CAP5415
Computer Vision

Yogesh S Rawat
yogesh@ucf.edu
HEC-241
Edge Detection

Lecture 4
Outline

• What is edge detection?
• Why we need edge detection?
• Challenges
  • Noise
• How to detect edges?
  • Prewit
  • Sobel
  • Marr-Hildreth
  • Canny
Edge Detection

• Identify sudden changes in an image
  • Semantic and shape information
  • Marks the border of an object
  • More compact than pixels
Origins of Edges

- Edges are caused by a variety of factors

- Surface normal discontinuity
- Depth discontinuity
- Surface color discontinuity
- Illumination discontinuity
Types of edges

• Edge models

Step edge  Ramp edge  Roof edge
Why edge detection?

- Extract useful information from images
  - Recognizing objects
- Recover geometry
Closeup of edges
Closeup of edges
Closeup of edges
Closeup of edges
Characterizing edges

• An edge is a place of rapid change in the image intensity function

Slide Credit: James Hays

9/3/2020
Intensity profile

![Intensity profile graph]

Source: D. Hoiem
With a little Gaussian noise
Effects of Noise

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

\[ f(x) \]

\[ \frac{d}{dx} f(x) \]

Where is the edge?

Credit to Seitz.
Effects of noise

• Difference filters respond strongly to noise
  • Image noise results in pixels that look very different from their neighbors
  • Generally, the larger the noise the stronger the response

• What can we do about it?
Solution: smooth first

To find edges, look for peaks in $\frac{d}{dx}(f \ast g)$.
Derivative theorem of convolution

• Convolution is differentiable:

\[
\frac{d}{dx} (f * g) = f * \frac{d}{dx} g
\]

• This saves us one operation:

Source: S. Seitz
Solution: Smoothing

- Smoothing removes noise, but blurs edge.
Evaluate Edge Detection

\[
\text{precision} = \frac{\text{GT} \cap \text{RM}}{\text{RM}} = \frac{\text{TP}}{\text{RM}}
\]

\[
\text{recall} = \frac{\text{GT} \cap \text{RM}}{\text{GT}} = \frac{\text{TP}}{\text{GT}}
\]

Ground Truth (GT)
Results of Method (RM)
True Positives (TP)
True Negatives (TN)
False Negatives (FN)
False Positives (FP)
Design Criteria for Edge Detection

• Good detection: find all real edges, ignoring noise or other artifacts
• Good localization
  • as close as possible to the true edges
  • one point only for each true edge point

True edge

Poor robustness to noise

Poor localization

Too many responses
45 years of boundary detection

[Pre deep learning]

Source: Arbelaez, Maire, Fowlkes, and Malik. TPAMI 2011 (pdf)
Prewitt and Sobel Edge Detector

• Compute derivatives
  • In $x$ and $y$ directions

• Find gradient magnitude

• Threshold gradient magnitude
Discrete derivative - revisit

$$\frac{df}{dx} = f(x) - f(x-1) = f'(x)$$  Backward difference

$$\frac{df}{dx} = f(x) - f(x+1) = f'(x)$$  Forward difference

$$\frac{df}{dx} = f(x+1) - f(x-1) = f'(x)$$  Central difference
## Derivative Masks

<table>
<thead>
<tr>
<th>Type</th>
<th>Mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backward difference</td>
<td>$[-1 \ 1]$</td>
</tr>
<tr>
<td>Forward difference</td>
<td>$[1 \ -1]$</td>
</tr>
<tr>
<td>Central difference</td>
<td>$[-1 \ 0 \ 1]$</td>
</tr>
</tbody>
</table>
Image derivative

Given function

\[ f(x, y) \]

Gradient vector

\[ \nabla f(x, y) = \begin{bmatrix} \frac{\partial f(x, y)}{\partial x} \\ \frac{\partial f(x, y)}{\partial y} \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \end{bmatrix} \]

Gradient magnitude

\[ |\nabla f(x, y)| = \sqrt{f_x^2 + f_y^2} \]

Gradient direction

\[ \theta = \tan^{-1} \frac{f_x}{f_y} \]
Example

- Original image
- Laplacian operator
- Horizontal derivative
- Vertical derivative
Prewitt Edge Detector

- **image**
  - Average smoothing in \( x \)
  - Blurred
  - Derivative filtering in \( x \)
  - Edges in \( x \)

- **image**
  - Average smoothing in \( y \)
  - Blurred
  - Derivative filtering in \( y \)
  - Edges in \( y \)
Sobel Edge Detector

The Sobel Edge Detector process involves the following steps:

1. **Image Blurring in x**: Applying an average smoothing filter in the x-direction. The filter mask is:
   \[
   \begin{bmatrix}
   1 & 1 \\
   2 & 2 \\
   1 & 1 
   \end{bmatrix}
   \]

2. **Derivative Filtering in x**: Using a derivative filter to detect edges in the x-direction. The filter mask is:
   \[
   \begin{bmatrix}
   1 & -1 
   \end{bmatrix}
   \]
   The result is:
   \[
   \begin{bmatrix}
   1 & 0 & -1 \\
   2 & 0 & -2 \\
   1 & 0 & -1 
   \end{bmatrix}
   \]

3. **Image Blurring in y**: Applying an average smoothing filter in the y-direction. The filter mask is:
   \[
   \begin{bmatrix}
   1 & 2 & 1 \\
   1 & 2 & 1 
   \end{bmatrix}
   \]

4. **Derivative Filtering in y**: Using a derivative filter to detect edges in the y-direction. The filter mask is:
   \[
   \begin{bmatrix}
   1 \\
   0 \\
   -1 \\
   -2 \\
   -1 
   \end{bmatrix}
   \]
   The result is:
   \[
   \begin{bmatrix}
   1 & 2 & 1 \\
   0 & 0 & 0 \\
   -1 & -2 & -1 
   \end{bmatrix}
   \]
Sobel Edge Detector

\[ \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \]

\[ \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \]

\[ \frac{d}{dx}I \]

\[ \frac{d}{dy}I \]

\[ \sqrt{\left(\frac{d}{dx}I\right)^2 + \left(\frac{d}{dy}I\right)^2} \]

Threshold → Edges
Sobel Edge Detector

\[ \frac{dI}{dx} \]

\[ \frac{dI}{dy} \]
Sobel Edge Detector

\[
\Delta = \sqrt{\left(\frac{d}{dx} I\right)^2 + \left(\frac{d}{dy} I\right)^2}
\]

\[\Delta \geq \text{Threshold} = 100\]
Sobel vs Prewitt

Source: Arbelaez, Maire, Fowlkes, and Malik. TPAMI 2011 (pdf)
Questions?

Sources for this lecture include materials from works by Mubarak Shah, Abhijit Mahalanobis, and D. Lowe

Other sources from James Hays, Lana Lazebnik, Steve Seitz, David Forsyth, David Lowe, Fei-Fei Li, and Derek Hoiem