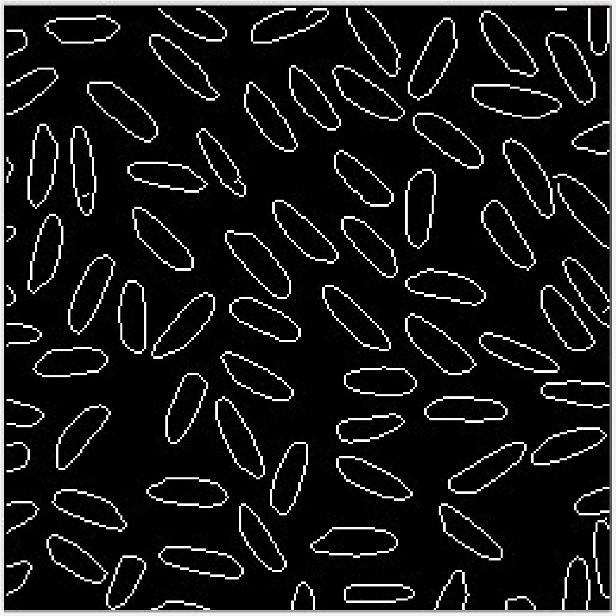
Department of Computer Science
University of Bristol

Image Processing and Computer Vision

www.ole.bris.ac.uk/bbcswebdav/courses/COMS30121_2018/content
www.ole.bris.ac.uk/bbcswebdav/courses/COMSM0020_2018/content



Lecture 05

Segmentation Basics

Andrew Calway | Tilo Burghardt | Sion Hanunna

31 Slides

Definition of Image Segmentation

- **Image Segmentation ...**

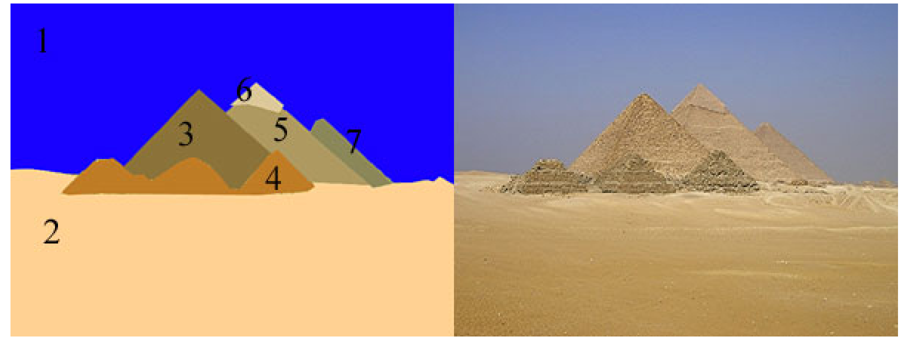
  ... is the process of spatial subsectioning of a (digital) image into multiple partitions of pixels (i.e. segments or regions) according to given criteria.

Example: segmentation of an image into locally coherent regions

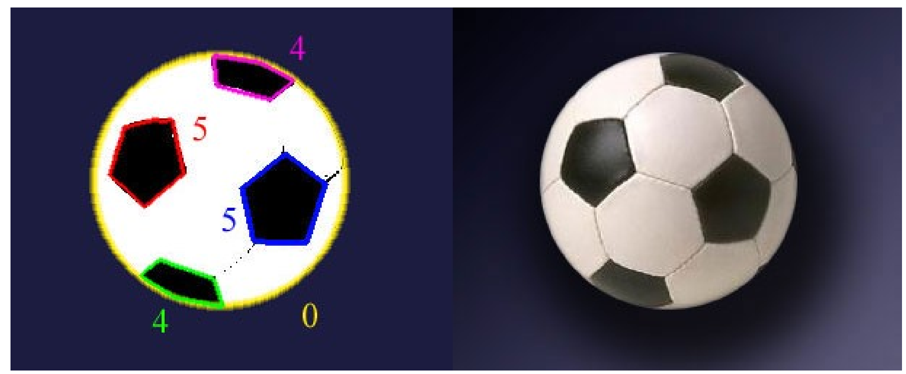
(Original Slide by M Mirmehdi)

Motivation: Why Segment Images?

- **Image Simplification**
  - an image may contain millions of pixels but only a few regions



- **Higher-level Object Description**
  - regions tend to belong to the same class of object
  - regions may provide object properties (e.g. shape, colour, ...)



- **Input for Content Classifiers**
  - region descriptions can be input data for higher level classifiers, e.g. Bayesian Classifiers or Neural Networks.

(Original Slide by M Mirmehdi)

Image Segmentation

Perfect image segmentation is difficult to achieve:

- a pixel may straddle the "real" boundary of objects such that it partially belongs to two or more objects
- effects of noise, non-uniform illumination, occlusions etc. give rise to the problem of over-segmentation and under-segmentation
- Over-segmentation: pixels belonging to the same object are classified as belonging to different segments
- Under-segmentation: pixels belonging to different objects are classified as belonging to the same object

(Original Slide by M Mirmehdi)

Example of Over- and Under Segmentation

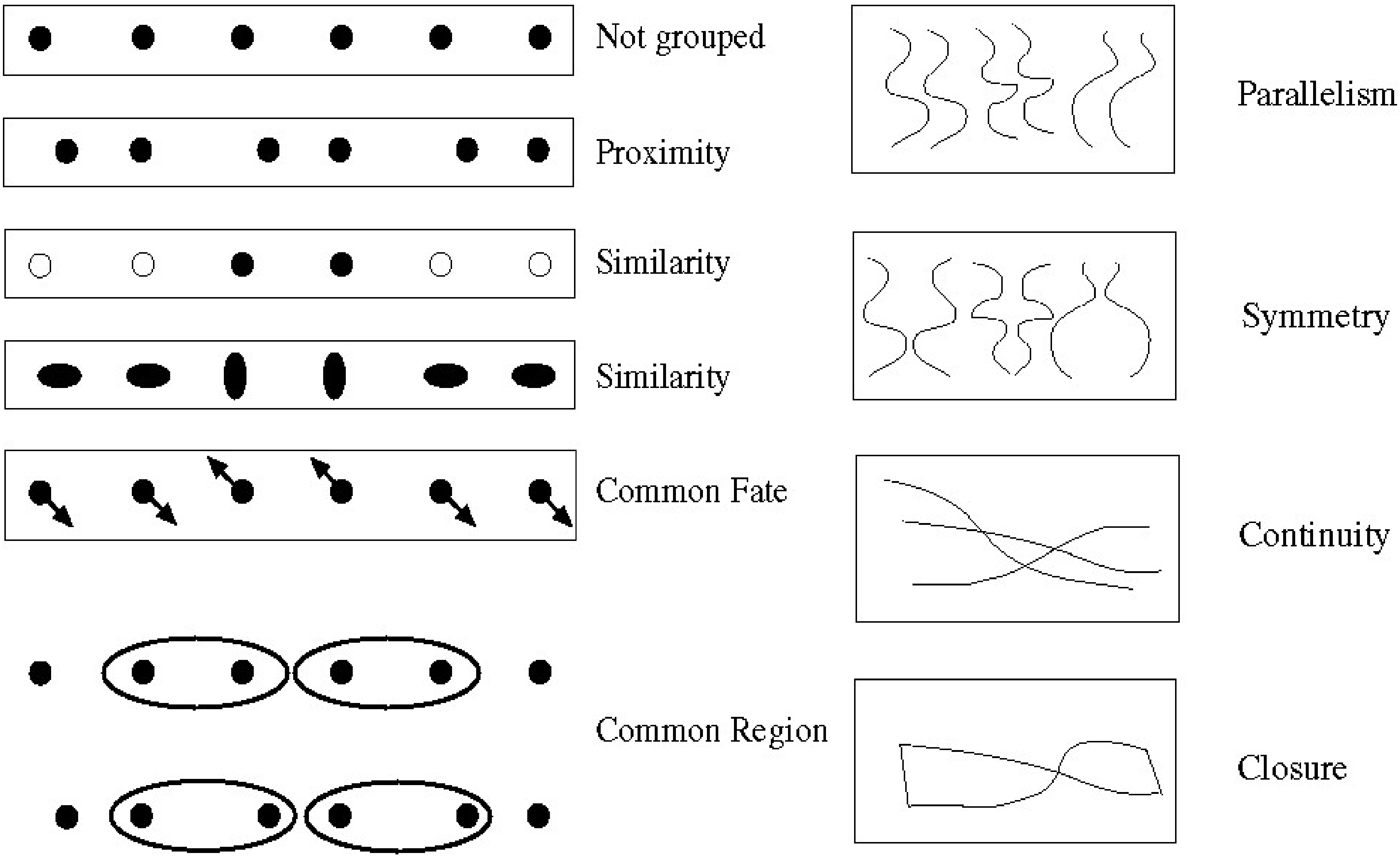
Original image

Over-segmentation

Under-segmentation

(Original Slide by M Mirmehdi)

Digression: Gestalt-Rules for Grouping and Segmentation



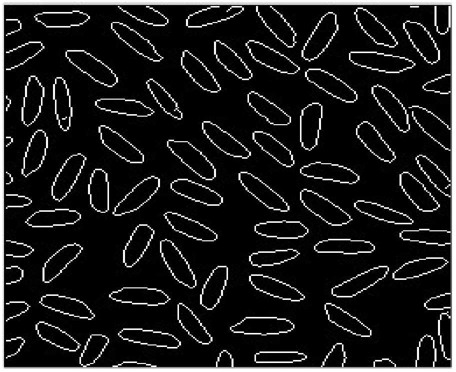
Concepts of Segmentation I

Thresholding Methods

- pixels are categorized based on intensity
- only useful when sufficient contrast exists

Edge-based Methods

- region boundaries are constructed from edgemaps



Region-based Methods

- region growing from seed pixels
- region splitting and merging for efficient spatial encoding

(Original Slide by M Mirmehdi)

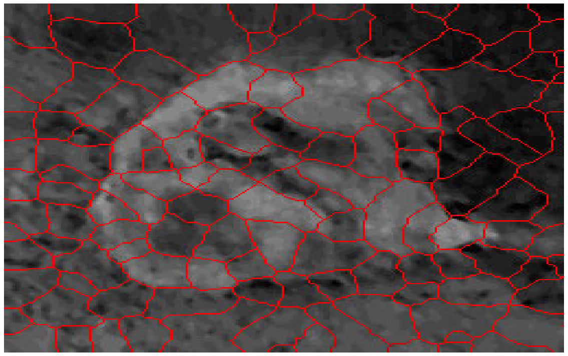
Concepts of Segmentation II

Clustering and Statistical Methods

- global, often histogram based image partitioning, e.g. k-means, Gaussian Mixture Model

Topographic Methods

- stepwise simplifications that take spatially wider (topographical) image configurations into account e.g. watershed transform, variational based methods



(Original Slide by M Mirmehdi)

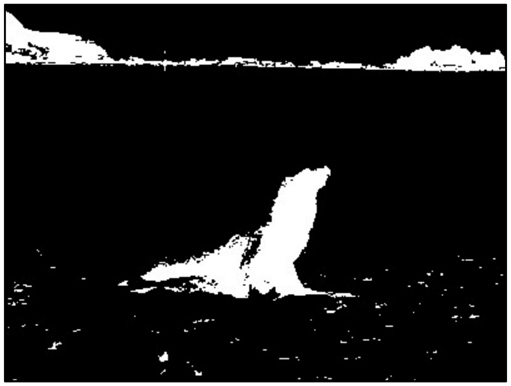
Thresholding Example

Original Image

T = 128

T = 96

T = 64



(Original Slide by M Mirmehdi)

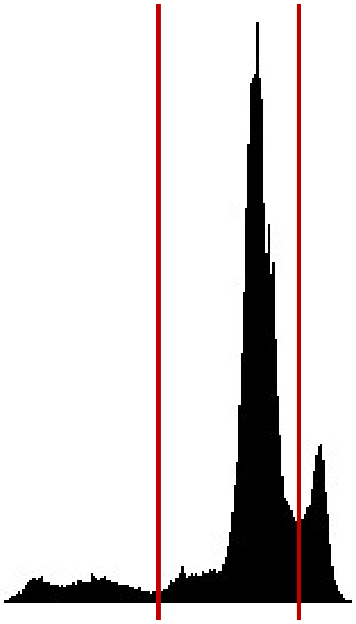
Using Histograms to Stipulate Regions

To find a threshold we can use an image histogram

- count how many pixels in the image have each value
- for simple images it shows peaks and valleys around regions of the image

The seal image shows three regions

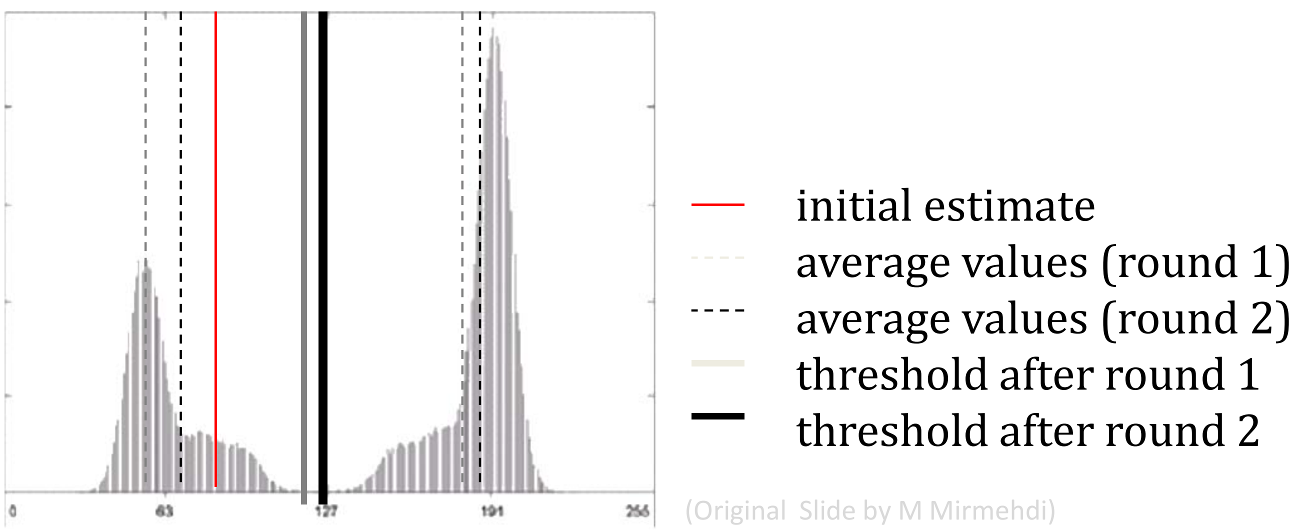
- one below T_1 = 80
- one above T_2 = 142
- one between the two thresholds



(Original Slide by M Mirmehdi)

Threshold Selection Algorithm

1. Select an initial estimate for the threshold T
2. Segment the image using T.
   This will produce two groups of pixels: G_1 consisting of all pixels with grey levels $>T$ and G_2 consisting of pixels with grey values $<T$.
3. Compute the average grey level values m_1 and m_2 for the pixels in regions G_1 and G_2.
4. Compute a new threshold value: $T = (m_1 + m_2)/2$
5. Repeat steps *(2.)* through *(4.)* until convergence



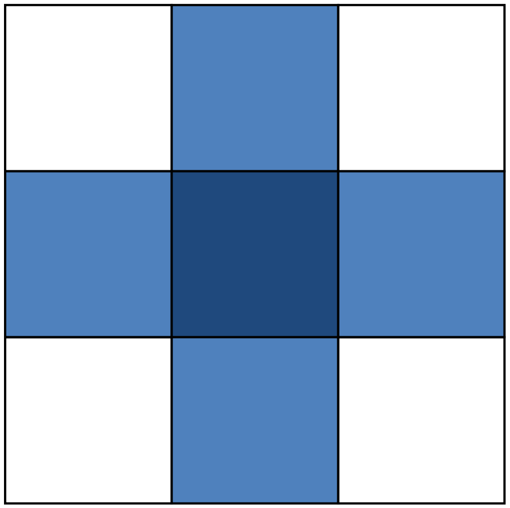
(Original Slide by M Mirmehdi)

Connectivity

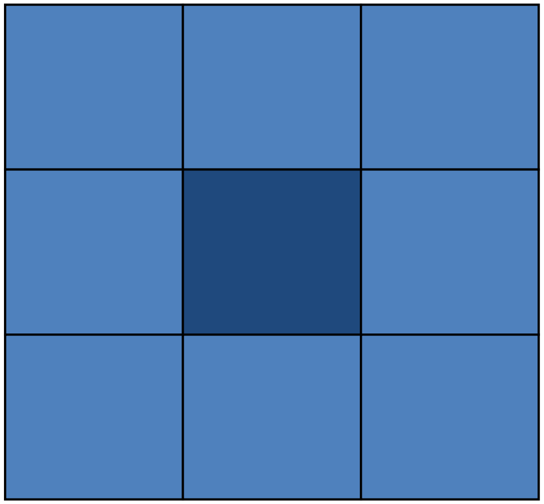
General idea:

There are several ways of defining the adjacent neighbourhood of a pixel given the image grid.

Two common paradigms:



4-connectivity



8-connectivity

Edge-based Segmentation I

General idea:

- detect strong edges
- extend or delete them in order to create closed boundaries that represent objects

- problems: noise or no edge presence where a real object border exits

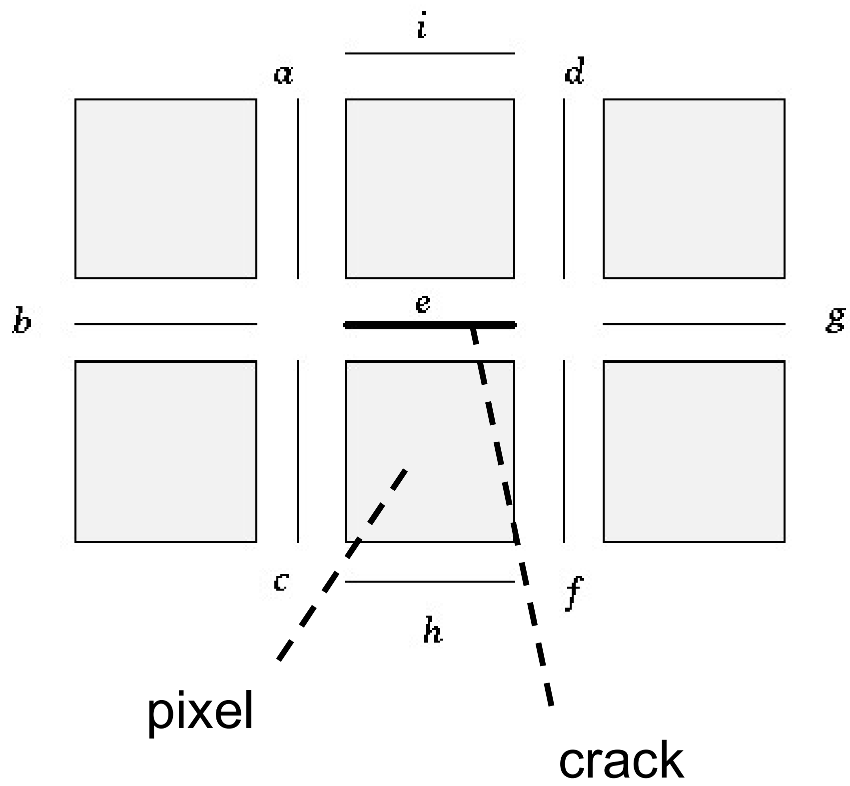
(Original Slide by M Mirmehdi)

Edge-based Segmentation II

Conceptual examples:

- A weak edge positioned between two strong edges provides an example of context; it is highly probable that this inter-positioned weak edge should be a part of a resulting boundary.

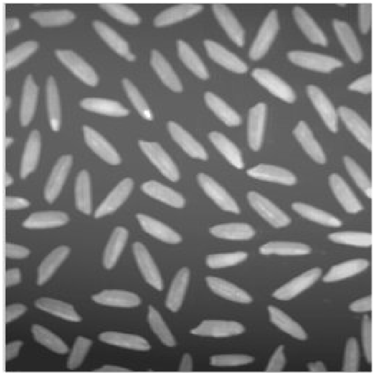
-If, on the other hand, an edge (even a strong one) is positioned by itself with no supporting context, it is probably not a part of any border.

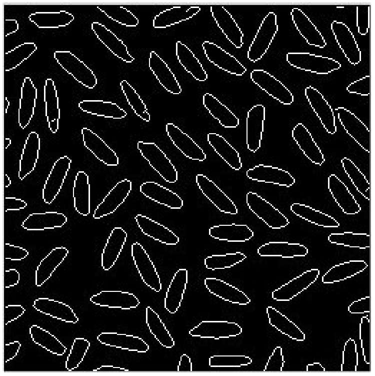


Edge-based Segmentation III

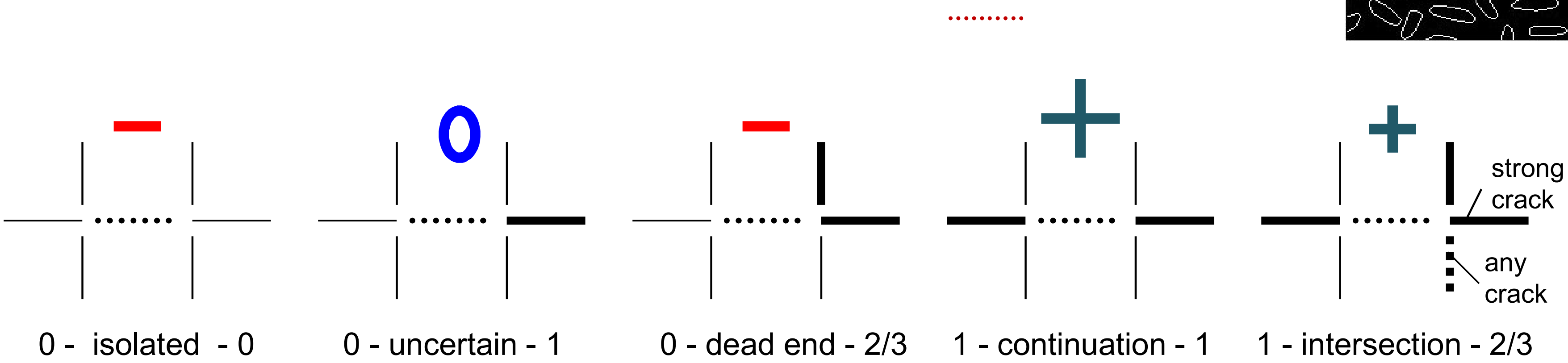
Example: Edge Relaxation

Consider each crack context (e.g. the adjacent 6 cracks) in order to iteratively establish stable, connected and closed edges, i.e. based on the strength of the edges in a specified neighbourhood, the confidence of each edge, e.g. C(e), is increased or decreased.





Relevant context-configurations and their influence on edge probability:



(Original Slide by M Mirmehdi)

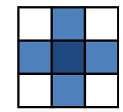
Region Growing I

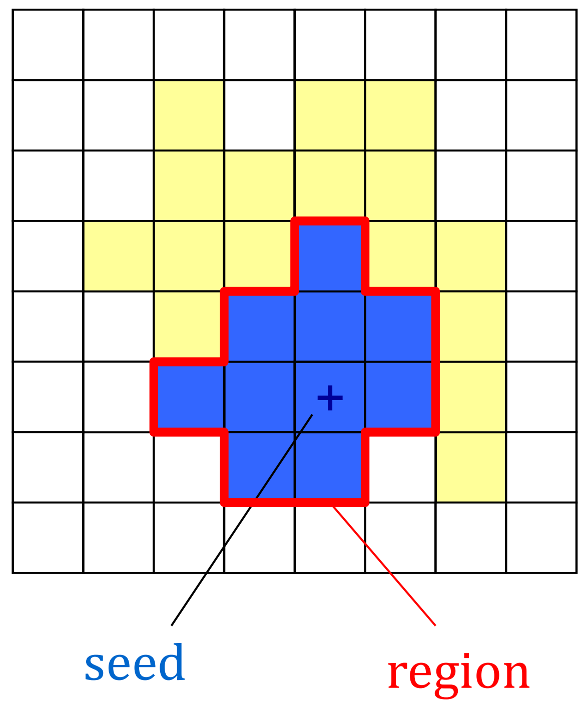
Find regions by growing homgeneous areas around initially chosen seed pixels.

Region Growing Algorithm:

- Start with an initial seed pixel.
- Grow area to neighbouring pixels (based on connectivity) if they satisfy a homogeneity condition test.
- If the region doesn't grow anymore select another seed and repeat the process until all pixels are accounted for.
- Final tidying operation is often performed to remove very small regions.

Example: for a single region using 4-connectivity:





accepted & grown
accepted
current tests

Region Growing II

Homogenity Condition:

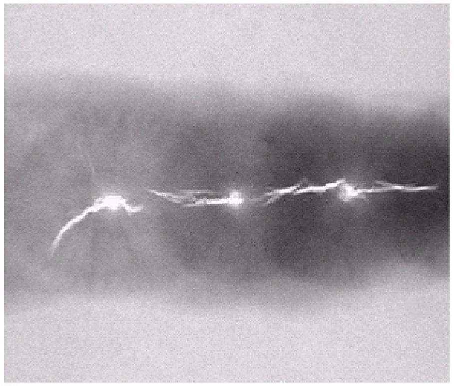
A characteristic function H that maps from parameters of the current region r and a new candidate pixel p to a binary decision whether to merge or not:

$$(r, p) \xrightarrow{H} \{0, 1\}$$

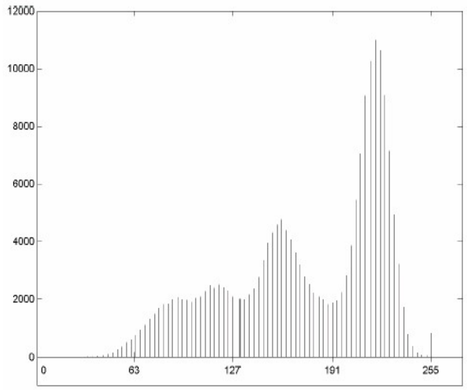
Example:

- Initialise seed region as grey level 255 (full luminance)
- Use 8-connectivity
- Basic homogeneity condition H:
  - Merge, if the luminance difference between the seed and a pixel is less than 65
  - Merge regions, if a pixel is connected to more than one region

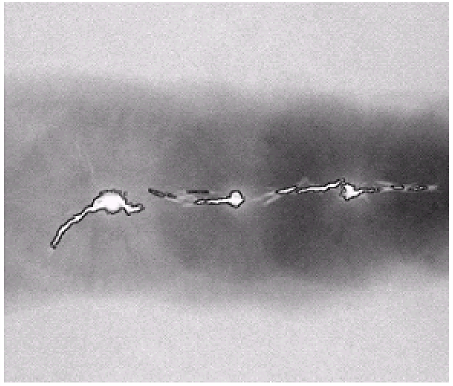
Original



Initialization



Histogram

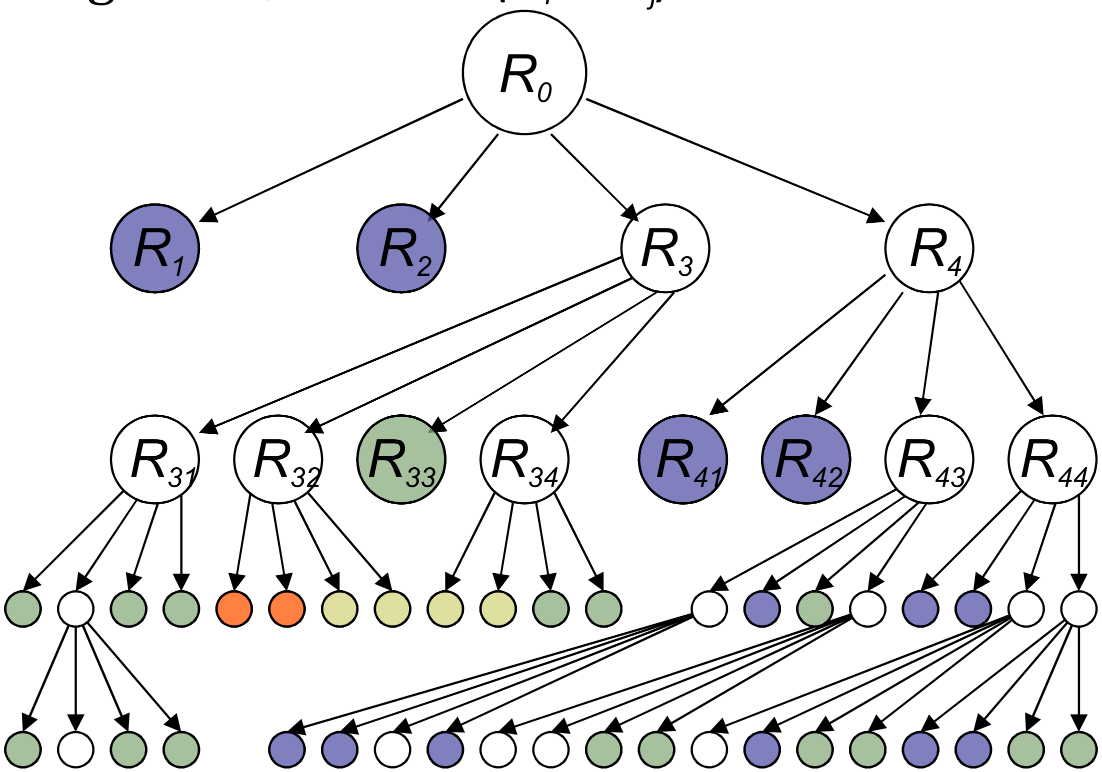


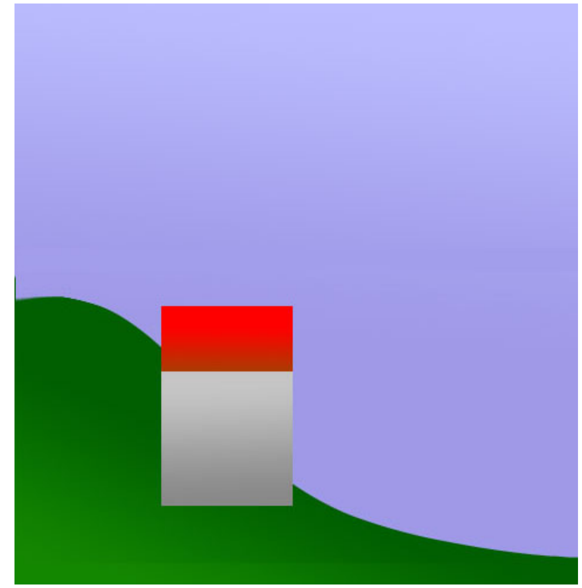
Segmentation Result

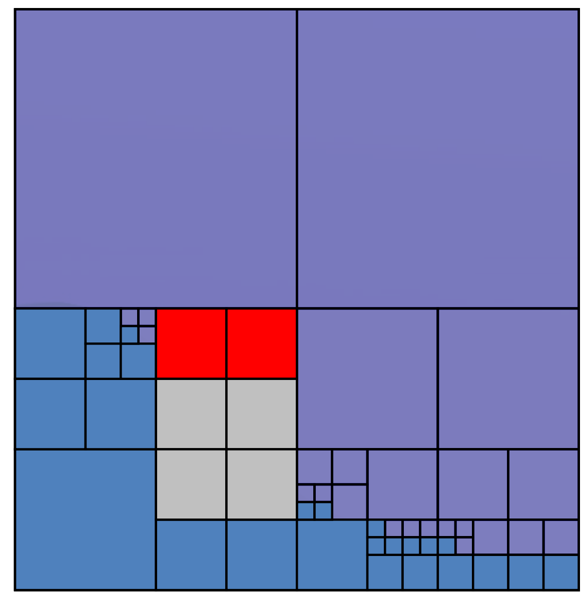
(Original Slide by M Mirmehdi)

Split & Merge – Divide & Conquer

1. Start with R_0 that represents the entire image
2. If $H(R_i)=0$ (=inhomogeneous) then
   {split area into 4 blocks (quadtree splitting) and
   process each area with step (2.)}
3. Merge all subregions that are (pairwise) homogenous, that is $H(R_i \cup R_j)=1$







Segmentation Result

Split & Merge – Summary

Conceptual Summary:

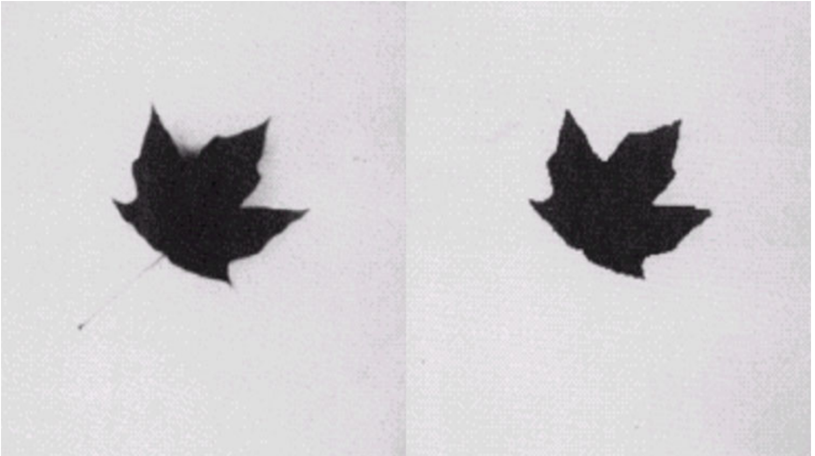
- Iteratively decompose an image into regions of a maximally sized selected shape (e.g. rectangle) that does not satisfy a homogeneity condition.(split step)
- Then merge regions that together satisfy a homogeneity condition. (merge step)

Some Comments:

- Using quadtrees, the results of split and merge tend to be *blocky*.
- Can have an adaptive homogeneity condition that, for instance, changes depending on the region size.

Example:

- $H(R_i)=1$ if at least 80% of the pixels in R_i have the property $| z_j - m_i | < 2\sigma_i$, where z_j is the grey level of the j^{th} pixel in R_i, m_i is the mean grey level of the region and σ_i is the standard deviation of the grey levels in R_i
- If $H(R_i)=1$ then set all the pixels in R_i to value m_i

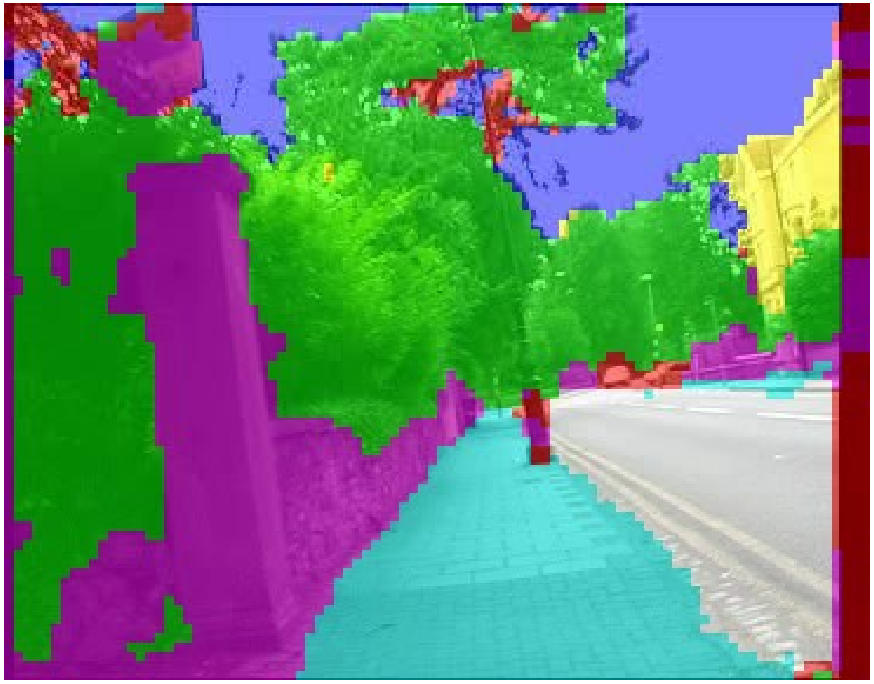


Original Result

(Original Slide by M Mirmehdi)

Split & Merge – Example

Original Video Frame



- Images are segmented using a Split-And-Merge technique. (Note the blocky nature of the regions!)

- Regions are then labelled by a Neural Network to associate the segments with semantics (colouration).

(Original Slide by M Mirmehdi)

Statistical Segmentation by Clustering

- Clustering involves segmenting the instance/feature space into regions of similar objects
  - no guidance through labelled training instances
    → unsupervised learning

- We can use the idea of a distance metric
  → K-means clustering

- We can also use the class as a 'hidden' variable
  → Gaussian mixture models

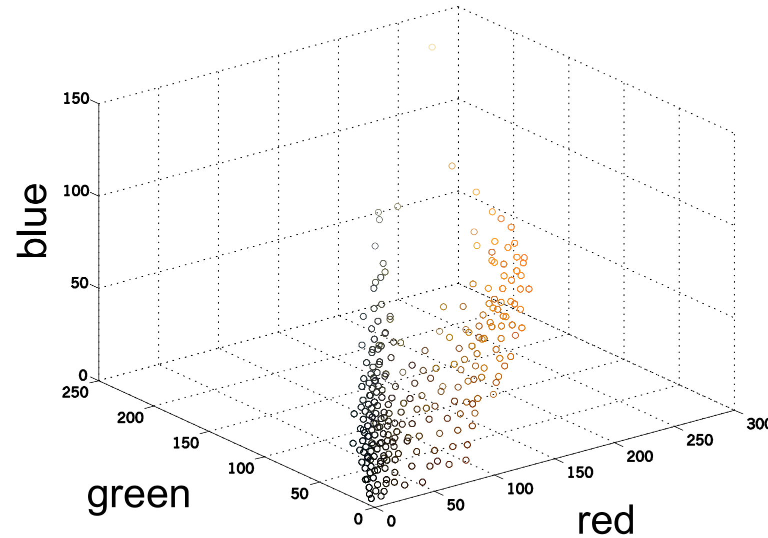
(Original Slide by M Mirmehdi)

From SPS

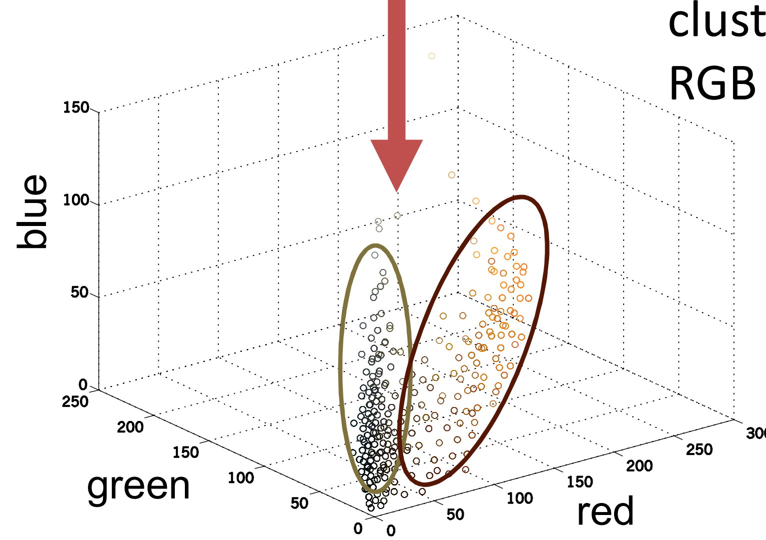
Example: Clustering for image segmentation

map to 3D

RGB space

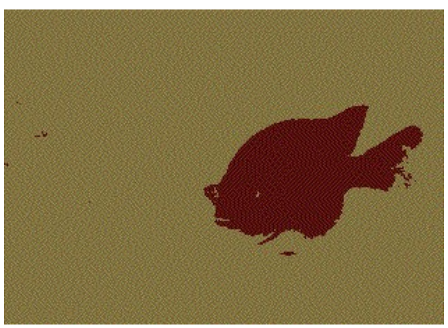


clustering in RGB space



map back to

pixel space



(Original Slide by M Mirmehdi)

K-means clustering – theoretical view

- We are effectively minimising the following objective function:

$$\Theta(clusters, data) = \sum_{j \in clusters} \left[\sum_{i \in j^{th} cluster} \left\| \mathbf{x}_i - \boldsymbol{\mu}_j \right\|^2 \right]$$

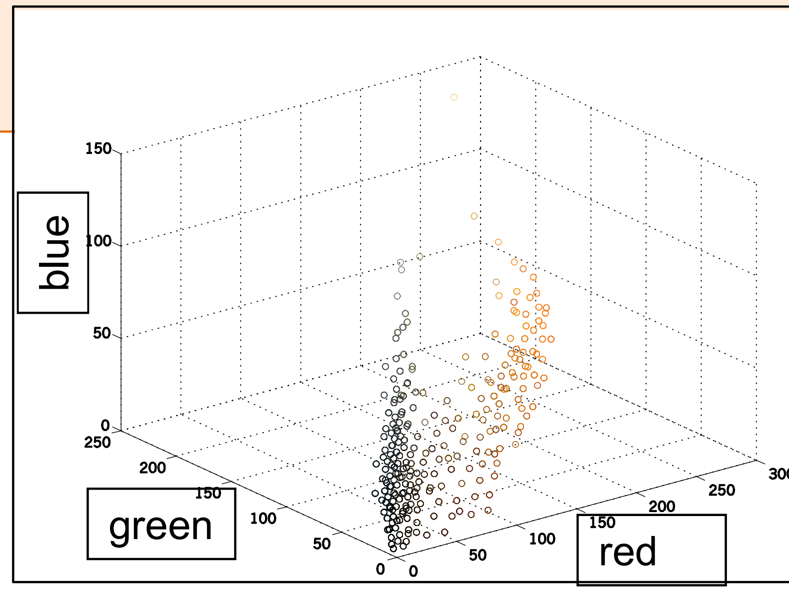
- **K-means iterates through two activities…**
  - Assumes the cluster centres are known and allocates each point to the closest cluster centre
  - Assumes the allocation is known and chooses a new set of cluster centres, with each centre being the mean of the points allocated to that cluster.
- **....until it converges to a local minimum of the objective function**.

- The starting K points are chosen randomly.

(Original Slide by M Mirmehdi)

K-means clustering

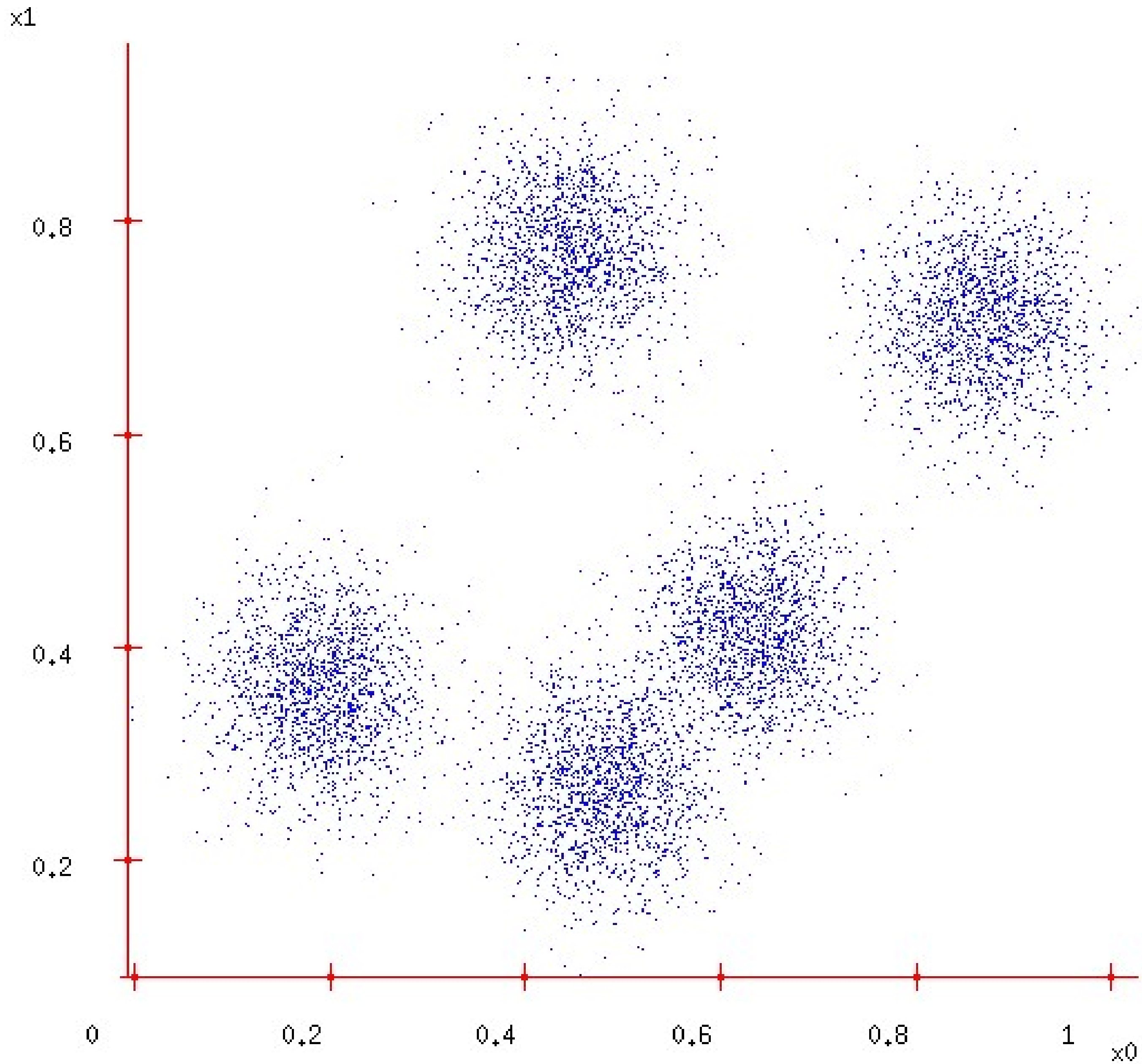
function KMeans(*Instances*,K)
  randomly initialise K vectors $\mu_1 \ldots \mu_K$;
  repeat
    assign each $x \in$ *Instances* to the nearest μ_j;
    recompute each μ_j as the mean of the
      instances assigned to it;
  until no change in $\mu_1 \ldots \mu_K$;
  return $\mu_1 \ldots \mu_K$;



(Original Slide by M Mirmehdi)

Example

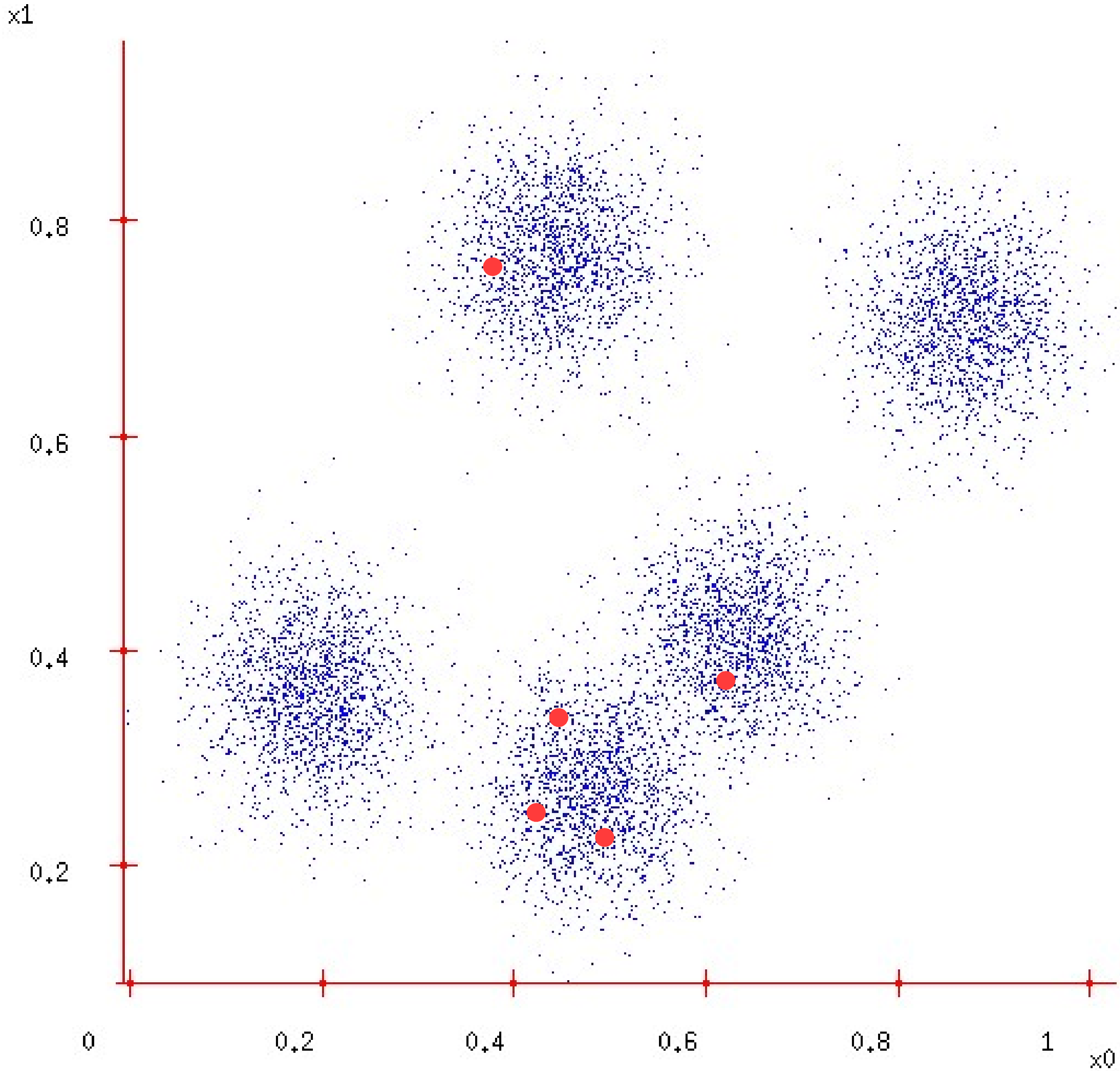
1. Ask user how many clusters they'd like (e.g., *K*=5)



Example courtesy of Andrew Moore

Example

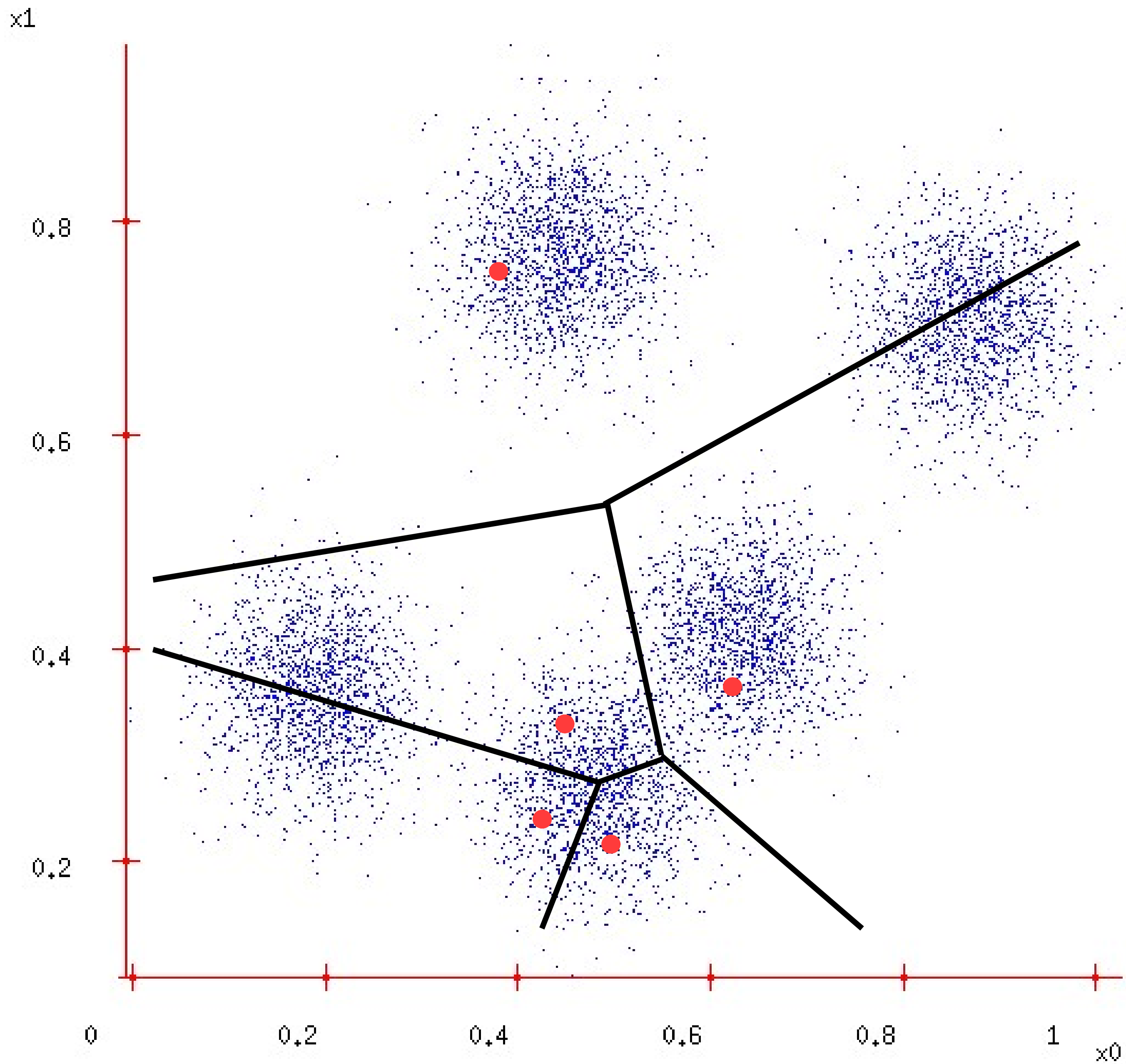
1. Ask user how many clusters they'd like (e.g., K=5)
2. Randomly guess K cluster centre locations (μ_1 … μ_K)



Example courtesy of Andrew Moore

Example

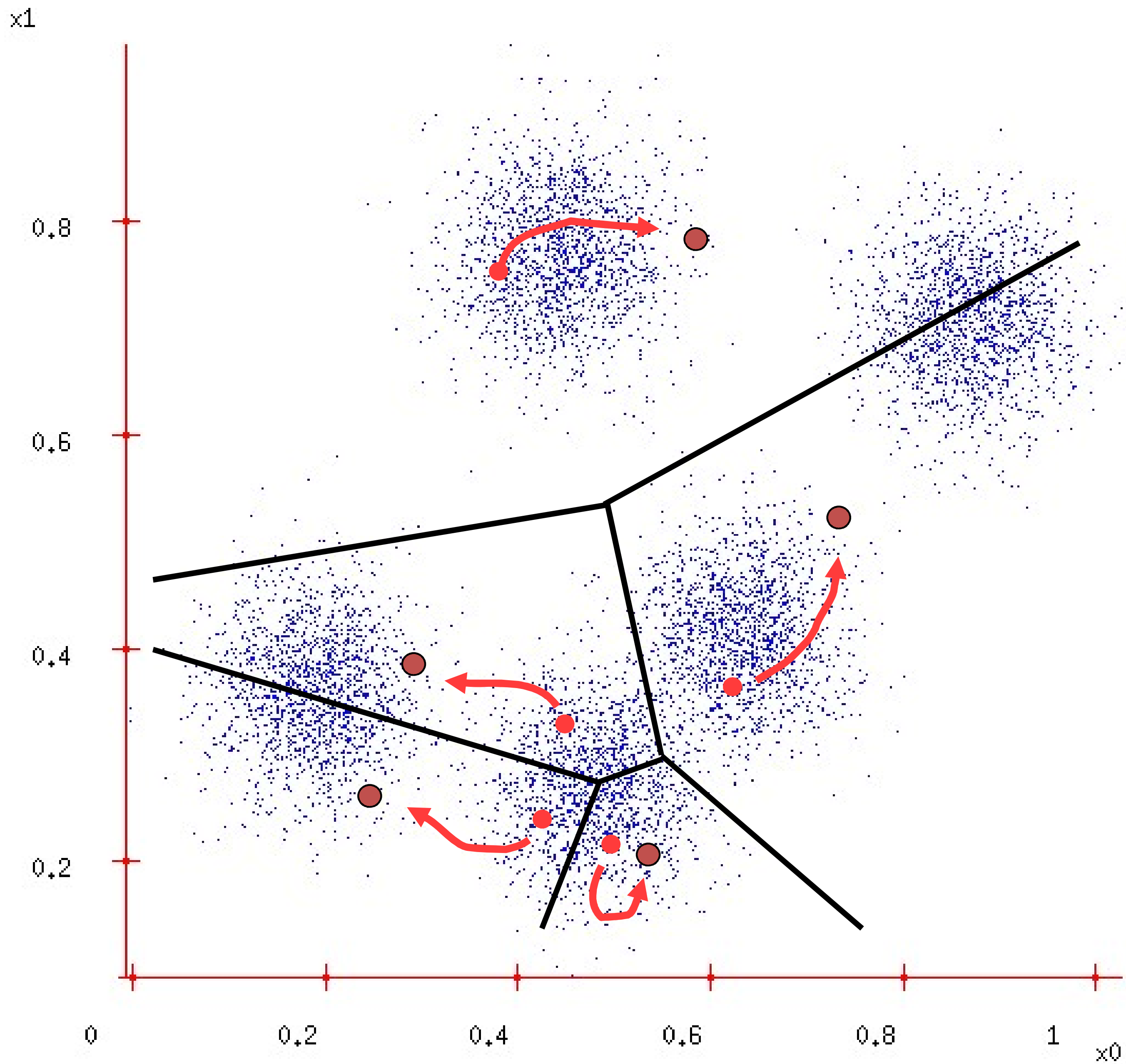
1. Ask user how many clusters they'd like (e.g., K=5)
2. Randomly guess K cluster centre locations ($\mu_1 \ \ldots \ \mu_K$)
3. Each datapoint finds out which centre it's closest to (thus each centre "owns" a set of datapoints)



Example courtesy of Andrew Moore

Example

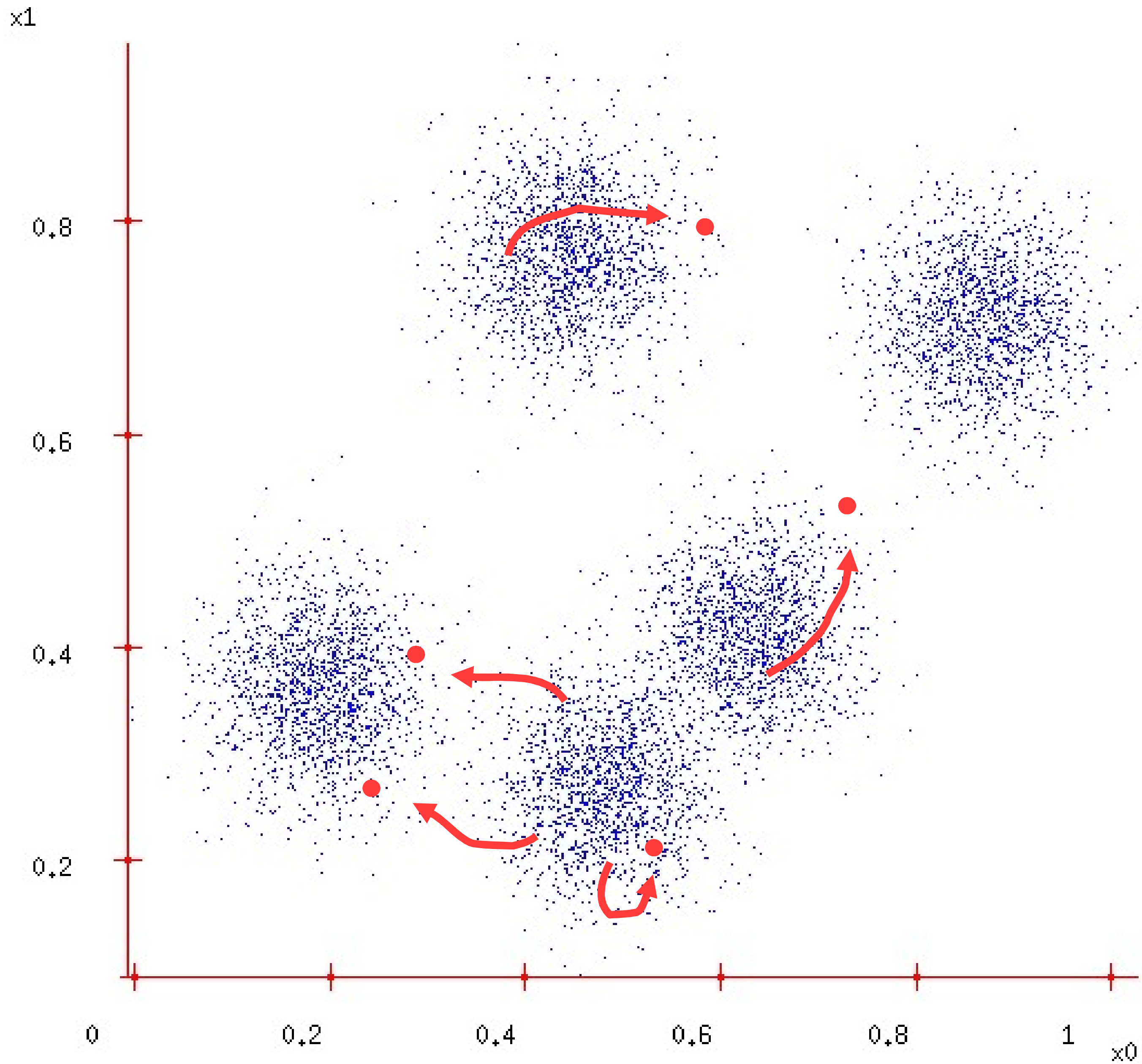
1. Ask user how many clusters they'd like (e.g., K=5)
2. Randomly guess K cluster centre locations ($\mu_1 \ \dots \ \mu_K$)
3. Each datapoint finds out which centre it's closest to
4. Each centre finds the centroid of the points it owns



Example courtesy of Andrew Moore

Example

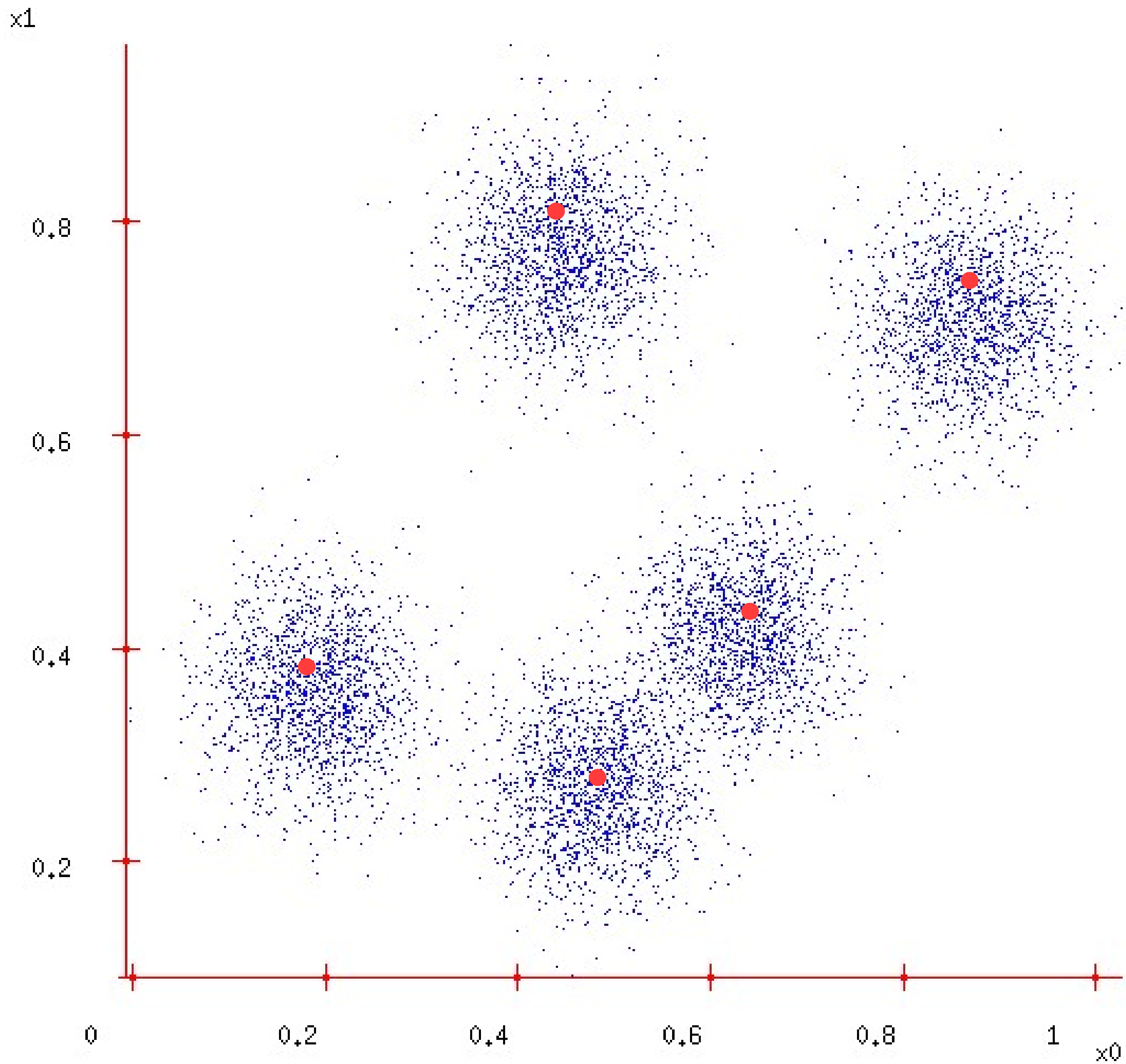
1. Ask user how many clusters they'd like (e.g., K=5)
2. Randomly guess K cluster centre locations (μ_1 … μ_K)
3. Each datapoint finds out which centre it's closest to
4. Each centre finds the centroid of the points it owns…
5. …and jumps there
6. Repeat until terminated!



Example courtesy of Andrew Moore

Example

1. Ask user how many clusters they'd like (e.g., K=5)
2. Randomly guess K cluster centre locations (μ_1 … μ_K)
3. Each datapoint finds out which centre it's closest to
4. Each centre finds the centroid of the points it owns…
5. …and jumps there
6. Repeat until terminated!



Example courtesy of Andrew Moore

Reflection on the k-means Algorithm

- **What does it do?**
  - K-means attempts to find a configuration $\mu_1 \ldots \mu_K$ that minimises within-cluster scatter: total squared distance between point xi and centroid μ_j in jth cluster:

$$\sum_i \left\| \mathbf{x}_i - \boldsymbol{\mu}_j \right\|^2$$

  - This is equivalent to maximising the between-cluster scatter (total squared distance between each cluster centroid and the global centroid of all points)

- **Does it work?**

1. The algorithm terminates.
2. It finds a local optimum from which no further improvement is possible by making local changes.
3. It does not necessarily find a global optimum.