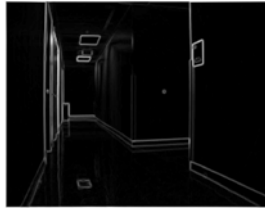


Edge Detection



original image



gradient magnitude

Canny edge detector

- Compute image derivatives
- if gradient magnitude $> \tau$ and the value is a local maximum along gradient direction - pixel is an edge candidate

CS682, Jana Kosecka

An edge is not a line...



- How can we detect *lines* ?

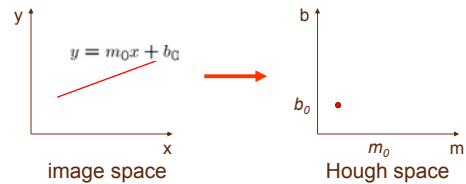
CS682, Jana Kosecka

Finding lines in an image

- Option 1:
 - Search for the line at every possible position/orientation
 - What is the cost of this operation?
- Option 2:
 - Use a voting scheme: Hough transform

CS682, Jana Kosecka

Finding lines in an image



- Connection between image (x,y) and Hough (m,b) spaces
 - A line in the image corresponds to a point in Hough space
 - To go from image space to Hough space:
 - given a set of points (x,y) , find all (m,b) such that $y = mx + b$

CS682, Jana Kosecka

Finding lines in an image

x_0 y_0 x y image space
 $b = -x_0m + y_0$ m b Hough space

- Connection between image (x,y) and Hough (m,b) spaces
 - A line in the image corresponds to a point in Hough space
 - To go from image space to Hough space:
 - given a set of points (x,y) , find all (m,b) such that $y = mx + b$
 - What does a point (x_0, y_0) in the image space map to?
 - A: the solutions of $b = -x_0m + y_0$
 - this is a line in Hough space

CS682, Jana Kosecka

Hough transform algorithm

- Typically use a different parameterization

$$d = x \cos \theta + y \sin \theta$$
 - d is the perpendicular distance from the line to the origin
 - θ is the angle this perpendicular makes with the x axis
 - Why?

Idea - keep an accumulator array (Hough space) and let each edge pixel contribute to it
 Line candidates are the maxima in the accumulator array

CS682, Jana Kosecka

Hough transform algorithm

- Typically use a different parameterization

$$d = x \cos \theta + y \sin \theta$$
 - d is the perpendicular distance from the line to the origin
 - θ is the angle this perpendicular makes with the x axis

CS682, Jana Kosecka

Typical Hough Transform

Basic Hough transform algorithm

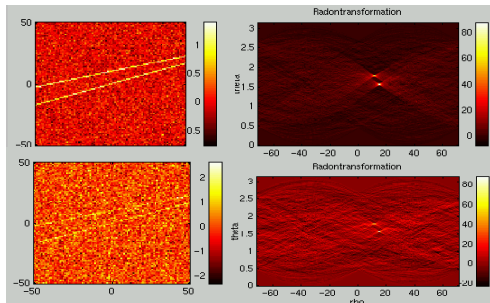
- Initialize $H[d, \theta] = 0$
- For each edge point $I[x,y]$ in the image
 - For $\theta = 0$ to 180

$$H[d, \theta] += d = x \cos \theta + y \sin \theta$$
 - point is now a sinusoid in Hough space
 - Find the value(s) of (d, θ) where $H[d, \theta]$ is maximum

$d = x \cos \theta + y \sin \theta$
 The detected line in the image is given by

What's the running time (measured in #)

Radon transform



CS682, Jana Kosecka

Extensions

- Extension 1: Use the image gradient
 1. same
 2. for each edge point $I[x,y]$ in the image compute unique (d, θ) based on image gradient at (x,y)
$$H[d, \theta] += 1$$
 3. same
 4. Same
- 1. Extension 2
 - give more votes for stronger edges
 - The same procedure can be used with circles, squares, or any other shape

CS682, Jana Kosecka

Hough Transform for Curves

- The H.T. can be generalized to detect any curve that can be expressed in parametric form:
 - $Y = f(x, a_1, a_2, \dots, a_p)$
 - a_1, a_2, \dots, a_p are the parameters
 - The parameter space is p -dimensional
 - The accumulating array is LARGE!

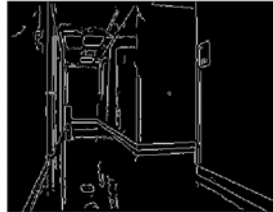
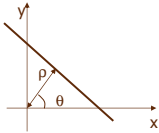
CS682, Jana Kosecka

H.T. Summary

- H.T. 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.

CS682, Jana Kosecka

Line fitting



Non-max suppressed gradient magnitude

- Edge detection, non-maximum suppression (traditionally Hough Transform - issues of resolution, threshold selection and search for peaks in Hough space)
- Connected components on edge pixels with similar orientation - group pixels with common orientation

CS682, Jana Kosecka

Line Fitting

$$A = \begin{bmatrix} \sum x_i^2 & \sum x_i y_i \\ \sum x_i y_i & \sum y_i^2 \end{bmatrix}$$

second moment matrix associated with each connected component
 v_1 - eigenvector of A

$$v_1 = [\cos(\theta), \sin(\theta)]^T$$

$$\theta = \arctan(v_1(2)/v_1(1))$$

$$\rho = \bar{x} \sin(\theta) - \bar{y} \cos(\theta)$$



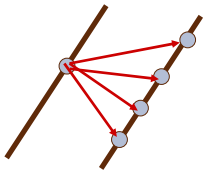
- Line fitting lines determined from eigenvalues and eigenvectors of A
- Candidate line segments - associated line quality

CS682, Jana Kosecka

Corner detection

Corners contain more edges than lines.

- A point on a line is hard to match.



CS682, Jana Kosecka

Finding Corners

Intuition:

- Right at corner, gradient is ill defined.
- Near corner, gradient has two different values.

CS682, Jana Kosecka

Formula for Finding Corners

We look at matrix:

Sum over a small region, the hypothetical corner

Gradient with respect to x, times gradient with respect to y

$$C = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

Matrix is symmetric

CS682, Jana Kosecka

First, consider case where:

$$C = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

This means all gradients in neighborhood are:

(k,0) or (0,c) or (0,0) (or off-diagonals cancel).

What is region like if:

1. $\lambda_1 = 0$?
2. $\lambda_2 = 0$?
3. $\lambda_1 = 0$ and $\lambda_2 = 0$?
4. $\lambda_1 > 0$ and $\lambda_2 > 0$?

CS682, Jana Kosecka

General Case:

From Linear Algebra, it follows that because C is symmetric:

$$C = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

With R a rotation matrix.

So every case is like one on last slide.

CS682, Jana Kosecka

So, to detect corners

- Filter image.
- Compute magnitude of the gradient everywhere.
- We construct C in a window.
- Use Linear Algebra to find λ_1 and λ_2 .
- If they are both big, we have a corner.

CS682, Jana Kosecka

Point Feature Extraction

$$G = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

- Compute eigenvalues of G
- If smallest eigenvalue σ of G is bigger than τ - mark pixel as candidate feature point

- Alternatively feature quality function (Harris Corner Detector)

$$C(G) = \det(G) + k \cdot \text{trace}^2(G)$$

CS682, Jana Kosecka

Harris Corner Detector - Example



CS682, Jana Kosecka