JACOBS UNIVERSITY

COMPUTER VISION
LECTURE 6 – EDGE DETECTION

Prof. Dr. Francesco Maurelli
2018-09-21

WHAT WE WILL LEARN TODAY

JACOBS UNIVERSITY

1. Edge detection
2. Image Gradients
3. A simple edge detector
4. Sobel edge detector
5. Canny edge detector
6. Hough Transform

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

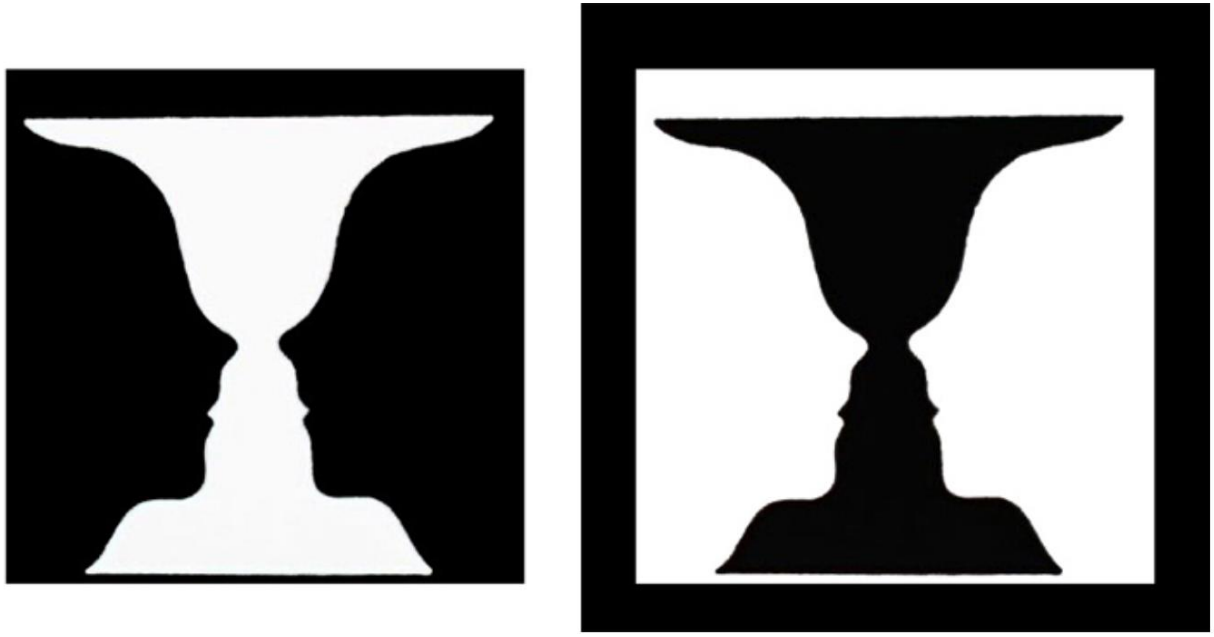
WHAT WE WILL LEARN TODAY

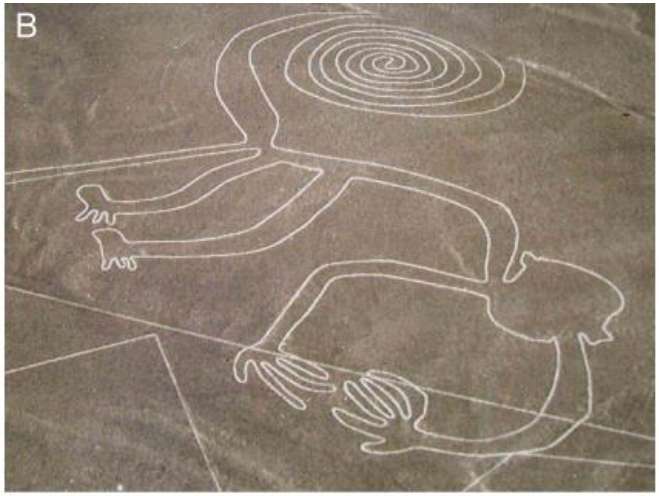
JACOBS UNIVERSITY

1. Edge detection
2. Image Gradients
3. A simple edge detector
4. Sobel edge detector
5. Canny edge detector
6. Hough Transform

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

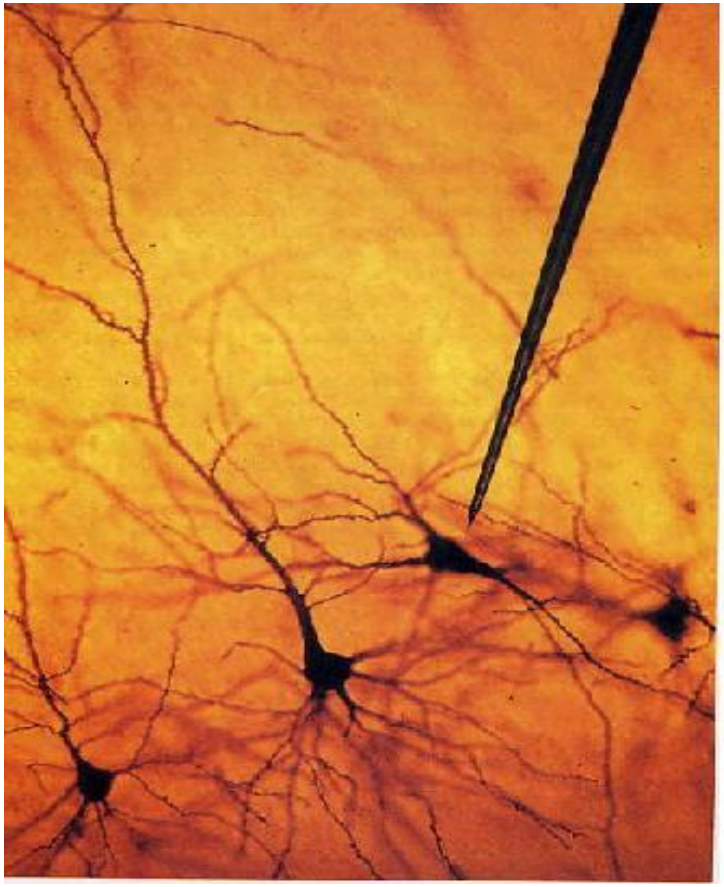
EDGE → MEANING?

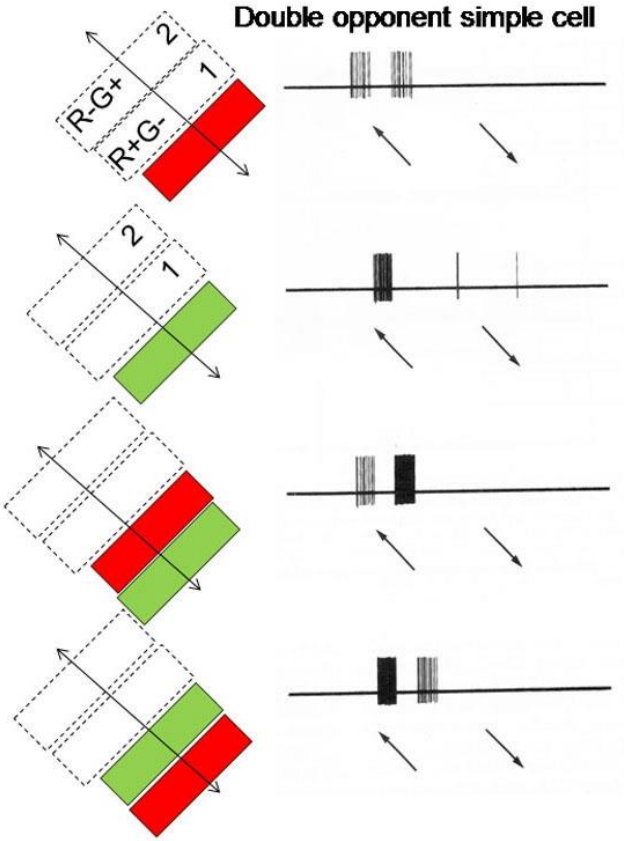




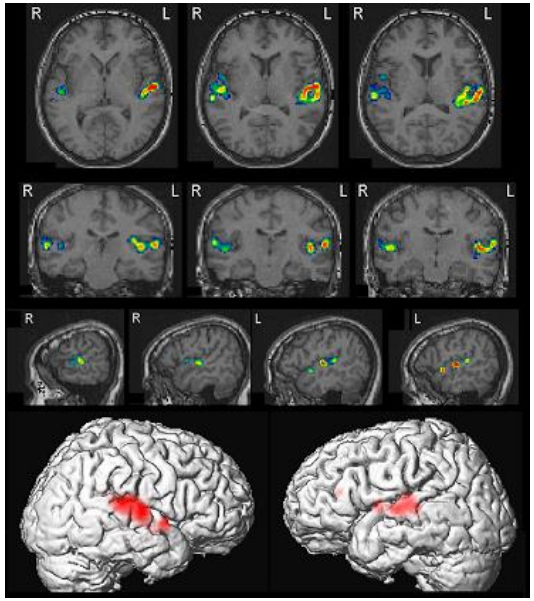
(A) Cave painting at Chauvet, France, about 30,000 B.C.;
(B) Aerial photograph of the picture of a monkey as part of the Nazca Lines geoglyphs, Peru, about 700 – 200 B.C.;
(C) Shen Zhou (1427-1509 A.D.): Poet on a mountain top, ink on paper, China;
(D) Line drawing by 7-year old I. Lleras (2010 A.D.).

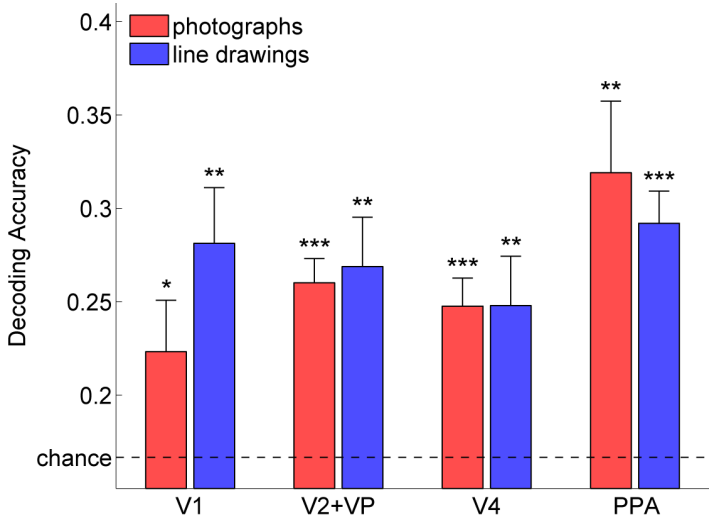
WE KNOW EDGES ARE SPECIAL FROM HUMAN (MAMMALIAN) VISION STUDIES





Hubel & Wiesel, 1960s





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

EDGE DETECTION

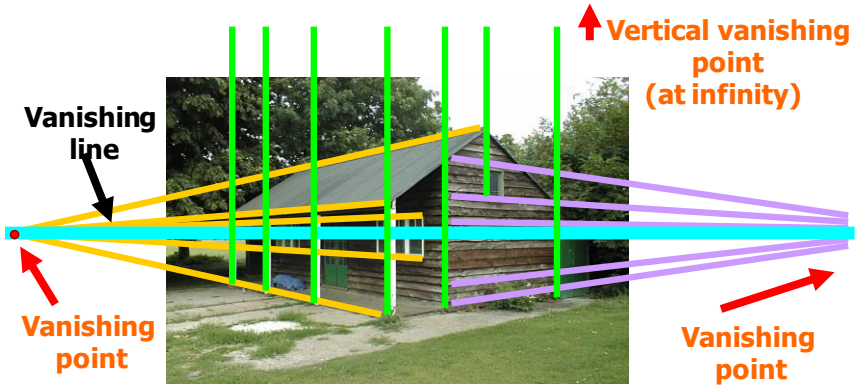
- **Goal:** Identify sudden changes (discontinuities) in an image
  - Intuitively, most semantic and shape information from the image can be encoded in the edges
  - More compact than pixels

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

WHY DO WE CARE ABOUT EDGES?

1. Extract information, recognize objects

1. Recover geometry and viewpoint



ORIGINS OF EDGES

surface normal discontinuity

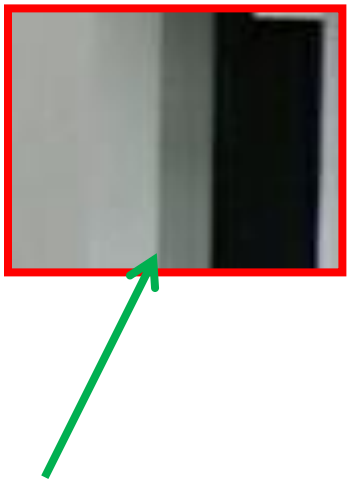
depth discontinuity

surface color discontinuity

illumination discontinuity

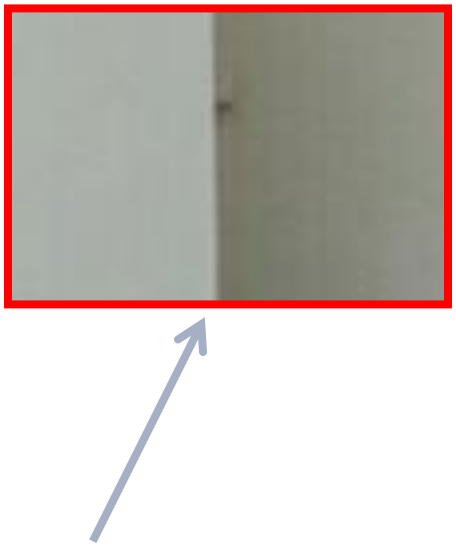
CLOSEUP OF EDGES

Surface normal discontinuity



CLOSEUP OF EDGES

Depth discontinuity



CLOSEUP OF EDGES

Surface color discontinuity

WHAT WE WILL LEARN TODAY

JACOBS UNIVERSITY

1. Edge detection
2. Image Gradients
3. A simple edge detector
4. Sobel edge detector
5. Canny edge detector
6. Hough Transform

DERIVATIVES IN 1D

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

DERIVATIVES IN 1D - EXAMPLE

JACOBS UNIVERSITY

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

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

DERIVATIVES IN 1D - EXAMPLE

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

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

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

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

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

JACOBS UNIVERSITY

Backward

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

Forward

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

Central

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

JACOBS UNIVERSITY

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]

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

DISCRETE DERIVATE IN 2D

JACOBS UNIVERSITY

Given function $f(x,y)$

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

Gradient magnitude $$\left|\nabla f(x,y)\right| = \sqrt{f_x^2 + f_y^2}$$

Gradient direction $$\theta = \tan^{-1} \frac{f_x}{f_y}$$

DISCRETE DERIVATE IN 2D

JACOBS UNIVERSITY

Given function $f(x,y)$

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

Gradient magnitude $$\left|\nabla f(x,y)\right| = \sqrt{f_x^2 + f_y^2}$$

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

JACOBS UNIVERSITY

What does this filter do?

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

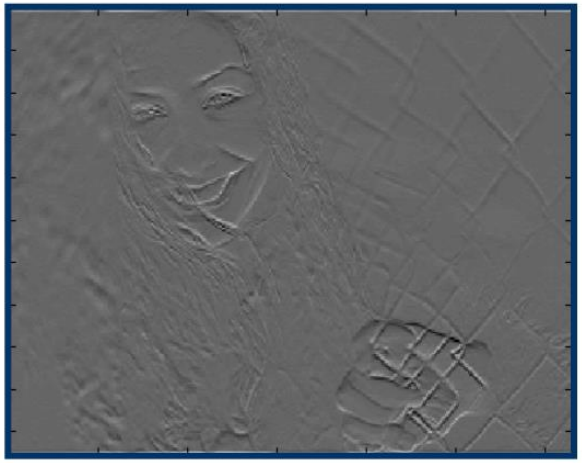
JACOBS UNIVERSITY

What about this filter?

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

3X3 IMAGE GRADIENT FILTERS

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



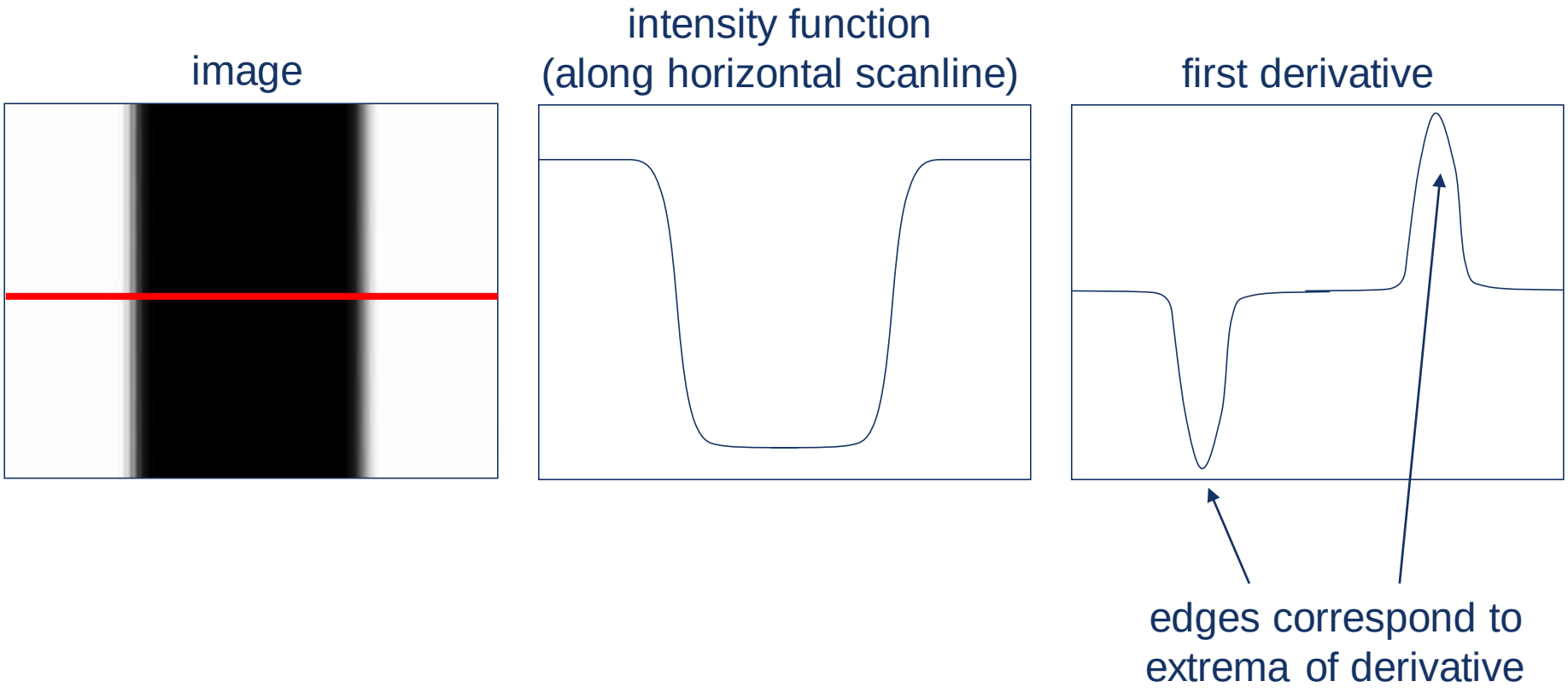
WHAT WE WILL LEARN TODAY

JACOBS UNIVERSITY

1. Edge detection
2. Image Gradients
3. A simple edge detector
4. Sobel edge detector
5. Canny edge detector
6. Hough Transform

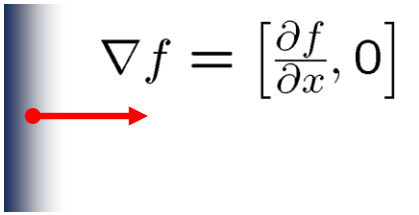
CHARACTERIZING EDGES

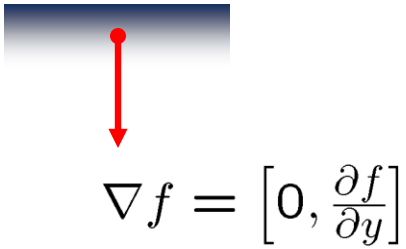
- An edge is a place of rapid change in the image intensity function

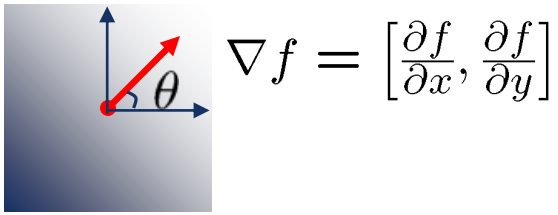


$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

1.The gradient of an image:







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

The gradient direction is given by $\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$

The *edge strength* is given by the gradient magnitude

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

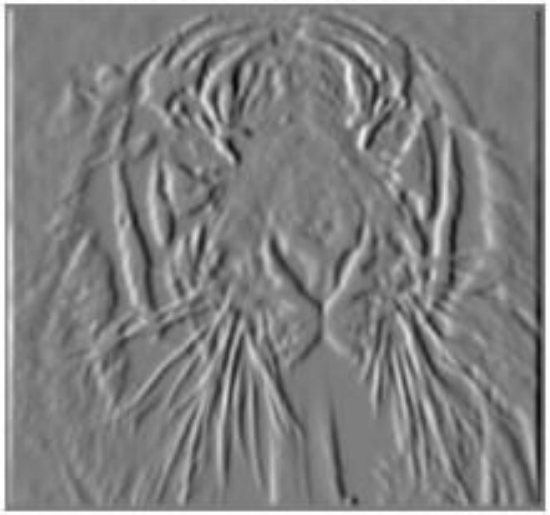
FINITE DIFFERENCES: EXAMPLE

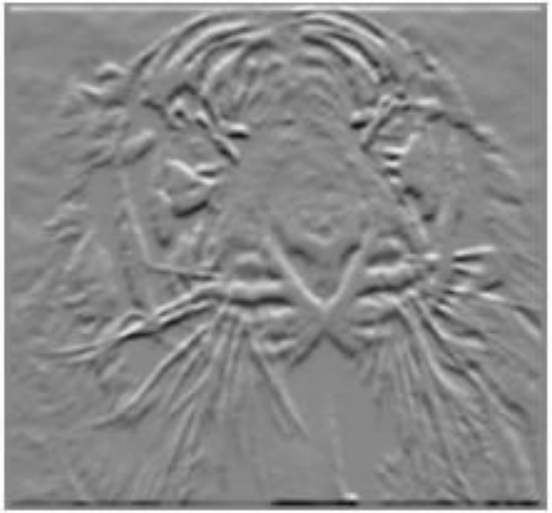
JACOBS UNIVERSITY

Original Image

Gradient magnitude

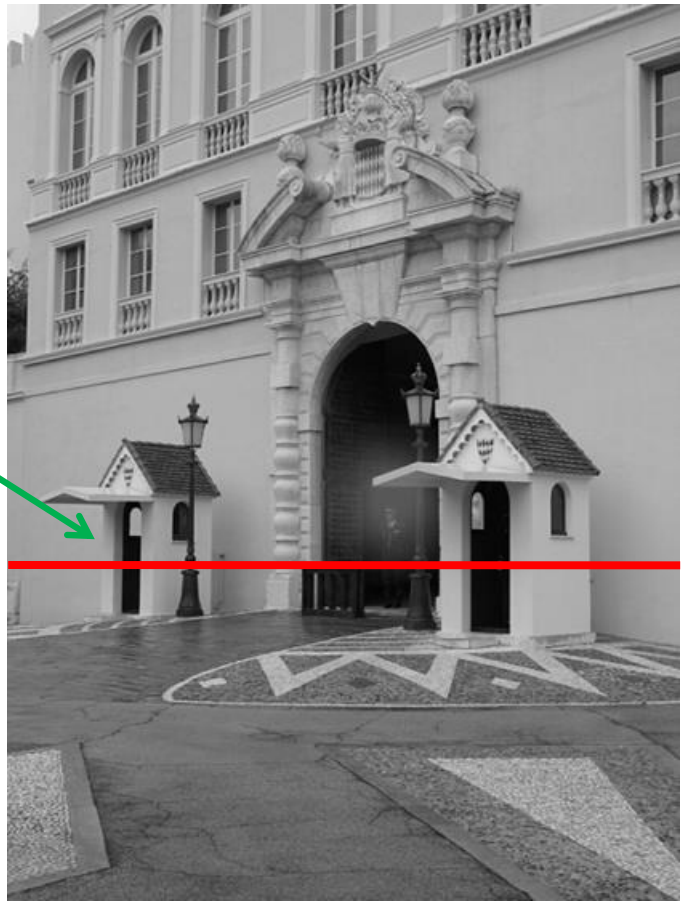
x-direction

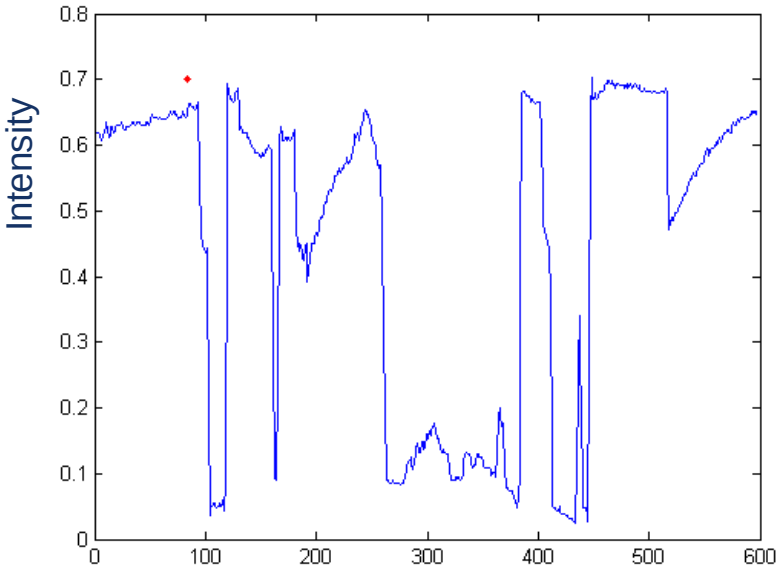


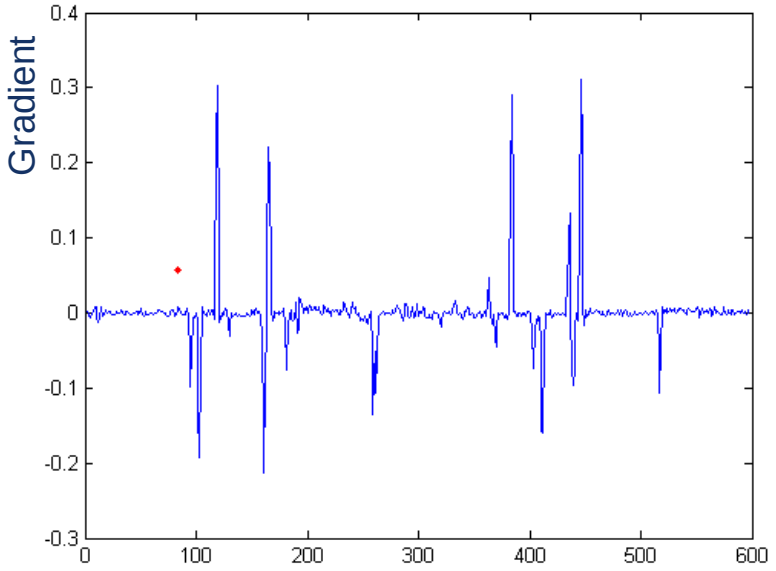


y-direction

INTENSITY PROFILE



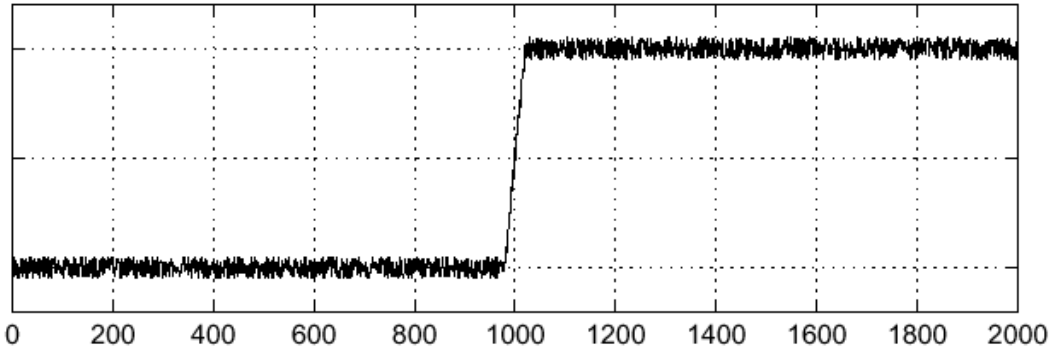




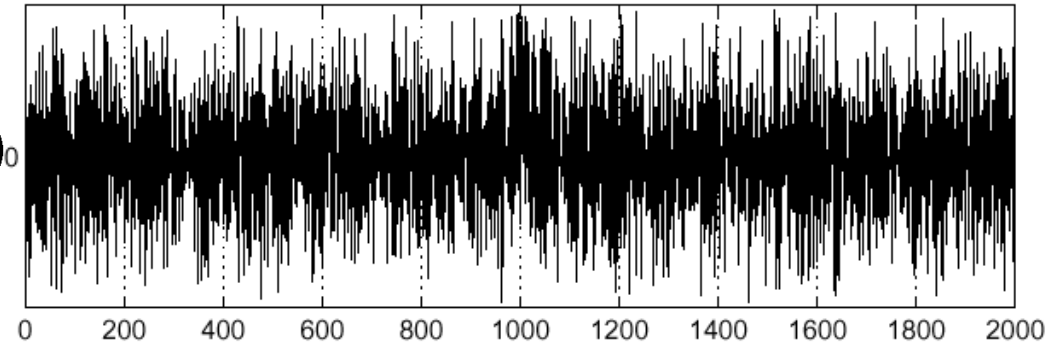
Consider a single row or column of the image

- Plotting intensity as a function of position gives a signal

$f(x)$

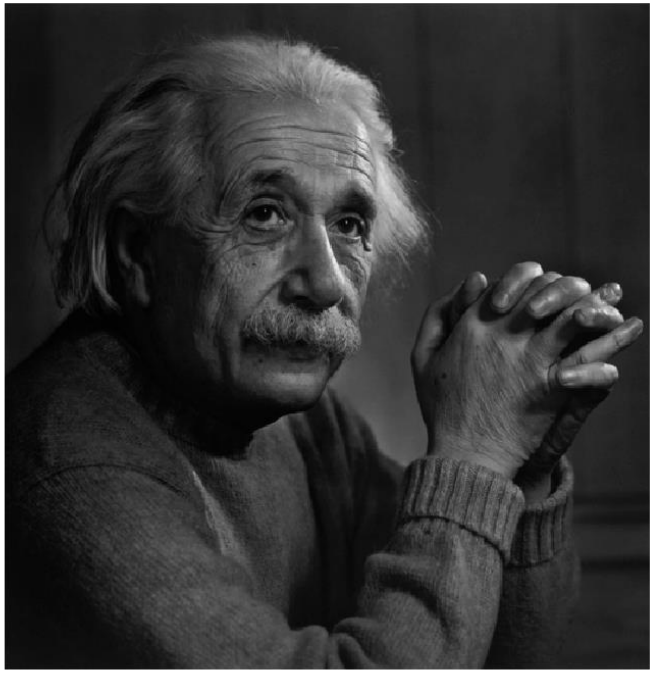


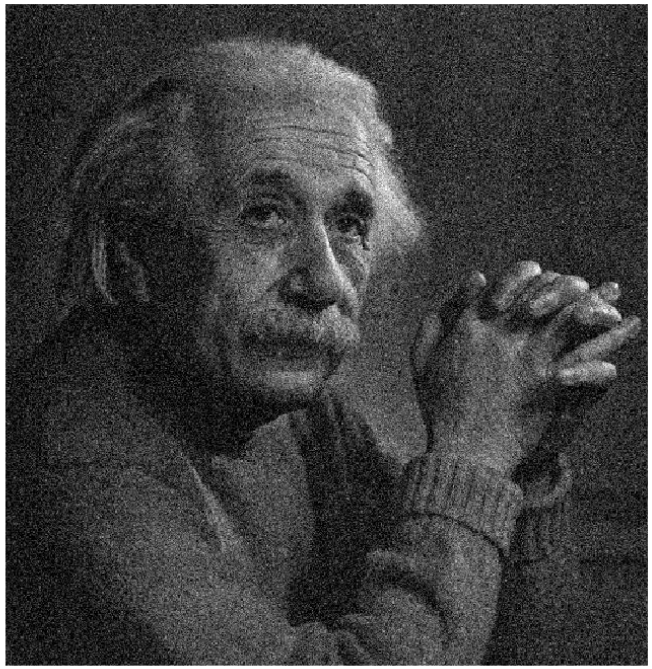
$\frac{d}{dx}f(x)$



Where is the edge?

EFFECTS OF NOISE





JACOBS UNIVERSITY

- 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

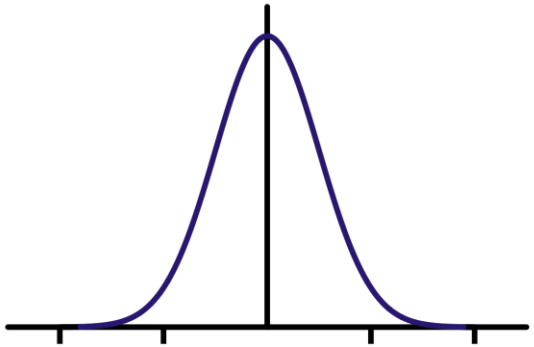
- What is to be done?
  - Smoothing the image should help, by forcing pixels different to their neighbors (=noise pixels?) to look more like neighbors

SMOOTHING WITH DIFFERENT FILTERS

1. Mean smoothing

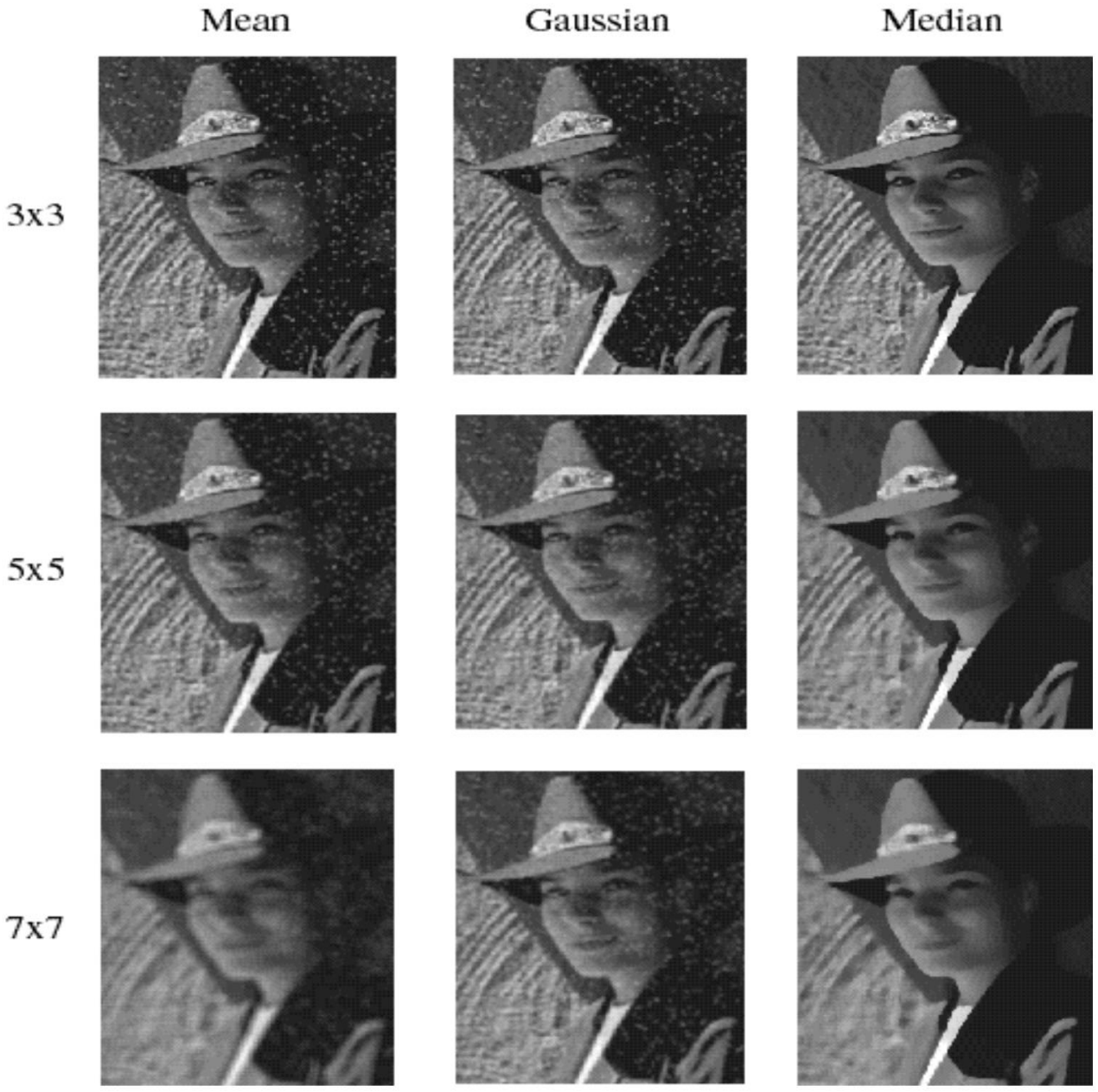
$$\begin{bmatrix}1\\1\\1\end{bmatrix} \quad \begin{bmatrix}1 & 1 & 1\end{bmatrix}$$

2. Gaussian (smoothing * derivative)

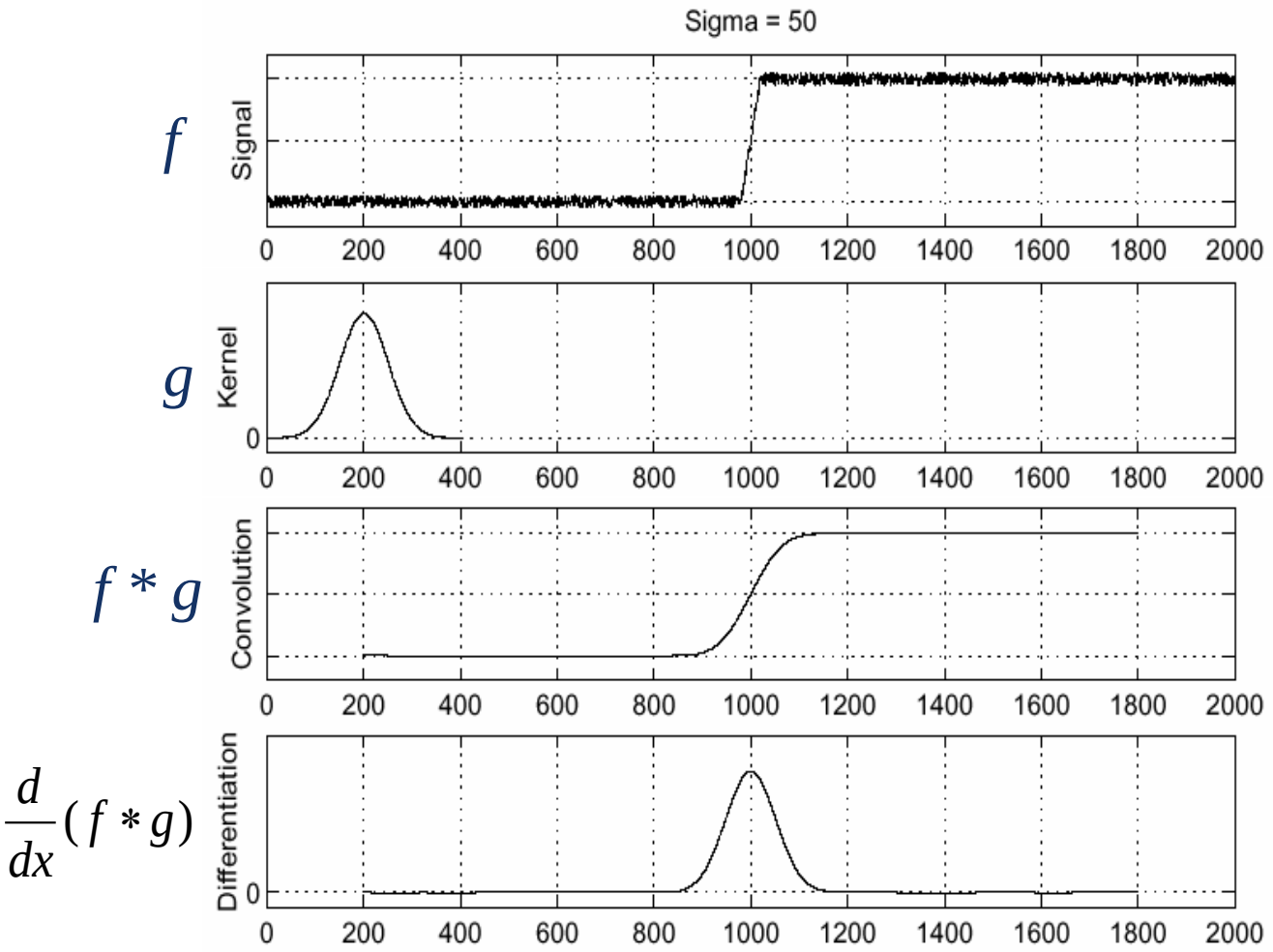


$$\begin{bmatrix}1\\2\\1\end{bmatrix} \quad \begin{bmatrix}1 & 2 & 1\end{bmatrix}$$

SMOOTHING WITH DIFFERENT FILTERS



SOLUTION: SMOOTH FIRST



f

g

$f * g$

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

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

DERIVATIVE THEOREM OF CONVOLUTION

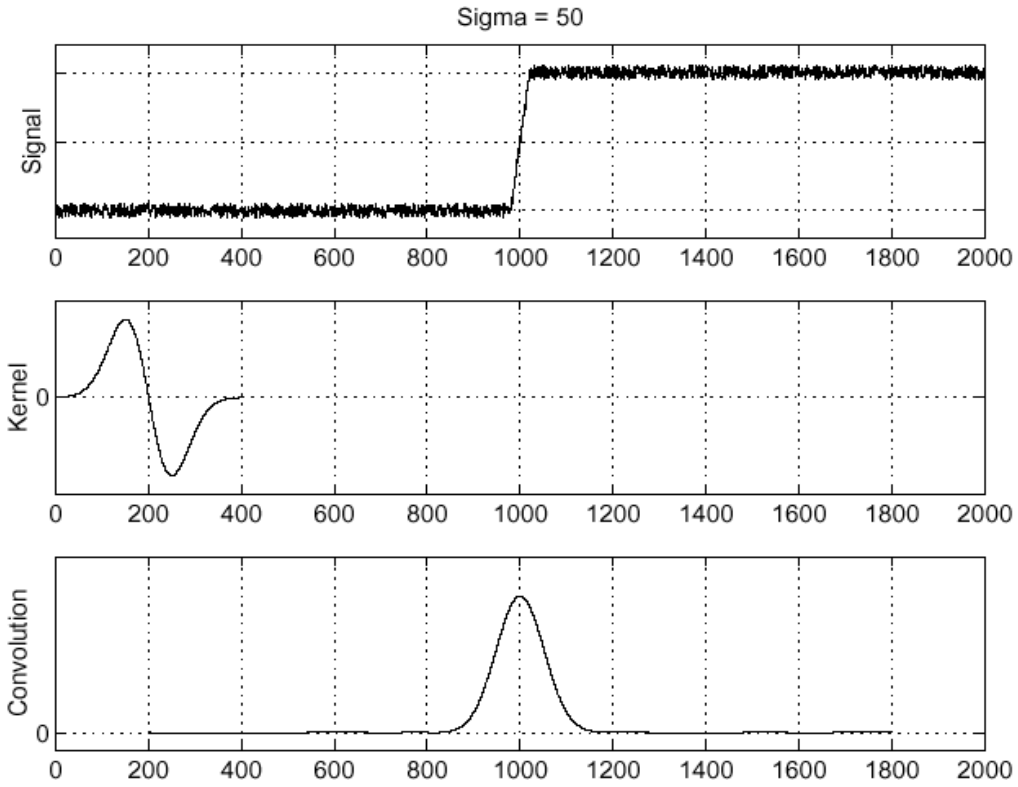
- This theorem gives us a very useful property: $\frac{d}{dx}(f * g) = f * \frac{d}{dx} g$

- This saves us one operation:

f

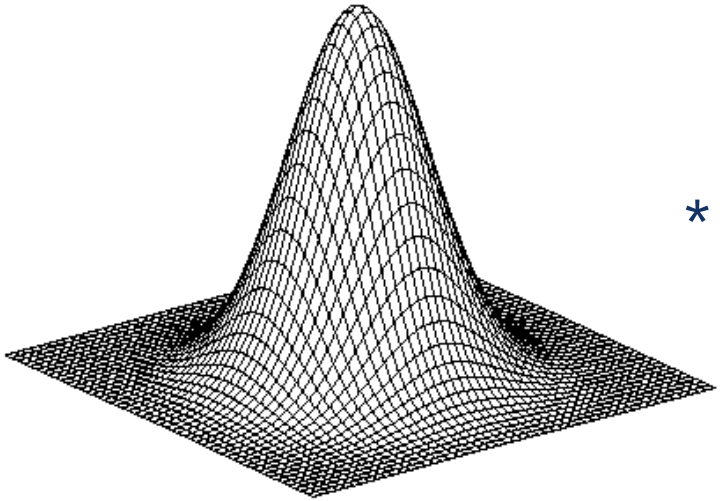
$\frac{d}{dx} g$

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

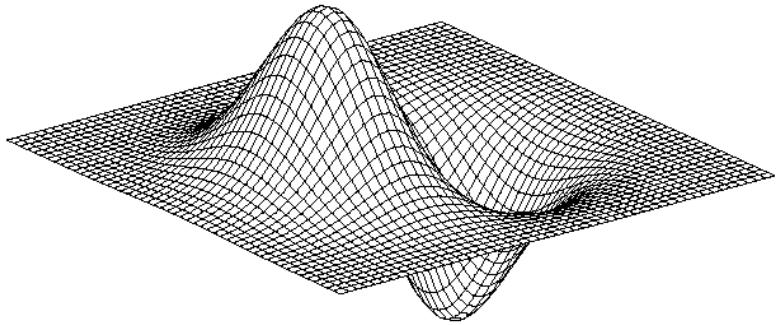


Source: S. Seitz

DERIVATIVE OF GAUSSIAN FILTER



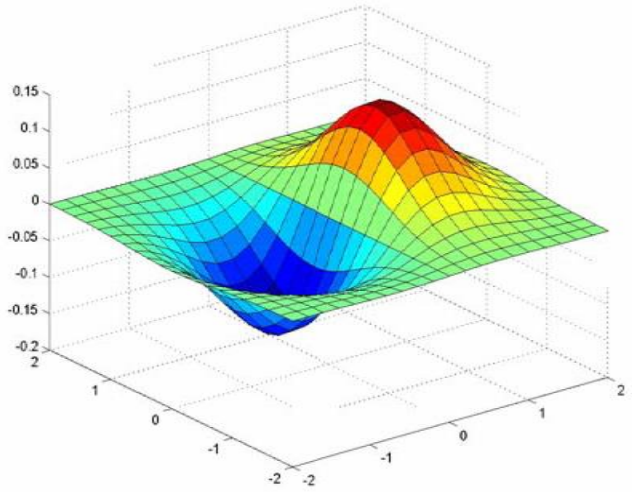
* [1 0 -1] =



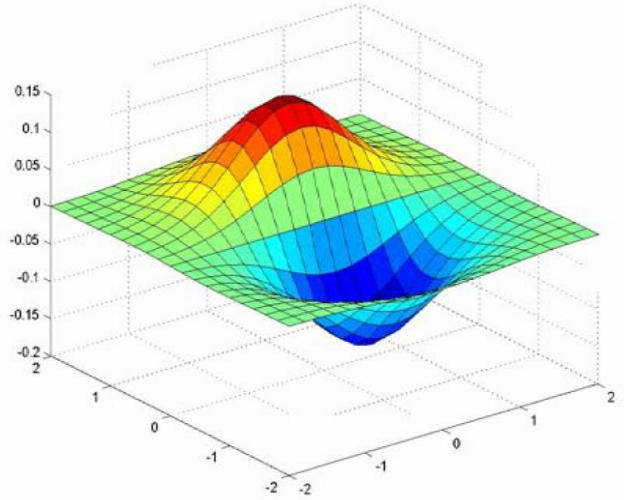
2D-Gaussian

x - derivative

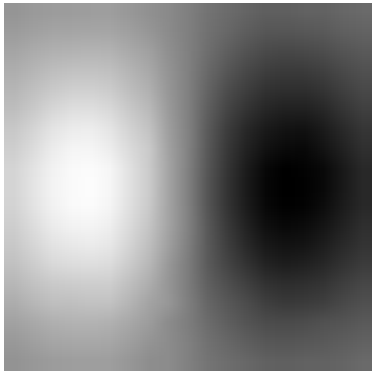
DERIVATIVE OF GAUSSIAN FILTER

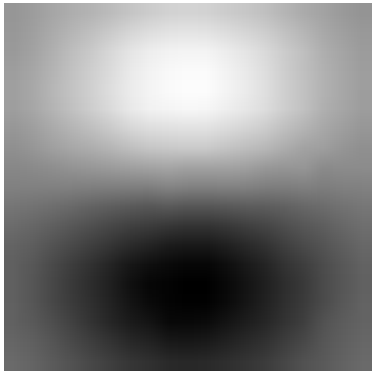


x-direction



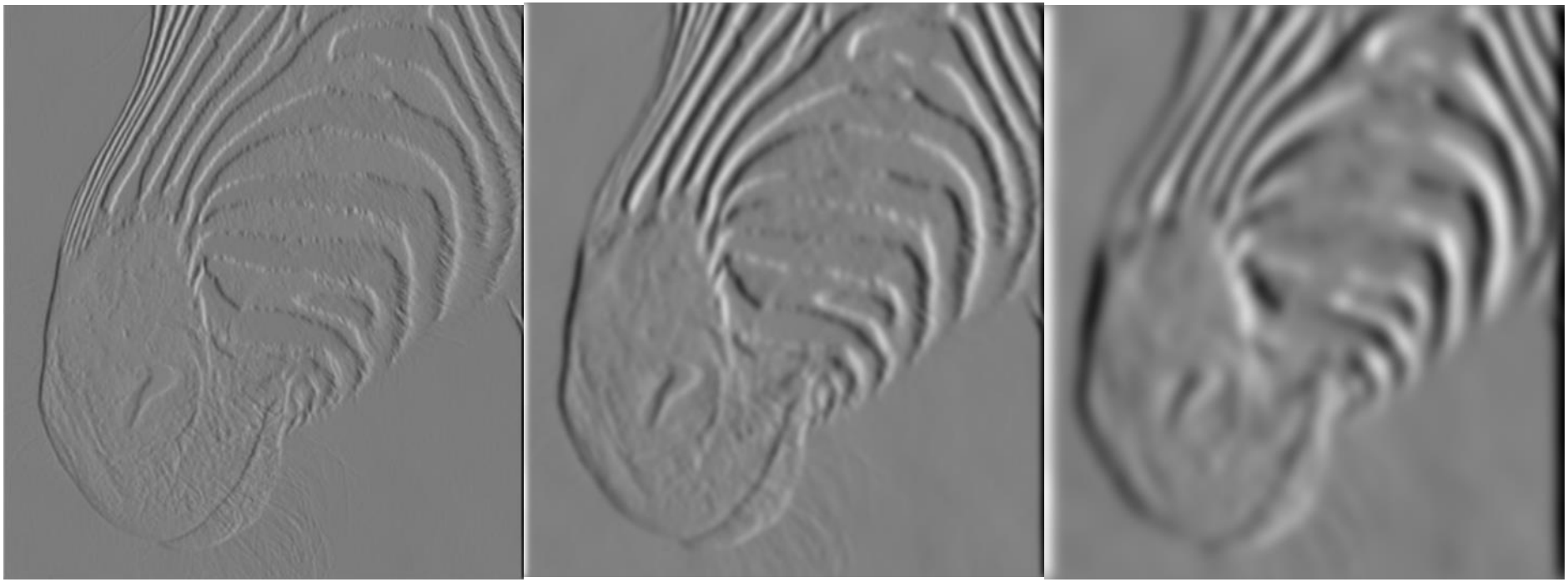
y-direction





DERIVATIVE OF GAUSSIAN FILTER

TRADEOFF BETWEEN SMOOTHING AT DIFFERENT SCALES



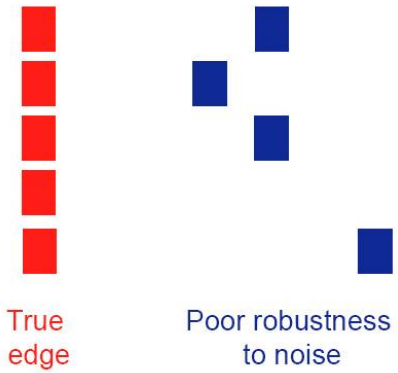
1 pixel 3 pixels 7 pixels

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

JACOBS UNIVERSITY

Criteria for an "optimal" edge detector:

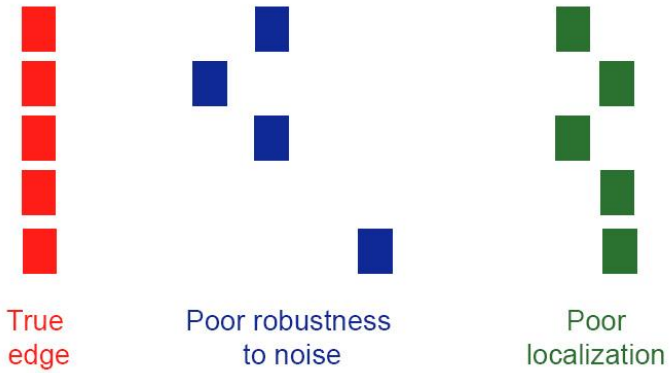
- **Good detection:** the optimal detector must minimize the probability of false positives (detecting spurious edges caused by noise), as well as that of false negatives (missing real edges)



JACOBS UNIVERSITY

Criteria for an "optimal" edge detector:

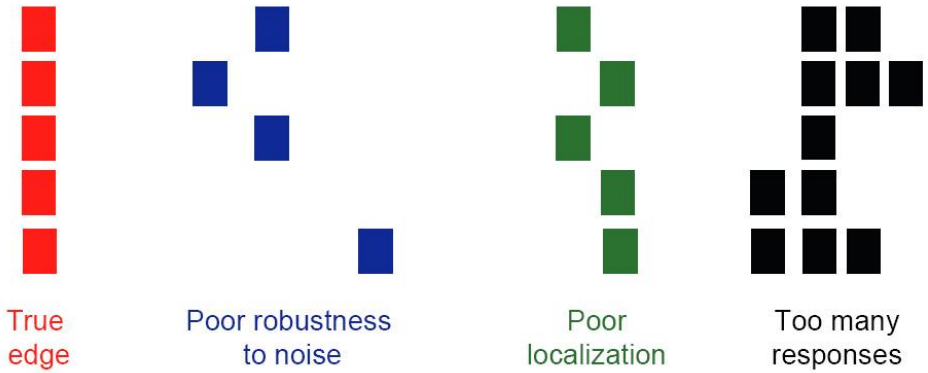
- **Good detection:** the optimal detector must minimize the probability of false positives (detecting spurious edges caused by noise), as well as that of false negatives (missing real edges)
- **Good localization:** the edges detected must be as close as possible to the true edges



JACOBS UNIVERSITY

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



WHAT WE WILL LEARN TODAY

JACOBS UNIVERSITY

1. Edge detection
2. Image Gradients
3. A simple edge detector
4. Sobel Edge detector
5. Canny edge detector
6. Hough transform

SOBEL OPERATOR

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

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

JACOBS UNIVERSITY

1. 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

SOBEL OPERATION

JACOBS UNIVERSITY

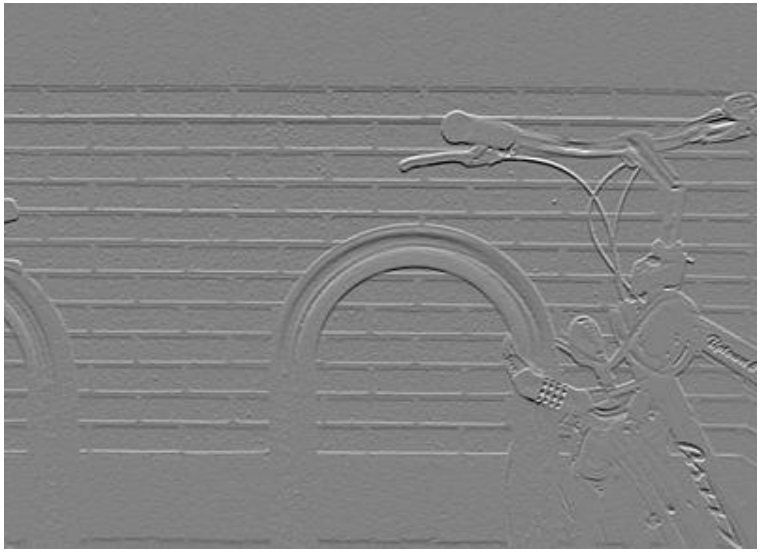
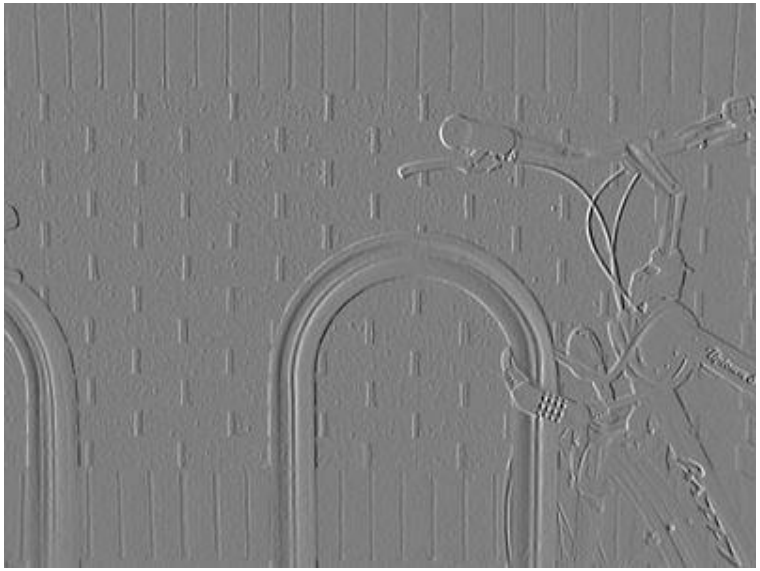
1. Magnitude:

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

2. Angle or direction of the gradient:

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

SOBEL FILTER EXAMPLE


WHAT WE WILL LEARN TODAY

JACOBS UNIVERSITY

1. Edge detection
2. Image Gradients
3. A simple edge detector
4. Sobel Edge detector
5. Canny edge detector
6. Hough Transform

CANNY EDGE DETECTOR

JACOBS UNIVERSITY

- 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.

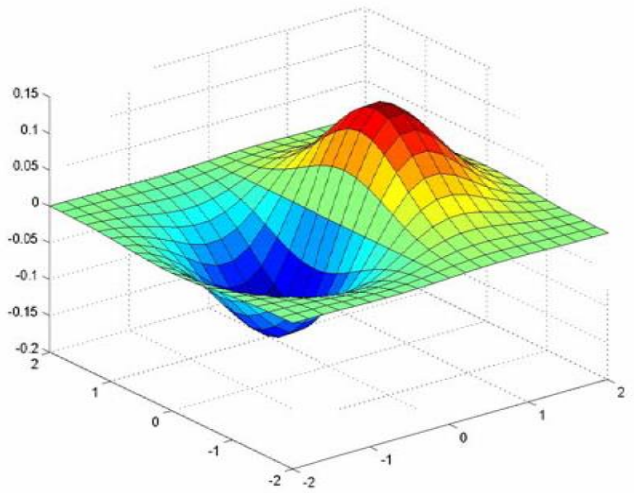
CANNY EDGE DETECTOR

JACOBS UNIVERSITY

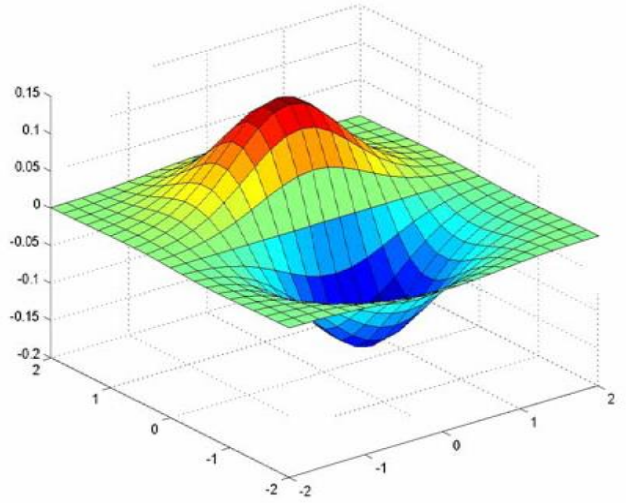
1. Suppress Noise
2. Compute gradient magnitude and direction
3. Apply Non-Maximum Suppression
   - Assures minimal response
4. Use hysteresis and connectivity analysis to detect edges

EXAMPLE

DERIVATIVE OF GAUSSIAN FILTER

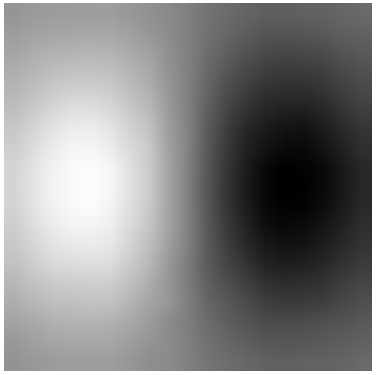


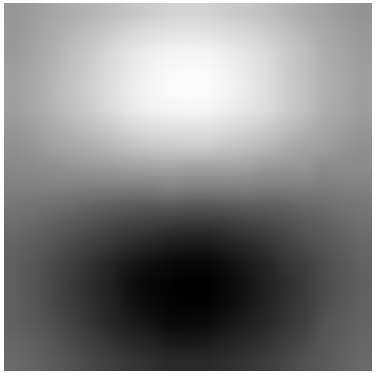
x-direction



y-direction

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

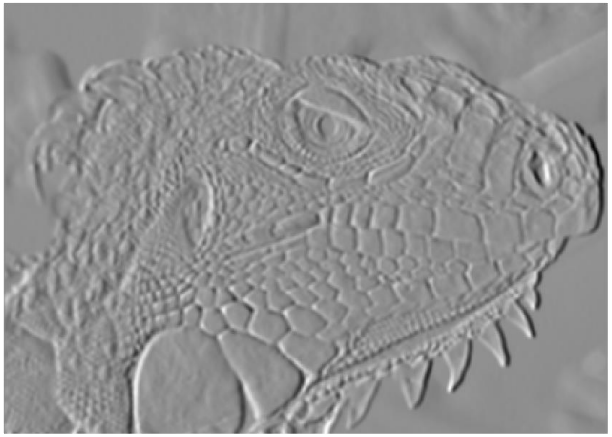




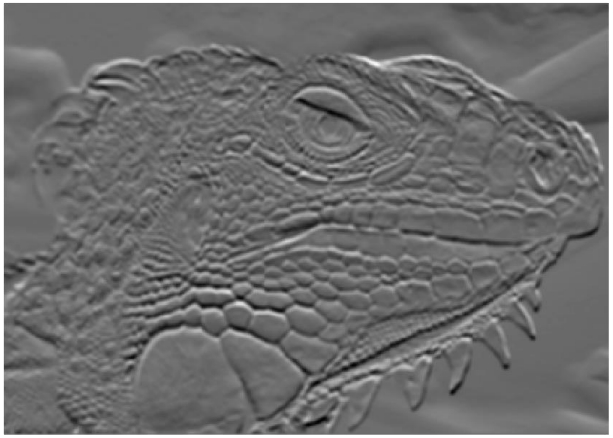
$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

COMPUTE GRADIENTS

JACOBS UNIVERSITY



X-Derivative of Gaussian



Y-Derivative of Gaussian

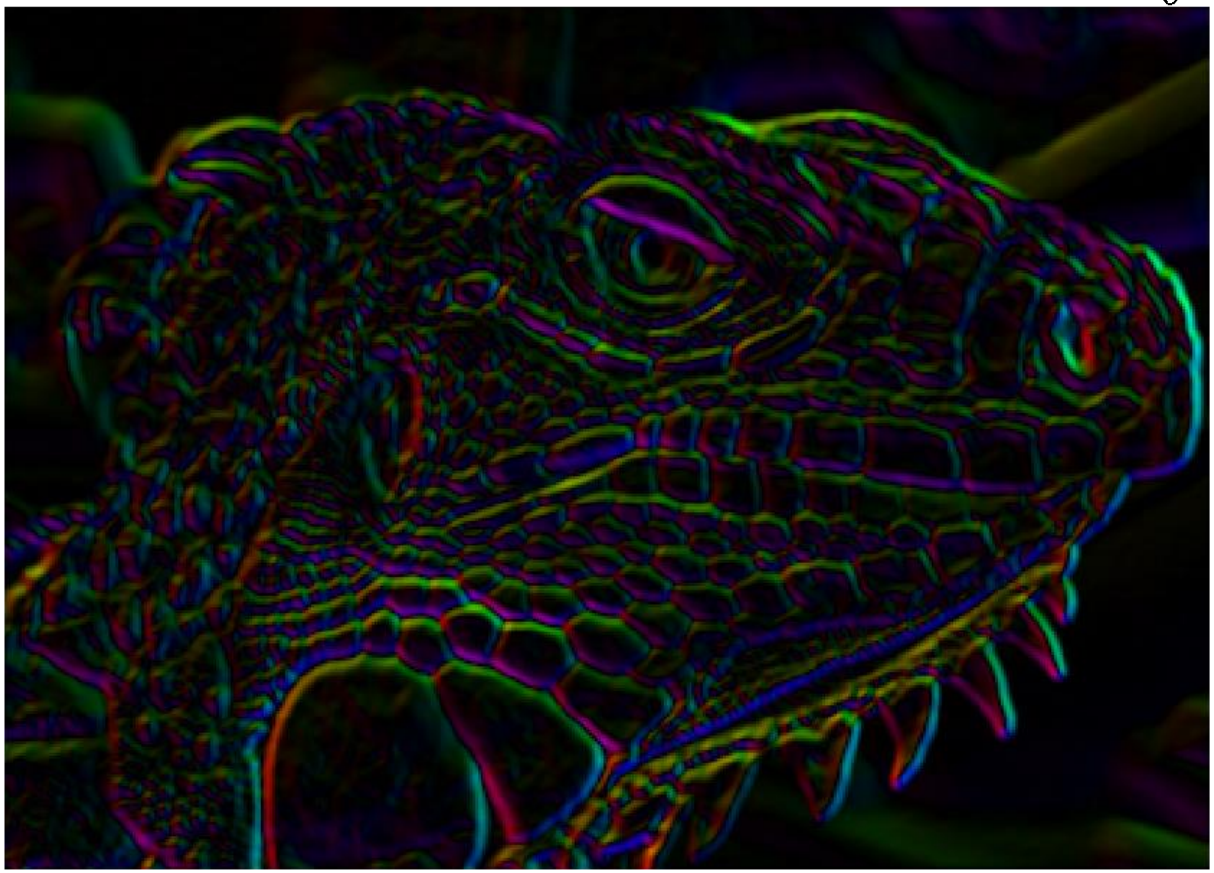
Gradient Magnitude

Source: J. Hayes

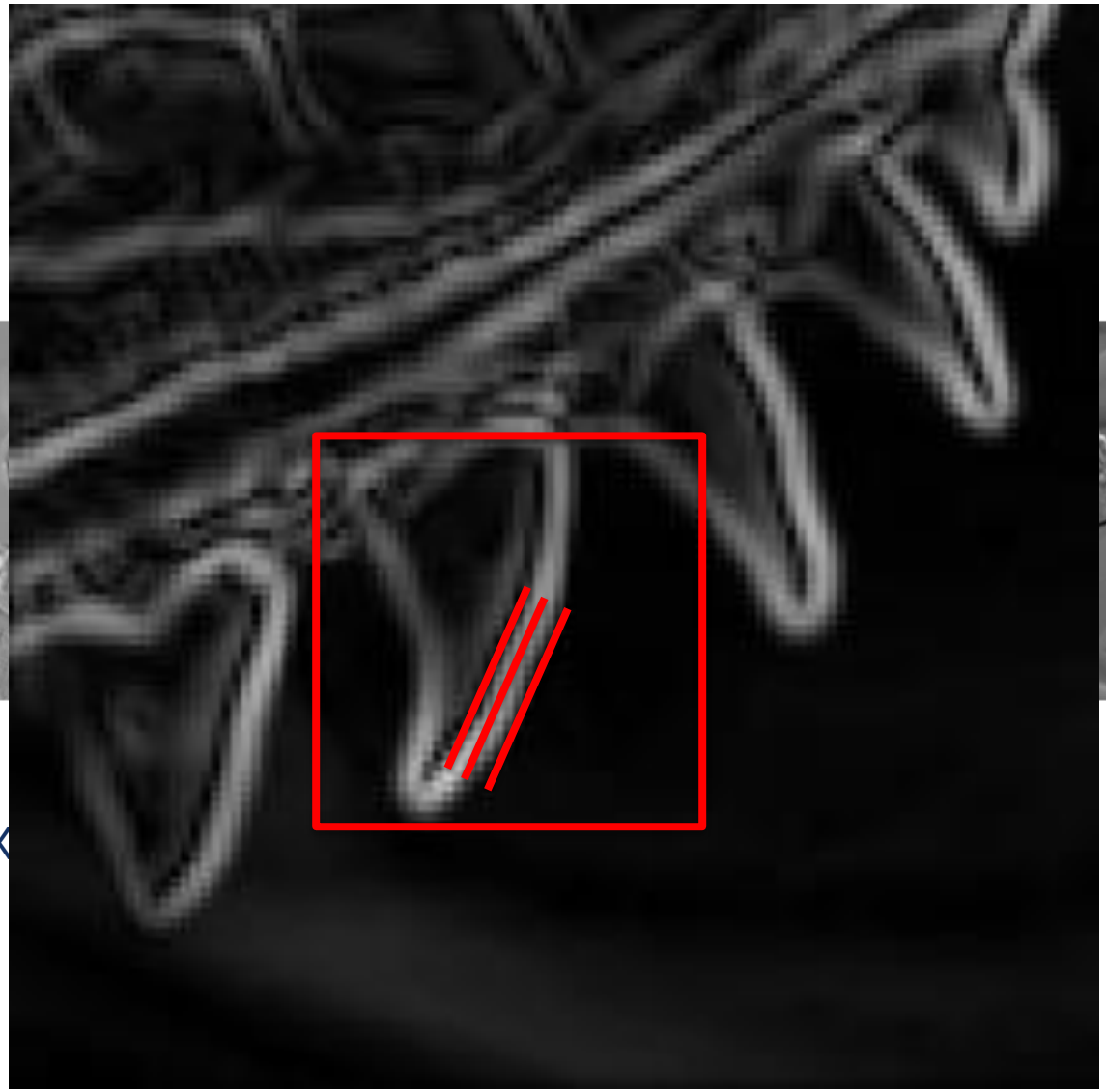
GET ORIENTATION AT EACH PIXEL

JACOBS UNIVERSITY

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



COMPUTE GRADIENTS



X

Gradient Magnitude

CANNY EDGE DETECTOR

JACOBS UNIVERSITY

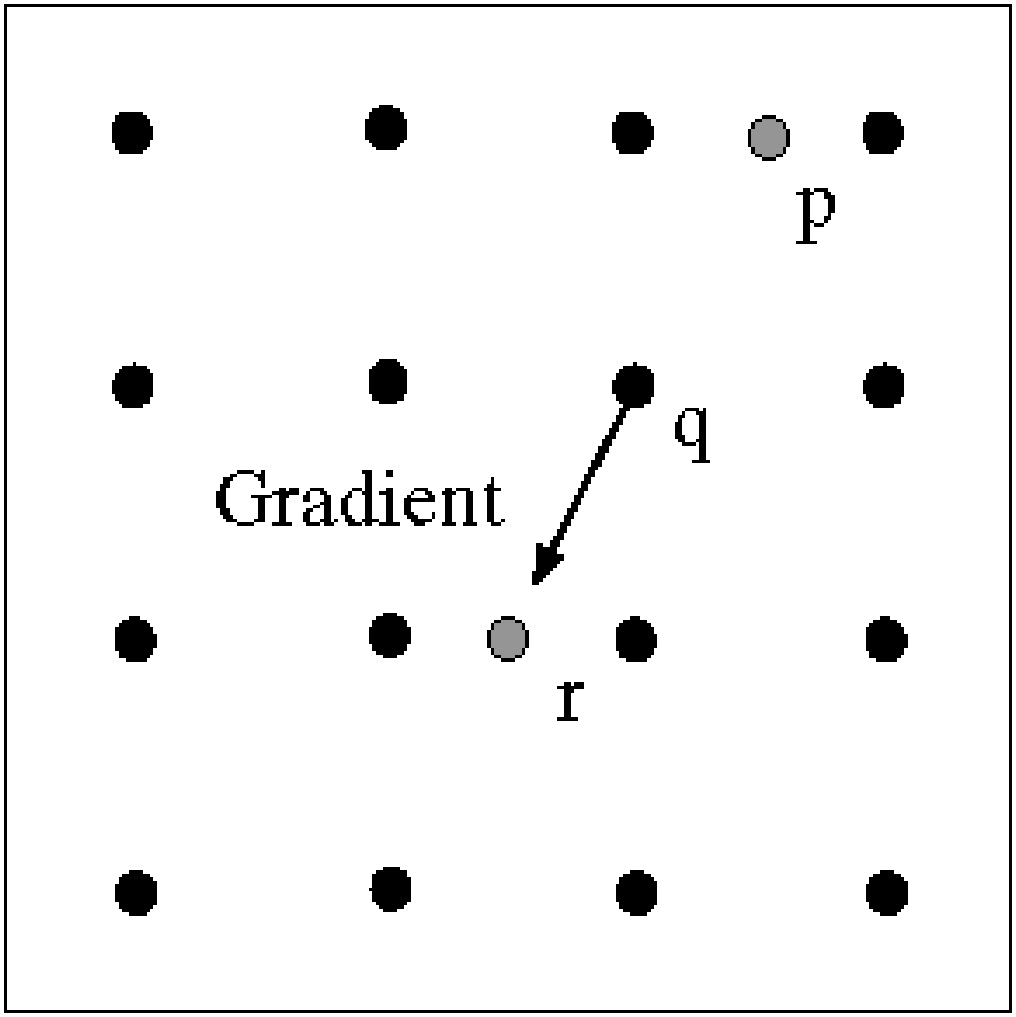
1. Suppress Noise
2. Compute gradient magnitude and direction
3. Apply Non-Maximum Suppression
   - Assures minimal response

NON-MAXIMUM SUPPRESSION

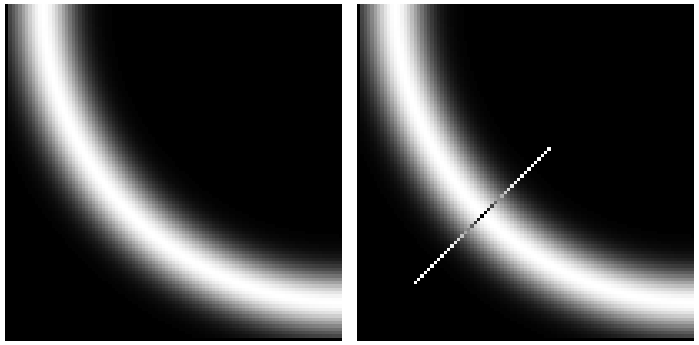
JACOBS UNIVERSITY

1. Edge occurs where gradient reaches a maxima
2. Suppress non-maxima gradient even if it passes threshold
3. Only eight angle directions possible
   - Suppress all pixels in each direction which are not maxima
   - Do this in each marked pixel neighborhood

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

NON-MAX SUPPRESSION

JACOBS UNIVERSITY

Before

After

CANNY EDGE DETECTOR

JACOBS UNIVERSITY

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

JACOBS UNIVERSITY

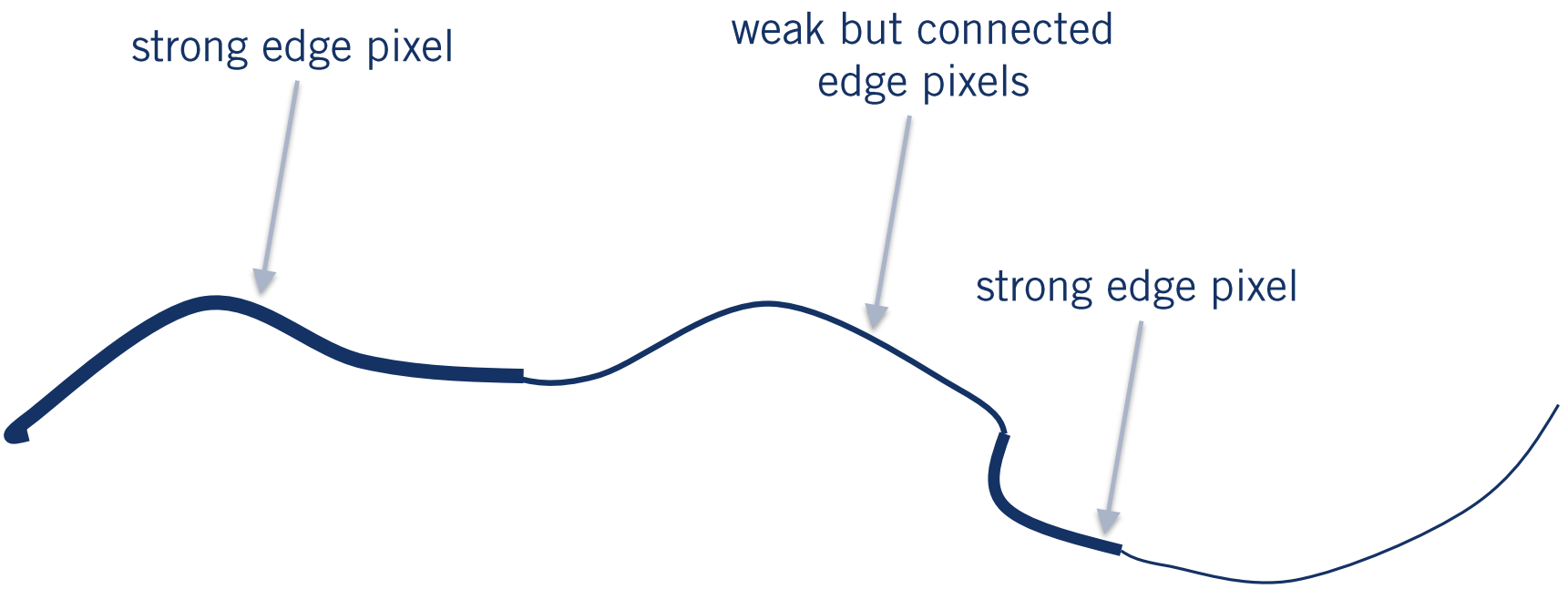
Define two thresholds: Low and High

- If less than Low, not an edge
- If greater than High, strong edge
- If between Low and High, weak edge

If the gradient at a pixel is

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

HYSTERESIS THRESHOLDING



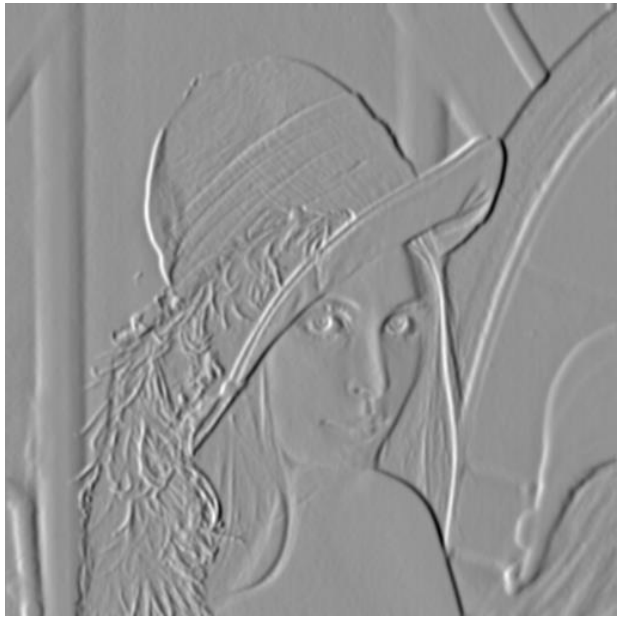
Source: S. Seitz

FINAL CANNY EDGES

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

Original Image

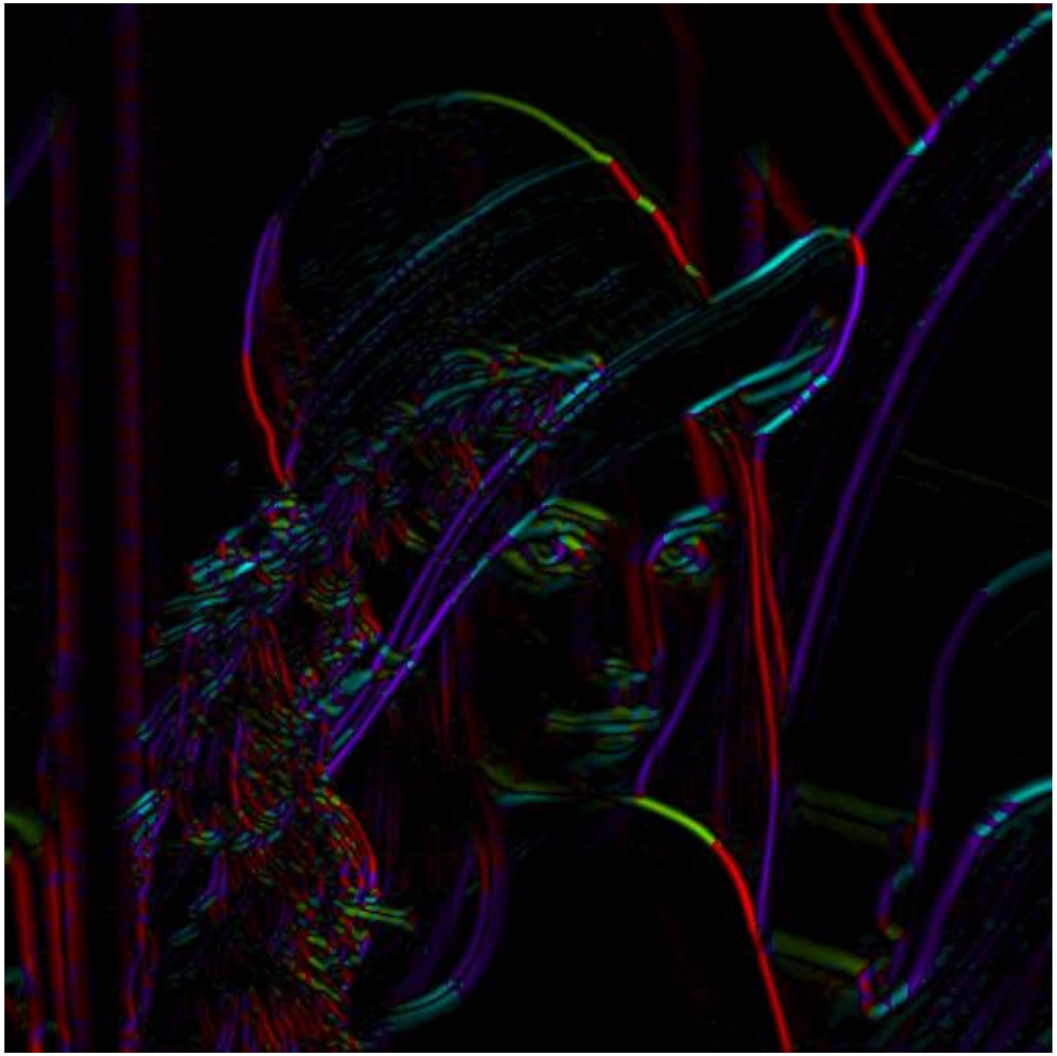
JACOBS UNIVERSITY



X-Derivative

Y-Derivative

Gradient Magnitude



Orientation at each pixel

Before non-max suppression

After non-max suppression

Hysteresis thresholding

Final Canny Edges

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