Intro to Media Computing

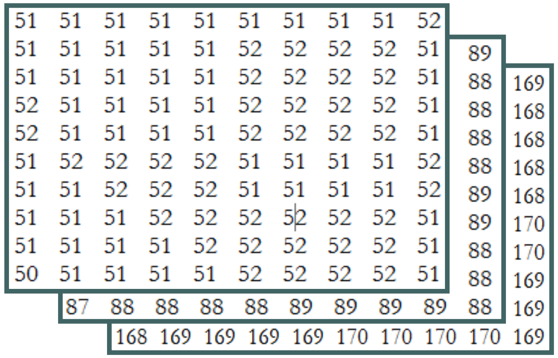
Lecture 3:
Image Content Analysis
and Search

Contents

- Color Representations
- Other Image Feature Extractions
- Similarity Measures and Matching
- Relevance Feedbacks
- Trends in Image/video Retrieval

How to represent images?

what we see



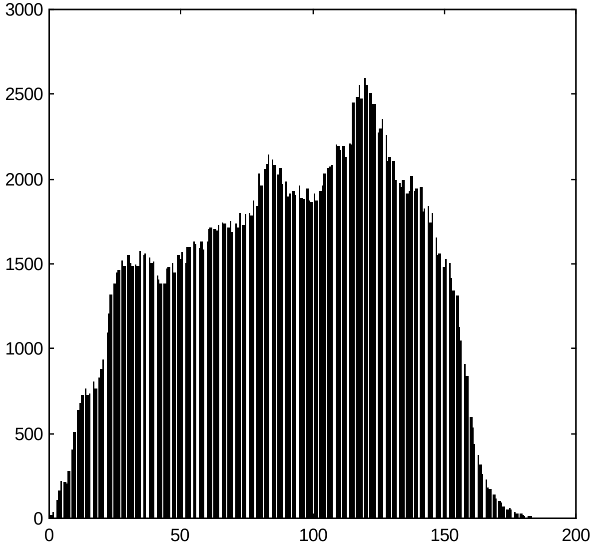
what computers see

Image Feature Extraction

- Simplest is as color histogram!!

Histogram Representation

- What is histogram?
  - The histogram function is defined over all possible intensity levels
  - For 8-bit representation, we have 256 levels or colors
  - For each intensity level, its value is equal to the number of the pixels with that intensity

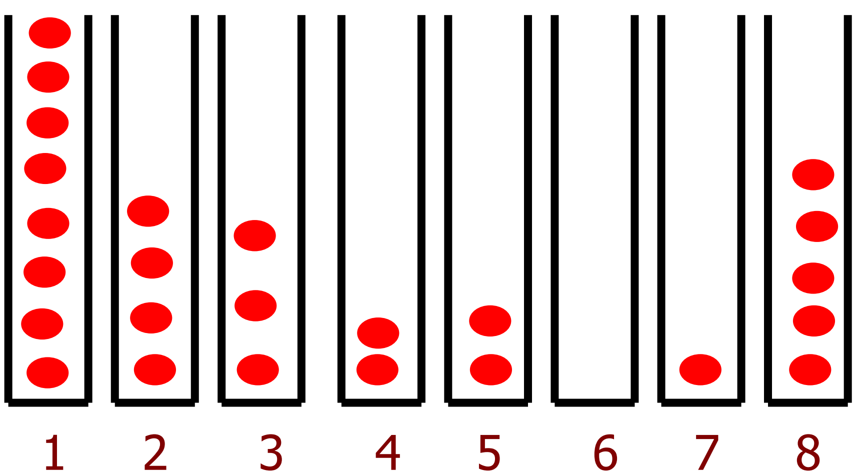


MATLAB function >imhist(x)

What is Histogram

- Example: Consider a 5x5 image with integer intensities in the range between of between 1 & 8, its histogram function $h(r_k)=n_k$ is:

1	8	4	3	4
1	1	1	7	8
8	8	3	3	1
2	2	1	5	2
1	1	8	5	2



Histogram Function:

$$h(r_1) = 8$$
$$h(r_2) = 4$$
$$h(r_3) = 3$$
$$h(r_4) = 3$$
$$h(r_5) = 2$$
$$h(r_6) = 0$$
$$h(r_7) = 1$$
$$h(r_8) = 5$$

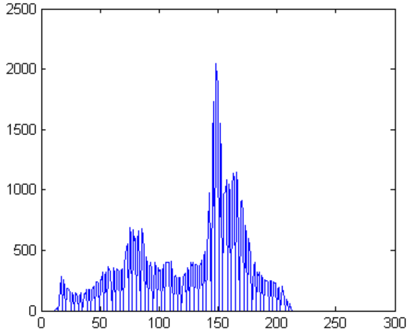
Normalized Histogram:

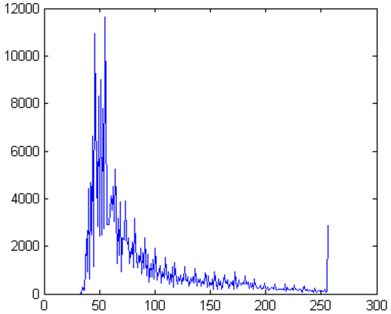
$$p(r_1) = 8/25 = 0.32$$
$$p(r_2) = 4/25 = 0.16$$
$$p(r_3) = 3/25 = 0.12$$
$$p(r_4) = 3/25 = 0.08$$
$$p(r_5) = 2/25 = 0.08$$
$$p(r_6) = 0/25 = 0.00$$
$$p(r_7) = 1/25 = 0.04$$
$$p(r_8) = 5/25 = 0.20$$

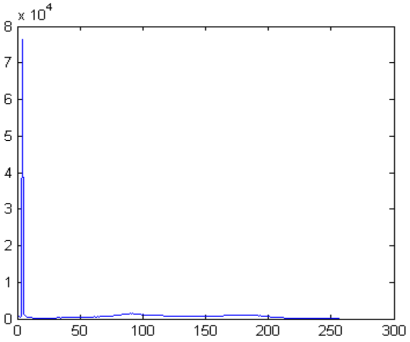
Examples of Image Histogram

Original image

Graph of the histogram function





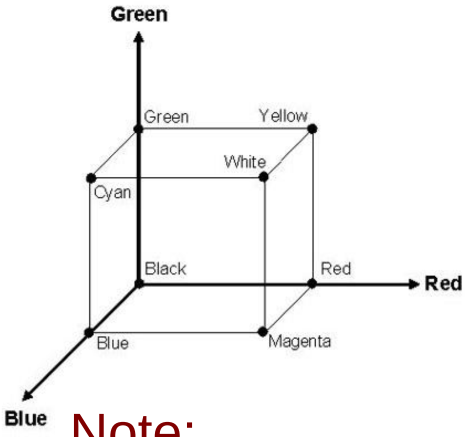


Observation:

- Image intensity is skewed (not fully utilizing the full range of intensities)
- What can be done??

Color Histogram -1

- Let image I be of dimension $p \times q$
  - For ease in representation, need to quantize $p \times q$ potential colors into m colors (for m << p x q)
  - For pixel p = (x,y)∈ I, the color of pixel is denoted by $I(p) = c_k$
- Construction of Color Histogram
  - Extract color value for each pixel in image
  - Quantize color value into one of m quantization levels
  - Collect frequency of color values in each quantization level



$$H[r,g,b] = \sum_p \sum_q \begin{cases} 1 & \text{if } I_R[p,q]=r, I_G[p,q]=g, I_B[p,q]=b \\ 0 & \text{otherwise} \end{cases}$$

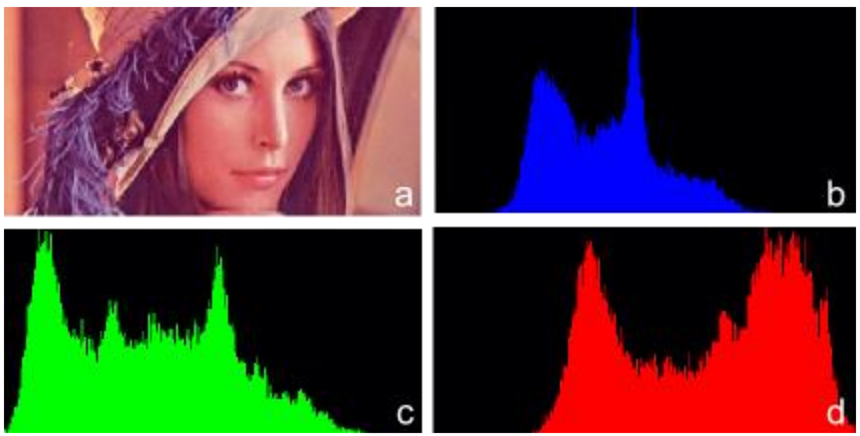
where each bin corresponds to a color in the quantized color space

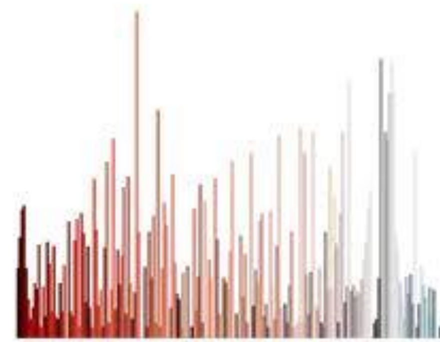
Note:

- Divide each color axis into n_r, n_g, n_b bins
- $m = n_r \times n_g \times n_b$

Color Histogram -2

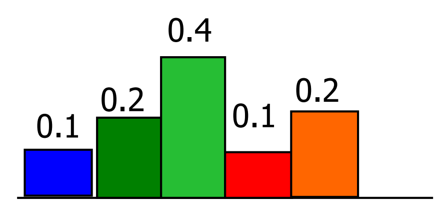
- Thus, image is represented as a color histogram H of size *m*
  - where H[i] gives # of pixels at intensity level I
- For example:





Into a single quantized histogram

- Normalize H to NH by dividing each entry by size of image p*q



Color Histogram -3

- Desirable properties of feature vector *f(I):*
  - |f(I)– f(I')| should be large iff I and I' are very different,
    f(.) should have property of monotonicity
  - f(.) should be fast to compute
  - f(I) should be small in dimension

- Color Histogram satisfies all these properties
  - But it has no spatial info
  - Not robust to large appearance changes

Color Representation -1

- Need some measurement of color differences
- RGB Color Space is used for display devices
  - Each color is represented as a triple (r_i, g_i, b_i)
  - But it is not designed for human, as it is perceptually non-linear
- Need to use perceptually linear color spaces
  - Luv, Lab, YUV, YC_rC_b
- In linear color space, say YC_rC_b, the difference between 2 colors, C_i and C_j, can be measured by (Euclidean) distance between them in the space:

$$Diff_{L_2}(i,j) = \sqrt{(Y_i - Y_j)^2 + (Cr_i - Cr_j)^2 + (Cb_i - Cb_j)^2}$$

and this corresponds well with human perception of color differences

Color Representation -2

- Describe color in terms of luminance & chrominance

- YUV Model:
  - Y: Luminance or Black-and-White component
  - U & V: Chrominance or color components
  - Basic color format used by the NTSC, PAL, SECOM
  - U and V subsampled to reduce bitrate

$$\begin{pmatrix} Y \\ U \\ V \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.515 & -0.100 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

Color Representation -3

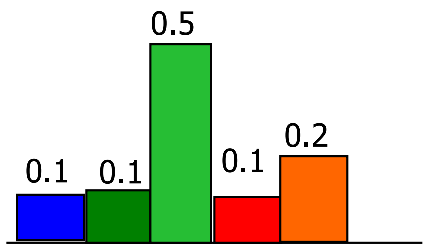
- YCrCb Model
  - Y: Luminance component
  - Cr & Cb: Chrominance or color components
  - Used in digital image/video compression standards

$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.16875 & -0.33126 & 0.5 \\ 0.5 & -0.41869 & 0.08131 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

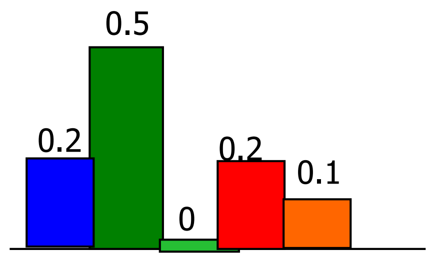
$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.0 & 0 & 1.402 \\ 1.0 & -0.34413 & -0.71414 \\ 1.0 & 1.772 & 0 \end{bmatrix} \begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix}$$

Metrics for Histogram Matching -1

- Given two images with histogram *Q and D*:



H(Q) =[0.1 0.1 0.5 0.1 0.2]



H(D) =[0.2 0.5 0.0 0.2 0.1]

- The difference between Q & D in L_1, or city block, dist is:

$$Diff_1[Q,D] = \sum_j | H(Q,j) - H(D,j) |$$

or the normalized version

$$NDiff_1[Q,D] = \sum_j [H(Q,j) * \frac{| H(Q,j) - H(D,j) |}{\max\{H(Q,j), H(D,j)\}}]$$

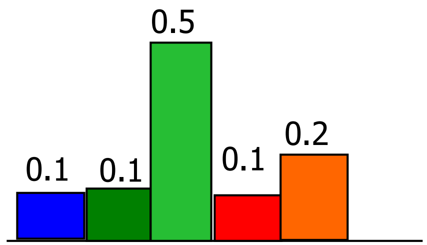
Diff1 = 0.1 + 0.4 + 0.5 + 0.1 + 0.1 = 1.2??

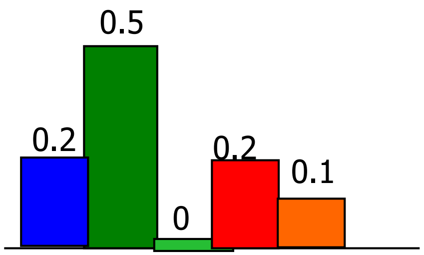
NDiff1 = 0.05 + 0.08 + 0.5 + 0.05 + 0.05 = 0.75 ??

Differences are larger than it is perceived!!

Metrics for Histogram Matching -2

- H(Q)=[0.1 0.1 0.5 0.1 0.2]   H(D) =[0.2 0.5 0.0 0.2 0.1]





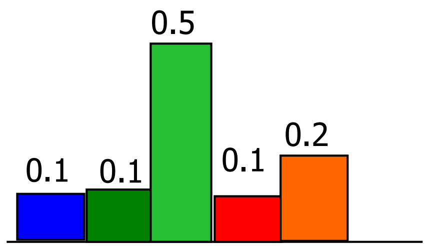
- It is more useful to consider similarity, rather than differences
- The normalized similarity between Q & D in L_1 is:

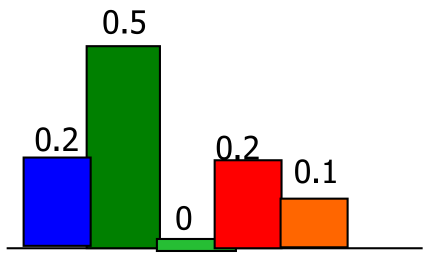
$$NSim_1[Q,D] = \sum_{j \in S_Q} [H(Q,j) * (1 - \frac{|H(Q,j) - H(D,j)|}{\max\{H(Q,j), H(D,j)\}})]$$

- $NSim_1$ (Q, D) = 0.05 + 0.02 + 0 + 0.05 + 0.05 = 0.17 ??
  Again, the similarity looks less than perceived!!

Metrics for Histogram Matching -3

- H(Q)=[0.1 0.1 0.5 0.1 0.2]   H(D) =[0.2 0.5 0.0 0.2 0.1]





- Why is similarity looks less than perceived!!
- In fact C_2 & C_3 (say S_{23}=0.8) are very similar, so are C_4 & C_5 (say S_{45}=0.7)

The actual similarity between H_1 and H_2 should consider this fact. Hence:

$$NSim_1(Q,D) = 0.05 + [0.02 + 0*0.8] + [0 + 0.02*0.8] + [0.05 + 0.05*0.7] + [0.05 + 0.05*0.7]$$

$$= 0.256 \quad \text{(instead of just 0.17)..}$$

Modeling Color Similarity -1

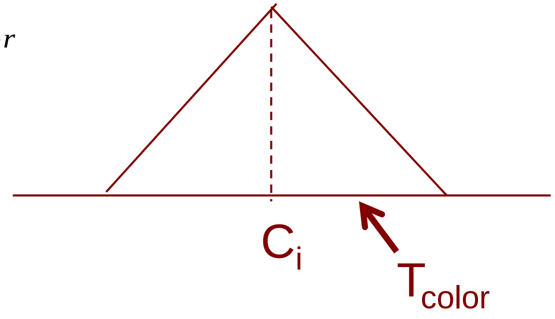
- In YC_rC_b, the difference between 2 colors, C_i & C_j, can be measured by the distance between them as:

$$Dist_{L_2}(i,j) = \sqrt{(Y_i - Y_j)^2 + (Cr_i - Cr_j)^2 + (Cb_i - Cb_j)^2}$$

& this corresponds well with human perception of color differences

- The similarity between two colors can be incorporated as:

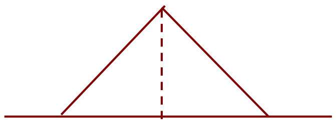
$$SIM(i,j) = \begin{cases} 0 & when\, Dist(i,j) > T_{color} \\ 1 - \dfrac{Dist(i,j)}{T_{color}} & otherwise \end{cases}$$



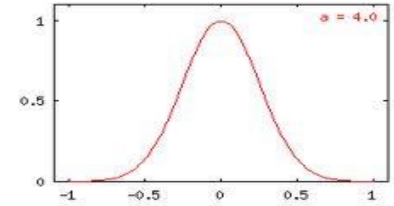
In practice T_{color} is small, say $T_{color} = 0.1$

Modeling Color Similarity -2

- In previous example, color similarity is approximated as hat function!!



- Other functions are possible, such as the Gaussian function:



C_i

- The perceptually similar color matrix S(i, j) is:

$$\mathbf{S} = \begin{bmatrix} 1 & s_{0,1} & \dots & s_{0,N} \\ s_{1,0} & 1 & \dots & \dots \\ s_{2,0} & \dots & \dots & s_{N-1,N} \\ s_{N,0} & \dots & s_{N,N-1} & 1 \end{bmatrix}$$

  - S(i,j) gives the similarity between colors i and j
  - This matrix is symmetrical and can be pre-computed

Metrics for Histogram Matching -4

- Recall the normalized difference between Q and D in L_1 distance for color *i* is:

$$NDiff_1(i) = H(Q,i) * \frac{|H(Q,i) - H(D,i)|}{\max\{H(Q,i), H(D,i)\}}$$

and its normalized similarity is:

$$NSim_1(i) = H(Q,i) * (1 - \frac{|H(Q,i) - H(D,i)|}{\max\{H(Q,i), H(D,i)\}})$$

- The overall similarity with perceptually similar colors is:

$$Sim_{1_{SimColor}}(Q,D) = \sum_i \sum_j NSim_1(i)\, S(i,j)\, NSim_1(j)$$

Color Moment

- Let the set of pixel be:
  $I = [p_1, p_2, \dots p_R]$, for a total of $R=(p \times q)$ pixels

- Represent color contents of image in terms of moments:

  1st Color moment (Mean): $\frac{1}{R}\sum_i X_i$

  2nd Color Moment about mean (Variance): $\frac{1}{R}\sum_i (X_i - \overline{X})^2$

- We can use these to model image contents
  - Advantages: Simple & efficient; Only one value for each representation
  - Disadvantage: Unable to model contents well
  - However, it can be effective at sub-image level, say sub-blocks

    HOW TO DO THIS??

Color Coherence Vector (CCV)

- Problems of color histogram rep^{n}
  - Easy to find 2 different images with identical color histogram
  - As it does not model local and location info

Exactly same color distribution & similar shape

- Need to take spatial info into consideration when utilizing colors:
  - Color Coherence Vector (CCV) representation
  - Color Correlogram representation

- CCV
  - A simple and elegant extension to color histogram
  - Not just count colors, but also check adjacency
  - Essentially form 2 color histograms – one where colors form sufficiently large regions, while the other for isolated colors

CCV Representation -2

- Example:
  - Define sufficiently large region as those > 5 pixels

2	1	2	2	1	1
2	2	1	2	1	1
2	1	3	2	1	1
2	2	2	1	3	3
2	2	1	1	3	3
2	2	1	1	3	3

→

2	1	2	2	1	1
2	2	1	2	1	1
2	1	3	2	1	1
2	2	2	1	3	3
2	2	1	1	3	3
2	2	1	1	3	3

→

Region	A	B	C	D	E
Color	2	1	3	1	3
Size	15	3	1	11	6

Color	1	2	3
Hα	11	15	6
Hβ	3	0	1

- Treats H_α and H_β separately

- Similarity measure:
  - Give higher weight to Hα, as it tends to correspond more to objects

$$Sim(Q, D) = \mu \, Sim(Q_\alpha, D_\alpha) + (1 - \mu) \, Sim(Q_\beta, D_\beta)$$

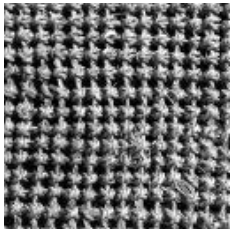
for $\mu > 0.5$

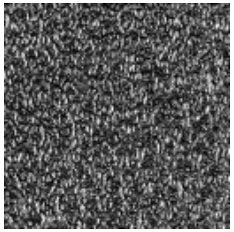
Contents

- Color Representations
- Other Image Feature Extractions
- Similarity Measures and Matching
- Relevance Feedbacks
- Trends in Image/video Retrieval

Texture Representation

- What is texture?
  - Something that repeats with variation
  - Must separate what repeats and what stays the same
  - Model as repeated trials of a random process


Fabric


Metal


Leaves


Flowers

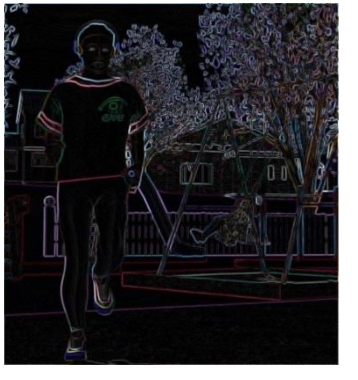
- Tamura representation: classifies textures based on psychology studies
  - Coarseness
  - Contrast
  - Directionality
  - Linelikeness
  - Regularity
  - Roughness
- Consider simple realization of Tamura features
  - May be simplified as distributions of edges or directions

Edge Representation -1

- Spatial Domain Edge-based texture histogram
  - To extract an edge-map for the image, the image is first converted to luminance Y (via $Y = 0.299R+0.587G+0.114B$)
  - A *Sobel edge operator* is applied to the Y -image by sliding the following 3×3 weighting matrices (*convolution masks*) over the image.

-1	0	1
-2	0	2
-1	0	1

1	2	1
0	0	0
-1	-2	-1



  - The edge magnitude D and the edge gradient ϕ are given by:

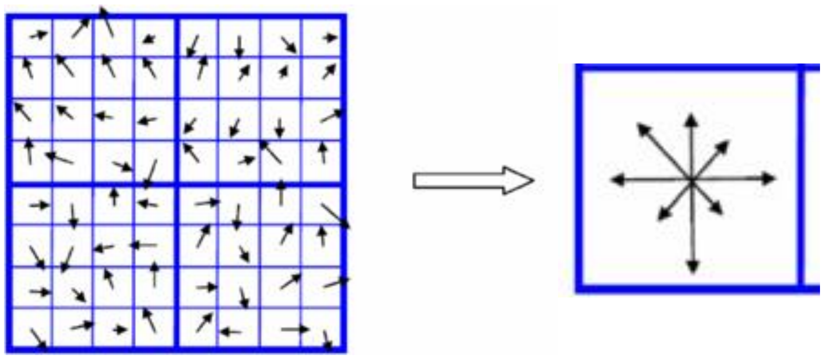
$$D = \sqrt{d_x^2 + d_y^2}, \quad \phi = \arctan \frac{d_y}{d_x}$$

Edge Representation -2

- Represent texture of image as 1 or 2 histograms:

Edge histogram

- Quantize the edge direction *Φ* into 8 directions:



- Setup *H(Φ)* (with 8 dimension)

Magnitude histogram

- Quantize the magnitude *D* into, say 16 values
- Setup *H(D)*, with 16 dimension.

- Edge Histogram is normally used

Segmented Image Representation

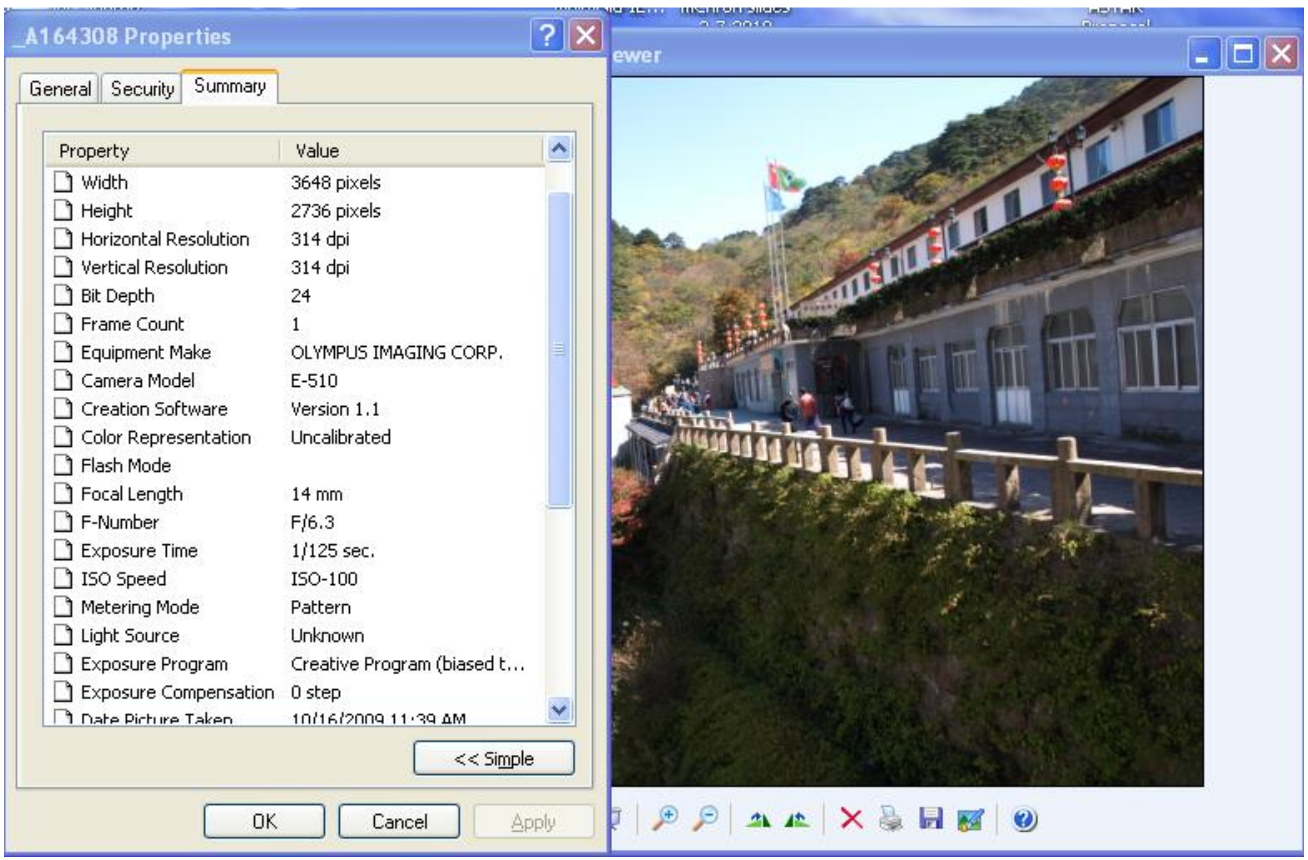
- Problems with global image representation – can't handle layout and object level matching very well
- One simple remedy: use segmented image (example, 4x4):

(1,1)	(1,2)	(1,3)	(1,4)
(2,1)	(2,2)	(2,3)	(2,4)
(3,1)	(3,2)	(3,3)	(3,4)
(4,1)	(4,2)	(4,3)	(4,4)

- Compute histograms for individual window
- Match at sub-window level between Q and D:
  - between corresponding sub-windows or
  - between all possible pairs of sub-windows
  - May give higher weights to central sub-windows
- Pros: able to capture some local information
- Cons: more expensive, may have mis-alignment problem

Metadata of Images

- Cameras store image metadata as "EXIF tags"
  - EXIF (**Exchangeable image file format**)
  - Timestamp, focal length, shutter speed, aperture, etc
  - Keywords can be embedded in images



Metadata of Images -2

- Other form of metadata: semantic tags (or concepts)
  - Supply manually by users
  - Reasonable thru social tagging

- With metadata, we can perform advanced analysis:
  - Use existing set of semantic tags
  - Automatic keyword generation (leveraging on EXIF info)
  - Camera knows *when* a picture was taken…
  - A GPS tracker knows *where* you were…
  - EXIF knows the conditions that picture was taken
  - Your calendar (or phone) knows *what* you were doing…

  - Combine these together into a list of keywords

Contents

- Color Representations
- Other Image Feature Extractions
- Similarity Measures and Matching
- Relevance Feedbacks
- Trends in Image/video Retrieval

Similarity Measure -1

How to measure the similarity between two images given the feature vectors?

Feature vectors:

$$F_1 = [\, f_{11}\ f_{12} \ldots f_{1n} \,] \qquad F_2 = [\, f_{21}\ f_{22} \ldots f_{2n} \,]$$

$$Dist(F_1, F_2) = ?$$

- Euclidean (L2) Distance
- Manhattan (L1) Distance
- Minkowski Distance
- Chebychev Distance
- Mahalanobis distance
- Cosine Similarity

- Similar to text measures

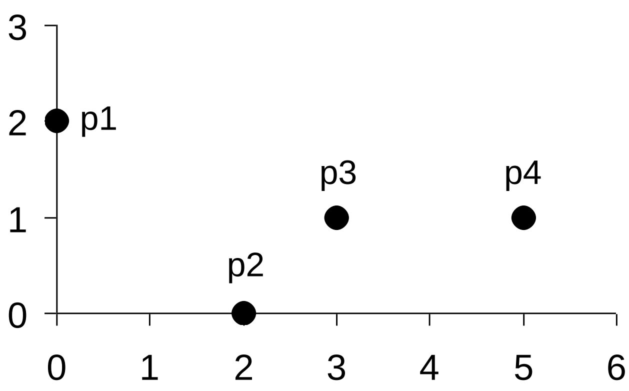
Similarity Measure -2

Euclidean (L2) Distance:

$$dist = \sqrt{\sum_{k=1}^{n} (f_{1k} - f_{2k})^2}$$

- The Euclidean Distance takes into account both the direction and the magnitude of the vectors
- More for correlated data

Examples:



point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

Distance Matrix:

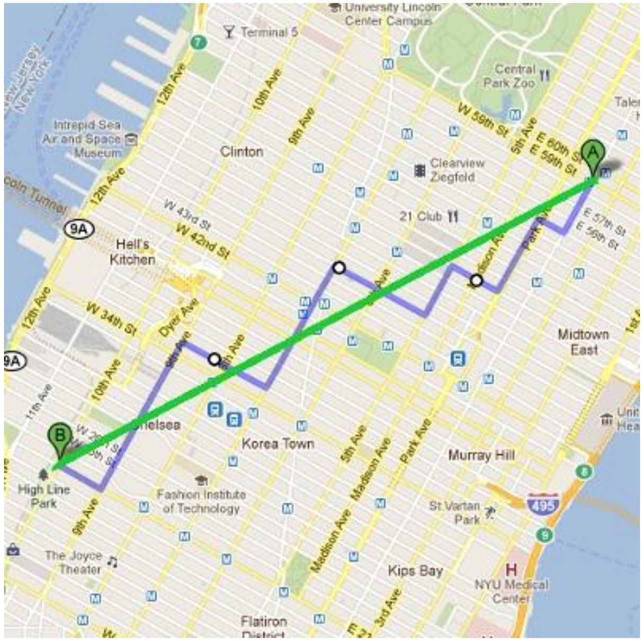
	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

Similarity Measure -3

Manhattan (L1) Distance

$$dist = \sum_{k=1}^{n} | f_{1k} - f_{2k} |$$

- Manhattan distance represents distance that is measured along directions that are parallel to all axes.
- More for uncorrelated data



- Green: Euclidean Distance
- Blue: Manhattan Distance

Similarity Measure -4

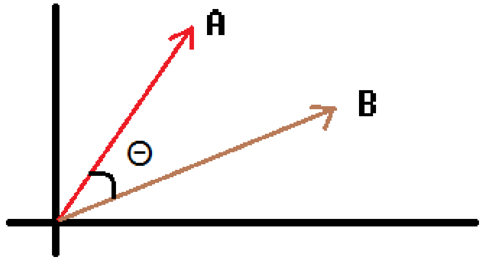
Mahalanobis Distance

$$dist = \sqrt{(F_1 - F_2)^T S^{-1} (F_1 - F_2)}$$

- Mahalanobis Distance takes into account the correlations of the data set and is scale-invariant.
- S is the distance (or color similarity) metric for Mahalanobis distance

Cosine Similarity

$$dist = \cos(\theta) = \frac{F_1 \cdot F_2}{\|F_1\| \cdot \|F_2\|}$$



- The Cosine Similarity takes into account only the angle and discards the magnitude.

Similarity Measure -5

Minkowski Distance

$$dist = \left\{ \sum_{k=1}^{n} \left| f_{1k} - f_{2k} \right|^{m} \right\}^{\frac{1}{m}}$$

- Minkowski distance is a generalization of Euclidean and Manhattan distance.

when m=1: Euclidean Distance
when m=2: Manhattan Distance

Chebychev Distance

$$dist = \max \left\{ \left| f_{1k} - f_{2k} \right| \right\}$$

- Chebychev distance simply picks the largest difference between any two corresponding coordinates.

Similarity Measure -6

- Histogram Intersection

$$Diff_{Int}[Q,D] = \frac{\sum_{j \in S_Q} \min\{H(Q,j), H(D,j)\}}{\min\{\| H(Q) \|, \| H(D) \|\}}$$

Similarity Measure -7

How to select the correct similarity measure?

Euclidean distance: The most popular distance.

Manhattan distance: between two items is the sum of the differences of their corresponding components.

Cosine distance (angle): Takes into consideration only the angle, not the magnitude.

Chebychev: Focuses on the most important differences.

Mahalanobis: Can warp the space in any convenient way. Usually, the space is warped using the correlation matrix of the data.

Contents

- Color Representations
- Other Image Feature Extractions
- Similarity Measures and Matching
- Relevance Feedbacks
- Trends in Image/video Retrieval

Relevance Feedback

- Performance of auto-retrieval is limited
  - Visual analysis not precise
  - Users' queries are ambiguous
- Develop interactive system
  - Users indicate which image is relevant to query
  - System update query for new retrieval

  - The histogram-based RF formula:

$$\underline{Q}^{(k+1)} = \underline{Q}^{(k)} + \alpha \sum \underline{R} - \beta \sum \underline{NR}$$

Contents

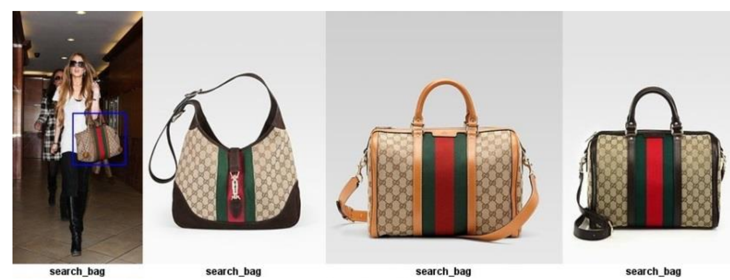
- Color Representations
- Other Image Feature Extractions
- Similarity Measures and Matching
- Relevance Feedbacks
- Trends in Image/video Retrieval

Current Trends

- Media search
  - Media Search 1.0 (uses text and visual features, with user interactions)
  - Media Search 1.2? (towards concept-based search)

    Needs to detect visual concepts in images, such as grass, tiger etc.

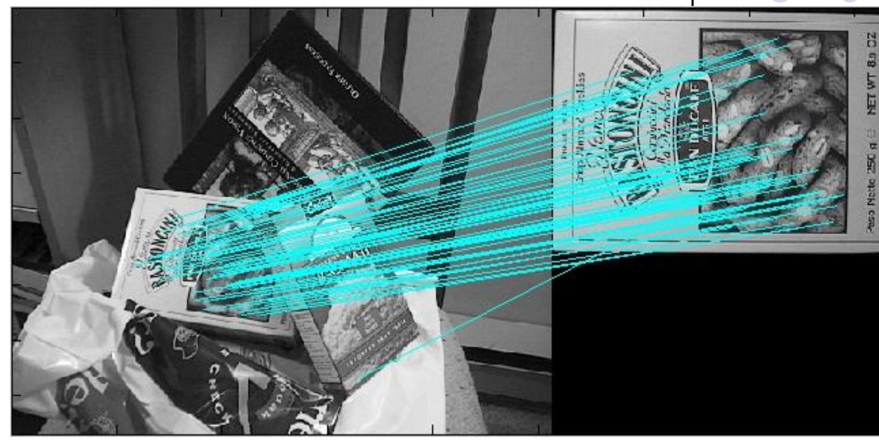
  - Media Search 2.0 (leverages on social tagging)
- Towards large-scale commercial applications
  - Consumer vs. Enterprise search
- Vertical domain search
  - Fashion search..



- Mobile search

Next Lesson

- Feature point extraction and matching



- Concept detection in images

Concepts present:

Tiger, grass.

- You will be ready for assignment 1: full details will be given next week