Edge Detection
Introduction to Computer Vision
CSE 152
Lecture 9

Announcements
• Assignment 2 assigned and due Tuesday, May 4.
• Midterm: Thursday, May 6.

Convolution
\[ R(i, j) = \sum_{h=-m/2}^{m/2} \sum_{k=-m/2}^{m/2} K(h, k)I(i-h, j-k) \]

Properties of convolution
Let \( f, g, h \) be images and \( \ast \) denote convolution
\[ f \ast g(x, y) = \int \int f(x-u, y-v)g(u, v) du dv \]
• Commutative: \( f \ast g = g \ast f \)
• Associative: \( f \ast (g \ast h) = (f \ast g) \ast h \)
• Linear: for scalars \( a \) & \( b \) and images \( f, g, h \)
  \( (af+bg) \ast h = af \ast h + bg \ast h \)
• Differentiation rule
  \[ \frac{\partial}{\partial x} (f \ast g) = \frac{\partial f}{\partial x} \ast g = f \ast \frac{\partial g}{\partial x} \]

Filters are templates
• Applying a filter at some point can be seen as taking a dot-product between the image and some vector
• Filtering the image is a set of dot products
• Insight
  – Filters look like the effects they are intended to find
  – Filters find effects they look like
Gaussian Noise: \( \sigma = 1 \)

Gaussian Noise: \( \sigma = 16 \)

An Isotropic Gaussian

- The picture shows a smoothing kernel proportional to

\[
\exp \left( -\frac{x^2 + y^2}{2\sigma^2} \right)
\]

(which is a reasonable model of a circularly symmetric fuzzy blob)

Smoothing with a Gaussian

Kernel:

Other Types of Noise

- Impulsive noise
  - randomly pick a pixel and randomly set to a value
  - saturated version is called salt and pepper noise

- Quantization effects
  - Often called noise although it is not statistical

- Unanticipated image structures
  - Also often called noise although it is a real repeatable signal.

Some other useful filtering techniques

- Median filter
- Anisotropic diffusion

Median filters: example

filters have width 5:
Median filters: analysis

median completely discards the spike, linear filter always responds to all aspects

median filter preserves discontinuities, linear filter produces rounding-off effects

DON’T become all too optimistic

Physical causes of edges

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

Object Boundaries

Surface normal discontinuities

Boundaries of materials properties
Boundaries of lighting

Profiles of image intensity edges

Noisy Step Edge

- Derivative is high everywhere.
- Must smooth before taking gradient.

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.

Numerical Derivatives

Take Taylor series expansion of \( f(x) \) about \( x_0 \):

\[
 f(x) = f(x_0) + f'(x_0)(x-x_0) + \frac{1}{2} f''(x_0)(x-x_0)^2 + \ldots
\]

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:

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

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

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.
2D Edge Detection: Canny

1. Filter out noise
   - Use a 2D Gaussian Filter. \( J = I * G \)
2. Take a derivative
   - Compute the magnitude of the gradient:
     \[
     \nabla J = (J_x, J_y) = \left( \frac{\partial J}{\partial x}, \frac{\partial J}{\partial y} \right) \text{ is the Gradient}
     \]
     \[
     \|\nabla J\| = \sqrt{J_x^2 + J_y^2}
     \]

Smoothing and Differentiation

- Need two derivatives, in x and y direction.
- Filter with Gaussian and then compute gradient, or
- Use a derivative of Gaussian
  - because differentiation can be implemented as convolution, and convolution is associative

Directional Derivatives

\[
\cos \theta \frac{\partial G_\theta}{\partial x} + \sin \theta \frac{\partial G_\theta}{\partial y}
\]

Finding derivatives

Is this \( dI/dx \) or \( dI/dy \)?

Next step

Can next step, be the same for 2-D edge detector as in 1-D detector??

NO!!

Let’s talk about corners first and return to edges later.

Corner Detection
Feature extraction: Corners

Why extract features?
• Motivation: panorama stitching
  – We have two images – how do we combine them?

Step 1: extract features
Step 2: match features

Corners contain more info than lines.
• A point on a line is hard to match.

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

Corners contain more info than lines.
• A corner is easier to match
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

Source: A. Efros

Edge Detectors Tend to Fail at Corners

Finding Corners

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

Formula for Finding Corners

We look at matrix:

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

Matrix is symmetric

WHY THIS?

For intuition, consider case where:

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

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 \( \lambda_2 > 0 \)?

General Case:

From Linear Algebra we haven’t talked about it follows that since C is symmetric:

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

So every case is like one on last slide.
General Case

Since $C$ is symmetric, we have

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

We can visualize $C$ as an ellipse with axis lengths determined by the eigenvalues and orientation determined by $R$.

Ellipse equation:

$$[u \ v] M [u \ v] = \text{const}$$

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 $\lambda_1$ and $\lambda_2$.
- 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

Corner Detection Sample Results

Threshold=25,000

Threshold=10,000

Threshold=5,000