EECS332 Digital Image Analysis

Edge Detection

Ying Wu

Electrical Engineering and Computer Science
Northwestern University
Evanston, IL 60208

http://www.ece.northwestern.edu/~yingwu
yingwu@ece.northwestern.edu

What we've learnt

- Binary image analysis
- Basic image enhancement
- Image segmentation
  - Color-based segmentation
  - Region-based segmentation
  - Watershed segmentation

- So, what should we learn next?
- Rough → precise

Why Precise ?

- Since we need
  - the precise localization of a target
    - ✓ 2D
    - ✓ 3D
  - the precise shape of a target
    - ✓ 2D
    - ✓ 3D
- Thus we need accurate visual features
- What are they?

You may want to know …

- A picture is worth 1000 words
- An edge map preserves about arguably 90% information of an image

Outline

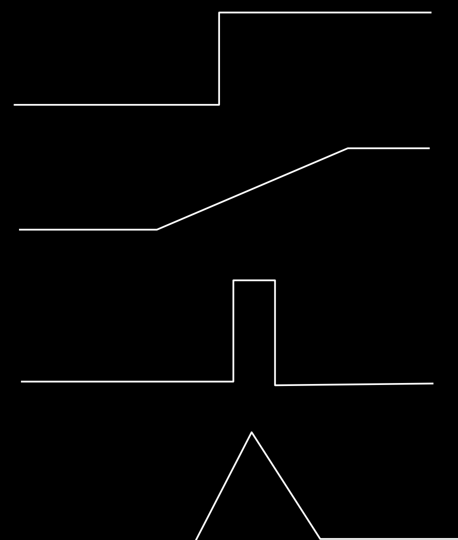
- Motivation
- What is image edge?
- Image gradient
  - DoG
  - Simple edge detectors
- Second order derivative methods
  - Zero-crossing
  - LoG
- Canny detector

Image Edge

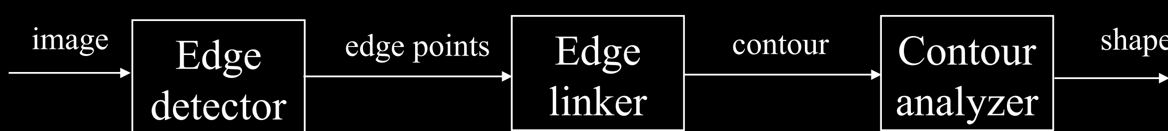
- What is an edge?
  - A significant local change in the image intensities
- How is an edge produced?
  - Discontinuity in image intensities
    - ✓ depth discontinuity
    - ✓ material discontinuity
    - ✓ shading discontinuity
    - ✓ color discontinuity
  - Discontinuity in the 1st order derivative of intensity

Edge

- Various types of edges
  - Step edge
  - Ramp edge
  - Line edge
  - Roof edge



Edge → Shape



- Edge point (or edge)
  - an image point where significant local changes present
- Edge fragment
- Edge detector
  - an algorithm that finds edge points
- Contour
  - a list of edge ← freeform
  - a curve that models the list of edge ← parametric form
- Edge linking
  - a process of forming an ordered list of edges
- Edge following
  - a process of searching the image to determine contours

Gradient

- The gradient is a measure of the change of a function

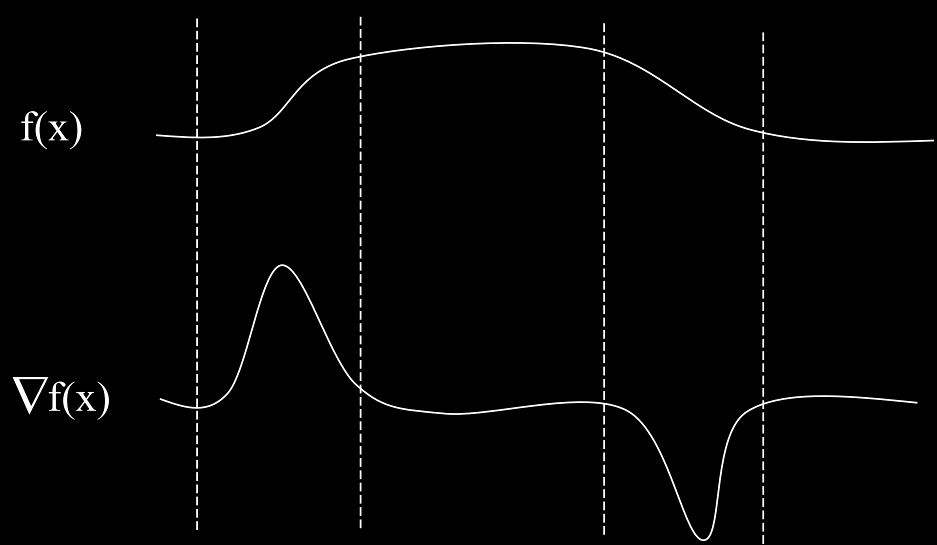


Image Gradient

- Image gradient measures the intensity changes
- Definition

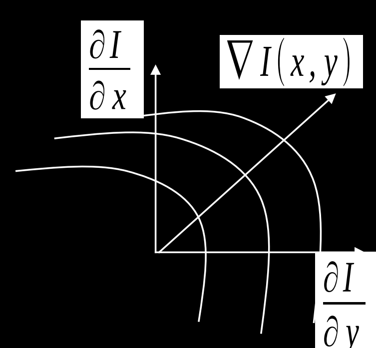
$$\nabla I(x,y) = \begin{bmatrix} \frac{\partial I(x,y)}{\partial x} \\ \frac{\partial I(x,y)}{\partial y} \end{bmatrix}$$

- Properties:
  - $\nabla I(x,y)$ points to the direction of the max rate of increase of the image I(x,y)
  - Its magnitude

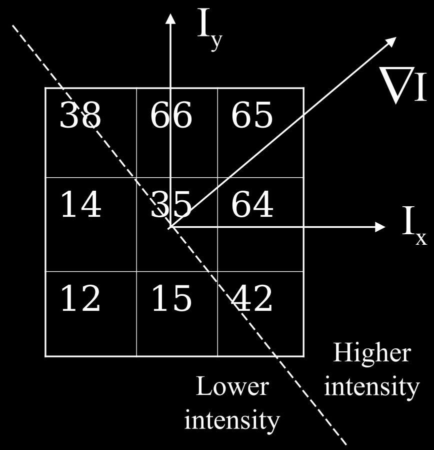
$$\|\nabla I(x,y)\| = \sqrt{\left(\frac{\partial I(x,y)}{\partial x}\right)^2 + \left(\frac{\partial I(x,y)}{\partial y}\right)^2}$$

  - Its direction

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



An Example



$I_y = (38-12) + 2(66-15) + (65-42) = 141$

$I_x = (65-38) + 2(64-14) + (42-12) = 157$

$\theta = \tan^{-1}(141/157) = 42^{o}$

$|\nabla I| = \mathrm{sqrt}(141^2+157^2) = 211$

Convolution kernel

1	2	1
0	0	0
-1	-2	-1

G_y

-1	0	1
-2	0	2
-1	0	1

G_x

Simple detectors

- Robert Cross operators

1	0
0	-1

G_x

0	-1
1	0

G_y

- Sobel operators

-1	0	1
-2	0	2
-1	0	1

G_x

1	2	1
0	0	0
-1	-2	-1

G_y

- Prewitt operators

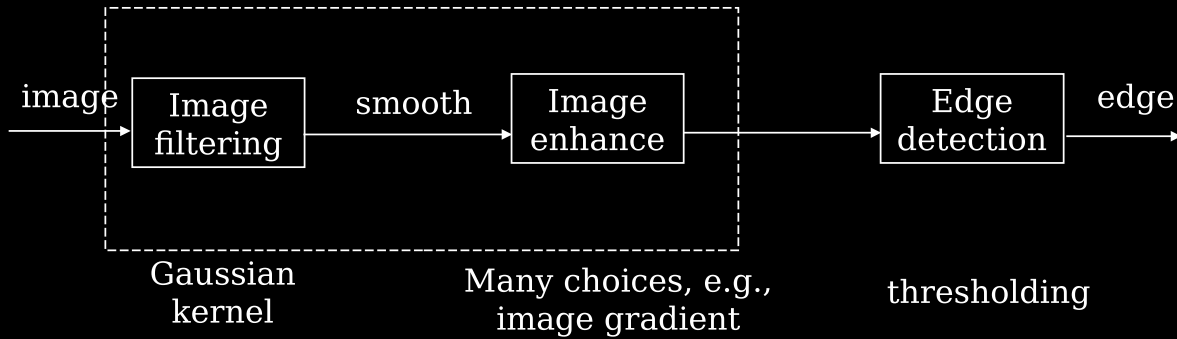
-1	0	1
-1	0	1
-1	0	1

G_x

1	1	1
0	0	0
-1	-1	-1

G_y

Steps in edge detection



- Filtering → get rid of image noise
- Enhancement → get salient information
- Detection → extracting strong edge contents

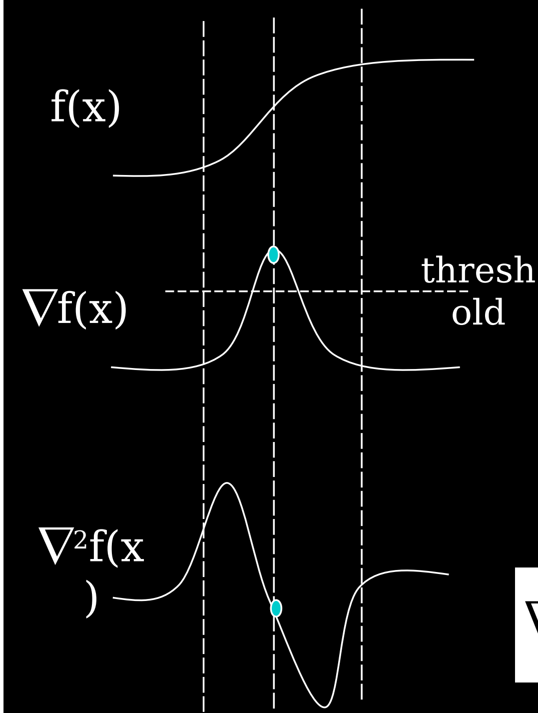
DoG

- Filter out noise before edge enhancement
- Derivative of Gaussian
  - Filtering ← Gaussian kernel
  - Enhance ← 1st order derivative
  - Detection ← thresholding

$$h(x,y)=\nabla\left[g(x,y)\otimes I(x,y)\right]=\left[\nabla g(x,y)\right]\otimes I(x,y)$$

$$\nabla g(x,y)=\begin{bmatrix} -\dfrac{x}{\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}} \\ -\dfrac{y}{\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}} \end{bmatrix}$$

∇ vs. ∇^2



A threshold applies to the gradient to find the edges.
→ thick edge

Is it good enough?

What we need is the point that has a local maximum gradient!

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

Laplacian operator

Approximation

$$\frac{\partial^2 I(x,y)}{\partial y^2} = \frac{\partial I_y}{\partial y} = \frac{\partial (I(x,y+1) - I(x,y))}{\partial y}$$
$$= \frac{\partial I(x,y+1)}{\partial y} - \frac{\partial I(x,y)}{\partial y}$$
$$= [I(x,y+2) - I(x,y+1)] - [I(x,y+1) - I(x,y)]$$
$$= I(x,y+2) - 2I(x,y+1) + I(x,y)$$
$$\xrightarrow{recenter} \quad I(x,y+1) - 2I(x,y) + I(x,y-1)$$

$$\frac{\partial^2 I(x,y)}{\partial x^2} = \frac{\partial I_x}{\partial x}$$
$$= I(x+1,y) - 2I(x,y) + I(x-1,y)$$

0	0	0
1	-2	1
0	0	0

+

0	1	0
0	-2	0
0	1	0

⇩

0	1	0
1	-4	1
0	1	0

Find zero-crossing

Ideal case

2	2	5	8	8
2	2	5	8	8
2	2	5	8	8

I(x,y)

	3	0	3	
	3	0	3	
	3	0	3	

$\nabla^2 I(x,y)$

Non-Ideal case

2	2	2	8	8
2	2	2	8	8
2	2	2	8	8

I(x,y)

	0	6	-6	
	0	6	-6	
	0	6	-6	

$\nabla^2 I(x,y)$

? Where is zerocrossi ng

How to solve this problem?

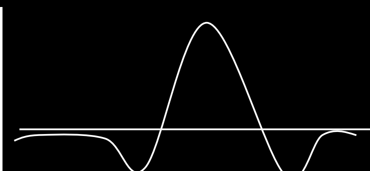
LoG

- 2nd order derivative is sensitive to noise
- Solution?
  - Filter out noise before edge enhancement
- Laplacian of Gaussian
  - Filtering ← Gaussian kernel
  - Enhance ← 2nd order derivative
  - Detection ← zero-crossing

$$h(x,y)=\nabla^2\left[g(x,y)\otimes I(x,y)\right]=\left[\nabla^2 g(x,y)\right]\otimes I(x,y)$$

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

Mexican hat



Example

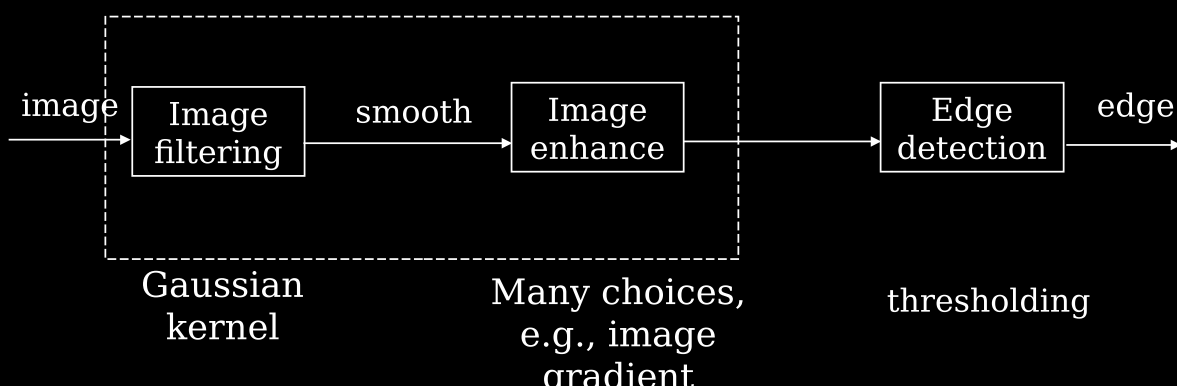
Lena

Robert cross

Sobel

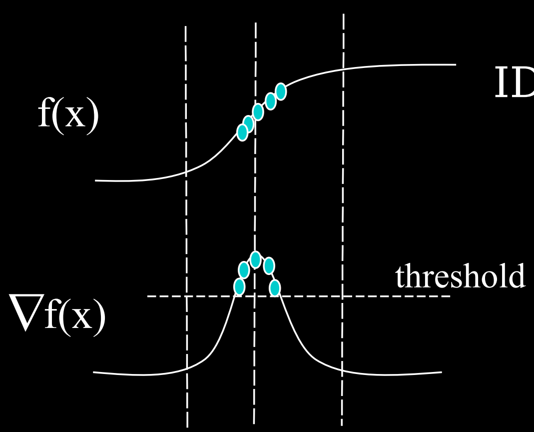
Zero-crossing

Steps in edge detection



- Filtering → get rid of image noise
- Enhancement → get salient information
- Detection → extracting strong edge contents

A review: idea I

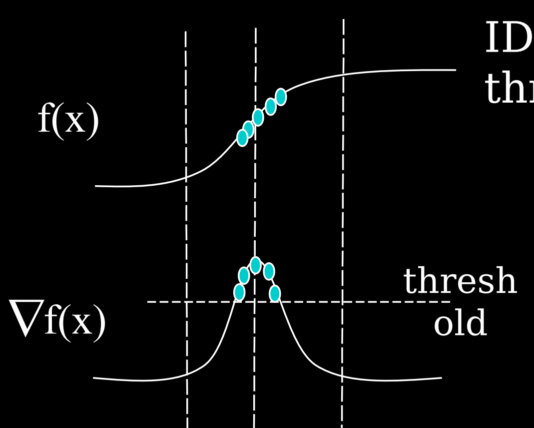


IDEA_1: ∇ + thresholding

Problems:

- How to find the best threshold?
- Thin vs. thick edge?
- Noise?

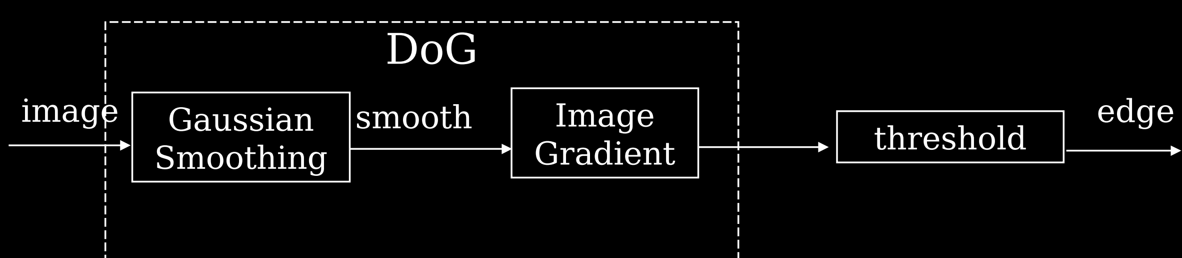
A review: idea II



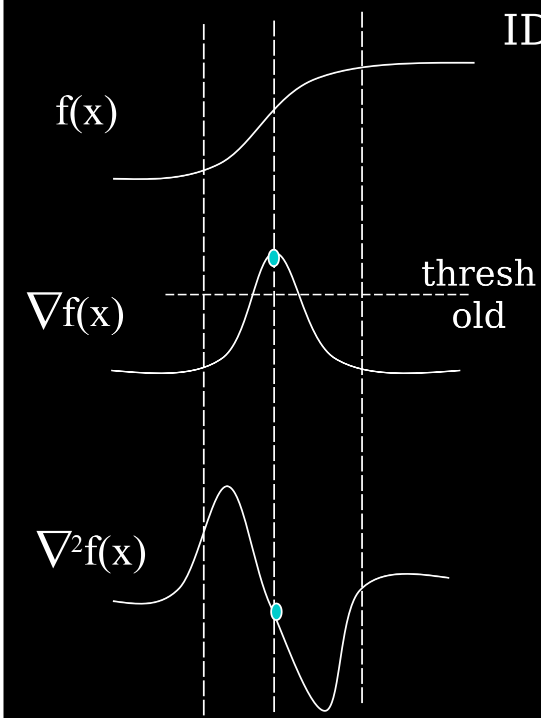
IDEA_2: DoG (∇ of Gaussian + threshold)

Problems:

- How to find the best threshold?
  - ✓ thin vs. thick edge?
- ~~Noise?~~



A review: idea III



IDEA_3: ∇^2 + zero-crossing

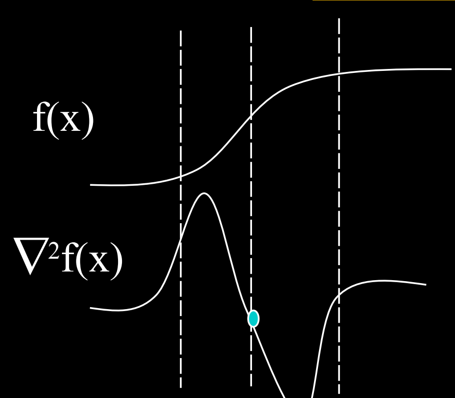
Old problems:

- ~~How to find the best threshold?~~
  - ✓ ~~thin vs. thick edge?~~
- Noise?

New problems:

- Computationally more intensive
  - ✓ to find zero-crossing
- Very sensitive to noise

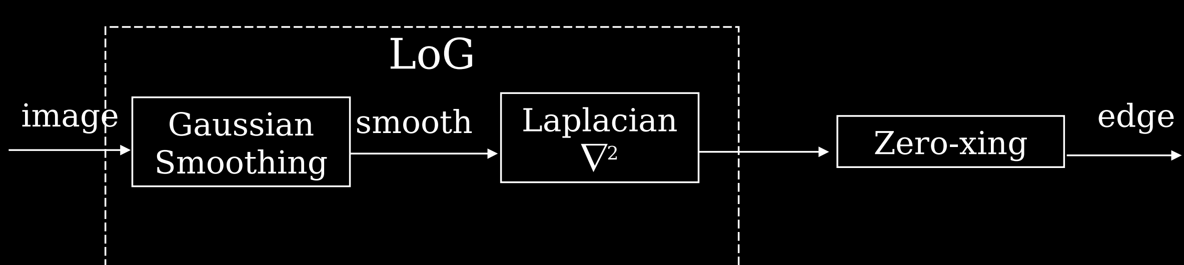
A review: idea IV



IDEA_4: LoG (∇^2G + zero-crossing)

New problems:

- Computationally more intensive
  - ✓ to find zero-crossing
- Very sensitive to noise



Research starts from here …

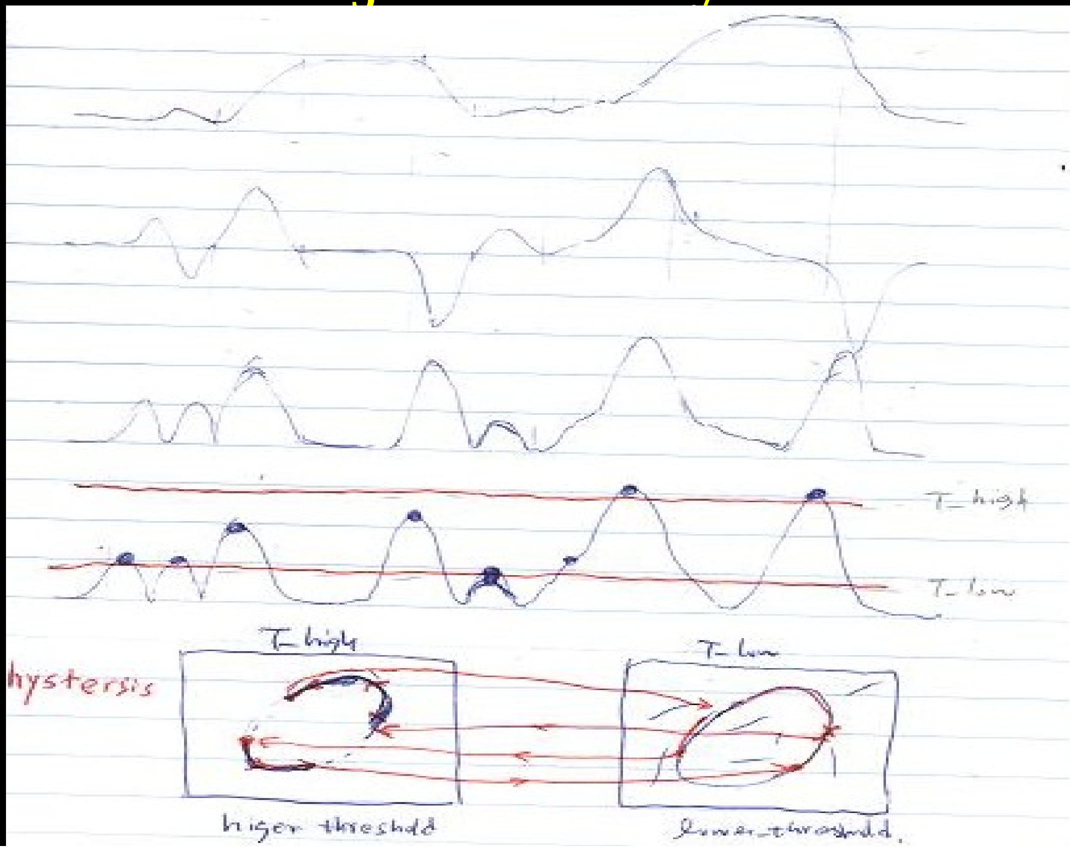
- Before you have a new idea, you need to:
  - Have a deep understanding of the existing work
  - Analyze their pros and cons
    - ✓ theoretical analysis
    - ✓ or just try
  - Distinguish “great”, “very good” and “good”
    - ✓ what you should keep or discard
  - Question their assumptions
    - ✓ break our common senses
    - ✓ find what they may have overlooked

These needs experiences

So …

- What are very good/good in the previous methods?
  - ✓ ∇^2 seems not so good, due to zero-xing
  - ✓ ∇ seems to be a better one: simple and less sensitive to noise
- What does it assume?
  - ✓ thresholding
    - → why thresholding? Why not finding the peaks directly?
  - ✓ using ONE threshold
    - → You may think: why ONE, but not TWO or more?
- What may it have overlooked?
  - ✓ NO neighborhoods used (i.e., pixel by pixel)!
    - → why don’t we use it?

John Canny



Overview

1. Gaussian smoothing
2. Calculating image gradient
3. Suppressing non-maxima
4. Finding two thresholds
5. Edge linking

1. Gaussian Smoothing

$$S(x,y) = G(x,y) \otimes I(x,y)$$

```
function S = GaussSmoothing(I, N, sigma)
% N = 3;
% Sigma = 3;
Gmask = fspecial(‘gaussian’, [N,N], sigma);
S = conv2(I, Gmask, ‘same’);
```

Lena

Lena original

Gaussian smoothing

2. Image Gradient

- Whatever.
- as long as you can find:
  - M(x,y) ← magnitudes of ∇S(x,y)
  - T(x,y) ← direction of ∇S(x,y)

Lena

Gaussian smoothing

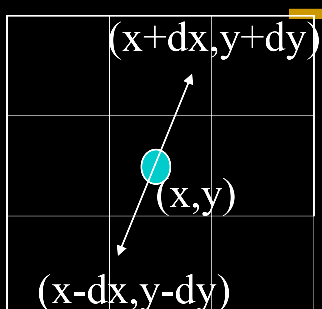
Robert-cross

3. Suppressing non-Maxima

- Only find local maxima (i.e., the “ridges”)
  - ✓ what do you experience when you walking along a ridge?
  - ✓ Oh, yes. The stuff on my both “sides” are lower.
- So, two things:
  - The direction of the “ridge”
  - The “sides” of the ridge
- You’ve got the idea now …
  - find the two side pixels
  - identify non-maxima
  - set them to 0

35	10	10	0	0	0
35	35	10	10	0	0
20	35	35	10	10	0
0	20	35	35	10	10

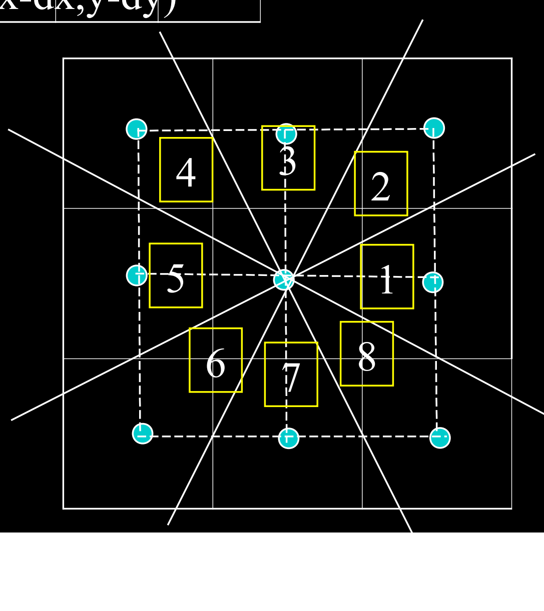
Method I: LUT



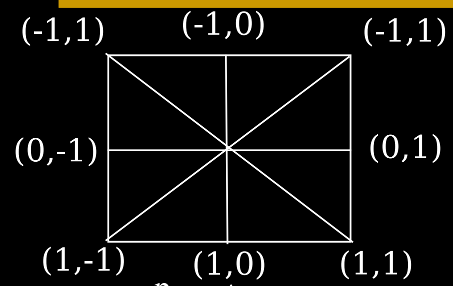
Digital thinking ← recall our first lecture

→ how many possible combinations??

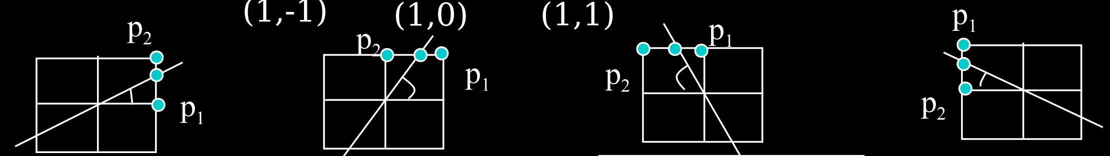
	dx	dy
1	0	1
2	-1	1
3	-1	0
4	-1	-1
5	0	-1
6	1	-1
7	1	0
8	1	1



Method II: interpolation



$$I(p) = \alpha I(p_1) + (1-\alpha) I(p_2)$$



$p_1 = [0,1] \quad p_2 = [-1,1]$
$\alpha = 1 - |tg\theta|$

$p_1 = [-1,1] \quad p_2 = [-1,0]$
$\alpha = |\frac{1}{tg\theta}|$

$p_1 = [-1,0] \quad p_2 = [-1,1]$
$\alpha = 1 - |\frac{1}{tg\theta}|$

$p_1 = [-1,1] \quad p_2 = [0,-1]$
$\alpha = |tg\theta|$

$p_1 = [0,-1] \quad p_2 = [1,-1]$
$\alpha = 1 - |tg\theta|$

$p_1 = [1,-1] \quad p_2 = [1,0]$
$\alpha = |\frac{1}{tg\theta}|$

$p_1 = [1,0] \quad p_2 = [1,1]$
$\alpha = 1 - |\frac{1}{tg\theta}|$

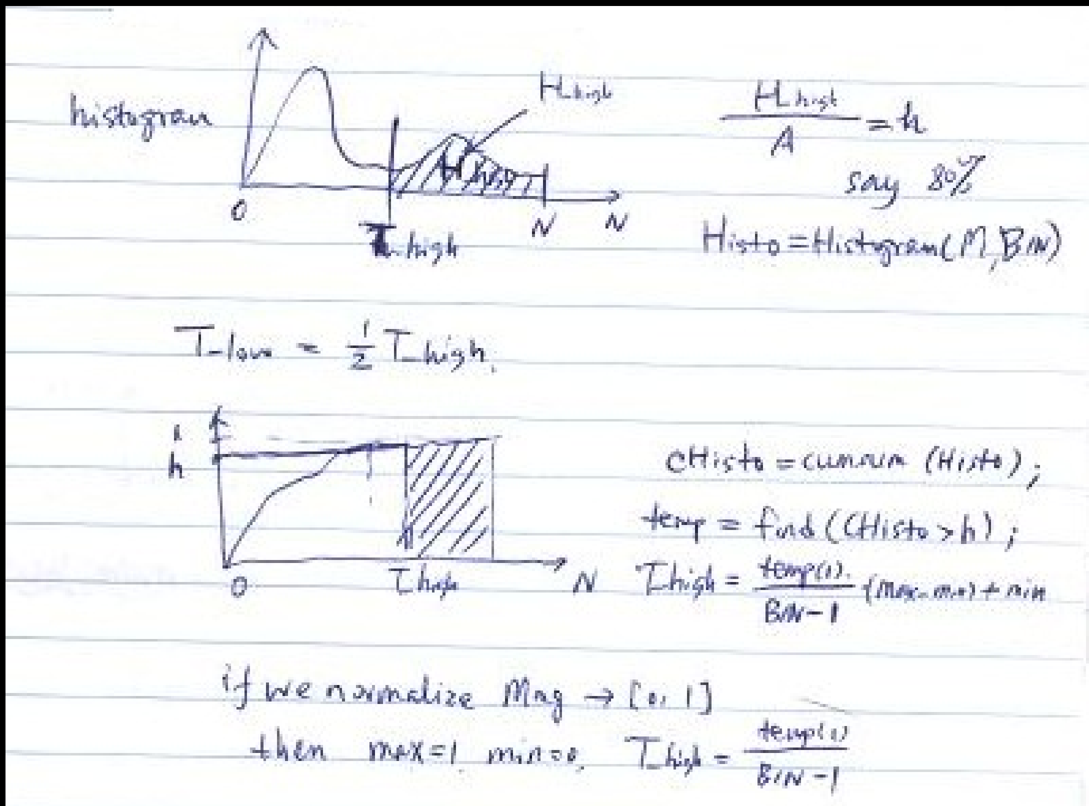
$p_1 = [1,1] \quad p_2 = [0,1]$
$\alpha = |tg\theta|$

Lena

Robert-cross

Non-maxima suppressing

4. Two thresholds



Lena

Applying T_low

Applying T_high

5.Edge Linking

- Pick a starting point
- Recursively check its 8-neighbor in strong edges

```
if the neighbor is an endpoint
  return;
else
  continue recursion;
```

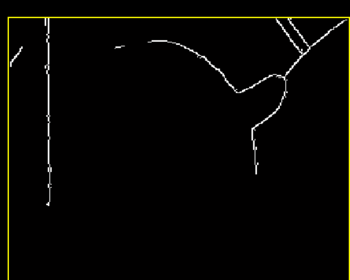
- For the end point, check weak edges, until the edge ends or it connects to a strong edge

```
if the neighbor is an endpoint, or a strong edge
return;
else
continue recursion;
```

Note: you may end up with a stack overflow!

`set(0, 'RecursionLimit', 2000);` in matlab

Lena



Stage 1

Stage 2

Stage 3

Stage 4

Final

Final Canny

Sobel