

Edge Detection, Corner Detection Lines

Introduction to Computer Vision
CSE 152
Lecture 10

CSE152, Winter 2013

Intro Computer Vision

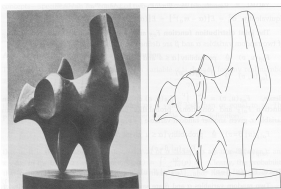
Announcements

- Assignment 1 will be returned on Tuesday
- Assignment 2 due Thursday, February 14 (But do it before).
- Midterm: Thursday, February 14
- Oscar will answer questions about the midterm on Tuesday.

CSE152, Winter 2013

Intro Computer Vision

Edges



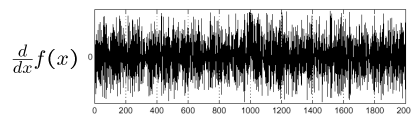
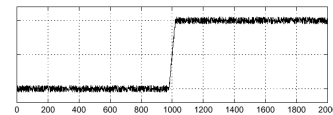
1. Object boundaries
2. Surface normal discontinuities
3. Reflectance (albedo) discontinuities
4. Lighting discontinuities (shadow boundaries)

CSE152, Winter 2013

Intro Computer Vision

Effects of Noise

- Consider a single row or column of the image
 - Plotting intensity as a function of position gives a signal



Where is the edge??

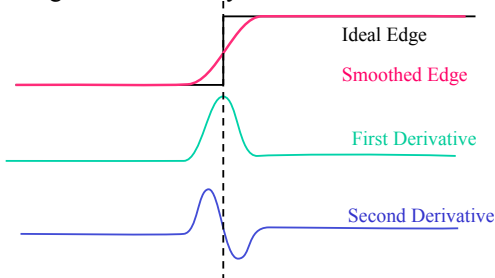
(from Srinivasa Narasmithan)

CSE152, Winter 2013

Intro Computer Vision

Edge is Where Change Occurs: 1-D

- Change is measured by derivative in 1D

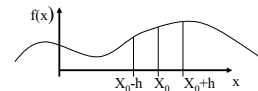


- Biggest change, derivative has maximum magnitude
- Or 2nd derivative is zero.

CSE152, Winter 2013

Intro Computer Vision

Numerical Derivatives



Take Taylor series expansion of $f(x)$ about x_0

$$f(x) = f(x_0) + f'(x_0)(x-x_0) + \frac{1}{2} f''(x_0)(x-x_0)^2 + \dots$$

Consider Samples taken at increments of h and first two terms, we have

$$f(x_0+h) = f(x_0) + f'(x_0)h + \frac{1}{2} f''(x_0)h^2$$

$$f(x_0-h) = f(x_0) - f'(x_0)h + \frac{1}{2} f''(x_0)h^2$$

Subtracting and adding $f(x_0+h)$ and $f(x_0-h)$ respectively yields

$$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h}$$

$$f''(x_0) = \frac{f(x_0+h) - 2f(x_0) + f(x_0-h))}{h^2}$$

Convolve with

First Derivative: $[-1/2h \ 0 \ 1/2h]$

Second Derivative: $[-1/h^2 \ 2/h^2 \ -1/h^2]$

Can often drop h or h^2 in denominator
Yielding $[-1 \ 0 \ 1]$ and $[-1 \ 2 \ -1]$

CSE152, Winter 2013

Intro Computer Vision

Implementing 1-D Edge Detection

1. Filter out noise: convolve with Gaussian
2. Take a derivative: convolve with $[-1 \ 0 \ 1]$
 - We can combine 1 and 2.
3. Find the peak of the magnitude of the convolved image: Two issues:
 - Should be a local maximum.
 - Should be sufficiently high.

CSE152, Winter 2013

Intro Computer Vision

Canny Edge Detector

1. Smooth image by filtering with a Gaussian
2. Compute gradient at each point in the image.
3. At each point in the image, compute the direction of the gradient and the magnitude of the gradient.
4. Perform non-maximal suppression to identify candidate edgels.
5. Trace edge chains using hysteresis thresholding.

CSE152, Winter 2013

Intro Computer Vision

2D Edge Detection: Canny

1. Filter out noise
 - Use a 2D Gaussian Filter. $J = G * I$
2. Take a derivative -- gradient
 - Compute the magnitude of the gradient
 - Compute the direction of the gradient

CSE152, Winter 2013

Intro Computer Vision

Gradient

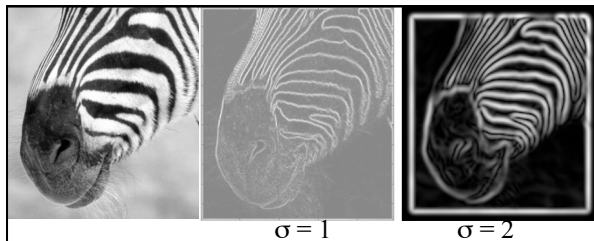
- Given a function $f(x,y)$ -- e.g., intensity is f
- Gradient equation: $\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$
- Represents direction of most rapid change in intensity

- Gradient direction: $\theta = \tan^{-1} \left(\frac{\partial f / \partial y}{\partial f / \partial x} \right)$
- The *edge strength* is given by the gradient magnitude

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

CSE152, Winter 2013

Intro Computer Vision

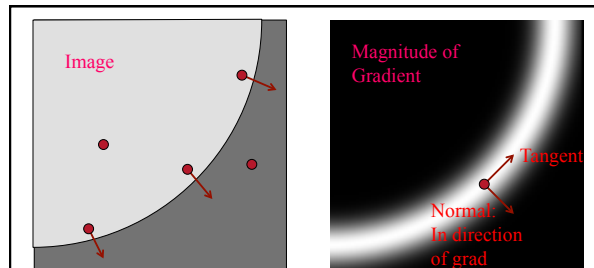


There are three major issues:

1. The gradient magnitude at different scales is different; which scale should we choose?
2. The gradient magnitude is large along thick trail; how do we identify the significant points?
3. How do we link the relevant points up into curves?

CSE152, Winter 2013

Intro Computer Vision



We wish to mark points along a curve where the magnitude is biggest. We can do this by looking for a maximum along a slice orthogonal to the curve (non-maximum suppression). These points should form a curve.

CSE152, Winter 2013

Intro Computer Vision

Non-maximum suppression

Loop over every point q in the image, decide whether q is a candidate edge point

Using gradient direction at q , find two points p and r on adjacent rows (or columns).

If $|\nabla I(q)| > |\nabla I(p)|$ and $|\nabla I(q)| > |\nabla I(r)|$ then q is a candidate edgel

CSE152, Winter 2013

Non-maximum suppression

Loop over every point q in the image, decide whether q is a candidate edge point

Using gradient direction at q , find two points p and r on adjacent rows (or columns). **p & r are found by interpolation**

If $|\nabla I(q)| > |\nabla I(p)|$ and $|\nabla I(q)| > |\nabla I(r)|$ then q is a candidate edgel

CSE152, Winter 2013

The Canny Edge Detector

original image (Lena)

(Example from Srinivasa Narasmihan) Intro Computer Vision

CSE152, Winter 2013

The Canny Edge Detector

magnitude of the gradient

(Example from Srinivasa Narasmihan) Intro Computer Vision

CSE152, Winter 2013

The Canny Edge Detector

After non-maximum suppression

(Example from Srinivasa Narasmihan) Intro Computer Vision

CSE152, Winter 2013

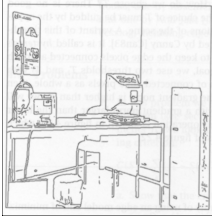
An Idea: Single Threshold

1. Smooth Image
2. Compute gradients & Magnitude
3. Non-maximal suppression
4. Compare to a threshold: T

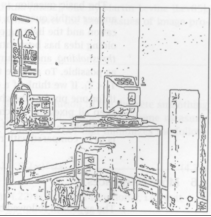
(Example from Srinivasa Narasmihan) Intro Computer Vision

CSE152, Winter 2013

An OK Idea: Single Threshold



T=15

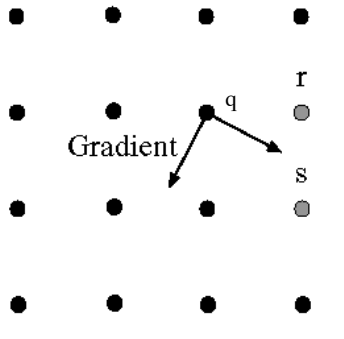


T=5

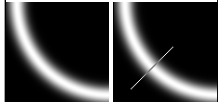
1. Smooth Image
2. Compute gradients & Magnitude
3. Non-maximal suppression
4. Compare to a threshold: T

CSE152, Winter 2013 Intro Computer Vision

A Better Idea: Linking + Two Thresholds



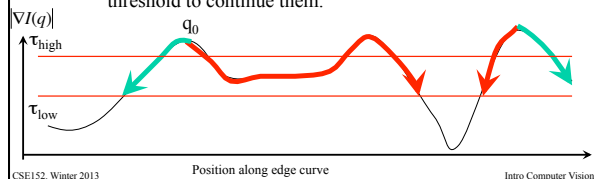
Linking: Assume the marked point q is an edge point. Then we construct the tangent to the edge curve (which is normal to the gradient at that point) and use this to predict the next points (either r or s).



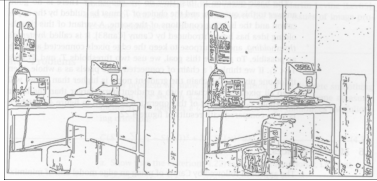
CSE152, Winter 2013 Intro Computer Vision

A Better Idea: Hysteresis Thresholding

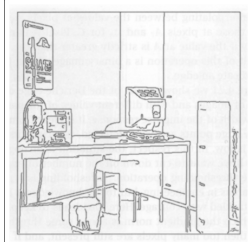
- Define two thresholds τ_{low} and τ_{high} .
- Starting with output of nonmaximal suppression, find a point q_0 where $|\nabla I(q_0)| > \tau_{high}$, and $|\nabla I(q_0)|$ is a local maximum.
- Start tracking an edge chain at pixel location q_0 in one of the two directions
- Stop when gradient magnitude $< \tau_{low}$.
– i.e., use a high threshold to start edge curves and a low threshold to continue them.



Single Threshold



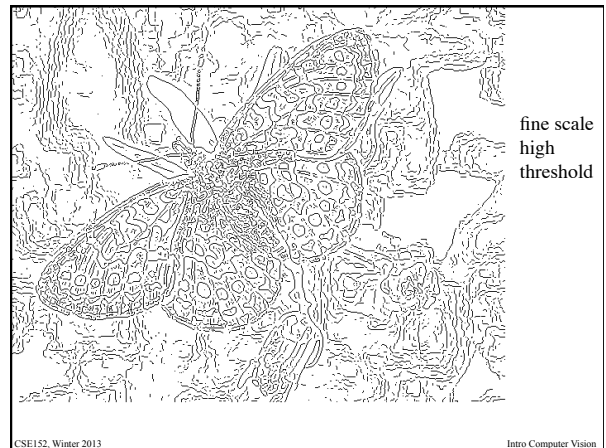
T=15 T=5

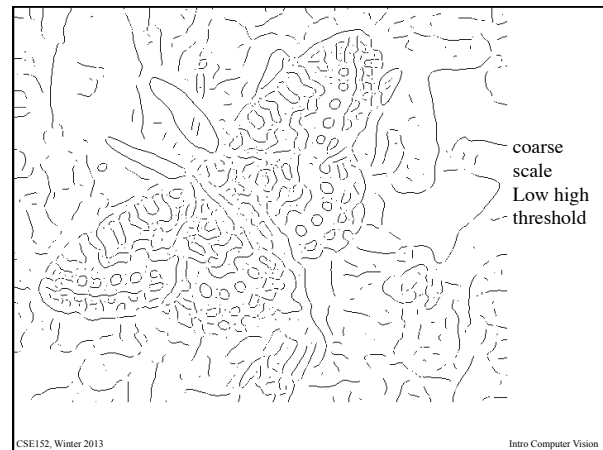
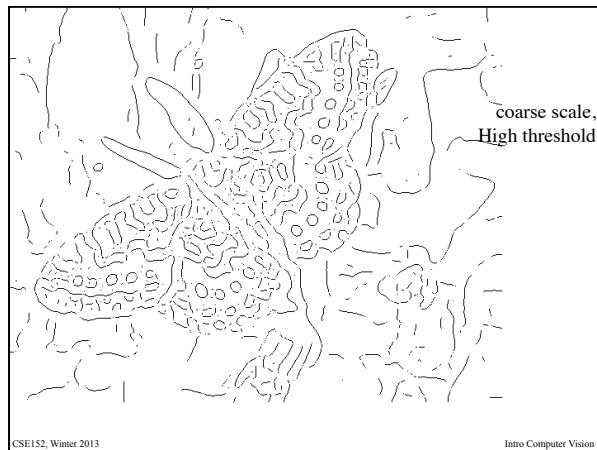


Hysteresis
 $T_h=15 \ T_l=5$

Hysteresis thresholding

CSE152, Winter 2013 Intro Computer Vision





Why is Canny so Dominant

- Still widely used after 20 years.
 1. Theory is nice
 2. Details good (magnitude of gradient, non-max suppression).
 3. Hysteresis thresholding an important heuristic.
 4. Code was distributed.

CSE152, Winter 2013

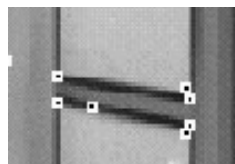
Intro Computer Vision

Corner Detection

CSE152, Winter 2013

Intro Computer Vision

Feature extraction: Corners and blobs

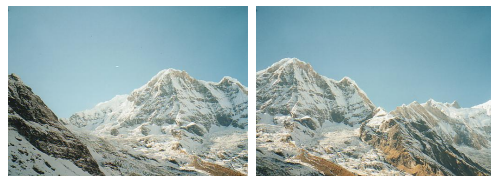


CSE152, Winter 2013

Intro Computer Vision

Why extract features?

- Motivation: panorama stitching
 - We have two images – how do we combine them?

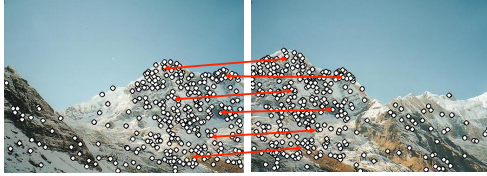


CSE152, Winter 2013

Intro Computer Vision

Why extract features?

- Motivation: panorama stitching
 - We have two images – how do we combine them?



Step 1: extract features

Step 2: match features

CSE152, Winter 2013

Intro Computer Vision

Why extract features?

- Motivation: panorama stitching
 - We have two images – how do we combine them?



Step 1: extract features

Step 2: match features

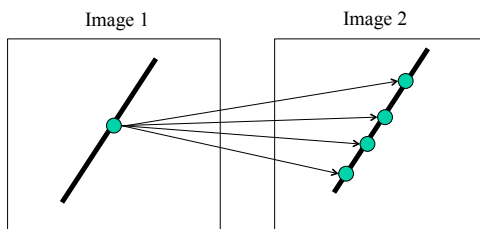
Step 3: align images

CSE152, Winter 2013

Intro Computer Vision

Corners contain more info than lines.

- A point on a line is hard to match.

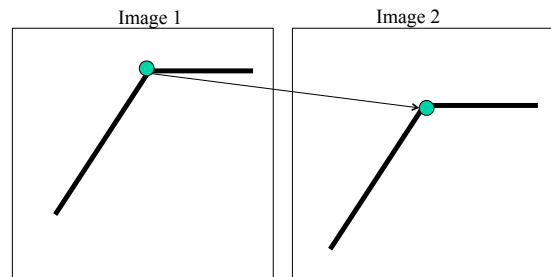


CSE152, Winter 2013

Intro Computer Vision

Corners contain more info than lines.

- A corner is easier to match

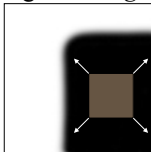


CSE152, Winter 2013

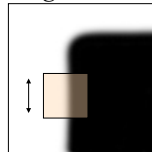
Intro Computer Vision

The Basic Idea

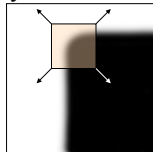
- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give a *large change* in intensity



“flat” region:
no change in
all directions



“edge”:
no change along
the edge
direction

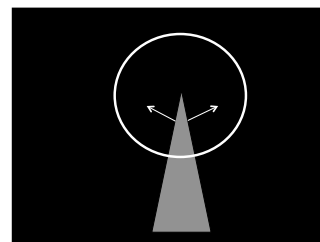


“corner”:
significant
change in all
directions

Source: A. Efros
CSE152, Winter 2013

Intro Computer Vision

Finding Corners



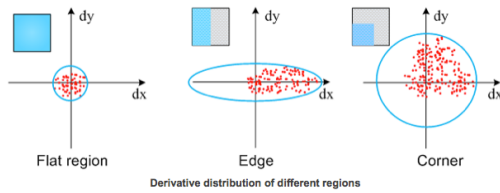
Intuition:

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

CSE152, Winter 2013

Intro Computer Vision

Distribution of gradients for different image patches



CSE152, Winter 2013

Intro Computer Vision

Finding Corners

For each image location (x,y) , we create a matrix $C(x,y)$:

Sum over a small region

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

$$C(x,y) = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

Matrix is symmetric

WHY THIS?

CSE152, Winter 2013

Intro Computer Vision

Because C is a symmetric positive definite matrix, it can be factored as:

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

where R is a 2×2 rotation matrix and λ_1 and λ_2 are non-negative.

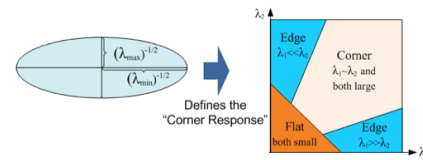
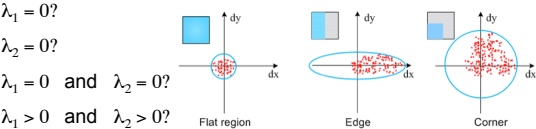
1. λ_1 and λ_2 are the Eigenvalues of C .
2. The columns of R are the Eigenvectors of C .
3. Eigenvalues can be found by solving the characteristic equation $\det(C - \lambda I) = 0$ for λ .

CSE152, Winter 2013

Intro Computer Vision

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 $\lambda_2 > 0$?



CSE152, Winter 2013

Intro Computer Vision

So, to detect corners

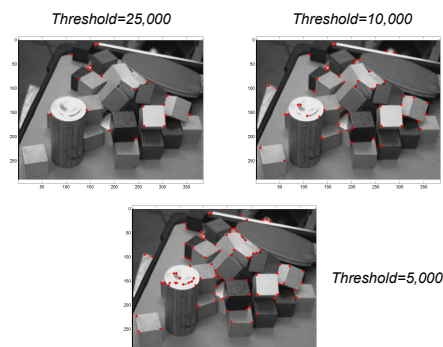
- Filter image with a Gaussian.
- Compute the gradient everywhere.
- Move window over image and construct C over the window.
- Use linear algebra to find λ_1 and λ_2 .
- If they are both big, we have a corner.
 1. Let $e(x,y) = \min(\lambda_1(x,y), \lambda_2(x,y))$
 2. (x,y) is a corner if it's local maximum of $e(x,y)$ and $e(x,y) > \tau$

Parameters: Gaussian std. dev, window size, threshold

CSE152, Winter 2013

Intro Computer Vision

Corner Detection Sample Results

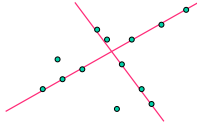


CSE152, Winter 2013

Intro Computer Vision

What to do with edges?

- Segment linked edge chains into curve features (e.g., line segments).
- Group unlinked or unrelated edges into lines (or curves in general).



- Accurately fitting parametric curves (e.g., lines) to grouped edge points.

