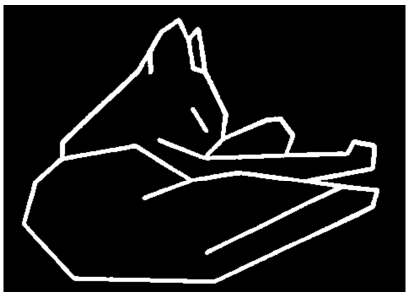
3D Computer Vision
and Video Computing

Introduction

CSc I6716
Spring 2011

Part I
Feature Extraction (2)



Edge Detection

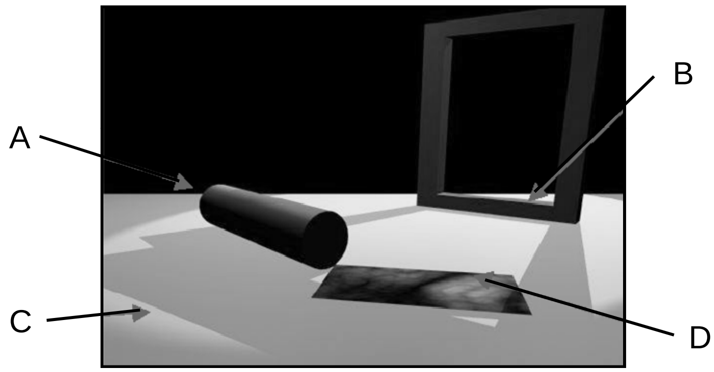
Zhigang Zhu, City College of New York zhu@cs.ccny.cuny.edu

3D Computer Vision
and Video Computing

Edge Detection

- What's an edge?
  - "He was sitting on the Edge of his seat."
  - "She paints with a hard Edge."
  - "I almost ran off the Edge of the road."
  - "She was standing by the Edge of the woods."
  - "Film negatives should only be handled by their Edges."
  - "We are on the Edge of tomorrow."
  - "He likes to live life on the Edge."
  - "She is feeling rather Edgy."
- The definition of Edge is not always clear.
- In Computer Vision, Edge is usually related to a discontinuity within a local set of pixels.

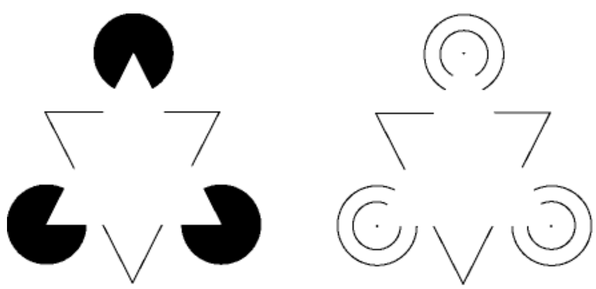
Discontinuities



- A: Depth discontinuity: abrupt depth change in the world
- B: Surface normal discontinuity: change in surface orientation
- C: Illumination discontinuity: shadows, lighting changes
- D: Reflectance discontinuity: surface properties, markings

Illusory Edges

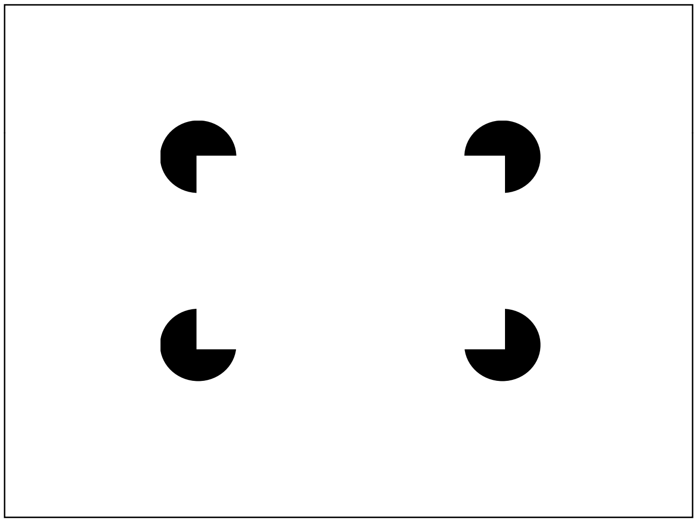
Kanizsa
Triangles



- Illusory edges will not be detectable by the algorithms that we will discuss
- No change in image irradiance - no image processing algorithm can directly address these situations
- Computer vision can deal with these sorts of things by drawing on information external to the image (perceptual grouping techniques)

3D Computer Vision
and Video Computing

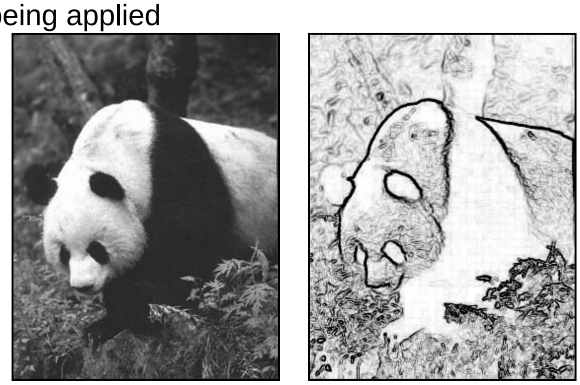
Another One



3D Computer Vision
and Video Computing

Goal

- Devise computational algorithms for the extraction of significant edges from the image.
- What is meant by significant is unclear.
  - Partly defined by the context in which the edge detector is being applied



Edgels

- Define a local edge or *edgel* to be a rapid change in the image function over a small area
  - implies that edgels should be detectable over a local neighborhood
- Edgels are NOT contours, boundaries, or lines
  - edgels may lend support to the existence of those structures
  - these structures are typically constructed from edgels
- Edgels have properties
  - Orientation
  - Magnitude
  - Position

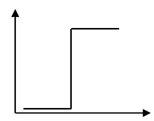
Outline

- First order edge detectors (lecture - required)
  - Mathematics
  - 1x2, Roberts, Sobel, Prewitt
- Canny edge detector (after-class reading)
- Second order edge detector (after-class reading)
  - Laplacian, LOG / DOG
- Hough Transform – detect by voting
  - Lines
  - Circles
  - Other shapes

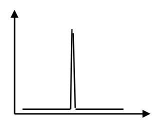
Locating Edgels

Rapid change in image => high local gradient => differentiation

f(x) = step edge

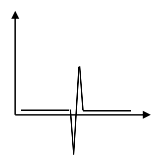


1^{st} Derivative f '(x)



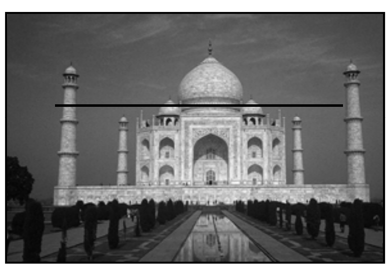
maximum

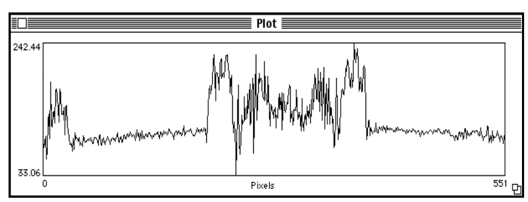
2^{nd} Derivative -f ''(x)

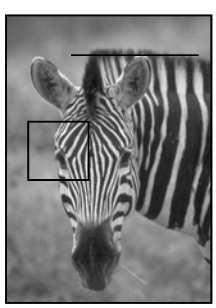


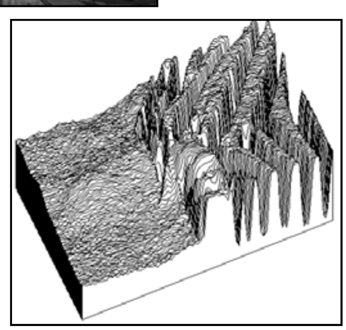
zero crossing

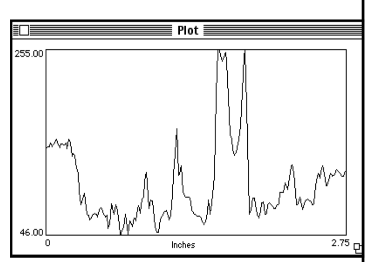
Reality



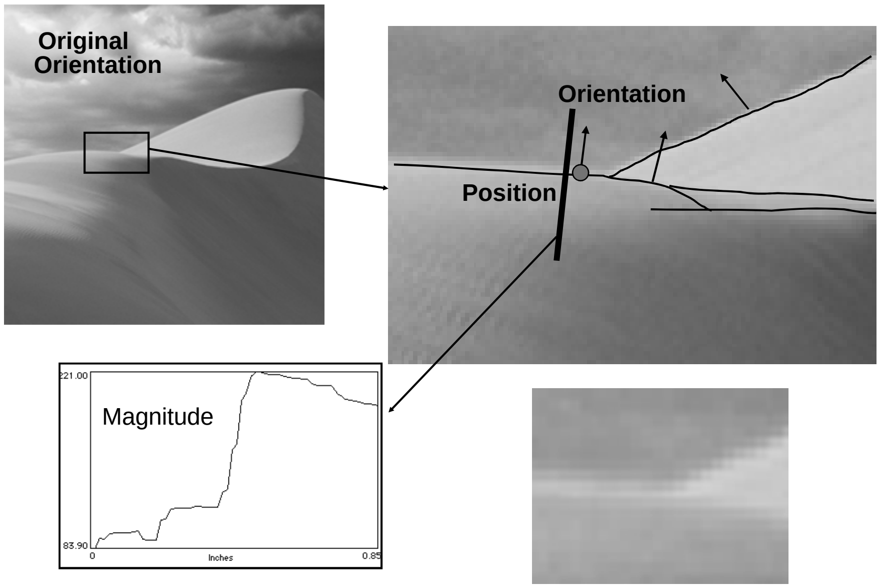








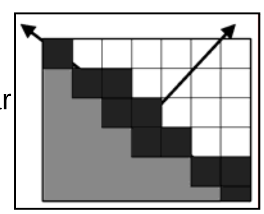
Properties of an Edge



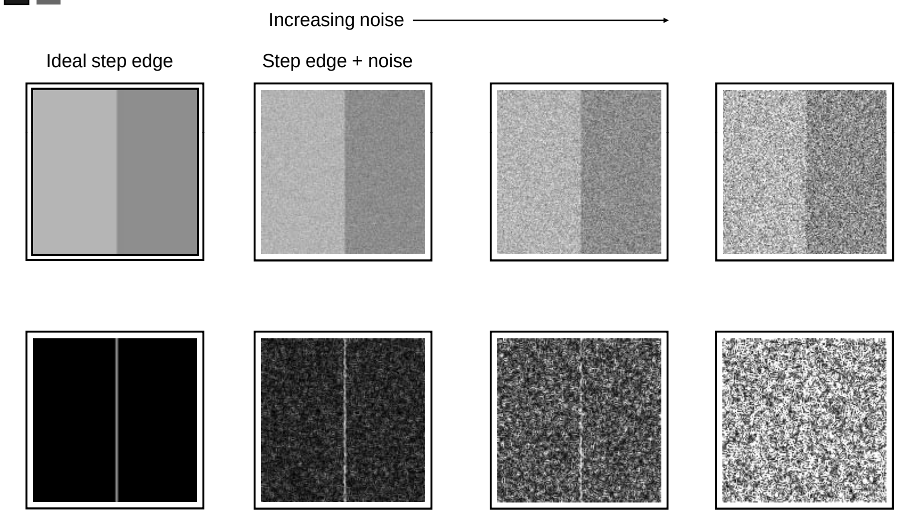
3D Computer Vision
and Video Computing

Quantitative Edge Descriptors

- Edge Orientation
  - Edge Normal - unit vector in the direction of maximum intensity change (maximum intensity gradient)
  - Edge Direction - unit vector perpendicular the edge normal
- Edge Position or Center
  - image position at which edge is located (usually saved as binary image)
- Edge Strength / Magnitude
  - related to local contrast or gradient - how rapid is the intensity variation across the edge along the edge normal.

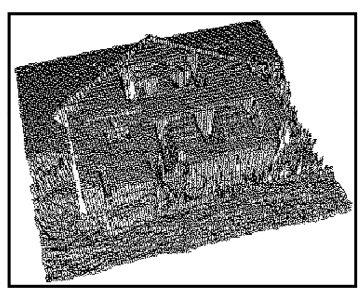


Edge Degradation in Noise



3D Computer Vision
and Video Computing

Real Image



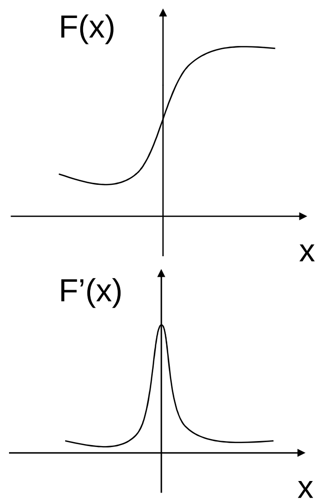
Edge Detection: Typical

- Noise Smoothing
  - Suppress as much noise as possible while retaining 'true' edges
  - In the absence of other information, assume 'white' noise with a Gaussian distribution
- Edge Enhancement
  - Design a filter that responds to edges; filter output high are edge pixels and low elsewhere
- Edge Localization
  - Determine which edge pixels should be discarded as noise and which should be retained
    - thin wide edges to 1-pixel width (nonmaximum suppression)
    - establish minimum value to declare a local maximum from edge filter to be an edge (thresholding)

Edge Detection Methods

- 1st Derivative Estimate
  - Gradient edge detection
  - Compass edge detection
  - Canny edge detector (*)
- 2nd Derivative Estimate
  - Laplacian
  - Difference of Gaussians
- Parametric Edge Models (*)

Gradient Methods



Edge= sharp variation

↓

Large first derivative

Gradient of a Function

- Assume f is a continuous function in (x,y). Then

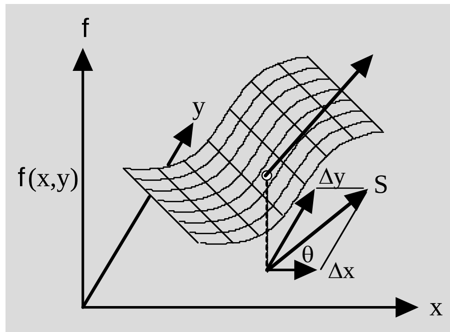
$$\Delta_x = \frac{\partial f}{\partial x}, \quad \Delta_y = \frac{\partial f}{\partial y}$$

- are the rates of change of the function f in the x and y directions, respectively.
- The vector (Δ_x, Δ_y) is called the gradient of f.
- This vector has a magnitude: $s = \sqrt{\Delta_x^2 + \Delta_y^2}$

  and an orientation: $\theta = \tan^{-1} \left(\frac{\Delta_y}{\Delta_x} \right)$

- θ is the direction of the maximum change in f.
- S is the size of that change.

Geometric Interpretation



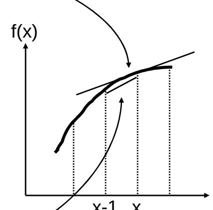
- But
  - I(i,j) is not a continuous function.
- Therefore
  - look for discrete approximations to the gradient.

3D Computer Vision
and Video Computing

Discrete Approximations

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

$$\frac{df(x)}{dx} \cong \frac{f(x) - f(x-1)}{1}$$

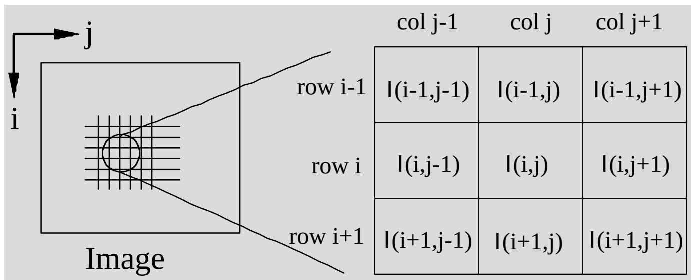


Convolve with

-1	1

In Two Dimensions

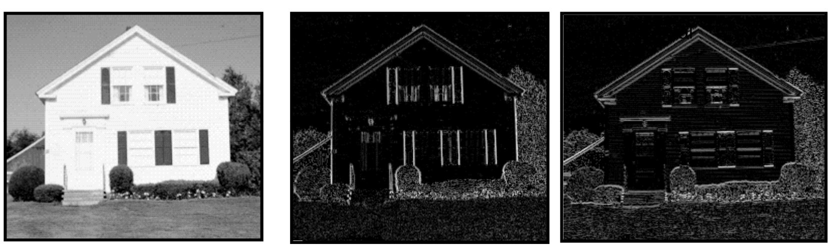
- Discrete image function I



- Derivatives ⇨ Differences

$\Delta_j I = \begin{bmatrix} -1 & 1 \end{bmatrix}$ $\qquad$ $\Delta_i I = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$

1x2 Example



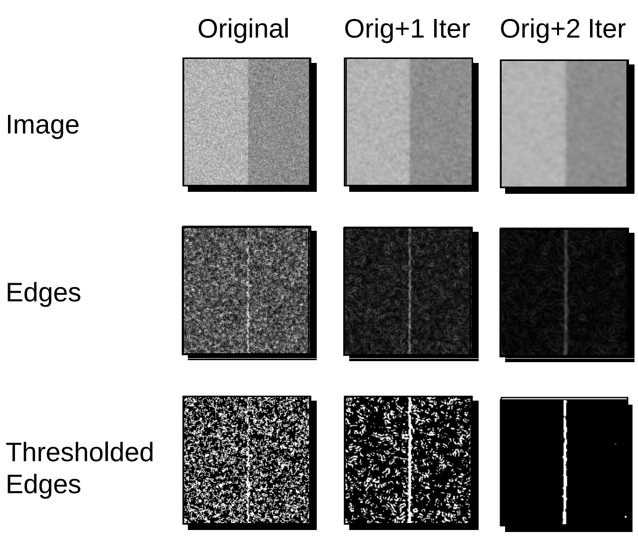
Smoothing and Edge Detection

- Derivatives are 'noisy' operations
  - edges are a high spatial frequency phenomenon
  - edge detectors are sensitive to and accent noise
- Averaging reduces noise
  - spatial averages can be computed using masks

$$\frac{1}{9}\times\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline\end{array} \qquad \frac{1}{8}\times\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 0 & 1 \\ \hline 1 & 1 & 1 \\ \hline\end{array}$$

- Combine smoothing with edge detection.

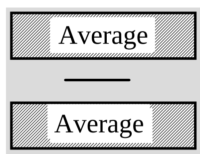
Effect of Blurring

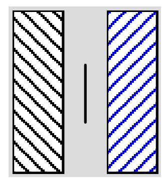


Combining the Two

- Applying this mask is equivalent to taking the difference of averages on either side of the central pixel.

-1	-1	-1
0	●	0
1	1	1





Many Different Kernels

- Variables
  - Size of kernel
  - Pattern of weights
- 1x2 Operator (we've already seen this one

$\Delta_j I = \begin{bmatrix} -1 & 1 \end{bmatrix}$ $\quad$ $\Delta_i I = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$

Roberts Cross Operator

- Does not return any information about the orientation of the edge

$$S = \sqrt{[\,I(x, y) - I(x+1, y+1)\,]^2 + [\,I(x, y+1) - I(x+1, y)\,]^2}$$

or

$$S = |\,I(x, y) - I(x+1, y+1)\,| + |\,I(x, y+1) - I(x+1, y)\,|$$

$$\left| \begin{array}{|c|c|} \hline 1 & 0 \\ \hline 0 & -1 \\ \hline \end{array} \right| + \left| \begin{array}{|c|c|} \hline 0 & 1 \\ \hline -1 & 0 \\ \hline \end{array} \right|$$

Sobel Operator

$$S_1 = \begin{array}{|c|c|c|} \hline -1 & -2 & -1 \\ \hline 0 & 0 & 0 \\ \hline 1 & 2 & 1 \\ \hline \end{array} \qquad S_2 = \begin{array}{|c|c|c|} \hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$$

$$\text{Edge Magnitude} = \sqrt{S_1^2 + S_2^2}$$

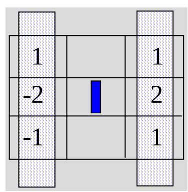
$$\text{Edge Direction} = \tan^{-1}\left(\frac{S_1}{S_2}\right)$$

Anatomy of the Sobel

$$\frac{1}{4}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} = 1/4 * [-1\ \ 0\ +1] \circledast \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

$$\frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} = 1/4 * [\ 1\ \ 2\ \ 1] \circledast \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

Sobel kernel
is separable!



Averaging done parallel to edge

3D Computer Vision
and Video Computing

Prewitt Operator

$$P_1 = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \qquad P_2 = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

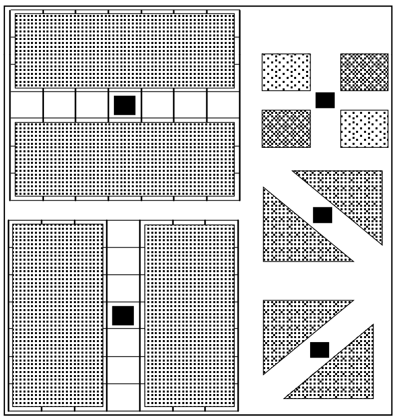
$$\text{Edge Magnitude} = \sqrt{P_1^2 + P_2^2}$$

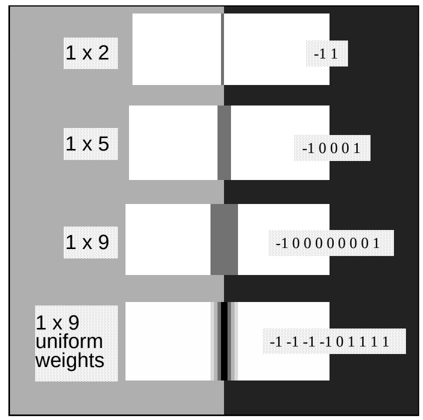
$$\text{Edge Direction} = \tan^{-1}\left(\frac{P_1}{P_2}\right)$$

3D Computer Vision
and Video Computing

Large Masks

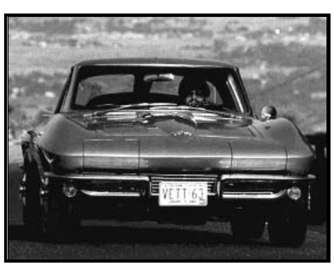
What happens as the mask size increases?

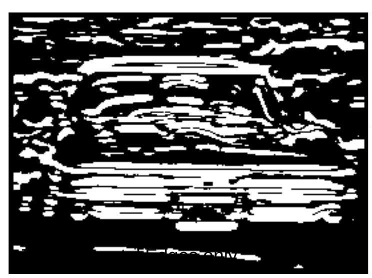


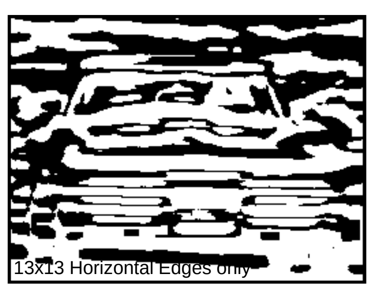


3D Computer Vision
and Video Computing

Large Kernels



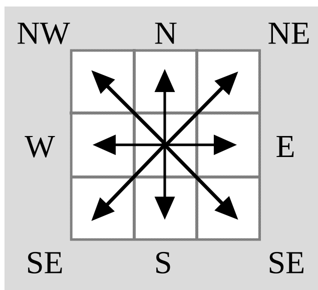


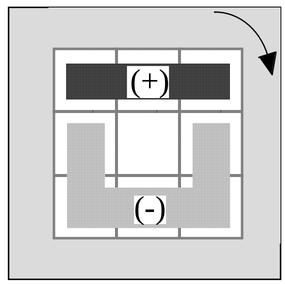


13X13 Horizontal Edges only

Compass Masks

- Use eight masks aligned with the usual compass directions
- Select largest response (magnitude)
- Orientation is the direction associated with the largest response





3D Computer Vision
and Video Computing

Many Different Kernels

1	1	1
1	-2	1
-1	-1	-1

Prewitt 1

5	5	5
-3	0	-3
-3	-3	-3

Kirsch

-1	$-\sqrt{2}$	-1
0	0	0
1	$\sqrt{2}$	1

Frei & Chen

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

Prewitt 2

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

Sobel

Robinson Compass Masks

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

→

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

↗

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

↑

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

↖

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

←

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

↙

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

↓

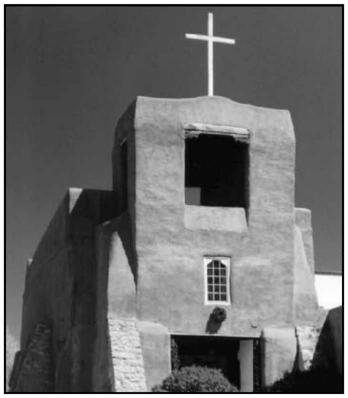
-2	-1	0
-1	0	1
0	1	2

↘

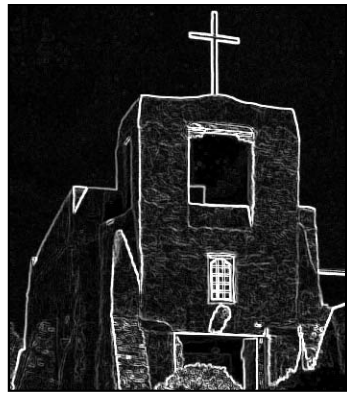
Analysis of Edge Kernels

- Analysis based on a step edge inclined at an angle θ (relative to y-axis) through center of window.
- Robinson/Sobel: true edge contrast less than 1.6% different from that computed by the operator.
- Error in edge direction
  - Robinson/Sobel: less than 1.5 degrees error
  - Prewitt: less than 7.5 degrees error
- Summary
  - Typically, 3 x 3 gradient operators perform better than 2 x 2.
  - Prewitt2 and Sobel perform better than any of the other 3x3 gradient estimation operators.
  - In low signal to noise ratio situations, gradient estimation operators of size larger than 3 x 3 have improved performance.
  - In large masks, weighting by distance from the central pixel is beneficial.

Prewitt Example



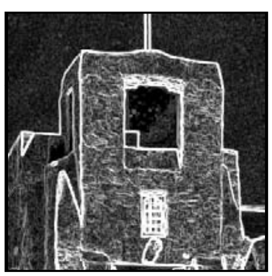
Santa Fe Mission

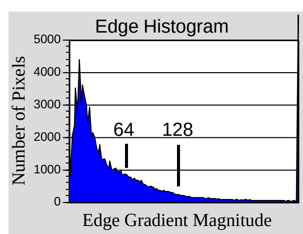


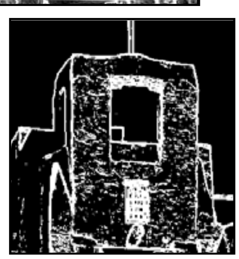
Prewitt Horizontal
and Vertical Edges
Combined

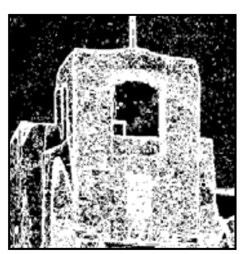
Edge Thresholding

- Global approach









See Haralick paper for thresholding based on statistical significance tests.

Demo in Photoshop

- Go through slides 40-71 after class
- Reading: Chapters 4 and 5
- Homework 2: Due after two weeks

You may try different operators in Photoshop, but do your homework by programming … …

Canny Edge Detector

- Probably most widely used
- LF. Canny, "A computational approach to edge detection", IEEE Trans. Pattern Anal. Machine Intelligence (PAMI), vol. PAMI vii-g, pp. 679-697, 1986.
- Based on a set of criteria that should be satisfied by an edge detector:
  - Good detection. There should be a minimum number of false negatives and false positives.
  - Good localization. The edge location must be reported as close as possible to the correct position.
  - Only one response to a single edge.

⇩

Cost function which could be optimized using variational methods

Canny Results

σ=1, T2=255, T1=1

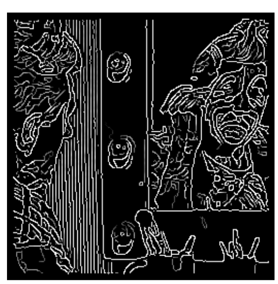
```
I = imread('image file name');
BW1 = edge(I,'sobel');
BW2 = edge(I,'canny');
imshow(BW1)
figure, imshow(BW2)
```

'Y' or 'T' junction problem with Canny operator

3D Computer Vision
and Video Computing

Canny Results

σ=1, T2=255, T1=220



σ=1, T2=128, T1=1

σ=2, T2=128, T1=1

M. Heath, S. Sarkar, T. Sanocki, and K.W. Bowyer, "A Robust Visual Method for Assessing the Relative Performance of Edge-Detection Algorithms" IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 19, No. 12, December 1997, pp. 1338-1359.

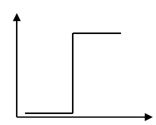
http://marathon.csee.usf.edu/edge/edge_detection.html

- Second derivatives…

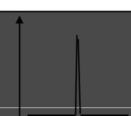
Edges from Second Derivatives

- Digital gradient operators estimate the first derivative of the image function in two or more directions.

f(x) = step edge



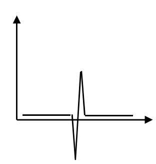
1st Derivative f'(x)



maximum

GRADIENT METHODS

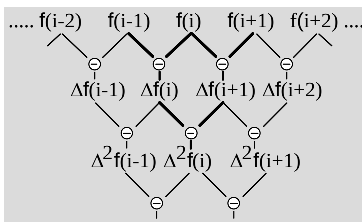
2nd Derivative f''(x)



zero crossing

Second Derivatives

- Second derivative = rate of change of first derivative.
- Maxima of first derivative = zero crossings of second derivative.
- For a discrete function, derivatives can be approximated by differencing.
- Consider the one dimensional case:



$\Delta^2 f(i) = \Delta\ f(i+1) - \Delta\ f(i)$

$= f(i+1) - 2\ f(i) + f(i-1)$

Mask:

1	**-2**	**1**

3D Computer Vision
and Video Computing

Laplacian Operator

- Now consider a two-dimensional function f(x,y).
- The second partials of f(x,y) are not isotropic.
- Can be shown that the smallest possible isotropic second derivative operator is the Laplacian:

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

- Two-dimensional discrete approximation is:

	1	
1	-4	1
	1	

Example Laplacian Kernels

-1	-1	-1	-1	-1
-1	-1	-1	-1	-1
-1	-1	24	-1	-1
-1	-1	-1	-1	-1
-1	-1	-1	-1	-1

5X5

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	+8	+8	+8	-1	-1	-1
-1	-1	-1	+8	+8	+8	-1	-1	-1
-1	-1	-1	+8	+8	+8	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

9X9

- Note that these are not the optimal approximations to the Laplacian of the sizes shown.

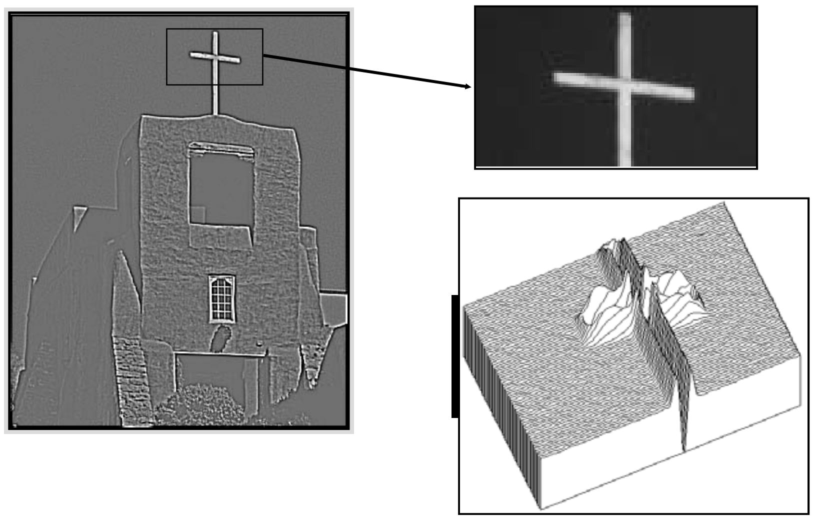
3D Computer Vision and Video Computing

Example Application

5x5 Laplacian Filter

9x9 Laplacian Filter

Detailed View of Results



Interpretation of the Laplacian

- Consider the definition of the discrete Laplacian:

$$\nabla^2 I = \underbrace{I(i+1,j)+I(i-1,j)+I(i,j+1)+I(i,j-1)}_{\text{looks like a window sum}} - 4I(i,j)$$

- Rewrite as:

$$\nabla^2 I = I(i+1,j)+I(i-1,j)+I(i,j+1)+I(i,j-1)+I(i,j) - 5I(i,j)$$

- Factor out -5 to get:

$$\nabla^2 I = -5 \{I(i,j) - \text{window average}\}$$

- Laplacian can be obtained, up to the constant -5, by subtracting the average value around a point (i,j) from the image value at the point (i,j)!
  - What window and what averaging function?

Enhancement using the Laplacian

- The Laplacian can be used to enhance images:

$$I(i,j) - \nabla^2 I(i,j) = 5\,I(i,j) - [I(i+1,j) + I(i-1,j) + I(i,j+1) + I(i,j-1)]$$

- If (i,j) is in the middle of a flat region or long ramp: $I-\nabla^2 I = I$
- If (i,j) is at low end of ramp or edge: $I-\nabla^2 I < I$
- If (i,j) is at high end of ramp or edge: $I-\nabla^2 I > I$

- Effect is one of deblurring the image

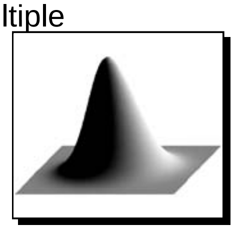
Laplacian Enhancement

Blurred Original

3x3 Laplacian Enhanced

Noise

- Second derivative, like first derivative, enhances noise
- Combine second derivative operator with a smoothing operator.
- Questions:
  - Nature of optimal smoothing filter.
  - How to detect intensity changes at a given scale.
  - How to combine information across multiple scales.
- Smoothing operator should be
  - 'tunable' in what it leaves behind
  - smooth and localized in image space.
- One operator which satisfies these two

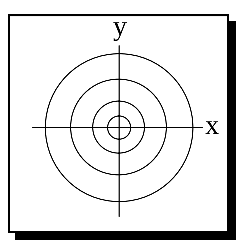


2D Gaussian Distribution

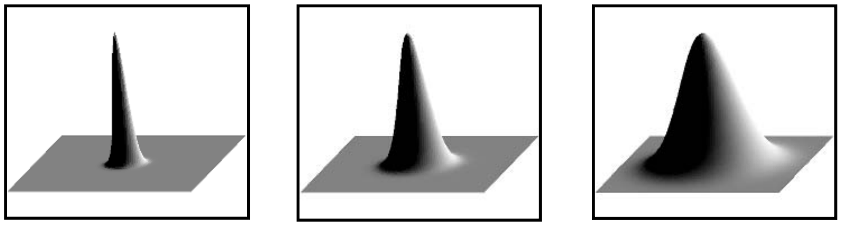
- The two-dimensional Gaussian distribution is defined by:

$$G(x,y) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{-\left[\frac{(x^2+y^2)}{2\sigma^2}\right]}$$

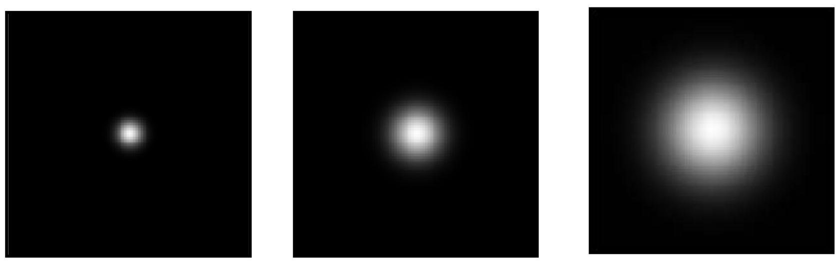
- From this distribution, can generate smoothing masks whose width depends upon σ:



σ Defines Kernel 'Width'



$\sigma^2 = .25$ $\sigma^2 = 1.0$ $\sigma^2 = 4.0$



Creating Gaussian Kernels

- The mask weights are evaluated from the Gaussian distribution:

$$W(i,j) = k * \exp\left(- \frac{i^2 + j^2}{2\,\sigma^2}\right)$$

- This can be rewritten as:

$$\frac{W(i,j)}{k} = \exp\left(- \frac{i^2 + j^2}{2\,\sigma^2}\right)$$

- This can now be evaluated over a window of size nxn to obtain a kernel in which the (0,0) value is 1.
- k is a scaling constant

3D Computer Vision
and Video Computing

Example

- Choose σ^2 = 2. and n = 7, then:

i \ j	-3	-2	-1	0	1	2	3
-3	.011	.039	.082	.105	.082	.039	.011
-2	.039	.135	.287	.368	.287	.135	.039
-1	.082	.287	.606	.779	.606	.287	.082
0	.105	.039	.779	1.000	.779	.368	.105
1	.082	.287	.606	.779	.606	$\boxed{.287}$	.082
2	.039	.135	.287	.368	.287	.135	.039
3	.011	.039	.082	.105	.082	.039	$\boxed{.011}$

$$\frac{W(1,2)}{k} = \exp\left(-\frac{1^2 + 2^2}{2*2}\right)$$

To make this value 1, choose k = 91.

3D Computer Vision
and Video Computing

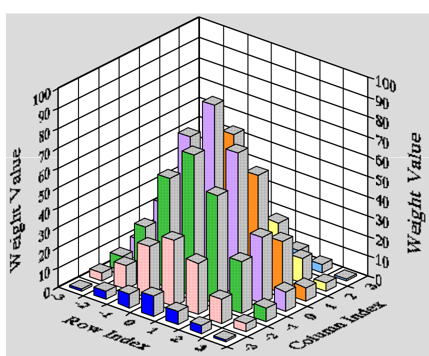
Example

1	4	7	10	7	4	1
4	12	26	33	26	12	4
7	26	55	71	55	26	7
10	33	71	91	71	33	10
7	26	55	71	55	26	7
4	12	26	33	26	12	4
1	4	7	10	7	4	1

7x7 Gaussian Filter

$$\sum_{i=-3}^{3}\sum_{j=-3}^{3} W(i,j) = 1,115$$

Plot of Weight Values



Kernel Application

7x7 Gaussian Kernel

15x15 Gaussian Kernel

Why Gaussian for Smoothing

- Gaussian is not the only choice, but it has a number of important properties
  - If we convolve a Gaussian with another Gaussian, the result is a Gaussian
    - This is called linear scale space

$$G_{\sigma_1} * G_{\sigma_2} = G_{\sqrt{\sigma_1^2+\sigma_2^2}}.$$

  - Efficiency: separable
  - Central limit theorem

Why Gaussian for Smoothing

- Gaussian is separable

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{(x^2+y^2)}{2\sigma^2}\right)$$
$$= \left(\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x^2)}{2\sigma^2}\right)\right) \times \left(\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(y^2)}{2\sigma^2}\right)\right),$$

Why Gaussian for Smoothing – cont.

- Gaussian is the solution to the diffusion equation

$$\frac{\partial \Phi}{\partial \sigma} = \frac{\partial^2 \Phi}{\partial x^2} + \frac{\partial^2 \Phi}{\partial y^2} = \nabla^2 \Phi,$$
$$\Phi(x, y, 0) = \mathcal{I}(x, y)$$

- We can extend it to non-linear smoothing

$$\frac{\partial \Phi}{\partial \sigma} = \nabla \cdot (c(x, y, \sigma)\nabla\Phi)$$
$$= c(x, y, \sigma)\nabla^2\Phi + (\nabla c(x, y, \sigma)) \cdot (\nabla\Phi)$$

∇^2G Filter

- Marr and Hildreth approach:
  1. Apply Gaussian smoothing using σ's of increasing size:

     $$G \circledast I$$

  2. Take the Laplacian of the resulting images:

     $$\nabla^2 (G \circledast I)$$

  3. Look for zero crossings.

- Second expression can be written as: $(\nabla^2 G) \circledast I$

- Thus, can take Laplacian of the Gaussian and use that as the operator.

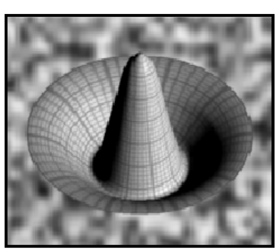
3D Computer Vision
and Video Computing

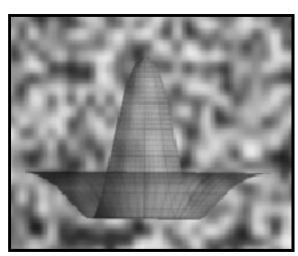
Mexican Hat Filter

- Laplacian of the Gaussian

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

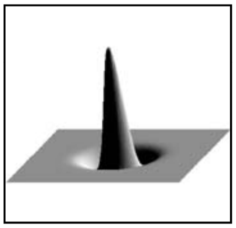
- ∇^2G is a circularly symmetric operator.
- Also called the hat or Mexican-hat operator.



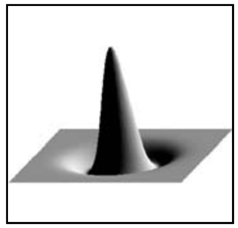


3D Computer Vision
and Video Computing

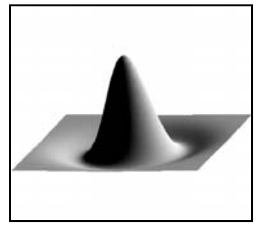
σ^2 Controls Size



$\sigma^2 = 0.5$



$\sigma^2 = 1.0$



$\sigma^2 = 2.0$

3D Computer Vision
and Video Computing

Kernels

5x5

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

17 x 17

0	0	0	0	0	0	-1	-1	-1	-1	-1	0	0	0	0	0	0
0	0	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0
0	0	-1	-1	-1	-2	-3	-3	-3	-3	-3	-2	-1	-1	-1	0	0
0	0	-1	-1	-2	-3	-3	-3	-3	-3	-3	-3	-2	-1	-1	0	0
0	-1	-1	-2	-3	-3	-3	-2	-3	-2	-3	-3	-3	-2	-1	-1	0
0	-1	-2	-3	-3	-3	0	2	4	2	0	-3	-3	-3	-2	-1	0
-1	-1	-3	-3	-3	0	4	10	12	10	4	0	-3	-3	-3	-1	-1
-1	-1	-3	-3	-2	2	10	18	21	18	10	2	-2	-3	-3	-1	-1
-1	-1	-3	-3	-3	4	12	21	24	21	112	4	-3	-3	-3	-1	-1
-1	-1	-3	-3	-2	2	10	18	21	18	10	2	-2	-3	-3	-1	-1
-1	-1	-3	-3	-3	0	4	10	12	10	4	0	-3	-3	-3	-1	-1
0	-1	-2	-3	-3	-3	0	2	4	2	0	-3	-3	-3	-2	-1	0
0	-1	-1	-2	-3	-3	-3	-3	-3	-3	-3	-3	-3	-2	-1	-1	0
0	0	-1	-1	-2	-3	-3	-3	-3	-3	-3	-3	-2	-1	-1	0	0
0	0	-1	-1	-1	-2	-3	-3	-3	-3	-3	-2	-1	-1	-1	0	0
0	0	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0

- Remember the center surround cells in the human system?

Example

13x13 Kernel

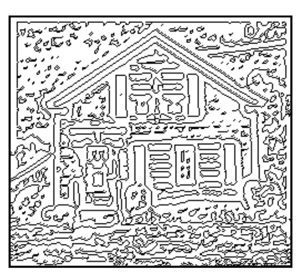
3D Computer Vision
and Video Computing

Example

13 x 13 Hat Filter

Thesholded Positive

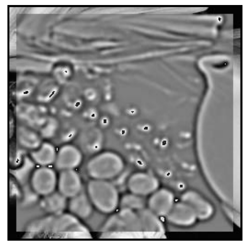
Thesholded Negative



Zero Crossings

3D Computer Vision and Video Computing

Scale Space



17x17 LoG Filter

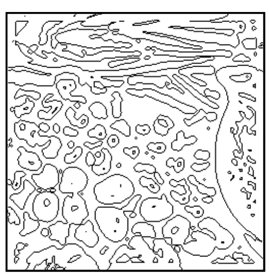
Thresholded Positive

Thresholded Negative

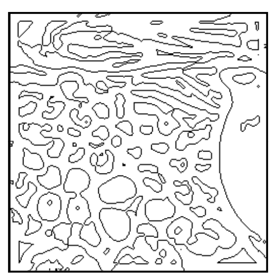
Zero Crossings

3D Computer Vision and Video Computing

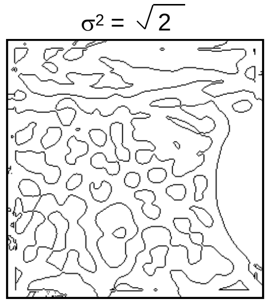
Scale Space



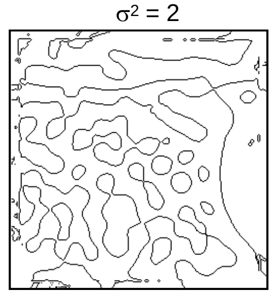
$\sigma^2 = \sqrt{2}$



$\sigma^2 = 2$



$\sigma^2 = 2\sqrt{2}$



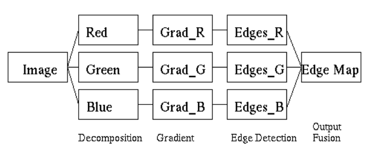
$\sigma^2 = 4$

Multi-Resolution Scale Space

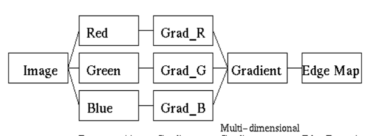
- Observations:
  - For sufficiently different σ 's, the zero crossings will be unrelated unless there is 'something going on' in the image.
  - If there are coincident zero crossings in two or more successive zero crossing images, then there is sufficient evidence for an edge in the image.
  - If the coincident zero crossings disappear as σ becomes larger, then either:
    - two or more local intensity changes are being averaged together, or
    - two independent phenomena are operating to produce intensity changes in the same region of the image but at different scales.
- Use these ideas to produce a 'first-pass' approach to edge detection using multi-resolution zero crossing data.
- Never completely worked out
- See Tony Lindbergh's thesis and papers

Color Edge Detection

- Typical Approaches
  - Fusion of results on R, G, B separately



  - Multi-dimensional gradient methods



  - Vector methods
  - Color signatures: Stanford (Rubner and Thomasi)

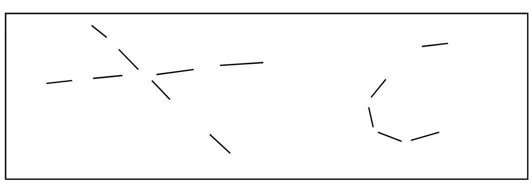
Hierarchical Feature Extraction

- Most features are extracted by combining a small set of primitive features (edges, corners, regions)
  - Grouping: which edges/corners/curves form a group?
    - perceptual organization at the intermediate-level of vision
  - Model Fitting: what structure best describes the group?
- Consider a slightly simpler problem…..

3D Computer Vision
and Video Computing

From Edgels to Lines

- Given local edge elements:



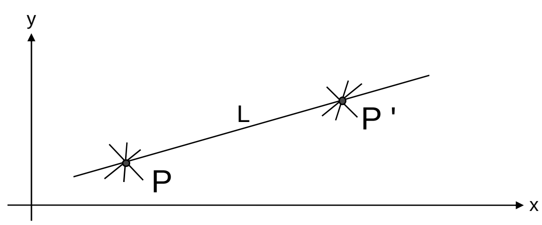
- Can we organize these into more 'complete' structures, such as straight lines?
- Group edge points into lines?
- Consider a fairly simple technique...

Edgels to Lines

- Given a set of local edge elements
  - With or without orientation information
- How can we extract longer straight lines?
- General idea:
  - Find an alternative space in which lines map to points
  - Each edge element 'votes' for the straight line which it may be a part of.
  - Points receiving a high number of votes might correspond to actual straight lines in the image.
- The idea behind the Hough transform is that a change in representation converts a point grouping problem into a peak detection problem

Edgels to Lines

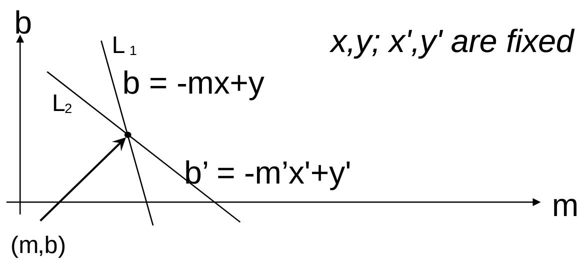
- Consider two (edge) points, P(x,y) and P'(x',y') in image space:



- The set of all lines through P=(x,y) is y=mx + b, for appropriate choices of m and b.
  - Similarly for P'
- But this is also the equation of a line in (m,b) space, or parameter space.

Parameter Space

- The intersection represents the parameters of the equation of a line y=mx+b going through both (x,y) and (x',y').



- The more colinear edgels there are in the image, the more lines will intersect in parameter space
- Leads directly to an algorithm

General Idea

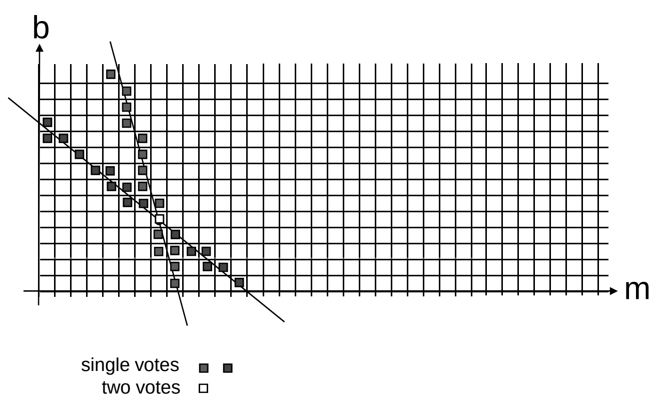
- General Idea:
  - The Hough space (m,b) is a representation of every possible line segment in the plane
  - Make the Hough space (m and b) discrete
  - Let every edge point in the image plane 'vote for' any line it might belong to.

Hough Transform

- Line Detection Algorithm: Hough Transform
  - Quantize b and m into appropriate 'buckets'.
    - Need to decide what's 'appropriate'
  - Create accumulator array H(m,b), all of whose elements are initially zero.
  - For each point (i,j) in the edge image for which the edge magnitude is above a specific threshold, increment all points in H(m,b) for all discrete values of m and b satisfying b = -mj+i.
    - Note that H is a two dimensional histogram
  - Local maxima in H corresponds to colinear edge points in the edge image.

Quantized Parameter Space

- Quantization



The problem of line detection in image space has been transformed into the problem of cluster detection in parameter space

Example

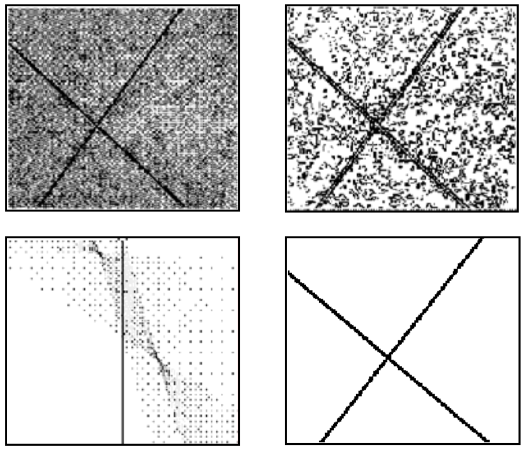
- The problem of line detection in image space has been transformed into the problem of cluster detection in parameter space

Image

Edges

Accumulator Array

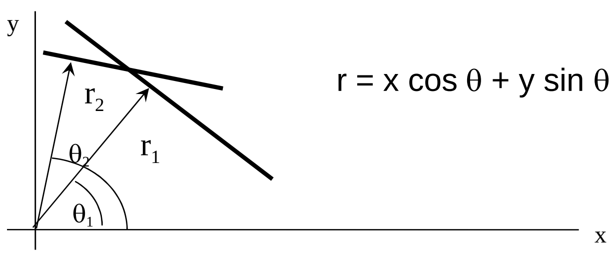
Result



Problems

- Vertical lines have infinite slopes
  - difficult to quantize m to take this into account.
- Use alternative parameterization of a line
  - polar coordinate representation

$$r_1 = x_1 \cos(\theta) + y_1 \sin(\theta)$$



$r = x \cos \theta + y \sin \theta$

3D Computer Vision
and Video Computing

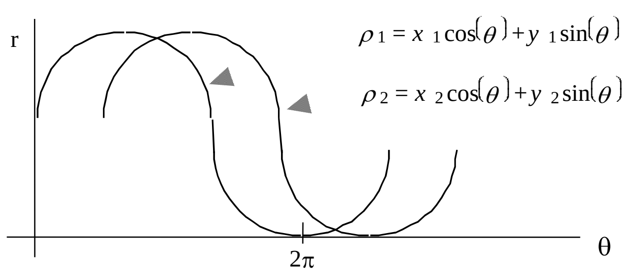
Why?

- (ρ,θ) is an efficient representation:
  - Small: only two parameters (like y=mx+b)
  - Finite: $0 \le \rho \le \sqrt{(row^2+col^2)}$, $0 \le \theta \le 2\pi$
  - Unique: only one representation per line

3D Computer Vision
and Video Computing

Alternate Representation

- Curve in (ρ,θ) space is now a sinusoid
  - but the algorithm remains valid.

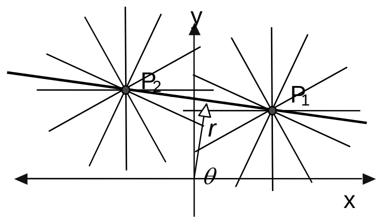


Example

$r = -3\cos(\theta) + 5\sin(\theta)$

$r = 4\cos(\theta) + 4\sin(\theta)$

Two Constraints



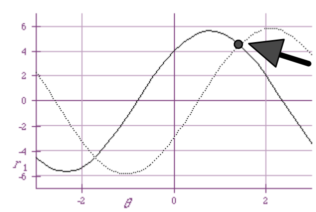
$P_1 = (4, 4)$

$P_2 = (-3, 5)$

$r = 4c + 4s$	$s = \frac{7}{50}\sqrt{50}$	$\theta = 1.4289$
$r = -3c + 5s$	$c = \frac{1}{50}\sqrt{50}$	$r = 4.5255$
$s^2 + c^2 = 1$		

Solve for r and θ

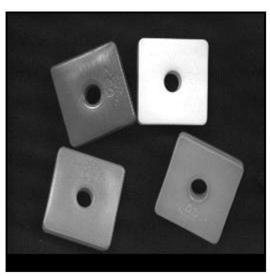
(r, θ) Space

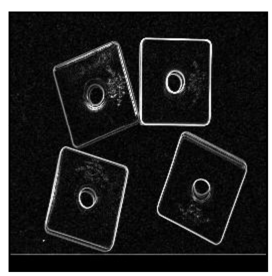


(r, θ)

Real Example

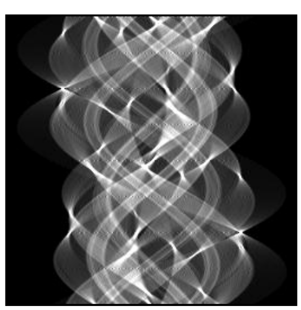
Image

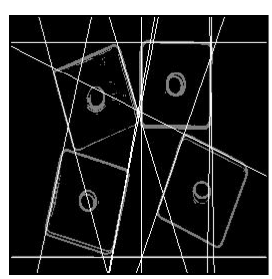




Edges

Accumulator Array

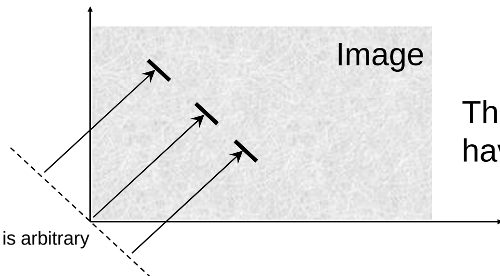




Result

Modifications

- Note that this technique only uses the fact that an edge exists at point (i,j).
- What about the orientation of the edge?
  - More constraints!

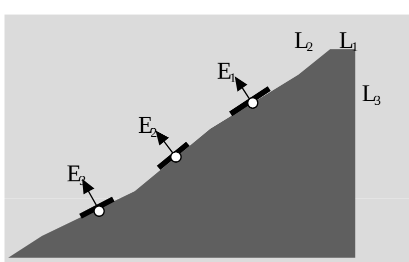


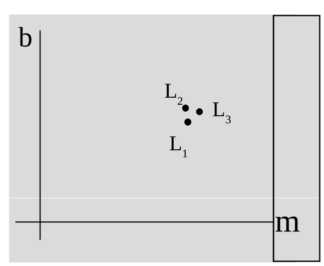
The three edges have same (r, θ)

- Use estimate of edge orientation as θ.
- Each edgel now maps to a point in Hough space.

Gradient Data

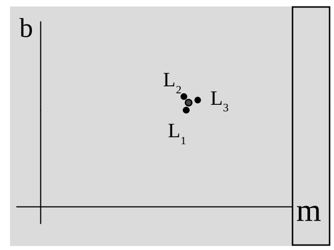
- Colinear edges in Cartesian coordinate space now form point clusters in (m,b) parameter space.



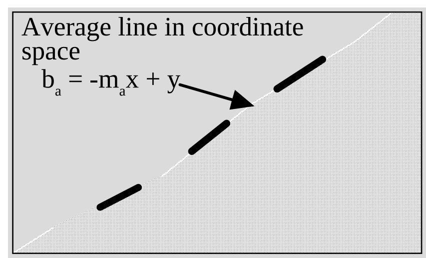


Gradient Data

- ‘Average’ point in Hough Space:

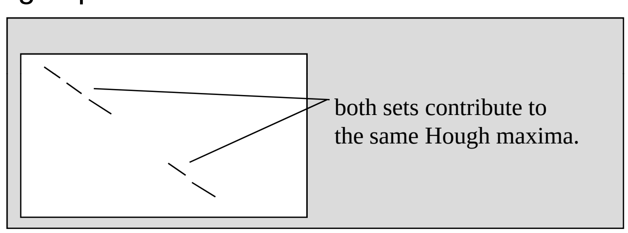


- Leads to an ‘average’ line in image space:



Post Hough

- Image space localization is lost:



- Consequently, we still need to do some image space manipulations, e.g., something like an edge 'connected components' algorithm.
- Heikki Kälviäinen, Petri Hirvonen, L. Xu and Erkki Oja, “Probabilistic and nonprobabilistic Hough Transforms: Overview and comparisons”, *Image and vision computing*, Volume 13, Number 4, pp. 239-252, May 1995.

Hough Fitting

- Sort the edges in one Hough cluster
  - rotate edge points according to θ
  - sort them by (rotated) x coordinate
- Look for Gaps
  - have the user provide a “max gap” threshold
  - if two edges (in the sorted list) are more than max gap apart, break the line into segments
  - if there are enough edges in a given segment, fit a straight line to the points

Generalizations

- Hough technique generalizes to any parameterized curve:

$$f(x,\underbrace{a}_{\text{parameter vector (axes in Hough space)}}) = 0$$

- Success of technique depends upon the quantization of the parameters:
  - too coarse: maxima 'pushed' together
  - too fine: peaks less defined
- Note that exponential growth in the dimensions of the accumulator array with the the number of curve parameters restricts its practical application to curves with few parameters

Example: Finding a Circle

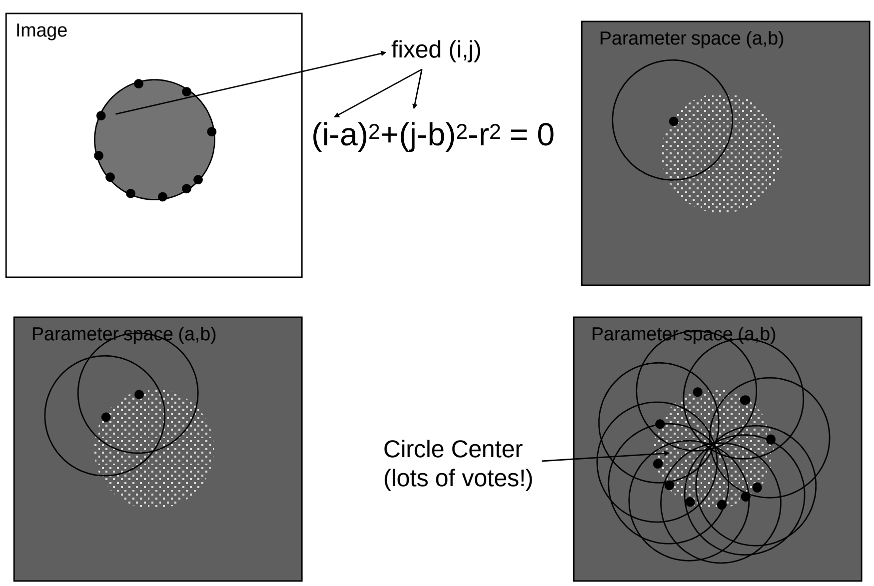
- Circles have three parameters
  - Center (a,b)
  - Radius r
- Circle $f(x,y,r) = (x-a)^2+(y-b)^2-r^2 = 0$
- Task:

> Find the center of a circle with known radius r given an edge image with no gradient direction information (edge location only)

- Given an edge point at (x,y) in the image, where could the center of the circle be?

3D Computer Vision
and Video Computing

Finding a Circle



Finding Circles

- If we don't know r, accumulator array is 3-dimensional
- If edge directions are known, computational complexity if reduced
  - Suppose there is a known error limit on the edge direction (say +/- $10^{o)}$ - how does this affect the search?
- Hough can be extended in many ways….see, for example:
  - Ballard, D. H. Generalizing the Hough Transform to Detect Arbitrary Shapes, Pattern Recognition 13:111-122, 1981.
  - Illingworth, J. and J. Kittler, Survey of the Hough Transform, Computer Vision, Graphics, and Image Processing, 44(1):87-116, 1988