ELEC 474 – Machine Vision

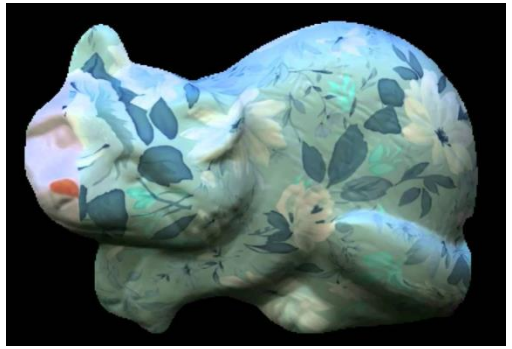
EDGE EXTRACTION

Contents

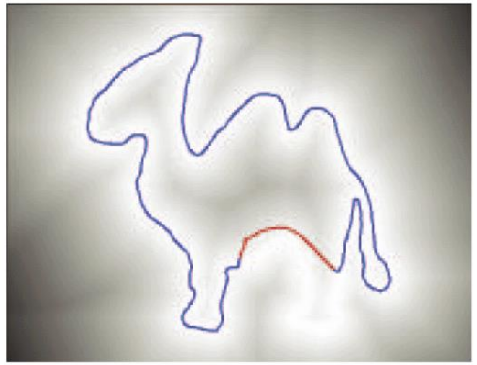
- Introduction
- Edge Detection
- Group Adjacent Pixels
- Canny Edge Detector

Edge detection operators

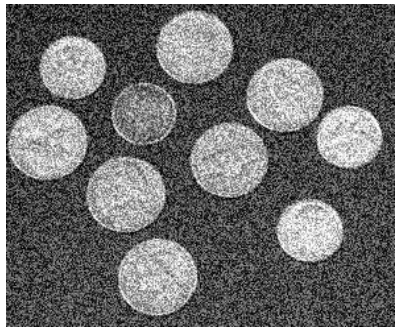
- Edges should be on boundaries, ideally, but
  - Object intrinsic color
  - Scene radiance
  - Occlusions
  - Noise

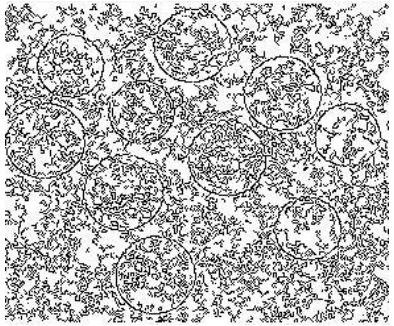


(a)

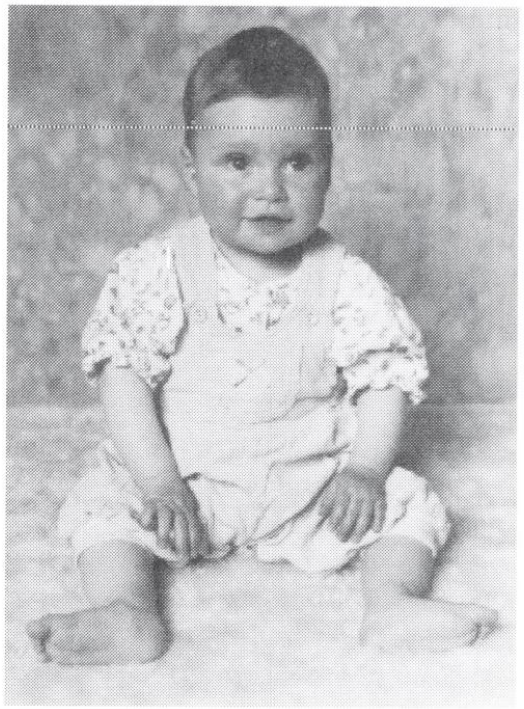


(b)

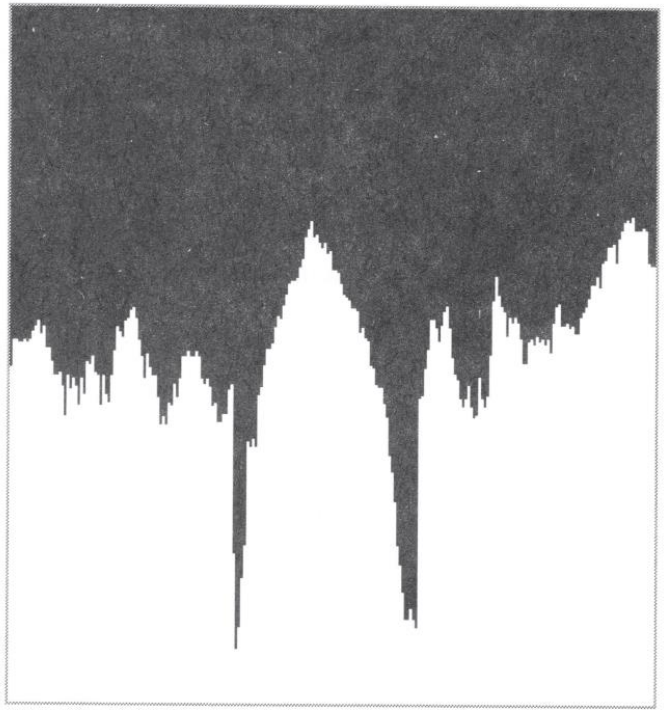




Introduction



(a)



(b)

Figure 4.1 (a) A 325 × 237-pixel image, with scanline $i = 56$ highlighted. (b) The intensity profile along the highlighted scanline. Notice how the main intensity variations indicate the borders of the hair region along the scanline.

Introduction

- The lowest level extraction methods are ***edge*** detection, and ***corner*** detection.
  - **Sharpening filters** identify **regions of high contrast** in an image.
  - These regions can be **single pixels**, or **larger neighborhoods** of adjacent pixels.
  - In edge detection, we wish **to group these adjacent pixels** into continuous curves that **represent the boundaries of objects**.

Edge Detection

- General Procedure
  - Smooth image I to reduce noise
  - Convolve I with orthogonal masks (horizontal $M_x$and vertical M_y) to obtain x-Gradient image G_x and y-Gradient image G_y
  - Estimate the gradient magnitude image as:

$$\mathrm{G(i,j)} = \sqrt{G_x^2(i,j) + G_y^2(i,j)}$$

OR/

$$\mathrm{G(i,j)} \approx |G_x| + |G_y|$$

  - Thresholding: Mark all pixels $I(i,j)$ as edges if $\mathrm{G(i,j)} > \tau$

Edge Detection

If in the above, the following are used, then this is known as Roberts edge detection:

$$M1 = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \quad M2 = \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}$$

Alternately, if the following masks are used, then this is known as Prewitt edge detection:

$$My = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}, \quad Mx = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

Similarly, if the following masks are used, then this is known as Sobel edge detection:

$$My = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}, \quad Mx = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Introduction

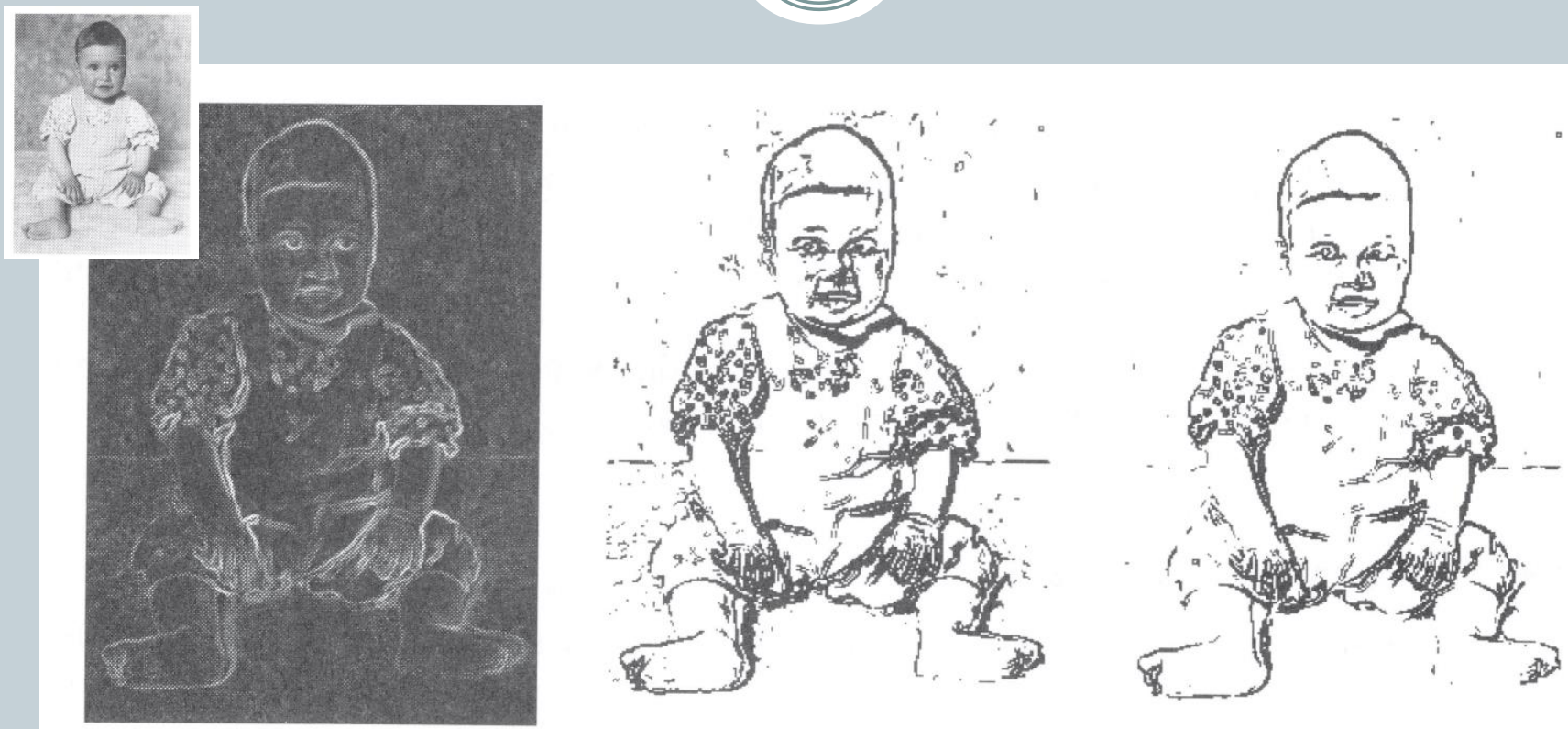


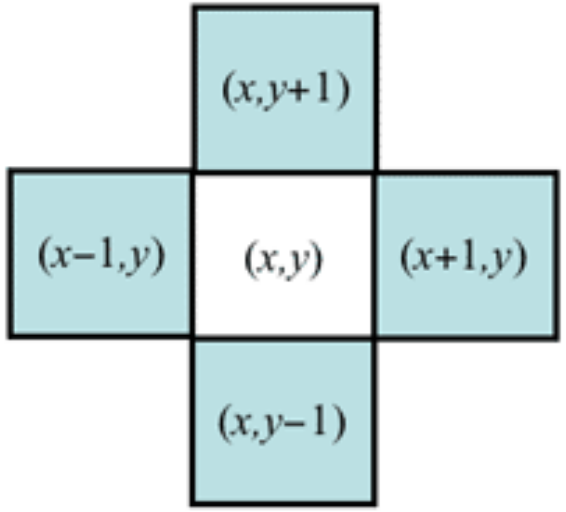
Figure 4.7 Left: output of Sobel edge enhancer run on Figure 4.1. Middle: edges detected by thresholding the enhanced image at 35. Right: same, thresholding at 50. Notice that some contours are thicker than one pixel (compare with Figure 4.5).

Group Adjacent Pixels

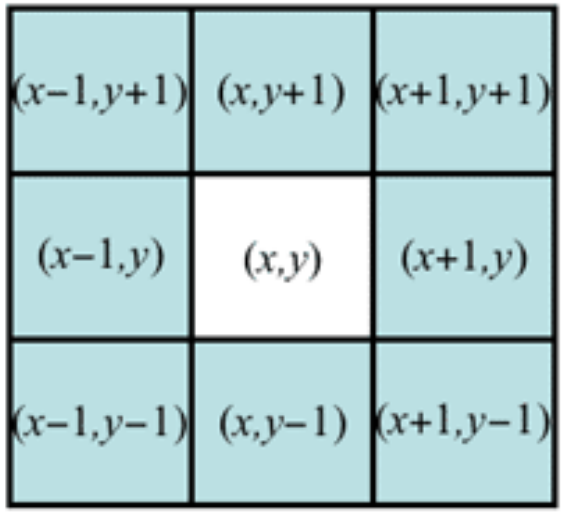
- Thresholding gives binary images
- Pixels on the edges represent 1, otherwise 0.
- How to find individual edge features
  - Group adjacent “ON” pixels
- Connected Components Algorithm
  - 4 connected
  - 8 connected

Group Adjacent Pixels

- 4 Connected Mask
- 8 Connected Mask



4-neighbourhood

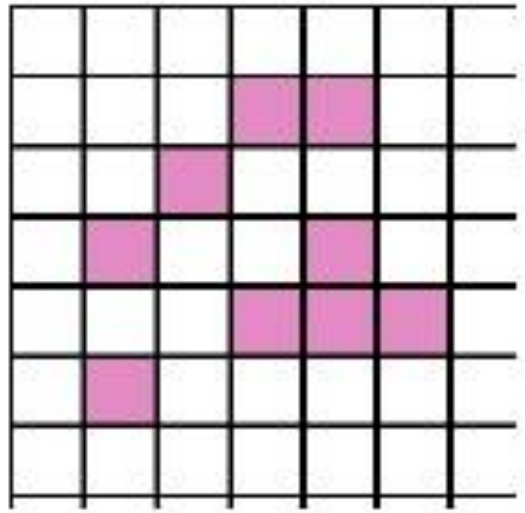


8-neighbourhood

Grouping Adjacent Pixels

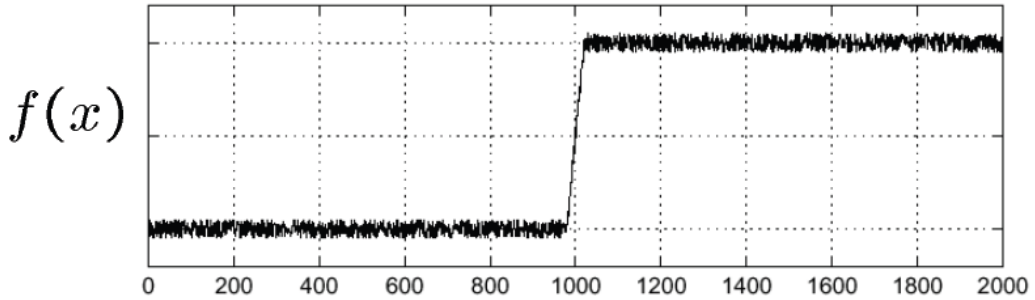
- The Connected Components Algorithm:

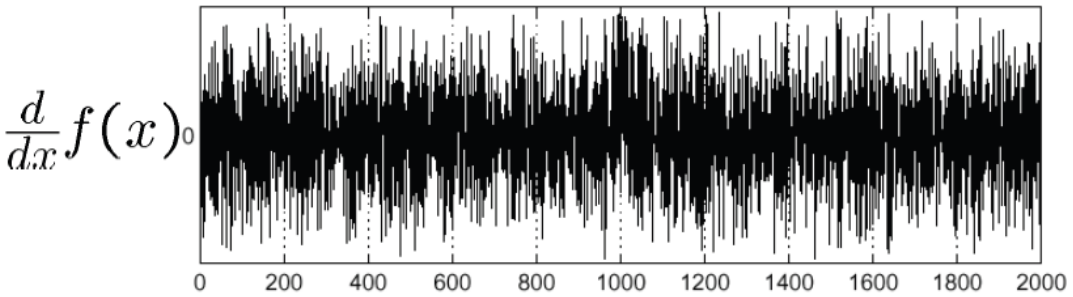
1. Set "curlab" (short for "current label") = 1.
2. Starting from the first pixel in the image, scan the next pixel.
3. If this pixel is a foreground pixel and it is not already labelled, then give it the label "curlab" and add it as the first element in a queue, then go to (4). If it is a background pixel, then repeat (2) for the next pixel in the image.
4. Pop out an element from the queue, and look at its neighbours (based on any type of connectivity). If a neighbour is a foreground pixel and is not already labelled, give it the "curlab" label and add it to the queue. Repeat (4) until there are no more elements in the queue.
5. Go to (2) for the next pixel in the image and increment "curlab" by 1.



Effect of Noise

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





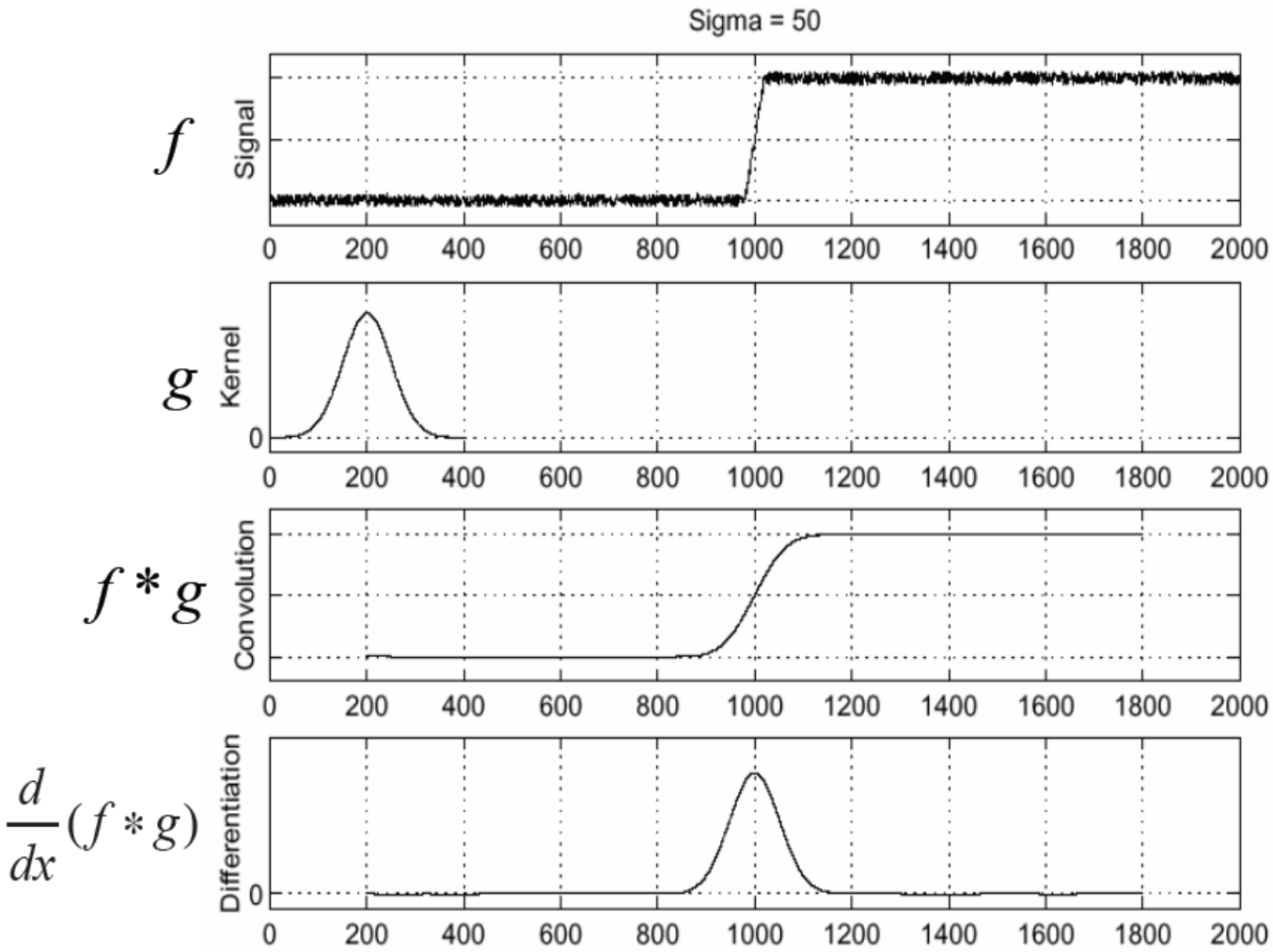
- Where is the Edge?

Effect 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.

Smooth First

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

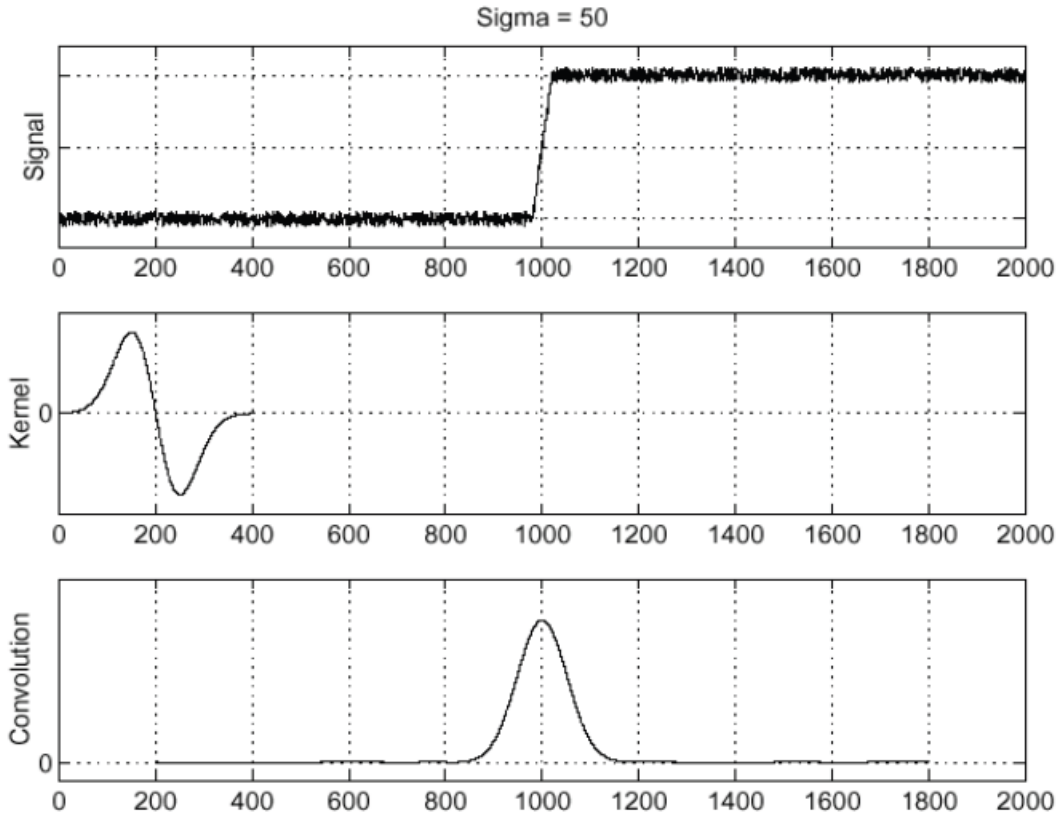


Convolution is Associative

- Differentiate the filter first

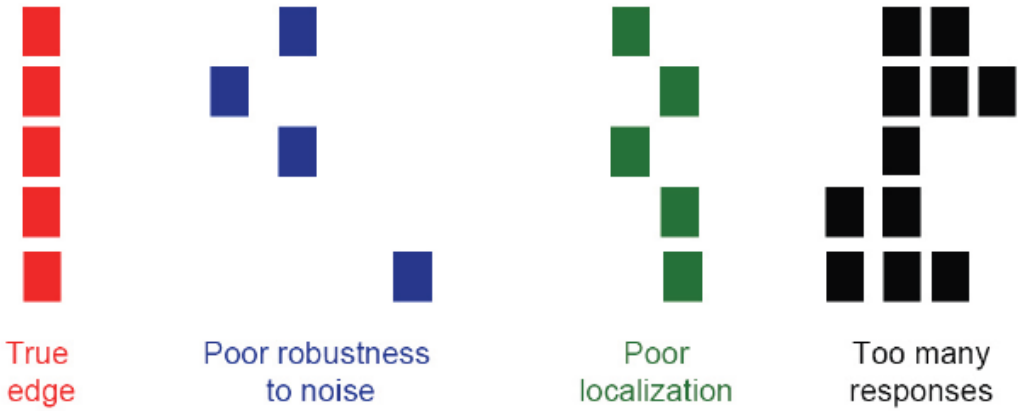
- Because convolution is associative, then:

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



Designing an edge detector

- Three objectives
  - Good Detection
    - There should be a low probability of both false negatives and false positives.
  - Good Localization
    - The detected edge points should be close to the true edge.
  - Single Response
    - In contrast to the Laplacian, each image edge should generate only a single output edge.



Canny Edge Detector

- The most widely used edge detector in computer vision
- Idea:
  - Step-edges are corrupted by additive Gaussian noise.
  - The first derivative of the Gaussian closely approximates the operator that optimizes the product of signal to noise ratio and localization.

Canny Edge Detector

- Algorithm
  1. Smoothing
  2. Differentiation

  } Derivative of Gaussian (DoG)

  3. Non-Maximum Suppression
  4. Hysteresis Thresholding

Canny Edge Detector

1. Convolve the image with a Gaussian mask

$$J = G * I$$

2. Differentiation: compute the gradient

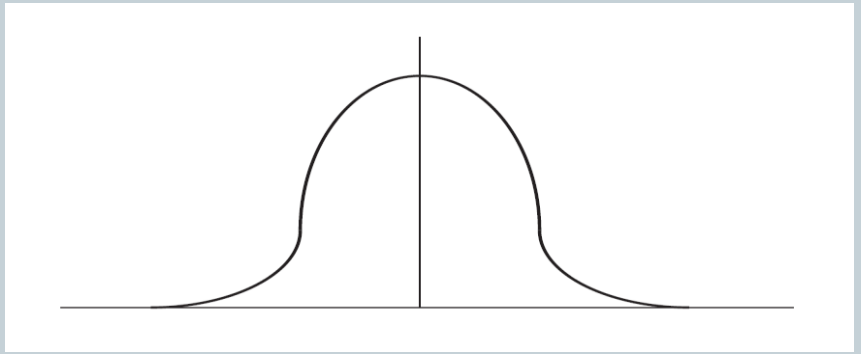
$$J = \nabla(G * I)$$
$$\nabla(G * I) = (\nabla G) * I$$

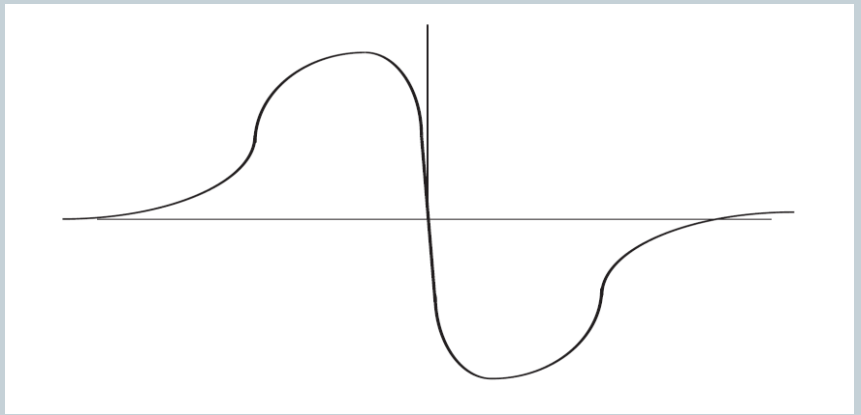
- Magnitude or Edge strength

$$e_s = \sqrt{G_x^2(i,j) + G_y^2(i,j)}$$

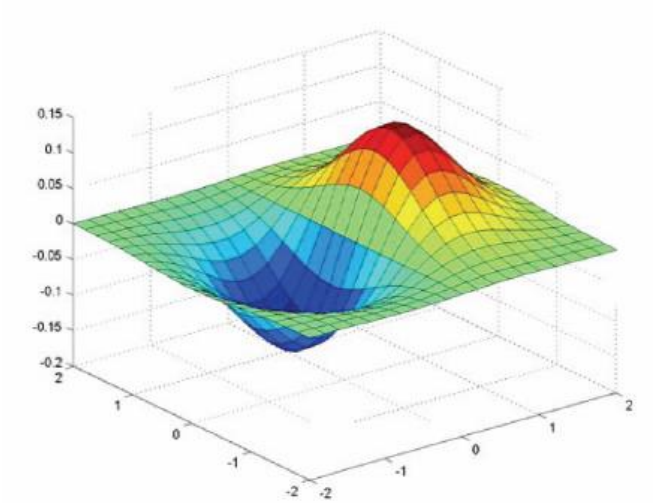
- Direction of edge e_θ

$$e_\theta = \text{atan}\left(\frac{G_y}{G_x}\right)$$

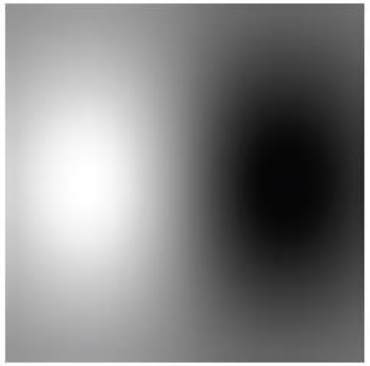


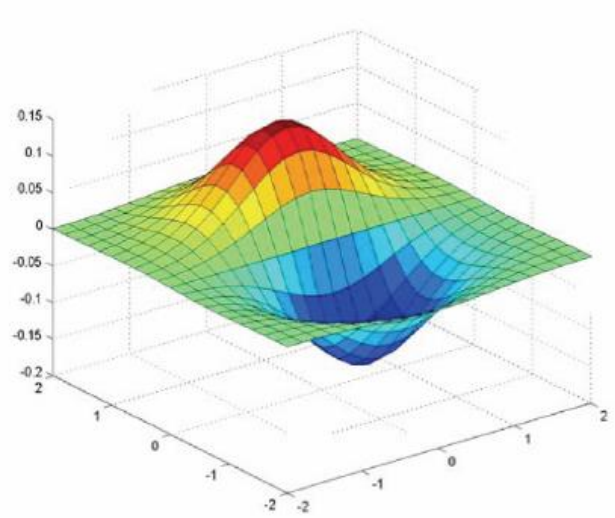


Derivative of Gaussian (DoG) – 2D

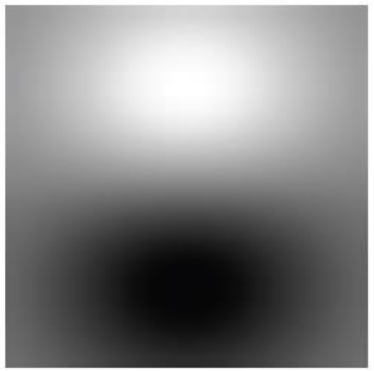


x-direction



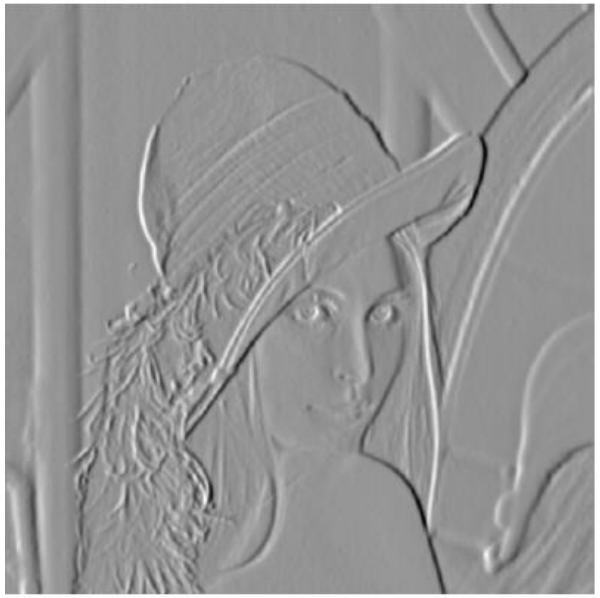


y-direction



Canny Edge Detector

Image Gradient



X-Derivative of Gaussian

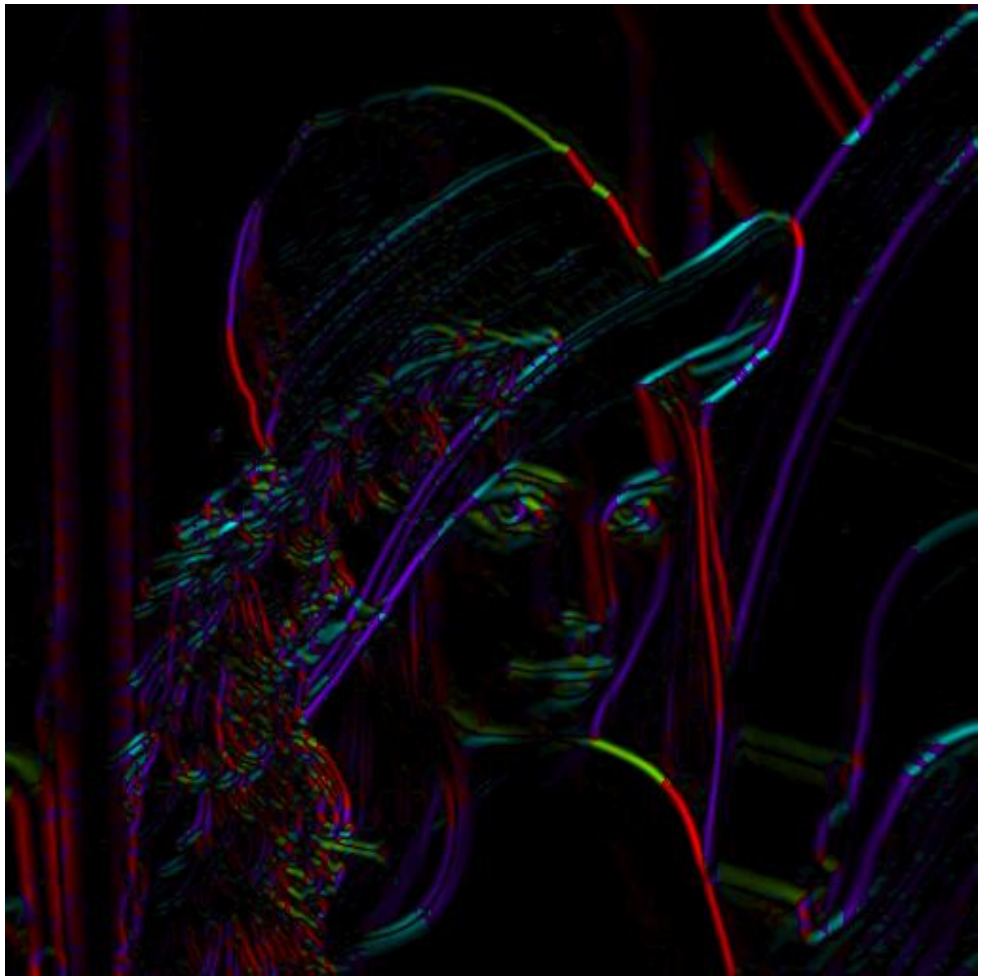
Y-Derivative of Gaussian

Gradient Magnitude

Gradient Orientation

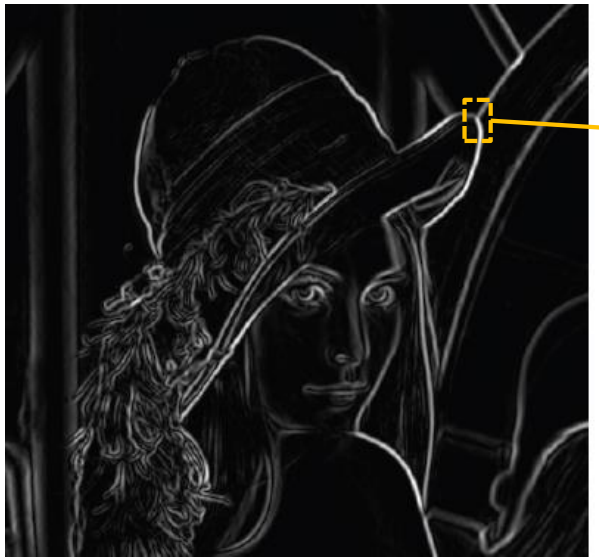
Direction of edge e_θ

$$e_\theta = \text{atan}\left(\frac{G_y}{G_x}\right)$$

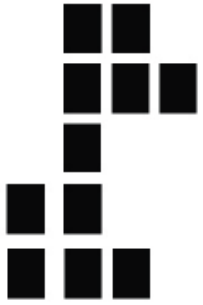


Canny Edge Detector

- Non- Maximum Suppression
  - Remove all the non maximum values from the neighbourhood.
    - For example, Use a 3x3 mask



5	4	3	1	1
9	8	8	2	1
3	2	7	1	1
4	5	8	7	1
3	2	1	7	6



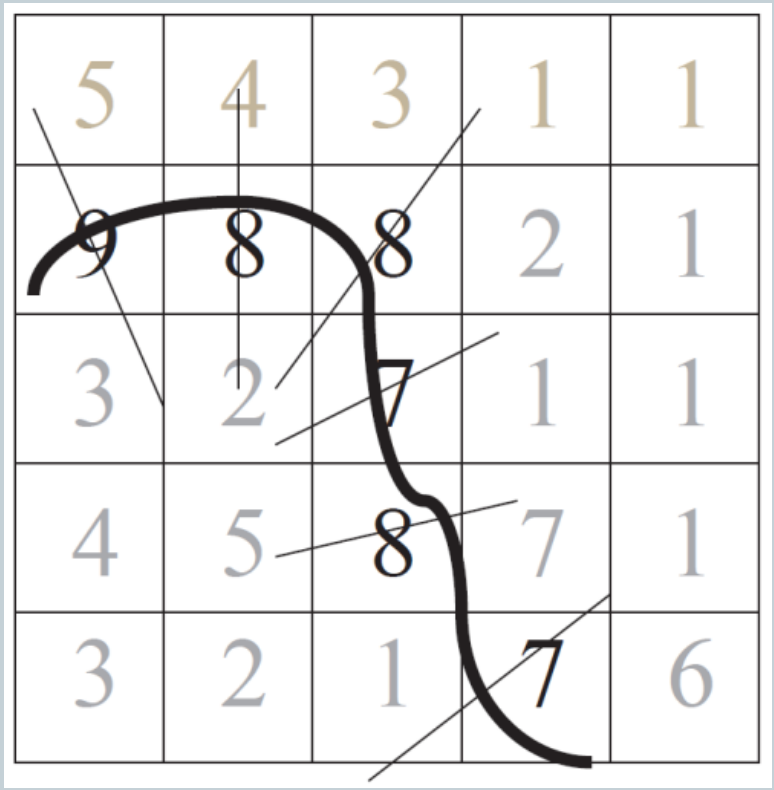
Too many responses

Poor localization

0	0	0	0	0
9	0	8	0	0
0	0	0	0	0
0	0	8	0	0
0	0	0	0	0

Canny Edge Detector

- Non- Maximum Suppression
  - consider the neighboring values of a pixel only in the direction that is perpendicular to the edge direction.
  - the 2-D problem is reduced to a series of simpler 1-D problems.



Canny Edge Detector

Before
non-max suppression

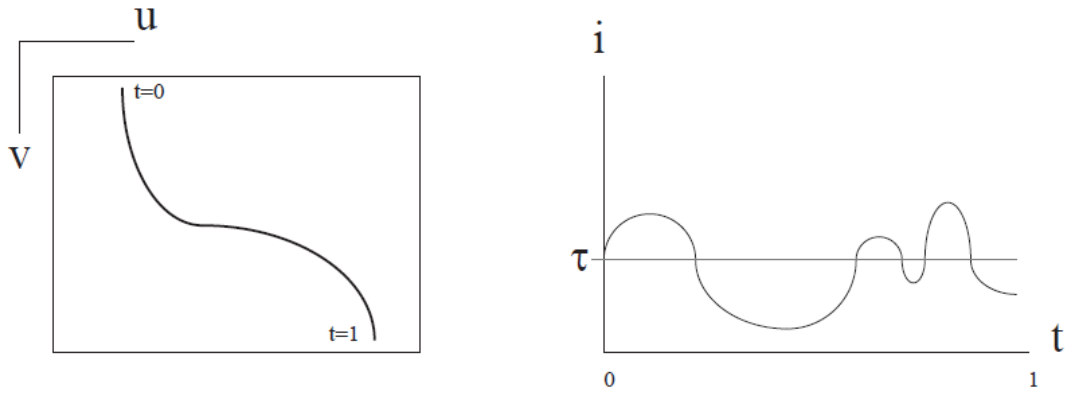
After
non-max suppression

Canny Edge Detector

- Thresholding
  - Set all pixels to zero whose values are less than some threshold τ

What are the disadvantages ?

  - Difficult to set the value of τ.
  - Streaking can occur with an edge oscillates above and below τ .



Canny Edge Detector

- Hysterisis Thresholding
  1. The filtered edge strength image is scanned until a pixel value is found greater than a high threshold value τ_H.
  2. This pixel is set to "on" as an edge pixel. The edge is tracked outward from this pixel.
  3. All connected pixels with values greater than a low threshold τ_L are also set to "on".
  4. Repeat until all pixels are accessed.
- Ratio of $\frac{\tau_H}{\tau_L} \approx 2$ or 3

Canny Edge Detector

Original

Final Canny Edges

Canny Edge Detector

- Summary of Canny Edge Detection
  1. Smoothing
  2. Differentiation
  3. Non-Maximum Suppression
  4. Hysteresis Thresholding
- Canny Edge Detection is optimal under certain assumptions.
  - Step edges
  - Gaussian noise

Resources

- J. Canny, “A Computational Approach To Edge Detection”, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8:679--714, 1986.