Morphological image processing

Morphological image processing is an instrument developed for extracting image components based on shape and not on pixels intensity. The morphological transformations are usually applied on binary images but the techniques can be adapted in order to work o grayscale images also.

In morphological image processing the image is explored with a small shape, known as structuring element. The structuring element is a small binary image which is shifted through all the pixels and certain comparisons are performed in order to extract shape information from the image. The 'language' for expressing morphological operation is set theory. The main operations are erosion and dilation; all the other morphological techniques are expressed in terms of erosion and dilation.

The reflection of a set $\boldsymbol{B}$, denoted $\bar{\boldsymbol{B}}$ is defined as

$$\bar{\boldsymbol{B}} = \{ \boldsymbol{w} \, ; \; \boldsymbol{w} = -\boldsymbol{b} \, , \text{ for } \; \boldsymbol{b} \in \boldsymbol{B} \; \}$$

The translation of a set B by point $\boldsymbol{z} = (\boldsymbol{z_1}, \boldsymbol{z_2})$, denoted $(\boldsymbol{B})_{\boldsymbol{z}}$ is defined as

$$(\boldsymbol{B})_z = \{ \boldsymbol{c} \, ; \boldsymbol{c} = \boldsymbol{z} + \boldsymbol{b} \text{ for } \boldsymbol{b} \in \boldsymbol{B} \; \}$$

Erosion and Dilation

Many of the morphological algorithms are based on these two primitive operations: *erosion* and *dilation*.

Erosion

Let $\boldsymbol{A}$ and $\boldsymbol{B}$ be to sets from $\mathbb{Z}^2$. The *erosion* of $\boldsymbol{A}$ by $\boldsymbol{B}$, denoted $\boldsymbol{A} \odot \boldsymbol{B}$ is defined as:

$$\boldsymbol{A} \odot \boldsymbol{B} = \{ \; \boldsymbol{z} \; ; \; (\boldsymbol{B})_z \subseteq \boldsymbol{A} \; \}$$

This definition indicates that the erosion of $\boldsymbol{A}$ by $\boldsymbol{B}$ is the set of all points $\boldsymbol{z}$ such that $\boldsymbol{B}$, translated by $\boldsymbol{z}$, is contained in $\boldsymbol{A}$. In the following, set $\boldsymbol{B}$ is assumed to be a structuring element. Because the statement that ***B*** has to be contained in ***A*** is equivalent to ***B*** not shearing any common elements with the background, erosion can be expressed equivalently:

$$\boldsymbol{A} \odot \boldsymbol{B} = \{\, z \,;\, (\boldsymbol{B})_z \cap \boldsymbol{A}^c = \varnothing \,\}$$

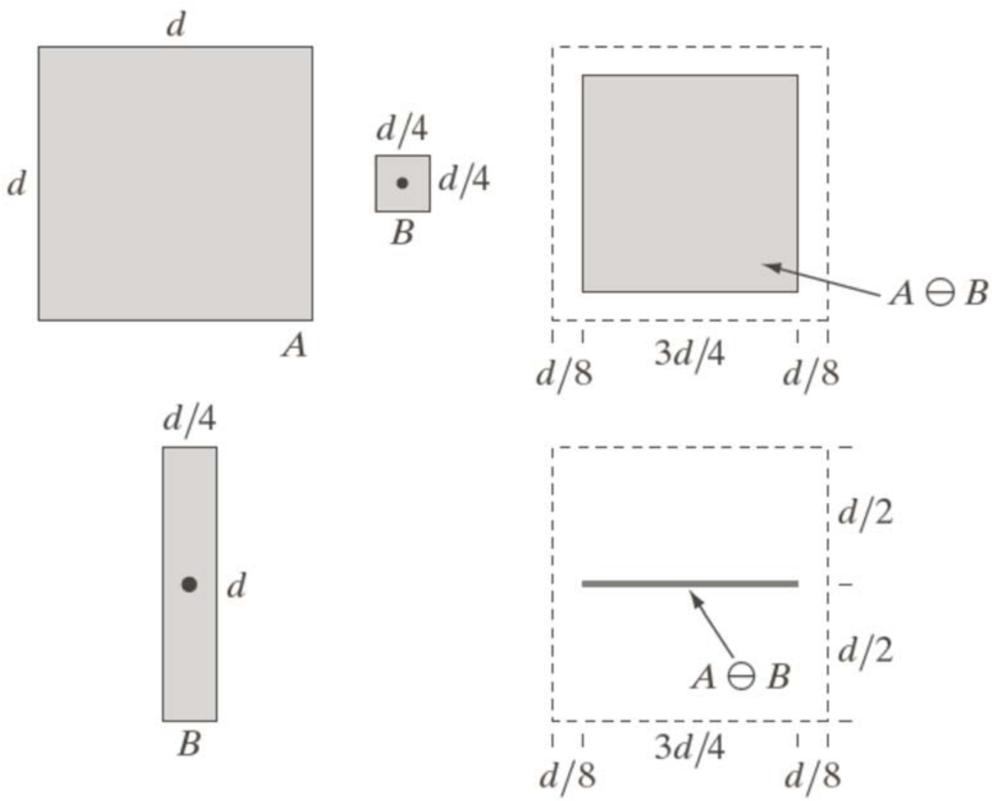


FIGURE 9.4 (a) Set A. (b) Square structuring element, B. (c) Erosion of A by B, shown shaded. (d) Elongated structuring element. (e) Erosion of A by B using this element. The dotted border in (c) and (e) is the boundary of set A, shown only for reference.

Equivalent definitions of erosion:

$$\boldsymbol{A} \odot \boldsymbol{B} = \{\, \boldsymbol{w} \in \mathbb{Z}^2 ; \boldsymbol{w} + \boldsymbol{b} \in \boldsymbol{A} \ \text{ for every } \ \boldsymbol{b} \in \boldsymbol{B} \,\}$$

$$\boldsymbol{A} \odot \boldsymbol{B} = \bigcap_{\boldsymbol{b} \in \boldsymbol{B}} (\boldsymbol{A})_{-\boldsymbol{b}}$$

Erosion shrinks or thins objects in a binary image. We can view erosion as a *morphological filtering* operation in which image details smaller than the structuring element are filtered (removed) from the image.

Dilation

Let $\boldsymbol{A}$ and $\boldsymbol{B}$ be to sets in $\mathbb{Z}^2$. The *dilation* of $\boldsymbol{A}$ by $\boldsymbol{B}$, denoted $\boldsymbol{A} \oplus \boldsymbol{B}$ is defined as:

$$\boldsymbol{A} \oplus \boldsymbol{B} = \{\, \boldsymbol{z} \,;\, (\bar{\boldsymbol{B}})_{\boldsymbol{z}} \cap \boldsymbol{A} \neq \varnothing \,\}$$

The dilation of $\boldsymbol{A}$ by $\boldsymbol{B}$ is the set of all displacements $\boldsymbol{z}$, such that $\bar{\boldsymbol{B}}$ and $\boldsymbol{A}$ overlap by at least one element. The above definition can be written equivalently as:

$$\boldsymbol{A} \oplus \boldsymbol{B} = \{\, \boldsymbol{z} ; \left[(\bar{\boldsymbol{B}})_{\boldsymbol{z}} \cap \boldsymbol{A} \ \right] \subseteq \boldsymbol{A} \}$$

We assume that $\boldsymbol{B}$ is a structuring element.

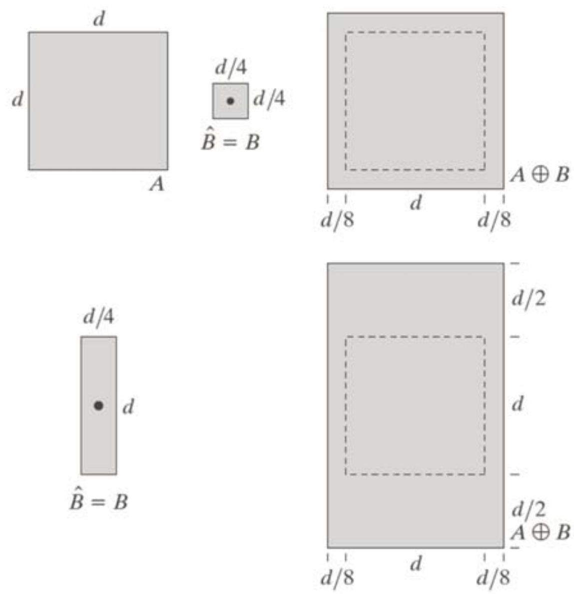
Equivalent definitions of dilation:

$$\boldsymbol{A} \oplus \boldsymbol{B} = \{\, \boldsymbol{w} \in \mathbb{Z}^2 \ ; \boldsymbol{w} = \boldsymbol{a} + \boldsymbol{b} \text{ , for some } \ \boldsymbol{a} \in \boldsymbol{A} \text{ and } \boldsymbol{b} \in \boldsymbol{B} \ \}$$

$$\boldsymbol{A} \oplus \boldsymbol{B} = \bigcup_{\boldsymbol{b} \in \boldsymbol{B}} (\boldsymbol{A})_{\boldsymbol{b}}$$

The basic process of rotating $\boldsymbol{B}$ about its origin and then successively displacing it so that it slides over set (image) $\boldsymbol{A}$ is analogous to spatial convolution. Dilation being based on set operations is a nonlinear operation, whereas convolution is a linear operation.

Unlike erosion which is a shrinking or thinning operation, dilation “grows” or “thickens” objects in a binary image. The specific manner and the extent of this thickening are controlled by the shape of the structuring element used.



a b c
d e

FIGURE 9.6
(a) Set A. (b) Square structuring element (the dot denotes the origin). (c) Dilation of A by B, shown shaded. (d) Elongated structuring element. (e) Dilation of A using this element. The dotted border in (c) and (e) is the boundary of set A, shown only for reference

More information on morphological techniques can be found in Course 4 (week 4) and the other posted files.

MATLAB functions for morphological processing: ***strel*** (for creating structuring elements), ***imerode, imdilate, imopen, imclose, bwmorph, bwhitmiss***, …

For this lab only binary image are to be processed. If the input images are color or grayscale apply edge detection filters and thresholding in order to transform them into binary images.

Test the morphological operations provided by MATLAB or OpenCV on image "DIP_GW_page.tif".

"Clean" the image "patrat_cu_pete.tif" using successive opening and closing operations. Find the combination that gives the best result.

For the faces in *faces95* apply the boundary extraction technique.

Using "hit-or-miss" transformation find the geometrical figures in "basic_shapes.png", "shapes.png". Count how many of each (circle, square, rectangle, triangle, …) are in a given image.