CHAPTER 10

IMAGE SEGMENTATION
Image Segmentation

- Goal: Segmentation subdivides an image into its constituent regions or objects that have similar features according to a set of predefined criteria.
- features?
  - Intensity, Histogram, mean, variance, Energy, Texture, ....etc.
- Level of detail depends on application.
- Segmentation of nontrivial images is one of the most difficult tasks in image processing and computer vision.
- Still a very active research area.
- Segmentation accuracy often determines the eventual success or failure of computerized analysis procedures.
Most segmentations algorithms we will consider are based on one of two basic properties of intensity values:

- **Discontinuity:** The strategy is to partition an image based on abrupt changes in intensity (Edge-based segmentation).

- **Similarity:** The strategy is to partition an image into regions that are similar according to a set of predefined criteria (Region-based segmentation).
Detection of Discontinuities

- detect the three basic types of gray-level discontinuities
  - points, lines, edges
- the common way is to run a mask through the image

\[ R = w_1 z_1 + w_2 z_2 + \ldots + w_9 z_9 \]

\[ = \sum_{k=1}^{9} w_k z_k \]
Detection of Isolated Point

Point detection based on the second derivative (laplacian)

\[ \nabla^2 f(x, y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \]

For two dimension image

\[ \nabla^2 f(x, y) = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y) \]

can be achieved simply using the mask below

With extension to the diagonal

Points are detected at those pixels in the subsequent filtered image that are above a set threshold

The output is obtained using

\[ g(x, y) = \begin{cases} 
1 & \text{if } |R(x, y)| \geq T \\
0 & \text{otherwise} 
\end{cases} \]
Example

T is set to 90% of the highest absolute pixel value of the image in Fig. 10.2(c).
**LINE DETECTION**

![Image of line detection masks](image)

**FIGURE 10.6** Line detection masks. Angles are with respect to the axis system in Fig. 2.18(b).

- These filter masks would respond more strongly to lines.
- Note that the coefficients in each mask sum to zero, indicating a zero response from the masks in areas of constant gray level.
- If we are interested in detecting lines in a specified direction, we could use the mask associated with that direction and threshold its output.
- Note: these filters respond strongly to lines of one pixel thick.
Line Detection

- Apply every masks on the image
- Let $R_1$, $R_2$, $R_3$, $R_4$ denotes the response of the horizontal, $+45$ degree, vertical and $-45$ degree masks, respectively.
- If, at a certain point in the image
  \[ |R_i| > |R_j|, \]
  for all $j \neq i$, that point is said to be more likely associated with a line in the direction of mask $i$. 
Line Detection

- Alternatively, if we are interested in detecting all lines in an image in the direction defined by a given mask, we simply run the mask through the image and threshold the absolute value of the result.

- The points that are left are the strongest responses, which, for lines one pixel thick, correspond closest to the direction defined by the mask.
Example

**FIGURE 10.4**
Illustration of line detection.
(a) Binary wire-bond mask.
(b) Absolute value of result after processing with $-45^\circ$ line detector.
(c) Result of thresholding image (b).
Edge Detection

- The most common approach for detecting meaningful discontinuities in gray level.
- We discuss approaches for implementing
  - First-order derivative (gradient operator)
  - Second-order derivative (laplacian operator)
- Here, we will talk only about their properties for edge detection.
- We have introduced both derivatives in chapter 3
Ideal and Ramp Edges

Model of an ideal digital edge

Model of a ramp digital edge

Gray-level profile of a horizontal line through the image

Gray-level profile of a horizontal line through the image

FIGURE 10.5
(a) Model of an ideal digital edge.
(b) Model of a ramp edge. The slope of the ramp is proportional to the degree of blurring in the edge.
Thick edge

- The slope of the ramp is inversely proportional to the degree of blurring in the edge.
- We no longer have a thin (one pixel thick) path.
- Instead, an edge point now is any point contained in the ramp, and an edge would then be a set of such points that are connected.
- The thickness is determined by the length of the ramp.
- The length is determined by the slope, which is in turn determined by the degree of blurring.
- **Blurred edges tend to be thick and sharp edges tend to be thin**
the signs of the derivatives would be reversed for an edge that transitions from light to dark
Second derivatives

- Produces 2 values for every edge in an image (an undesirable feature)
- An imaginary straight line joining the extreme positive and negative values of the second derivative would cross zero near the midpoint of the edge. (Zero-crossing property) quite useful for locating the centers of thick edges
Noise images

- Unfortunately, derivatives (differences) are very sensitive to noise.
- Fairly little noise can have such a significant impact on the two key derivatives used for edge detection in images.
- Image smoothing should be serious consideration prior to the use of derivatives in applications where noise is likely to be present.
Edge detection steps

- Three fundamental steps in edge detection:
  - Image smoothing for noise reduction: The need for this step is cleared in the previous slide.
  - Detection of edge points: Local operation that extracts from an image all points that are potential candidates to become edge point.
  - Edge localization: Select from candidate edge points only points that are true members of the set of points comprising an edge.
Basic Edge Detection

- We define a point in an image as being an edge point if its two-dimensional first-order derivative is greater than a specified threshold.
- A set of such points that are connected according to a predefined criterion of connectedness is defined as an edge.
- The term edge segment generally is used if the edge is short in relation to the dimensions of the image.
- A key problem in segmentation is to assemble edge segments into longer edges.
Gradient operator

First-order derivatives of a digital image are based on various approximations of the 2-D gradient. The gradient is defined as the two-dimensional column vector

\[ \nabla f \equiv \text{grad}(f) = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \]

The vector points in the direction of the greatest rate of change of at location \((x,y)\).

The magnitude of the gradient vector often is the value of the rate of change in the direction of the gradient

\[ M(x, y) = \text{mag}(\nabla f) = \sqrt{g_x^2 + g_y^2} \approx |g_x| + |g_y| \]

\(M, g_x, g_y\) are images of the same size as the original
Gradient operator

- The direction of the gradient vector
  \[ \alpha(x, y) = \tan^{-1}\left( \frac{g_y}{g_x} \right) \]
- The direction of an edge at \((x, y)\) is perpendicular to the direction of the gradient vector at the point.
- The gradient vector is called edge normal.
Gradient operator

- Obtaining the gradient of an image requires computing the partial derivatives at every pixel.
  \[ g_x = \frac{\partial f(x, y)}{\partial x} = f(x+1, y) - f(x, y) \]

- and
  \[ g_y = \frac{\partial f(x, y)}{\partial y} = f(x, y+1) - f(x, y) \]

- One dimensional mask to implement gradient in y-and x-directions
Gradient operator - Roberts

- The Roberts cross-gradient operators are based on implementing the diagonal differences

\[ g_x = \frac{\partial f}{\partial x} = (z_9 - z_5) \]

- and

\[ g_y = \frac{\partial f}{\partial y} = (z_8 - z_6) \]

| \( z_1 \) | \( z_2 \) | \( z_3 \) |
| \( z_4 \) | \( z_5 \) | \( z_6 \) |
| \( z_7 \) | \( z_8 \) | \( z_9 \) |

| -1 | 0 | 0 | -1 |
| 0 | 1 | 1 | 0 |

Roberts
The simplest digital approximation to partial derivatives using mask of size $3\times3$ are given by

$$g_x = \frac{\partial f}{\partial x} = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)$$

and

$$g_y = \frac{\partial f}{\partial y} = (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)$$

- Difference between third and first row -> $x$ direction
- Difference between third and first columns -> $y$ direction
Gradient operator - Sobel

- A slight variation of the preceding two equations uses a weight of 2 in the centre coefficient

$$g_x = \frac{\partial f}{\partial x} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

- and

$$g_y = \frac{\partial f}{\partial y} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

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

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

Sobel
Diagonal Edge Masks

Prewitt

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

Sobel

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

FIGURE 10.10
(a) Original image. (b) $|G_x|$, component of the gradient in the x-direction.
(c) $|G_y|$, component in the y-direction.
(d) Gradient image, $|G_x| + |G_y|$. 
Example

**FIGURE 10.11**
Same sequence as in Fig. 10.10, but with the original image smoothed with a $5 \times 5$ averaging filter.
Example
The Laplacian generally is not used in its original form for edge detection for several reasons:

- It is unacceptably sensitive to noise
- The magnitude of the Laplacian produces double edges
- It is unable to detect edge direction

The role of the Laplacian in segmentation:

- Finding the location of edge using its zero crossing property
- Judging whether a pixel is on the dark or light side of an edge
The Laplacian operator is a linear operator defined as:

\[
\nabla^2 f = \frac{\partial^2 f(x, y)}{\partial x^2} + \frac{\partial^2 f(x, y)}{\partial y^2}
\]

or equivalently,

\[
\nabla^2 f = [f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)]
\]

**Figure 10.13**
Laplacian masks used to implement Eqs. (10.1-14) and (10.1-15), respectively.

<table>
<thead>
<tr>
<th></th>
<th>-1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-1</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>4</td>
<td>-1</td>
</tr>
<tr>
<td>-1</td>
<td>8</td>
<td>-1</td>
</tr>
<tr>
<td>0</td>
<td>-1</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>
Laplacian of Gaussian (LoG)

- Laplacian combined with smoothing as a precursor to find edges via zero-crossing.

\[ h(r) = -e^{-\frac{r^2}{2\sigma^2}} \]

where \( r^2 = x^2 + y^2 \), and \( \sigma \) is the standard deviation.

\[ \nabla^2 h(r) = -\left[ \frac{r^2 - \sigma^2}{\sigma^4} \right] e^{-\frac{r^2}{2\sigma^2}} \]
Laplacian of Gaussian (LoG)

**FIGURE 10.14**
Laplacian of a Gaussian (LoG).
(a) 3-D plot.
(b) Image (black is negative, gray is the zero plane, and white is positive).
(c) Cross section showing zero crossings.
(d) $5 \times 5$ mask approximation to the shape of (a).

The coefficient must be sum to zero.
Laplacian of Gaussian (LoG)

- Second derivation is a linear operation
- Thus, $\nabla^2 f$ is the same as convolving the image with gaussian smoothing function first and then computing the laplacian of the result
Laplacian of Gaussian (LoG)

- The purpose of the Gaussian function in the LoG is to smooth the image to reduce the effect of noise.
- The purpose of the Laplacian operator is to provide an image with zero crossing used to establish the location of edges.
Example

a). Original image
b). Sobel Gradient
c). Spatial Gaussian smoothing function
d). Laplacian mask
e). LoG
f). Threshold LoG
g). Zero crossing
Zero crossing & LoG

- Approximate the zero crossing from LoG image
- To threshold the LoG image by setting all its positive values to white and all negative values to black.
- The zero crossing occur between positive and negative values of the thresholded LoG.
Zero crossing vs. Gradient

- Attractive
  - Zero crossing produces thinner edges
  - Noise reduction

- Drawbacks
  - Zero crossing creates closed loops. Sophisticated computation.

- Gradient is more frequently used.
Edge Linking and Boundary Detection

- edge detection algorithm are followed by linking procedures to assemble edge pixels into meaningful edges.

- Basic approaches
  - Local Processing
  - Global Processing via the Hough Transform
  - Global Processing via Graph-Theoretic Techniques
Local Processing

- Analyze the characteristics of pixels in a small neighborhood say, (3x3), or (5x5) about every edge pixels (x, y) in an image.

- All points that are similar according to a set of predefined criteria are linked, forming an edge of pixels that share those criteria.
Local Processing

1. The strength of the response of the gradient operator used to produce the edge pixel
   + An edge pixel with coordinates \((x_0, y_0)\) in a predefined neighborhood of \((x, y)\) is similar in magnitude to the pixel at \((x, y)\) if
     \[
     |\nabla F(x, y) - \nabla f(x_0, y_0)| \leq e
     \]

2. The direction of the gradient vector
   + An edge pixel with coordinates \((x_0, y_0)\) in a predefined neighborhood of \((x, y)\) is similar in angle to the pixel at \((x, y)\) if
     \[
     |\alpha(x, y) - \alpha(x_0, y_0)| < A
     \]
Local Processing

- A point in the predefined neighborhood of \((x,y)\) is linked to the pixel at \((x,y)\) if **both magnitude and direction criteria are satisfied**.
- The process is repeated at every location in the image.
- A record must be kept. Simply by assigning a different gray level to each set of linked edge pixels.
Example

- Use horizontal and vertical sobel operators.
- Link conditions:
  - Gradient value > 25
  - Gradient direction differs < 15°
- Eliminate isolated short segments

Search for license plate
Global Processing via the Hough Transform

- We attempt to link edge pixels that lie on specified curves

- **Brute force method:**
  - When the specified curve is a straight line, the line between each pair of edge pixels in the image is considered. The distance between every other edge pixel and the line in question is then calculated.
  - When the distance is less than a specified threshold, the pixel is considered to be part of the line
Brute force method

Number of calculations for \( n \) edge pixels:

Number of possible lines: \( \sum_{k=1}^{n-1} k = \frac{n(n-1)}{2} \approx n^2 \)

Distances per line: \( n \)

Total number of distances: \( \frac{n^2(n-1)}{2} \approx n^3 \)

When \( n = 256^2 \) then number of calculations is \( 1.4 \times 10^{14} \) !!!
When different values for $a$ and $b$ are considered, then $y_i = ax_i + b$ gives all possible lines through the point $(x_i, y_i)$.

The equation $b = -x_i a + y_i$ gives one line in the $ab$-plane for a specific $(x_i, y_i)$.
When another point \((x_j, y_j)\) is considered, \(b = -x_ja + y_j\) represents another line in the \(ab\)-plane. Suppose that these two lines intersect at the point \((a', b')\), then \(y = a'x + b'\) represents the line in the \(xy\)-plane on which both \((x_i, y_i)\) and \((x_j, y_j)\) lie.
Accumulator cells

- Since a computer can only deal with a finite number of straight lines, we subdivide the parameter space $ab$ into a finite number of accumulator cells...
Accumulator cells

- $(a_{\text{max}}, a_{\text{min}})$ and $(b_{\text{max}}, b_{\text{min}})$ are the expected ranges of slope and intercept values.
- All are initialized to zero.
- If a choice of $a_p$ results in solution $b_q$ then we let $A(p,q) = A(p,q)+1$.
- At the end of the procedure, value $Q$ in $A(i,j)$ corresponds to $Q$ points in the $xy$-plane lying on the line $y = a_i x + b_j$.

$$b = -ax_i + y_i$$
Algorithm

- (1) Set all cells equal to zero
- (2) For every \((x_k, y_k)\)
  - (2.1) Let \(a = \) every subdivision on the \(a\)-axis
  - (2.2) Calculate \(b = -x_k a + y_k\)
  - (2.3) Round off \(b\) to the nearest allotted value on the \(b\)-axis
  - (2.4) Increment accumulator cell \((a, b)\) with 1
When there are $K$ subdivisions on the $a$-axis, we need only $nK$ calculations, which is linear (recall that we needed $n^3$ calculations for the brute force method).

We still have a problem though, since $-\infty < a < \infty$ and $-\infty < b < \infty$!

In order to deal with this problem we use the $\rho$-$\theta$ plane
The ρ-θ plane

- we now represent a straight line as follows:
  \[ x \cos \theta + y \sin \theta = \rho \]
  so that \((a, b) \rightarrow (\rho, \theta)\)

- We have \(\rho \in [-\sqrt{2D}, \sqrt{2D}]\) and \(\theta \in [-90^\circ, 90^\circ]\), where \(\sqrt{2D}\) is the diagonal distance between two opposite corners in the image.
P collinear points lying on a line \( x \cos \theta_j + y \sin \theta_j = \rho_i \) yield \( P \) sinusoidal curves that intersect at \((\rho_i, \theta_j)\).
FIGURE 10.20
Illustration of the Hough transform.
(Courtesy of Mr. D. R. Cate, Texas Instruments, Inc.)
Accumulator cells Algorithm

- (1) Set all cells equal to zero
- (2) For every \((x_k, y_k)\)
  + (2.1) Let \(\theta = \) every subdivision on the \(\theta\)-axis
  + (2.2) Calculate \(\rho = x_k \cos \theta + y_k \sin \theta\)
  + (2.3) Round off \(\rho\) to the nearest allotted value on the \(\rho\)-axis
  + (2.4) Increment accumulator cell \((\rho, \theta)\) with 1
Edge-linking based on Hough Transformation

- Algorithm for edge linking
  
  + (1) Compute $|\nabla f|$ and isolate edge pixels through thresholding
  
  + (2) Specify subdivisions in the $\rho \theta$-plane
  
  + (3) Apply Hough transform to edge pixels
  
  + (4) Identify accumulator cells with highest values
  
  + (5) Examine continuity of pixels that constitute cell
  
  + (6) Link these pixels if gaps are smaller than threshold
Continuity

- based on computing the distance between disconnected pixels identified during traversal of the set of pixels corresponding to a given accumulator cell.
- a gap at any point is significant if the distance between that point and its closest neighbor exceeds a certain threshold.
Example

FIGURE 10.21
(a) Infrared image.
(b) Thresholded gradient image.
(c) Hough transform.
(d) Linked pixels
(Courtesy of Mr. D. R. Cate, Texas Instruments, Inc.)

link criteria: 1) the pixels belonged to one of the set of pixels linked according to the highest count
2) no gaps were longer than 5 pixels
Thresholding

- Because of its intuitive properties and simplicity of implementation, image thresholding enjoys a central position in applications of image segmentation.
Thresholding

**FIGURE 10.26** (a) Gray-level histograms that can be partitioned by (a) a single threshold, and (b) multiple thresholds.
Multilevel thresholding

- A point \((x,y)\) belongs to
  + To an object class if \(T_1 < f(x,y) \leq T_2\)
  + To another object class if \(f(x,y) > T_2\)
  + To background if \(f(x,y) \leq T_1\)

- \(T\) depends on
  + Only \(f(x,y)\): only on gray-level values \(\Rightarrow\) **global threshold**
  + Both \(f(x,y)\) and \(p(x,y)\): on gray-level values and its neighbors \(\Rightarrow\) **local threshold**
  + On \(f, p,\) and coordinates \(x, y\): \(\Rightarrow\) **dynamic or adaptive**
Basic Global Thresholding

- Based on visual inspection of histogram to select an initial estimate for $T$:
  1. Heuristic approach based on visual inspection of histogram
  2. Algorithm used to obtain $T$ automatically

Visual inspection of histogram
Algorithm used to obtain $T$ automatically

(1) Select an initial estimate for $T$
(2) Segment image using $T \rightarrow$ Group $G_1$ (values $> T$)  
Group $G_2$ (values $< T$)
(3) Compute average gray level values for $G_1, G_2$  
$\rightarrow \mu_1, \mu_2$
(4) Compute a new threshold value $T = \frac{1}{2}(\mu_1 + \mu_2)$
(5) Repeat (2) through (4) until the difference in $T$  
in successive iterations is smaller than $T_0$
Example: Automated Method

Note: the clear valley of the histogram and the effective of the segmentation between object and background

\[ T_0 = 0 \]
3 iterations with result \( T = 125 \)
Basic Adaptive Thresholding

- Subdivide original image into small areas.
- Utilize a different threshold to segment each of the subimages.
- Since the threshold used for each pixel depends on the location of the pixel in terms of the subimages, this type of thresholding is adaptive.
Example: Adaptive Thresholding

- $T$ is obtained automatically with $T_0$ midway between the minimum and maximum gray level.
Further subdivision

a) Properly and improperly segmented subimages from previous example.
b)–c) Corresponding histograms.
d) Further subdivision of the improperly segmented subimage.
e) Histogram of small subimage at top.
f) Result of adaptively segmenting d).
Optimal Global and Adaptive Thresholding

- Try to estimate threshold value that produces minimum average segmentation error.
- Assume two principal gray level regions...

**FIGURE 10.32**
Gray-level probability density functions of two regions in an image.
Optimal Global and Adaptive Thresholding

- Mixture PDF describing overall gray level variation...
- \( p(z) = P_1 p_1(z) + P_2 p_2(z)\)
- \( P_1 \): probability that pixel is object pixel
- \( P_2 \): probability that pixel is background pixel
- \( P_1 + P_2 = 1\)
- Select \( T \) that minimizes average error in making decision
Optimal Global and Adaptive Thresholding

- Probability in erroneously classifying background as object:

\[ E_1(T) = \int_{-\infty}^{T} p_2(z) \, dz \]

- Probability in erroneously classifying object as background

\[ E_2(T) = \int_{T}^{\infty} p_1(z) \, dz \]
Optimal Global and Adaptive Thresholding

- Overall probability of error is:
  \[ E(T) = P_2 \, E_1(T) + P_1 \, E_2(T) \]

- Threshold value for which the error is minimal
  \[ P_1 \, p_1(T) = P_2 \, p_2(T) \quad \cdots \quad (**) \]

- Approximate \( p_1(z) \) and \( p_2(z) \) with Gaussian densities...
  \[ p(z) = \frac{P_1}{\sqrt{2\pi}} e^{-\frac{(z-\mu_1)^2}{2\sigma_1^2}} + \frac{P_2}{\sqrt{2\pi}} e^{-\frac{(z-\mu_2)^2}{2\sigma_2^2}} \]
Optimal Global and Adaptive Thresholding

- Using this equation in (**) results in: \( AT^2 + BT + C = 0 \), where:

  \[
  A = \sigma_1^2 - \sigma_2^2 \\
  B = 2 (\mu_1 \sigma_2^2 - \mu_2 \sigma_1^2) \\
  C = \sigma_1^2 \mu_2^2 - \sigma_2^2 \mu_1^2 + 2 \sigma_1^2 \sigma_2^2 \ln(\frac{\sigma_2 P_1}{\sigma_1 P_2})
  \]

- Note that two threshold values are generally required.
- Also note that if \( \sigma^2 = \sigma_1^2 = \sigma_2^2 \), one threshold is sufficient:

  \[
  T = \frac{\mu_1 + \mu_2}{2} + \frac{\sigma^2}{\mu_1 - \mu_2} \ln\left(\frac{P_2}{P_1}\right)
  \]

- When \( P_1 = P_2 \) and/or \( \sigma = 0 \) the optimal threshold is the average of the means.
Example

- Use of optimal thresholding for image segmentation

**FIGURE 10.33** A cardioangiogram before and after preprocessing. (Chow and Kaneko.)
(1) Preprocessing
   (1.1) $s = c \log(1 + r)$
   (1.2) Image subtraction
   (1.3) Image averaging (several images) $\rightarrow$ reduce random noise
(2) Image subdivided into subimages and their histograms calculated
(3) Unimodal histograms rejected and bimodal histograms fitted by bimodal Gaussian density curves
(4) At first only those regions with bimodal histograms are assigned thresholds
(5) Interpolate these thresholds to obtain thresholds for the other (unimodal) regions, so that every pixel has a threshold
(7) Apply thresholds $\rightarrow$ binary image
FIGURE 10.34
Histograms (black dots) of (a) region A, and (b) region B in Fig. 10.33(b). (Chow and Kaneko.)
FIGURE 10.35
Cardioangiogram showing superimposed boundaries. (Chow and Kaneko.)
Region-Based Segmentation

- **Basic Formulation**

\[ \bigcup_{i=1}^{n} R_i = R \]

- \( (a) \) \( \bigcup_{i=1}^{n} R_i = R \)

- \( (b) \) \( R_i \) is a connected region, \( i = 1, 2, \ldots, n \)

- \( (c) \) \( R_i \cap R_j = \emptyset \) for all \( i \) and \( j, i \neq j \)

- \( (d) \) \( P(R_i) = \text{TRUE} \) for \( i = 1, 2, \ldots, n \)

- \( (e) \) \( P(R_i \cup R_j) = \text{FALSE} \) for \( i \neq j \)

\( P(R_i) \) is a logical predicate property defined over the points in set \( R_i \)

ex. \( P(R_i) = \text{TRUE} \) if all pixel in \( R_i \) have the same gray level
Region Growing

- Start from a set of seed points and from these points grow the regions by appending to each seed those neighbouring pixels that have similar properties (seeded region growing).

- The selection of the seed points depends on the problem. When a priori information is not available, (unseeded region growing) clustering techniques can be used.
Region Growing

- Unseeded region starts off with a single region $A_1$.
- At each iteration it considers the neighbouring pixels in the same way as seeded region growing.
- If the difference between a pixel's intensity value and the region's mean, $\delta$ is less than a predefined threshold $T$ then it is added to the respective region $A_j$.
- If not, then the pixel is considered significantly different from all current regions $A_j$ and a new region $A_{n+1}$ is created with this pixel.
Region Growing

- The selection of similarity criteria depends on the problem under consideration and the type of image data that is available, when the image is monochrome, region analysis must be carried out with a set of descriptors based on gray levels and spatial properties such as (moment and texture).
- Descriptors must be used in conjunction with connectivity (adjacency) information.
Region Growing

- Formulation of a “stopping rule”. Growing a region should stop when no more pixels satisfy the criteria for inclusion in that region.
- When a model of the expected results is partially available, the consideration of additional criteria like the size of the region, the likeliness between a candidate pixel and the pixels grown so far, and the shape of the region can improve the performance of the algorithm.
Example

select all seed points with gray level 255

criteria:
1. the absolute gray-level difference between any pixel and the seed has to be less than 65
2. the pixel has to be 8-connected to at least one pixel in that region (if more, the regions are merged)

FIGURE 10.40
(a) Image showing defective welds. (b) Seed points. (c) Result of region growing. (d) Boundaries of segmented defective welds (in black). (Original image courtesy of X-TEK Systems, Ltd.)
Example contd.

- Histogram of original image: used to find the criteria of the difference gray-level between each pixel and the seeds
Region splitting and merging

- Subdivide an image initially into a set of arbitrary, disjoint regions and then merge and/or split the regions in an attempt to satisfy the necessary conditions.

- Let $R$ represent entire image region and select a predicate $P$.
  1. Split into four disjoint quadrants any region $R_i$ for which $P(R_i) = \text{FALSE}$
  2. Merge any adjacent regions $R_j$ and $R_k$ for which $P(R_j \cup R_k) = \text{TRUE}$
  3. Stop when no further merging or splitting is possible.

Several variations of this theme are possible.
Region splitting and merging

- Quadtree

**FIGURE 10.42**
(a) Partitioned image.
(b) Corresponding quadtree.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_1$</td>
<td></td>
<td>$R_2$</td>
</tr>
<tr>
<td>$R_3$</td>
<td>$R_{41}$</td>
<td>$R_{42}$</td>
</tr>
<tr>
<td></td>
<td>$R_{43}$</td>
<td>$R_{44}$</td>
</tr>
</tbody>
</table>

Root $R$
Quad Tree Example
Binary Image
Example

\[ P(R_i) = \text{TRUE if at least } 80\% \text{ of the pixels in } R_i \text{ have the property } |z_j - m_i| \leq 2\sigma_i, \]

where \( z_j \) is the gray level of the jth pixel in \( R_i \), \( m_i \) is the mean gray level of that region, \( \sigma_i \) is the standard deviation of the gray levels in \( R_i \).