Announcements

- HW1 report handed in today
- HW2 is posted, written assignment.
- Last lecture filtering
- Today: Edges

Convolution: \( R = K \ast I \)

Kernel size is \( m+1 \) by \( m+1 \)

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

What is image filtering?

- Modify the pixels in an image based on some function of a local neighborhood of the pixels.

<table>
<thead>
<tr>
<th>Local image data</th>
<th>Some function</th>
<th>Modified image data</th>
</tr>
</thead>
</table>
| \( \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \) | \( \begin{bmatrix} 3 \end{bmatrix} \) | Shift

Shift

Blurring

original

Pixel offset

Coefficient

0.1

shifted

original

Pixel offset

Coefficient

0.1

Blurred (filter applied in both dimensions).
Additive noise

- \( I = S + N \). Noise doesn’t depend on signal.
- We’ll consider:
  \[ I_i = s_i + n_i \text{ with } E(n_i) = 0 \]
  \( s_i \) deterministic.
  \( n_i, n_j \) independent for \( i \neq j \)
  \( n_i, n_j \) identically distributed

Filtering to reduce noise

- Noise is what we’re not interested in.
- We’ll discuss simple, low-level noise today:
  - Light fluctuations; Sensor noise; Quantization effects; Finite precision
  - Not complex: shadows; extraneous objects.
- A pixel’s neighborhood contains information about its intensity.
- Averaging noise reduces its effect.

Smoothing by Averaging

Kernel

Sharpening

original

Sharpened original

Sharpening example

original

Sharpened

(differences are accentuated; constant areas are left untouched).
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)

Efficient Implementation

- Both, the BOX filter and the Gaussian filter are separable:
  - First convolve each row with a 1D filter
  - Then convolve each column with a 1D filter.

Fourier Transform

Discrete Fourier Transform (DFT) of \(I[x,y]\)

\[
F[u,v] = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} I[x,y] e^{-\frac{2\pi i}{N} (ux+vy)}
\]

Inverse DFT

\[
I[x,y] = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F[u,v] e^{\frac{2\pi i}{N} (ux+vy)}
\]

x,y: spatial domain
u,v: frequency domain
Implemented via the “Fast Fourier Transform” algorithm (FFT)

Here u and v are larger than in the previous slide.
### Phase and Magnitude

\[ \theta^2 = \cos \theta + i \sin \theta \]

- Fourier transform of a real function is complex
  - difficult to plot, visualize
  - instead, we can think of the phase and magnitude of the transform
- Phase is the phase of the complex transform
- Magnitude is the magnitude of the complex transform

### Using Fourier Representations

**Dominant Orientation**

**Limitations**: not useful for local segmentation

### The Fourier Transform and Convolution

- If \( H \) and \( G \) are images, and \( \mathcal{F}(\cdot) \) represents Fourier transform, then
  \[ \mathcal{F}(H \ast G) = \mathcal{F}(H) \mathcal{F}(G) \]
- Thus, one way of thinking about the properties of a convolution is by thinking of how it modifies the frequencies of the image to which it is applied.
- In particular, if we look at the power spectrum, then we see that convolving image \( H \) by \( G \) attenuates frequencies where \( G \) has low power, and amplifies those which have high power.
- This is referred to as the **Convolution Theorem**

### Various Fourier Transform Pairs

- **Important facts**
  - scale function down \( \Leftrightarrow \) scale transform up
  - i.e. high frequency = small details
  - The FT of a Gaussian is a Gaussian.
  
  [Compare to box function transform]

### Other Types of Noise

- **Impulsive noise**
  - randomly pick a pixel and randomly set ot 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: principle

Method:
1. rank-order neighbourhood intensities
2. take middle value
   - non-linear filter
   - no new grey levels emerge...

Median filters: example

Filters have width 5:

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

Physical causes of edges

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

Object Boundaries

Surface normal discontinuities
Edges are where change occurs in 1D.
- Change is measured by the derivative in 1D.
  - Biggest change, derivative has maximum magnitude.
  - Or the 2nd derivative is zero.

Nhiny step edge:
- Derivative is high everywhere.
- Must smooth before taking the gradient.

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: Two issues:
   - Should be a local maximum.
   - Should be sufficiently high.
2D Edge Detection: Canny

1. Filter out noise
   - Use a 2D Gaussian Filter. \( J = I \otimes G \)

2. Take a derivative
   - Compute the magnitude of the gradient:
     \[
     \nabla J = (J_x, J_y) = \left( \frac{\partial J}{\partial x}, \frac{\partial J}{\partial y} \right)
     \]
     is the Gradient
     \[
     ||\nabla J|| = \sqrt{J_x^2 + J_y^2}
     \]

What is the gradient?

\[
\left( \frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right) = (k, 0)
\]

Gradient direction is perpendicular to edge.
Gradient Magnitude measures edge strength.

Finding derivatives

Is this \( \frac{dl}{dx} \) or \( \frac{dl}{dy} \)?

Smoothing and Differentiation

- Need two derivatives, in x and y direction.
- We can use a derivative of Gaussian filter
  - because differentiation is convolution, and convolution is associative
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 trails; how do we identify the significant points?
3. How do we link the relevant points up into curves?

\[ \sigma = 1 \quad \sigma = 2 \]

The scale of the smoothing filter affects derivative estimates.

1 pixel \quad 3 pixels \quad 7 pixels

We wish to mark points along the curve where the magnitude is biggest. We can do this by looking for a maximum along a slice normal to the curve (non-maximum suppression). These points should form a curve. There are then two algorithmic issues: at which point is the maximum, and where is the next one?

Non-maximum suppression
At \( q \), we have a maximum if the value is larger than those at both \( p \) and at \( r \). Interpolate to get these values.

Hysteresis
- Track edge points by starting at point where gradient magnitude \( > \tau_{\text{high}} \).
- Follow edge in direction orthogonal to gradient.
- Stop when gradient magnitude \( < \tau_{\text{low}} \).
  - i.e., use a high threshold to start edge curves and a low threshold to continue them.

Predicting the next edge point
Assume the marked point 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 (here either \( r \) or \( s \)).
fine scale high threshold

coarse scale, high high threshold

coarse scale Low high threshold