Introduction

TA: Sahar Sajadieh

Email: sahar@umail.ucsb.edu

or Class Email: cs181b@cs.ucsb.edu

Office Hours: Wednesday 10AM-12PM------------------------CSIL

Thursday 9AM-11AM-------------------------------CSIL

Edge Detection: Marr–Hildreth Algorithm

- 1. Convert RGB picture to Gray Scale
- 2. Smoothing: using Gaussian filter
- 3. Second order derivative: using Laplacian filter
- 4. Zero crossing

1. Convert RGB picture to Gray Scale

- imgGry=Rgb2gray(img)

How to apply a filter on an image

- Use Convolution Operation
- **Convolution notations:** $R = H * F = H \otimes F$
- outputImg = conv2(H, F); %H is the filter, and F is the image

  figure;

  imshow(outputImg);

Convolution

$$R_{ij} = \sum_{u,v} H_{i-u,j-v} \, F_{uv}$$

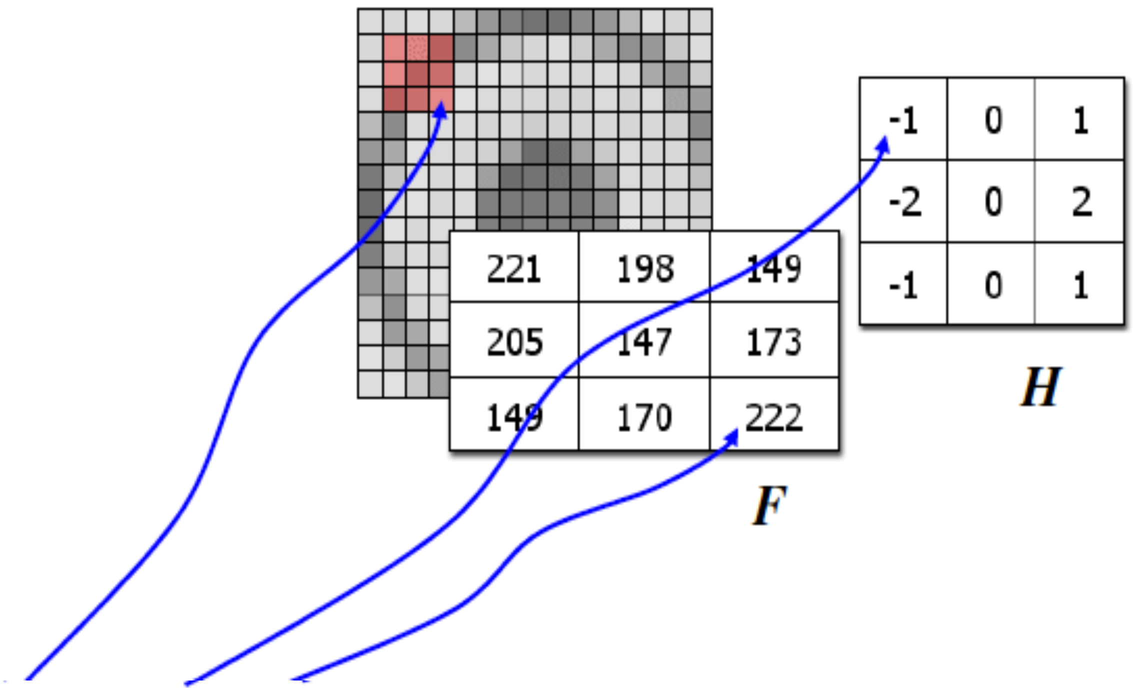
Output image

Convolution filter kernel

Input image

- For every pixel (i,j):
  - Line up the image at (i,j) with the filter kernel
  - Flip the kernel in both directions (vertical and horizontal)
  - Multiply and sum (dot product) to get output value *R*(*i,j*)

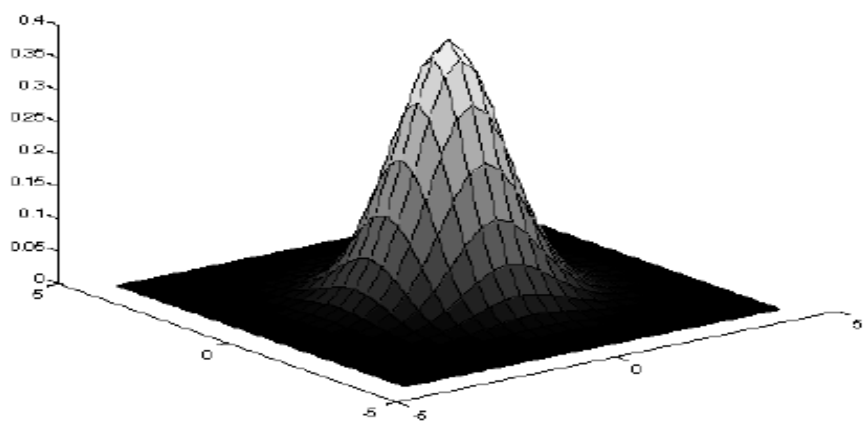
Convolution



$$R_{4,4} = H_{0,0}F_{4,4} + H_{0,1}F_{4,3} + H_{0,2}F_{4,2} + H_{1,0}F_{3,4} + H_{1,1}F_{3,3} + H_{1,2}F_{3,2} + H_{2,0}F_{2,4} + H_{2,1}F_{2,3} + H_{2,2}F_{2,2}$$

2. Gaussian Filter

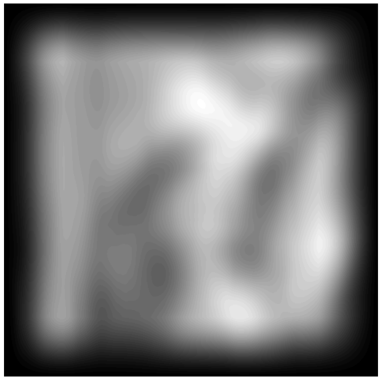
- What is Gaussian?
  - A Low Pass Filter: to get rid of high-frequency noise
  - Smoothing by Averaging
  - Rotationally symmetric
  - Weights of nearby pixels are more than distant ones



2. Gaussian Filter

Sigma=3

Sigma=12



Sigma=48

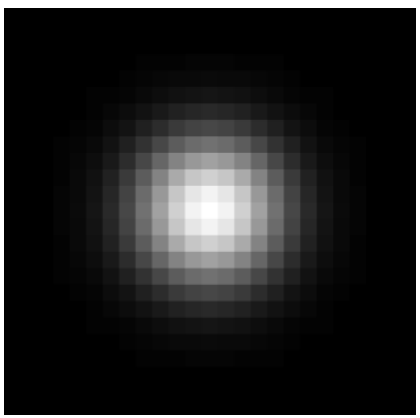
2. Gaussian Filter

- How does a Gaussian Kernel looks like?

2. Gaussian Filter

- How does a Gaussian Kernel looks like?

The filter looks something like this:



$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

2. Gaussian Filter

- Formula:

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp \frac{-(x^2+y^2)}{\sigma^2}$$

- In Matlab: 1<= x, y<=n (n is the dimension of the filter)
- What is n?
- Don't forget to normalize the filter

2. Gaussian Filter

- Please Note: Even though the Gaussian equation in the previous page is correct, to do this HW in Matlab, you need to use a modified version of that equation:

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp \frac{-(i^2+j^2)}{\sigma^2}$$

- i=x-1-floor(n/2), j=y-1-floor(n/2)
- x, y are the indices of the Gaussian Kernel matrix in Matlab→ 1<= x, y<=n
- -floor(n/2)<i, j<+floor(n/2) (so that the Gaussian filter is centered at zero, with the max value at i, j=0)

2. Gaussian Filter

- What is n?

2. Gaussian Filter

- What is n?

n is based on your choice, but:

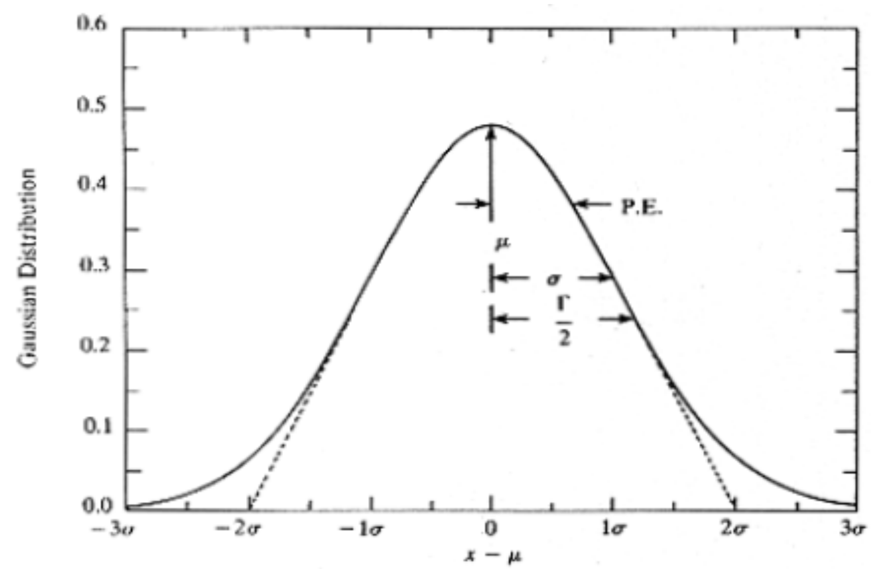
2. Gaussian Filter

- What is n?

n is based on your choice, but:

- Should contain at least one sigma (in each side)
- Should be an odd number, since the Gaussian filter is symmetric and we want to have a center point to do the averaging for.
- For example n= (6*sigma)+1

(With 6*sigma, we can get most of the Gaussian energy inside of the window.)



2. Gaussian Filter

- How to normalize the filter?

2. Gaussian Filter

- How to normalize the filter?

$$\frac{1}{16}\boxed{\begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}}$$

3. Laplacian Filter

- 2nd order Edge detection filter:

∇^2 is the **Laplacian** operator:

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

Laplacian is the sum of second partial derivatives of the function.

3. Laplacian Filter

- What is a gradient of an image:

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

- How do you approximate it (in a discrete space/grid with finite elements): "Discrete Gradient"

$$\frac{\partial f}{\partial x}[x, y] \approx f[x+1, y] - f[x, y]$$

3. Laplacian Filter

- What is a Laplacian of an image:

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

- How do you approximate it (in a discrete space/grid with finite elements):

"Discrete Laplacian"

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

3. Laplacian Filter

- How to make a Matrix:

A= [1 2 3; 4 5 6; 7 8 9]

- You can use either of these filters:

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

-1	-1	-1
-1	8	-1
-1	-1	-1

1	-2	1
-2	4	-2
1	-2	1

Most Commonly-used Laplacian kernel

LoG filter (Mexican Hat)

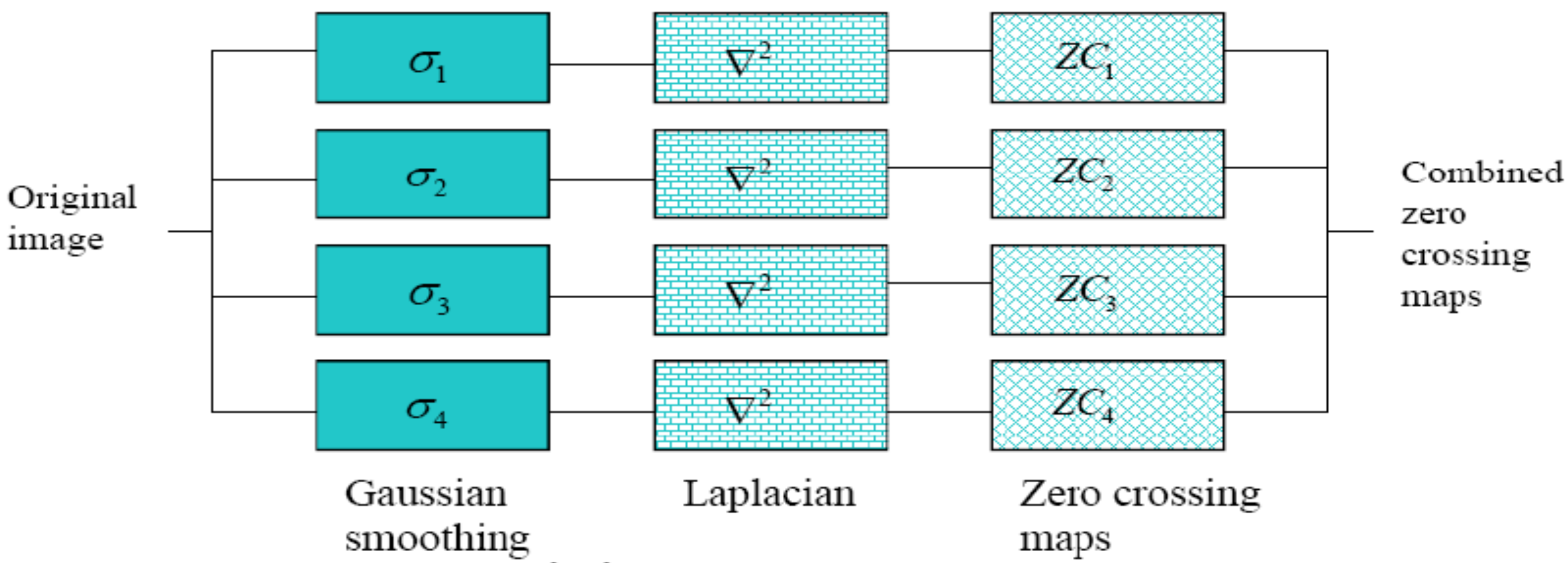
- What is LoG?

LoG filter (Mexican Hat)

- What is LoG? Convolution of Laplacian and Gaussian

LoG filter (Mexican Hat)

- What is LoG? Convolution of Laplacian and Gaussian



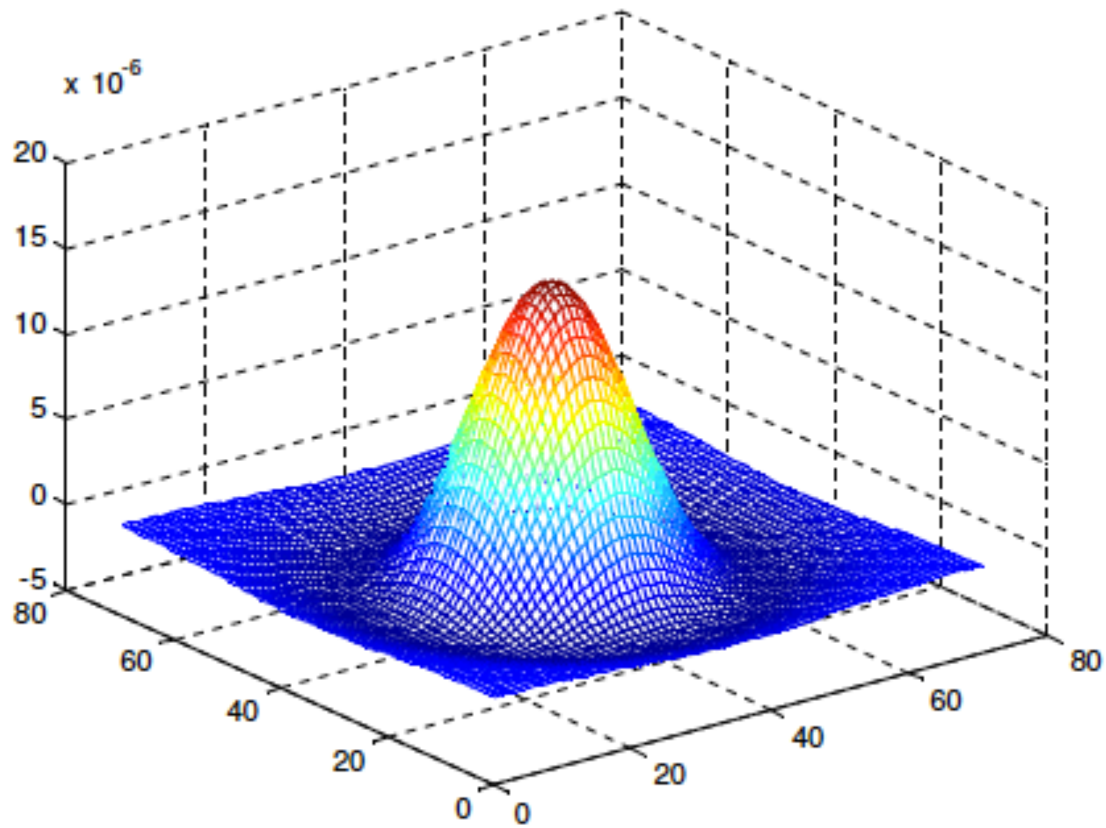
$$G = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad \nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

$$\nabla^2 G = \frac{1}{\pi\sigma^4}\left(1 - \frac{x^2+y^2}{2\sigma^2}\right) e^{-\frac{x^2+y^2}{2\sigma^2}}$$

LoG filter (Mexican Hat)

- What is LoG? Convolution of Laplacian and Gaussian or "The Mexican Hat"

LoG=conv2(Gaus, Lap)



LoG filter (Mexican Hat)

- Why does it work?

LoG filter (Mexican Hat)

- Why does it work?
  - Convolution is commutative:

    conv2(A, B)=conv2(B,A)
  - Convolution is associative:

    conv2(conv2(img, A),B)=conv2(img, conv2(A,B))

4. Zero Crossing

- Why do we need to get the zero crossing of the Laplacian of Gaussian of the image to find the edges?

4. Zero Crossing

- Why do we need to get the zero crossing of the Laplacian of Gaussian of the image to find the edges?
- What happens to the first order derivative of f, at the zero crossing of the second derivative of f?

4. Zero Crossing

- Why do we need to get the zero crossing of the Laplacian of Gaussian of the image to find the edges?
- What happens to the first order derivative of f, at the zero crossing of the second derivative of f?
- What happens to f, at the zero crossing of the second derivative of f?

4. Zero Crossing

- How to do Zero Crossing?

4. Zero Crossing

- How to do Zero Crossing?

  img(x, y) is a zero crossing if:

  - img(x, y)*img(x+1, y)<0

  Or:

  - img(x, y)*img(x, y+1)<0

If img(x, y) is zero crossing, set the corresponding pixel (imgEdge(x,y)) in the edge image to 1, otherwise to 0.

* Finding the exact 0s are hard, due to precision and discretization.

Functions to use; Functions not to use

- Matlab Functions that are OK to be used:
  - Basic Functions such as: Conv2, imfilter, imread, imwrite, rgb2gray, size, figure, imshow, …
- Matlab Functions that are NOT OK to be used:
  - Complex functions such as: Mexihat, edgeDetector, fspecial functions (ie. Of Gaussian, or Laplacian type), …

Functions to use; Functions not to use

- Matlab Functions that are OK to be used:
  - Basic Functions such as: Conv2, imfilter, imread, imwrite, rgb2gray, size, figure, imshow, …
- Matlab Functions that are NOT OK to be used:
  - Complex functions such as: Mexihat, edgeDetector, fspecial functions (ie. Of Gaussian, or Laplacian type), …

Basically you need to construct Laplacian and Gaussian filters, and use convolution operation(conv2) to apply them to the image: As simple as that!

Questions?