Feature Extraction

Image features represent dominant and distinctive image regions. They can be used to characterize the shape or appearances of the image or the objects in the images. Features include photometric and geometric features. Features are widely used by many different computer vision tasks including camera calibration, stereo, object detection, tracking, recognition, and 3D reconstruction.

Feature Extraction (cont'd)

Image features can be divided into geometric features and appearance features. They include

- Geometric features: linear features like edges, point features, including the Scale Invariant Feature Transform (SIFT), corners, and junctions , and complex shape features like lines and curves. They characterize object shape information.
- Appearance features: Local Binary Pattern (LBP), Histogram of Gradients (HOG) features, and Gabor features. They capture object reflectance, materials, illumination, etc.. properties.

These features are collectively called hand-crafted features. The latest developments are to learn feature representations using a deep learning model. The learnt

features are better than hand-crafted features.

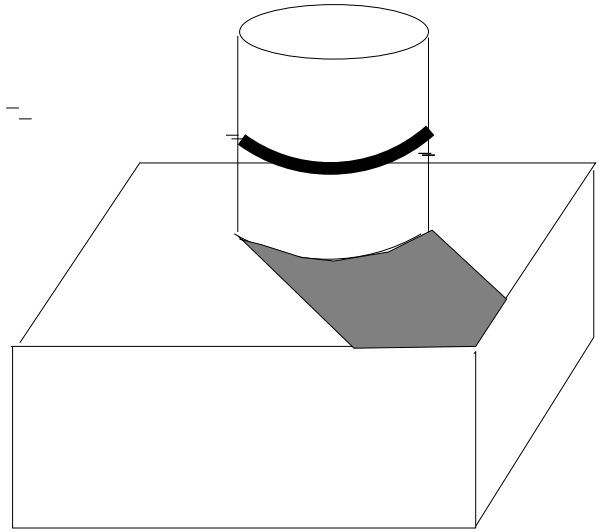
Edge Detection

An edge is defined to be a sequence of pixels that border two homogeneous regions of different intensities. It can concisely represent an object's shape. An edge indicates discontinuity in image intensity function. Edge detection amounts to locating each edge pixel (also referred to as edgel) and determining its orientation and strength.

Example

Edges may be generated from different physical sources. But the resultant edges all exhibit various degrees of discontinuities in image intensity.

Physical Edge Sources



- different surface properties

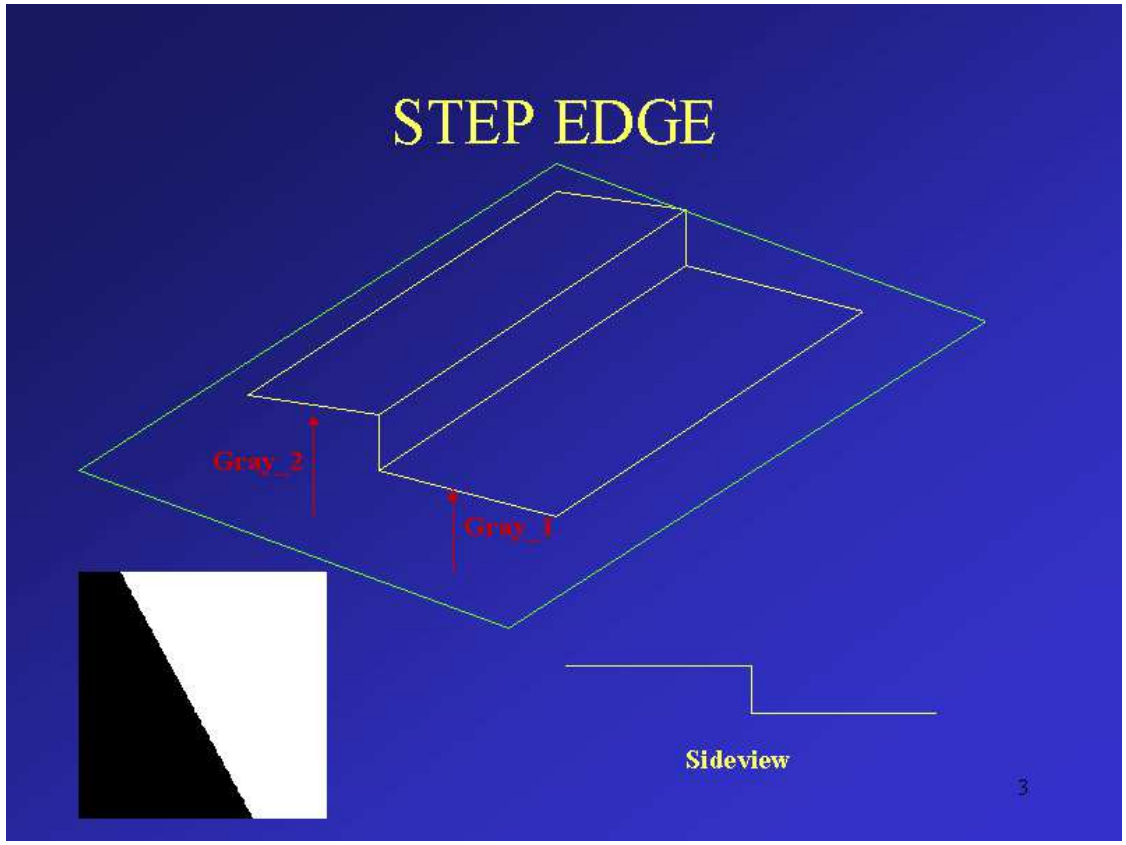
  color, reflective properties, texture

- discontinuity in distance and surface orientation

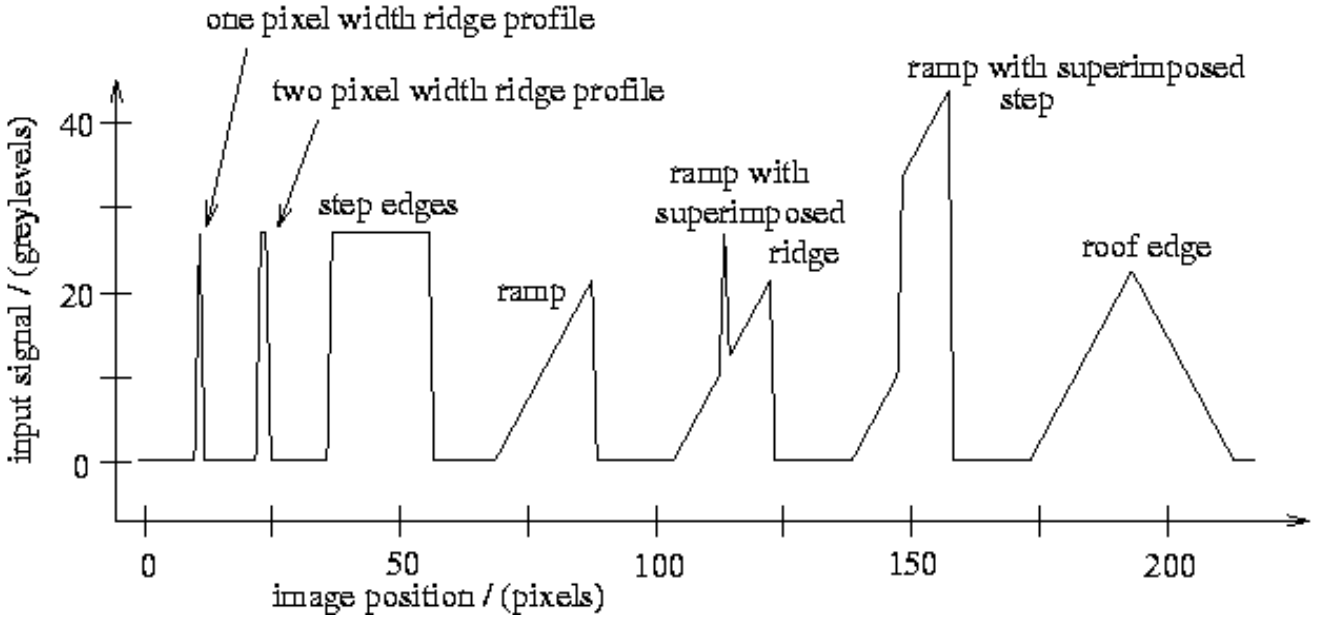
- shadows

Edge Detection Is Difficult

- Different types of edges: step, ridge, ramp, roof edge
- Subject to image noise (intensity noise and spatial quantization errors)

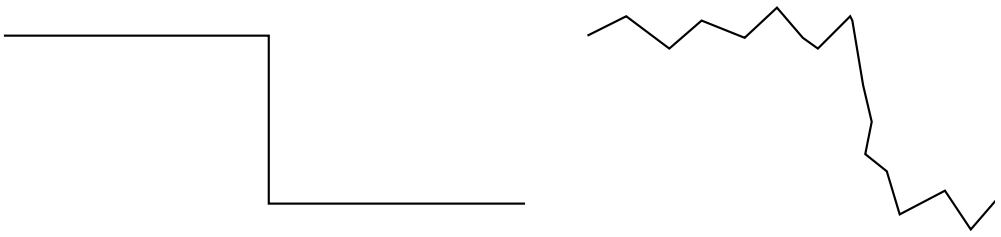

STEP EDGE
Gray_2
Gray_1
Sideview
3

Edge Detection Is Difficult (cont'd)



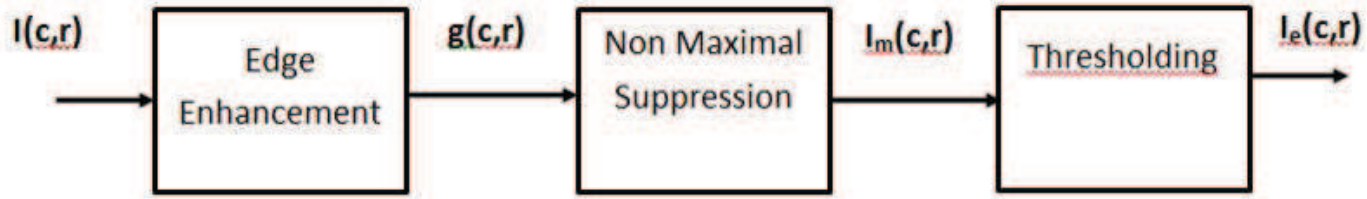
perturbed ste

ideal step edge



see figure 4.2

Edge Detection Paradigm



Edge Enhancement

Edge enhancement is often formulated a filtering processing by enhancing the high frequency components in the image. Design a filter and convolve it with the image. The result of convolution should be large at edge pixels and low elsewhere.

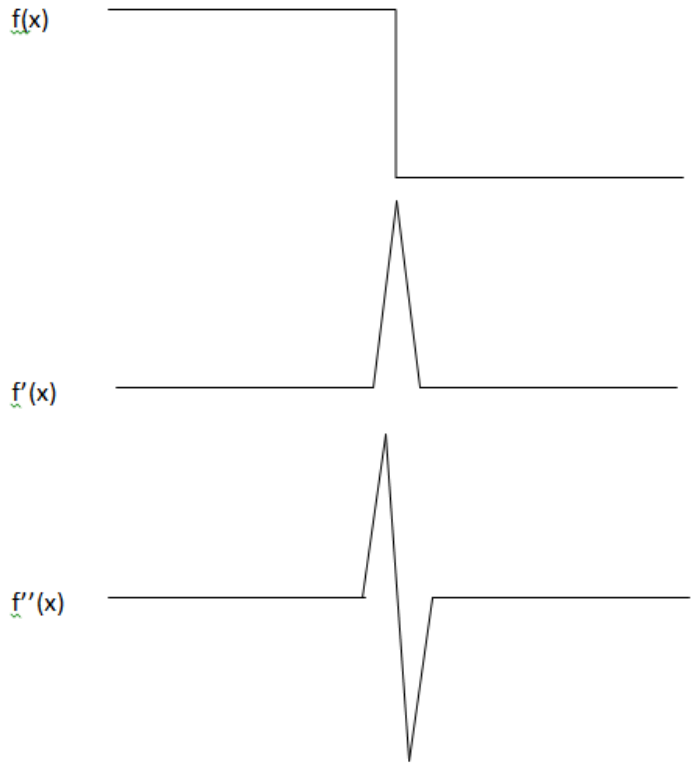
Non-Maximal Suppression

Remove edge candidates that are not locally maximal.

Edge Thresholding

Based on the response of each pixel from edge enhancement, decide which pixel is edge pixel and which is non-edge pixel, yielding a binary edge map.

Edge Enhancement



- First order edge enhancer
- Second order edge enhancer

First Order Edge Filter

Let I(c,r) be the image intensity at pixel (c,r), then the first order image gradient can be approximated by

$$\bigtriangledown I(c,r) = \begin{pmatrix} \frac{\partial I(c,r)}{\partial c} \\ \frac{\partial I(c,r)}{\partial r} \end{pmatrix}$$

where the partial derivatives are numerically approximated via (see Appendix 2)

$$\begin{aligned} \frac{\partial I}{\partial c} &= I(c+1,r) - I(c,r) \\ \frac{\partial I}{\partial r} &= I(c,r+1) - I(c,r) \end{aligned}$$

Gradient magnitude and direction are defined as

$$
\begin{aligned}
g(c,r) &= \sqrt{(\frac{\partial I(c,r)}{\partial c})^2 + (\frac{\partial I(c,r)}{\partial r})^2} \\
\theta &= \arctan \frac{\frac{\partial I(c,r)}{\partial r}}{\frac{\partial I(c,r)}{\partial c}}
\end{aligned}
$$

where θ represents the steepest slope direction.

This leads to two first order image derivatives filters

$$h_x = [-1 \quad 1] \qquad h_y = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

where h_x computes the first order horizontal image derivative and h_y computes the first order vertical image derivative.

If we take first order derivatives in 45 and 135 degrees, we obtain the Roberts edge operator:

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

Note the image gradients computed using Roberts operator differ from those using h_x and h_y in gradient direction (by 45 degree).

Classical First Order Edge Filters

In order to be more robust to noise, differentiation must be performed on a smoothed version of the image. Therefore, edge detection is often preceded by a smoothing operation, i.e., $(I * s) * h$ If an image is smoothed with the impulse response (averaging smoothing)

$$s = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

and the resulting image is then convolved with a first order horizontal derivative h_x, we have the classical vertical Prewitt operator, because of **the associative property** of convolution,

i.e., $(I * s) * h = I * (s * h)$

Prewitt

$$s * h_x = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

The horizontal Prewitt operator

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

can be generated in a similar fashion .

If the image is smoothed with the impulse response (weighted averaging)

$$s = \begin{bmatrix} 1 & 1 \\ 2 & 2 \\ 1 & 1 \end{bmatrix}$$

is then convolved with h_y, we have the horizontal Sobel operator

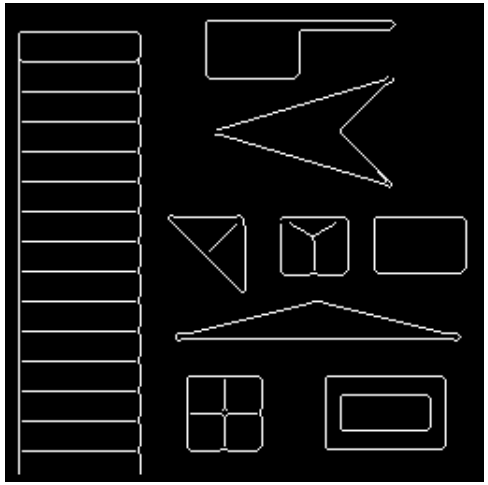
$$s * h_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

The vertical Sobel operator can be generated in a similar fashion

$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

If the image is smoothed with the Gaussian response (Gaussian averaging) and the smoothed image is then convolved with difference operators (h_x and h_y), this produces the well-known Canny edge detector (named after John Canny). Also called Difference of Gaussian or DoG filter, it is created by taking the first order derivative of the Gaussian filter.

An Example of Edge Detection



Edgel Descriptors

- Edge direction (perpendicular to gradient direction see figure 4.3)
- Edge position (specified by the pixel position (c,r))
- Edge strength (gradient magnitude)

Gradient-based Facet Edge Detector

The gray levels in a small neighborhood centered at pixel (c, r) on the image is approximated by a sloped facet model $\alpha c + \beta r + \gamma = 0$, where $\hat{\alpha}$ and $\hat{\beta}$ are estimates of α and β based on a least-squares fitting, i.e.,

$$\begin{aligned} \hat{\alpha} &= \frac{\sum_r \sum_c cI(c,r)}{\sum_r \sum_c c^2} \\ \hat{\beta} &= \frac{\sum_r \sum_c rI(c,r)}{\sum_r \sum_r r^2} \end{aligned}$$

Then, the gradient at this point is

$$\begin{aligned} g(c,r) &= \sqrt{\hat{\alpha}^2 + \hat{\beta}^2} \\ \theta &= \arctan \frac{\hat{\beta}}{\hat{\alpha}} \end{aligned}$$

Design a First Order Edge Filter

A good edge kernel should yield accurate edge gradient (both magnitude and direction) estimation. Assume the gray level in a 3 x 3 neighborhood can be described by a simple linear relationship $I(c, r) = \alpha c + \beta r + \gamma$. Design a 3×3 edge detector that maximizes the accuracy of the estimated image gradient, i.e. find the kernel coefficients a and b by minimizing the variance of convolution output.

Design a First Order Edge Filter (cont'd)

Let a filter be parameterized by

$$\begin{bmatrix} -a & -b & -a \\ 0 & 0 & 0 \\ a & b & a \end{bmatrix}$$

Noise model

$$I(c,r) = \alpha c + \beta r + \gamma + \epsilon(c,r)$$

where $\epsilon(r,c)$ is independent noise having mean 0 and variance $\sigma^2(c,r)$

Given the above parameterized filter, the noise model, and the linear facet model, find a and b that minimizes the variance of the estimated gradient. Using different noise models, the

minimization leads to the Sobel or Prewitt filter (see Haralick's book pages 341 and 342, Vol. 1).

The Optimal First Order Filter

- High SNR for noisy step edge (assume noise is white and additive)
- Good location accuracy
- Single response

The Optimal First Order Filter (cont'd)

We model the ideal 1-D step edge as

$$f(x) = \begin{cases} 0 & x < 0 \\ A & x \geq 0 \end{cases}$$

The ideal step edge is perturbed by a white noise $n(x)$, generating the observed step edge $\hat{f}(x)$

$$\hat{f}(x) = f(x) + n(x)$$

The response of a linear filter h to the observed step edge f is

$$\int_{W}^{-W} \hat{f}(x-t)h(t)dt = A\int_{-W}^{0} h(t)dt + \int_{-W}^{W} n(x-t)h(t)dt$$

The first two criteria can be defined as

$$SNR = \frac{A||\int_{-W}^{0} h(t)dt||}{\sigma_0\sqrt{\int_{-W}^{W} h^2(t)dt}}$$

$$LOC = \frac{A||h'(0)||}{\sigma_0\sqrt{\int_{-W}^{W} h'^2(t)dt}}$$

where LOC measures the location accuracy.

The product of SNR and LOC is a measure of how well the filter h satisfies both criteria simultaneously. So we look for h that maximizes the product of SNR and LOC. A simple difference operator (e.g., h_x) maximizes the product.

The Optimal First Order Filter (cont'd)

The difference operator obtained by maximizing the product of SNR and LOC does not satisfy the third criterion, i.e., it may yield multiple local maxima for noise step edge as shown in Figure 4.4. We can obtain the optimal step edge filter by maximizing the product of SNR and LOC subject to the single response constraint.

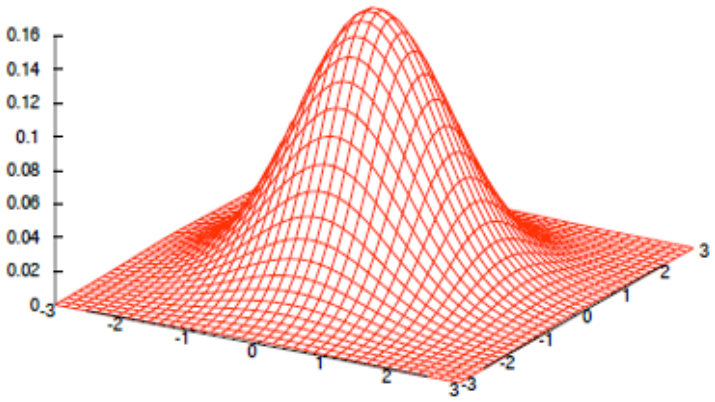
Canny Edge Enhancer

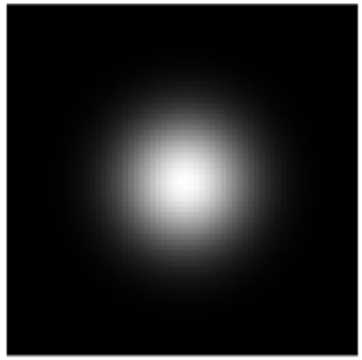
The optimal filter obtained by optimizing the above three criteria for a step edge can be approximated by the first order Derivative of a Gaussian (DoG). This is the Canny edge enhancer (see figure 4.4 and algorithm Canny_Enhancer).

$$h = \bigtriangledown G(c, r)$$

where $G =$ is a 2D Gaussian as

2D Gaussian





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

Canny Edge Enhancer (cont'd)

Taking derivatives with respect to x and y, we obtain the horizontal and vertical DoG filters as follows

$$\frac{\partial}{\partial x} G_\sigma(x,y) = \frac{-x}{2\pi\sigma^4} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

$$\frac{\partial}{\partial y} G_\sigma(x,y) = \frac{-y}{2\pi\sigma^4} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

Canny Edge Enhancer (cont'd)

Let $\sigma = 0.6$ (W=5σ), we can create 3x3 DoG filters as follows

H_x

0.0764	0.3062	0.0764
0	0	0
-0.0764	-0.3062	-0.0764

H_y

0.0764	0	-0.0764
0.3062	0	-0.3062
0.0764	0	-0.0764

Summary of First Order Edge Enhancer

The first order image intensity derivative can be used to enhance the edges in an image. Different versions of first order edge enhancer (high pass filter) have been developed:

- Roberts edge enhancer, a 2x2 filter resulted from rotating the horizonal and vertical difference operators by 45 degree.
- Prewitt edge enhancer, a 3x3 filter resulted from convolving the horizonal and vertical difference operators by an averaging smoothing
- Soble edge enhancer, a 3x3 filter resulted from convolving the horizonal and vertical difference operators by a weighted averaging smoothing
- Canny edge enhancer or Difference of Gaussian (DoG), a 3x3

filter resulted from convolving the horizonal and vertical difference operators by a Gaussian smoothing

Edge Thresholding

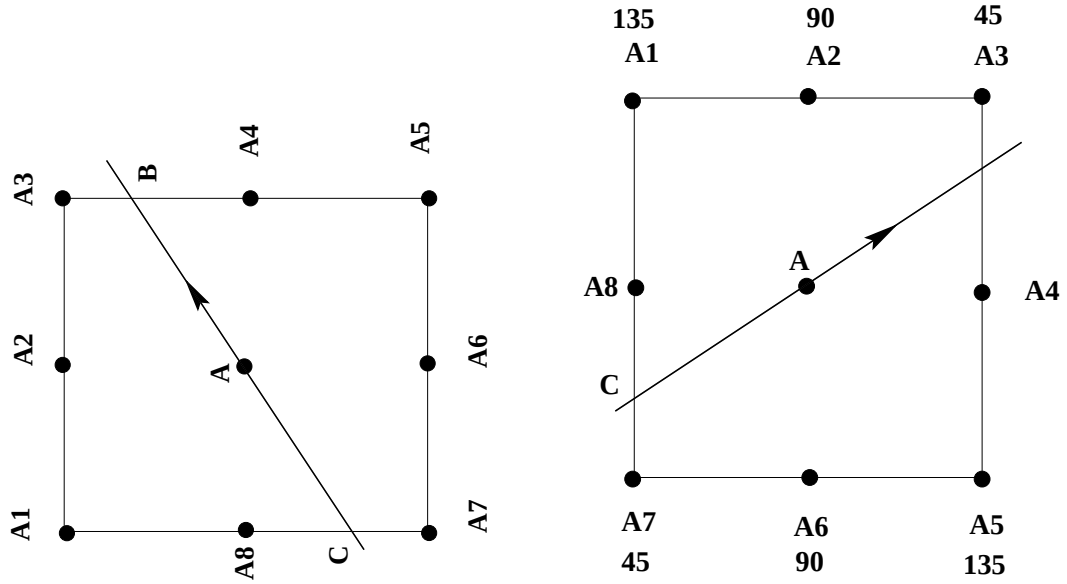
Edge thresholding identifies edge pixels from non-edge pixels based image gradients. Edge thresholding consists of two steps: non-maximum suppression and thresholding. Non-maximum suppression selects the potential candidates of edgles by identifying pixels that are local maxima in gradient magnitude. The final edge pixels are selected from the local maxima via a thresholding operation.

Non-maximum Suppression

To decide whether a pixel A is a local maximum, we need to do

- define *local* by selecting a neighborhood size
- search in the gradient direction of A within the defined neighborhood. If the magnitude of A is strictly larger than gradient magnitudes of other pixels located within the neighborhood and in the gradient direction of A, then A is marked as edgel candidate. Repeat for every pixel.

Non-maximum Suppression



The result of non-maximum supression should be a binary image with edge pixel candidates marked as white and the non-edge pixels marked as black (see Figure 4.5).

Edge Thresholding

- Simple gradient magnitude thresholding
- Thresholding using both gradient magnitude and direction (e.g. orientation consistency and Canny's hysteresis thresholding)
- A statistical approach

Edge Thresholding (cont'd)

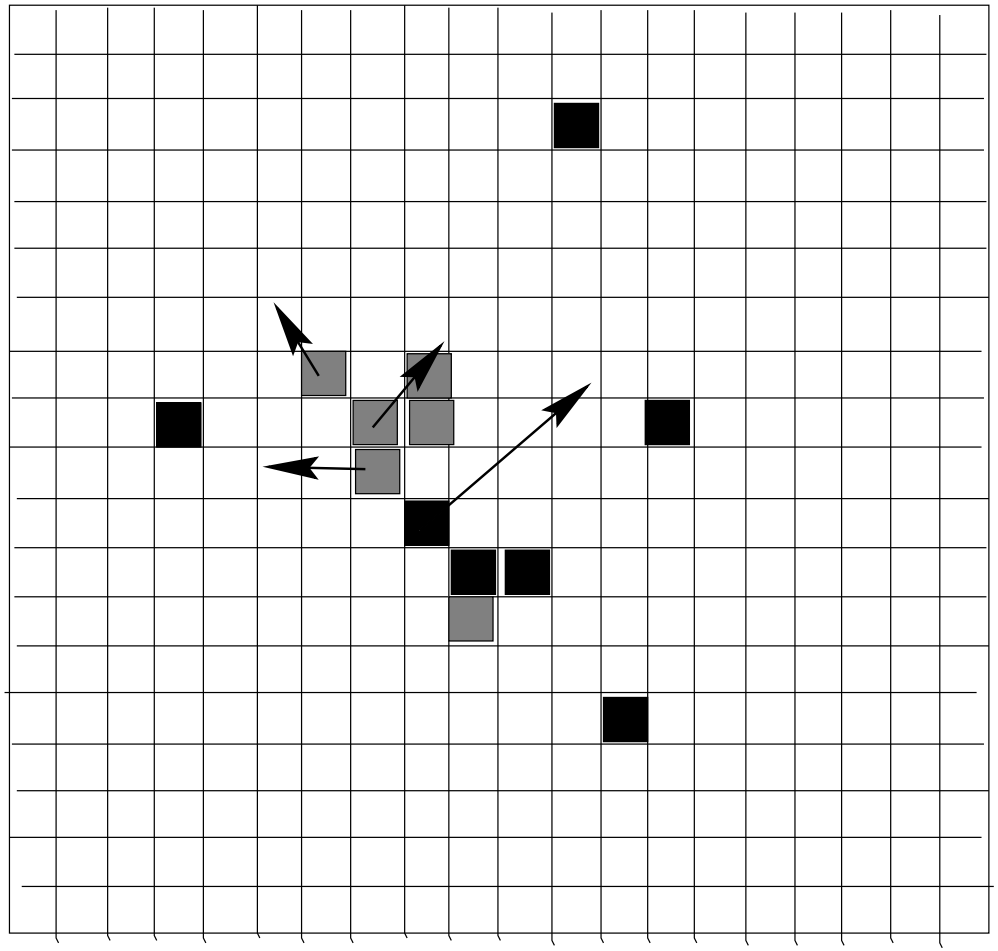
Edge detection with enforced orientation consistency is to examine every pixel labeled as edge pixel. Check each one of its eight neighbors to see if it has at least one edge-labeled neighbor whose direction is consistent with its own orientation. If so, the pixel in the output image is labeled as edge pixel. This may iterate.

Canny's Hysteresis Thresholding

The hysteresis thresholding method consists in eliminating the local maxima which are not significant. Given a high and a low thresholds T_h and T_l, we eliminate the local maxima whose gradient norm are either:

1) less than T_l; or 2) less than T_h and not connected (8-connectivity) to a local maximum whose gradient norm is greater than the high threshold.

This thresholding technique gives better results than a one level threshold because it takes into account that some contours can be "hidden" in the noise and it preserves the connectivity.



Edge Thresholding: a Statistical Approach

Based on the facet model, the gray levels in a small neighborhood centered at pixel (c, r) on the image is approximated by by a sloped facet model $\alpha c + \beta r + \gamma = 0$, the gradient at this point is

$$\begin{aligned} g(c,r) &= \sqrt{\hat{\alpha}^2 + \hat{\beta}^2} \\ \theta &= \arctan \frac{\hat{\beta}}{\hat{\alpha}} \end{aligned}$$

where $\hat{\alpha}$ and $\hat{\beta}$ are estimates of α and β based on a least-squares fitting, i.e.,

$$\begin{aligned} \hat{\alpha} &= \frac{\sum_r \sum_c cI(c,r)}{\sum_r \sum_c c^2} \\ \hat{\beta} &= \frac{\sum_r \sum_c rI(c,r)}{\sum_r \sum_r r^2} \end{aligned}$$

A Statistical Approach (cont'd)

Assume the following perturbation model

$$I(c, r) = \alpha c + \beta r + \gamma + \epsilon(c, r)$$

where $\epsilon(r, c)$ is independent noise having mean 0 and variance $\sigma^2(c, r)$

We have variances of $\hat{\alpha}$ and $\hat{\beta}$ as follows

$$\sigma^2_{\hat{\alpha}} = \frac{\sigma^2}{\sum_r \sum_r c^2}$$

$$\sigma^2_{\hat{\beta}} = \frac{\sigma^2}{\sum_r \sum_r r^2}$$

A Statistical Approach (cont'd)

Since $\hat{\alpha} \sim N(\alpha, \sigma^2_{\hat{\alpha}})$, we have

$$\frac{(\hat{\alpha} - \alpha)^2}{\sigma^2_{\hat{\alpha}}} \sim \chi^2_1$$

Similarly, we have

$$\frac{(\hat{\beta} - \beta)^2}{\sigma^2_{\hat{\beta}}} \sim \chi^2_1$$

$$\frac{(\hat{\alpha}-\alpha)^2}{\sigma_{\hat{\alpha}}^2}+\frac{(\hat{\beta}-\beta)^2}{\sigma_{\hat{\beta}}^2}\sim\chi_2^2$$

For a symmetric window, we have $\sigma_{\hat{\alpha}}^2=\sigma_{\hat{\beta}}^2=\sigma_p^2$. Thus,

$$\frac{(\hat{\alpha}-\alpha)^2+(\hat{\beta}-\beta)^2}{\sigma_p^2}\sim\chi_2^2$$

We can therefore formulate the edge detection problem as a hypothesis testing problem, i.e.,

H_0: (c,r) is not an edgel ($\alpha=\beta=0$) v.s. H_1: (c,r) is an edgel

Let $t=\frac{(\hat{\alpha})^2+(\hat{\beta})^2}{\sigma_p^2}$ and select a significant level of 0.05. If $P(\chi_2^2>t)>0.05$, then the null hypothesis is accepted, i.e., (c,r) is not an edgel or (c,r) is an edgel. We can control the false alarm rate of the edge detector by adjusting the significant level(larger significant level, larger false alarm).

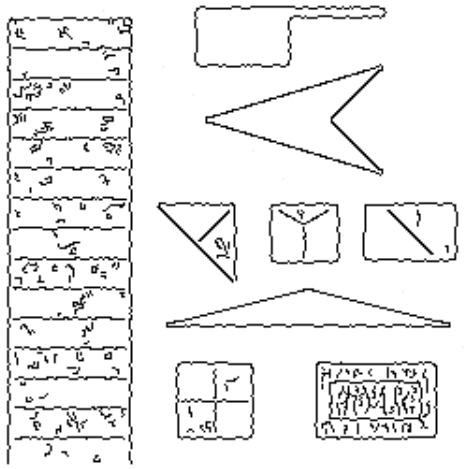
A Statistical Approach (cont'd)

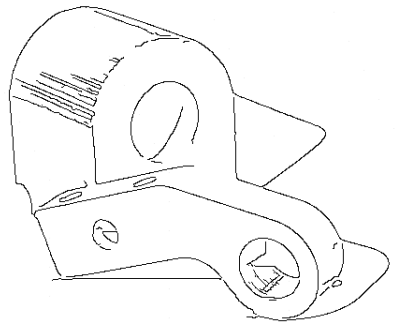
To use this technique, we must know noise variance σ^2. We can obtain an estimate of σ^2 via

$$\hat{\sigma}^2 = \frac{E^2}{\sum_r \sum_c 1 - 3}$$

where $E^2 = \sum_r \sum_c \{I(c,r) - \hat{\alpha}c - \hat{\beta}r - \gamma\}^2$. It is assumed that $E^2 \sim \chi^2_{\sum_r \sum_c 1-3}$. Assume each pixel is perturbed identically and independently, we can obtain a more stable estimate of σ^2, $\hat{\sigma}^2_{avg} = \frac{\sum_{n=1}^{N} \hat{\sigma}^2_n}{N}$, where N represents the N neighborhoods.

Canny Edge Detection Examples





Second Order Edge Filter

The place where the first order image derivative of an step edge is maximum is exactly the place where the second derivative of the step has a zero-crossing. The isotropic generalization of the second derivative in 2D is the Laplacian. Laplacian of image intensity $I(c, r)$ is

$$\bigtriangledown^2 I = (\frac{\partial^2}{\partial c^2} + \frac{\partial^2}{\partial r^2})I = \frac{\partial^2 I}{\partial c^2} + \frac{\partial^2 I}{\partial r^2}$$

Note Laplacian operation does not provide strength and orientation of the edge point.

Second Order Edge Filter

Numerical approximations of $\frac{\partial^2 I}{\partial c^2}$ and $\frac{\partial^2 I}{\partial r^2}$ can be found in appendix A. 2. A 3×3 Laplacian mask is

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Alternatively, $\frac{\partial^2 I}{\partial c^2}$ and $\frac{\partial^2 I}{\partial r^2}$ can be obtained analytically with facet fitting.

Second Order Edge Filter (cont'd)

Laplacian of a Gaussian (LoG) edge detector. The Gaussian serves the purpose of smoothing for noise reduction. Let G be a 2D Gaussian function, I be an image, and because of the associative property of the Laplacian operator

$$\bigtriangledown^2(I * G) = I * \bigtriangledown^2 G$$

where

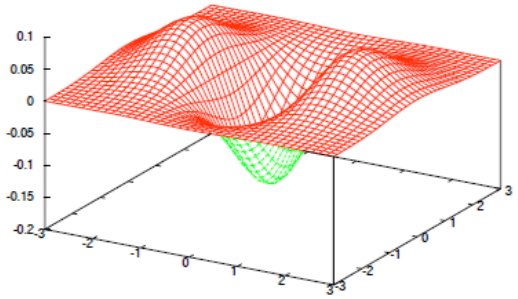
$$
\begin{aligned}
\bigtriangledown^2 G &= (\frac{\partial^2}{\partial c^2} + \frac{\partial^2}{\partial r^2})G \\
&= (\frac{\partial^2}{\partial c^2} + \frac{\partial^2}{\partial r^2})\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}(\frac{c^2+r^2}{\sigma^2})} \\
&= \frac{1}{\sqrt{2\pi}\sigma^3} e^{-\frac{1}{2}(\frac{c^2+r^2}{\sigma^2})}(\frac{c^2+r^2}{\sigma^2} - 2)
\end{aligned}
$$

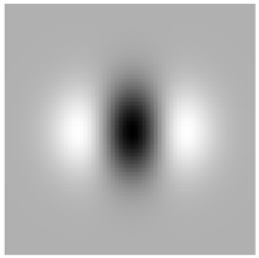
To avoid the negative area of the kernel, the radius of LOG kernel must be larger than $\sqrt{2}\sigma$. In practice, to avoid truncation, the width of the kernel is usually larger than $W = 3\sqrt{2}\sigma$.

Let $\sigma = 1$, W=5, a LOG kernel can then be constructed using above equation with c and r range from -2 to 2.

LoG X Derivative

2D Gaussian Second X-Derivative

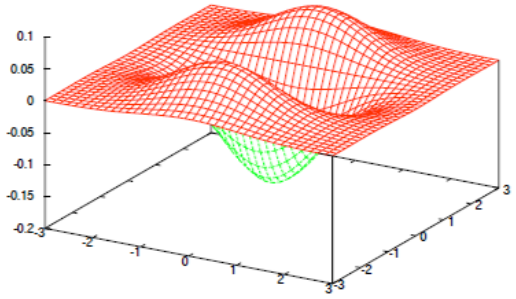


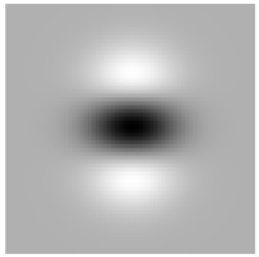


$$\frac{\partial^2}{\partial x^2} G_\sigma(x,y) = \frac{x^2-\sigma^2}{2\pi\sigma^6} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

LoG Y Derivative

2D Gaussian Second X-Derivative





$$\frac{\partial^2}{\partial y^2} G_\sigma(x,y) = \frac{y^2-\sigma^2}{2\pi\sigma^6} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

Second Order Edge Filter (cont'd)

Once an image is convolved with a LOG kernel, the zero crossings can be detected as follows: A pixel is declared to have a zero crossing if it is less than -t and one of its neighbors is greater than t or vice verse, where t is a threshold. This process is equivalent to the non-maximum supression process for gradient operator. Finally, a threshold is performed such that zero-crossings pixels with gradient magnitude larger than a threshold are retained.

Integrated Gradient Edge Detector

Let $I(c,r)$ be image intensity at (c,r) location, the image gradient at (c,r) is

$$\bigtriangledown I(c,r) = \begin{pmatrix} \frac{\partial I}{\partial c} \\ \frac{\partial I}{\partial r} \end{pmatrix}$$

For a given direction $(\sin\theta, \cos\theta)$, the first order directional derivative projected in θ direction

$$\begin{aligned} I'_\theta &= (\cos\theta, \sin\theta) \bigtriangledown I(c,r) \\ &= \frac{\partial I}{\partial c}\cos\theta + \frac{\partial I}{\partial r}\sin\theta \end{aligned}$$

Integrated Gradient Edge Detector (cont'd)

For a given $2N \times 2N$ neighborhood, the integrated directinal derivative I_θ for all pixels in the neighborhood is

$$I_\theta = \frac{1}{4N^2} \int_{-N}^{N} \int_{-N}^{N} I'_\theta dcdr$$

The optimal edge direction is

$$\theta^* = \arg \max_\theta I_\theta$$

Edge Detector Evaluation

- Good localization
- Accurate orientation estimation
- The number of spurious edges (false alarm) and the number of true edges missed (misdetection).

2D Image Features

Two dimensional image features are interesting local structures. They include junctions of different types like 'Y', 'T', 'X', and 'L'. Much of the work on 2D features focuses on junction 'L', aks, *corners*.

Corner

Corners are the intersections of two edges of sufficiently different orientations.

Corner Detection

- Corners are important two dimensional features.
- They can concisely represent object shapes, therefore playing an important role in matching, pattern recognition, robotics, and mensuration.

Previous Research

- Corner detection from the underlying gray scale images.
- Corner detection from binary edge images (digital arcs).

The Harris Corner Detector

The Harris corner detector detects corners directly from gray scale images. Corners are located in the region with large intensity variations in certain directions.

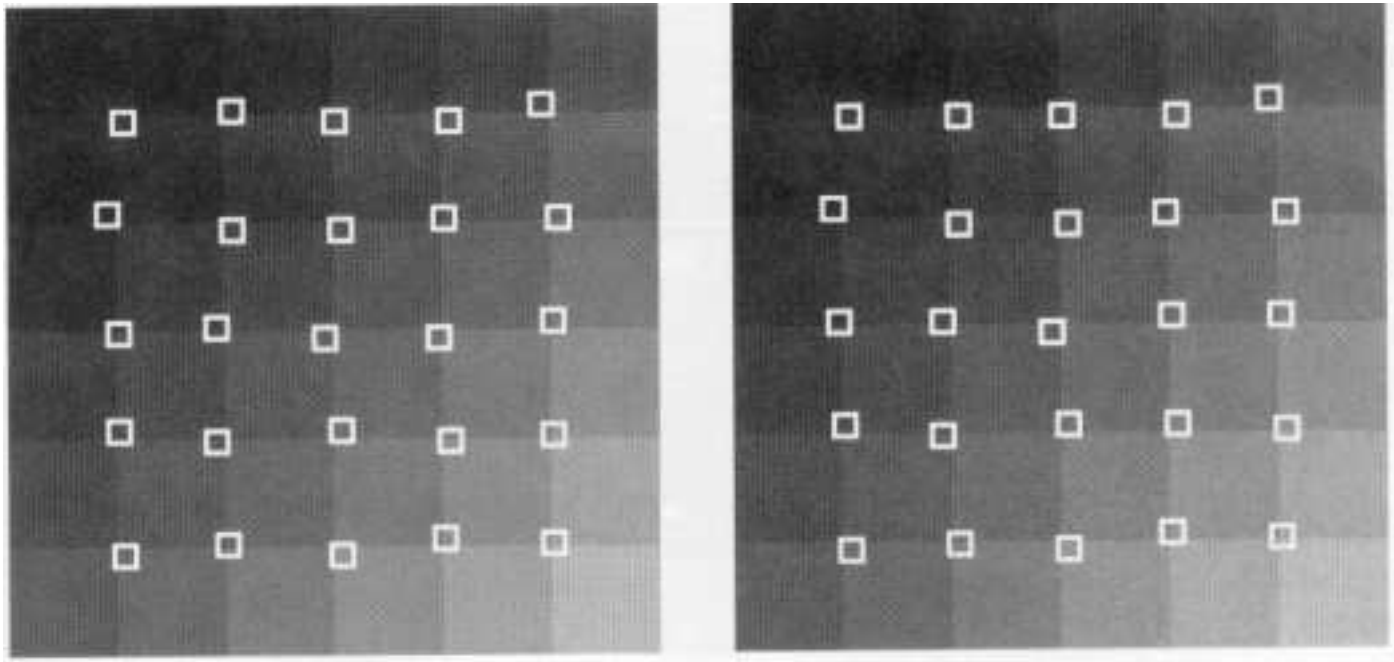
Let I_c and I_r be image gradients in horizontal and vertical directions, we can defined a matrix C (moment matrix or structure tensor) that summarizes I_c and I_r distribution in a local neighborhood

$$C = \begin{bmatrix} \sum I_c^2 & \sum I_c I_r \\ \sum I_c I_r & \sum I_r^2 \end{bmatrix}$$

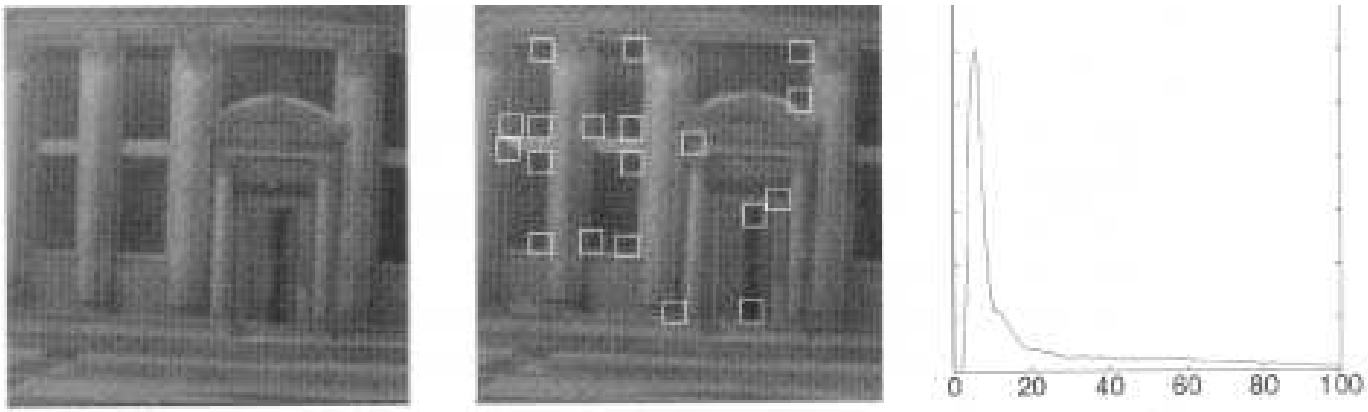
where the sums are taken over a small neighborhood

Compute the eigenvalue of C, λ_1 and λ_2. If the minimum of the two eigen values is larger than a threshold, the the point is

declared as a corner. It is good for detecting corners with orthogonal edges.



(a) (b)



(a) (b) (c)

Corner Detection from Gray Scale Image II

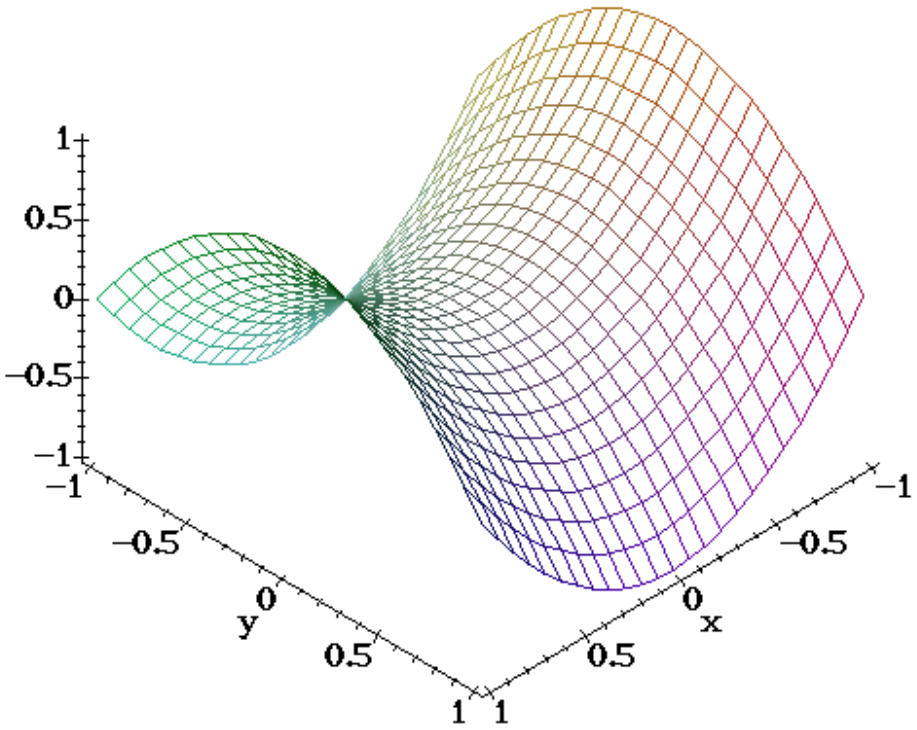
Assume corners are formed by two edges of sufficiently different orientations, in a $N \times N$ neighborhood, compute the direction of each point and then construct a histogram of edgel orientations. A pixel point is declared as a corner if two distinctive peaks are observed in the histogram.

Corner Detection from Gray Scale Image II

Fit the image intensities of a small neighborhood with a local quadratic or cubic facet surface. Look for saddle points by calculating image Gaussian curvature (the product of two principle curvatures, see appendix A 5).

Corner Detection from Gray Scale Image II

Saddle points are points with zero gradient, and a max in one direction but a min in the other. Different methods are used to detect saddle points.


1
0.5
0
−0.5
−1
−1
−0.5
0
0.5
1
y
1
0.5
0
−0.5
−1
x

Corner Detection from Digital Arcs

- Objective

  Given a list of connected edge points (a digital arc) resulted from an edge detection, identify arc points that partition the digital arc into maximum arc subsequences.

Corner Detection from Digital Arcs

- Criteria

  maximum curvature.

  deflection angle.

  maximum deviation.

  total fitting errors.

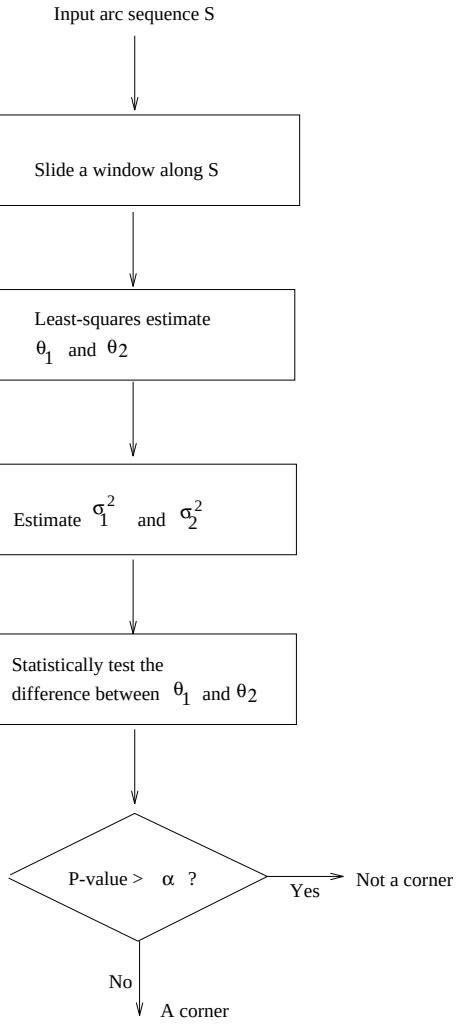
A Statistical Approach

- Problem Statement

  Given an arc sequence $S = \{ \begin{pmatrix} \hat{x}_n \\ \hat{y}_n \end{pmatrix} | n = 1, \ldots N \}$,

  statistically determine the arc points along S that are most likely corner points.

Approach Overview



Approach Overview (cont'd)

- Slide a window along the input arc sequence S.
- Estimate $\hat{\theta}_1$ and $\hat{\theta}_2$, the orientations of the two arc subsequences located to the right and left of the window center, via least-squares line fitting.
- Analytically compute σ_1^2 and σ_2^2, the variances of $\hat{\theta}_1$ and $\hat{\theta}_2$.
- Perform a hypothesis test to statistically test the difference between $\hat{\theta}_1$ and $\hat{\theta}_2$.

Details of the Proposed Approach

- Noise and Corner Models
- Covariance Propagation
- Hypothesis Testing

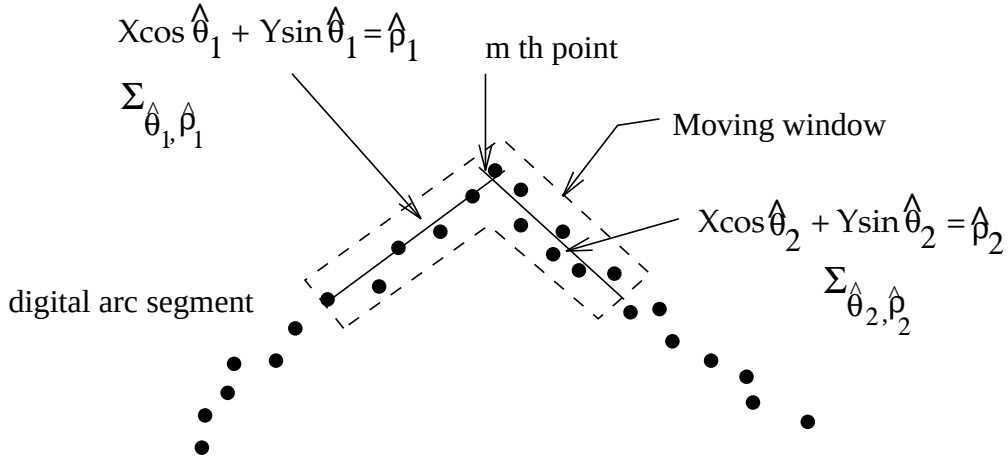
Noise Model

Given an observed arc sequence $S = \{(\hat{x}_n\ \hat{y}_n)^t | n = 1, \ldots N\}$, it is assumed that $(\hat{x}_n, \hat{y}_n)$ result from random perturbations to the ideal points (x_n, y_n), lying on a line $x_n \cos\theta + y_n \sin\theta - \rho = 0$, through the following noise model:

$$\begin{pmatrix} \hat{x_n} \\ \hat{y_n} \end{pmatrix} = \begin{pmatrix} x_n \\ y_n \end{pmatrix} + \xi_n \begin{pmatrix} \cos\theta \\ sin\theta \end{pmatrix}; n = 1, \ldots, N$$

where ξ_n are iid as N(0, σ^2).

Corner Model



$$H_0 : \theta_{12} < \theta_0 \qquad H_1 : \theta_{12} \geq \theta_0$$

where $\hat{\theta}_{12} = |\hat{\theta}_1 - \hat{\theta}_2|$, θ_{12} is the population mean of $\hat{\theta}_{12}$, and θ_0 a threshold.

Covariance Propagation

- Problem statement

  Analytically estimate $\Sigma_{\Delta\Theta}$, the covariance matrix of least-squares estimate $\hat{\Theta} = (\hat{\theta}\ \hat{\rho})^t$, in terms of the input covariance matrix $\Sigma_{\Delta X}$.

Covariance Propagation (cont.)

From Haralick's covariance propagation theory, define

$$F(\hat{\Theta}, \hat{X}) = \sum_{n=1}^{N} (\hat{x}_n cos\hat{\theta} + \hat{y}_n \sin\hat{\theta} - \hat{\rho})^2$$

and

$$g^{2\times 1}(\Theta, X) = \frac{\partial F}{\partial \Theta}$$

then

$$\begin{aligned} \sum_{\Delta\Theta} \quad = \quad & (\frac{\partial g(X,\Theta)}{\partial \Theta})^{-1} (\frac{\partial g(X,\Theta)}{\partial X})^t \sum_{\Delta X} \\ & (\frac{\partial g(X,\Theta)}{\partial X})[(\frac{\partial g(X,\Theta)}{\partial \Theta})^{-1}]^t \end{aligned}$$

Covariance Propagation (cont.)

Define

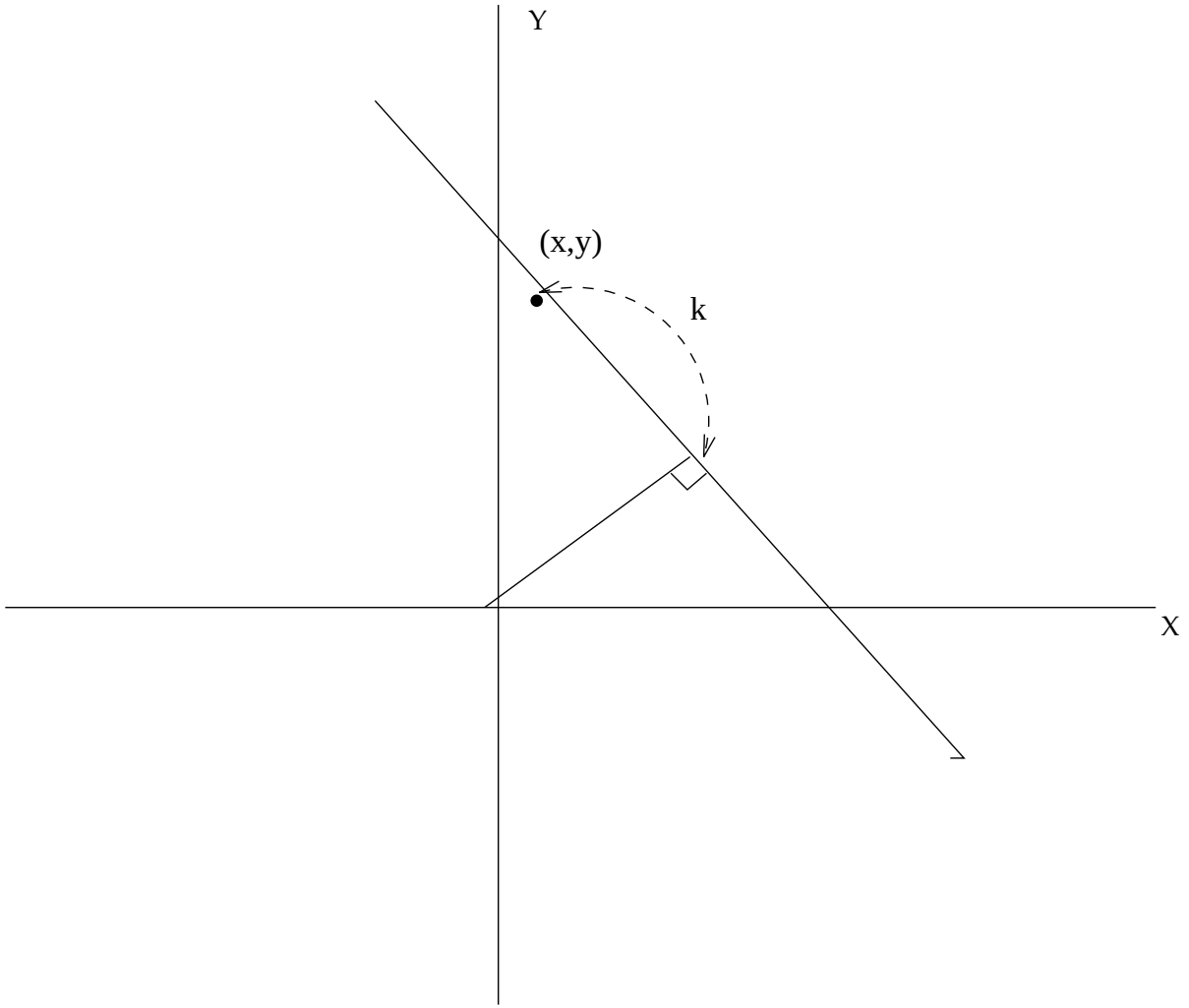
$$k = \begin{cases} +\sqrt{x^2 + y^2 - \rho^2} & \text{if } y\cos\theta \geq x\sin\theta \\ -\sqrt{x^2 + y^2 - \rho^2} & \text{otherwise} \end{cases}$$

and

$$\mu_k = \frac{1}{N}\sum_{n=1}^{N} k_n$$

$$\sigma_k^2 = \sum_{n=1}^{N}(k_n - \mu_k)^2$$

Geometric Interpretation of k



Covariance Propagation (cont.)

$$\sum_{\Delta\Theta} = \sigma^2 \begin{pmatrix} \frac{1}{\sigma_k^2} & \frac{\mu_k}{\sigma_k^2} \\ \frac{\mu_k}{\sigma_k^2} & \frac{1}{N} + \frac{\mu_k^2}{\sigma_k^2} \end{pmatrix}$$

Hypothesis Testing

$$H_0 : \theta_{12} < \theta_0 \qquad H_1 : \theta_{12} \geq \theta_0$$

where θ_0 is an angular threshold and θ_{12} is the population mean of RV $\hat{\theta}_{12} = |\hat{\theta}_1 - \hat{\theta}_2|$.

Let

$$T = \frac{\hat{\theta}_{12}^2}{\hat{\sigma}_{\theta_1}^2 + \hat{\sigma}_{\theta_2}^2}$$

Under null hypothesis

$$T \sim \chi_2^2$$

if $P(T) < \alpha$, then a corner else not a corner.

Corner Detection Example

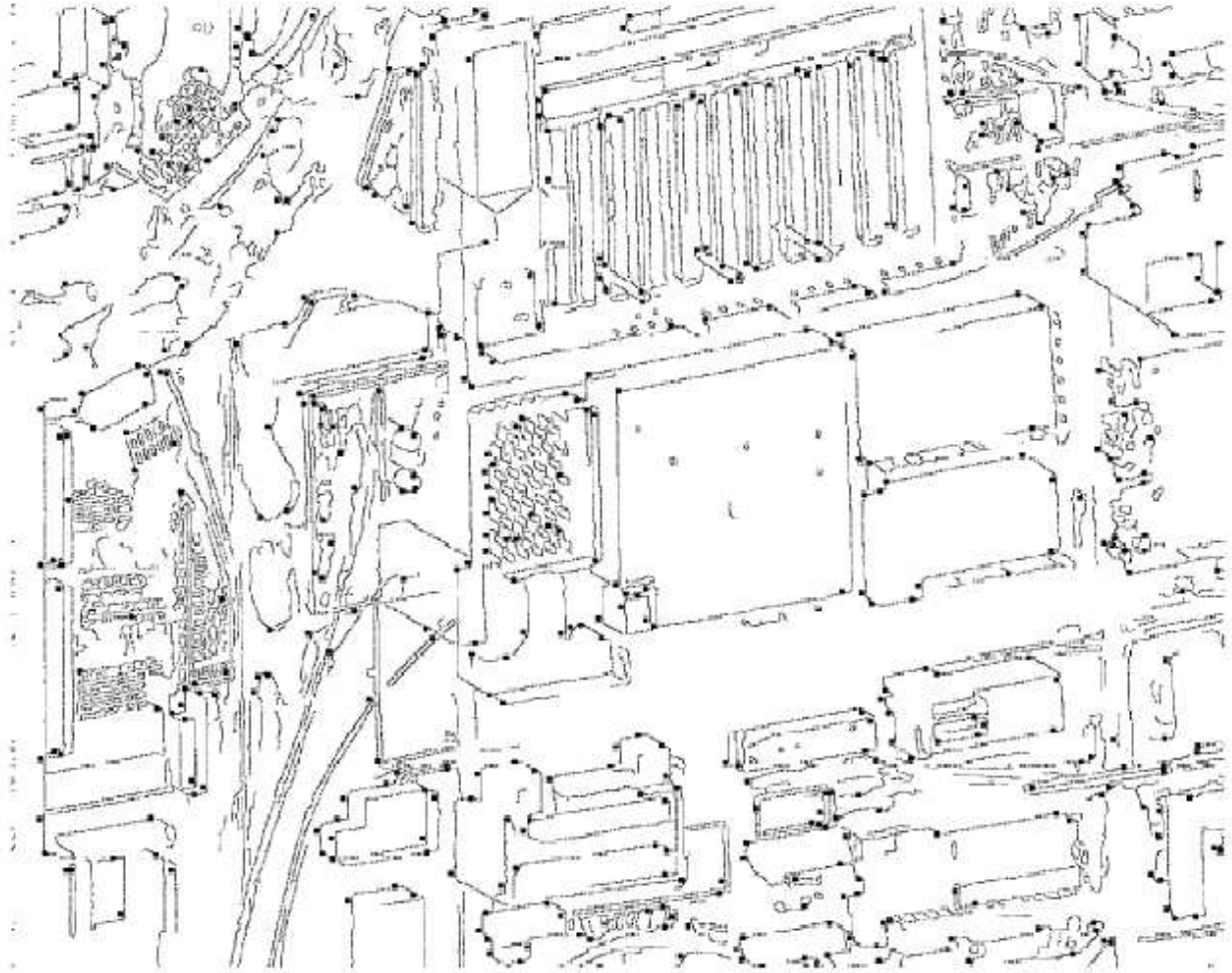


Fig. 2. Extracted edges of building model for a RADIUS image with detected corners (represented by black square dots) overlaid.

Performance Evaluation

- Evaluation criteria

  Misdetection rate.

  False alarm rate.

- Images: Eighty RADIUS Images

Evaluation Results

- Average Misdetection Rate: 2.3%
- Average False Alarm Rate: 2.1%

Performance Comparison

- Lowe's Algorithm
- SUSAN Corner Detector

Results from Radius Images

- figure 1
- figure 2
- figure 3
- figure 9

Results from Comparative Study

- Low's algorithm.

  figure 4

  figure 5

  figure 6

  figure 7

  figure 8

- SUSAN corner detector.

  figure 10

Conclusions

- Present a statistical approach for detecting corners on digital arc sequences.
- Performance evaluation shows the robustness of the proposed algorithm and its superiority over the existing algorithms.

Conclusions (cont.)

- Contributions include
  - Explicitly introduce a noise model and a corner model to account for the image noise.
  - Develop an analytical method for estimating the covariance matrix of the fitted line parameters.

Corner Detection Example

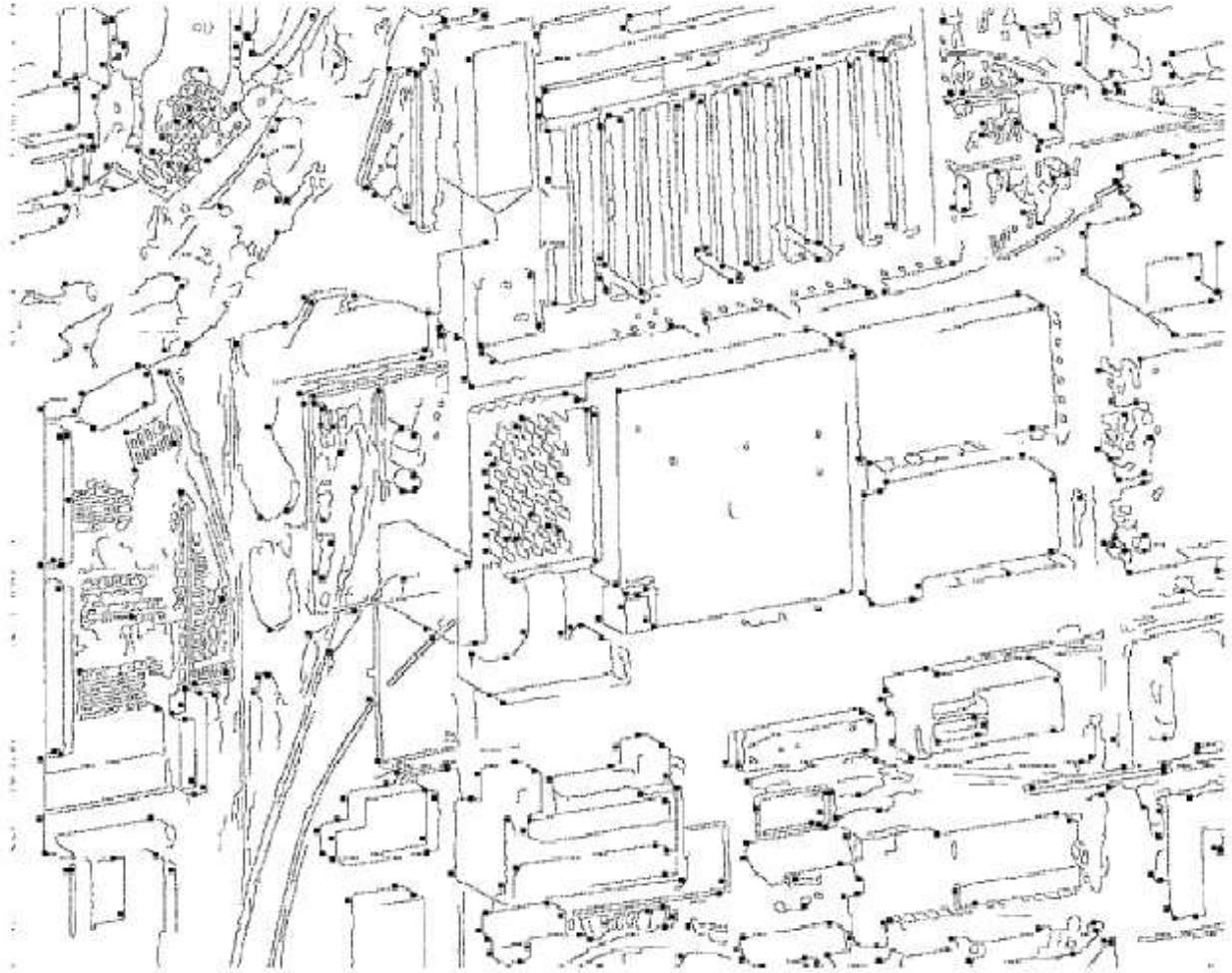


Fig. 2. Extracted edges of building model for a RADIUS image with detected corners (represented by black square dots) overlaid.

Scale Invariant Feature Transform (SIFT) Features

See attached

Line and Curves Detection

Given an edge image, find line or curve segments present in the image.

Issues with Curve Detection

- Grouping (e.g., the Canny hysteresis thresholding procedure)
- Model fitting

They can be performed sequentially or simultaneously.

The Hough Transform

- The Hough Transform (HT) is a popular technique for detecting straight lines and parameterizable curves on gray-scale images.
- It was invented by Richard Duda and Peter Hart (RPI alumnus) in 1972
- It maps image data from image space to a parameter space, where curve detection becomes peak detection problem, i.e., searching the parameter space for peaks.
- Check the Wikipedia on HT for more information. It includes a citation of my prior work [3, 1].

The HT for Line detection

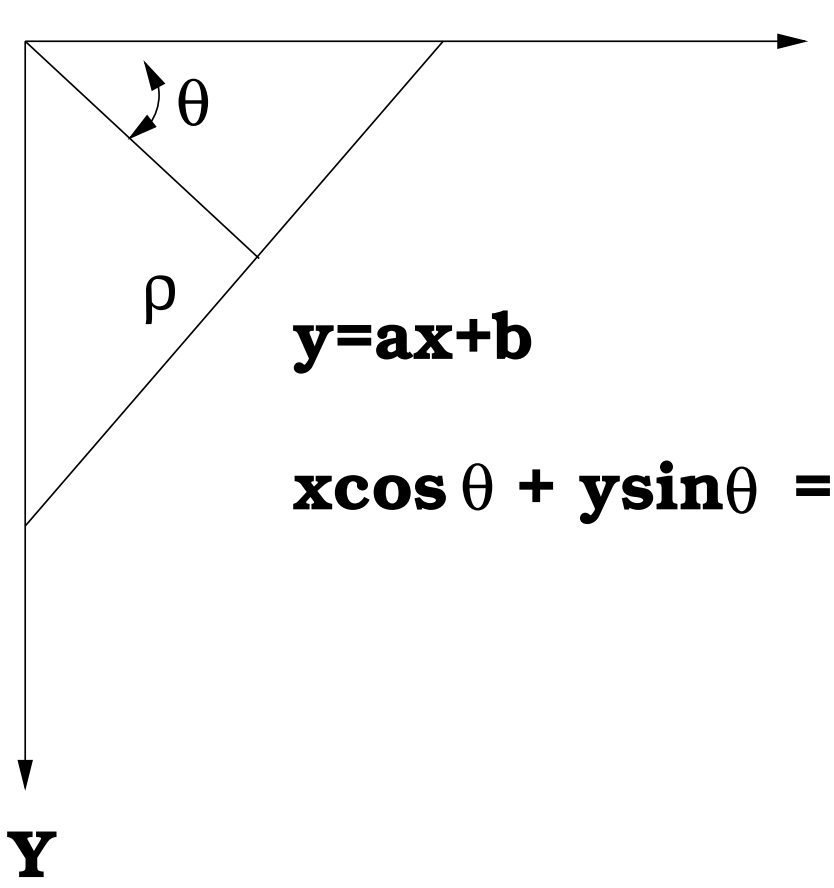
1. **Parameterize the line**

$$y = ax + b$$

In the parameter spaces determined by a and b, each parameter pair (a, b) represents a line. For example (0,4) represents a horizontal line. This line representation can not represent a vertical line.

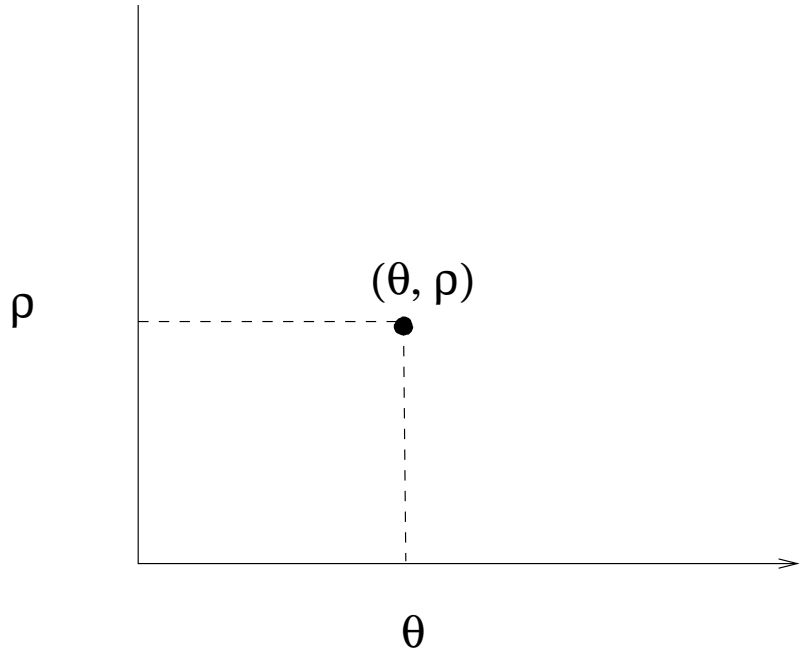
Alternatively, we can also parameterize a line in polar coordinates as

$$x \cos\theta + y \sin\theta - \rho = 0$$



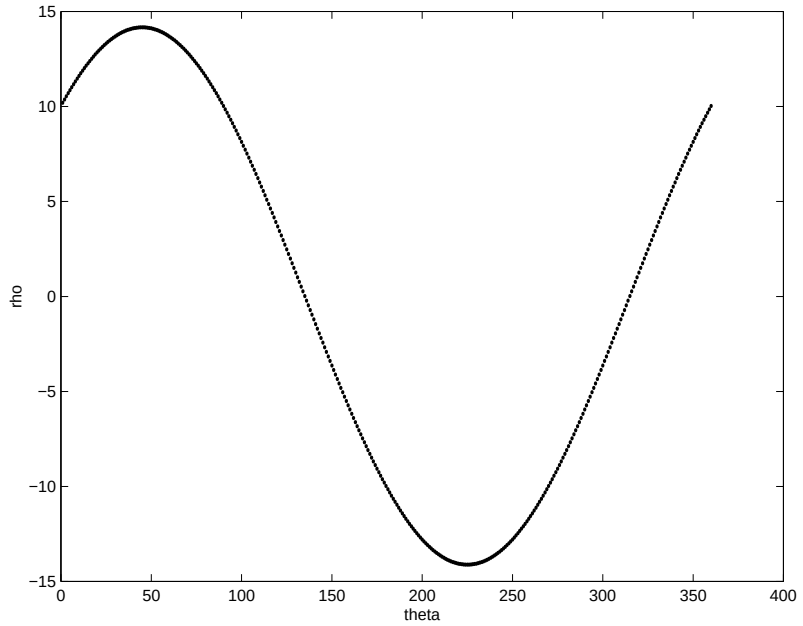
y=ax+b

xcos θ **+ ysin** θ **=** ρ

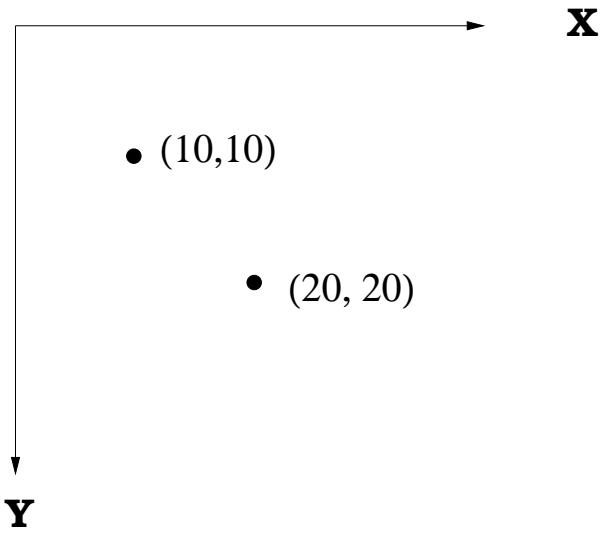


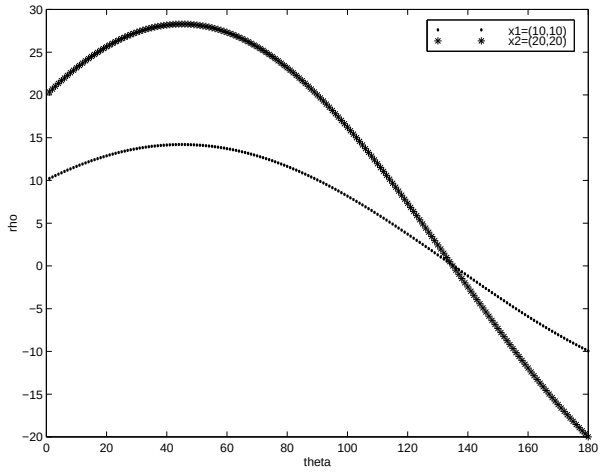
In parameter space determined by θ and ρ, each pair of (θ, ρ) represents a line. For example, a vertical line my be represented as $(0, 4)$.

An image point (x_0, y_0) represents the intersection of all lines that satisfy $x_0 \cos\theta + y_0 \sin\theta - \rho = 0$, each of which can be represent by a pair of (θ, ρ). In the parameter space, the image point is represented by a sinusoid curve



The intersection of two sinusoid curves representing the line going through two image points (e.g., (10,10) and (20,20)) uniquely determines a line.





If lines are parameterized using a and b, then an image point is represented by as a line in the parameter space (fig. 5.1).

2. **Quantize parameters ρ and θ**

   To keep the parameter space finite, it is often necessary to quantize the parameter space. By quantization, the parameter space is divided into finite grid of cells, the dimensions of each cell are determined by the quantization intervals. Each cell is indexed by a pair of quantized (θ, ρ).

3. **Build an accumulator array** $A(\theta, \rho)$

   The accumulator array is used to accrue the contribution of each image point to a quantized (θ, ρ). The standard HT scheme for updating the accumulator array $\mathrm{A}(\theta, \rho)$ is

$$A(\theta, \rho) = \#\{(x, y) \in X \times Y | x \cos\theta + y \sin\theta + \rho = 0\}$$

   where $X \times Y$ are the image domain.

   For example, if we have N image points lying on a line with parameter pair (θ_0, ρ_0), then we have $A(\theta_0, \rho_0) = N$ and the accumulator array values for all other parameter pairs are 1 or zero.

   For each quantized θ, we use the line equation to compute the corresponding ρ. The computed ρ is then quantized. The accumulator corresponding to the quantized θ and ρ are then incremented by 1.

4. **Search $A(\theta, \rho)$ for peaks**

   The image lines are therefore detected by the peaks of $\mathrm{A}(\rho, \theta)$.

   To detect multiple lines, look for all local maxima of $A(\theta, \rho)$.

HT Algorithm Outline

For an input gray scale image I of $M \times N$.

1. Locate the HT coordinate system
2. Identify the ranges for θ and ρ. Let the range for for θ be between θ_l and θ_h, and the range for ρ between ρ_l and ρ_h.
3. Choose quantization intervals $\delta\theta$ and $\delta\rho$ for θ and ρ respectively.
4. Discretize the parameter space of ρ and θ using sampling steps $\delta\rho$ and $\delta\theta$. Let θ_d and ρ_d be 1D arrays containing the discretized θ and ρ.
5. Let A(T, R) be an array of integer counter; initialize all elements of A to zero, where $R = \frac{\rho_h - \rho_l}{\delta\rho}$ and $T = \frac{\theta_h - \theta_l}{\delta\theta}$.
6. Let L(T,R) be an array of a list of 2D coordinates.

7. For each image pixel (c, r), if its gradient magnitude $g(c, r) > \tau$, where τ is a gradient magnitude threshold.
   for i=1 to T
   - $$\rho = c * \cos(\theta_d(i)) + r * \sin(\theta_d(i))$$
   - find the index k, for the element of ρ_d closest to ρ
   - increment A(i,k) by one.
   - Add point (c,r) to L(i,k)
8. Find all local $A(i_p, k_p)$ such that $A(i_p, k_p) > t$, where t is a threshold.
9. The output is a set of pairs $(\theta_d(i_p), \rho_d(k_p))$ and lists of points in $L(i_p, k_p)$, describing the lines detected in the image, along with points located on these lines

HT Coordinate Systems

- The HT coordinate system coincides with the row-column coordinate system.

  Ranges of ρ and θ

  $0 < \rho < \sqrt{M^2 + N^2}$

  $-90 < \theta < 180$

  or

  $-M < \rho < \sqrt{M^2 + N^2}$

  $0 < \theta < 180$

- The HT coordinate system locates at the center of the image

  Ranges of ρ and θ

  $0 < \rho < \frac{\sqrt{M^2+N^2}}{2}$

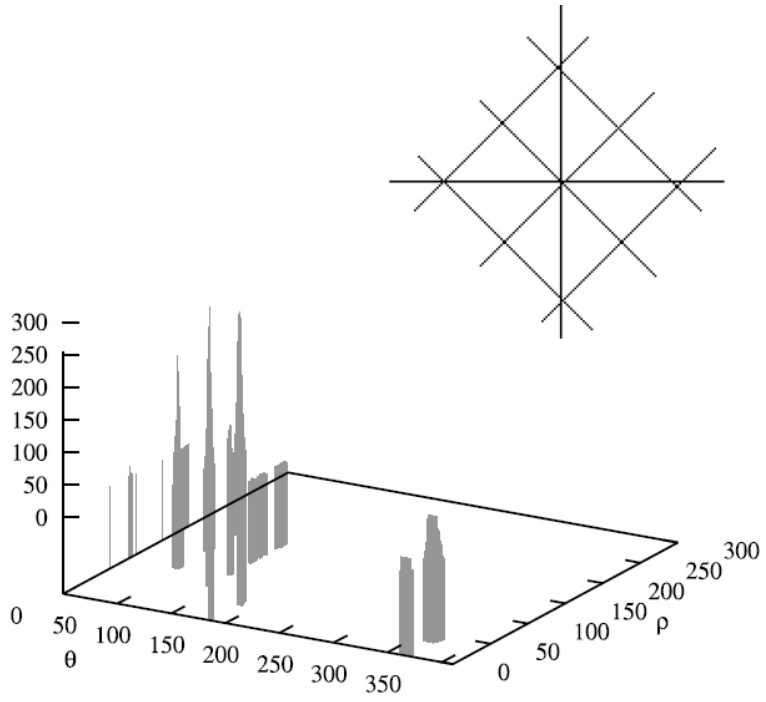
  $0 < \theta < 360$

or

$$-\frac{\sqrt{M^2+N^2}}{2} < \rho < \frac{\sqrt{M^2+N^2}}{2}$$

$$0 < \theta < 180$$

An Example



Top: An image with lines, Bottom: the display of the accumulator array.

Incorporating Gradient Direction

If gradient direction at each image point is available, it can be used to restrict the range of values of θ that the image point may vote for. Instead of checking for all possible θ s as in the standard HT, we only increase the accumulators for those lines whose orientations are close to the direction that is orthogonal to the gradient direction of the image point, therefore significantly reducing the required computations.

Incorporating Gradient Magnitude

Instead of updating the accumulator array by an unit uniformly, the contribution of each point to the accumulator array can be determined based on the gradient magnitude of the point.

HT for Curves

The HT can be generalized to detect any parameterizable curves such as circles and ellipse. For example for circle detection, a circle can be parameterized as

$$(x - x_0)^2 + (y - y_0)^2 = R^2$$

where the parameters are (x_0, y_0), and R. We can therefore construct a 3D accumulator $A(x_0, y_0, R)$. For each edge point vary (x_0, y_0), compute the corresponding R using the circle equation, and then update A. In the end, search A for peaks.

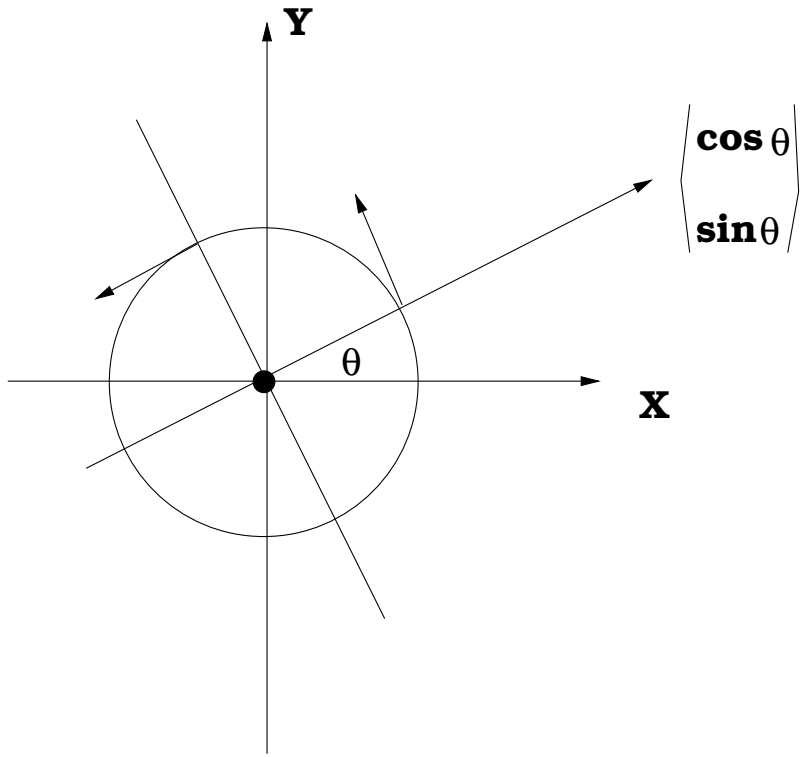
If we know the gradient direction of each point, we may use a different parameterizable representation for circle to take advantage of the gradient direction

$$\begin{aligned} x &= R\cos\theta + x_0 \\ y &= R\sin\theta + y_0 \end{aligned}$$

where θ represents the gradient direction of point (x, y). We can then reduce search from 2D to 1D. We can vary R and then use the above equation to compute x_0 and y_0, then updating the corresponding accumulator accordingly.

Another HT Technique for Circle Detection

- Detect circle center first
- Determine circle radius



For the parametric circle equation, we have the tangent of the

circle at (x, y) is

$$\frac{dx}{dy} = -\tan\theta$$

From circle equation $(x - x_0)^2 + (y - y_0)^2 = R^2$, we have

$$\frac{dx}{dy} = -\frac{y - y_0}{x - x_0}$$

Equating the above two equations yields

$$\frac{y - y_0}{x - x_0} = \tan\theta$$

where θ represents the gradient direction at (x, y).

This equation does not contain circle radius R, therefore allowing to find circle centers first.

Another HT Technique for Circle Detection

Given (x_0, y_0), we can then use equation
$(x - x_0)^2 + (y - y_0)^2 = R^2$ to determine Rs for each (x_0, y_0).
Note different Rs may be identified for each (x_0, y_0), indicating concentric circles.

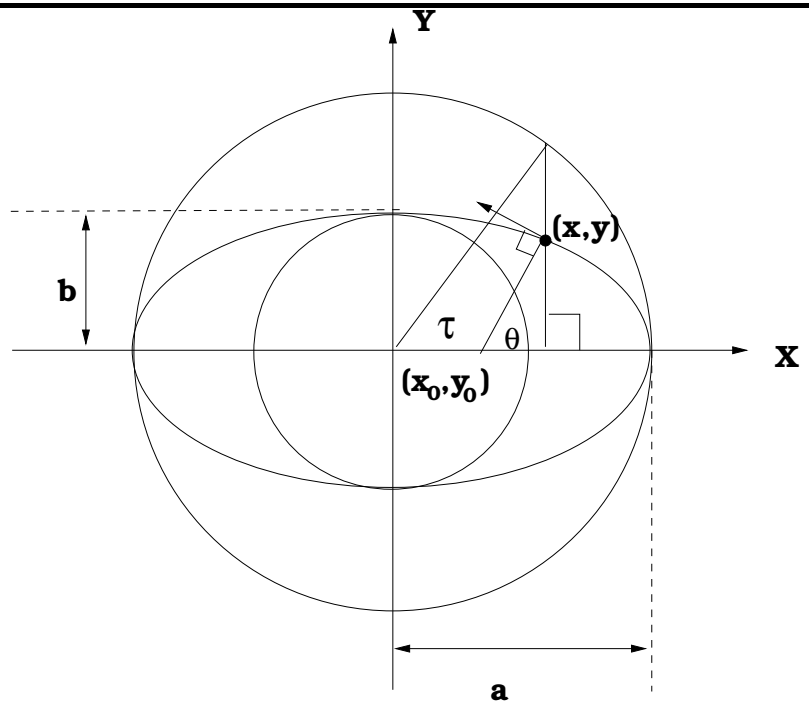
My prior work on circle detection can be found in [2].

HT for Ellipse Detection

For a nominal ellipse (i.e., major and minor axes align with X and Y axes), it can be represented as

$$\frac{(x - x_0)^2}{a^2} + \frac{(y - y_0)^2}{b^2} = 1$$

HT for Ellipse Detection (cont'd)



$$x = x_0 + a\cos\tau$$

$$y = y_0 + b\sin\tau$$

HT for Ellipse Detection (cont'd)

From the tangent definition,

$$\frac{dx}{dy} = -\tan\theta$$

From the parametric ellipse equation, we have

$$\frac{dx}{dy} = -\frac{a}{b}\tan\tau$$

As a result, we have

$$\tan\tau = \frac{b}{a}\tan\theta$$

Substituting the above equation to the parametric ellipse equation yields the parametric representation of an ellipse using θ

$$
\begin{aligned}
x &= x_0 \pm \frac{a}{\sqrt{1 + \frac{b^2}{a^2} \tan^2 \theta}} \\
y &= y_0 \pm \frac{b}{\sqrt{1 + \frac{a^2}{b^2 \tan^2 \theta}}}
\end{aligned}
$$

This still, however, involves 4 parameters.

HT for Ellipse Detection (cont'd)

From the previous equations, we have

$$\frac{y - y_0}{x - x_0} = \frac{b}{a} \tan \tau$$

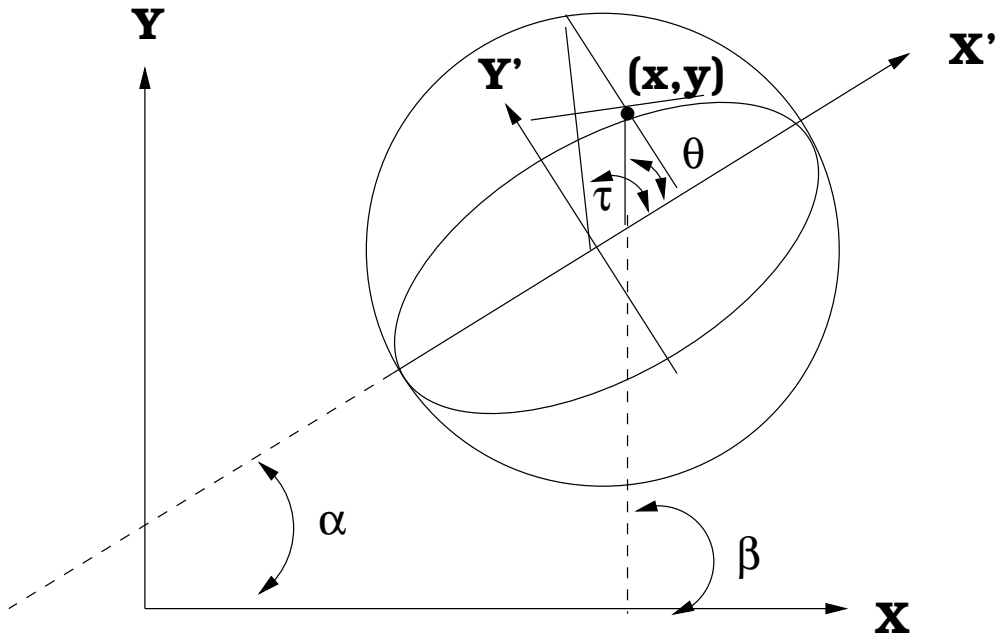
We also know $\tan \tau = \frac{b}{a} \tan \theta$ As a result, we have

$$\frac{y - y_0}{x - x_0} = \frac{b^2}{a^2} \tan \theta$$

In the above equation, we can treat $\frac{b^2}{a^2}$ as a single unknown, effectively reducing to three dimensions. We can therefore construct an accumulator array involving x_0, y_0, and $\frac{b^2}{a^2}$.

Detection of Ellipse with an Unknown Orientation

For a general ellipse, its orientation is unknown. This introduces another parameter. Let $X' - Y'$ represent the coordinate system where the major and minor axes align with X' and Y' respectively as shown in the figure.



$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} + \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} a\cos\tau \\ b\sin\tau \end{pmatrix}$$

From the previous discussion, we have

$$\tan\tau = \frac{b}{a}\tan\theta$$

Since $\theta = \beta - \alpha$, we therefore have

$$\tan\tau = \frac{b}{a}\tan(\beta - \alpha)$$

where β is the gradient direction of point (x, y) in X-Y coordinate frame.

Detection of Ellipse with an Unknown Orientation

From the above equation, we have

$$\frac{(x - x_0)\sin\alpha + (y - y_0)\cos\alpha}{(x - x_0)\cos\alpha - (y - y_0)\sin\alpha} = \frac{b^2}{a^2}\tan(\beta - \alpha)$$

When $\alpha = 0$, the above equation reduces to the equation for a nominal ellipse, with $\theta = \beta$. Again, we reduce the dimension of parameter array from 5 to 4.

Other Efficient Ellipse Detection using HT

One method is to scan the image horizontally and identify any possible edge points on each row. For each row, identify the centroid of any two selected edge points. The idea is that if the two points happen to be on an ellipse, then their centroid must be on the vertical center line of the ellipse. So, by voting, we may detect the vertical center line. Similarly, if scan vertically, the horizontal centerline can be detected. The ellipse center is the intersection of the vertical and horizontal center lines. Given the center of the ellipse, other parameters are then determined.

One of my prior work on ellipse detection [3] was cited over 250 times and was incorporated into OpenCV.

Generalized HT

HT transform can be generalized to detect any shapes, even for those without analytic (parametric) representation.

HT Summary

- Advantages:

  Robust to image noise and occlusion

  Can detect multiple curves

- Disadvantages

  computational intense

  noise and quantization errors may introduce spurious peaks

  standard HT only detects line not line segments

Model Fitting

Given a model and some data, model fitting is concerned with estimating model parameters that could best explain the observed data. *best* may be interpreted as *smallest* sum of distances of all image points to the model curve.

Model Fitting

Mathematically, model fitting may be stated as follows:

Given a vector of points $X = (X_1, X_2, \ldots, X_N)$ and model f, find model parameters Θ by minimizing

$$\sum_{n=1}^{N} D^2(X_n, \Theta)$$

where D is the distance of a point X_n to the model curve f and D is a function of f.

Distance Measures

- Algebraic distance
- Geometric (Euclidean) distance

Distance Measures (cont'd)

The algebraic distance refers to the vertical distance between point (x_n, y_n) and a point (x_n, y'_n) on the curve, where $f(x_n, y'_n) = 0$. On the other hand, the Euclidean distances refers to the distance between point (x_n, y_n) and a point on the curve closest to (x_n, y_n).

Distance Measures (cont'd)

While the geometric distance improves the stability of fitting, it necessitates an iterative procedure, substantially increasing the computational requirements. The algebraic distance, on the other hand, may suffer from lack of fitting stability, it however can be solved non-iteratively, with a direct solution.

Distance Measures (cont'd)

The geometric distance measure and the algebraic distance is related via

$$d(X_n) = \frac{|f(X_n, \Theta)|}{|| \bigtriangledown f(X_n, \Theta)||}$$

where d(X_n) is the geometric distance and $f(X_n)$ is the algebraic distance.

Model Fitting Methods

The model fitting problem can be formulated as a least-squares problem. Solution to a least-squares problem can be linear or non-linear. A linear least-squares problem can be solved as a generalized eigenvector problem. Non-linear least-squares problems are usually solved iteratively.

Least-squares Curve Fitting

Least-squares curve fitting determines the free parameters Θ of an analytical curve $f(x, y, \Theta) = 0$ such that the curve is the best fit to a set of observed points $(\hat{x}_n, \hat{y}_n)$, where $n = 1, \ldots, N$, in the least-squares sense.

Least-squares Curve Fitting (cont'd)

A "best" fit is defined as a fit that minimizes the total residual error E

$$E = \sum_{n=1}^{N} (x_n - \hat{x}_n)^2 + (y_n - \hat{y}_n)^2 - 2\lambda f(x_n, y_n, \Theta) \qquad (1)$$

where (x_n, y_n) is the point on the curve f(x,y) that is closest to point $(\hat{x}_n, \hat{y}_n)$.

Least-squares Ellipse Fitting

Given a conic function expressed as

$$f(x, y) = ax^2 + bxy + cy^2 + dx + ey + f$$

to ensure the resulting fit curve be ellipse instead of hyperbolic or parabolic, the inequality constraint $b^2 - 4ac < 0$ must be imposed. This constraint This inequality constraint, however, renders the ellipse-fitting a non-linear problem.

An Analytic Ellipse Fitting Method

Here we introduce an analytic technique for ellipse-fitting using the approximate constraint $b^2 - 4ac = -1$.

Given the conic equation

$$f(x, y) = ax^2 + bxy + cy^2 + dx + ey + f$$

and a set of points (x_n, y_n), where $n = 1, 2, \ldots, N$, we want to find parameter vector $\Theta = (a\ b\ c\ d\ e\ f)$ by minimizing

$$\sum_{n=1}^{N} f^2(x_n, y_n, \Theta) \qquad \text{subject to } b^2 - 4ac = -1$$

An Analytic Ellipse Fitting Method (cont'd

Let $X_n = (x_n^2 \;\; x_n y_n \;\; y_n^2 \;\; x_n \;\; y_n \;\; 1)$, we then have

$$f(x_n, y_n, \Theta) = X_n^t \Theta$$

Let D be a matrix as

$$D = \begin{pmatrix} X_1 \\ X_2 \vdots \\ X_n \end{pmatrix}$$

This leads to

$$\sum_{n=1}^{N} f^2 = (x_n, y_n, \Theta) = ||D\Theta||^2$$

In matrix format, the problem can be restated as:

determine Θ be minimizing

$$||D\Theta||^2 \quad \text{subject to} \quad \Theta^t C \Theta = -1$$

where

$$\begin{pmatrix} 0 & 0 & -2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ -2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Line and Curves Detection

Given an edge image, find line or curve segments present in the image.

Issues with Curve Detection

- Grouping (e.g., the Canny hysteresis thresholding procedure)
- Model fitting

They can be performed sequentially or simultaneously.

Model Fitting

Given a model and some data, model fitting is concerned with estimating model parameters that could best explain the observed data. *best* may be interpreted as *smallest* sum of distances of all image points to the model curve.

Model Fitting

Mathematically, model fitting may be stated as follows:

Given a vector of points $X = (X_1, X_2, \ldots, X_N)$ and model f, find model parameters Θ by minimizing

$$\sum_{n=1}^{N} D^2(X_n, \Theta)$$

where D is the distance of a point X_n to the model curve f and D is a function of f.

Distance Measures

- Algebraic distance
- Geometric (Euclidean) distance

Distance Measures (cont'd)

The algebraic distance refers to the vertical distance between point $(\hat{x}_n, \hat{y}_n)$ and a point $(\hat{x}_n, \hat{y}'_n)$ on the curve, where $f(\hat{x}_n, \hat{y}'_n) = 0$. On the other hand, the Euclidean distances refers to the distance between point $(\hat{x}_n, \hat{y}_n)$ and a point on the curve that is closest to $(\hat{x}_n, \hat{y}_n)$.

Distance Measures (cont'd)

While the geometric distance improves the stability of fitting, it necessitates an iterative procedure, substantially increasing the computational requirements. The algebraic distance, on the other hand, may suffer from lack of fitting stability, it however can be solved non-iteratively, with a direct solution.

Distance Measures (cont'd)

The geometric distance measure and the algebraic distance is related via

$$d(X_i) = \frac{|f(X_i, \Theta)|}{|| \bigtriangledown f(X_i, \Theta)||}$$

where d(X_i) is the geometric distance and $f(X_i)$ is the algebraic distance.

Model Fitting Methods

The model fitting problem can be formulated as a least-squares problem. Solution to a least-squares problem can be linear or non-linear. A linear least-squares problem can be solved as a generalized eigenvector problem. Non-linear least-squares problems are usually solved iteratively.

Least-squares Curve Fitting

Least-squares curve fitting determines the free parameters Θ of an analytical curve $f(x, y, \Theta) = 0$ such that the curve is the best fit to a set of points $(\hat{x}_n, \hat{y}_n)$, where $n = 1, \ldots, N$, in the least-squares sense. A "best" fit is defined as a fit that minimizes the total residual error E

$$E = \sum_{n=1}^{N} (x_n - \hat{x}_n)^2 + (y_n - \hat{y}_n)^2 - 2\lambda f(x_n, y_n, \Theta) \qquad (2)$$

where (x_n, y_n) is the point on the curve f(x,y) that is closest to corresponding observed point $(\hat{x}_n, \hat{y}_n)$.

Least-squares Ellipse Fitting

When representing the ellipse using a conic equation as shown above, to avoid the trivial solution of Θ=0, the vector Θ is constrained in some way. Many of the published algorithms differ only in the form constraint applied to the parameters: Many suggest $|||\Theta||^2$=1. Rosin [13, Fit] impose a+c=1 or f=1. Rosin [PA letter, 1993, vol. 14, p799] has a paper investigating the merits of using a+c=1 or f=1. Bookstein proposes quadratic constraint $a^2 + 0.5b^2 + c^2 = 1$. This constraint allows the fit ellipse parameters invariant to affine transformation. While the more accurate ellipse fitting requires an iterative procedure, which is computationally expensive and requires an good initial estimate, a less accurate method is the direct method by solving a general eigensystem. It gains speed at the expense of accuracy. t may be used as an initial estimate of the iterative procedure.

While fitting ellipses using the conic equation, to ensure the resulting fit curve be ellipse, the inequality constraint $b^2 - 4ac < 0$ must be imposed. Haralick proposed a method to effectively incorporating the inequality constraint into the error function by working with different set of parameters
Imposing the inequality constraint to the conic equation renders the ellipse-fitting a non-linear problem. Thus, most researchers use the general conic equation for ellipse fitting. However, this often leads to hyperbolic and parabolic arcs being fitted in place of elliptic arcs. Even for approximately elliptical data a hyperbolic or parabolic may give a lower fitting error than an elliptic arc. In particular, sections of low curvature data tend to be fitted best by hyperbolic arcs. Since we are interested in only ellipse and circle features, we need to ignore no-elliptic fit like hyperbolic or or parabolic features. Different options have been

studied to perform ellipse-fitting using the general conic equation:
1) Imposing the $b^2 - 4ac < 0$ constraint [Haralick, Fitzgibbon]
2) Impose some constraints for each iteration to ensure

it converges to an ellipse.[Rosin,Int. Con. CV, 1990] [Porrill, Image and Vision Computing, 8(1), Feb. 1990] 3) Adding a few artificial data points at appropriate locations to force the best fitting conic be ellipse [Cooper, TR, 1975 Brown]. 4) Selecting five data points ...[Bolles, IJCAI, 1981, p637]

References

[1] Qiang Ji and Robert M Haralick. Error propagation for the hough transform. *Pattern Recognition Letters*, 22(6-7):813–823, 2001.

[2] Qiang Ji and Yonghong Xie. Randomised hough transform with error propagation for line and circle detection. *Pattern Analysis & Applications*, 6(1):55–64, 2003.

[3] Yonghong Xie and Qiang Ji. A new efficient ellipse detection method. In *Object recognition supported by user interaction for service robots*, volume 2, pages 957–960. IEEE, 2002.