Image Segmentation

Source: https://www.commsp.ee.ic.ac.uk/~tania/

Introduction to image segmentation

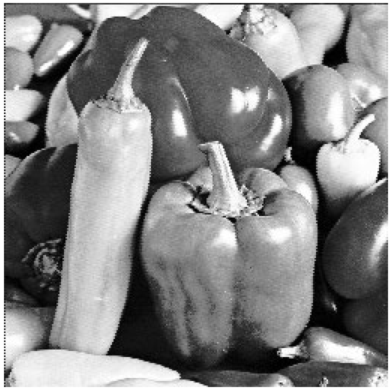
- The purpose of image segmentation is to partition an image into *meaningful* regions with respect to a particular application
- The segmentation is based on measurements taken from the image and might be *greylevel*, *colour*, *texture*, *depth* or *motion*

Introduction to image segmentation

- Usually image segmentation is an initial and vital step in a series of processes aimed at overall image understanding
- Applications of image segmentation include
  - Identifying objects in a scene for object-based measurements such as size and shape
  - Identifying objects in a moving scene for *object-based video compression (MPEG4)*
  - Identifying objects which are at different distances from a sensor using depth measurements from a laser range finder enabling path planning for a mobile robots

Introduction to image segmentation

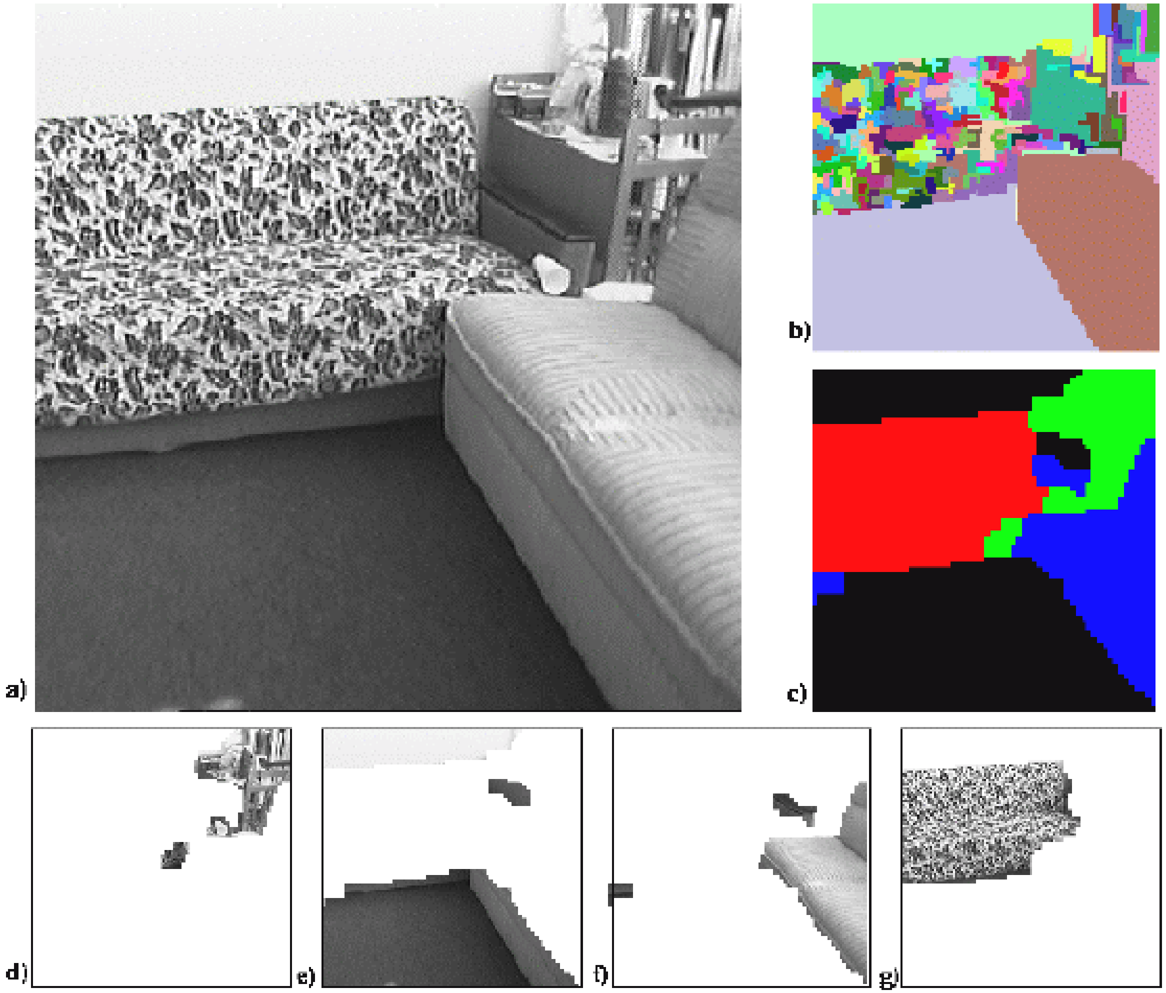
- Example 1
  - Segmentation based on greyscale
  - Very simple ‘model’ of greyscale leads to inaccuracies in object labelling



Introduction to image segmentation

- Example 2
  - Segmentation based on texture
  - Enables object surfaces with varying patterns of grey to be segmented

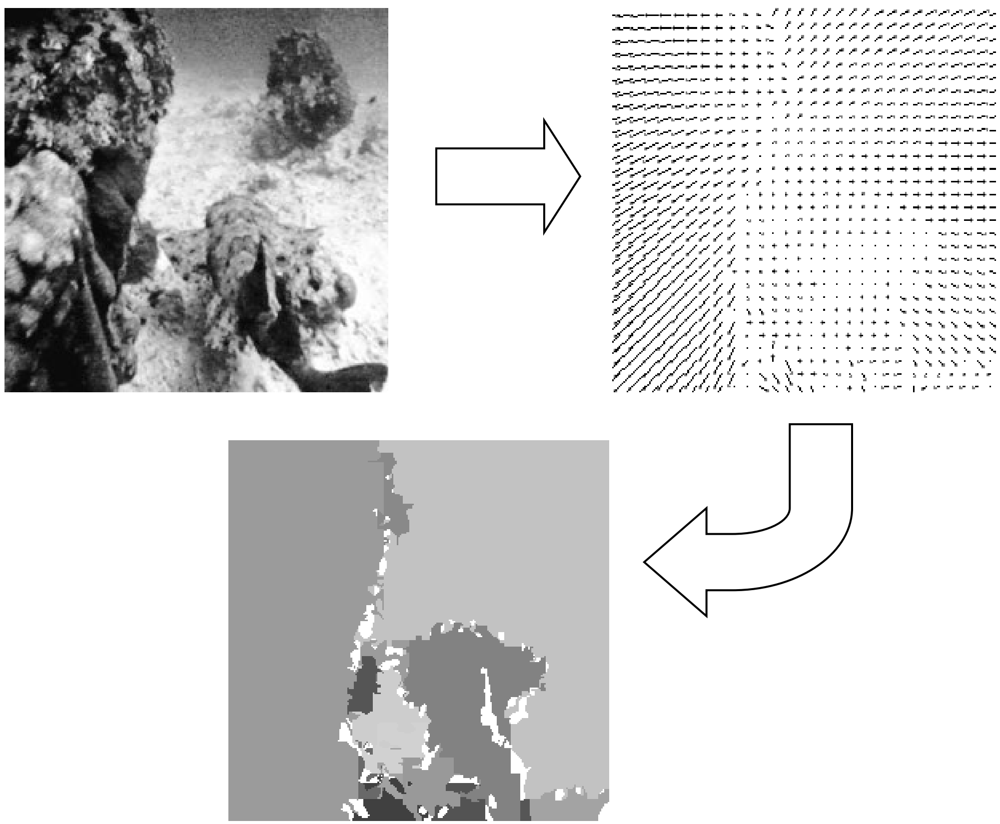
Introduction to image segmentation



Introduction to image segmentation

- Example 3
  - Segmentation based on motion
  - The main difficulty of motion segmentation is that an intermediate step is required to (either implicitly or explicitly) estimate an *optical flow field*
  - The segmentation must be based on this estimate and not, in general, the true flow

Introduction to image segmentation

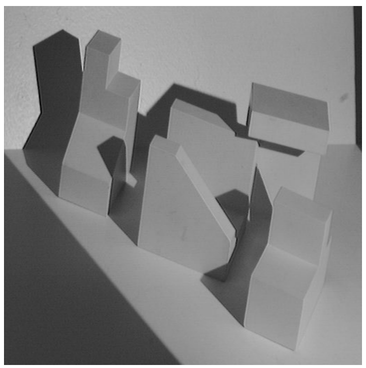


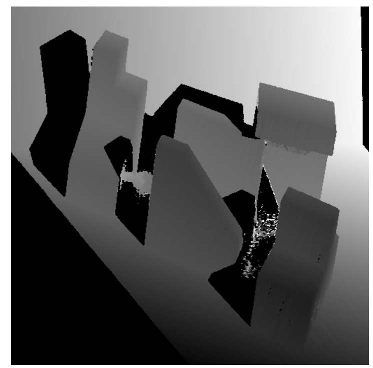
Introduction to image segmentation

- Example 3
  - Segmentation based on depth
  - This example shows a range image, obtained with a laser range finder
  - A segmentation based on the range (the object distance from the sensor) is useful in guiding mobile robots

Introduction to image segmentation

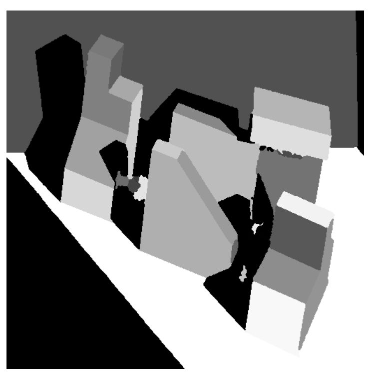
Original image





Range image

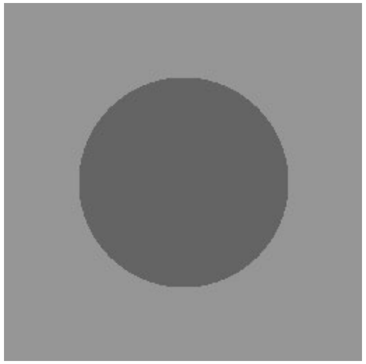
Segmented image



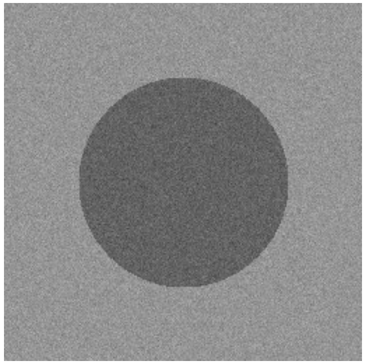
Greylevel histogram-based segmentation

- We will look at two very simple image segmentation techniques that are based on the greylevel histogram of an image
  - Thresholding
  - Clustering
- We will use a very simple object-background test image
  - We will consider a zero, low and high noise image

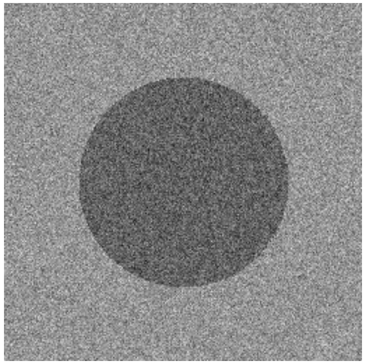
Greylevel histogram-based segmentation



Noise free



Low noise

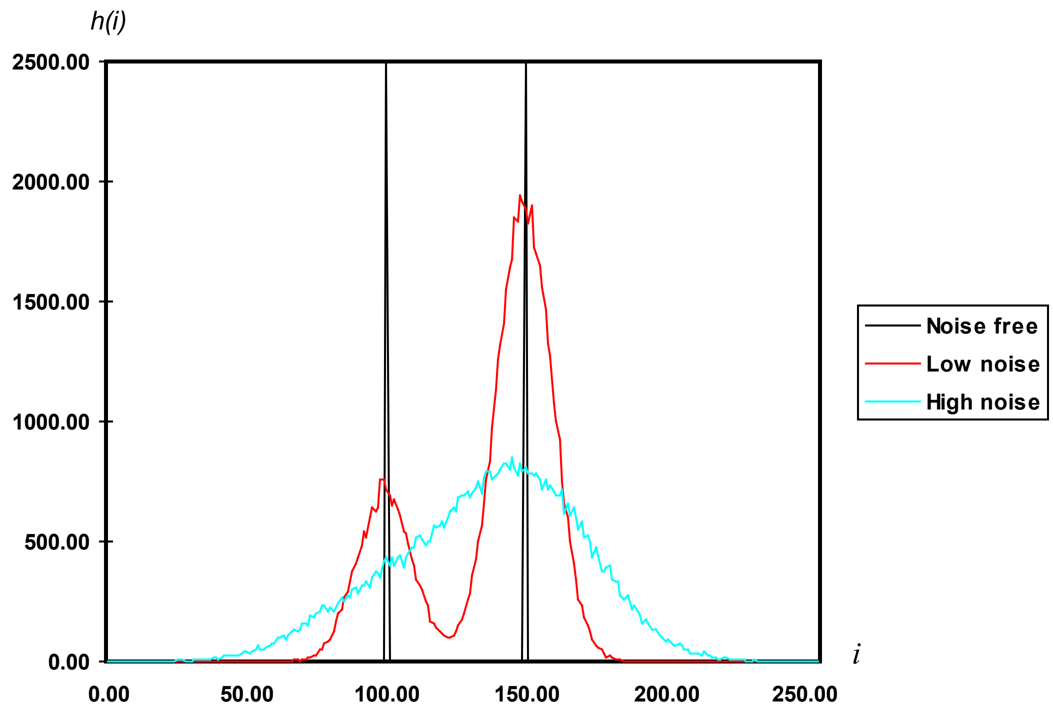


High noise

Greylevel histogram-based segmentation

- How do we characterise low noise and high noise?
- We can consider the histograms of our images
  - For the noise free image, its simply two spikes at i=100, $i=150$
  - For the low noise image, there are two clear peaks centred on i=100, $i=150$
  - For the high noise image, there is a single peak – two greylevel populations corresponding to object and background have merged

Greylevel histogram-based segmentation



Greylevel histogram-based segmentation

- We can define the input image *signal-to-noise ratio* in terms of the mean greylevel value of the object pixels and background pixels and the additive noise standard deviation

$$S/N = \frac{|\mu|}{\sigma}$$

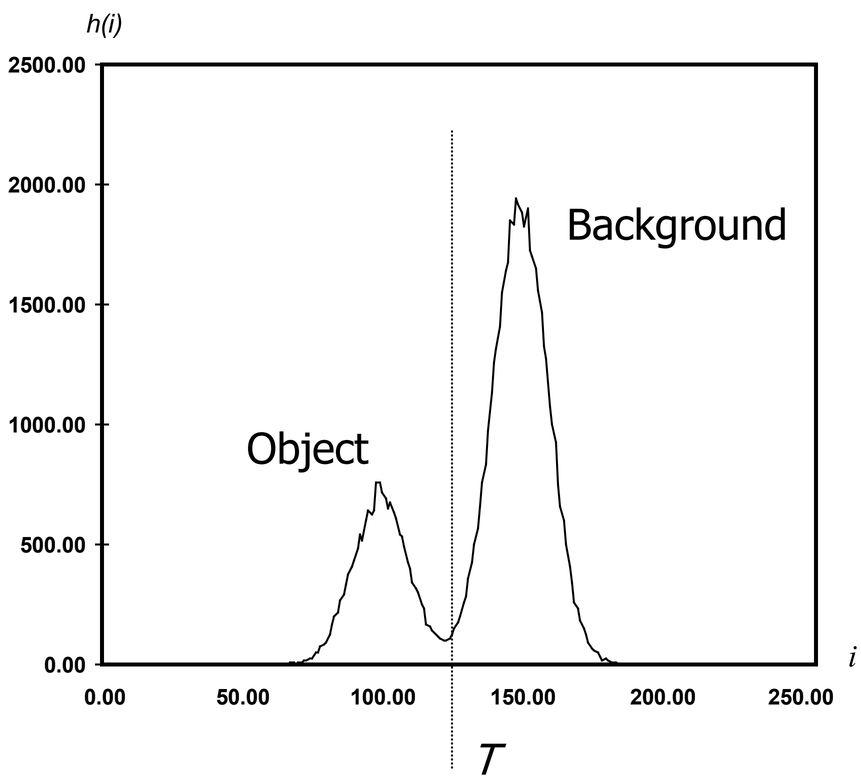
Greylevel histogram-based segmentation

- For our test images :
  - S/N (noise free) = ∞
  - S/N (low noise) = 5
  - S/N (low noise) = 2

Greylevel thresholding

- We can easily understand segmentation based on thresholding by looking at the histogram of the low noise object/background image
  – There is a clear ‘valley’ between to two peaks

Greylevel thresholding



Greylevel thresholding

- We can define the greylevel thresholding algorithm as follows:
  - If the greylevel of pixel p <=T then pixel p is an object pixel

  else

  - Pixel p is a background pixel

Greylevel thresholding

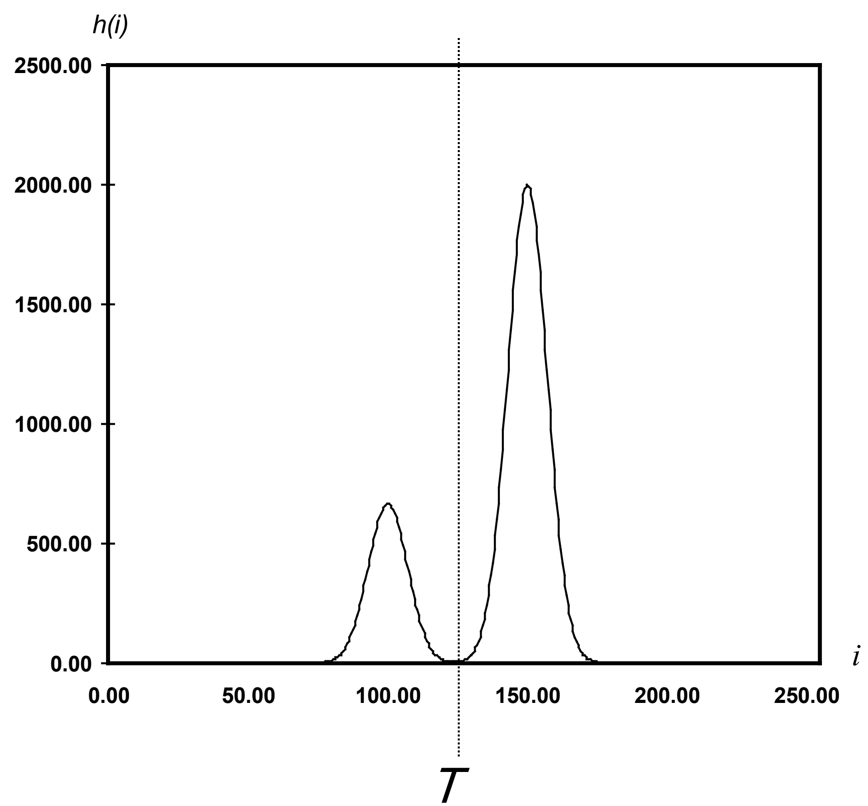
- This simple threshold test begs the obvious question how do we determine the threshold ?
- Many approaches possible
  - Interactive threshold
  - Adaptive threshold
  - Minimisation method

Greylevel thresholding

- We will consider in detail a minimisation method for determining the threshold
  - Minimisation of the *within group variance*
  - Robot Vision, Haralick & Shapiro, volume 1, page 20

Greylevel thresholding

- Idealized object/background image histogram



Greylevel thresholding

- Any threshold separates the histogram into 2 groups with each group having its own statistics (mean, variance)
- The homogeneity of each group is measured by the *within group variance*
- The optimum threshold is that threshold which minimizes the within group variance thus maximizing the homogeneity of each group

Greylevel thresholding

- Let group *o* (object) be those pixels with greylevel <=*T*
- Let group *b* (background) be those pixels with greylevel >*T*
- The prior probability of group *o* is $p_o(T)$
- The prior probability of group *b* is $p_b(T)$

Greylevel thresholding

- The following expressions can easily be derived for prior probabilities of object and background

$$p_o(T) = \sum_{i=0}^{T} P(i)$$

$$p_b(T) = \sum_{i=T+1}^{255} P(i)$$

$$P(i) = h(i) / N$$

- where $h(i)$ is the histogram of an *N* pixel image

Greylevel thresholding

- The mean and variance of each group are as follows :

$$\mu_o(T) = \sum_{i=0}^{T} iP(i) / p_o(T)$$

$$\mu_b(T) = \sum_{i=T+1}^{255} iP(i) / p_b(T)$$

$$\sigma_o^2(T) = \sum_{i=0}^{T} \left[i - \mu_o(T)\right]^2 P(i) / p_o(T)$$

$$\sigma_b^2(T) = \sum_{i=T+1}^{255} \left[i - \mu_b(T)\right]^2 P(i) / p_b(T)$$

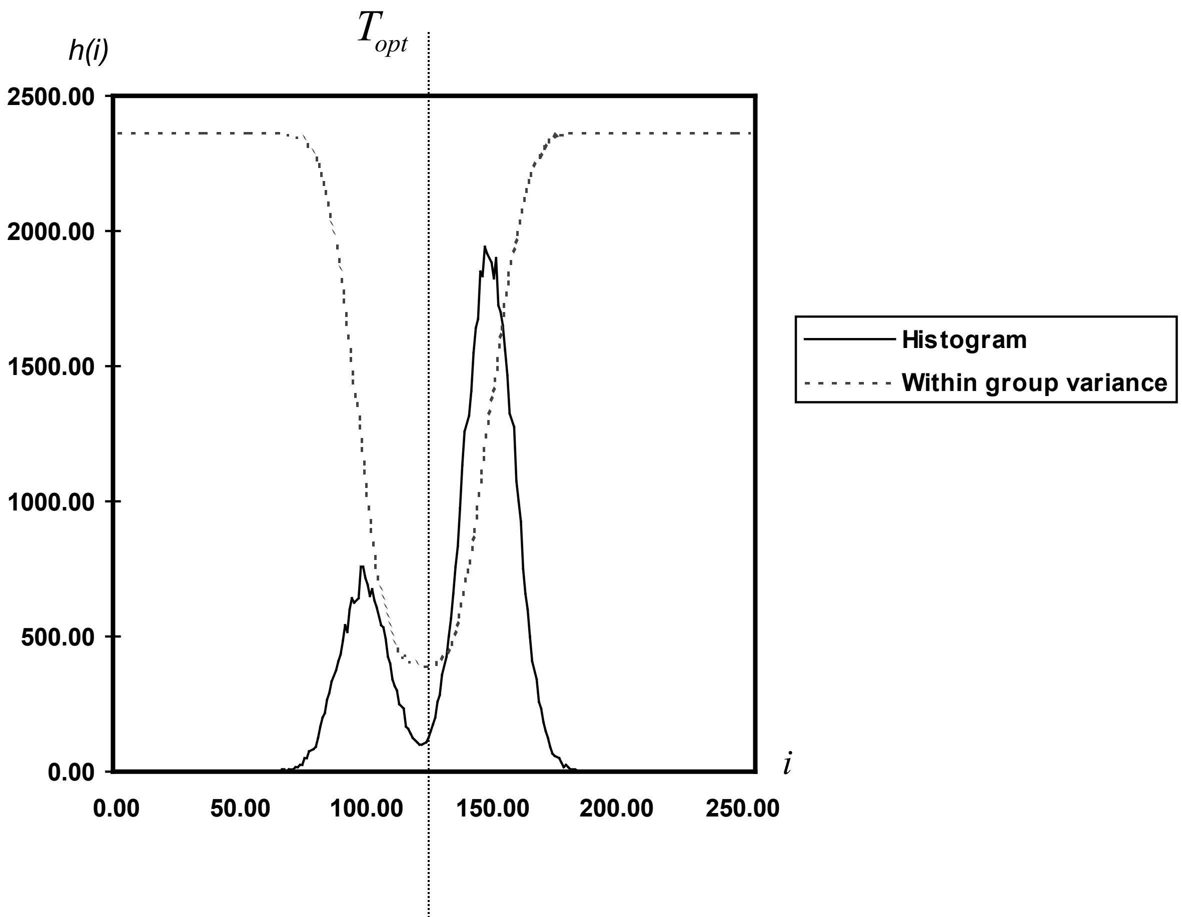
Greylevel thresholding

- The within group variance is defined as :

$$\sigma_W^2(T) = \sigma_o^2(T)p_o(T) + \sigma_b^2(T)p_b(T)$$

- We determine the optimum T by minimizing this expression with respect to T
  - Only requires 256 comparisons for and 8-bit greylevel image

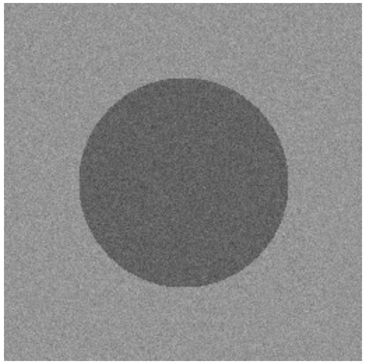
Greylevel thresholding



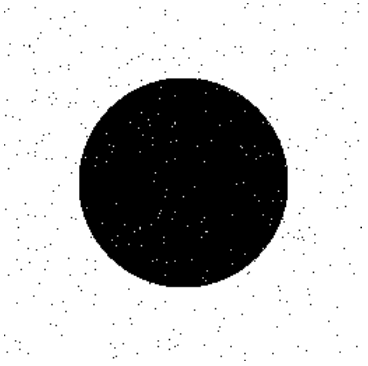
Greylevel thresholding

- We can examine the performance of this algorithm on our low and high noise image
  - For the low noise case, it gives an optimum threshold of T=124
  - Almost exactly halfway between the object and background peaks
  - We can apply this optimum threshold to both the low and high noise images

Greylevel thresholding

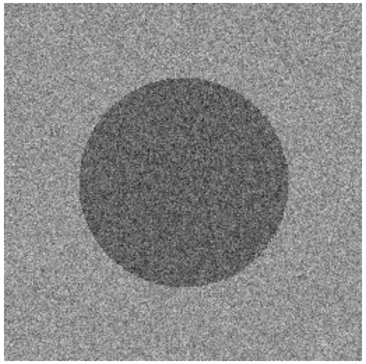


Low noise image

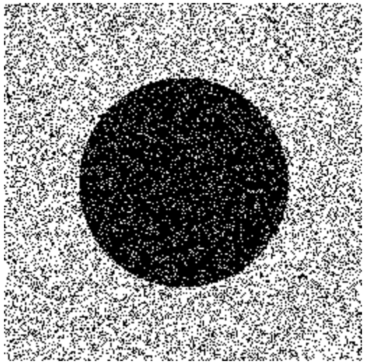


Thresholded at T=124

Greylevel thresholding



Low noise image

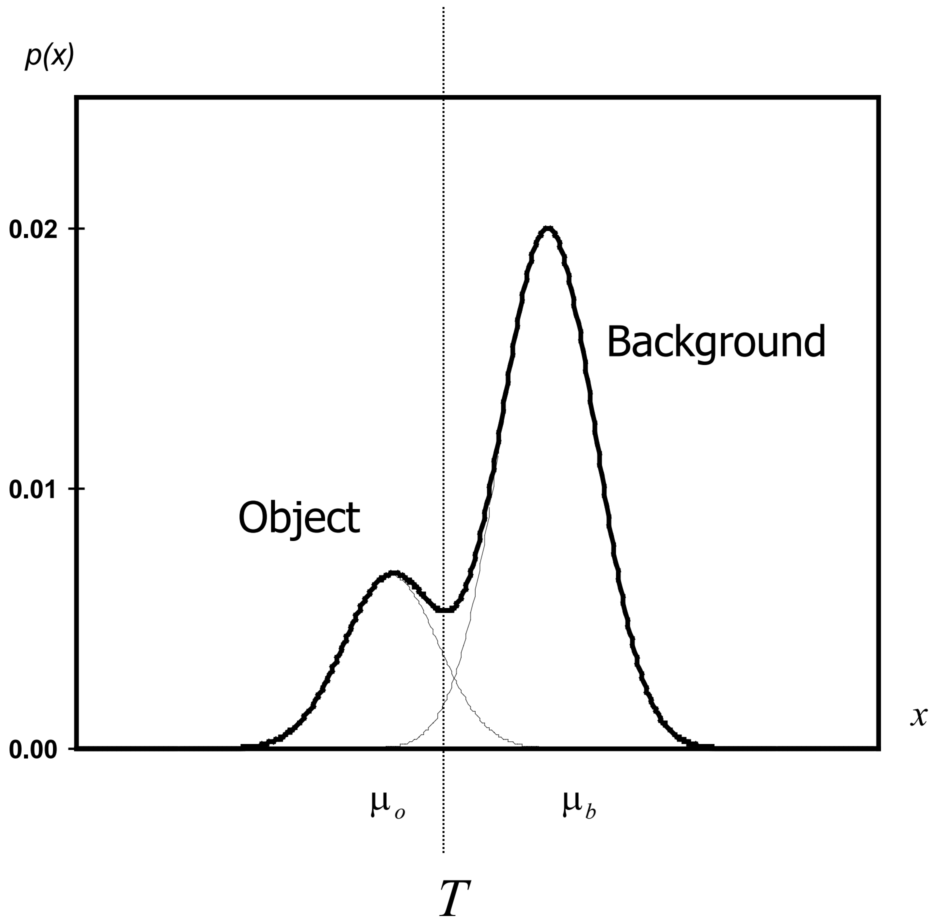


Thresholded at T=124

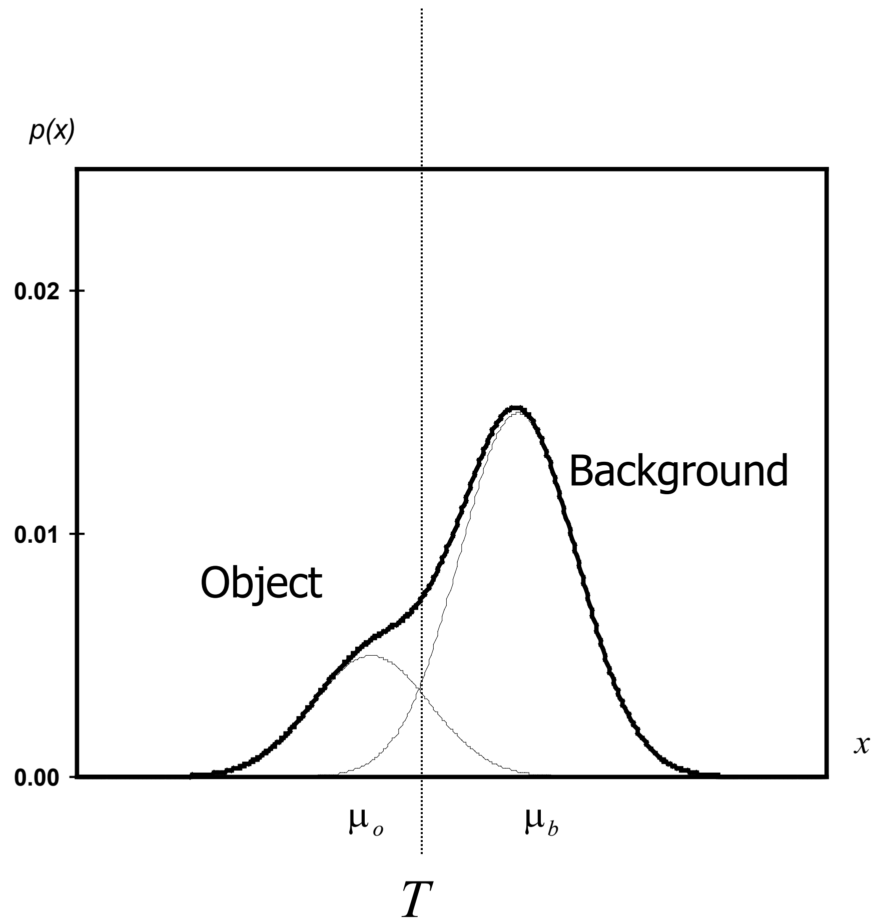
Greylevel thresholding

- High level of pixel miss-classification noticeable
- This is typical performance for thresholding
  - The extent of pixel miss-classification is determined by the overlap between object and background histograms.

Greylevel thresholding



Greylevel thresholding

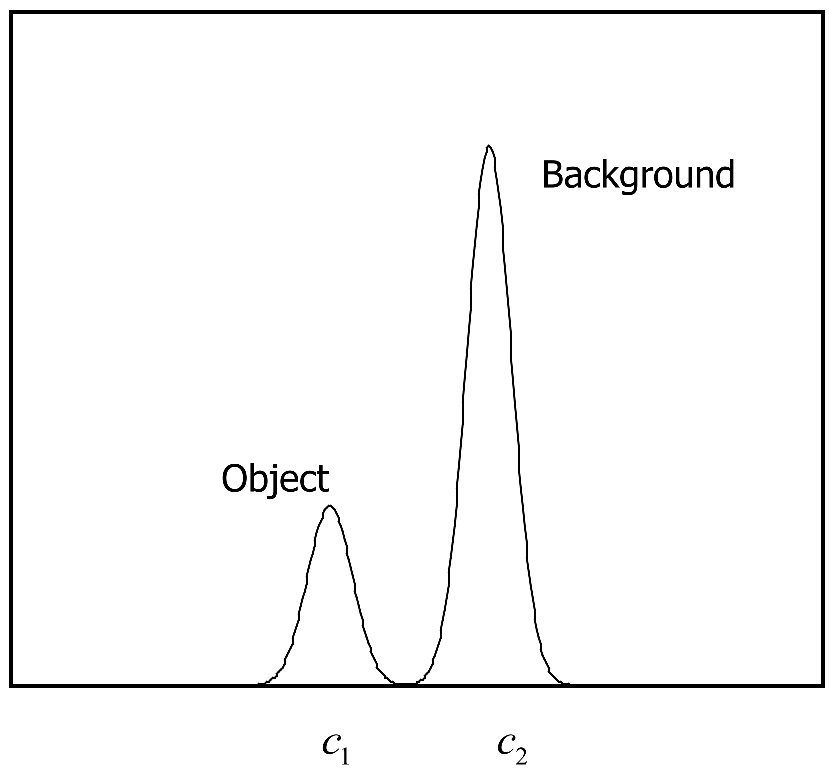


Greylevel thresholding

- Easy to see that, in both cases, for any value of the threshold, object pixels will be miss-classified as background and vice versa
- For greater histogram overlap, the pixel miss-classification is obviously greater
  - We could even quantify the probability of error in terms of the mean and standard deviations of the object and background histograms

Greylevel clustering

- Consider an idealized object/background histogram



Greylevel clustering

- Clustering tries to separate the histogram into 2 groups
- Defined by two cluster centres c_1 and c_2
  – Greylevels classified according to the nearest cluster centre

Greylevel clustering

- A *nearest neighbour* clustering algorithm allows us perform a greylevel segmentation using clustering
  - A simple case of a more general and widely used *K-means* clustering
  - A simple iterative algorithm which has known convergence properties

Greylevel clustering

- Given a set of greylevels

$$\{g(1), g(2)......g(N)\}$$

- We can partition this set into two groups

$$\{g_1(1), g_1(2)......g_1(N_1)\}$$

$$\{g_2(1), g_2(2)......g_2(N_2)\}$$

Greylevel clustering

- Compute the local means of each group

$$c_1 = \frac{1}{N_1}\sum_{i=1}^{N_1} g_1(i)$$

$$c_2 = \frac{1}{N_2}\sum_{i=1}^{N_2} g_2(i)$$

Greylevel clustering

- Re-define the new groupings

$$\left|g_1(k)-c_1\right|<\left|g_1(k)-c_2\right| \;\; k=1\ldots N_1$$

$$\left|g_2(k)-c_2\right|<\left|g_2(k)-c_1\right| \;\; k=1\ldots N_2$$

- In other words all grey levels in set 1 are nearer to cluster centre c_1 and all grey levels in set 2 are nearer to cluster centre c_2

Greylevel clustering

- But, we have a *chicken and egg* situation
  - The problem with the above definition is that each group mean is defined in terms of the partitions and vice versa
  - The solution is to define an iterative algorithm and worry about the convergence of the algorithm later

Greylevel clustering

- The iterative algorithm is as follows

Initialize the label of each pixel randomly

Repeat

c_1= mean of pixels assigned to object label

c_2= mean of pixels assigned to background label

Compute partition $\{g_1(1), g_1(2) \ldots g_1(N_1)\}$

Compute partition $\{g_2(1), g_2(2) \ldots g_2(N_2)\}$

Until none pixel labelling changes

Greylevel clustering

- Two questions to answer
  - Does this algorithm converge?
  - If so, to what does it converge?
- We can show that the algorithm is guaranteed to converge and also that it converges to a sensible result

Greylevel clustering

- Outline proof of algorithm convergence
  - Define a ‘cost function’ at iteration *r*

$$E^{(r)} = \frac{1}{N_1}\sum_{i=1}^{N_1}\left(g_1^{(r)}(i) - c_1^{(r-1)}\right)^2 + \frac{1}{N_2}\sum_{i=1}^{N_2}\left(g_2^{(r)}(i) - c_2^{(r-1)}\right)^2$$

$$E^{(r)} > 0$$

Greylevel clustering

- Now update the cluster centres

$$c_1^{(r)} = \frac{1}{N_1} \sum_{i=1}^{N_1} g_1^{(r)}(i)$$

$$c_2^{(r)} = \frac{1}{N_2} \sum_{i=1}^{N_1} g_2^{(r)}(i)$$

- Finally, update the cost function

$$E_1^{(r)} = \frac{1}{N_1} \sum_{i=1}^{N_1} \left(g_1^{(r)}(i) - c_1^{(r)}\right)^2 + \frac{1}{N_2} \sum_{i=1}^{N_2} \left(g_2^{(r)}(i) - c_2^{(r)}\right)^2$$

Greylevel clustering

- Easy to show that

$$E^{(r+1)} < E_1^{(r)} < E^{(r)}$$

- Since $E^{(r)} > 0$, we conclude that the algorithm must converge
  - *but*
- What does the algorithm converge to?

Greylevel clustering

- E_1 is simply the sum of the variances within each cluster which is minimised at convergence
  - Gives sensible results for well separated clusters
  - Similar performance to thresholding

Greylevel clustering