Edge Detection
(points and lines too)

Image analysis

• Image Analysis
  - Derive “features” from an image
  - segment an image into its constituent parts or objects
    • how and why depends on application

• Segmentation
  - Hard
  - Maybe the hardest part
Marr’s Paradigm
(David Marr)

Low-Level Processing
(Early Vision)

Segmentation Feature Detection

Fish at (x,y)
Fish at (u,v)

Analysis
(Higher Vision)

Simple Feature Detection

• Points
• Lines
• Edges
Points

• Single pixel point

• Neighbors will have roughly the same intensity

• Point will be different intensity

Detection Mask

Apply this mask to each point in image

\[
\begin{array}{ccc}
-1 & -1 & -1 \\
-1 & 8 & -1 \\
-1 & -1 & -1 \\
\end{array}
\]

\(|R| > T\)

R is the response, T is a non-negative threshold. \(|\text{Responses}|\) greater than T are points
Detection Masks

Construct a mask to find “features” of interest

\[
\begin{array}{ccc}
-1 & -1 & -1 \\
2 & 2 & 2 \\
-1 & -1 & -1 \\
\end{array}
\]

\[|R| > T\]

T is a non-negative threshold

Line Detection

- Horizontal Line
- Vertical Line
- Diagonal Lines
**Line Detection Masks**

<table>
<thead>
<tr>
<th>Horizontal</th>
<th>+45°</th>
<th>Vertical</th>
<th>-45°</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1 -1 -1</td>
<td>-1 -1 2</td>
<td>-1 2 -1</td>
<td>2 -1 -1</td>
</tr>
<tr>
<td>2 2 2</td>
<td>-1 2 -1</td>
<td>-1 2 -1</td>
<td>-1 2 -1</td>
</tr>
<tr>
<td>-1 -1 -1</td>
<td>2 -1 -1</td>
<td>-1 2 -1</td>
<td>-1 -1 2</td>
</tr>
</tbody>
</table>

**Edge Detection**

- One strong salient image feature is an edge
- Discontinuity of pixel intensity
What is an edge?

- Real Edge
- False Edge (Occluding Edge)
- False Edge (Shadow)

Edge in an Image?
Edge Detection

• Gradient Operators
  - Sobel and Prewitt
  - Detect intensity discontinuity

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

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

Edge Detection

• Recall the Gradient for each pixel has two values

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

\[
\nabla f(x, y) = \begin{bmatrix}
G_x \\
G_y
\end{bmatrix}
\]
Edge Detection

• Compute Gradient Magnitude

\[ \nabla f = \text{mag}(\nabla f) = \left[ \left( \frac{\partial f}{\partial x} \right)^2 + \left( \frac{\partial f}{\partial y} \right)^2 \right]^{1/2} \]

or

\[ \nabla f = \text{mag}(\nabla f) \approx |G_x| + |G_y| \]

Edge Detection

• Assumption
  - Pixels on an edge should have a high gradient response
  - Choose pixels with high-response

\[ \nabla f(x, y) > T \]
Example

![Example Image](image1)

\[ \nabla f(x, y) \]

---

Example

Results with different threshold $T$
(Lets call this image $I_e$)
Edge Linking

- This gives us pixels candidates that lie on an edge

- What about the edges?
  - How do we extract edges from $I_e$

Examine the Gradient Angle

- Gradient Direction (or angle)
- This can be thought of as the normal to the edge

$$\Psi(\nabla f) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$
Edge Linking

Edge Direction

Edge Normal

\[ \Psi(\nabla f) = \tan^{-1}\left(\frac{G_y}{G_x}\right) \]

\[ G_y = 2 \]
\[ G_x = -2 \]

\[ \text{atan}(1) = -45^\circ \]

(which is really 45°)

Assume the Gradient was computed via Prewitt Mask

Edge Linking

\[ |\Psi(x, y)| \]

Edges should have similar edge normals
Simple Algorithm

• for each edge pixel \((x,y)\) in \(I_e\)
  - search the \(N_8\) neighbors \((x',y')\)
    • if \(|\nabla f(x, y) - \nabla f(x', y')| \leq T\)
    • and
    • if \(|\Psi(x, y) - \Psi(x', y')| \leq A\)
      - Link Pixels

Edge Linking

• This procedure produces a “chain” of pixels
  - We can think of the individual pixels as “edglets”

• These chains will be used as input for higher-level processing
Post Processing of Edge Chains

- Remove small edges
  - exclude based on the number of edglets in the chain

- Remove large edges
  - remove very long edges (look at the sum of the distance)

Example

Edges (Just an example)
Edge Detection

- Lots of parameters involved

- How do you chose T for edges, T for similar pixels, A for angles

- Application specific
  - We call these “Magic Numbers”

Edge Detection

- One thing we would like to do is have a one-pixel wide edge

- Previous method didn’t do this

- Use the Canny Edge Detector
Canny Edge Detection

• Three Step Detection
  - Noise Smoothing
    - Suppress as much noise as possible (without blurring out the true edge)
  - Edge Enhancement
    • Apply a filter that responds to edges
  - Edge Localization
    • thins wide edges to 1-pixel
    • establish a 2-level threshold
      Global edge + local edge

Canny Edge Detector

• apply CANNY_ENHANCER to I

• apply NONMAX_SUPPRESSION to output of CANNY_ENHANCER

• apply HYSTERESIS_THRESH to the output of NONMAX_SUPPRESSION
CANNY_ENHANCER

• Apply Gaussian smoothing to I
  - Gaussian filter
• For each pixel (i,j)
  - (a) Compute $G_x$ and $G_y$
  - (b) Compute $\text{mag}(G) = e_s(i,j)$
    - (call $e_s(i,j)$ edge strength)
  - (c) estimate the orientation of the edge
    - $\arctan(G_y/G_x)$
    - (call this $e_a(i,j)$ edge angle)
    - often $e_a(i,j) = \arctan(-G_y/G_x)$
    » Need to negate the Y-direction to flip the axis
    » Add $\pi$ to the negative values
    » Making the angles range from 0 to $\pi$

Gaussian Filter for Canny

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

\([-N,N]\) [\([-2,2]\) with $\sigma=1$]
NONMAX_SUPPRESSION

• for each pixel \((i,j)\)
  - Use \(e_a(i,j)\) to classify pixel’s direction
    - \(d = \{0, 45, 90, 135\}\)
    - check two neighbors that correspond to these directions (next slide)
  - If \(e_s(i,j)\) is less than either of its neighbors, \(In(i,j) = 0\)
  - else
    - \(In(i,j) = e_s(i,j)\)

<table>
<thead>
<tr>
<th>Direction</th>
<th>Neighbors</th>
<th>Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(d = 0)</td>
<td>n4, n6</td>
<td></td>
</tr>
<tr>
<td>(d = 45)</td>
<td>n7, n3</td>
<td></td>
</tr>
<tr>
<td>(d = 90)</td>
<td>n2, n8</td>
<td></td>
</tr>
<tr>
<td>(d = 135)</td>
<td>n1, n9</td>
<td></td>
</tr>
</tbody>
</table>
NONMAX_SUPPRESSION
(Before and After)

HISTTERESIS_THRESH

- Inputs: In, Eo, and two thresholds
  - tl, th (where tl < th)
- for each pixel
  - Locate the next unvisited edge pixel In(i,j) such that In(i,j) > th
  - Starting from In(i,j)
    - Follow the pixels in the directions perpendicular to the edge normal that have values In > tl

\( \text{th is global threshold. At least one pixel in the edge has to satisfy this. } \text{tl is local, the connecting edgelets have to satisfy this.} \)
## Perpendicular Directions

<table>
<thead>
<tr>
<th>Direction</th>
<th>P-Neighbors</th>
<th>Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>d = 0</td>
<td>n2 n8</td>
<td><img src="image1" alt="Representation" /></td>
</tr>
<tr>
<td>d = 45</td>
<td>n1 n9</td>
<td><img src="image2" alt="Representation" /></td>
</tr>
<tr>
<td>d = 90</td>
<td>n4 n6</td>
<td><img src="image3" alt="Representation" /></td>
</tr>
<tr>
<td>d = 135</td>
<td>n7 n3</td>
<td><img src="image4" alt="Representation" /></td>
</tr>
</tbody>
</table>

### Example

*th* = 150, *tl* = 50
Comparison

- Input
- Sobel
- Canny

Edge Linking

- Still need to link the edges
- Use technique from slide 23
- Or, if you don't care about direction and magnitude
  - Use connected component detector
Canny in practice

• Don’t perform the gaussian smoothing
• Instead use the Sobel operators

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

- Sobel operator combines a smoothing filter with a derivative filter
  • Saves time

Summary

• Simple Detection
  - Build an appropriate mask to generate the desired response

• Edge Detection
  - Relies on Image Gradient \( G_x \) and \( G_y \), Gradient Magnitude, Gradient Angle

• Canny Detector
  - Very common edge detector
  - Introduces NON_MAXSUPPRESSION
  - Uses a two level threshold
Active Research Areas

• Real-time edge detection
  - Video Rate edge detection

• Use other heuristics

• Edge detection in the face of noise

Active Research Areas

• Application specific edge detection
  - A-priori knowledge of the image
  - MRI, CT scans, etc.

• Compressed-domain edge detection
  - Use the compressed-components to find edges