MORPHOLOGY ON BINARY IMAGES

Ole-Johan Skrede

10.05.2017

INF2310 - Digital Image Processing

Department of Informatics
The Faculty of Mathematics and Natural Sciences
University of Oslo

After original slides by Fritz Albregtsen

TODAY'S LECTURE

- Introduction
- Fundamental morphological operators
- Composed morphological operators
- Application examples
- Sections from G&W:
  - 9 - 9.5.2
  - 9.5.5
  - 2.6.4

INTRODUCTION

INTRODUCTION

- Morphological operations are used as a step in image processing and analysis.
- It is used to modify the shape of objects in an image, by using local operations.
- It can be used to *remove unwanted effects* in segmentation post-processing
  - Remove small objects (that is assumed to be noise)
  - Smooth the edges of larger objects
  - Fill holes in objects
  - Link objects together
- It can be used as a part of *object description and analysis*
  - Locate the boundary of objects
  - Thin objects
  - Locating objects with a certain structure
  - Locating patterns in an image
- The operations are small, and often very fast.

MOTIVATIONAL EXAMPLE

In this example, the morphological operation called *dilation* is used to join fragmented characters in a poor resolution image of a text.

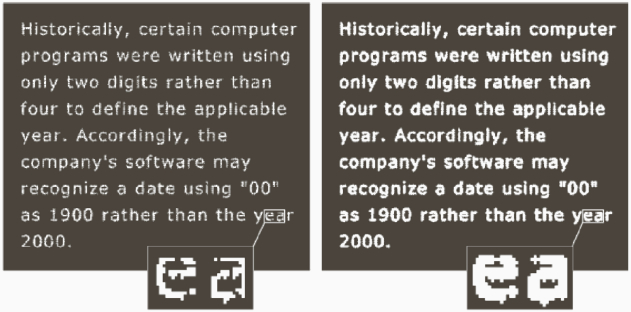


Figure 1: Image before (left) and after (right) dilation with the structuring element shown at the bottom

GENERAL DEFINITION

- We start by formally defining two fundamental operations, *dilation* and *erosion*, for graylevel images.
- Note that the curriculum is only for *binary images*, but we include the more general case for context.
- Let $f : \Omega_f \to [0, 2^b - 1]$ be a $2D$, b-bit grayscale image.
- Let $s : \Omega_s \to \mathbb{Z}$ be a $2D$ *structuring element*.
- *Dilation* is denoted with the symbol $\oplus$ and is used to "enlarge objects" in an image

$$(f \oplus s)(x) = \max_{\substack{y \in \Omega_s \\ x-y \in \Omega_f}} \{f(x-y) + s(y)\} \tag{1}$$

- *Erosion* is denoted with the symbol $\ominus$ and is used to "shrink objects" in an image

$$(f \ominus s)(x) = \min_{\substack{y \in \Omega_s \\ x+y \in \Omega_f}} \{f(x+y) - s(y)\}. \tag{2}$$

- These definitions also apply to binary images, but we will take a simpler approach.

BACKGROUND: SET-THEORETICAL NOTATION

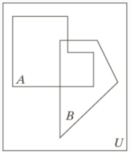
- A *set* is a collection of unique *elements*.
- If an element a is in a set A, we write it as $a \in A$ ("a in A").
- If an element a is not in a set A, we write it as $a \notin A$ ("a not in A").
- $\emptyset$ denotes a set with no elements in it, and it is called *the empty set*.
- We often use curly braces to denote a set: $A = \{a, b, c\}$.
- We use symbol : and to mean "where" or "such that":
  - $A = \{x : x \in B, x < c\}$ is the set of elements in B smaller than some constant c

- A^c is called the *complement* of A and contains all elements not in A.
- $A \setminus B$ is the set difference between A and B and contains all elements in A that are not in B, $A \setminus B = \{x \in A : x \notin B\}$.
- A is a *subset* of B if all elements of A is also in B, and this is denoted as $A \subseteq B$.
- B is then a *superset* of A, which we write $B \supseteq A$.
- The *union* of two sets A and B is the set of all elements in A *or* B (or both). We write this as $A \cup B$.
- The *intersection* of two sets A and B is the set of all elements in A *and* B. We write this as $A \cap B$.
- For an element $x \in \Omega$ and some set $A \subseteq \Omega$, let $x + A$ be the resulting set when we *shift* all elements in A by x

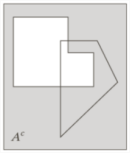
$$x + A = \{x + a : a \in A\}. \tag{3}$$

- Let $f : \Omega \to \{0, 1\}$ be an image.
- In the case of a 2D image, $\Omega \subset \mathbb{Z}^2$, and every pixel (x, y) in f is in the set Ω, written $(x, y) \in \Omega$.
- A binary image can be described by the set of foreground pixels, which is a subset of Ω.
- Therefore, we might use notation and terms from set theory when describing binary images, and operations acting on them.
- The *complement* of a binary image f is

$$h(x, y) = f^c(x, y) = \begin{cases} 0 & \text{if } f(x, y) = 1, \\ 1 & \text{if } f(x, y) = 0. \end{cases}$$



(a) Contours of sets A and B inside a set U



(b) The complement A^C (gray) of A (white)

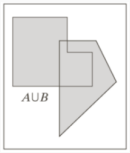
Figure 2: Set illustrations, gray is foreground, white is background.

- The *union* of two binary images f and g is

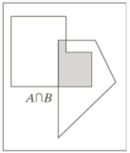
$$h(x,y) = (f \cup g)(x,y) = \begin{cases} 1 & \text{if } f(x,y) = 1 \text{ or } g(x,y) = 1, \\ 0 & \text{otherwise.} \end{cases}$$

- The *intersection* of two binary images f and g is

$$h(x,y) = (f \cap g)(x,y) = \begin{cases} 1 & \text{if } f(x,y) = 1 \text{ and } g(x,y) = 1, \\ 0 & \text{otherwise.} \end{cases}$$



(a) Union of A and B



(b) Intersection of A and B

Figure 3: Set illustrations, gray is foreground, white is background.

FUNDAMENTAL OPERATIONS

STRUCTURING ELEMENT

- A *structuring element* in morphology is used to determine the acting range of the operations.
- It is typically defined as a binary matrix where pixels valued 0 are not acting, and pixels valued 1 are acting.
- When hovering the structuring element over an image, we have three possible scenario for the structuring element (or really the location of the 1's in the structuring element):
  - It is *not overlapping* the image foreground (a *miss*).
  - It is *partly overlapping* the image foreground (a *hit*).
  - It is *fully overlapping* the image foreground (it *fits*).

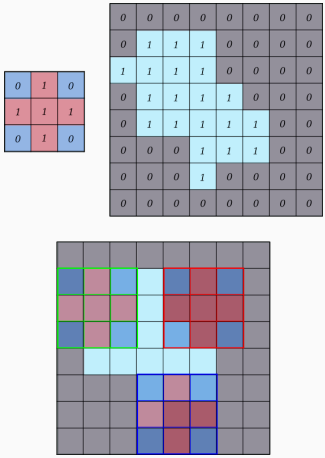


Figure 4: A structuring element (top left) and a binary image (top right). Bottom: the red misses, the blue hits, and the green fits.

ORIGIN OF STRUCTURING ELEMENT

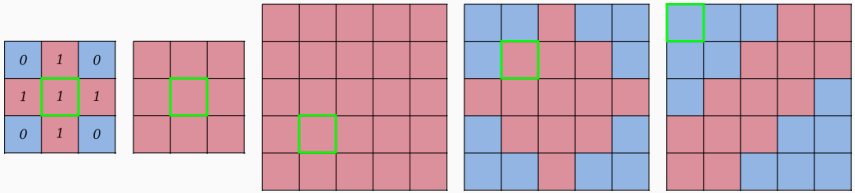


Figure 5: Some structuring elements. The red pixel contour highlights the origin.

- The structuring element can have different shapes and sizes.
- We need to determine an *origin*.
  - This origin denotes the pixel that (possibly) changes value in the result image.
  - The origin *may* lay outside the structuring element.
  - The origin should be highlighted, e.g. with a drawed square.
  - We will assume the origin to be at the center pixel of the structuring element, and not specify the location unless this is the case.

- Let $f : \Omega_f \to \{0, 1\}$ be a $2D$ binary image.
- Let $s : \Omega_s \to \{0, 1\}$ be a $2D$ binary structuring element.
- Let $x, y \in \mathbb{Z}^2$ be $2D$ points for notational convenience.
- We then have 3 equivalent definitions of erosion.
  - The one mentioned at the beginning of the lecture

$$(f \ominus s)(x) = \min_{\substack{y \in \Omega_s^+ \\ x+y \in \Omega_f}} \{f(x + y)\}, \tag{4}$$

    where Ω_s^+ is the subset of Ω_s with foreground pixels.
  - Place the s such that its origin overlaps with x, then

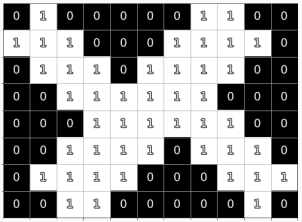
$$(f \ominus s)(x) = \begin{cases} 1 & \text{if } s \text{ fits in } f, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

  - Let $F(g)$ be the set of all foreground pixels of a binary image g, then

$$F(f \ominus s) = \{x \in \Omega_f : F(x + \Omega_s^+) \subseteq F(f)\}. \tag{6}$$

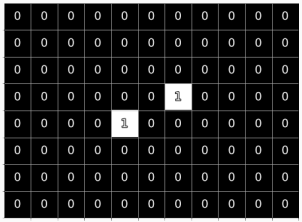
    Note that the $F()$ is often ommitted, as we often use set operations in binary morphology.

EROSION EXAMPLES

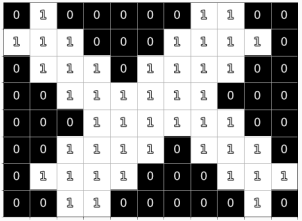


(a) f

(b) s_1

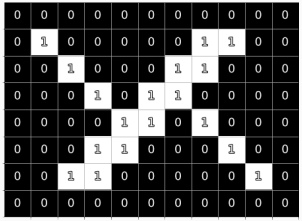


(c) $f \ominus s_1$



(a) f

(b) s_2

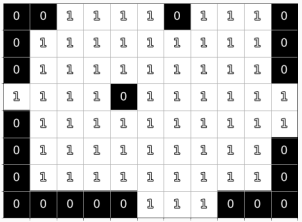


(c) $f \ominus s_2$

EFFECTS OF EROSION

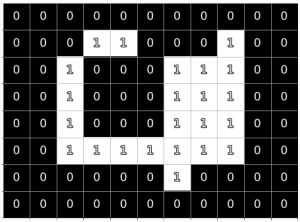
- Erosion *shrinks* foreground objects.
- Foreground holes are enlarged.
- Small (relative to the structuring element size) foreground ”peaks” are removed.
- Background ”fjords” are enlarged.
- The result is dependent on the structuring element.
- Larger structuring elements means more erosion.

EROSION EXAMPLES

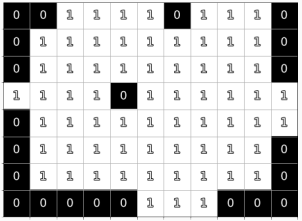


(a) f

(b) s_1

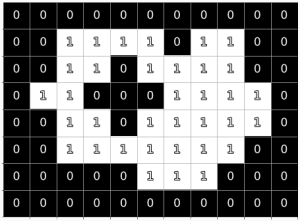


(c) $f \ominus s_1$



(a) f

(b) s_2



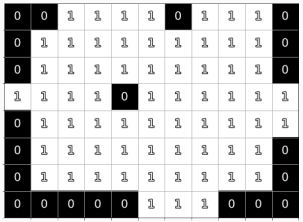
(c) $f \ominus s_2$

ITERATIVE EROSION

- Larger structuring elements erodes more.
- Repeated erosion with a small structuring element is similar to one erosion with a larger structuring element.
- If s_2 is similar to s_1 in shape, but twice as large, then

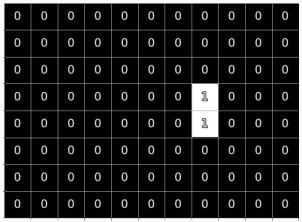
$$f \ominus s_2 \approx (f \ominus s_1) \ominus s_1 \tag{7}$$

ITERATIVE EROSION EXAMPLES

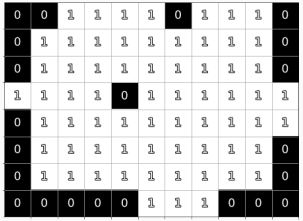


(a) f

(b) s_1

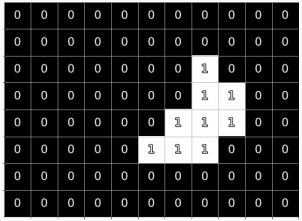


(c) $(f \ominus s_1) \ominus s_1$



(a) f

(b) s_2



(c) $(f \ominus s_2) \ominus s_2$

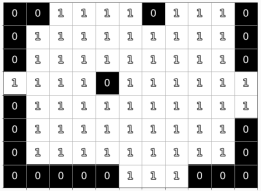
EDGE DETECTION WITH EROSION

- Erosion removes pixels along the boundary of the foreground object.
- We can locate the edges by subtracting the eroded image from the original

$$g = f - (f \ominus s)$$

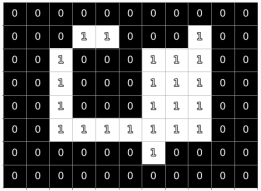
- The shape of the structuring element determines the connectivety of the countour. That is, if the contour is connected using a 4 or 8 connected neighbourhood.

EDGE DETECTION WITH EROSION: EXAMPLES

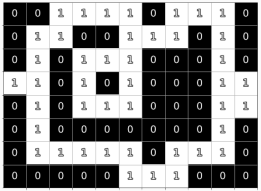


(a) f

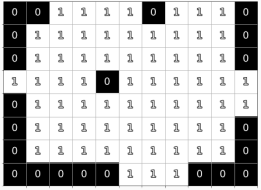
(b) s_1



(c) $f \ominus s_1$

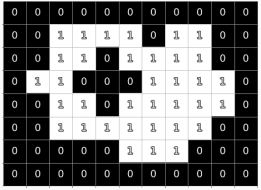


(d) $f - (f \ominus s_1)$

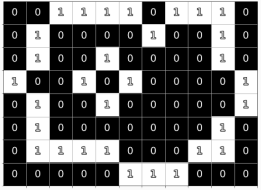


(a) f

(b) s_2

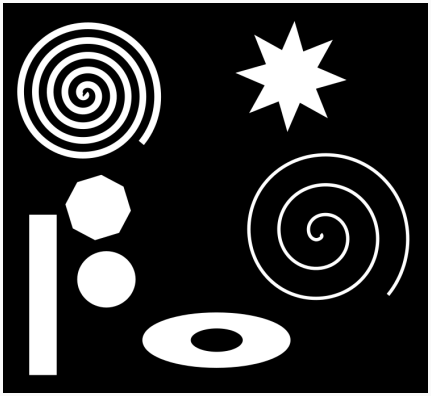


(c) $f \ominus s_2$



(d) $f - (f \ominus s_2)$

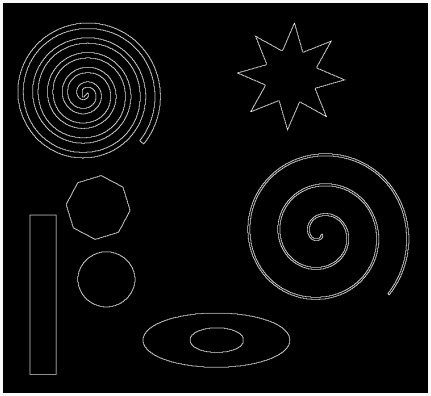
EDGE DETECTION WITH EROSION: EXAMPLES



(a) f

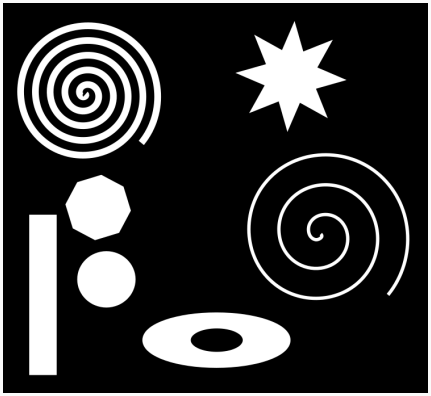
1	1	1
1	1	1
1	1	1

(b) s

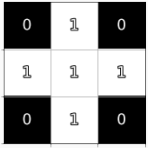


(c) $f - (f \ominus s)$

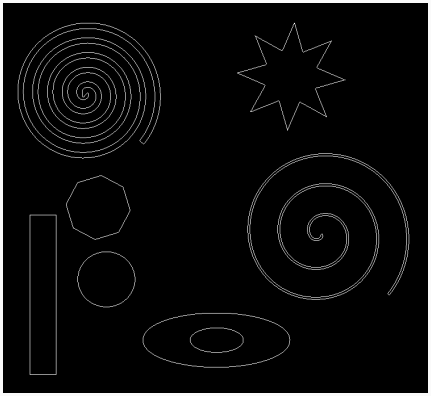
EDGE DETECTION WITH EROSION: EXAMPLES



(a) f



(b) s



(c) $f - (f \ominus s)$

- Let $f : \Omega_f \to \{0, 1\}$ be a $2D$ binary image.
- Let $s : \Omega_s \to \{0, 1\}$ be a $2D$ binary structuring element.
- Let $x, y \in \mathbb{Z}^2$ be $2D$ points for notational convenience.
- We then have 3 equivalent definitions of dilation.
  - The one mentioned at the beginning of the lecture

$$(f \oplus s)(x) = \max_{\substack{y \in \Omega_s^+ \\ x - y \in \tilde{\Omega}_f}} \{f(x - y)\}, \tag{8}$$

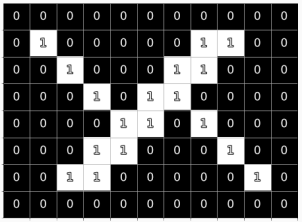
    where Ω_s^+ is the subset of Ω_s with foreground pixels.
  - Place the s such that its origin overlaps with x, then

$$(f \oplus s)(x) = \begin{cases} 1 & \text{if } \tilde{s} \text{ hits } f, \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

    where $\tilde{s}$ is s rotated 180 degrees.
  - Let $F(g)$ be the set of all foreground pixels of a binary image g, then

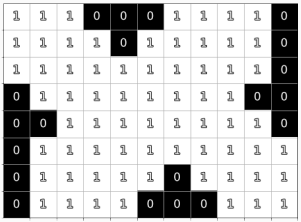
$$F(f \oplus s) = \{x \in \Omega_f : F(x + \Omega_s^+) \cap F(f) \neq \emptyset\} \tag{10}$$

DILATION EXAMPLES

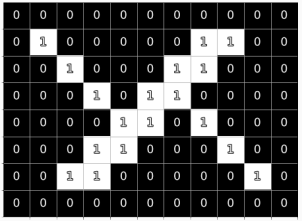


(a) f

(b) s_1

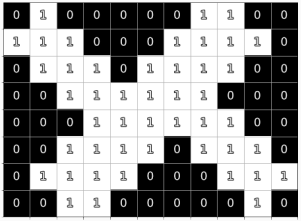


(c) $f \oplus s_1$



(a) f

(b) s_2

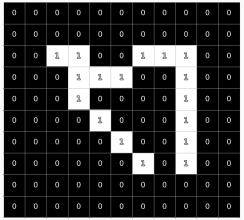


(c) $f \oplus s_2$

EFFECTS OF DILATION

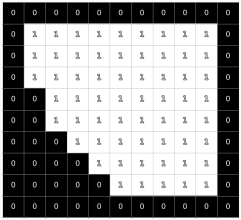
- Dilation *enlarges* foreground objects.
- Dilation fills holes in the foreground (if the structuring element is large enough).
- Dilation smooths background ”fjords” in the foreground.
- The result is dependent on the structuring element.
- Larger structuring element gives larger dilation effect.

DILATION EXAMPLES

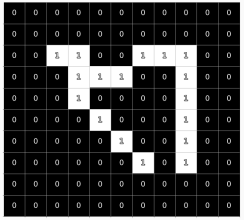


(a) f

(b) s_1

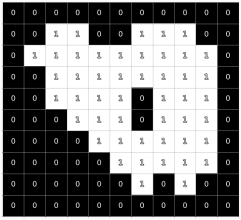


(c) $f \oplus s_1$



(a) f

(b) s_2



(c) $f \oplus s_2$

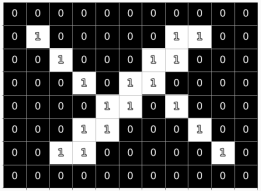
EDGE DETECTION WITH DILATION

- Dilation adds pixels along the boundary of the foreground object.
- We can locate the edges by subtracting the original image from the dilated image

$$g = (f \oplus s) - f$$

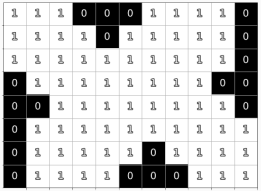
- The shape of the structuring element determines the connectivety of the countour. That is, if the contour is connected using a 4 or 8 connected neighbourhood.

EDGE DETECTION WITH DILATION: EXAMPLES

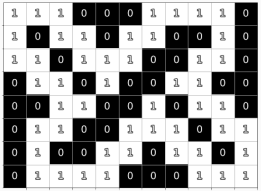


(a) f

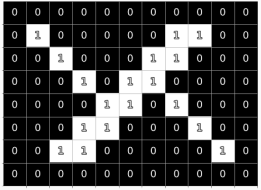
(b) s_1



(c) $f \oplus s_1$

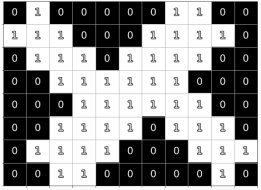


(d) $(f \ominus s_1) - f$

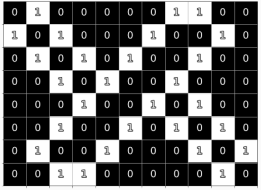


(a) f

(b) s_2

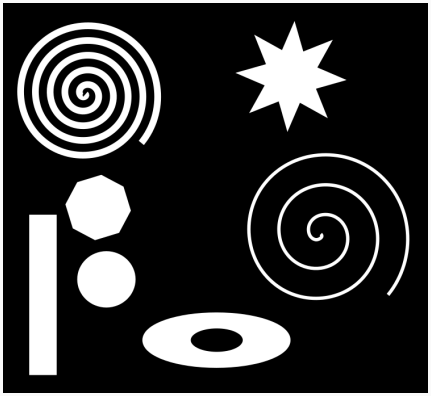


(c) $f \oplus s_2$



(d) $(f \oplus s_2) - f$

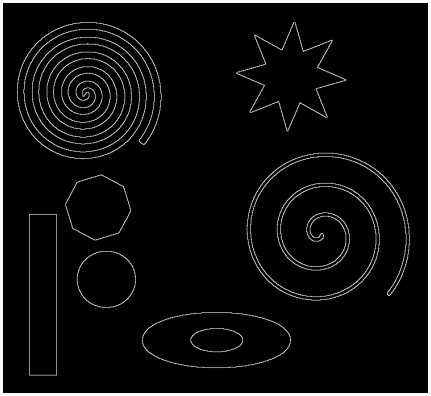
EDGE DETECTION WITH DILATION: EXAMPLES



(a) f

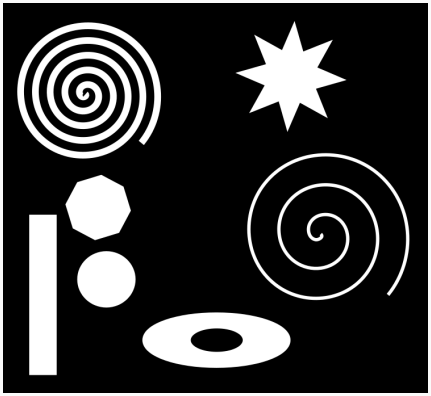
1	1	1
1	1	1
1	1	1

(b) s

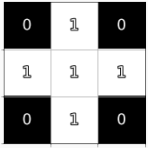


(c) $(f \oplus s) - f$

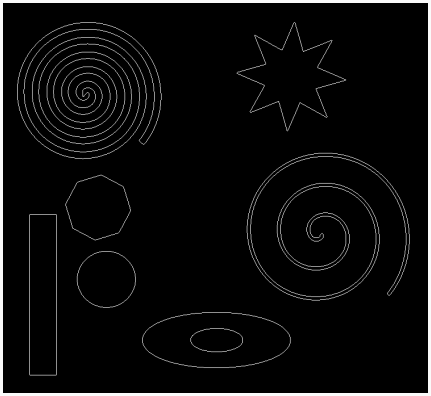
EDGE DETECTION WITH DILATION: EXAMPLES



(a) f



(b) s



(c) $(f \oplus s) - f$

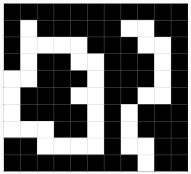
REGION FILLING

- Region filling is a simple application of dilation.
- We can use it to extract connected foreground componets.
- Conversly, it can be used to extract holes in foreground regions. This can then be used to fill said holes.

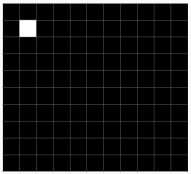
EXTRACTING CONNECTED FOREGROUND COMPONENTS

- Assume you have a binary image f and a binary, square structuring element s.
- To extract all connected components (with 8-connectivity)
  - Select a 2D point x on the structure.
  - Let c_0 be the same shape as f, and initialize it with 0 in every element, except at x, where it gets value 1.
  - Then, let
$$c_k = (c_{k-1} \oplus s) \cap f, \quad k = 1, \ldots, K$$
  - Continue until no change is observed, that is, when $c_k = c_{k-1}$.

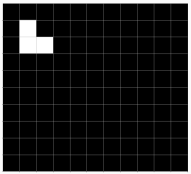
CONNECTED COMPONENT EXAMPLE



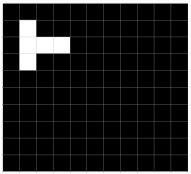
(a) f



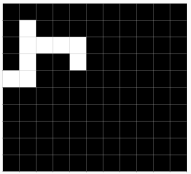
(b) c_0



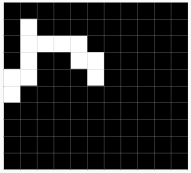
(c) c_1



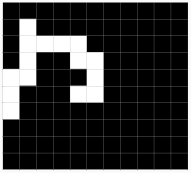
(d) c_2



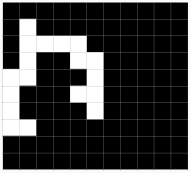
(e) c_3



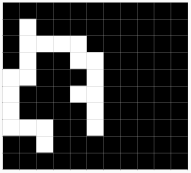
(f) c_4



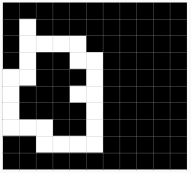
(g) c_5



(h) c_6

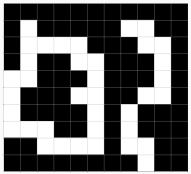


(i) c_7

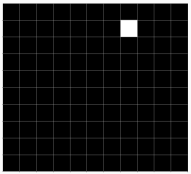


(j) c_8

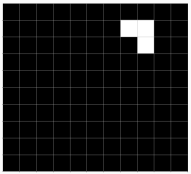
CONNECTED COMPONENT EXAMPLE



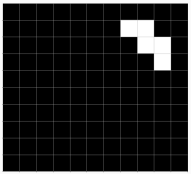
(a) f



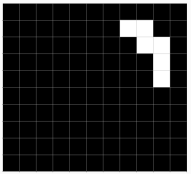
(b) c_0



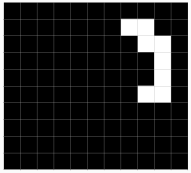
(c) c_1



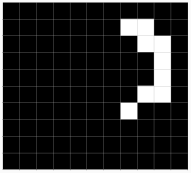
(d) c_2



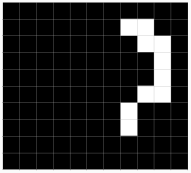
(e) c_3



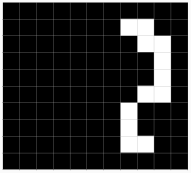
(f) c_4



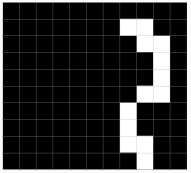
(g) c_5



(h) c_6



(i) c_7

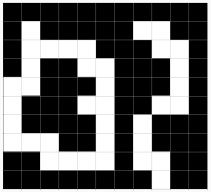


(j) c_8

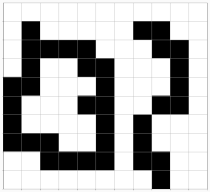
EXTRACTING HOLES IN FOREGORUND REGIONS

- This is almost identical to the connected component method.
- Assume you have a binary image f and a binary, square structuring element s.
- To extract all elements inside a foreground hole in f (with 4-connectivity)
  - Select a 2D point x inside the hole.
  - Let c_0 be the same shape as f, and initialize it with 0 in every element, except at x, where it gets value 1.
  - Then, let
$$c_k = (c_{k-1} \oplus s) \cap f^c, \quad k = 1, \ldots, K$$
  - Continue until no change is observed, that is, $c_k = c_{k-1}$.
- Note that we used the complement of f here, f^c.

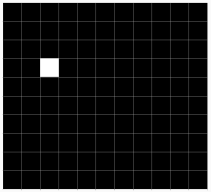
CONNECTED COMPONENT EXAMPLE



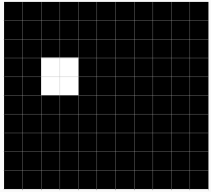
(a) f



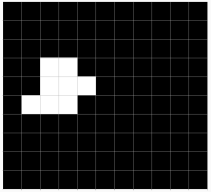
(b) f^c



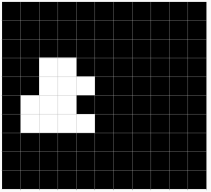
(c) c_0



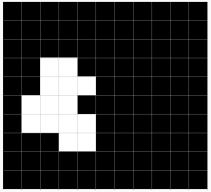
(d) c_1



(e) c_2



(f) c_3



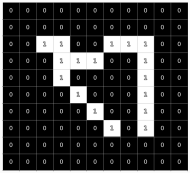
(g) c_4

DUALITY BETWEEN DILATION AND EROSION

- Dilation and erosion are *dual* operations w.r.t. complements and reflections (180 degree rotation). That is, erosion and dilation can be expressed as

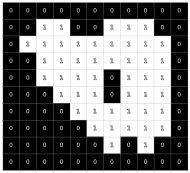
$$f \oplus s = (f^c \ominus s)^c$$
$$f \ominus s = (f^c \oplus s)^c$$

- This means that dilation and erosion can be performed by the same procedure, given that we can rotate the structuring element and find the complement of the binary image.

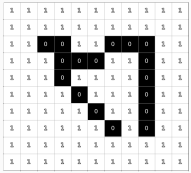


(a) f

(b) s

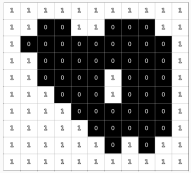


(c) $f \oplus s$



(a) f^c

(b) s



(c) $f^c \ominus s$

- Dilation is commutative

$$f \oplus s = s \oplus f.$$

  Even if it is convention to place the image as the left operand and the structuring element as the right operand, this is not necessary.

- Dilation is associative

$$f \oplus (s_1 \oplus s_2) = (f \oplus s_1) \oplus s_2$$

  If $s = s_1 \oplus s_2$ we can utilize this separability for computational benefit. This is especially true if s_1 and s_2 are one-dimensional. Example:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix} \oplus \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

SOME PROPERTIES OF EROSION

- Erosion is *not* commutative

$$f \ominus s \neq s \ominus f.$$

- Erosion is *not* associative, but successive erosion of the image f with s_1 and s_2 is equal to the erosion of f with s_1 dilated with s_2

$$(f \ominus s_1) \ominus s_2 = f \ominus (s_1 \oplus s_2).$$

CIRCULAR STRUCTURING ELEMENTS ON CORNERS

- Both dilation and erosion with *rectangular* structuring elements preserves the shape of corners.
- Dilation of *concave* corners with circular structuring elements *preserves* the shape of the corners.
- Dilation of *convex* corners with circular structuring elements *rounds* the shape of the corners.
- The opposite is true for erosion
  - Erosion of *concave* corners give rounded corners.
  - Erosion of *convex* corners preserves the shape.

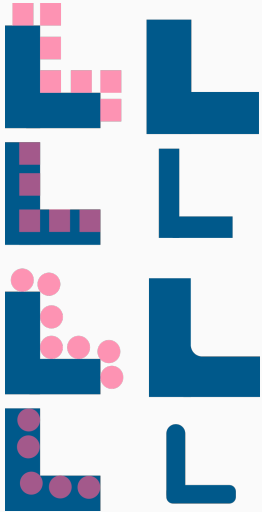


Figure 29

MORE ON DUALITY

- Erosion can be thought of finding the positions where the structuring element fits the foreground.
- Dilation can be thought of finding the positions where the (rotated) structure elemnt fits the background.
- This duality implies the behaviour on corners discussed in the previous slide.
- Since erosion of concave corners with a circular structuring element rounds the corners, dilation of convex corners will round the corners when using the same circular structure element.
- Note that a concave foreground corner is a convex background corner, and vice versa.

COMPOSITE OPERATIONS

MORPHOLOGICAL OPENING

- Erosion of an image removes all regions that cannot fit the structuring element, and shrinks all other regions.
- We can then dilate the result of the erosion, with this
  - the regions that where shrinked are (approximately) restored.
  - the regions too small to survive an erosion, are not restored.
- This is *morphological opening*

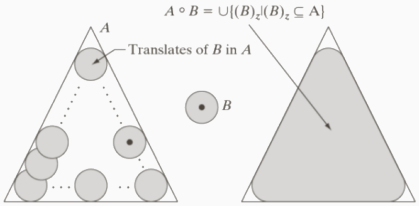
$$f \circ s = (f \ominus s) \oplus s$$

- The name stems from that this operation can provide an opening (a space) between regions that are connected through thin ”bridges”, almost without affecting the original shape of the larger regions.
- Using erosion only will also open these bridges, but the shape of the larger regions is also altered.
- The size and shape of the structuring element is vital.

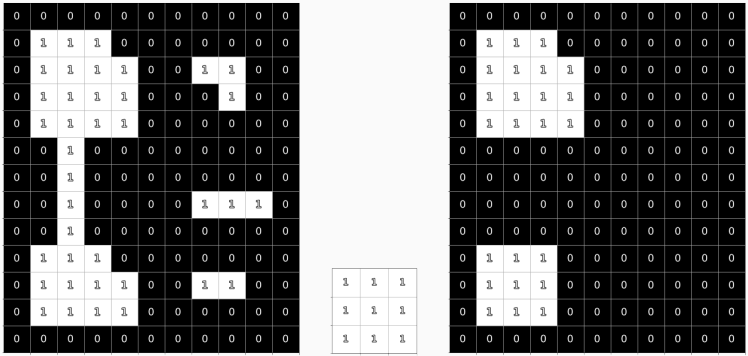
GEOMETRICAL INTERPRETATION OF OPENING

- Imagine translating the structuring element over the image such that it always fits inside the foreground regions.
- Do this wherever possible
- An opening is then the regions covered by the structuring element when doing the above operation.
- The opening operator is *idempotent*

$$(f \circ s) \circ s = f \circ s$$



EXAMPLE: OPENING



(a) f (b) s (c) $f \circ s$

Figure 30: Morphological opening of f with s.

MORPHOLOGICAL CLOSING

- Dilation of an image expands foreground regions and fills small (relative to the structuring element) holes in the foreground.
- We can then erode the result of the dilation, then
  - the regions that where expanded are (approximately) restored.
  - the holes that where filled, are not opened again.
- This is *morphological closing*

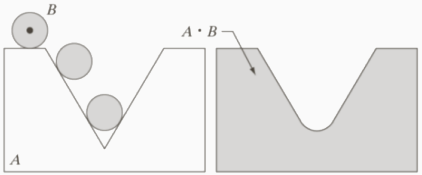
$$f \bullet s = (f \oplus s) \ominus s$$

- The name stems from that this operation can close small gaps between foreground regions, without altering the larger foreground shapes too much.
- Using dilation only will also close these gaps, but the shape of the larger regions is also altered.
- The size and shape of the structuring element is vital.

GEOMETRICAL INTERPRETATION OF CLOSING

- We can use the same image as for morphological opening.
- Imagine translating the structuring element over the image such that it always is outside foreground regions.
- Do this wherever possible
- A closing is then the regions *not covered* by the structuring element when doing the above operation.
- The opening operator is also *idempotent*

$$(f \bullet s) \bullet s = f \bullet s$$



EXAMPLE: CLOSING

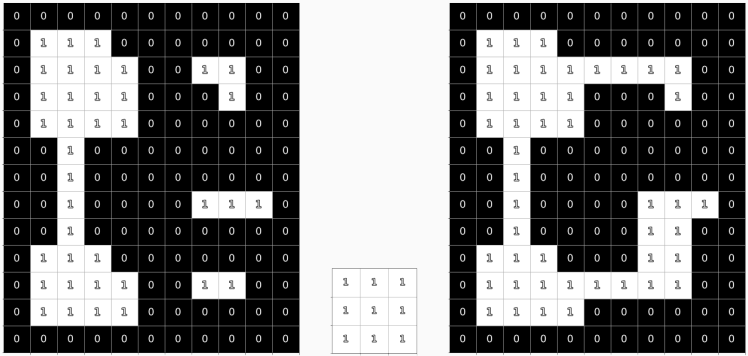
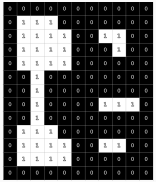


Figure 31: Morphological opening of f with s.

- Opening and closing are *dual operations* w.r.t. complements and rotation

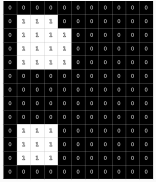
$$f \circ s = (f^c \bullet s)^c$$
$$f \bullet s = (f^c \circ s)^c$$

- This means that closing can be performed by complementing the image, opening the complement by the rotated structuring element, and complement the result. The corresponding is true for opening.

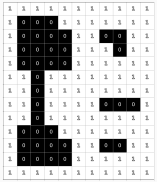


(a) f

(b) s

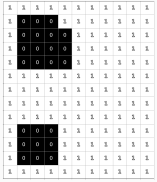


(c) $f \circ s$



(a) f^c

(b) s



(c) $f^c \bullet s$

- Dilation and closing are *extending operations*, meaning that foreground pixels are added to the image.
- Erosion and opening are *narrowing operations*, meaning that foreground pixels are removed.
- For a binary image f and a binary structuring element s, we have that

$$(f \ominus s)(x) \leq (f \circ s)(x) \leq f(x) \leq (f \bullet s)(x) \leq (f \oplus s)(x)$$

- On a similar note, if $F(g)$ is the set of foreground pixels in g,

$$F(f \ominus s) \subseteq F(f \circ s) \subseteq F(f) \subseteq F(f \bullet s) \subseteq F(f \oplus s)$$

EXAMPLE: NOISE REMOVAL

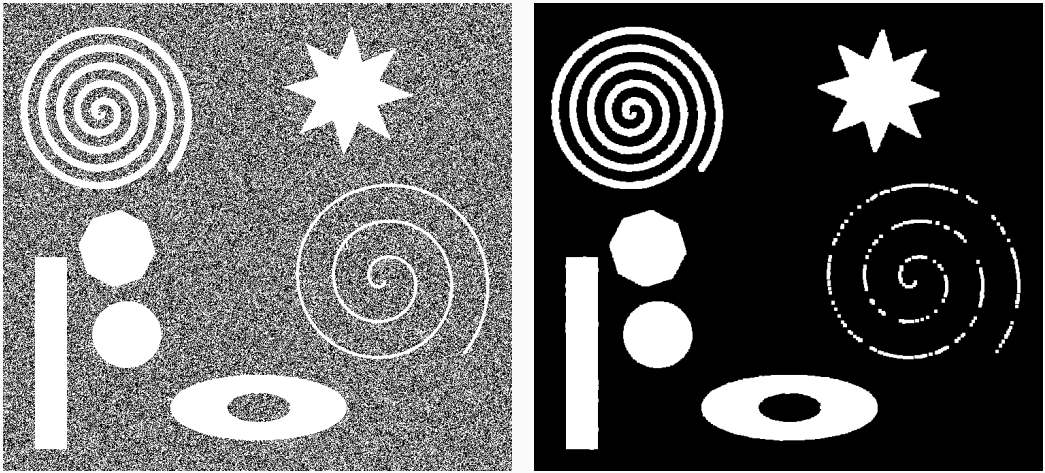


Figure 34: Opening with a 5×5 square structuring element

EXAMPLE: FORM SEPARATION

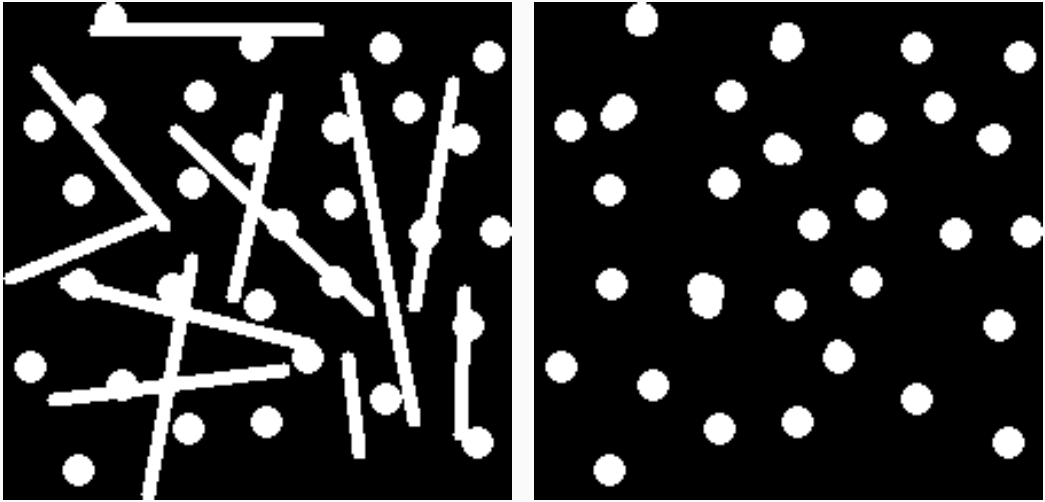


Figure 35: Opening with a circular structuring element.

EXAMPLE: FILTERING BY CLOSING

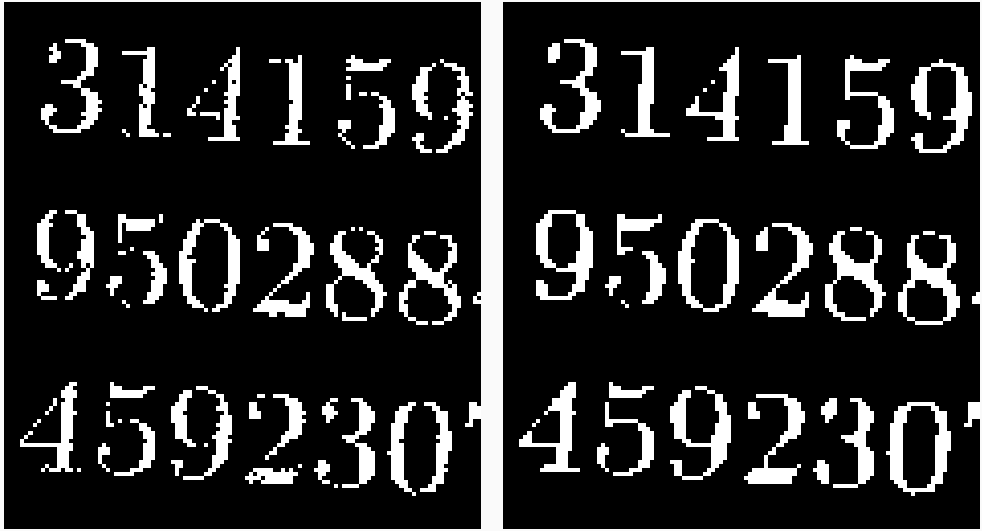


Figure 36: Closing with a 3×3 quadratic structuring element

EXAMPLE: FILTERING BY OPENING AND CLOSING

Figure 37: *G&W Figure 9.11.* Closing the opening of an image to remove noise.

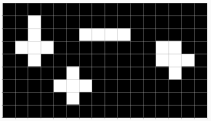
HIT-OR-MISS TRANSFORMATION

- Let f be a binary image as usual, but define s to be a tuple of two structuring elements $s = (s^{hit}, s^{miss})$.
- The hit-or-miss transformation is then defined as

$$f \circledast s = (f \ominus s^{hit}) \cap (f^c \ominus s^{miss}),$$

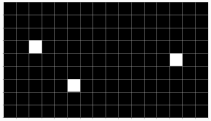
- A foreground pixel in the out-image is only achieved if
  - s^{hit} fits the foreground around the pixel, and
  - s^{miss} fits the background around the pixel.
- This can be used in several applications, including
  - finding certain patterns in an image,
  - removing single pixels,
  - thinning and thickening foreground regions.

HIT-OR-MISS EXAMPLE

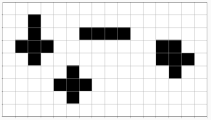


(a) f

(b) s^{hit}

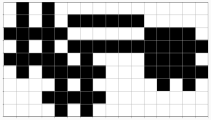


(c) $f \ominus s^{hit}$



(a) f^c

(b) s^{miss}



(c) $f^c \ominus s^{miss}$

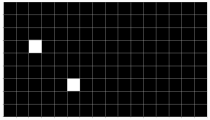


Figure 40: $f \circledast s$

MORPHOLOGICAL THINNING

- Morphological thinning of an image f with a structuring element tuple s, is defined as

$$\begin{aligned} f \otimes s &= f \setminus (f \circledast s) \\ &= f \cap (f \circledast s)^c \end{aligned}$$

- In order to thin a foreground region, we perform a sequential thinning with multiple structuring elements $s_1, \ldots, s_8$, which, when defined with the hit-or-miss notation above are

$$s_1^{hit} = \begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 1 & 1 & 1 \\ \hline \end{array}, \quad s_1^{miss} = \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 0 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array}.$$

$$s_2^{hit} = \begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 1 & 1 & 0 \\ \hline 1 & 1 & 0 \\ \hline \end{array}, \quad s_2^{miss} = \begin{array}{|c|c|c|}\hline 0 & 1 & 1 \\ \hline 0 & 0 & 1 \\ \hline 0 & 0 & 0 \\ \hline \end{array}.$$

Following this pattern, where s_{i+1} is rotated clockwise w.r.t. s_i, we continue this until

$$s_8^{hit} = \begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 1 & 1 \\ \hline 0 & 1 & 1 \\ \hline \end{array}, \quad s_8^{miss} = \begin{array}{|c|c|c|}\hline 1 & 1 & 0 \\ \hline 1 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array}.$$

- With these previously defined structuring elements, we apply the iteration

$$\begin{aligned} d_0 &= f \\ d_k &= d_{k-1} \otimes \{s_1, \cdots, s_8\} \\ &= (\cdots((d_{k-1} \otimes s_1) \otimes s_2) \cdots) \otimes s_8 \end{aligned}$$

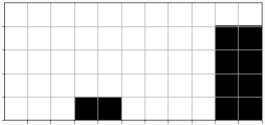
  for K iterations until $d_K = d_{K-1}$ and we terminate with d_K as the result of the thinning.

- In the same manner, we define the dual operator *thickening*

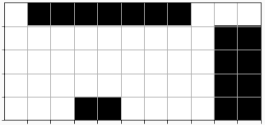
$$f \odot s = f \cup (f \circledast s),$$

  which also can be used in a sequential manner analogous to thinning.

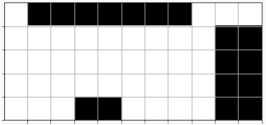
THINNING EXAMPLE



(a) $d_0 = f$

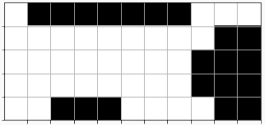


(b) $d_{01} = d_0 \setminus (d_0 \circledast s_1)$

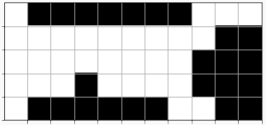


(c) $d_{02} = d_{01} \setminus (d_{01} \circledast s_2)$

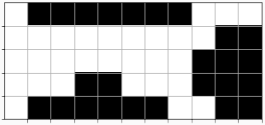
(d) $d_{03} = d_{02} \setminus (d_{02} \circledast s_3)$



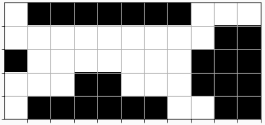
(e) $d_{04} = d_{03} \setminus (d_{03} \circledast s_4)$



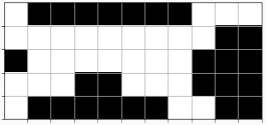
(f) $d_{05} = d_{04} \setminus (d_{04} \circledast s_5)$



(g) $d_{06} = d_{05} \setminus (d_{05} \circledast s_6)$

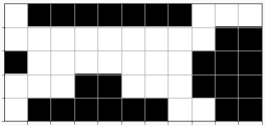


(h) $d_{07} = d_{06} \setminus (d_{06} \circledast s_7)$

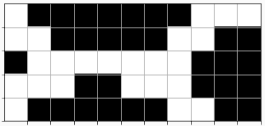


(i) $d_{08} = d_{07} \setminus (d_{07} \circledast s_8)$

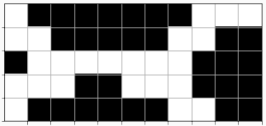
THINNING EXAMPLE



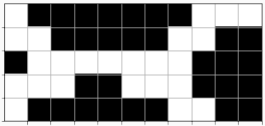
(a) $d_1 = d_0$



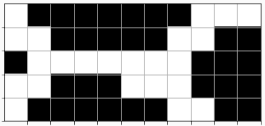
(b) $d_{11} = d_1 \setminus (d_1 \circledast s_1)$



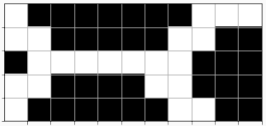
(c) $d_{12} = d_{11} \setminus (d_{11} \circledast s_2)$



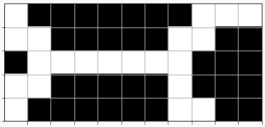
(d) $d_{13} = d_{12} \setminus (d_{12} \circledast s_3)$



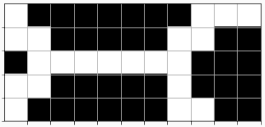
(e) $d_{14} = d_{13} \setminus (d_{13} \circledast s_4)$



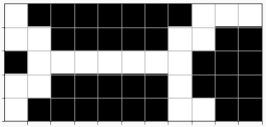
(f) $d_{15} = d_{14} \setminus (d_{14} \circledast s_5)$



(g) $d_{16} = d_{15} \setminus (d_{15} \circledast s_6)$

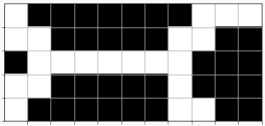


(h) $d_{17} = d_{16} \setminus (d_{16} \circledast s_7)$

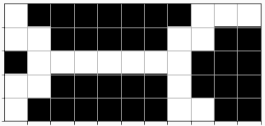


(i) $d_{18} = d_{17} \setminus (d_{17} \circledast s_8)$

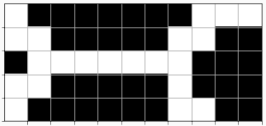
THINNING EXAMPLE



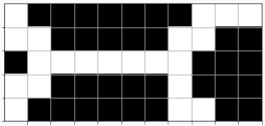
(a) $d_2 = d_1$



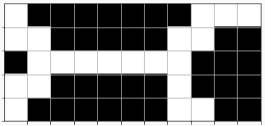
(b) $d_{21} = d_2 \setminus (d_2 \circledast s_1)$



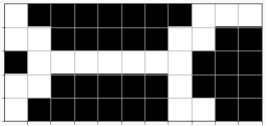
(c) $d_{22} = d_{21} \setminus (d_{21} \circledast s_2)$



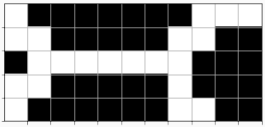
(d) $d_{23} = d_{22} \setminus (d_{22} \circledast s_3)$



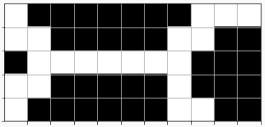
(e) $d_{24} = d_{23} \setminus (d_{23} \circledast s_4)$



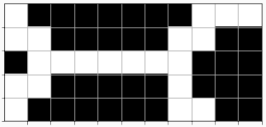
(f) $d_{25} = d_{24} \setminus (d_{24} \circledast s_5)$



(g) $d_{26} = d_{25} \setminus (d_{25} \circledast s_6)$



(h) $d_{27} = d_{26} \setminus (d_{26} \circledast s_7)$



(i) $d_{28} = d_{27} \setminus (d_{27} \circledast s_8)$

SUMMARY

- Binary morphology
- Structuring element
- Duality
- Operations
  - Erosion
  - Dilation
  - Opening
  - Closing
  - Hit-or-miss
  - Thinning
- Some applications

Questions?