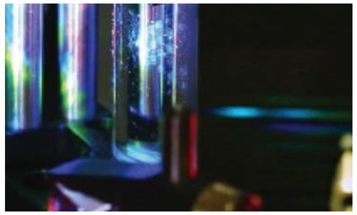
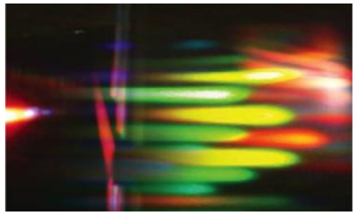
Imperial College London





COMP70058 Computer Vision

Lecture 8 – Texture and Region-based Segmentation

Stamatia (Matina) Giannarou, PhD
The Hamlyn Centre for Robotic Surgery

stamatia.giannarou@imperial.ac.uk

The Hamlyn Centre for Robotic Surgery

Contents

Image Segmentation

- The goal of segmentation is to divide an image into regions or classes that have similar properties. Ideally, pixels in same category have similar properties and are connected while, neighbouring pixels that are in different classes have dissimilar properties

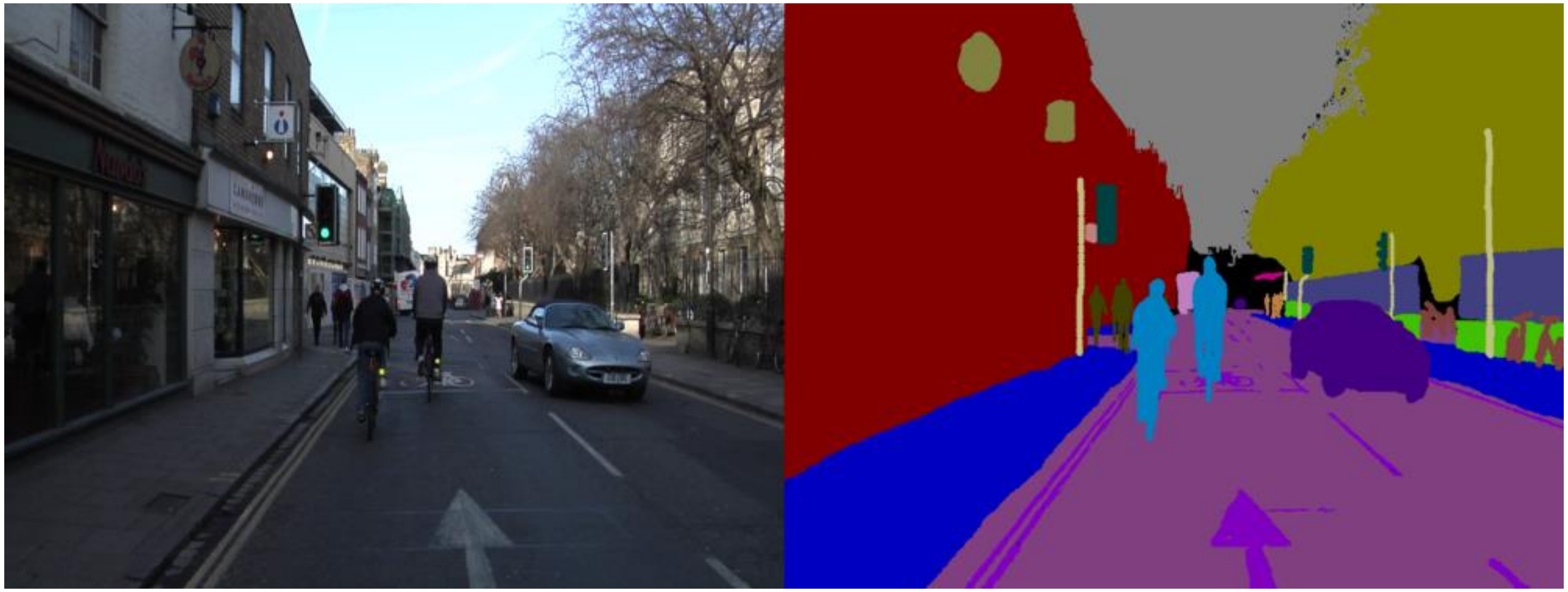
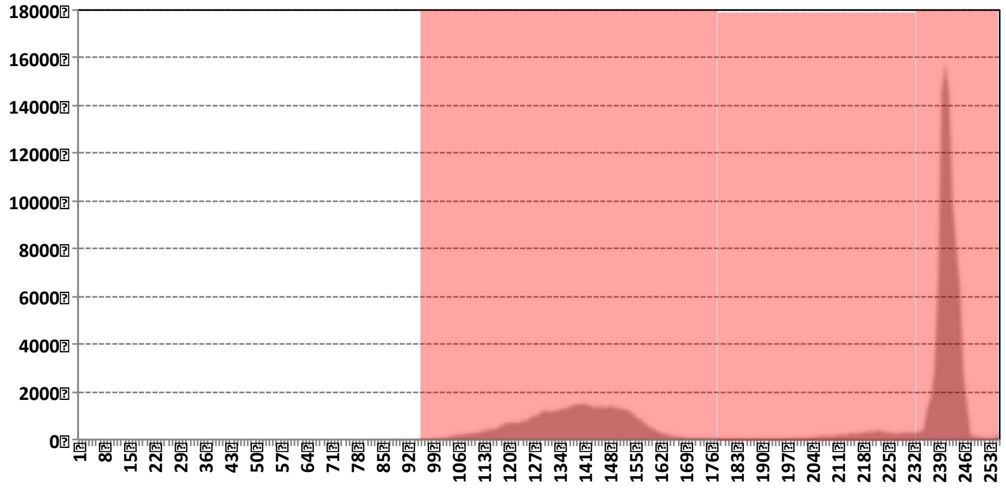


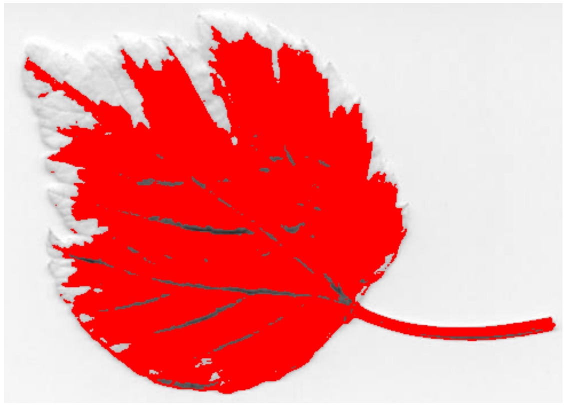
Image Segmentation

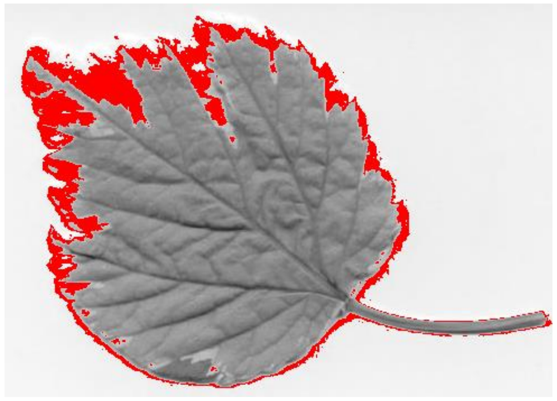
- A number of techniques have been proposed for segmenting images by identifying regions of some common property.
  - **Edge-based** techniques that locate the boundaries of specific regions
  - **Pixel based** techniques use similarities in pixel intensity values
  - **Region based** techniques expand segmented regions about specific seed points to form connected regions
- The extracted information is then analysed and modified as needed to form **closed regions** belonging to the objects in the image.

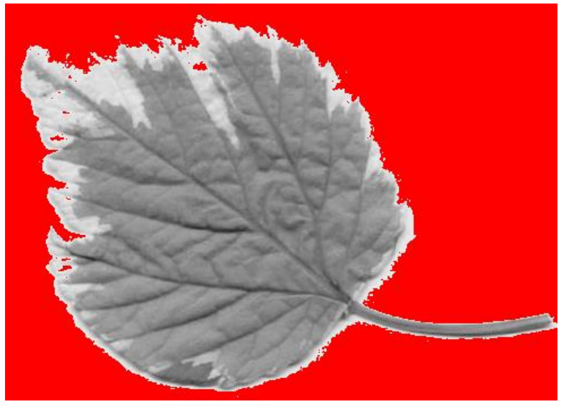
Segmentation by Histogram

- **Histogram**: the frequencies (occurrences) of pixels within the image with a given intensity value. They are allocated to 'bins', so for an eight-bit image, there will be 256 bins.



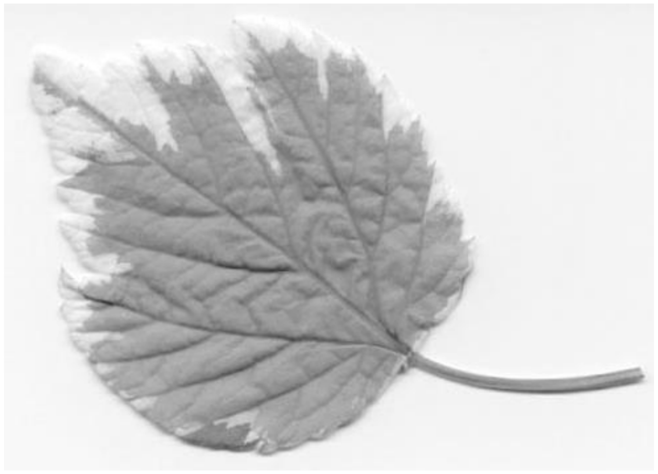
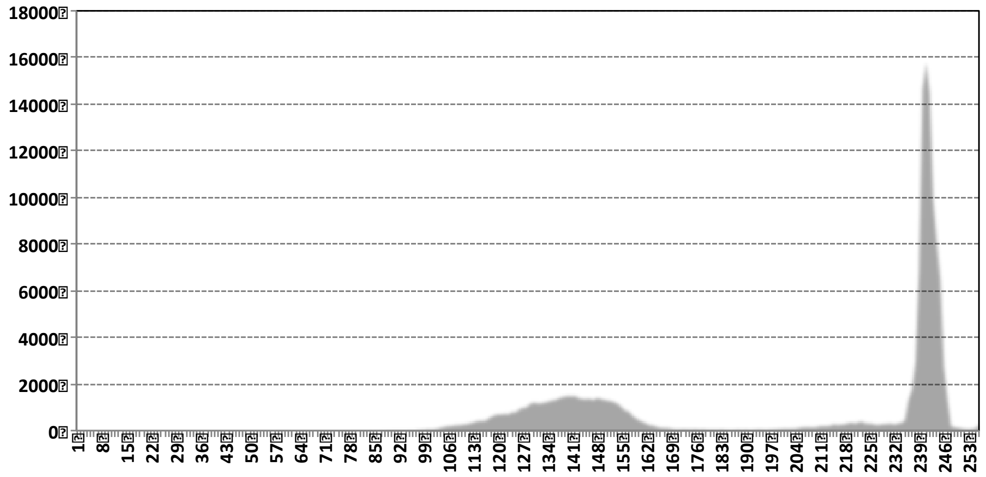
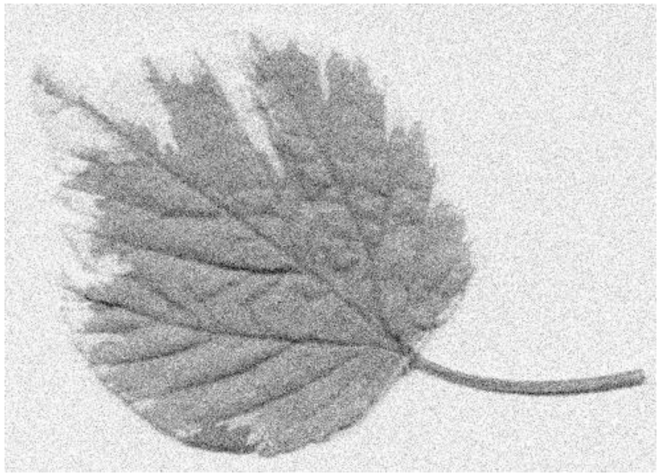
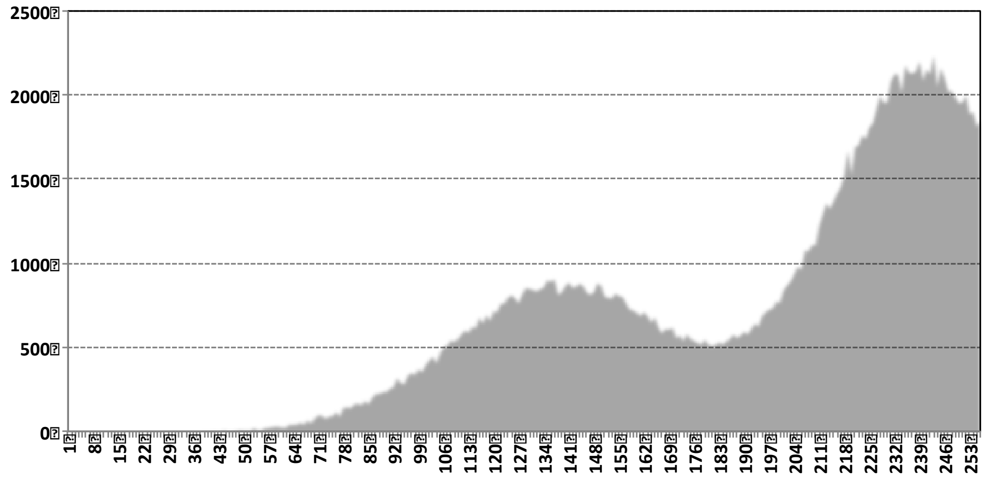






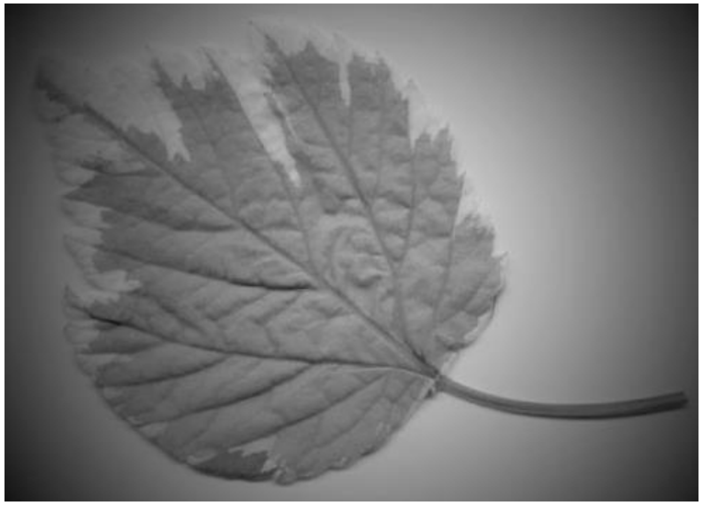
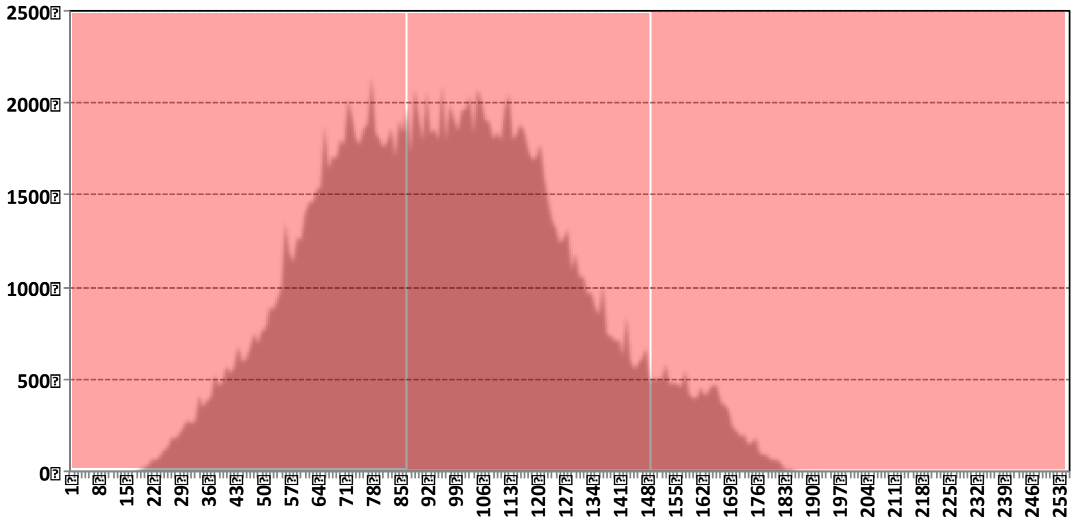
Segmentation by Histogram

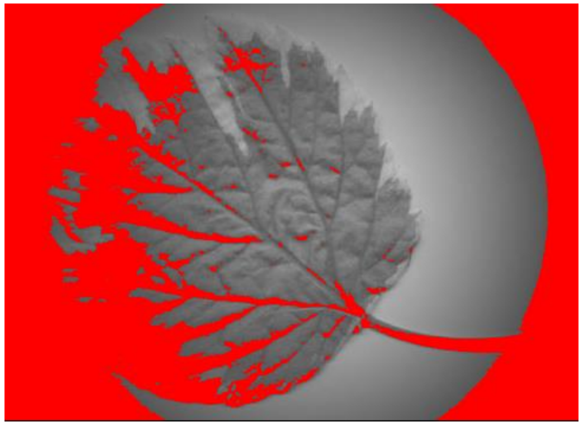
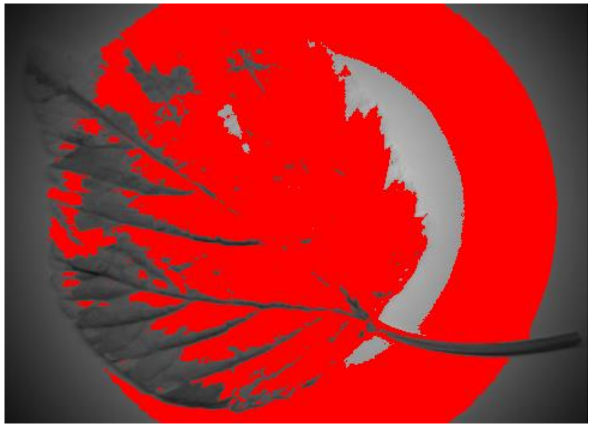
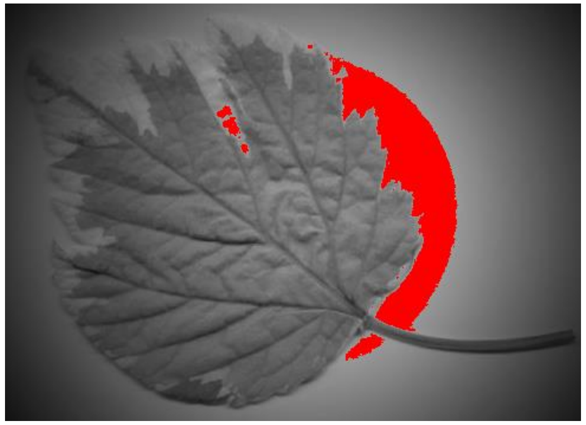
- **Histogram**: the effect of noise (depends on the distribution of noise, generally can cause a broadening effect of the histogram).

Segmentation by Histogram

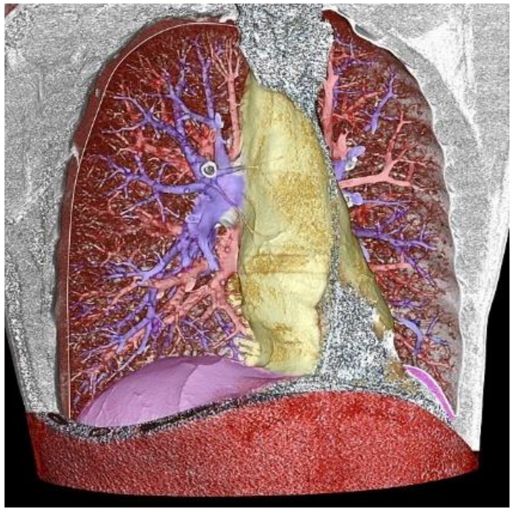
- **Effect of Lighting:** this creates a background bias field (which can be additive or multiplicative, distorting the distribution and making simple thresholding difficult.

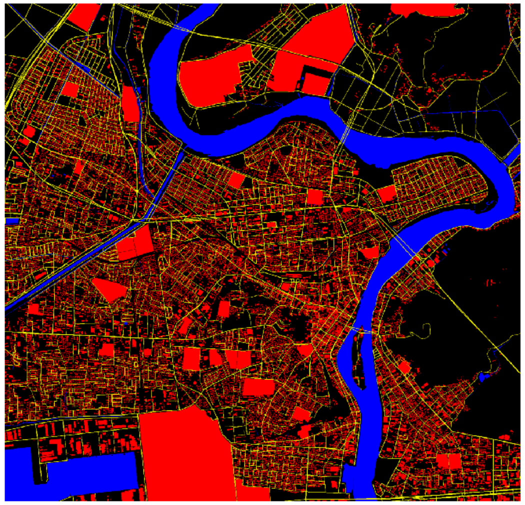



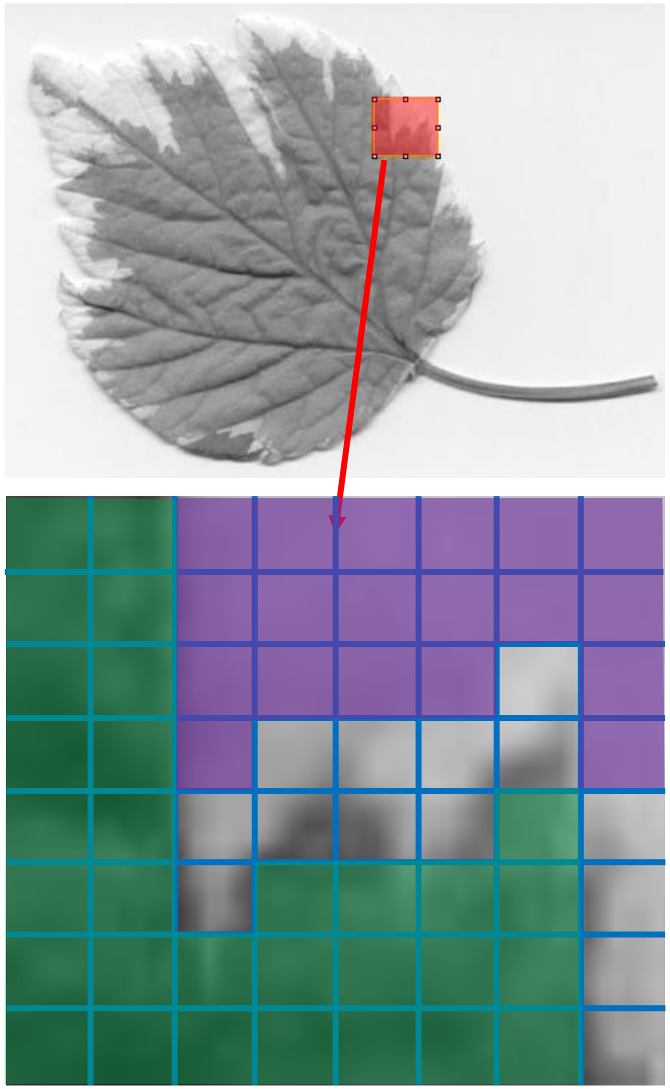
Region-based image segmentation

- These techniques examine the local neighbourhood of a pixel to look for relationships between groups of pixels.
- One common technique for achieving this is *region growing*, or *region merging.*
- A seed point is selected manually or automatically.
- Neighbouring pixels are assessed one by one and added to the growing region provided they satisfy a *uniformity test.*
- If a pixel fails the test then it is labelled as an edge.
- Performance depends on the uniformity test used – may result in merging of disconnected regions.
- This method segments areas with the same properties and generates connected regions.





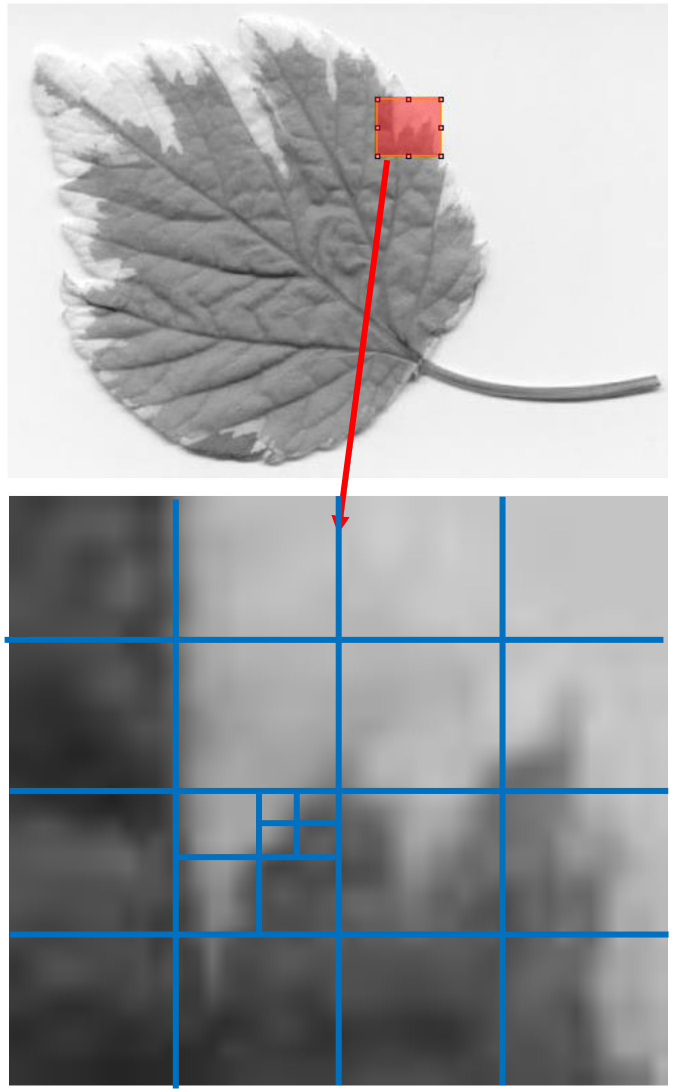
Region Merging



- Merging must start from a uniform seed region. Some work has been done in discovering a suitable seed region. One method is to divide the image into 2x2 or 4x4 blocks and check each one. Another is to divide the image into strips, and then subdivide the strips further. In the worst case the seed will be a single pixel. Once a seed has been found, its neighbours are merged **until no more neighbouring regions conform to the uniformity criterion**. At this point the region is extracted from the image, and a further seed is used to merge another region.

There are some drawbacks which must be noted with this approach. The process is inherently **sequential**, and if fine detail is required in the segmentation then the computing time will be long. Moreover, since in most cases the merging of two regions will change the value of the property being measured, the resulting area will depend on the **search strategy** employed among the neighbours, and the seed chosen.

Region Splitting

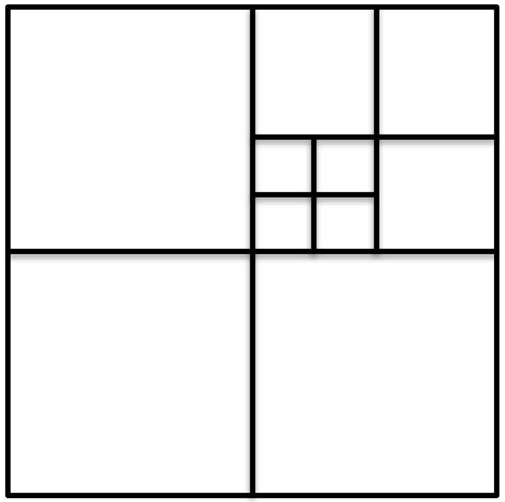


- Begins from the whole image, and divide it up until each sub region is uniform. The usual criterion for stopping the splitting process is when **the properties of a newly split pair do not differ from those of the original region** by more than a threshold.

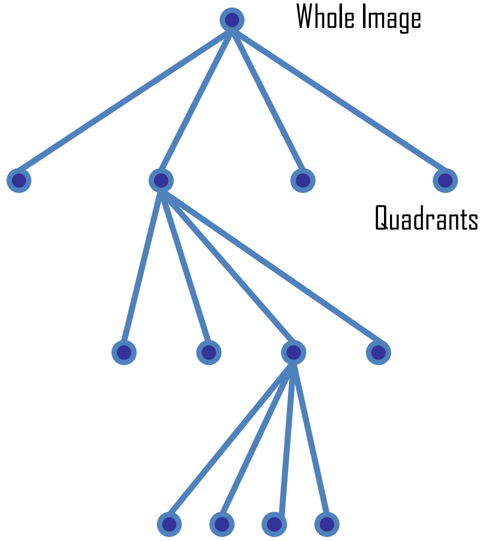
The main problem with this algorithm is the difficulty of deciding **where to make the partition**. Early algorithms used some regular decomposition methods and for some classes these are satisfactory; however, in most cases, splitting is used as a first stage of a split/merge algorithm.

Split and Merge

- Region splitting results in many small homogeneous areas, but neighbouring areas may be similar and should be merged.
- Various algorithms combine both processes
    - Either refine a boundary by both techniques. Increasing detail as the boundary becomes more accurate
    - Or split image to find large seeds, then merge (usually through regular decomposition)
- An important data structure used in split and merge algorithms is the quadtree



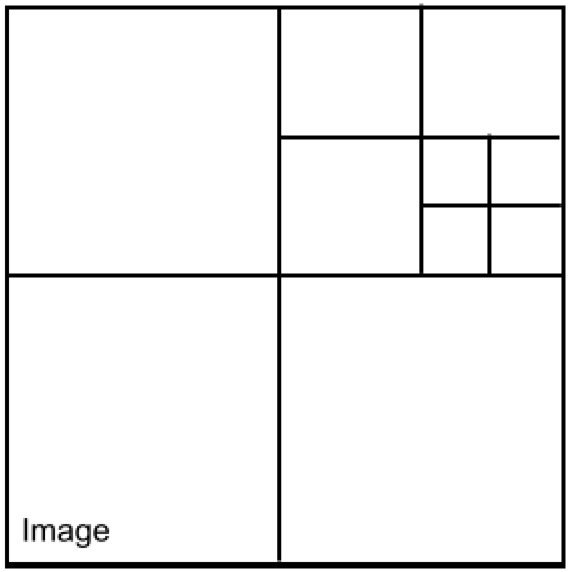
Image

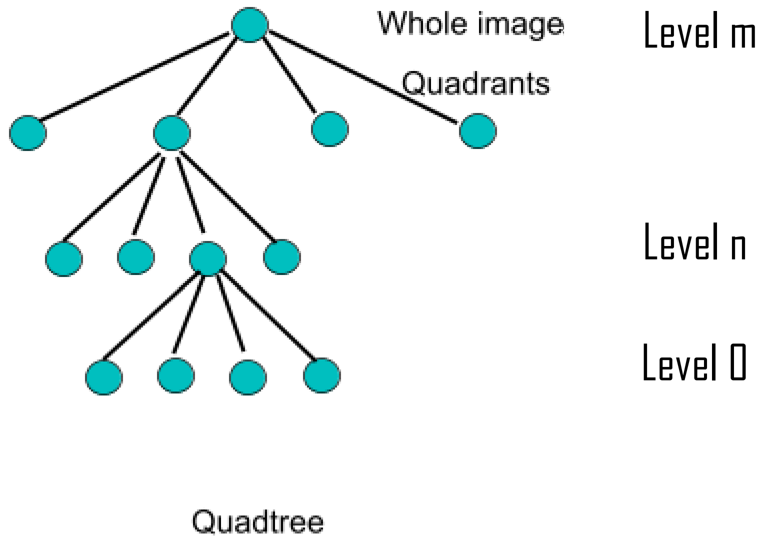


Quadtree

Note that in graphics the quadtree is used in a region splitting algorithm (Warnock's Algorithm) which breaks a graphical image down recursively from the root node, which represents the whole image, to the leaf nodes which each represent a coherent region, which can be rendered without further hidden line elimination calculations. The same use is made of quadtrees for vision.

Quadtrees

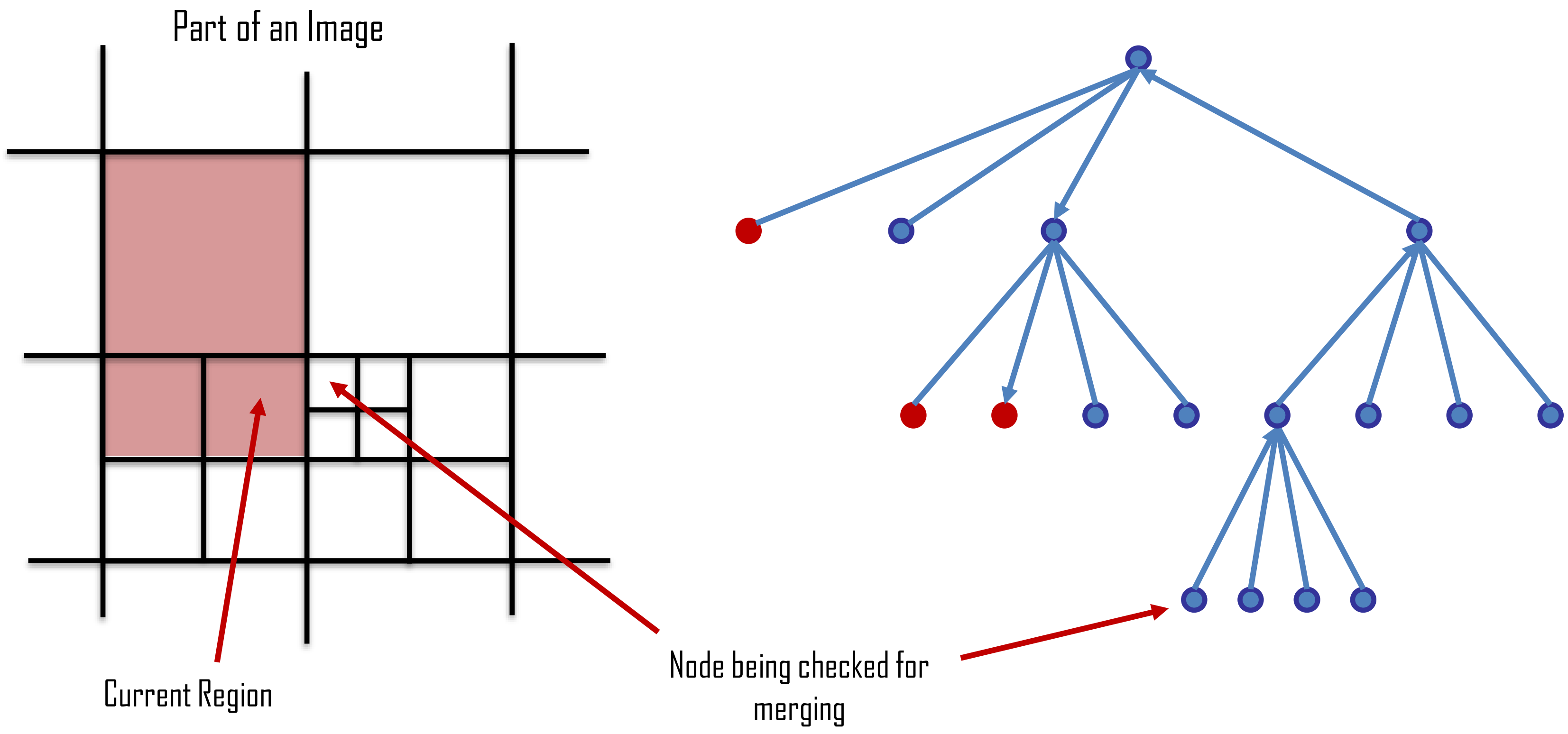




- The image is partitioned into four regions that are represented by nodes in a quad tree. Each region is checked for homogeneity. If the region is homogeneous, no further action is taken for the respective node. If the region is not homogeneous, it is further split into four regions.
- For an image with resolution $2^m x 2^m$, Quadtree has m+1 levels and $(2^{2(m+1)})-1)/3$ nodes
- level 0 - each node corresponds to a pixel
- level m - the root (whole image)
- level n - each node corresponds to region of side 2^n pixels

Quadtrees for Region Extraction

- Quadtrees impose one type of regular decomposition onto an image. To complete the segmentation process this must be followed by a merging phase. Thus, the problem of finding adjacent neighbours to a given node has been studied. The problem is one of tree search and efficient algorithms have been published.



Splitting and Merging with Quadtrees

Quadtree Splitting and Merging Algorithm

1. Define an initial segmentation into regions, a homogeneity criterion, and a pyramid data structure.

2. If a region R in the pyramid data structure is not homogeneous, split it into four child-regions; if any of the four regions with the same parent can be merged into a single homogeneous region, merge them. If no region can be split or merged, go to step (3).

3. If there are any two adjacent regions Ri, Rj (even if they are in different pyramid levels or do not have the same parent) that can be merged into a homogeneous region, merge them.

4. Merge small regions with the most similar adjacent region if it is necessary to remove small-size regions.

Uniformity Criteria

- What criteria can we use to define thresholds for splitting and merging? How do we define uniformity?
- If the mean of the pixel in a subtree is represented by

$$\mu = \frac{1}{n}\sum f(x,y)$$

- Then the variance is defined as:

$$\mathrm{v} = \frac{1}{n}\sum (f(x,y) - \mu)^2 = \frac{1}{n}\sum f(x,y)^2 - \mu^2$$

- Min/Max

Uniformity Criteria

- What criteria can we use to define thresholds for splitting and merging? How do we define uniformity?

- Mean
  - Gives no uniformity information ✗
- Min/Max
  - Works in good images, however random noise can lead to wrong results ✗
- Variance
  - Is a statistical measure of how close to the mean a set of data is ✓

The measure of mean value along alone gives us no measure of uniformity which is essential in practical cases. Instead we need to use some statistical measure, and it turns out that variance is a convenient, and easy to compute property. Hence at each node in the quadtree we will store both the mean and the variance of the pixels in the subtree.

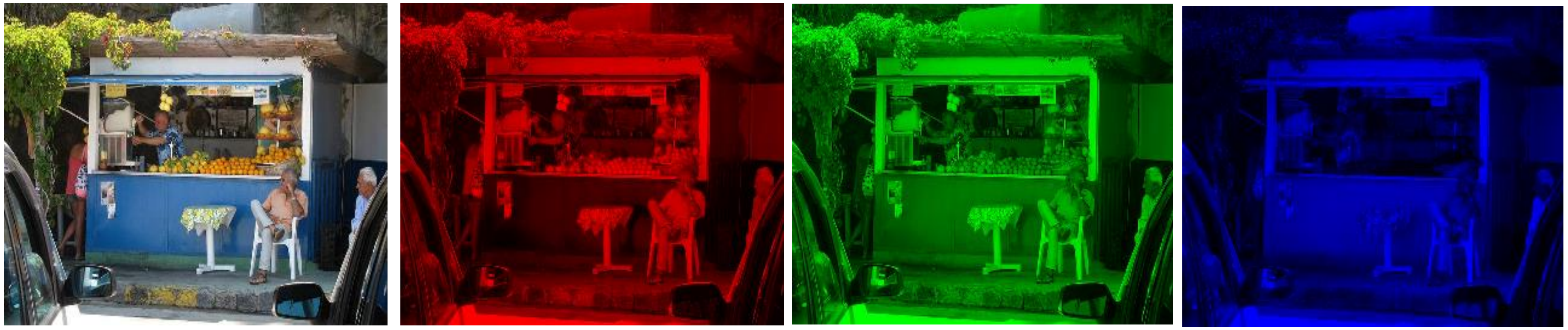
Fischer’s Criterion

- Analysis based on mean and variance makes an assumption that the feature upon which the segmentation is based is **distributed normally**
- Use Fischer’s Criterion to discriminate between adjacent areas with differing means and standard deviations:

$$\frac{|\mu_1 - \mu_2|}{\sqrt{v_1^2 + v_2^2}} > \lambda$$

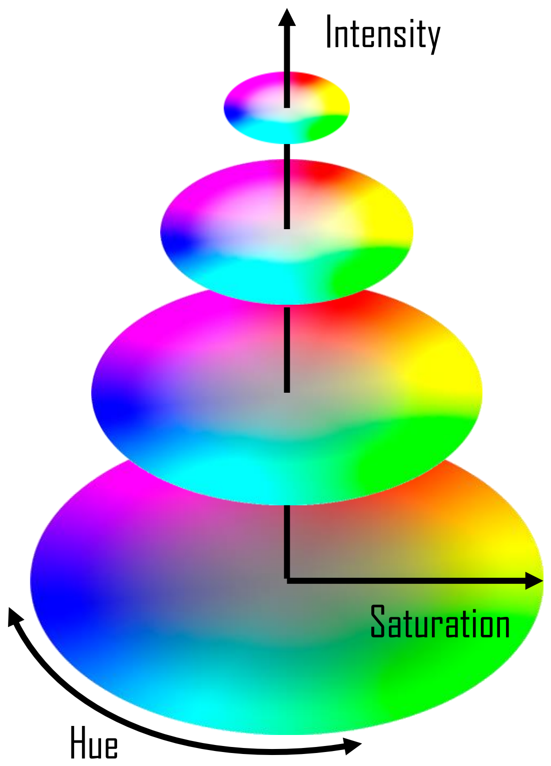
- Where λ is a threshold
- If two regions have **good separation in their means** and **low variance**, then we can discriminate them. However, if the variance becomes high and the mean difference is low, it is not possible to separate them.

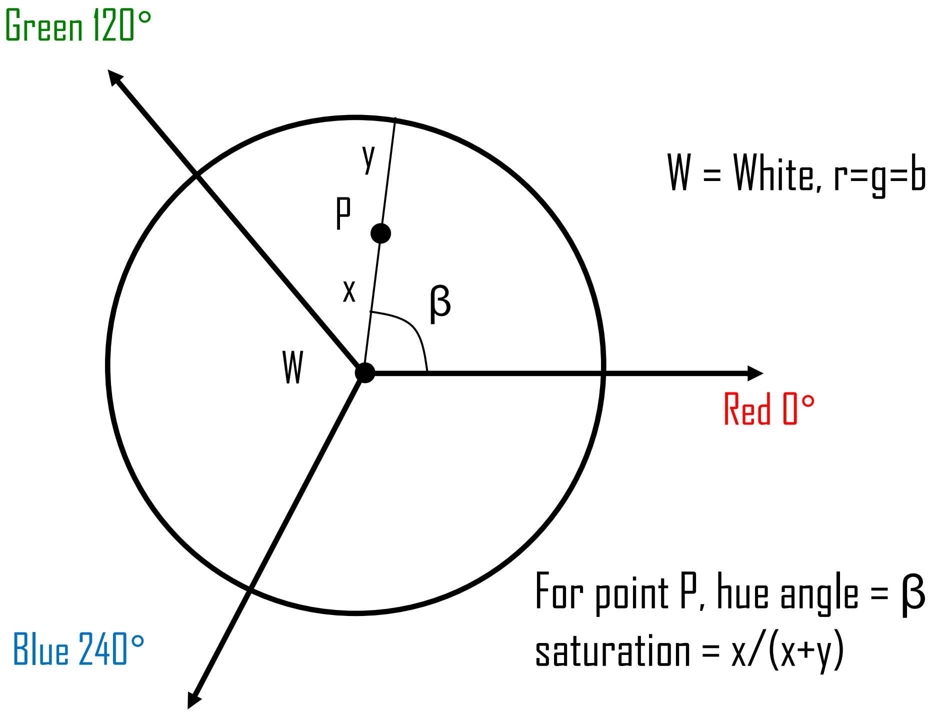
Colour Space – Tri-Stimulus Representation



Colour provides an important property on which we can segment images and it has advantages over intensity in some cases. In practical systems, colour is normally represented using the tri-stimulus model, with each pixel specified using 3 bytes – RGB. We could choose to perform the segmentation on **one of these colour planes**. This approach turns out to be of limited application and a more appropriate method is to utilise a **different colour space**, where the same information is represented in a way that corresponds better to the segmentation method.

Conceptual Colour Circle





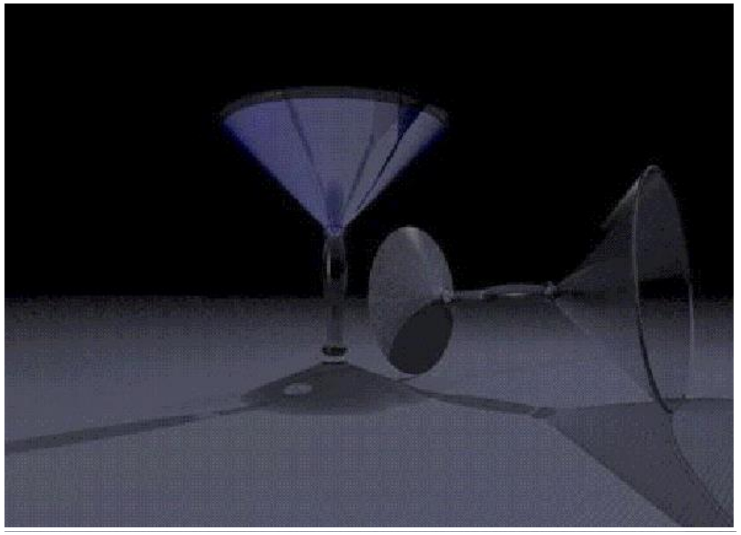
W = White, r=g=b

For point P, hue angle = β and saturation = x/(x+y)

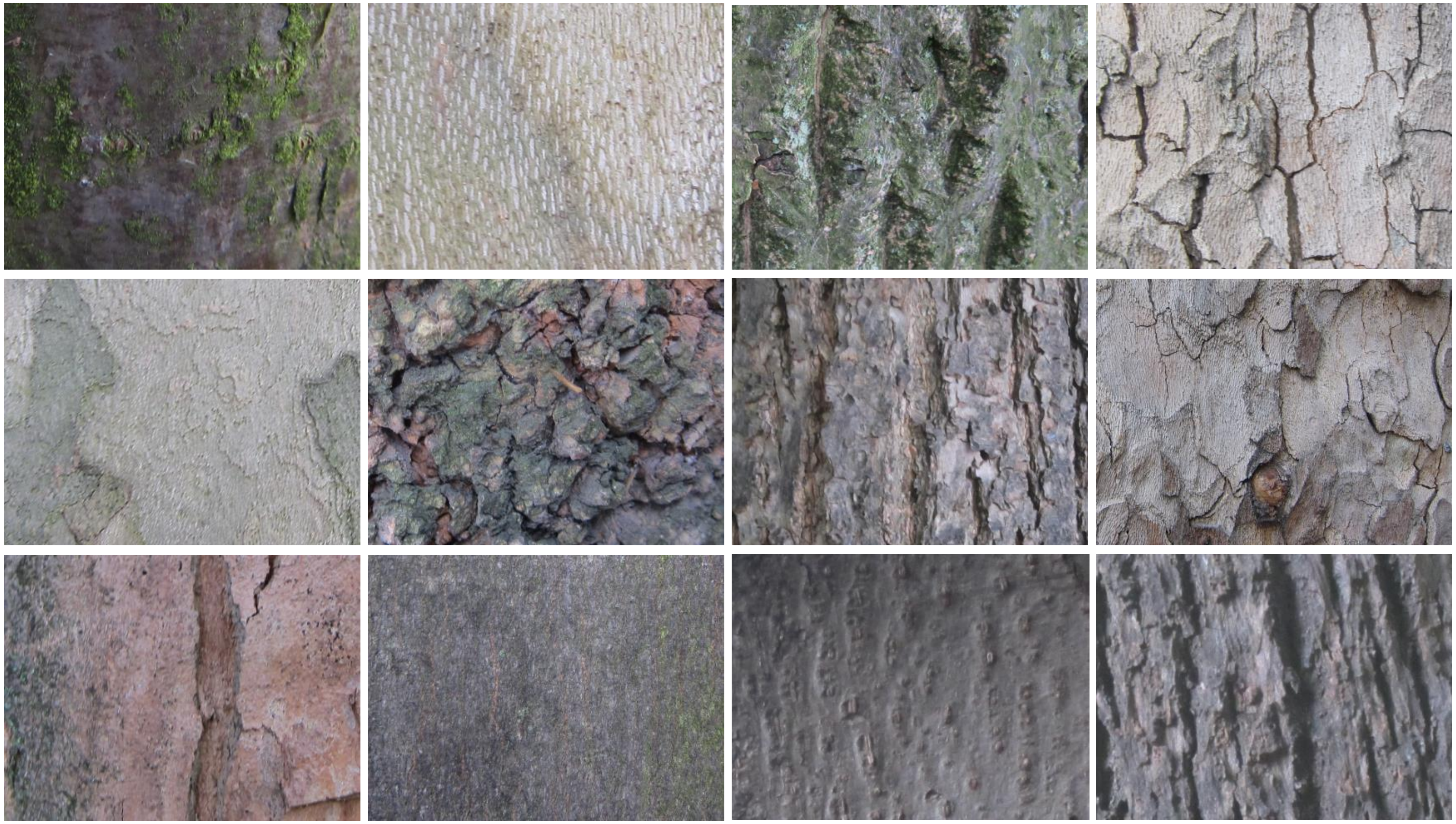
Saturation is the proportion of pure colour. Since hue is represented by an angle in the range 0-2π and saturation in the range 0-1, we can represent each point in RGB space on a unit circle. The centre of the circle is white and the perimeter is the fully saturated pure colours.

Advantages of Using Hue

- Using hue for segmentation has several immediate advantages over intensity
- Shadows: no change in hue, only intensity
- Specular Reflections: no change in hue, only a change in saturation
    - Saturation, like the r, g, and b planes may be helpful in a particular application



Textures



- Surface inspection
- Scene classification
- Surface orientation and shape determination

Textures and Co-occurrence

- A general method for determining the characteristics of a textured region is to use a region **histogram**
- These must be normalised to compare histograms of two textured regions
- One possible matching technique – **Euclidean distances** of the histogram separation
- This cannot separate patterns which have the same pixel intensity distributions
- Instead, can we identify repeated spatial patterns?
- Co-occurrence matrices
    - Measure the number of times a pair of pixels at some defined separation have a given pair of intensities
    - Size = number of grey levels in the image, and is computed for a given direction and a given distance

Co-occurrence Matrices

$$\begin{matrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 2 & 2 & 2 \\ 2 & 2 & 3 & 3 \end{matrix}$$

- This image has four grey levels, therefore the co-occurrence matrices will have a size of 4x4

Co-occurrence Matrices

Texture Image

$$\begin{array}{ccccc}
0 & 0 & — & 1 & 1 \\
0 & 0 & — & 1 & 1 \\
0 & 2 & & 2 & 2 \\
2 & 2 & & 3 & 3
\end{array}$$

This image has 4 grey levels, therefore the co-occurrence matrices will have a size of 4 × 4.

Co-occurrence matrix

$$\begin{array}{c|cccc}
 & 0 & 1 & 2 & 3 \\
\hline
0 & 4 & 2 & 1 & 0 \\
1 & \boxed{2} & 4 & 0 & 0 \\
2 & 1 & 0 & 6 & 1 \\
3 & 0 & 0 & 1 & 2
\end{array}$$

$$\longleftrightarrow \quad P_{0^\circ,1}$$

Co-occurrence Matrices

Texture Image

$$\begin{matrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 2 & 2 & 2 \\ 2 & 2 & 3 & 3 \end{matrix}$$

This image has 4 grey levels, therefore the co-occurrence matrices will have a size of 4 × 4.

Co-occurrence Matrix

$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 2 & 1 & 3 & 0 \\ 1 & 1 & 2 & 1 & 0 \\ 2 & \boxed{3} & 1 & 0 & 2 \\ 3 & 0 & 0 & 2 & 0 \end{array}$$

$\longleftrightarrow$ $P_{135^\circ,1}$

Co-occurrence Matrices

- Non-normalised frequencies of co-occurrence as functions of angle and distance can be represented formally as:

$$P_{0°,d}(a,b) = \left| \left\{ \begin{array}{c} (k,l),(m,n) \in (M \times N) \times (M \times N) : \\ k-m=0, |l-n| = d, f(k,l) = a, f(m,n) = b \end{array} \right\} \right|$$

- Similar equations can be derived for 90° and diagonal directions
- Practical Considerations
  - Quantise the image into a small number of grey levels, typically no more than 64
  - Restrict **distances** to < four pixels
  - Restrict the **directions** to four (horizontal, vertical, and two diagonal directions)
    - So from a texture window of any size we would have **sixteen** 64 × 64 matrices

Texture Matching

- We must normalise the co-occurrence matrices so that the entries become probabilities of co-occurrence, and are therefore independent of window size
- Matching can be carried out by extracting a number of characteristic properties of the matrices
- For example, energy (or angular second moment. It is an image homogeneity measure – the more homogeneous the image, the larger the value)

$$Energy = \sum_{a,b} P^2_{\varphi,d}(a,b)$$

- If we consider the energy of the sixteen 64 × 64 matrices to form a characteristic vector (16 dimensional), then the difference between two windows containing texture can be measured by either the Euclidean or the Manhattan distance between them

Texture Matching – Other Properties

$$Entropy = -\sum_{a,b} P_{\varphi,d}(a,b)\log\left(P_{\varphi,d}(a,b)\right)$$

$$Contrast = \sum_{a,b}(a-b)^2 P_{\varphi,d}(a,b)$$

$$Homogeneity = \sum_{a,b}\frac{P_{\varphi,d}(a,b)}{1+|a-b|}$$

Co-occurence matrices work well in specific applications but are limited in that they cannot cope with stochastic textures or with textures where the granularity cannot be made to match the pixel size.

Model Based Methods

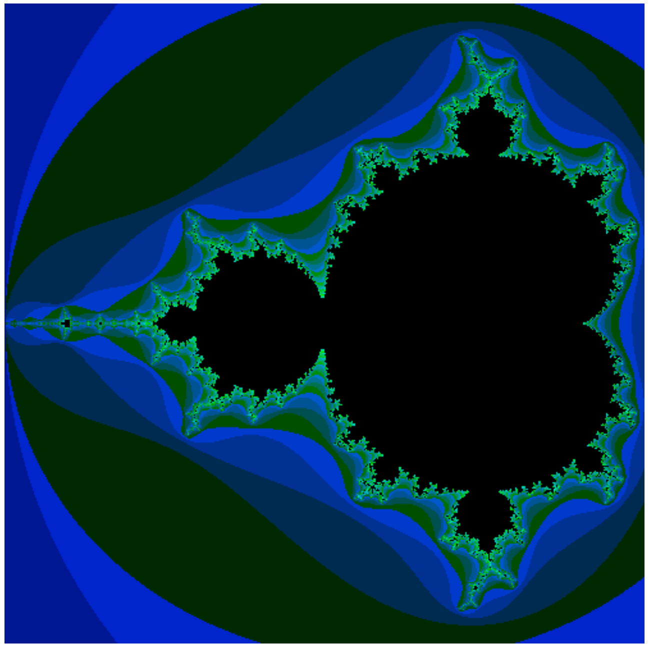
- Another approach is to use an analytical model of the texture
- Markov Random Fields (MRFs)
  - The grey level at any pixel is modelled as a linear combination of grey levels of its neighbours plus an additive noise term

$$f(i,j) = \sum_{k,l} f(i-k, j-l)h(k,l) + n(i,j)$$

The summation is carried out over a specified set of pixels which are neighbours of pixel (i,j). The weights h(k,l) are the parameters of this model and are computed from the texture image using a **least-mean-squares** method. These estimated parameters are then compared with those of the known texture classes to determine the class of the particular texture being analysed.

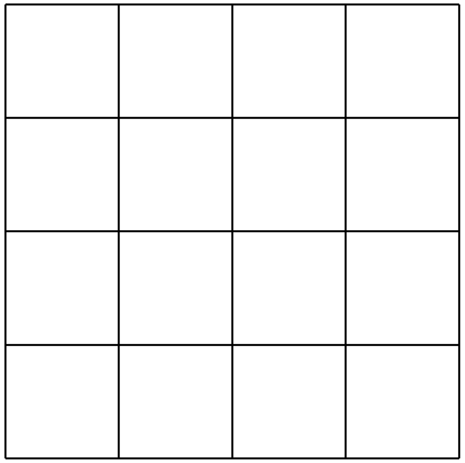
Model Based Methods

- When the pattern forming texture has the property of **self-similarity at different scales**, fractal-based methods may be used
- A set is said to have the property of self-similarity if it can be decomposed as a non-overlapping union of N copies of itself scaled down by a factor r
- Such a texture is characterised by its fractal dimension D

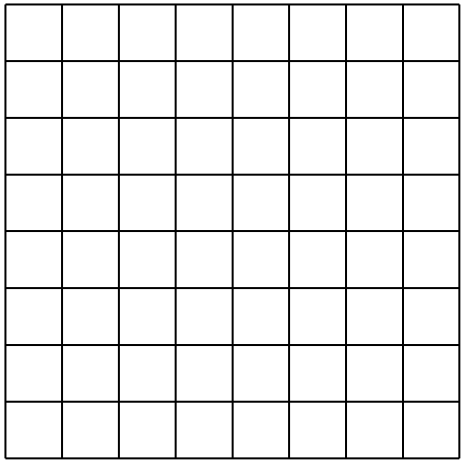


Fractal Dimension

- A square may be broken into N^2 self-similar pieces, each with magnification factor N



$N = 16, r = 1/4$

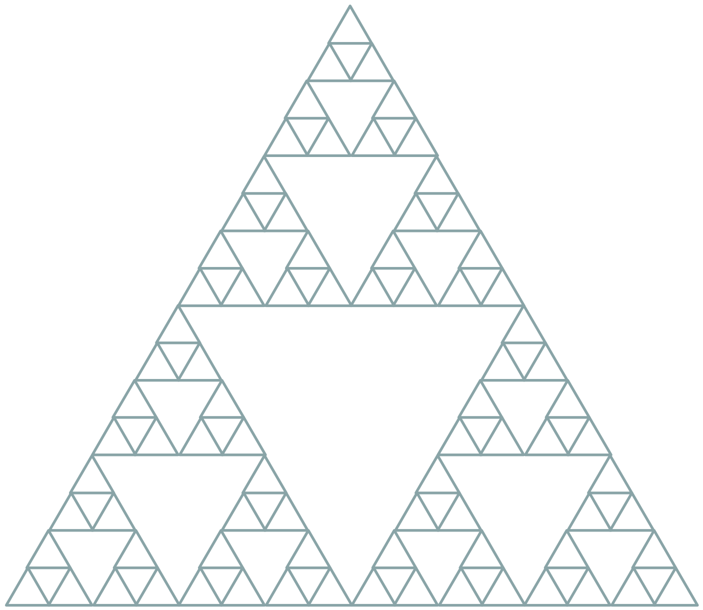


$N = 64, r = 1/8$

$$\boxed{D = \frac{\log N}{\log(1/r)}}$$

$$D = \log 16/\log 4 = 2\log 4/\log 4 = 2$$

Fractal Dimension – Sierpinski Triangle



$N = 3, r = 1/2$

$$D = \frac{\log N}{\log(1/r)}$$

$$D = \log 3 / \log 2 = 1.58$$

The fractal dimension is a useful feature for texture characterisation although estimation of D from an image is very difficult due to the fact that natural textures do not strictly follow the deterministic repetitive model of fractals assumed above but have statistical variations.

Conclusions

- Segmentation by histogram
- Region merging and splitting
- Quadtrees for region extraction
- Uniformity criteria
- Region features and merging criteria
- Texture and co-occurrence
- Model based methods for texture analysis