Introduction

- **Morphology** – a branch of biology concerned with the form and structure of plants and animals
- **Mathematical morphology** – a tool for extracting image components useful in the representation and description of image shape including:
  - Boundaries
  - Skeletons
  - Convex hull
- We will also look at morphological techniques for
  - Filtering
  - Thinning
  - Pruning

Preview

- Language of mathematical morphology is set theory
- Sets in mathematical morphology represent objects in an image
  - For example, the set of all black pixels in a binary image is a complete morphological description of the image
- For binary images, sets are members of the 2-D integer space \( \mathbb{Z}^2 \)
  - Each element of the set is a tuple (2-D vector) whose coordinates are the \((x,y)\) coordinates of a black (or white depending on convention) pixel in the image
- Gray-scale digital images are represented as sets in \( \mathbb{Z}^3 \)
  - Coordinates and gray-scale value
- Higher dimensioned sets could represent attributes such as color, time varying components, etc.

Basic Concepts from Set Theory

- Let \( A \) be a set in \( \mathbb{Z}^2 \). If \( a = (a_1, a_2) \) is an element of \( A \), then we write \( a \in A \)
- If \( a \) is not an element of \( A \) we write \( a \notin A \)
- A set with no elements is called the null or empty set and is denoted by the symbol \( \emptyset \)
- A set is specified by the contents of two braces: \( \{ \} \)
- For binary images, the elements of the sets are the coordinates of pixels representing objects
- If we write an expression of the form \( C = \{ w \mid w = -d, \text{for} \ d \in D \} \) we mean that \( C \) is the set of elements, \( w \), such that \( w \) is formed by multiplying each of the two coordinates of all the elements of set \( D \) by \(-1\)
Basic Concepts from Set Theory

- If every element of set $A$ is also an element of another set $B$, then $A$ is a subset of $B$ and we write $A \subseteq B$.
- The union of two sets $A$ and $B$, denoted by $C = A \cup B$, is the set of all elements belonging to either $A$, $B$, or both.
- The intersection of two sets $A$ and $B$, denoted by $D = A \cap B$, is the set of all elements belonging to both $A$ and $B$.
- Two sets are disjoint or mutually exclusive if they have no elements in common.
- $A \cap B = \emptyset$.
- The complement of a set $A$ is the set of elements not in $A$.
- $A^c = \{ w \mid w \notin A \}$.
- The difference of two sets $A$ and $B$, denoted $A-B$, is defined as $A-B = \{ w \mid w \in A, w \notin B \} = A \cap B^c$.

Logical Operations Involving Binary Images

- Principal logic operations
  - AND
  - OR
  - NOT (COMPLEMENT)
- Functionally complete
  - Combined to form any other logic operation
- Logic operations described have a one-to-one correspondence with the set operations intersection, union, and complement.
- Logic operations are restricted to binary images (not the case for general set operations).

Structuring Elements

- Set reflection and translation are used extensively in morphological operations based on structuring elements (SE).
- An SE is a small set (or subimage) used to “probe” an area of interest for certain properties.
- May be of arbitrary shape and size.
- In practice an SE is generally a regular geometric shape (square, rectangle, diamond, etc.).
- Generally padded to a rectangular array for image processing.
Structuring Elements (continued)

- Two-dimensional, or flat, structuring elements consist of a matrix of 0's and 1's, typically much smaller than the image being processed.
- The center pixel of the structuring element, called the origin, identifies the pixel of interest—the pixel being processed.
- The pixels in the structuring element containing 1's define the neighborhood of the structuring element.

Operation with a Structuring Element (example)

![Structuring Element Diagram](image)

**FIGURE 14.3** (a) A set (each shaded square is a member of the set); (b) A structuring element. (c) The set padded with background elements to form a rectangular array and provide a background border. (d) Structuring element as a rectangular array. (e) Set processed by the structuring element.

Structuring Elements: Matlab Functions

### `strnd`:
Create an m-ary structuring element.

**Syntax**: `SE = strnd(BOOL, R, C)`

**Description**: `strnd(BOOL, R, C)` returns a structuring element defined by the following:

- `R` = Number of rows in the structuring element.
- `C` = Number of columns in the structuring element.
- `BOOL` = Boolean scalar or 2-D array of logical values that define the structuring element. `true` elements are structuring elements, and `false` elements are background elements.

**Example**:
```
SE = strnd(true, 3, 3) % Create a 3x3 structuring element
```

**Table**:

<table>
<thead>
<tr>
<th>Structuring Element</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void</code></td>
<td></td>
</tr>
</tbody>
</table>

### `reflect`:
Reflect structuring element.

**Syntax**: `SE = reflect(SE)`

**Description**: `reflect(SE)` reflects a structuring element through its center. The effect is the same as if you rotated the structuring element's domain 180 degrees around its center for a 2-D structuring element. If `SE` is an array of structuring element objects, then `reflect(SE)` reflects each element of `SE`, and `SE` has the same size as `SE`.

**Class Support**: `SE` and `SE2` are `STREL` objects.

**Example**:
```
SE = strnd(true, 3, 3) % Create a 3x3 structuring element
refSE = reflect(SE) % Reflect the structuring element
```
Structuring Elements: Matlab Functions

translate

Translate structuring element

Syntax

\[ \text{SE} = \text{translate}(\text{SE}, \text{Y}) \]

Description

\( \text{SE} = \text{reflect}(\text{SE}, \text{Y}) \) translates a structuring element \( \text{SE} \) in \( \mathbb{R}^2 \) space. \( \text{Y} \) is an \( N \times 1 \) vector containing the offsets of the desired translation in each dimension.

Class Support

\( \text{SE} \) and \( \text{Y} \) are \text{struct} objects; \( \text{Y} \) is a vector of double precision values.

Dilation

- With \( A \) and \( B \) as sets in \( \mathbb{Z}^2 \), the dilation of \( A \) by \( B \), denoted \( A \oplus B \), is defined as
  \[ A \oplus B = \{ z \mid (B_z) \cap A \neq \emptyset \} \]
- This formulation is based on the reflection of \( B \) about its origin and shifting this reflection by \( z \)
- The dilation of \( A \) by \( B \) is the set of all displacements, \( z \), such that \( B \) and \( A \) overlap by at least one element
- Therefore, another expression for the dilation of \( A \) by \( B \) is
  \[ A \oplus B = \{ z \mid [(B_z) \cap A] \subseteq A \} \]
- Set \( B \) is the structuring element

Dilation Example

- \( B = \hat{B} \) because \( B \) is symmetric with respect to its origin
- The dashed line shows the original set \( A \) and the solid boundary shows the limit beyond which any further displacements of the origin of \( \hat{B} \) by \( z \) would cause the intersection of \( \hat{B} \) and \( A \) to be empty
- All points inside this boundary constitute the dilation of \( A \) by \( B \)
- The second case shows more dilation vertically than horizontally

Dilation Application

- One simple application of dilation is for bridging gaps
- In the image below, the maximum break length is two pixels
- Although low pass filtering can be used to accomplish the same task, this generates a gray-scale image that must then be thresholded to produce a resulting binary image
Matlab: Dilation Example

\[ I = \text{imread('brokenlines.bmp');} \]
\[ \text{imshow}(I), \text{title('Original');} \]
\[ \text{SE = strel('square',3);} \]
\[ J = \text{imdilate}(I, \text{SE}); \]
\[ \text{figure, imshow}(J), \text{title('Dilated');} \]

\[
\begin{array}{ccc}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
\end{array}
\]

Erosion

- With A and B as sets in \( \mathbb{Z}^2 \), the erosion of A by B, denoted \( A \ominus B \), is defined as

\[ A \ominus B = \{ z \mid (B)_z \subseteq A \} \]

- In words, the erosion of A by B is the set of all points z such that B, translated by z, is contained in A

- Dilation and erosion are duals of each other with respect to set complementation and reflection

- Therefore

\[ (A \ominus B)^c = A^c \oplus B \]

and

\[ (A \ominus B)^e = A^e \ominus B \]

Erosion Example
Erosion and Dilation Application

- One simple application of erosion is for eliminating irrelevant detail (in terms of size) from a binary image.
- Note: In general, dilation does not restore fully the eroded objects.

Erosion of a binary image with a 13x13 size structuring element and subsequent dilation of the result with the same element.