EECS 442 – Computer vision

Detectors part I

- Edge feature detectors
- Corner feature detectors

Reading: [FP] Chapters: 8,9

Some slides of this lectures are courtesy of prof F. Li, prof S. Lazebnik, and various other lecturers
Goal:
Identify interesting regions from the images (edges, corners, blobs…)

Descriptors

e.g. SIFT

Matching / Indexing / Recognition
Linear filtering

• Convolution:

\[(f * g)[m, n] = \sum_{k,l} f[k, l] g[m - k, n - l]\]

• Smoothing

• Differentiation
Smoothing with a Gaussian

• Weight contributions of neighboring pixels by nearness

\[ G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}} \]

5 x 5, \( \sigma = 1 \)

Slide credit: Christopher Rasmussen
Smoothing with a Gaussian
Differentiation and convolution

\[
\frac{\partial f}{\partial x} \approx \frac{f(x_{n+1}, y) - f(x_n, y)}{\Delta x}
\]

Original Image

![Original Image](Image)

2D Kernel

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>-1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>-1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

2D Kernel

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-2</td>
<td>-1</td>
<td>0</td>
</tr>
<tr>
<td>-2</td>
<td>-1</td>
<td>0</td>
</tr>
<tr>
<td>-2</td>
<td>-1</td>
<td>0</td>
</tr>
<tr>
<td>-2</td>
<td>-1</td>
<td>0</td>
</tr>
</tbody>
</table>

Rudimentary edge detector!
Edge detection
What causes an edge?

Identifies sudden changes in an image

- Depth discontinuity
- Surface orientation discontinuity
- Reflectance discontinuity (i.e., change in surface material properties)
- Illumination discontinuity (e.g., shadow)
Edge Detection

- Criteria for **optimal edge detection** (Canny 86):
  - **Good detection accuracy**:
    - minimize the probability of false positives (detecting spurious edges caused by noise),
    - false negatives (missing real edges)
  - **Good localization**:
    - edges must be detected as close as possible to the true edges.
  - **Single response constraint**:
    - minimize the number of local maxima around the true edge (i.e. detector must return single point for each true edge point)
Edge Detection

- Examples:
  - True edge
  - Poor robustness to noise
  - Poor localization
  - Too many responses
Designing an edge detector

• **Two ingredients:**

  • Use derivatives (in x and y direction) to define a location with high gradient.

  • Need **smoothing** to reduce noise prior to taking derivative.
\[ \frac{d}{dx} (f \ast g) = \left( \frac{d}{dx} g \right) \ast f = \text{“derivative of Gaussian” filter} \]
Canny Edge Detection

- Most widely used edge detector in computer vision.
- First derivative of the Gaussian closely approximates the operator that optimizes the product of signal-to-noise ratio and localization.
- Analysis based on "step-edges" corrupted by "additive Gaussian noise".
Canny Edge Detection

**Steps:**

1. Gaussian smoothing
2. & Derivative = Derivative of Gaussian
3. Find magnitude and orientation of gradient
4. Extract edge points: ‘Non-maximum suppression’
5. Linking and thresholding ‘Hysteresis’:

- **Matlab:** `edge(I, ‘canny’)`
Canny Edge Detector
First 2 Steps

• Smoothing

\[ I' = g(x, y) * I \]

\[ g(x, y) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2 + y^2}{2\sigma^2}} \]

• Derivative

\[ S = \nabla(g * I) = (\nabla g) * I = \nabla g = \begin{bmatrix} \frac{\partial g}{\partial x} \\ \frac{\partial g}{\partial y} \end{bmatrix} = \begin{bmatrix} g_x \\ g_y \end{bmatrix} \]

\[ = \begin{bmatrix} g_x \\ g_y \end{bmatrix} * I = \begin{bmatrix} g_x * I \\ g_y * I \end{bmatrix} = \begin{bmatrix} S_x \\ S_y \end{bmatrix} = \text{gradient vector} \]
Canny Edge Detector
Derivative of Gaussian

\[ g(x, y) \]

\[ g_x(x, y) \]

\[ g_y(x, y) \]
Canny Edge Detector
First 2 Steps
Canny Edge Detector
Third Step

- magnitude and direction of

\[ S = \begin{bmatrix} S_x & S_y \end{bmatrix} \]

magnitude = \[ \sqrt{(S_x^2 + S_y^2)} \]

direction = \( \theta = \tan^{-1} \frac{S_y}{S_x} \)
Increased smoothing:
• Eliminates noise edges.
• Makes edges smoother and thicker.
• Removes fine detail.
Canny Edge Detector - Fourth Step

Non maximum suppression
Canny Edge Detector - Fourth Step

1. Initialize:
   - Slice gradient magnitude along the gradient direction
   - Mark the point along the slide where the magnitude is max

2. Propagate chain from current point:
   - Predict next points using the normal to the gradient at that point
   - Find which point is a local max magnitude in gradient direction
   - Retain in magnitude > T
Examples:

Non-Maximum Suppression

Original image  Gradient magnitude  Non-maxima suppressed

courtesy of G. Loy

Slide credit: Christopher Rasmussen
Canny Edge Detector
Step 5: Thresholding

- Set a threshold $T$ to suppress gradients with magnitude $< T$
high threshold
(strong edges)

low threshold
(weak edges)
**Hysteresis**: A lag or momentum factor

- **Idea**: Maintain two thresholds $k_{\text{high}}$ and $k_{\text{low}}$
  - Use $k_{\text{high}}$ to find strong edges to start edge chain
  - Use $k_{\text{low}}$ to find weak edges along the edge chain

- **Typical ratio of thresholds** is roughly $k_{\text{high}} / k_{\text{low}} = 2$
hysteresis threshold
Effect of $\sigma$ (Gaussian kernel spread/size)

- The choice of $\sigma$ depends on desired behavior
  - large $\sigma$ detects large scale edges
  - small $\sigma$ detects fine features

Source: S. Seitz
Demo

http://www.cs.washington.edu/research/imagedatabase/demo/edge/
Other edge detectors:

- Sobel
- Canny-Deriche
- Differential
Extract useful building blocks: Corners
Extract useful building blocks: blobs
• **Repeatability**
  – The same feature can be found in several images despite geometric and photometric transformations

• **Saliency**
  – Each feature is found at an “interesting” region of the image

• **Locality**
  – A feature occupies a “relatively small” area of the image;
Repeatability

• Rotation
• Affine

Illumination invariance

Scale invariance

Pose invariance
• Rotation
• Affine
• Saliency

• Locality
Harris corner detector

Harris Detector: Basic Idea

Explore intensity changes within a window as the window changes location.

"flat" region: no change in all directions

"edge": no change along the edge direction

"corner": significant change in all directions
Harris Detector: Mathematics

Change of intensity for the shift \([u, v]\):

\[
E(u, v) = \sum_{x,y} w(x, y) \left[ I(x+u, y+v) - I(x, y) \right]^2
\]

Window function \(w(x, y)\) =

1 in window, 0 outside

or

Gaussian
Harris Detector: Mathematics

For small shifts \([u, v]\) we have a \textit{bilinear} approximation:

\[
E(u, v) \approx [u, v] \begin{bmatrix} u \\ v \end{bmatrix}
\]

where \(M\) is a 2\(\times\)2 matrix computed from image derivatives:

\[
M = \sum_{x,y} w(x, y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \sum_{w} I_x^2 \\ \sum_{w} I_x I_y \\ \sum_{w} I_x I_y \\ \sum_{w} I_y^2 \end{bmatrix}
\]
Second-moment matrix

Sum over a small region around the hypothetical corner (we can omit “w”)

\[ M = \begin{bmatrix}
\sum W I_x^2 & \sum W I_x I_y \\
\sum W I_x I_y & \sum W I_y^2
\end{bmatrix} \]

Matrix is symmetric

How to compute this?

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

\[ (g_x * I)(g_y * I) \]
Second-moment matrix

First, consider case where dominant gradient directions aligned with x or y

\[ M = \begin{bmatrix}
\sum I_x^2 & \sum I_x I_y \\
\sum I_x I_y & \sum I_y^2 
\end{bmatrix} = ? = \begin{bmatrix}
\sum I_x^2 & 0 \\
0 & \sum I_y^2 
\end{bmatrix} \]

\[ \text{Eig}(M) = ? = \begin{bmatrix}
\lambda_1 & 0 \\
0 & \lambda_2 
\end{bmatrix} \]
Second-moment matrix

First, consider case where dominant gradient directions aligned with x or y

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

\[ \text{Eig}(M) = \begin{bmatrix}
\lambda_1 & 0 \\
0 & 0
\end{bmatrix} \]

If either \( \lambda \) is close to 0, then this is \textbf{an edge}
Second-moment matrix

First, consider case where dominant gradient directions aligned with x or y

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

If both \( \lambda \)s are close to 0, then this is a flat region
Second-moment matrix

For generic window alignments, the eigenvalue decomposition of $M$ returns similar information:

\[
M = \begin{bmatrix}
\sum I_x^2 & \sum I_x I_y \\
\sum I_x I_y & \sum I_y^2 \\
\end{bmatrix} = ?
= U^{-1} \begin{bmatrix}
\lambda_1 & 0 \\
0 & \lambda_2 \\
\end{bmatrix} U
\]

- Lambda 1, 2 are the eigenvalues of $M$
- Eigenvectors allow to compute the transformation to “rectify” the corner
Classification of image points using eigenvalues of $M$:

- $\lambda_1$ and $\lambda_2$ are small; $E$ is almost constant in all directions.
- $\lambda_2 \gg \lambda_1$; “Corner” $\lambda_1$ and $\lambda_2$ are large, $\lambda_1 \sim \lambda_2$; $E$ increases in all directions.
- $\lambda_1 \gg \lambda_2$; “Edge” $\lambda_1$ and $\lambda_2$ are large, $\lambda_1 \sim \lambda_2$; $E$ increases in all directions.
- “Flat” region $\lambda_1$ and $\lambda_2$ are small; $E$ is almost constant in all directions.
Harris Detector: Mathematics

Measure of corner response:

\[ R = \det M - k (\text{trace } M)^2 \]

\[ \det M = \lambda_1 \lambda_2 \]

\[ \text{trace } M = \lambda_1 + \lambda_2 \]

\[(k \text{ – empirical constant, } k = 0.04-0.06)\]
Harris Detector: Mathematics

- $R$ depends only on eigenvalues of $M$
- $R$ is large for a corner
- $R$ is negative with large magnitude for an edge
- $|R|$ is small for a flat region

\[ R > 0 \]
\[ R < 0 \]
\[ |R| \text{ small} \]
Harris Detector: Algorithm

- Filter image with Gaussian to reduce noise
- Compute magnitude of the x and y gradients at each pixel
- Construct $M$ in a window around each pixel (Harris uses a Gaussian window)
- Compute $\lambda$s of $M$
- Compute $R = \det M - k \left( \text{trace } M \right)^2$
- If $R > T$ a corner is detected; retain point of local maxima
Harris Detector: Workflow
Harris Detector: Workflow

Compute corner response $R$
Harris Detector: Workflow

Find points with large corner response: $R > \text{threshold}$
Harris Detector: Workflow

Take only the points of local maxima of $R$
Harris Detector: Workflow
Harris Detector: Some Properties

- Is Harris detector rotational invariant?

Corner response $R$ is invariant to image rotation

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

$R = R(\lambda_1, \lambda_1)$ doesn't change!
Harris Detector: Some Properties

• Is it scale invariant?

*Corner response $R$ is not scale invariant!*

All points will be classified as edges
Harris Detector: Some Properties

- Partial invariance to *affine intensity* changes
  \[ l \rightarrow s \ l + b \]

- Invariance to intensity shift \( l \rightarrow l + b \) (*why?*)
  (only derivatives are used)

- Not invariant to intensity scale: \( l \rightarrow a \ l \)
Next lecture:

- Detectors part 2
- Descriptors