# Basic Feature Detection

# Outline

**Identifying points of interest**

1. Edge detection
   a. Gradient operators
   b. Zero-crossings of Laplacians
   c. Canny Edge Detector
2. Corner detection

## Computer Vision

**Hubel DH (1988) Eye, Brain and Vision.**
**Olshausen & Field, 1997**

# Learning objectives: what can you do after today?

- Extract edges from images
- Describe different gradient operators
- Describe and compare different edge detection methods
- Describe Canny detector and implement it
- Extract corners from images
- Describe structure tensor

# Edge Detection

edges arise from changes in :

- 1. reflectance
- 2. orientation
- 3. Illumination (e.g. shadows)

Thus, edges are not necessarily relevant to e.g. shape

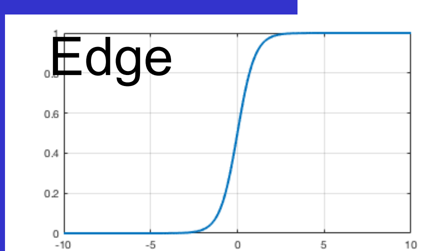
Methods introduced here are only 1st step, edge linking is the hard part

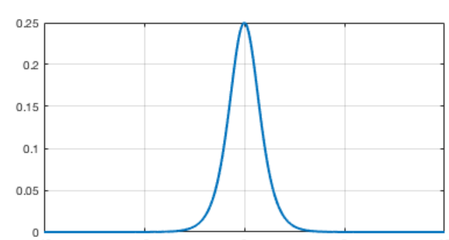
# Edge detection methods

we investigate three approaches :

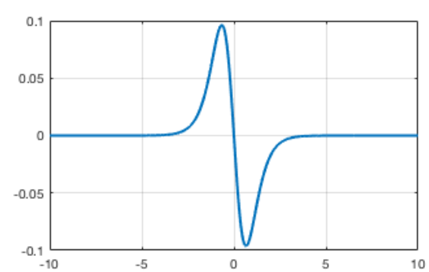
- ❑ 1. locating high intensity gradient magnitudes
- ❑ 2. locating inflection points in the intensity profile
- ❑ 3. signal processing view (optimal detectors)
  Canny edge detector

we will only consider isotropic operators

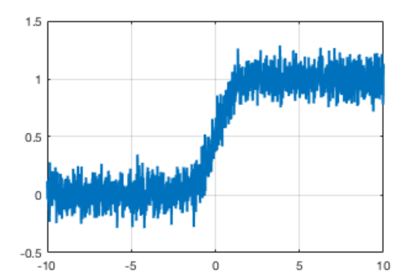


Gradient magnitude



Inflection points



Optimal detector

# Gradient operators : principle

image $f(x,y)$ : locate edges at $f$'s steep slopes

measure the gradient magnitude

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

easy to check that this operator is isotropic

the direction of steepest change : rotate coordinate frame and find θ that maximizes

$$\frac{\partial f}{\partial x'} = \frac{\partial f}{\partial x}\cos\theta + \frac{\partial f}{\partial y}\sin\theta$$

differentiation w.r.t. θ yields

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

the corresponding magnitude is the one defined above

# Gradient operators : implementation

Gradient magnitude is a non-linear operator

$\frac{\partial}{\partial x}$ and $\frac{\partial}{\partial y}$ are linear and shift-invariant

they can thus be implemented as a convolution

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

| | |
|---|---|
| -1 | 1 |

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

| |
|---|
| -1 |
| 1 |

Prone to noise!
We want something that will smooth and compute gradients

# Gradient operators : Sobel

discrete approximation (finite differences) :

$$\frac{\partial}{\partial x} \begin{array}{|c|c|c|} \hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array} \qquad \frac{\partial}{\partial y} \begin{array}{|c|c|c|} \hline -1 & -2 & -1 \\ \hline 0 & 0 & 0 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

these are the *Sobel masks*

# Gradient operators : Sobel

one mask primarily for vertical and one for horizontal edges

combine their outputs :

- 1. take the square root of the sum of their squares
- 2. take arctan of their proportion to obtain edge orientation

these masks are separable, e.g.

$$(-1,0,1) \otimes (1,2,1)^T$$

easy to implement in hardware

Computer Vision

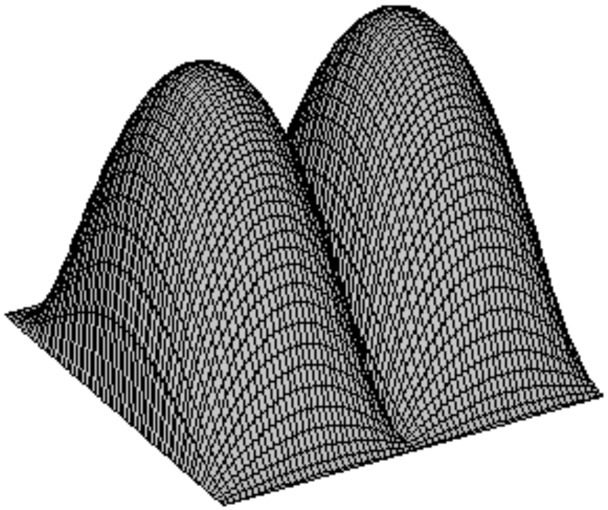
# Gradient operators : MTF shows smoothing effect of Sobel mask

example : MTF of the vertical Sobel mask

$$(2i \sin 2\pi u)(2\cos 2\pi v + 2)$$

this is a pure imaginary function, resulting in $\pi/2$ phase shifts

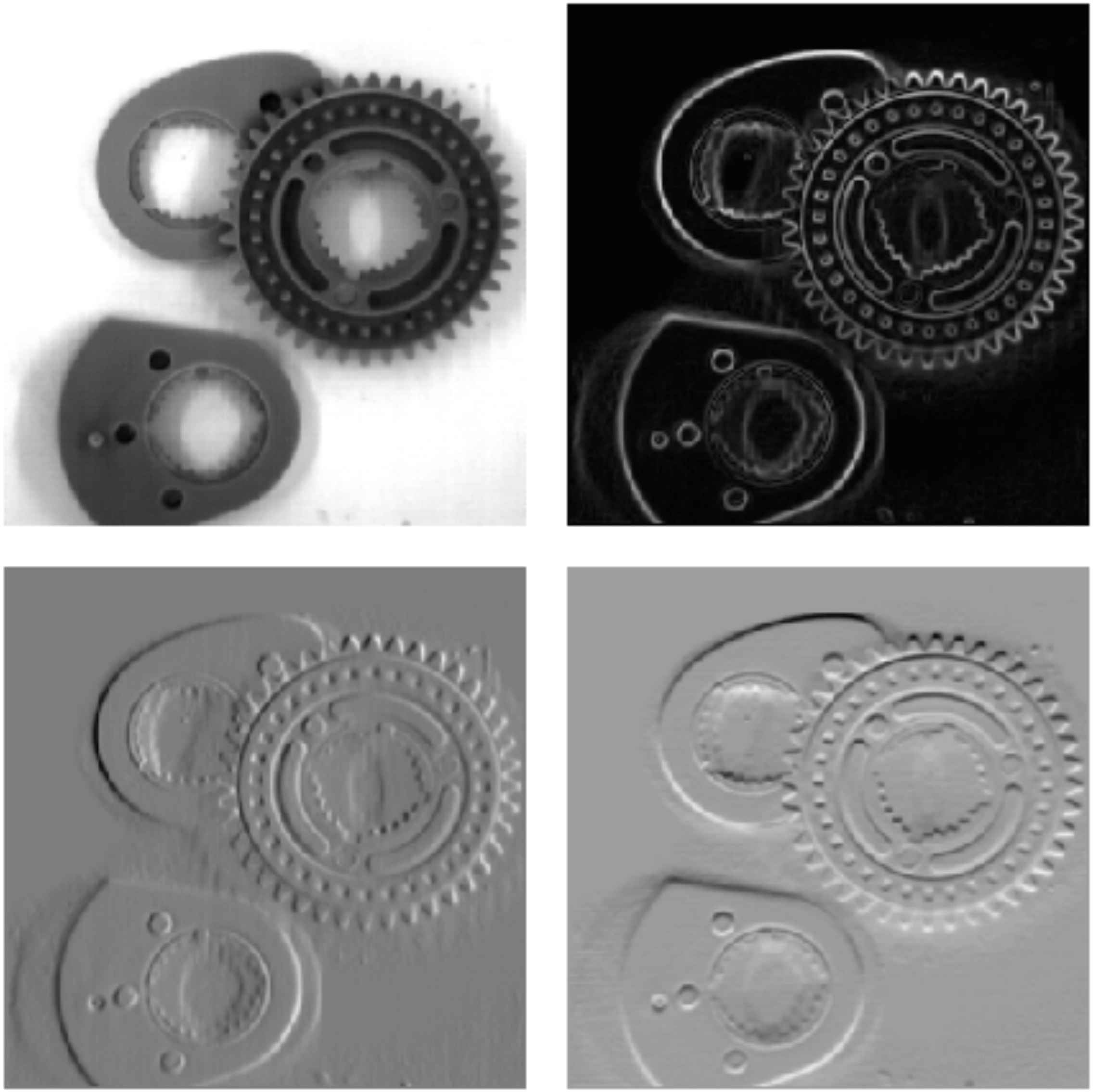
power spectrum :



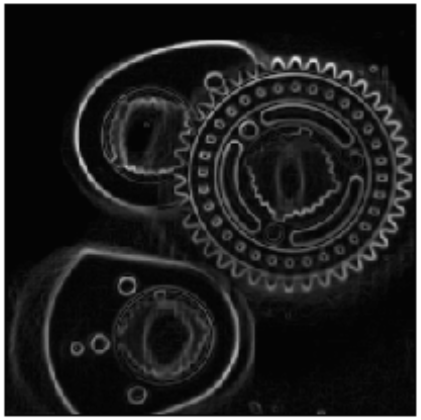
$u$-dir. : band-pass, $v$-dir. : low-pass

Computer Vision

# Gradient operators : example



# Gradient operators : analysis



result far from a perfect line drawing :

1. gaps
2. several pixels thick at places
3. some edges very weak , whereas others are salient

Sobel masks are the optimal 3 x 3 convolution filters with integer coefficients for step edge detection

# Outline

**Identifying points of interest**

1. Edge detection
   a. Gradient operators
   **b. Zero-crossings of Laplacians**
   c. Canny Edge Detector
2. Corner detection

# Zero-crossings : principle

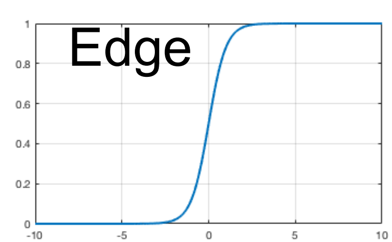
consider edges to lie at intensity inflections

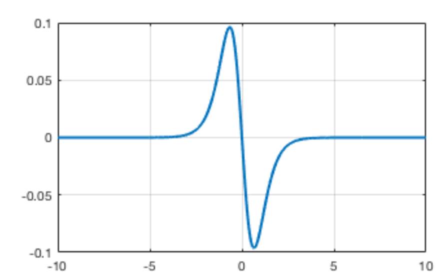
can be found at the zero-crossings of the Laplacian :

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = 0$$

= linear + shift-invariant $\Rightarrow$ convolution

= also isotropic




Inflection points

# Discrete approximations of the Laplacian

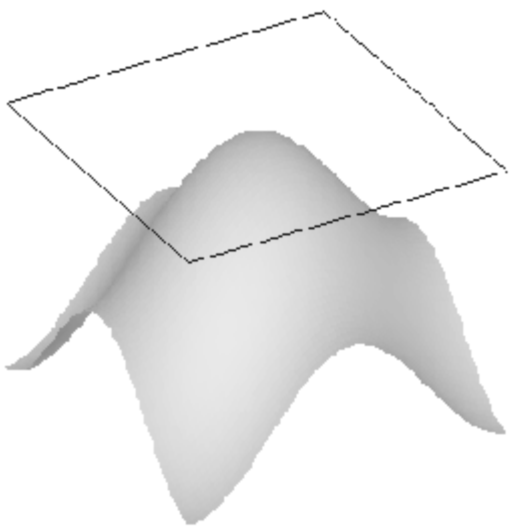
| | | |
|---|---|---|
| 0 | 1 | 0 |
| 1 | - 4 | 1 |
| 0 | 1 | 0 |

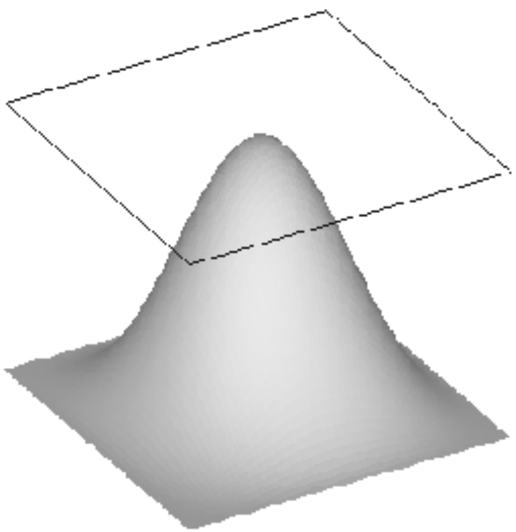
| | | |
|---|---|---|
| 1 | 2 | 1 |
| 2 | -12 | 2 |
| 1 | 2 | 1 |

MTF of left filter :

$$2\cos(2\pi u)+2\cos(2\pi v)-4$$

MTF's





Computer Vision

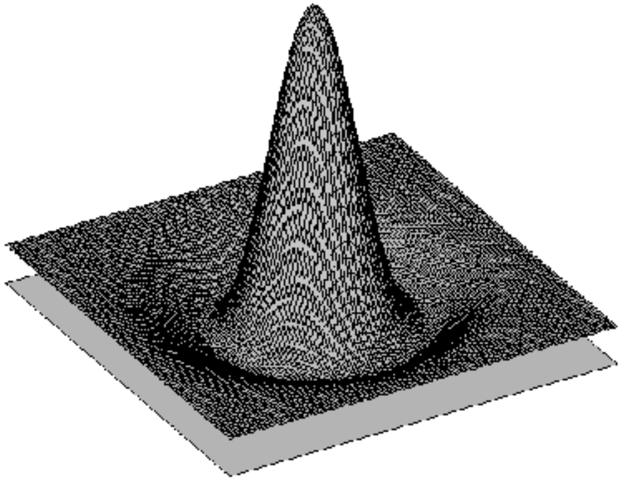
# Zero-crossings : implementation

sensitive to noise (2nd order der.)

therefore combined with smoothing, e.g. a Gaussian :

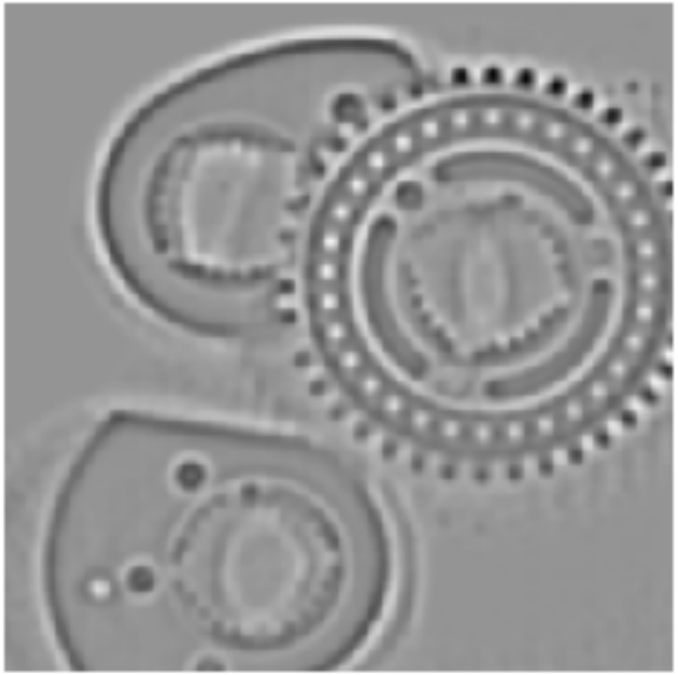
$$L * (G * f) = (L * G) * f$$

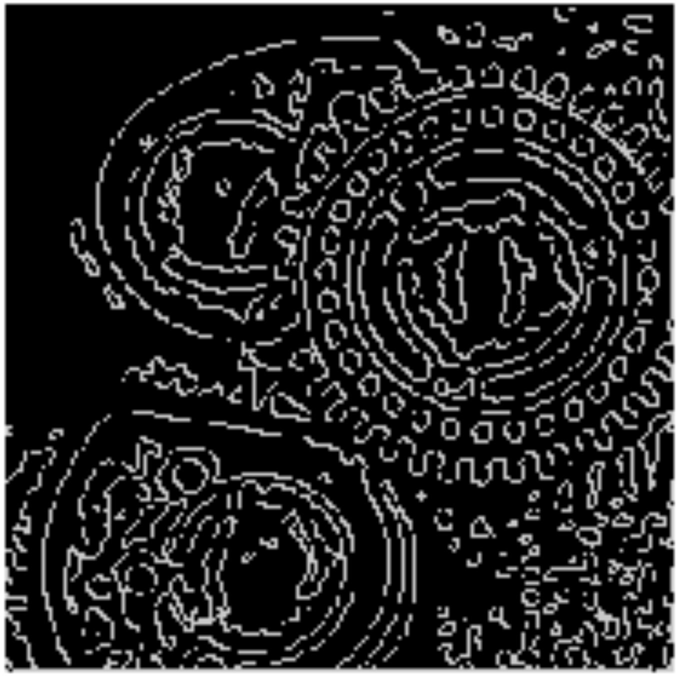
yields *"Mexican hat"* filter :



also implemented as DOG (difference of Gaussians)

# Zero-crossings : example





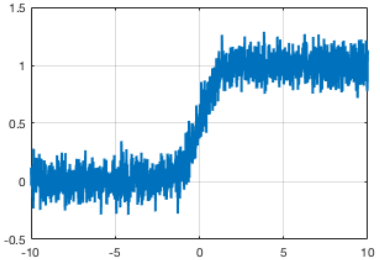
one-pixel thick edges
closed contours
yet not convincing

# Outline

**Identifying points of interest**

1. Edge detection
   a. Gradient operators
   b. Zero-crossings of Laplacians
   **c. Canny Edge Detector**
2. Corner detection

# The Canny edge detector




Optimal
detector

A (1D) signal processing approach

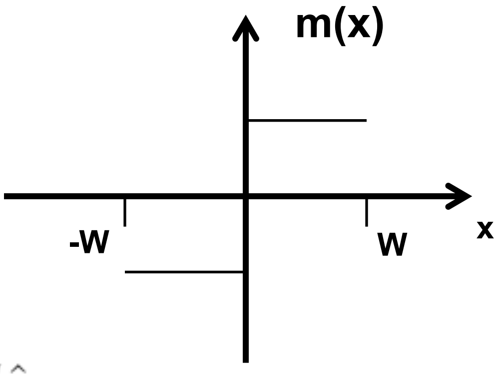
Looking for "optimal" filters

Optimality criteria

- Good SNR
  strong response to edges
  low (no) response to noise
- Good localization
  edges should be detected on the right position
- Uniqueness
  edges should be detected only once

Computer Vision

# Characterization of SNR




- Response of system ***h*** to signal
  deterministic signal model *m*(*x*)
  step edge at the origin

$$\boldsymbol{h}[m(x)] = \int_{-W}^{W} h(x-\hat{x})m(\hat{x})d\hat{x}$$

  at the edge position

$$\boldsymbol{h}[m](0) = \int_{-W}^{W} h(-\hat{x})m(\hat{x})d\hat{x}$$

- Response to noise *n*(*x*)
  noise is stochastic (Gaussian, white, uncorrelated)
  can be characterized by expected value

$$\left.\sqrt{E\left[\left(\int_{-W}^{W} h(\hat{x})n(x-\hat{x})d\hat{x}\right)^2\right]}\right|_{x=0} = \sigma\sqrt{\int_{-W}^{W} h^2(\hat{x})d\hat{x}}$$

# Characterization of SNR

$$SNR = \frac{\int_{-W}^{W} h(-\hat{x}) m(\hat{x}) d\hat{x}}{\sigma \sqrt{\int_{-W}^{W} h^2(\hat{x}) d\hat{x}}}$$

# Characterization of localization

- Edge location: maximum of the system response
  extremum of ***h****(m(x)+n(x))* at $x_0$
  again stochastic (depending on the noise)
  will deviate from the ideal edge position at 0
- Quantification through expected value of the deviation from the real edge location $\sqrt{E\left[x_0^2\right]}$
- Localization measure

$$LOK = \frac{1}{\sqrt{E\left[x_0^2\right]}} = \frac{\left|\int_{-W}^{W} h'(\hat{x}) m'(-\hat{x}) d\hat{x}\right|}{\sigma \sqrt{\int_{-W}^{W} \left[h'(\hat{x})\right]^2 d\hat{x}}}$$

# The matched filter

- Optimal filter ***h****(x)* for which

$$\max\left(\mathrm{SNR} \times \mathrm{LOC}\right)$$

- can be shown that

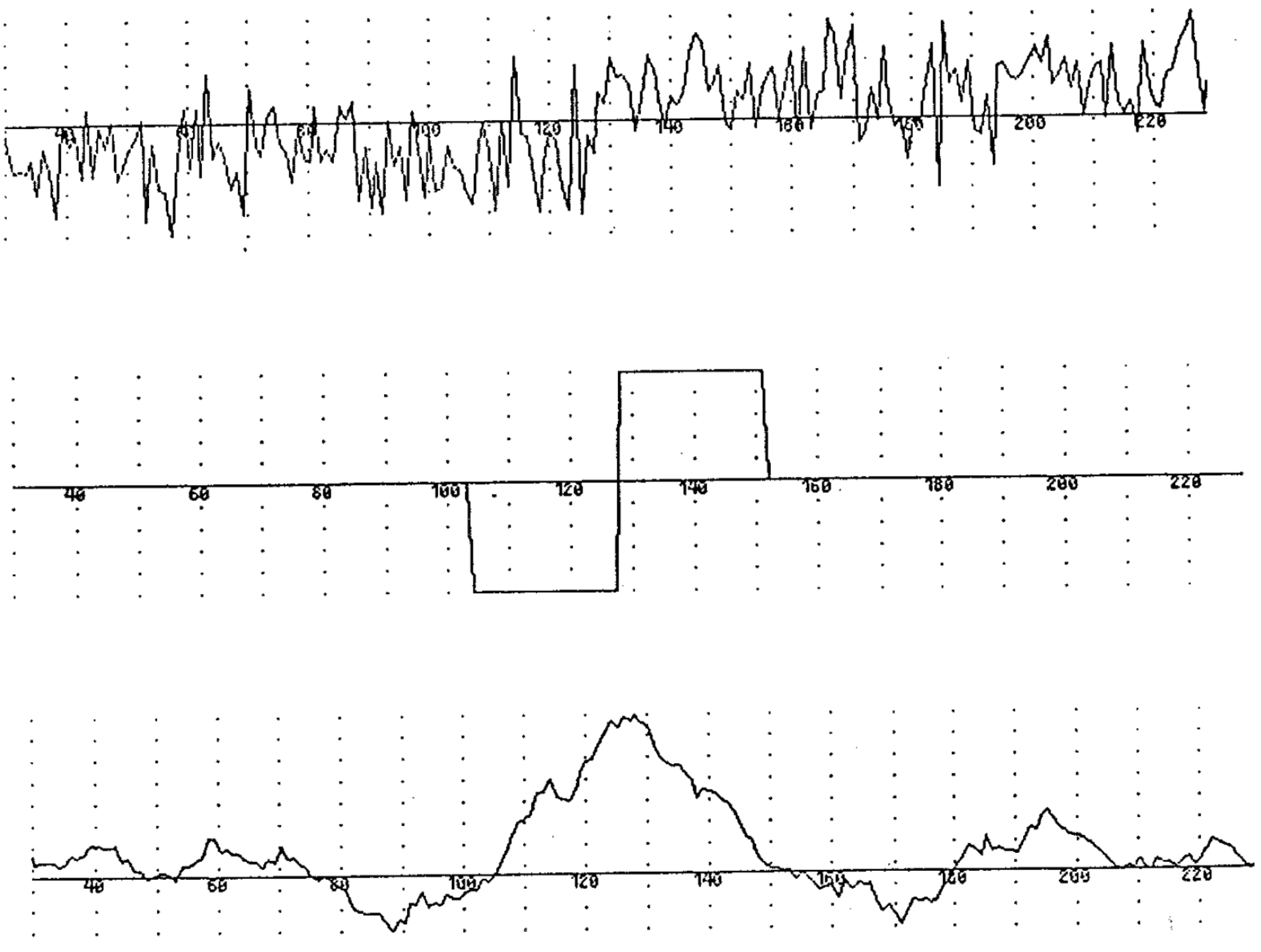
$$h(x) = \lambda m(x) \quad x \in [-W, W]$$

Essentially identical with the signal to be detected

- For the edge model used: difference of boxes DOB Filter

Computer Vision

# The matched filter



# Uniqueness

- Filtering with DOB generates many local maxima due to noise
- Hinders unique detection
- Remedy: minimize the number of maxima within the filter support
- Caused by noise (stochastic)
  Characterized by the average distance between subsequent zero crossings of the noise response derivative (*f* = ***h'****(n)*)
  Rice theorem

$$x_{ave} = \pi \sqrt{\frac{-\Phi_{ff}(0)}{\Phi''_{ff}(0)}}$$

# 1D optimal filter

- Average number of maxima within filter support

$$N_{max} = \frac{2W}{x_{max}} = \frac{W}{x_{ave}}$$

should be minimized

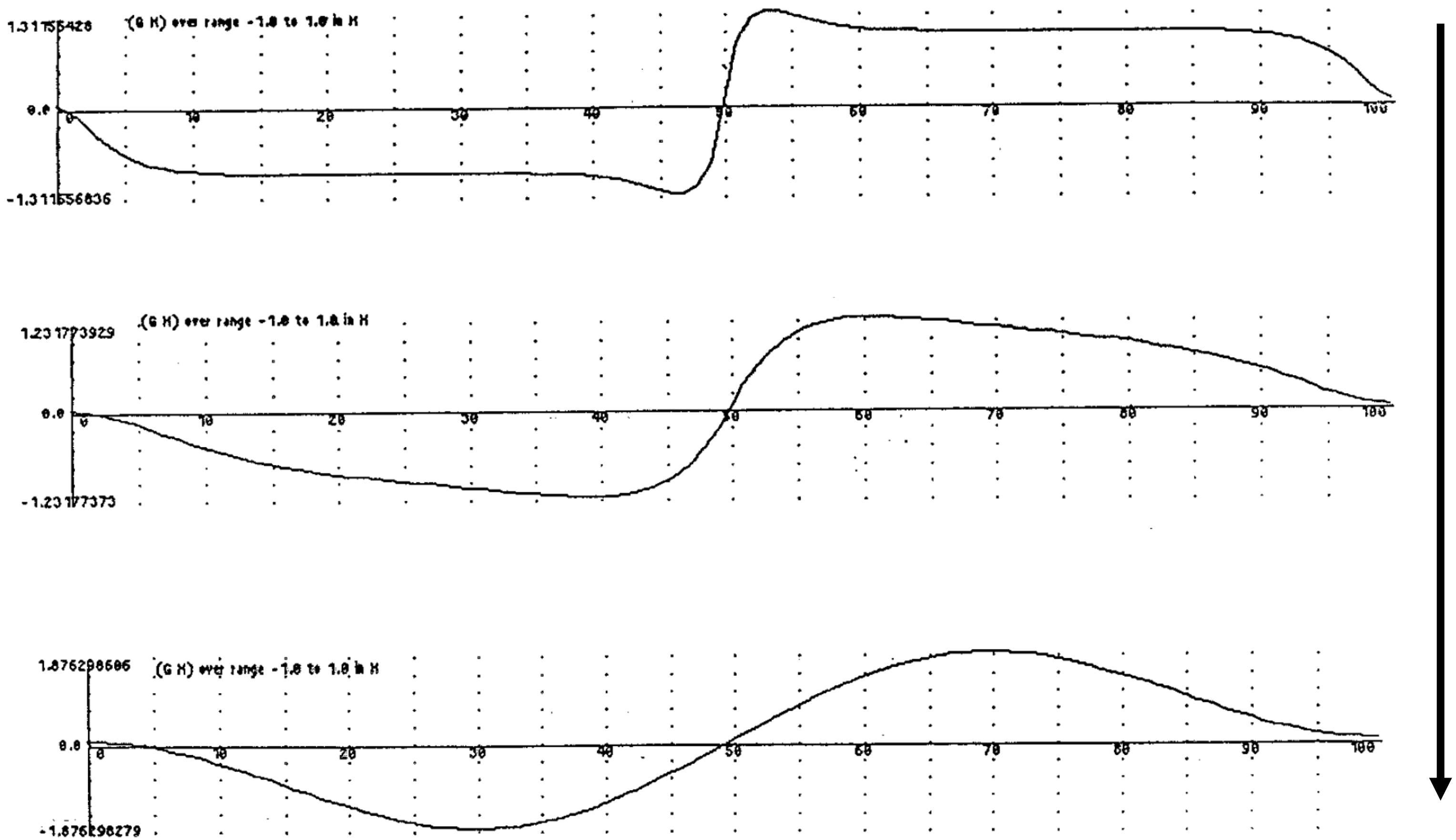
- Overall goal function is a linear combination of the two criteria

$$\max\left(\mathrm{SNR} \times \mathrm{LOC} + c\frac{1}{N_{\max}}\right)$$

the solution depends on c
empirically selected

# 1D optimal filter

Small c

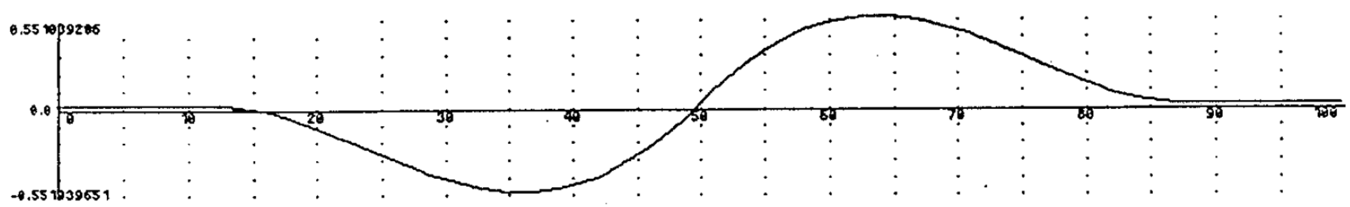



Large c

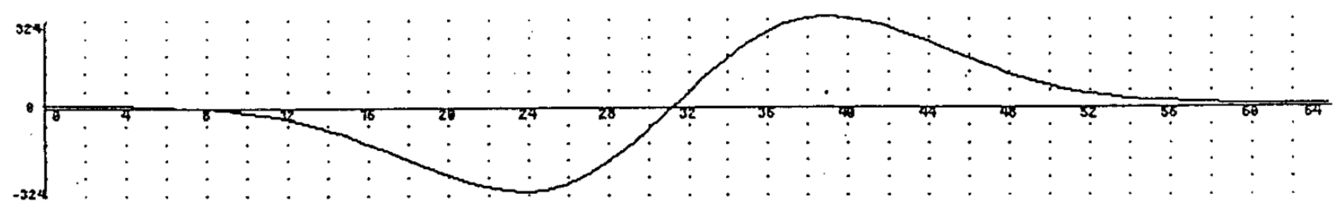
# 1D optimal filter

Resembles the first derivative of the Gaussian

Canny selection



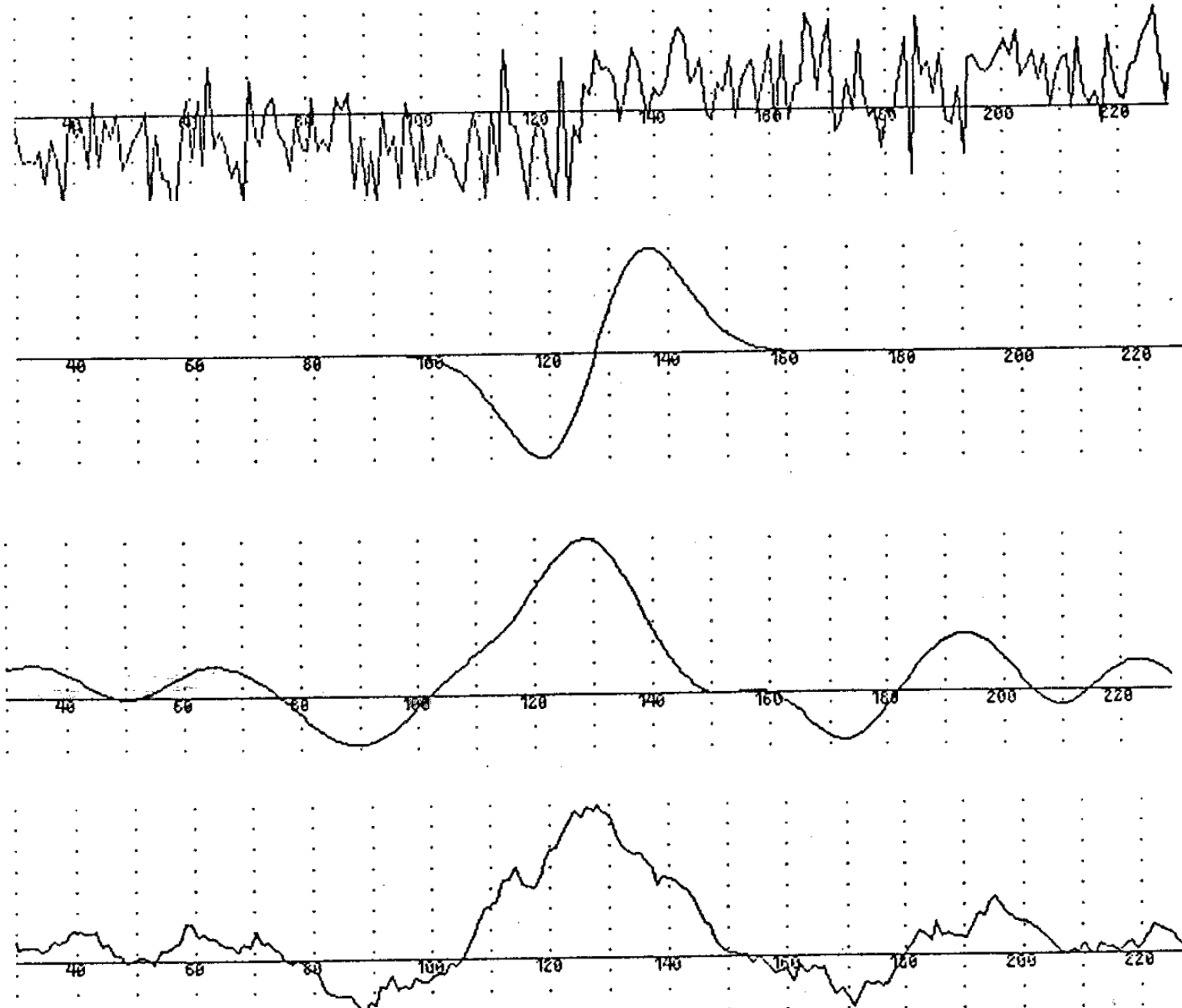

Gaussian derivative




Another first derivative based detector

Computer Vision

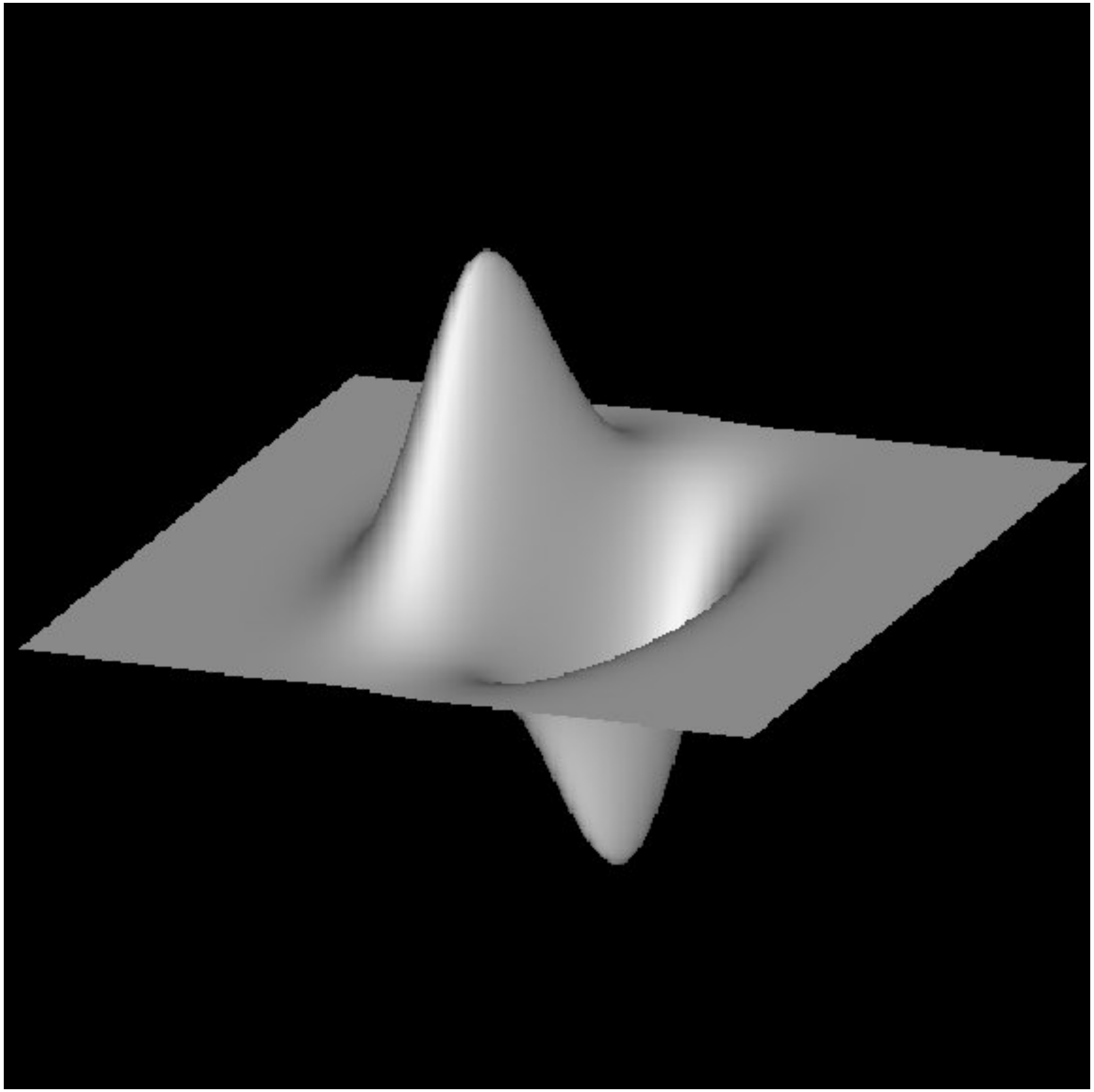
# Optimal filter



# Canny filter in nD

- The Canny filter is essentially 1D
- Extension to higher dimensions
  - simplified edge model
  - intensity variation only orthogonal to the edge
  - no intensity change along the edge
- Combination of two filtering principles
  - 1D Canny filter across the edge
  - (n-1)D smoothing filter along the edge
    Gaussian smoothing is used
- The effective filter is a directional derivative of Gaussian

# The 2D Canny filter

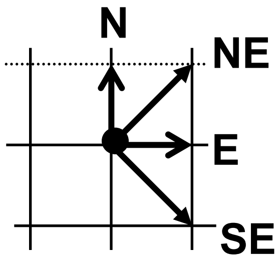


Computer Vision

# 2D implementation on the discrete image raster

- Faithful implementation by selecting gradient direction: does not respect discretization

- Estimation of directional derivatives instead considering neighbours on the image raster




  - Start with 2D Gaussian smoothing $f = G * I$
  - Directional derivatives from discrete differences

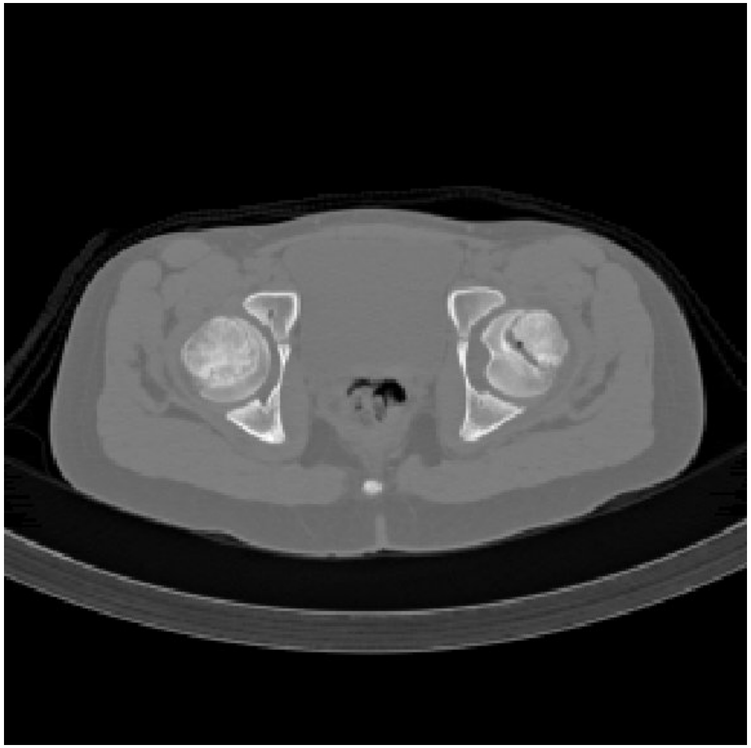
$$f'_N = f(i, j+1) - f(i,j); f'_{NE}(i,j) = (f(i+1, j+1) - f(i,j))/\sqrt{2}$$

$$f'_E = f(i+1, j) - f(i,j); f'_{SE}(i,j) = (f(i+1, j-1) - f(i,j))/\sqrt{2}$$

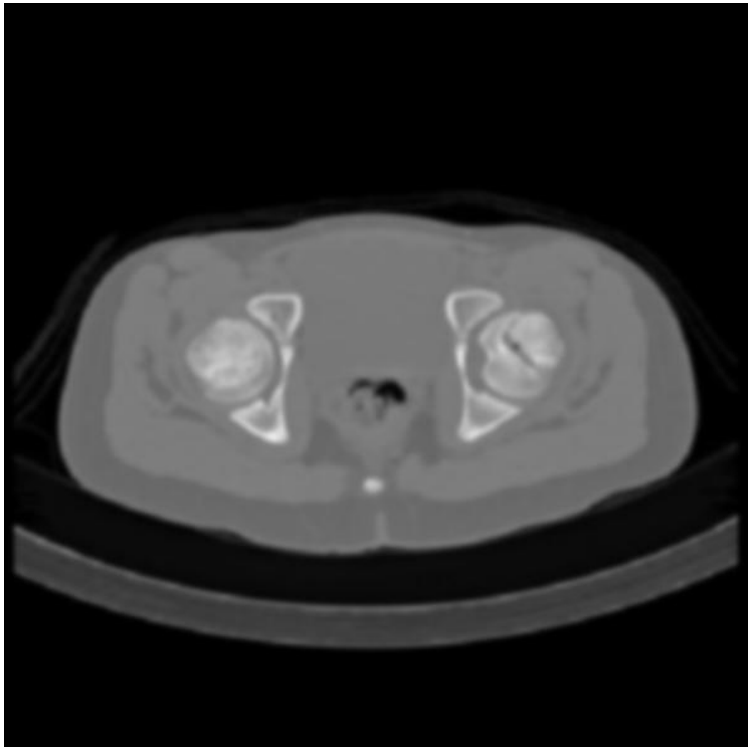
  - Selecting the maximum as gradient approximation

Computer Vision

# Canny 2D results

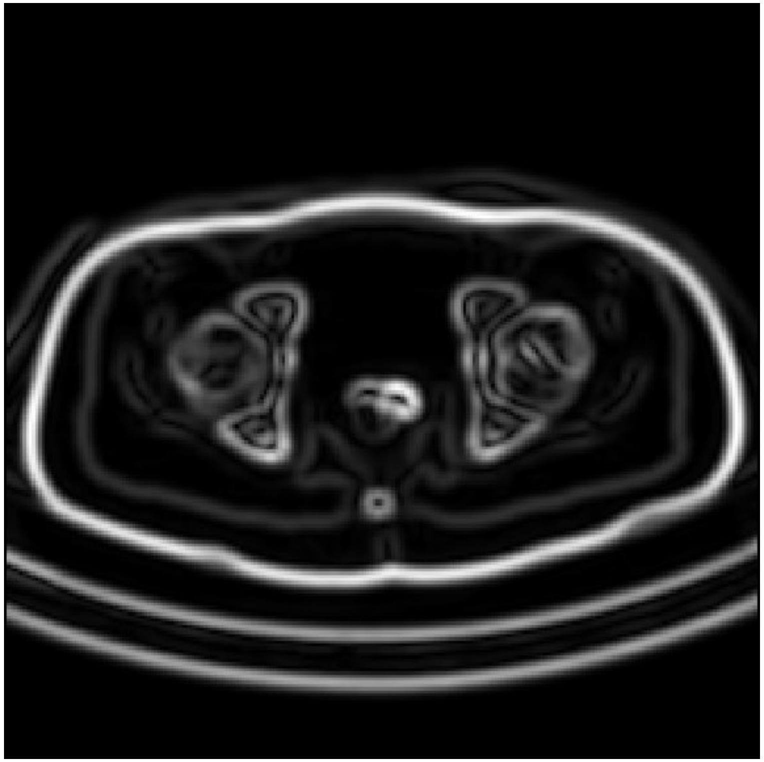


original image



Gaussian smoothing

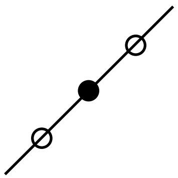
# Canny 2D results



Gradient approximation

# Post-processing steps

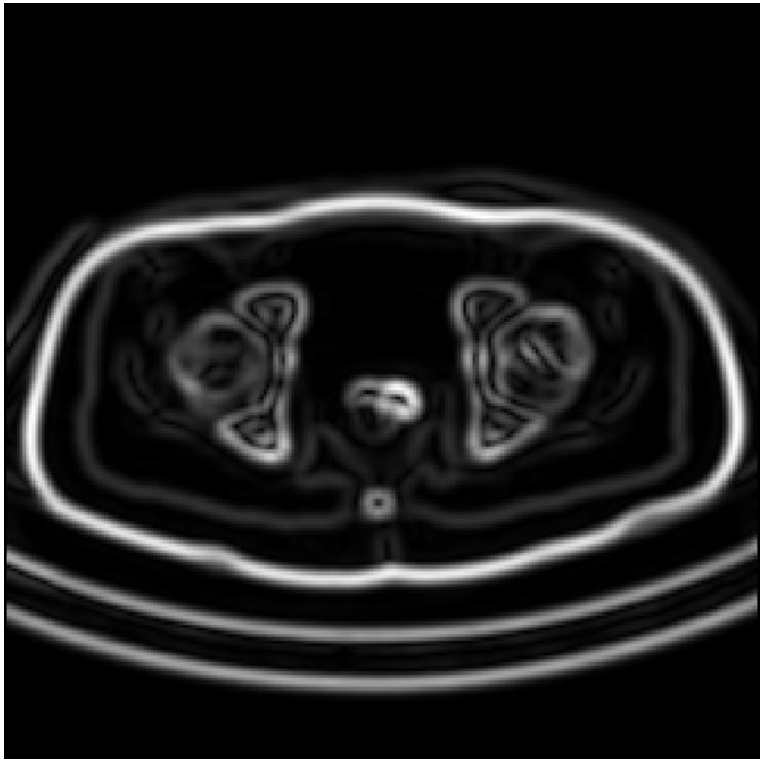
- Non-maximum suppression
  - Comparing derivatives at the two neighbours along the selected direction



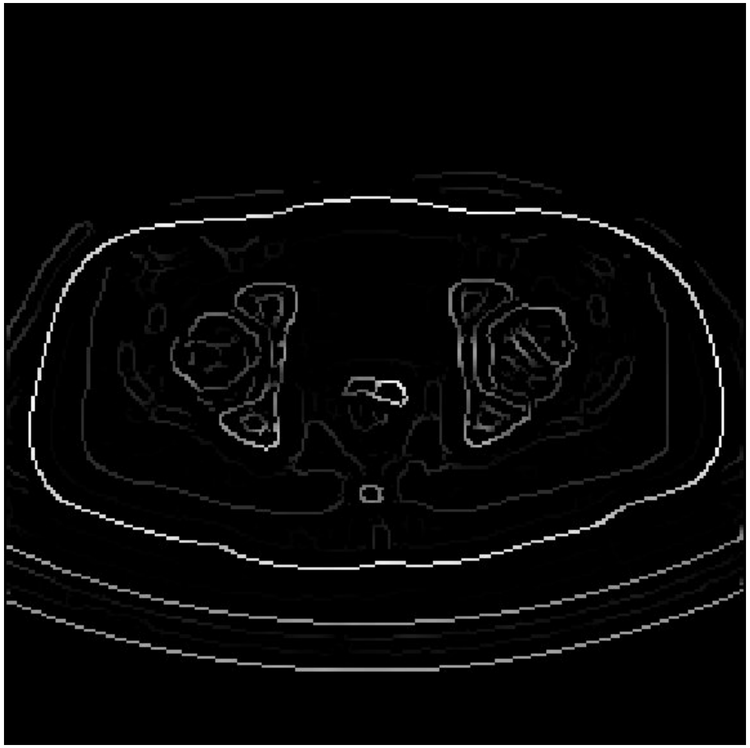
  - Keeping only values which are not smaller than any of them

- Hysteresis thresholding
  - using two treshold values $t_{low}$ and $t_{high}$
  - keep class 1 edge pixels for which $|f(i,j)| \geq t_{high}$
  - discard class 2 edge pixels for which $|f(i,j)| < t_{low}$
  - for class 3 edge pixels $t_{high} > |f(i,j)| \geq t_{low}$ keep them only if connected to class 1pixels through other class 3 pixels

# Canny 2D results



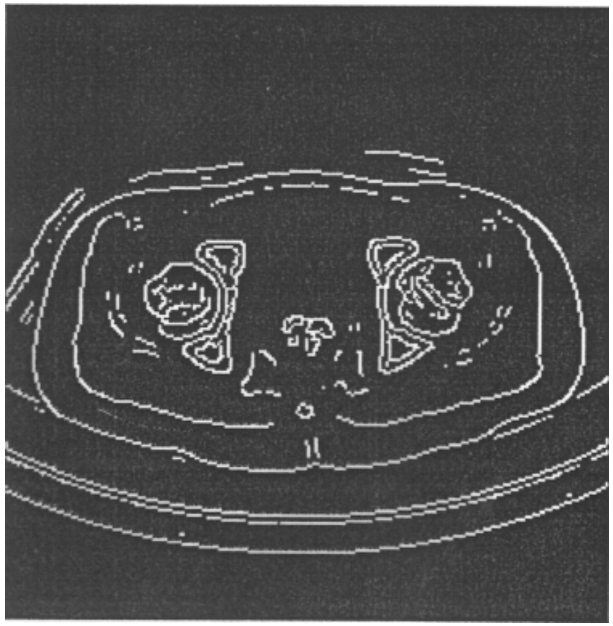
Gradient approximation



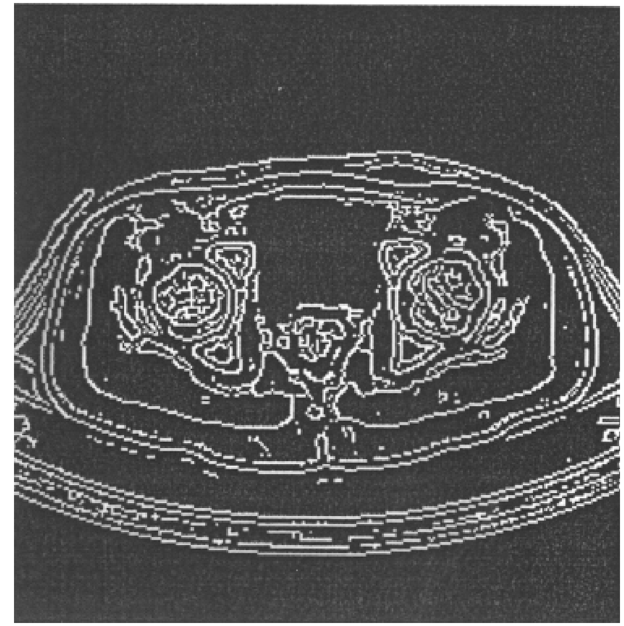
non-maximum suppresion

Computer Vision

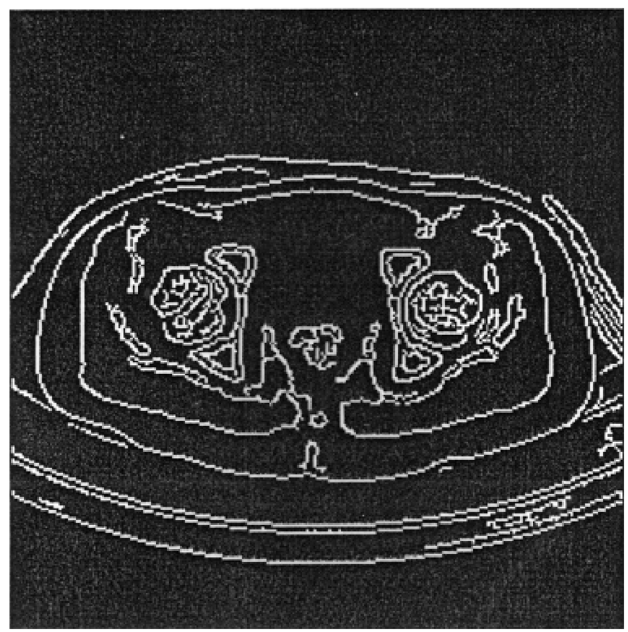
# Canny 2D results



threshold $t_{high}$



threshold $t_{low}$



hysteresis thresholding

➔

# Remarks to the Canny filter

- The state-of-the-art edge detector even today
- Very efficient implementation
  no interpolation is needed as respecting the raster
- Post-processing is the major contribution
- Can be applied to any gradient-based edge detection scheme
- Fails where the simplified edge model is wrong
  - crossing, corners, …
  - gaps can be created
  - mainly due to non-maximum suppression
- Hysteresis thresholding is only effective in 2D

# A Post Scriptum

e.g. Sobel filter reflects intensity pattern to be found

other example, line detector

| | | |
|---|---|---|
| -1 | -1 | -1 |
| 2 | 2 | 2 |
| -1 | -1 | -1 |

in line with the *matched filter theorem*

# Outline

**Identifying points of interest**

1. Edge detection
   a. Gradient operators
   b. Zero-crossings of Laplacians
   c. Canny Edge Detector
2. **Corner detection**

# The Harris corner detector

Goal : one approach that distinguishes

- *1. homogeneous areas*
- *2. edges*
- *3. corners*

Key : looking at intensity variations in different directions :

- *1. small everywhere*
- *2. large in one direction, small in the others*
- *3. Large in all directions*

Computer Vision

# The Harris corner detector

Homogeneous region | Edge region | Corner region

Approach : find the directions of minimal and maximal change

Second order moments of the intensity variations also called the structure tensor:

$$\begin{pmatrix} \left\langle \left(\frac{\partial f}{\partial x}\right)^2 \right\rangle & \left\langle \left(\frac{\partial f}{\partial x}\frac{\partial f}{\partial y}\right) \right\rangle \\ \left\langle \left(\frac{\partial f}{\partial x}\frac{\partial f}{\partial y}\right) \right\rangle & \left\langle \left(\frac{\partial f}{\partial y}\right)^2 \right\rangle \end{pmatrix}$$

Look for the *eigenvectors* and *eigenvalues*

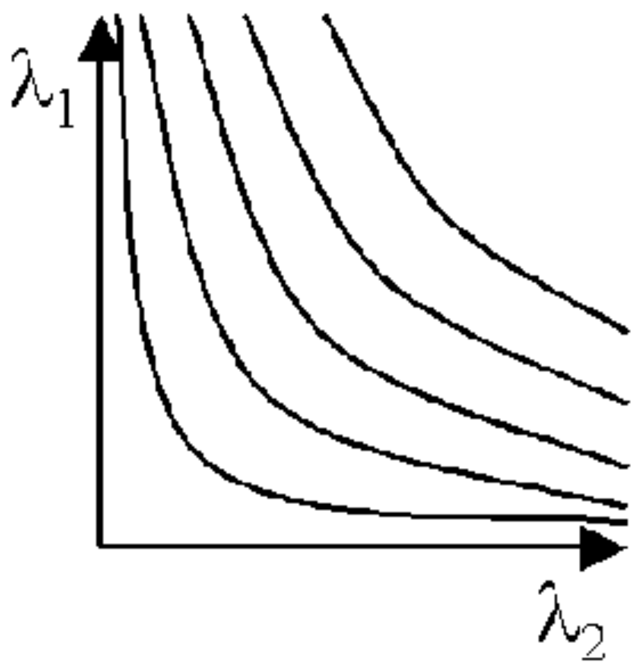
# The Harris corner detector

The classification can be made as

- 1. two small eigenvalues
- 2. one large and one small eigenvalue
- 3. two large eigenvalues

First attempt : determinant of the 2nd-order matrix,

i.e. the product of the eigenvalues :

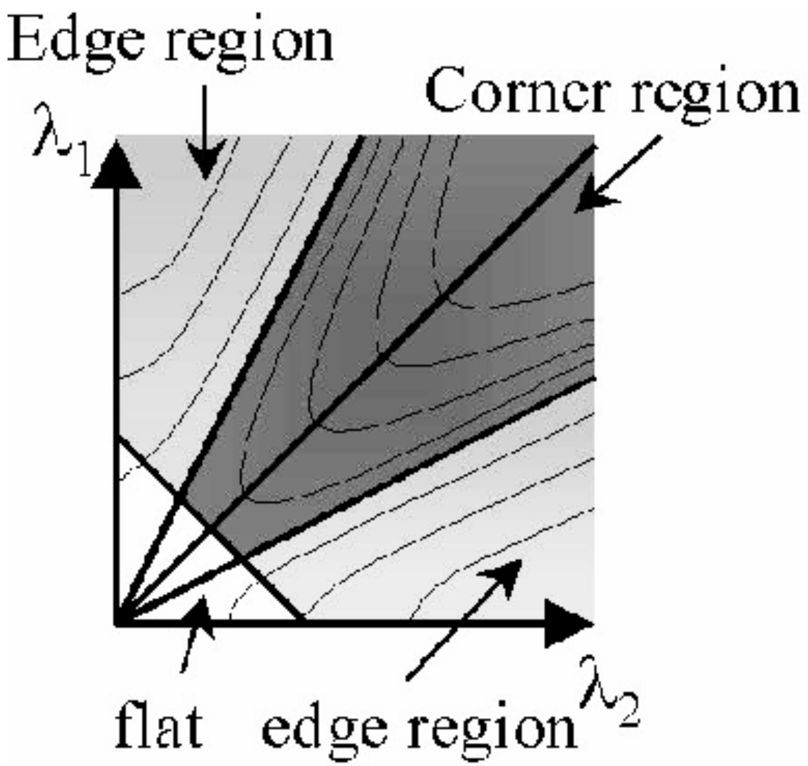


# The Harris corner detector

Edges are now too vague a class : one very large eigenvalue can still trigger a corner response.

A refined strategy :

Use iso-lines for *Determinant - k (Trace)*$^2$.



Computer Vision

# The Harris corner detector

# The Harris corner detector