T-61.5070 COMPUTER VISION, Exercise 7/08

1.

Segmentation methods based on thresholds detected from image histograms are described in Sec. 5.1, pp. 127–131.

a) The given histogram is depicted in Fig. 1.a. The histogram has many local minima[1] and therefore it is smoothed by averaging over five-element neighborhoods, Fig. 1.b (see Sec. 2.3.2. in the book). To obtain the five-element average at the histogram edge, the edge bin is copied. The

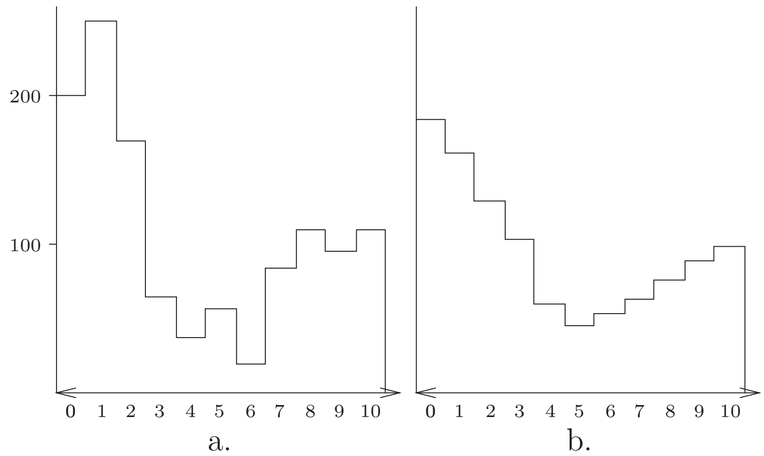

Figure 1: The given histogram (a.) and the smoothed histogram (b.)

thresholds are now detected by finding all minima in the smoothed histogram. The smoothed histogram has only one minimum at graylevel value 5.

b) If some regions in the image can be regarded as potential background or object pixels, then Algorithm 5.2 in the textbook (p. 129), iterative threshold selection, may be used. The step 1. is replaced with computation of μ_B^0 and μ_O^0 according to the background and object pixels. Set $T^1 = (\mu_B^0 + \mu_O^0)/2$ and continue to step 2.

2.

Border tracing is described in Sec. 5.2.3, pp. 142–148. Border tracing methods are used to detect borders of regions obtained by thresholding. The image is segmented by thresholding to background and object regions:

$$\begin{cases} g(x,y) = 1, & \text{when } f(x,y) \geq 4 \\ g(x,y) = 0, & \text{when } f(x,y) < 4 \end{cases} \qquad (1)$$

$f(x,y)$

1	2	0	1	0
1	5	2	5	1
0	6	7	5	1
1	2	7	5	1
1	7	2	7	0
0	1	1	0	1

$\rightarrow$

$g(x,y)$

0	0	0	0	0
0	1	0	1	0
0	1	1	1	0
0	0	1	1	0
0	1	0	1	0
0	0	0	0	0

[1] 'Noise' in the definition of the problem does not refer to image noise but to unevennesses in the histogram. Addition of noise to an image smoothes peaks in its histogram.

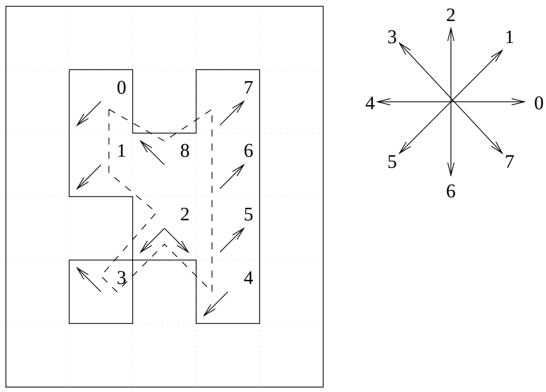

Figure 2: Finding the boundary in a binary image using inner boundary tracing. Direction notation in 8-connectivity is depicted on the right.

The textbook introduces four algorithms for border tracing:

Algorithm 5.8, inner boundary tracing (p. 142),

Algorithm 5.9, outer boundary tracing (p. 143) which is suitable for detection of shape, shape of perimeter or compact regions, for instance.

Algorithm 5.10, extended boundary tracing (p. 146) which defines a single common border between adjacent regions.

If object regions are not defined then Algorithm 5.11, border tracing in gray level images (p. 147), may be used. The border is represented by a simple path of high-gradient pixels.

Fig. 2 shows the path obtained by the inner boundary tracing algorithm in 8-connectivity. The arrows depict the direction at the beginning of each neighborhood search. The dashed lines depict the detected inner borders. The procedure is described in detail in the following.

1. Starting pixel $P_0 = 0$ is found in the upper left corner of the region. Assign $dir = 7$.
2. Search the 3×3 neighborhood in an anti-clockwise direction beginning in direction $(7 + 6) \bmod 8 = 5$. $P_1 = 1$, $dir = 6$.
3. Search begins in direction $(6 + 7) \bmod 8 = 5$. $P_2 = 2$, $dir = 7$.
4. Search begins in direction $(7 + 6) \bmod 8 = 5$. $P_3 = 3$, $dir = 5$.
5. Search begins in direction $(5 + 6) \bmod 8 = 3$. $P_4 = 2$, $dir = 1$.
6. Search begins in direction $(1 + 6) \bmod 8 = 7$. $P_5 = 4$, $dir = 7$.
7. Search begins in direction $(7 + 6) \bmod 8 = 5$. $P_6 = 5$, $dir = 2$.
8. Search begins in direction $(2 + 7) \bmod 8 = 1$. $P_7 = 6$, $dir = 2$.
9. Search begins in direction $(2 + 7) \bmod 8 = 1$. $P_8 = 7$, $dir = 2$.
10. Search begins in direction $(2 + 7) \bmod 8 = 1$. $P_9 = 8$, $dir = 5$.
11. Search begins in direction $(5 + 6) \bmod 8 = 3$. $P_{10} = 0$, $dir = 3$.
12. Search begins in direction $(3 + 6) \bmod 8 = 1$. $P_{11} = 1$, $dir = 6$.
13. The current boundary element $P_n = P_{11} = P_1$ and $P_{n-1} = P_{10} = P_0$. Stop.

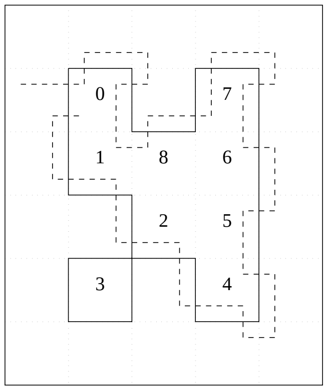

Figure 3: Finding the boundary in a binary image using Papert's turtle.

The inner boundary is represented by pixels

$$P_0, \cdots, P_{n-2} = P_0, \cdots, P_9 = 0, 1, 2, 3, 2, 4, 5, 6, 7, 8. \tag{2}$$

3.

Edge following is described in Sec. 5.2.4, pp. 148-163. The textbook introduces three algorithms for edge following:

> Algorithm 5.12: A-algorithm graph search (p. 149) which yields the optimal border connecting two image points, i.e., the starting and end points must be known. The optimal border minimizes some preselected cost function. The method is suitable for the detection of approximately straight contours.
>
> Algorithm 5.13: Heuristic search for image borders (p. 157). Local edges are merged into chains on the basis of their magnitudes and directions. The use of the method does not require any knowledge about the border locations.
>
> Algorithm 5.14: Boundary tracing as dynamic programming (p. 161) is suitable if the starting and end points of an edge are not known and, especially, if the cost functions are simple. The method is more efficient for some problems than the A-algorithm graph search (see the book, p. 161).

Papert's turtle searches for the boundaries of distinct image regions. The procedure is as follows:

1. Scan the image until a region pixel is encountered.
2. At a region pixel, turn left and step; else, turn right and step.
3. Terminate upon return to the starting pixel.

Regions with 4-connectivity are required for a consistent boundary detection; in an 8-connected region some parts may be missed. Some bookkeeping is necessary to generate an exact sequence of boundary pixels without duplications. Papert's turtle is a border tracing method rather than an edge following method; it requires no other knowledge than that about some object region. Fig. 3 shows the path traced out by the procedure. The detected border pixels with duplicates removed are

$$P_0, \ldots, P_8 = 0, 1, 8, 7, 6, 5, 4, 2, 1. \tag{3}$$

The procedure missed one 8-connected border pixel in the lower left corner.