Recap (Edge Detection)

- **Marr-Hildreth and Canny edge detectors**
  - Gaussian smoothing
  - Compute 2\textsuperscript{nd} order derivatives
    - In $x$ and $y$ directions
  - Find zero crossings
  - Threshold zero crossings

- **Difference between Marr-Hildreth and Canny**
  - Marr-Hildreth use 2\textsuperscript{nd} order derivative
  - Marr-Hildreth thresholds slope of zero-crossings
Marr-Hildreth Edge Detector

- Image
- $\Delta^2 g(x)$
- Find zero-crossings
- Compute slope
- Threshold

Canny Edge Detector

- Image
- $g(x,y)$
- Gradient magnitude
- Non-maximum suppression
- Hysteresis thresholding
- Gradient direction
Region Segmentation

Applications of Segmentation

- Object recognition
- MPEG-4 video compression
Object Recognition Using Region Properties

- Training
  - For all training samples of each model object
    - Segment the image
    - Compute region properties (features)

- Recognition
  - Given an image of unknown object,
    - Segment the image
    - Compute its feature vector
    - Compare with the training set

MPEG4 Compression
Object Based Compression

- Advantages of OBC
  - High compression ratio
  - Allows insertion deletion of objects

- How does it work?
  - Find objects (Object Segmentation)
  - Code objects and their locations
  - Build mosaics of globally static objects
  - Render scene at receiver
Clustering

Segmentation-Clustering
Region Segmentation

Layer Representation
Segmentation

- Find set of regions $R_1$, $R_2$, ..., $R_n$ such that
  \[ \bigcup_{i=1}^{n} R_i = I \]
  \[ \forall i \neq j, R_i \cap R_j = \emptyset \]
- All pixels in region $i$ satisfy some similarity constraint

Similarity Constraints

- All pixels in any sub-image must have the **same** gray levels.
- All pixels in any sub-image **must not differ** more than some threshold
- All pixels in any sub-image **may not differ** more than some threshold from the mean of the gray of the region
- The **standard deviation** of gray levels in any sub-image must be small.
Simple Segmentation

\[
B(x, y) = \begin{cases} 
1 & \text{if } I(x, y) < T \\
0 & \text{Otherwise}
\end{cases}
\]

\[
B(x, y) = \begin{cases} 
1 & \text{if } T_1 < I(x, y) < T_2 \\
0 & \text{Otherwise}
\end{cases}
\]

\[
B(x, y) = \begin{cases} 
1 & \text{if } I(x, y) \in Z \\
0 & \text{Otherwise}
\end{cases}
\]

Image Histogram

- Histogram graphs the number of pixels with a particular gray level as a function of the image of gray levels.
Segmentation Using Histogram
Simple Case

\[
\begin{align*}
B_1(x, y) &= \begin{cases} 
1 & \text{if } 0 < f(x, y) < T_1 \\
0 & \text{Otherwise}
\end{cases} \\
B_2(x, y) &= \begin{cases} 
1 & \text{if } T_1 < f(x, y) < T_2 \\
0 & \text{Otherwise}
\end{cases} \\
B_3(x, y) &= \begin{cases} 
1 & \text{if } T_2 < f(x, y) < T_3 \\
0 & \text{Otherwise}
\end{cases}
\end{align*}
\]
Realistic Histograms

- Smooth out noise
  - Convolve hist. by averaging or 1D Gaussian filter
Segmentation Using Histogram
Real image histograms

1. Compute the histogram of a given image.
2. Smooth the histogram by averaging peaks and valleys in the histogram.
3. **Detect good peaks by applying thresholds at the valleys.**
4. Segment the image into several binary images using thresholds at the valleys.
5. **Apply connected component algorithm to each binary image find connected regions.**

Good Peaks
Peakiness Test

\[
Peakiness = \left(1 - \frac{V_a + V_b}{2P}\right) \left(1 - \frac{N}{W.P}\right)
\]
Segmentation Using Histograms

- Select the valleys as thresholds
  - Apply threshold to histogram
  - Label the pixels within the range of a threshold with the same label, i.e., a, b, c … or 1, 2, 3 …

Connected Components

- Disjoint segments with the same labels need to be split

\[
\begin{bmatrix}
0 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 \\
\end{bmatrix}
\rightarrow
\begin{bmatrix}
0 & 0 & 0 & a & 0 \\
0 & 0 & 0 & a & a \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & c & c & 0 \\
0 & d & 0 & c & 0 \\
\end{bmatrix}
\]

May be added to segment c
Recursive Connected Component Algorithm

1. Scan the binary image left to right, top to bottom.
2. If there is an unlabeled pixel with a value of ‘1’ assign a new label to it.
3. Recursively check the neighbors of the pixel in step 2 and assign the same label if they are unlabeled with a value of ‘1’.
4. Stop when all the pixels of value ‘1’ have been labeled.

Sequential Connected Component Algorithm

1. Scan the binary image left to right, top to bottom.
2. If an unlabeled pixel has a value of ‘1’, assign a new label to it according to the following rules:
   \[0 \rightarrow 0\]
   \[1 \rightarrow L\]
   \[L \rightarrow L\]
   \[M \rightarrow M\]
   \[(Set \ L = M)\]
3. Determine equivalence classes of labels.
4. In the second pass, assign the same label to all elements in an equivalence class.

Figure 3.7: Recursive Connected Component Algorithm.

Figure 3.8: Sequential Connected Component Algorithm.
Sequential Connected Component Algorithm

\[
\begin{bmatrix}
0 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 \\
0 & 1 & 1 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
0 & 0 & 0 & a & 0 \\
b & b & 0 & a & a \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & c & c & 0 \\
0 & d & c & c & 0
\end{bmatrix}
\]

Equivalence class

Example Detecting Finger Tips (marked white)
Example
Segmenting a bottle image

Smoothed histogram
(averaging using mask
of size 5)
54 peaks (once)
After peakiness 18

Smoothed histogram
21 peaks (twice)
After peakiness 7

Smoothed histogram
11 peaks (three times)
After peakiness 4

93 peaks
Example
Segmenting a bottle image

Exercise

- Implement histogram based region segmentation (histogram computation, peakiness test, recursive connected component analysis).
- Report should include: Histograms, resulting region as binary images with various histograms generated with several smoothing operations and hard copy of the source code.
  - Smooth once find peaks.
  - Smooth twice find peaks.
  - Smooth 5 times find peaks.
- Due date: 28 September 2005
Suggested Reading

- Chapter 3, Mubarak Shah, “Fundamentals of Computer Vision”