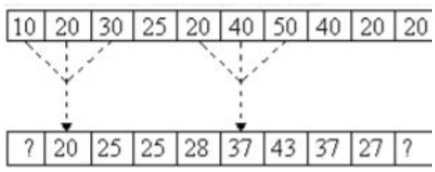
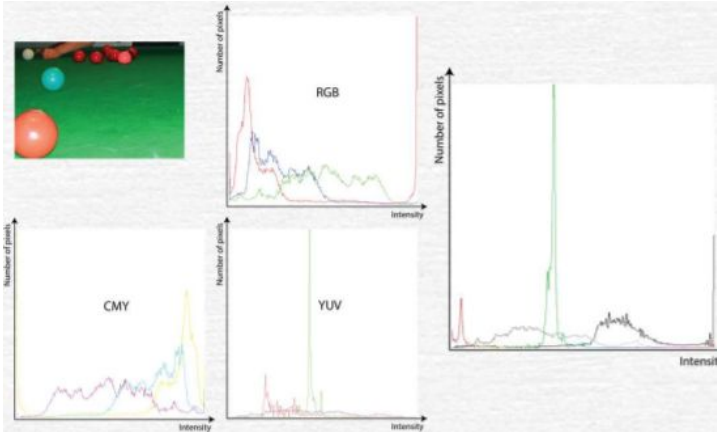
Histograms

- **1D Histograms**
    - In the case of greyscale - 256 greyscale intensities
        - Initialise histogram as all 0
        - For every pixel in the image, calculate the intensity, and increment the counter for that intensity
        - The histogram records the number of pixels per intensity
    - Uses
        - **Global information**
            - Histograms represent global information about an image - independent of the orientation of objects within the scene for example
        - **Can be used to classify**
            - Histograms, or information derived from histograms (avg intensity), can be used for classification - e.g. apple's with bruises will have a darker average intensity
        - **But..**
            - Not unique
            - Images can be very different and have very similar/the same histograms
    - Local minima and maxima are useful
        - But histograms are spikey - many local min/max
        - **How to deal with noise?**
            - **Smoothing**
                - Create a new array of values, where each value is the average of its neighbours in the original
                - 

                - What to do with first and last values? No right answer..
                    - Discard them?
                    - Replace with a constant?
                    - Create a wraparound?
    - **Colour histograms**
        - Calculate a histogram per each channel
        - Colour space has a huge impact in the usefulness of the histogram

- ■ 

- **3D Histograms**
    - **Channels aren't independent**
    - Better discrimination comes from considering all channels simultaneously
    - **8 bits per channel -> 16.8 million cells**
        - ■ Far too many, cannot use 16.8 million cells as a summary... to reduce this we shorten the number of bits in each channel
        - ■ **2 bits per channel gives us 64 cells**

- **Equalisation**
    - If an image has insufficient contrast, becomes difficult for human to interpret
        - ■ Can distinguish between 700 and 900 shades of grey under optimal conditions
            - In Brighter/Darker sections of the image this is more difficult
    - **If we improve the distribution of greyscale values in an image, we further the possible human interpretation**
        - ■ Equalisation
            - Attempts to make a flat histogram of greyscale values
            - Not really feasible, so becomes resulting histogram has greyscale values with zero associated pixels interspersed with high values
            - **When done on a colour image, we only change the luminance values**
            -

```
// Create a lookup table to map the luminances
// h[x] = histogram of luminance values image f(i,j).
pixels_so_far = 0
num_pixels = image.rows * image.cols
output = 0
for  input = 0 to 255
  pixels_so_far = pixels_so_far + h[ input ]
  new_output = (pixels_so_far*256) / (num_pixels+1)
  LUT[ input ] = (output+1+new_output) / 2
  output = new_output
// Apply the lookup table LUT(x) to the image:
for  every pixel f(i,j)
  f'(i,j) = LUT[ f(i,j) ]
```

- **Histogram Comparison**
    - Common approaches to finding similar images…
        - Use metadata tags
        - Compare images
    - To compare images we need to compare colour distributions - this will require a metric
        - $D_{Correlation}(h_1, h_2) = \frac{\sum_i (h_1(i) - \overline{h_1})(h_2(i) - \overline{h_2})}{\sqrt{\sum_i (h_1(i) - \overline{h_1})^2 \sum_i (h_2(i) - \overline{h_2})^2}}$
        - $D_{Chi-Square}(h_1, h_2) = \sum_i \frac{(h_1(i) - h_2(i))^2}{(h_1(i) + h_2(i))}$
        - $D_{Intersection}(h_1, h_2) = \sum_i min(h_1(i), h_2(i))$
        - $D_{Bhattacharyya}(h_1, h_2) = \sqrt{1 - \frac{1}{\sqrt{\overline{h_1}.\overline{h_2}.N^2}} \sum_i \sqrt{h_1(i).h_2(i)}}$
        - where
            - N is the number of bins in the histograms,
            - $\overline{h_k} = \sum_i (h_k(i)) / N$
        - **Alternatively: Earth Mover's Distance**
            - Calculates the minimum cost of turning one distribution (**In this case, histograms**) into another as a way to compare images
            - 1D solution:
                - EMD(-1) = 0
                - EMD(i) = h_1(i) + EMD(i-1) - h_2(i)
                - Earth Mover's Distance = Σ_i | EMD(i)
            - EMD is more difficult to compute for colour images…

- **Back Projection**
    - A better approach to select colours, based on sampling
        - Obtain a representative sample set of the colours
        - Histogram the samples
        - Normalize the histogram - max = 1.0
        - Back project the normalized histogram onto an image f(i,j)
        - **Creates a probability image p(i,j) indicating the similarity between f(i,j) and the sample set.**
    - **Key considerations**
        - **Size of the histogram bins - especially when sample set is limited**
        - **Colour space**

- **K-Means Clustering**
    - Large range of colour values is a problem (16.8 million in 8 bit colour space)
    - Need to reduce this number regularly
        - E.g. trying to represent the colour of someone's clothes, or compressing images
    - K-Means clustering as a way of reducing the variation in 3D colour space

- **Searching for k exemplars (specific colours) to best represent the image, with k specified in advance**
    - The colour of each pixel in an image is a **pattern** and a group of patterns associated with an exemplar is a **cluster**
- **Algorithm**
    - Get starting exemplars
        - Random / Choose first k patterns / distribute evenly
            - **Random assignment introduces non-determinism**
    - First pass
        - For all patterns in the image, allocate each one to the nearest exemplar
        - Recompute the exemplar as the centre of gravity of all the associated patterns after a new pattern is assigned to it
    - Second pass
        - Use final exemplars from first pass to reallocate all patterns to the exemplars
        - This might not change anything, but the exemplars will stay the same regardless
- How do we know how many exemplars to use?
    - Fewer may reflect large features better (e.g. felt on a snooker table), but more may capture some important detail (e.g. the balls on a snooker table)
    - **One approach is the Davies-Bouldin index:**

        Definition [edit]

        Let $R_{i,j}$ be a measure of how good the clustering scheme is. This measure, by definition has to account for $M_{i,j}$ the separation between the i^{th} and the j^{th} cluster, which ideally has to be as large as possible, and S_i, the within cluster scatter for cluster i, which has to be as low as possible. Hence the Davies–Bouldin index is defined as the ratio of S_i and $M_{i,j}$ such that these properties are conserved:

        1. $R_{i,j} \geqslant 0$.
        2. $R_{i,j} = R_{j,i}$.
        3. When $S_j \geqslant S_k$ and $M_{i,j} = M_{i,k}$ then $R_{i,j} > R_{i,k}$.
        4. When $S_j = S_k$ and $M_{i,j} \leqslant M_{i,k}$ then $R_{i,j} > R_{i,k}$.

        With this formulation, the lower the value, the better the separation of the clusters and the 'tightness' inside the clusters.

        A solution that satisfies these properties is:

        $$R_{i,j} = \frac{S_i + S_j}{M_{i,j}}$$

        This is used to define D_i:

        $$D_i \equiv \max_{j \neq i} R_{i,j}$$

        If N is the number of clusters:

        $$DB \equiv \frac{1}{N} \sum_{i=1}^{N} D_i$$

        -
        - **Short Comings:** No consideration for cluster size, so performs badly in situations with large clusters and small clusters