Announcements

- HW0 due tomorrow at midnight
- HW1 coming soon
Convolution

Image (I)

Kernel (K)

Note: Typically Kernel is relatively small in vision applications.

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

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

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

• 2d convolution
  – Kernel is flipped over both axes
  \[
  R(i, j) = \sum_{h=-m/2}^{m/2} \sum_{k=-m/2}^{m/2} K(h, k)I(i-h, j-k)
  \]
  \[
  = \sum_{h=-m/2}^{m/2} \sum_{k=-m/2}^{m/2} K(-h, -k)I(i+h, j+k)
  \]

• 2d correlation
  – Kernel is not flipped
  \[
  R(i, j) = \sum_{h=-m/2}^{m/2} \sum_{k=-m/2}^{m/2} K(h, k)I(i+h, j+k)
  \]

• When kernel is symmetric, convolution and correlation give the same result

Properties of convolution

For images A,B,C and scalar k
• Linear: A*(kB+C) = kA*B+A*C
• Shift invariant: A*B(x-k) = (A*B)(x-k)
• Commutative: A*B = B*A
• Associative: A*(B*C) = (A*B)*C
• Identity kernel: impulse I = [0, 0, 1, 0, 0]
• Differentiation for continuous functions
  \[
  \frac{\partial}{\partial x} (f * g) = \frac{\partial f}{\partial x} * g = f * \frac{\partial g}{\partial x}
  \]
Convolving two kernels

- By associativity $k_1 *(k_2 * I) = (k_1 * k_2) * I$, but how do you convolve two kernels?
  - If $k_1$ and $k_2$ are both $m \times m$, won’t $k_1 * k_2$ just be $1 \times 1$?
  - You need to pad one of kernels with zeros first and then convolve
  - E.g., let $k_1$ be $m \times m$ and $k_2$ be $n \times n$, then create $k_1'$ pad $k_1$ with $|n-m|/2$ rows/cols of zeros on each side, and then compute convolution: $k_1' * k_2$.
  - Then $k_1 * (k_2 * I) = (k_1' * k_2) * I$

Recap cont

- Image noise
- Box filter
- Gaussian filter
- Median filter
Today

- Image gradients, edges
- Image correspondence
- Corner detection
- Image feature descriptors – SIFT

Image Segmentation and Edges

- Image Segmentation is the process of dividing an image into connected regions such that pixels within a region share certain characteristics (color, texture, brightness, etc.)
- Boundaries or edges divide segmented regions.

[From Berkeley Segmentation Dataset] 13 Regions
Image Segmentation

[From Berkeley Segmentation Dataset]

Related Topics:
Semantic and Instance Segmentation
Edges in Natural Images

What Causes an Edge?

- Depth discontinuity
- Surface orientation discontinuity
- Illumination discontinuity (e.g., shadow)
- Specular reflection of other discontinuity
- Reflectance discontinuity (i.e., change in surface material properties)
Noisy 1D 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, first derivative has maximum magnitude
- Or, second derivative is zero.
Numerical Derivatives of Sampled Signal

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 of the expansion, we have $f(x_0)$ and

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

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

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

Derivatives can be implemented as a convolution with kernels

First Derivative: $\begin{bmatrix} \frac{1}{2h} & 0 & -\frac{1}{2h} \end{bmatrix}$
Second Derivative: $\begin{bmatrix} \frac{1}{h^2} & -\frac{2}{h^2} & \frac{1}{h^2} \end{bmatrix}$

When we only want to find min/max of first derivative or zero crossings of second derivative, we can ignore the denominators and use kernels

First Derivative: $[1 0 -1]$
Second Derivative: $[1 -2 1]$
Implementing 1-D Edge Detection

1. Filter out noise: convolve with Gaussian
2. Take a derivative: convolve with \( [1 \ 0 \ -1] \)
   We can combine steps 1 and 2
3. Find the peaks of \(|df/dx|\)
   Two issues:
   - Should be a local maximum of \(|df/dx|\)
   - Should be greater than a threshold: \(|df/dx| > \tau\)

![](image)

Extension to 2D

1. Convolve image I with a 2D Gaussian kernel to remove noise: \( J = I \ast G_{\sigma} \)
2. In 2D, the equivalent of the derivative is the gradient (a vector).
   Convolve with \( k_x = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix} \), \( k_y = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} \)
   \( \nabla J = (J_x, J_y) = \left( \frac{\partial J}{\partial x}, \frac{\partial J}{\partial y} \right) \) is the gradient
   \( \|\nabla J\| = \sqrt{J_x^2 + J_y^2} \) is the magnitude of the gradient
   \( \tan^{-1}(J_x, J_y) \) is the direction of the gradient
Finding derivatives

Is this $dI/dx$ or $dI/dy$?

Magnitude of Gradient

- Edges are where the magnitude of the gradient is large
- Finding local maxima and/or thresholding won’t work in 2D
  - Maxima are isolated points, not curves
  - Thresholding can lead to thick edges if too high or gaps if too low
Magnitude of Gradient

- Edges are where the magnitude of the gradient is large
- Solution
  1. Non-maxima supression
  2. Hysteresis Thresholding
- See Canny Edge Detector for details.

Correspondence
Correspondence Estimation

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

Step 1: extract features
Step 2: match features
Correspondence Estimation

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

Step 1: Extract features in each image
Step 2: Match features across images
Step 3: Align images and determine a transformation

Image matching

by Diva Sian
by xwashford
Harder case

by Diva Sian

by scgbt

Harder still?

NASA Mars Rover images
Answer below (look for tiny colored squares…)  

![Image features and descriptors](image)

Image features and descriptors

- Given input image
  - Detect feature locations (aka corners)
  - Compute a feature descriptor (vector of numbers)
- Would like features and descriptors to be independent of:
  - Position in the image
  - Scale
  - Image orientation
  - Lighting
  - Viewpoint
Corner Detection

Rotation & Position Covariant

Feature extraction: Corners and blobs
Finding Corners

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

Distribution of gradients for different image patches [if they were noisy]

Derivative distribution of different regions
Formula for Finding Corners

The distribution of gradients in the patch at location \((x,y)\) can be described by the second moment matrix \(C(x,y)\)

Sum over a patch \(W\)

\[
C(x, y) = \begin{bmatrix}
\sum_{w} \left( \frac{\partial I}{\partial x} \right)^2 & \sum_{w} \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \\
\sum_{w} \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} & \sum_{w} \left( \frac{\partial I}{\partial y} \right)^2
\end{bmatrix}
\]

Matrix is symmetric

General Case

Since \(C(x,y)\) is symmetric, it can be factored as

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

where \(R\) is a 2x2 rotation matrix. \(\lambda_1, \lambda_2\) are non-negative and are Eigenvalues of \(C\). The columns of \(R\) are Eigenvectors.

We can visualize \(C\) as an ellipse with axis lengths determined by the eigenvalues and orientation determined by \(R\)
Shi-Tomasi Corner Detector

1. Filter image with a Gaussian
2. Compute the gradient everywhere
3. Move window W over image and construct C for each window location (x,y)

\[
C(x, y) = \frac{\sum (\frac{\partial I}{\partial x})^2}{\sum \frac{\partial I}{\partial x} \frac{\partial I}{\partial y}} \left[ \frac{\sum (\frac{\partial I}{\partial y})^2}{\sum \frac{\partial I}{\partial x} \frac{\partial I}{\partial y}} \right]
\]

4. For each window location, use linear algebra to find \( \lambda_1 \) and \( \lambda_2 \)
5. If \( \lambda_1 \) and \( \lambda_2 \) are both large, we have a corner
   a. Let \( r(x, y) = \min(\lambda_1(x, y), \lambda_2(x, y)) \)
   b. \((x, y)\) is a corner if it is a local maximum of \( r(x, y) \) and \( r(x, y) > \tau \)
6. Parameters: Gaussian std. dev, window size, threshold
Corner Detection Sample Results

Threshold=25,000

Threshold=10,000

Threshold=5,000

Corner Detector: Workflow

Slide credit: http://vims.cis.udel.edu/~chandra/
Corner Detector: Workflow

Compute corner response $r(x,y)$

Find points with large corner response: $r(x,y) > \tau$
Corner Detector: Workflow

Take only the points of local maxima of \( r(x,y) \)
Invariance and covariance

• Are locations invariant to photometric transformations
• and covariant to geometric transformations?
  – Invariance: image is transformed and corner locations do not change
  – Covariance: if we have two transformed versions of the same image, features should be detected in corresponding locations

• Note: Shi-Thomasi corners are position and orientation covariant, but they are not scale covariant due to fixed Gaussian filter standard deviation $\sigma$ and the window size.

• They are largely invariant to scaling of intensity, except for thresholding
Idea of SIFT

- Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters.

SIFT Features