Binary Image Analysis

1

Outline

- Introduction to binary image analysis
- Thresholding
- Mathematical morphology
- Pixels and neighborhoods
- Connected components analysis

2

2

Binary image analysis

- Binary image analysis consists of a set of operations that are used to produce or process binary images, usually images of 0's and 1's where
  - 0 represents the background,
  - 1 represents the foreground.

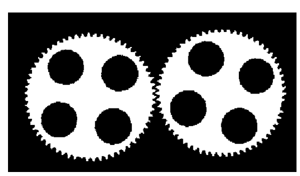
```
00010010001000
00011110001000
00010010001000
```


3

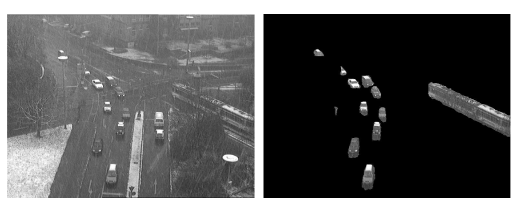
Application areas

- Document analysis

- Industrial inspection



- Surveillance



Adapted from Shapiro and Stockman;
Cheung and Kamath

4

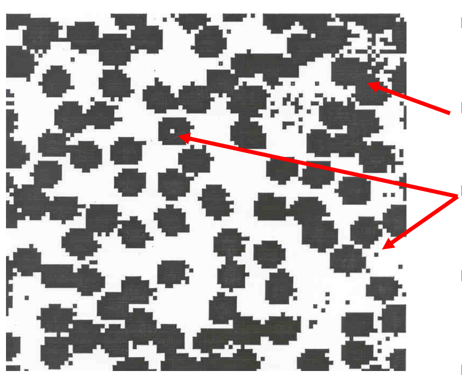
Operations

- Separate objects from background and from one another.
- Aggregate pixels for each object.
- Compute features for each object.

5

5

Example: red blood cell image



- Many blood cells are separate objects.
- Many touch each other → bad!
- Salt and pepper noise is present.
- How useful is this data?
- 63 separate objects are detected.
- Single cells have area of about 50 pixels.

Adapted from Linda Shapiro, U of Washington

6

6

Thresholding

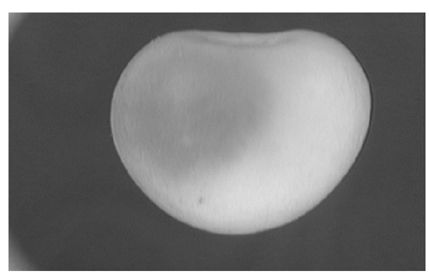
- Binary images can be obtained by thresholding.
- Assumptions for thresholding:
  - Object region-of-interest has intensity distribution different from background.
  - Object pixels likely to be identified by intensity alone:
    - intensity > a
    - intensity < b
    - a < intensity < b
- Works OK with flat-shaded scenes or engineered scenes.
- Does not work well with natural scenes.

7

7

Use of histograms for thresholding

- Background is black.
- Healthy cherry is bright.
- Bruise is medium dark.
- Histogram shows two cherry regions (black background has been removed).



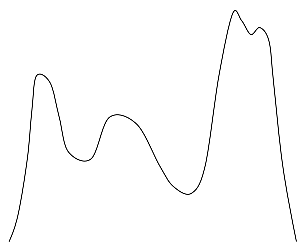
Adapted from Shapiro and Stockman

8

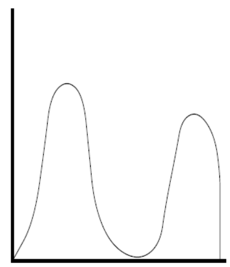
8

Automatic thresholding

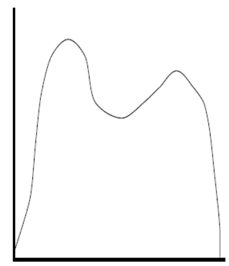
- How can we use a histogram to separate an image into 2 (or several) different regions?



Is there a single clear threshold? 2? 3?



Two distinct modes



Overlapped modes

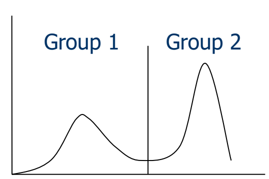
Adapted from Shapiro and Stockman

9

9

Automatic thresholding: Otsu's method

- Assumption: the histogram is bimodal.
- Method: find the threshold t that minimizes the <u>weighted sum of within-group variances</u> for the two groups that result from separating the gray levels at value t.
- The best threshold t can be determined by a simple sequential search through all possible values of t.



- If the gray levels are strongly dependent on the location within the image, local or dynamic thresholds can also be used.

10

10

Otsu's method

Weighted sum of within-group variances: $\sigma_w^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t)$

Probabilities of the two classes (L bins):

$$\omega_0(t) = \sum_{i=0}^{t-1} p(i)$$

$$\omega_1(t) = \sum_{i=t}^{L-1} p(i)$$

Minimizing within-group variances is same as maximizing between-group variances!

$$\sigma_b^2(t) = \sigma^2 - \sigma_w^2(t) = \omega_0(\mu_0 - \mu_T)^2 + \omega_1(\mu_1 - \mu_T)^2$$
$$= \omega_0(t)\omega_1(t)[\mu_0(t) - \mu_1(t)]^2$$

where class means are

$$\mu_0(t) = \frac{\sum_{i=0}^{t-1} ip(i)}{\omega_0(t)}$$

$$\mu_1(t) = \frac{\sum_{i=t}^{L-1} ip(i)}{\omega_1(t)}$$

$$\mu_T = \sum_{i=0}^{L-1} ip(i)$$

$$\omega_0\mu_0 + \omega_1\mu_1 = \mu_T$$
$$\omega_0 + \omega_1 = 1$$

11

11

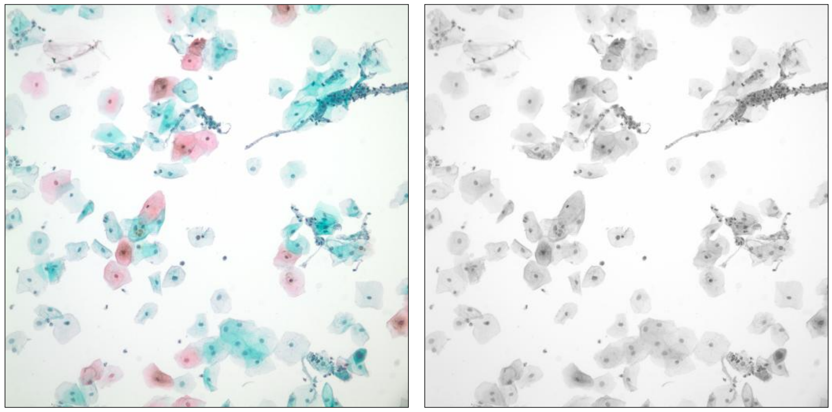
Otsu's algorithm

1. Compute histogram and probabilities of each intensity level
2. Set up initial $\omega_i(0)$ and $\mu_i(0)$
3. Step through all possible thresholds $t = 1, \ldots$ maximum intensity
    1. Update ω_i and μ_i
    2. Compute $\sigma_b^2(t)$
4. Desired threshold corresponds to the maximum $\sigma_b^2(t)$

12

12

Automatic thresholding

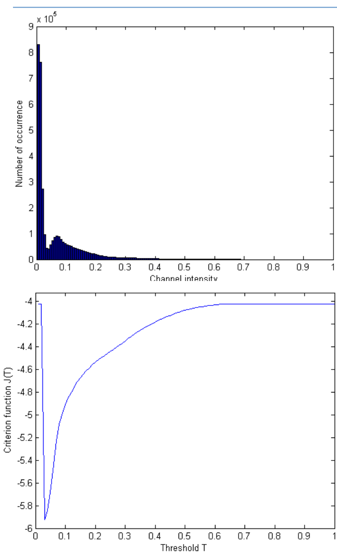


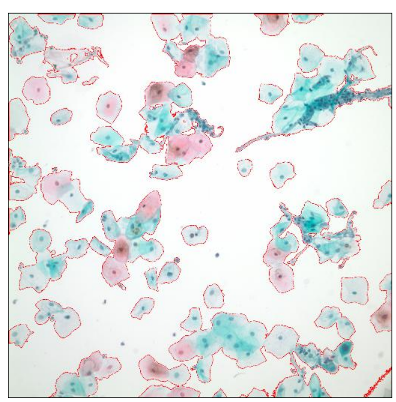
A Pap smear image example: RGB image (left) and grayscale image (right).

13

13

Automatic thresholding





Histogram of the image (top-left), sum of within-group variances versus the threshold (bottom-left), resulting mask overlayed as red on the original image (top).

14

14

Mathematical morphology

- The word morphology refers to form and structure.
- In computer vision, it is used to refer to the shape of a region.
- The language of mathematical morphology is set theory where sets represent objects in an image.
- We will discuss morphological operations on binary images whose components are sets in the 2D integer space Z^2.

15

15

Mathematical morphology

- Mathematical morphology consists of two basic operations
  - dilation
  - erosion

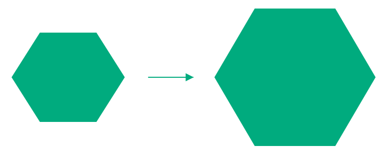
  and several composite relations
  - opening
  - closing
  - conditional dilation
  - …

16

16

Dilation

- Dilation expands the connected sets of 1s of a binary image.
- It can be used for
  - growing features

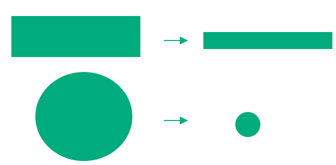


  - filling holes and gaps

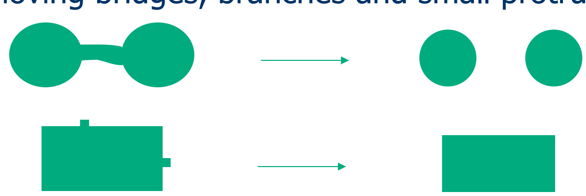
Adapted from Linda Shapiro, U of Washington

Erosion

- Erosion shrinks the connected sets of 1s of a binary image.
- It can be used for
  - shrinking features



  - removing bridges, branches and small protrusions



Adapted from Linda Shapiro, U of Washington

Basic concepts from set theory

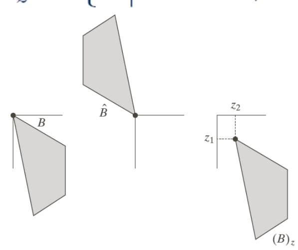
- Let A be a set in Z^2. If $a = (a_1, a_2)$ is an element of A, we write $a \in A$; otherwise, we write $a \notin A$.
- Set A being a *subset* of set B is denoted by $A \subseteq B$.
- The *union* of two sets A and B is denoted by $A \cup B$.
- The *intersection* of two sets A and B is denoted by $A \cap B$.
- The *complement* of a set A is the set of elements not contained in A:
$$A^c = \{w | w \notin A\}.$$
- The *difference* of two sets A and B, denoted by $A - B$, is defined as
$$A - B = \{w | w \in A, w \notin B\} = A \cap B^c.$$

19

19

Basic concepts from set theory

- The *reflection* of set B, denoted by $\check{B}$, is defined as
$$\check{B} = \{w | w = -b, \forall b \in B\}.$$
- The *translation* of set A by point $z = (z_1, z_2)$, denoted by A_z, is defined as
$$A_z = \{w | w = a + z, \forall a \in A\}.$$



a b c

FIGURE 9.1 (a) A set, (b) its reflection, and (c) its translation by z.

Adapted from Gonzales and Woods

20

20

Structuring elements

- Structuring elements are small binary images used as shape masks in basic morphological operations.
- They can be of any shape and size that is digitally representable.
- One pixel of the structuring element is denoted as its origin.
- Origin is often the central pixel of a symmetric structuring element but may in principle be any chosen pixel.

21

21

Structuring elements

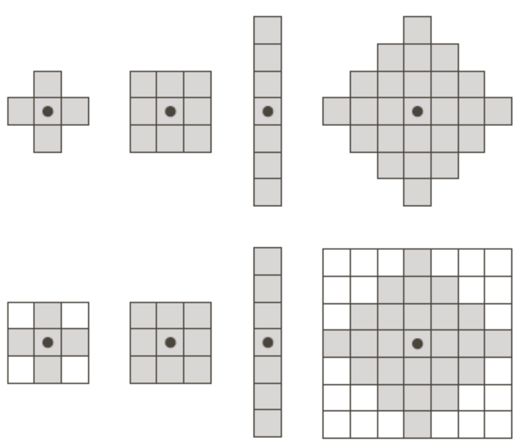


FIGURE 9.2 First row: Examples of structuring elements. Second row: Structuring elements converted to rectangular arrays. The dots denote the centers of the SEs.

Adapted from Gonzales and Woods

22

22

Dilation

- The *dilation* of binary image A by structuring element B is denoted by $A \oplus B$ and is defined by

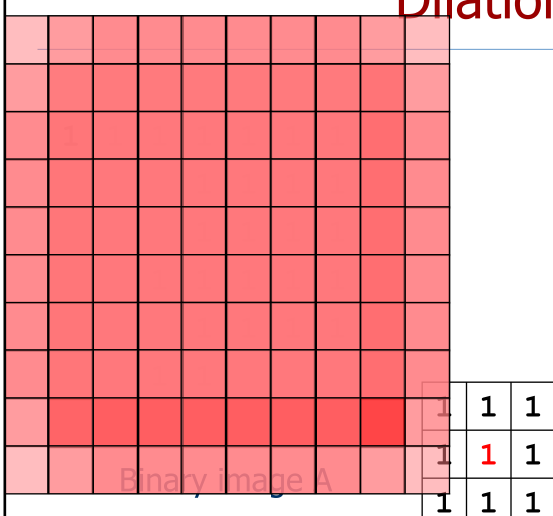
$$A \oplus B = \{z | \check{B}_z \cap A \neq \emptyset\},$$
$$= \bigcup_{a \in A} B_a.$$

  - First definition: The dilation is the set of all displacements z such that $\check{B}_z$ and A overlap by at least one element.
  - Second definition: The structuring element is swept over the image. Each time the origin of the structuring element touches a binary 1-pixel, the entire translated structuring element is ORed to the output image, which was initialized to all zeros.

23

23

Dilation



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

Dilation result

Structuring element B

(1st definition)

24

24

Dilation

1	1	1	1	1	1	1	
			1			1	
			1			1	
		1	1			1	
			1	1	1	1	
		1	1				

Binary image A

1	1	1
1	1	1
1	1	1

Structuring element B

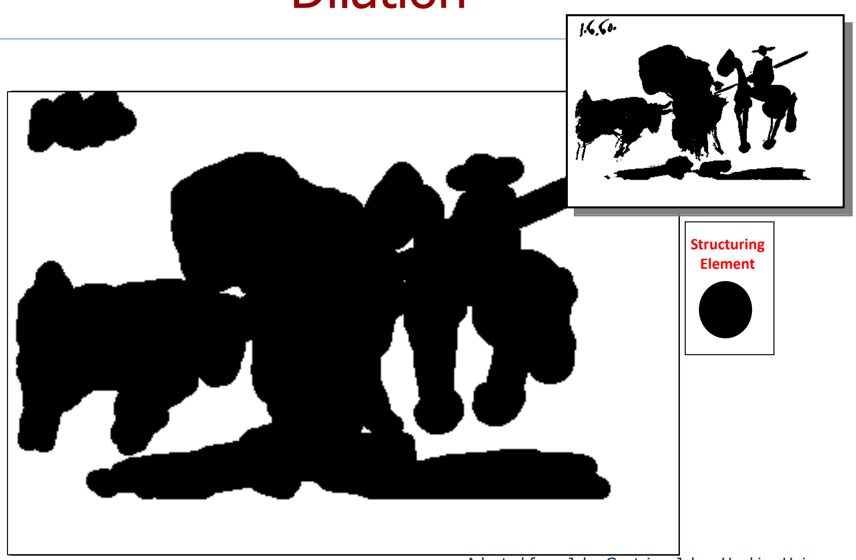
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
	1	1	1	1	1	1	1
	1	1	1	1	1	1	1
	1	1	1	1	1	1	1
	1	1	1	1	1	1	1
	1	1	1	1			

Dilation result

(2nd definition)

25

Dilation

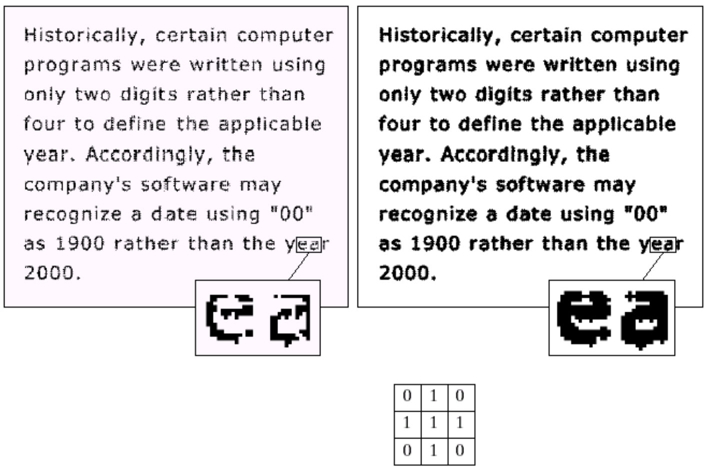


Adapted from John Goutsias, Johns Hopkins Univ.

Pablo Picasso, *Pass with the Cape*, 1960

26

Dilation



a c
b

FIGURE 9.5
(a) Sample text of poor resolution with broken characters (magnified view).
(b) Structuring element.
(c) Dilation of (a) by (b). Broken segments were joined.

Adapted from Gonzales and Woods

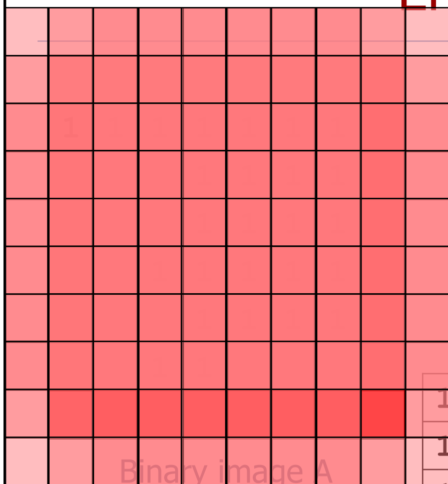
Erosion

- The *erosion* of binary image A by structuring element B is denoted by $A \ominus B$ and is defined by

$$A \ominus B = \{z | B_z \subseteq A\},$$
$$= \{a | a + b \in A, \forall b \in B\}.$$

  - First definition: The erosion is the set of all points z such that B, translated by z, is contained in A.
  - Second definition: The structuring element is swept over the image. At each position where every 1-pixel of the structuring element covers a 1-pixel of the binary image, the binary image pixel corresponding to the origin of the structuring element is ORed to the output image.

Erosion



Binary image A

1	1	1
1	1	1
1	1	1

Structuring element B

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

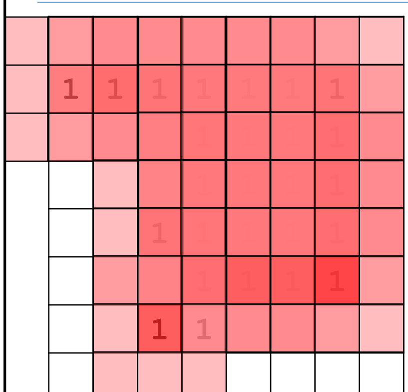
Erosion result

(1st definition)

29

29

Erosion



Binary image A

1	1	1
1	1	1
1	1	1

Structuring element B

				1	1		
				1	1		
				1	1		

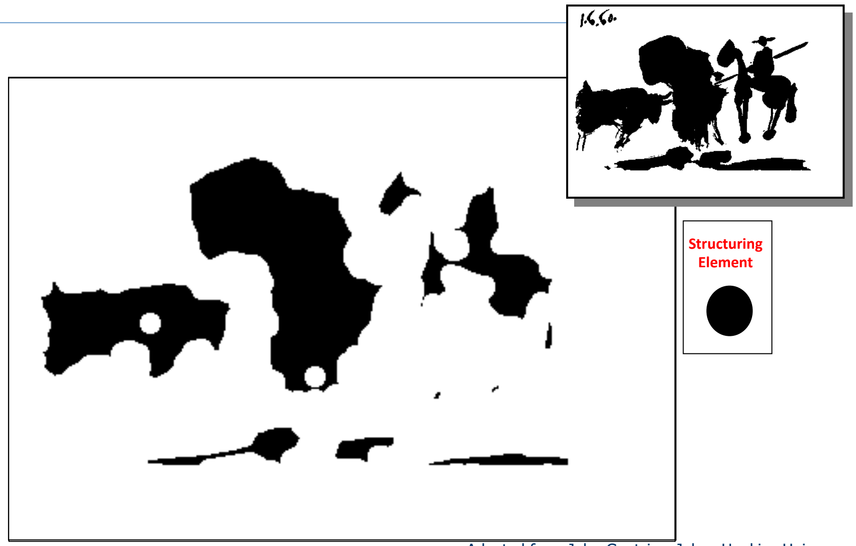
Erosion result

(2nd definition)

30

30

Erosion



Pablo Picasso, *Pass with the Cape*, 1960

Adapted from John Goutsias, Johns Hopkins Univ.

31

31

Opening

- The *opening* of binary image A by structuring element B is denoted by $A \circ B$ and is defined by

$$A \circ B = (A \ominus B) \oplus B.$$

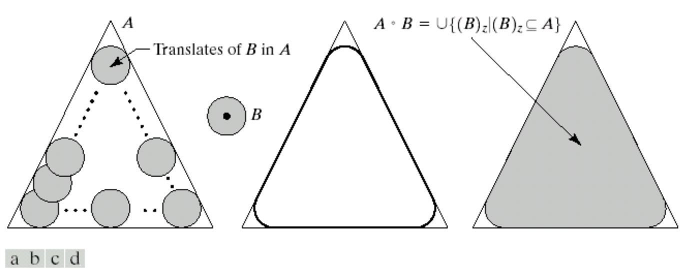


FIGURE 9.8 (a) Structuring element B "rolling" along the inner boundary of A (the dot indicates the origin of B). (c) The heavy line is the outer boundary of the opening. (d) Complete opening (shaded).

32

32

Opening

1	1	1	1	1	1	1	
			1	1	1	1	
			1	1	1	1	
		1	1	1	1	1	
			1	1	1	1	
		1	1				

Binary image A

1	1	1
1	1	1
1	1	1

Structuring element B

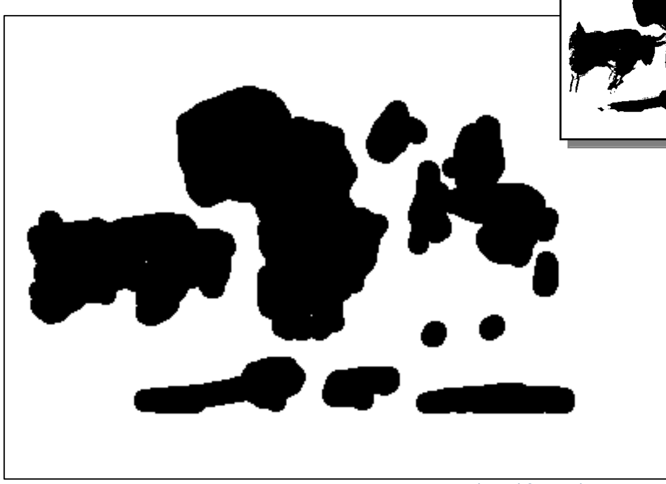
			1	1	1	1	
			1	1	1	1	
			1	1	1	1	
			1	1	1	1	
			1	1	1	1	

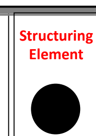
Opening result

33

33

Opening





Pablo Picasso, *Pass with the Cape*, 1960

Adapted from John Goutsias, Johns Hopkins Univ.

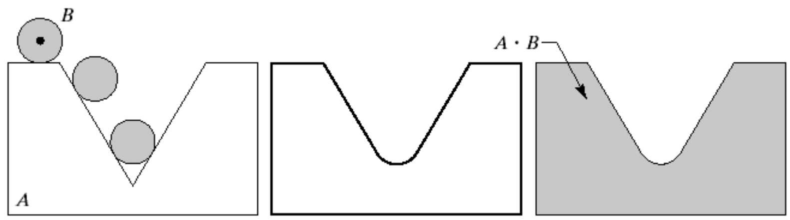
34

34

Closing

- The *closing* of binary image A by structuring element B is denoted by $A \bullet B$ and is defined by

$$A \bullet B = (A \oplus B) \ominus B.$$



a b c

FIGURE 9.9 (a) Structuring element B "rolling" on the outer boundary of set A. (b) Heavy line is the outer boundary of the closing. (c) Complete closing (shaded).

Closing

1	1	1	1	1	1	1	
			1	1	1	1	
			1	1	1	1	
		1	1	1	1	1	
			1	1	1	1	
		1	1				

Binary image A

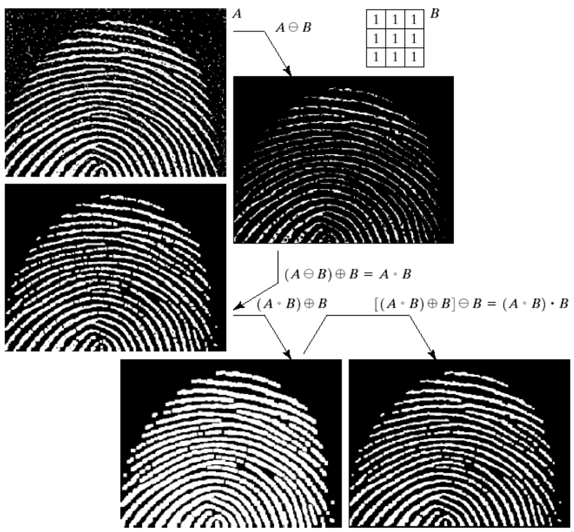
1	1	1
1	1	1
1	1	1

Structuring element B

1	1	1	1	1	1	1	
		1	1	1	1	1	
		1	1	1	1	1	
		1	1	1	1	1	
		1	1	1	1	1	
		1	1				

Closing result

Examples



a b c d e f

FIGURE 9.11 (a) Noisy image. (c) Eroded image. (d) Opening of A. (d) Dilation of the opening. (e) Closing of the opening. (Original image for this example courtesy of the National Institute of Standards and Technology.)

Adapted from Gonzales and Woods

37

37

Properties

- Dilation and erosion are duals of each other with respect to set complementation and reflection, i.e.,

$$(A \ominus B)^c = A^c \oplus \check{B}.$$

- Opening and closing are duals of each other with respect to set complementation and reflection, i.e.,

$$(A \bullet B)^c = A^c \circ \check{B}.$$

38

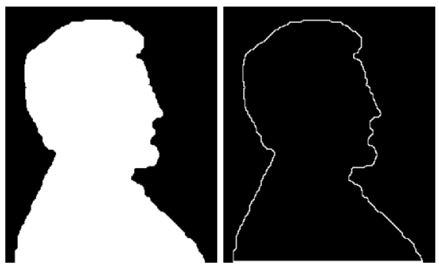
38

Boundary extraction

- The *boundary* of a set A can be obtained by first eroding A by B and then performing the set difference between A and its erosion, i.e.,

$$\text{boundary}(A) = A - (A \ominus B)$$

where B is a suitable structuring element.



a b
FIGURE 9.14 (a) A simple binary image, with 1's represented in white. (b) Result of using Eq. (9.5-1) with the structuring element in Fig. 9.13(b).

40

40

Region filling

- Given set A containing the boundary points of a region, and a point p inside the boundary, the following procedure *fills the region* with 1's:

$$X_k = (X_{k-1} \oplus B) \cap A^c, \quad k = 1, 2, 3, \ldots$$

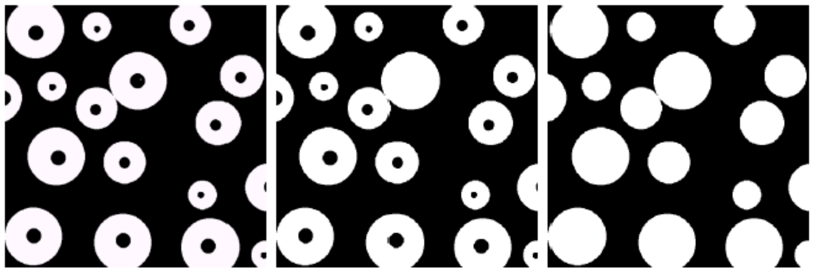
where $X_0 = p$ and B is the cross structuring element. The procedure terminates at iteration step k if $X_k = X_{k-1}$.

- The set union of X_k and A contains the filled set and its boundary.

41

41

Region filling



a b c

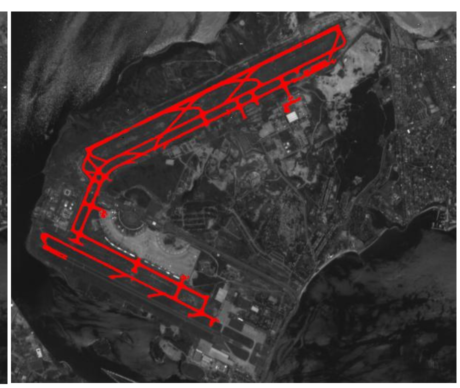
FIGURE 9.16 (a) Binary image (the white dot inside one of the regions is the starting point for the region-filling algorithm). (b) Result of filling that region (c) Result of filling all regions.

Adapted from Gonzales and Woods

42

42

Examples

Detecting runways in satellite airport imagery

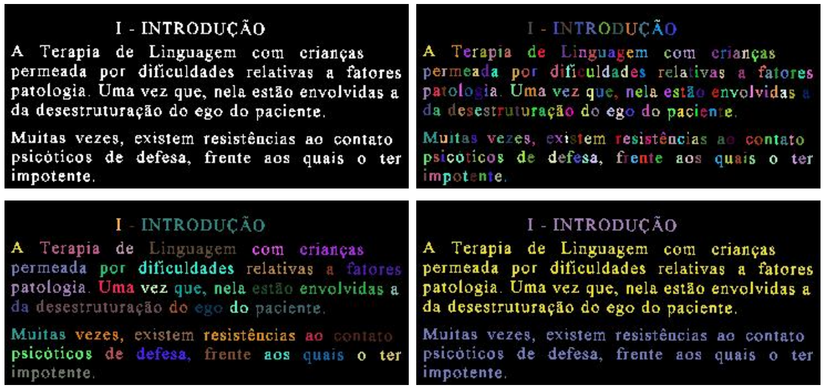
http://www.mmorph.com/mxmorph/html/mmdemos/mmdairport.html

Note: These links do not work anymore, but you can use the Wayback Machine for older versions.

43

43

Examples



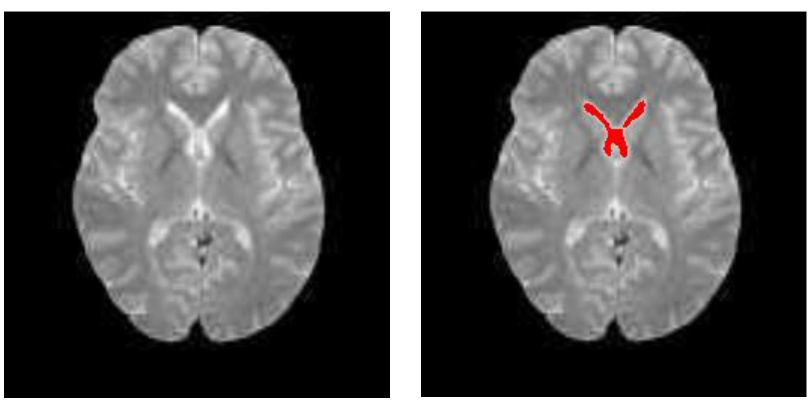
Segmenting letters, words and paragraphs

http://www.mmorph.com/mxmorph/html/mmdemos/mmdlabeltext.html

44

44

Examples



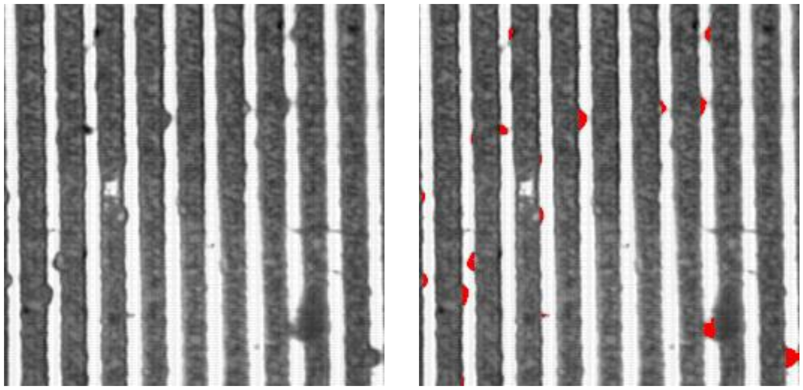
Extracting the lateral ventricle from an MRI image of the brain

http://www.mmorph.com/mxmorph/html/mmdemos/mmdbrain.html

45

45

Examples



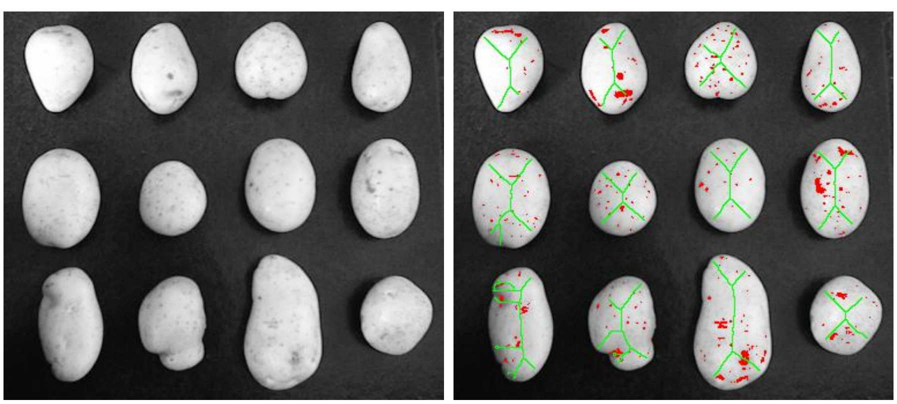
Detecting defects in a microelectronic circuit

http://www.mmorph.com/mxmorph/html/mmdemos/mmdlith.html

46

46

Examples



Grading potato quality by shape and skin spots

http://www.mmorph.com/mxmorph/html/mmdemos/mmdpotatoes.html

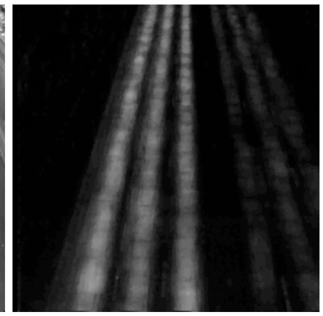
47

47

Examples


Traffic scene


Temporal average


Average of differences

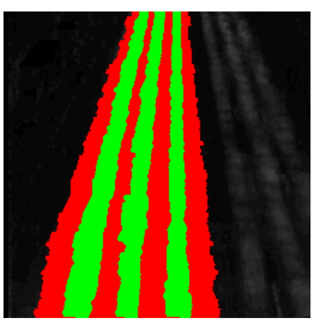
Lane detection example

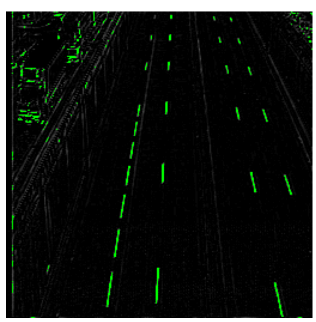
Adapted from CMM/ENSMP/ARMINES

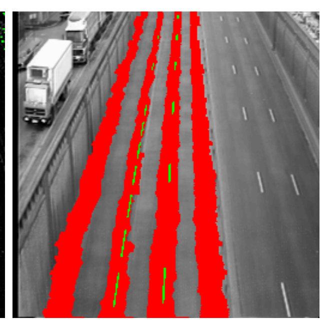
48

48

Examples


Threshold and dilation to detect lane markers


White line detection (top hat)


Detected lanes

Lane detection example

Adapted from CMM/ENSMP/ARMINES

49

49

Pixels and neighborhoods

- In many algorithms, not only the value of a particular pixel, but also the values of its neighbors are used when processing that pixel.
- The two most common definitions for neighbors are the 4-neighbors and the 8-neighbors of a pixel.

	N	
W	*	E
	S	

a) four-neighborhood N_4

NW	N	NE
W	*	E
SW	S	SE

b) eight-neighborhood N_8

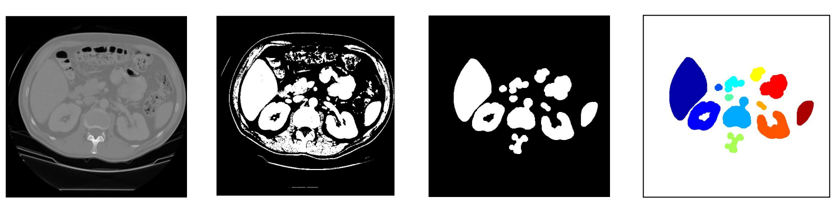
Figure 3.2: The two most common neighborhoods of a pixel.

50

50

Connected components analysis

- Once you have a binary image, you can identify and then analyze each connected set of pixels.
- The connected components operation takes in a binary image and produces a labeled image in which each pixel has the integer label of either the background (0) or a component.



Original image Thresholded image After morphology Connected components

51

51

Connected components analysis

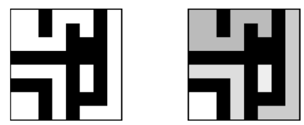
- Methods for connected components analysis:
  - Recursive labeling (almost never used)
  - Parallel growing (needs parallel hardware)
  - Row-by-row (most common)
    - Classical algorithm
    - Run-length algorithm (see Shapiro-Stockman)

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

a) binary image

1	1	0	1	1	1	0	2
1	1	0	1	0	1	0	2
1	1	1	1	0	0	0	2
0	0	0	0	0	0	0	2
3	3	3	3	0	4	0	2
0	0	0	3	0	4	0	2
5	5	0	3	0	0	0	2
5	5	0	3	0	2	2	2

b) connected components labeling



c) binary image and labeling, expanded for viewing

Adapted from Shapiro and Stockman

52

Connected components analysis

- Recursive labeling algorithm:
  1. Negate the binary image so that all 1s become -1s.
  2. Find a pixel whose value is -1, assign it a new label, call procedure *search* to find its neighbors that have values -1, and recursively repeat the process for these neighbors.

Compute the connected components of a binary image.
B is the original binary image.
LB will be the labeled connected component image.

```
procedure recursive_connected_components(B, LB);
{
LB := negate(B);
label := 0;
find_components(LB, label);
print(LB);
}

procedure find_components(LB, label);
{
for L := 0 to MaxRow
   for P := 0 to MaxCol
      if LB[L,P] == -1 then
         {
         label := label + 1;
         search(LB, label, L, P);
         }
}

procedure search(LB, label, L, P);
{
LB[L,P] := label;
Nset := neighbors(L, P);
for each (L',P') in Nset
   {
   if LB[L',P'] == -1
   then search(LB, label, L', P');
   }
}
```

53

Connected components analysis

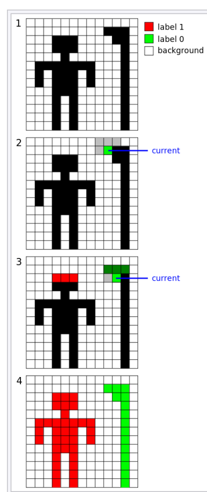
- Row-by-row labeling algorithm:
  1. The first pass propagates a pixel's label to its neighbors to the right and below it.
     (Whenever two different labels can propagate to the same pixel, these labels are recorded as an equivalence class.)
  2. The second pass performs a translation, assigning to each pixel the label of its equivalence class.
- A union-find data structure is used for efficient construction and manipulation of equivalence classes represented by tree structures.

54

54

The pseudocode is:

```
algorithm TwoPass(data) is
    linked = []
    labels = structure with dimensions of data, initialized with the value of Background

    First pass

    for row in data do
        for column in row do
            if data[row][column] is not Background then

                neighbors = connected elements with the current element's value

                if neighbors is empty then
                    linked[NextLabel] = set containing NextLabel
                    labels[row][column] = NextLabel
                    NextLabel += 1

                else

                    Find the smallest label

                    L = neighbors labels
                    labels[row][column] = min(L)
                    for label in L do
                        linked[label] = union(linked[label], L)

    Second pass

    for row in data do
        for column in row do
            if data[row][column] is not Background then
                labels[row][column] = find(labels[row][column])

    return labels
```



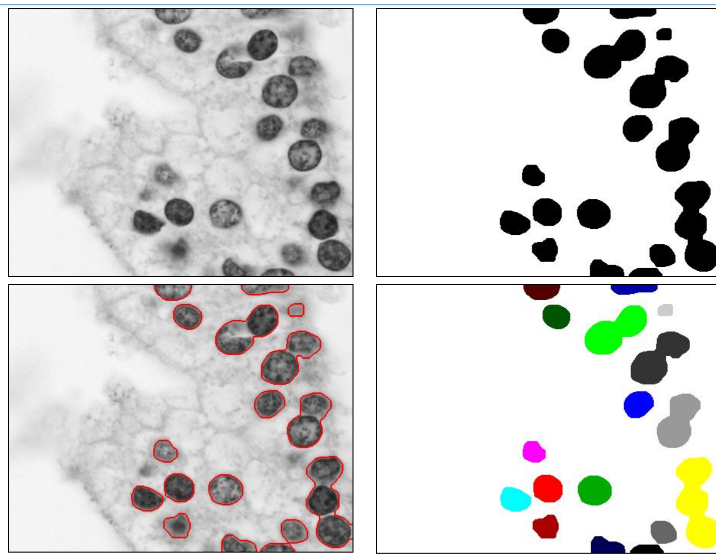
Sample graphical output from running the two-pass algorithm on a binary image. The first image is unprocessed, while the last one has been recolored with label information. Darker hues indicate the neighbors of the pixel being processed.

Source Wikipedia

55

55

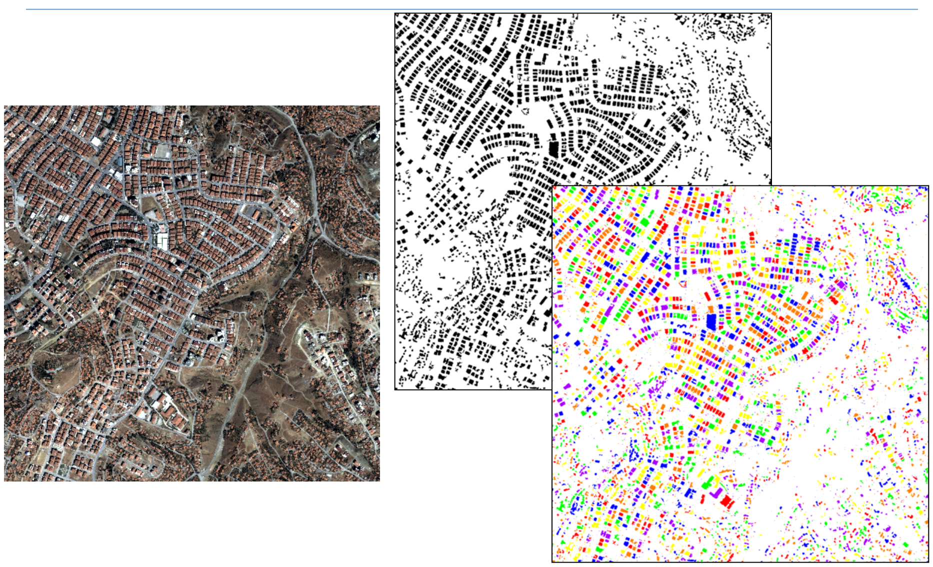
Connected components analysis



60

60

Connected components analysis



61

61