AMME4710: COMPUTER VISION AND IMAGE PROCESSING WEEK 6

Dr. Mitch Bryson

School of Aerospace, Mechanical and Mechatronic Engineering, University of Sydney

Last Week

- Introduction to projective geometry and stereo vision
  - Camera Models
  - Stereo Vision, Triangulation and Dense Stereo
  - Camera Geometric Calibration

This Week's Lecture

- Image Segmentation

- Learning Objectives:
  - To gain an understanding of image-based segmentation algorithms and the underlying fundamentals in data clustering

Image Segmentation

- Segmentation is the process of partitioning an image into a series of regions that assist in analysis
- Regions may represent objects (or parts thereof) of pixels that have a meaningful grouping or relationship

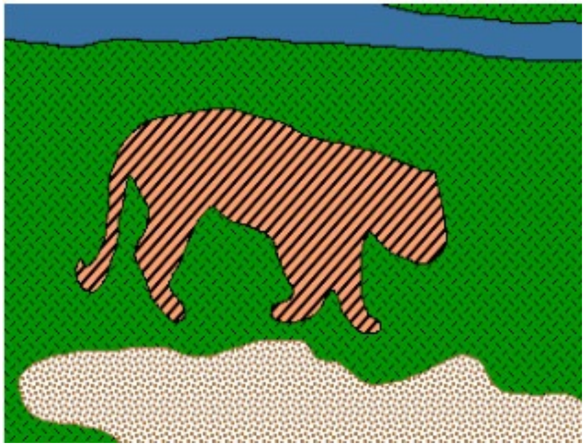
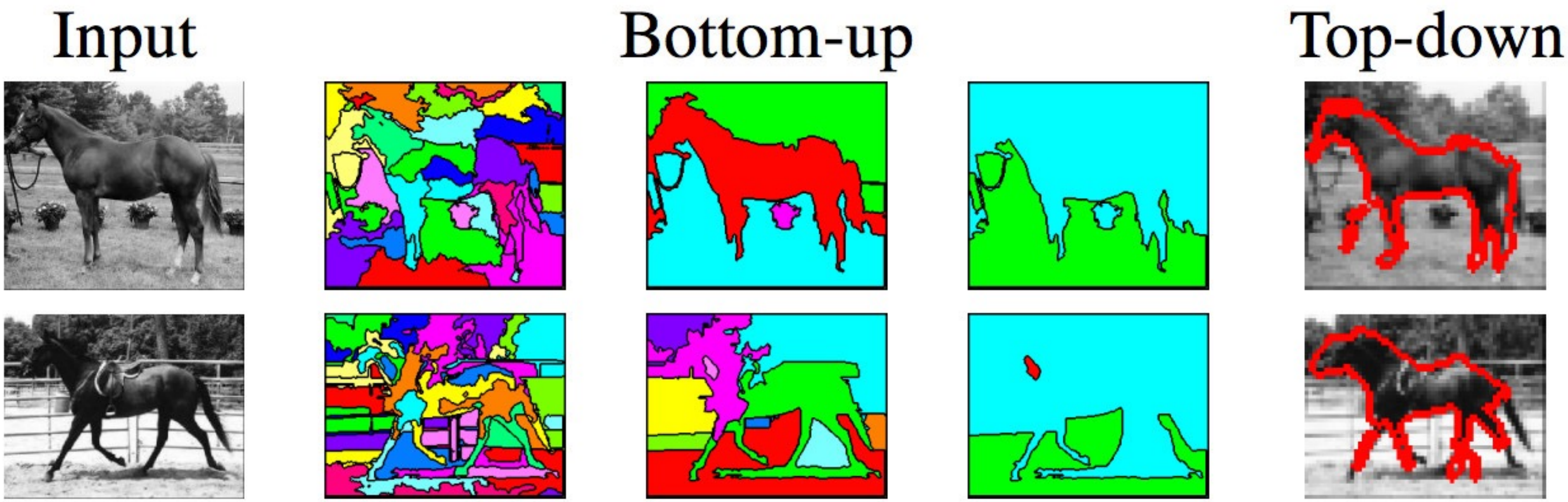


Image Segmentation

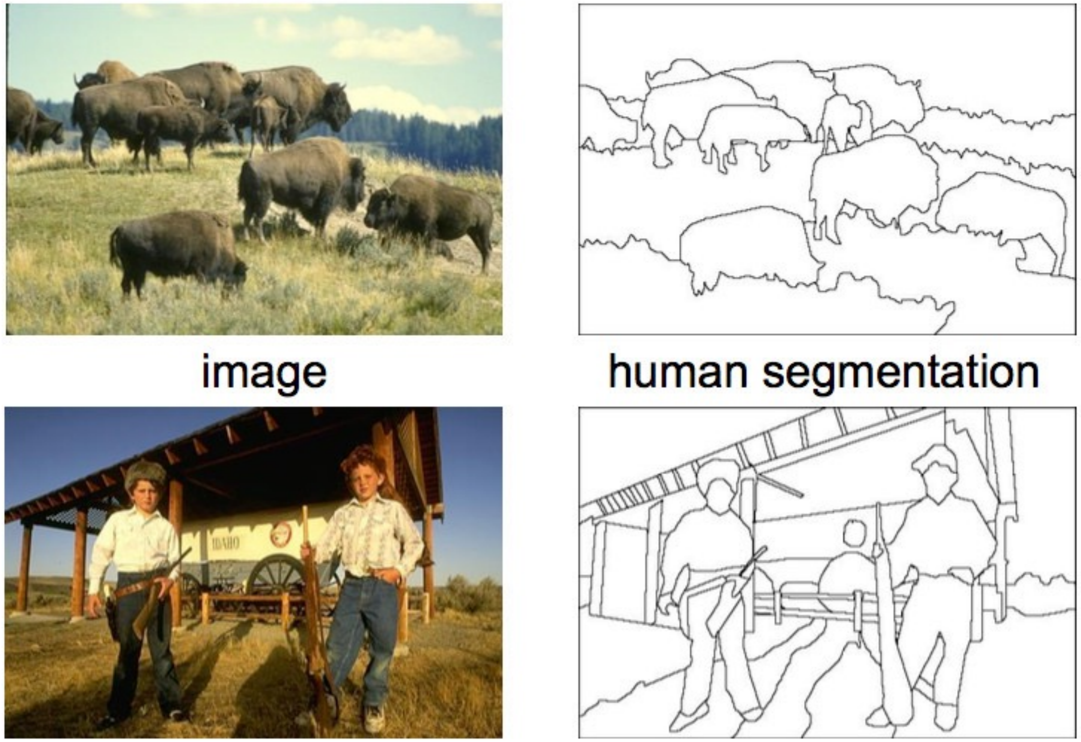
- Image segmentation can be approached from a ”bottom-up” or “top-down” way:
- **Bottom-up**: find groups of pixels or image regions that go together based on similarity (colour, texture etc.)
  - Typically unsupervised, and provides inputs to higher level reasoning tasks
- **Top-down**: delineate pixels based on a specific object
  - Typically supervised, related to classification



E. Borenstein, E. Sharon, S. Ullman, “Combining Top-down and Bottom-up Segmentation”, Computer Vision and Pattern Recognition Workshop, 2004.

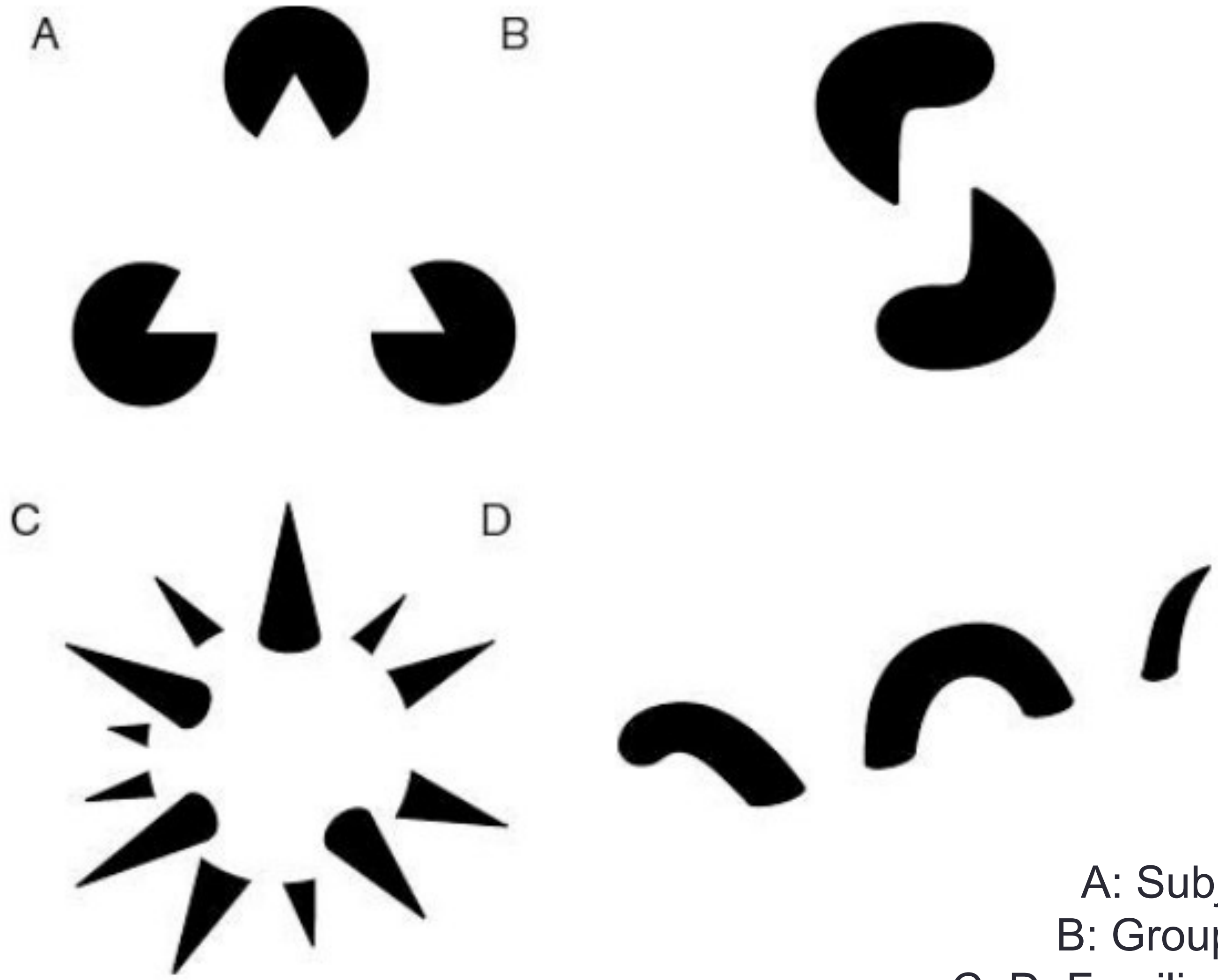
Image segmentation and visual perception

- Humans segment objects based on a variety of perceptive and psychological cues
- Humans typically account for contextual factors in segmentation
- Human perception usually occurs in a top-down fashion: challenge in image segmentation is to develop bottom-up approaches that capture segmentation properties in a similar way



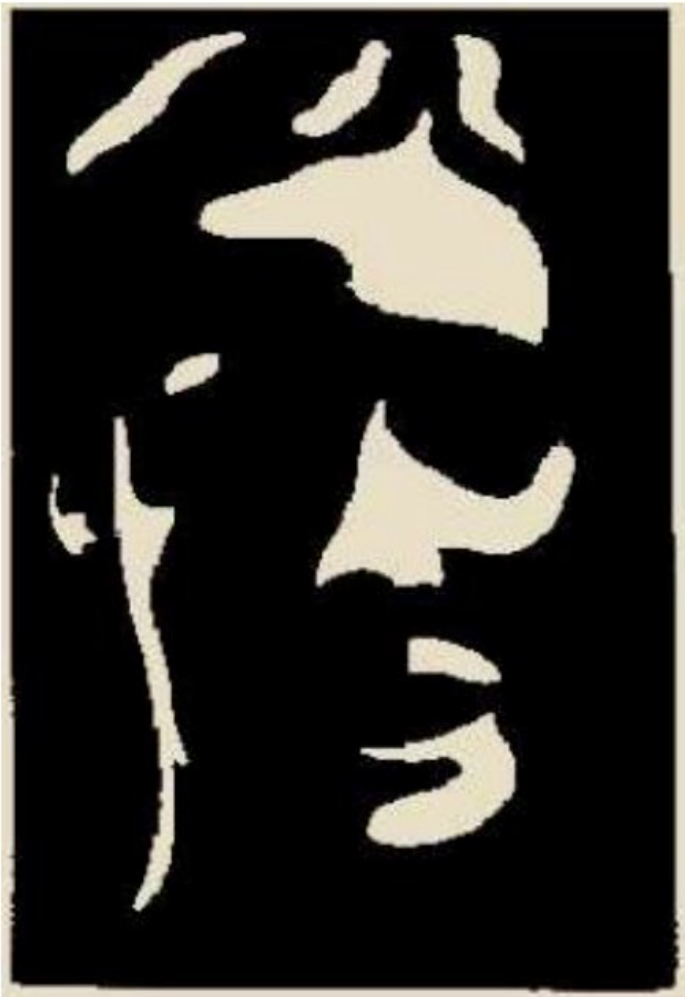
Berkeley Segmentation Database:
https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

Image segmentation and visual perception



A: Subjective contour
B: Group by Occlusion
C, D: Familiar Configuration

Image segmentation and visual perception

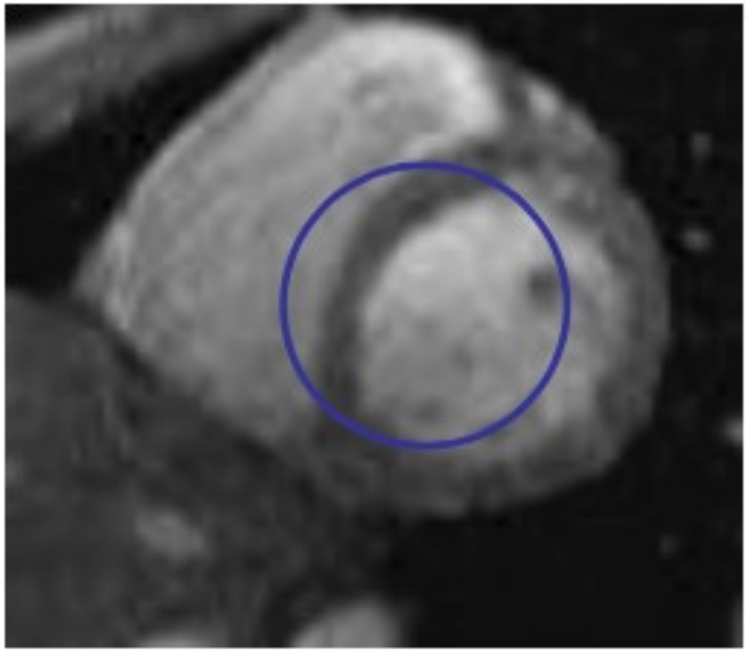


Familiarity

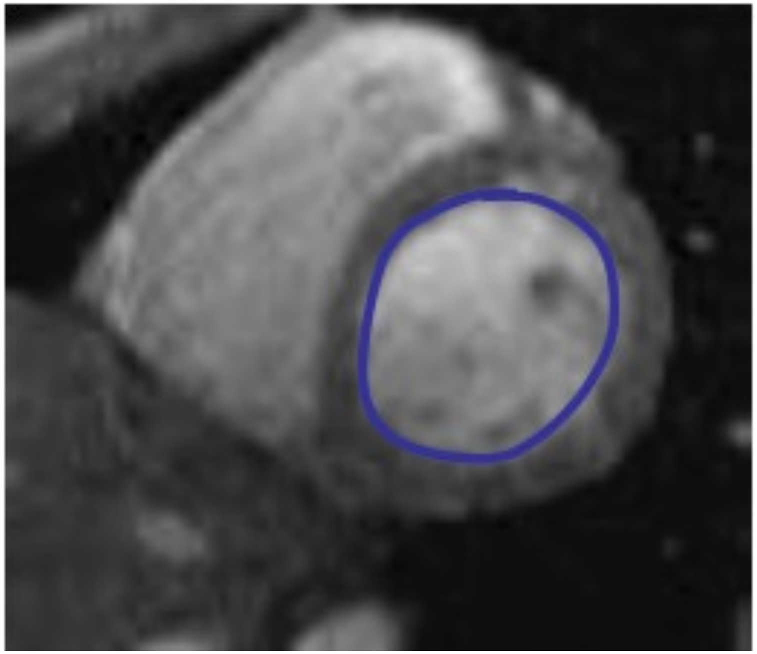
Common Fate

Active Contours and Snakes

- Active contours is a segmentation technique that attempts to refine an existing contour to best “fit” a target shape in the image
- An iterative refinement is used to drive the contour to overlap an area with high edge magnitude and maintain a specific type of “shape”
- Example of a top-down approach to segmentation



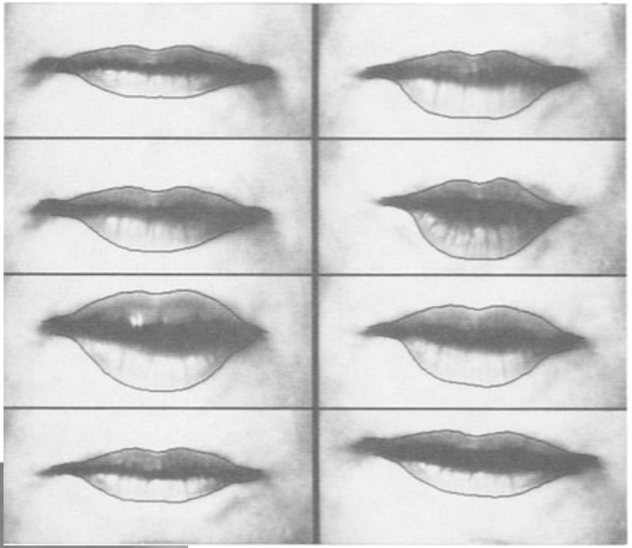
Initial contour

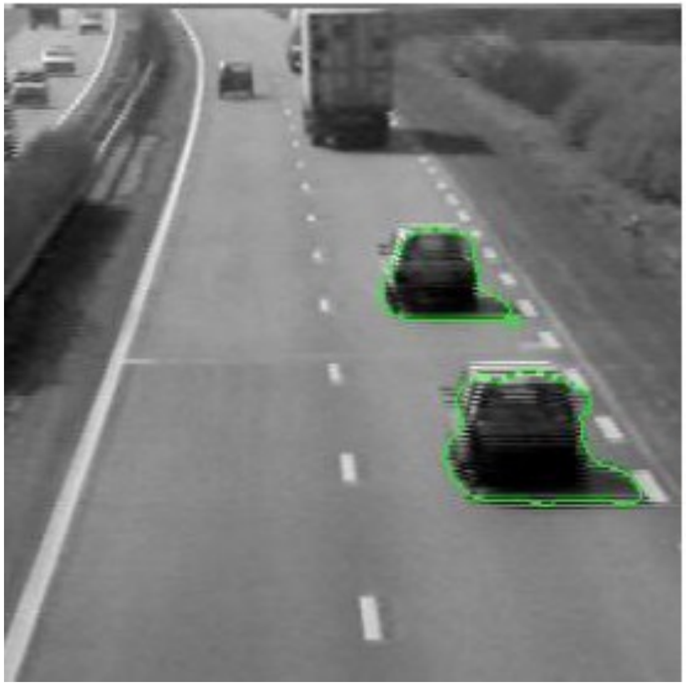


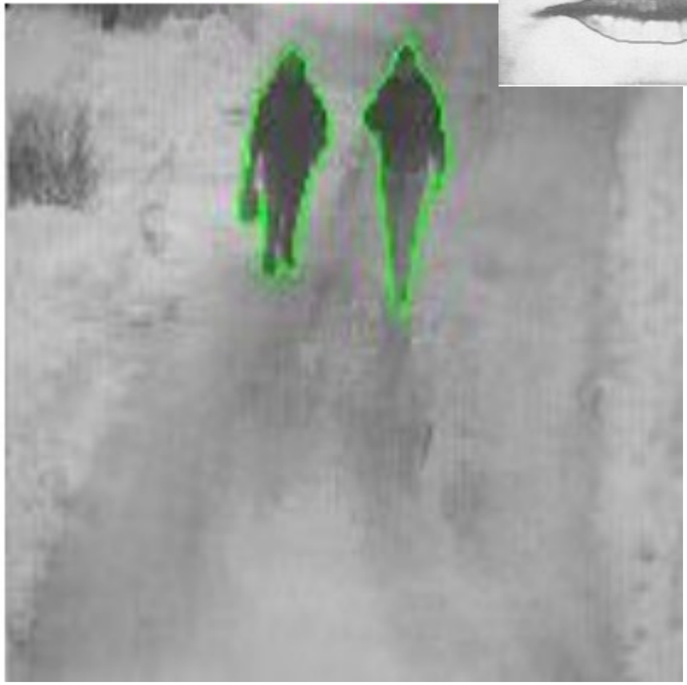
Final contour

Active Contours and Snakes

- Active contours can be used to track a deformable objects outline when an initial object segmentation has been achieved

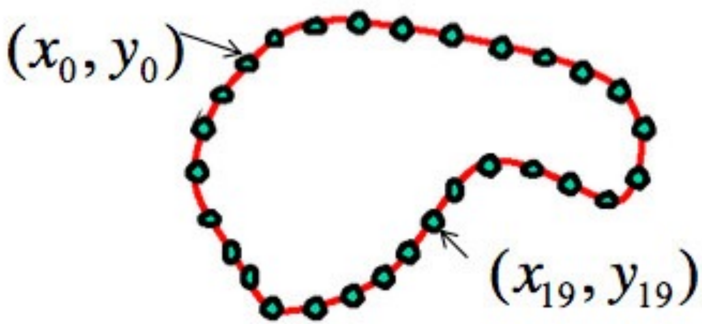






Active Contours and Snakes

- A contour is represented by a series of connected 2D vertices



$$v_i = (x_i, y_i),$$

for $i = 0, 1, \ldots, n-1$

- In fitting the active contour to the image, an energy function is minimised:

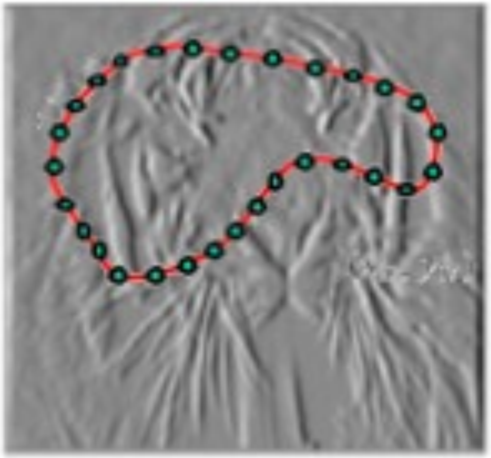
$$E_{total} = E_{internal} + E_{external}$$

- Where $E_{internal}$ is a function of the current shape of the contour and $E_{external}$ represents the interaction with an edge image based on the contours current position

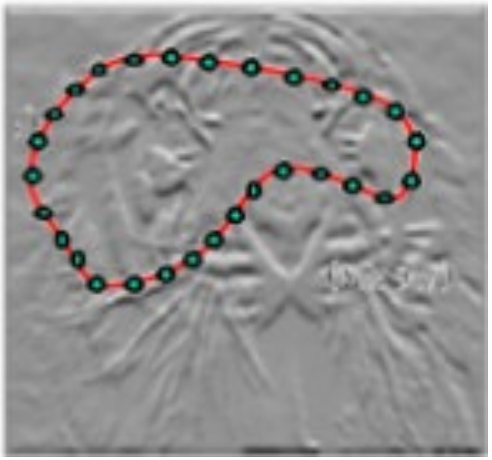
Active Contours and Snakes: External Energy

- The external energy of the contour is negatively proportionate to the magnitude of the gradient of the image pixels associated with the current position of each vertex in the contour

Gx



Gy



$$E_{external}(v) = -(|G_x(v)|^2 + |G_y(v)|^2)$$

$$E_{external} = -\sum_{i=0}^{n-1} |G_x(x_i, y_i)|^2 + |G_y(x_i, y_i)|^2$$

Where Gx is the x-axis gradient, Gy the y-axis gradient, E(v) the energy per vertex v

Active Contours and Snakes: Internal Energy

- The internal energy of the contour is based on two terms that control the elasticity and curvature of the snake based on the relative location of vertices

$$E_{internal}(v(s)) = \boxed{\alpha\left|\frac{dv}{ds}\right|^2} + \boxed{\beta\left|\frac{d^2v}{d^2s}\right|^2}$$

Tension, Elasticity

Stiffness, Curvature

- The spatial derivatives can be approximated at each vertex using the surrounding vertex values:

$$E_{internal} = \sum_{i=0}^{n-1} \alpha\left\|v_{i+1} - v_i\right\|^2 + \beta\left\|v_{i+1} - 2v_i + v_{i-1}\right\|^2$$

Active Contours and Snakes: Internal Energy

- The curvature terms penalises sharp changes in curvature to help maintain a smooth spatial fit:

$$E_{curvature}(v_i) = \left\| v_{i+1} - 2v_i + v_{i-1} \right\|^2$$
$$= (x_{i+1} - 2x_i + x_{i-1})^2 + (y_{i+1} - 2y_i + y_{i-1})^2$$

- The elasticity term serves to minimise the perimeter of the shape and maintain a tight fit to the object

$$E_{elastic} = \sum^{n-1} \alpha \left\| v_{i+1} - v_i \right\|^2$$
$$= \alpha \cdot \sum_{i=0}^{n-1} \left((x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 - \bar{d} \right)^2$$

Where $\bar{d}$ is the average distance between vertices

Active Contours and Snakes

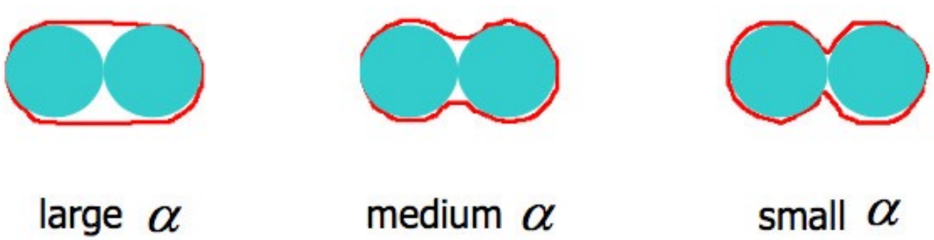
- The final energy term is represented as a weighted sum of external, curvature and elasticity terms:

$$E_{total} = E_{internal} + \gamma E_{external}$$

$$E_{external} = -\sum_{i=0}^{n-1} |G_x(x_i, y_i)|^2 + |G_y(x_i, y_i)|^2$$

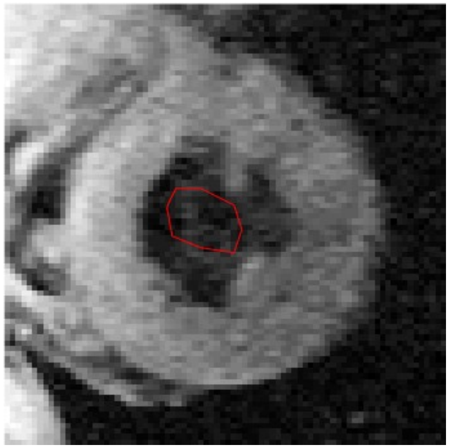
$$E_{internal} = \sum_{i=0}^{n-1} \alpha\left(\bar{d} - \left\|v_{i+1} - v_i\right\|\right)^2 + \beta\left\|v_{i+1} - 2v_i + v_{i-1}\right\|^2$$

- The different parameter weights control the final fit parameters

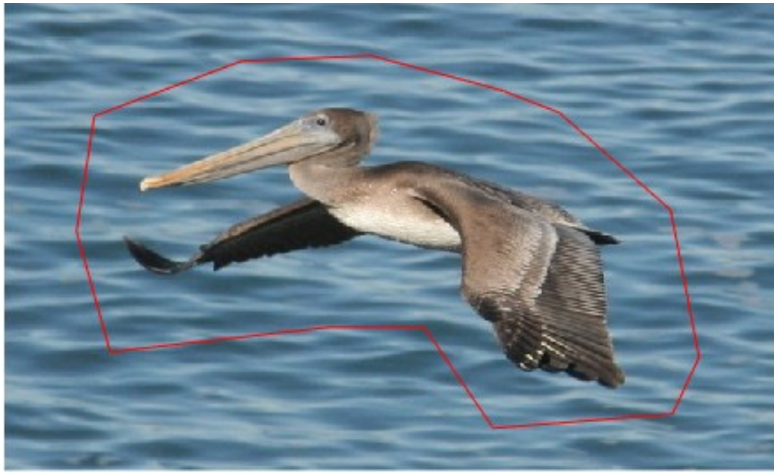


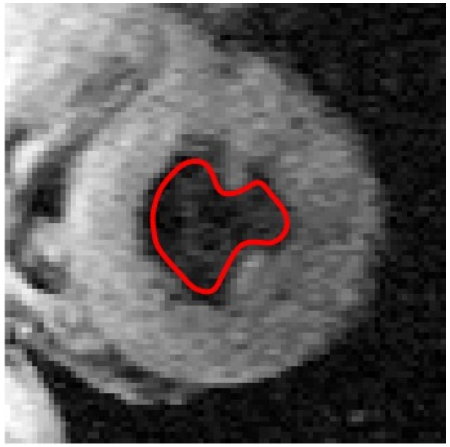
Active Contours and Snakes

- An optimisation strategy is then used to iteratively refine the positions of each vertex to reach a minimal energy state

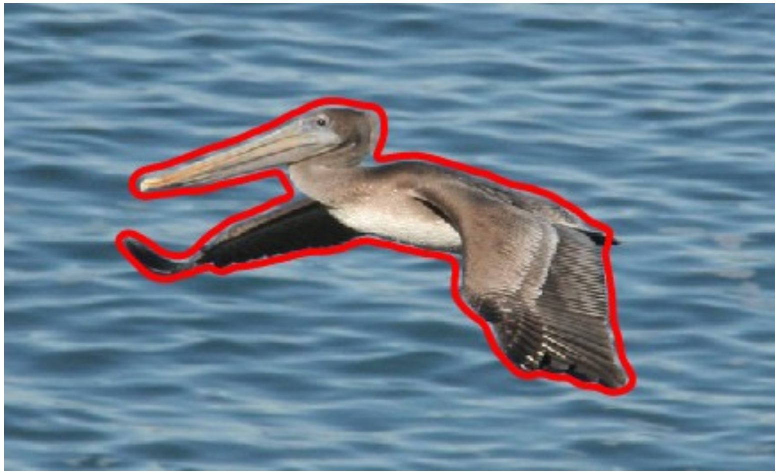


Initial contour





Final contour

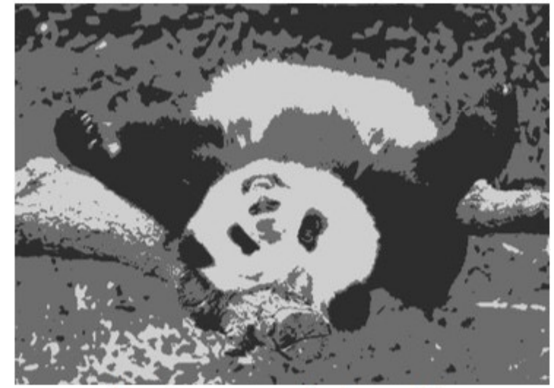


Segmentation by Clustering

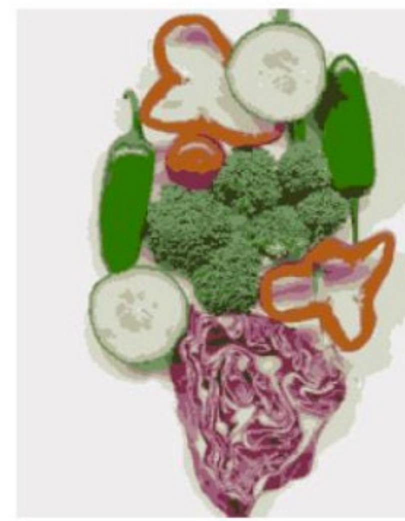
- **Clustering** is the process of splitting a group of objects up into subsets (clusters) for which objects in one cluster are more "similar" to one another than objects in another cluster

- In the context of image segmentation, the objects are pixels and the characteristics that make them similar or not can be based on any local image properties (i.e. intensity, colour, texture, spatial location etc.)


image


clusters on intensity


image

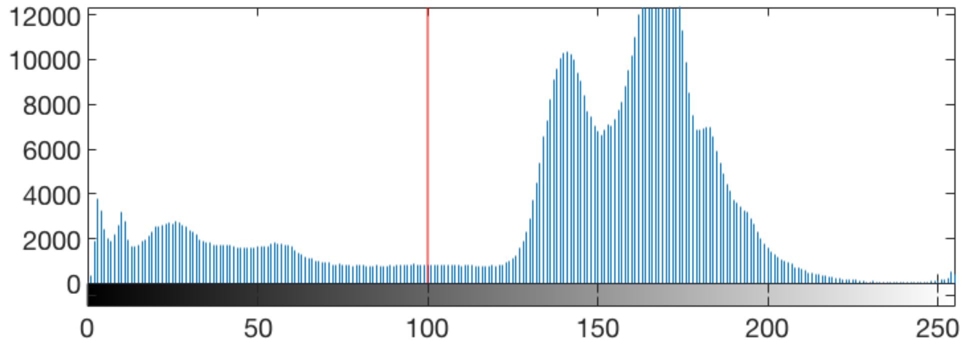

clusters on color

Clustering into two groups: thresholding

- When data is split into two clusters the process is equivalent to thresholding

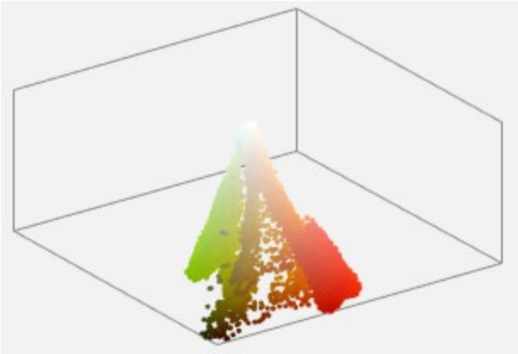
$$\boxed{\sigma^2_{intra}(t) = \omega_0(t)\sigma^2_0(t) + \omega_1(t)\sigma^2_1(t)}$$

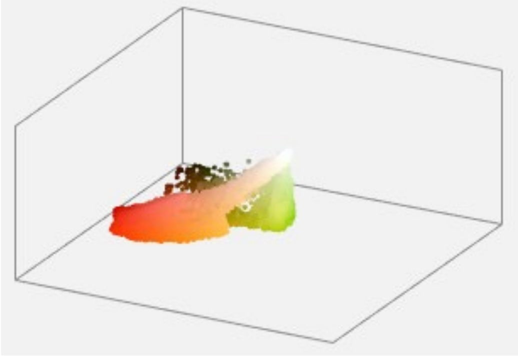
- For example, Otsu's method defines a threshold that minimises the intra-class variance in pixel intensity distributions of foreground and background classes
  - Clustering (as opposed to thresholding) is a more general technique that can provide more than two clusters and work with multi-dimensional feature/inputs spaces



Clustering: Feature Space

- During clustering-based image segmentation, pixels are treated as points in a feature space
- For example, colour-based clustering might base a feature space around the RGB coordinates of the pixel values
- Similarity between points is then measured based on the distance in this feature space



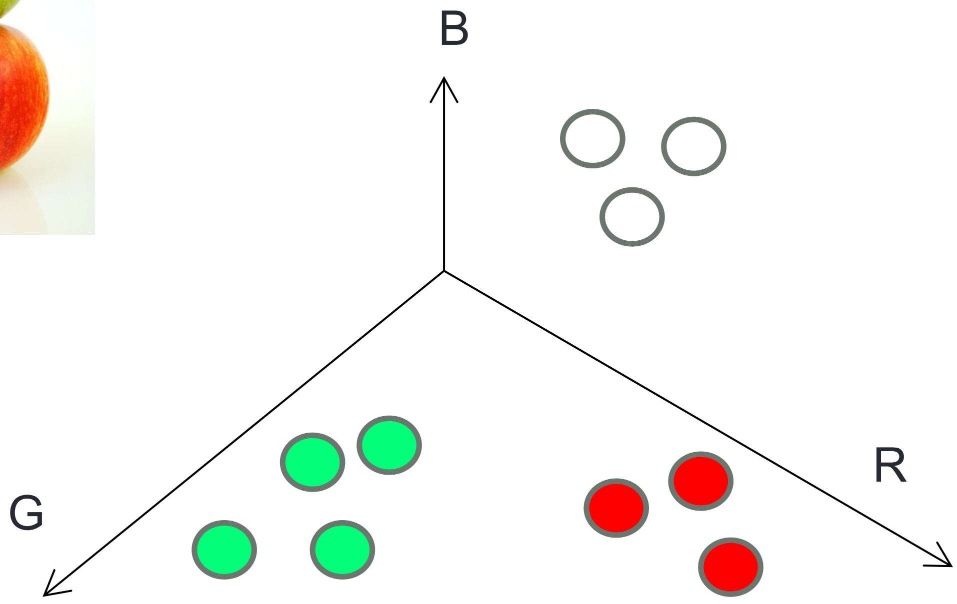


Clustering: Feature Space

- During clustering-based image segmentation, pixels are treated as points in a feature space

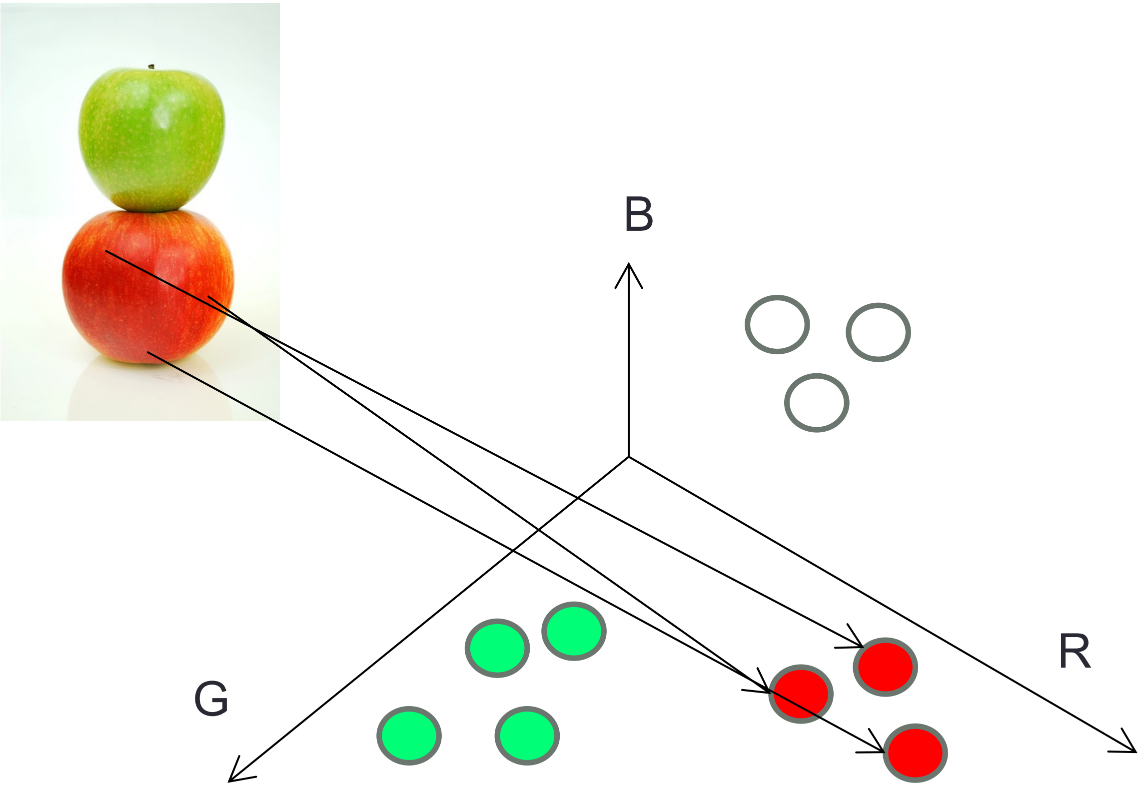
- For example, colour-based clustering might base a feature space around the RGB coordinates of the pixel values

- Similarity between points is then measured based on the distance in this feature space



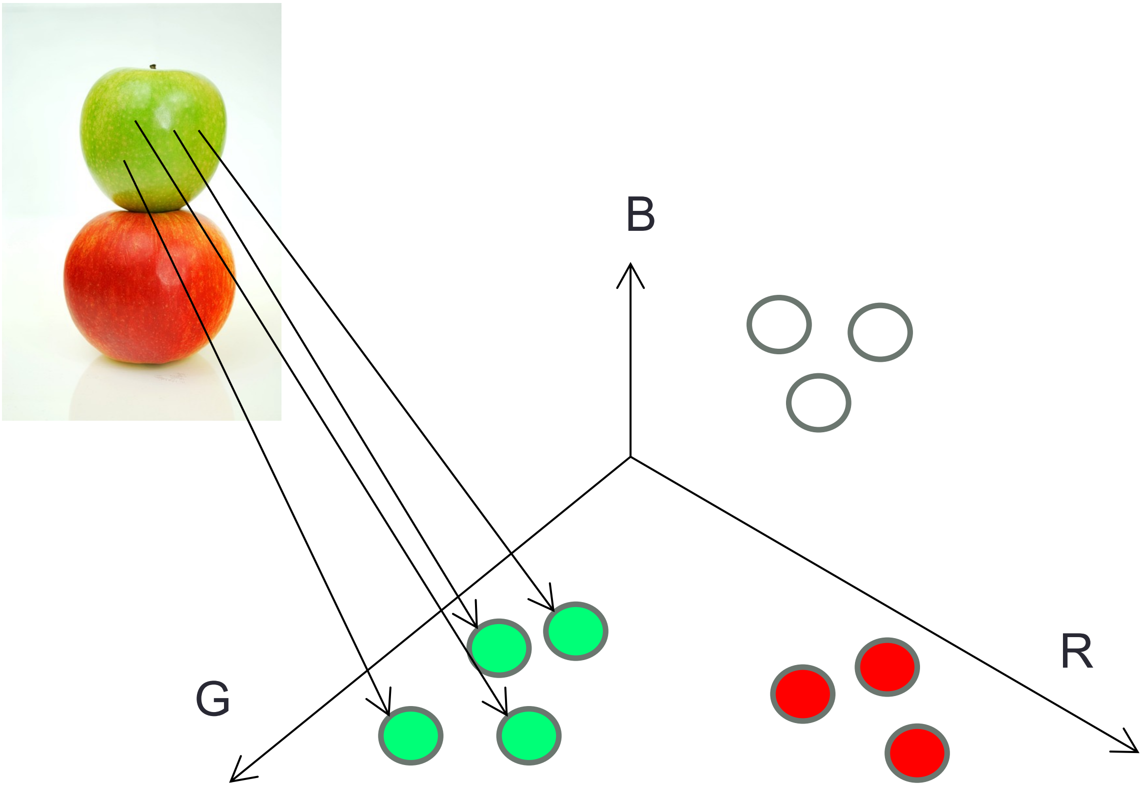
Clustering: Feature Space

- During clustering-based image segmentation, pixels are treated as points in a feature space
- For example, colour-based clustering might base a feature space around the RGB coordinates of the pixel values
- Similarity between points is then measured based on the distance in this feature space



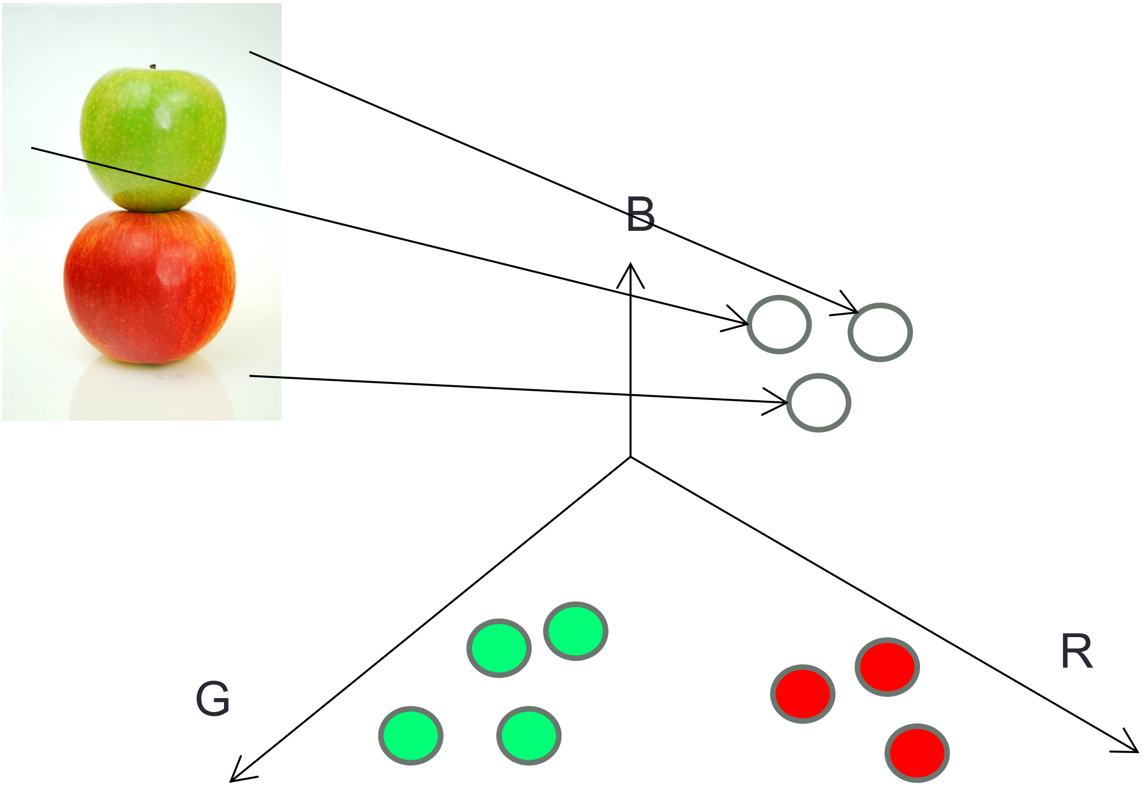
Clustering: Feature Space

- During clustering-based image segmentation, pixels are treated as points in a feature space
- For example, colour-based clustering might base a feature space around the RGB coordinates of the pixel values
- Similarity between points is then measured based on the distance in this feature space



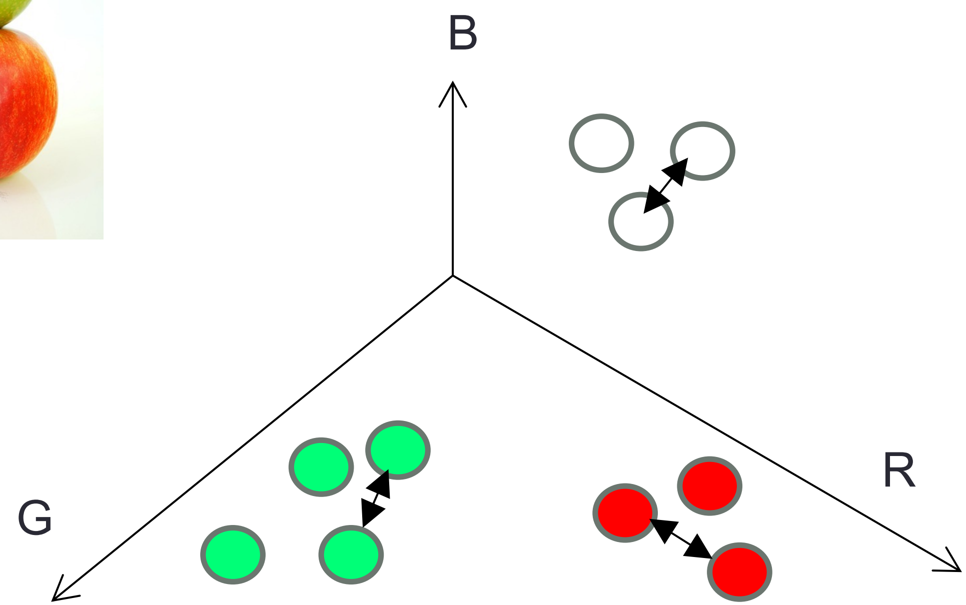
Clustering: Feature Space

- During clustering-based image segmentation, pixels are treated as points in a feature space
- For example, colour-based clustering might base a feature space around the RGB coordinates of the pixel values
- Similarity between points is then measured based on the distance in this feature space



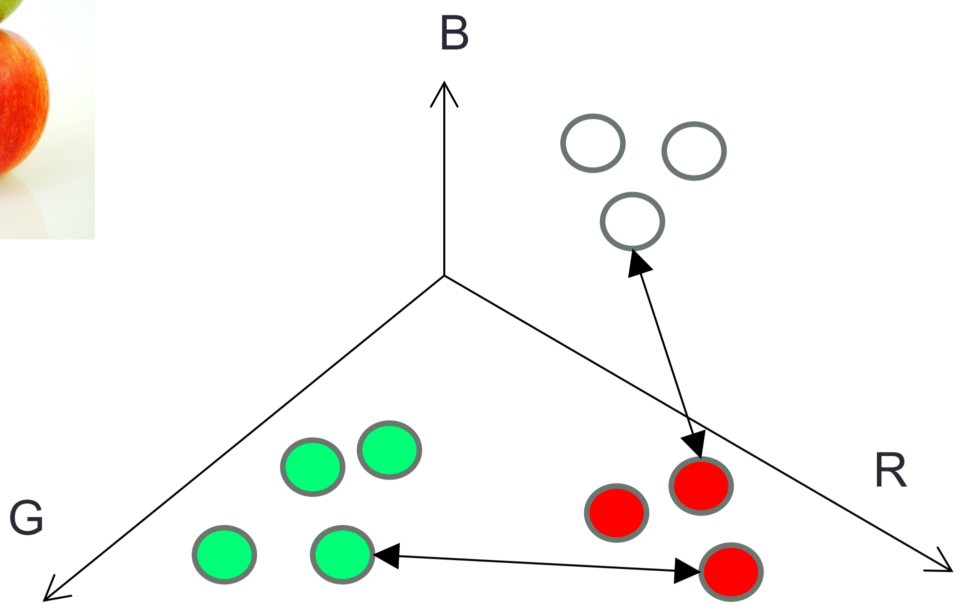
Clustering: Feature Space

- During clustering-based image segmentation, pixels are treated as points in a feature space
- For example, colour-based clustering might base a feature space around the RGB coordinates of the pixel values
- Similarity between points is then measured based on the distance in this feature space



Clustering: Feature Space

- During clustering-based image segmentation, pixels are treated as points in a feature space
- For example, colour-based clustering might base a feature space around the RGB coordinates of the pixel values
- Similarity between points is then measured based on the distance in this feature space

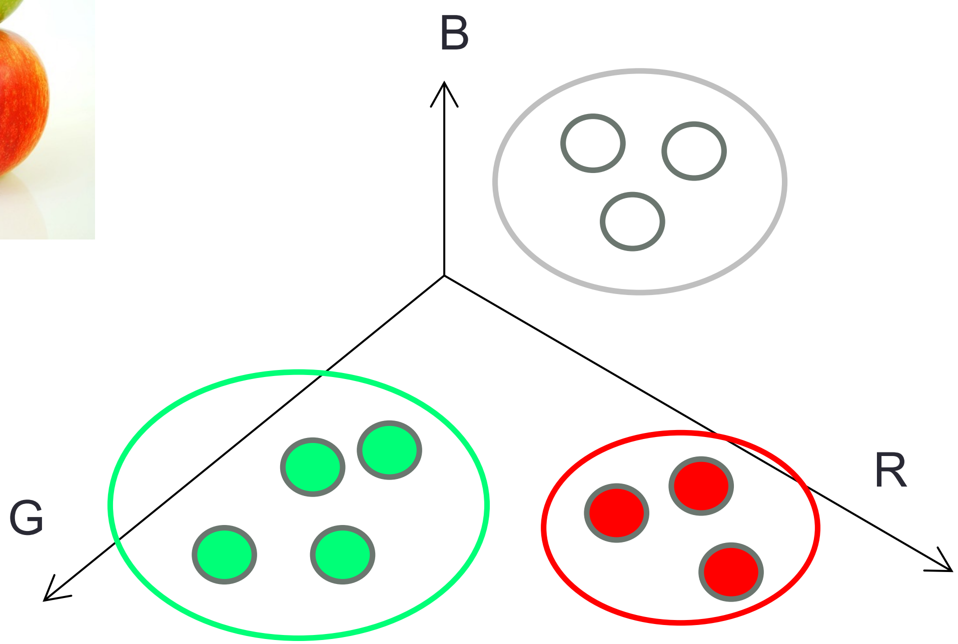


Clustering: Feature Space

- During clustering-based image segmentation, pixels are treated as points in a feature space

- For example, colour-based clustering might base a feature space around the RGB coordinates of the pixel values

- Similarity between points is then measured based on the distance in this feature space



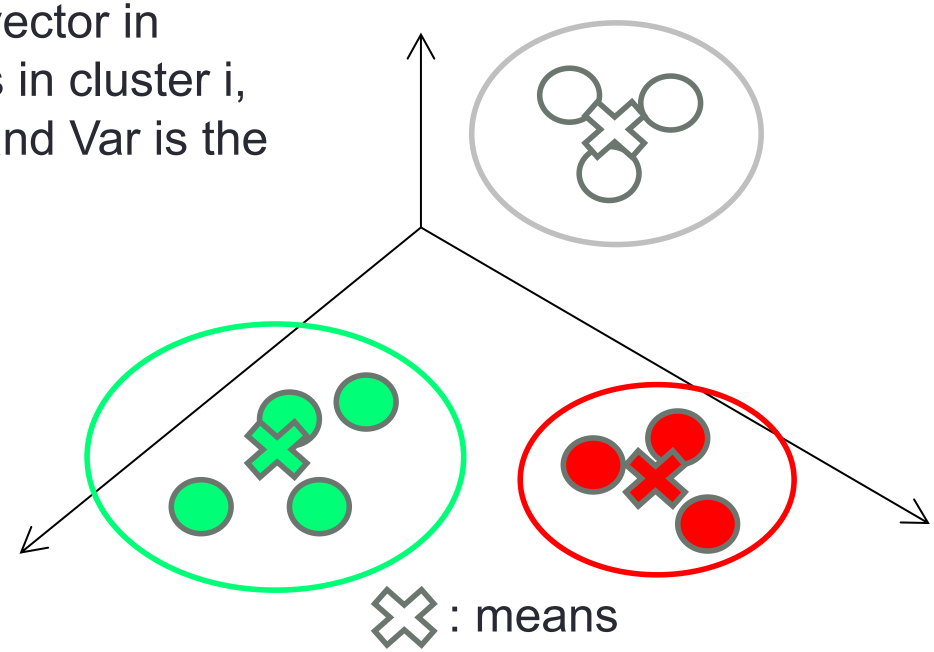
Clusters

K-means Clustering

- K-means is a simple clustering technique that splits data into a pre-defined number of clusters (K)
- The algorithm break the data into K clusters such that the within-cluster sum of squares is minimised:

$$\underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \left\| \mathbf{x} - \boldsymbol{\mu}_i \right\|^2 = \underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{k} |S_i| \operatorname{Var} S_i$$

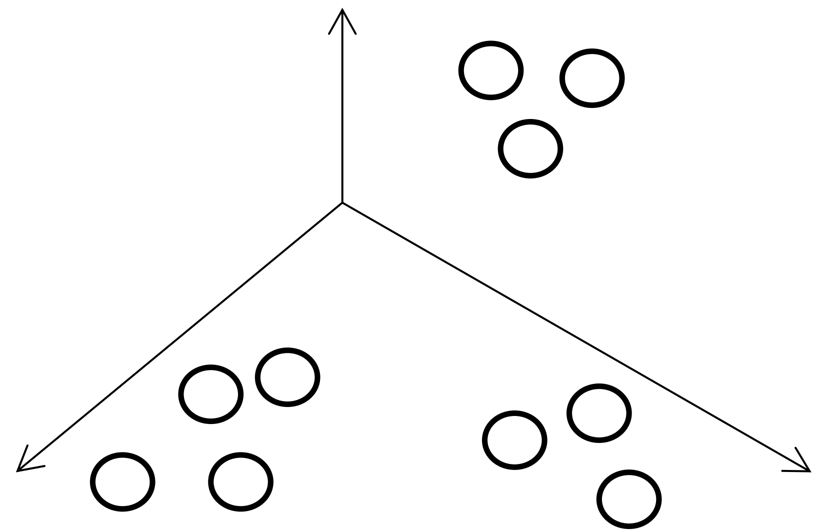
Where $\boldsymbol{\mu}_i$ is the mean of each cluster (vector in feature space), S_i is the set of all points in cluster i, $|S_i|$ is the number of points in cluster i and Var is the variance



K-means Clustering

- In-general minimising the within-cluster sum of squares objective function is computationally complex (NP-hard)
- In practice, heuristic methods can achieve approximate minimisations (local minima) in a computationally efficient manner

- Lloyd's Algorithm* is a common approach to solving k-means clustering:
  - (1) Pick K random points and assign each to one of the K clusters
  - (2) Assign all N points to each cluster based on their distance to the mean of all points in each cluster (to begin with each cluster only has a single point)
  - (3) Recalculate the means of each cluster based on all the points in each
  - (4) Repeat from step (2) until the assignments do not change from the last iteration (convergence)

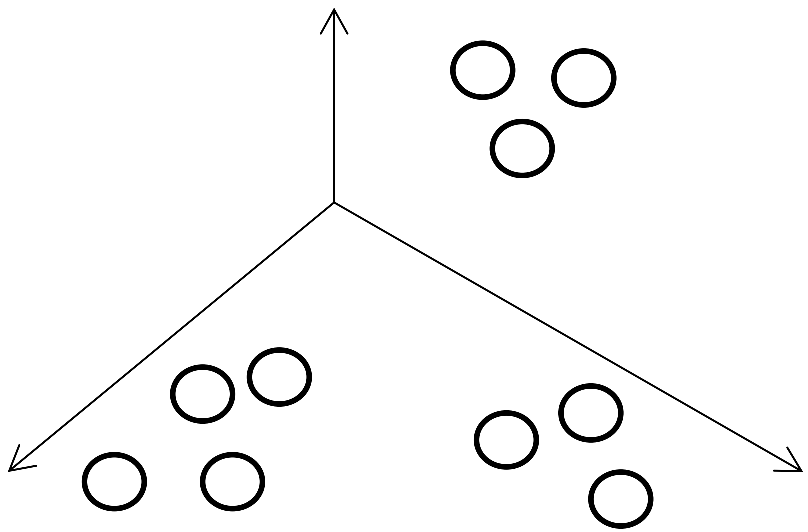


* S.P. Lloyd, "Least squares quantization in PCM", IEEE Transactions on Information Theory, 28-2:129–137, 1982.

K-means Clustering

- In-general minimising the within-cluster sum of squares objective function is computationally complex (NP-hard)
- In practice, heuristic methods can achieve approximate minimisations (local minima) in a computationally efficient manner

- Lloyd's Algorithm* is a common approach to solving k-means clustering:
  - (1) Pick K random points and assign each to one of the K clusters
  - (2) Assign all N points to each cluster based on their distance to the mean of all points in each cluster (to begin with each cluster only has a single point)
  - (3) Recalculate the means of each cluster based on all the points in each
  - (4) Repeat from step (2) until the assignments do not change from the last iteration (convergence)

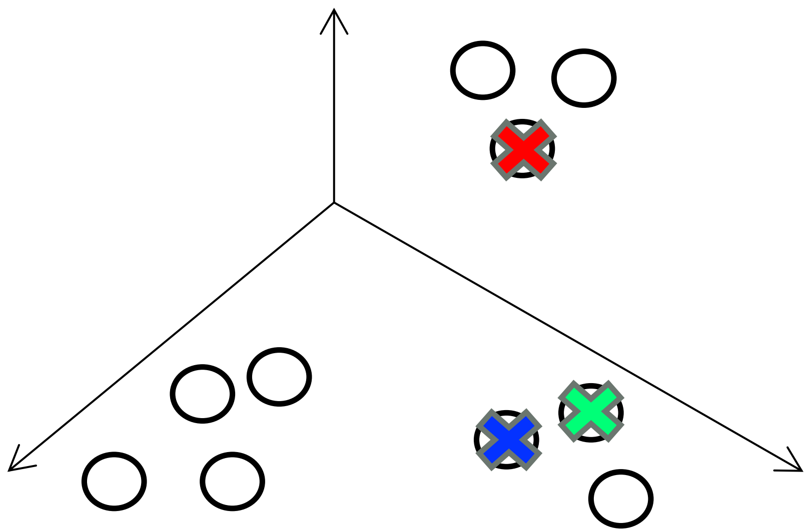


* S.P. Lloyd, "Least squares quantization in PCM", IEEE Transactions on Information Theory, 28-2:129–137, 1982.

K-means Clustering

- In-general minimising the within-cluster sum of squares objective function is computationally complex (NP-hard)
- In practice, heuristic methods can achieve approximate minimisations (local minima) in a computationally efficient manner

- Lloyd's Algorithm* is a common approach to solving k-means clustering:
  - (1) Pick K random points and assign each to one of the K clusters
  - (2) Assign all N points to each cluster based on their distance to the mean of all points in each cluster (to begin with each cluster only has a single point)
  - (3) Recalculate the means of each cluster based on all the points in each
  - (4) Repeat from step (2) until the assignments do not change from the last iteration (convergence)

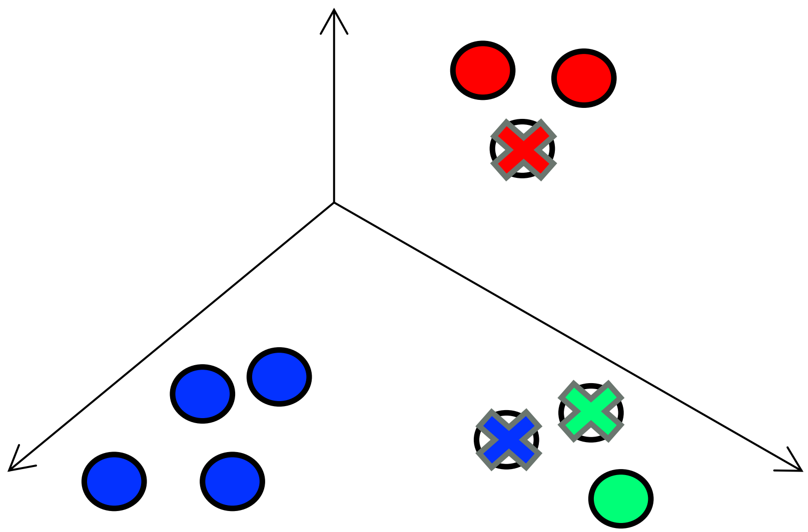


* S.P. Lloyd, "Least squares quantization in PCM", IEEE Transactions on Information Theory, 28-2:129–137, 1982.

K-means Clustering

- In-general minimising the within-cluster sum of squares objective function is computationally complex (NP-hard)
- In practice, heuristic methods can achieve approximate minimisations (local minima) in a computationally efficient manner

- Lloyd's Algorithm* is a common approach to solving k-means clustering:
  - (1) Pick K random points and assign each to one of the K clusters
  - (2) Assign all N points to each cluster based on their distance to the mean of all points in each cluster (to begin with each cluster only has a single point)
  - (3) Recalculate the means of each cluster based on all the points in each
  - (4) Repeat from step (2) until the assignments do not change from the last iteration (convergence)

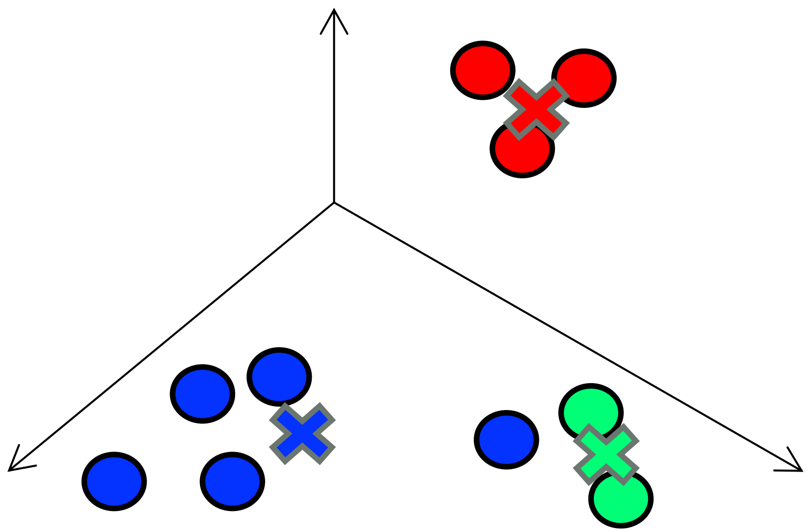


* S.P. Lloyd, "Least squares quantization in PCM", IEEE Transactions on Information Theory, 28-2:129–137, 1982.

K-means Clustering

- In-general minimising the within-cluster sum of squares objective function is computationally complex (NP-hard)
- In practice, heuristic methods can achieve approximate minimisations (local minima) in a computationally efficient manner

- Lloyd’s Algorithm* is a common approach to solving k-means clustering:
  - (1) Pick K random points and assign each to one of the K clusters
  - (2) Assign all N points to each cluster based on their distance to the mean of all points in each cluster (to begin with each cluster only has a single point)
  - (3) Recalculate the means of each cluster based on all the points in each
  - (4) Repeat from step (2) until the assignments do not change from the last iteration (convergence)

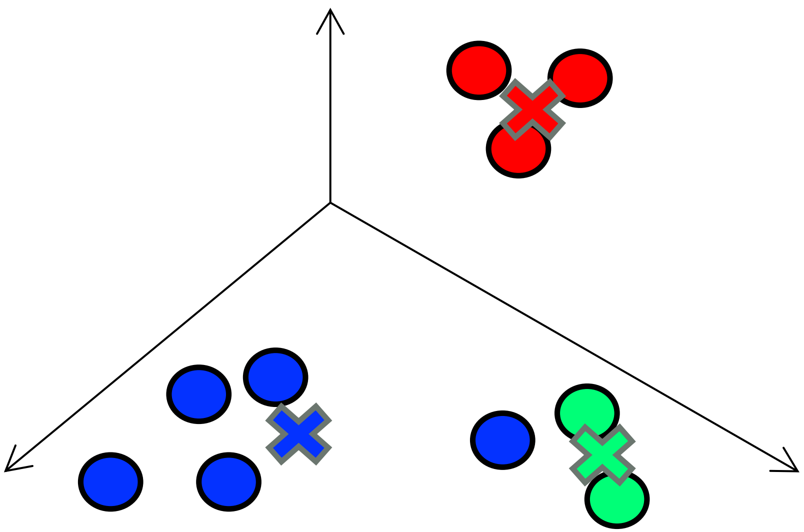


* S.P. Lloyd, “Least squares quantization in PCM", IEEE Transactions on Information Theory, 28-2:129–137, 1982.

K-means Clustering

- In-general minimising the within-cluster sum of squares objective function is computationally complex (NP-hard)
- In practice, heuristic methods can achieve approximate minimisations (local minima) in a computationally efficient manner

- Lloyd's Algorithm* is a common approach to solving k-means clustering:
  - (1) Pick K random points and assign each to one of the K clusters
  - (2) Assign all N points to each cluster based on their distance to the mean of all points in each cluster (to begin with each cluster only has a single point)
  - (3) Recalculate the means of each cluster based on all the points in each
  - (4) Repeat from step (2) until the assignments do not change from the last iteration (convergence)

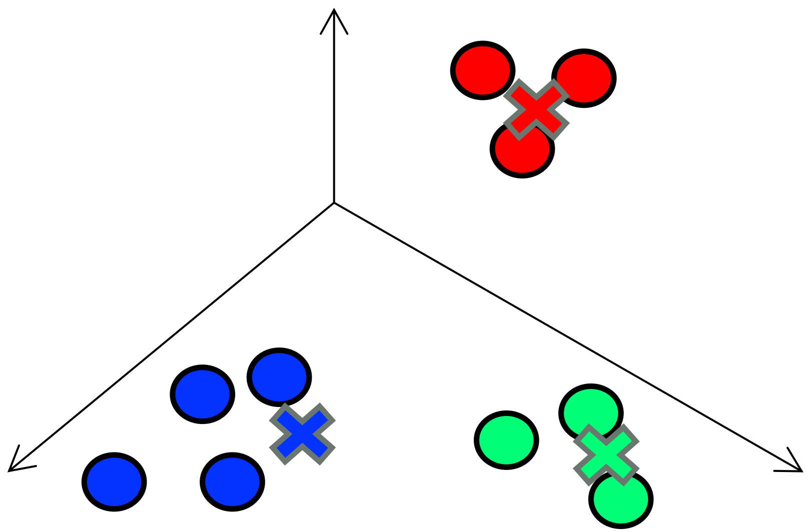


* S.P. Lloyd, "Least squares quantization in PCM", IEEE Transactions on Information Theory, 28-2:129–137, 1982.

K-means Clustering

- In-general minimising the within-cluster sum of squares objective function is computationally complex (NP-hard)
- In practice, heuristic methods can achieve approximate minimisations (local minima) in a computationally efficient manner

- Lloyd's Algorithm* is a common approach to solving k-means clustering:
  - (1) Pick K random points and assign each to one of the K clusters
  - (2) Assign all N points to each cluster based on their distance to the mean of all points in each cluster (to begin with each cluster only has a single point)
  - (3) Recalculate the means of each cluster based on all the points in each
  - (4) Repeat from step (2) until the assignments do not change from the last iteration (convergence)

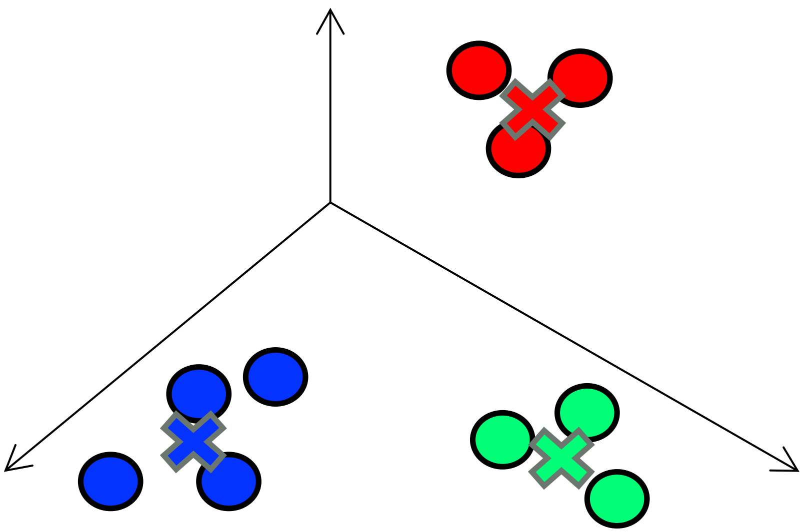


* S.P. Lloyd, "Least squares quantization in PCM", IEEE Transactions on Information Theory, 28-2:129–137, 1982.

K-means Clustering

- In-general minimising the within-cluster sum of squares objective function is computationally complex (NP-hard)
- In practice, heuristic methods can achieve approximate minimisations (local minima) in a computationally efficient manner

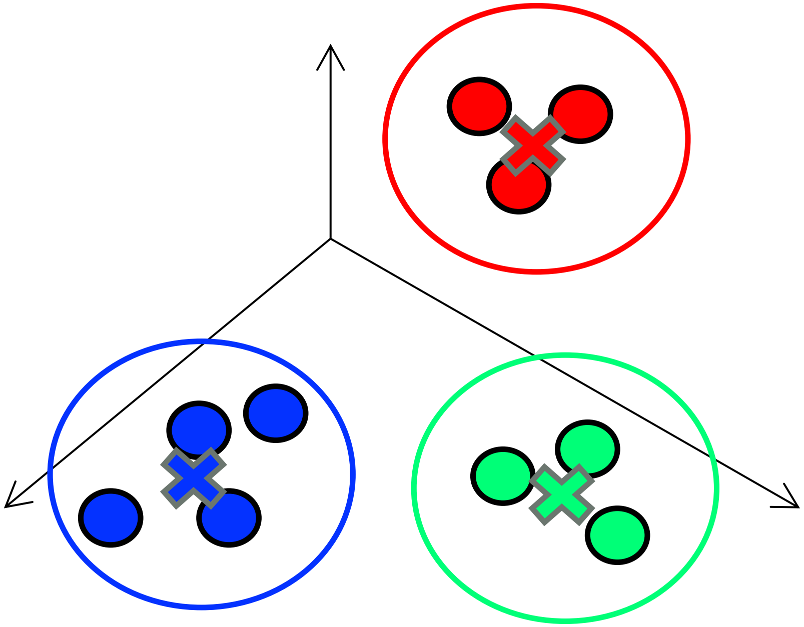
- Lloyd's Algorithm* is a common approach to solving k-means clustering:
  - (1) Pick K random points and assign each to one of the K clusters
  - (2) Assign all N points to each cluster based on their distance to the mean of all points in each cluster (to begin with each cluster only has a single point)
  - (3) Recalculate the means of each cluster based on all the points in each
  - (4) Repeat from step (2) until the assignments do not change from the last iteration (convergence)



* S.P. Lloyd, "Least squares quantization in PCM", IEEE Transactions on Information Theory, 28-2:129–137, 1982.

K-means Clustering

- In-general minimising the within-cluster sum of squares objective function is computationally complex (NP-hard)
- In practice, heuristic methods can achieve approximate minimisations (local minima) in a computationally efficient manner
- Lloyd's Algorithm* is a common approach to solving k-means clustering:
  - (1) Pick K random points and assign each to one of the K clusters
  - (2) Assign all N points to each cluster based on their distance to the mean of all points in each cluster (to begin with each cluster only has a single point)
  - (3) Recalculate the means of each cluster based on all the points in each
  - (4) Repeat from step (2) until the assignments do not change from the last iteration (convergence)



* S.P. Lloyd, "Least squares quantization in PCM", IEEE Transactions on Information Theory, 28-2:129–137, 1982.

K-means Clustering

- In practice, Lloyds algorithm converges relatively quickly to a local minima of the objective function
- Multiple repeated runs (with different random starting points) are often used to surf into better local minima

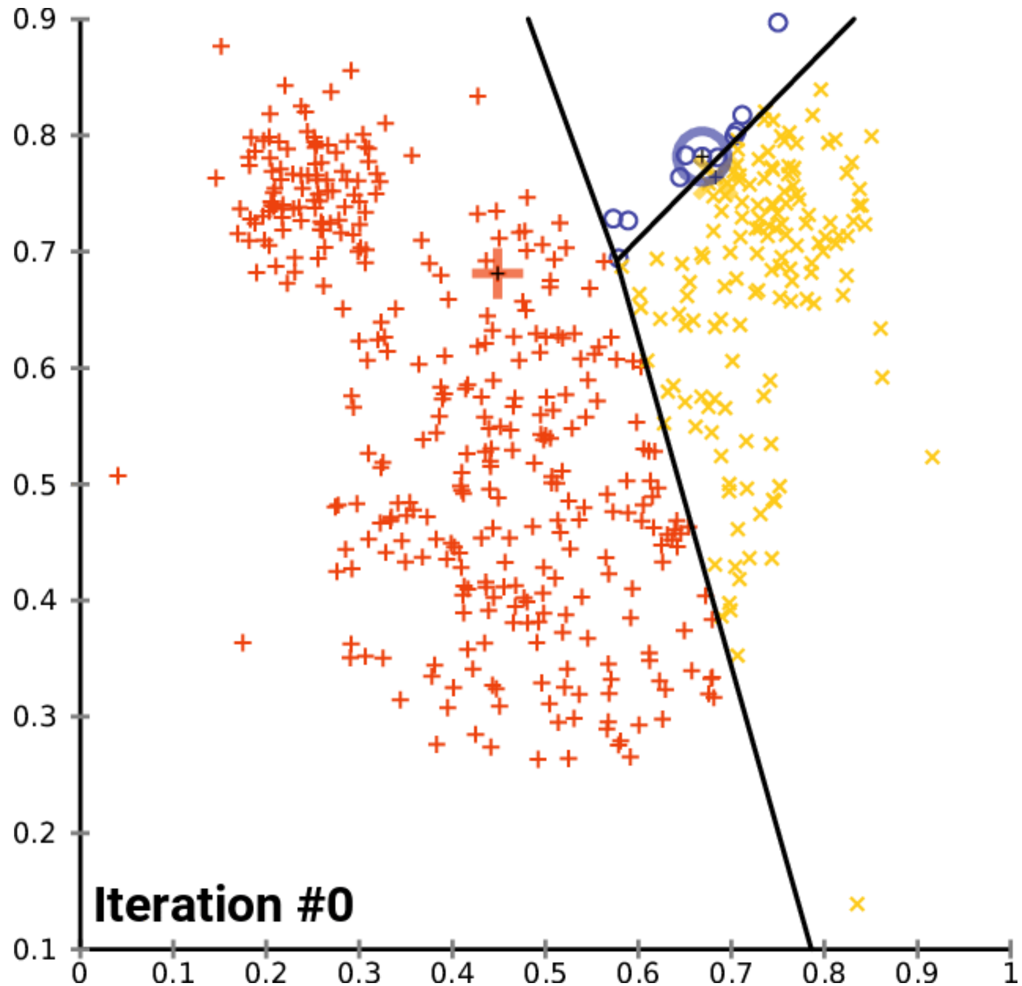


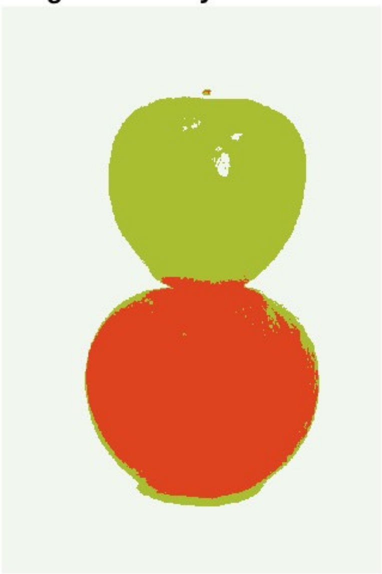
Image Courtesy of Chire (CC-SA-4.0)
https://en.wikipedia.org/wiki/File:K-means_convergence.gif

K-means Clustering

- Clustering algorithms can be applied to various feature spaces beyond colour (i.e. intensity, texture, other filtered outputs)

- Advantages of k-means:
  - Simple and fast

- Disadvantages of k-means:
  - Need to pick the number of clusters K beforehand
  - Can be sensitive to initially selected points
  - Clusters feature space into spheres: not good at finding non-spherical clusters



5 minute break

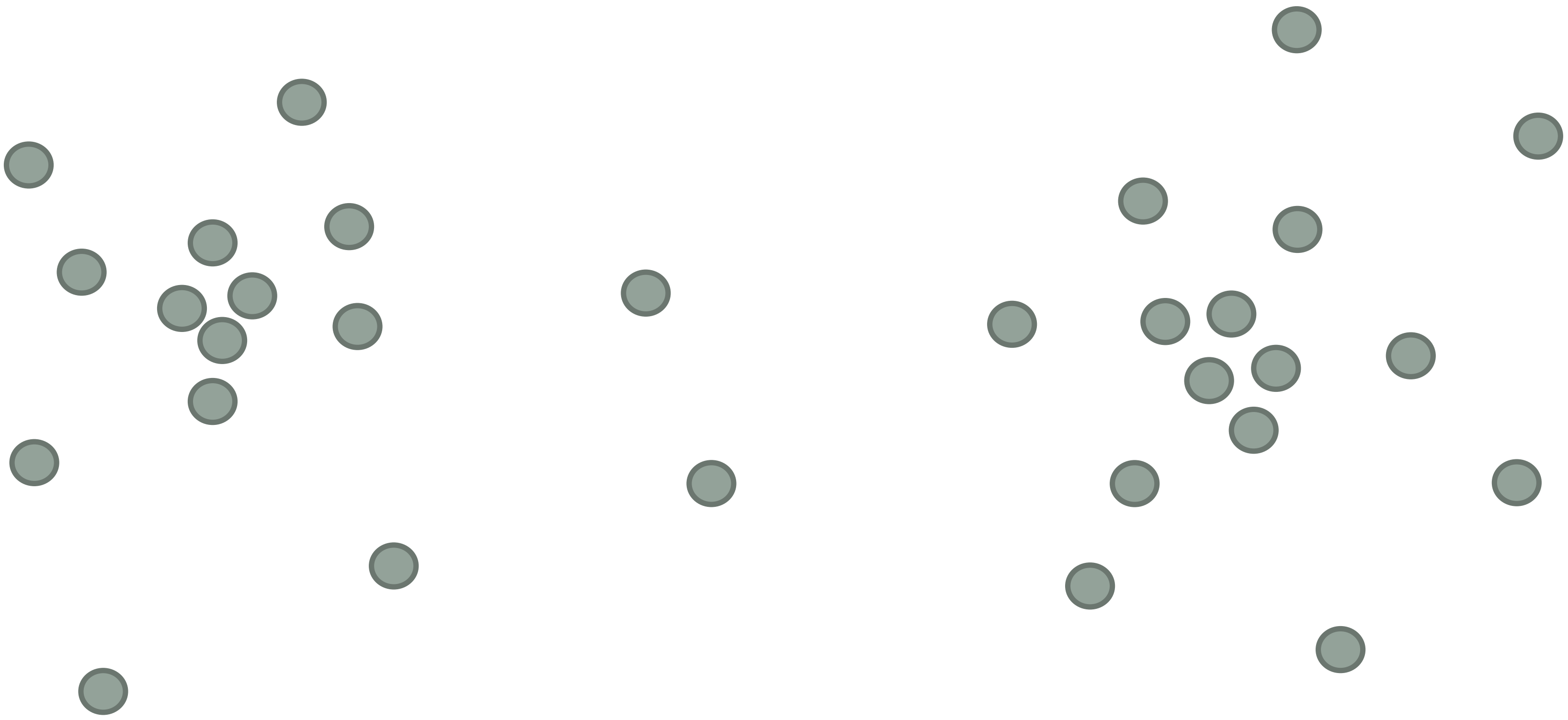
Mean Shift

- **Mean shift** is a clustering technique that can be applied to image segmentation that overcomes two of the main disadvantages of k-means:
  - Does not require the number of clusters to be pre-specified
  - Can find clusters in the feature space that are non-circular in shape

D. Comaniciu, P. Meer, “Mean shift: a robust approach toward feature space analysis”, IEEE Trans. on Pattern Analysis and Machine Intelligence, 2002.

Mean Shift

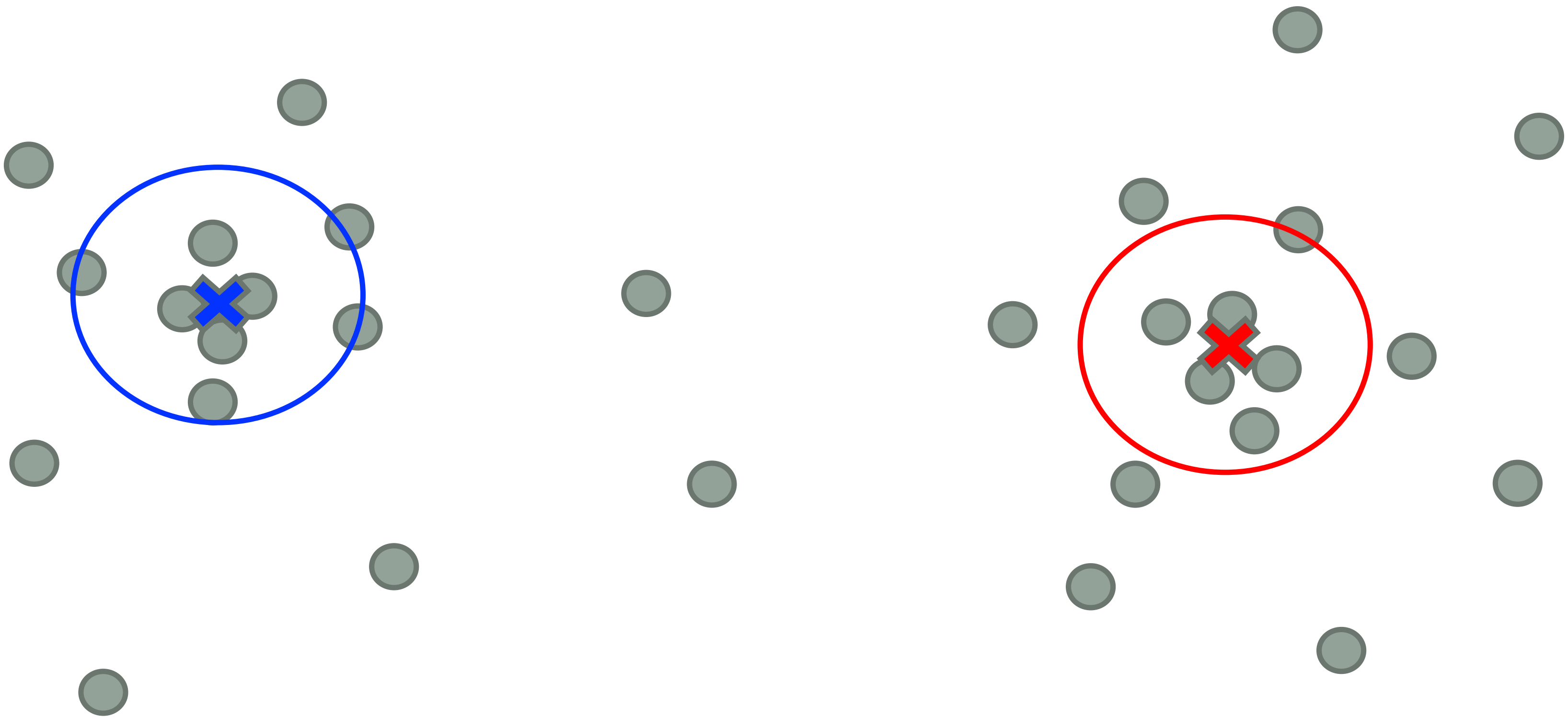
- Mean Shift clustering is a “mode-seeking” algorithm: it finds locally-maximal regions of point density in the feature space



Feature space points

Mean Shift

- Mean Shift clustering is a “mode-seeking” algorithm: it finds locally-maximal regions of point density in the feature space

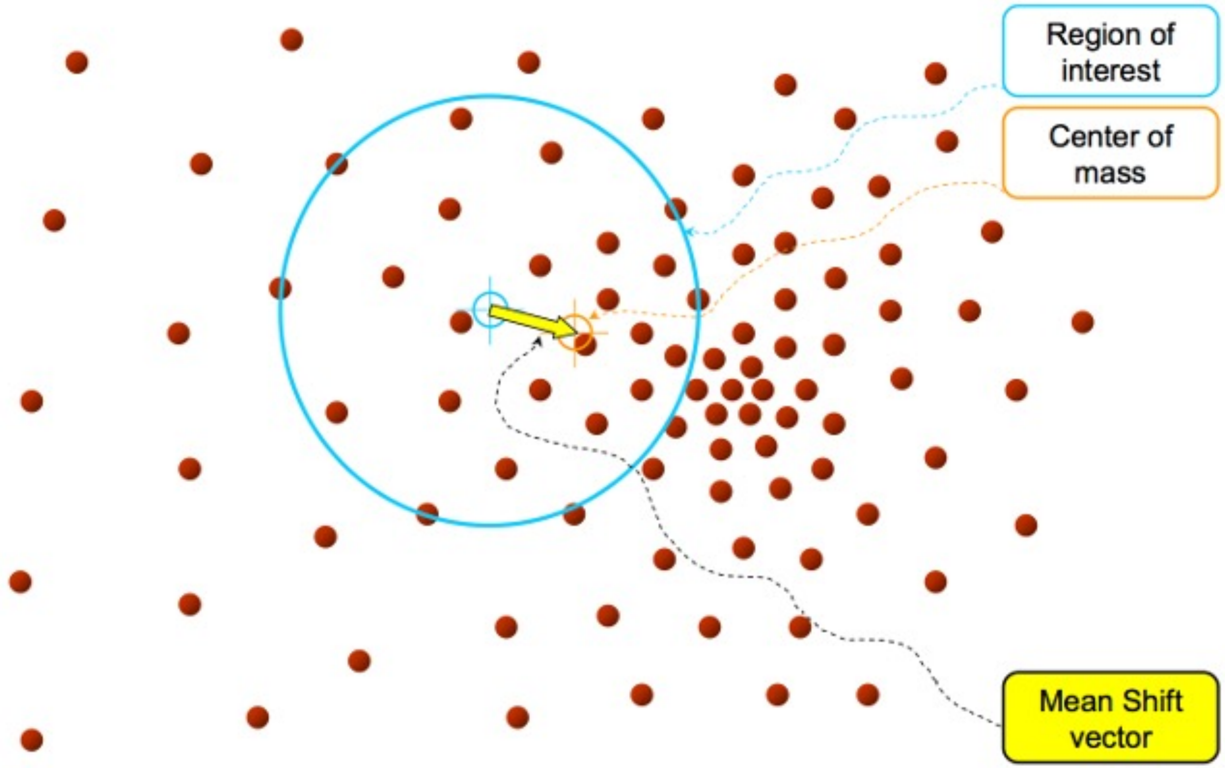


Locally-maximal density

Mean Shift

- Mean Shift clustering is a “mode-seeking” algorithm: it finds locally-maximal regions of point density in the feature space

- Finding density maxima in feature space:
  - (1) From a local search window compute the mean
  - (2) Shift the local window to the mean
  - (3) Repeat this process until convergence

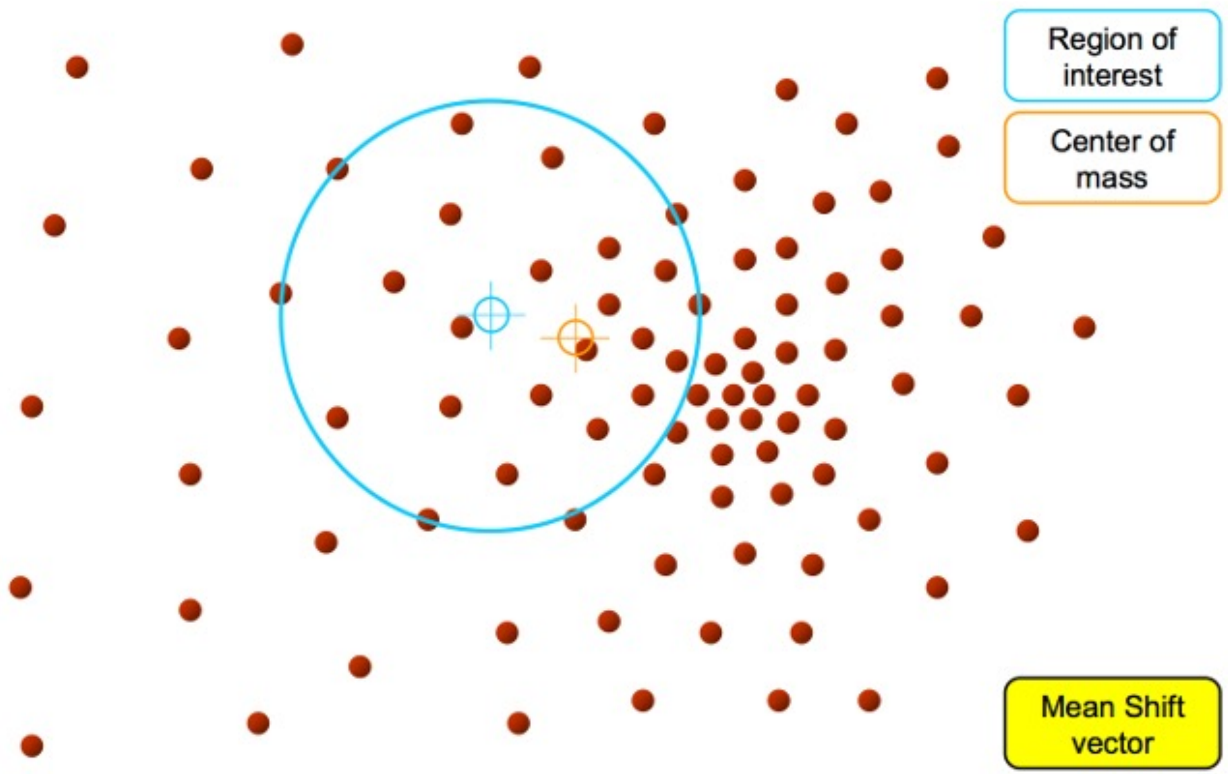


Slide by Y. Ukrainitz & B. Sarel

Mean Shift

- Mean Shift clustering is a “mode-seeking” algorithm: it finds locally-maximal regions of point density in the feature space

- Finding density maxima in feature space:
  - (1) From a local search window compute the mean
  - (2) Shift the local window to the mean
  - (3) Repeat this process until convergence

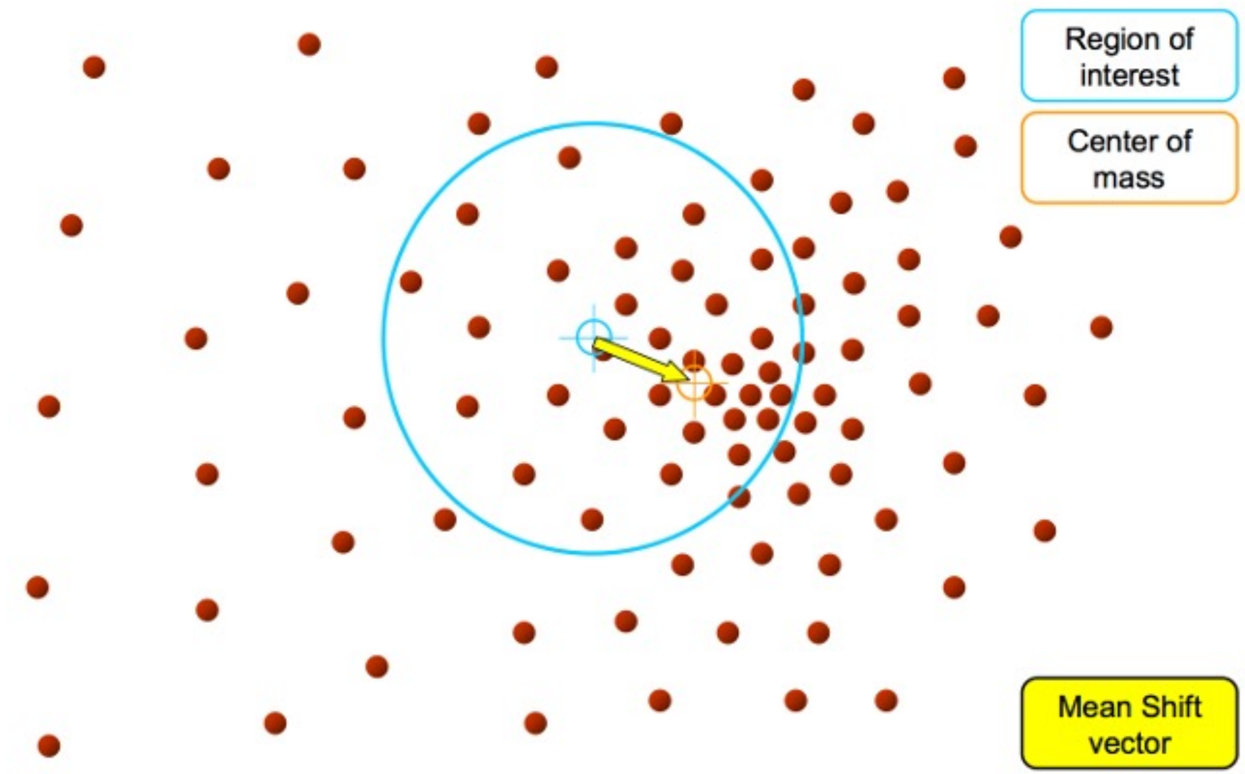


Slide by Y. Ukrainitz & B. Sarel

Mean Shift

- Mean Shift clustering is a “mode-seeking” algorithm: it finds locally-maximal regions of point density in the feature space

- Finding density maxima in feature space:
  - (1) From a local search window compute the mean
  - (2) Shift the local window to the mean
  - (3) Repeat this process until convergence

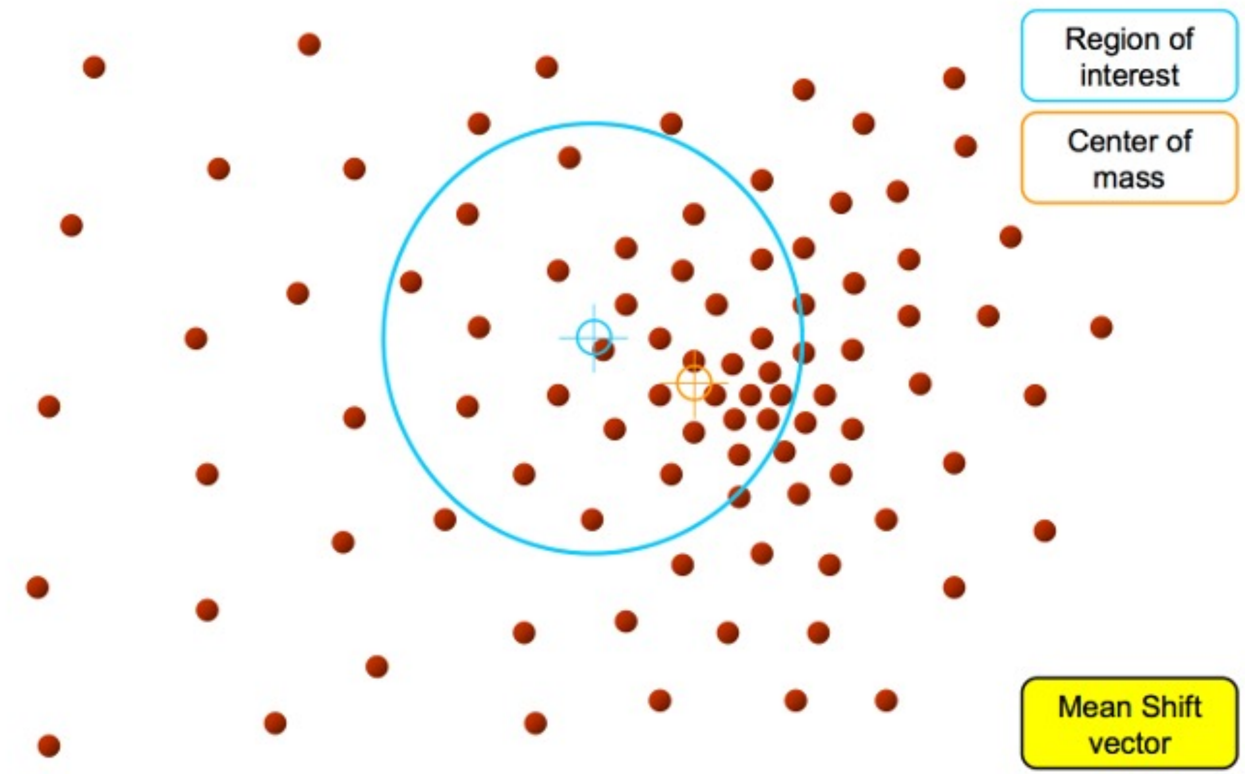


Slide by Y. Ukrainitz & B. Sarel

Mean Shift

- Mean Shift clustering is a “mode-seeking” algorithm: it finds locally-maximal regions of point density in the feature space

- Finding density maxima in feature space:
  - (1) From a local search window compute the mean
  - (2) Shift the local window to the mean
  - (3) Repeat this process until convergence

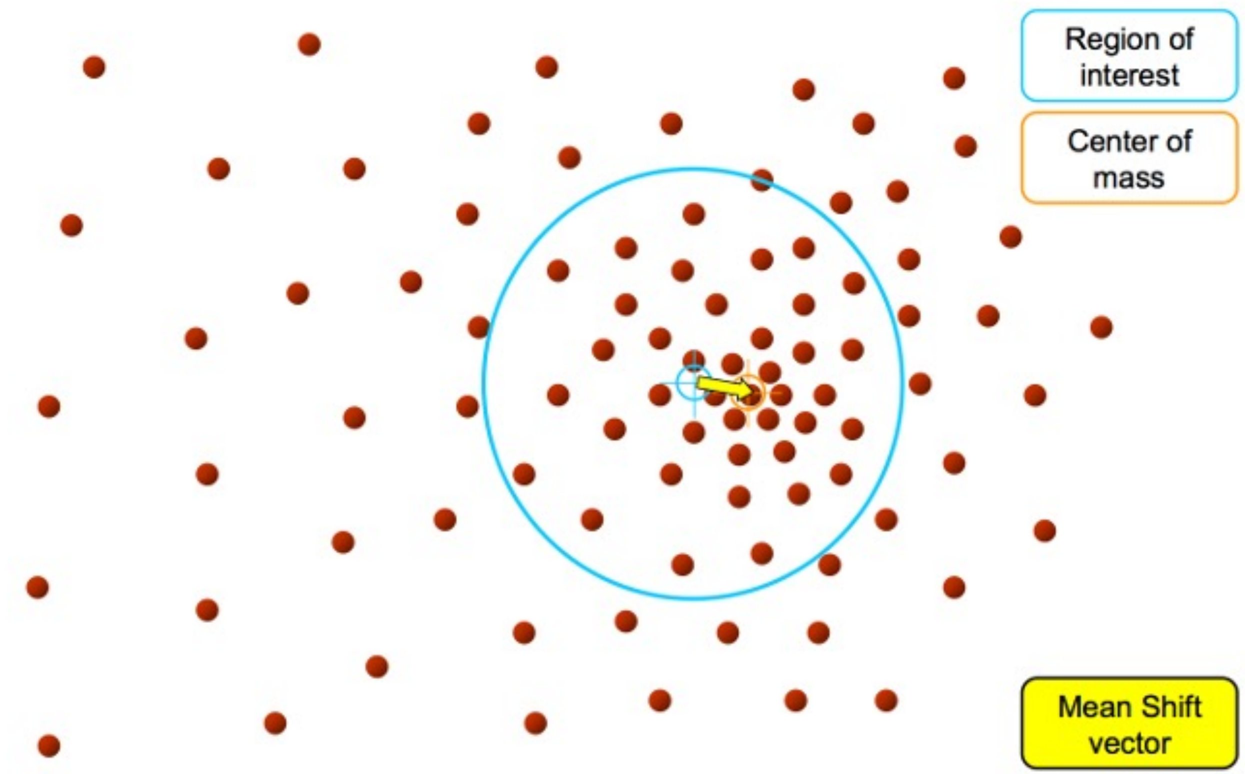


Slide by Y. Ukrainitz & B. Sarel

Mean Shift

- Mean Shift clustering is a “mode-seeking” algorithm: it finds locally-maximal regions of point density in the feature space

- Finding density maxima in feature space:
  - (1) From a local search window compute the mean
  - (2) Shift the local window to the mean
  - (3) Repeat this process until convergence

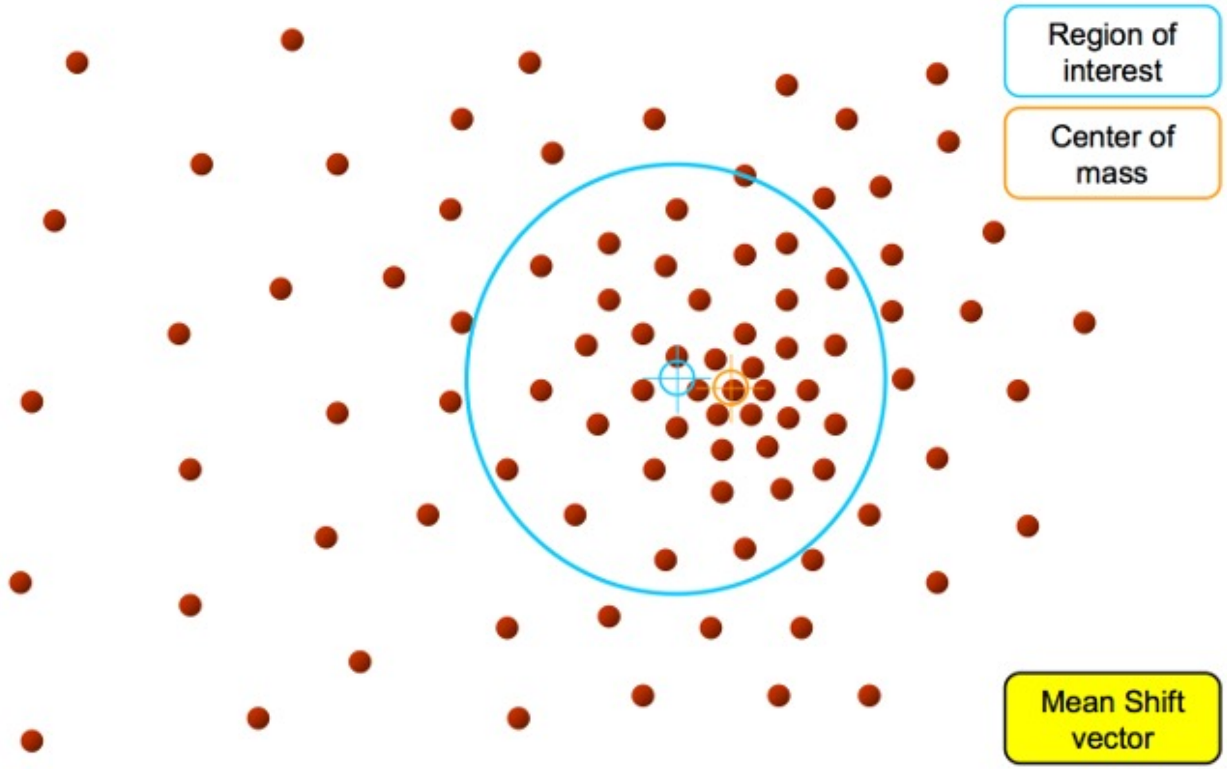


Slide by Y. Ukrainitz & B. Sarel

Mean Shift

- Mean Shift clustering is a “mode-seeking” algorithm: it finds locally-maximal regions of point density in the feature space

- Finding density maxima in feature space:
  - (1) From a local search window compute the mean
  - (2) Shift the local window to the mean
  - (3) Repeat this process until convergence

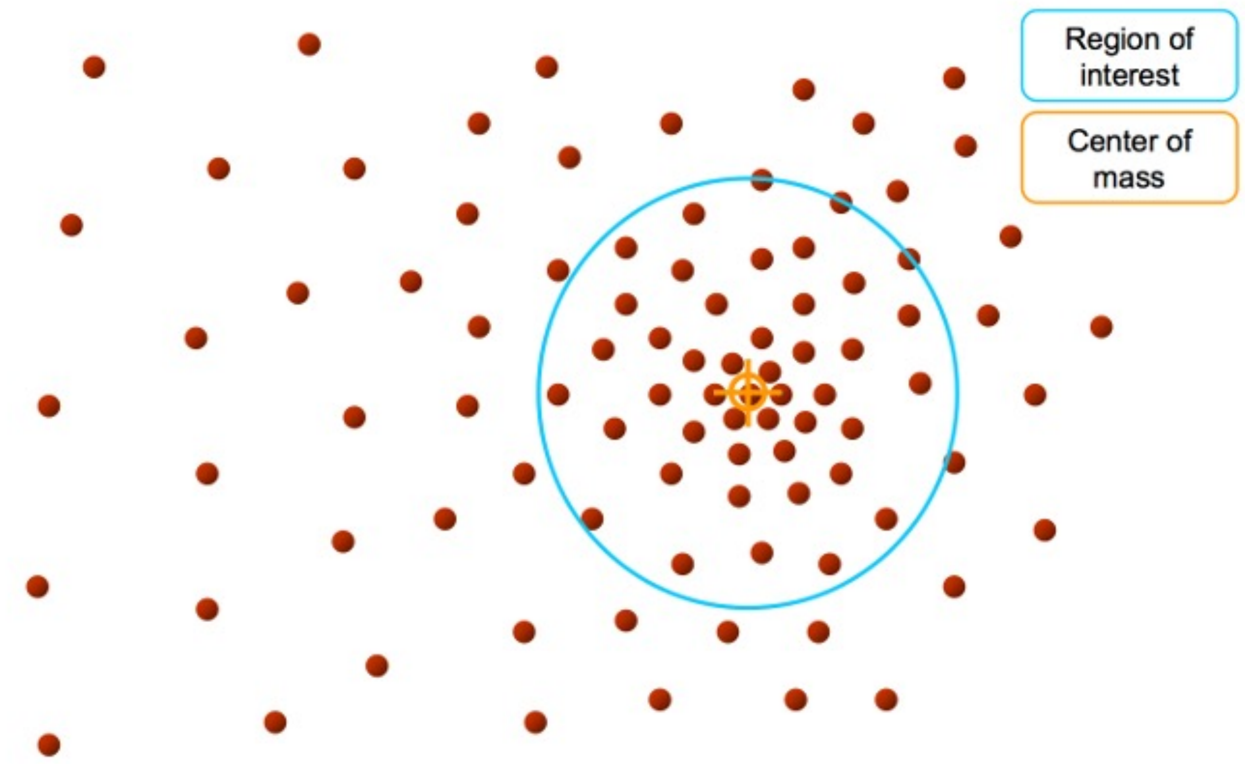


Slide by Y. Ukrainitz & B. Sarel

Mean Shift

- Mean Shift clustering is a “mode-seeking” algorithm: it finds locally-maximal regions of point density in the feature space

- Finding density maxima in feature space:
  - (1) From a local search window compute the mean
  - (2) Shift the local window to the mean
  - (3) Repeat this process until convergence

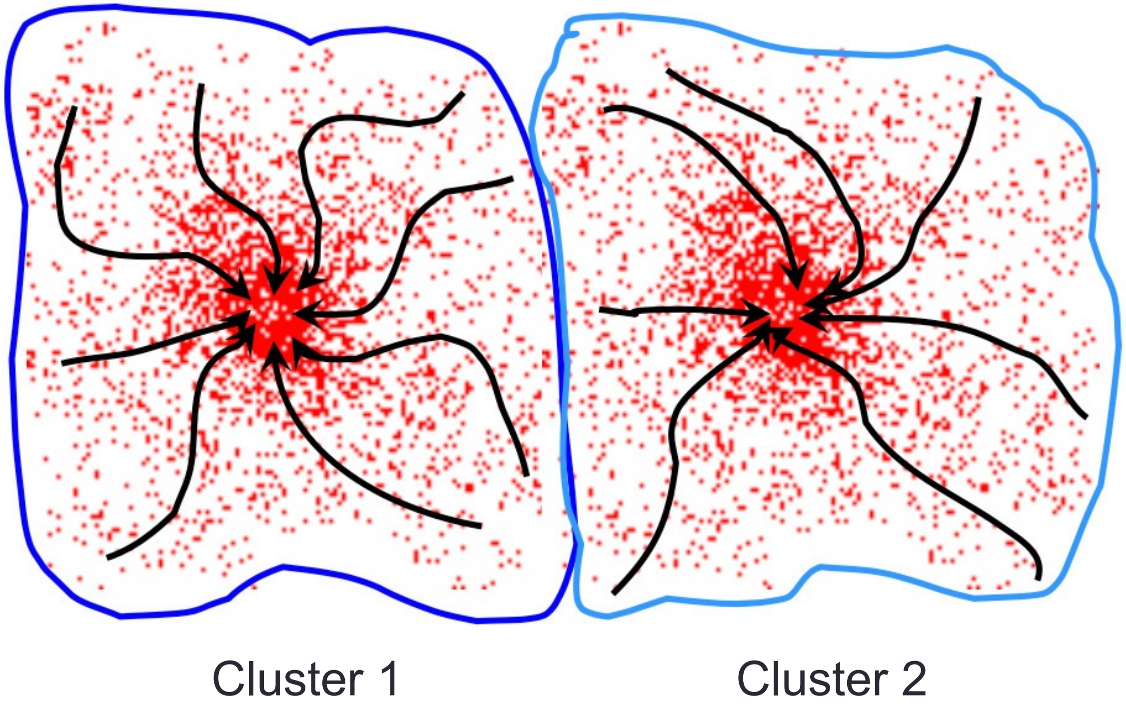


Slide by Y. Ukrainitz & B. Sarel

Mean Shift

- Algorithm begins by finding modes in feature space, initialising based on where pixel values lie in the feature space

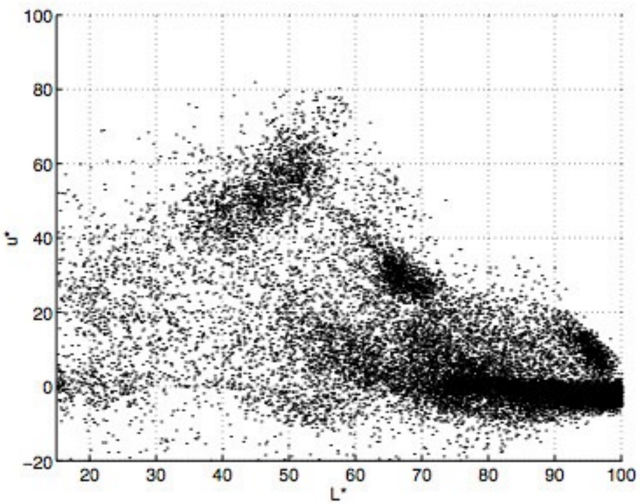
- Each location in feature space is then assigned to a mode (cluster) based on where they fall along a basin of attraction

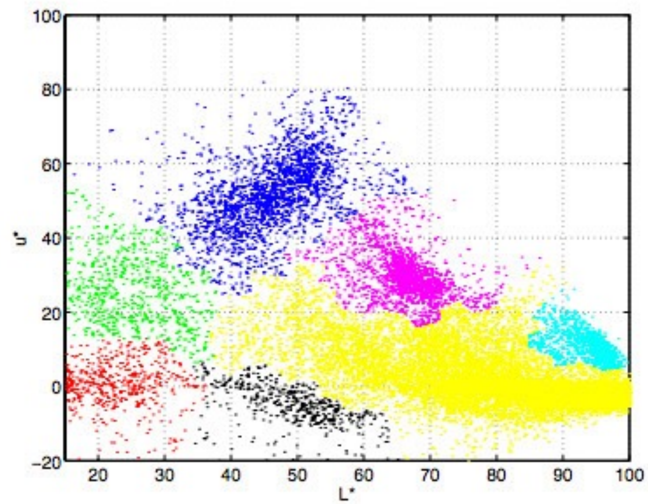


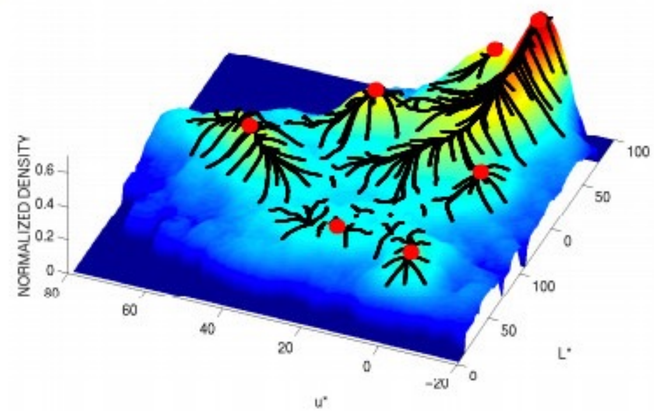
Mean Shift

- Algorithm begins by finding modes in feature space, initialising based on where pixel values lie in the feature space

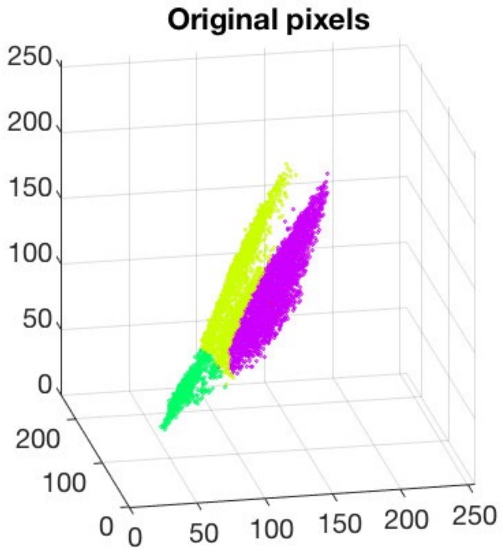
- Each location in feature space is then assigned to a mode (cluster) based on where they fall along a basin of attraction

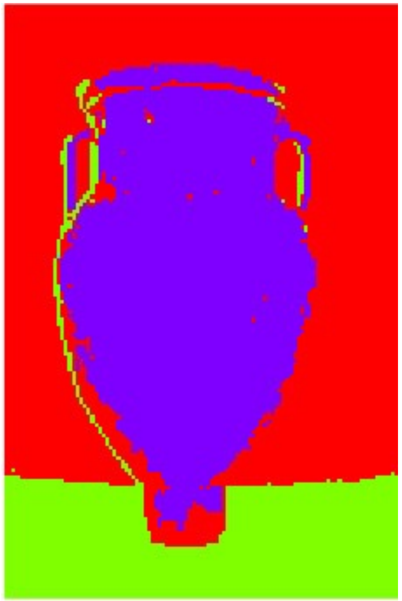


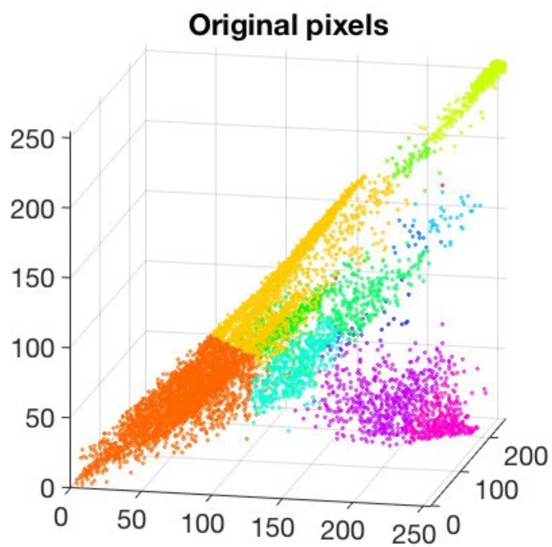




Mean Shift

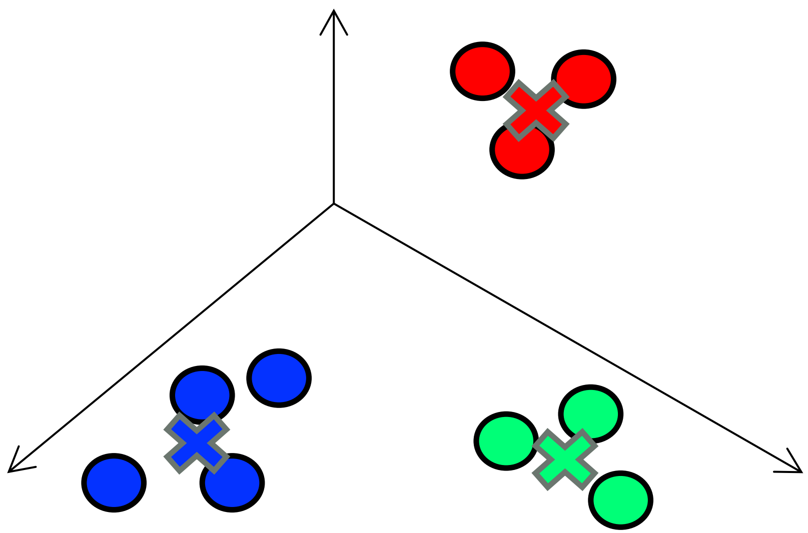

Original pixels
250
200
150
100
50
0
200
100
0
0
50
100
150
200
250

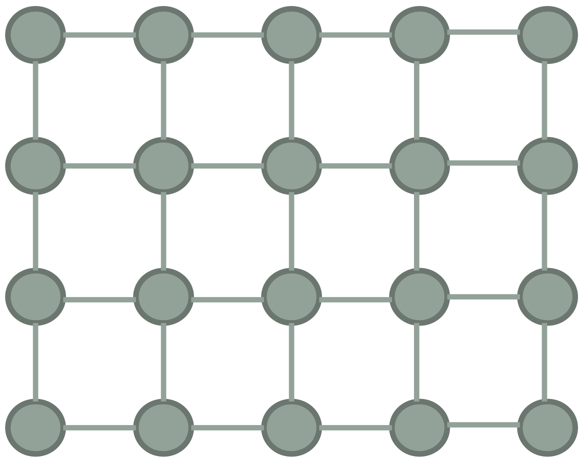



Original pixels
250
200
150
100
50
0
0
50
100
150
200
250
200
100
0

Maintaining spatial discontinuities in segments

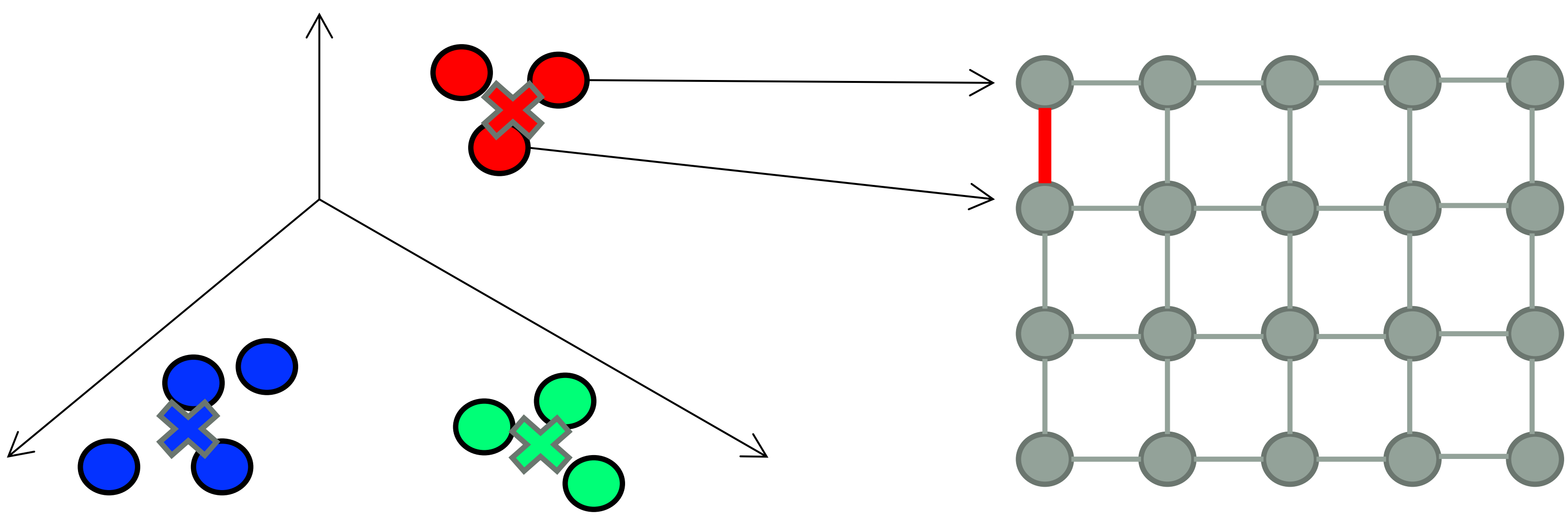
- Segmentations made based on a feature space can leave disparate spatial regions in the image belonging to the one segment
- One way to separate disparate regions is by treating the image as a graph, where edge values are the distance in mode values in feature space, corresponding to the basin each pixel sits in





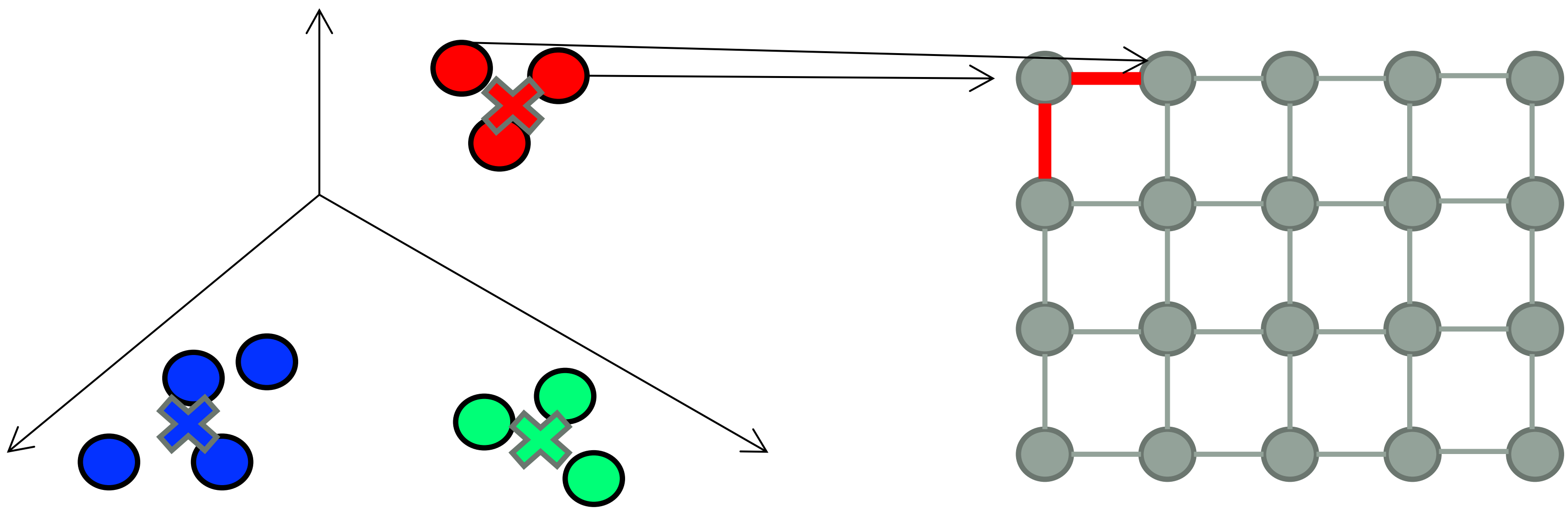
Maintaining spatial discontinuities in segments

- Segmentations made based on a feature space can leave disparate spatial regions in the image belonging to the one segment
- One way to separate disparate regions is by treating the image as a graph, where edge values are the distance in mode values in feature space, corresponding to the basin each pixel sits in



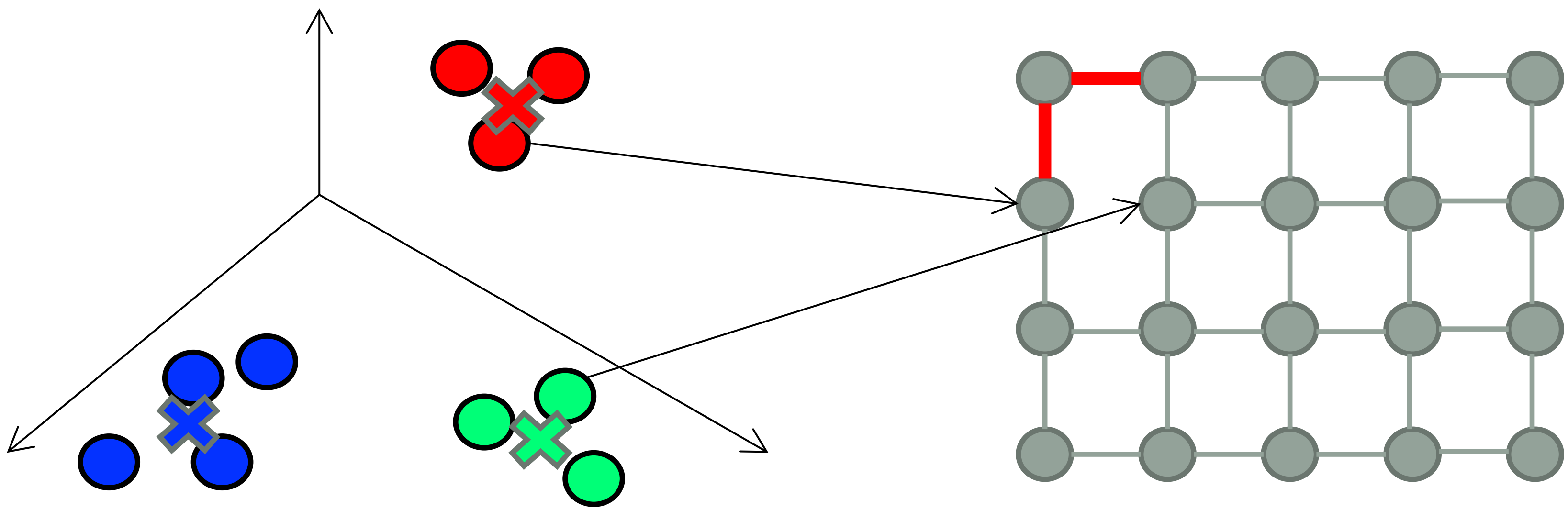
Maintaining spatial discontinuities in segments

- Segmentations made based on a feature space can leave disparate spatial regions in the image belonging to the one segment
- One way to separate disparate regions is by treating the image as a graph, where edge values are the distance in mode values in feature space, corresponding to the basin each pixel sits in



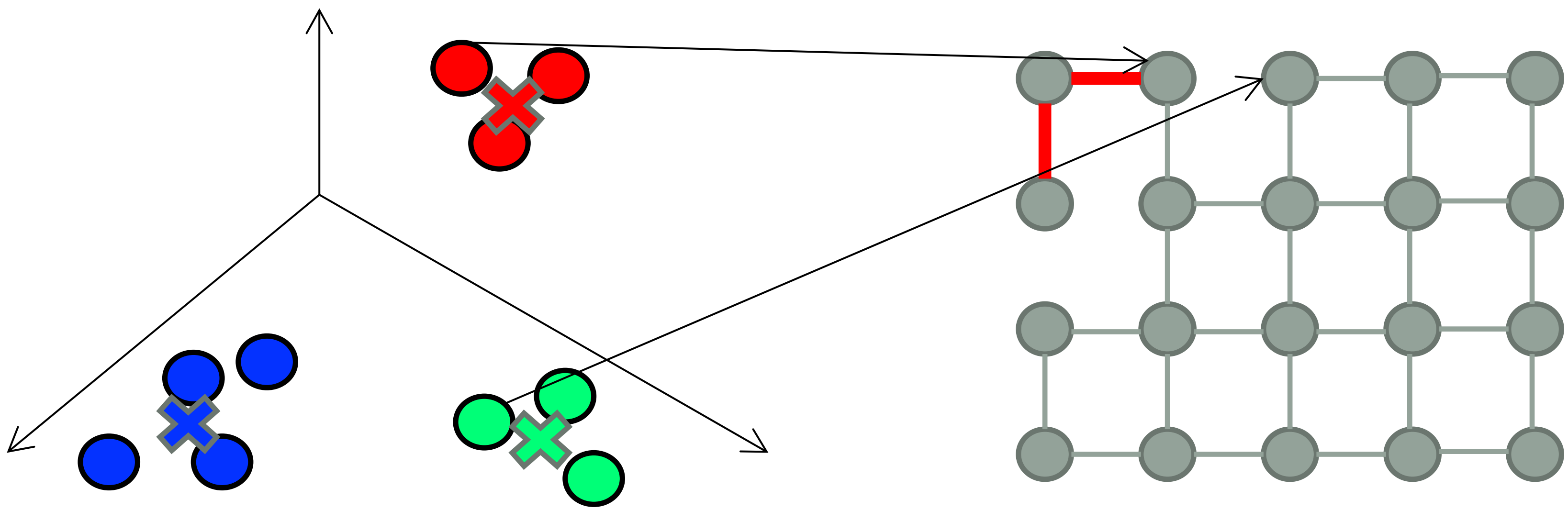
Maintaining spatial discontinuities in segments

- Segmentations made based on a feature space can leave disparate spatial regions in the image belonging to the one segment
- One way to separate disparate regions is by treating the image as a graph, where edge values are the distance in mode values in feature space, corresponding to the basin each pixel sits in



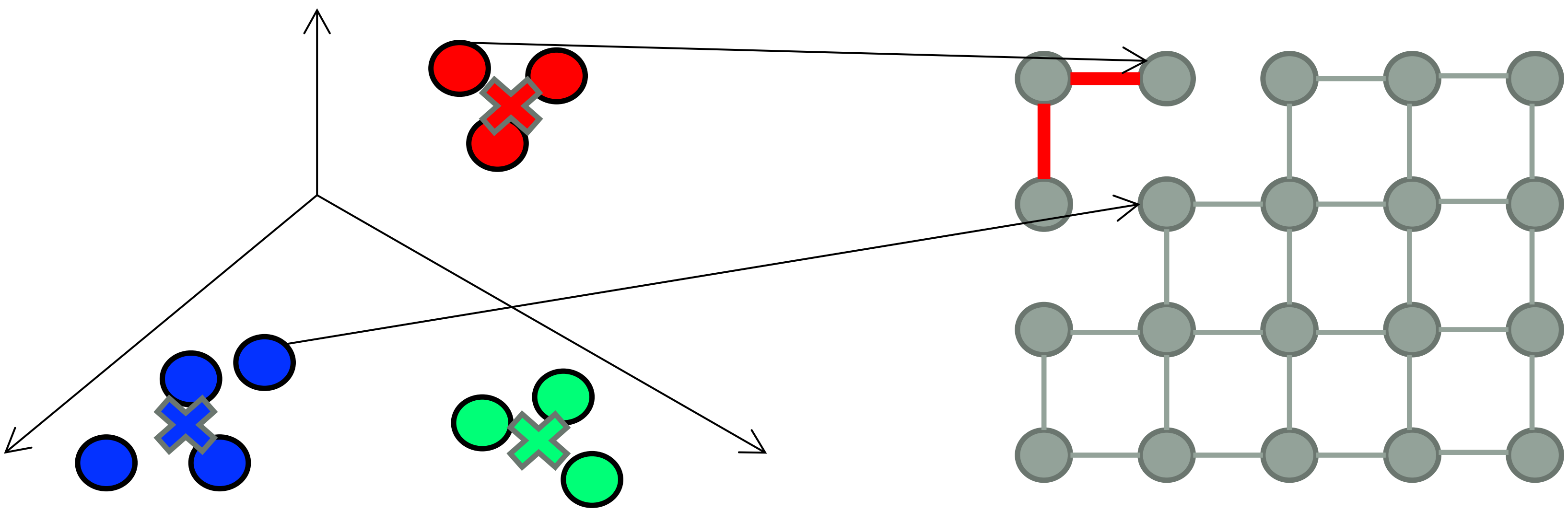
Maintaining spatial discontinuities in segments

- Segmentations made based on a feature space can leave disparate spatial regions in the image belonging to the one segment
- One way to separate disparate regions is by treating the image as a graph, where edge values are the distance in mode values in feature space, corresponding to the basin each pixel sits in



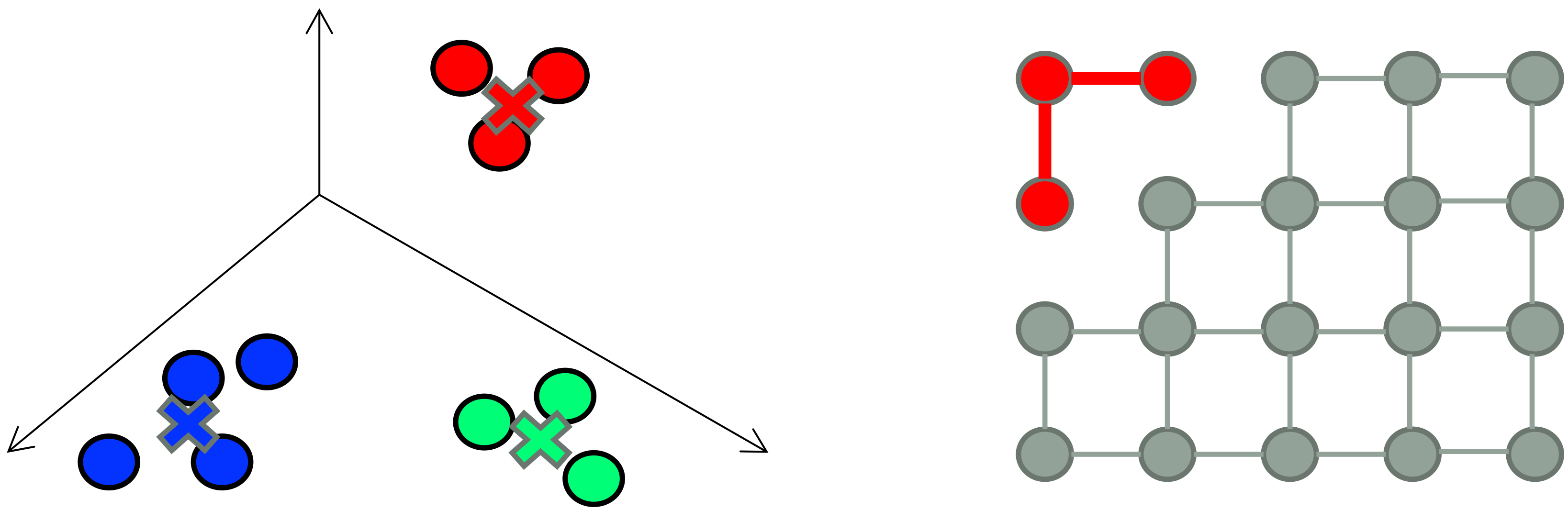
Maintaining spatial discontinuities in segments

- Segmentations made based on a feature space can leave disparate spatial regions in the image belonging to the one segment
- One way to separate disparate regions is by treating the image as a graph, where edge values are the distance in mode values in feature space, corresponding to the basin each pixel sits in

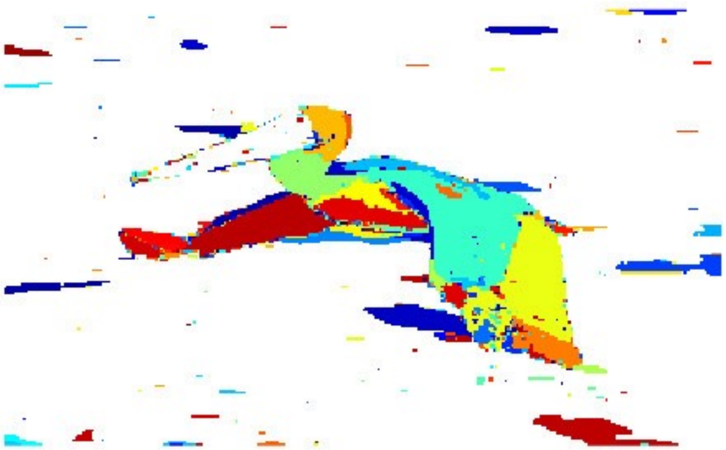


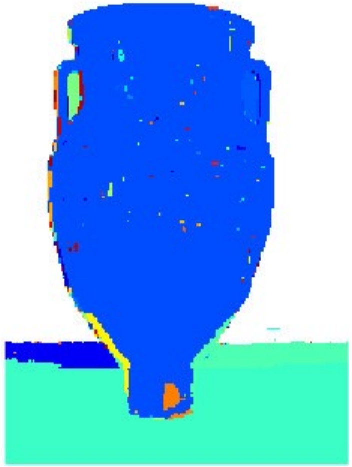
Maintaining spatial discontinuities in segments

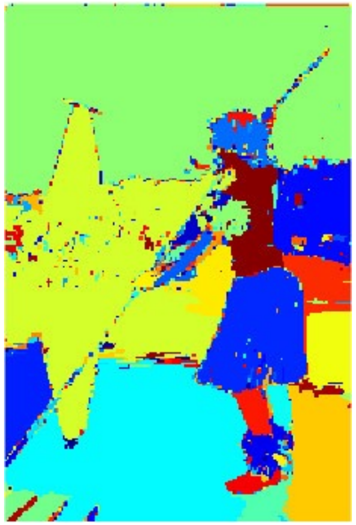
- Segmentations made based on a feature space can leave disparate spatial regions in the image belonging to the one segment
- One way to separate disparate regions is by treating the image as a graph, where edge values are the distance in mode values in feature space, corresponding to the basin each pixel sits in



Maintaining spatial discontinuities in segments



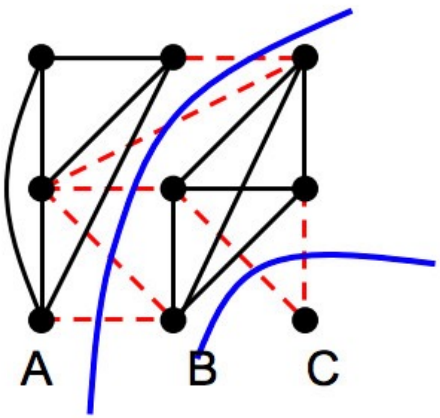


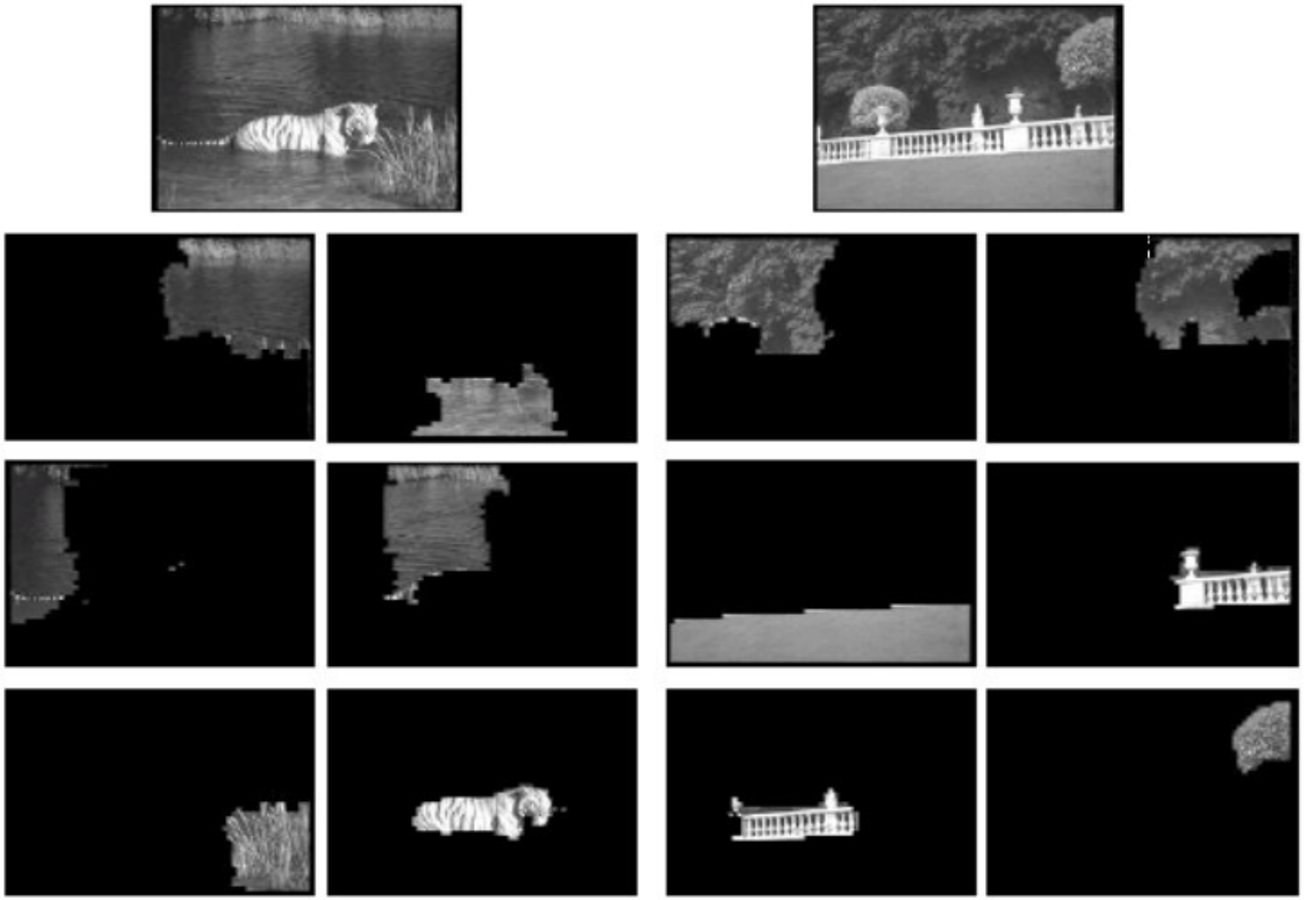


Graph-based Segmentation

- More sophisticated graph-based segmentations optimise sets of adjoining edges to remove based on minimal energy loss, associated with similarity

Cost of cut = sum of similarity weights in removed edges





J. Shi and J. Malik, “Normalized cuts and image segmentation”, IEEE Trans. PAMI, 2000

Further Reading and Next Week

- References:

  - Cohen, L., Cohen, I., 1993. Finite element methods for active contour models and balloons for 2D and 3D images. IEEE Transactions on Pattern Analysis and Machine Intelligence 15 (11), 1131–1147.
  - D. Comaniciu, P. Meer, “Mean shift: a robust approach toward feature space analysis”, IEEE Trans. on Pattern Analysis and Machine Intelligence, 2002.
  - D. A. Forsyth and J. Ponce, “Computer Vision - A Modern Approach”, Prentice Hall, 2002
  - R. Szeliski, “Computer Vision: Algorithms and Applications”, Springer, 2010

- Next Week:
  - Introduction to Machine Learning and Image Classification