Mathematical Morphology
first part: 2D
Mathematical Morphology was developed in France (G. Motheron e J. Serra, Ecole des Mines) and in different form with the name Image Algebra in USA (S. R. Sternberg, Michigan University).
Preliminary Statements

A \subseteq \mathbb{E}^n, t \in \mathbb{E}^n

- Translation of A by a vector t
  \[ A_t = \{ c \in \mathbb{E}^n \mid c = a + t, a \in A \} \]

- Reflection of A
  \[ A_r = \{ c \mid c = -a, a \in A \} \]

- Complement of A
  \[ A_c = \mathbb{E}^n - A \]
Minkowski sum (Dilation)

\[ A \oplus B = \{ c \in \mathbb{E}^n \mid c = a + b, \ a \in A, \ b \in B \} \]

\[ A \oplus B = \bigcup A_b, \ b \in B \]

It can be easily shown that: \( A \oplus B = B \oplus A \)

\[ A \ (0,0) \quad A \ (1,0) \quad A \oplus B \]

\[ B = \{ (0,0), (1,0) \} \]
Dilation

- B is usually called **structuring element**

B = \{(-1,0), (1,0) \}

A \oplus B

A_{(-1,0)}  A_{(1,0)}  A \oplus B
Dilation

Structuring element:

\[
\begin{align*}
\hat{B} &= B \\
&= \begin{cases}
\text{d/4} & \text{d/4} \\
\hat{B} &= B
\end{cases}
\end{align*}
\]

Structuring element:

\[
\begin{align*}
\hat{B} &= B \\
&= \begin{cases}
\text{d/4} & \text{d/4} \\
\hat{B} &= B
\end{cases}
\end{align*}
\]
Dilation

Structuring element:

\[ A \oplus B \]
Dilation

Structuring element:

\[ A \oplus B \]
Dilation

Structuring element:

A

B

A ⊕ B
Dilation

Structuring element:

\[ A \oplus B \]
Dilation

Structuring element 1:

Structuring element 2:
Dilation

A

B

C

A \oplus B \oplus C
Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.
Minkowski difference (Erosion)

- \( A \ominus B = \{ c \in \mathbb{E}^n \mid c + b \in A, \text{ per ogni } b \in B \} \)
- \( A \ominus B = \cap A_{-b} \ b \in B \)
- \( A \ominus B = \{ c \in \mathbb{E}^n \mid B_c \subset A \} \)

Structuring element:

\[ A \rightarrow B = \{ (0,0), (1,0) \} \]
**Erosion**

Structuring element:

\[ A \ominus B \]
Erosion

Structuring element 1:

Structuring element 2:
Erosion

Structuring element:

\[ A \ominus B \]
Erosion

A

Structuring element:

B

A ⊖ B
Erosion

A

Structuring element:

B

A ∩ B
**Erosion**

Original image  
Eroded once  
Eroded twice  

Structuring element: .
Dilation (+) and Erosion (-) properties

\[ A + \{\emptyset\} = A - \{\emptyset\} = A \]
\[ A + B = (A^c - B^r)^c \]
\[ A - B = (A^c + B^r)^c \]
\[ (A+B)^c = A^c - B^r \]
\[ A+B_t = (A+B)_t \]
\[ A-B_t = (A-B)_t \]
\[ A + \{a\} = A - \{a\}^r = A_a, \text{ Translation} \]

Decomposition:
\[ B = B_1 + B_2 + B_3 + \ldots + B_n \]
\[ A + B = (\ldots(((A + B_1) + B_2) + B_3) + \ldots) + B_n \]
\[ A - B = (\ldots(((A - B_1) - B_2) - B_3) - \ldots) - B_n \]

Erosion and Dilation Duality Theorem: Dilation and Erosion transformation bear a similarity, what one does to image foreground and the other does for the image background.

Similar but not identical to De Morgan rule in Boolean Algebra
\( (A+B)+C=A+(B+C) \) \( \) \( (A-B)-C=A-(B+C) \)

\( (A\cup B)+C=(A+C)\cup(B+C) \) \( (A \cap B)-C=(A-C)\cap(B-C) \)

\( A+(B\cup C)=(A+B)\cup(A+C) \) \( A-(B\cup C)=(A-C)\cap(B-C) \)

\( A \subseteq B \implies (A+C) \subseteq (B+C) \) \( A \subseteq B \implies (A-C) \subseteq (B-C) \)

\( B \subseteq C \implies (A-B) \supseteq (A-C) \)

\( (A\cap B)+C \subseteq (A+C)\cap(B+C) \) \( (A\cup B)-C \supseteq (A-C)\cup(B-C) \)

\( A-(B\cap C) \supseteq (A-C)\cup(B-C) \)
Erosion and Dilation summary

FIGURE 9.7  (a) Image of squares of size 1, 3, 5, 7, 9, and 15 pixels on the side. (b) Erosion of (a) with a square structuring element of 1’s, 13 pixels on the side. (c) Dilation of (b) with the same structuring element.

Structuring element: □
Closing operator

- \( C(A, K) = (A \oplus K) \ominus K \)
- Operator idempotent (the reapplication has not further effects): \( A \subseteq C(A, K) = C(C(A, K), K) \)

![Diagram showing the effect of the closing operator](image)

Structuring element: \( K \)

- \( A \)
- \( A \oplus K \)
- \( (A \oplus K) \ominus K \)
**Closing operator**

\[ C(A, K) = (A \oplus K) \ominus K \]

Structuring element:

![Structuring element](image-url)
Closing

Structuring element:

A

B
Closing operator

\[ C(A, K) = (A \oplus K) \Theta K \]

Structuring element:

\[ A \oplus K \]

\[ (A \oplus K) \Theta K \]
Opening operator

- $O(A, K) = (A \Theta K) \oplus K$
- Operator idempotent (the reapplication has not further effects): $O(O(A,K), K) = O(A, K) \subseteq A$

![Diagram of opening operator with examples of structuring elements]
Opening operator

\[ O(A, K) = (A \ominus K) \oplus K \]

Structuring element:

\[ \begin{array}{c}
A \\
A \ominus K \\
(A \ominus K) \oplus K
\end{array} \]
Opening

Structuring element:

Translates of $B$ in $A$
Opening

- Erode, then dilate
- Remove small objects, keep original shape

Structuring element: ●

Before opening
Opening

- Erode, then dilate
- Fill holes, but keep original shape

Structuring element: •

Before Opening  After Opening
Opening Example

- 3x9 and 9x3 Structuring Element
Opening a picture is describable as pushing object B under the scan-line graph, while traversing the graph according the curvature of B. The valleys usually remain in their original form.

Closing a picture is describable as pushing object B on top of the scan-line graph, while traversing the graph according the curvature of B. The peaks usually remain in their original form.
The ‘good’ contour

- Opening and Closing operator with a circle as structural elements change the boundaries as shown in figure: closing extends the boundary as if a ball rolls over the outer border; opening restricts it rolling the inner border.
- The larger the circle the smoothed the result. The maximum resulting curvature is that of the structural element.
Opening vs Closing

Structuring element:

A
Hit or Miss operator

- \( A \otimes (J,K) = (A \ominus J) \cap (A^c \ominus K) \)
- \( \text{con il vincolo } J \cap K = \emptyset \)
- Suitable for ‘template’ matching

Two structuring elements J and K
J and K can be seen as a single template with three values:

- Foreground points
- Background points
- Do not care points
**Hit or Miss**

- Search of isolated points (8-connection)
- $A \ominus J = A$

**A**

**$A^c$**

**Final Result**
Hit or Miss

Search of isolated points (4-connection)

$A \Theta J = A$

$A^c \Theta K$

Final Result
Hit or Miss

Pixels satisfying the background constraints

Pixels satisfying the foreground constraints
Contour example

A - (A ⊕ B)
Examples: Boundary Extraction

- Contour
  - Internal: \( A - (A \Theta K) \)
  - External: \( (A \oplus K) \cap \bar{A} \) or \( (A \oplus K) - A \)
  - Double: \( (A \oplus K) \cap (A \Theta K) = (A \oplus K) - (A \Theta K) \)
Iteration: disks in 4 and 8 connectivity

Structuring element:

\[ X = \{ C \}; I=1 \]

for \( i=1, R \) do \( X = (X \oplus K) \)

C = center pixel

X = evolving image

R = radius (4 in ex.)
Recursion: Propagation

Propagation in a connected component

Let A be a set containing one or more connected components (mask), and consider an array $X_0$ (of the same size of the array A) whose elements are 0s, except to a point of A foreground (marker).

- $X = \{ X_0 \}$; $X = \text{evolving image}$
- do $D = X$
  - $X = (X \oplus K) \cap A$ $K$ is the unitary circle
  - while($D \neq X$) in the adopted metric

- $A = \text{original image}$

\[ \text{while}(D \neq X) \]
Recursion example: Connected Components

Structuring element: $B$

$A$, $X_0$, $X_1$, $X_2$, $X_3$, $X_6$
Distance transform

- DT implementation using dilation and addition operators:

- \[ R = \emptyset \]  
- \[ R = \text{evolving image} \]  
- \[ \text{while} (A \neq \emptyset) \text{ do} \]
- \[ R = R + A \]
- \[ A = A \ominus K \]
- done

At the end DT
Distance Tranform - algorithm

Structuring element:

```
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 2 2 2 2
1 2 2 2 2
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 2 2 3 3
1 1 1 1 1
```

```
1 1 1 1 1
1 2 2 3 3
1 2 2 3 3
1 1 1 1 1
```

```
1 1 1 1 1
1 2 2 3 3
1 2 2 3 3
1 1 1 1 1
```

```
1 1 1 1 1
1 2 2 3 3
1 2 2 3 3
1 1 1 1 1
```

```
1 1 1 1 1
1 2 2 2 2
1 1 1 1 1
```

```
1 1 1 1 1
1 2 2 2 2
1 1 1 1 1
```

```
1 1 1 1 1
1 2 2 2 2
1 1 1 1 1
```

54
The local maxima set is a compact object representation

The object can be rebuilt as union of the maximal digital disks
Distance transform and MAT

- The Distance Transform (DT) is obtained by labeling all the pixels inside a binary object with their distance to the background.
- In figure applying twenty iterations of the erosion operator (structural element: unit disk) twenty successive colored layers showing equi-distant contours from the background for a Manhattan distance metric are obtained.
- Every pixel has a color corresponding to its distance label which increases going inwards. In practice, this value represents the side of the greatest digital disk having its centre on this pixel, which is completely contained in the binary object.
- Any pattern can be interpreted as the union of all its maximal digital disks (local maximum in DT). A maximal disk is a disk contained in the object that is not completely overlapped by any other disk.
- The set of the centers of the maximal disks with their labels, constitutes the MAT.
Reverting progressively MAT

- A procedure to derive the MAT from the DT is based on the comparison of neighboring labels to establish whether a local maximum exists.
- **This transform is complete** in the sense that it is possible to revert it, so obtaining the original object back.
- This recovery process can be implemented by expanding every pixel belonging to the MAT, using the corresponding maximal disc whose size is given by the pixel label. The logical union of all such discs reconstructs the original object.
- This figure shows the progressive reconstruction, starting from the set of disks corresponding to the highest level (two white disks) until the sixth and last monk’s profile, where discs, reduced to just one pixel, have been included.
- **This transform is compact** since the full object may be described only by its labeled disk centers.

\[ U = \{ \emptyset \} \]
\[ \forall i, j : \text{MAT}_{i,j} > 0 \rightarrow U = U \cup D_{\text{MAT}_{i,j}} \]
**Distance between two points**

- **Distance between** $X, Y \in \mathbb{Z}$:
  - $A = \{X\}; D = A \bigcap \{\emptyset\}$  
    - $A$ = evolving binary image
  - while ($Y \notin A$) do
    - $Z = A$
    - $A = (A \oplus K) \cap F$
    - $D = D + A$
  - Done
  - If $A \equiv (A \oplus K) \cap F$ and $Y \notin A$ : $Z$ is not connected and $Y$ is not reachable from $X$

- Following back the path of max gradient we can always find one of **the minimum paths** between $X$, $Y
Weighted DT

In this case all neighbors are not considered at the same distance (e.g. 8-connectivity)

Example: a good approximation to the Euclidean distance in 8-connectivity (the result is about doubled) is given by:

\[ w = \]

\[
\begin{array}{ccc}
3 & 2 & 3 \\
2 & 2 & 2 \\
3 & 2 & 3 \\
\end{array}
\]
Example
Minimum path
4-conn
Minimum path
8-conn
Minimum path
4-conn
Minimum path
8-conn
Minimum path
4-conn
8-Convex Hull

- A set \(A\) is said to be convex iff the straight line segment joining any two points in \(A\) lies entirely within \(A\).
- The convex hull is the minimum \(n\)-sided convex polygon that completely circumscribes an object, gives another possible description of a binary object. An example is given in figure where a constrained 8-sided polygon has been chosen to coarsely describe the monk silhouette.
- To obtain the convex hull a simple algorithm propagates the object along the eight (more generally \(2n\)) orientations and then: i) logically OR the opposite propagated segments; and ii) logically AND the four (more generally \(n\)) resulting segments. The contour of the obtained polygon is the convex hull.
**Use of thickening: Convex hull**

- **Convex hull:** union of thickenings, each up to idempotence

![Diagram showing the process of thickening and forming the convex hull](image)

Original shape → Thickening with first mask → Thickening with second mask → Thickening with third mask → Thickening with fourth mask → Union of four thickenings

D.A. Forsyth
Example of using convex hull

Morfologia binària

Morfologia binària
Pixel Parallelism: Processor arrays

- Processor Element (PE) includes local memory
- Image distributed over all PE
- All PE run the same program (SIMD)
- The atomic data is anymore the unstructured pixel but the connected component
Propagation: examples

[Diagram showing examples of propagation issues with annotations like 'Spacing', 'Mousebyte', 'Cut', 'Short']
Mousebyte
Minimum distance
Global OR (Or-sum-tree)
Global OR (Or-sum-tree)