Images
Information Systems M

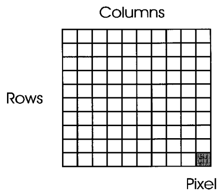
Prof. Paolo Ciaccia
http://www-db.deis.unibo.it/courses/SI-M/

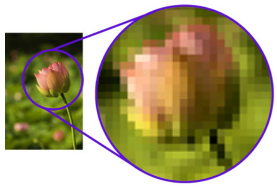
One image is worth 1,000 words…

- Undoubtedly, images are the most wide-spread MM data type, second only to text data
- Thus, it's not surprising that most efforts related to the management of MM data have concentrated on images, in particular:
  - Automatic extraction of features
  - Similarity measures
  - Indexing
  - …
- In the following we will provide basic information on the basic features of images

Image representation (1)

- Physically speaking a digital image represents a 2-D array of samples, where each sample is called pixel





- The word pixel is derived from the two words “picture” and “element” and refers to the smallest element in an image
- Color depth is the number of bits used to represent the color of a single pixel in a bitmapped image or video frame buffer (also known as bits per pixel – bpp)
  - Higher color depth gives a broader range of distinct colors

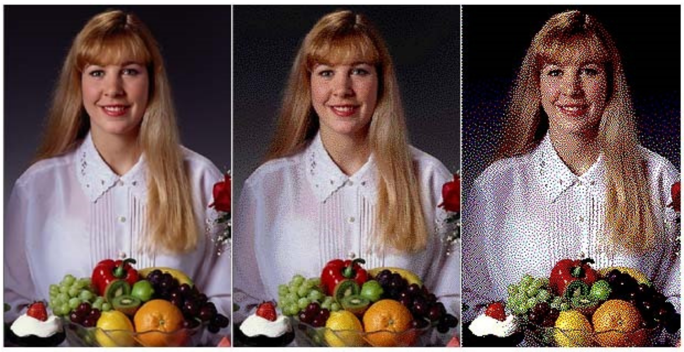
Image representation (2)

- According to the color depth, images can be classified into:
  - Binary images: 1 bpp (2 colors), e.g, black white photographic
  - Computer graphics: 4 bpp (16 colors), e.g., icon
  - Grayscale images: 8 bpp (256 colors)
  - Color images: 16 bpp, 24 bpp or more, e.g., color photography
- The table shows the color depths used in PCs today:

Color depth	# displayed colors	Bytes of storage per pixel	Common name
4-bit	16	0.5	Standard VGA
8-bit	256	1.0	256-Color Mode
16-bit	65.536	2.0	True Color
24-bit	16.777.216	3.0	High Color

- Dimension is the number of pixels in an image; identified by the *width* and *height* of the image as well as the *total number of pixels* in the image (e.g., an image 2048 wide and 1536 high (2048 x 1536) contains 3,145,728 pixels - 3.1 Mp)
- Spatial resolution is the *number of pixels per inch – bpi*; the higher the bpi, the better the resolution (clarity) of the image. Resolution changes according to the size at which the image is being reproduced
- Size [Byte] = *(width * height) * color depth/8*

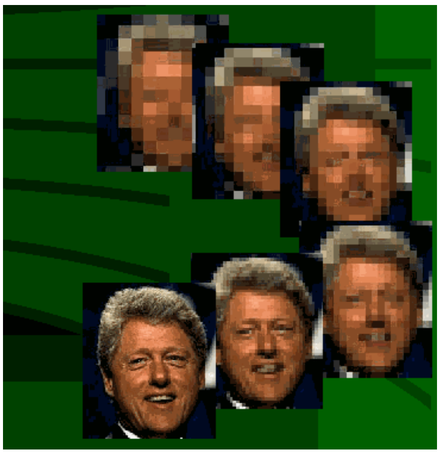
Color depth



16.7 Million Colors | 256 Colors | 16 Colors

Spatial resolution

Example: these images of Former President Clinton demonstrate the effects of different spatial resolutions
Each higher level of resolution allows you to distinguish more detail

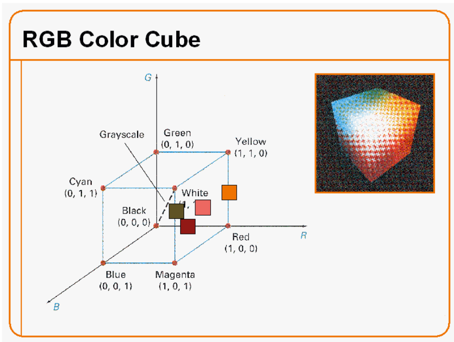


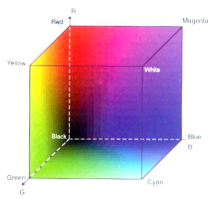
Color

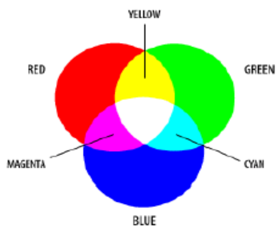
- According to the tri-chromatic theory, the sensation of color is due to the stimulation of 3 different types of receptors (*cones*) in the eyes
  - Each color has a wavelength, in the range 400÷700 nanometers (10^{-9} meters)
- Consequently, each color can be obtained as the combination of 3 component values (one per receptor type)
- A *color space* defines 3 color channels and how values from such channels have to be combined in order to obtain a given color
- There is a large variety of color spaces (e.g, RGB, CMY, XYZ, HSV, HSI, HLS, Lab, UVW, YUV, YCrCb, Luv, $L^* u^* v^*$), each designed for specific purposes, such as displaying (RGB), printing (CMY), compression (YIQ), recognition (HSV), etc.
- It is important to understand that a certain "distance" value in a color space does not directly correspond to an equal difference in colors' perception
  - E.g., distance in the RGB space badly matches human's perception

Color spaces: RGB

- The RGB space is a 3-D cube with coordinates Red,Green, and Blue
- The line of equation R=G=B corresponds to gray levels
- It can represent only a small range of potentially perceivable colors

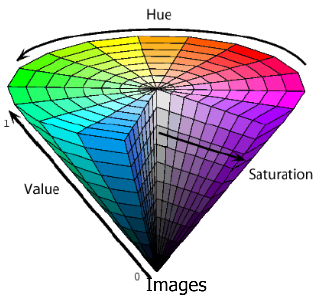


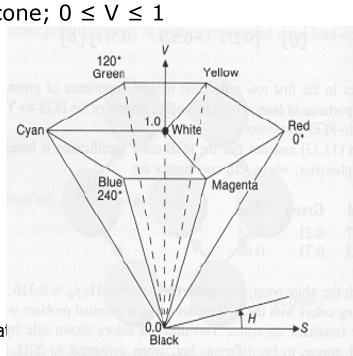




Color spaces: HSV

- The HSV space is a 3-D cone with coordinates Hue,Saturation, and Value:
- Hue is the “color”, as described by a wavelength
  - Hue is the angle around the circle or the regular hexagon; $0 \leq H \leq 360$
- Saturation is the amount of color that is present (e.g., red vs. pink)
  - Saturation is the distance from the center; $0 \leq S \leq 1$
    - The axis S = 0 corresponds to gray levels
- Value is the amount of light (intensity, brightness)
  - Value is the position along the axis of the cone; $0 \leq V \leq 1$





Saturation of colors

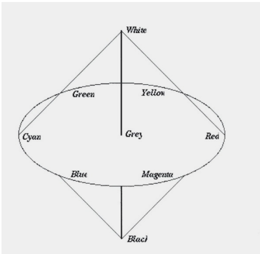
Original image

Saturation decreased by 20%

Saturation increased by 40%

What the 3 channels represent

- The figure contrasts the information carried out by each channel of the RGB and HSI color spaces
  - HSI: similar to HSV, the color space is a "bi-cone"



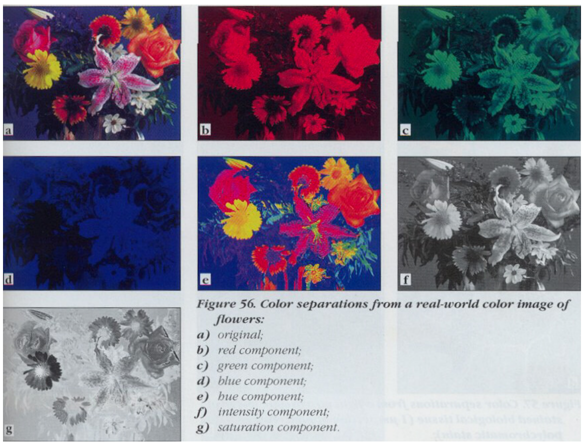


Figure 56. Color separations from a real-world color image of flowers:
a) original;
b) red component;
c) green component;
d) blue component;
e) hue component;
f) intensity component;
g) saturation component.

Color spaces: from RGB to HSV

- The conversion from RGB to HSV values is based on the following equations:

$$H = \cos^{-1} \frac{[(R-B)+(R-G)]/2}{[(R-G)^2+(R-B)(G-B)]^{1/2}}$$
$$S = 1 - 3 \times \min\{R,G,B\}/(R+G+B)$$
$$V = (R+G+B)/3$$

- HSV is much more suitable than RGB to support similarity search, since it better preserves perceptual distances

Representing color

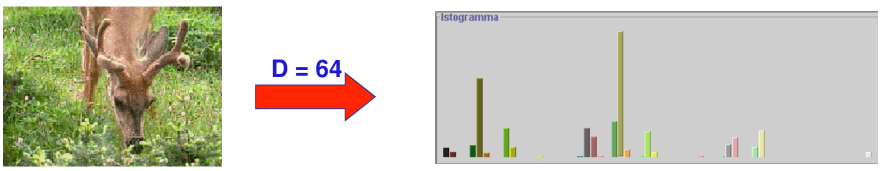
- In a digital image, the color space that encodes the color content of each pixel of the image is necessarily discretized
  - This depends on how many bits per pixel (bpp) are used

  Example:
  - if one represents images in the RGB space by using $8 \times 3 = 24$ bpp, the number of possible distinct colors is $2^{24} = 16{,}777{,}216$
  - With 8 bits per channel, we have 256 possible values on each channel
- Although discrete, the possible color values are still too many if one wants to compactly represent the color content of an image
  - This also aims at achieving some robustness in the matching process (e.g., the two RGB values (123,078,226) and (121,080,230) are almost indistinguishable)
- In practice, a common approach to represent color is to make use of histograms…

Color histograms

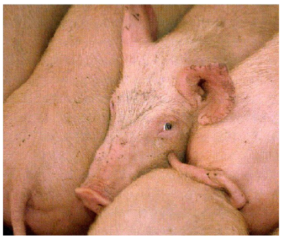
- A color histogram h is a D-dimensional vector, which is obtained by quantizing the color space into D distinct color regions
  - Typical values of D are 32, 64, 256, 1024, …

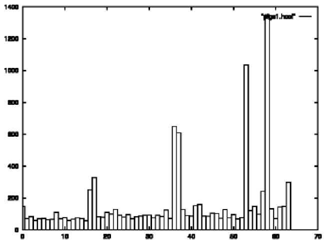
  Example: the HSV color space can be quantized into D=32 colors:
  H is divided into 8 intervals, and S into 4.
  V = 0 guarantees invariance to light intensity
- The i-th component (also called *bin*) of h stores the percentage (number) of pixels in the image whose color is mapped to the i-th color
- Although conceptually simple, color histograms are widely used since they are relatively invariant to translation, rotation, scale changes and partial occlusions

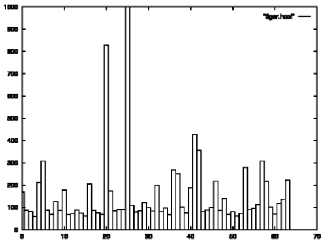


Examples of color histograms

- Two D=64 color histograms



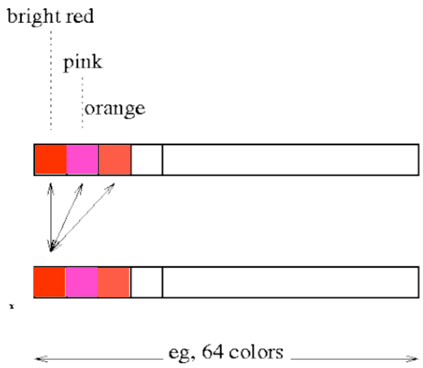




Comparing color histograms

- Since histograms are vectors, we can use any Lp-norm to measure the distance (dissimilarity) of two color histograms
- However, Lp-norms do not take into account colors' correlation (similarity)
  - Depending on the query and the dataset, we might therefore obtain low-quality results
  - Weighted Lp-norms and relevance feedback can partially alleviate the problem…

> - The problem is that Lp-norms just consider the difference of corresponding bins, i.e., they perform a 1-1 comparison
> - With color histograms, our "coordinates" are not unrelated ("cross-talk" effect)

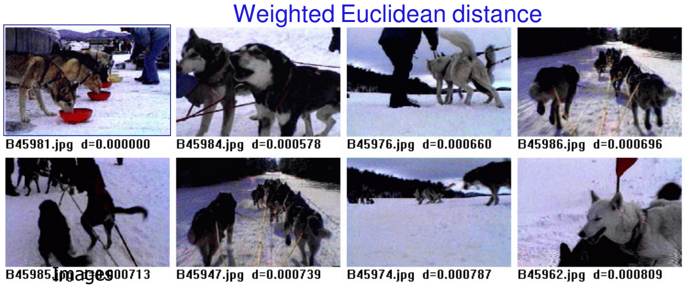


Sample queries based on color (1)

32-D HSV histograms

QueryImage — Euclidean distance

Weighted Euclidean distance



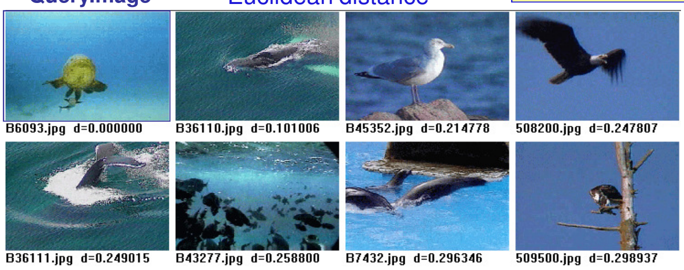
Images

17

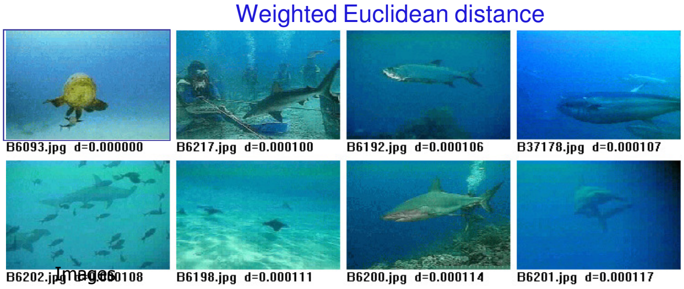
Sample queries based on color (2)

32-D HSV histograms

QueryImage — Euclidean distance



Weighted Euclidean distance



Images

18

Quadratic distance

- Consider two histograms h and q, both with D bins
- Their quadratic distance [FBF+94] is defined as:

$$L_A(h,q;A) = \sqrt{\sum_{i=1}^{D}\sum_{j=1}^{D} a_{i,j}(h_i - q_i)(h_j - q_j)}$$
$$= \sqrt{(h-q)^T \times A \times (h-q)}$$

where $A = \{a_{i,j}\}$ is called the (color-)similarity matrix

- The value of $a_{i,j}$ is the “similarity” of the i-th and the j-th colors ($a_{i,i} = 1$)
- Note that
  - when A is a diagonal matrix we are back to the weighted Euclidean distance,
  - when A = I (the identity matrix) we obtain the L_2 distance
- In order to guarantee that L_A is indeed a distance ($L_A(h,q;A) \geq 0\ \forall h,q$), it is sufficient that A is a symmetric positive definite matrix

Quadratic distance vs. Euclidean distance

- As a simple example, let D = 3, with colors red, orange, and blue
- Consider 3 pure-color images and the corresponding histograms:

h1=(1,0,0) **h2=(0,1,0)** **h3=(0,0,1)**

- Using L_2, the distance between two different images is always $\sqrt{2}$
- On the other hand, let the color-similarity matrix be defined as:

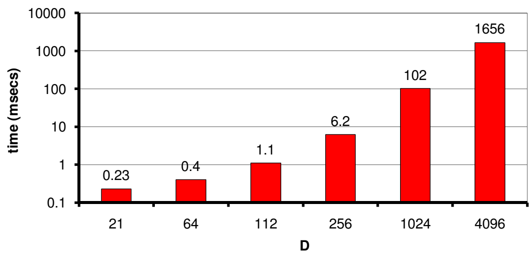
A	red	orange	blue
red	**1**	**0.8**	**0**
orange	**0.8**	**1**	**0**
blue	**0**	**0**	**1**

- Now we have $L_A(h1,h2) = \sqrt{0.4}$, whereas $L_A(h1,h3) = L_A(h2,h3) = \sqrt{2}$

Approximating the quadratic distance (1)

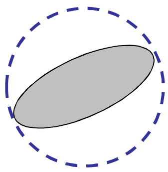
- From a geometric point of view, the quadratic distance defines iso-distance (hyper-)surfaces that are arbitrarily oriented (hyper-)ellipsoids

- Since computing the quadratic distance of two points (histograms) requires $O(D^2)$ time, for moderately large values of D the cost becomes prohibitive



Approximating the quadratic distance (2)

- Graphically, we can speed-up the computation of L_A by enclosing the query (hyper-)ellipsoid into a minimum bounding (hyper-)sphere



- Analytically, it can be proved that

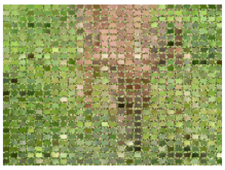
$$\boxed{L_2(h,q) \leq 1/\min_j\{\lambda_j\} \times L_A(h,q;A)}$$

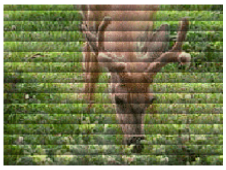
where the λ_j's are the eigenvalues of the matrix A

- Other possibilities to approximate L_A exist, which are based on dimensionality-reduction techniques applied to the indexed images [SK97]

Texture

- Unlike color, texture is not a property of the single pixel, rather it is a collective property of a pixel and its, suitably defined, "neighborhood"





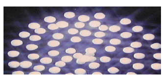
"mosaic" effect **"blinds" effect**

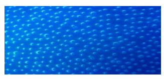
- Intuitively, texture provides information about the uniformity, granularity and regularity of the image surface
- It is usually computed just considering the gray-scale values of pixels (i.e., the V channel in HSV)

What texture measures

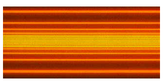
- A common model to define texture is based on the properties of coarseness, contrast e directionality:
  - Coarseness - coarse vs. fine: it provides information about the "granularity" of the pattern

  - Contrast - high vs. low contrast: it measures the amount of local changes in brightness





  - Directionality - directional vs. non-directional: it's a global property of the image

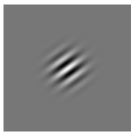


Texture extraction with Gabor filters

- A Gabor filter is a Gaussian modulated by a sinusoid, which can reveal the presence of a pattern along a certain direction and at a certain scale (frequency)

Scale: 3 at 72°

Scale: 4 at 108°



Scale: 5 at 144°

- To extract texture information, one chooses a number of directions/orientations (e.g.,6) and scales (e.g., 5) according to which the image has to be analyzed [MM96]
- For each orientation and scale, the average and the variance (standard deviation) of the filter output are computed
  - This leads to, say, 2×6×5 = 60-dimensional feature vectors, which are usually compared using the L_1 (Manhattan) distance
  - By the way, there is strong evidence that some cells in the primary visual cortex can be modeled by Gabor functions tuned to detect different orientations and scales…

Gabor filter

- Let I be an image, with I(x,y) being the gray-scale value of the pixel in position (x,y)
- A Gabor function is written as $$G(x,y) = \frac{1}{2\pi\sigma_x\sigma_y}\exp\left(-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y^2}\right)\right)\cos(2\pi\omega x)$$

  and is completely determined by its frequency (ω) and bandwidth (σ_x,σ_y)
- The Gabor filter $G_{m,n}(x,y)$ for scale m and orientation n is then defined as

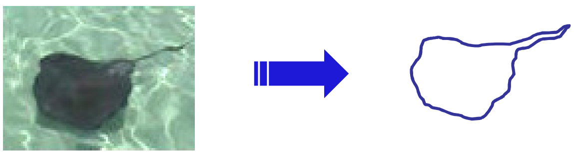
$$G_{m,n}(x,y) = a^{-m}G(x',y')$$
$$x' = a^{-m}(x\cos\theta_n + y\sin\theta_n),\ y' = a^{-m}(-x\sin\theta_n + y\cos\theta_n),\ \theta_n = n\pi/K$$

  where K is the total number of orientations
- Finally, the image is analyzed by convolution with the filter:

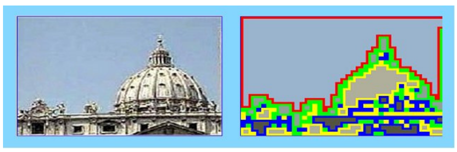
$$w_{m,n}(x,y) = \sum_i\sum_j G_{m,n}(x-i, y-j)I(i,j)$$

Shape

- Strictly speaking, an image has no relevant shape at all ☺
- When we talk about shape, we refer to that of the "object(s)" represented by the image
- Object recognition is a hard task, hardly solvable by any algorithm that operates in a general scenario (i.e., no knowledge about what to look for)
- In practice, shape information is often obtained by "segmenting" the image into a set of "regions", and then recovering the contours of such regions
  - …and segmentation is typically performed by analyzing color and texture information…



An example of segmentation



- A classical problem with segmentation is the trade-off between homogeneity of a region and number/significance of regions:

How many regions?

How "homogeneous" pixels within a same region should be?

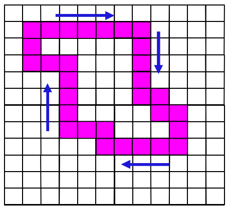
No general answer!

- In the limit cases: a single region(!?), each pixel is a region(!?)

Shape representation

- Once one has succeeded in extracting an object's contour, the next step is how to represent/encode it
- A common approach is to *navigate* the contour, which leads to an ordering of the pixels in the contour:

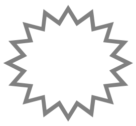
  { (x(t),y(t)) : t = 1…,M }

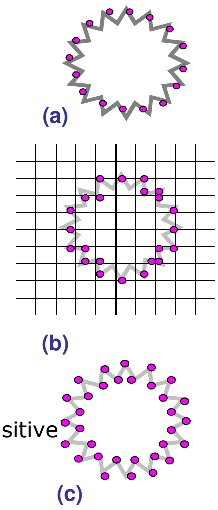


- A 2nd step is to represent the resulting curve in a parametric form
- For instance, a possibility is to resort to complex values, by setting z(t) = x(t)+ j y(t)
- Thus, now we have vectors of complex values…
- The problem is that each vector has a different length (i.e., M depends on the specific image)…

Representative points

- The idea is to keep only the D most "interesting" points
- Some methods are:
  - Equally-spaced sampling (a)
  - Grid-based sampling (b)
  - Maximum curvature points (c)
  - Fourier-based methods, which first compute the DFT of the contour, and then keep only the first D coefficents
- Working in the frequency domain has several advantages:
  - It can be proved that by properly modifying Fourier coefficients one can achieve invariance to scale, translation and rotation
  - Further, by viewing shape as a "signal", one can adopt distance measures that have been developed for the comparison of time series and that are somewhat insensitive to signals' modifications



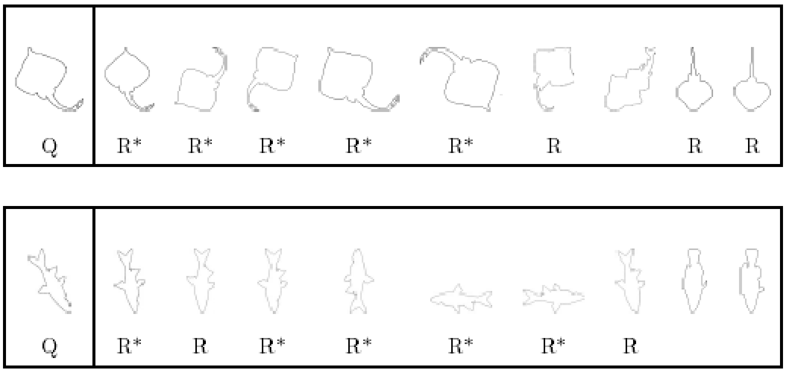


Sample queries based on shape ([BCP05])

R = relevant
(same type of fish)

1100 objects' contours

QueryImage



Final observations

- Effective and efficient image retrieval is not an easy task
- We have just scratched the surface of available techniques and ideas
- An impressive amount of work indeed exists, mainly originated in the pattern recognition area
  - Look at the [SWS+00] survey for detailed pointers
- Besides "generic" features, any specific image domain/application needs to extract and manage specific features, which in general require much more sophisticated tools than the one we have seen
  - E.g., face recognition
- Nonetheless, the problem of how to search in large image DB's remains (almost) the same