M30242 Graphics and Computer Vision

Lecture 7 Colour Analysis

Overview

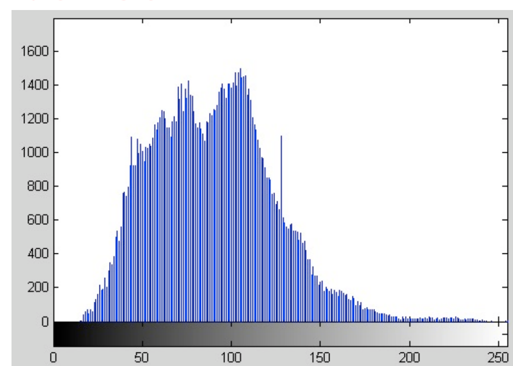
- Grayscale v.s. colour processing
- Colour histograms
- Compare colour histograms
- RGB colour space and colour vectors
- Colour-based image segmentation
  - Colour similarity measures: Euclidean and Mahalanobis distances

Limitations of Grayscale Processing

- Consider the problem of separating the flowers from the background

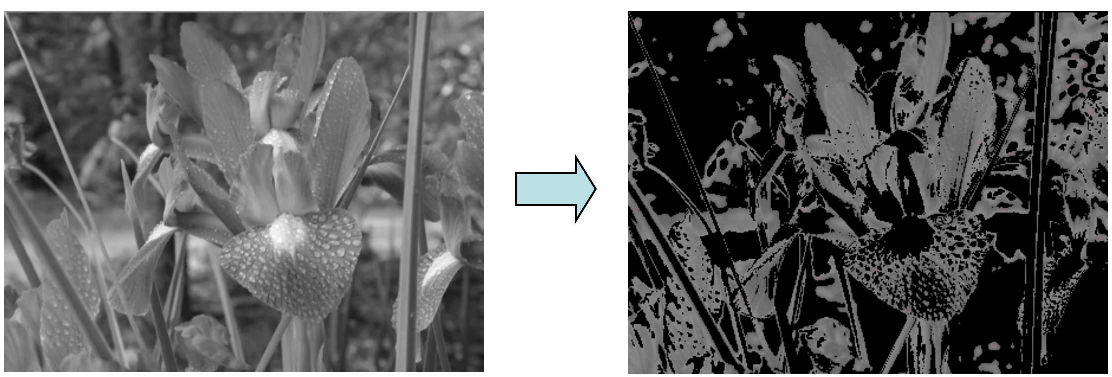
Cont’d

- If grayscale values are used, then a processing procedure might be:
  - convert the colour image to grayscale,
  - filter to remove noise (or do this in an earlier step),
  - threshold the gray image (double thresholds and a histogram might be needed to get a good result).



Cont'd

- Result of double thresholding: removal of pixels with intensities lower than 80 and higher than 120
- Is the result acceptable?



Many Mis-classified Areas

- For this image, the result would be better if we use colours to classify the pixels, because the flower's colour is dominantly blue, which is very different from the background

Using Colour

Separate out the blue channel

The intensity image of the blue channel

After thresholding (keep pixels with intensity>150)

Binary

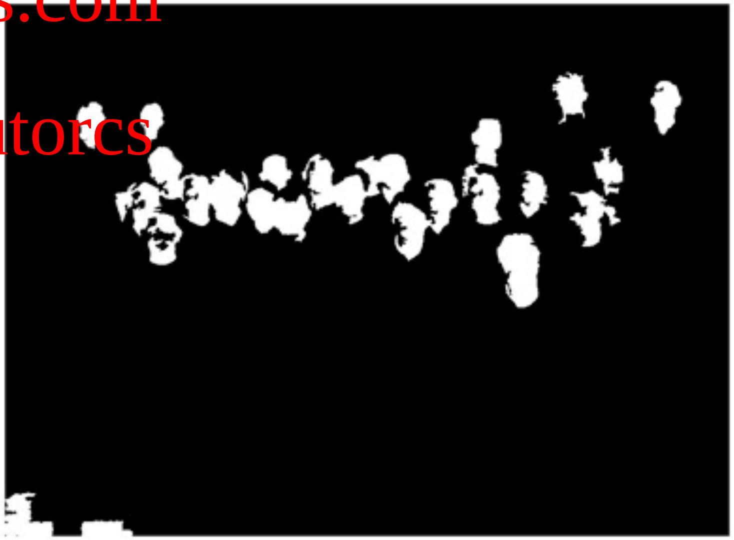
In General…

- Pixel colours provide rich information about objects. When processed as grayscale images, the information is lost.

- Colour information
  - Mainly used in *image segmentation* where pixels are grouped/classified into regions according to their colours, intensities, or neighbourhood connections.
  - Also used in *recognition and identification* where the aim is to detect the presence of an object or to find its identity.

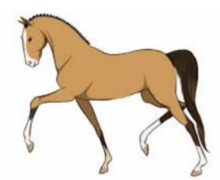
An Example

- Use of colour cues in face detection.



Another Example

- In most search engines, a search with the key word “horse” will return image of horses, as well as image of items that have a horse tag.





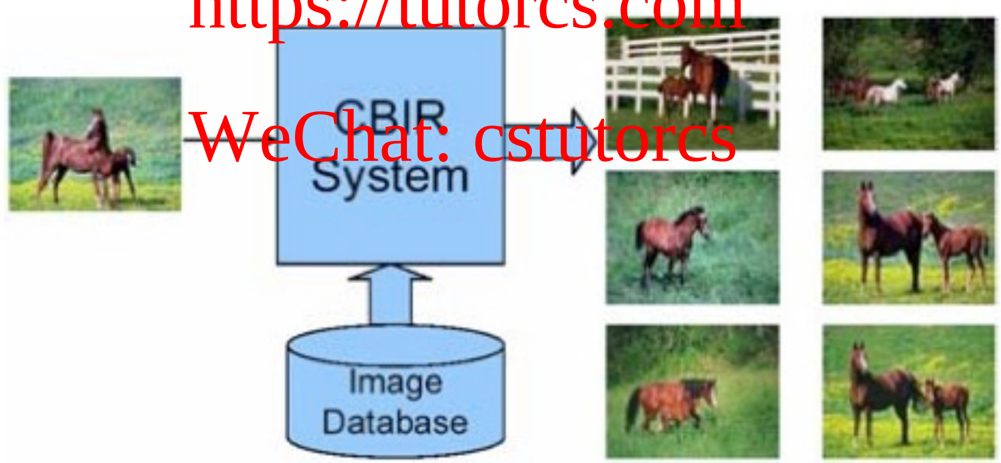

Cont’d

- How about searching for “horse in water” and “horse in snow”?

- For search engines to answer such queries, the images have to have richer annotations.
- Is annotation the way to go? A picture worth thousands of words!

Cont’d

- Content-base image retrieval aims to address this problem.
  - Colour statistics (global or local/regional statistics),
  - Texture-based, etc



One More Example

- Colours are used for stereo matching.
- Figure (a) shows a stereo pair.
- Figure (b) shows the depth maps obtained from
  - direct pixel matching (left), and,
  - colour informed matching (right).

(a)

(b)

Working with Colours

- In the example of flower detection, the flowers can be easily separated from other objects by thresholding the values of the blue channel of the image.
- In general, however, simple interpretation of pixel colours is error-prone:
  - Highlights change surface colours.
  - Lighting variations change colours.
- More sophisticated methods would use colour statistics:
  - Colour histogram,
  - Colour covariance,
  - Colour distances: Euclidean or others.

Colour Histogram

- Colour histograms show the statistics of colour distribution of images:
  - Divide the full range of colours into a fixed intervals and each interval corresponds to a “bin”,
  - Then counts the numbers of pixels fall in each bin.
- In principle, colour histogram is invariant to translation, rotation, small off-axis rotations, scale change and occlusions.
  - That is, the distribution of pixel colours remains unchanged under small changes to the scene or imaging conditions
- Colour histogram is a useful representation for image retrieval and object recognition.

Create Colour Histogram

- We usually do not perform histogram algorithm directly on a colour image.
  - It requires to quantise the whole colour space (all possible colour values of a pixel: 256^3 =16,777,216 colours).

- Two ways to get simplified colour histograms:
  - 1. Take out the higher two bits of each colour channel of a pixel, hence we have 2^6 = 64 values/colours
  - 2. Create a separate histogram for each colour channel and assemble them together as a single histogram.

Blue channel

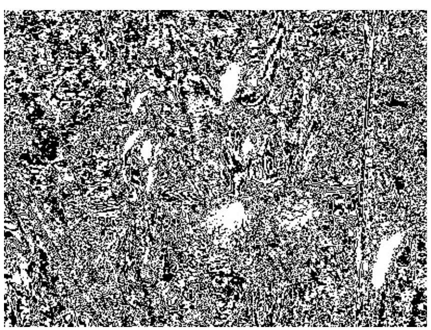
Images for each of the eight bits of the blue channel

Bit 8

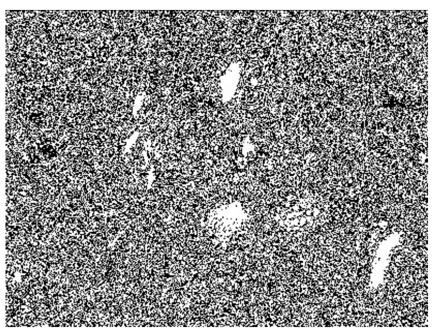
Bit 7

Bit 6

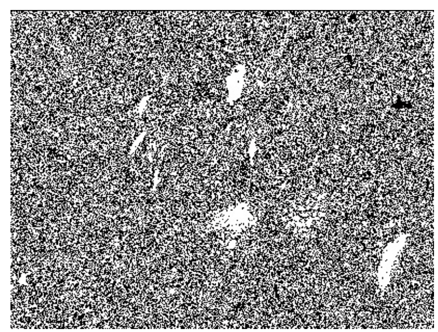
Bit 5



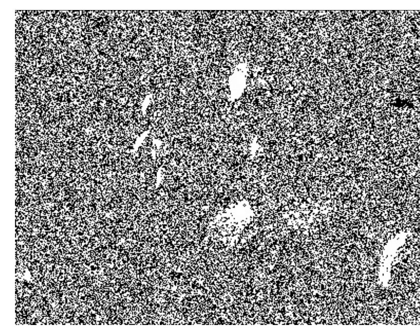
Bit 4



Bit 3



Bit 2



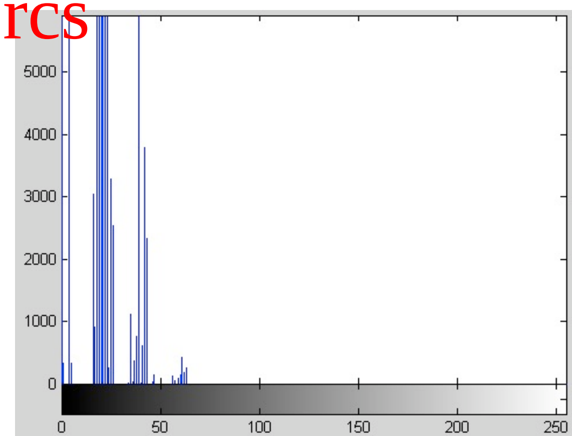
Bit 1

Histogram of Two Most Significant Bits from Each Channel

- Take the highest two bits from each of the three colour of a pixel.
- We have 6 bits for each pixel (2^6=64 colours). Arranging the bits, we can get a new image.
- Find the histogram of the newly formed image

R8	R7	G8	G7	B8	B7

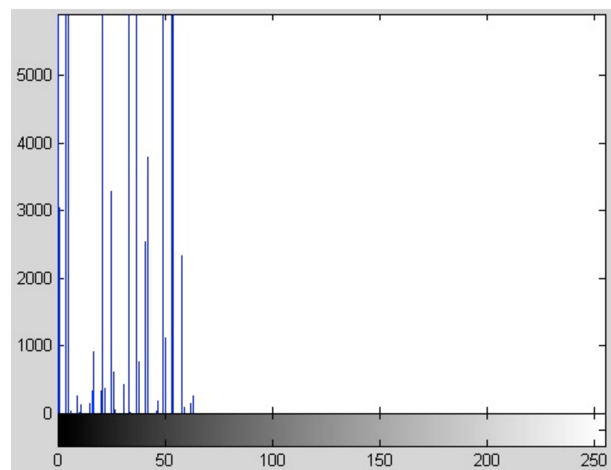
Any potential problems?



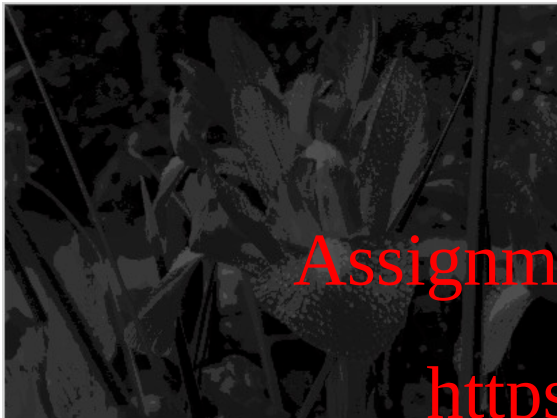
Histogram of Two Most Significant Bits from Each Channel

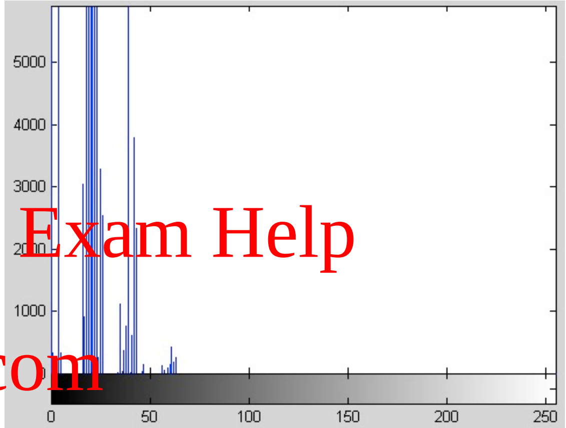
- One colour dominates the histogram.
- The positions the bits of a colour decide its importance in the histogram.
- We have a totally different histogram if we put the blue bit at the most significant position.

B8	B7	G8	G7	R8	R7

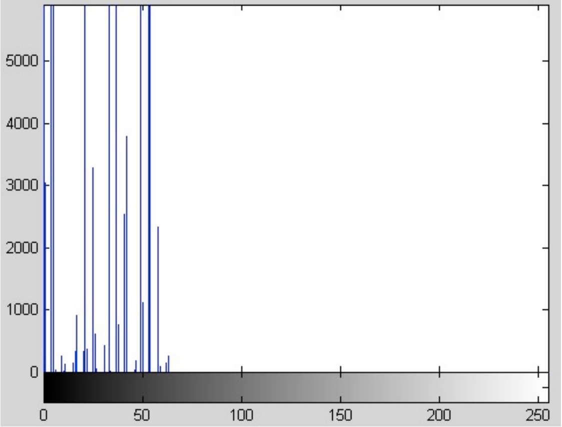


Compare





R8	R7	G8	G7	B8	B7



B8	B7	G8	G7	R8	R7

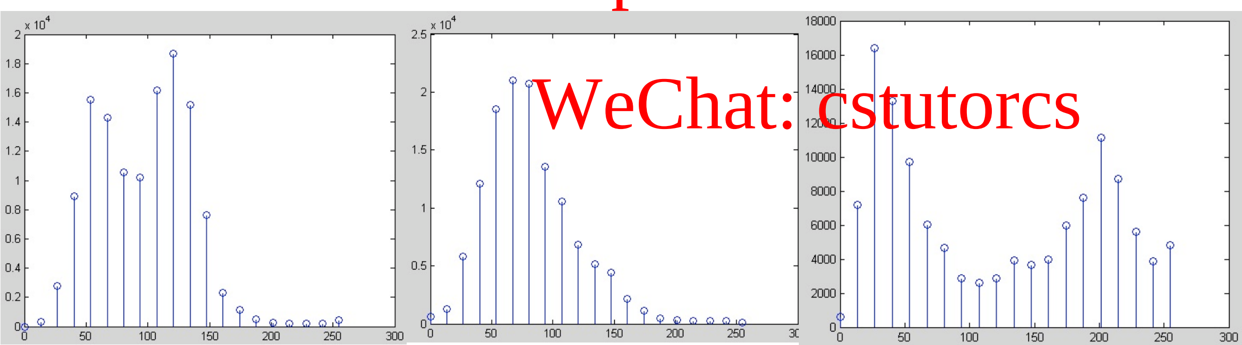
Create Histogram for Each Channel

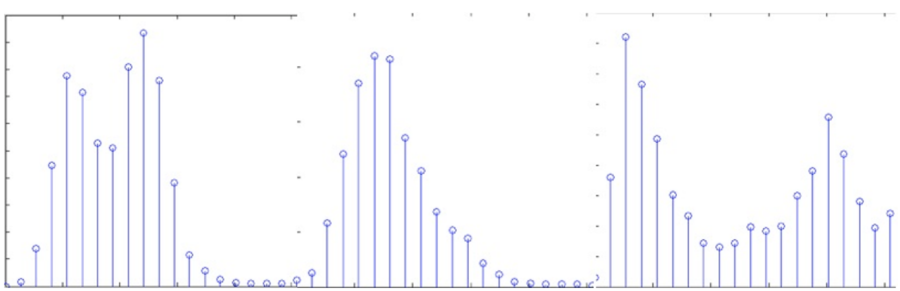
- Create a histogram for each colour channel and combine the histograms into one composite histogram.

- As a result
  - All colour have equal importance.
  - When comparing histograms of two images, we need to compare three instead of one.

Histograms of Color Channels

Red Green Blue





The profile of the composite histogram

20 bins for each colour:

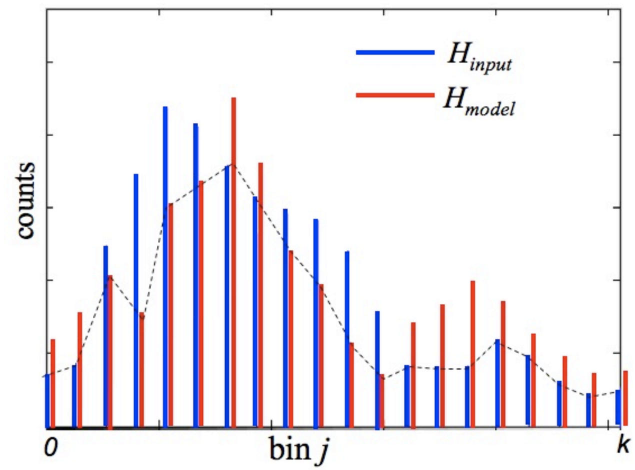
0
13.4211
26.8421
40.2632
53.6842
67.1053
80.5263
93.9474
107.3684
120.7895
134.2105
147.6316
161.0526
174.4737
187.8947
201.3158
214.7368
228.1579
241.5789
255.0000

Compare Histograms

- Comparing histograms is about comparing their profiles (i.e., bin-wise comparison).
- The similarity (or dissimilarity) of two histograms can be quantified by:

  - Intersection of the histograms

$$intersection(H_{input}, H_{model}) = \sum_{j=1}^{k} \min\{H_{input}[j], H_{model}[j]\}$$



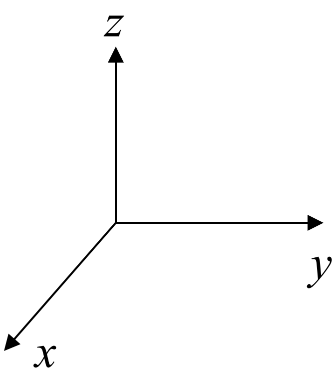
  - Match of one histogram with the other

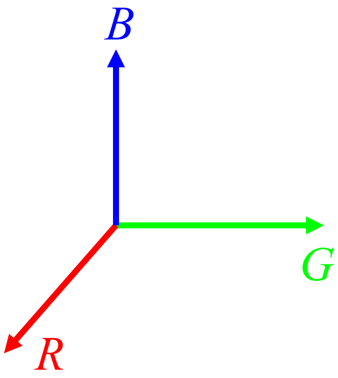
$$match(H_{input}, H_{model}) = \frac{\sum_{j=1}^{k} \min\{H_{input}[j], H_{model}[j]\}}{\sum_{j=1}^{k} H_{model}[j]} = \frac{intersection}{\sum_{j=1}^{k} H_{model}[j]}$$

The value of match varies from 0 (no similarity) to 1 (exactly the same)

Distance-Based Approaches

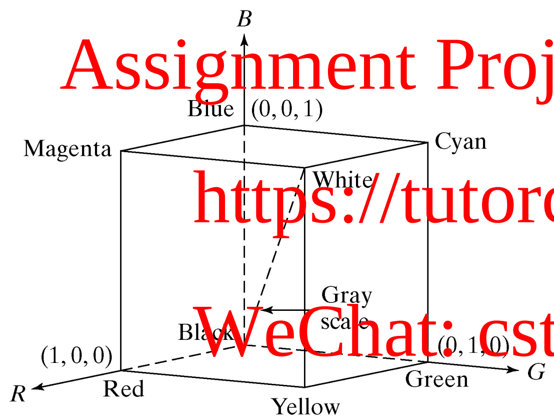
- Calculating the "distance" between the colours of two images.
- To do so, we need to:
  - define a coordinate system using R, G, B as three mutually orthogonal axes. This coordinate system is the so-called RGB Colour Space.
  - treat a colour as a point in this space,
  - calculate the distance between two points as we do in normal 3D Cartesian coordinate system.

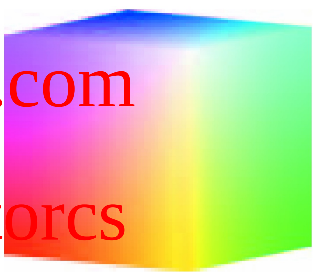




RGB Space and Colour Coordinates

- In the RGB colour space, a colour can be viewed as a point having coordinates $c(r,g,b)$





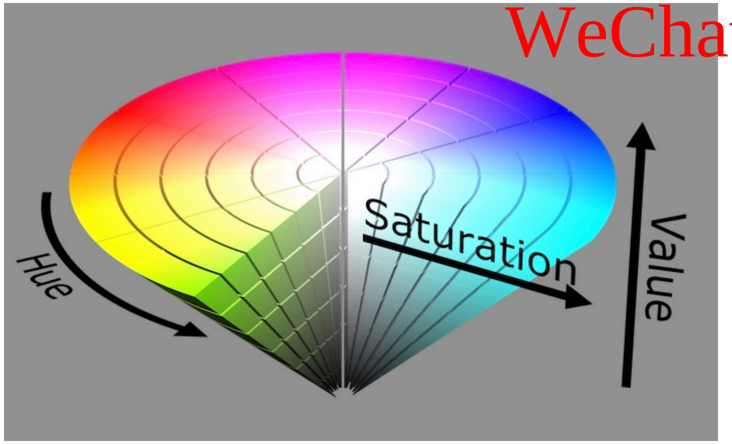
All colours in the space form a *colour cube*

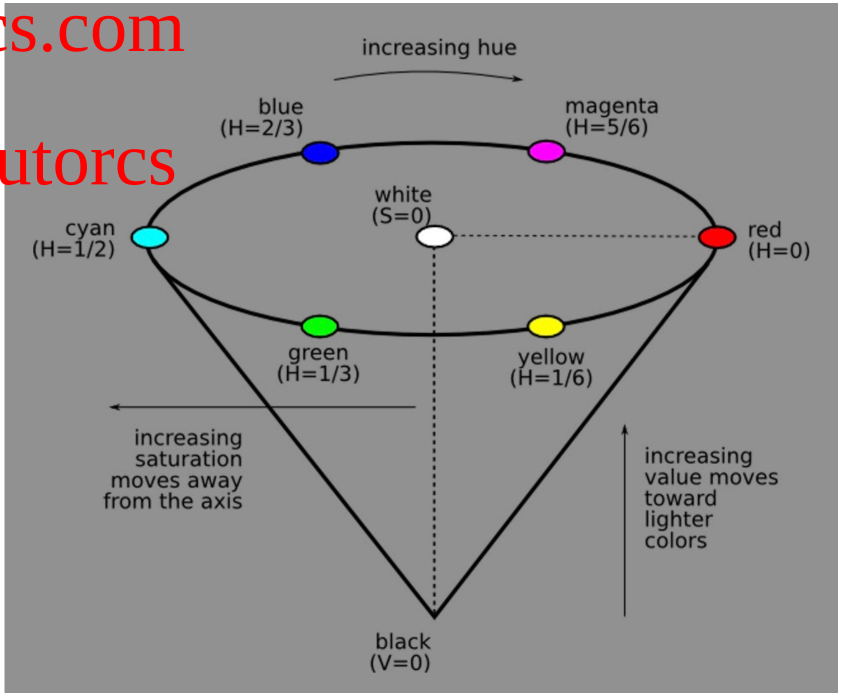
- The coordinates can also be expressed as a colour vector:

$$\mathbf{c} = \begin{bmatrix} r \\ g \\ b \end{bmatrix}$$

Other Colour Spaces

- In computer vision, other colour spaces are also used, e.g., HSV colour space.
- In HSV space, a colour is define by its hue, saturation and (ligthness) value.





Other Colour Spaces

- HSV space defines colour in a way that is more similar to how human perceives colours.
- In this space, the images of H and S channel are less dependent on the lightness (V channel). Therefore HSV images are used to remove/reduce the effect of uneven illumination (by ignoring the V channel).
- You can convert RGB image to and from HSV images using Matlab functions `rgb2hsv` and `hsv2rgb`.

Colour Classification

- In practical applications, we normally need to classify the pixels of an image according to the similarity of its colours to that of a sample or reference image(s).
- For example, we might use the colours of human skins to segment/isolate all pixels belonging to human faces in images.
- About the samples
  - Predefined or specified by the user in form of a small image region (e.g., by drawing a polygon around the sample pixels).
  - Usually have very similar but NOT exactly the same colours, (i.e., a collection of skin pixels may have very similar colours, but the colours are never exactly the same).

Cont'd

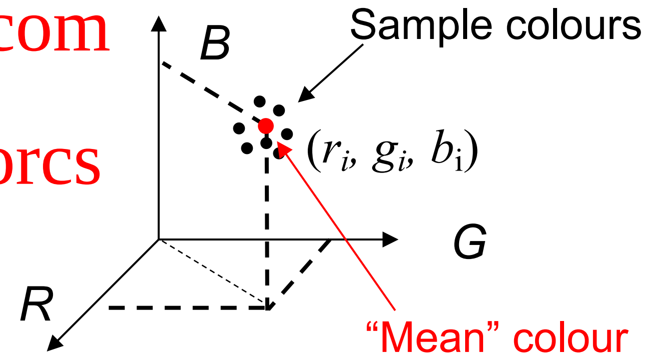
- We have two problems when evaluating the similarity between a pixel and the samples:

  - How can we define a "general colour" that can reasonably represents all the colours in the samples?
  - How do we measure the similarity between two colours?

The "Mean" Colour

- Given a set of sample points (pixels) $\mathbf{X}_i$, the *mean* (average) of colours of the sample pixels can be used as the sample (model) colour:

$$\mathbf{m} = \begin{bmatrix} m_r \\ m_g \\ m_b \end{bmatrix} = \frac{1}{K} \sum_{i=1}^{K} \begin{bmatrix} r_i \\ g_i \\ b_i \end{bmatrix}$$



- Mean colour is a reasonable representation of sample clours when the samples distribute as a spherical "lump" in the RGB colour space.

Similarity Measure

- One of the simplest measures is the Euclidean distance:
  - Let **z** be a point (a colour vector) in RGB space. We say **z** is similar to sample **m** if the distance between them is less than a specified threshold, t
  - The Euclidean distance between **z** and **m** is given by

$$D(\mathbf{z},\mathbf{m}) = \|\mathbf{z}-\mathbf{m}\|$$

where ||•|| is called the **norm** (or length) of **z-m**, which is defined as

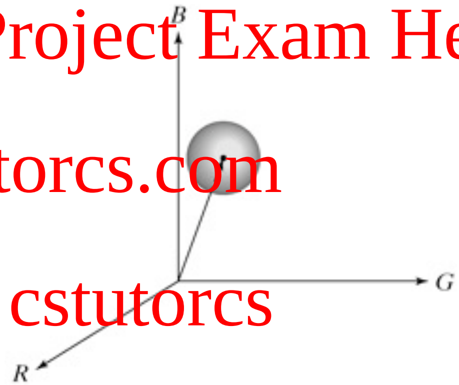
$$\|\mathbf{z}-\mathbf{m}\| = [(\mathbf{z}-\mathbf{m})^T(\mathbf{z}-\mathbf{m})]^{1/2} = \left(\begin{bmatrix} z_r - m_r & z_g - m_g & z_b - m_b \end{bmatrix} \begin{bmatrix} z_r - m_r \\ z_g - m_g \\ z_b - m_b \end{bmatrix} \right)^{\frac{1}{2}}$$

$$= \sqrt{(z_r - m_r)^2 + (z_g - m_g)^2 + (z_b - m_b)^2}$$

Colour Classification

- When Euclidean distance is used as similarity measure, all colours within the distance of **t** from **m** form a sphere.

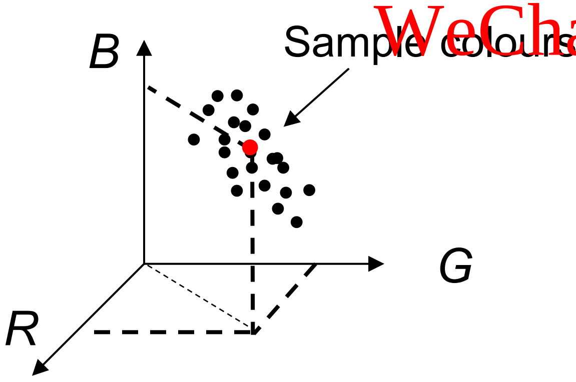
$$D(\mathbf{z},\mathbf{m}) \le t$$

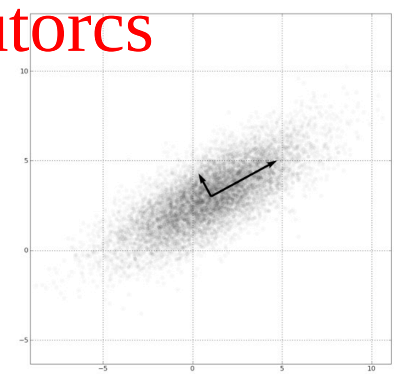


- This sphere separates the RGB colour space into two sets of points: the points inside and outside the sphere.
- Coding these two sets of points produces a binary image – the segmented image.

Limitations of "Mean" Colour

- A single mean colour becomes inadequate when the distribution of the samples cannot not be described as a spherical lump.
- For example, the distribution of the sample colours might form an ellipsoidal shape in R-G-B space:





Ellipsoidal sample distribution

In this case, instead of using a mean colour and the Euclidean distance, a new distance must be defined to account for the distribution of the samples.

A General Distance

- A more general (and useful) distance, *Mahalanobis distance*, is defined as:

$$D(\mathbf{z},\mathbf{m}) = \left[(\mathbf{z}-\mathbf{m})^T \mathbf{C}^{-1} (\mathbf{z}-\mathbf{m})\right]^{1/2}$$

where **z** is the colour of a pixel that one wish to compare with the samples; **m** is the sample mean, $\mathbf{C}$ is the 3x3 ***covariance matrix*** of the samples, with

$$\mathbf{C} = \frac{1}{K}\sum_{k=1}^{K}(\mathbf{x}_k - \mathbf{m})(\mathbf{x}_k - \mathbf{m})^T$$

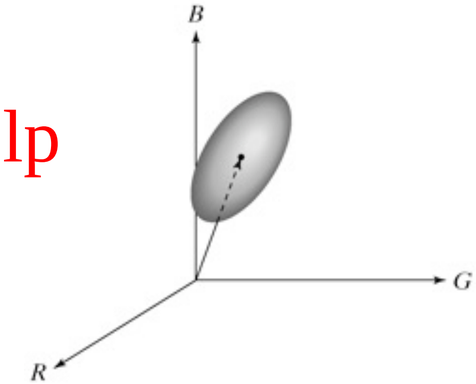
where x_k is the colour of a sample pixel.

Cont’d

- With the new distance, the classification criteria

$$D(\mathbf{z},\mathbf{m}) \le t$$

defines a solid ellipsoid with its principal axis pointing to the direction of maximum data spread.



- In Mahalanobis distance, if the covariance matrix becomes the identity matrix (i.e., $\mathbf{C}=\mathbf{I}$, when the sample distribution is spherical), the Mahalanobis reduces to Euclidean distance.

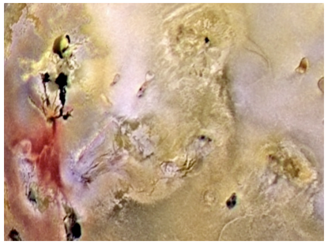
Matlab Implementation

- Select sample pixels interactively
  - roipoly(i): create a polygon that encloses a Region of Interest (ROI).

- Calculate sample mean and covariance matrix
  - covmatrix(sample_image)
- Image segmentation
  - Use Euclidean distance:

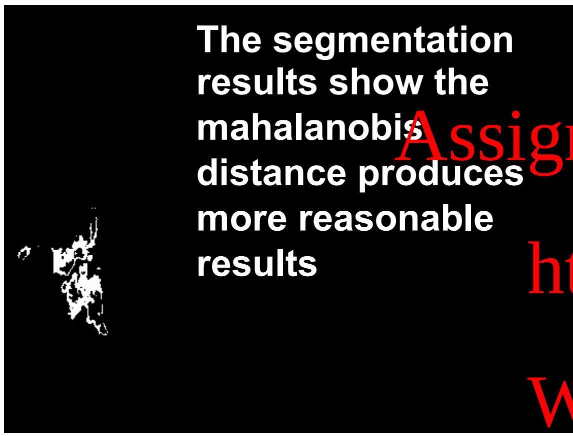
    ```
    Colorseg('euclidean', image, thresh,
    sample_mean)
    ```
  - Use Mahalanobis distance:

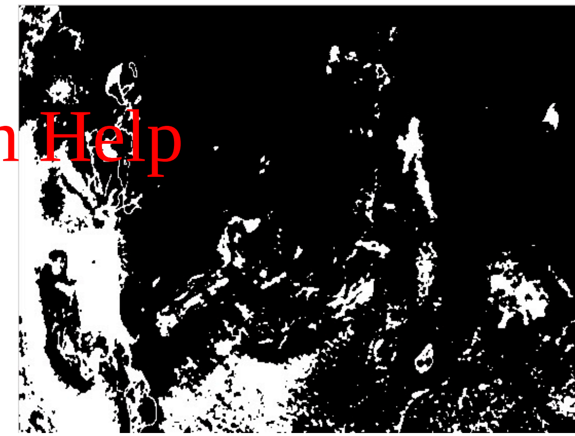
    ```
    Colorseg('mahalanobis', image, thresh,
    sample_mean, sample_covariance)
    ```



The image on the left is the surface of Jupiter's moon. The reddish materials are newly ejected from a volcano.

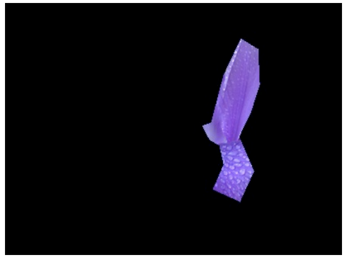
t=25 *t=50* *t=100*





Segmentation using *Euclidean* distance

Segmentation using *Mahalanobis* distance



Standard deviation of sample pixels:

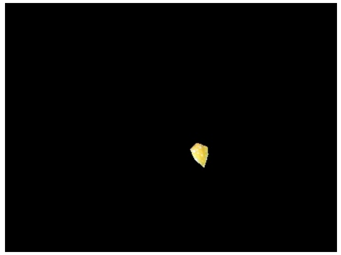
$$\sigma = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(x_k - \overline{m})^2}$$

R: 15.7298
G: 25.7281
B: 20.7891

t=25 *t=50* *t=100*

Segmentation using *Euclidean* distance

Segmentation using *Mahalanobis* distance



However, in this case, Euclidean distance gives better results

$t=25$ $t=50$ $t=100$

Segmentation using *Euclidean* distance

Segmentation using *Mahalanobis* distance

Readings

- Shapiro, L.G., Stockman, G.C., Computer Vision, Prentice-Hall, 2001, ISBN 0-13-030796-3
  - Section 6.1~ 6.5