VC 11/12 – T10
Advanced Segmentation

Mestrado em Ciência de Computadores

Mestrado Integrado em Engenharia de Redes e Sistemas Informáticos

Miguel Tavares Coimbra

U.PORTO FC

Outline

- Introduction
- Simple clustering
- K-means clustering
- Graph-theoretic clustering
- Fitting lines

Acknowledgements: These slides follow Forsyth and Ponce's "Computer Vision: A Modern Approach", Chapters 14 and 15.

Topic: Introduction

- Introduction
- Simple clustering
- K-means clustering
- Graph-theoretic clustering
- Fitting lines

What is ‘Segmentation’? (again?)

- Traditional definition:

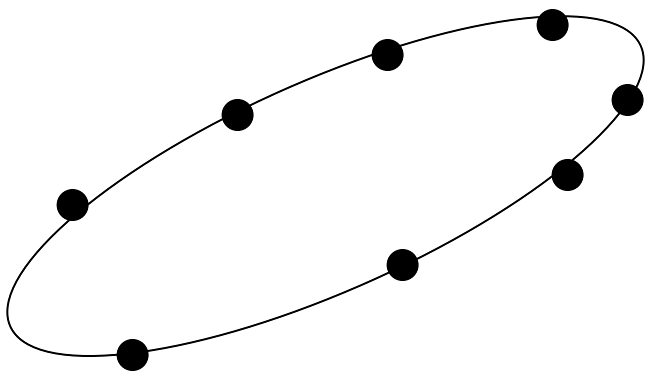
  “Separation of the image in different areas“

  - Decompose an image into “superpixels”.
  - Colour and texture coherence.

- Aren’t there other ways to look at the ‘Segmentation’ concept?

Other ‘*Segmentation*’ problems

- Fitting lines to edge points.



We can’t see this as ‘separating an image in different areas’!

- Fitting a fundamental matrix to a set of feature points.

This one is complicated!
Check Forsyth and Ponce, chap.14

Segmentation as Clustering

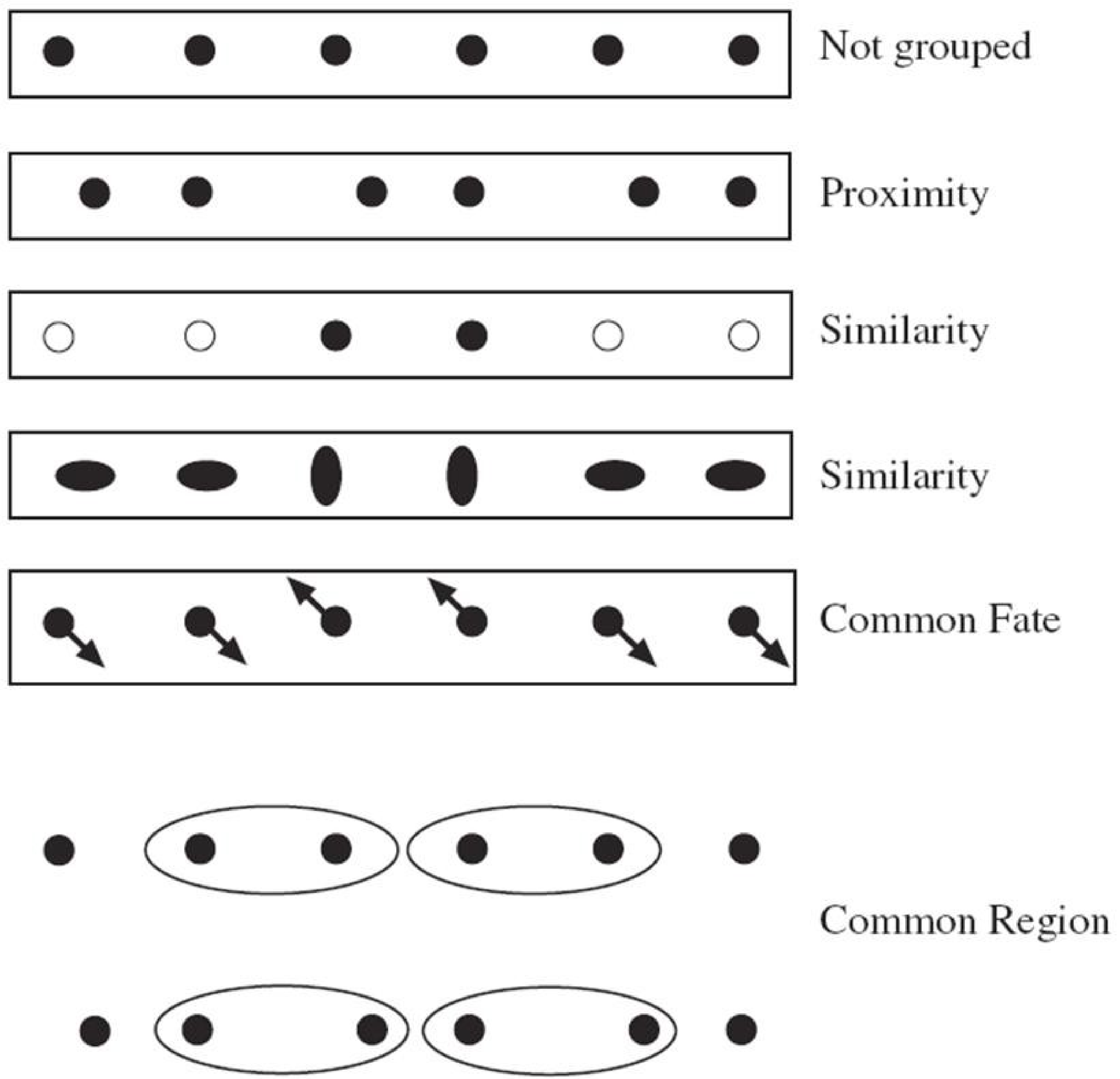
- Tries to answer the question:

  "Which components of the data set naturally belong together?"

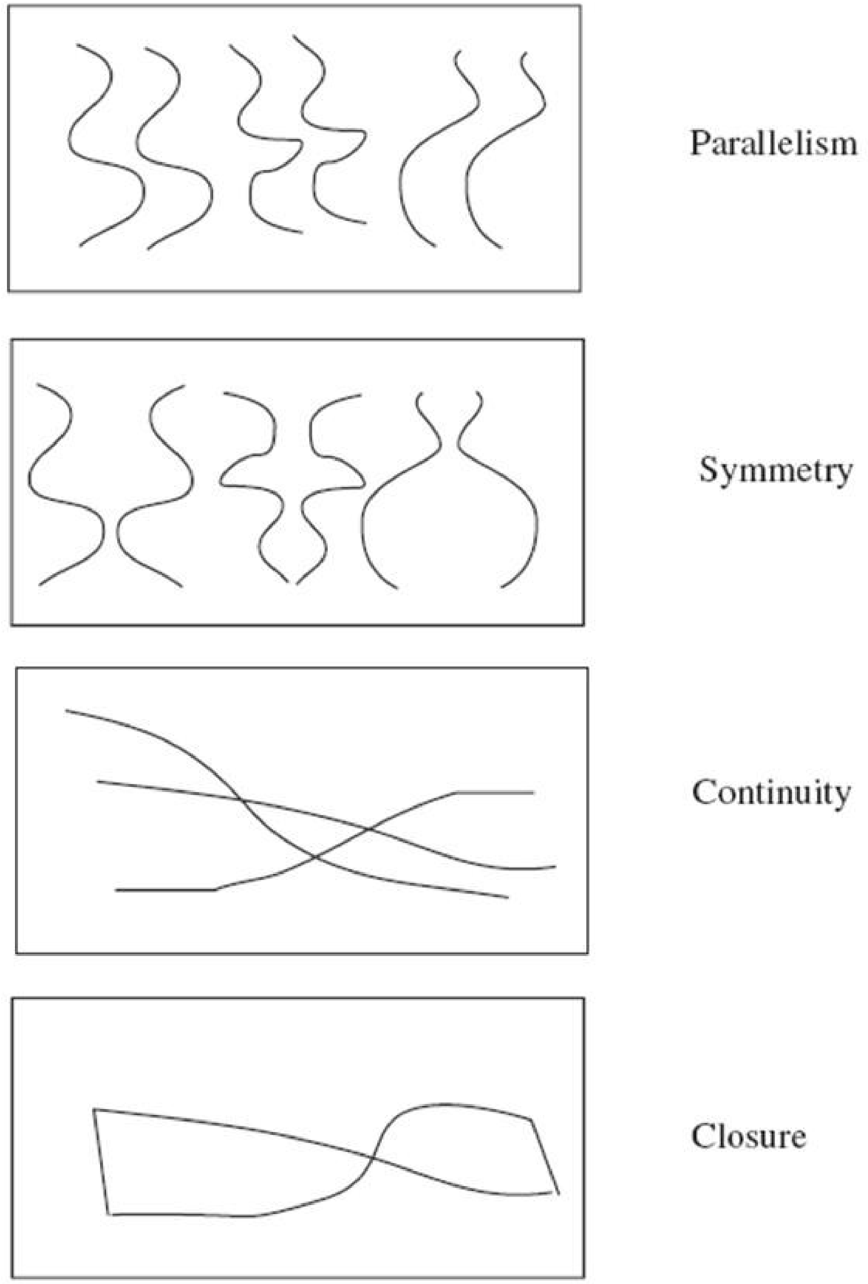
- Two approaches:
  - Partitioning
    - Decompose a large data set into pieces that are 'good' according to our model.
  - Grouping
    - Collect sets of data items that 'make sense' according to our model.

Human Clustering

- How de we humans *cluster* images?
  - Well… we don't really know...
- Gestalt school of psychologists
  - Attempts to study this problem.
  - Key ideas:
    - Context affects perception. So…
    - Responses to stimuli are not important.
    - Grouping is the key to understanding visual perception.



Examples of Gestalt factors that lead to grouping



Examples of Gestalt factors that lead to grouping

Gestalt in Practice

- Rules function fairly well as explanations.
- However, they are insufficient to form an algorithm.
- So, how is Gestalt useful?
  - Gives us ‘hints’ on where to go.
  - Shatters the traditional definition of segmentation, clearly showing us that we need something better.
  - Context is vital! Grouping is vital!

Topic: Simple clustering

- Introduction
- Simple clustering
- K-means clustering
- Graph-theoretic clustering
- Fitting lines

What do we mean by 'clustering'?

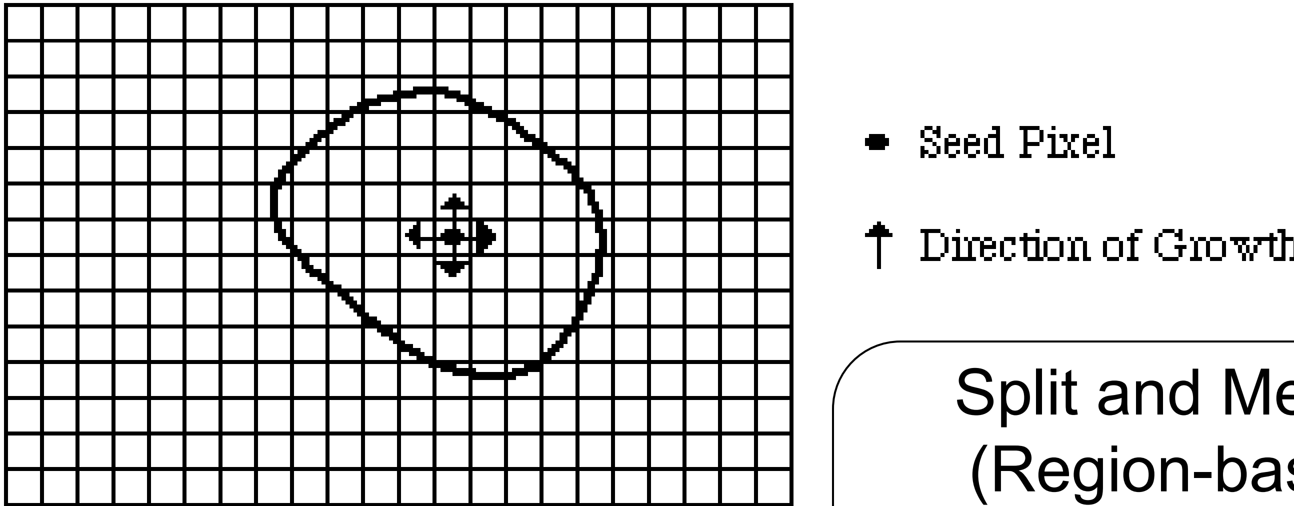
- "Clustering is a process whereby a data set is replaced by **clusters**, which are collections of data points that belong together"

Forsyth and Ponce, "Computer Vision: A modern approach"

- Why do points "belong together"?
  - Same colour.
  - Same texture.
  - Same… something!

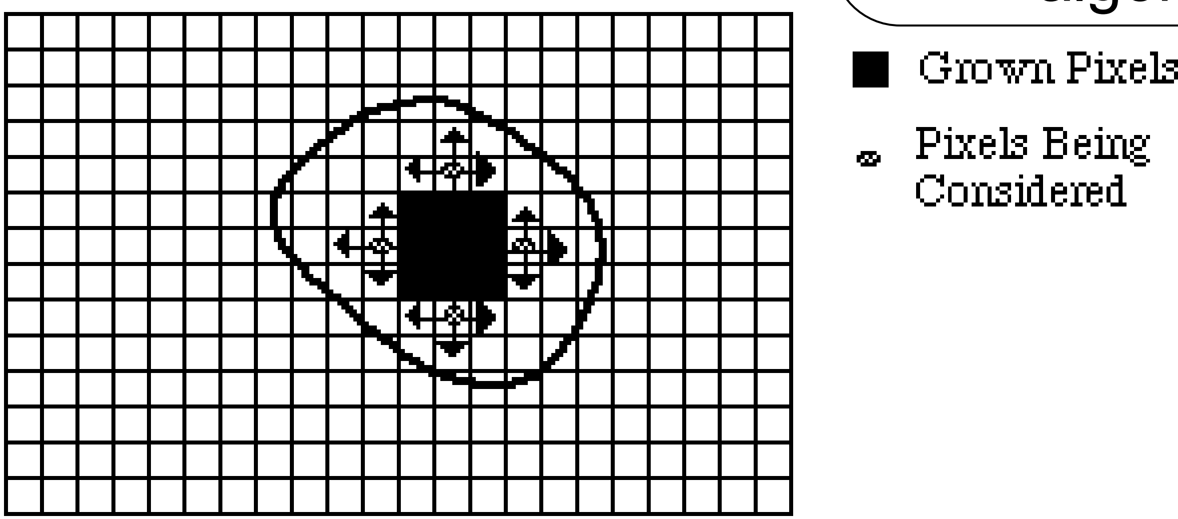
Simple clustering

- Two natural types of clustering:
  - Divisive clustering
    - Entire data set is regarded as a cluster.
    - Clusters are recursively split.
  - Agglomerative clustering
    - Each data item is a cluster.
    - Clusters are recursively merged.
- Where have I seen this before?



(a) Start of Growing a Region

Split and Merge (Region-based segmentation) is in fact a *clustering* algorithm.



(b) Growing Process After a Few Iterations

Generic simple clustering algorithms

- Divisive Clustering
  - Construct a single cluster containing all points.
  - Until the clustering is satisfactory
    - Merge the two clusters with smallest inter-cluster distance.
  - end
- Agglomerative Clustering
  - Make each point a separate cluster
  - Until the clustering is satisfactory
    - Split the cluster that yields the two components with the largest inter-cluster distance.
  - end

Which inter-cluster distance?

What does this mean?

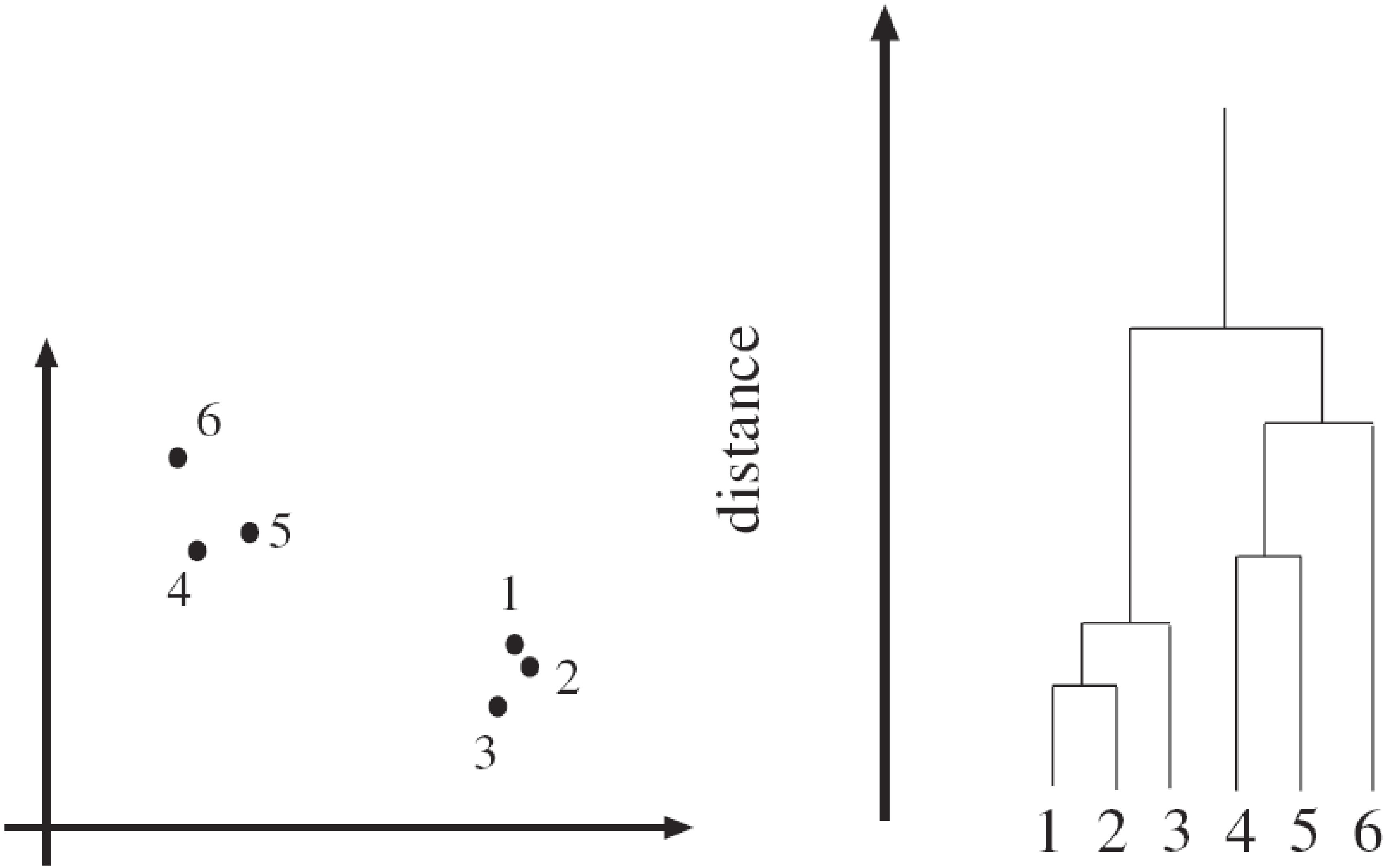


Figure 16.12. Left, a data set; right, a dendrogram obtained by agglomerative clustering using single link clustering. If one selects a particular value of distance, then a horizontal line at that distance will split the dendrogram into clusters. This representation makes it possible to guess how many clusters there are, and to get some insight into how good the clusters are.

Simple clustering with images

- Some specific problems arise:
  - Lots of pixels! Graphical representations are harder to read.
  - Segmentation: It is desirable that certain objects are connected. How to enforce this?
  - When do we stop splitting/merging process?
- Complex situations require more complex clustering solutions!

Topic: K-means clustering

- Introduction
- Simple clustering
- K-means clustering
- Graph-theoretic clustering
- Fitting lines

Objective function

- What if we know that there are *k* clusters in the image?
- We can define an *objective function*!
  - Expresses how good my representation is.
- We can now build an algorithm to obtain the *best* representation.

Caution! "*Best*" given my objective function!

K-means Clustering

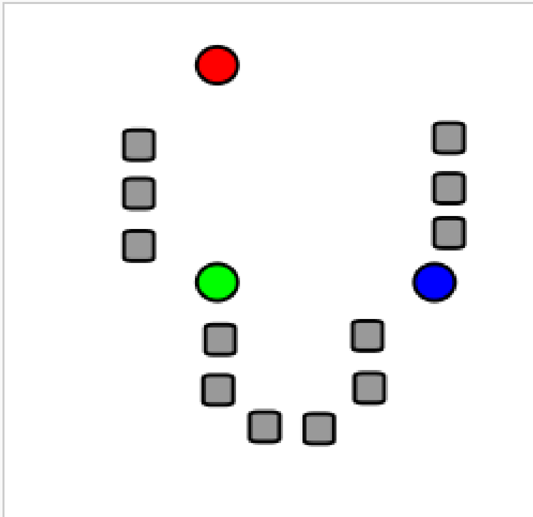
- Assume:
  - We have k clusters.
  - Each cluster i has a centre c_i.
  - Element j to be clustered is described by a feature vector x_j.
- Our objective function is thus:

What does this mean?

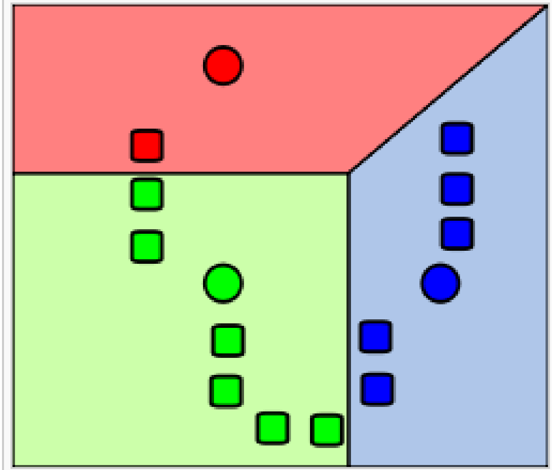
$$\Phi(clusters, data) = \sum_{i \in clusters} \left\{ \sum_{j \in cluster(i)} (x_j - c_i)^T (x_j - c_i) \right\}$$

Iteration step

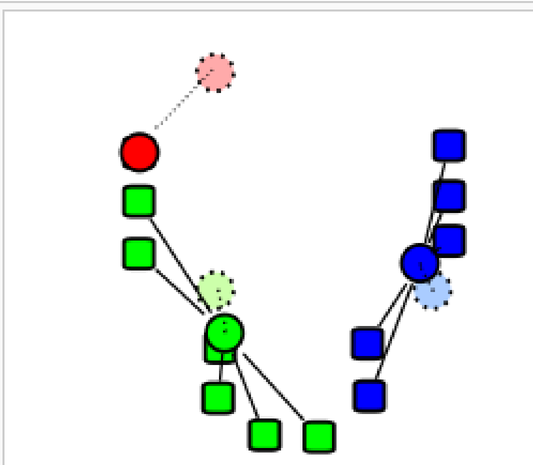
- Too many possible allocations of points to clusters to search this space for a minimum.
- Iterate!
  - Assume cluster centres are known and allocate each point to the closest cluster centre.
  - Assume the allocation is known and choose a new set of cluster centres. Each centre is the mean of the points allocated to that cluster.



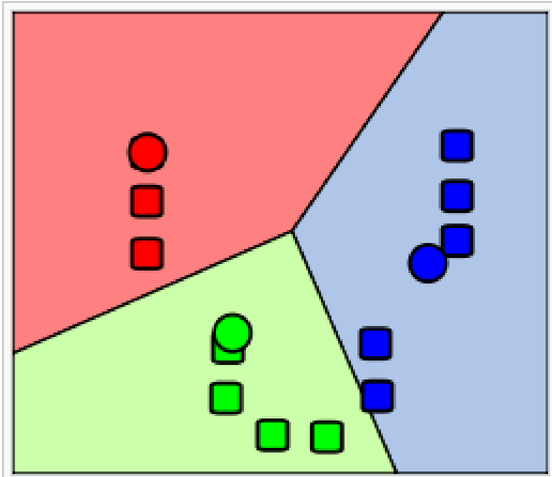
Shows the initial randomized centroids and a number of points.



Points are associated with the nearest centroid.



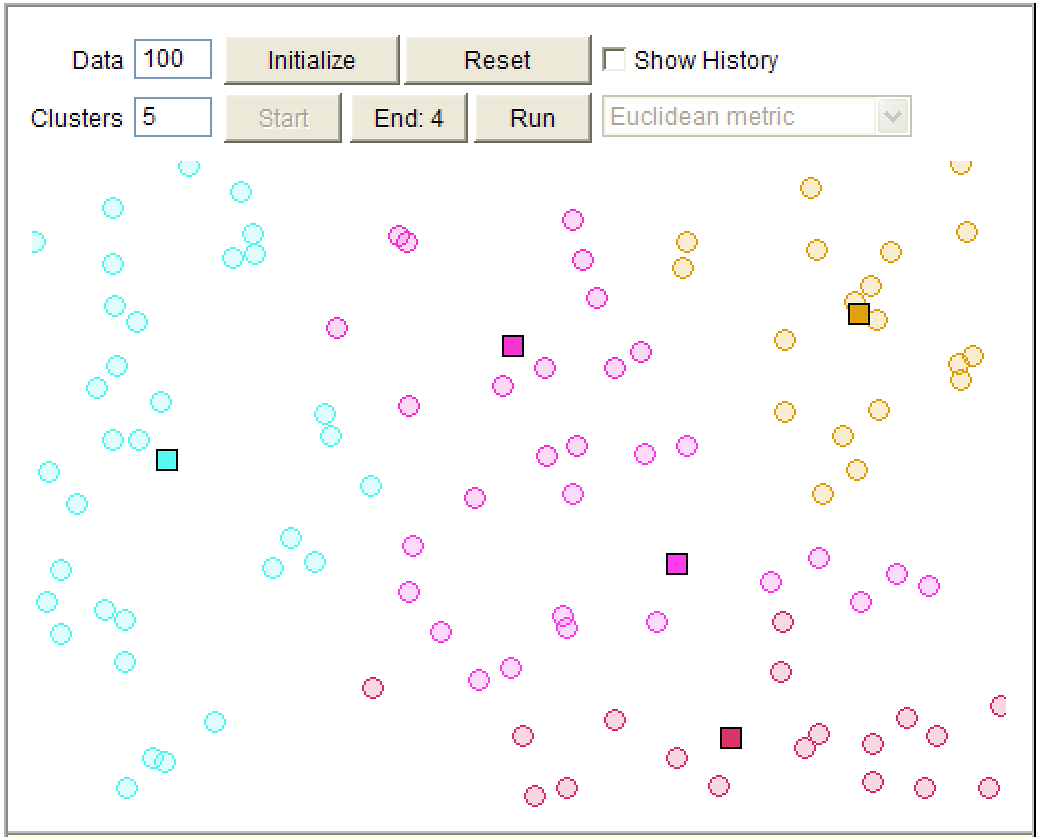
Now the centroids are moved to the center of their respective clusters.



Steps 2 & 3 are repeated until a suitable level of convergence has been reached.

[Wikipedia]

Interactive Java Tutorial



http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/AppletKM.html

Topic: Graph-theoretic clustering

- Introduction
- Simple clustering
- K-means clustering
- Graph-theoretic clustering
- Fitting lines

Using graphs

- Clustering can be seen as a problem of "*cutting graphs into good pieces*".
- Data Items
  - Vertex in a weighted graph.
  - Weights are large if elements are similar.
- Cut edges
  - Cut edges with small weights.
  - Keep connected components with large interior weights.

Regions!

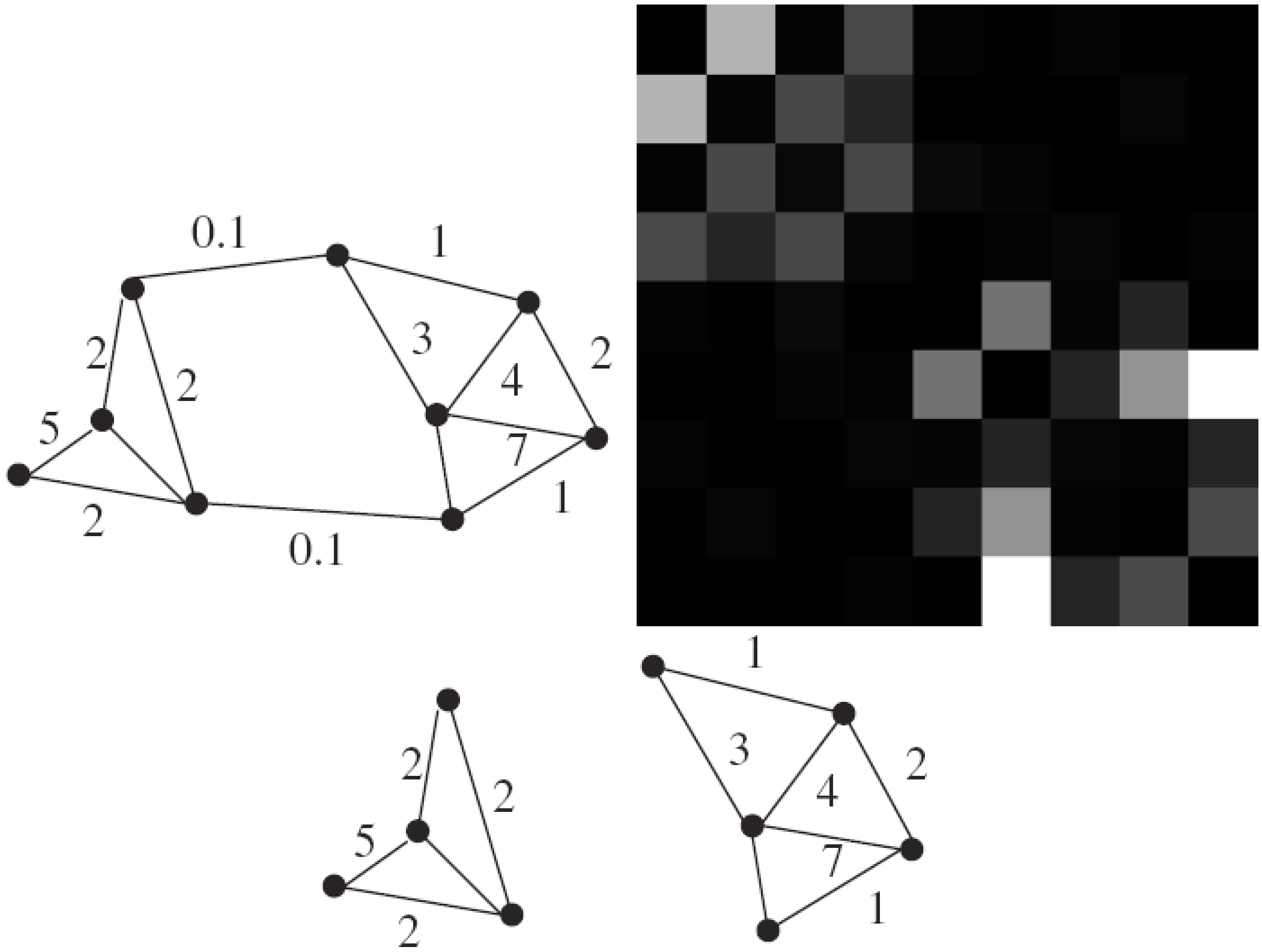


Figure 16.16. On the **top left**, a drawing of an undirected weighted graph; on the **top right**, the weight matrix associated with that graph. Larger values are lighter. By associating the vertices with rows (and columns) in a different order, the matrix can be shuffled. We have chosen the ordering to show the matrix in a form that emphasizes the fact that it is very largely block-diagonal. The figure on the **bottom** shows a cut of that graph that decomposes the graph into two tightly linked components. This cut decomposes the graph's matrix into the two main blocks on the diagonal.

Graphs and Clustering

- Associate each element to be clustered with a vertex on a graph.
- Construct an edge from every element to every other.
- Associate a weight with each edge based on a similarity measure.
- Cut the edges in the graph to form a good set of connected components.

Weight Matrices

- Typically look like block diagonal matrices.
- Why?
  - Interclusters similarities are strong.
  - Intracluster similarities are weak.
- Split a matrix into smaller matrices, each of which is a block.
- Define *Affinity Measures*.

More on this

- Affinity measures
  - – Affinity by Distance
  - – Affinity by Intensity
  - – Affinity by Colour
  - – Affinity by Texture
- Popular method: Normalized cuts

  Jianbo Shi and Jitendra Malik, “Normalized Cuts and Image Segmentation”, IEEE Transactions on Pattern Analysis And Machine Intelligence, Vol. 22, No. 8, August 2000

Want to know more?
Check out: Forsyth and Ponce, Section 14.5

Topic: Fitting lines

- Introduction
- Simple clustering
- K-means clustering
- Graph-theoretic clustering
- Fitting lines

Fitting and Clustering

- Another definition for segmentation:
  - Pixels belong together because they conform to some model.
- Sounds like “Segmentation by Clustering”...
- Key difference:
  - The model is now **explicit**.

We have a mathematical model for the object we want to segment.
E.g. A line

Hough Transform

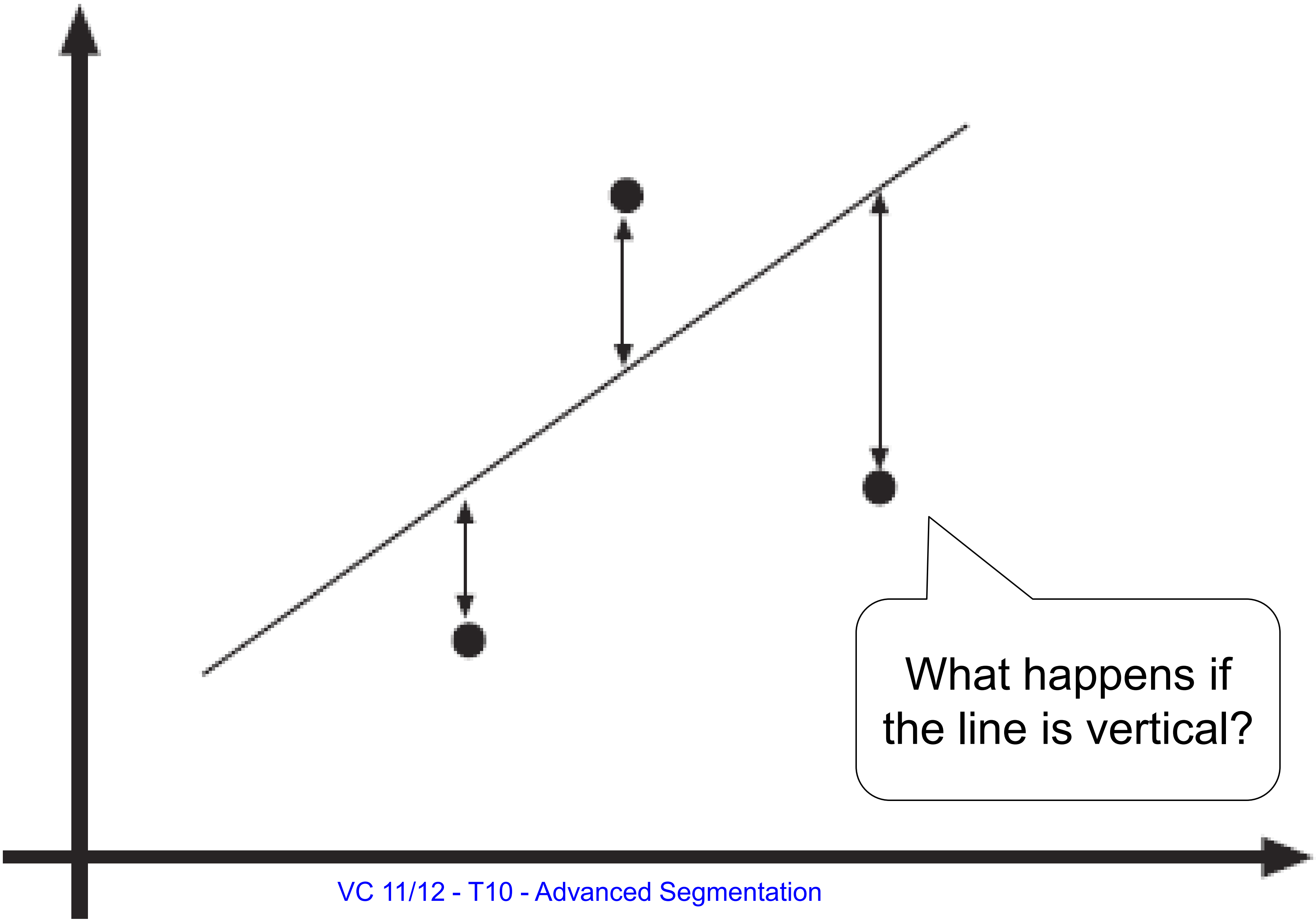
- Elegant method for direct object recognition
- Edges need not be connected
- Complete object need not be visible
- Key Idea: Edges **VOTE** for the possible model

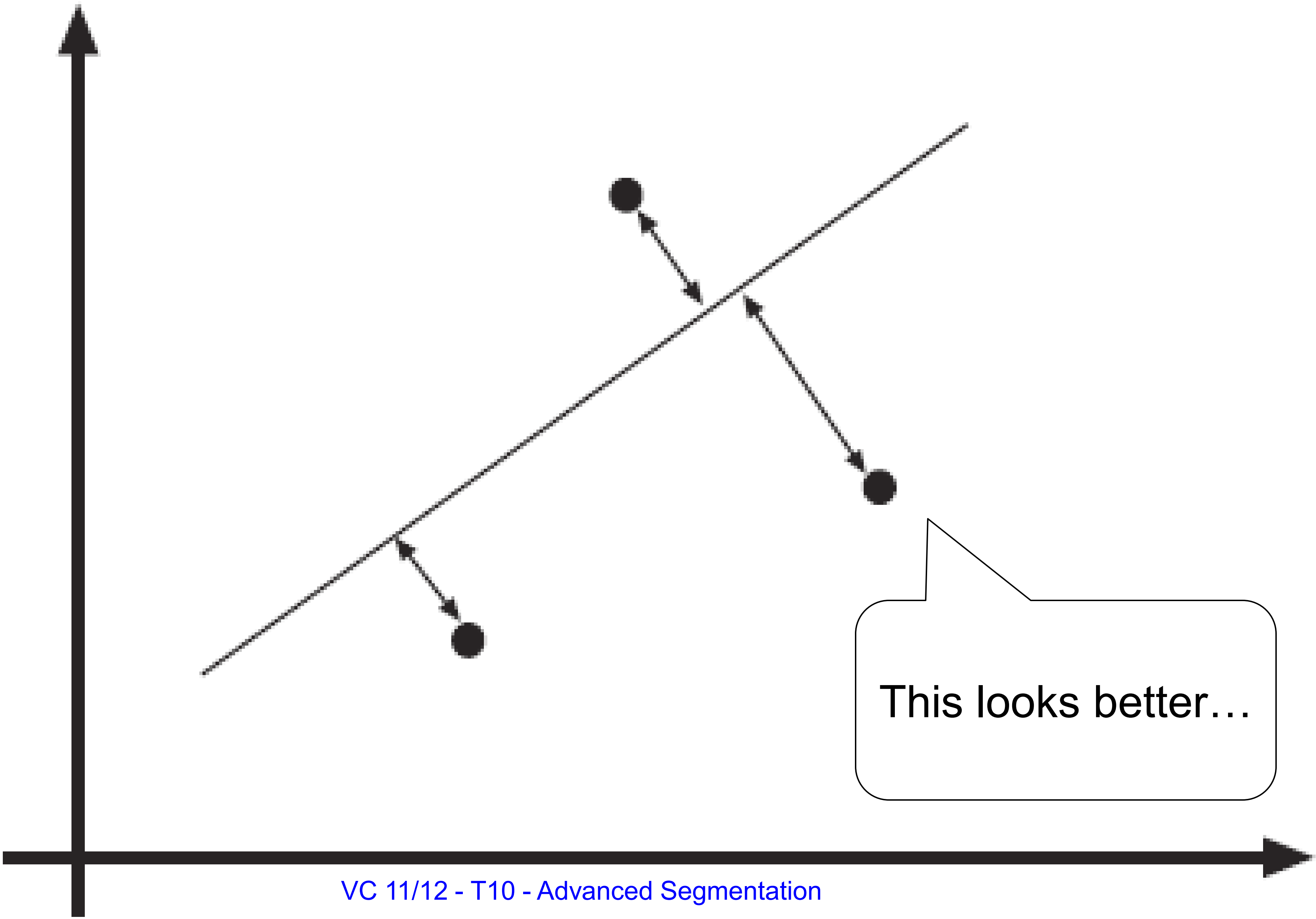
Least Squares

- Popular fitting procedure.
- Simple but biased (why?).
- Consider a line:

y = ax + b

- What is the line that best predicts all observations (x_i,y_i)?
  - Minimize: $$\sum_i (y_i - ax_i - b)^2$$


What happens if
the line is vertical?


This looks better…

Total Least Squares

- Works with the actual distance between the point and the line (rather than the vertical distance).
- Lines are represented as a collection of points where:

$$ax + by + c = 0$$

- And:

$$a^2 + b^2 = 1$$

Again... Minimize the error, obtain the line with the 'best fit'.

Point correspondence

- We can estimate a line but, **which points are on which line**?
- Usually:
  - We are fitting lines to edge points, so…
  - Edge directions can give us hints!
- What if I only have isolated points?
- Let's look at two options:
  - Incremental fitting.
  - Allocating points to lines with K-means

Incremental Fitting

- Start with connected *curves* of edge points
- Fit *lines* to those points in that curve.
- Incremental fitting:
  - Start at one end of the *curve*.
  - Keep fitting all points in that curve to a line.
  - Begin another line when the fitting deteriorates too much.
- Great for closed curves!

```
Put all points on curve list, in order along the curve
empty the line point list
empty the line list

Until there are two few points on the curve
  Transfer first few points on the curve to the line point list
  fit line to line point list

  while fitted line is good enough
    transfer the next point on the curve
    to the line point list and refit the line
  end

  transfer last point back to curve
  attach line to line list
end
```

K-means allocation

- What if points carry no hints about which line they lie on?
- Assume there are *k* lines for the *x* points.
- Minimize: $$\sum_{\text{lines}} \sum_{\text{points}} dist(\text{line}, \text{point})^2$$
- Iteration:
  - Allocate each point to the closest line.
  - Fir the best line to the points allocated to each line.

```
Hypothesize k lines (perhaps uniformly at random)
or
hypothesize an assignment of lines to points
and then fit lines using this assignment

Until convergence
  allocate each point to the closest line
  refit lines
```

Resources

- Forsyth and Ponce, Chapter 14
- Forsyth and Ponce, Chapter 15