From Pixels to “Blobs”

15-463: Rendering and Image Processing
Alexei Efros

Today

Blobs

- Need for blobs
- Extracting blobs
- Image Segmentation

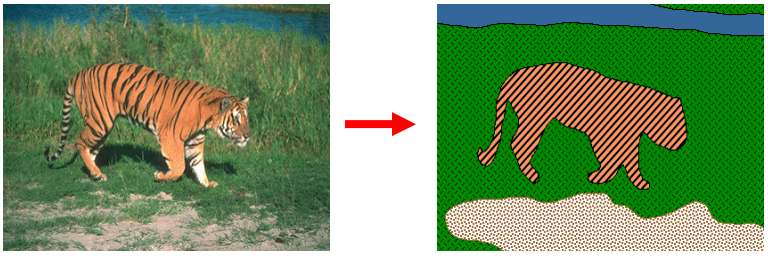
Working with binary images

- Mathematical Morphology
- Blob properties

Further Reading:

- Gonzalez and Woods, Ch. 9 & 10

Goal: Extract “Blobs”



What are “blobs”?

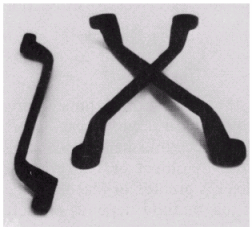
- Regions of an image that are somehow coherent

Why?

- Object extraction, object removal, compositing, etc.
- …but are “blobs” objects?
- No, not in general

Blob’s coherence

Simplest way to define blob coherence is as similarity in brightness or color:



The tools become blobs

The house, grass, and sky make different blobs

The meaning of a blob

Other interpretations of blobs are possible, depending on how you define the input image:

- Image can be a response of a particular detector gx^2+gy^2
  - Color Detector
  - Face detector
  - Motion Detector
  - Edge Detector

Why is this useful?

AIBO RoboSoccer (VelosoLab)

Ideal Segmentation

Result of Segmentation

Thresholding

Basic segmentation operation:

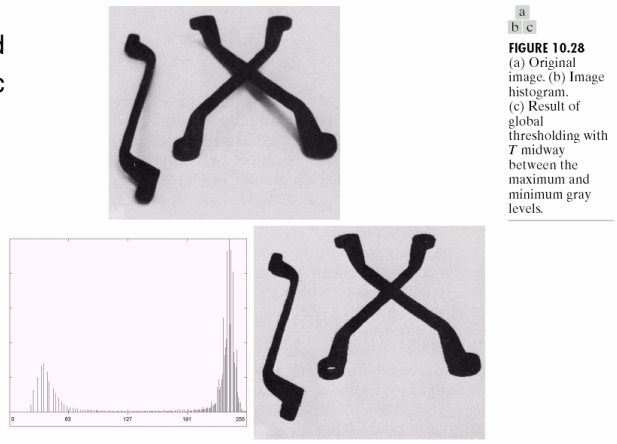
mask(x,y) = 1 if im(x,y) > T

mask(x,y) = 0 if im(x,y) < T

T is threshold

- User-defined
- Or automatic

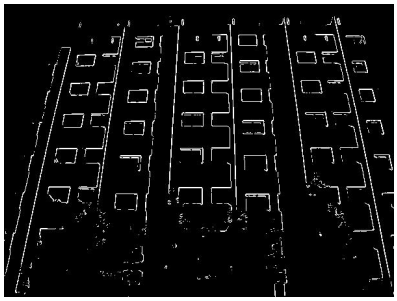
Same as histogram partitioning:



a
b c
FIGURE 10.28
(a) Original image. (b) Image histogram. (c) Result of global thresholding with T midway between the maximum and minimum gray levels.

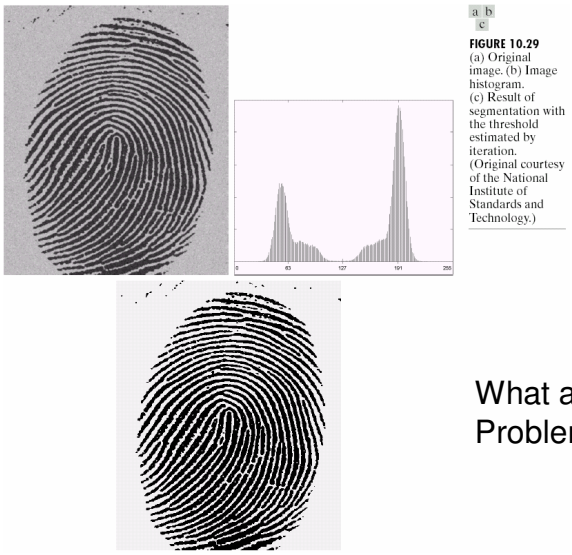
As Edge Detection

gx^2+gy^2



$gx^2+gy^2 > T$

Sometimes works well…



a b
c

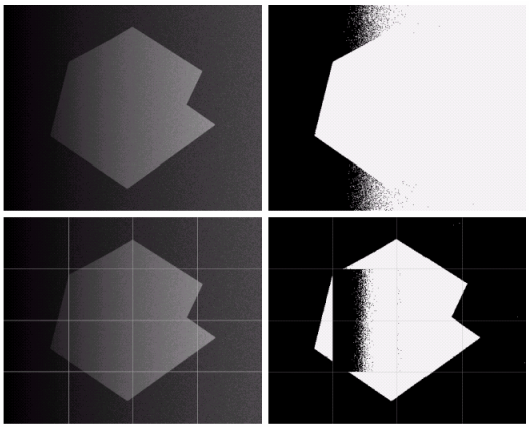
FIGURE 10.29
(a) Original image. (b) Image histogram. (c) Result of segmentation with the threshold estimated by iteration. (Original courtesy of the National Institute of Standards and Technology.)

What are potential Problems?

…but more often not

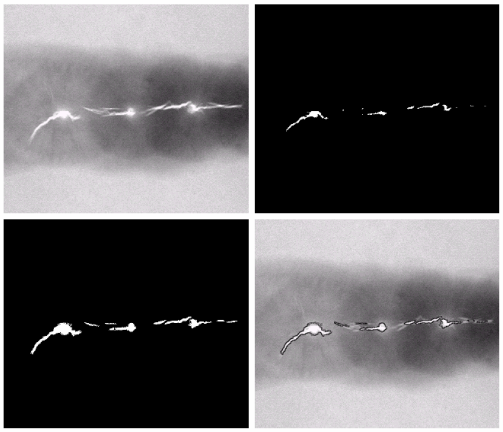
a b
c d

FIGURE 10.30
(a) Original image. (b) Result of global thresholding. (c) Image subdivided into individual subimages. (d) Result of adaptive thresholding.



Adaptive thresholding

Region growing



a b
c d

FIGURE 10.40
(a) Image showing defective welds. (b) Seed points. (c) Result of region growing. (d) Boundaries of segmented defective welds (in black). (Original image courtesy of X-TEK Systems, Ltd.).

- Start with initial set of pixels K
- Add to K any neighbors, if they are within similarity threshold
- Repeat until nothing changes

Is this same as global threshold?

What can go wrong?

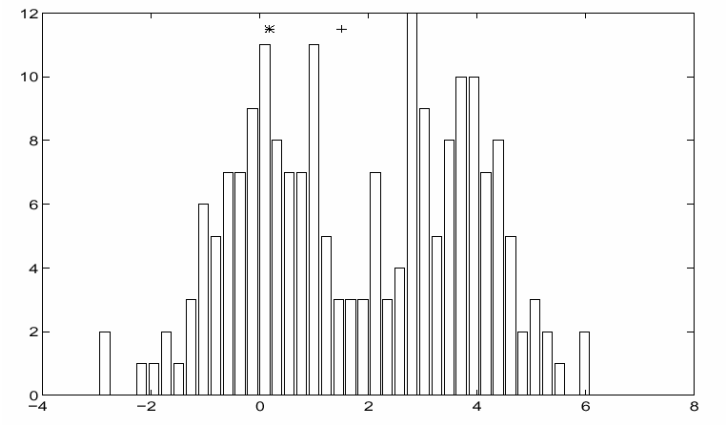
Color-Based Blob Segmentation

Automatic Histogram Partitioning

- Given image with N colors, choose K
- Each of the K colors defines a region
  - not necessarily contiguous
- Performed by computing color histogram, looking for modes

- This is what happens when you downsample image color range, for instance in Photoshop

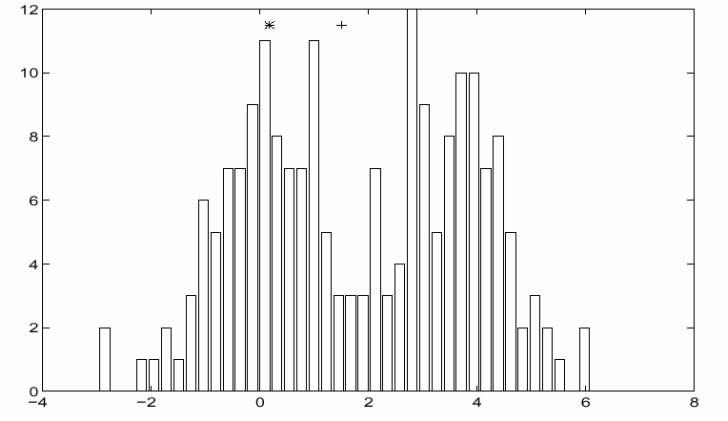
Finding Modes in a Histogram



How Many Modes Are There?

- Easy to see, hard to compute

Mean Shift [Comaniciu & Meer]



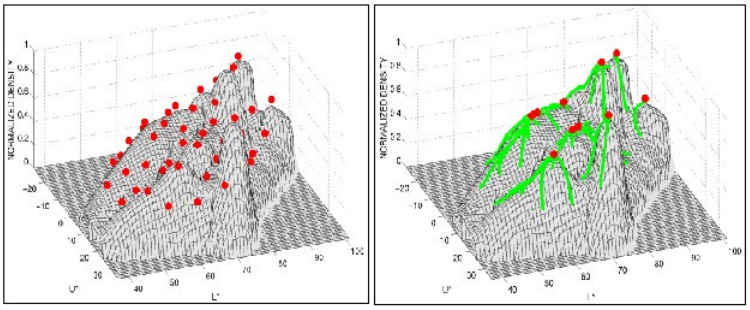
Iterative Mode Search

1. Initialize random seed, and fixed window
2. Calculate center of gravity of the window (the “mean”)
3. Translate the search window to the mean
4. Repeat Step 2 until convergence

Mean-Shift

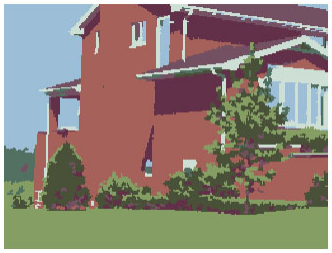
Multimodal Distributions

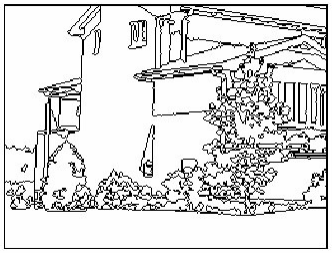
- Parallel processing of an initial tessellation.
- Pruning of mode candidates.
- Classification based on the basin of attraction.



Mean shift trajectories

Mean-shift results





More Examples: http://www.caip.rutgers.edu/~comanici/segm_images.html

Issues:

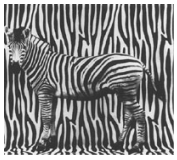
Although often useful, all these approaches work only some of the time, and are considered rather “hacky”.

Can’t even handle our tiger:

Problem is that blobs != objects!

Image Segmentation

Detour into Computer Vision…



"I stand at the window and see a house, trees, sky. Theoretically I might say there were 327 brightnesses and nuances of colour. Do I have "327"? No. I have sky, house, and trees." **--Max Wertheimer**

From Pixels to ***Objects***, not mere “blobs”…

What Defines an Object?

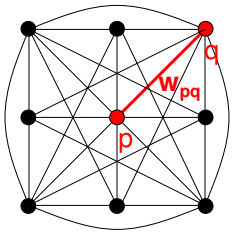
- Subjective problem, but has been well-studied
- Gestalt Laws seek to formalize this:
  - proximity, similarity, continuation, closure, common fate

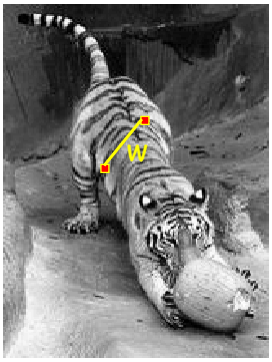
Region-Based Segmentation

We Want Regions

- why not build this in as a constraint?

Images as Graphs [Shi & Malik]

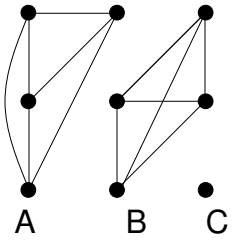




Graph G = (V, E, W)

- node for every pixel
- edge between every pair of pixels, **p**,**q**
- weight $\mathbf{w}_{pq}$ for each edge
  - $\mathbf{w}_{pq}$ measures *similarity*
    - » similarity: difference in color and position (or other things)

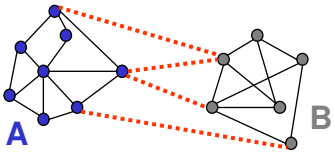
Segmentation by cutting a Graph



Break Graph into Segments

- Delete edges that cross between segments
- Easiest to break edges that have low weight:
  - similar pixels should be in the same segments
  - dissimilar pixels should be in different segments

Cuts in a graph



Edge Cut

- set of edges whose removal makes a graph disconnected
- cost of a cut:

$$cut(A,B) = \sum_{p \in A, q \in B} w_{p,q}$$

Normalized Cut

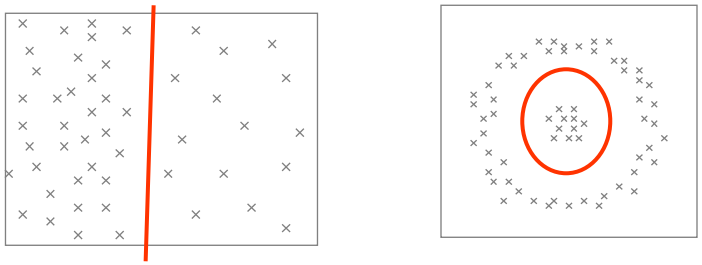
- a min cut penalizes large segments
- fix by normalizing for size of segments

$$Ncut(A,B) = cut(A,B)\left[\underbrace{\frac{1}{\sum_{p \in A} w_{p,q}}}_{\text{Vol(A)}} + \underbrace{\frac{1}{\sum_{q \in B} w_{p,q}}}_{\text{Vol(B)}}\right]$$

The Normalized Cut (NCut) criterion

Given a Graph G = (V, E, W)

- Find A in V that minimizes $Ncut(A, \overline{A})$



NP-Hard!

Normalize Cut in Matrix Form

$\mathbf{W}$ is the weight matrix : $\mathbf{W}(i, j) = w_{i,j}$;

$\mathbf{D}$ is the sum of weights from node i : $\mathbf{D}(i,i) = \sum_j \mathbf{W}(i, j)$;

$\mathbf{x}$ is a vector in $\{1,-1\}^N$, $\mathbf{x}(i) = 1 \Leftrightarrow i \in A$.

$$Ncut(\mathrm{A},\mathrm{B}) = \frac{cut(\mathrm{A},\mathrm{B})}{Vol(\mathrm{A})} + \frac{cut(\mathrm{A},\mathrm{B})}{Vol(\mathrm{B})}$$

$$= \frac{(1+x)^T(\mathbf{D}-\mathbf{W})(1+x)}{k1^T\mathbf{D}1} + \frac{(1-x)^T(\mathbf{D}-\mathbf{W})(1-x)}{(1-k)1^T\mathbf{D}1};\ k = \frac{\sum_{x_i>0}\mathbf{D}(i,i)}{\sum_i \mathbf{D}(i,i)}$$

Eigenvalue Problem

After lot's of math, we get:

$$Ncut(A,B) = \frac{\mathbf{y}^{\mathbf{T}}(\mathbf{D}-\mathbf{W})\mathbf{y}}{\mathbf{y}^{\mathbf{T}}\mathbf{D}\mathbf{y}}, \quad \text{with } \mathbf{y}_i \in \{1, -b\}, \mathbf{y}^T\mathbf{D}\mathbf{1} = 0.$$

This is a Rayleigh Quotient

- Solution given by "generalized" eigenvalue problem:

$$(\mathbf{D}-\mathbf{W})\mathbf{y} = \lambda\mathbf{D}\mathbf{y}$$

- Solved by converting to standard eigenvalue problem:

$$\mathbf{D}^{-\frac{1}{2}}(\mathbf{D}-\mathbf{W})\mathbf{D}^{-\frac{1}{2}}\mathbf{z} = \lambda\mathbf{z}, \quad where\ \mathbf{z} = \mathbf{D}^{\frac{1}{2}}\mathbf{y}$$

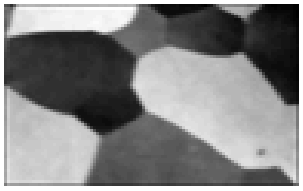
Subtleties

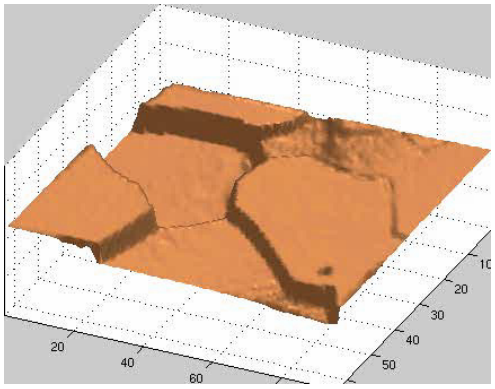
- optimal solution is second smallest eigenvector
- gives real result—must convert into discrete values of y

Interpretation as a Dynamical System

Weights are Springs

- eigenvectors correspond to vibration modes



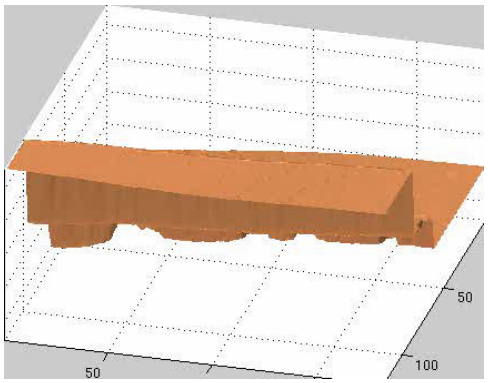


Movie by Serge Belongie

Interpretation as a Dynamical System

Weights are Springs

- eigenvectors correspond to vibration modes



Movie by Serge Belongie

Segmentation result

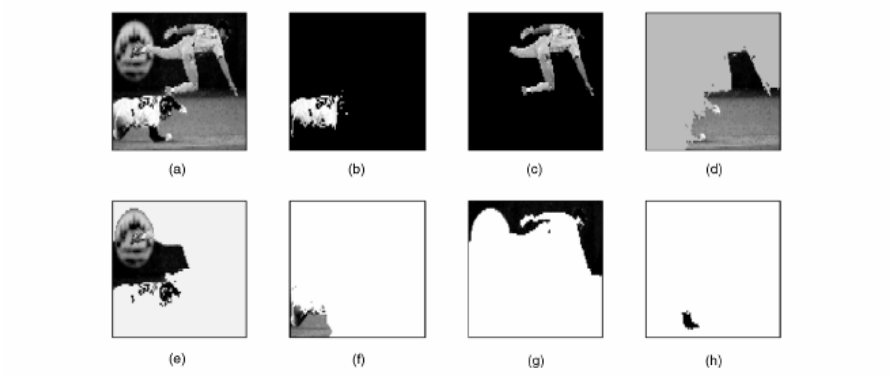
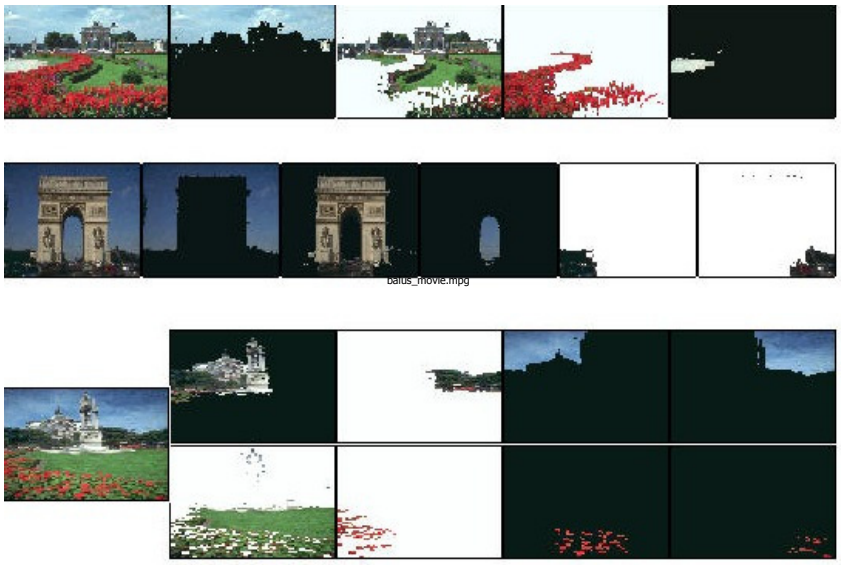


Fig. 4. (a) shows the original image of size 80×100. Image intensity is normalized to lie within 0 and 1. Subplots (b)-(h) show the components of the partition with $Ncut$ value less than 0.04. Parameter setting: $\sigma_I = 0.1$, $\sigma_X = 4.0$, $r = 5$.

Color Image Segmentation



Video Matching

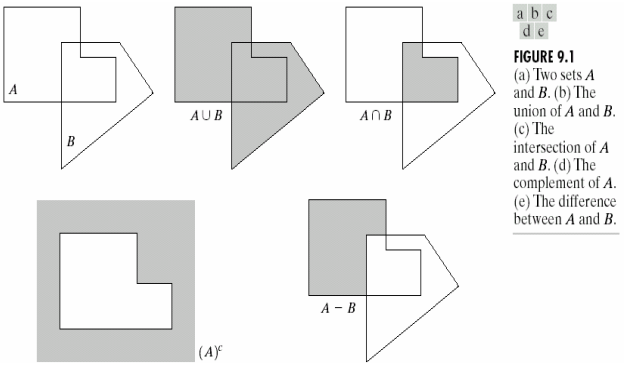
Binary Image Processing

The result of all these operations is a binary mask

Binary images are handy in many cases (sprite extraction, compositing, etc).

Binary image processing is a well-studied field, based on set theory, called Mathematical Morphology

Preliminaries



a b c
d e

FIGURE 9.1 (a) Two sets A and B. (b) The union of A and B. (c) The intersection of A and B. (d) The complement of A. (e) The difference between A and B.

Preliminaries

TABLE 9.1
The three basic logical operations.

p	q	p AND q (also $p \cdot q$)	p OR q (also $p + q$)	NOT (p) (also $\bar{p}$)
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Preliminaries

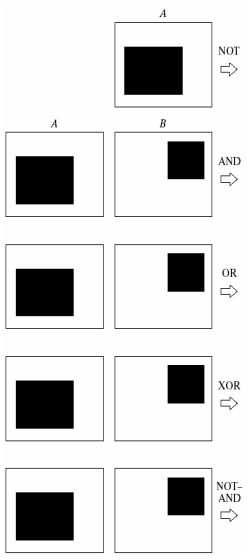
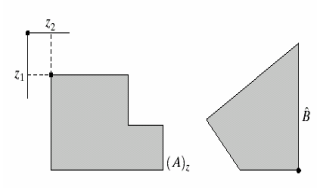


FIGURE 9.3 Some logic operations between binary images. Black represents binary 1s and white binary 0s in this example.

Basic Concepts in Set Theory

A is a set in Z^2, a=(a_1,a_2) an element of A, a$\in$A
If not, then a$\notin$A
$\varnothing$: null (empty) set
Typical set specification: C={w|w=-d, for d $\in$ D}
A subset of B: A$\subseteq$B
Union of A and B: C=A$\cup$B
Intersection of A and B: D=A$\cap$B
Disjoint sets: A$\cap$B= $\varnothing$
Complement of A: $A^c = \{w \mid w \notin A\}$
Difference of A and B: A-B={w|w $\in$ A, w $\notin$ B}= $A \cap B^c$

Preliminaries



a b
FIGURE 9.2
(a) Translation of A by z.
(b) Reflection of B. The sets A and B are from Fig. 9.1.

$$\hat{B} = \{w \mid w = -b, \; for \;\; b \in B\}$$
$$(A)_z = \{c \mid c = a + z, \; for \;\; a \in A\}$$

Dilation and Erosion

0	1	0
1	1	1
0	1	0

Two basic operations:

- A is the image, B is the "structural element", a mask akin to a kernel in convolution

Dilation : $A \oplus B = \{z \mid (\hat{B})_z \cap A \neq \phi\}$

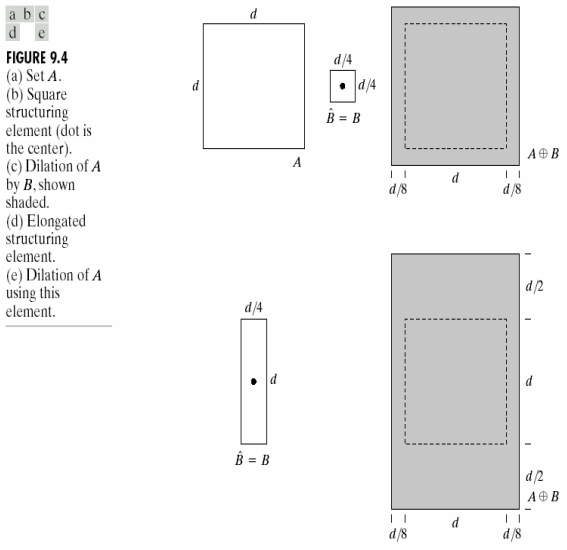
$A \oplus B = \{z \mid [(\hat{B})_z \cap A] \subseteq A\}$

(all shifts of B that have a non-empty overlap with A)

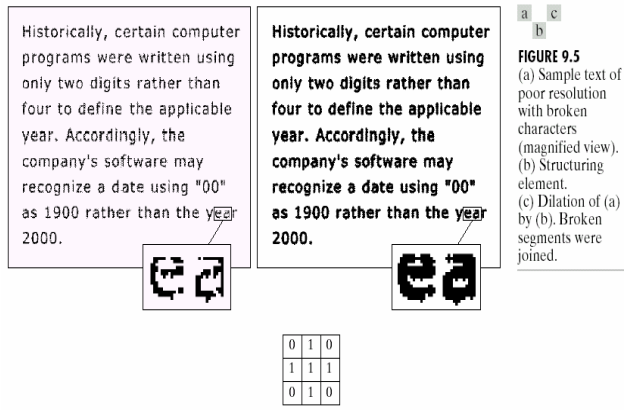
Erosion : $A \Theta B = \{z \mid (B)_z \subseteq A\}$

(all shifts of B that are fully contained within A)

Dilation



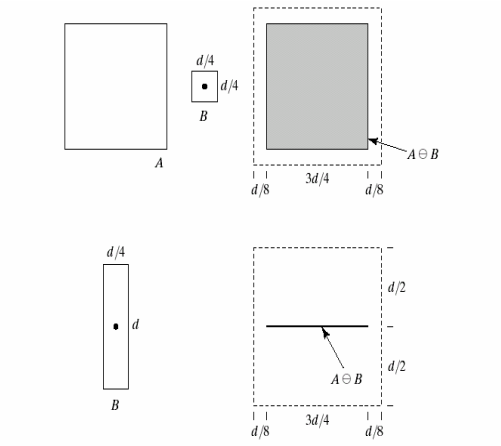
Dilation



a c
b

FIGURE 9.5
(a) Sample text of poor resolution with broken characters (magnified view). (b) Structuring element. (c) Dilation of (a) by (b). Broken segments were joined.

Erosion

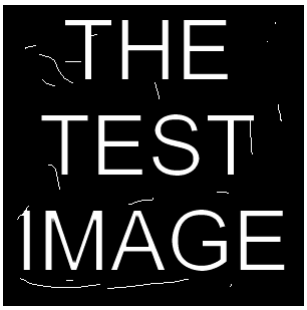


a b c
d e

FIGURE 9.6 (a) Set A. (b) Square structuring element. (c) Erosion of A by B, shown shaded. (d) Elongated structuring element. (e) Erosion of A using this element.

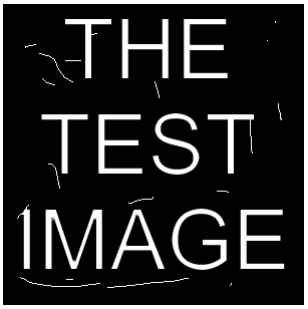
Erosion

Original image



Eroded image

Erosion



Eroded once

Eroded twice

Opening and Closing

Opening : smoothes the contour of an object, breaks narrow isthmuses, and eliminates thin protrusions

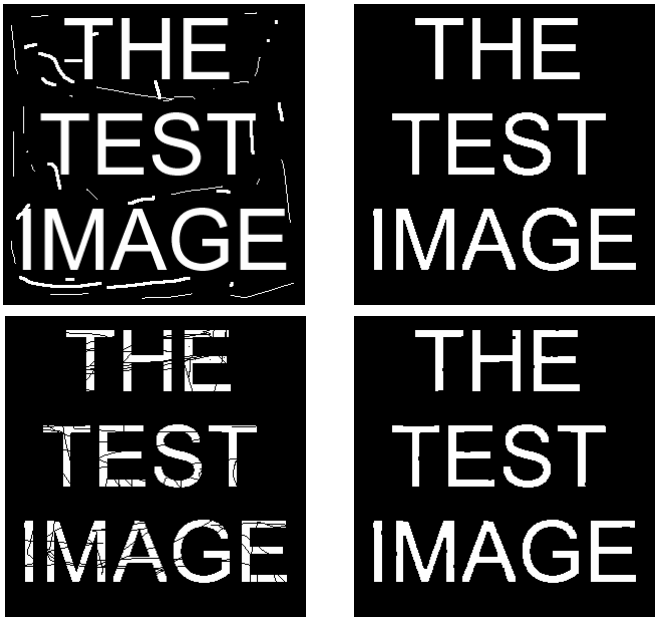
$$A \circ B = (A \Theta B) \oplus B$$

Closing : smooth sections of contours but, as opposed to opning, it generally fuses narrow breaks and long thin gulfs, eliminates small holes, and fills gaps in the contour

$$A \bullet B = (A \oplus B) \Theta B$$

Prove to yourself that they are not the same thing. Play around with `bwmorph` in Matlab.

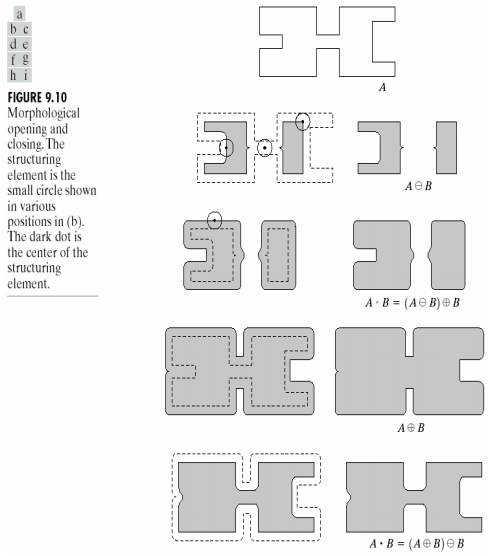
Opening and Closing



OPENING: The original image eroded twice and dilated twice (opened). Most noise is removed

CLOSING: The original image dilated and then eroded. Most holes are filled.

Opening and Closing

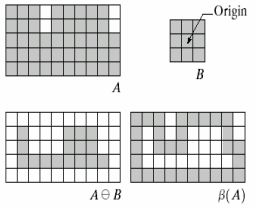


a b c d e f g h i

FIGURE 9.10 Morphological opening and closing. The structuring element is the small circle shown in various positions in (b). The dark dot is the center of the structuring element.

Boundary Extraction

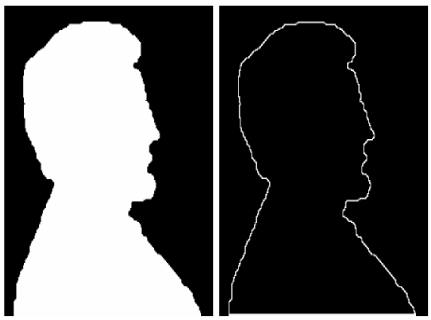
$$\beta(A) = A - (A \Theta B)$$



a b c d

FIGURE 9.13 (a) Set A. (b) Structuring element B. (c) A eroded by B. (d) Boundary, given by the set difference between A and its erosion.

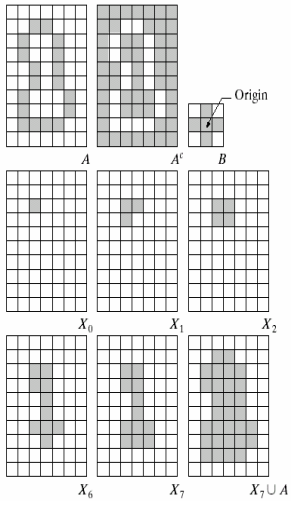
Boundary Extraction



a b

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

Region Filling



a b c
d e f
g h i

FIGURE 9.15
Region filling.
(a) Set A.
(b) Complement of A.
(c) Structuring element B.
(d) Initial point inside the boundary.
(e)–(h) Various steps of Eq. (9.5-2).
(i) Final result [union of (a) and (h)].

$$X_k = (X_{k-1} \oplus B) \bigcap A^c \quad k = 1,2,3....$$

Extraction of Connected Components

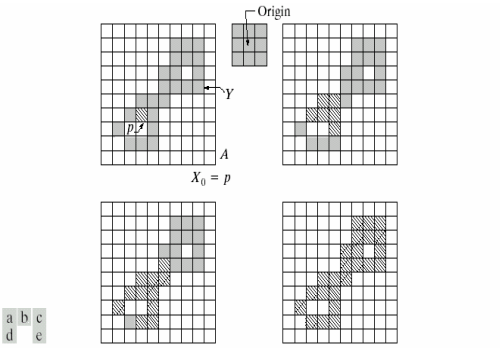


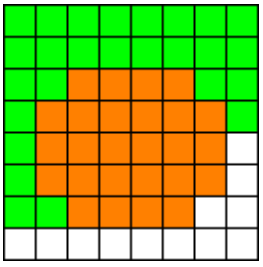
FIGURE 9.17 (a) Set A showing initial point p (all shaded points are valued 1, but are shown different from p to indicate that they have not yet been found by the algorithm). (b) Structuring element. (c) Result of first iterative step. (d) Result of second step. (e) Final result.

$$X_k = (X_{k-1} \oplus B) \bigcap A \quad k = 1,2,3,....$$

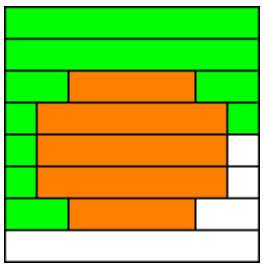
First Step : Run Length Encoding

Segment each image row into groups of similar pixels called *runs*

- Runs store a start and end point for each contiguous row of color

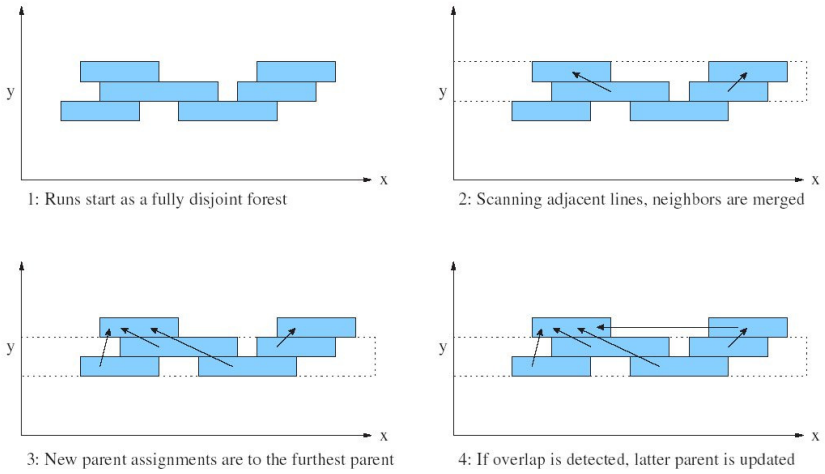


Original image



RLE image

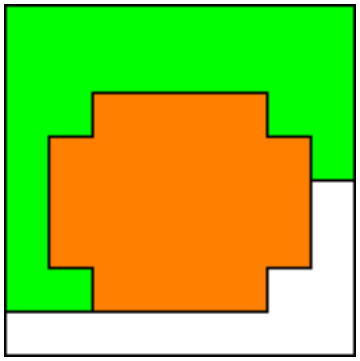
Second Step : Merging Regions



Final Results

Runs are merged into multi-row regions

Image is now described as contiguous regions instead of just pixels



Blob Properties

Now that we have nice, clean blobs, what can we do with them?

Compute Statistics:

- Area
- Perimeter
- Aspect ratio
- Center of mass
- best-fitting ellipse
- Average color
- Etc.

All this can be used to classify blobs and decide if they hold the objects we are interested in.