Mathematical Morphology

- The study of shape...
- Using Set Theory

- Most easily understood for binary images.
Binary Morphology: Basic Idea

1. Make multiple copies of a shape
2. Translate those copies around
3. Combine them with either their:
   - Union, $\cup$, in the case of dilation, $\Theta$
   - Intersection, $\cap$, in the case of erosion, $\Theta$

Dilation makes things bigger
Erosion makes things smaller

Q: How do we designate:
   - The number of copies to make?
   - The translation to apply to each copy?

A: With a structuring element (s.e.)
   - A (typically) small binary image.
   - We will assume the s.e. always contains the origin.

For each marked pixel in the s.e.:
   - Make a new copy of the original image
   - Translate that new copy by the coordinates of the current pixel in the s.e.
Dilation Example

\[ f_A = \{ (2,8),(3,8),(7,8),(8,8),(5,6),(2,4),(3,4), (3,3),(4,3),(5,3),(6,3),(7,3),(7,4),(8,4) \} \]

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

Erosion Example

- For erosion, we translate by the negated coordinates of the current pixel in the s.e.

\[ A = \{ (0,0),(0,-1) \} \]

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

- A (binary) image: $f_A$
- The set of marked pixels in $f_A$: $A$
  - $A = \{ (x_1,y_1), (x_2,y_2), \ldots \}$
- A translated image or set: $f_{A_{(dx,dy)}}$ or $A_{(dx,dy)}$
- The number of elements in $A$: $\#A$
- Complement (inverse) of $A$: $A^c$
- Reflection (rotation) of $A$: $\bar{A}$
  - $\bar{A} = \{ (-x,-y) \mid (x,y) \in A \}$

![Image of A and its complement]

Properties

- Dilation:
  - Commutative, Associative, & Distributive
  - Increasing: If $A \subseteq B$ then $A \oplus K \subseteq B \oplus K$
  - Extensive: $A \subseteq A \oplus B$
- Erosion:
  - Anti-extensive ($A \ominus B \subseteq A$), ... (see the text)
- Duality:
  - $(A \ominus B)^c = A^c \ominus \bar{B}$
  - $(A \oplus B)^c = A^c \ominus \bar{B}$
- Not Inverses:
  - $A \neq (A \ominus B) \oplus B$
  - $A \neq (A \oplus B) \ominus B$

![Diagram showing opening and closing of a set]
Opening

- \( f_A \circ f_B = (f_A \Theta f_B) \Theta f_B \)
- Preserves the geometry of objects that are “big enough”
- Erases smaller objects
- Mental Concept:
  - “Pick up” the s.e. and place it in \( f_A \).
  - Never place the s.e. anywhere it covers any pixels in \( f_A \) that are not marked.
  - \( f_A \circ f_B \) = the set of (marked) pixels in \( f_A \) which can be covered by the s.e.

Opening Example

- Use a horizontal s.e. to remove 1-pixel thick vertical structures:

![Erosion Example](image)

Erosion -> Dilation
Gray-Scale Morphology

- Morphology operates on sets
- Binary images are just a set of marked pixels
- Gray-scale images contain more information
- How can we apply morphology to this extra intensity information?
- We need to somehow represent intensity as elements of a set

The Umbra

- Gray-scale morphology operates on the umbra of an image.
- Imagine a 2D image as a pixilated surface in 3D
- We can also “pixilate” the height of that surface
- The 2D image is now a 3D surface made of 3D cells
The Distance Transform (DT)

- Records at each pixel the distance from that pixel to the nearest boundary (or to some other feature).
- Used by other algorithms
- The DT is a solution of the Diff. Eq.:
  \[ |\nabla DT(x)| = 1, \]
  \[ DT(x) = 0 \text{ on boundary} \]
- Can compute using erosion
  - \( DT(x) = \text{iteration when } x \text{ disappears} \)
  - Details in the book

\[
\begin{array}{cccccccc}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 2 & 2 & 1 & 2 & 2 & 1 & 2 \\
1 & 2 & 2 & 3 & 2 & 2 & 1 & 2 \\
1 & 2 & 2 & 1 & 2 & 1 & 2 & 1 \\
1 & 2 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\end{array}
\]

DT of a region’s interior

Voronoi Diagram

- Divides space
- Related to DT
- Q: To which of a set of regions (or points) is this point the closest?
- Voronoi Diagram’s boundaries = points that are equi-distant from multiple regions
- Voronoi Domain of a region = the “cell” of the Voronoi Diagram that contains the region
- Details in the text

The voronoi diagram of a set of 10 points is public domain from:
http://en.wikipedia.org/wiki/File:2Ddim-l2norm-10site.png
Imaging Matching (ch. 13)

- Matching iconic images
- Matching graph-theoretic representations

- Most important:
  - Eigenimages
  - Springs & Templates

Template Matching

- Template ≈ a relatively small reference image for some feature we expect to see in our input image.
- Typical usage: Move the template around the input image, looking for where it “matches” the best (has the highest correlation).
- Rotation & scale can be problematic
  - Often require multiple passes if they can’t be ruled out a-priori
- How “big” do we make each template?
  - Do we represent small, simple features
  - Or medium-size, more complex structures?
Eigenimages

- **Goal:** Identify an image by comparing it to a database of other images
- **Problem:** Pixel-by-pixel comparisons are too expensive to run across a large database
- **Solution:** Use PCA

**PCA (K-L Expansion)**

- **Big Picture:** Fitting a hyper-ellipsoid & then (typically) reducing dimensionality by flattening the shortest axes
- Same as fitting an (N+1)-dimensional multivariate Gaussian, and then taking the level set corresponding to one standard deviation
- Mathematically, PCA reduces the dimensionality of data by mapping it to the first n eigenvectors (principal components) of the data’s covariance matrix
- The first principal component is the eigenvector with the largest eigenvalue and corresponds to the longest axis of the ellipsoid
- The variance along an eigenvector is exactly the eigenvector’s eigenvalue
- This is VERY important and VERY useful. Any questions?
Eigenimages: Procedure

- Run PCA on the training images
  - See the text for efficiency details
- Store in the database:
  - The set of dominant Eigenvectors
    - = the principle components
    - = the Eigenimages
  - For each image, store its coefficients when projected onto the Eigenimages
- Match a new image:
  - Project it onto the basis of the Eigenimages
  - Compare the resulting coefficients to those stored in the database.

Eigenimages Example

The face database and the derived Eigenface examples are all from AT&T Laboratories Cambridge:

Which training image(s) does each face most resemble?
Matching Simple Features

- Classification based on features
  - Ex: mean intensity, area, aspect ratio
- Idea:
  - Combine a set of shape features into a single feature vector
  - Build a statistical model of this feature vector between and across object classes in a sequence of training shapes
  - Classification of a new shape = the object class from which the new shape’s feature vector most likely came.

Graph Matching: Association Graphs

- Match nodes of model to segmented patches in image
- Maximal cliques represent the most likely correspondences
  - Clique = a totally connected subgraph
- Problems: Over/under segmentation, how to develop appropriate rules, often > 1 maximal clique
Graph Matching: Springs & Templates

- Idea: When matching simple templates, we usually expect a certain arrangement between them.
- So, arrange templates using a graph structure.
- The springs are allowed to deform, but only “so” much.

Fischler and Elschlager’s “Pictorial Structures” spring & template model for image matching from the early 1970s

Graph Matching: Springs & Templates

- A match is based on minimizing a total cost.
- Problem: Making sure missing a point doesn’t improve the score.

\[
\text{Cost} = \sum_{d \in \text{templates}} \text{TemplateCost} \left( d, F(d) \right) + \sum_{d \in \text{ref} \times \text{ref}} \text{SpringCost} \left( F(d), F(e) \right) + \sum_{c \in \text{R}_{\text{source}} \times \text{R}_{\text{source}}} \text{MissingCost} \left( c \right)
\]