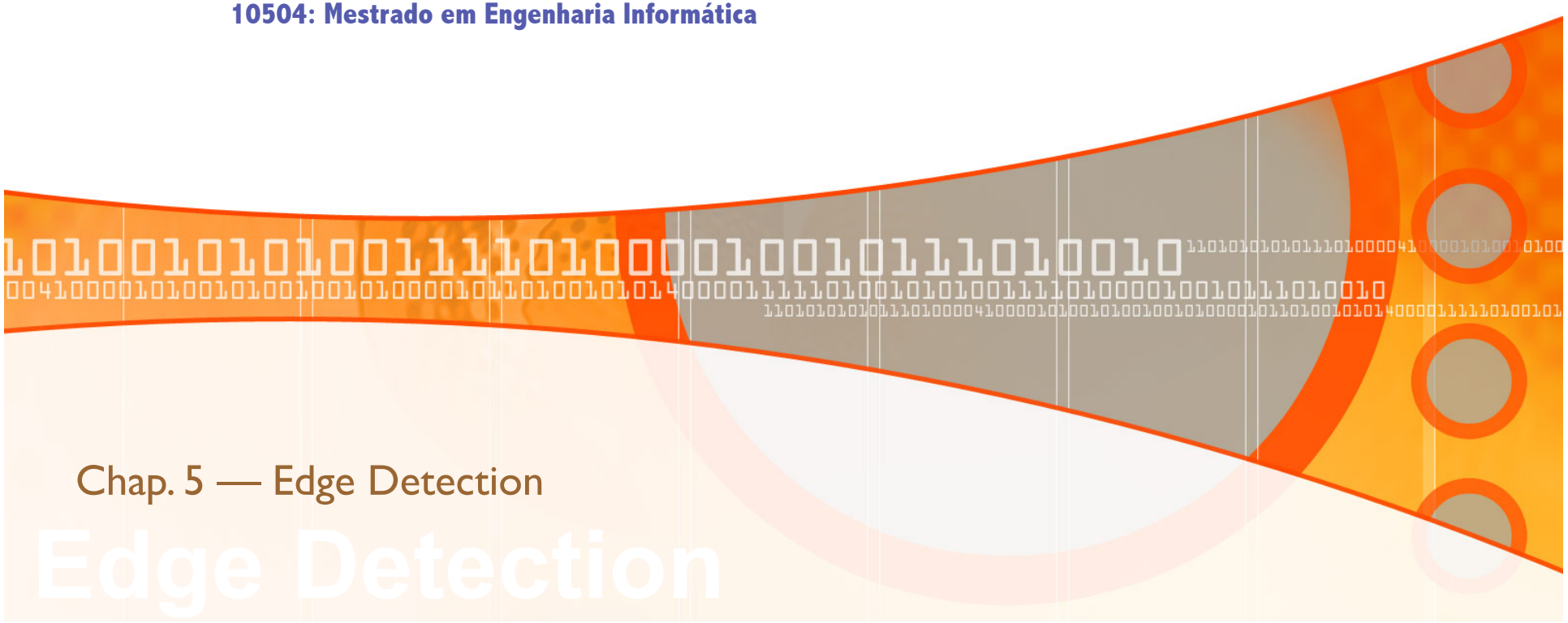
Computação Visual e Multimédia

10504: Mestrado em Engenharia Informática



Chap. 5 — Edge Detection

Edge Detection

Outline

...:

- Image segmentation algorithms: overview
  - Discontinuity-based algorithms (or edge-based algorithms)
  - Similarity-based algorithms (or region-based algorithms)
- Edge detection:
  - Point detection
  - Line detection
  - Edge detection:
    - Basic gradient-based
    - LoG
    - Canny

Monochrome image segmentation algorithms

Definition:

- Segmentation is to subdivide an image into its component regions or objects.
- Segmentation should stop when the objects of interest have been isolated.

Categories:

- Segmentation algorithms for monochrome images generally are based on one of two basic properties of image intensity values: discontinuity and similarity.
- Discontinuity-based algorithms. The idea is to partition an image based on abrupt change in intensity, such as edge.
  - Discontinuities detection: point, line, edge.
- Similarity-based algorithms. The principal approaches in the second category are based on portioning an image into regions that are similar according to a set of predefined criteria.
  - Thresholding techniques, region-oriented approaches, watershed segmentation (generally involve a prior knowledge).

DISCONTINUITY-BASED ALGORITHMS

(edge-based algorithms)

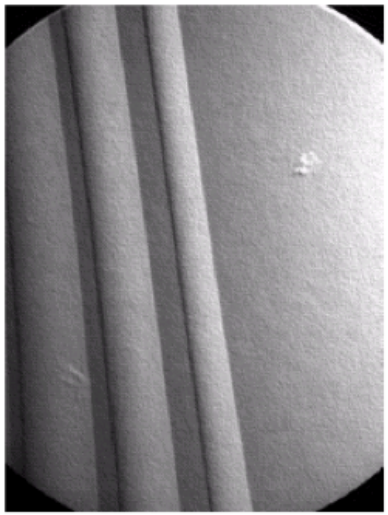
Isolated point detection (Laplacian-based)

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Laplacian mask

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

other typical mask



original image

http://www.cs.bgu.ac.il/~klara/ATCS111/gonzales_10.1_10.2.pdf

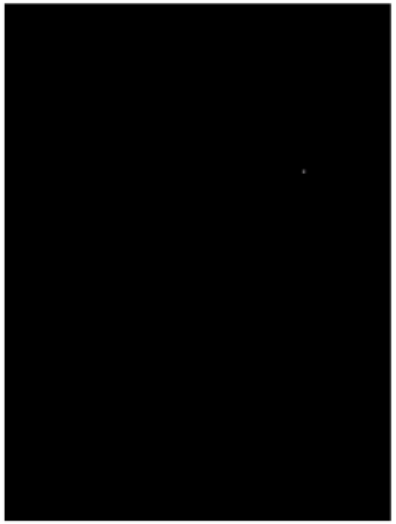
Principle:

- The response of a mask must be the strongest when the mask is centered on an isolated point, and that the response be in areas of constant intensity.

Laplacian-based algorithm:

- Step 1: *Apply the mask over the image*. The detection of an isolated point is carried out using the Laplacian mask because the second order derivative is very sensitive to sudden intensity changes.
- Step 2: *Apply thresholding*. We will now decide about a threshold T, and any pixel whose value is larger than T after being masked will be considered as an isolated point.

masked image



thresholded image

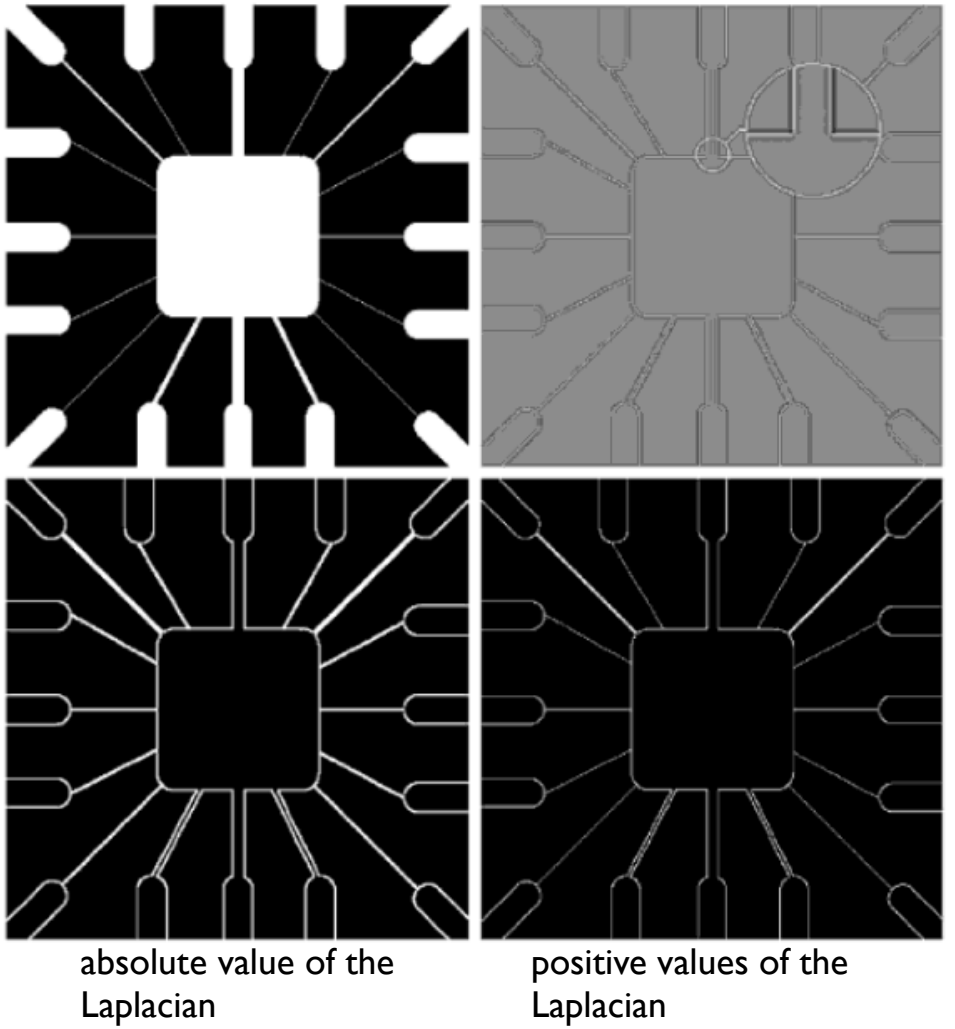
Line detection (Laplacian-based)

Principle:

- Taking into account that the Laplacian is sensitive to sudden changes and thin lines, we can use it also for detection of lines.

Be aware that:

- We must note that since the second derivative changes its sign on a line it creates a "double line effect" and it must be handled.
- Second derivative can have negative results and we need to scale the results.



http://www.cs.bgu.ac.il/~klara/ATCS111/gonzales_10.1_10.2.pdf

Line detection

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

horizontal 45° vertical -45°

Principle:

- The Laplacian is isotropic, i.e. independent of direction.
- If we would like to detect lines on a certain direction only we might want to use masks that would emphasize a certain direction and be less sensitive to other directions.

binary wire-bond mask

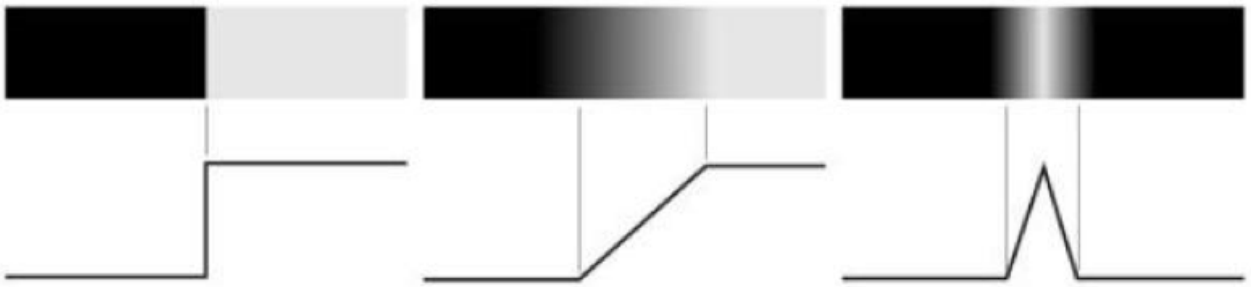
absolute value with a -45° line detector

Result of threshold image

Edge models

Three edge types and definitions:

- **Step** edge – Transition of intensity level over 1 pixel only in ideal, or few pixels on a more practical use
- **Ramp** edge – A slow and graduate transition
- **Roof** edge – A transition to a different intensity and back. Some kind of spread line.



step edge ramp edge roof edge

intensity profiles for different types of edges

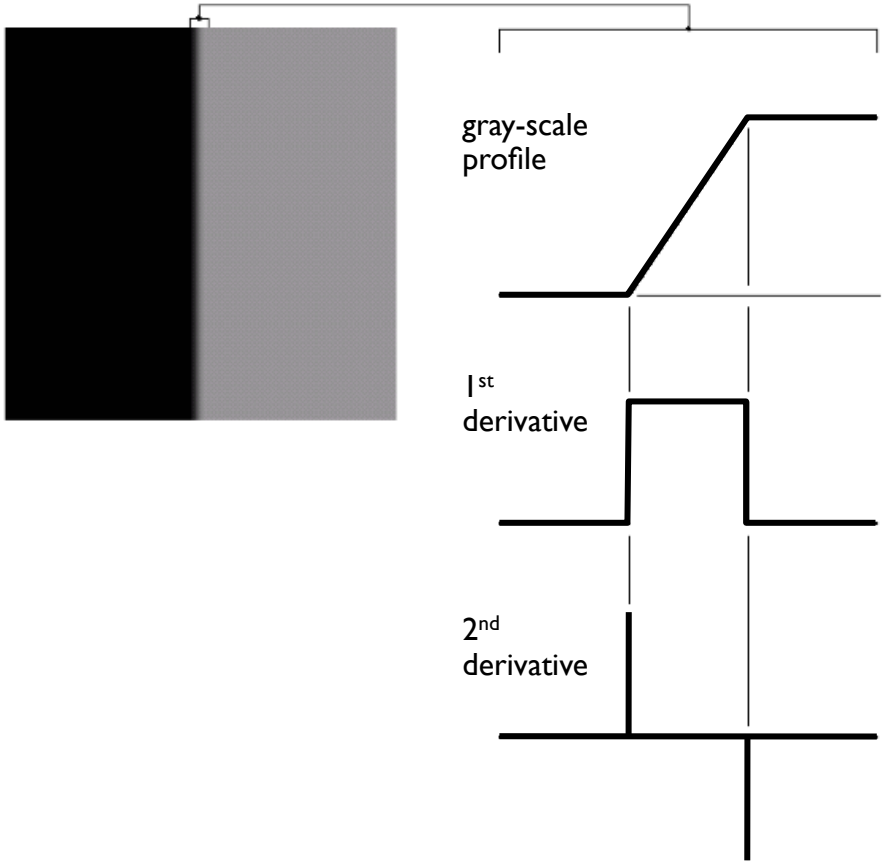
Edge models: derivatives

1st Derivative:

- Magnitude of the first derivative can be used to detect an edge.

2nd Derivative:

- The sign of the second derivative indicates the direction of the edge (black to white or white to black)



Edge models: derivatives (cont'd)

1st Derivative:

- Magnitude of the first derivative can be used to detect an edge.
- *Example* (in the x-direction):
  - The gradient can be calculated by convolving the original data with a mask [-1/2 0 +1/2]:

    $G_X(x) = -1/2 \cdot f(x-1) + 0 \cdot f(x) + 1/2 \cdot f(x+1)$

5	7	6	4	152	148	149

$$G_X(4) = -\frac{1}{2} \times 6 + 0 \times 4 + \frac{1}{2} \times 152$$

2nd Derivative:

- The sign of the second derivative indicates the direction of the edge (black to white or white to black)
- *Example* (in the x-direction):
  - The Laplcian can be calculated by convolving the original data with a mask [+1 -2 +1]:

    $L_X(x) = 1 \cdot f(x-1) - 2 \cdot f(x) + 1 \cdot f(x+1)$

$$L_X(4) = 1 \times 6 - 2 \times 4 + 1 \times 152$$

Basic edge detection (gradient-based)

Algorithm:

- Step 1: filter noise using mean filter
- Step 2: compute gradient or variants (e.g., Sobel, Prewitt, etc.)
- Step 3: mark points > *threshold* as edges

Sobel filter provides a better noise supression than Prewitt filter

original image | $|G_X|$ | $|G_Y|$ | $|G_X|+|G_Y|$

Advanced edge detection

What does "advanced" word mean?

- The basic edge detection method is based on simple filtering without taking note of image characteristics and other information such as noise, scaling etc.

Two advanced techniques:

- Marr-Hildreth [1980] ; Canny [1986]

The leading idea:

- Marr and Hildreth argued that:
  - Intensity of changes is not independent of image scale.
  - Sudden intensity change will cause a zero-crossing of the second derivative.
- To cope with these two requirements, an edge detection operator should
  - be capable of being tuned to any scale
  - be capable of computing the first and second derivatives.

LoG edge detection (Marr-Hildreth)

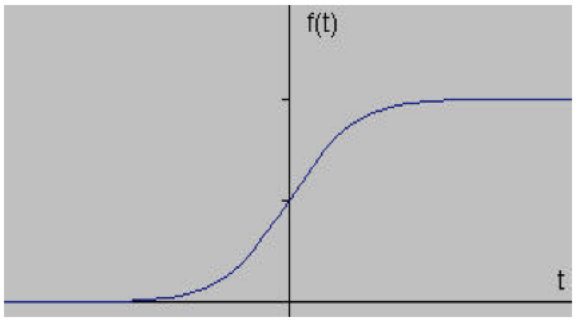
Definition:

- This method combines Gaussian filtering with the Laplacian for edge detection. It is known as "Laplacian of Gaussian" (**LoG**).

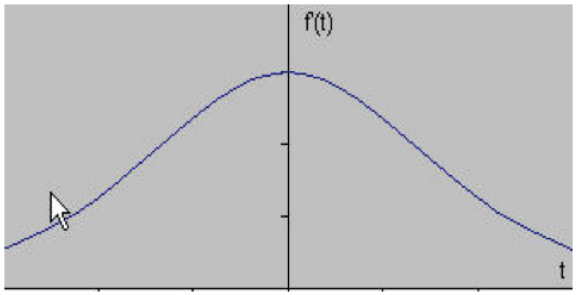
How does it work?

- The edge points of an image can be detected by finding the zero crossings of the 2nd derivative of the image intensity.
- But, the 2nd derivative is very sensitive to noise. This means that the noise should be filtered out before edge detection. A Gaussian filter is used to swipe away noise from the image.

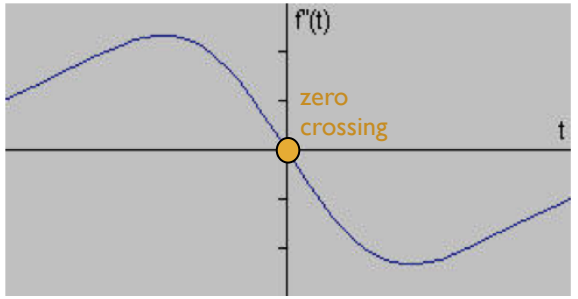
function f



1st derivative f'



2nd derivative f"



http://euclid.ii.metu.edu.tr/~ion528/demo/lectures/6/3/index.html

LoG edge detection (Marr-Hildreth)

Two alternatives:

- Convolve the image with a Gaussian smoothing filter and compute the Laplacian of the result:

$$L(x,y) \otimes G(x,y) \otimes f(x,y)$$

- Convolve the image with the linear filter that is the Laplacian of the Gaussian filter LoG(f):

$$LoG = \nabla^2 G(x,y) = \left[\frac{x^2 + y^2 - 2\sigma^2}{\sigma^4}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}$$

where the Gaussian is given by

$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Since LoG filter is linear, the two constituent filters (Gaussian and Laplacian) can be applied separately, thus allowing us to use different sized filters for each of the actions.

LoG edge detection (Marr-Hildreth) (1st alternative)

LoG edge detection algorithm (1st)

- Filtering.
  - nxn Gaussian filter is used to filter out noise
- Enhancement.
  - Laplacian is used as the enhancement step, i.e., enhancement is done by transforming edges into zero crossings. This is done using a 3x3 mask.
- Detection.
  - Detection is done by finding the zero crossings, i.e., the detection criterion is the presence of a zero crossing in the second derivative with the corresponding large peak in the first derivative.
  - To find a zero crossing it is possible to use 3x3mask that checks sign changes around a pixel.

LoG edge detection (Marr-Hildreth) (2nd alternative)

$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

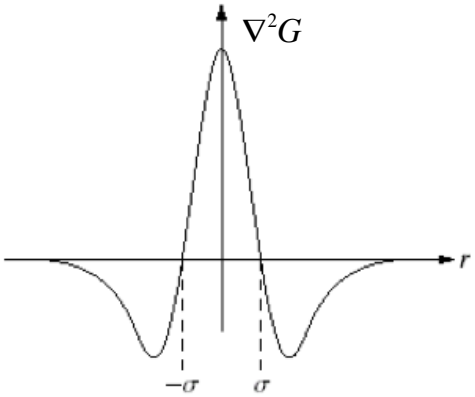
$$LoG = \nabla^2 G(x,y) = \left[\frac{x^2+y^2-2\sigma^2}{\sigma^4}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}$$

LoG edge detection algorithm (2nd):

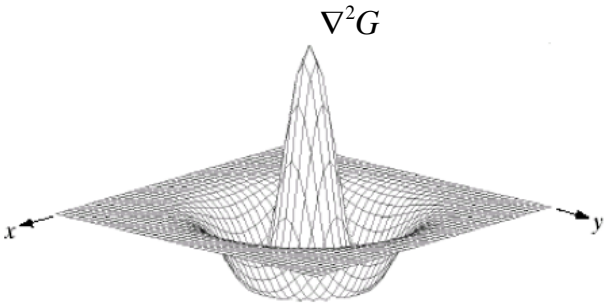
- Filtering and Enhancement (LoG):
- Detection: finding the zero crossings.

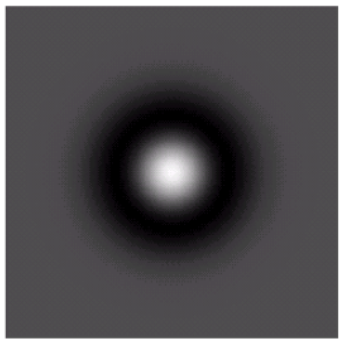
$$\begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}$$

LoG mask = approximation to the shape of $\nabla^2 G$



cross section showing zero crossings





- *black* is negative;
- *gray* is the zero plane
- *white* is positive

Canny edge detection

Two basic objectives:

- *Low error rate*: All edges should be found, with a minimum of spurious responses.
- *Edge points should be well localized:* The distance between one detected point and the true edge point should be minimum.
- *Single edge point response*: Only one point should be detected for each true edge point.

Canny algorithm:

- *Smooth image*: Smooth the input image with a Gaussian filter.
- *Compute gradient*: Compute the gradient magnitude and angles images.
- *Non-maxima suppression*: Apply nonmaxima suppression to the gradient magnitude image.
- *Double thresholding*: Use the double thresholding and connectivity analysis to detect and link edges.

Canny edge detection (1st step): *smooth image*

Smooth image:

- Let f(x,y) denote the image; be $G_\sigma(x,y)$ a Gaussian smoothing filter where σ is the spread of the Gaussian that controls the degree of smoothing. The result of convolution of f(x,y) with $G_\sigma(x,y)$ gives an array of smoothed data as follows:

$$S(x,y) = G_\sigma(x,y) \otimes f(x,y)$$

Output: a smoothed/nonnoisy image S(x,y)

Canny edge detection (2nd step): *compute gradient*

Compute gradient:

- Firstly, the gradient of the smoothed array S(x,y) is used to produce the x and y partial derivatives $G_X(x,y)$ and $G_Y(x,y)$ respectively as follows:

$$G_X(x,y) \approx \left[S(x,y+1) - S(x,y) + S(x+1,y+1) - S(x+1,y)\right]/2$$
$$G_Y(x,y) \approx \left[S(x,y) - S(x+1,y) + S(x,y+1) - S(x+1,y+1)\right]/2$$

  i.e., as a result from averaging the finite differences over the 2x2 square.

- From the standard formulas for rectangular-to-polar conversion, the magnitude and orientation of the gradient can be computed as:

$$G(x,y) = \sqrt{G_X^2(x,y) + G_Y^2(x,y)}$$
$$\theta(x,y) = \tan^{-1}\left(\frac{G_Y^2(x,y)}{G_X^2(x,y)}\right)$$

Output: 2 arrays of the same size as the image:
- *gradient magnitude array* G(x,y)
- *gradient angle array* θ (x,y)

Canny edge detection (3rd step): *non-maxima suppression*

Non-maxima suppression (NMS):

- *Problem*: Edges generated using gradient typically contain wide ridges around local maxima.
- We use non-maxima suppression to thin those ridges, i.e., to find thin edges corresponding to local maxima.

NMS algorithm (for each pixel):

- *Pre-condition*: A 3x3 region centered at every pixel (x,y) defines 4 discrete orientations d_i of the edge normal (gradient vector): horizontal, 45°, vertical, -45°
- Find the direction d_i that is closest to θ (x,y)
- If the value of G(x,y) is less that at least one of its two neighbors along d_i, let N(x,y) =0 (suppression); otherwise, let N(x,y)=G(x,y), where N(x,y) is the non.maxima suppressed image.

Output: I array of the same size as the image:
- *non-maxima suppressed array* N(x,y)

Canny edge detection (4th step): *double thresholding (hysteresis thresholding)*

Double thresholding = false edge point reduction + edge linking

Double thresholding:

- *Problem*: The received image may still contain false edge points.

False edge point reduction algorithm:

- *Input*:
  - Non-maxima suppressed array N(x,y)
  - Low and high thresholds, T_L and T_H (the suggested ratio is 1:3 or 2:3)
- Determine $N_L(x,y)$ after thresholding N(x,y) with T_L
- Determine $N_H(x,y)$ after thresholding N(x,y) with T_H.

  (It is clear that $N_L(x,y)$ contains $N_H(x,y)$)
- $N_L(x,y) = N_L(x,y) - N_H(x,y)$

  (It is clear that $N_L(x,y)$ include the "weak" edge pixels and $N_H(x,y)$ include the "strong" edge pixels

Canny edge detection (cont'd): *double thresholding*

Double thresholding = false edge point reduction + edge linking

Edge linking:

- *Problem*: The edges in $N_H(x,y)$ typically have gaps.
- *Input*:
  - Thresholded array $N_L(x,y)$
  - Thresholded array $N_H(x,y)$
- For each unvisited non-zero pixel p in $N_H(x,y)$
  - Mark as valid pixels all the weak pixels in $N_L(x,y)$ that are connected to p
- Set all the unmarked pixels in $N_L(x,y)$ to 0
- Return $N_H(x,y)+N_L(x,y)$

Canny edge detection: revisited

Algorithm:

- Step 0: Convert to grayscale (for color images)
- Step 1: Noise reduction
- Step 2: Compute gradient magnitude and angle
- Step 3: Non-maximum suppression
- Step 4: Hysteresis thresholding

original image

Step 2: G(x,y) with σ=2.0

Step 3

Step 4: T_L=10 and T_H=50

Summary:

…:

- Image segmentation algorithms: overview
  - Discontinuity-based algorithms (or edge-based algorithms)
  - Similarity-based algorithms (or region-based algorithms)
- Edge detection:
  - Point detection
  - Line detection
  - Edge detection:
    - Basic gradient-based
    - LoG
    - Canny