Digital Image Processing Lecture 6

Lab 3!

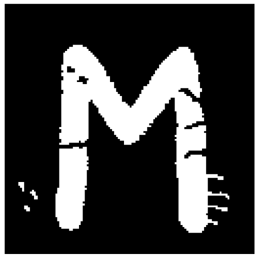
- Morphological image processing
  - Dilation (expansion) and erosion (contraction).
  - Opening and closing.
- Connectivity, 4- and 8-adjacency
- Labelling with the Flood fill algorithm
- Connectivity preserving shrinking (to point or skeleton)
- Detection of ramifications and end points
- Distance maps and measures
- The true skeleton, the medial axis transformation (MAT)
- Numbers according to Gonzales & Woods, Global Edition, 4th edition. (Numbers in other editions may vary).
- Gonzales & Woods:
  - Section 2.5
  - Section 9.1-9.5
  - Section 11.2, last part
  - G&W lacks some of the contents in this lecture.

Maria Magnusson, Computer Vision Lab., Dept. of Electrical Engineering, Linköping University

Morphological image processing

- Fundamental operations for morphological processing
  - Dilation (expansion)
  - Erosion (contraction)
- Opening = Erosion + Dilation
- Closing = Dilation + Erosion

Fundamental mor-phological operations



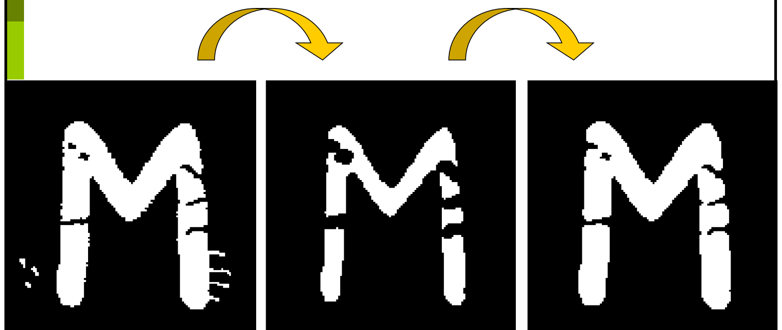
Dilation (expansion)

Erosion (contraction)

Opening = Erosion + Dilation

erosion

dilation



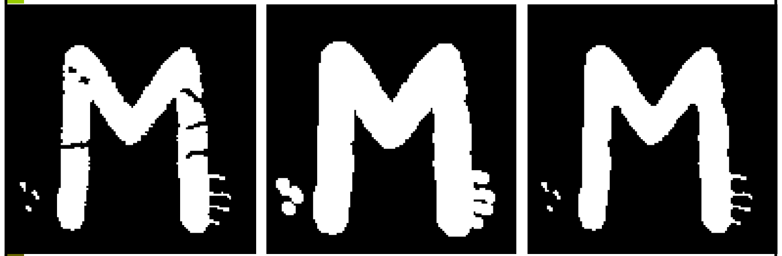
Removes small objects and spurs

Reset size

Closing = dilation + erosion

dilation

erosion



Fills small holes and cracks

Reset size

Opening followed by closing

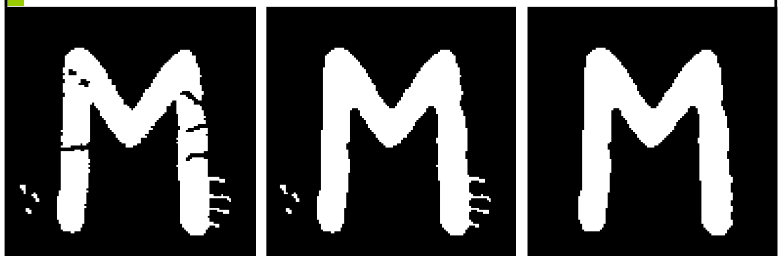
opening

closing

Closing followed by opening

closing

opening



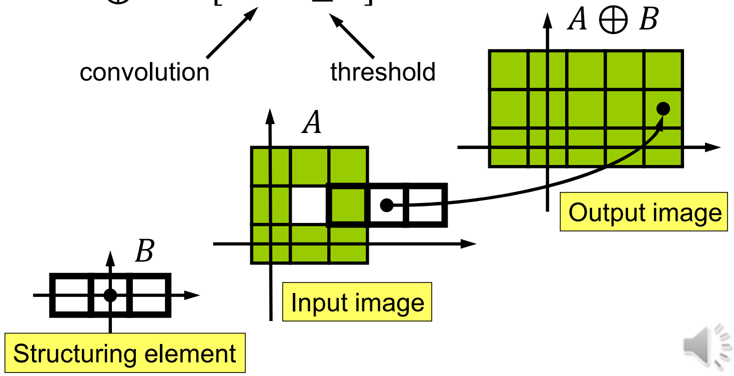
Very similar result!

Opening followed by closing

Closing followed by opening

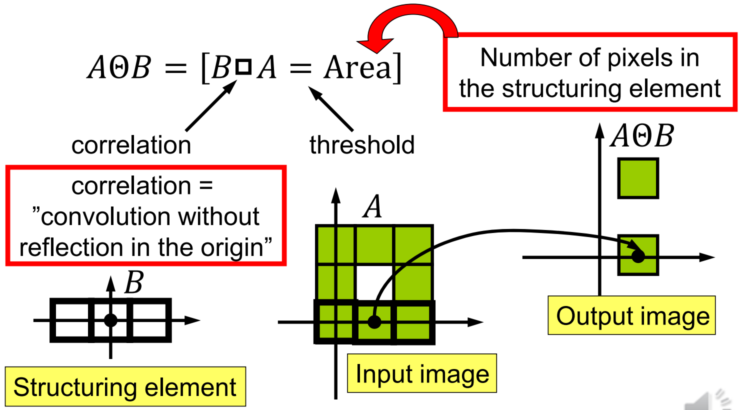
Dilation, formal description

$$A \oplus B = [A * B \geq 1]$$



Erosion, formal description

$$A \Theta B = [B \square A = \text{Area}]$$



Description of erosion and dilation according to G&W

Erosion of A by B is the set of all point z such that B, translated by z, is contained in A.

$$A \Theta B = \{ z | (B)_z \subseteq A \}$$

(9-3)

Reflect B in the origin and translate it by z. **Dilation** of A by B is the set of all displacements z, such that $\hat{B}$ and A overlap by at least one element.

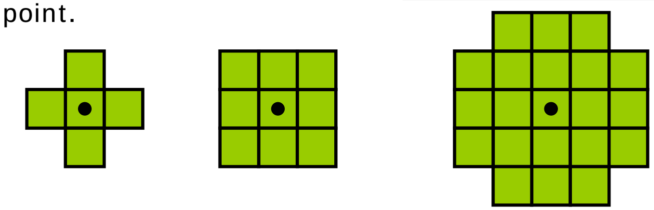
$$A \oplus B = \left\{ z \middle| (\hat{B})_z \cap A \neq \emptyset \right\}$$

(9-6)

Common structuring element

- Binary filters are often called structuring elements.
- The most common structuring elements are illustrated below.
- The origin is marked with a point.

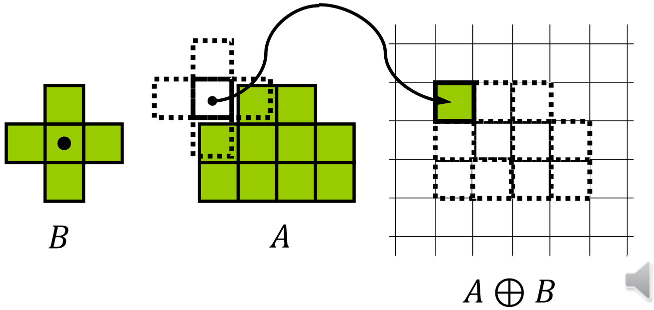
This “octagonal” structuring element gives a more symmetrical dilation/erosion. The same effect can be obtained by alternately using $d^{(4)}$ and $d^{(8)}$.



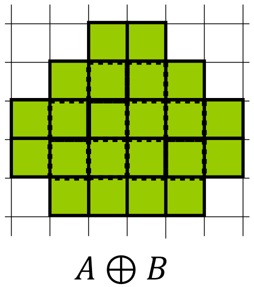
$d^{(4)}$ $d^{(8)}$ $d^{(oct)}$

Exercise: Finish the dilation!

$$A \oplus B = \{A * B \geq 1\} = \left\{z \,\middle|\, (\hat{B})_z \cap A \neq \emptyset\right\}$$

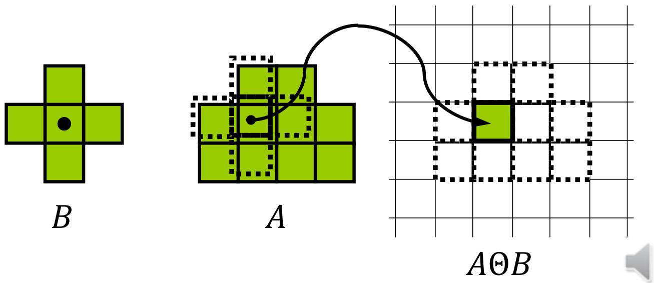


Answer: Dilation finished

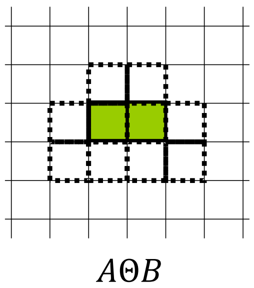


Exercise: Finish the erosionen!

$$A \Theta B = \{A \square B = \text{Area}\} = \{z | (B)_z \subseteq A\}$$



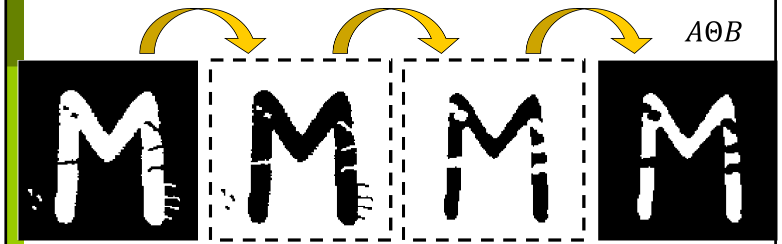
Answer: erosion finished



The two exercises give:

- dilation
  - If the structuring element touches the object, the pixel under the centre of the structuring element will be set to 1.
  - A $d^{(4)}$–structuring element makes pixel neighbors to the side to be set to 1.
  - A $d^{(8)}$–structuring element makes pixel neighbors to the side and corner to be set to 1.
- erosion
  - If the structuring element is contained inside the object, the pixel under the centre of the structuring element will be kept to 1.
  - A $d^{(4)}$–structuring element makes pixel neighbors to the side to be set to 0.
  - A $d^{(8)}$–structuring element makes pixel neighbors to the side and corner to be set to 0.

Duality for erosion and dilation



erosion = dilation of the background (C – denotes the binary inverse)

$$A \Theta B = \left(A^C \oplus \hat{B}\right)^C \quad (9\text{-}8)$$

Similarly: Dilation = erosion of the background

$$A \oplus B = \left(A^C \Theta \hat{B}\right)^C \quad (9\text{-}9)$$

Properties for opening and closing

- Properties for opening

$$A \circ B = (A \Theta B) \oplus B \quad (9\text{-}10)$$

$A \circ B$ is a subset of A

$$(A \circ B) \circ B = A \circ B$$

More than one opening makes no sense!

- Properties for closing

$$A \bullet B = (A \oplus B) \Theta B \quad (9.11)$$

A is a subset of $A \bullet B$

$$(A \bullet B) \bullet B = A \bullet B$$

More than one closing makes no sense!

Example with opening and closing

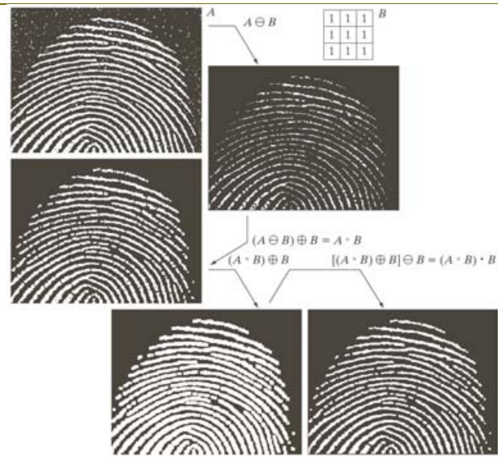
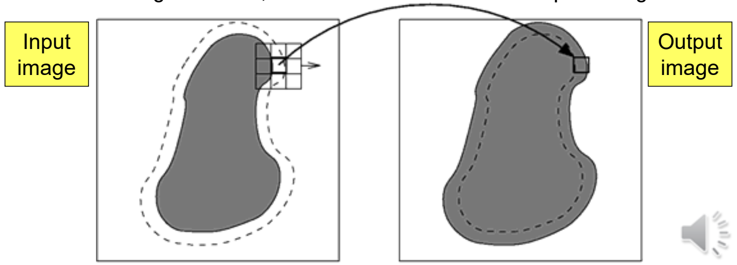


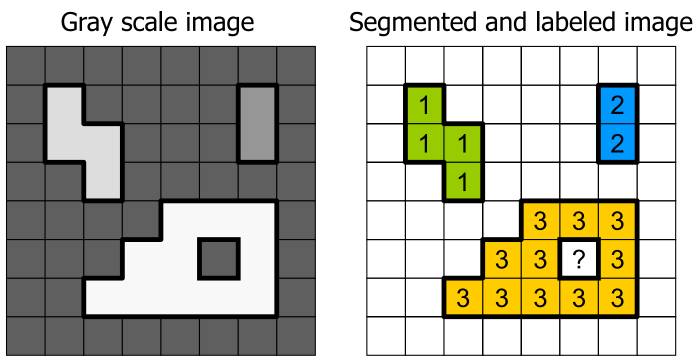
Fig. 9.11

Parallel implementation is used for dilation and erosion

- Works in parallel with one input image and one output image.
- Is applied on the whole image in one run.
- Each pixel is affected only by the area under the structuring element.
- Simple implementation: Loop over the input image, apply the structuring element, write the result to the output image



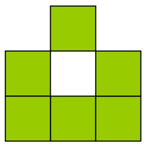
Segmentation and labeling of objects



Which pixels are connected?

Which pixels are adjacent and connected?

- $d^{(4)}$-adjacency: 2 pixels are adjacent if they have a common side.
- $d^{(8)}$-adjacency: 2 pixels are adjacent if they have a common side or corner.



- $d^{(4)}$-connectivity: 2 objects, 1 connected background
- $d^{(8)}$- connectivity: 1 object with a hole

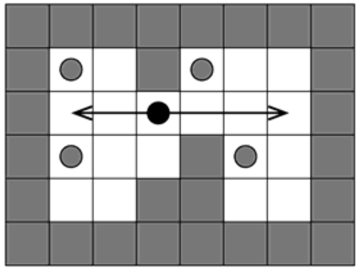
- $d^{(4)}$-connected object <=> $d^{(8)}$-conn. background
- $d^{(8)}$-connected object <=> $d^{(4)}$-conn. background

Labeling with the Flood fill algorithm

- The Flood fill algorithm fills a connected region or area with a label value.
- The region is specified with an arbitrary selected starting point, from which the algorithm searches the area and fills its pixels with the desired label value.
- One variant of the Flood fill algorithm is the following:
- **Put the start pixel on the stack.**
- **As long as the stack is not empty:**
  - **Fetch a pixel *p* from the stack.**
  - **Look to the right and to the left from *p* to locate the interval *I* of pixels on the row.**
  - **Along *I*, check the pixels above and below the row. All pixels that are first in an interval are put on the stack.**
  - **Fill the interval *I* with the label value.**
  - **Fetch a new pixel *p* from the stack.**

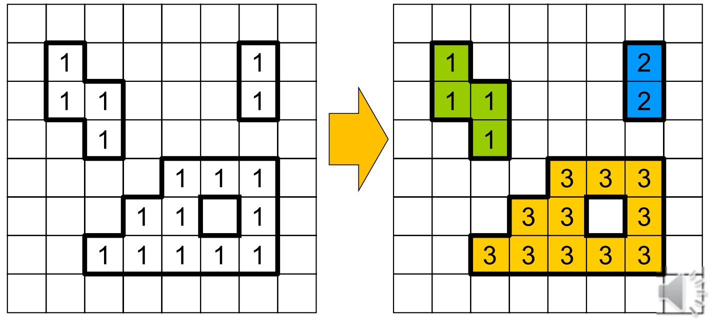
Labeling with the Flood fill algorithm, cont.

- The figure illustrates one step of the algorithm.
- The start pixel is the black dot.
- The interval I for this pixel is marked with arrows.
- The row above and below is examined, and four intervals are found, starting with the four pixels marked with gray dots.
- These four pixels are put on the stack.
- Then the interval is filled and the process is repeated recursively until the stack is empty.



Labeling with the Flood fill algorithm, cont.

- To apply this algorithm on a binary image is trivial.
- Scan the image row-by-row and look for non-labelled pixels,
- When such a pixel is found, use the Flood fill algorithm to fill the region with a unique label value.



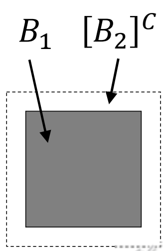
The Hit-or-Miss transformation: A tool for shape matching

$$A \circledast B = (A \Theta B_1) \cap [A^C \Theta B_2] \qquad (9\text{-}16)$$

The object B is found in the image A

The object B_1 matches

The local background B_2 matches



The Hit-or-Miss transformation: A simple example

Detection of right-oriented end-points

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

A

	0	
1	1	0
	0	

B^1

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

$A \circledast B^1$

The object B^1 is found in the image A.

Detection of end points (for 4-connectivity)

B^1

	0	
1	1	0
	0	

B^2

	0	
0	1	0
	1	

B^3

	0	
0	1	1
	0	

B^4

	1	
0	1	0
	0	

- Call the input image A.
- Call the output image C.

$$C = \bigcup_{i=1,2,3,4} \left(A \circledast B^i\right)$$

Detection of ramifications (for 4-connectivity)

B^1

	1	
1	1	0
	1	

B^2

	1	
1	1	1
	0	

B^3

	1	
0	1	1
	1	

B^4

	0	
1	1	1
	1	

- Call the input image A.
- Call the output image C.

$$C = \bigcup_{i=1,2,3,4} \left(A \circledast B^i\right)$$

Ex) Detection of end points and ramifications

For end points:

	0	
1	1	0
	0	

	0	
0	1	0
	1	

	0	
0	1	1
	0	

	1	
0	1	0
	0	

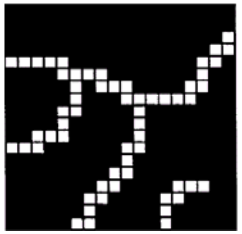
For ramifications:

	1	
1	1	0
	1	

	1	
1	1	1
	0	

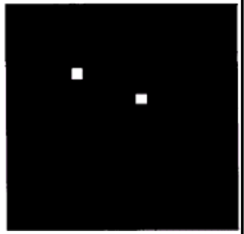
	1	
0	1	1
	1	

	0	
1	1	1
	1	



A part of a finger-print after thinning

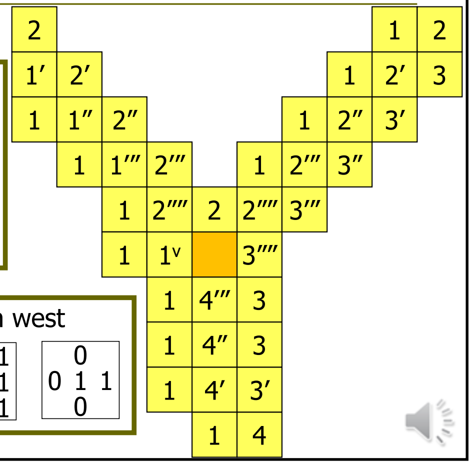
Detected end points



Detected ramifications

Ex) Connectivity preserving shrinking (to point)

1=disappears in phase 1
1′=disappears in phase 1, 2:nd iteration
etc. ...
= final result



phase 1: shrinking from west

	0	
0	1	1
	1	1

	1	1
0	1	1
	0	

		1
0	1	1
		1

	0	
0	1	1
	0	

p. 33

Structuring elements for connectivity preserving shrinking (to point)

phase 1: $P_{1a}=\begin{bmatrix} & 0 & \\ 0 & 1 & 1 \\ & 1 & 1 \end{bmatrix}$ $P_{1b}=\begin{bmatrix} & 1 & 1 \\ 0 & 1 & 1 \\ & 0 & \end{bmatrix}$ $P_{1c}=\begin{bmatrix} & & 1 \\ 0 & 1 & 1 \\ & & 1 \end{bmatrix}$ $P_{1d}=\begin{bmatrix} & 0 & \\ 0 & 1 & 1 \\ & 0 & \end{bmatrix}$ shrinking from west

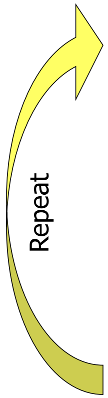
phase 2: $P_{2a}=\begin{bmatrix} & 0 & \\ 1 & 1 & 0 \\ 1 & 1 & \end{bmatrix}$ $P_{2b}=\begin{bmatrix} & 0 & \\ 0 & 1 & 1 \\ & 1 & 1 \end{bmatrix}$ $P_{2c}=\begin{bmatrix} & 0 & \\ & 1 & \\ 1 & 1 & 1 \end{bmatrix}$ $P_{2d}=\begin{bmatrix} & 0 & \\ 0 & 1 & 0 \\ & 1 & \end{bmatrix}$ shrinking from north

phase 3: $P_{3a}=\begin{bmatrix} 1 & 1 & \\ 1 & 1 & 0 \\ & 0 & \end{bmatrix}$ $P_{3b}=\begin{bmatrix} & 0 & \\ 1 & 1 & 0 \\ 1 & 1 & \end{bmatrix}$ $P_{3c}=\begin{bmatrix} 1 & & \\ 1 & 1 & 0 \\ 1 & & \end{bmatrix}$ $P_{3d}=\begin{bmatrix} & 0 & \\ 1 & 1 & 0 \\ & 0 & \end{bmatrix}$ shrinking from east

phase 4: $P_{4a}=\begin{bmatrix} & 1 & 1 \\ 0 & 1 & 1 \\ & 0 & \end{bmatrix}$ $P_{4b}=\begin{bmatrix} 1 & 1 & \\ 1 & 1 & 0 \\ & 0 & \end{bmatrix}$ $P_{4c}=\begin{bmatrix} 1 & 1 & 1 \\ & 1 & \\ & 0 & \end{bmatrix}$ $P_{4d}=\begin{bmatrix} & 1 & \\ 0 & 1 & 0 \\ & 0 & \end{bmatrix}$ shrinking from south

p. 34

Connectivity preserving shrinking (to point), algorithm



- Apply the structuring elements in phase 1 on the input image. If match, set the output image to 0.
- Input image := Output image
- Apply the structuring elements in phase 2 on the input image. If match, set the output image to 0.
- Input image := Output image
- Apply the structuring elements in phase 3 on the input image. If match, set the output image to 0.
- Input image := Output image
- Apply the structuring elements in phase 4 on the input image. If match, set the output image to 0.
- Input image := Output image

p. 35

Connectivity preserving shrinking (to point), algorithm

The structuring elements was defined on two slides before.
Call the in−image $A = A_0^4$

$$A_i^1 = A_{i-1}^4 - (A_{i-1}^4 \circledast P_{1a}) \cup (A_{i-1}^4 \circledast P_{1b}) \cup (A_{i-1}^4 \circledast P_{1c}) \cup (A_{i-1}^4 \circledast P_{1d})$$
$$A_i^2 = A_i^1 - (A_i^1 \circledast P_{2a}) \cup (A_i^1 \circledast P_{2b}) \cup (A_i^1 \circledast P_{2c}) \cup (A_i^1 \circledast P_{2d})$$
$$A_i^3 = A_i^2 - (A_i^2 \circledast P_{3a}) \cup (A_i^2 \circledast P_{3b}) \cup (A_i^2 \circledast P_{3c}) \cup (A_i^2 \circledast P_{3d})$$
$$A_i^4 = A_i^3 - (A_i^3 \circledast P_{4a}) \cup (A_i^3 \circledast P_{4b}) \cup (A_i^3 \circledast P_{4c}) \cup (A_i^3 \circledast P_{4d})$$

Iterate $i = 1, 2, 3, \ldots$

p. 36

Structuring elements for thinning, connectivity preserving shrinking (to skeleton)

phase 1: $\begin{bmatrix} & 0 & \\ 0 & 1 & 1 \\ & 1 & 1 \end{bmatrix}$ $\begin{bmatrix} & 1 & 1 \\ 0 & 1 & 1 \\ & 0 & \end{bmatrix}$ $\begin{bmatrix} & & 1 \\ 0 & 1 & 1 \\ & & 1 \end{bmatrix}$ ~~$\begin{bmatrix} & 0 & \\ 0 & 1 & 1 \\ & 0 & \end{bmatrix}$~~ shrinking from west

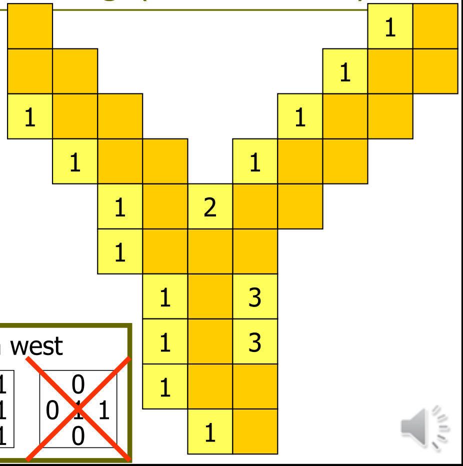
phase 2: $\begin{bmatrix} & 0 & \\ 1 & 1 & 0 \\ 1 & 1 & \end{bmatrix}$ $\begin{bmatrix} & 0 & \\ 0 & 1 & 1 \\ & 1 & 1 \end{bmatrix}$ $\begin{bmatrix} & 0 & \\ & 1 & \\ 1 & 1 & 1 \end{bmatrix}$ ~~$\begin{bmatrix} & 0 & \\ 0 & 1 & 0 \\ & 1 & \end{bmatrix}$~~ shrinking from north

phase 3: $\begin{bmatrix} 1 & 1 & \\ 1 & 1 & 0 \\ & 0 & \end{bmatrix}$ $\begin{bmatrix} & 0 & \\ 1 & 1 & 0 \\ 1 & 1 & \end{bmatrix}$ $\begin{bmatrix} 1 & & \\ 1 & 1 & 0 \\ 1 & & \end{bmatrix}$ ~~$\begin{bmatrix} & 0 & \\ 1 & 1 & 0 \\ & 0 & \end{bmatrix}$~~ shrinking from east

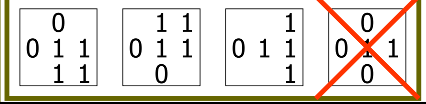
phase 4: $\begin{bmatrix} & 1 & 1 \\ 0 & 1 & 1 \\ & 0 & \end{bmatrix}$ $\begin{bmatrix} 1 & 1 & \\ 1 & 1 & 0 \\ & 0 & \end{bmatrix}$ $\begin{bmatrix} 1 & 1 & 1 \\ & 1 & \\ & 0 & \end{bmatrix}$ ~~$\begin{bmatrix} & 1 & \\ 0 & 1 & 0 \\ & 0 & \end{bmatrix}$~~ shrinking from south

Ex) Thinning, connectivity preserving shrinking (to skeleton)

1 = disappears in phase 1
2 = disappears in phase 2
3 = disappears in phase 3
etc. ...
= final result

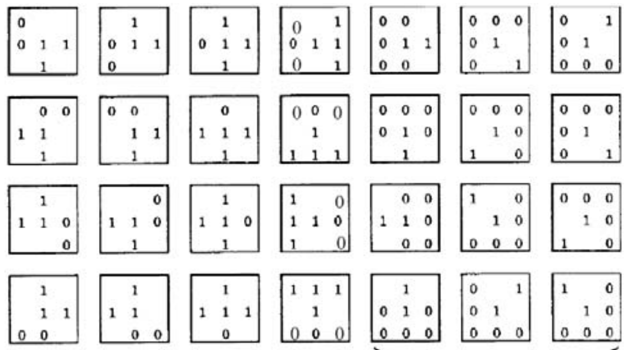


phase 1: shrinking from west



Structuring elements for 8-connective connectivity preserving shrinking (to point)

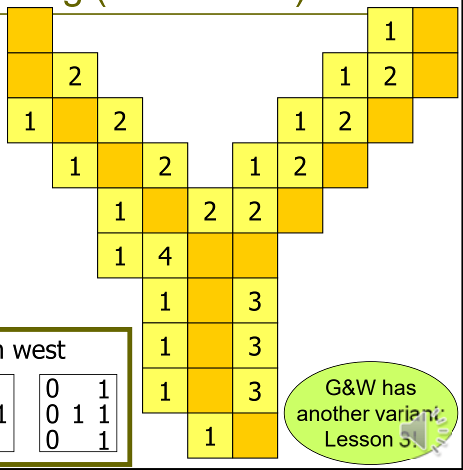
phase1:
phase2:
phase3:
phase4:



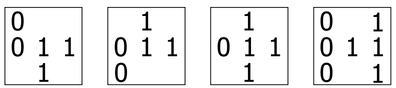
Excluded during thinning

Ex) Thinning, 8-connective connectivity preserving shrinking (to skeleton)

1 = disappears in phase 1
2 = disappears in phase 2
3 = disappears in phase 3
etc. ...
= final result



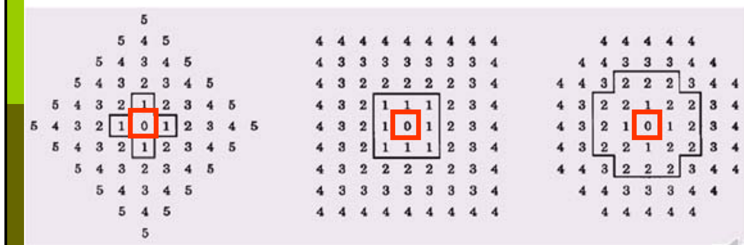
phase 1: Shrinking from west



G&W has another variant: Lesson 3!

The structuring element gives a metric

The structuring element is applied several times on the object (here a point). Note the iteration number. It gives the distance to the object. For every iteration, the distance is incremented.



$d^{(4)}$-metric $d^{(8)}$-metric $d^{(oct)}$-metric

Distances in digital images

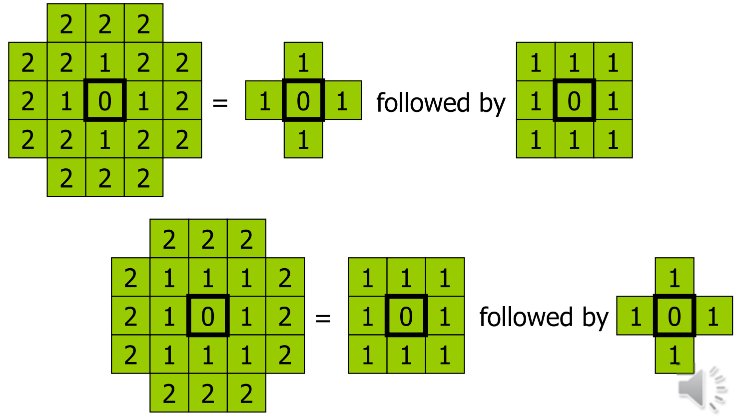
The distance measures below are approximate, but easy to calculate. Sometimes they can replace the correct **Euclidian distance.**

- $d^{(4)}$-metric
- $d^{(8)}$-metric
- $d^{(oct)}$-metric

d is a distance function or metric if:

- d(p,q) >= 0 with d(p,q) = 0 iff p = q
- d(p,q) = d(q,p)
- d(p,q) <= d(p,r) + d(r,q)

The octagonal metric can be implemented by alternating $d^{(4)}$ & $d^{(8)}$

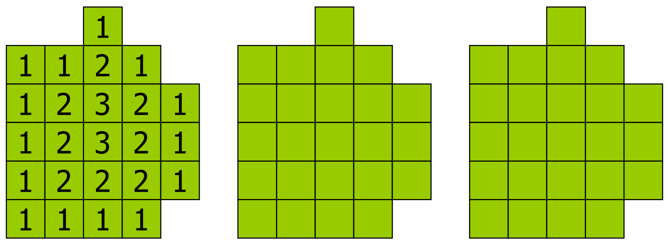


The Euclidian distance map

- Gives an exact distance map.
- Cannot be implemented by dilating a structural element.
- Implemented in MATLAB together with $d^{(4)}$ and $d^{(8)}$ distance maps.
- $d^{(4)}$ and $d^{(8)}$ distance maps are faster to calculate than the Euclidian distance map.
- However, a rather fast implementation was suggested by Breu et al., 1995.

√2	1	√2	√5
1	0	1	2
√2	1	√2	√5

The object has a $d^{(4)}$ distance map. Perform $d^{(8)}$ and $d^{(oct)}$ distance mapping!



$d^{(4)}$-metric $d^{(8)}$-metric $d^{(oct)}$-metric

p. 45

Answer: The object shown with $d^{(4)}$ and $d^{(8)}$ distance maps

		1		
1	1	2	1	
1	2	3	2	1
1	2	3	2	1
1	2	2	2	1
1	1	1	1	

$d^{(4)}$-metric

		1		
1	1	1	1	
1	2	2	1	1
1	2	2	2	1
1	2	2	1	1
1	1	1	1	

$d^{(8)}$-metric

p. 46

Answer: The object shown with $d^{(oct)}$ distance maps

Here: $d^{(8)}$- $d^{(4)}$- $d^{(8)}$- ...

		1		
1	1	1	1	
1	2	2	1	1
1	2	3	2	1
1	2	2	1	1
1	1	1	1	

$d^{(oct)}$-metric

Here: $d^{(4)}$- $d^{(8)}$- $d^{(4)}$- ...

		1		
1	1	2	1	
1	2	2	2	1
1	2	3	2	1
1	2	2	2	1
1	1	1	1	

$d^{(oct)}$-metric

p. 47

Distance measures

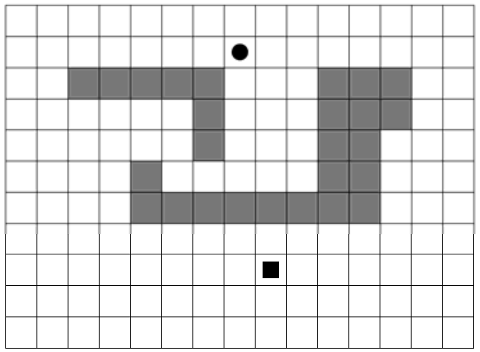
- $\text{Euclidian distance} = \sqrt{x^2 + y^2}$ — max error: 0% (2.19)
- $d^{(4)}(x, y) = |x| + |y|$ — max error: -41% (2.20)
- $d^{(8)}(x, y) = \max(|x|, |y|)$ — max error: $+41\%$ (2.21)
- $d^{(oct)}(x, y) = \max\left(|x|, \left\lfloor \frac{2}{3}(|x| + |y| + 1) \right\rfloor, |y|\right)$

max error: 11% (for distances ≥ 2)

p. 48

Shortest path

- The task is to connect the circular start point with the box-shaped end point. The grey pixels are obstacles and must be avoided.



Shortest path, cont.

- A $d^{(oct)}$ distance map is generated from the circular start point.

7	6	5	4	3	2	2	1	2	2	3	4	5	6	7
7	6	5	4	3	2	1	○	1	2	3	4	5	6	7
7	6						1	2	2				6	7
8	7	8	8	7	7		2	2	3				7	8
8	8	8	7	6	6		3	3	4			8	8	8
9	9	8	8		5	4	4	4	4			9	9	9
10	10	9	9									10	10	10
11	10	10	10	10	11	12	13	14	14	13	12	11	11	11
12	11	11	11	11	12	12	13	14	14	13	12	12	12	12
12	12	12	12	12	12	13	14	14	14	14	13	13	13	13
13	13	13	13	13	13	14	14	15	15	14	14	14	14	14

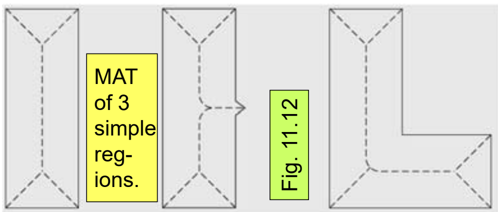
Shortest path, cont.

- Start the tracking from the end point. Check the eight neighbors. Choose the smallest one. If there are 2 or more with the same value, choose the closest one, i.e. not a diagonal.

7	6	5	4	3	2	2	1	2	2	3	4	5	6	7
7	6	5	4	3	2	1	○	1	2	3	4	5	6	7
7	6						1	2	2				6	7
8	7	8	8	7	7		2	2	3				7	8
8	8	8	7	6	6		3	3	4			8	8	8
9	9	8	8		5	4	4	4	4			9	9	9
10	10	9	9									10	10	10
11	10	10	10	10	11	12	13	14	14	13	12	11	11	11
12	11	11	11	11	12	12	13	14	14	13	12	12	12	12
12	12	12	12	12	12	13	14	14	14	14	13	13	13	13
13	13	13	13	13	13	14	14	15	15	14	14	14	14	14

The true skeleton, the medial axis transformation (MAT)

- The MAT of a region R with border B is as follows. For each point p in R, we find its closest neighbor in B. If p has more than one such neighbor, it is said to belong to the medial axis (skeleton) of R.
- “The prairie fire concept”: Consider an image region as uniform, dry grass, and suppose that a fire is lit along its border. All fire fronts will advance into the region at the same speed. The MAT of the region is the set of points reached by more than one fire front at the same time.



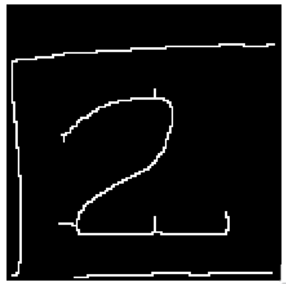
MAT can be computed in many different ways. G&W use a special thinning algorithm. We will show an example with the Euclidean distance Transform.

Comparison between a skeleton received by thinning and MAT

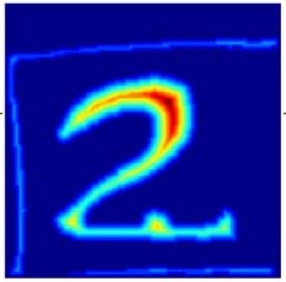
- This and next slide: own experiments with Matlab

Binary number

Normal thinning skeleton (4-connected)

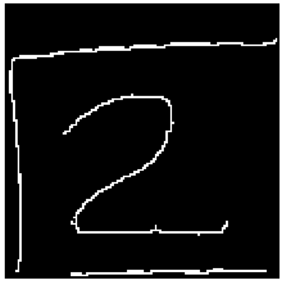


Euclidian distance transform



Lo-cal max

- MAT was received in this way:
  - Compute the Euclidian distance transform
  - Detect local maxima
  - Perform “4-connectivity preserving shrinking to point” provided that the local maxima are kept.

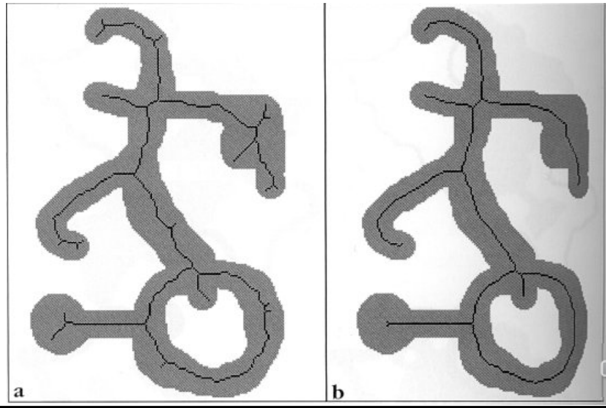


MAT-skele-ton

Comparison between a skeleton received by thinning and MAT

- From Russ: Image Processing Handbook, Ch. 7, Fig. 90.

A thinning skeleton has often spurs that must be removed by post-processing.



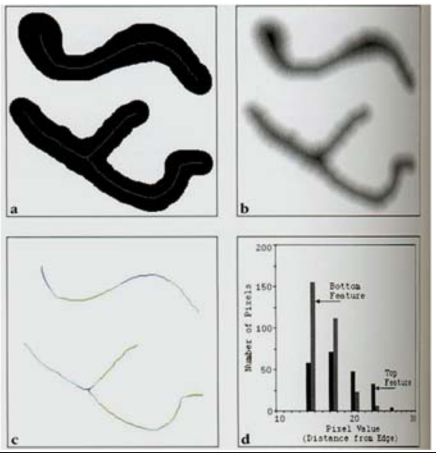
MAT is smooth and looks nice.

Ex) Distance map + skeleton can be used to measure thickness

Lesson 3

Fibers (?) with skeleton

Distance map



The skeleton multiplied with the distance transform

From Russ: Image Processing Handbook, Ch. 7, Fig. 91

Statistics regarding thickness