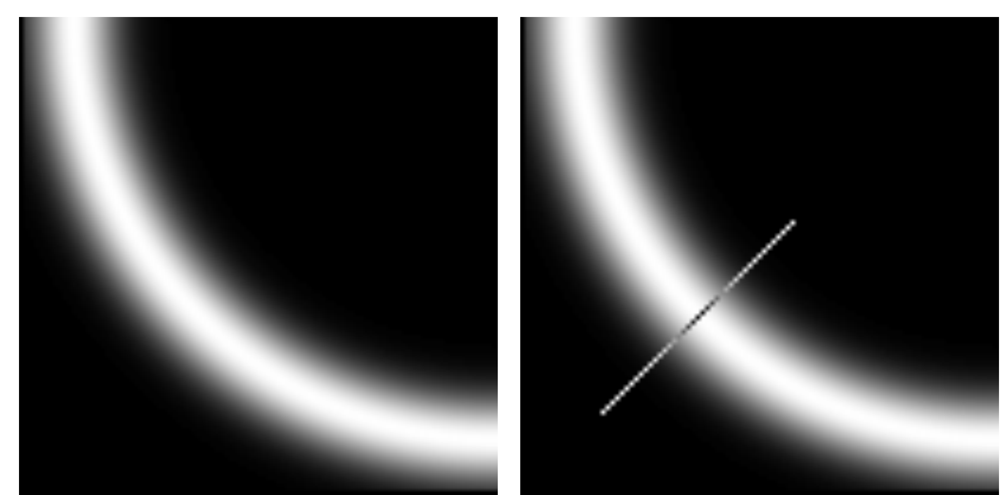
1. Filter image with x, y derivatives of Gaussian
2. Find magnitude and orientation of gradient
3. Non-maximum suppression:
   – Thin multi-pixel wide “ridges” to single pixel width
4. ‘Hysteresis’ Thresholding

Source: D. Lowe, L. Fei-Fei

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).

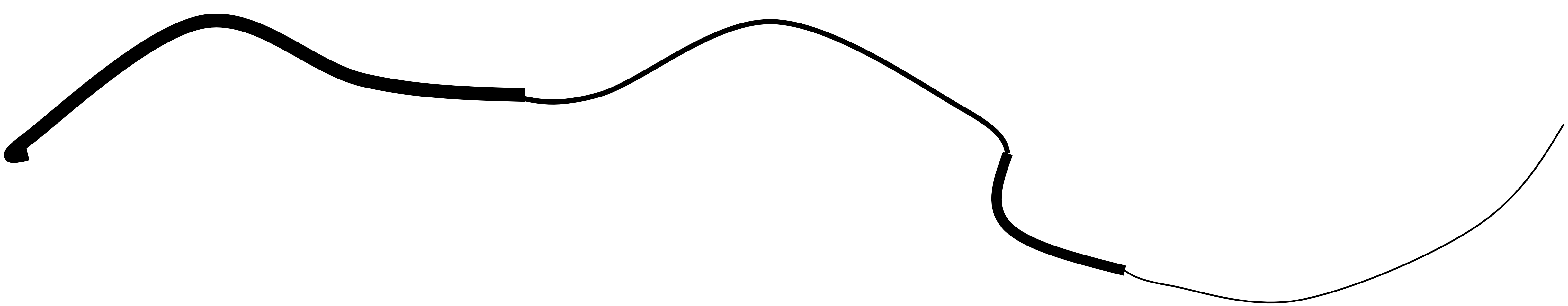


Source: D. Forsyth

‘Hysteresis’ thresholding

- Two thresholds – high and low
- Grad. mag. > high threshold? = strong edge
- Grad. mag. < low threshold? noise
- In between = weak edge

- ‘Follow’ edges starting from strong edge pixels
- Continue them into weak edges
    - Connected components (Szeliski 3.3.4)



Source: S. Seitz

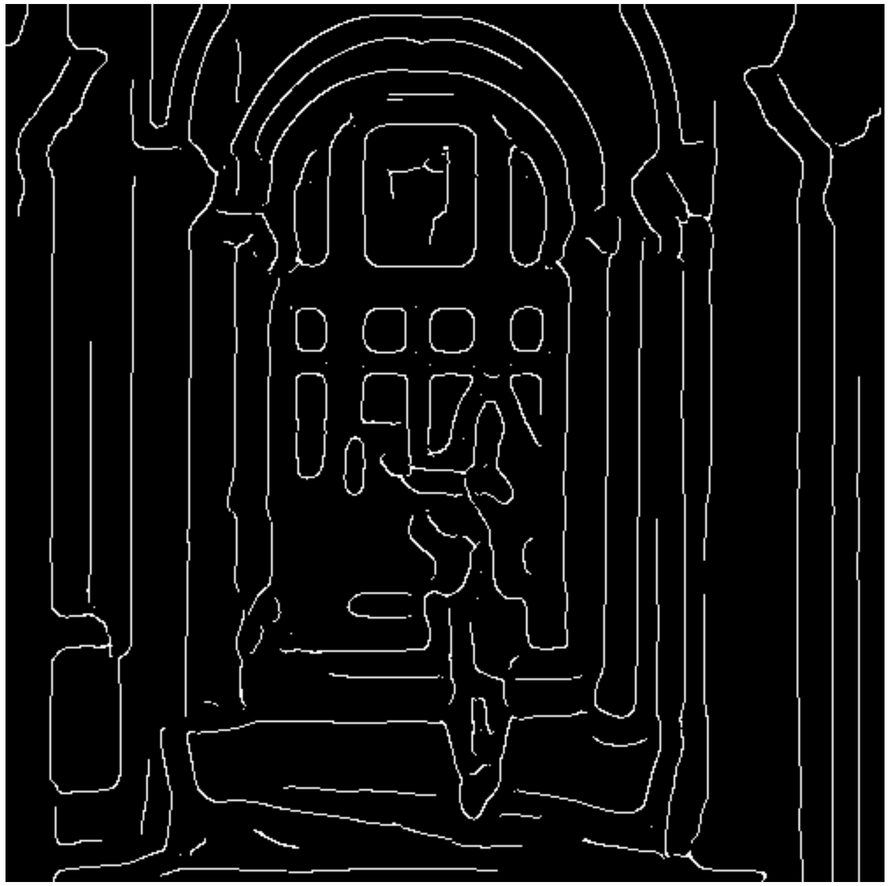
Final Canny Edges

$$\sigma = \sqrt{2},\ t_{low} = 0.05, t_{high} = 0.1$$

Effect of σ (Gaussian kernel spread/size)

Original

$\sigma = \sqrt{2}$



$\sigma = 4\sqrt{2}$

The choice of σ depends on desired behavior

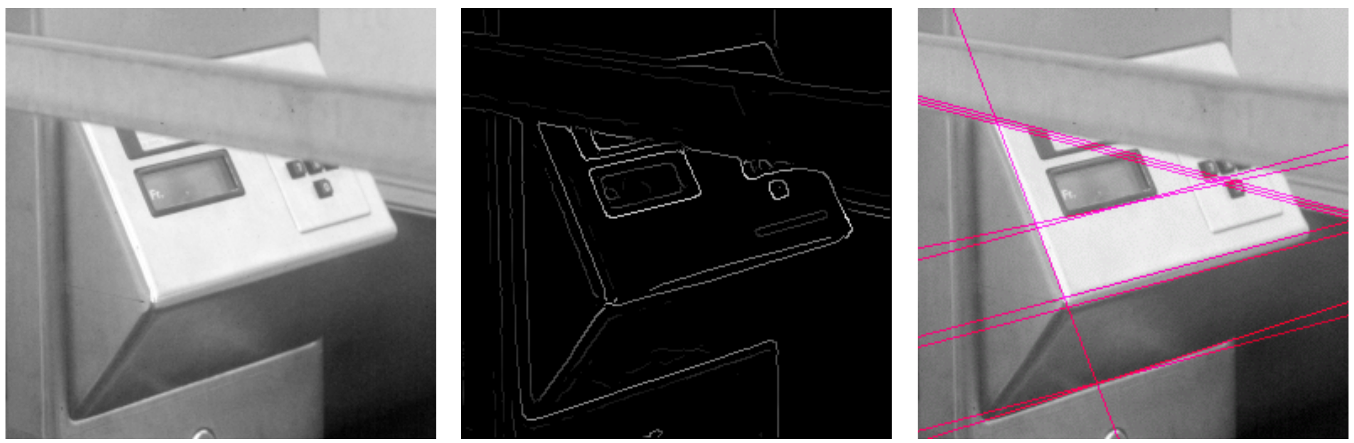
- large σ detects large scale edges
- small σ detects fine features

Source: S. Seitz

Fitting

Fitting

- Want to associate a model with observed features



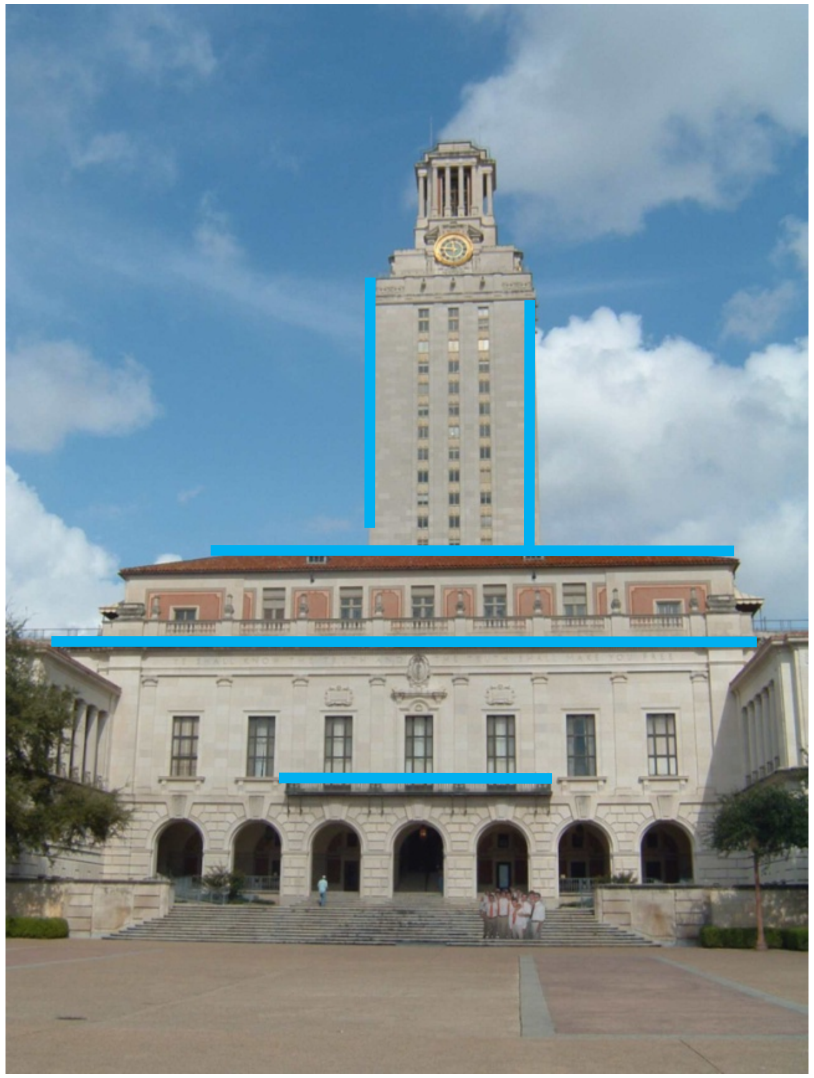
[Fig from Marszalek & Schmid, 2007]

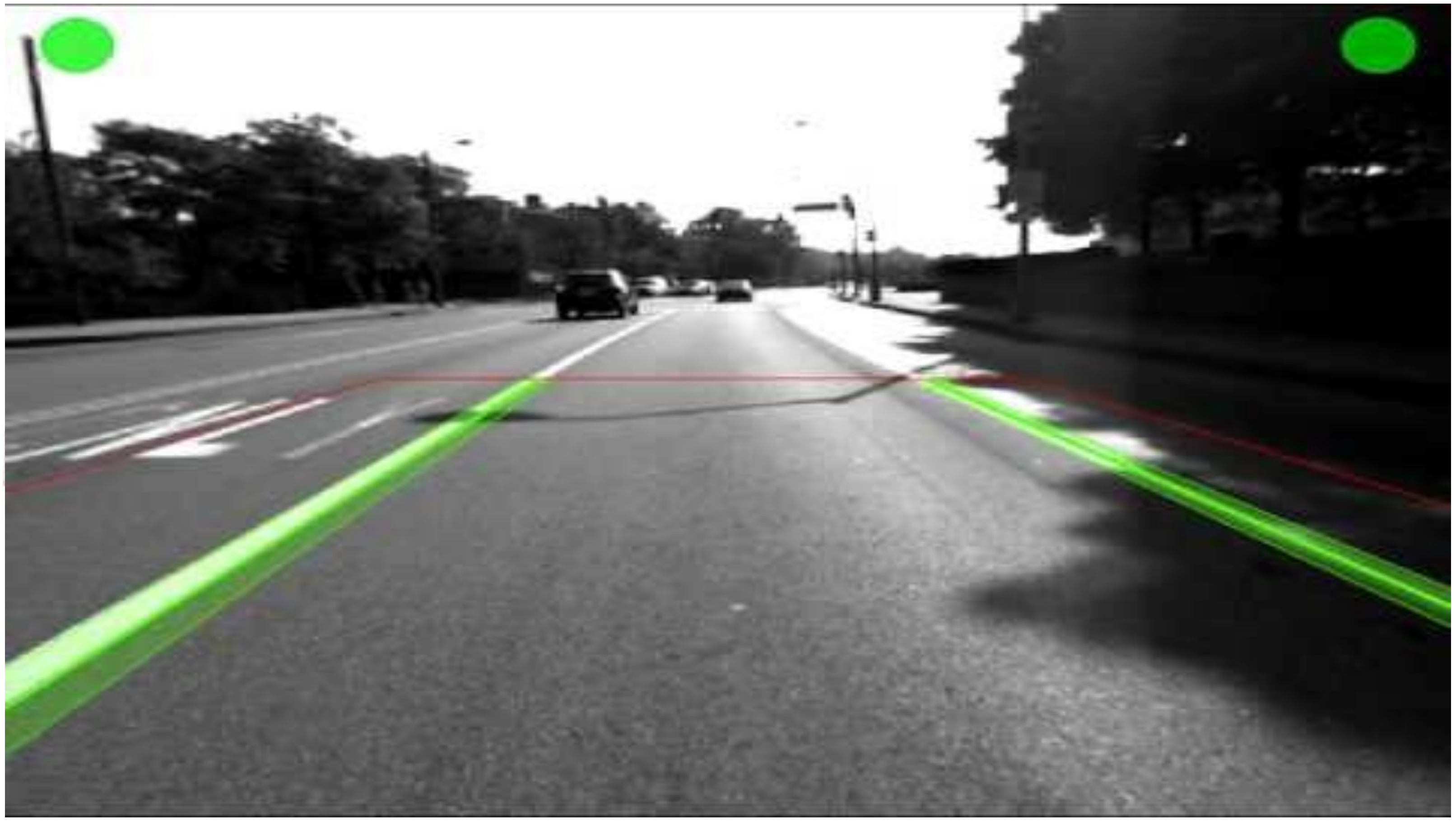
For example, the model could be a line, a circle, or an arbitrary shape.

Slide credit: K. Grauman

Case study: Line fitting

- Why fit lines? Many objects characterized by presence of straight lines

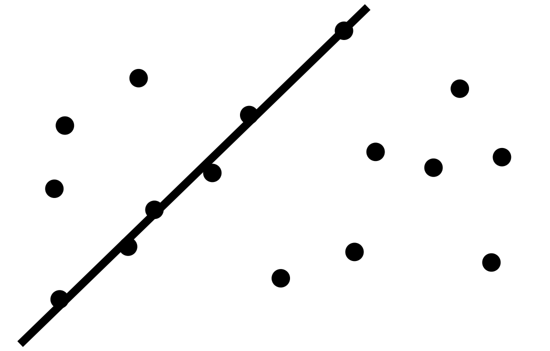


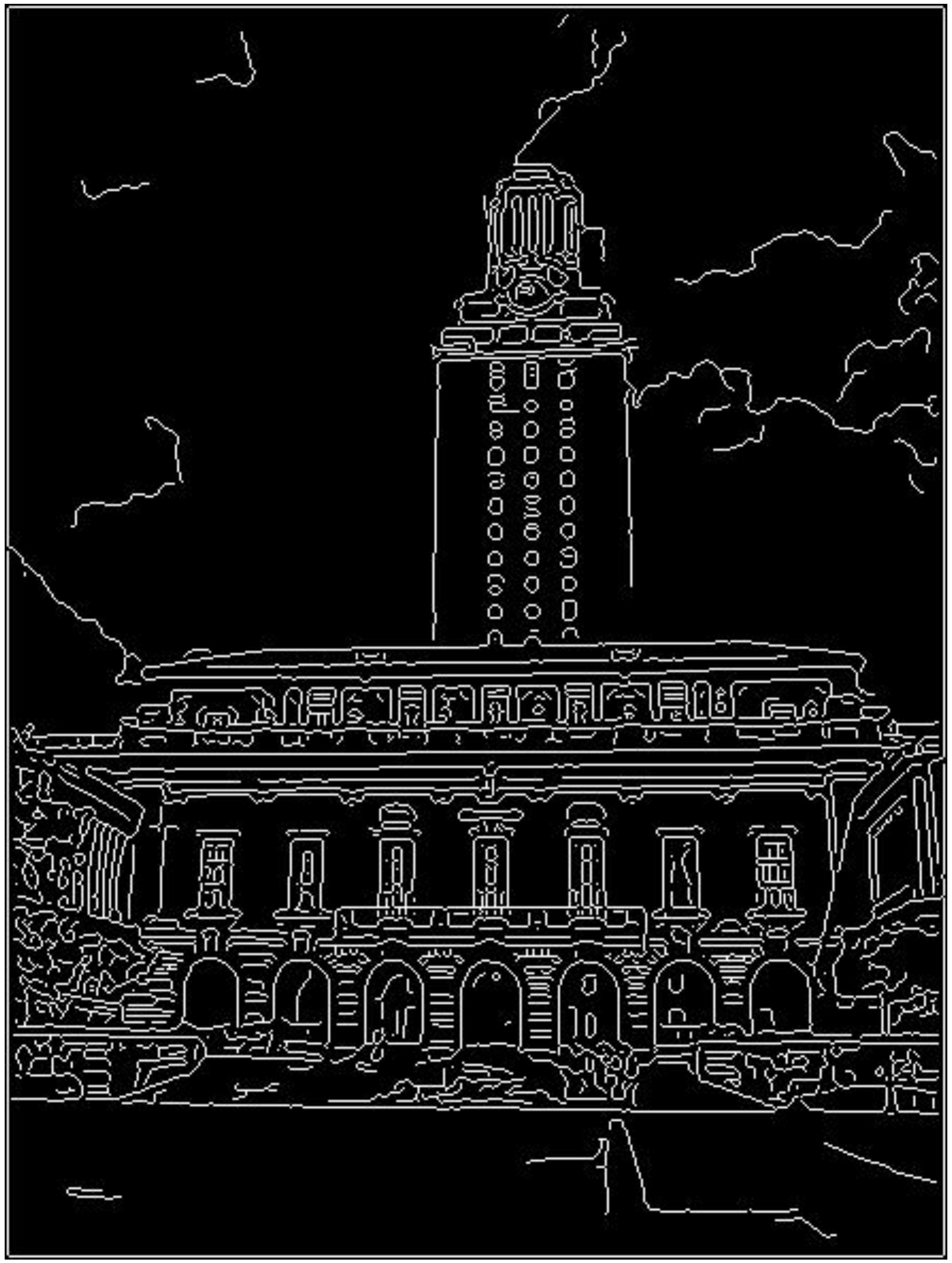


- Wait, why aren't we done just by running edge detection?

Slide credit: K. Grauman

Difficulty of line fitting





- **Extra** edge points (clutter), multiple models:
  - which points go with which line, if any?
- Only some parts of each line detected, and some parts are **missing:**
  - how to find a line that bridges missing evidence?
- **Noise** in measured edge points, orientations:
  - how to detect true underlying parameters?

Fitting: Main idea

- Choose a parametric model to represent a set of features
- Membership criterion is not local
  - Can’t tell whether a point belongs to a given model just by looking at that point
- Three main questions:
  - What model represents this set of features best?
  - Which of several model instances gets which feature?
  - How many model instances are there?
- Computational complexity is important
  - It is infeasible to examine every possible set of parameters and every possible combination of features

Slide credit: L. Lazebnik

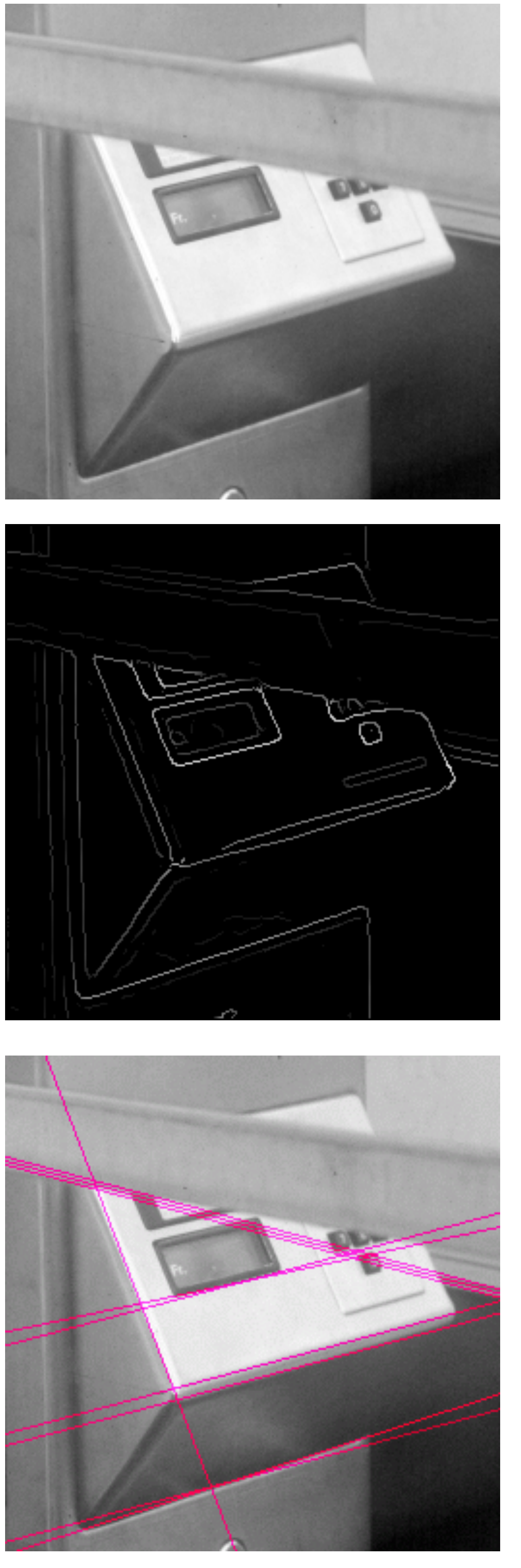
Fitting lines: Hough transform

- Given points that belong to a line, what is the line?
- How many lines are there?
- Which points belong to which lines?

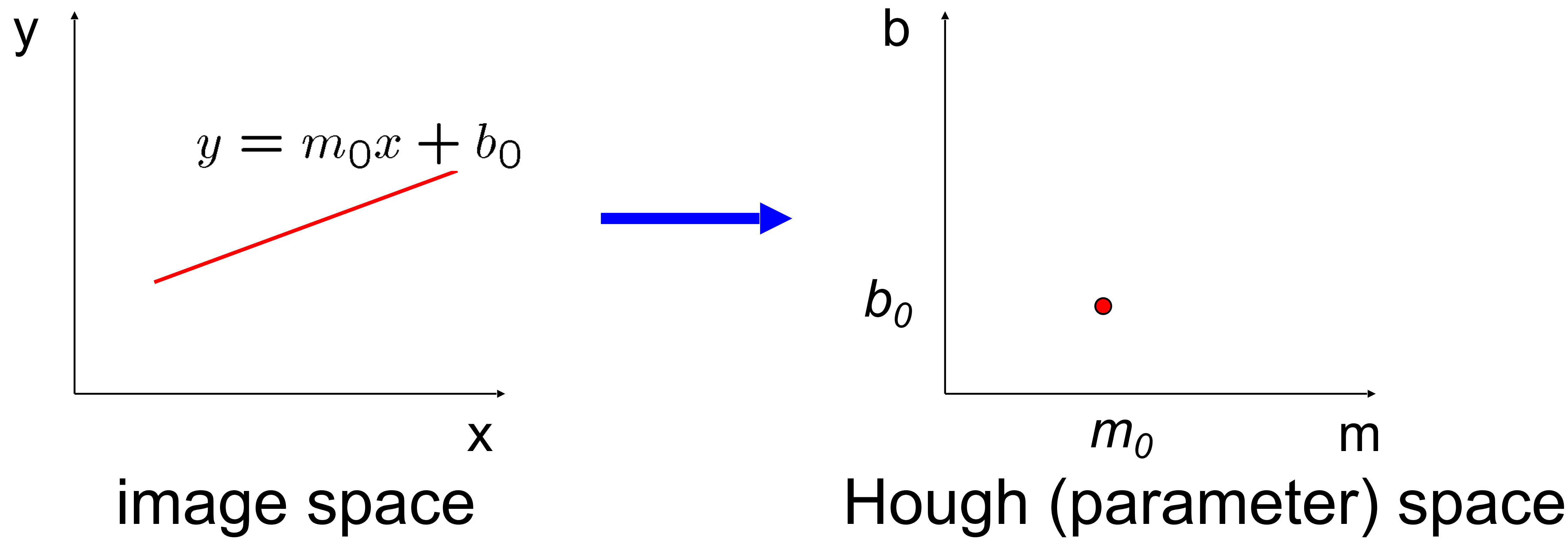
- **Hough Transform** is a voting technique that can be used to answer all of these questions.

  Main idea:

  1. Record vote for each possible line on which each edge point lies.
  2. Look for lines that get many votes.



Finding lines in an image: Hough space

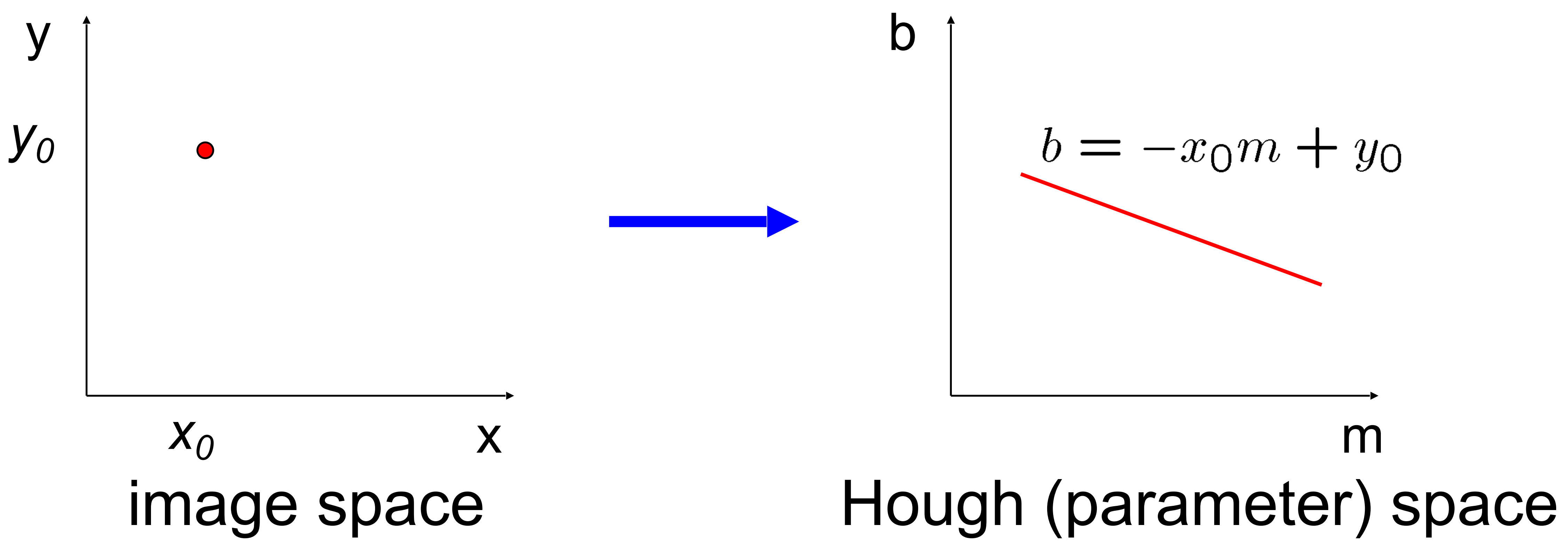


Connection between image (x,y) and Hough (m,b) spaces

- A line in the image corresponds to a point in Hough space
- To go from image space to Hough space:
  - given a set of points (x,y), find all (m,b) such that y = mx + b

Slide credit: Steve Seitz

Finding lines in an image: Hough space



Connection between image (x,y) and Hough (m,b) spaces

- A line in the image corresponds to a point in Hough space
- To go from image space to Hough space:
  - given a set of points (x,y), find all (m,b) such that y = mx + b
- What does a point (x_0, y_0) in the image space map to?
  - Answer: the solutions of $b = -x_0 m + y_0$
  - this is a line in Hough space

Slide credit: Steve Seitz

Finding lines in an image: Hough space

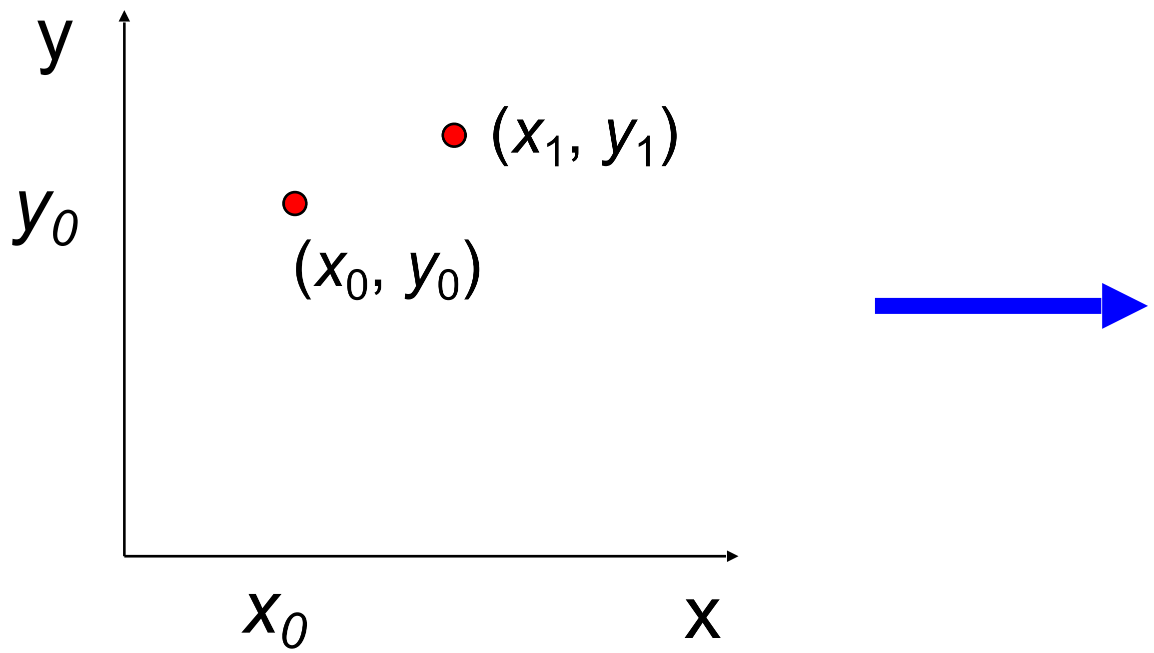
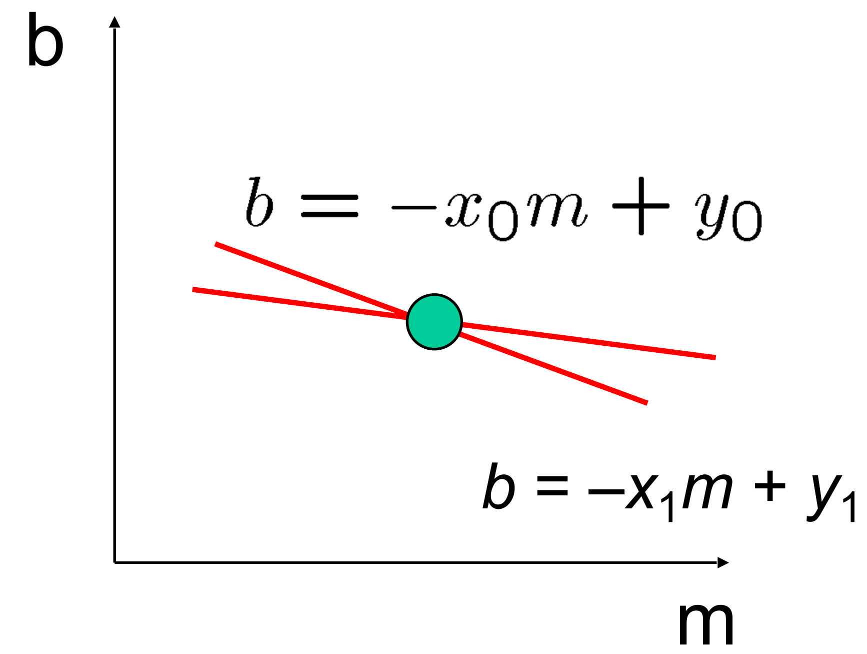


image space



Hough (parameter) space

What are the line parameters for the line that contains both (x_0, y_0) and (x_1, y_1)?

- It is the intersection of the lines $b = -x_0 m + y_0$ and $b = -x_1 m + y_1$

Finding lines in an image: Hough algorithm

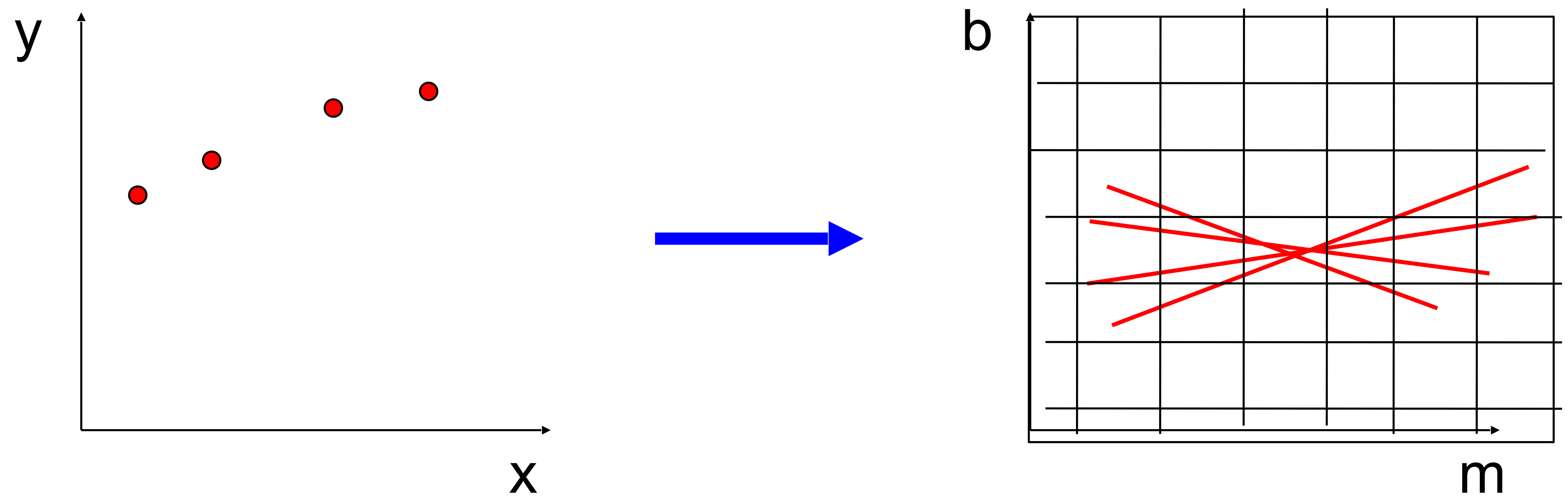


image space Hough (parameter) space

How can we use this to find the most likely parameters (m,b) for the most prominent line in the image space?

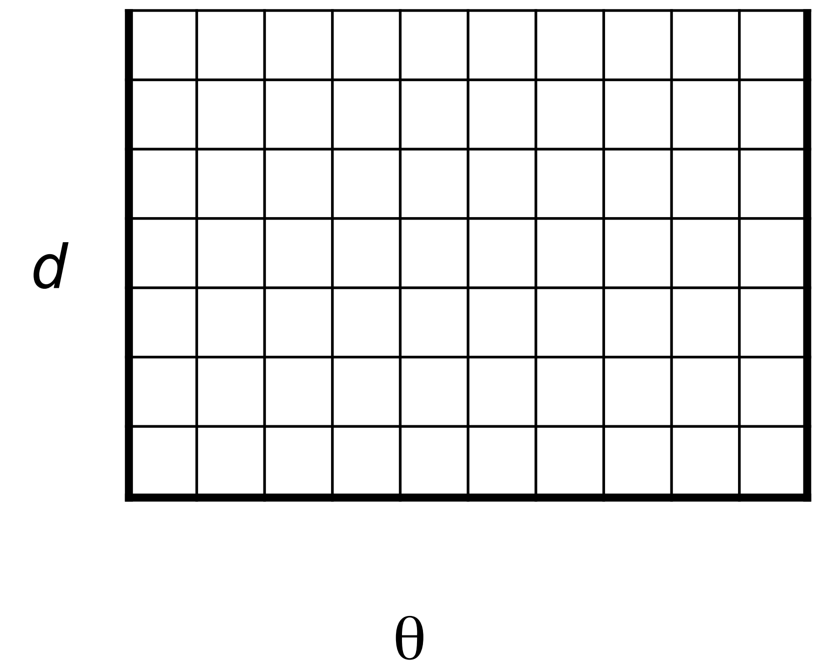
- Let each edge point in image space *vote* for a set of possible parameters in Hough space
- Accumulate votes in discrete set of bins*; parameters with the most votes indicate line in image space.

Hough transform algorithm

Using the polar parameterization:

$$x\cos\theta - y\sin\theta = d$$

H: accumulator array (votes)



Basic Hough transform algorithm

1. Initialize H[d, θ]=0
2. for each edge point I[x,y] in the image

   for θ = [θ_{min} to θ_{max}] // some quantization

   $$d = x\cos\theta - y\sin\theta$$

   H[d, θ] += 1
3. Find the value(s) of (d, θ) where H[d, θ] is maximum
4. The detected line in the image is given by $d = x\cos\theta - y\sin\theta$

Time complexity (in terms of number of votes per pt)?

Source: Steve Seitz

Extensions

Extension: Use the image gradient

1. same
2. for each edge point I[x,y] in the image

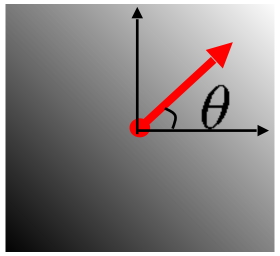
   θ = gradient at (x,y)

   $$d = x\cos\theta - y\sin\theta$$

   H[d, θ] += 1

3. same
4. same

(Reduces degrees of freedom)



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

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

Hough transform for circles

- Circle: center (a,b) and radius r

$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

- For a fixed radius r, unknown gradient direction

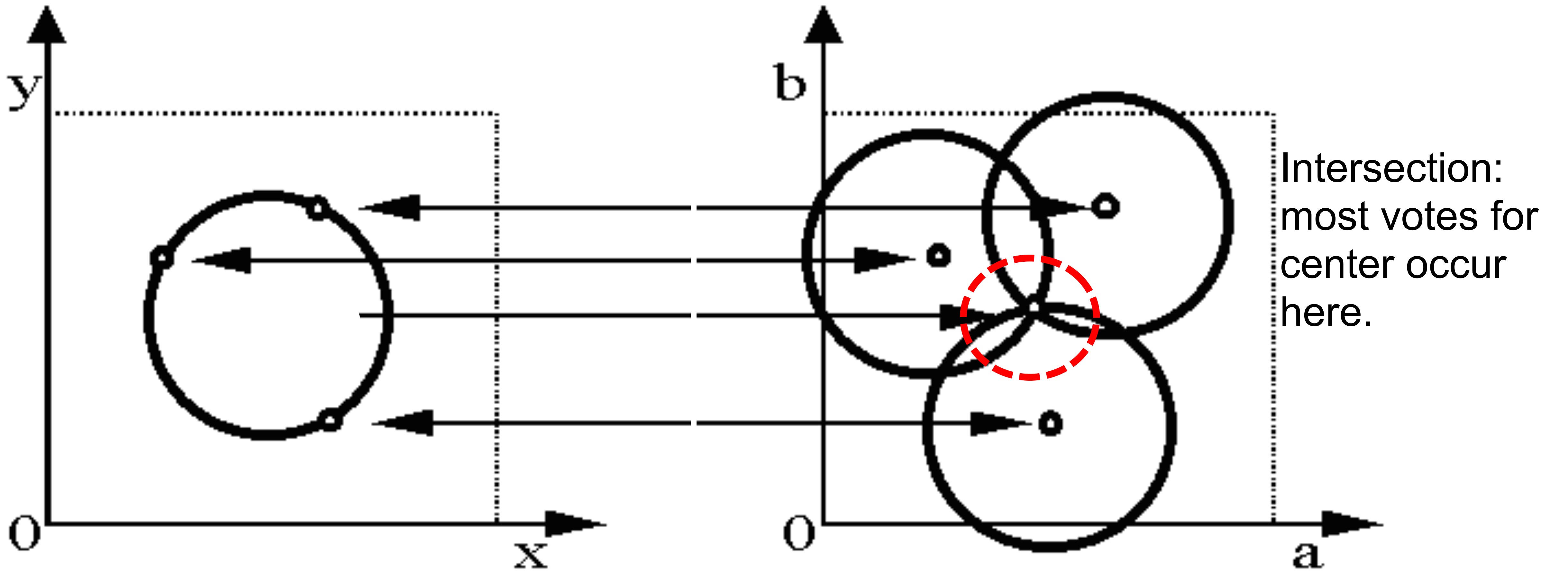


Image space

Hough space

Hough transform for circles

- Circle: center (a,b) and radius r

$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

- For an unknown radius r, unknown gradient direction

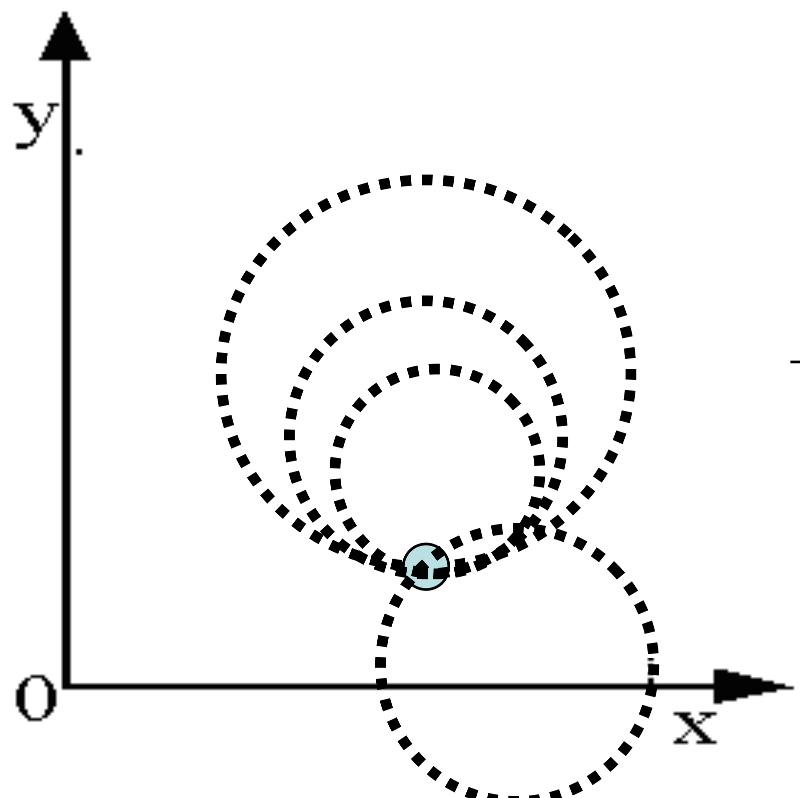
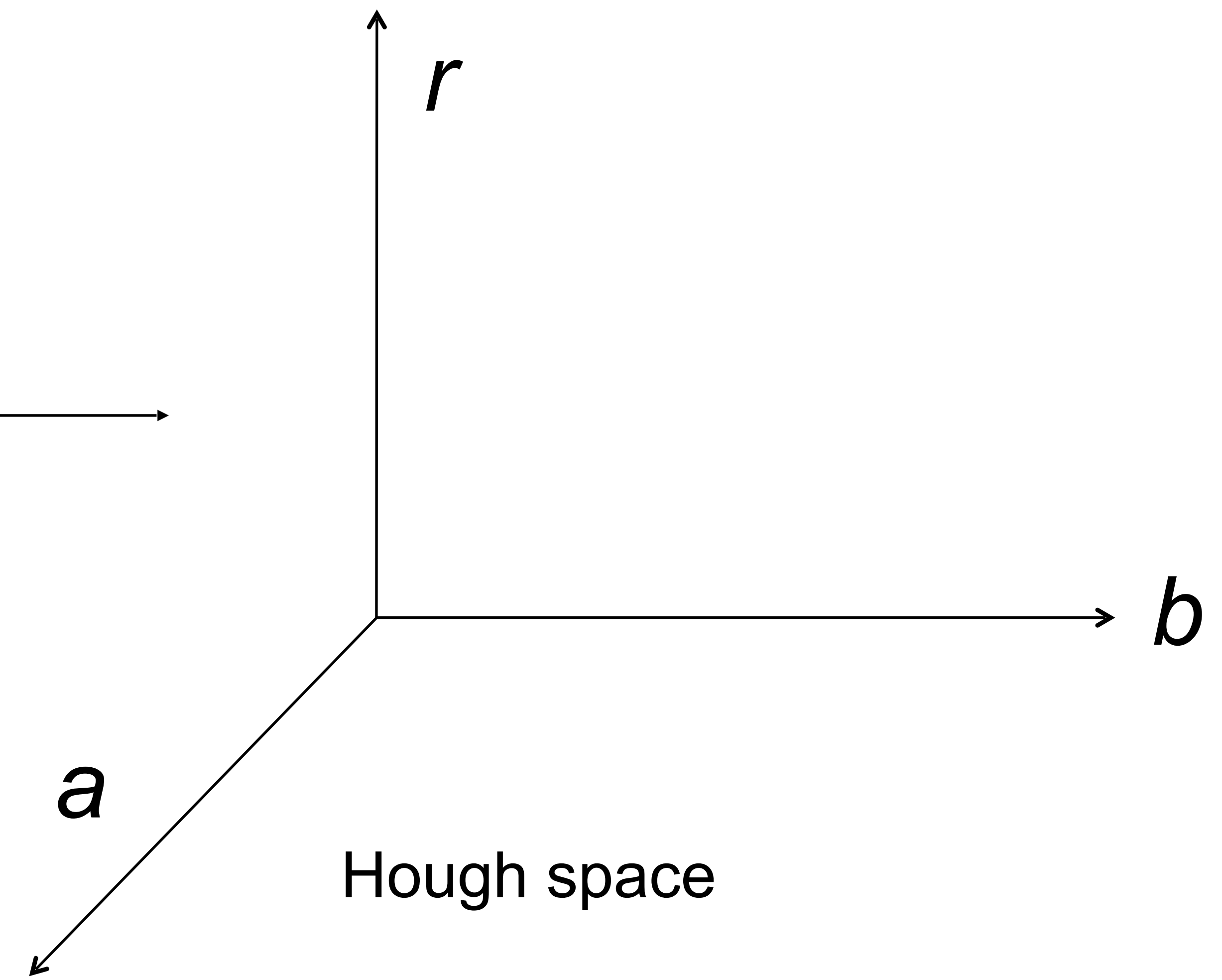


Image space



Hough space

Hough transform for circles

- Circle: center (a,b) and radius r

$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

- For an unknown radius r, unknown gradient direction

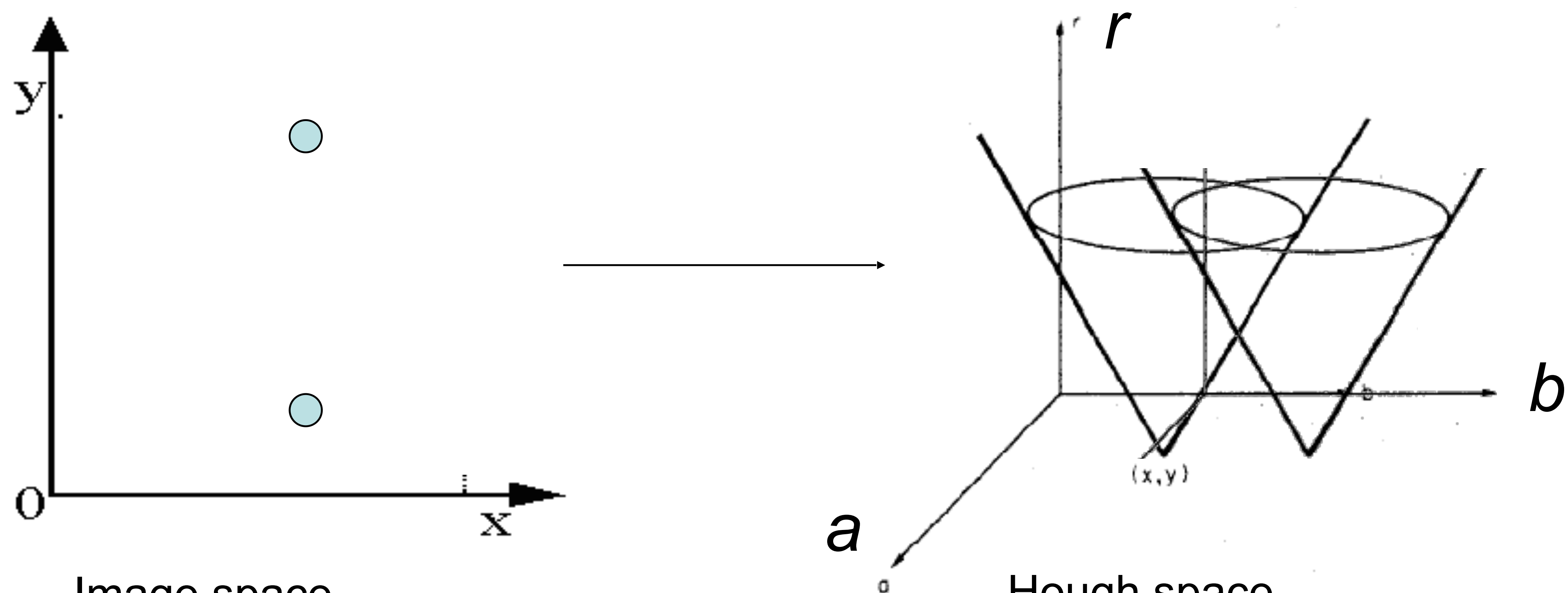
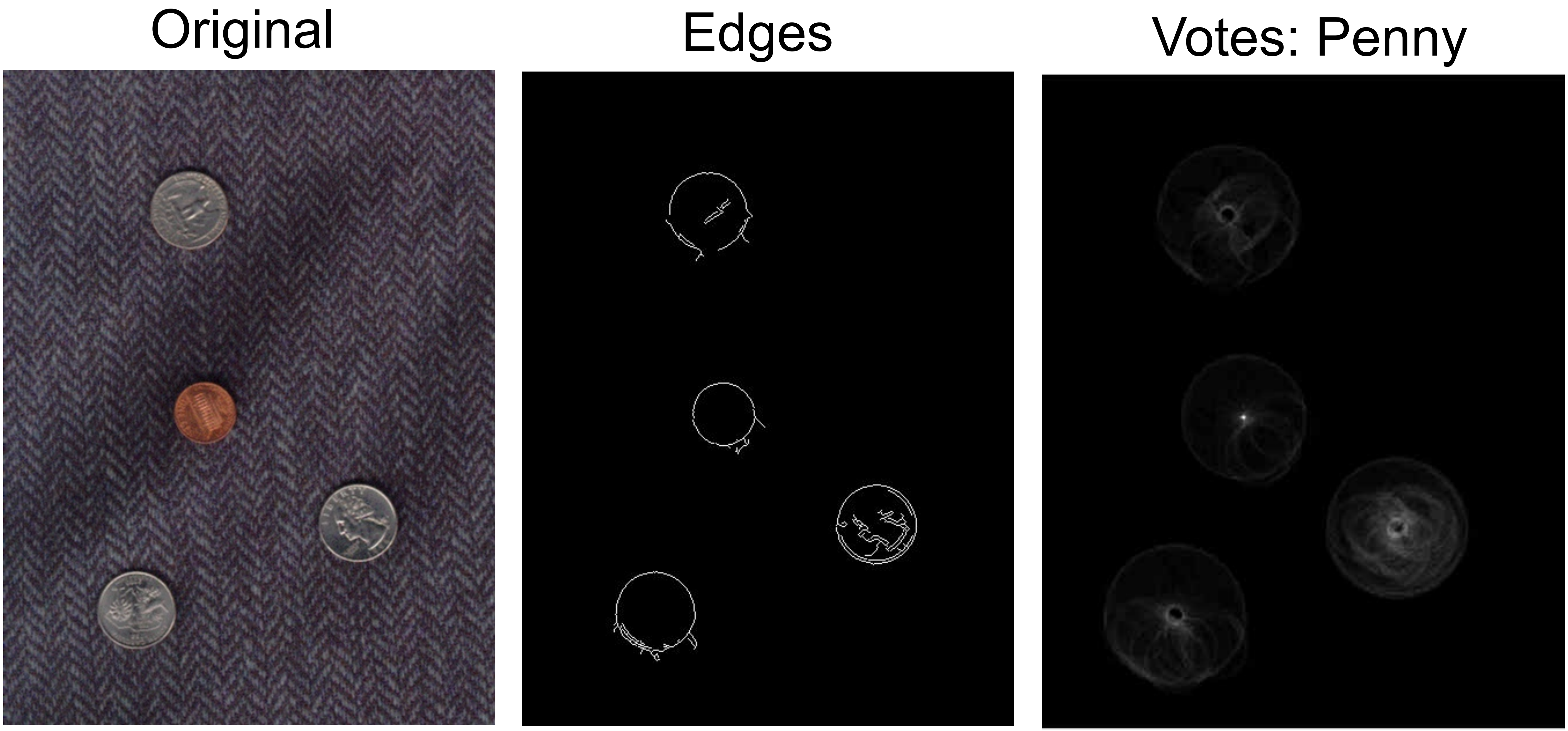


Image space

Hough space

Example: detecting circles with Hough

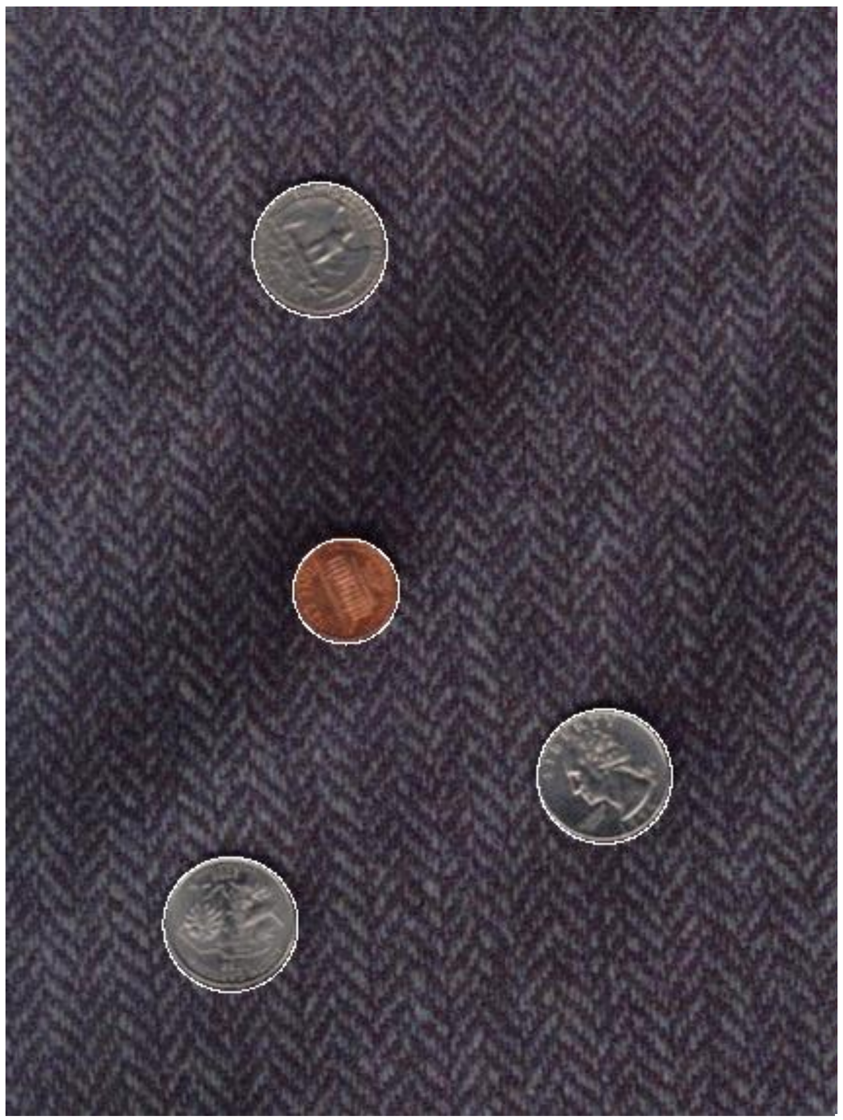


Note: a different Hough transform (with separate accumulators) was used for each circle radius (quarters vs. penny).

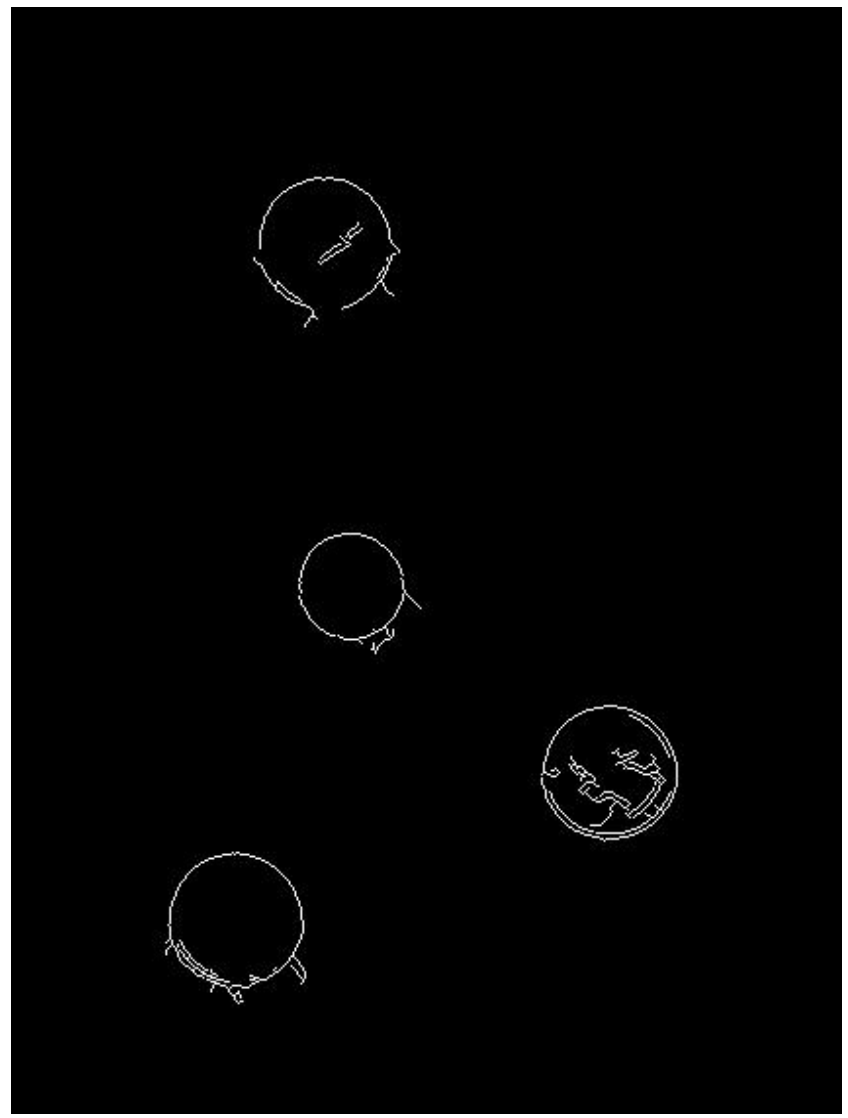
Coin finding sample images from: Vivek Kwatra

Example: detecting circles with Hough

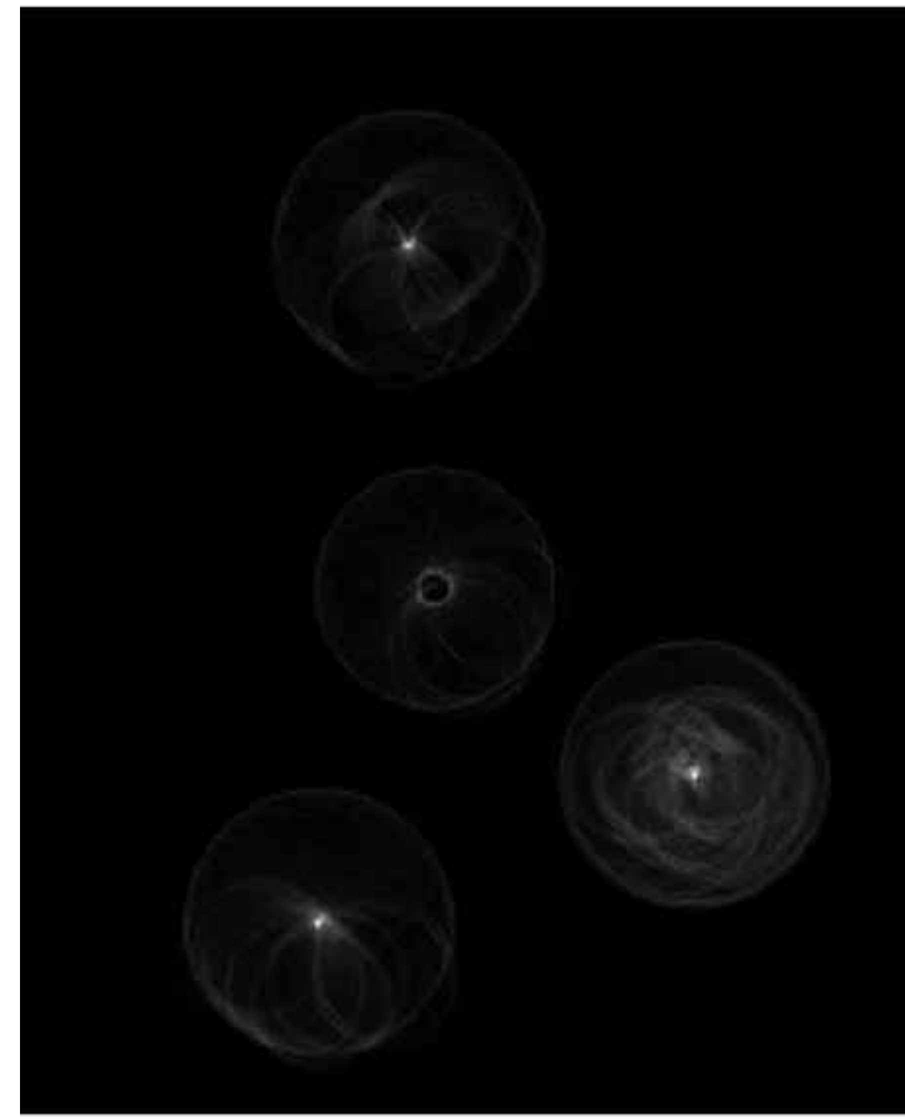
Original



Edges

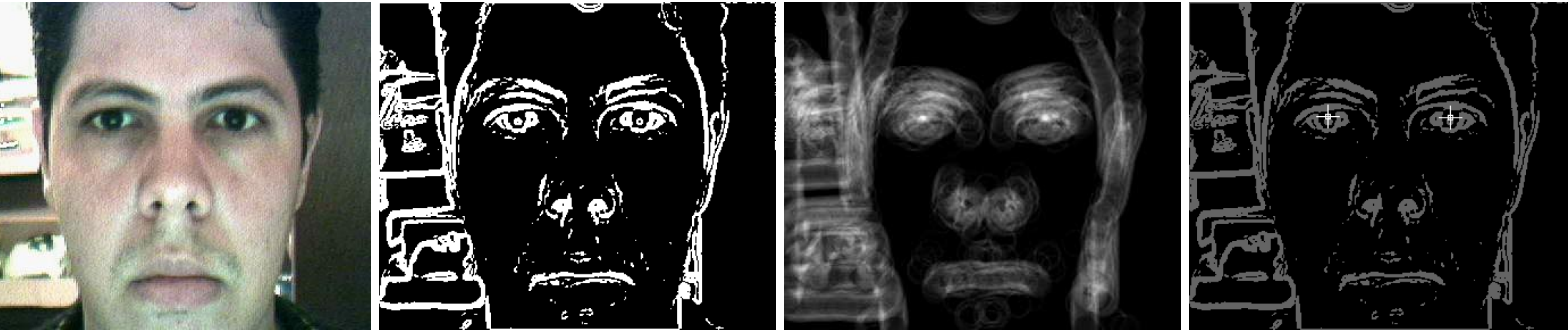


Votes: Quarter



Coin finding sample images from: Vivek Kwatra

Example: iris detection



Gradient+threshold

Hough space (fixed radius)

Max detections

- Hemerson Pistori and Eduardo Rocha Costa http://rsbweb.nih.gov/ij/plugins/hough-circles.html

Example: iris detection


Figure 2. Original image


Figure 3. Distance image


Figure 4. Detected face region

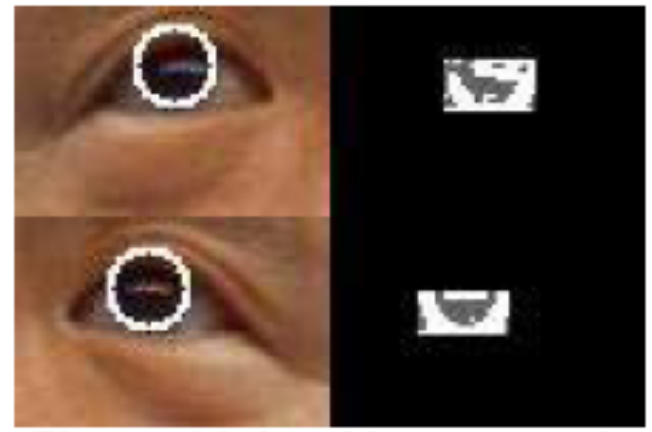


Figure 14. Looking upward

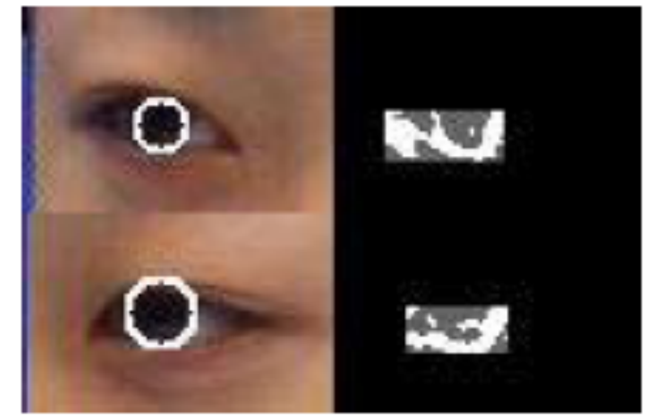


Figure 15. Looking sideways

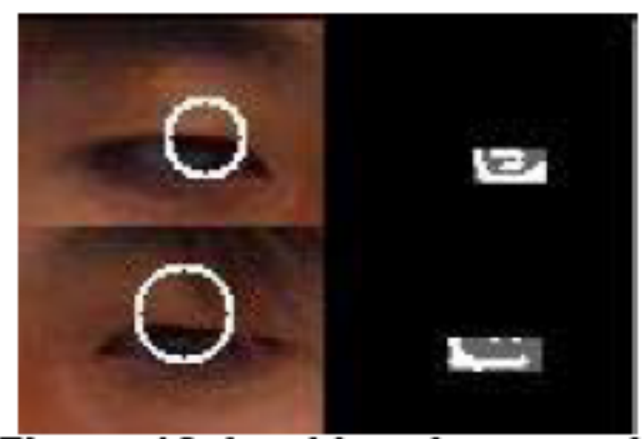


Figure 16. Looking downward

- An Iris Detection Method Using the Hough Transform and Its Evaluation for Facial and Eye Movement, by Hideki Kashima, Hitoshi Hongo, Kunihito Kato, Kazuhiko Yamamoto, ACCV 2002.

Voting: practical tips

- Minimize irrelevant tokens first
- Choose a good grid / discretization

Too fine ? Too coarse

- Vote for neighbors, also (smoothing in accumulator array)
- Use direction of edge to reduce parameters by 1
- To read back which points voted for “winning” peaks, keep tags on the votes.

Hough transform: pros and cons

Pros

- All points are processed independently, so can cope with occlusion, gaps
- Some robustness to noise: noise points unlikely to contribute *consistently* to any single bin
- Can detect multiple instances of a model in a single pass

Cons

- Complexity of search time increases exponentially with the number of model parameters
- Non-target shapes can produce spurious peaks in parameter space
- Quantization: can be tricky to pick a good grid size

Seam Carving

Seam carving: main idea

Content-aware resizing

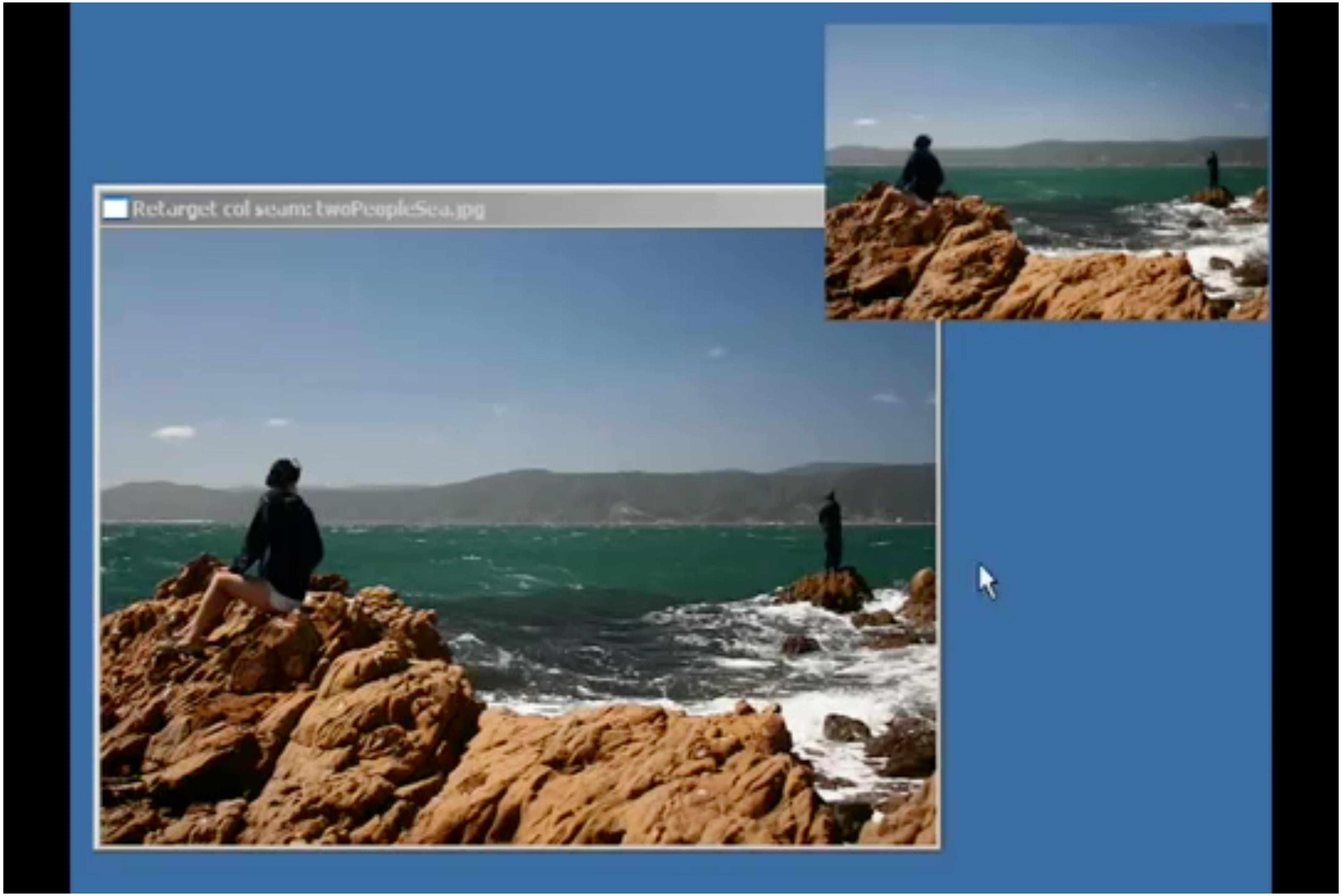
Traditional resizing

[Shai & Avidan, SIGGRAPH 2007]

Seam carving: main idea

[Shai & Avidan, SIGGRAPH 2007]

Seam Carving



Seam carving: main idea

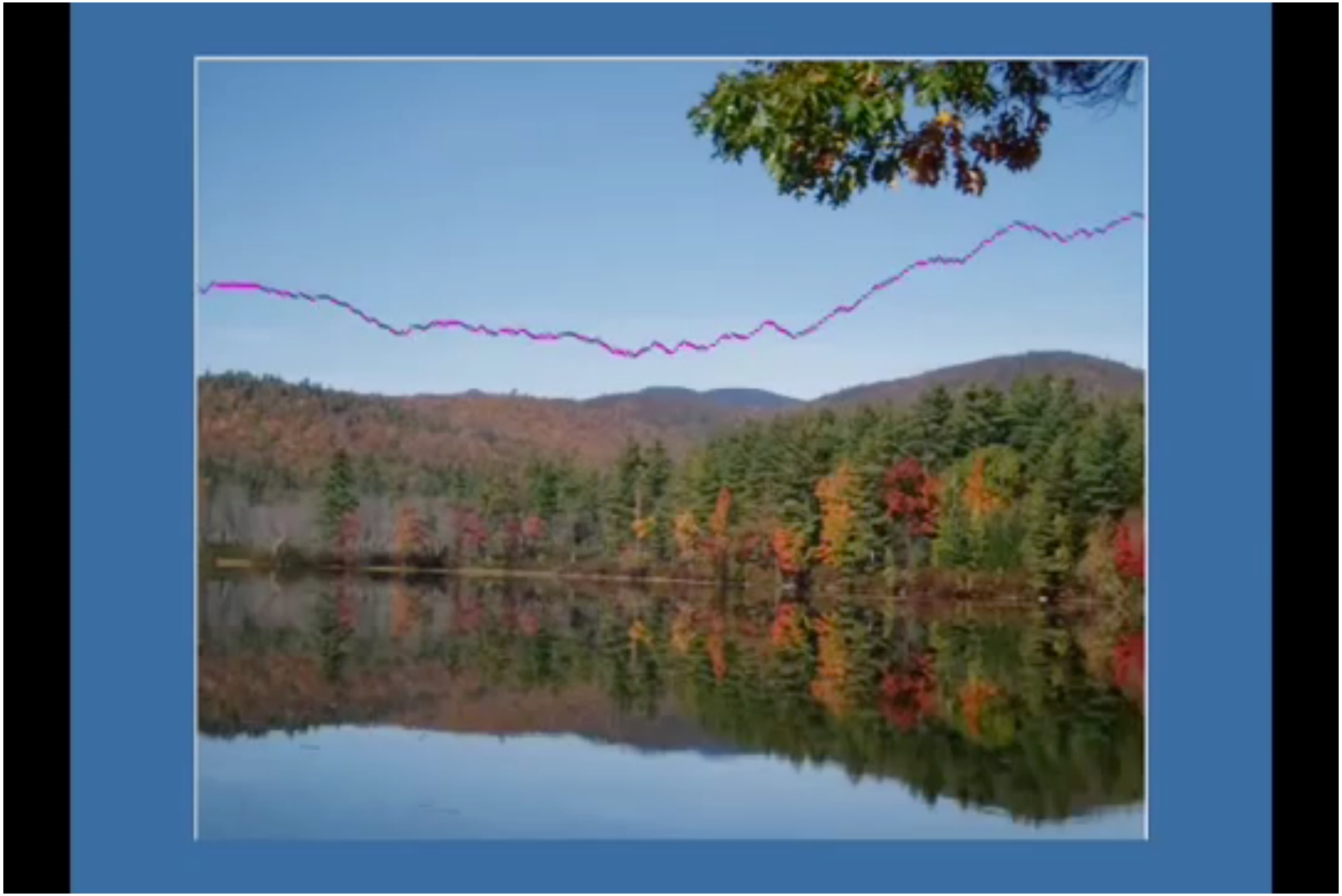
Content-aware resizing

Intuition:

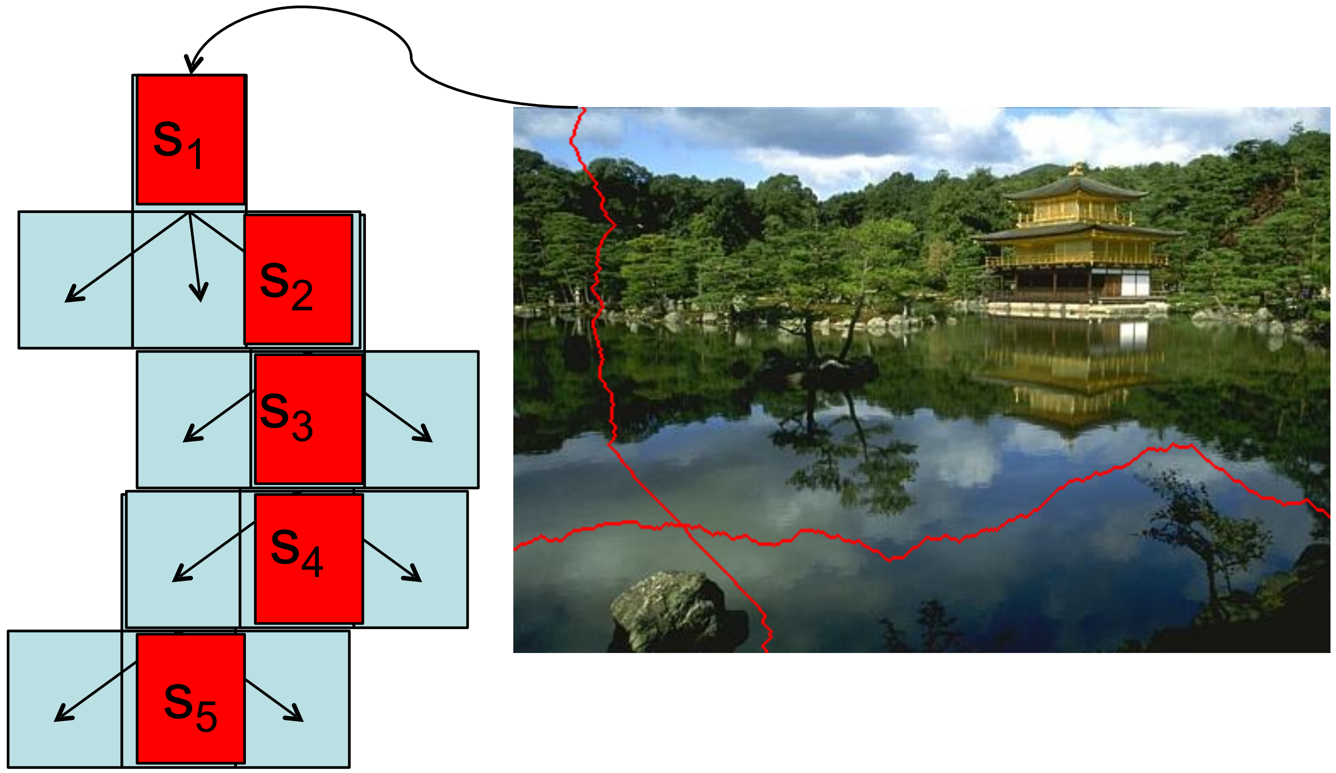
- Preserve the most "interesting" content
  - → Prefer to remove pixels with low gradient energy
- To reduce or increase size in one dimension, remove irregularly shaped "seams"
  - → Optimal solution via dynamic programming.

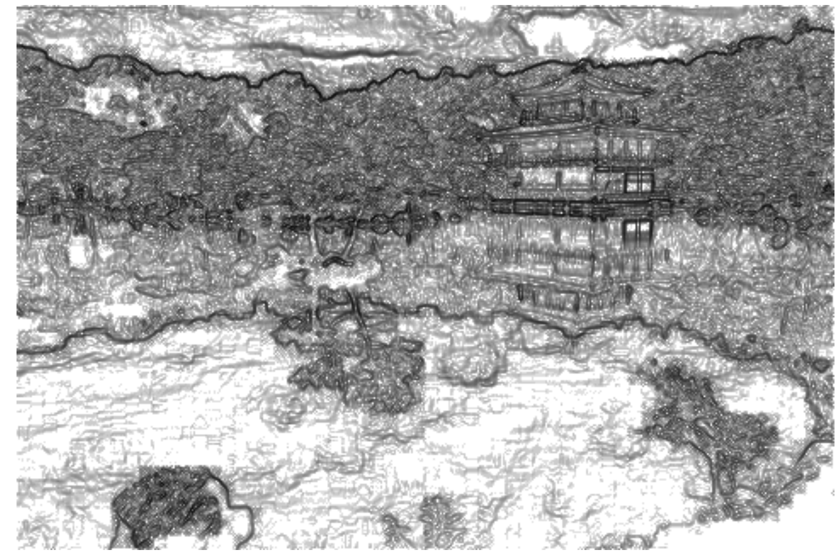
Slide credit: Kristen Grauman

Seam Carving



Seam carving: algorithm





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

Let a vertical seam **s** consist of *h* positions that form an 8-connected path.

Let the cost of a seam be:

$$Cost(\mathbf{s}) = \sum_{i=1}^{h} Energy(f(s_i))$$

Optimal seam minimizes this cost:

$$\mathbf{s}^* = \min_{\mathbf{s}} Cost(\mathbf{s})$$

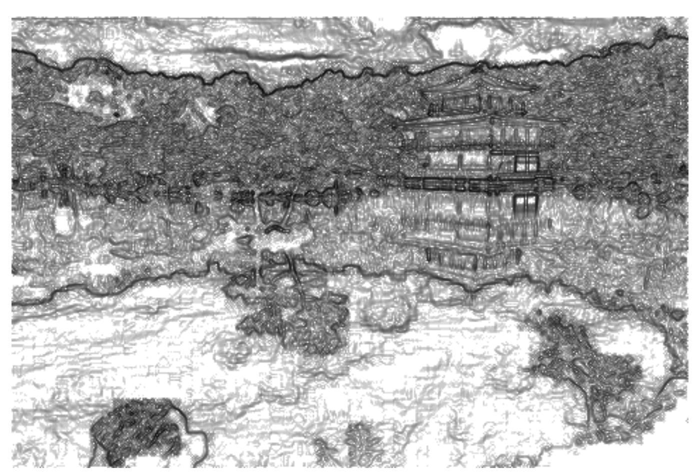
Compute it efficiently with dynamic programming.

Slide credit: Kristen Grauman

How to identify the minimum cost seam?

- First, consider a **greedy** approach:

$$\begin{matrix} 1 & 3 & \boxed{0} \\ 2 & \boxed{8} & 9 \\ 5 & \boxed{2} & 6 \end{matrix}$$



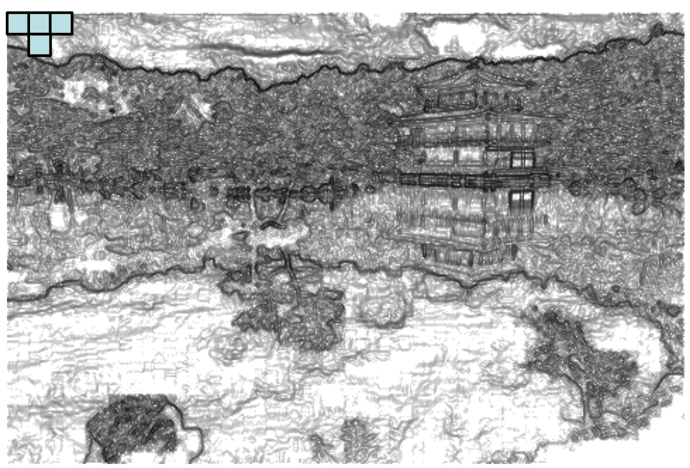
Energy matrix (gradient magnitude)

Slide credit: Kristen Grauman

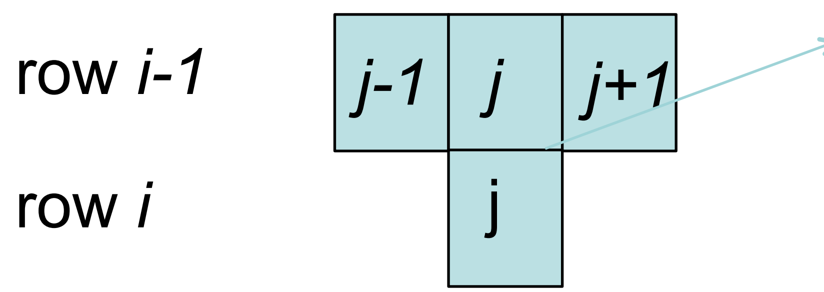
Seam carving: algorithm

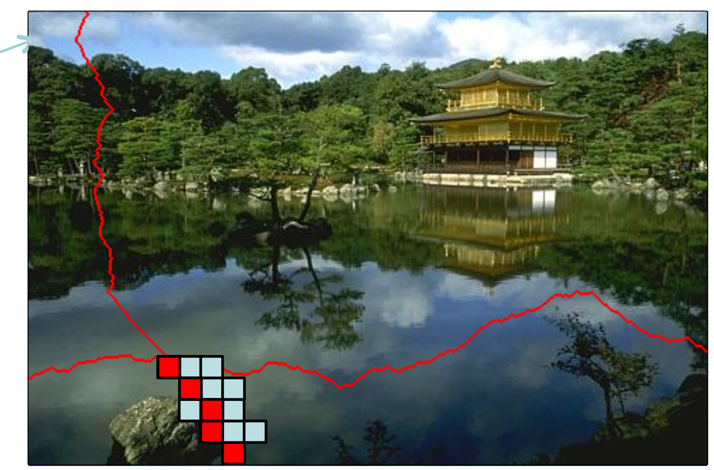
- Compute the cumulative minimum energy for all possible connected seams at each entry *(i,j)*:

$$\mathbf{M}(i,j) = Energy(i,j) + \min\big(\mathbf{M}(i-1,j-1), \mathbf{M}(i-1,j), \mathbf{M}(i-1,j+1)\big)$$



Energy matrix (gradient magnitude)





M matrix: cumulative min energy (for vertical seams)

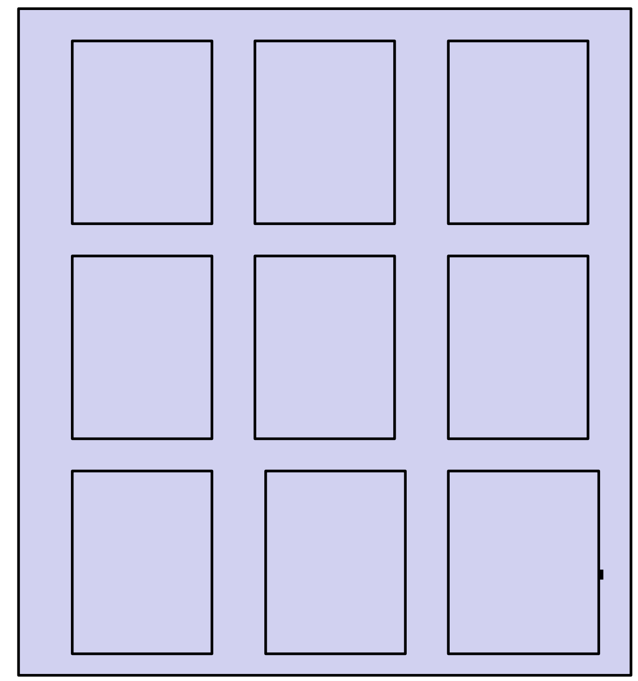
- Then, min value in last row of **M** indicates end of the minimal connected vertical seam.
- Backtrack up from there, selecting min of 3 above in **M**.

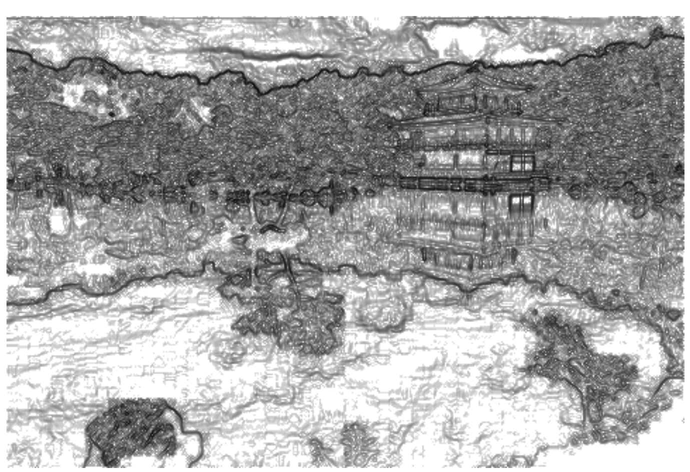
Slide credit: Kristen Grauman

Example

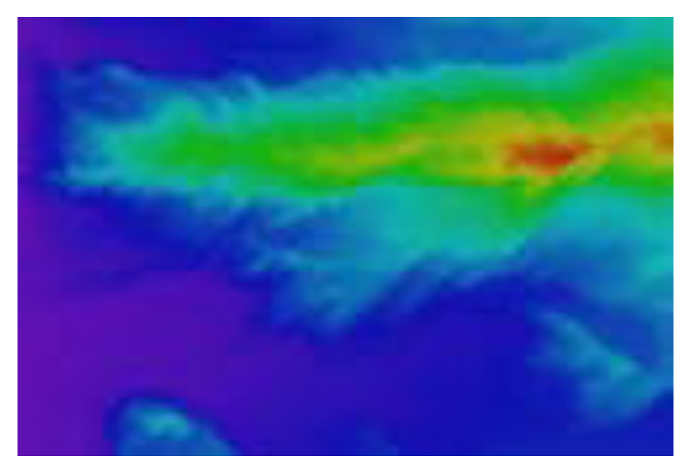
$$\mathbf{M}(i,j) = Energy(i,j) + \min\big(\mathbf{M}(i-1,j-1), \mathbf{M}(i-1,j), \mathbf{M}(i-1,j+1)\big)$$

$$\begin{matrix} 1 & 3 & 0 \\ 2 & 8 & 9 \\ 5 & 2 & 6 \end{matrix}$$





Energy matrix (gradient magnitude)



M matrix (for vertical seams)

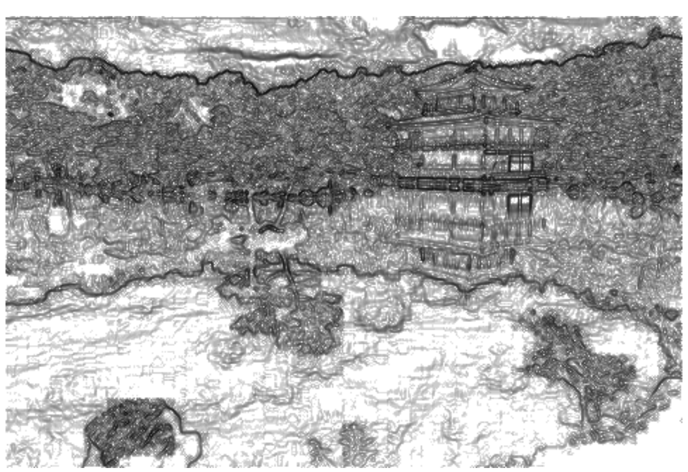
Slide credit: Kristen Grauman

Example

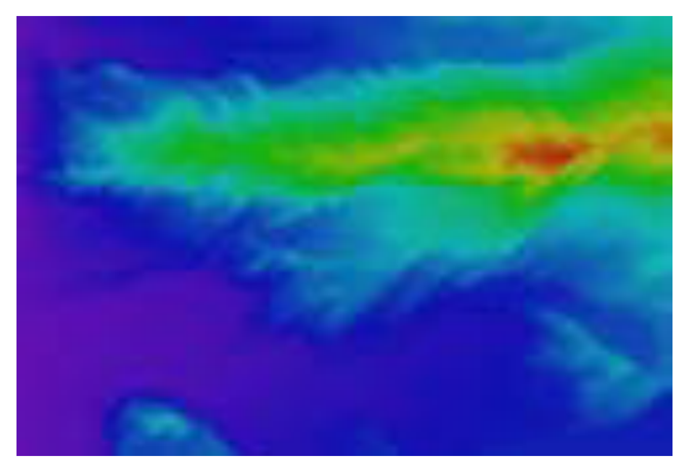
$$\mathbf{M}(i, j) = Energy(i, j) + \min\big(\mathbf{M}(i-1, j-1), \mathbf{M}(i-1, j), \mathbf{M}(i-1, j+1)\big)$$

$\boxed{1}$	3	0
$\boxed{2}$	8	9
5	$\boxed{2}$	6

$\boxed{1}$	3	0
$\boxed{3}$	8	9
8	$\boxed{5}$	14



Energy matrix (gradient magnitude)



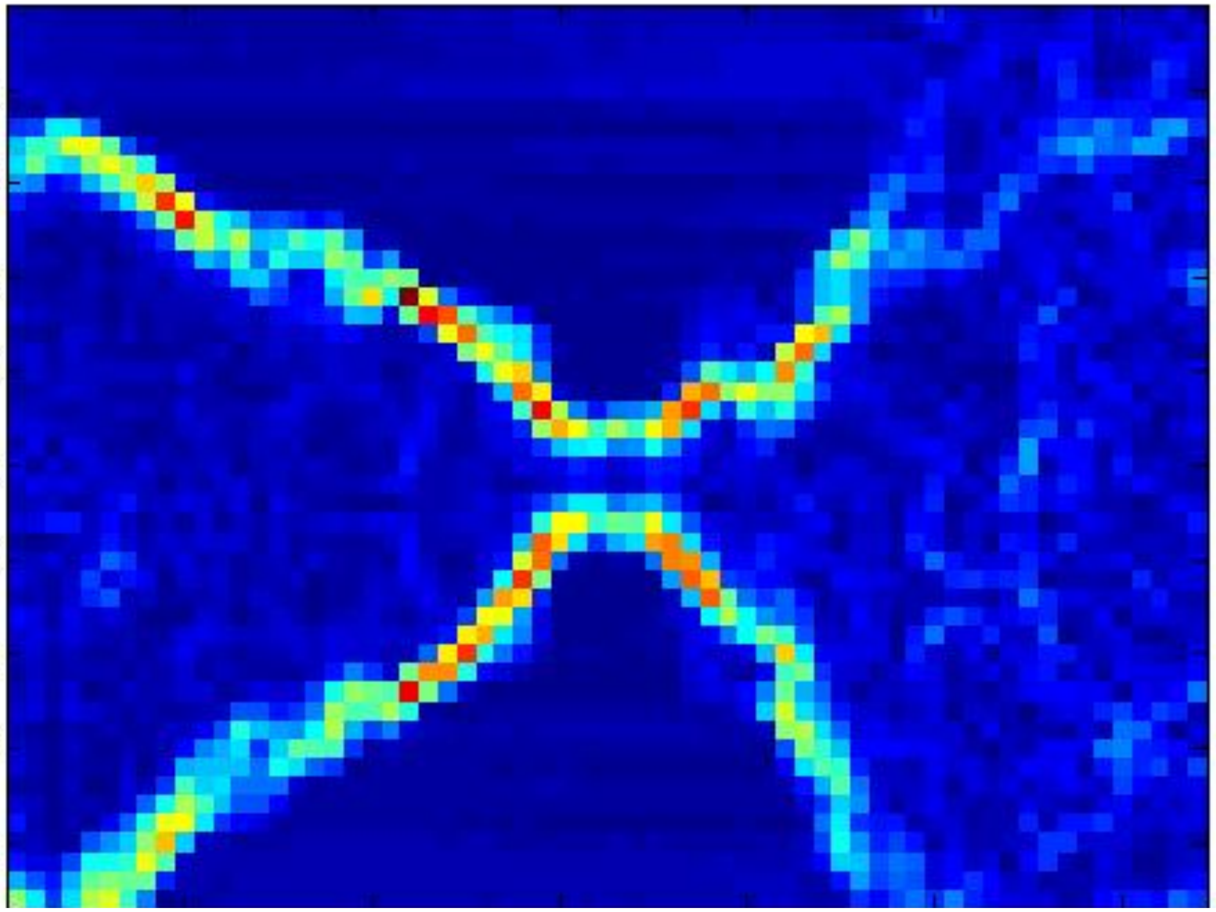
M matrix (for vertical seams)

Slide credit: Kristen Grauman

Real image example

Original Image

Energy Map



Blue = low energy
Red = high energy

Slide credit: Kristen Grauman

Other notes on seam carving

- Analogous procedure for horizontal seams
- Can also insert seams to *increase* size of image in either dimension
  - Duplicate optimal seam, averaged with neighbors
- Other energy functions may be plugged in
  - E.g., color-based, interactive,…
- Can use combination of vertical and horizontal seams

Seam Carving

Why did it fail?

Original

Resized

Why did it fail?

Original

Resized