Lecture:
Edge Detection

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

What we will learn today

- Edge detection
- Image Gradients
- A simple edge detector
- Sobel edge detector
- Canny edge detector

Some background reading:
Forsyth and Ponce, Computer Vision, Chapter 8

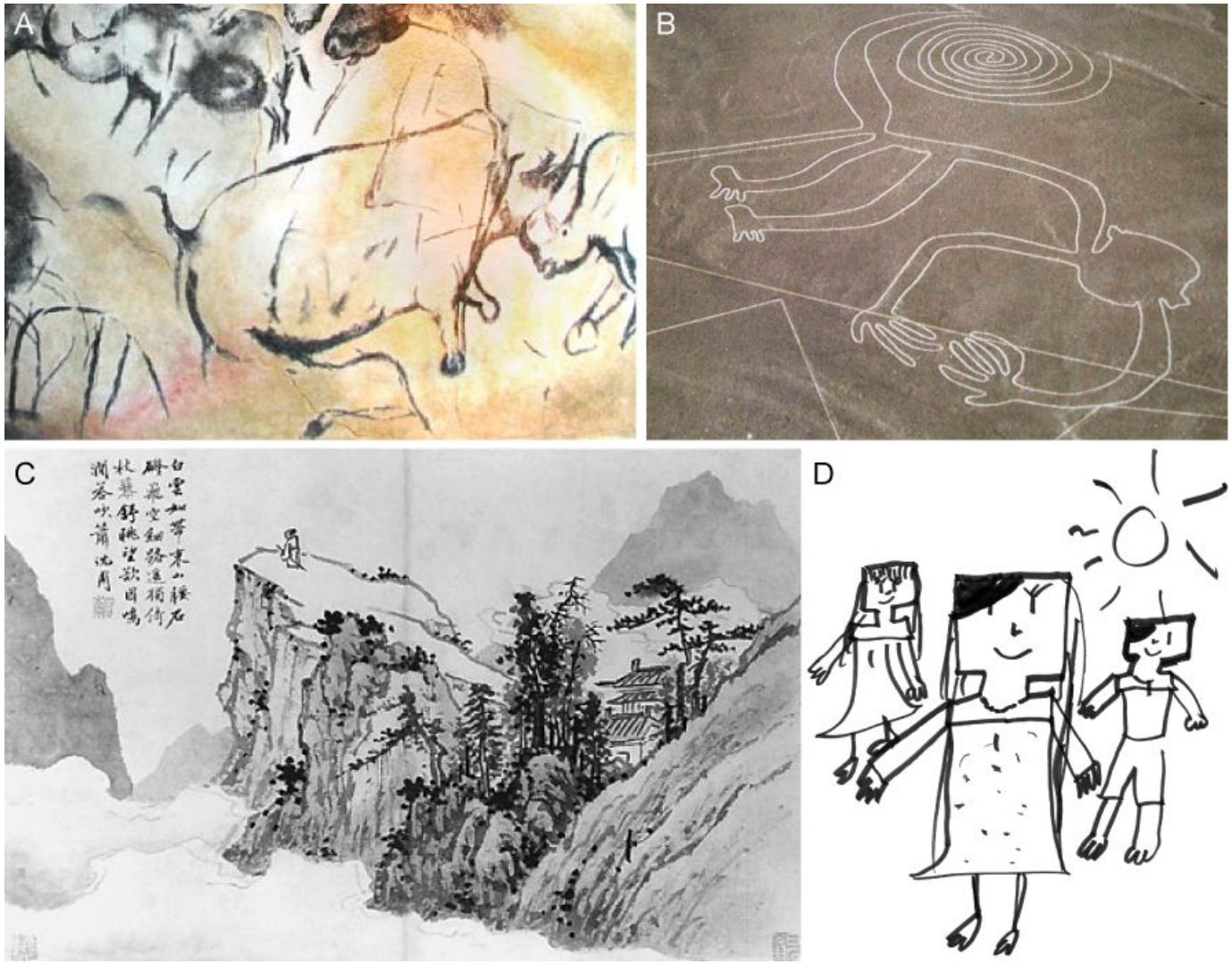
Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

What we will learn today

- Edge detection
- Image Gradients
- A simple edge detector
- Sobel edge detector
- Canny edge detector

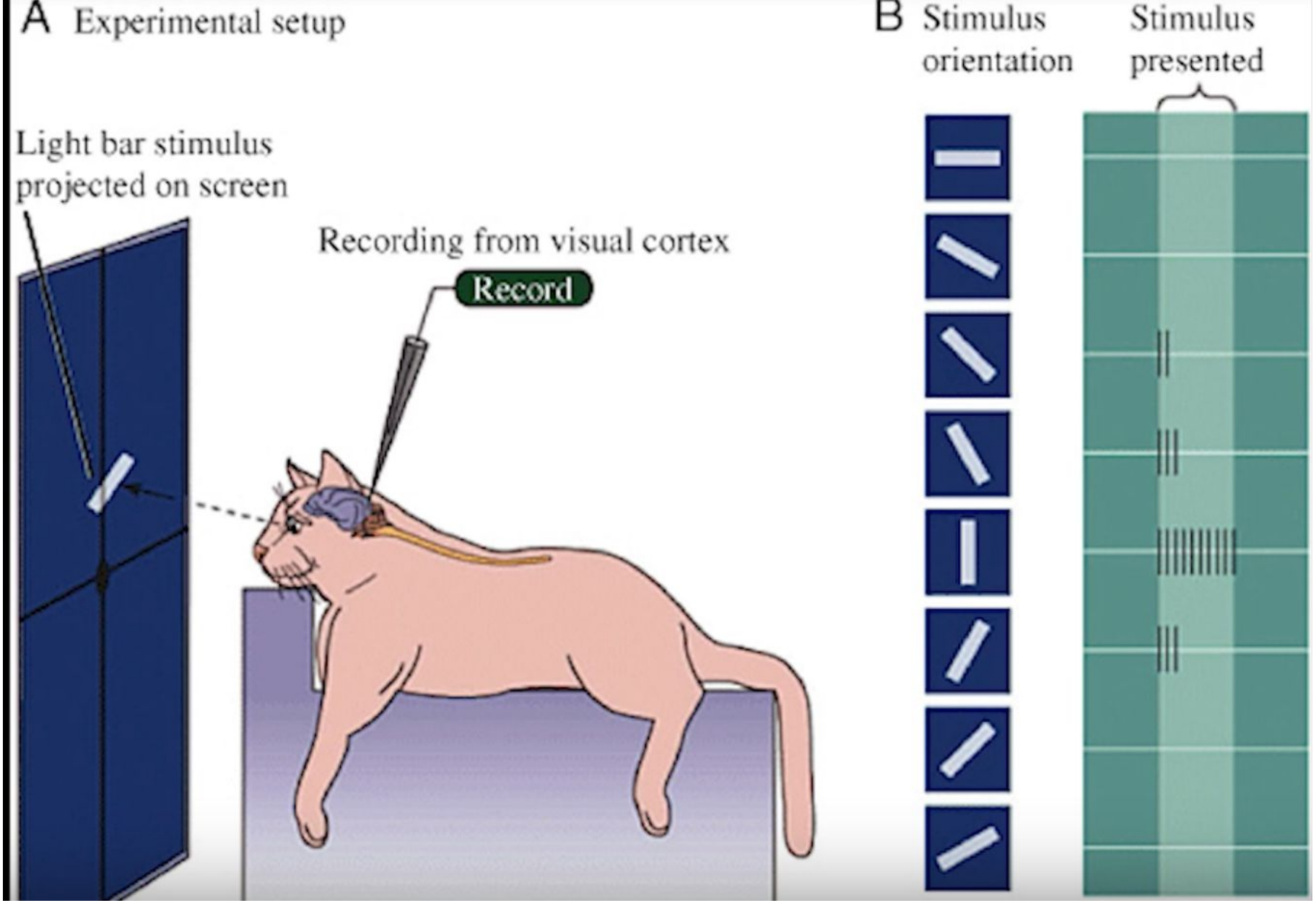
Some background reading:
Forsyth and Ponce, Computer Vision, Chapter 8

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna



(A) Cave painting at Chauvet, France, about 30,000 B.C.;
(B) Aerial photograph of the picture of a monkey as part of the Nazca Lines geoglyphs, Peru, about 700 – 200 B.C.;
(C) Shen Zhou (1427-1509 A.D.): Poet on a mountain top, ink on paper, China;
(D) Line drawing by 7-year old I. Lleras (2010 A.D.).

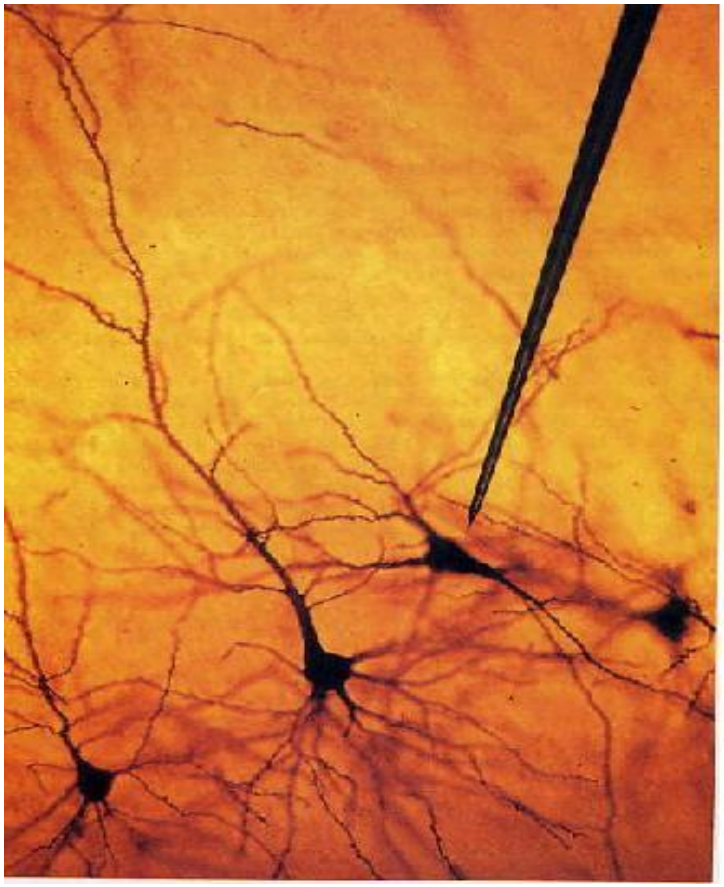
Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

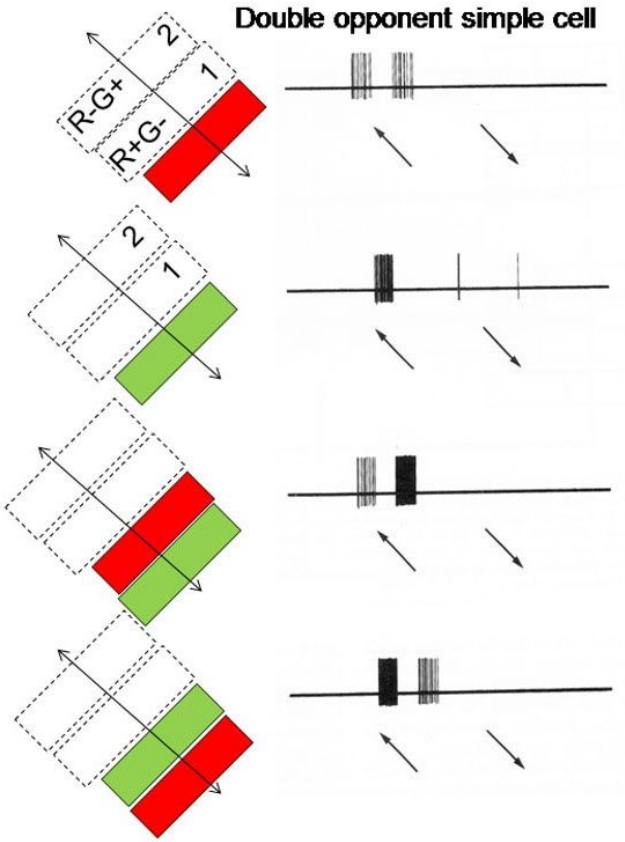


Hubel & Wiesel, 1960s

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

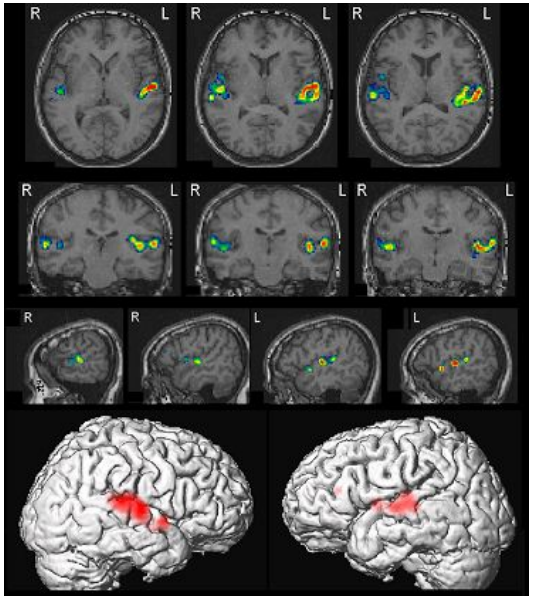
We know edges are special from human (mammalian) vision studies

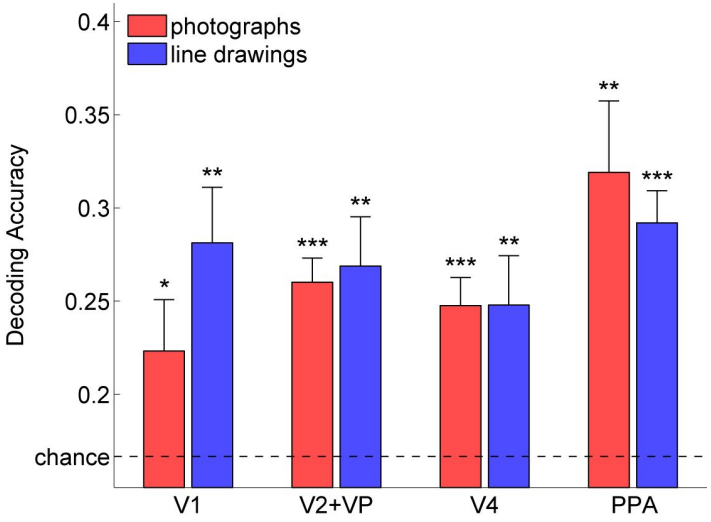




Hubel & Wiesel, 1960s

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna





Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Walther, Chai, Caddigan, Beck & Fei-Fei, *PNAS, 2011*

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)

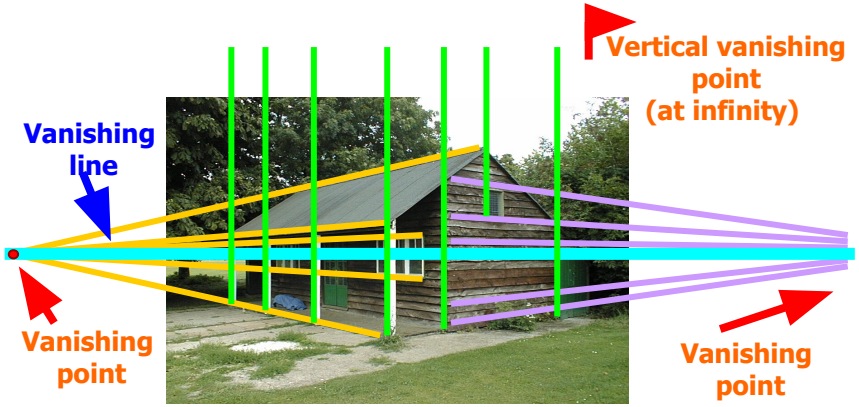
Source: D. Lowe

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Why do we care about edges?

- Extract information, recognize objects

- Recover geometry and viewpoint



Source: J. Hayes

Origins of edges

surface normal discontinuity

depth discontinuity

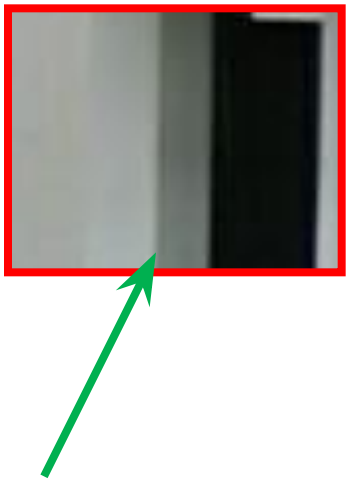
surface color discontinuity

illumination discontinuity

Source: D. Hoiem

Closeup of edges

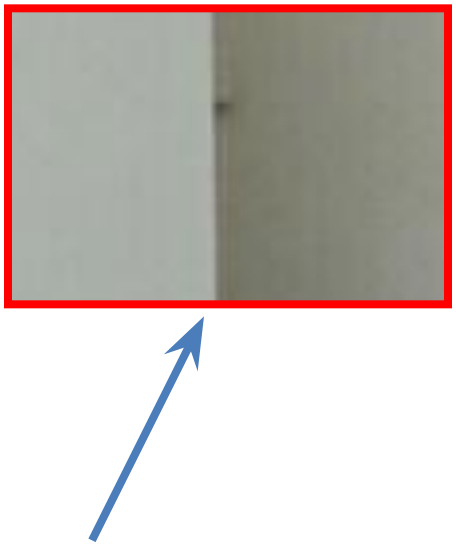
Surface normal discontinuity



Source: D. Hoiem

Closeup of edges

Depth discontinuity



Source: D. Hoiem

Closeup of edges

Surface color discontinuity

Source: D. Hoiem

What we will learn today

- Edge detection
- Image Gradients
- A simple edge detector
- Sobel edge detector
- Canny edge detector

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Derivatives in 1D

$$\frac{df}{dx} = \lim_{\Delta x \to 0} \frac{f(x) - f(x - \Delta x)}{\Delta x} = f'(x) = f_x$$

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Derivatives in 1D - example

$$y = x^2 + x^4$$

$$\frac{dy}{dx} = 2x + 4x^3$$

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Derivatives in 1D - example

$$y = x^2 + x^4 \qquad y = \sin x + e^{-x}$$

$$\frac{dy}{dx} = 2x + 4x^3 \qquad \frac{dy}{dx} = \cos x + (-1)e^{-x}$$

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Slide credit: Dr Mubarak

Discrete Derivative in 1D

$$\frac{df}{dx} = \lim_{\Delta x \to 0} \frac{f(x) - f(x - \Delta x)}{\Delta x} = f'(x)$$

$$\frac{df}{dx} = \frac{f(x) - f(x - 1)}{1} = f'(x)$$

$$\frac{df}{dx} = f(x) - f(x - 1) = f'(x)$$

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Types of Discrete derivative in 1D

Backward $$\frac{df}{dx} = f(x) - f(x-1) = f'(x)$$

Forward $$\frac{df}{dx} = f(x) - f(x+1) = f'(x)$$

Central $$\frac{df}{dx} = f(x+1) - f(x-1) = f'(x)$$

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

1D discrete derivate filters

- Backward filter: [0 1 -1]

$$f(x) - f(x-1) = f'(x)$$

- Forward: [-1 1 0]

$$f(x) - f(x+1) = f'(x)$$

- Central: [1 0 -1]

$$f(x+1) - f(x-1) = f'(x)$$

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

1D discrete derivate example

$$f(x) = 10 \quad 15 \quad 10 \quad 10 \quad 25 \quad 20 \quad 20 \quad 20$$

$$f'(x) = \quad 0 \quad 5 \quad -5 \quad 0 \quad 15 \quad -5 \quad 0 \quad 0$$

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Discrete derivate in 2D

Given function $f(x, y)$

Discrete derivate in 2D

Given function $f(x, y)$

Gradient vector $$\nabla f(x, y) = \begin{bmatrix} \frac{\partial f(x, y)}{\partial x} \\ \frac{\partial f(x, y)}{\partial y} \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \end{bmatrix}$$

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

2D discrete derivative filters

What does this filter do?

$$\frac{1}{3}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

2D discrete derivative filters

What about this filter?

$$\frac{1}{3}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

$$\frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

2D discrete derivative - example

$$I = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

2D discrete derivative - example

What happens when we apply this filter?

$$I = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

$$\frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

2D discrete derivative - example

What happens when we apply this filter?

$$I = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

$$\frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

$$I_y = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Adapted f y Krishna

2D discrete derivative - example

Now let's try the other filter!

$$I = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

$$\frac{1}{3}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

2D discrete derivative - example

What happens when we apply this filter?

$$I = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

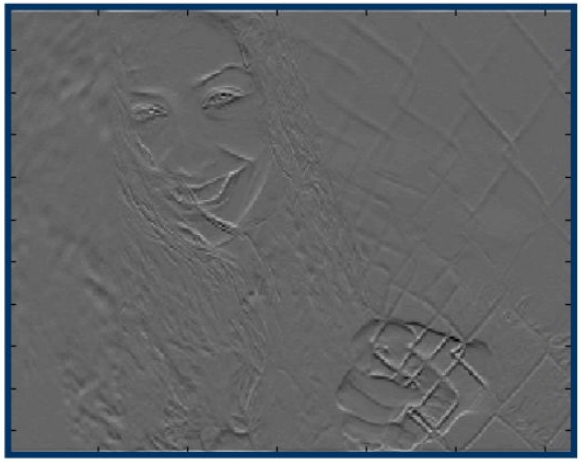
$$\frac{1}{3}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

$$I_x = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & -10 & -10 & 0 & 0 \\ 0 & -10 & -10 & 0 & 0 \\ 0 & -10 & -10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Adapted fr... Ranjay Krishna

3x3 image gradient filters

$$\frac{1}{3}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$



Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

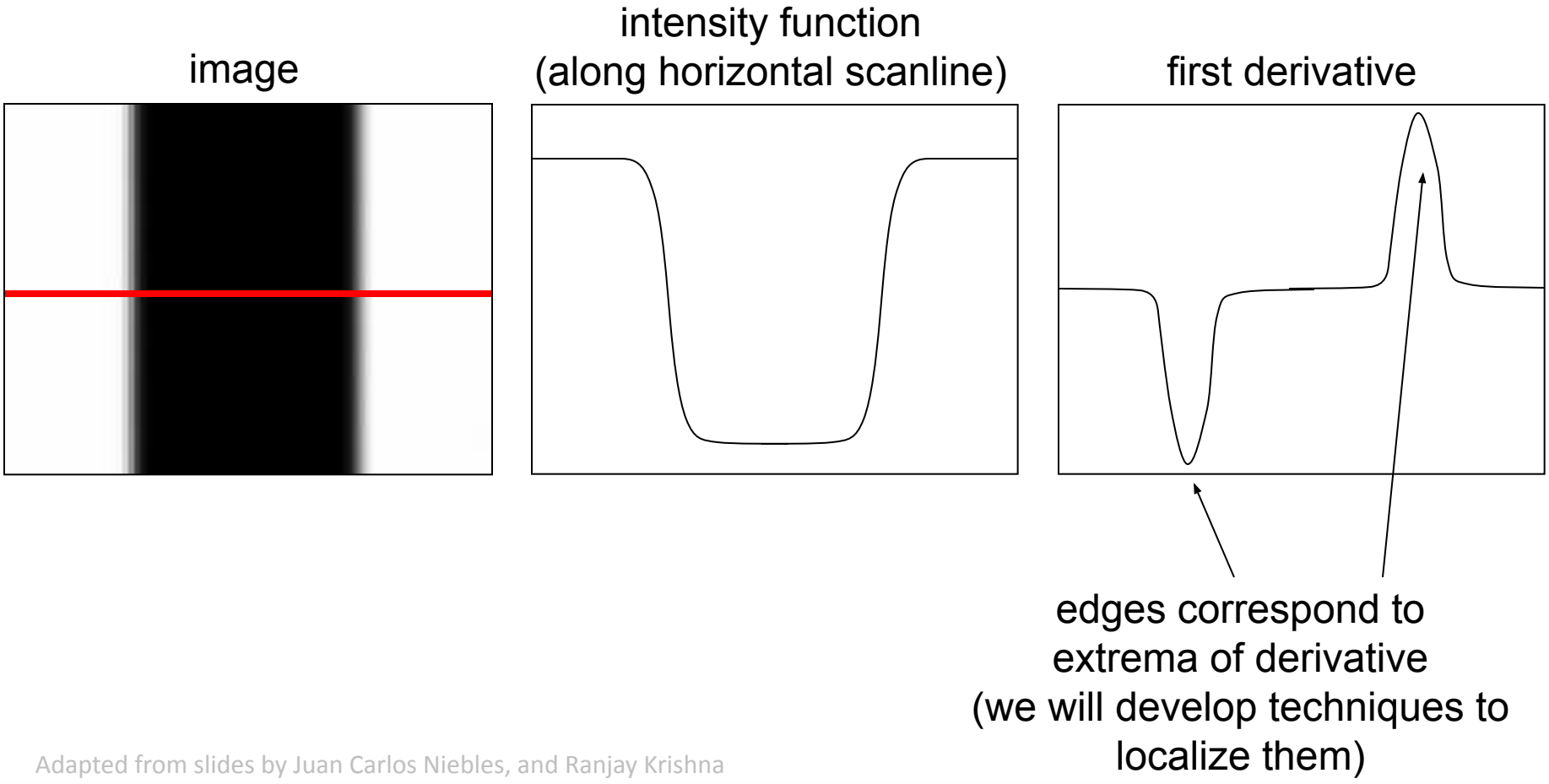
What we will learn today

- Edge detection
- Image Gradients
- A simple edge detector
- Sobel edge detector
- Canny edge detector

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Characterizing edges

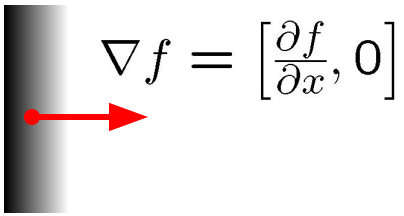
- An edge is a place of rapid change in the image intensity function

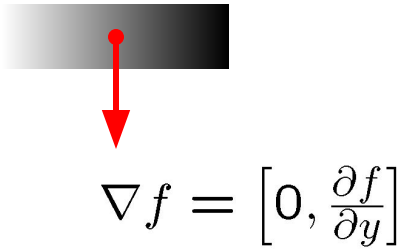


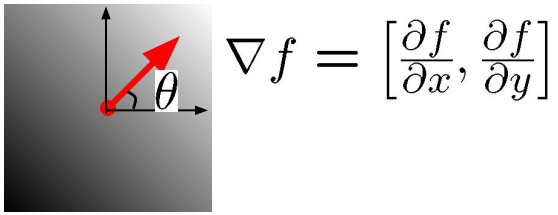
Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Image gradient

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







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

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?

The *edge strength* is given by the gradient magnitude

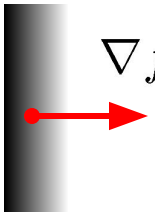
$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Source: Steve Seitz

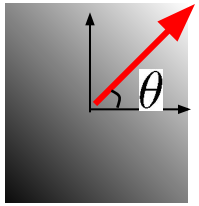
Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Image gradient

- The gradient of ar $\left[\frac{f}{x}, \frac{\partial f}{\partial y}\right]$



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



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

The gradient vector points i crease in intensity

The gradient direction is giv

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

The *edge strength* is given by the gradient magnitude

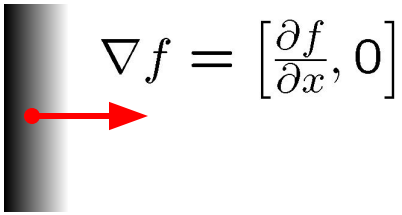
$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Source: Steve Seitz

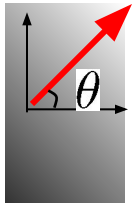
Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Image gradient

- The gradient of $\left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$



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



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

The gradient vector poin ase in intensity

The gradient direction is

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

The *edge strength* is given by the gradient magnitude

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

Source: Steve Seitz

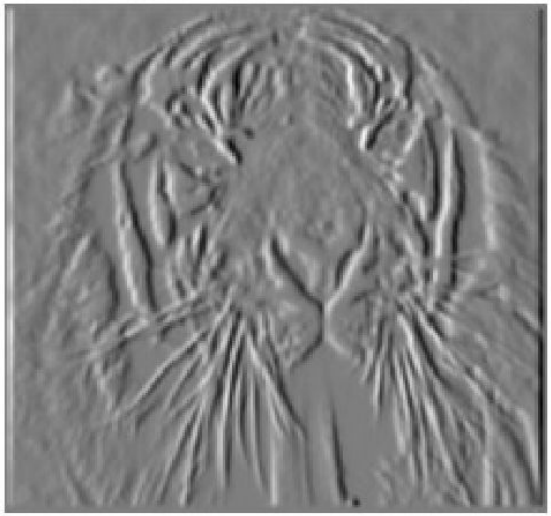
Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

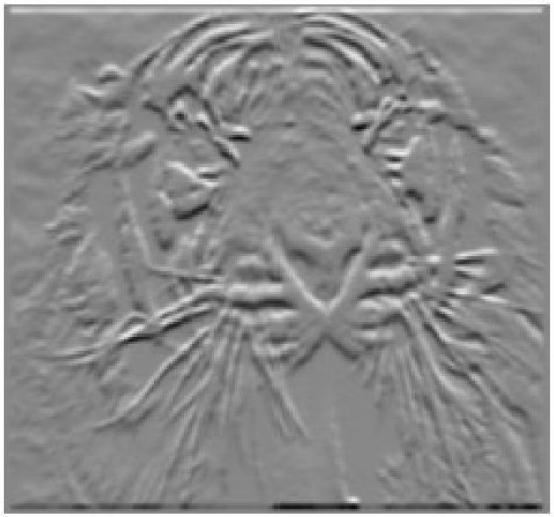
Finite differences: example

Original Image

Gradient magnitude

x-direction



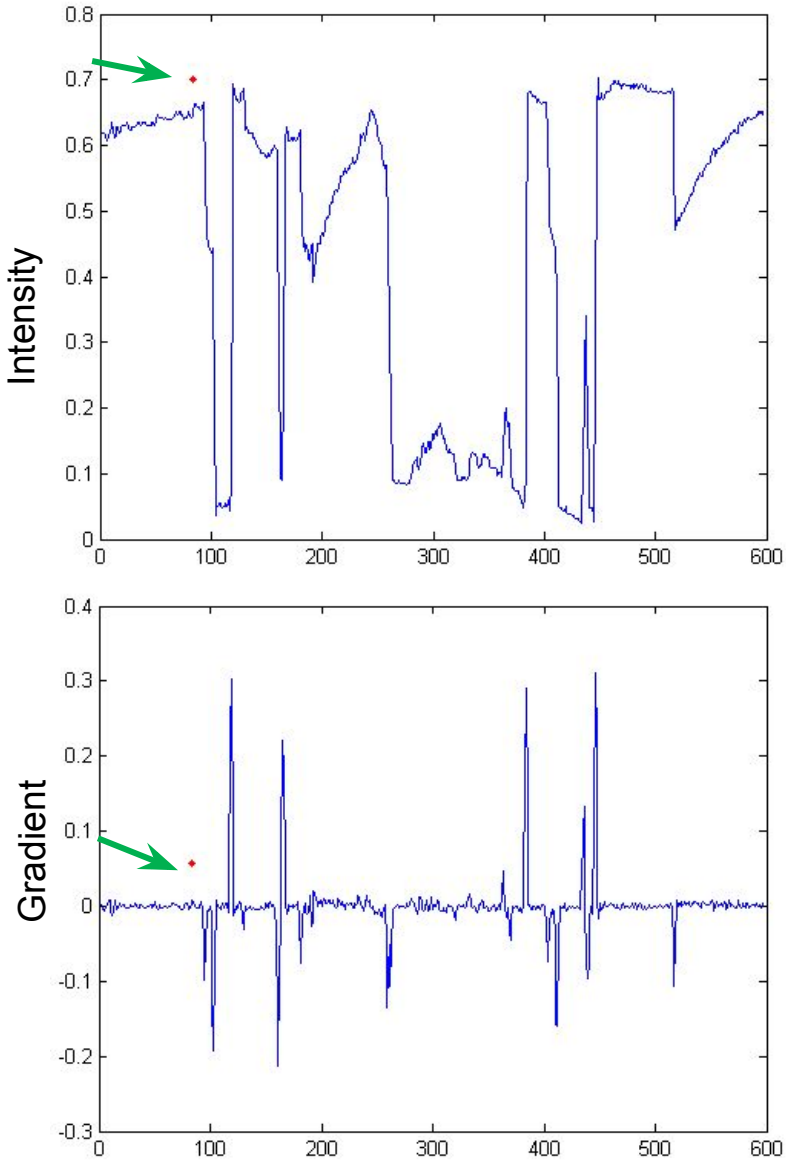


y-direction

- Which one is the gradient in the x-direction? How about y-direction?

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Intensity profile

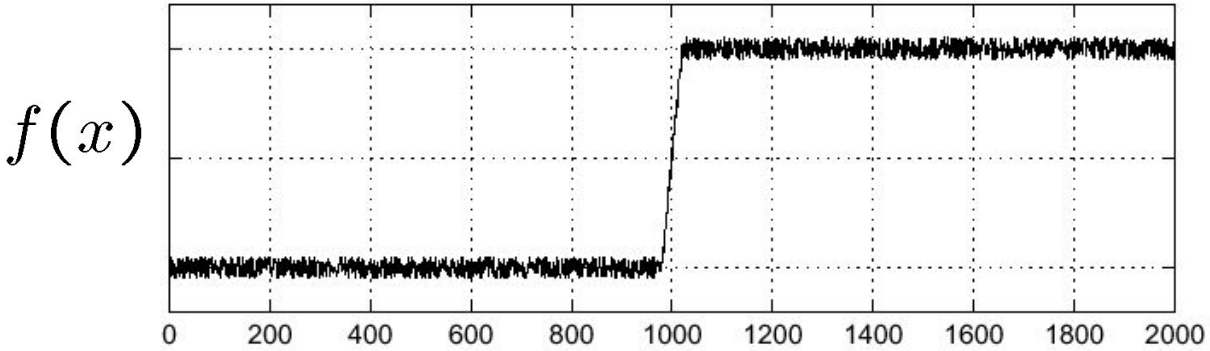


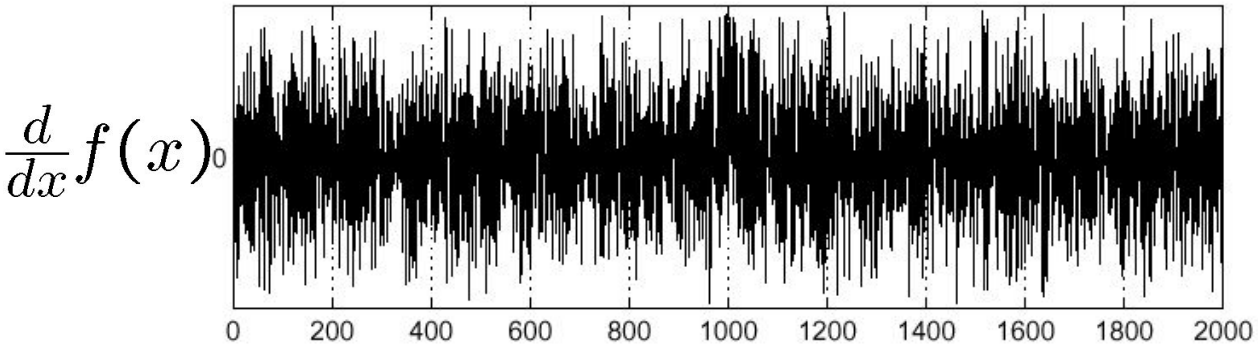
Source: D. Hoiem

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

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

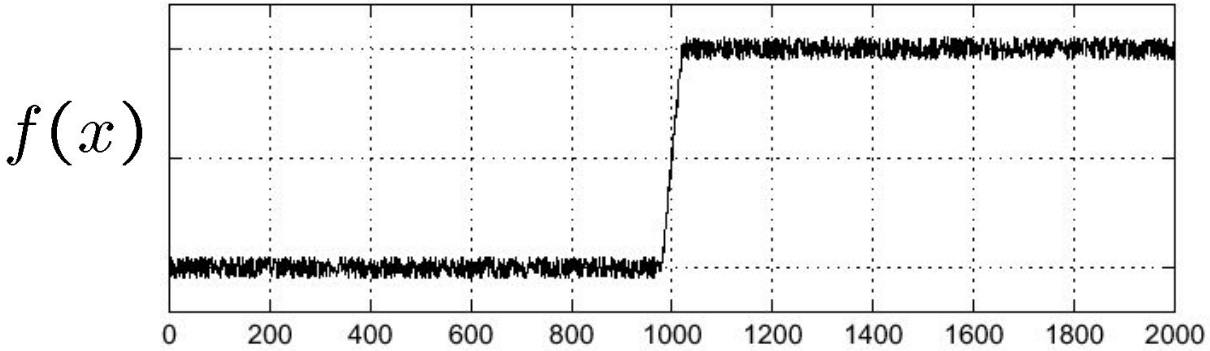
Understanding *noise*

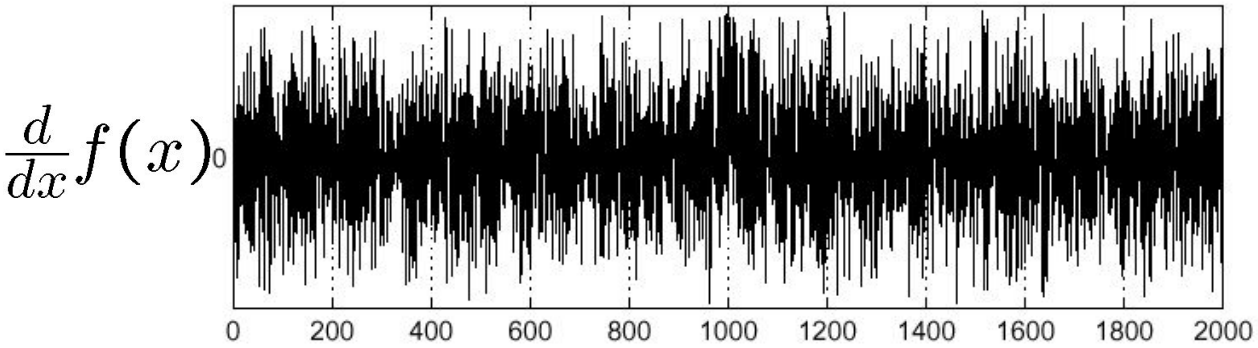
Concepts to have an understanding of:

- The conceptual relationship between noise and input scale
  - Something *noise* in a scale can be the texture or even the shape in another
- The processing-wise relationship between noise, smoothing and filter scale
  - Input down-sing is like smoothing + larger filter
  - We'll soon understand this mathematically too

Effects of noise

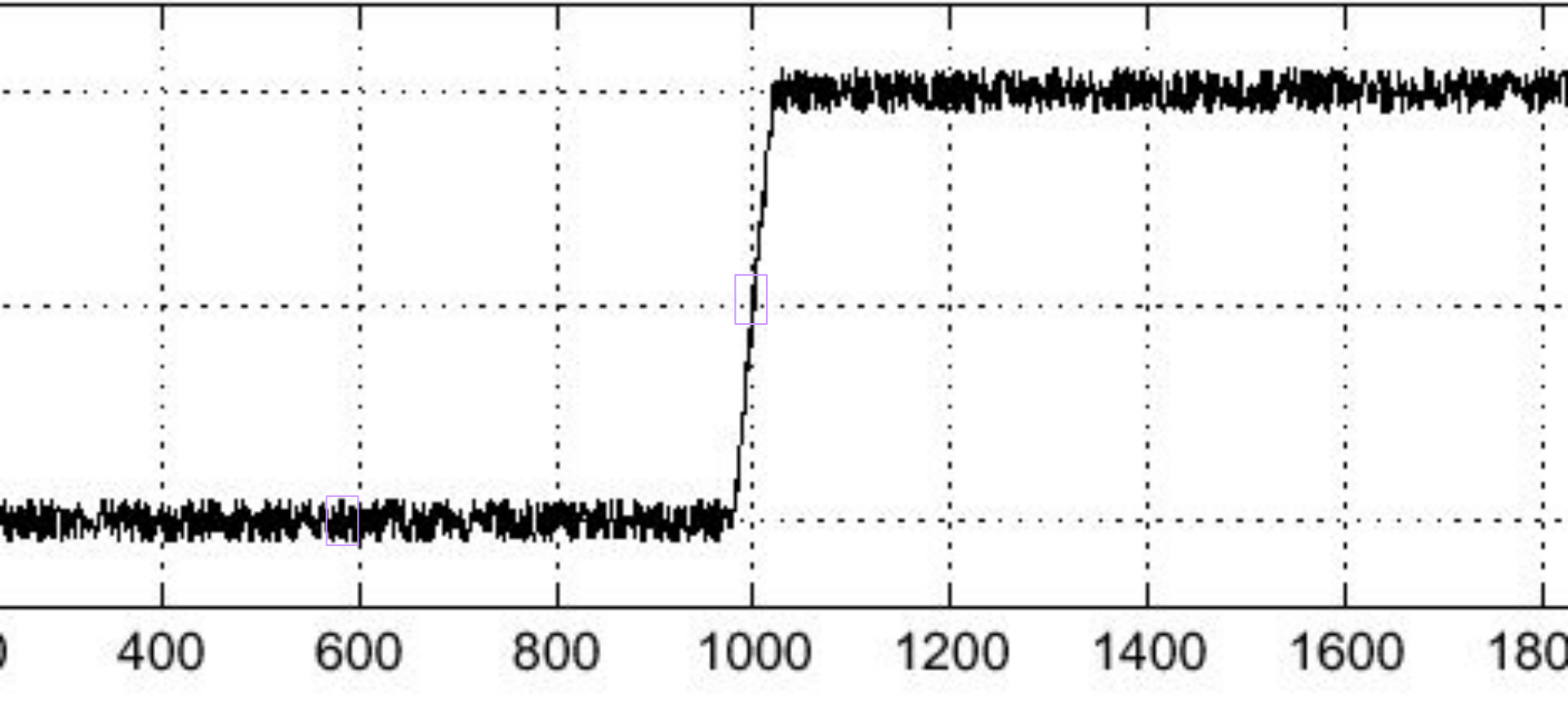
- Consider a single row or column of the image
  - Plotting intensity as a function of position gives a signal





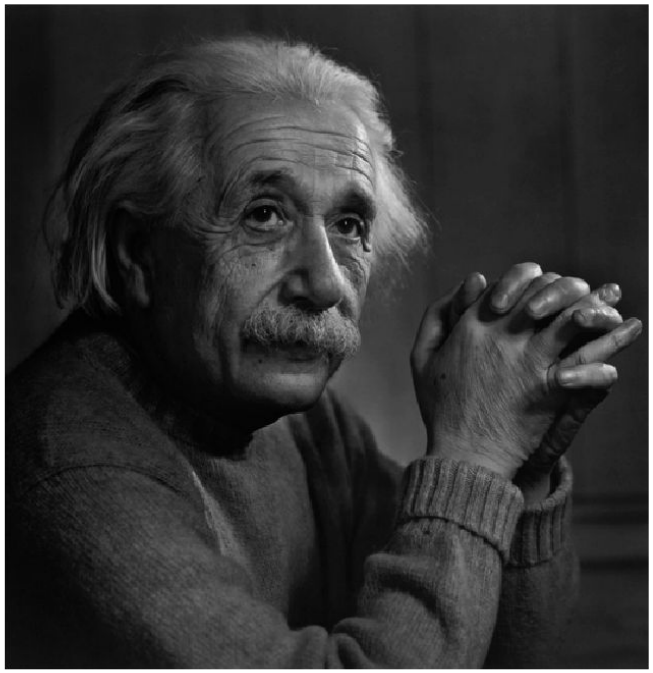
Where is the edge?

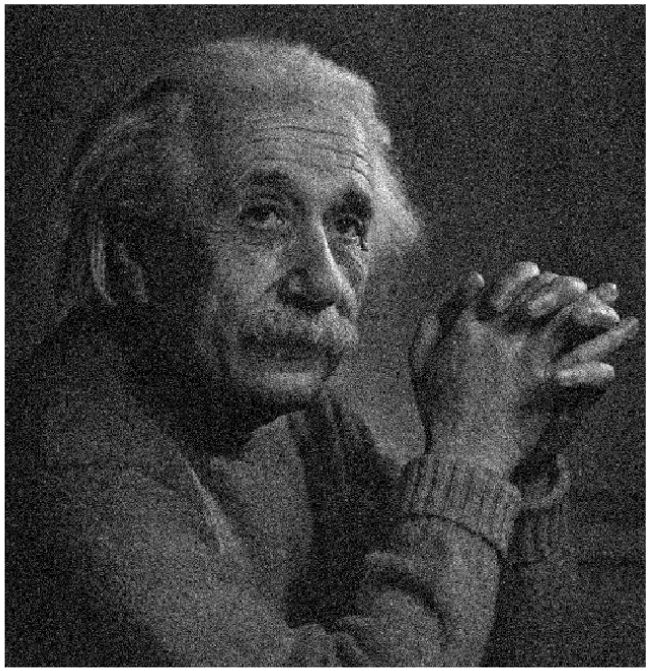
Source: S. Seitz



Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Effects of noise





Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Effects of noise

- Finite 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 is to be done?

Source: D. Forsyth

Effects of noise

- Finite 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 is to be done?
  - Smoothing the image should help, by forcing pixels different to their neighbors (=noise pixels?) to look more like neighbors

Source: D. Forsyth

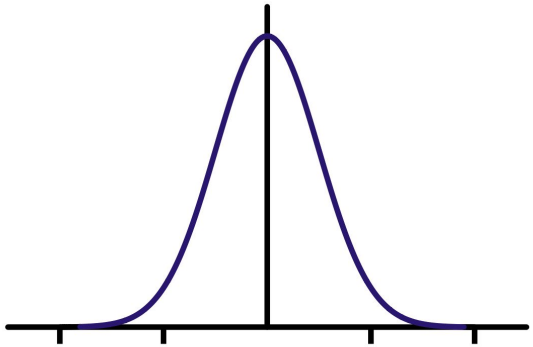
Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Smoothing with different filters

- Mean smoothing

$$\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

- Gaussian (smoothing * derivative)

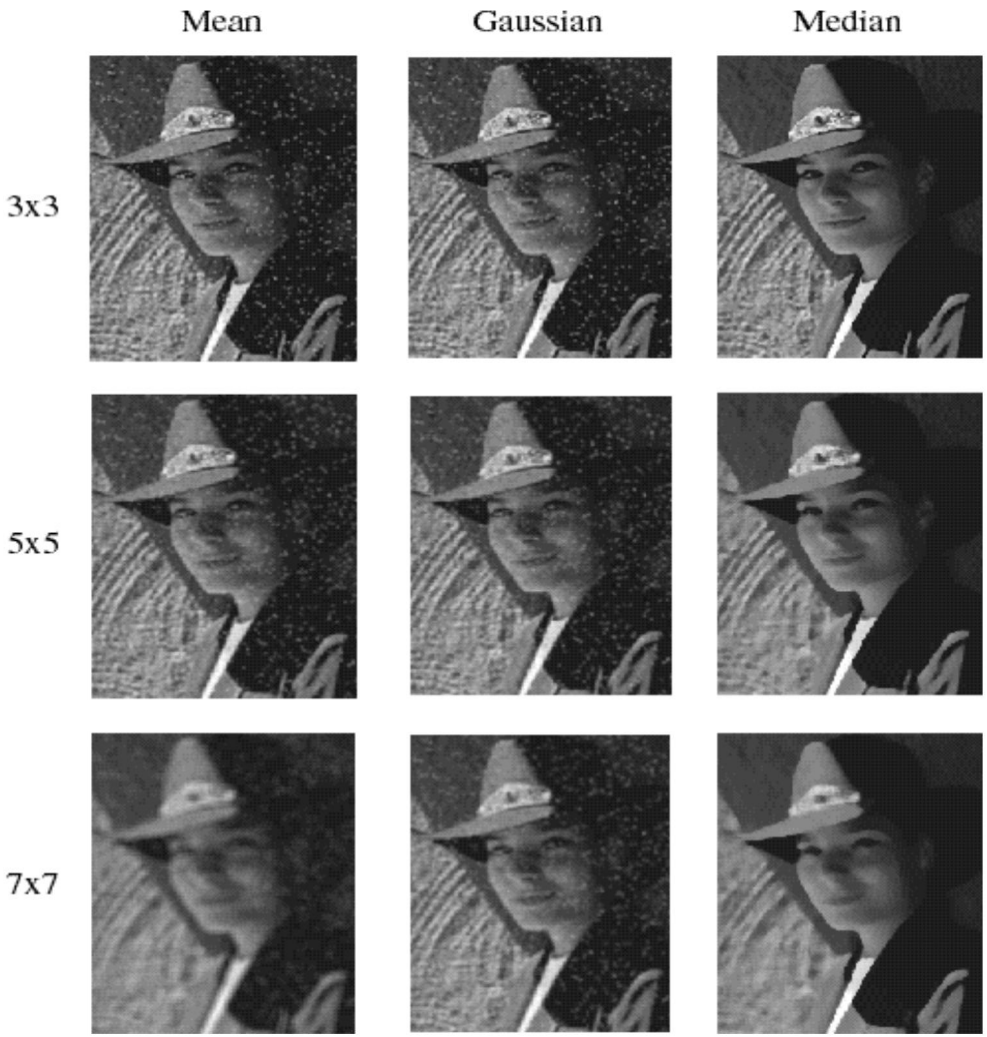


$$\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

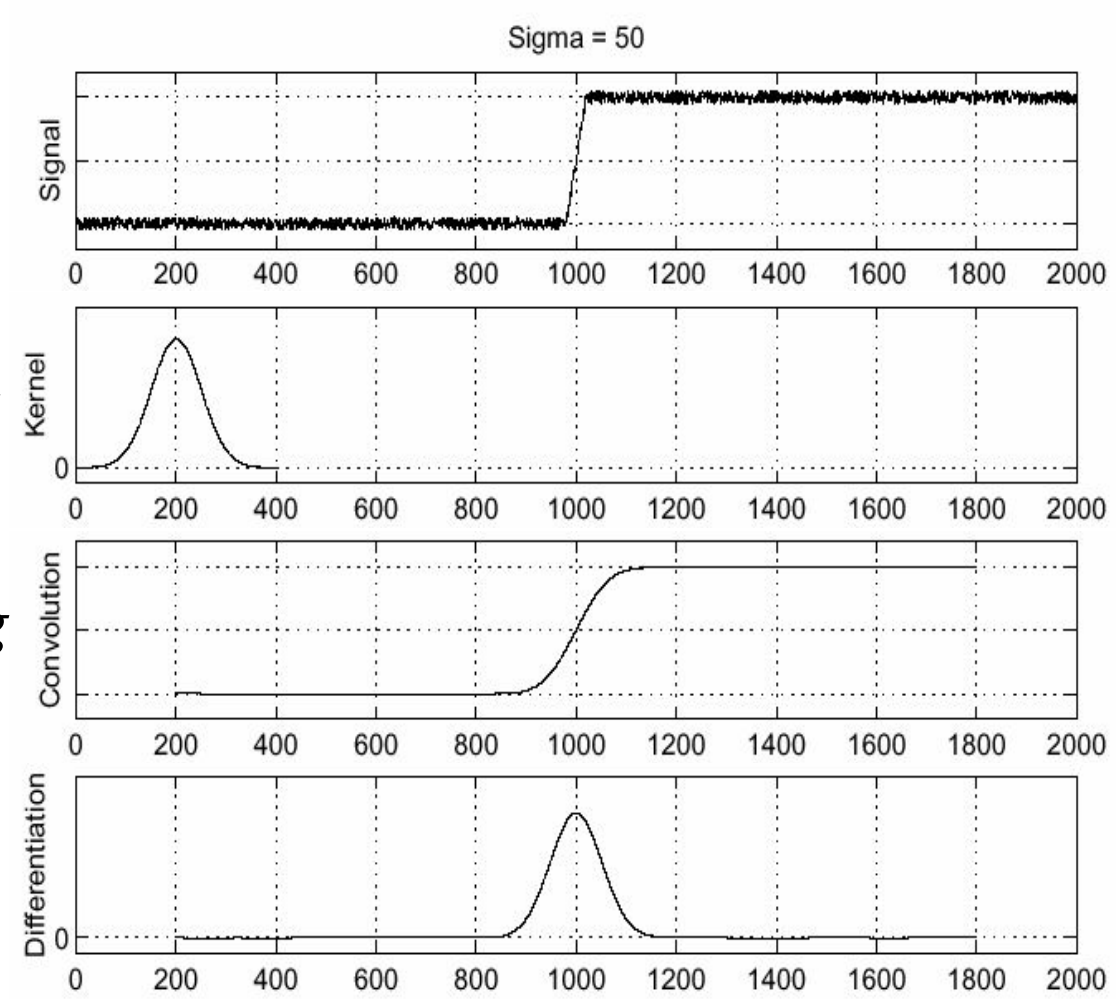
Slide credit: Steve Seitz

Smoothing with different filters



Slide credit: Steve Seitz

Solution: smooth first



f

g

$f * g$

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

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

Source: S. Seitz

Derivative theorem of convolution

- This theorem gives us a very useful property:

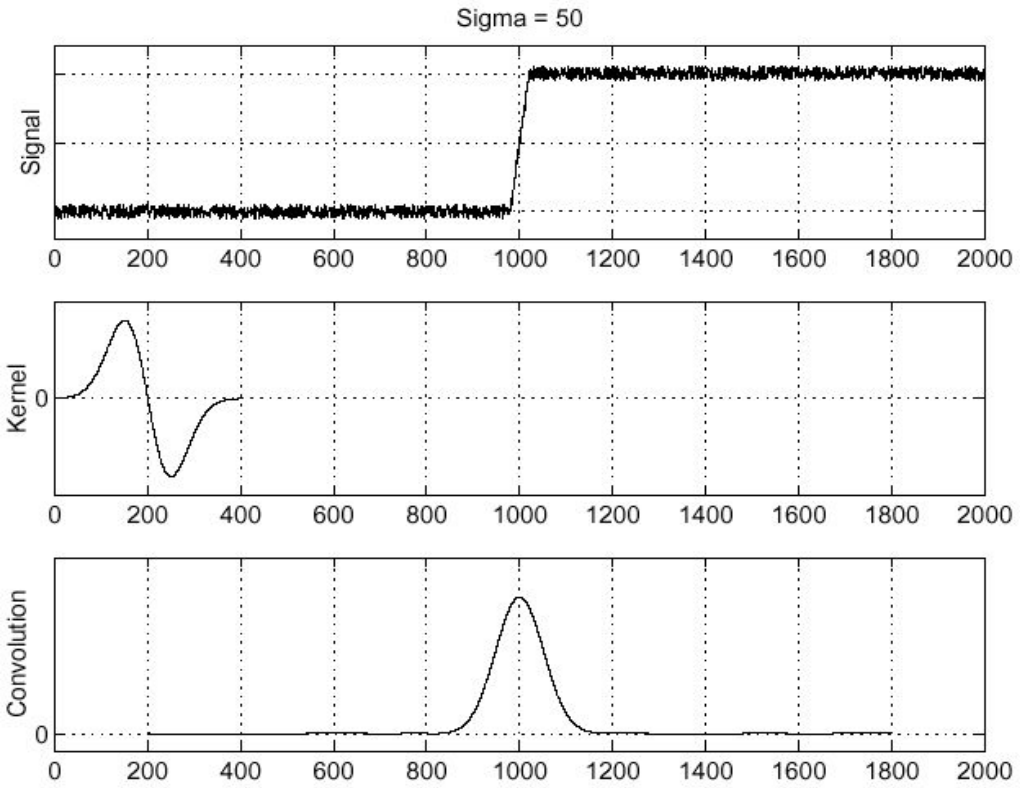
$$\frac{d}{dx}(f * g) = f * \frac{d}{dx} g$$

- This saves us one operation:

f

$\frac{d}{dx} g$

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



Source: S. Seitz

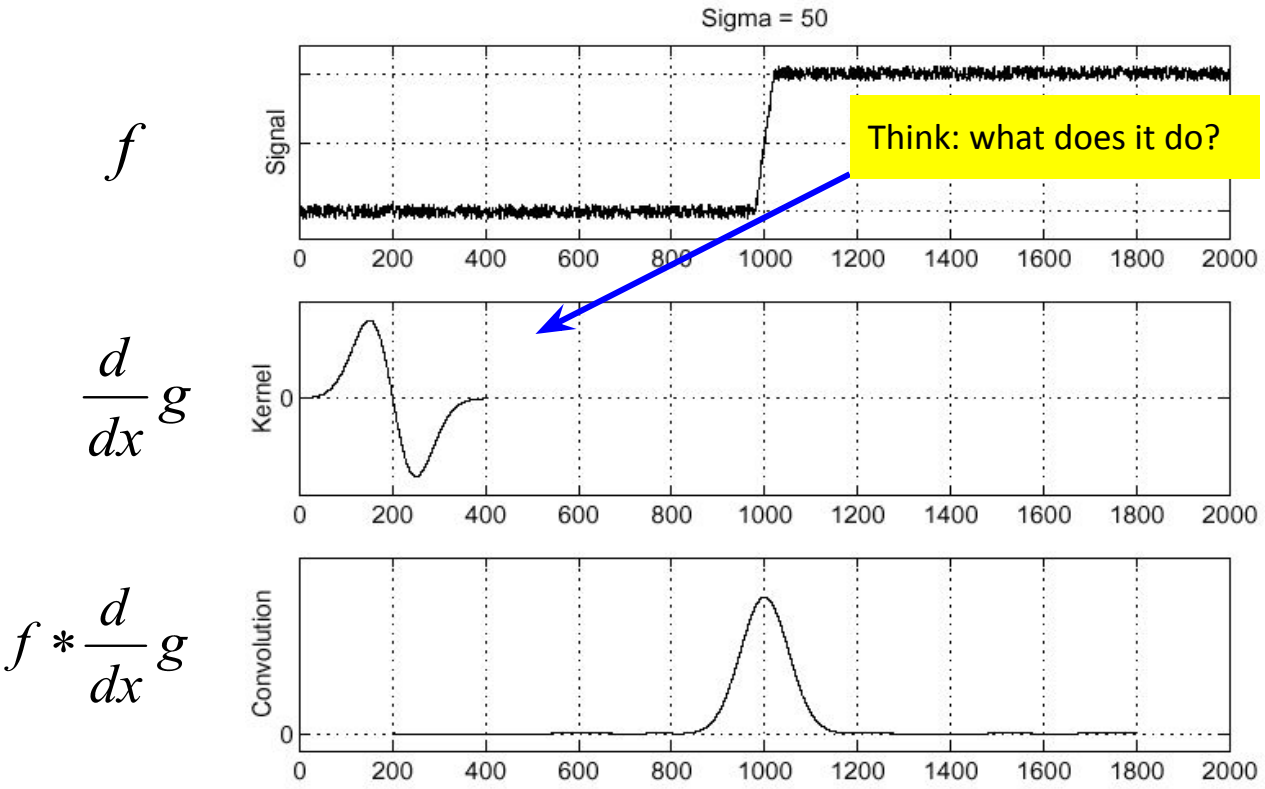
Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Derivative theorem of convolution

- This theorem gives us a very useful property:

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

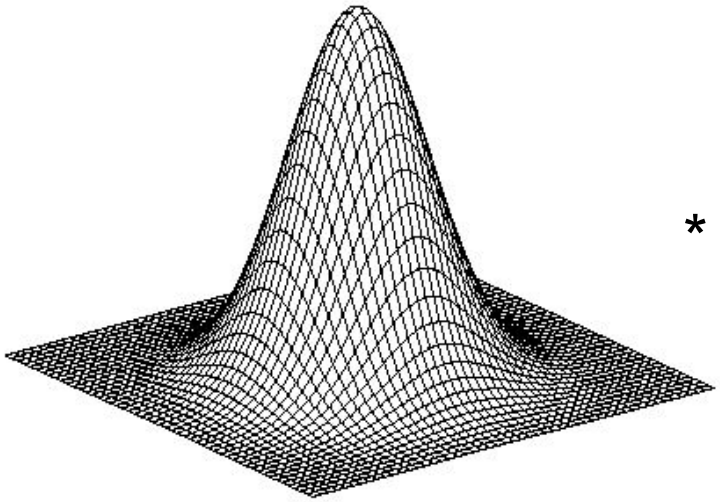
- This saves us one operation:



Source: S. Seitz

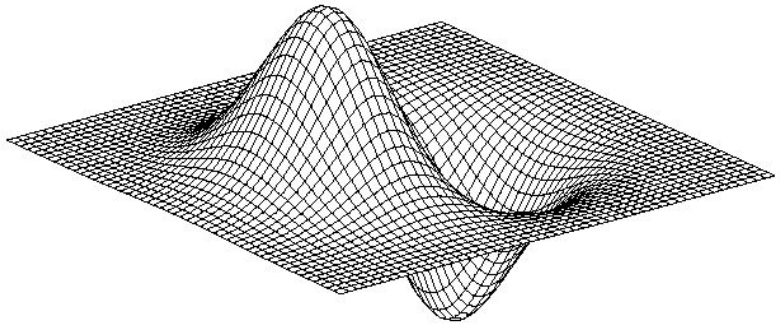
Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Derivative of Gaussian filter



2D-gaussian

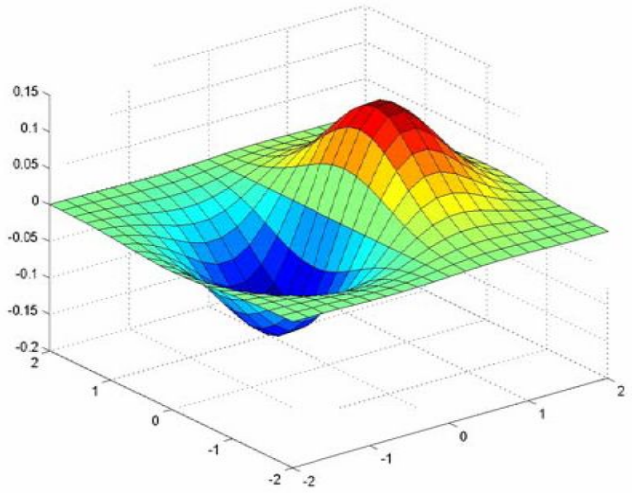
* [1 0 -1] =



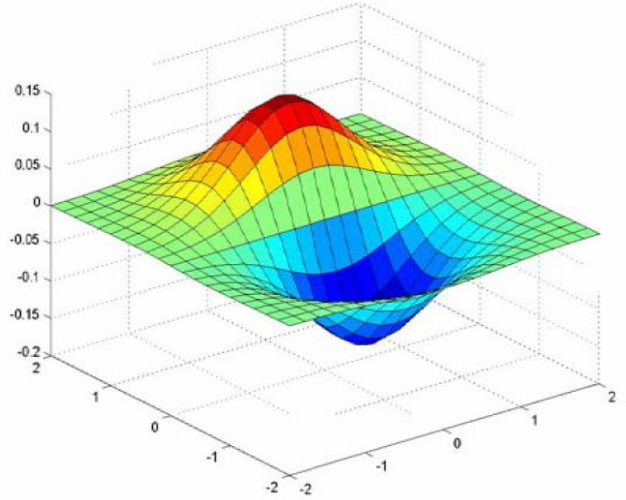
x - derivative

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

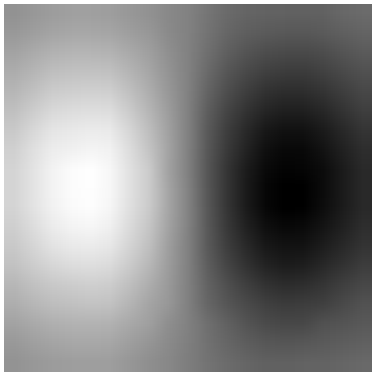
Derivative of Gaussian filter

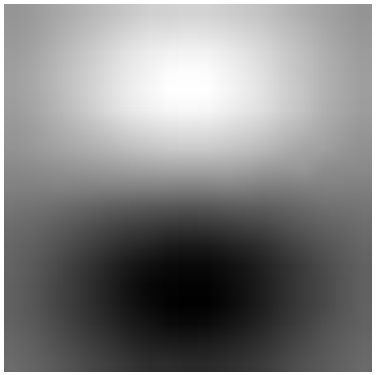


x-direction



y-direction



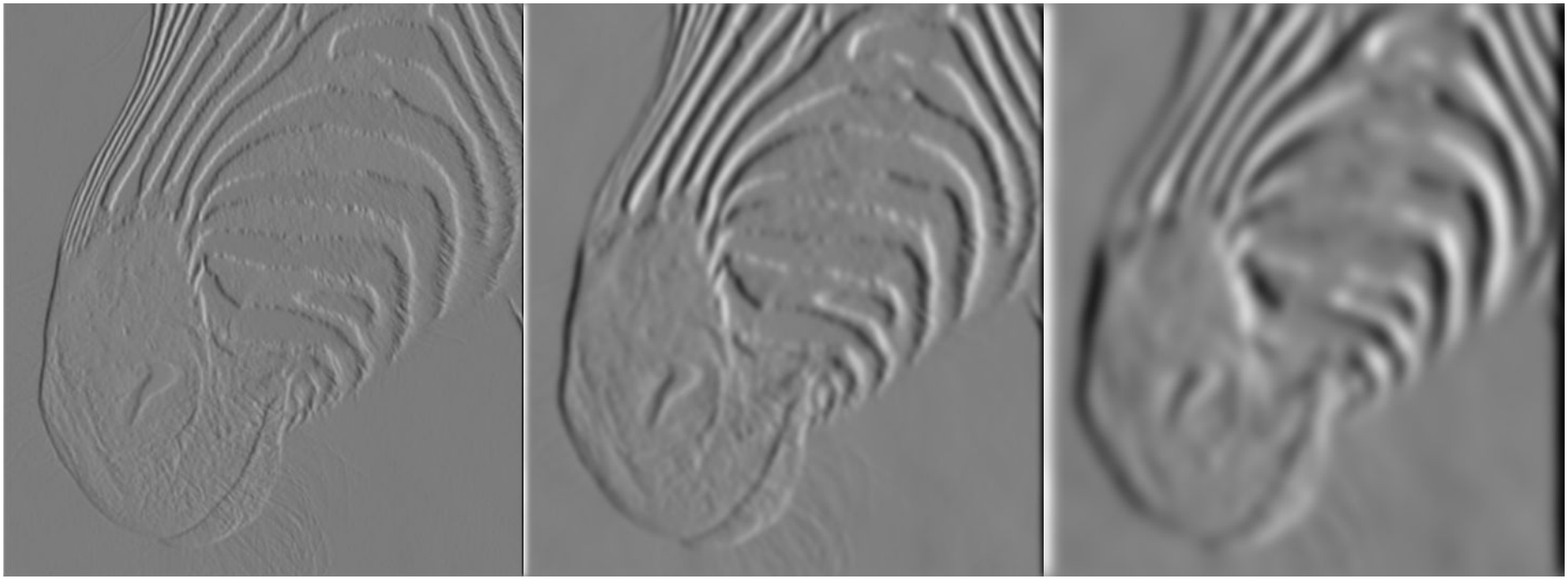


Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Derivative of Gaussian filter

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Tradeoff between smoothing at different scales



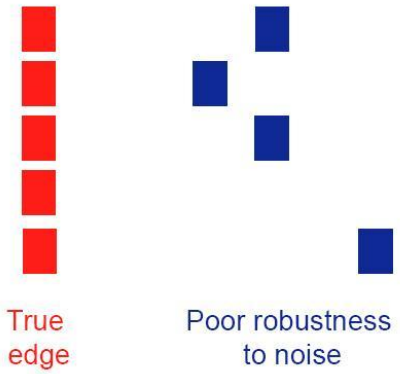
1 pixel 3 pixels 7 pixels

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

Source: D. Forsyth

Designing an edge detector

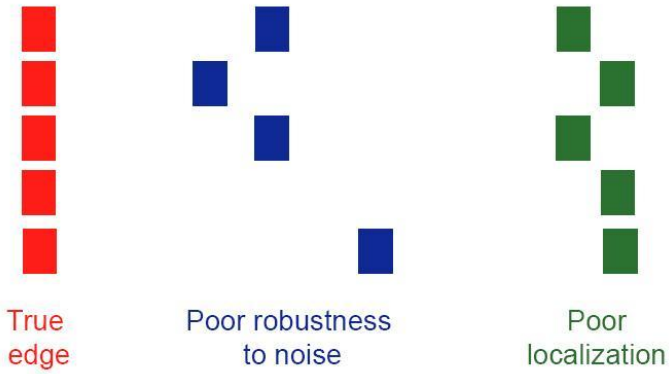
- Criteria for an “optimal” edge detector:
  - **Good detection:** the optimal detector must minimize the probability of false positives (detecting spurious edges caused by noise), as well as that of false negatives (missing real edges)



Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Designing an edge detector

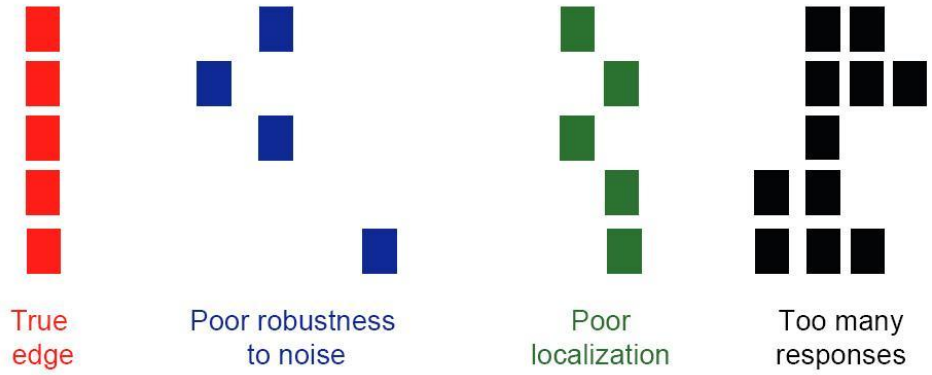
- Criteria for an "optimal" edge detector:
  - **Good detection:** the optimal detector must minimize the probability of false positives (detecting spurious edges caused by noise), as well as that of false negatives (missing real edges)
  - **Good localization:** the edges detected must be as close as possible to the true edges



Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Designing an edge detector

- Criteria for an "optimal" edge detector:
  - **Good detection:** the optimal detector must minimize the probability of false positives (detecting spurious edges caused by noise), as well as that of false negatives (missing real edges)
  - **Good localization:** the edges detected must be as close as possible to the true edges
  - **Single response:** the detector must return one point only for each true edge point; that is, minimize the number of local maxima around the true edge



Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

What we will learn today

- Edge detection
- Image Gradients
- A simple edge detector
- Sobel Edge detector
- Canny edge detector

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Sobel Operator

- uses two 3×3 kernels which are convolved with the original image to calculate approximations of the derivatives
- one for horizontal changes, and one for vertical

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} \qquad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Sobel Operation

- Smoothing + differentiation

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} +1 & 0 & -1 \end{bmatrix}$$

Gaussian smoothing

differentiation

Intuition: take a weighted average to obtain smoothed f(x-1,y) and f(x+1,y) data.

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Sobel Operation

- Magnitude:

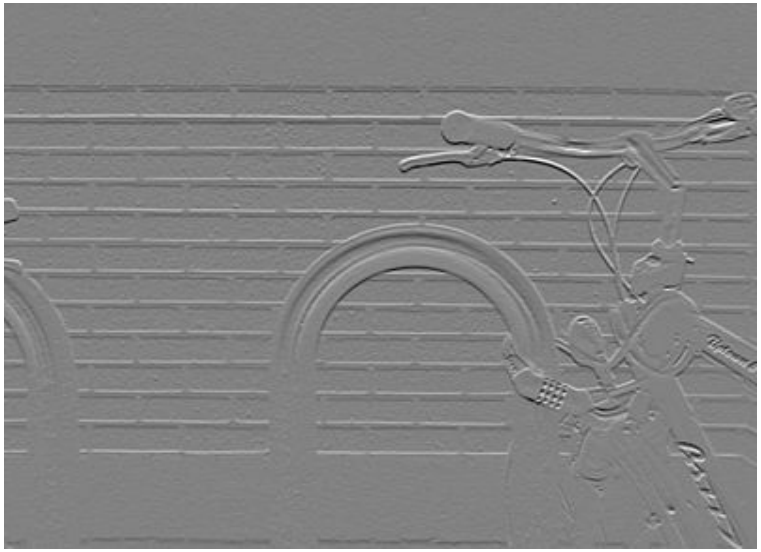
$$\mathbf{G} = \sqrt{\mathbf{G}_x{}^2 + \mathbf{G}_y{}^2}$$

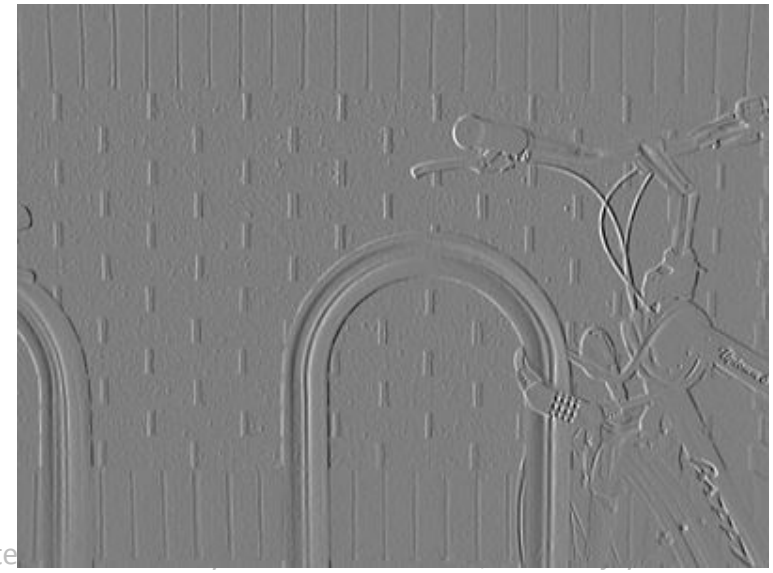
- Angle or direction of the gradient:

$$\mathbf{\Theta} = \operatorname{atan}\left(\frac{\mathbf{G}_y}{\mathbf{G}_x}\right)$$

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

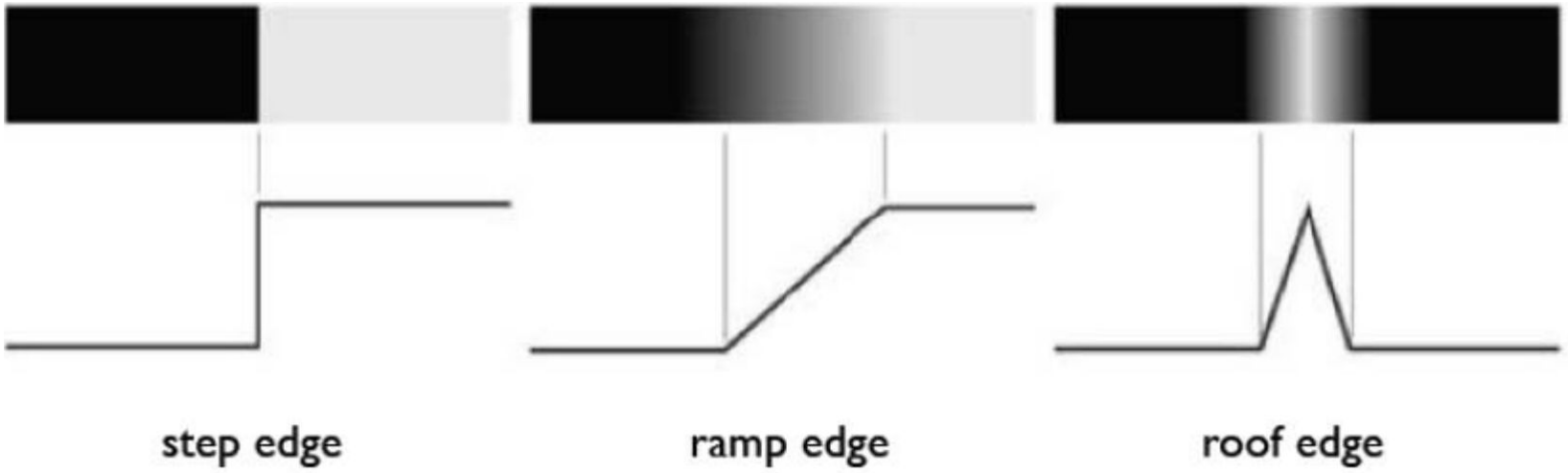
Sobel Filter example





Adapte

Sobel Filter Problems



- Poor Localization (Trigger response in multiple adjacent pixels)
- Thresholding value favors certain directions over others
  - Can miss oblique edges more than horizontal or vertical edges
  - False negatives

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

What we will learn today

- Edge detection
- Image Gradients
- A simple edge detector
- Sobel Edge detector
- Canny edge detector

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Canny edge detector

- This is probably the most widely used classical 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.

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Canny edge detector

- Suppress Noise
- Compute gradient magnitude and direction
- Apply Non-Maximum Suppression
  - Assures minimal response
- Use hysteresis and connectivity analysis to detect edges

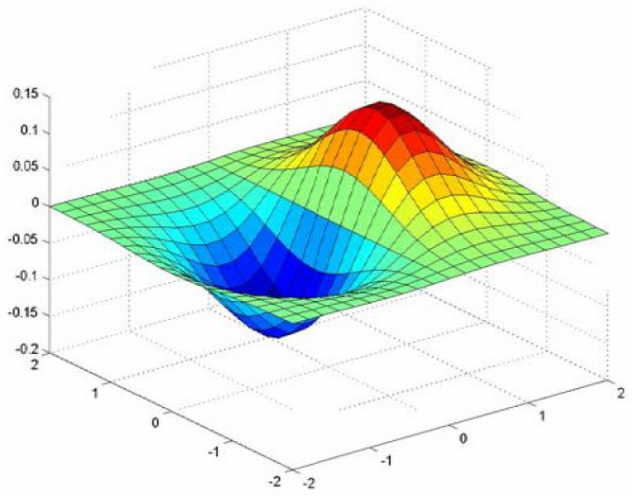
Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Example

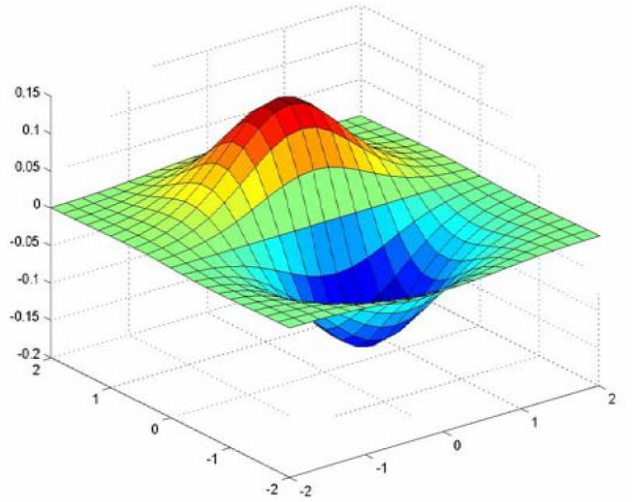
- original image

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Derivative of Gaussian filter

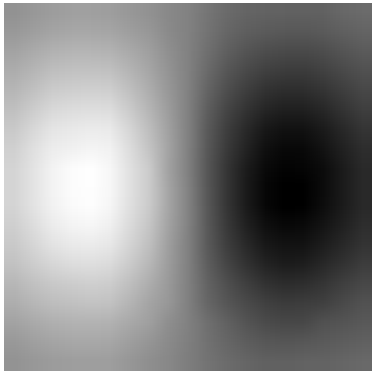


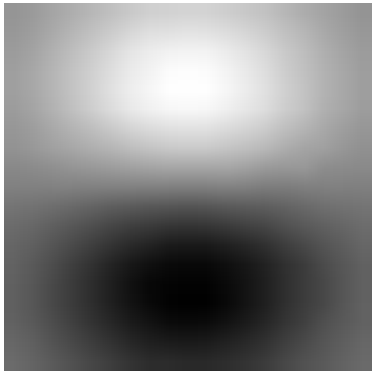
x-direction



y-direction

$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

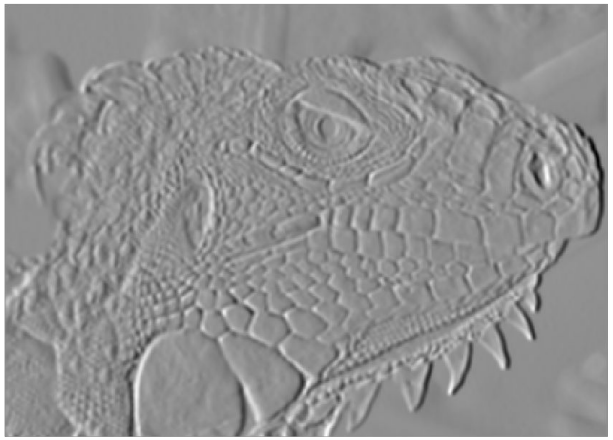




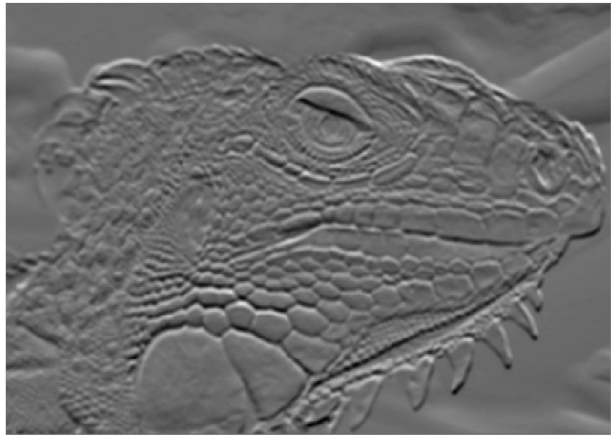
$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Compute gradients (DoG)



X-Derivative of Gaussian



Y-Derivative of Gaussian

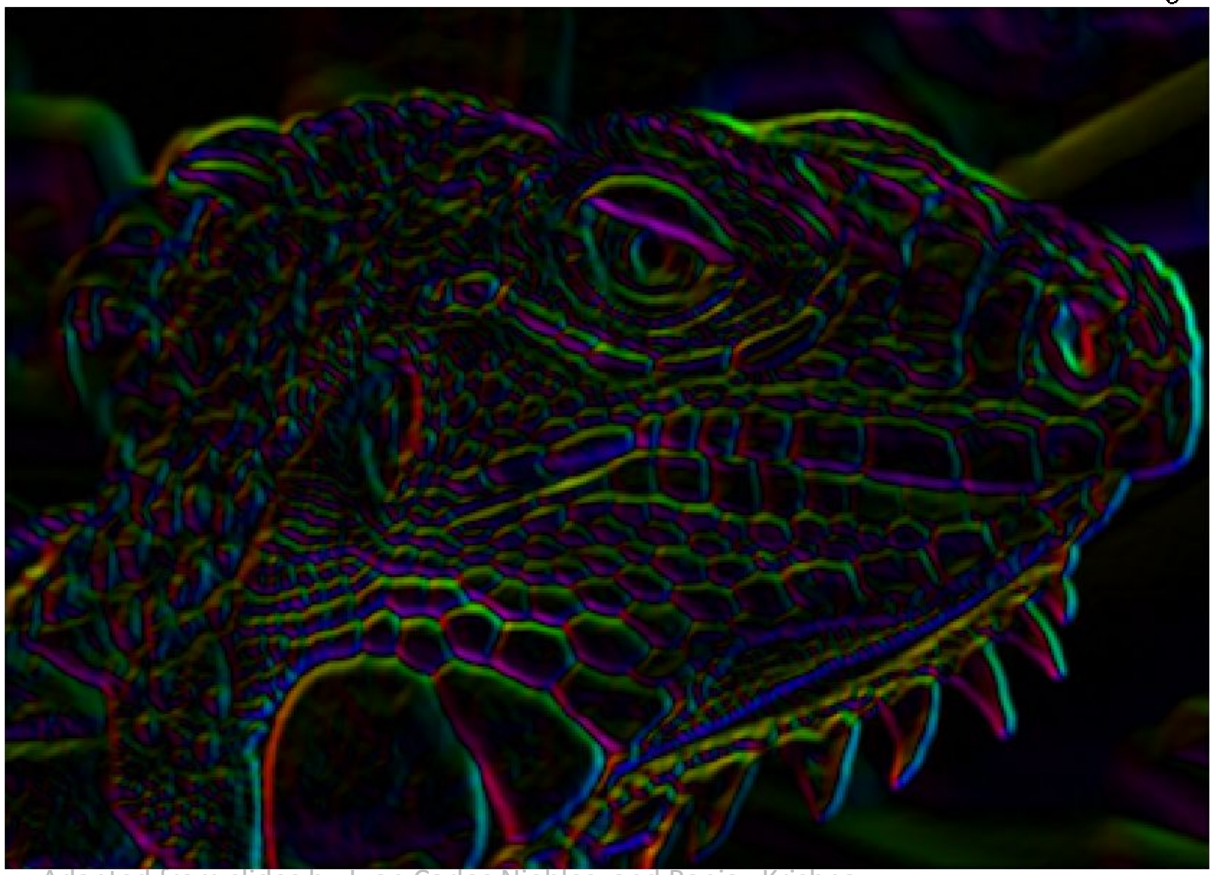
Gradient Magnitude

Source: J. Hayes

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Get orientation at each pixel

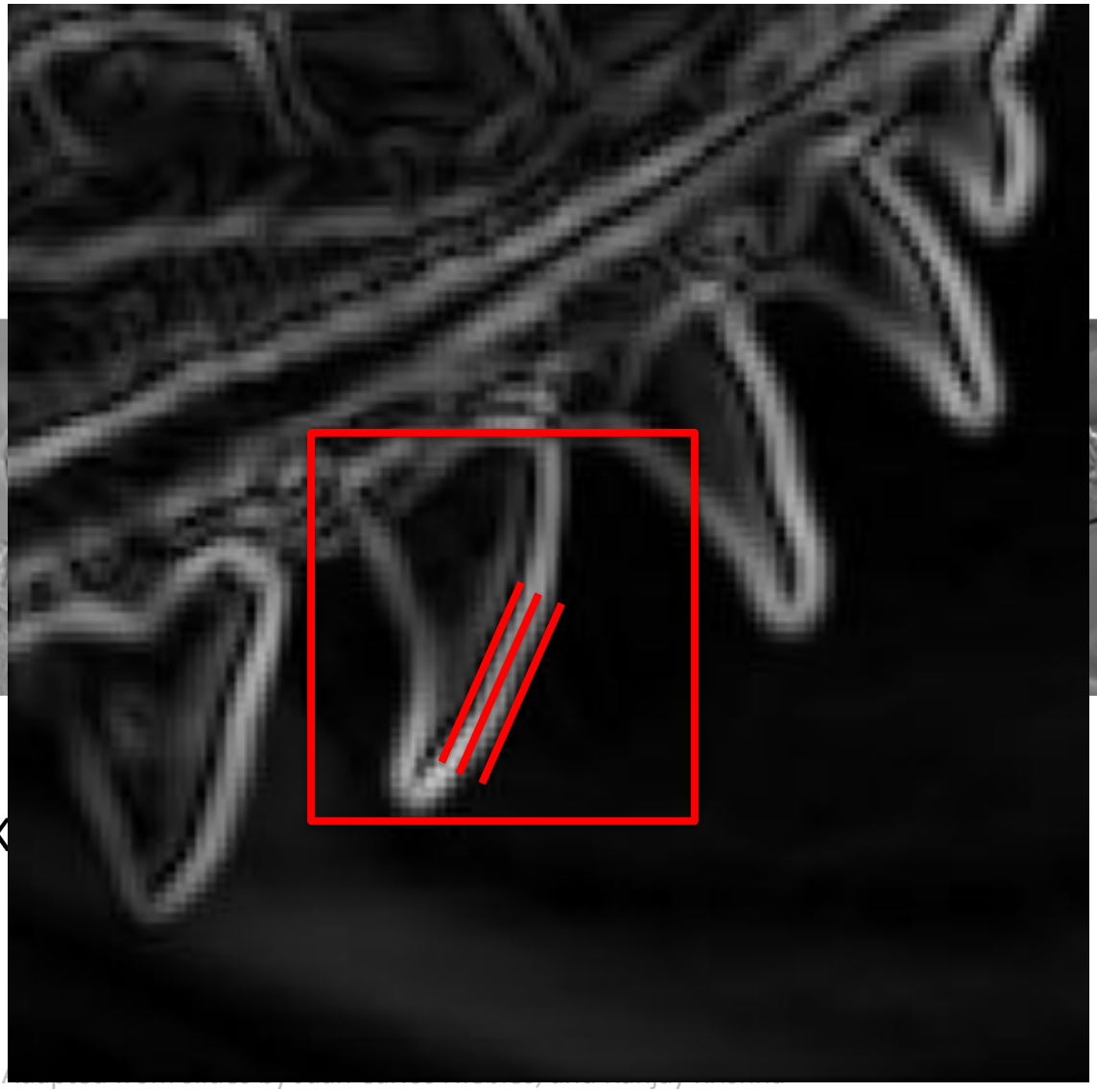
$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$



Source: J. Hayes

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Compute gradients (DoG)



Gradient Magnitude

Canny edge detector

- Suppress Noise
- Compute gradient magnitude and direction
- Apply Non-Maximum Suppression
  - Assures minimal response

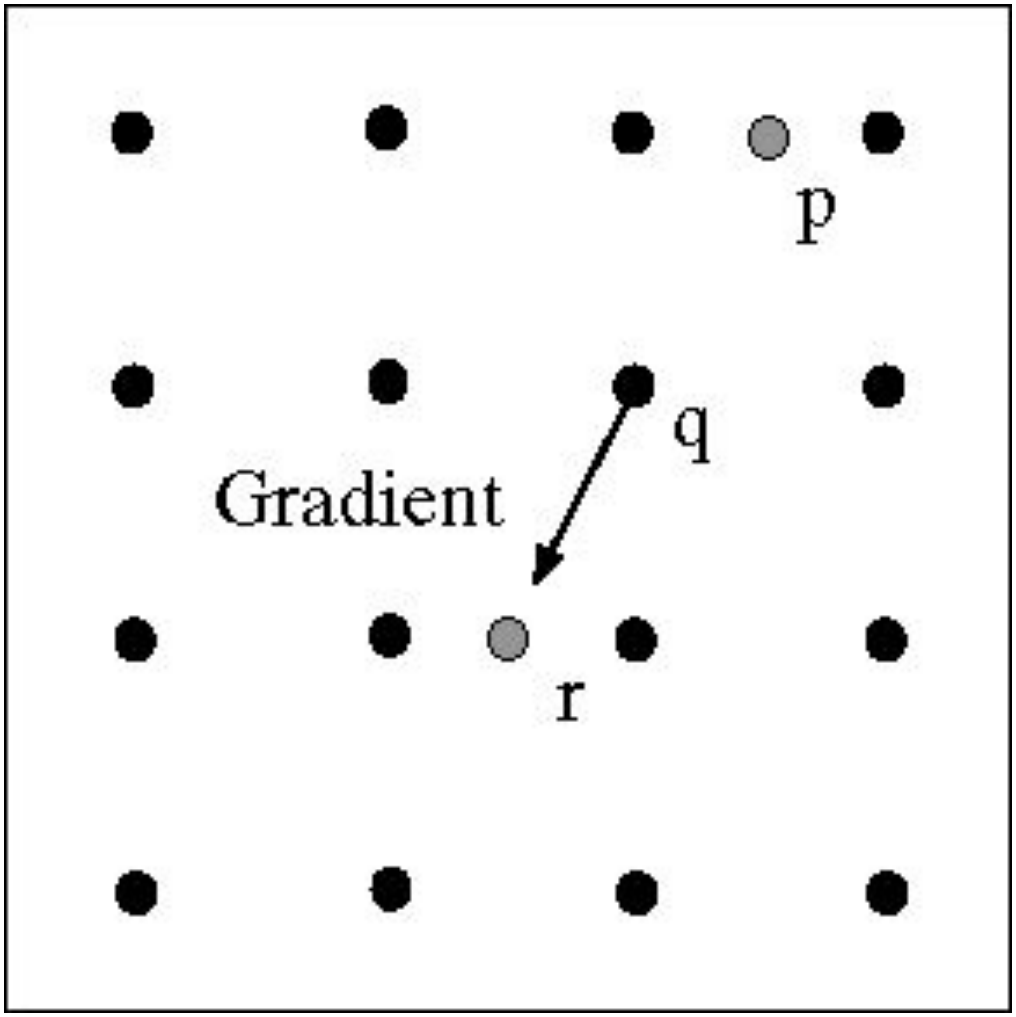
Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Non-maximum suppression

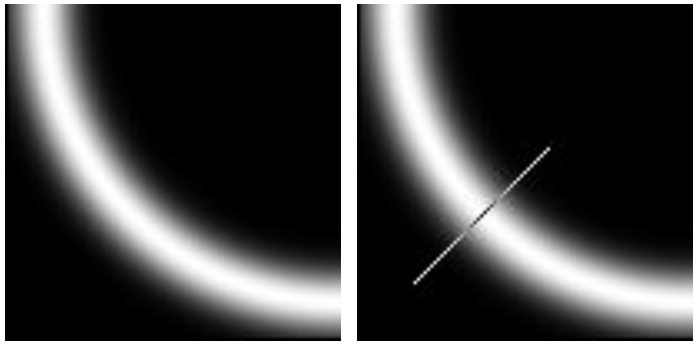
- Edge occurs where gradient reaches a maxima
- Suppress non-maxima gradient even if it passes threshold

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

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.
- Set to zero if not the largest.



Source: D. Forsyth

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Non-max Suppression

Before

After

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Canny edge detector

- Suppress Noise
- Compute gradient magnitude and direction
- Apply Non-Maximum Suppression
  - Assures minimal response
- Use hysteresis and connectivity analysis to detect edges

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Hysteresis thresholding

- Define two thresholds: Low and High
  - If less than Low, not an edge
  - If greater than High, strong edge
  - If between Low and High, weak edge

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

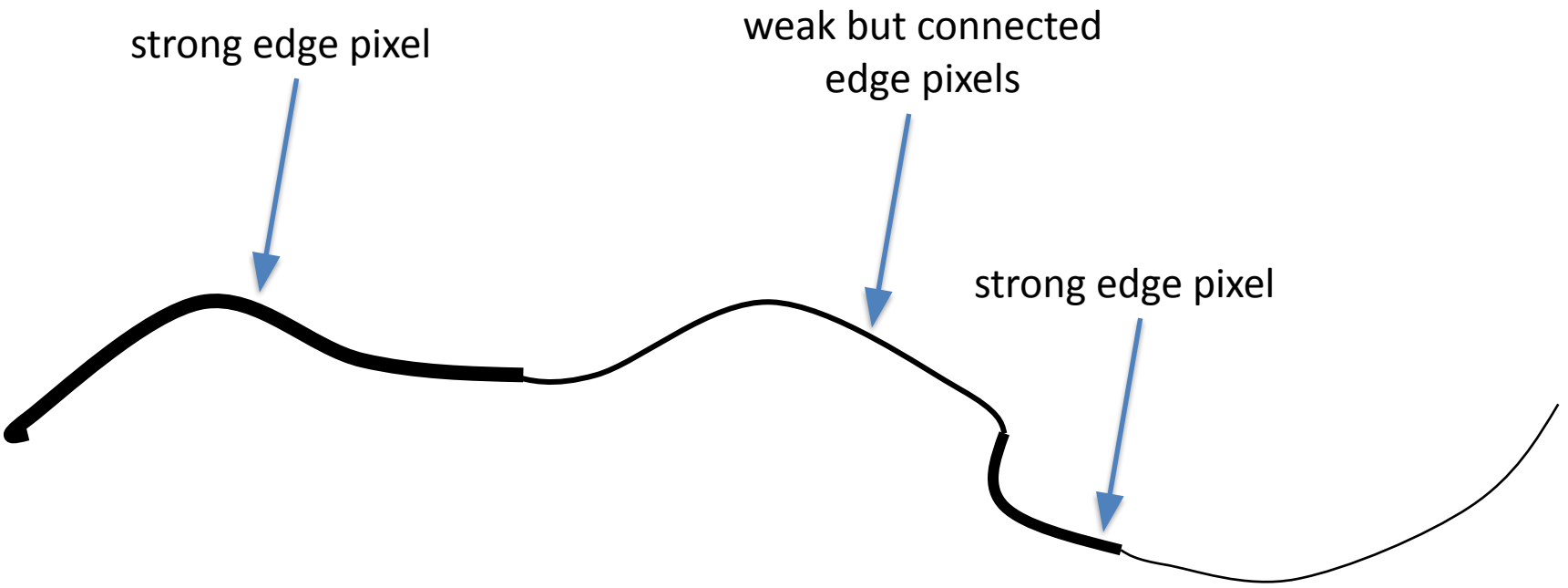
Hysteresis thresholding

If the gradient at a pixel is

- above High, declare it as a ‘strong edge pixel’
- below Low, declare it as a “non-edge-pixel”
- between Low and High
  - Consider its neighbors iteratively then declare it an “edge pixel” if it is connected to an ‘strong edge pixel’ directly or via pixels between Low and High

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Hysteresis thresholding



Source: S. Seitz

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Final Canny Edges

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Canny edge detector summary

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

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

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

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna

Gradients (e.g. Canny)

Color

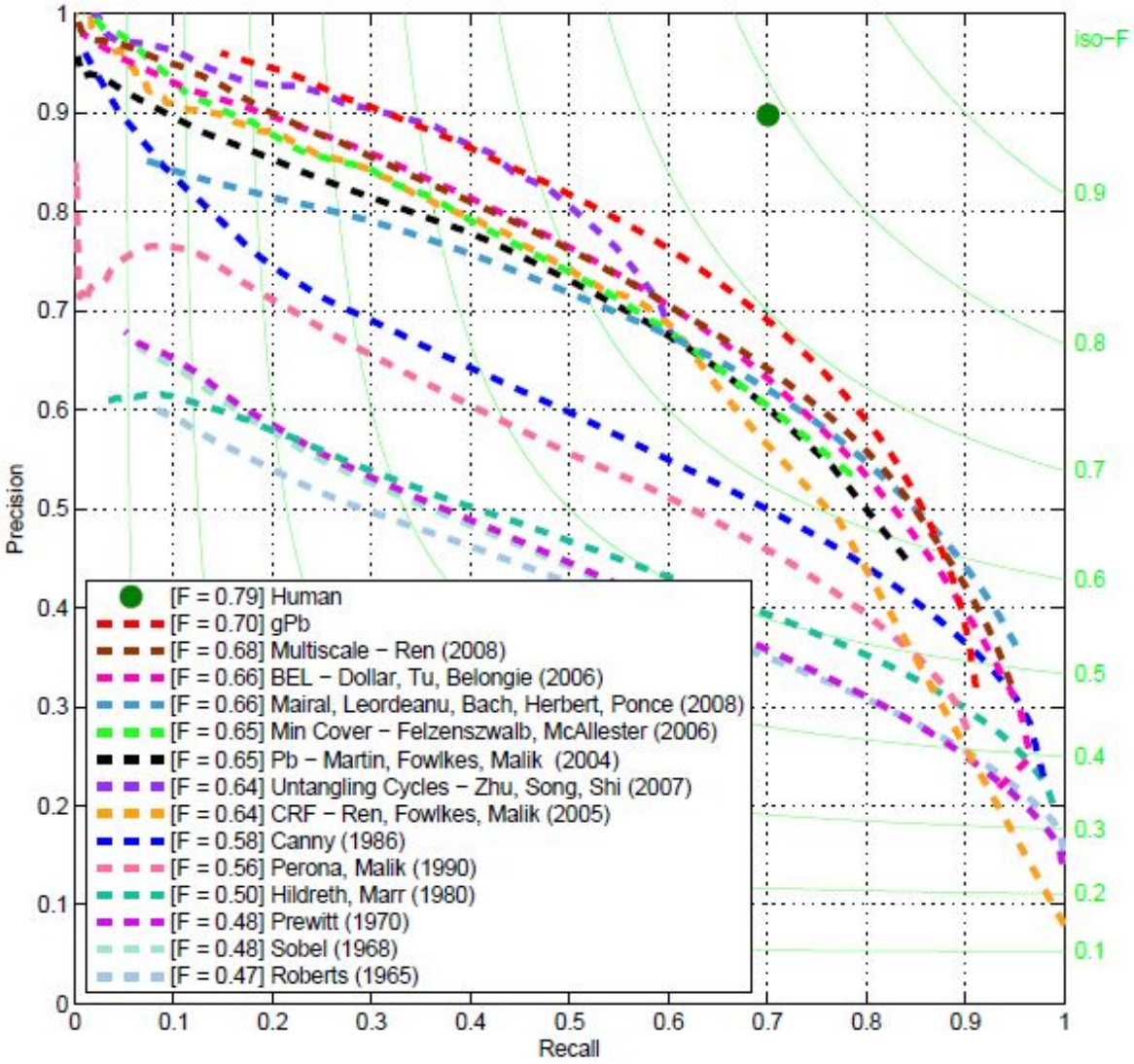
Texture

Combined

Human

Adapted from sli

45 years of boundary detection



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

Adapte

The End

That‘s all about edge detection for now.

Adapted from slides by Juan Carlos Niebles, and Ranjay Krishna