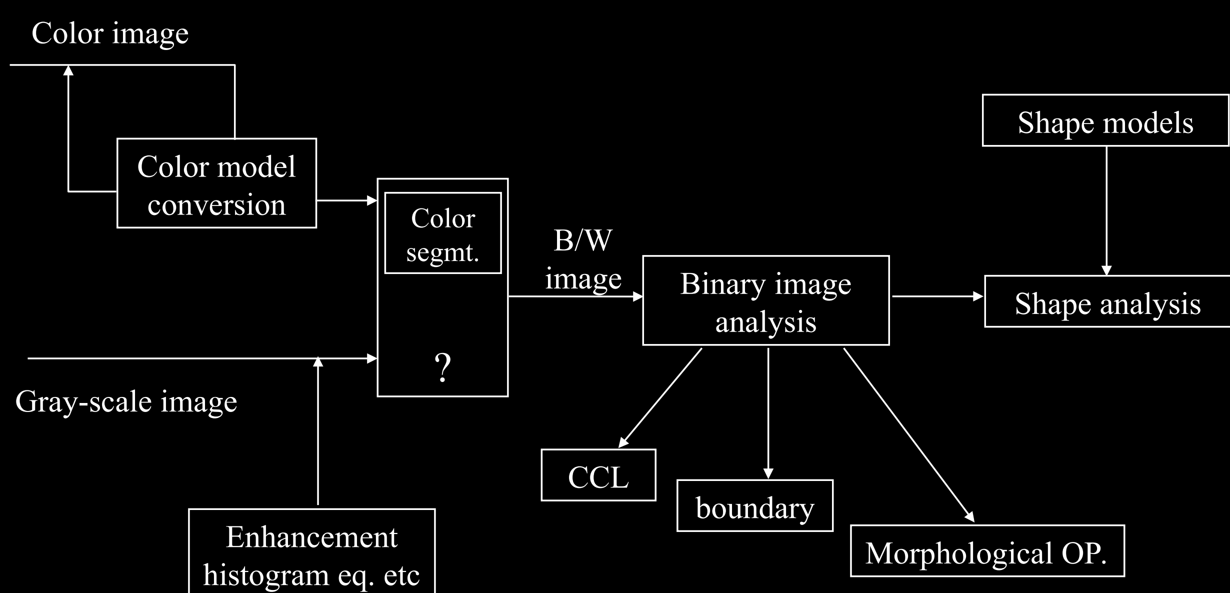
Image Segmentation

Ying Wu

Electrical Engineering and Computer Science
Northwestern University
Evanston, IL 60208

http://www.ece.northwestern.edu/~yingwu
yingwu@ece.northwestern.edu

Where are we?



Outline

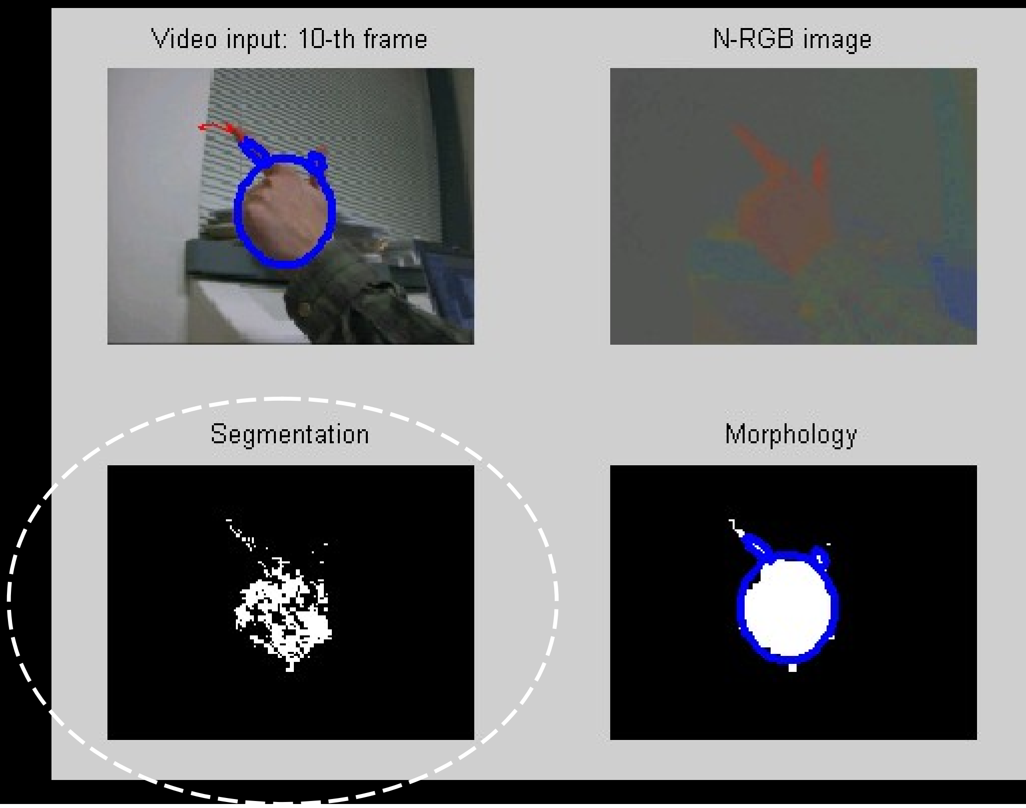
- Motivation
- Region-based representations
  - Array representation
  - Hierarchical representation
  - Symbolic representation
  - RAG (region adjacency graph)
- Region-based segmentation
  - Region-growing
  - Region-splitting
- The “watershed” segmentation algorithm

A simple approach

- We’ve learnt the histogram-based thresholding method for segmentation
  - E.g., color segmentation
- Advantages:
  - Fast
  - Easy to implement
- Disadvantages
  - Difficult to find the best thresholds
- What is missing?

  It throws away the spatial information among the pixels!

Motivation

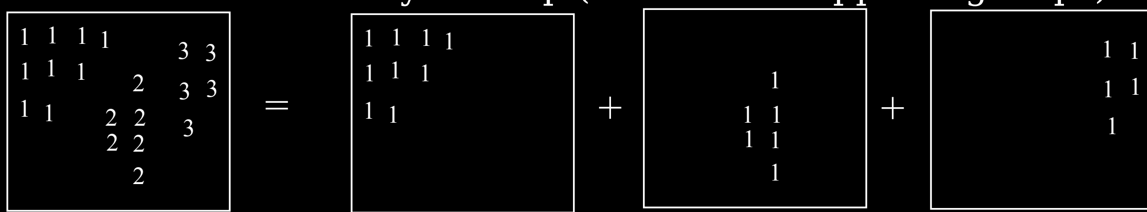


How do we represent a region?

- Let's discuss

Array representation

- A region is represented by a region mask (an array)
- Hard masks vs. soft masks
  - Soft mask are weighted masks
  - Soft masks may overlap (also called supporting maps)

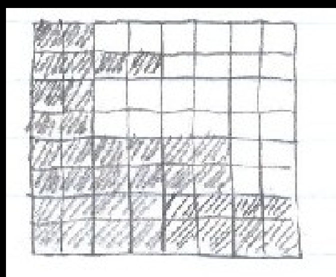


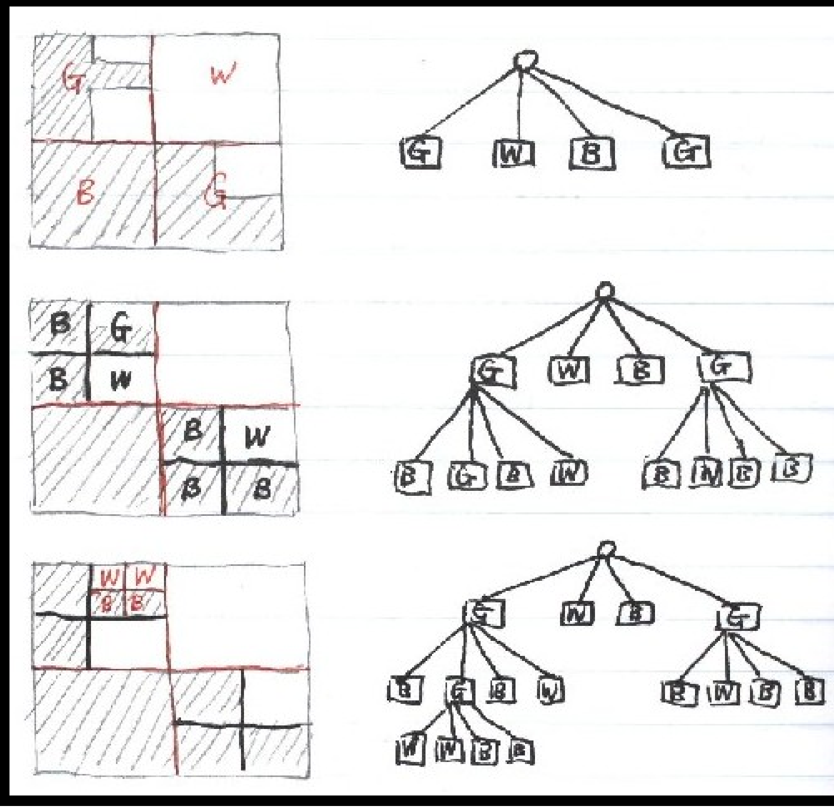
- Advantages:
  fast, hardware support → widely used
- Disadvantages:
  low-level,
  not much geometric information (e.g,who is close to whom?)

Hierarchical Representation

- Motivation:
  - Human visual perception is quite complex. It is still a mysterious thing for scientific research.
  - E.g.,

Quad tree



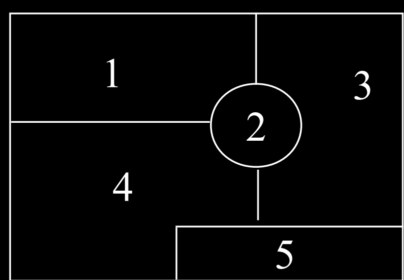


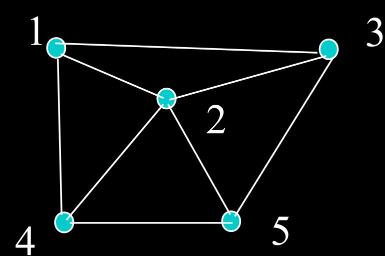
Symbolic representation

- Describing the characteristics of a region
- Symbolic characteristics
  - Shape
    - ✓ Boundary box
    - ✓ Centroid
    - ✓ Moments
    - ✓ Euler number
  - Intensity
    - ✓ Mean/variance of the intensities
    - ✓ Approximated intensity surfaces

RAG

- The relations among regions
- Region adjacency graph (RAG)
  - A vertex → a region
  - A edge → the adjacency
    - ✓ if the two regions adjuncts, connect the two nodes





Formulation of Segmentation

- A process that partitions image R into subregions $\{R_1, R_2, \ldots, R_n\}$, s.t.,

$$
\begin{array}{ll}
(a) & \bigcup_{k=1}^{n} R_k = R \\
(b) & R_k \text{ is a connected region} \\
(c) & R_i \cap R_j = \varphi, \quad \forall i \neq j \\
(d) & P(R_k) = \text{true}, \quad \forall k \\
(e) & P(R_i \cup R_j) = \text{false}, \quad \forall i \neq j \\
 & \text{where, } P(\cdot) \text{ is a logical predicate}
\end{array}
$$

Thoughts

- The thresholding method is not good for segmentation, since
  - It is pixel-based, not region-based, no connectivity
  - Optimal thresholds may not exist
- So?
- Using regions/connectivity/neighborhood

- Basic idea:

  If a pixel $x \in R_k$,

  Check its neighbor N(x)

  If P(N(x))=true,

  then N(x) should also be in R_k.

Region Growing

- Basic idea
  - Group/grow pixels of subregions into larger regions based on predefined criteria.
- Grouping pixels
  - Selecting seeds
  - Selecting a merging criterion (this is critical)
    - ✓ intensity?
    - ✓ color?
    - ✓ texture?
    - ✓ …?
  - The stop criterion
- Grouping subregions
  - Constructing RAG
  - Merging criteria?

A formal method

- There are two sets of pixels from two regions
  - R1, $\{x^1_1, x^1_2, \ldots, x^1_n\}$
  - R2, $\{x^2_1, x^2_2, \ldots, x^2_m\}$
- For two regions, we need to test two hypotheses
  - H0: they belong to the same region, i.e., the intensities are drawn from a Gaussian density, $N(\mu_0, \sigma_0)$
  - H1: they don't, and the intensities are drawn from two Gaussian densities: $R_1 \sim N(\mu_1, \sigma_1)$, and $R_2 \sim N(\mu_2, \sigma_2)$.
- Which hypothesis is more likely?
  - Check the data likelihood

Hypothesis testing

$$p(g_1^1, \cdots, g_n^1, g_1^2, \cdots, g_m^2 | H_0) = \prod_{k=1}^{n+m} p(g_k | H_0) = \frac{1}{(\sqrt{2\pi}\sigma_0)^{n+m}} e^{-\frac{(m+n)}{2}}$$

$$p(g_1^1, \cdots, g_n^1, g_1^2, \cdots, g_m^2 | H_1) = \frac{1}{(\sqrt{2\pi}\sigma_1)^{n}} e^{-\frac{n}{2}} \frac{1}{(\sqrt{2\pi}\sigma_2)^{m}} e^{-\frac{m}{2}}$$

$$\Rightarrow \quad L = \frac{p(g_1^1, \cdots, g_n^1, g_1^2, \cdots, g_m^2 | H_1)}{p(g_1^1, \cdots, g_n^1, g_1^2, \cdots, g_m^2 | H_0)} = \frac{\sigma_0^{n+m}}{\sigma_1^n \sigma_2^m}$$

Likelihood ratio

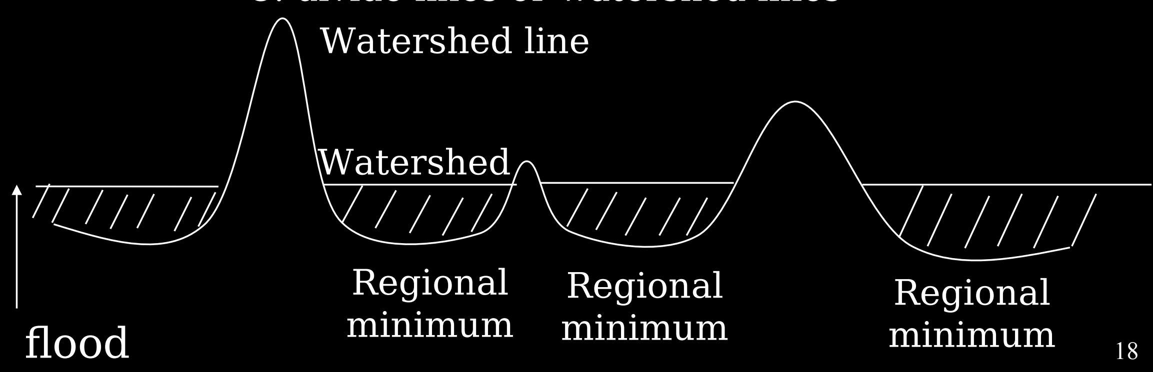
$$\Rightarrow \quad \begin{array}{lll} \text{if} \quad L > t, & \text{we accept } H_1 \\ \quad \text{else} & \text{we accept } H_0 \end{array}$$

Region Splitting

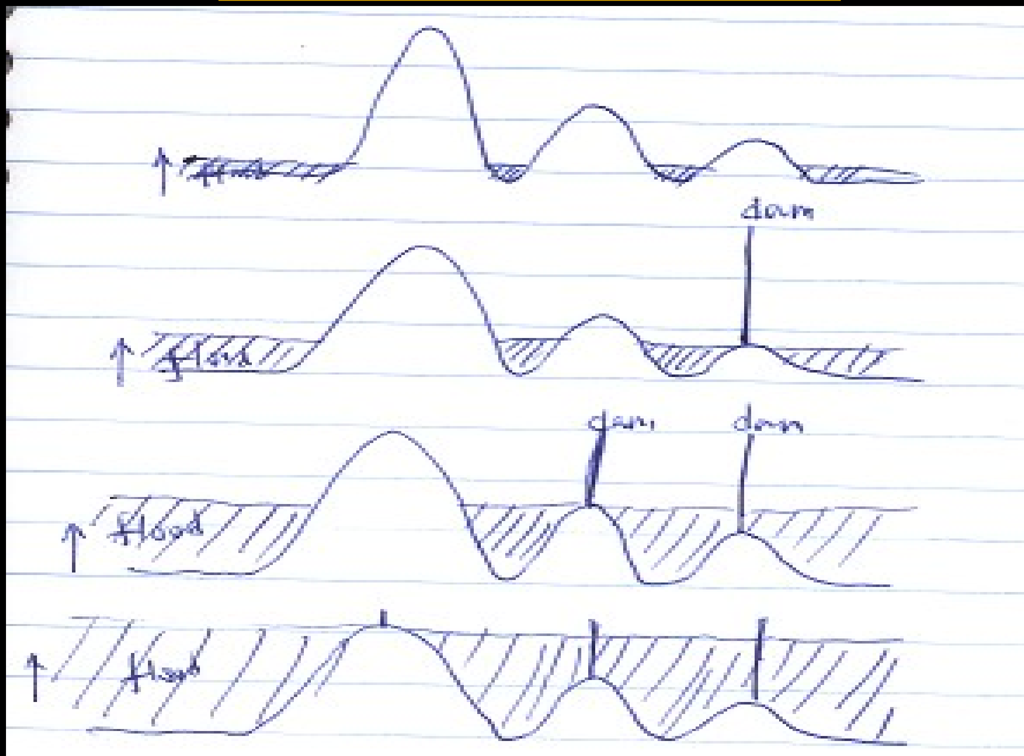
- Region growing is from bottom-up
- Region splitting is from top-down
- The quad tree is a simple example
- Splitting&Merging Algorithm
  - Split region into 4 disjoint quadrants using quad tree algorithm, s.t. $\forall R_k$, $P(R_k)$=true;
  - Merge any adjacent region R_i and R_k, if $P(R_i \cup R_k)$=true;
  - Stop when no further merging or splitting is possible

Watershed Segmentation

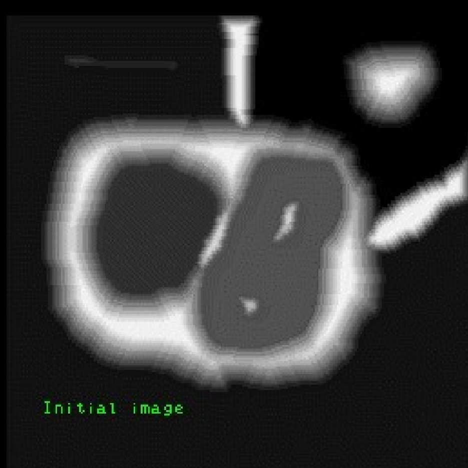
- Basic idea
  - We can visualize an image in 3-D (topographic surface)
  - We have
    - ✓A. regional minimum
    - ✓B. catchment basin or watershed
    - ✓C. divide lines or watershed lines



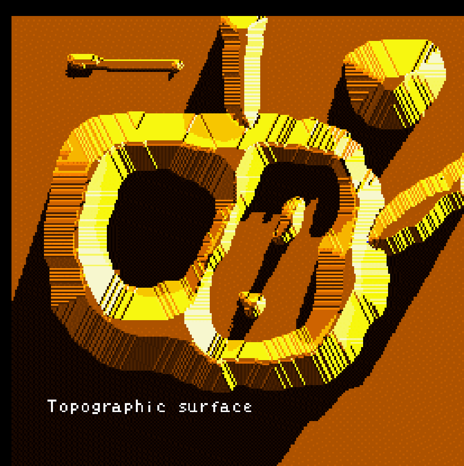
Flooding



3D illustration

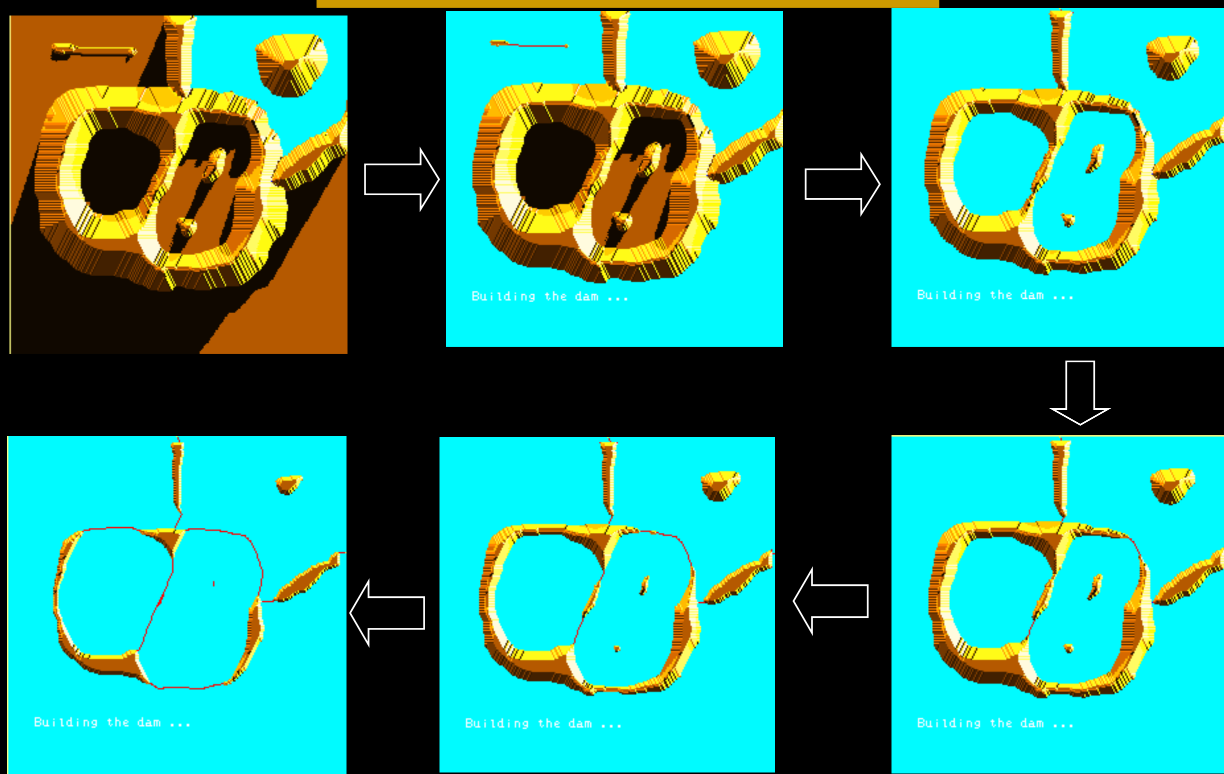


Original image

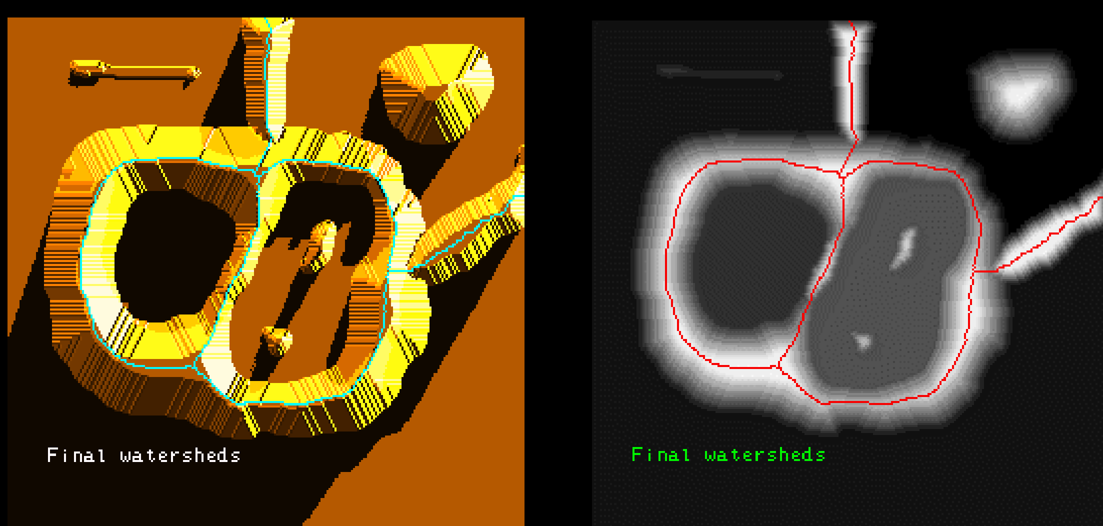


Topographic view

Flooding



Results



A trick

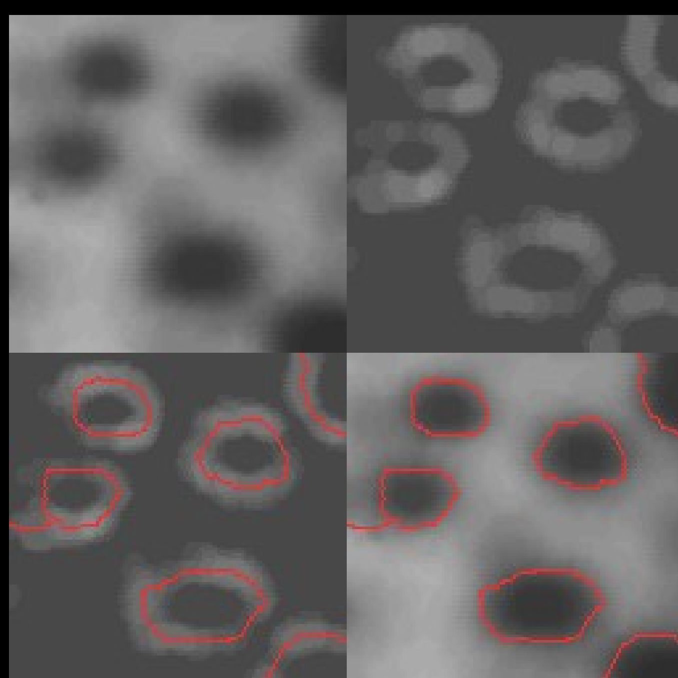
- How can we use this idea?
- Can we perform it directly on the original image?

  - No!
  - We need to perform it on image gradient, i.e., the change of image intensities

A real example

Original image

Image gradient



Watershed lines

Watershed lines

A Problem

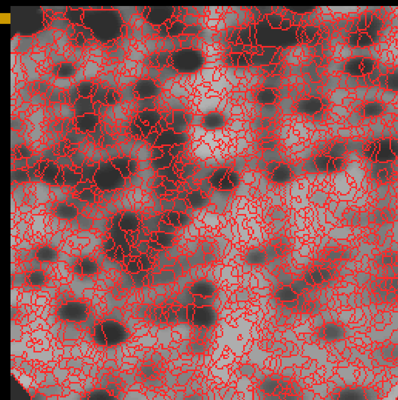
- Watershed may end up with too many regions, called over-segmentation
  - Since the noise in image produces too many regional minima
  - While each regional minimum corresponds to a region
- Solution?
  - We can smooth the image to reduce the noise
  - Then use masks to specify the minima

A solution

- Smooth image (pre-processing)
- Use threshold to get the markers
  - A marker is a region that is surrounded by points of higher altitude
  - S.t., points in the region form a connect component
  - All the points in the component have the same intensity
- Apply watershed to the smoothed image

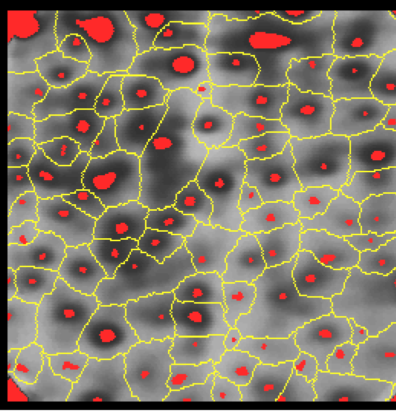
An example

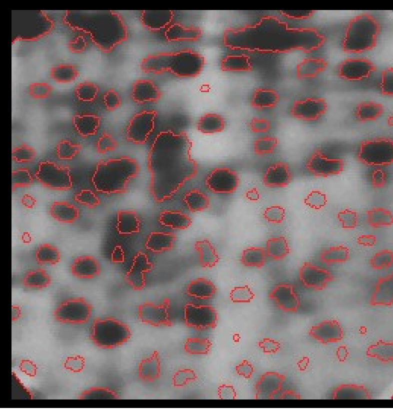
Electrophoresis image



Watershed result (oversegment)

Initial markers





Final result

Segmentation by Graph Cut

- Represent the set of data $\{X_1,\dots,X_n\}$ by a graph $G=\{V,E\}$
- Each vertex: an individual data point
- Each edge: the adjacency of two data points
- Edge weight: the affinity of the two points
  - e.g.

$$A_{ij} = \exp\left\{-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma^2}\right\}$$

- Thus, the data set can be viewed as a weight adjacency graph
- More importantly, it can also be viewed as an affinity matrix A.

Block-diagonalization: Idea

- If the data are grouped, then the affinity matrix is pretty much block-diagonalized
- Now, clustering can be treated as the task of finding the best re-permutation to block-diagonalize A
- More specifically, the summation of the affinity values of those off-diagonal block matrices is minimized
- Or the sum of diagonal block matrices is maximized

Formulation

- Introduce an association vector (i.e., a projection) for each cluster component $\mathbf{w}_k$,

  $$\mathbf{w}_k = [w_{k1}, \; w_{k2}, \ldots, w_{kn}]^T$$

  where w_{ki} is the association of x_i to the cluster k
- Positive w_{ki} indicates that x_i is in cluster k to some extent, and negative otherwise
- Usually, such projection vector is normalized, or

  $$\mathbf{w}_k^T \mathbf{w}_k = 1, \quad \forall k = 1, \ldots, K$$
- Now, we can formulate the problem as:

  $$\begin{aligned} \mathbf{w}_k^* &= \arg\max_{w_k} \; \mathbf{w}_k^T \mathbf{A} \mathbf{w}_k \\ s.t. & \quad \mathbf{w}_k^T \mathbf{w}_k = 1 \end{aligned}$$

Solution

- The solution is easy
- Let's see the Lagrangian

$$L = \mathbf{w}_k^T \mathbf{A} \mathbf{w}_k + \lambda(1 - \mathbf{w}_k^T \mathbf{w}_k)$$

- It is clear that

$$\frac{\partial L}{\partial \mathbf{w}_k} = 2\mathbf{A}\mathbf{w}_k - 2\lambda \mathbf{w}_k = 0 \quad \Rightarrow \mathbf{A}\mathbf{w}_k = \lambda \mathbf{w}_k$$

- w_k, an eigenvector, indicates the association of data with cluster k

An Issue

- Ideally, we can check the values of w_{ki} for grouping
- But there is an complication
- Suppose A has two identical (repeated) eigenvalues

$$\mathbf{A}\mathbf{w}_1 = \lambda \mathbf{w}_1, \quad \textit{and } \mathbf{A}\mathbf{w}_2 = \lambda \mathbf{w}_2$$

- It is easy to see that any linear combination of w_1 and w_2 also gives a valid eigenvector
- $$\mathbf{A}(a_1\mathbf{w}_1 + a_2\mathbf{w}_2) = \lambda(a_1\mathbf{w}_1 + a_2\mathbf{w}_2)$$
- This means that we cannot simply use the values of $w=a_1w_1+a_2w_2$ for grouping
- Instead of using the 1-D subspace, we need to go to the 2D subspace spanned by $\{w_1,w_2\}$