Chapter 2 Binary Machine Vision: Thresholding and Segmentation

2.1 Introduction

binary value 1: considered part of object
binary value 0: background pixel
binary machine vision: generation and analysis of binary image

2.2 Thresholding

$B(r,c) = 1$ if $I(r,c) \geq T$
$B(r,c) = 0$ if $I(r,c) < T$
r: row, c: column
I: grayscale intensity, B: binary intensity
T: intensity threshold
original gray scale image: no numbers means value 0
=====Fig. 2.1=====
thresholded gray scale image: threshold 0
=====Fig. 2.2=====
=====lena.bin.128=====

histogram $h(m) = \#\{(r,c)|I(r,c) = m\}$

m spans each gray level value e.g. $0 - 255$
#: operator counts the number of elements in a set
image of a metal part
=====Fig. 2.3=====
histogram of the image of Fig. 2.3: two dominant modes
=====Fig. 2.4=====
metal-part image thresholded at gray level 148
=====Fig. 2.5=====
=====lena.histogram=====
BNC T-connector against a dark background
=====Fig. 2.6=====
histogram of the BNC T-connector image
=====Fig. 2.7=====
image of the BNC T-connector thresholded at: 110, 130, 150, 170
=====Fig. 2.8=====

2.2.1 Minimizing Within-Group Variance

$P(1), ..., P(I)$: histogram probabilities of gray values $1, ..., I$

$$P(i) = \frac{\#\{(r,c)|Intensity(r,c) = i\}}{R \times C}$$

$R \times C$: the spatial domain of the image within-group variance σ_W^2: weighted sum of group variances

$$\sigma_W^2(t) = q_1(t)\sigma_1^2(t) + q_2(t)\sigma_2^2(t)$$

$\sigma_1^2(t)$: variance for the group with values $\leq t$
$\sigma_2^2(t)$: variance for the group with values $> t$
$q_1(t)$: probability for the group with values $\leq t$
$q_2(t)$: probability for the group with values $> t$

$$q_1(t) = \sum_{i=1}^{t} P(i)$$

$$q_2(t) = \sum_{i=t+1}^{I} P(i)$$

$$\mu_1(t) = \sum_{i=1}^{t} iP(i)/q_1(t)$$

$$\mu_2(t) = \sum_{i=t+1}^{I} iP(i)/q_2(t)$$

$$\sigma_1^2(t) = \sum_{i=1}^{t} [i - \mu_1(t)]^2 P(i)/q_1(t)$$

$$\sigma_2^2(t) = \sum_{i=t+1}^{I} [i - \mu_2(t)]^2 P(i)/q_2(t)$$

find t which minimizes $\sigma_W^2(t)$
binary image produced by thresholding T-connector with Otsu threshold

=====Fig. 2.9=====
=====New 3:9=====

2.2.2 Minimizing Kullback Information Distance

minimize the Kullback directed divergence J

$$J = \sum_{i=1}^{I} P(i) \log[\frac{P(i)}{f(i)}]$$

mixture distribution of the two Gaussians in histogram:

$$f(i) = \frac{q_1}{\sqrt{2\pi}\sigma_1} e^{-\frac{1}{2}(\frac{i-\mu_1}{\sigma_1})^2} + \frac{q_2}{\sqrt{2\pi}\sigma_2} e^{-\frac{1}{2}(\frac{i-\mu_2}{\sigma_2})^2}$$

binary image produced by thresholding with Kittler-Illingworth threshold
=====Fig. 2.10=====
T-connector histogram with Otsu (left) and Kittler-Illingworth (right) thr.
=====Fig. 2.11=====
mixture of two Gaussians
=====Fig. 2.12=====

2.3 Connected Components Labeling

Connected components analysis of a binary image consists of the connected components labeling of the binary-1 pixels followed by property measurement of the component regions and decision making.

All pixels that have value binary 1 and are connected to each other by a path of pixels all with value binary 1 are given the same identifying label.

label: unique name or index of the region
label: identifier for a potential object region
connected components labeling: a grouping operation
pixel property: position, gray level or brightness level
region property: shape, bounding box, position, intensity statistics

2.3.1 Connected Components Operators

Two 1-pixels p and q belong to the same connected component C if there is a sequence of 1-pixels $(p_0, p_1, ..., p_n)$ of C where $p_0 = p, p_n = q$, and p_i is a neighbor of p_{i-1} for $i = 1, ..., n$.

4-connected: north, south, east, west
8-connected: north, south, east, west, northeast,
northwest, southeast, southwest
border: subset of 1-pixels also adjacent to 0-pixels
(a) 4-connected (b) 8-connected
=====Fig. 2.13=====
(a) original image (b) connected components
=====Fig. 2.14=====

Rosenfeld has shown that if C is a component of 1s and D is an adjacent component of 0s, and if 4-connectedness is used for 1-pixels and 8-connectedness is used for 0-pixels then either C surrounds D (D is a hole in C) or D surrounds C (C is a hole in D).
true when 8-connectedness for 1-pixels, 4-connectedness for 0-pixels
common to use one type of connectedness for 1-pixels and other for 0-pixels
phenomenon associated with using 4- and 8-adjacency
=====Fig. 2.15=====

2.3.2 Connected Components Algorithms

All the algorithms process a row of the image at a time
All the algorithms assign new labels to the first pixel of each component
attempt to propagate the label of a pixel to right or below neighbors
This process continues until the pixel marked A in row 4 encountered
propagation process
=====Fig. 2.16=====
The differences among the algorithms of three types:

1. What label should be assigned to pixel A?
2. How to keep track of the equivalence of two (or more) labels?
3. How to use the equivalence information to complete processing?

2.3.3 An Iterative Algorithm

1. Initialization of each 1-pixel to a unique label
2. Iteration of top-down followed by bottom-up passes until no change

iterative algorithm for connected components labeling

=====Fig. 2.17=====

=====Oldie 34:13=====

2.3.4 The Classical Algorithm

makes two passes but requires a large global table for equivalences

1. performs label propagation as above
2. when two different labels propagate to the same pixel,
   the smaller label propagates and equivalence entered into table
3. equivalence classes are found by transitive closure
4. second pass performs a translation

classical connected components labeling algorithm

=====Fig. 2.18 (a)=====Fig. 2.18(b)=====

main problem: global equivalence table may be too large for memory

2.3.5 A Space-Efficient Two-Pass Algorithm That Uses a Local Equivalence Table

small table stores only equivalences from current and preceding lines

maximum number of equivalences = image width
relabel each line with equivalence labels when equivalence detected
results after the top-down pass of local table method
=====Fig. 2.19=====

2.3.6 An Efficient Run-Length Implementation of the Local Table Method

run-length encoding: transmits lengths of runs of zeros and ones
$01111000110000 \rightarrow [1, 4, 3, 2, 4]$
(a) binary image (b), (c) run-length encoding
=====Fig. 2.20=====
data structures used for keeping track of equivalence classes
=====Fig. 2.21=====
diagram showing the bottom-up pass (8-connected)
=====Fig. 2.22=====

2.4 Signature Segmentation and Analysis

signature: histogram of the nonzero pixels of the resulting masked image
signature: a projection
projections can be vertical, horizontal, diagonal, circular, radial ...
=====Fig. 2.23: $45°$ diagonal projection=====

vertical projection of a segment: column c between s, t
horizontal projection: row r between u, v
vertical and horizontal projection define a rectangle R:

$$R = \{(r, c) | u \leq r \leq v \text{ and } s \leq c \leq t\}$$

horizontal and vertical projection
=====Fig. 2.25=====
binary image segmented into regions based on initial projections
=====Fig. 2.27=====
binary image further segmented
=====Fig. 2.28=====
1. segment the vertical and horizontal projections
2. treat each rectangular subimage as the image
=====check=====

Diagonal projections:
P_D: from upper left to lower right
P_E: from upper right to lower left
diagonal projections P_D and P_E
=====Fig. 2.29=====
object area: sum of all the projection values in the segment

2.5 Summary

when components spaced away and relatively few, use signature segmentation

=====joke=====

Project due Oct. 19:

Write a program to generate

a. a binary image (threshold at 128)

b. a histogram

c. connected components (regions with + at centroid, bounding box)