IMAGE PROCESSING & ANALYSIS

References

- Zhigang Zhu, City College of New York, CSc I6716 - Spring 2011, course slides
- E. Trucco, A. Verri, Introductory techniques for 3D computer vision
- R. Szeliski, Computer Vision: Algorithms and Applications, http://szeliski.org/Book/, 2010
- Some publications referenced in the slides.

IMAGE PROCESSING & ANALYSIS

Image enhancement

Point operations

Local operations

Noise reduction / Image smoothing

Feature detection (lines, corners)

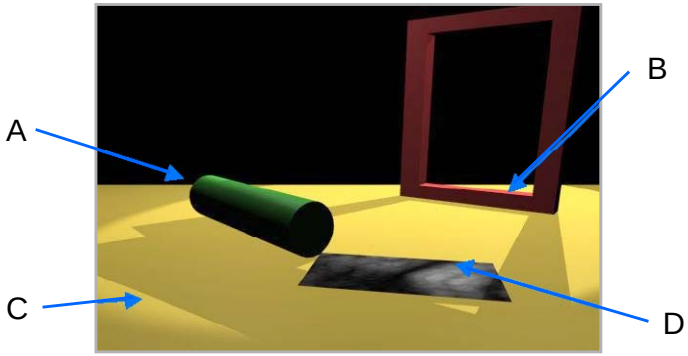
Edge detection

First order & second order detectors

Edge Detection

- What's an edge?
  - "He was sitting on the edge of his seat."
  - "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."
  - " She paints with a hard edge."
  - "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 / depth ➔
  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)

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

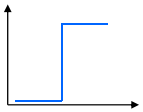
Edgels

- 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

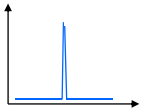
Locating Edgels

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

f(x) - step edge

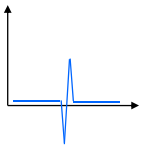


1st Derivative - f '(x)



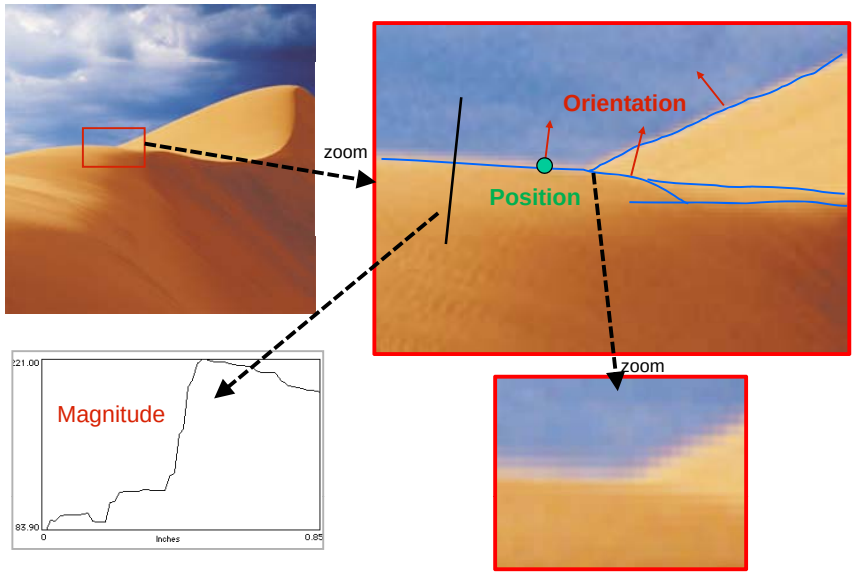
maximum

2nd Derivative - f ''(x)



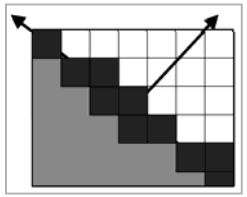
zero crossing

Properties of an Edge



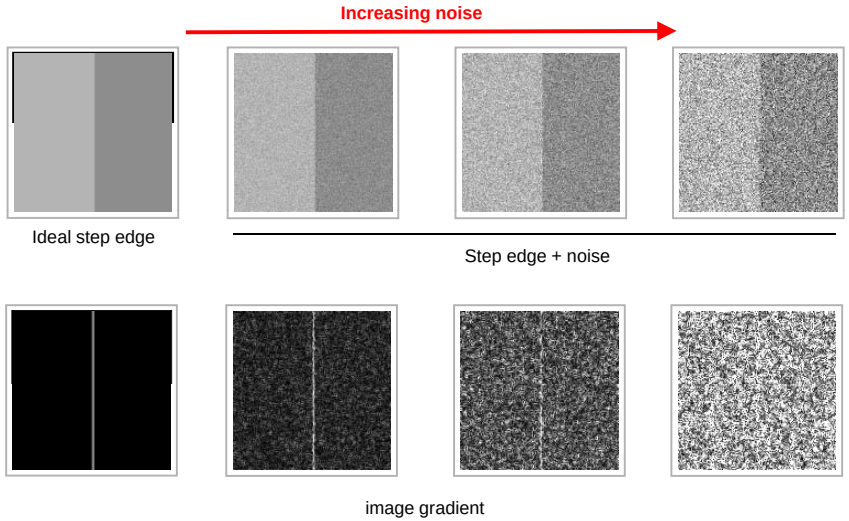
Quantitative Edge Descriptors

- Edge Orientation
  - Edge Normal
    - unit vector in the direction of maximum intensity change (maximum intensity gradient)
  - Edge Direction
    - unit vector perpendicular to the edge normal

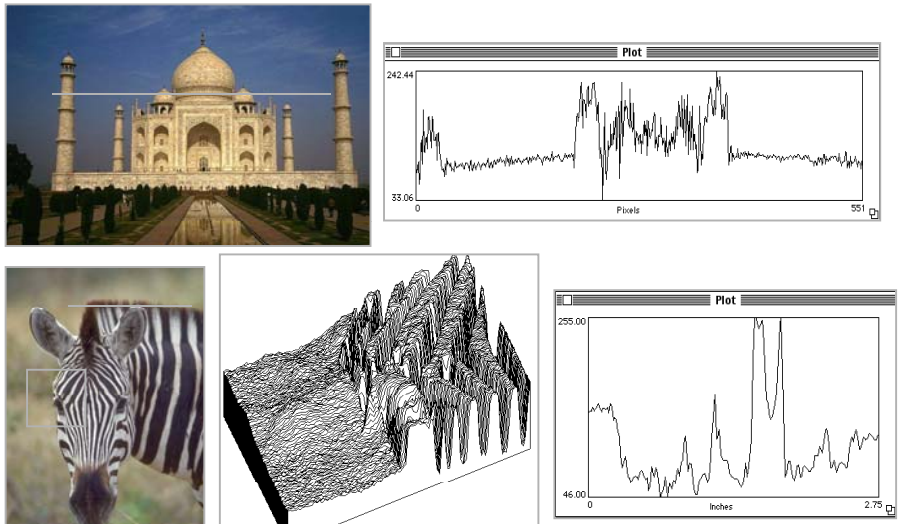


- Edge Position or Center
  - image position at which edge is located (usually saved as binary image)
- Edge Strength / Magnitude
  - related to local contrast or gradient – how rapid is the intensity variation across the edge (along the edge normal).

Edge Degradation by Noise



Real Image



Real Image

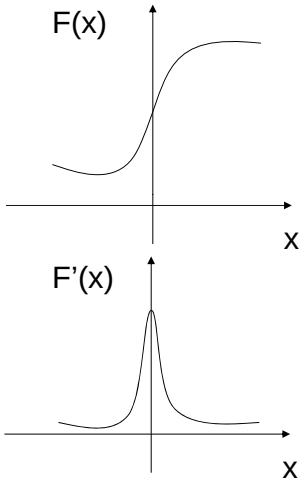
Edge Detection: typical steps

- 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

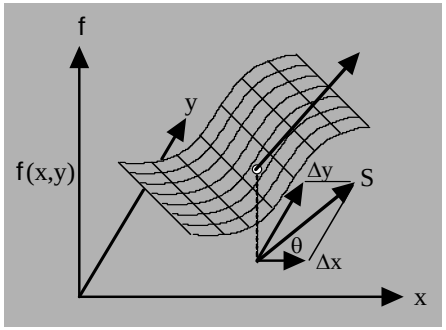
$$\Delta_x = \frac{\partial f}{\partial x}, \quad \Delta_y = \frac{\partial f}{\partial y}$$

- 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

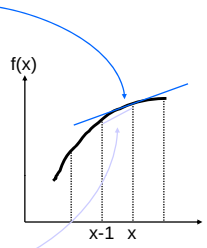


- But
  - I(i,j) is not a continuous function.
- Therefore
  - look for discrete approximations to the gradient.

Discrete Approximations

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

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

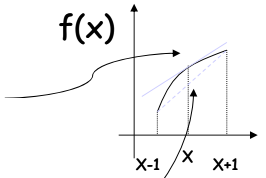


Convolve with

-1	1

Digital approximation

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



Convolve with:

-1	1

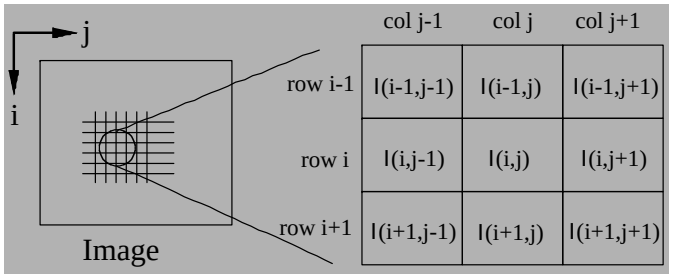
$$\frac{df(x)}{dx} \cong \frac{f(x+1) - f(x-1)}{2}$$

Convolve with:

-1	0	1

In Two Dimensions

- Discrete image function I



- Derivatives ⇨ Differences

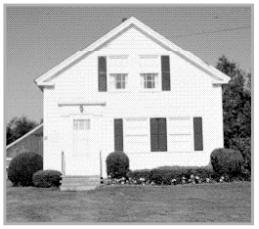
$\Delta_j I =$

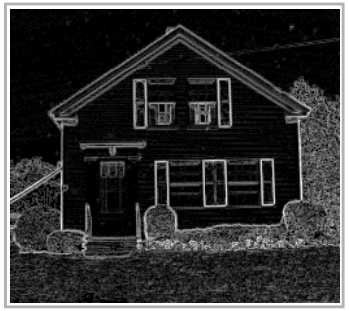
-1	1

$\Delta_i I =$

-1
1

1x2 Example





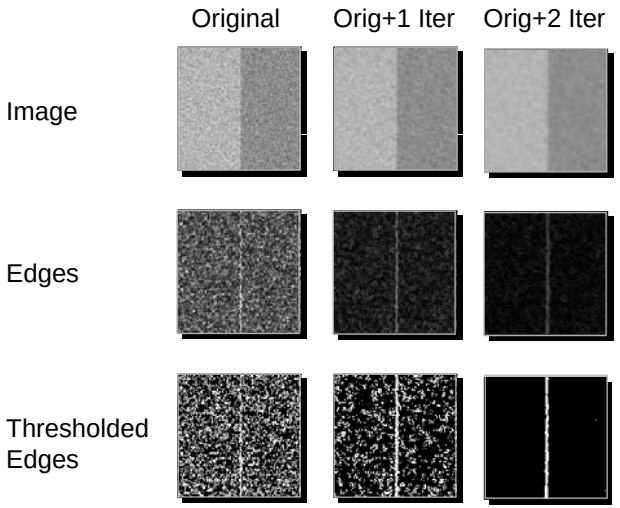
Smoothing and Edge Detection

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

$1/9 \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ $1/8 \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$

- Combine smoothing with edge detection.

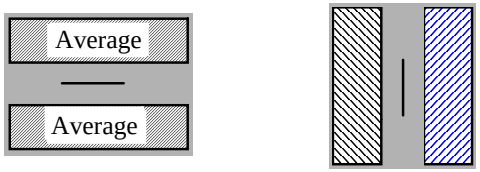
Effect of Blurring



Combining the Two

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

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



Many Different Kernels

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

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

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

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

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

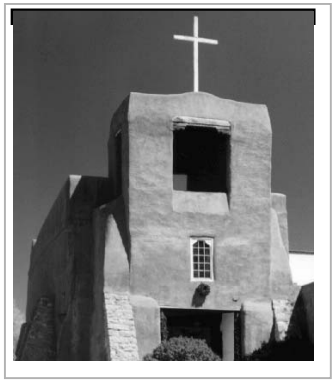
Prewitt Operator

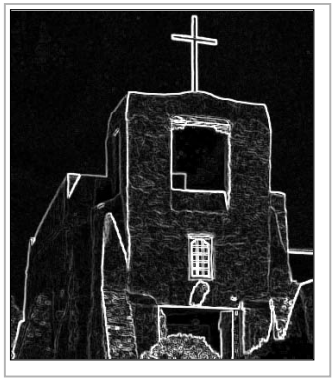
$$P_1 = \begin{array}{|ccc|} \hline -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \\ \hline \end{array} \qquad P_2 = \begin{array}{|ccc|} \hline -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \\ \hline \end{array}$$

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

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

Prewitt Example





Prewitt
horizontal and vertical edges
combined

Sobel Operator

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

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

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

Anatomy of the Sobel

1/4

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

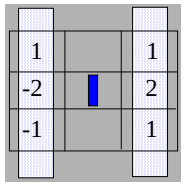
$= 1/4 * [-1 \;\; 0 \;\; -1] \circledast \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$

1/4

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

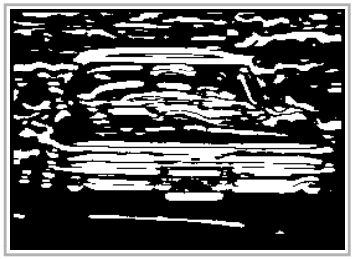
$= 1/4 * [\, 1 \;\; 2 \;\; 1] \circledast \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$

Sobel kernel is separable!

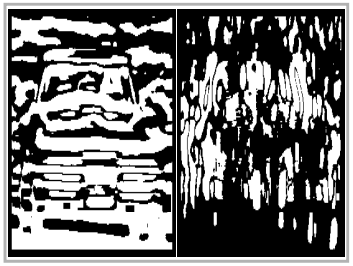


Averaging done parallel to edge

Large Kernels



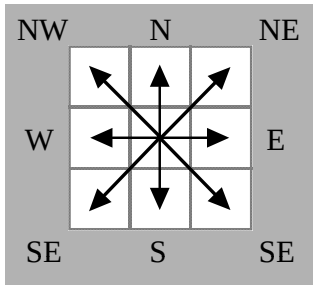
7x7 horizontal edges only

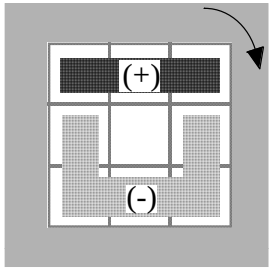


13x13 horizontal edges only

Compass Masks

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





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

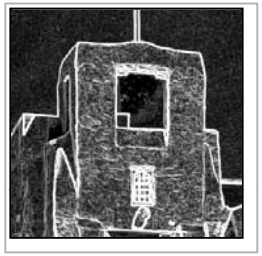
0	-1	-2
-1	0	-1
2	1	0

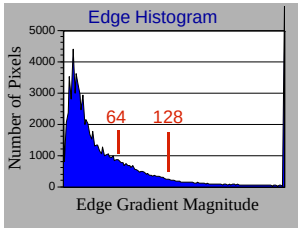
-1	-2	-1
0	0	0
1	2	1

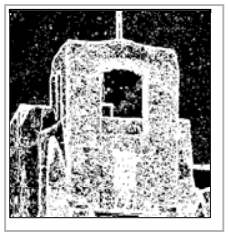
-2	-1	0
-1	0	1
0	1	2

Edge Thresholding

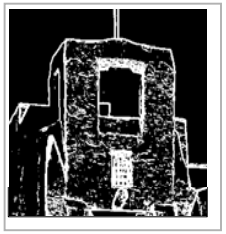
- Global approach







T=64



T=128

Canny Edge Detector

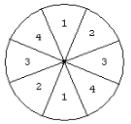
- Edge detection involves 3 steps:
  - Noise smoothing
  - Edge enhancement
  - Edge localization
- J. Canny formalized these steps to design an optimal edge detector
  - LF. Canny, "A computational approach to edge detection", IEEE Trans. Pattern Anal. Machine Intelligence (PAMI), vol. PAMI vii-g, pp. 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.
  - Single response. Only one response to a single edge.
- Probably most widely used edge detector

Canny Edge Detector

- Stage 1. Image Smoothing
  - The image data is smoothed by a Gaussian function of width specified by the user parameter.
- Stage 2. Differentiation
  - The smoothed image, retrieved at Stage 1, is differentiated with respect to the directions x and y.
  - From the computed gradient values x and y, the magnitude and the angle of the gradient can be calculated

Canny Edge Detector

- Stage 3. Non-Maximum Suppression
  - The edges can be located at the points of local maximum gradient magnitude.
  - It is done via suppression of non-maxima, that is, points, whose gradient magnitudes are not local maxima.
  - Non-maxima perpendicular to the edge direction, rather than those in the edge direction, have to be suppressed,
  - The algorithm starts off by reducing the angle of gradient to one of the four sectors shown in Figure.
  - The algorithm passes the 3x3 neighborhood across the magnitude array. At each point the center element of the neighborhood is compared with its two neighbors along line of the gradient given by the sector value.
  - If the central value is non-maximum, that is, not greater than the neighbors, it is suppressed.



Canny Edge Detector

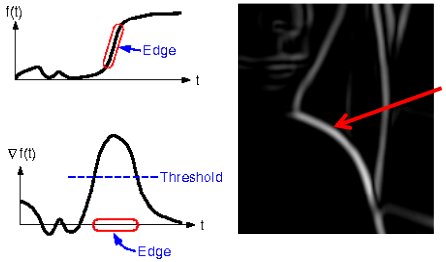
- Stage 4. Edge Thresholding
  - The Canny operator uses the so-called "hysteresis" thresholding.
  - Most thresholders use a single threshold limit, which means that if the edge values fluctuate above and below this value, the line appears broken.
    This phenomenon is commonly referred to as "streaking".
  - Hysteresis counters streaking by setting an upper and lower edge value limit.
  - Considering a line segment,
    - If a value lies above the upper threshold limit it is immediately accepted.
    - If the value lies below the low threshold it is immediately rejected.
    - Points which lie between the two limits are accepted if they are connected to pixels which exhibit strong response.
  - The likelihood of streaking is reduced drastically since the line segment points must fluctuate above the upper limit and below the lower limit for streaking to occur.
  - J. Canny recommends in [Canny86] the ratio of high to low limit to be in the range of two or three to one, based on predicted signal-to-noise ratios.

Canny Edge Detector

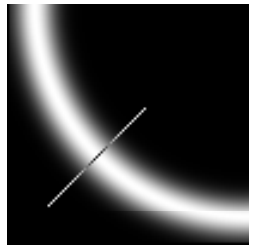
Original image

Gradient magnitude

Thresholding



How to turn these thick regions of the gradient into curves?



Non-maximum supression

Hysteresis thresholding

Problem: pixels along this edge didn't survive the thresholding

Hysteresis thresholding

- Use a high threshold to start edge curves, and a low threshold to continue them.

original image

high threshold (strong edges)

low threshold (weak edges)

hysteresis threshold

Canny Results

σ=1, T2=255, T1=1

- Hysteresis threshold method allows to add weaker edges (those above *T1)* *if they are neighbors of* stronger edges (those above *T2).*

Canny Results

σ=1, T2=255, T1=220

σ=1, T2=128, T1=1

σ=2, T2=128, T1=1

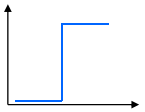
M. Heath, S. Sarkar, T. Sanocki, and K.W. Bowyer, "A Robust Visual Method for Assessing the Relative Performance of Edge-Detection Algorithms" IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 19, No. 12, December 1997, pp. 1338-1359.

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

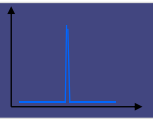
Edges from Second Derivatives

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

f(x) = step edge



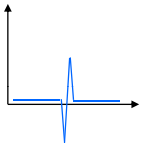
1st Derivative f'(x)



maximum

GRADIENT METHODS

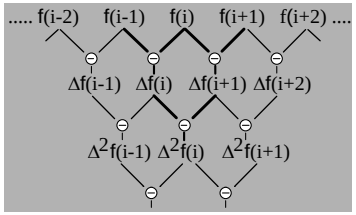
2nd Derivative f''(x)



zero crossing

Second Derivatives

- Second derivative = rate of change of 1st derivative.
- Maxima of first derivative = zero crossings of 2nd 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

Laplacian Operator

- Now consider a two-dimensional function f(x,y).
- 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 possible approximations:

	1	
1	-4	1
	1	

1	1	1
1	-8	1
1	1	1

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

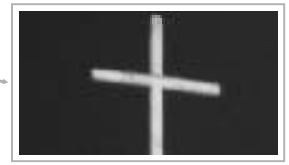
- Note that these are <u>not</u> the optimal approximations to the Laplacian of the sizes shown.

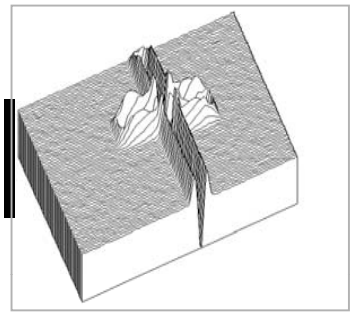
Example Application

5x5 Laplacian filter

9x9 Laplacian filter

Detailed View of Results





Computer Vision — Interpretation of the Laplacian

- Consider the definition of the discrete Laplacian:

$$\nabla^2 I = I(i+1,j)+I(i-1,j)+I(i,j+1)+I(i,j-1) - 4I(i,j)$$

looks like a window sum

- 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?*

Computer Vision — 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

Computer Vision — Laplacian Enhancement

Blurred original

3x3 Laplacian enhanced

Computer Vision — Observations on Enhancement

- Set of general techniques
- Varied goals:
  - enhance perceptual aspects of an image for human observation
  - preprocessing to aid a vision system achieve its goal
- Techniques tend to be simple, ad-hoc, and qualitative
- Not universally applicable
  - results depend on image characteristics
  - determined by interactive experimentation
- Can be embedded in larger vision system
  - selection must be done carefully
  - some techniques introduce artifacts into the image data
- Developing specialized techniques for specific applications can be tedious and frustrating.

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

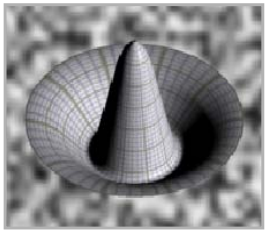
- Thus, can take Laplacian of the Gaussian (LoG) and use that as the operator.

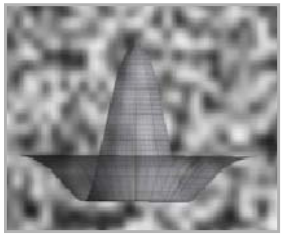
Mexican-Hat Filter

- Laplacian of the Gaussian

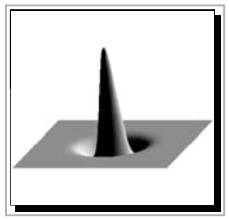
$$\nabla^2 G(x,y) = \frac{-1}{\pi\sigma^4}\left[1 - \frac{(x^2+y^2)}{2\sigma^2}\right] e^{-\left[\frac{(x^2+y^2)}{2\sigma^2}\right]}$$

- $\nabla^2 G$ is a circularly symmetric operator.
- Also called the hat or Mexican-hat operator.
- Sometimes approximated as the difference of 2 Gaussians with different sigma values

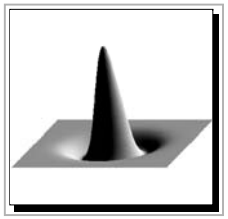




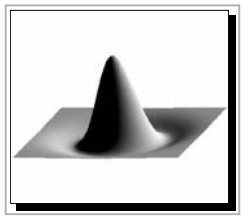
σ^2 Controls Size



$\sigma^2 = 0.5$



$\sigma^2 = 1.0$



$\sigma^2 = 2.0$

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	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	-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	0	0	0	0	

Example

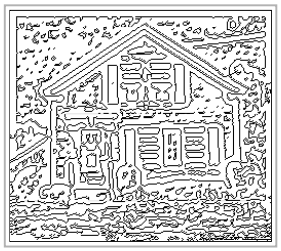
13x13 Kernel

Example

13 x 13 Hat filter

Thesholded Positive

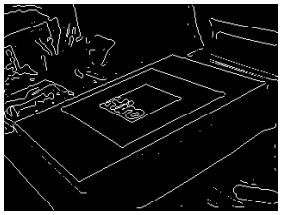
Thesholded Negative



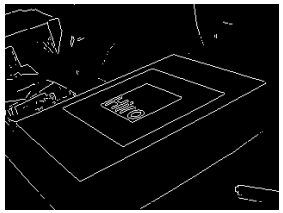
Zero Crossings

Edge Detectors

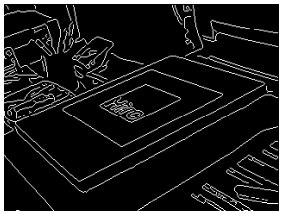
Original image



LoG



Sobel



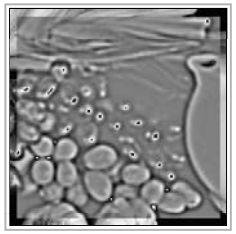
Canny

(detection thresholds automatically chosen by Matlab functions)

Scale Space

- Gaussian filter
  - the amount of blurring depends on the standard deviation (σ)
  - there is a trade-off between noise removal (better for larger σ), and edge enhancement (better for smaller σ).
  - small filters result in too many noise points and large filters tend to dislocate the edges.
- Marr and Hildreth filter
  - analyse the behavior of edges at **different scales of filtering**
  - combine the information from different scales:
  - *"If a zero-crossing segment is present in a set of independent $\nabla^2 G$ channels (scales) over a continuous range of sizes and the segment has the same position and orientation in each channel, then the set of such zero crossing segments may be taken to indicate the presence of an intensity change in the image that is due to single physical phenomenon (a change in reflectance, illumination, depth or surface orientation".*
- Other conclusions (by other authors)
  - The location of a zero crossing in the Gaussian filtered image tends to its true location as $\sigma \rightarrow 0$.
  - No new zero crossing are introduced as σ increases.

Scale Space



17x17 LoG Filter

Thresholded Positive

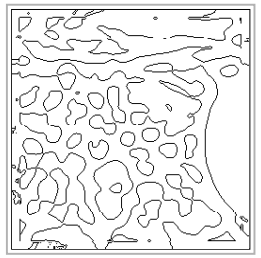
Thresholded Negative

Zero Crossings

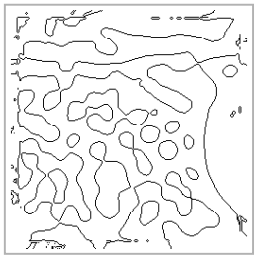
Scale Space

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

$\sigma^2 = 2$

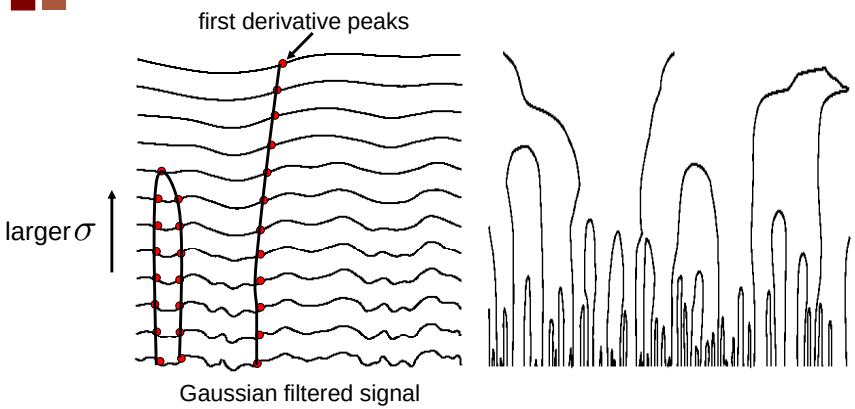


$\sigma^2 = 2\sqrt{2}$



$\sigma^2 = 4$

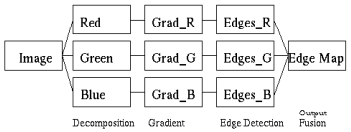
Scale Space



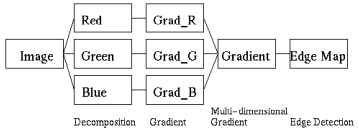
- Properties of scale space (w/ Gaussian smoothing) – Witkin'83
  - edge position may shift with increasing scale (σ)
  - two edges may merge with increasing scale
  - an edge may ***not*** split into two with increasing scale

Color Edge Detection

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



    - may produce poor results
      (see C.Tomasi at al., Bilateral Filtering for Gray and Color, ICCV 1998)
  - Multi-dimensional gradient methods



Corner detection

- Basic idea:
  - find points where two edges meet—
    i.e., high gradient in orthogonal directions
- **Harris corner detector**
  - Allows the calculation of the "strength of a corner", λ_p, for each pixel, *p*, of an image window
  - The calculation is based on the gradient inside the window

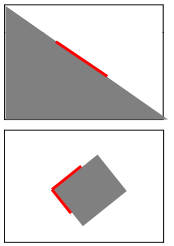
– window gradient = $[E_x\ E_y]^T$,
where E_x E_y are the spatial derivatives
  - $E_x = \partial E / \partial x$; $E_y = \partial E / \partial y$
  - calculated using, for example, Sobel filters

$$C = \begin{bmatrix} \sum E_x^2 & \sum E_x E_y \\ \sum E_x E_y & \sum E_y^2 \end{bmatrix}$$

– *p* – image pixel
– *Q* – neighborhood of (2*N*+1) x (2*N*+1) *pixels*, around *p*
– *C* – the following matrix, calculated in the neighborhood *Q* of *p*

Harris corner detector

- Matrix *C* caracterizes the graylevel "structure".
- Using the eigenvalues and eigenvectors of *C*
  it is possible to detect the existence of corners in the windows
  – let λ_1 and λ_2 be the eigenvalues of *C* (both positive)
  – if the intensity of the pixels inside the window is approximately constant, then
    - $\lambda_1 \approx \lambda_2 \approx 0$
  – if the window contains an ideal step edge
    (transition black↔white) we have:
    - $\lambda_1 > 0$ and $\lambda_2 = 0$
    - eigenvector associated with λ_1 is parallel to the gradient (perpendicular to the edge)
  – if the window contains a corner, then:
    - $\lambda_1 >= \lambda_2 > 0$
    - the greater the λ's, the greater the contrast
    - eigenvectors are parallel to the gradients



- In conclusion:
  – eigenvectors of *C* codify edge orientation
  – eigenvalues of *C* codify edge magnitude

Eigenvalues and eigenvectors

(

Eigenvalues
and
Eigenvectors

Eigenvalues and eigenvectors

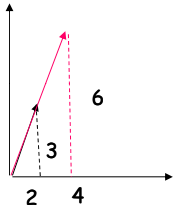
- Consider that
  – *A* is a square matrix
  – *v* is a column vector
  – λ is a scalar
- If $A.v = \lambda v$
  – assuming a non-trivial solution ($v \neq 0$)
  – *v* is an *eigenvector* of A
  – λ is an *eigenvalue* of A

(http://mathworld.wolfram.com/Eigenvalue.html)

- Example:

$$A = \begin{bmatrix} 5 & -2 \\ 6 & -2 \end{bmatrix}$$

$$v = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$



$$Av = \begin{bmatrix} 5 & -2 \\ 6 & -2 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 4 \\ 6 \end{bmatrix} = 2 \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

v is an eigenvector with eigenvalue 2

$$Av = \lambda v$$

$$Av - \lambda v = 0$$

$$(A - \lambda I)v = 0$$

homogeneous system of linear equations

v is an eigenvector if it is not the trivial solution, *v*=0;

according to Cramer's rule,
a system of linear equations has non-trivial solutions
if and only if the determinant is zero

$$det(A - \lambda I) = 0$$

- Example:

$$A = \begin{bmatrix} 5 & -2 \\ 6 & -2 \end{bmatrix}$$

$$\det(A - \lambda I) = \det \begin{vmatrix} 5-\lambda & -2 \\ 6 & -2-\lambda \end{vmatrix} = 0$$

$$(5 - \lambda)(-2 - \lambda) - (6)(-2) = 0$$

$$\lambda^2 - 3\lambda + 2 = 0 \Rightarrow \lambda = \frac{3 \pm \sqrt{9 - 8}}{2}$$

$$\lambda_1 = 1, \lambda_2 = 2$$

- Eigenvector for λ=1

$$\begin{aligned} (5-1)v_{11} - 2v_{12} &= 0 \\ 6v_{11} + (-2-1)v_{12} &= 0 \end{aligned} \quad \Rightarrow \quad \begin{aligned} 4v_{11} - 2v_{12} &= 0 \\ 6v_{11} - 3v_{12} &= 0 \end{aligned}$$

$$\Rightarrow 2v_{11} = v_{12} \Rightarrow \quad v_1 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

Eigenvalues and eigenvectors

- eigenvector for λ=2

$$\begin{aligned}(5-2)v_{21}-2v_{22} &= 0\\ 6v_{21}+(-2-2)v_{22} &= 0\end{aligned} \quad \Rightarrow \quad \begin{aligned}3v_{21}-2v_{22} &= 0\\ 6v_{21}-4v_{22} &= 0\end{aligned}$$

$$\Rightarrow 3v_{21} = 2v_{22} \Rightarrow \quad v_2 = \begin{bmatrix}2\\3\end{bmatrix}$$

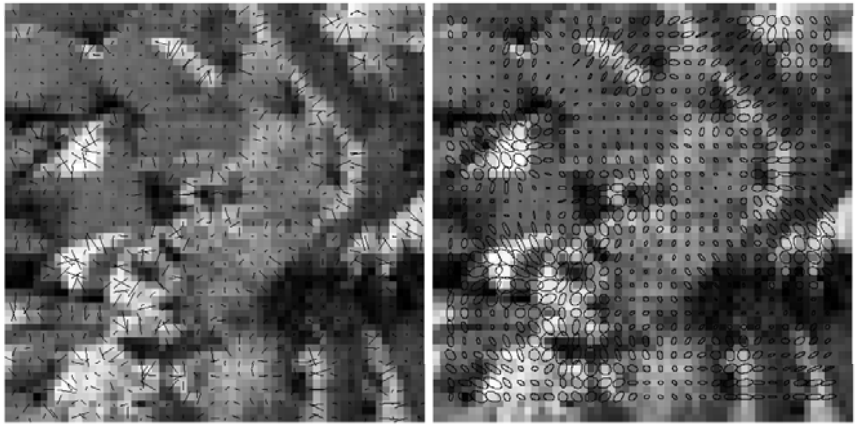
Eigenvalues and eigenvectors

)

Harris corner detector

Original Gradient

Harris corner detector

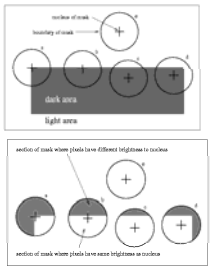


Zooming of a region of the previous image.
The ellipses indicate the eignvalues and eigenvectors of the C matrices

Corners & Feature detection

- Other corner detectors
  (http://www.ee.surrey.ac.uk/Research/VSSP/demos/corners/survey.html)
  - Moravec detector (one of the 1st ones)
    (http://rfv.insa-lyon.fr/~jolion/PAPIERS/ptint/node2.html)
  - SUSAN (Smallest Univalue Segment Assimilating Nucleus) detector
    (http://www.fmrib.ox.ac.uk/~steve/susan/)
  - Modern variants of the Harris corner detector
    (see Szeliski book)

SUSAN
- circular masks
- USANs are the white parts similar to the nucleus (cross)
- minimum values of the USANs occur at the corners



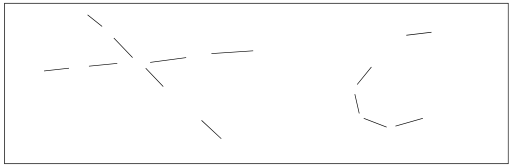
- **Feature detectors** (to be analyzed later)
  - SIFT - Scale Invariant Feature Transform
  - SURF- Speeded Up Robust Features
    (a variant of SIFT)

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…
  finding edges from lines

From Edgels to Lines

- Given local edge elements:



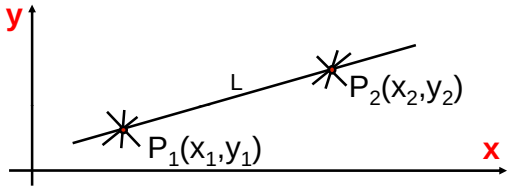
- Can we organize these into more 'complete' structures, such as straight lines?
- Group edge points into lines?
- Consider a fairly simple technique...

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

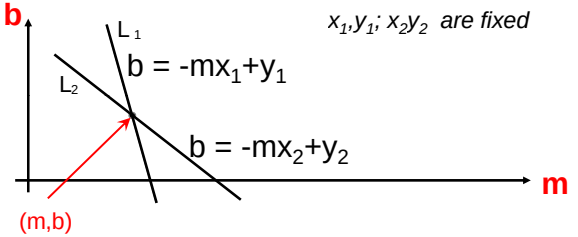
- Consider two (edge) points, P(x,y) and P'(x',y') in image space:



- The set of all lines through P=(x,y) is **y**=m**x** + 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
  - **b** = – x**m** + y

Parameter Space

- The intersection represents the parameters of the equation of a line y=mx+b going through both (x1,y1) and (x2,y2).



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

General Idea

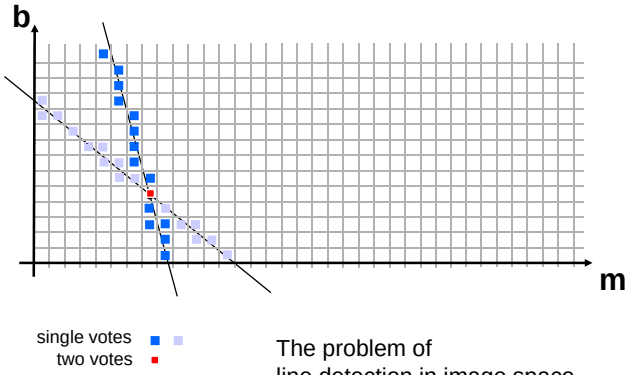
- General idea:
  - The **Hough space (m,b)** is a representation of every possible line segment in the plane
  - Make the Hough space (**m** and **b**) discrete
  - Let every edge point in the image plane 'vote for' any line it might belong to.

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

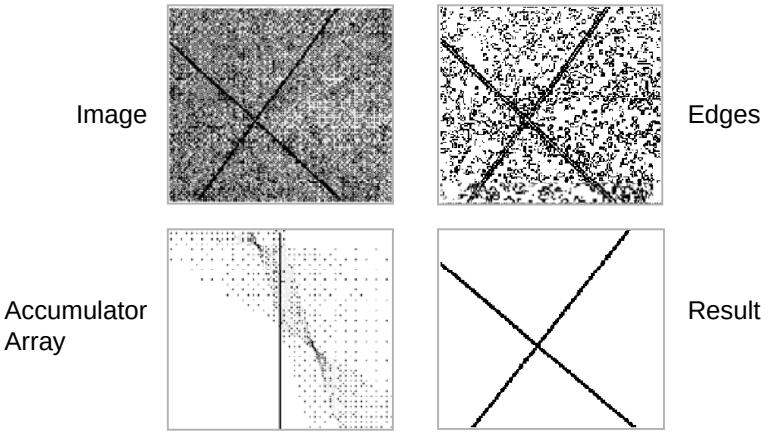
- Quantization



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

Example

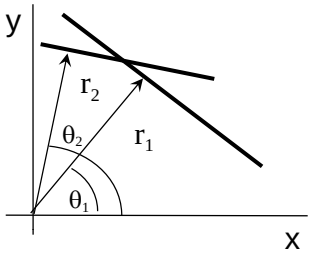
- The problem of line detection in image space has been transformed into the problem of cluster detection in parameter space



Parameter Space (r,θ)

- Problem: vertical lines have infinite slopes
  - difficult to quantize **m** to take this into account.
- Solution: use alternative parameterization of a line
  - polar coordinate representation:

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



$r_1 = x \cos\theta_1 + y \sin\theta_1$

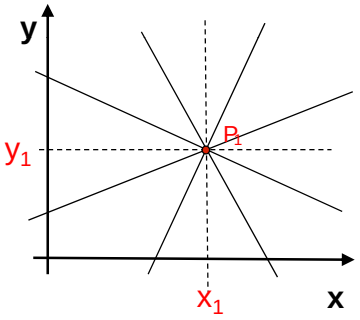
$r_2 = x \cos\theta_2 + y \sin\theta_2$

Parameter Space (r,θ)

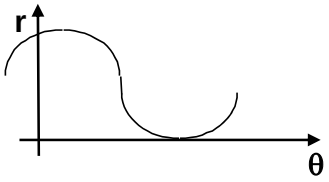
- (r,θ) is an efficient representation:
  - Small: only two parameters (like y=mx+b)
  - Finite: $0 \le r \le \sqrt{(row^2+col^2)}$, $0 \le \theta \le 2\pi$
  - Unique: only one representation per line

Representation of a point

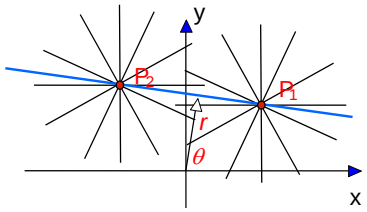
- A point P1=(x1,y1) in image space is represented by a sinusoid in (r,θ) space



$r = x_1 \cos\theta + y_1 \sin\theta$



Example

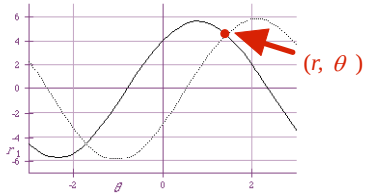


$P_1 = (4,4) \rightarrow r = x_1 \cos\theta + y_1 \sin\theta = 4\cos\theta + 4\sin\theta$

$P_2 = (-3,5) \rightarrow r = x_2 \cos\theta + y_2 \sin\theta = -3\cos\theta + 5\sin\theta$

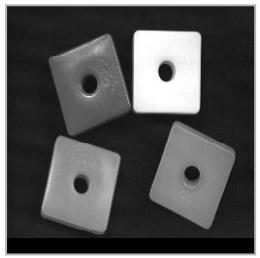
2 eq.s on (r ,θ)
Possible to solve for (r ,θ)

(r, θ) Space



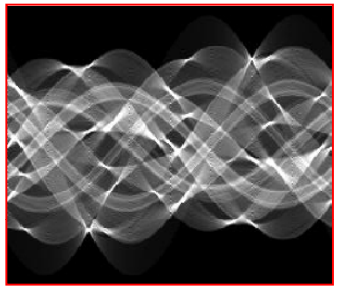
Real Example

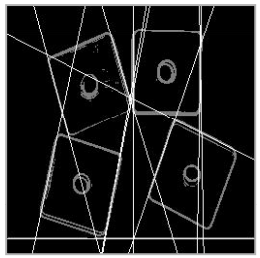
Image



Edges

Accumulator Array

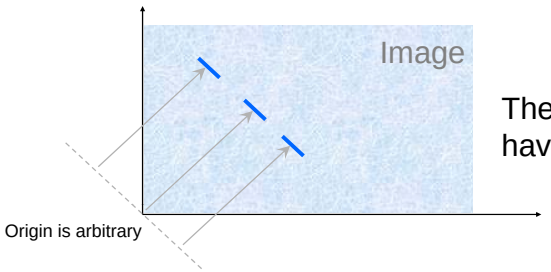




Result

Hough Transform Speed up

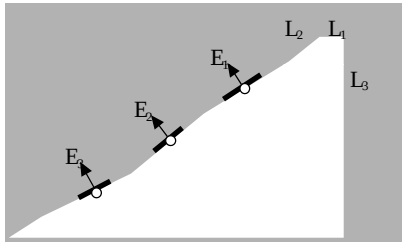
- If we know the orientation of the edge – usually available from the edge detection step
- We fix θ in the parameter space and increment only one counter!
- We can allow for orientation uncertainty by incrementing a few counters around the “nominal” counter.

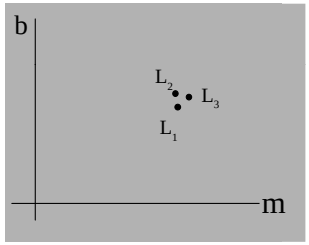


The three edges have same (r, θ)

Gradient Data

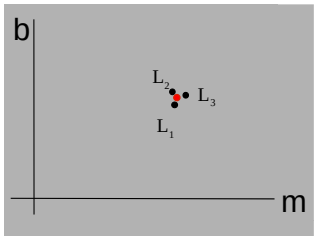
- Colinear edges in Cartesian coordinate space form point clusters in (m,b) parameter space.



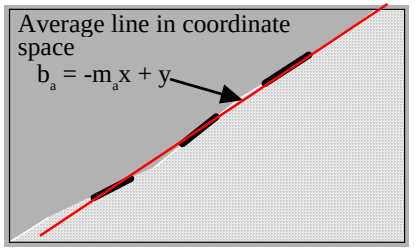


Gradient Data

- 'Average' point in Hough Space:

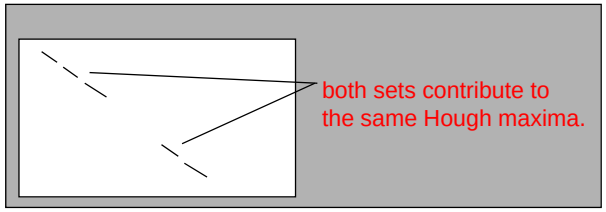


- Leads to an 'average' line in image space:



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:

$f(\underline{x},\underline{a}) = 0$

($\underline{a}$: parameter vector (axes in Hough space))

- 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

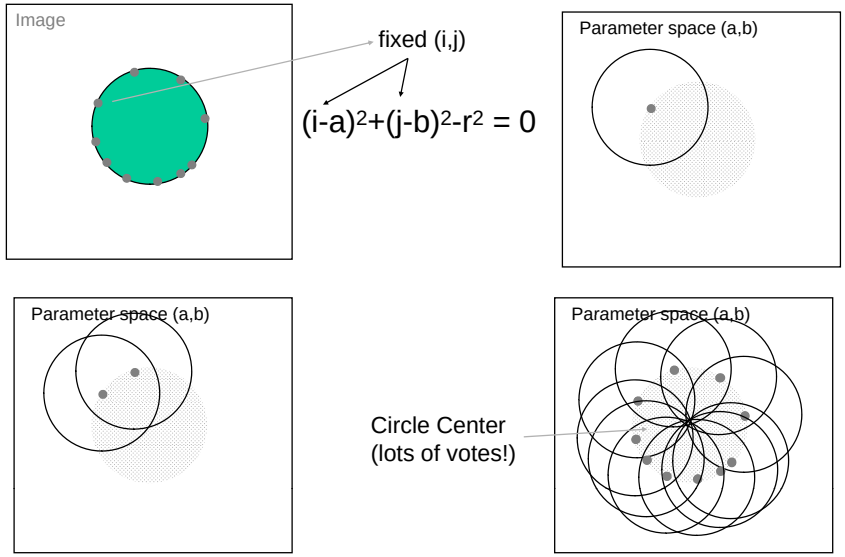
Example: Finding a Circle

- 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?

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,
    CVGIP, 44(1):87-116, 1988

Hough Transform - Summary

- Hough Transform is a “voting” scheme
  - points vote for a set of parameters describing a line or curve.
- The more votes for a particular set
  - the more evidence that the corresponding curve is present in the image.
- Can detect MULTIPLE curves in one shot.
- Computational cost increases with the number of parameters describing the curve.