Digital Image Processing
Module 5: Part 2

Image Description and Representation

Image Representation and Description?

Objective:
To represent and describe information embedded in an image in other forms that are more suitable than the image itself.

Benefits:
- Easier to understand
- Require fewer memory, faster to be processed
- More “ready to be used”

What kind of information we can use?
- Boundary, shape
- Region
- Texture
- Relation between regions

Shape Representation by Using Chain Codes

Why we focus on a boundary?
The boundary is a good representation of an object shape and also requires a few memory.

Chain codes: represent an object boundary by a connected sequence of straight line segments of specified length and direction.

4-directional chain code

8-directional chain code

Examples of Chain Codes

Object boundary (resampling)

Boundary vertices

4-directional chain code

8-directional chain code

The First Difference of a Chain Codes

Problem of a chain code:
a chain code sequence depends on a starting point.
Solution: treat a chain code as a circular sequence and redefine the starting point so that the resulting sequence of numbers forms an integer of minimum magnitude.

The first difference of a chain code: counting the number of direction change (in counterclockwise) between 2 adjacent elements of the code.

Example:

<table>
<thead>
<tr>
<th>Chain code</th>
<th>The first difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 → 1</td>
<td>1</td>
</tr>
<tr>
<td>0 → 2</td>
<td>2</td>
</tr>
<tr>
<td>0 → 3</td>
<td>3</td>
</tr>
<tr>
<td>2 → 3</td>
<td>1</td>
</tr>
<tr>
<td>2 → 0</td>
<td>2</td>
</tr>
<tr>
<td>2 → 1</td>
<td>3</td>
</tr>
</tbody>
</table>

Example:

- a chain code: 10103322
- The first difference = 3133030
- Treating a chain code as a circular sequence, we get the first difference = 33133030

The first difference is rotational invariant.

Polygon Approximation

Represent an object boundary by a polygon

Object boundary

Minimum perimeter polygon

Minimum perimeter polygon consists of line segments that minimize distances between boundary pixels.
**Polygon Approximation: Splitting Techniques**

1. Find the line joining two extreme points
2. Find the farthest points from the line
3. Draw a polygon

**Distance-Versus-Angle Signatures**

Represent an 2-D object boundary in terms of a 1-D function of radial distance with respect to $\theta$.

**Boundary Segments**

**Concept:** Partitioning an object boundary by using vertices of a convex hull.

**Convex Hull Algorithm**

**Input:** A set of points on a cornea boundary  
**Output:** A set of points on a boundary of a convex hull of a cornea

1. Sort the points by $x$-coordinate to get a sequence $p_1, p_2, \ldots, p_n$
2. Put the points $p_1$ and $p_2$ in a list $L_{\text{upper}}$ with $p_1$ as the first point
3. For $i = 3$ to $n$
4.   Do append $p_i$ to $L_{\text{upper}}$
5.   While $L_{\text{upper}}$ contains more than 2 points and the last 3 points in $L_{\text{upper}}$ do not make a right turn
6.     Do delete the middle point of the last 3 points from $L_{\text{upper}}$
7.     Remove the first and the last points from $L_{\text{upper}}$
8.     Append $L_{\text{lower}}$ to $L_{\text{upper}}$ resulting in the list $L$
9.     Return $L$

**Convex Hull Algorithm (cont.)**

**Skeletons**

**Medial axes (dash lines)**

Obtained from thinning or skeletonizing processes.
**Thinning Algorithm**

**Concept:**
1. Do not remove end points
2. Do not break connectivity
3. Do not cause excessive erosion

**Apply only to** contour pixels: pixels “1” having at least one of its 8 neighbor pixels valued “0”

**Notation:**

Let

\[
\begin{align*}
N(p_i) &= p_2 + p_3 + \ldots + p_7 + p_9 \\
T(p_i) &= \text{the number of transition 0-1 in the ordered sequence } p_2, p_3, \ldots, p_7, p_8, p_9.
\end{align*}
\]

**Example**

\[
\begin{array}{ccc}
0 & 0 & 1 \\
1 & p_1 & 0 \\
1 & 0 & 1
\end{array}
\]

\[N(p_i) = 4, \quad T(p_i) = 3\]

**Step 1.** Mark pixels for deletion if the following conditions are true.

a) \(2 \leq N(p_i) \leq 6\)

b) \(T(p_i) = 1\)

c) \(p_2 \cdot p_4 \cdot p_6 = 0\)

d) \(p_2 \cdot p_4 \cdot p_6 = 0\)

(Apply to all border pixels)

**Step 2.** Delete marked pixels and go to Step 3.

**Step 3.** Mark pixels for deletion if the following conditions are true.

a) \(2 \leq N(p_i) \leq 6\)

(Apply to all border pixels)

b) \(T(p_i) = 1\)

c) \(p_2 \cdot p_4 \cdot p_6 = 0\)

d) \(p_2 \cdot p_4 \cdot p_6 = 0\)

**Step 4.** Delete marked pixels and repeat Step 1 until no change occurs.

**Boundary Descriptors**

1. Simple boundary descriptors:
   - Length of the boundary
   - The size of smallest circle or box that can totally enclosing the object

2. Shape number

3. Fourier descriptor

4. Statistical moments

**Shape Number**

**Shape number of the boundary definition:**
the first difference of smallest magnitude

**The order n of the shape number:**
the number of digits in the sequence

<table>
<thead>
<tr>
<th>Order 4</th>
<th>Order 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chain code: 0 3 2 1</td>
<td>0 0 3 2 2 1</td>
</tr>
<tr>
<td>Difference: 3 3 3 3</td>
<td>3 0 3 3 0 3</td>
</tr>
<tr>
<td>Shape no.: 3 3 3 3</td>
<td>0 3 3 0 3 3</td>
</tr>
</tbody>
</table>

**Shape Number (cont.)**

**Shape numbers of order 4, 6 and 8**

<table>
<thead>
<tr>
<th>Order 4</th>
<th>Order 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chain code: 0 3 2 1</td>
<td>0 0 3 2 2 1</td>
</tr>
<tr>
<td>Difference: 3 3 3 3</td>
<td>1 0 3 3 0 3</td>
</tr>
<tr>
<td>Shape no.: 3 3 3 3</td>
<td>0 3 3 0 3 3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Order 5</th>
<th>Order 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chain code: 0 3 3 2 2 1</td>
<td>0 0 3 3 2 2 1</td>
</tr>
<tr>
<td>Difference: 3 0 3 0 3 0</td>
<td>3 3 1 3 3 0 3</td>
</tr>
<tr>
<td>Shape no.: 0 3 3 0 3 3</td>
<td>0 0 3 3 0 3 3</td>
</tr>
</tbody>
</table>
Example: Shape Number

1. Original boundary
2. Find the smallest rectangle that fits the shape
3. Create grid
4. Find the nearest Grid.

Chain code:
0 0 0 3 2 2 2 3 2 2 2 1 2 1 1
First difference:
3 0 0 0 3 1 0 3 3 0 1 3 0 0 3 1 3 0
Shape No.
0 0 0 3 1 0 3 0 1 3 0 0 3 1 3 0 3

Example: Fourier Descriptor

Fourier descriptor: view a coordinate (x, y) as a complex number (x = real part and y = imaginary part) then apply the Fourier transform to a sequence of boundary points.

Let s(k) be a coordinate of a boundary point k:

\[ s(k) = x(k) + jy(k) \]

Fourier descriptor:

\[ a(u) = \frac{1}{K} \sum_{k=0}^{K-1} e^{-j2\pi ku/K} \]

Fourier Descriptor Properties

Some properties of Fourier descriptors:

<table>
<thead>
<tr>
<th>Transformation</th>
<th>Boundary</th>
<th>Fourier Descriptor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identity</td>
<td>s(k) =</td>
<td>a(u)</td>
</tr>
<tr>
<td>Rotation</td>
<td>s(k) = s(k) e^{j \phi}</td>
<td>a(u) = a(u) e^{j \phi}</td>
</tr>
<tr>
<td>Translation</td>
<td>s(k) = s(k) + \Delta x</td>
<td>a(u) = a(u) + \Delta x a'(u)</td>
</tr>
<tr>
<td>Scaling</td>
<td>s(k) = s(k)</td>
<td>a(u) = \alpha a'(u)</td>
</tr>
<tr>
<td>Starting point</td>
<td>s(k) = s(k)</td>
<td>a(u) = a(u) e^{j \phi}</td>
</tr>
</tbody>
</table>

Statistical Moments

Definition: the \(n\)th moment

\[ \mu_n = \sum_{i=1}^{L} (r_i - m)^n g(r_i) \]

where

\[ m = \frac{\sum_{i=1}^{L} r_i g(r_i)}{\sum_{i=1}^{L} g(r_i)} \]

Example of moment:
The first moment = mean
The second moment = variance

Boundary segment

1. Convert a boundary segment into 1D graph
2. View a 1D graph as a PDF function
3. Compute the \(n\)th order moment of the graph

Fourier descriptor:

\[ s(k) = x(k) + jy(k) \]

Boundary points

Reconstruction formula

\[ s(k) = \frac{1}{K} \sum_{u=0}^{K-1} a(u) e^{j2\pi uk/K} \]

P is the number of Fourier coefficients used to reconstruct the boundary

Original (K = 44)

\( P = 2 \)
\( P = 4 \)
\( P = 8 \)
\( P = 16 \)
\( P = 32 \)
\( P = 64 \)

1D graph