EE795: Computer Vision and Intelligent Systems

Spring 2012
TTh 17:30-18:45 WRI C225

Lecture 05
130205

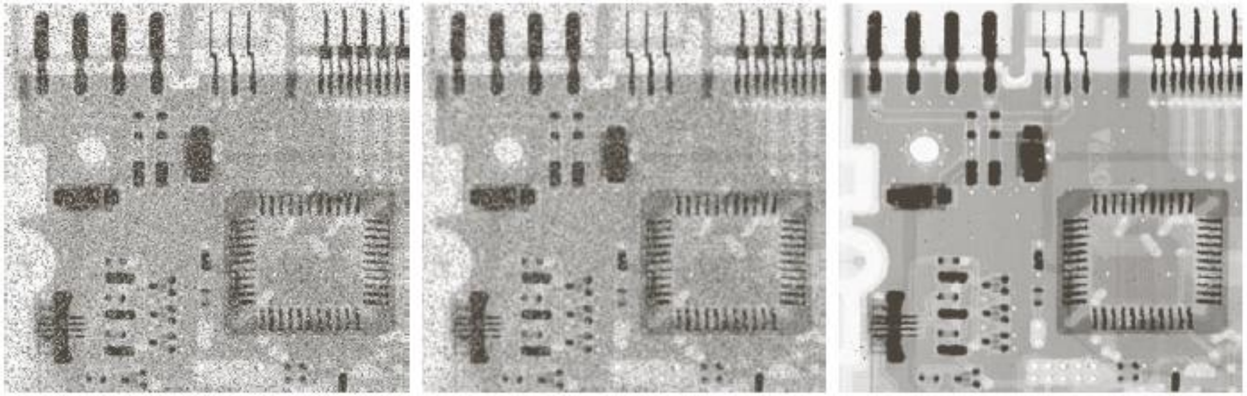
http://www.ee.unlv.edu/~b1morris/ecg795/

Outline

- Review
  - Non-linear Filters
  - Sharpening Filters
- Morphology
- Connected Components

Median Filtering

- Sometimes linear filtering is not sufficient
  - Non-linear neighborhood operations are required
- Median filter – replaces the center pixel in a mask by the median of its neighbors
  - Non-linear operation, computationally more expensive
  - Provides excellent noise-reduction with less blurring than smoothing filters of similar size (edge preserving)
    - For impulse and salt-and-pepper noise



a b c

FIGURE 3.35 (a) X-ray image of circuit board corrupted by salt-and-pepper noise. (b) Noise reduction with a 3 × 3 averaging mask. (c) Noise reduction with a 3 × 3 median filter. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

Bilateral Filtering

- Combine the idea of a weighted filter kernel with a better version of outlier rejection
  - α-trimmed mean calculates average in neighborhood excluding the α fraction that are smallest or largest
- $w(i,j,k,l) = d(i,j,k,l) \times r(i,j,k,l)$
  - $d(i,j,k,l)$ - domain kernel specifies “distance” similarity between pixels (usually Gauassian)
  - $r(i,j,k,l)$ – range kernel specifies “appearance (intensity)” similarity between pixels

Bilateral Filtering Example

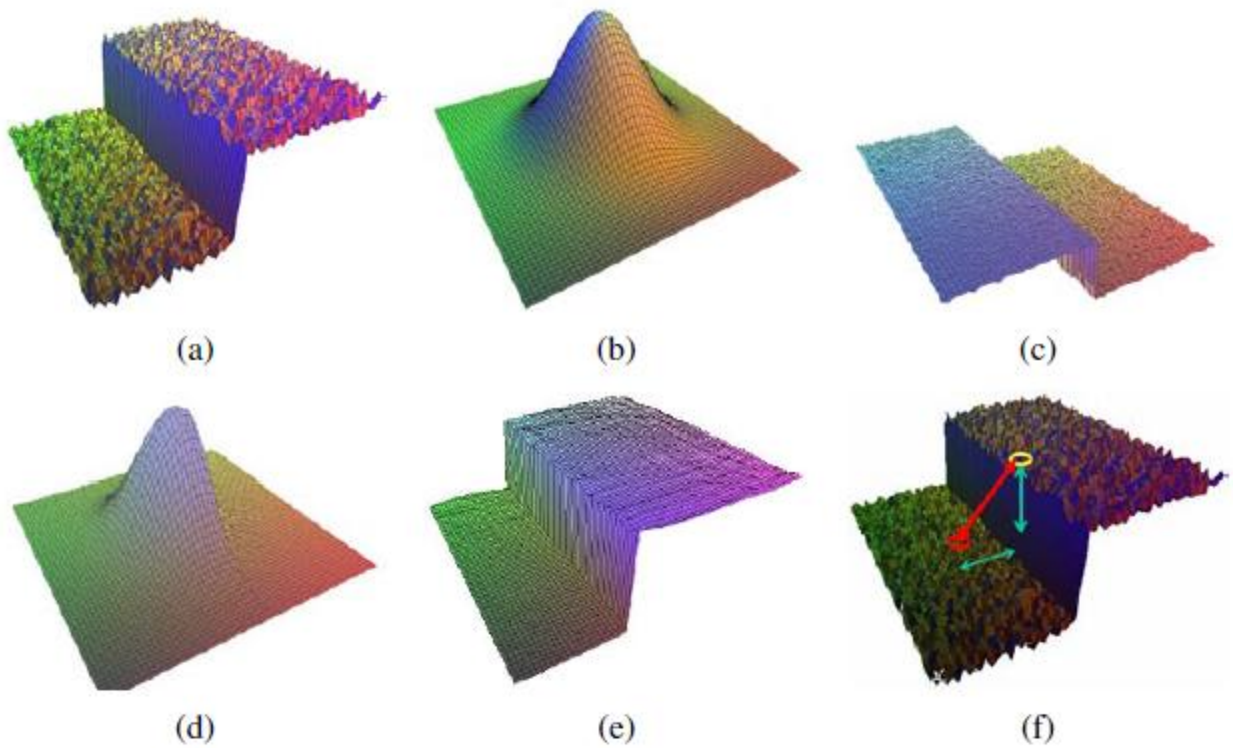


Figure 3.20 Bilateral filtering (Durand and Dorsey 2002) © 2002 ACM: (a) noisy step edge input; (b) domain filter (Gaussian); (c) range filter (similarity to center pixel value); (d) bilateral filter; (e) filtered step edge output; (f) 3D distance between pixels.

Sharpening Filters

- Sharpening filters are used to highlight fine detail or enhance blurred detail
  - Differentiation (difference in discrete case)

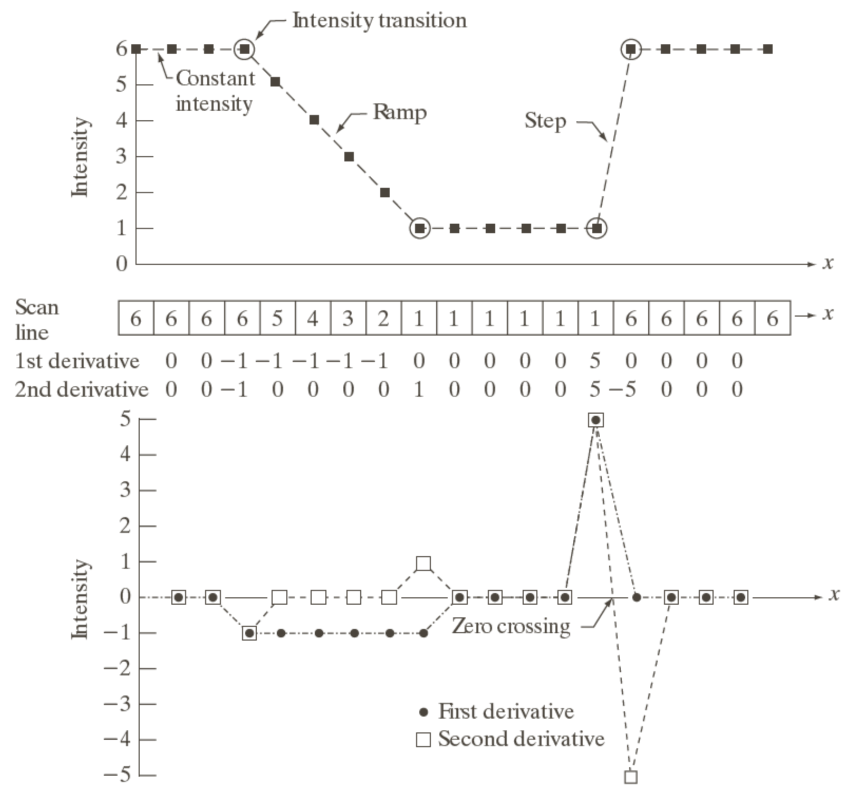
- Discrete approximation of a derivative
  - $\frac{\partial f}{\partial x} = f(x+1) - f(x)$
  - $\frac{\partial f}{\partial x} = f(x+1) - f(x-1)$
    - Center symmetric
- Second-order derivative
  - $\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x)$

Difference Properties

- Zero in constant segments
- Non-zero at intensity transition

- 1st derivative
  - Non-zero along ramps
    - Thick edges
- 2nd derivative
  - Zero along ramps
    - Thin edges

- 2nd order filter is more aggressive at enhancing sharp edges
  - Notice: the step gets both a negative and positive response → double line



Discrete Laplacian

- The Laplacian is simplest isotropic derivative operator
  - $\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$
  - Isotropic – rotation invariant
    - Rotations of 90° → zeros in corners
    - Rotations of 45° → ones in corners
- Center pixel sign indicates light-to-dark or dark-to-light transitions

(a)

0	1	0
1	−4	1
0	1	0

(b)

1	1	1
1	−8	1
1	1	1

(c)

0	−1	0
−1	4	−1
0	−1	0

(d)

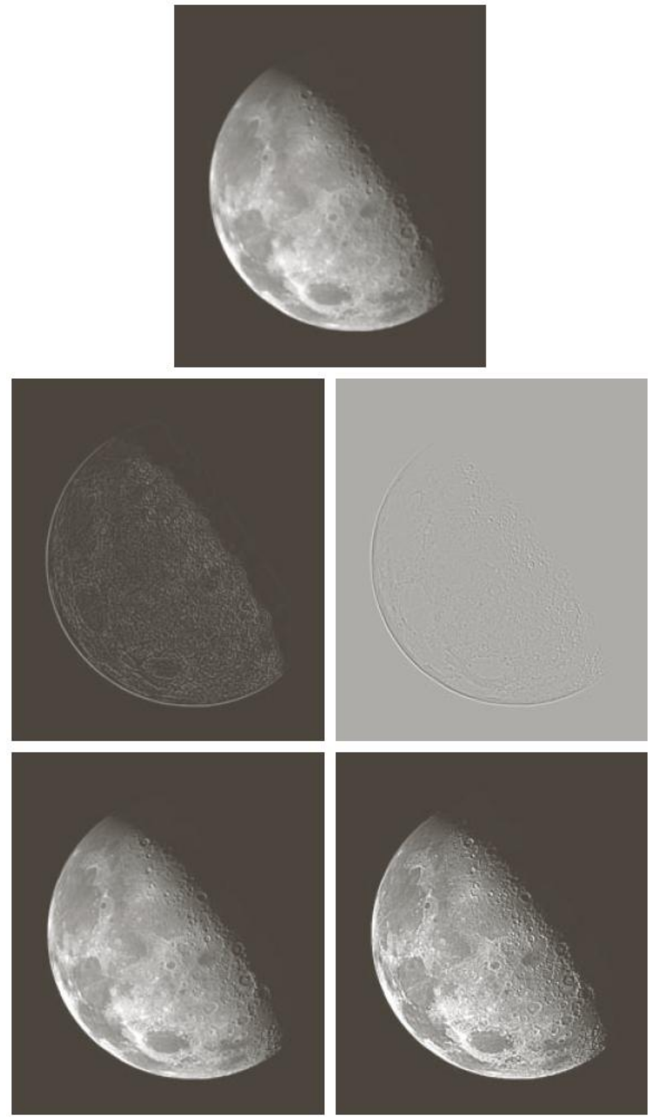
−1	−1	−1
−1	8	−1
−1	−1	−1

a b
c d

FIGURE 3.37 (a) Filter mask used to implement Eq. (3.6-6). (b) Mask used to implement an extension of this equation that includes the diagonal terms. (c) and (d) Two other implementations of the Laplacian found frequently in practice.

Sharpening Images

- Sharpened image created by addition of Laplacian
  - $g(x,y) = \begin{cases} f(x,y) - \nabla^2 f(x,y) & w(0,0) < 0 \\ f(x,y) + \nabla^2 f(x,y) & w(0,0) > 0 \end{cases}$
- Notice: the use of diagonal entries creates much sharper output image
- How can we compute $g(x,y)$ in one filter pass without the image addition?
  - Think of a linear system

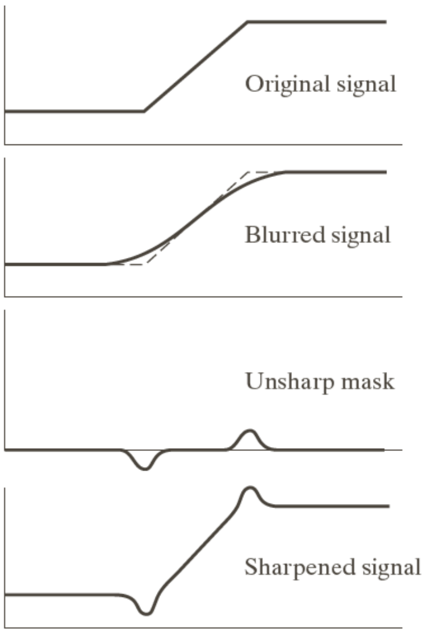


a
b c
d e

FIGURE 3.38 (a) Blurred image of the North Pole of the moon. (b) Laplacian without scaling. (c) Laplacian with scaling. (d) Image sharpened using the mask in Fig. 3.37(a). (e) Result of using the mask in Fig. 3.37(b). (Original image courtesy of NASA.)

Unsharp Masking

- Edges can be obtained by subtracting a blurred version of an image
  - $f_{us}(x,y) = f(x,y) - \bar{f}(x,y)$
  - Blurred image
    - $\bar{f}(x,y) = h_{\text{blur}} * f(x,y)$
- Sharpened image
  - $f_s(x,y) = f(x,y) + \gamma f_{us}(x,y)$



a b c d

FIGURE 3.39 1-D illustration of the mechanics of unsharp masking. (a) Original signal. (b) Blurred signal with original shown dashed for reference. (c) Unsharp mask. (d) Sharpened signal, obtained by adding (c) to (a).

a b c d e

FIGURE 3.40 (a) Original image. (b) Result of blurring with a Gaussian filter. (c) Unsharp mask. (d) Result of using unsharp masking. (e) Result of using highboost filtering.

The Gradient

- 1st derivatives can be useful for enhancement of edges
  - Useful preprocessing before edge extraction and interest point detection
- The gradient is a vector indicating edge direction
  - $\nabla \mathbf{f} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$
- The gradient magnitude can be approximated as
  - $\nabla f \approx |G_x| + |G_y|$
  - This give isotropic results for rotations of 90°
- Sobel operators
  - Have directional sensitivity
  - Coefficients sum to zero
    - Zero response in constant intensity region

−1	−2	−1
0	0	0
1	2	1

G_y

−1	0	1
−2	0	2
−1	0	1

G_x

Morphological Image Processing

- Filtering done on binary images
  - Images with two values [0,1], [0, 255], [black,white]
  - Typically, this image will be obtained by thresholding
    - $g(x,y) = \begin{cases} 1 & f(x,y) > T \\ 0 & f(x,y) \leq T \end{cases}$
- Morphology is concerned with the structure and shape
- In morphology, a binary image is convolved with a structuring element *s* and results in a binary image

- See Chapter 9 of Gonzalez and Woods for a more complete treatment
- Matlab
  - http://www.mathworks.com/help/images/pixel-values-and-image-statistics.html

Mathematical Morphology

- Tool for extracting image components that are useful in the representation and description of region shape
  - Boundaries, skeletons, convex hull, etc.
- The language of mathematical morphology is set theory
  - A set represents an object in an image

- This is often useful in video processing because of the simplicity of processing and emphasis on “objects”
  - Handy tool for “clean up” of a thresholded image

Morphological Operations

- Threshold operation
  - $\theta(f,t) = \begin{cases} 1 & f \geq t \\ 0 & \text{else} \end{cases}$
- Structuring element
  - s – e.g. 3 x 3 box filter (1's indicate included pixels in the mask)
  - S – number of "on" pixels in s
- Count of 1s in a structuring element
  - $c = f \otimes s$
  - Correlation (filter) raster scan procedure
- Basic morphological operations can be extended to grayscale images
- Dilation
  - $\text{dilate}(f,s) = \theta(c,1)$
  - Grows (thickens) 1 locations
- Erosion
  - $\text{erode}(f,s) = \theta(c,S)$
  - Shrink (thins) 1 locations
- Opening
  - $\text{open}(f,s) = \text{dilate}(\text{erode}(f,s),s)$
  - Generally smooth the contour of an object, breaks narrow isthmuses, and eliminates thin protrusions
- Closing
  - $\text{close}(f,s) = \text{erode}(\text{dilate}(f,s),s)$
  - Generally smooth the contour of an object, fuses narrow breaks/separations, eliminates small holes, and fills gaps in a contour

Morphology Example

Note: Black is "1" location

(a) (b) (c) (d) (e) (f)

Figure 3.21 Binary image morphology: (a) original image; (b) dilation; (c) erosion; (d) majority; (e) opening; (f) closing. The structuring element for all examples is a 5×5 square. The effects of majority are a subtle rounding of sharp corners. Opening fails to eliminate the dot, since it is not wide enough.

- Dilation - grows (thickens) 1 locations
- Erosion - shrink (thins) 1 locations
- Opening - generally smooth the contour of an object, breaks narrow isthmuses, and eliminates thin protrusions
- Closing - generally smooth the contour of an object, fuses narrow breaks/separations, eliminates small holes, and fills gaps in a contour

More Morphology

- Chapter 9 Gonzalez and Woods

- Logic Operations on Binary Images
- Erosion
- Dilation
- Opening
- Closing

Binary Image Logic Operations

- Extension of basic logic operators
    - NOT, AND, OR, XOR

- Often use for "masking"

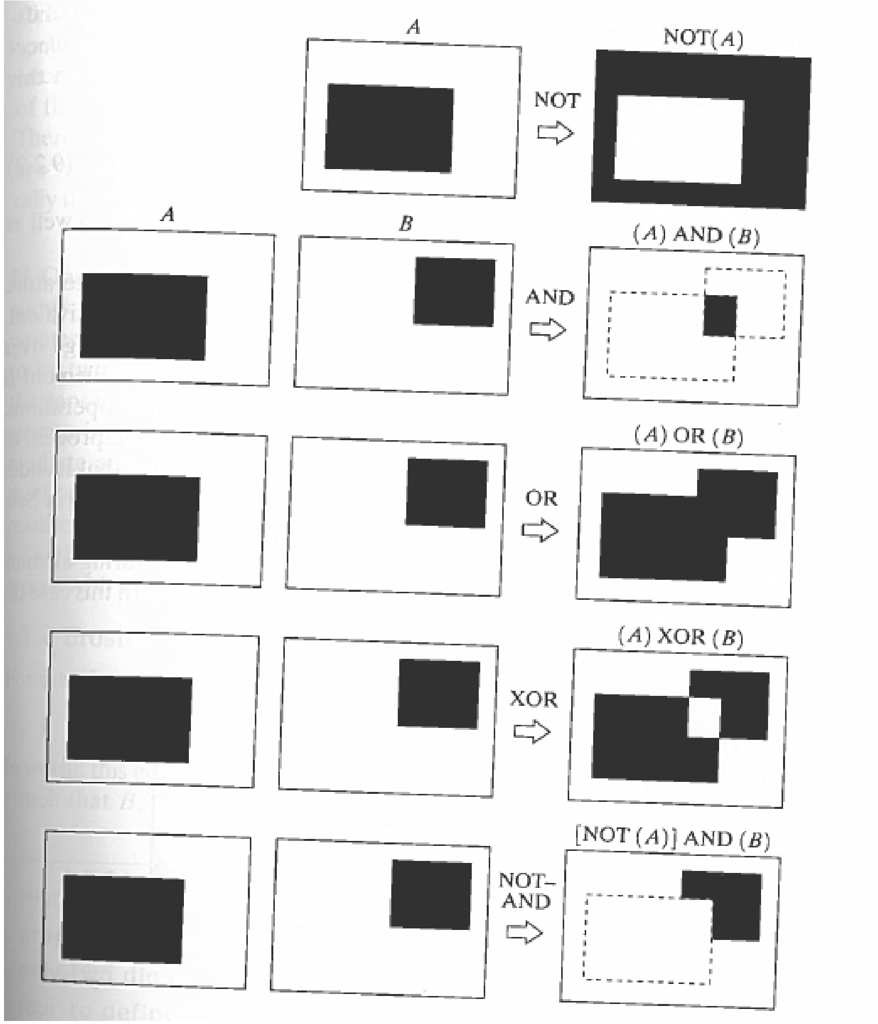


FIGURE 9.3 Some logic operations between binary images. Black represents binary 1s and white binary 0s in this example.

Structuring Elements

- The structuring element (SE) can be any shape
  - This is a mask of "on" pixels within a rectangular container
  - Typically, the SE is symmetric

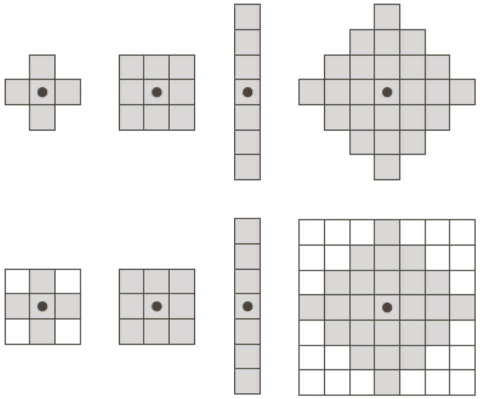


FIGURE 9.2 First row: Examples of structuring elements. Second row: Structuring elements converted to rectangular arrays. The dots denote the centers of the SEs.

Erosion

- Retain only pixels where the entire SE is overlapped
  - $A \ominus B = \{z | (B)_z \subseteq A\}$
    - A – image
    - B – SE
    - z – displacements (x,y locations)

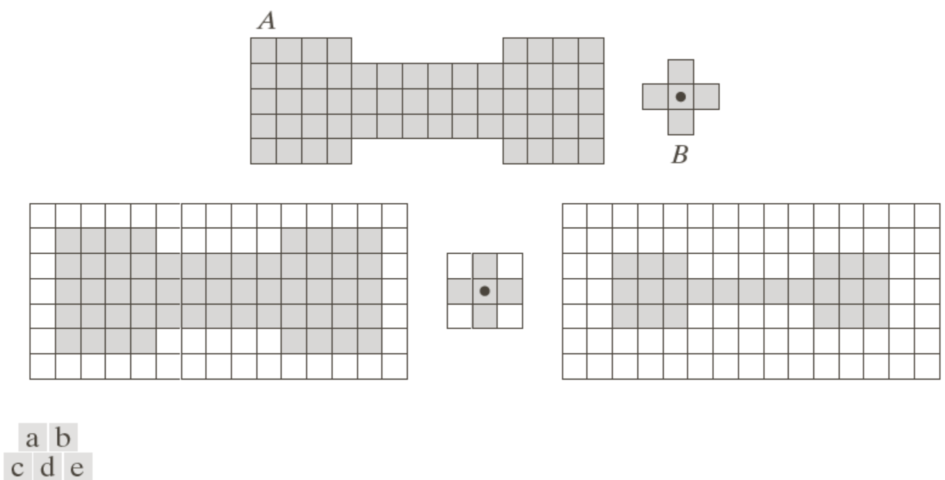


FIGURE 9.3 (a) A set (each shaded square is a member of the set). (b) A structuring element. (c) The set padded with background elements to form a rectangular array and provide a background border. (d) Structuring element as a rectangular array. (e) Set processed by the structuring element.

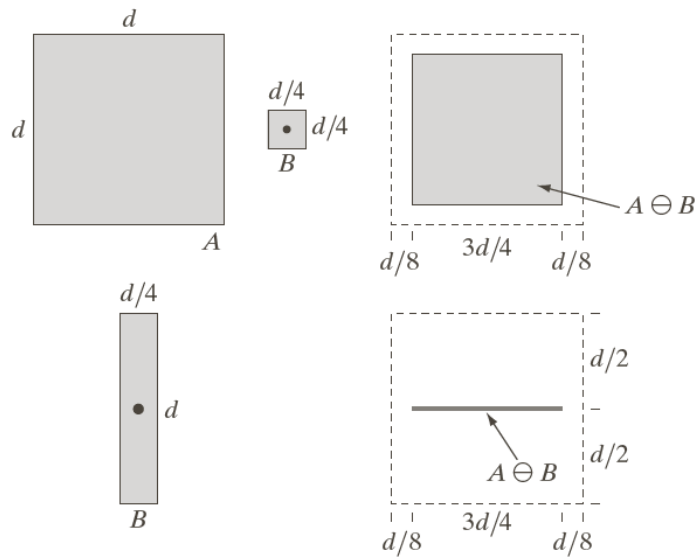
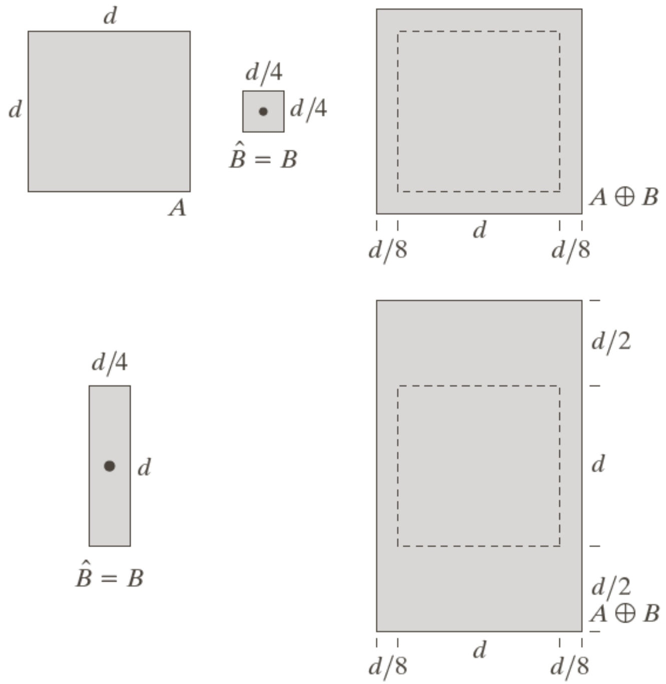


FIGURE 9.4 (a) Set A. (b) Square structuring element, B. (c) Erosion of A by B, shown shaded. (d) Elongated structuring element. (e) Erosion of A by B using this element. The dotted border in (c) and (e) is the boundary of set A, shown only for reference.

Dilation

- Output image is "on" anywhere the SE touches an "on" pixel
  - $A \oplus B = \{z | (B)_z \cap A \neq \emptyset\}$
    - A – image
    - B – SE
    - z – displacements (x,y locations)



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

Opening

- All pixels that fit inside when the SE is “rolled” on the inside of a boundary
  - $A \circ B = \cup \{(B)_z | (B)_z \subseteq A\}$

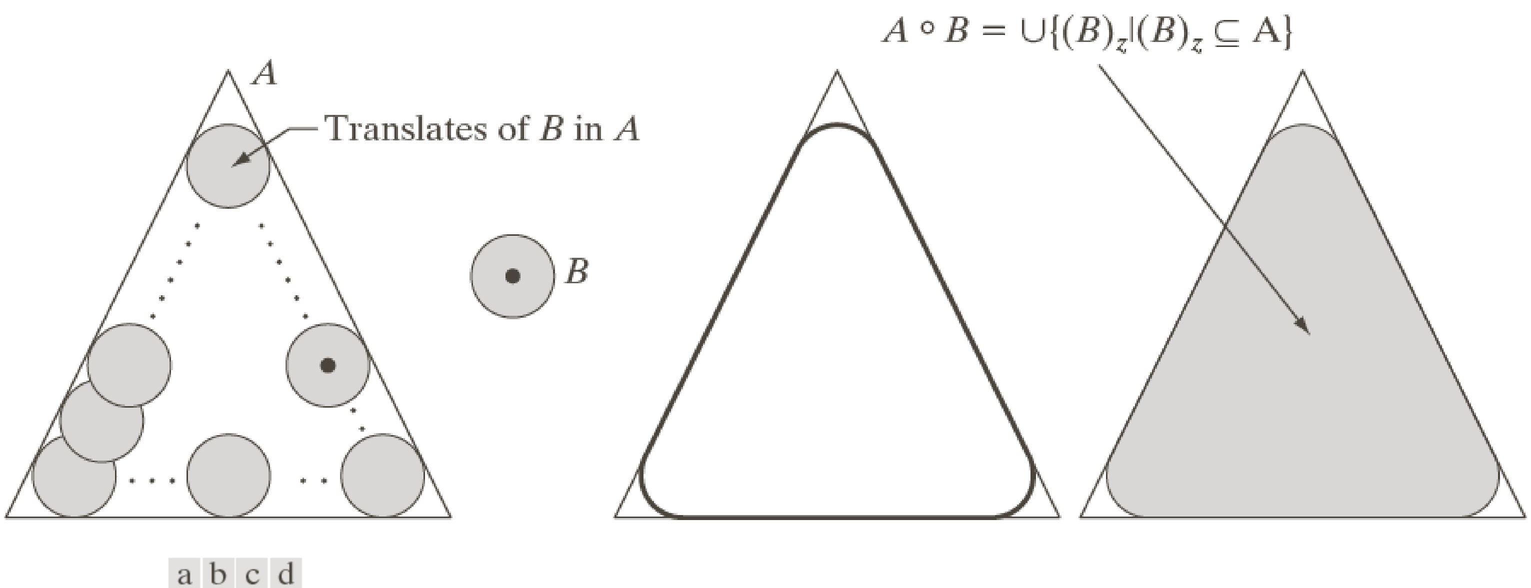


FIGURE 9.8 (a) Structuring element B “rolling” along the inner boundary of A (the dot indicates the origin of B). (b) Structuring element. (c) The heavy line is the outer boundary of the opening. (d) Complete opening (shaded). We did not shade A in (a) for clarity.

Closing

- All pixels that fit inside when the SE is “rolled” on the outside of a boundary

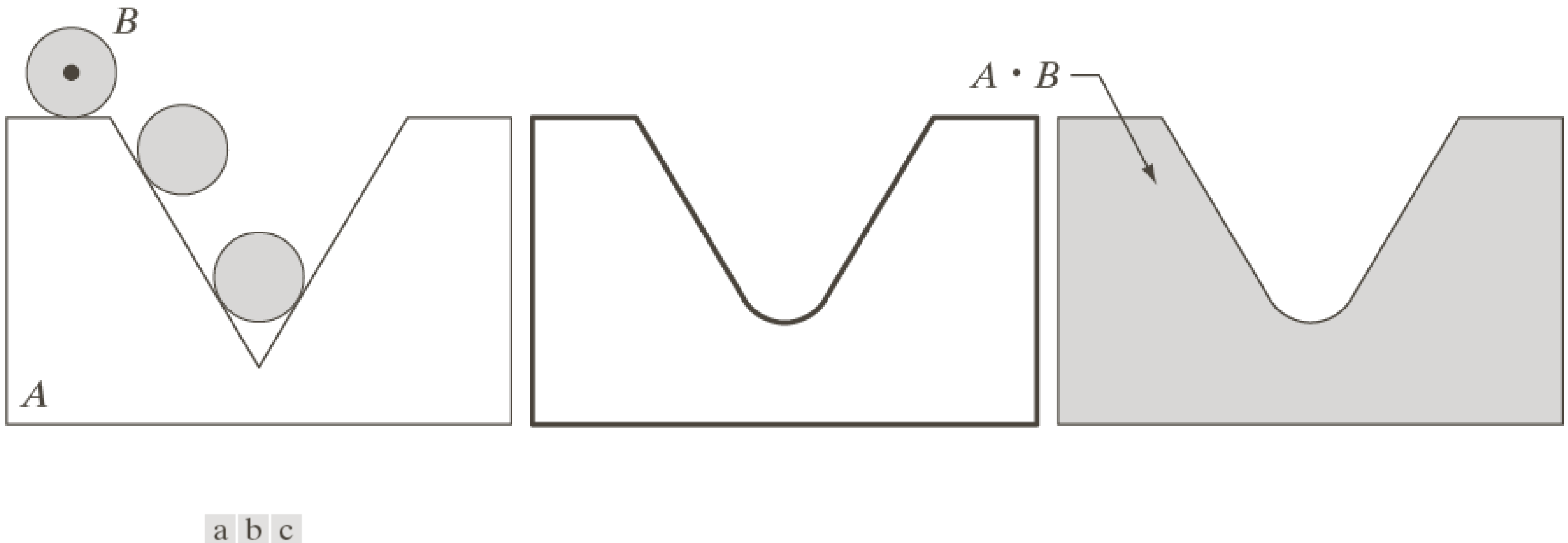


FIGURE 9.9 (a) Structuring element B “rolling” on the outer boundary of set A. (b) The heavy line is the outer boundary of the closing. (c) Complete closing (shaded). We did not shade A in (a) for clarity.

Opening/Closing Examples

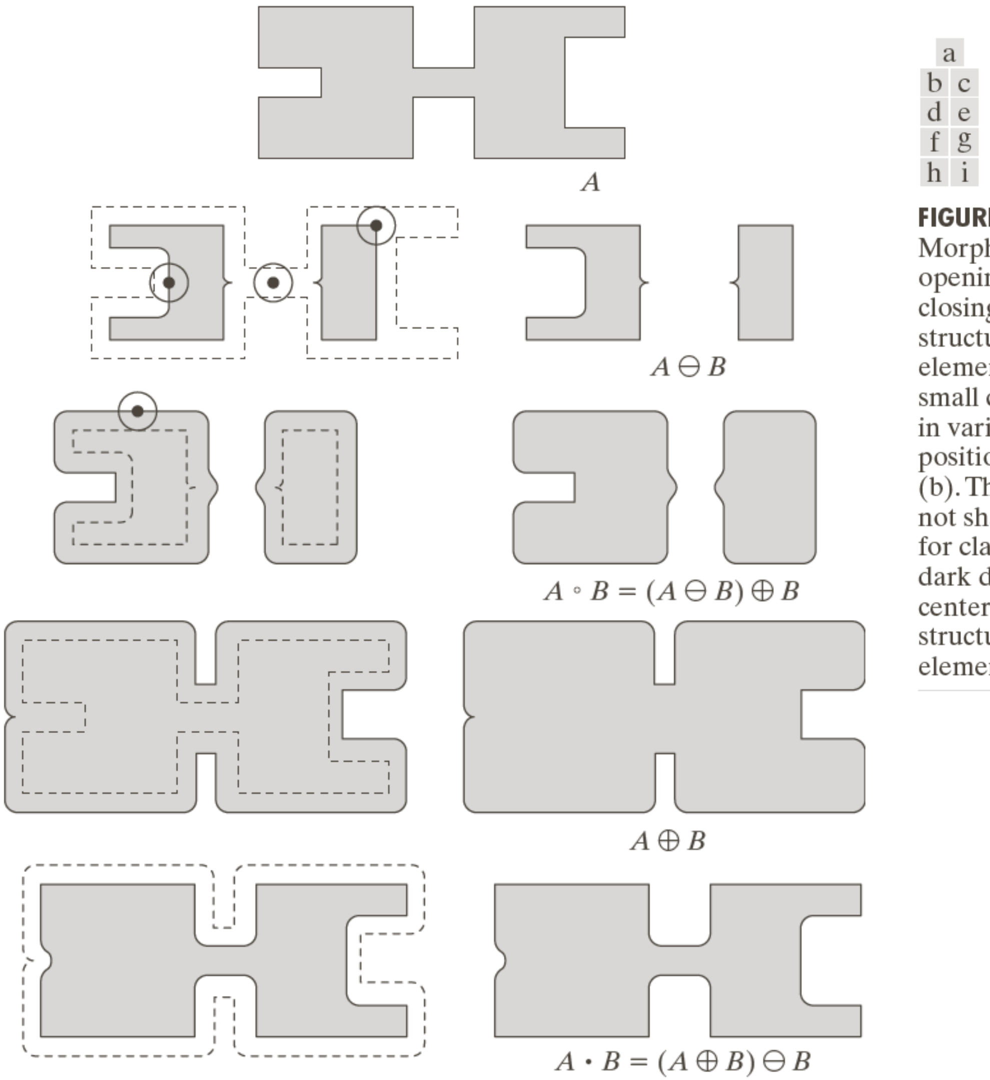


FIGURE 9.10 Morphological opening and closing. The structuring element is the small circle shown in various positions in (b). The SE was not shaded here for clarity. The dark dot is the center of the structuring element.

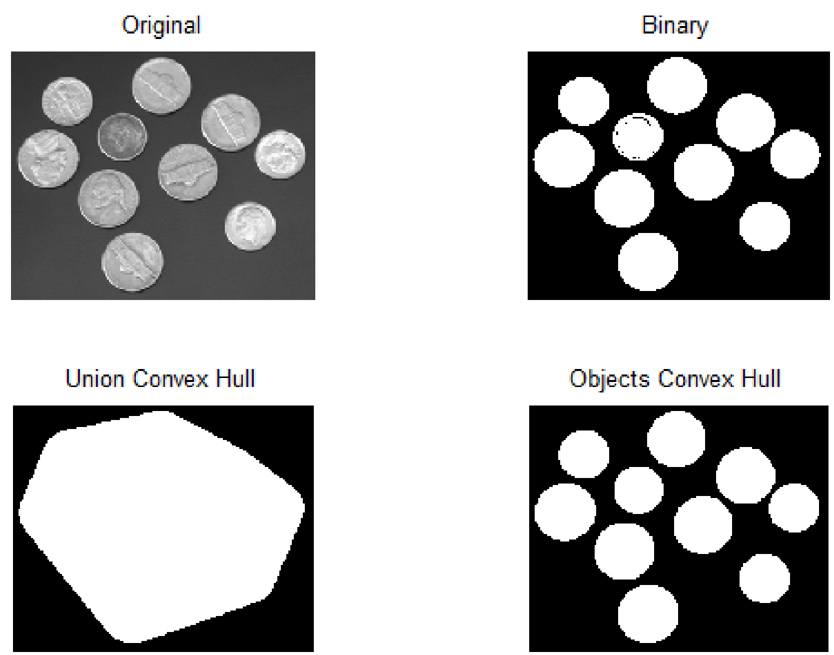
Other Morphological Operations

- Boundary extraction
  - $\beta(A) = A - (A \ominus B)$
  - Subtract erosion from original
  - Notice this is an edge extraction

a b

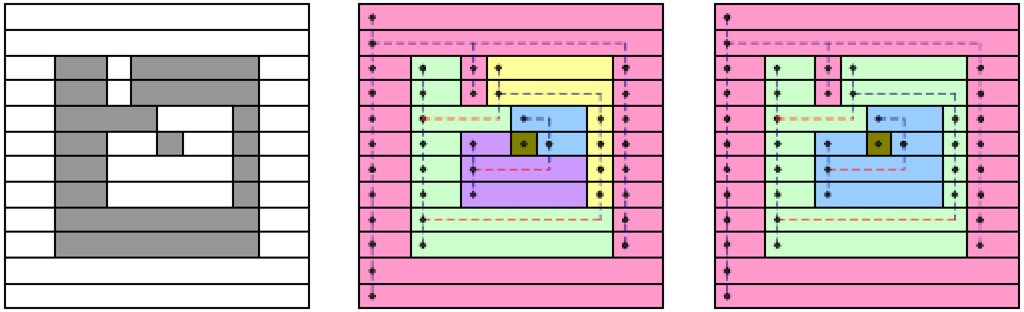
FIGURE 9.14 (a) A simple binary image, with 1s represented in white. (b) Result of using Eq. (9.5-1) with the structuring element in Fig. 9.13(b).

- Convex hull (H)
  - Smallest convex set that contains another set S
  - This is often done for a collection of 2D or 3D point
  - `bwconvhull.m`



Connected Components

- Semi-global image operation to provide consistent labels to similar regions
  - Based on adjacency concept
- Most efficient algorithms compute in two passes



(a) (b) (c)

Figure 3.23 Connected component computation: (a) original grayscale image; (b) horizontal runs (nodes) connected by vertical (graph) edges (dashed blue)—runs are pseudocolored with unique colors inherited from parent nodes; (c) re-coloring after merging adjacent segments.

- More computational formulations (iterative) exist from morphology
  - $X_k = (X_{k-1} \oplus B) \cap A$

X_k ↑ Connected component

B ↑ Structuring element

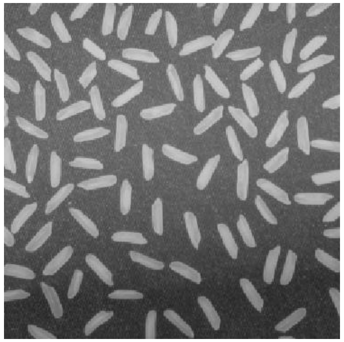
A ← Set

More Connected Components

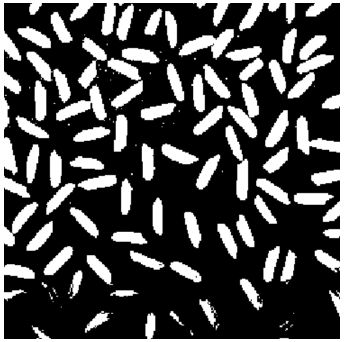
- Typically, only the "white" pixels will be considered objects
  - Dark pixels are background and do not get counted
- After labeling connected components, statistics from each region can be computed
  - Statistics describe the region – e.g. area, centroid, perimeter, etc.

- Matlab functions
  - `bwconncomp.m`, `labelmatrix.m` (`bwlabel.m`)- label image
  - `label2rgb.m` – color components for viewing
  - `regionprops.m` – calculate region statistics

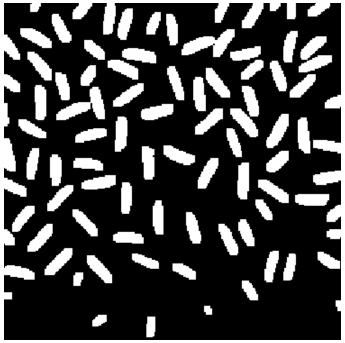
Connected Component Example



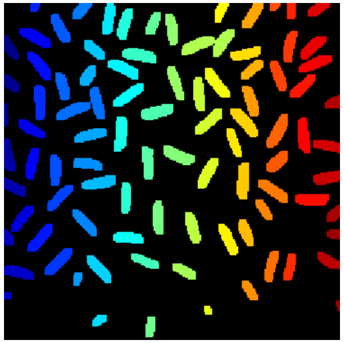
Grayscale image



Threshold image



Opened Image



Labeled image – 91 grains of rice