EE 604
Digital Image Processing

IIT KANPUR
Indian Institute of Technology, Kanpur

Tanaya Guha
Aug - Nov 2017

Clustering

K-means clustering

- One of the most popular clustering algorithms.
- Originally developed for quantization in signal processing.
  - The standard algorithm was used by Llyod 1957
- Related to expectation-maximization(EM) algorithms

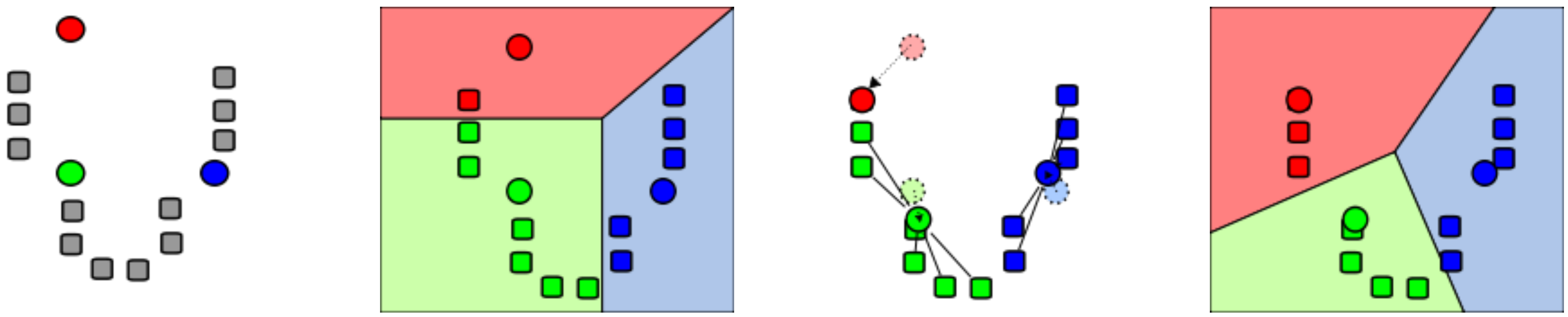
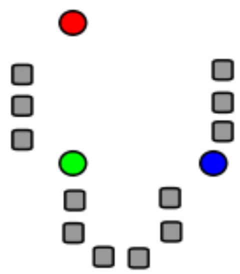


Image from Wikipedia

K-means clustering

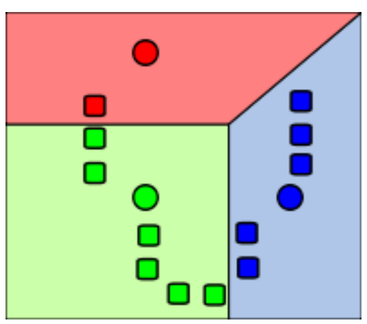
- **Input:** $x^{(1)}, x^{(2)}, ..., x^{(n)}$
- **Output:** Set of clusters $C_1, C_2, ...C_k$
- **Initialization:** Randomly pick ***k*** centroids $z^{(1)}, z^{(2)}, ..., z^{(k)}$



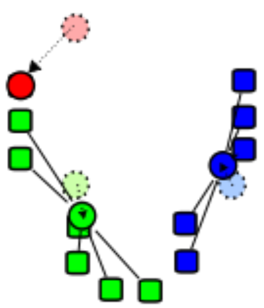
- **Itereate** until convergence
  - **Assignment:** Assign each point to its closest centroid

    for each $j = 1, .., k$

    $C_j = \{i | \text{s.t. } x^{(i)} \text{ is closest to } z^{(j)}\}$



  - **Update:** Recompute centroids with newly assigned points



$$z^{(j)} = \frac{1}{|C_j|} \sum_{i \in C_j} x^{(i)}$$

K-means clustering

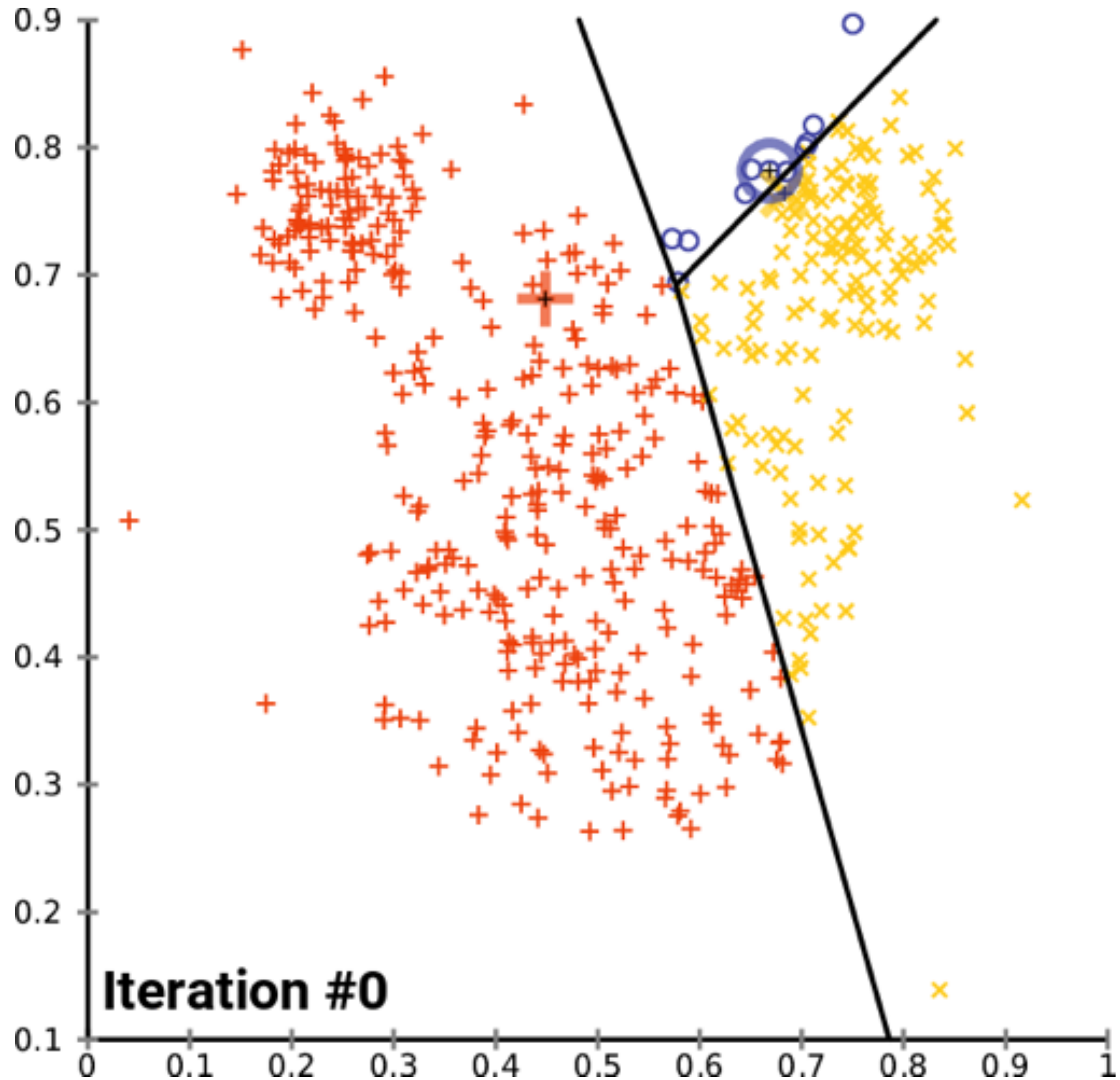


Image from Wikipedia

Properties of k-means

- Guaranteed to converge within a finite number of iterations.
- Cost function:

$$\min_{z^{(1)},\ldots,z^{(k)}} \min_{C_1,..C_k} \sum_{j=1}^{k} \sum_{i \in C_j} \|x^{(i)} - z^{(j)}\|^2$$

- **Assignment:** Fix z, optimize for C

$$\min_{C_1,..C_k} \sum_{j=1}^{k} \sum_{i \in C_j} \|x^{(i)} - z^{(j)}\|^2 = \sum_{i=1}^{n} \min_{j=1:k} \|x^{(i)} - z^{(j)}\|^2$$

Properties of k-means

- Guaranteed to converge within a finite number of iterations.
- Cost function:

$$\min_{z^{(1)},\ldots,z^{(k)}} \min_{C_1,..C_k} \sum_{j=1}^{k} \sum_{i \in C_j} \|x^{(i)} - z^{(j)}\|^2$$

- **Update:** Fix C, optimize for z

$$\min_{z^{(1)},\ldots,z^{(k)}} \sum_{j=1}^{k} \sum_{i \in C_j} \|x^{(i)} - z^{(j)}\|^2$$

- Take partial derivative w.r.t. $z^{(j)}$ and set to 0

$$z^{(j)} = \frac{1}{|C_j|} \sum_{i \in C_j} x^{(i)}$$

Properties of k-means

- Connections to well-known optimization methods
  - An alternate minimization approach
  - Can also be cast as a **gradient descent** problem

- At each iteration, the error reduces.

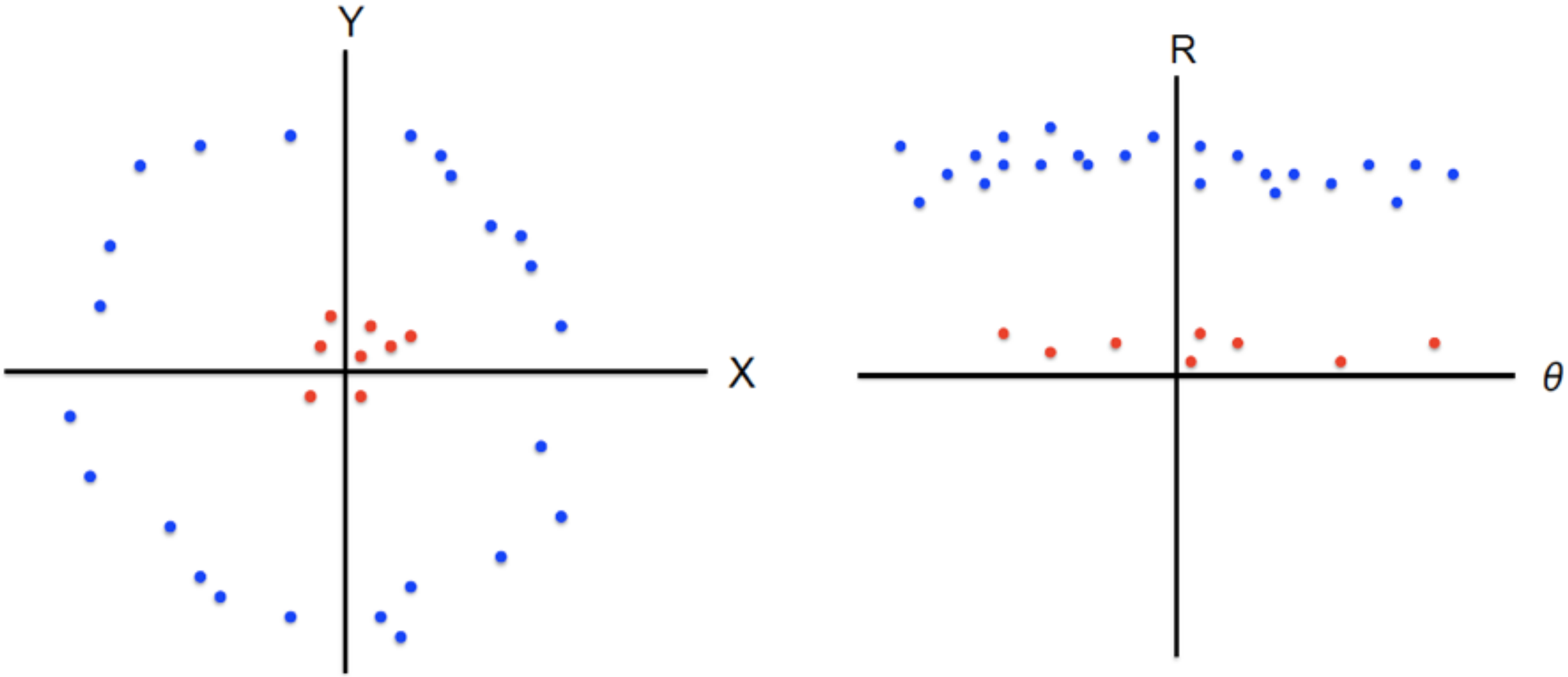
- Guaranteed to converge, but no guarantee that the algorithm will converge to a global minima.

- Complexity per iteration
  - Assignment step: $\boldsymbol{O(kn)}$
  - Update step: $\boldsymbol{O(n)}$

K-means clustering

- How to choose the initial points?
  - Smartly choose the initial points
  - Run multiple times and choose the best result
- How to choose ***k*?**
  - Usually unknown and difficult
  - Use ***k*** that minimizes Bayesian information criterion (BIC) or Akaike information criterion (AIC)
- The similarity function matters
  - Euclidean, cosine similarity are common choices
  - Other distances can also help
- If in the feature domain, the choice of feature matters

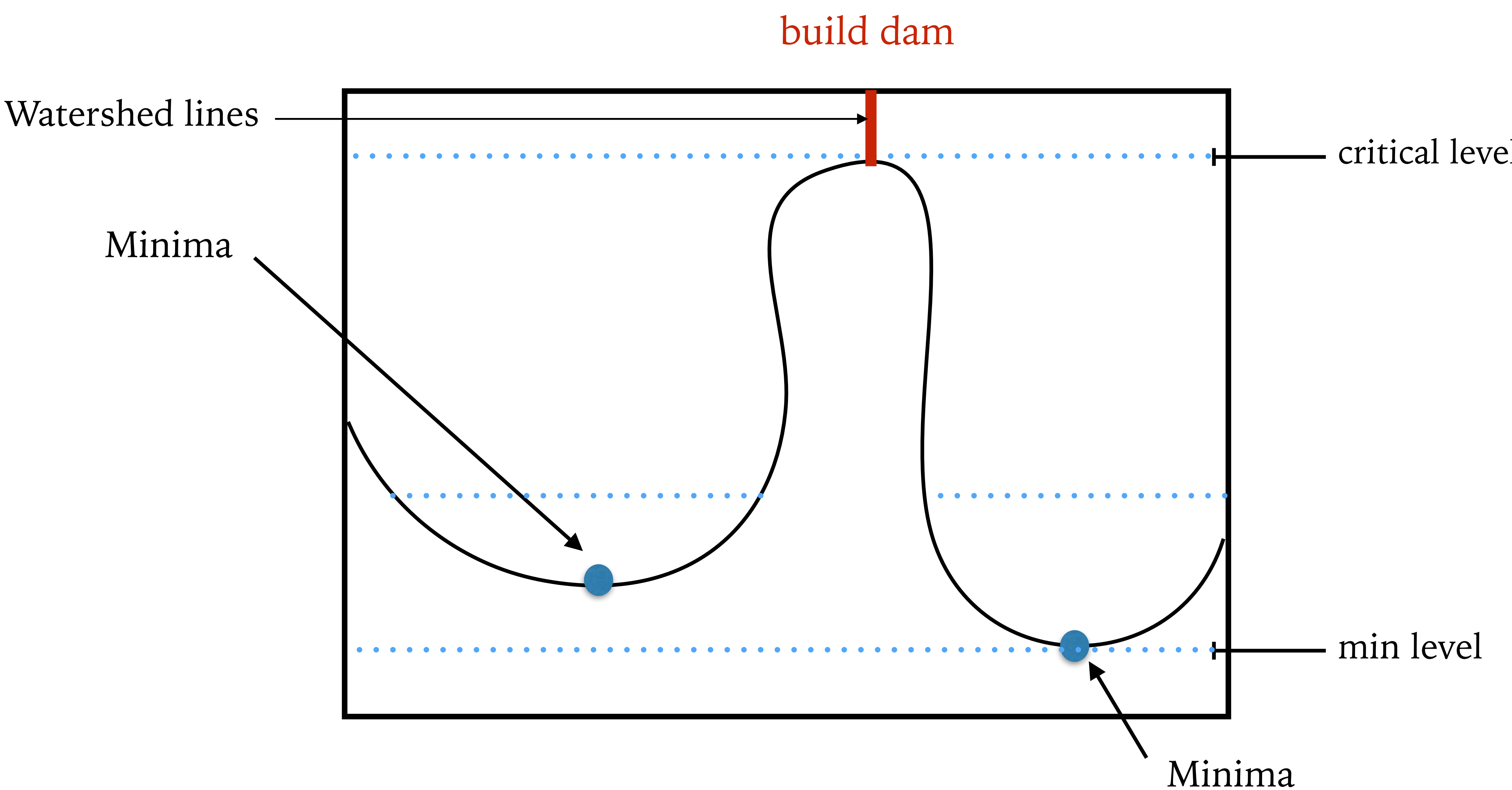
K-means clustering



Watershed Segmentation

Watershed

- **Physical interpretation**: Consider a gray level image as a topological surface. where the pixel intensity corresponds to the height of the surface



Watershed algorithm

- **Input:** a gray-level image ***I*** with gray scale ***[h_{min} h_{max}]***
- **Define**:
  - Minima points: ***M_1, …, M_R***
  - Thresholded set: $T_h = \{p \in I | I(p) \leq h\}$, where ***p*** is an pixel in ***I*** and ***h*** is some intensity level.
- **Initialize:**
  - ***h = h_{min}***
  - Immersed set: $$X_h = X_{h_{min}} = T_{h_{min}}$$ $$= \{p \in I | I(p) \leq h_{min}\}$$

Watershed algorithm

- **Loop** until $\boldsymbol{h_{max}}$

$$X_{h+1} = X_h \bigcup IZ_{h+1}(M_1) \ldots \bigcup IZ_{h+1}(M_R)$$

Influence set of minima $\boldsymbol{M_1}$ at level $\boldsymbol{h+1}$

- *Watershed*(***I***) = Set of all pixels in $\boldsymbol{I} \setminus \boldsymbol{X_{hmax}}$

- Let's define Influence set

$$C(M_i) = \text{cluster associated with } M_i$$

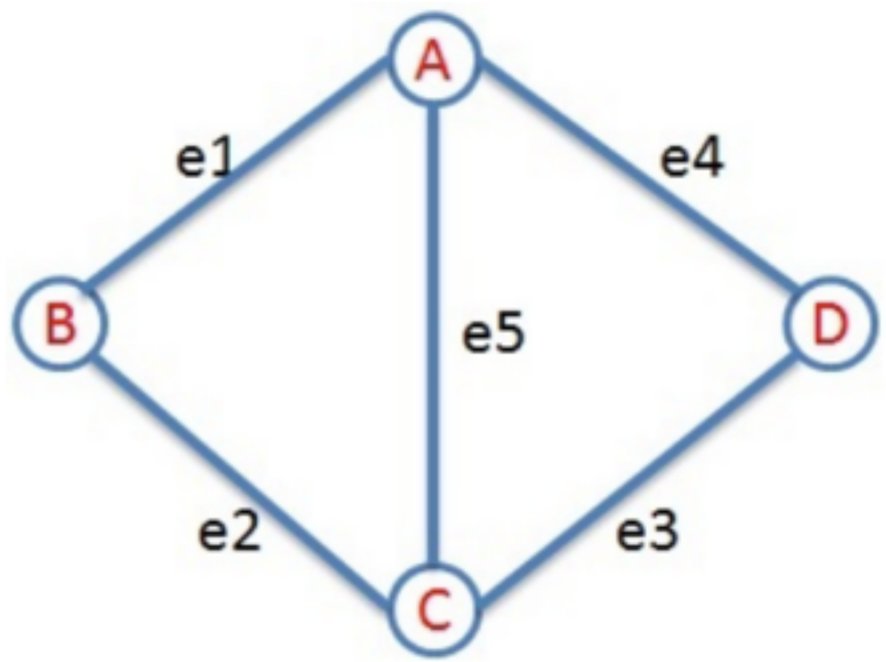
$$IZ_{h+1}(M_i) = \{p \in T_{h+1} | d(p, C(M_i)) < d(p, C(M_j))\}$$

$$\forall j,\ i \neq j$$

Graph-based Segmentation

What is a graph?

- A graph **G = (V, E)** has two components
  - a set of vertices **V**
  - a set of edges **E**, which characterize the pairwise relationship between nodes/vertices

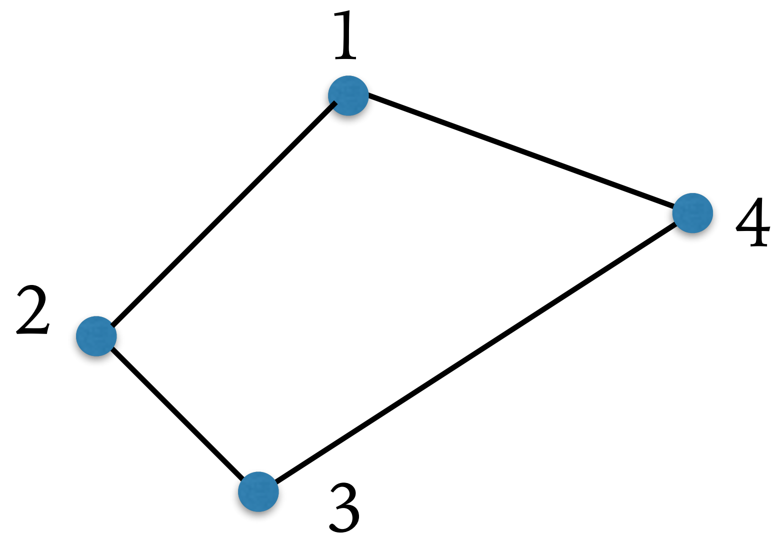


$V = \{A, B, C, D\}$

$E = \{e_1, e_2, e_3, e_4\}$

Adjacency matrix

- A graph is often represented as an Adjacency matrix $\mathbf{A_d}$



$$\mathbf{A_d} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$

- **A** here is binary -> indicates presence or absence of connection
- **A** here is unweighted, undirected.

Images as graphs

- **Main idea:** Represent images as graphs
    - Pixels as nodes
    - Pixel relationship as edges

- **Why graphs?**
    - Compact representation
    - Computational convenience and scalability
        - Easy to extend to higher dimensions
    - Mathematical convenience
        - Take advantage of Graph theory

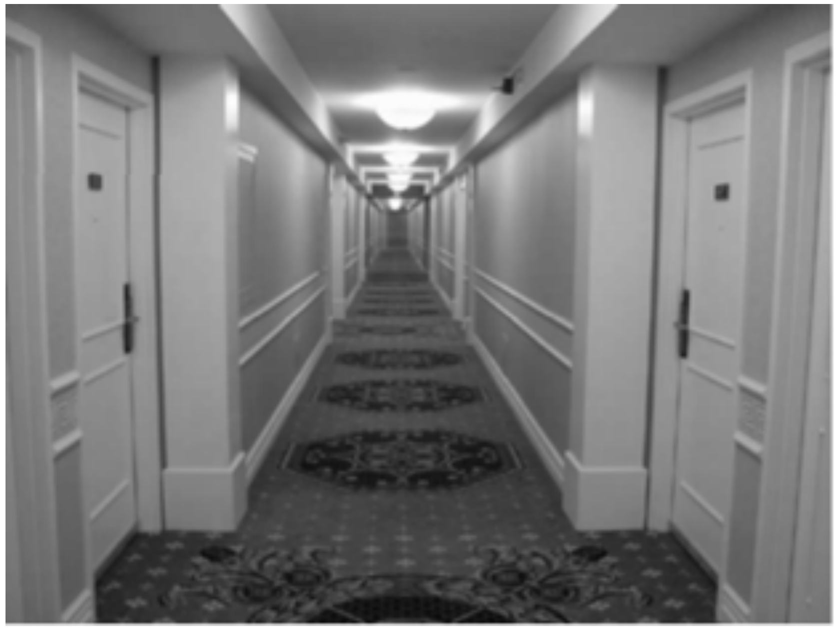
Images as graphs

- **Images as graphs**
  - Each pixels is a **node**: $\mathbf{V = \{p_1, \ldots, p_N\}}$
  - Each pair of neighboring pixels share an **edge**
    - The concept of neighborhood can be specified according to requirement (4-connect is popular)
    - edges can be weighted or unweighted
    - edges can be directed or undirected
- $\mathbf{p_i, p_j}$ are two pixels,
  - Unweighted edge: $\mathbf{w_{ij} = 1}$ ($\mathbf{p_i, p_j}$ are neighbors), $\mathbf{w_{ij}=0}$ (otherwise)
  - Weighted edge: $\mathbf{w_{ij} = 1/\, d(p_i, p_j)}$ ($\mathbf{p_i, p_j}$ are neighbors), $\mathbf{w_{ij}=0}$ (otherwise)

Images as graphs

$\sqrt{N}$

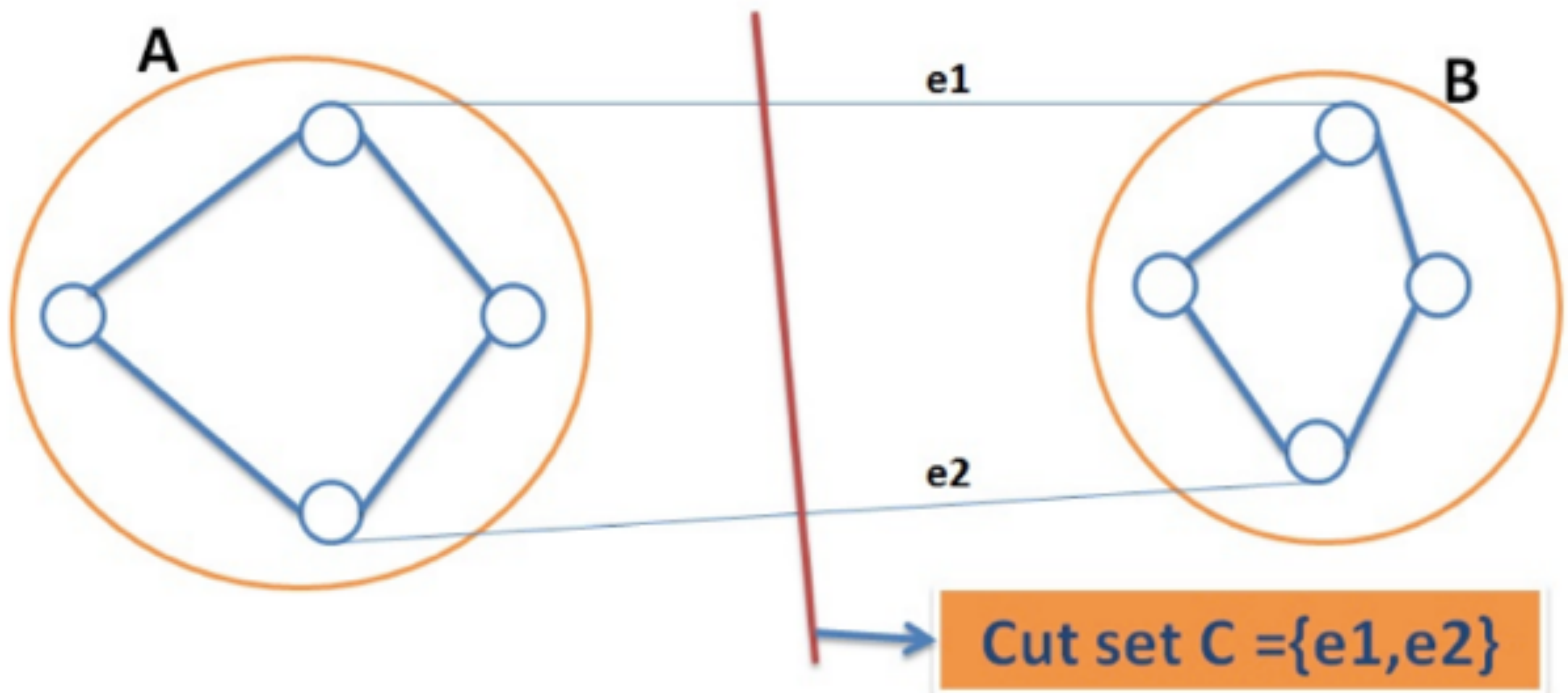
$\sqrt{N}$



$$\mathbf{A_d} = \begin{bmatrix} w_{11} & \dots & w_{1N} \\ w_{21} & \dots & w_{21} \\ . & . & . \\ . & . & . \\ w_{N1} & \dots & w_{1N} \end{bmatrix}$$

Graph cut

A **cut** is a partition of nodes **V** into two non-empty sets **A** and **B(= V\A)**.



{e1, e2}: crossing edges (has one node in **A** and the other in **B**)

$$cut(A, B) = |C| = 2 \qquad cut(A, B) = \sum_{V_i \in A, V_j \in B} w_{ij}$$

Images as graphs

- In segmentation, we are interested in partitioning the image into a number of disjoint sets.

- This is graph partition, if an image is represented as a graph.

- Often an objective is to have sets which are most dissimilar (or similar).

- So, we are interested in a partition (or cut) that has lowest number of crossing edges. - **Min cut**

- There are many standard algorithms for Mincut.

Kerger's mincut

Karger's Algorithm:

- Randomly choose an edge from the graph.
- Collapse the two nodes in to a supernode. Ignore self edges, keep all (parallel) edges to other nodes
- Loop until 2 nodes are left.
- The algorithm returns a mincut with a probability $\binom{n}{2}^{-1}$

Kerger's mincut

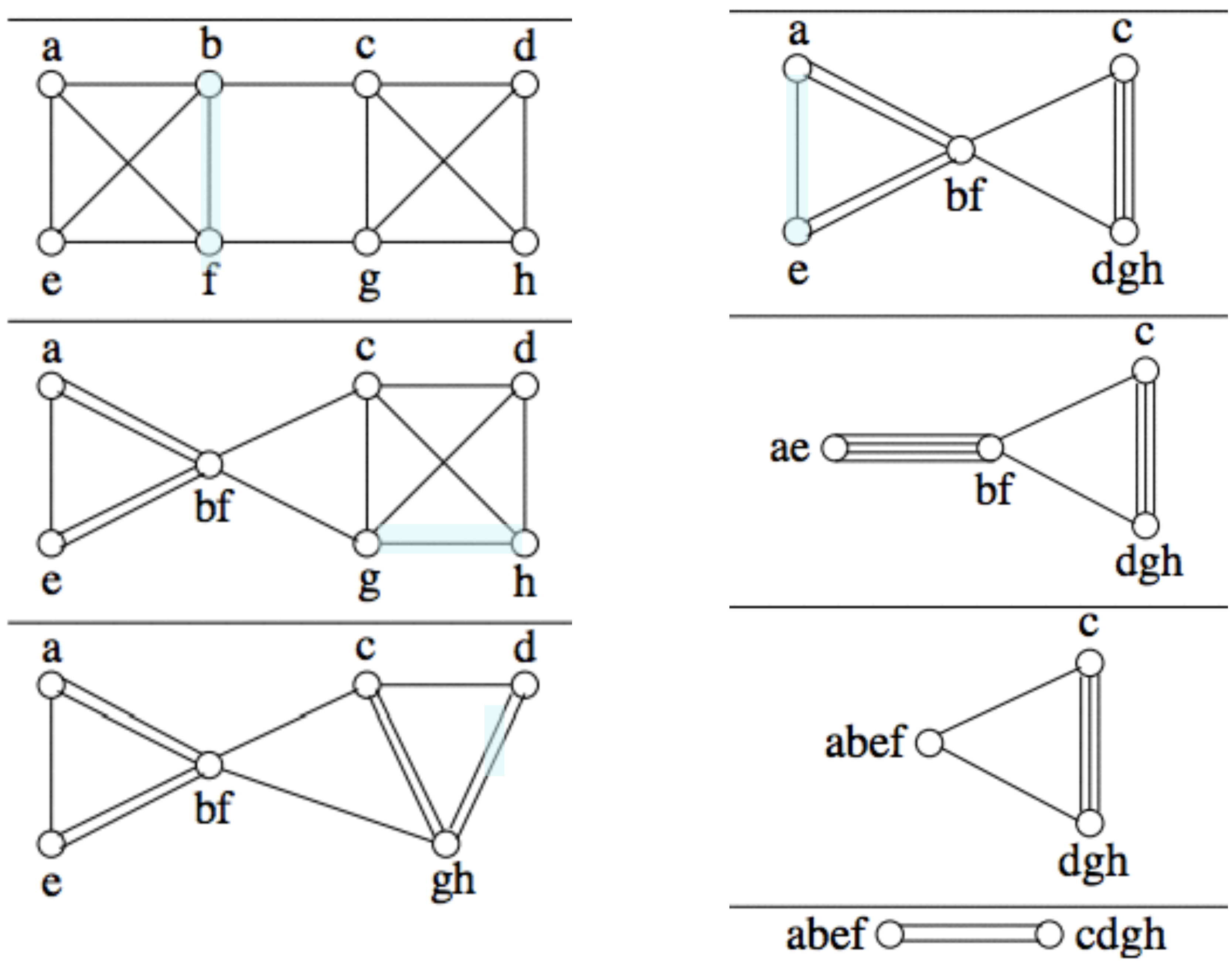


image source: https://www.cs.princeton.edu/courses/archive/fall13/cos521/lecnotes/lec2final.pdf

Summary

- Broad classes of segmentation approaches
  - Shape segmentation (Hough transform, Active appearance model, Snake, …)
  - Thresholding (Optimal vs. approximate, global vs. local)
  - Region growing (surface fitting, cellular automata …)
  - Clustering (Agglomerative, K-means, …)
  - Graph-based (Min cut, normalized cut …)
  - Supervised (when enough labels are available, train a binary classifier and label new pixels)