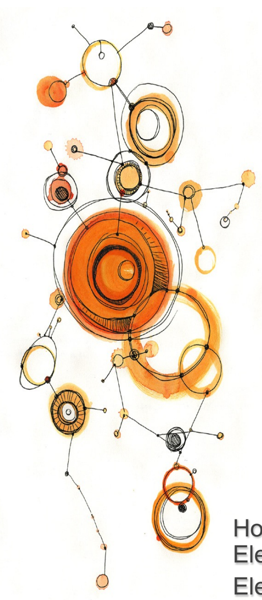
ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA



Modeling and Processing of Multimedia Data

International Second cycle degree programme (LM) in Digital Humanities and Digital Knowledge (DHDK)
University of Bologna

Multimedia Information Retrieval – Part III

Home page: http://www-db.disi.unibo.it/courses/DMMMDB/
Electronic version: 2.03.MultimediaInformationRetrieval-III.pdf
Electronic version: 2.03.MultimediaInformationRetrieval-III-2p.pdf

1

Outline

- MM queries
- Query formulation paradigms
- Semantic gap and MM data annotation
- Query results presentation
- Interactive queries
- Demo of some applications
  - SCENIQUE and SHIATSU systems

2

MM queries

- From previous lesson we know how to effectively represent MM data content by means of *low-level features*…
  - Global, e.g., color and texture of images
  - Local, e.g., the shape of an image region, or keypoints of an image
- …and how to compare such features, using suitable *distance* (or *dissimilarity*) *functions*, in order to compute their distance, i.e., their *dissimilarity* score
  - with the hypothesis that such score is a numeric value in the range [0,1], we can define the

similarity score $\cong$ *(1 – dissimilarity score)*

- Today we focus on how to query a MM collection

Problem: "*Given an input MM data object (i.e., a query) Q and a MM DB, we want to compute which are the objects in the MM DB that represent the query results wrt Q*"

MM query formulation paradigms (1)

- By text (attributes/annotations)

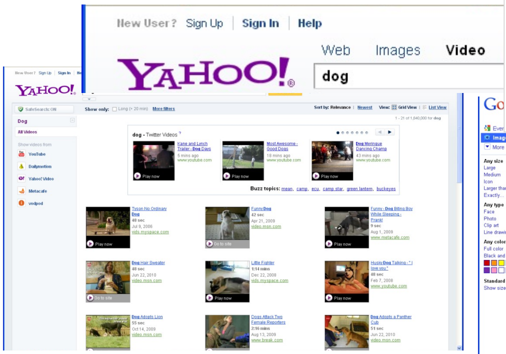
1. using *SQL statements* (if data is maintained into a DB)
   - E.g.: we suppose to have created the table **SONG** *with song records storing information such as the song artist, the title, etc.*

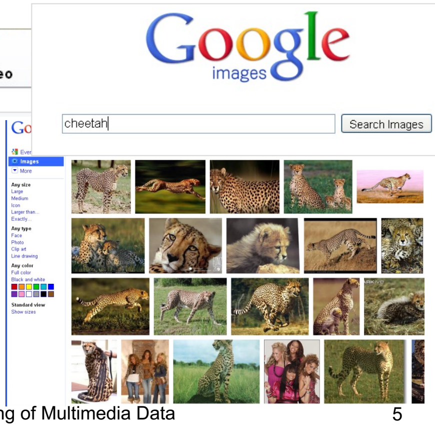
```
Select mp3
From   SONG
Where  artist = 'Manowar'
```

   - The execution of this type of searches exploits *standard DBMS technologies*

MM query formulation paradigms (2)

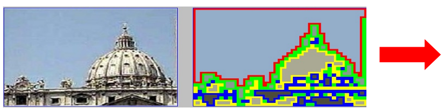
2. using the "*query-by-keyword*" paradigm
   - The execution of this type of queries exploits traditional *Information Retrieval* (IR) techniques
   - With the assumption that an association has been recorded between MM objects and keywords
     - .. will detail it very soon! ☺

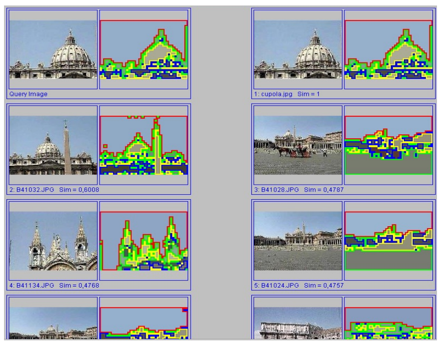




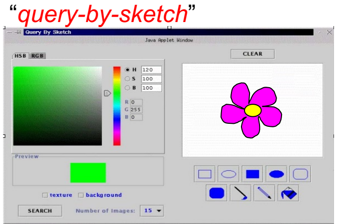
MM query formulation paradigms (3)

- By low-level features (e.g., color, texture, etc.)
  - following the "*query-by-example*" (QBE) paradigm first adopted in the IBM's query by image content (*QBIC*) system
    - *Full* vs. *partial* queries
    - The execution of this type of queries is based on *content-based MM data retrieval* techniques



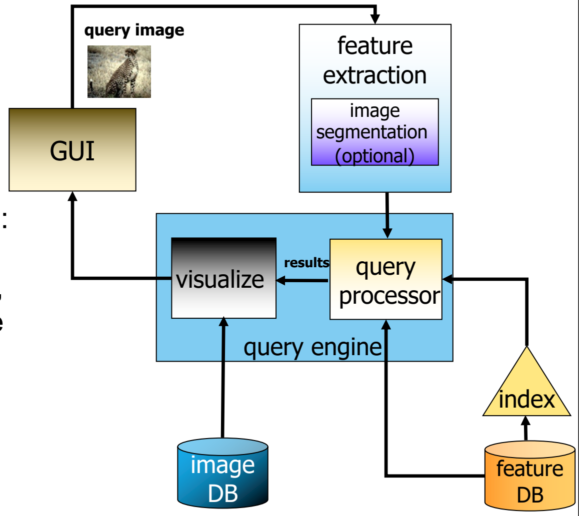


"*query-by-sketch*"



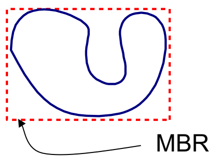
MM queries evaluation strategies

- Sequential evaluation: Q is compared with *all* DB objects
  - no feasible solution for large MM data sets
- Index-based execution: Q is compared with a subset of DB objects, with "guarantee" on the result correctness
  - to speed-up query evaluation time



Access methods for MM data

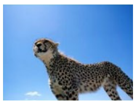
- Among the plethora of proposed access methods for multidimensional data, here we focus on two basic cases
- *R-tree* (Guttman, 1984)
  - *B-trees in multiple dimensions*
  - *MM object represented as a point in a vector space*
  - Index based on the concept of *Minimum Bounding Rectangle* (MBR)
  - Variants
    - R^+-tree (Sellis et al 1987)
    - R^*-tree (Beckmann et al 1990)



- *M-tree* (Ciaccia et al., 1997)
  - Intuitively, it *generalizes "R-tree principles" to arbitrary metric spaces*
  - The M-tree treats the distance function as a "black box"

The semantic gap problem

- Characterizing the object content by means of low level features (e.g., color, texture, and shape of an image) represents a completely automatic solution to MM data retrieval
  - However low level feature are not always able to properly characterize the semantic content of objects
    - e.g., two images should be considered “similar” even if their semantic content is completely different




- This is due to the semantic gap existing between the user subjective notion of similarity and the one according to which a low level features-based retrieval system evaluate two objects to be similar
  - prevents to reach 100% precision results

Possible solution

- (Semi-)automatically provide a semantic characterization (e.g., by means of *keywords* or *tags*) for each object able to capture its content
  - e.g., ([sky, cheetah] vs. [sky, eagle])

- Combine *visual features* with *tags* by taking the best of the two approaches


[sky, cheetah]


[sky, eagle]

Automatically infer semantics to MM objects

- *Automatic objects annotation* requires user intervention
  1) Relevant feedback
     - Exploiting user feedback to understand which are real relevant objects to the query… will see how very soon
  2) Learning
     - The system is trained by means of a set of objects that are manually annotated by the user (training phase)
     - Exploiting the training set, the system is able to predict labels for uncaptioned objects: the test object is compared to training objects; labels associated to the "best" objects are proposed for labeling (labeling & testing phases)

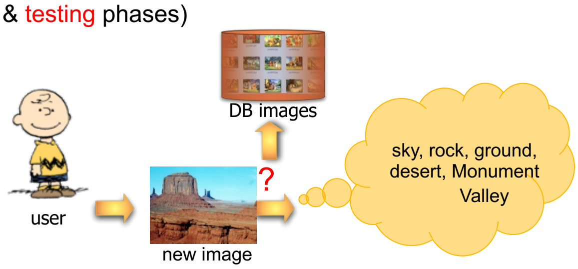


Image annotation problem

- How current commercial systems tackle the problem:
- Image search extensions of Google and Yahoo consider the original Web context, e.g.:
  - file name
  - title
  - surrounding text

  to support *keyword-based search*
- Microsoft's Photo Gallery, Google Picasa, and Yahoo's Flickr rely on user-provided tags or labels
- Apple iPhoto uses meta-data and user provided annotations
- Google similar images labs allows users to search for images using pictures rather than words (i.e., to find other images that look like the selected one)

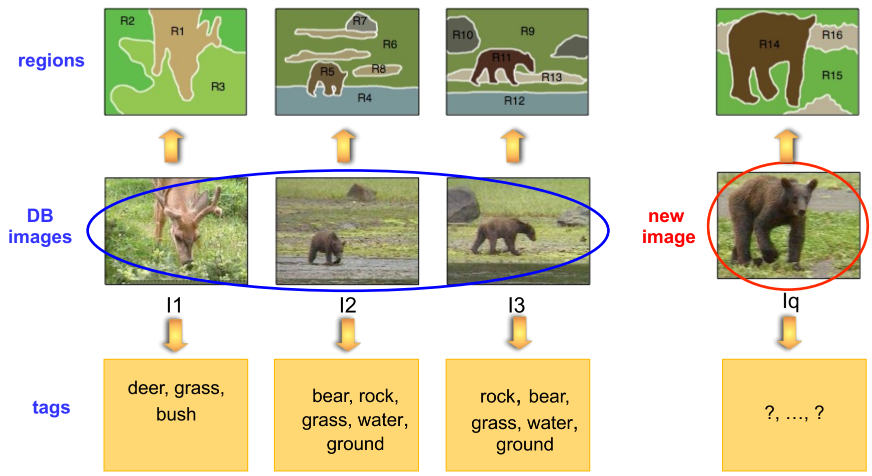
Imagination case study [BC08a]

- Imagination: IMAGe (semI-)automatic anNotATION
- Images as set of regions (à la *Windsurf*)

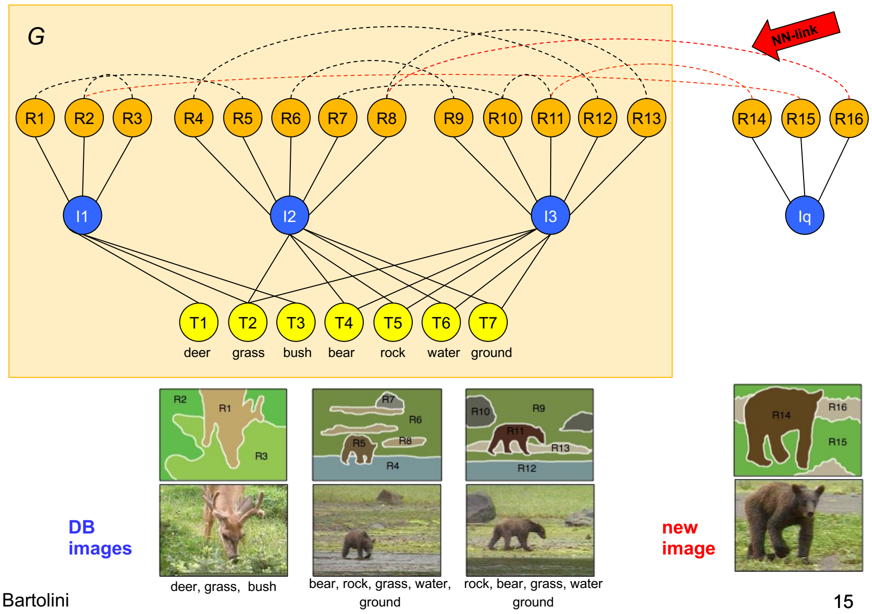
- Labels are *tags* which are associated at the image level
- *Graph-based* approach (à la "*Page Rank*")
  - 3-level of graph objects
    - Images
    - Regions with low level features (i.e., color and texture)
    - Tags assigned to images
  - plus *K*-NN links computed on region similarities

"Given a new image provide tags that are affine to the image and semantically correlated to each other"

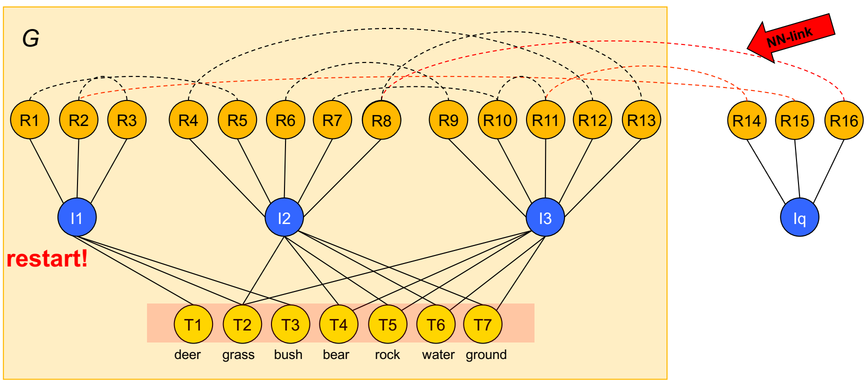
Intuitive example



Graph construction



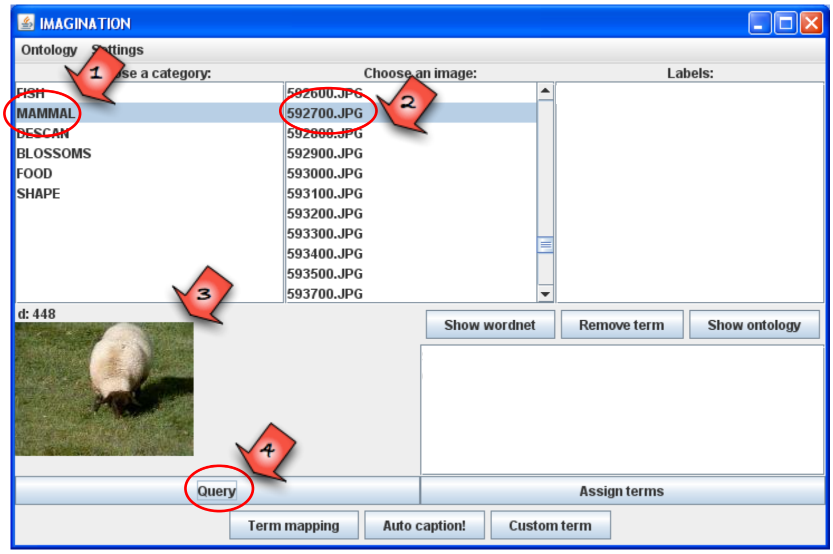
Random walk of the graph



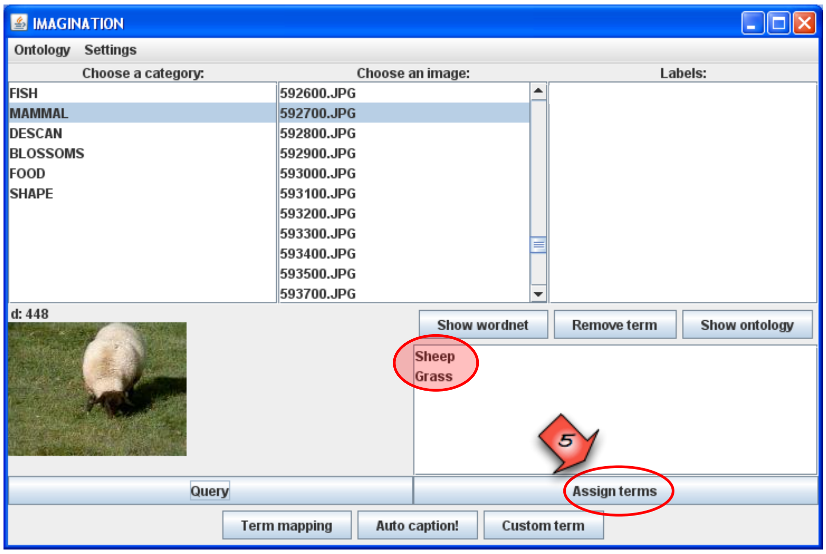
- restart at the query node (with probability *p*)
- randomly walk to one link (with probability *1-p*)

For each tag node a *relative frequency* (i.e., the *affinity*) is computed approximating the *steady state probability*

Imagination user interface



Predicted tags



Semantic-based image retrieval (1)

- How to compare two images based on its annotation (i.e., its tags)?
  - Exact text matching
    - simple ☺
    - not flexible ☹

…

Query: "flower"

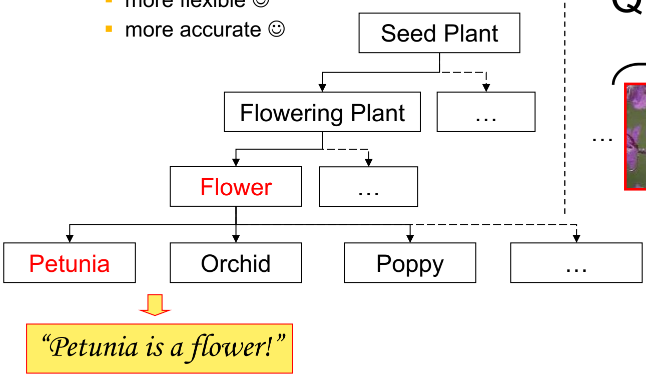
flower ☺

petunia ☹

…

Semantic-based image retrieval (2)

- Is it possible to perform better?
  - Yes! Exploiting the semantic relations of keywords belonging to a preexistent taxonomy or ontology (e.g., WordNet [Mil95]) and applying a "fuzzy" text matching
    - still simple
    - more flexible ☺
    - more accurate ☺



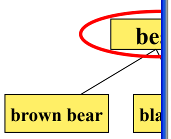
Query: "flower"

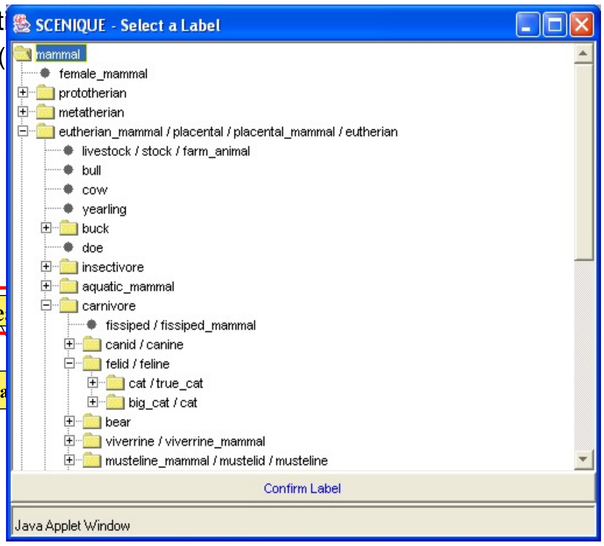
…   …

flower ☺ petunia ☺

Semantic relations

- Among possible semant…
  - hyponymy/hypernymy (… relation)
    - Define hierarchy
  - …

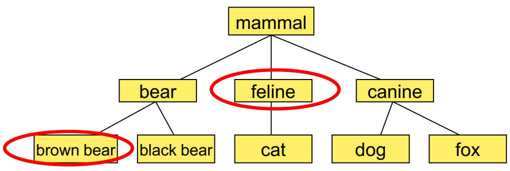


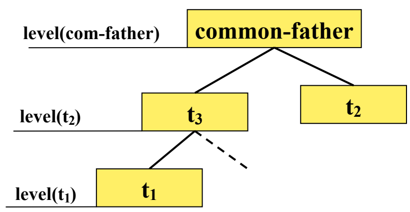


21

Semantic similarity

- Problem: Quantify the similarity between two terms of the hierarchy (e.g., brown bear and feline)



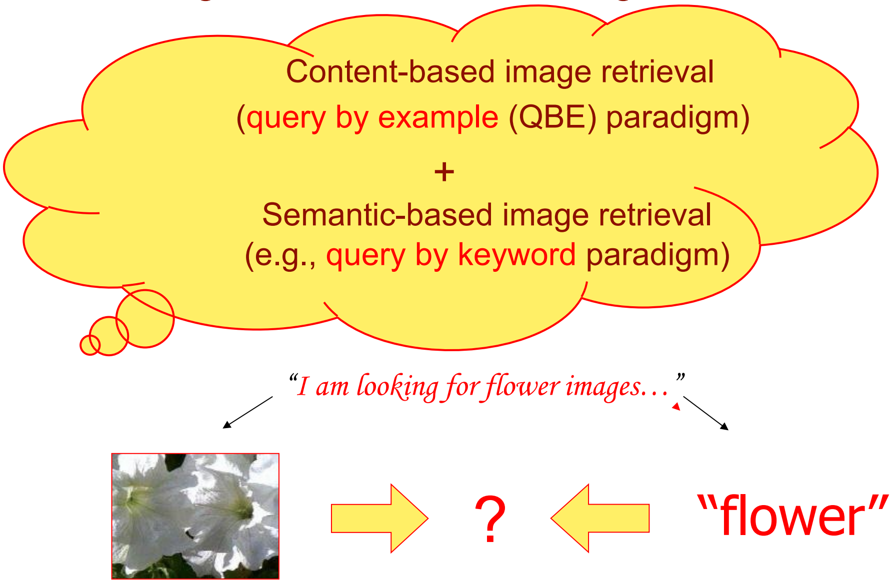


$$\text{Sim}(t_1,t_2) = \frac{2 * \text{level(common-father)}}{\text{level}(t_1) + \text{level}(t2)}$$

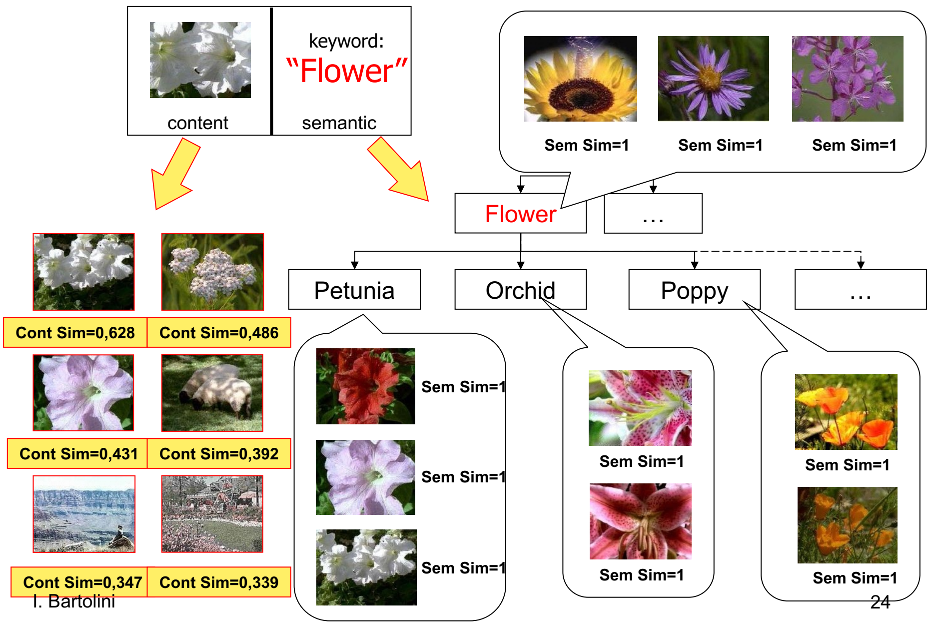
- The similarity between a term of the hierarchy and terms belonging its sub-tree is equal to one (sim=1)

22

Combining visual features with tags



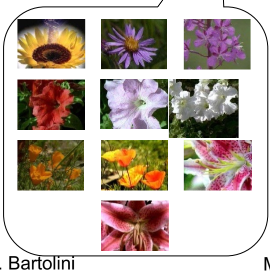
Practical example (1)



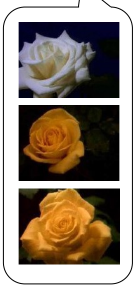
Practical example (2)

Keyword:"Flower"
Relaxation=92%

- livello 3, peso=90: Vascular Plant
  - livello 4, peso=83: Seed Plant
    - livello 5, peso=68: Flowering Plant
      - livello 6, peso=55: Flower
        - livello 7, peso=42: Petunia
        - Orchid
        - Poppy
        - …
      - …
    - …
  - …
  - Ligneous Plant
    - …
    - Bush
      - …
      - Rose



Sem Sim=1



Sem Sim=0,5

Sem Sim("Flower","Rose")=

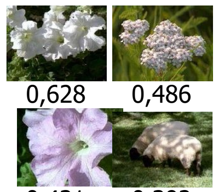
$$\frac{2*\text{livello("Vascular Plant")}}{\text{livello("Flower")}+\text{livello("Rose")}}$$

Integration policies

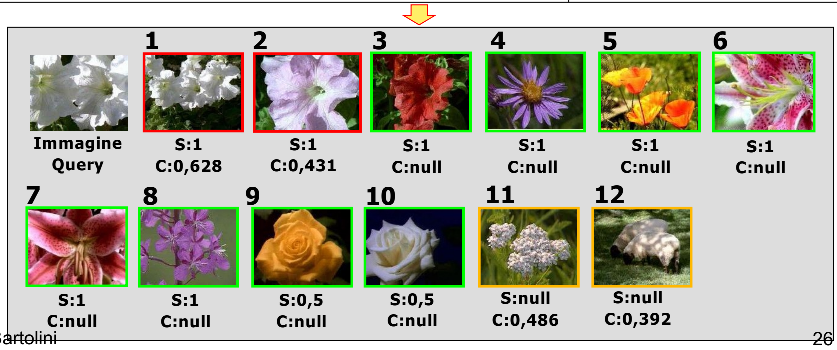
1) Semantic similarity

(Content similarity = null)

2) Content similarity



(Semantic similarity = null)



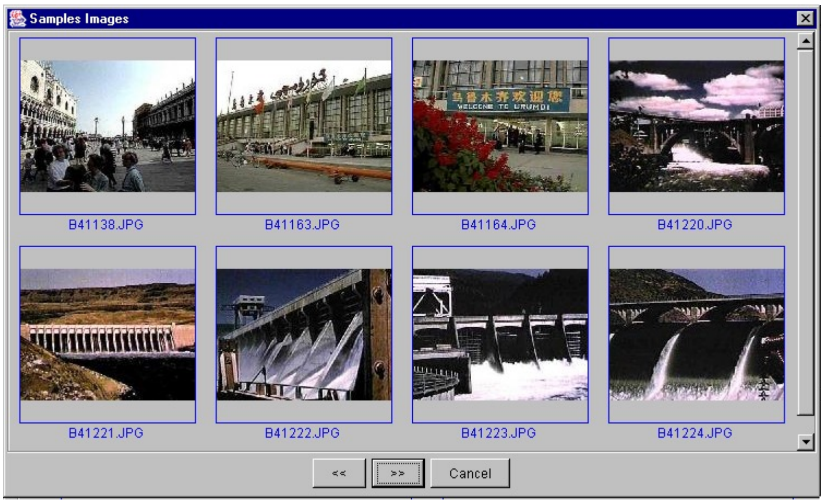
Content-based MM browsing

- Till now we have implicitly assumed that the user "knows"
  - *what she is looking for*
  - *how to formulate her queries*
    - e.g., QBE paradigm
- In some cases the user does not know at all what to look for; in these cases a "browsing" activity should be supported
  - to determine a good starting point for searching
  - to get an overall view of the DB contents
  - to give the user the ability to organize her MM collections (e.g. personal photos albums) in a semi-automatic way

MM browsing paradigms

- Browsing paradigms can be classified from *different/orthogonal* dimensions
- "*Graphical exploration direction*" dimension
  - Horizontal (or flat) vs. vertical (or hierarchical) vs. spatial
  - Effective graphical tools are essential in order to opportunely drive user during her browsing experience
- "*Object content*" dimension
  - Low-level feature-based vs. semantic-based vs. a mix of them
    - How MM objects are organized/clustered/grouped for improving the effectiveness of the user browsing session

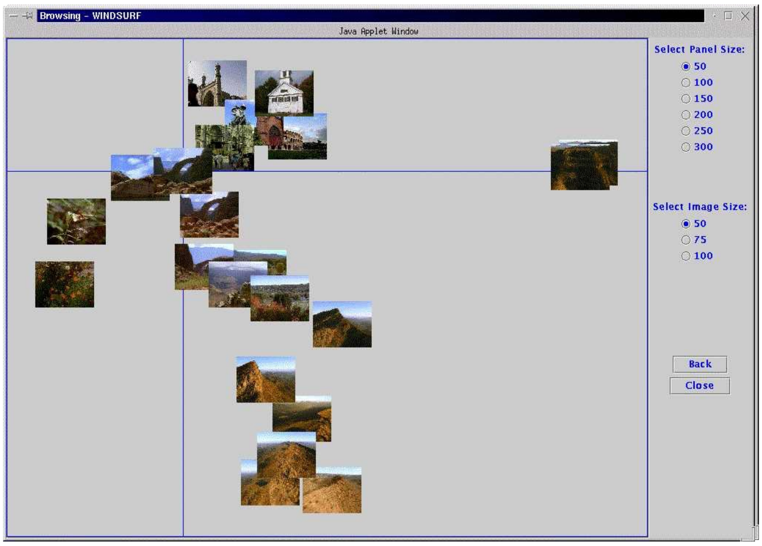
Flat browsing example



I. Bartolini Modeling and Processing of Multimedia Data 29

29

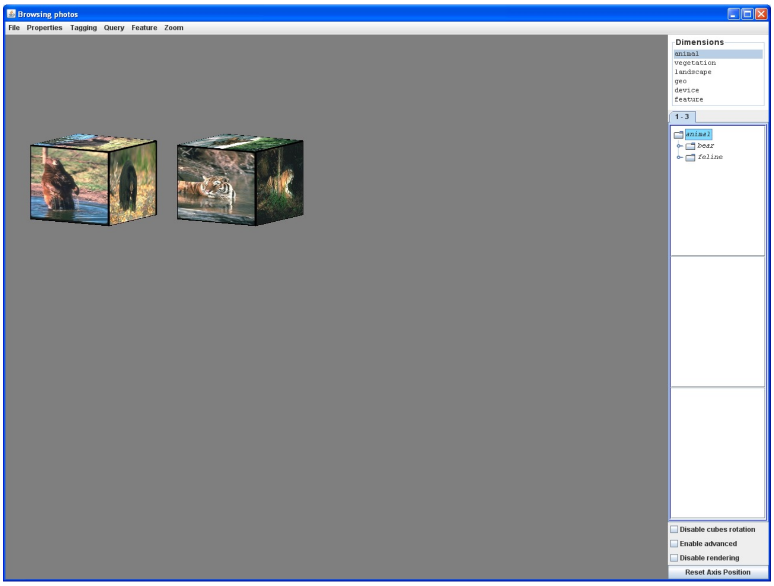
Vertical&Spatial browsing example



I. Bartolini Modeling and Processing of Multimedia Data 30

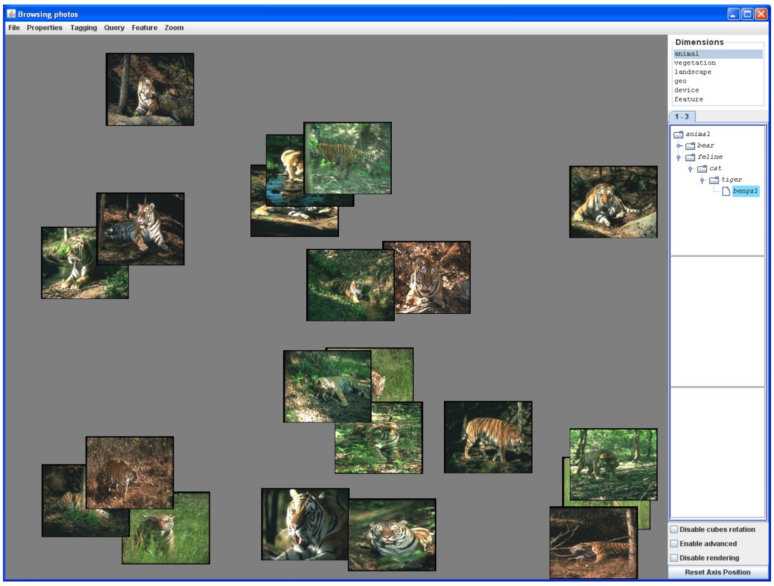
30

Semantic-based browsing example



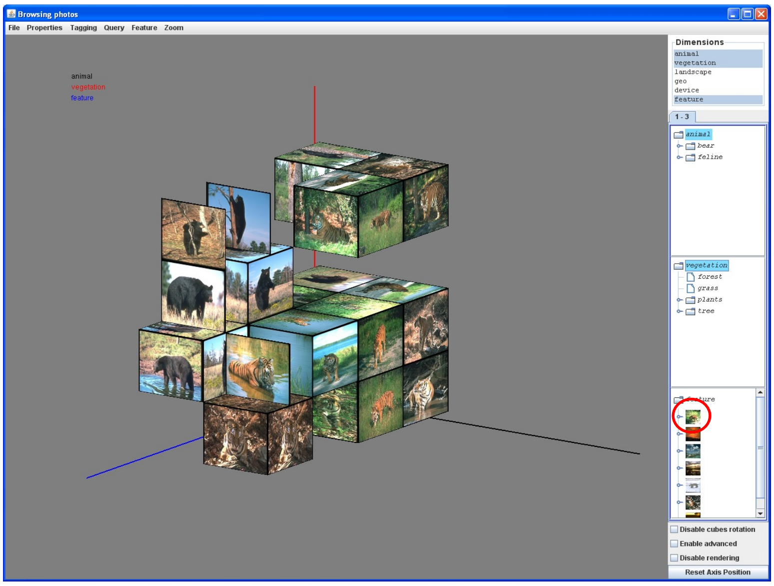
31

Semantic-based browsing example



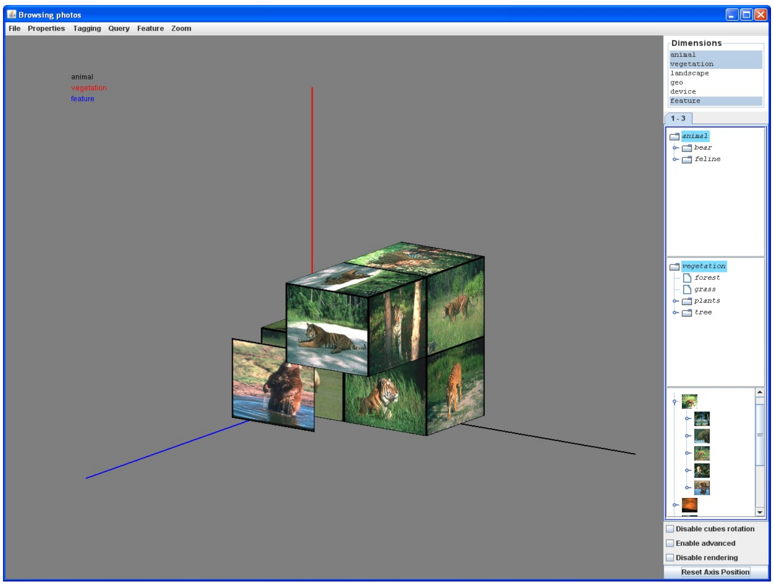
32

Putting it all together



33

Putting it all together



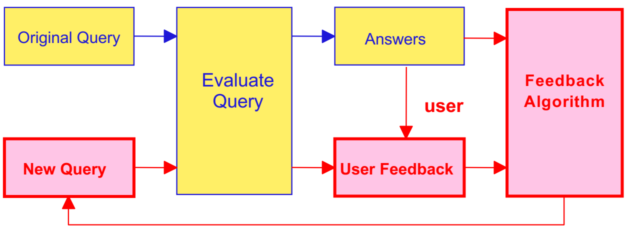
34

How can a user effectively search?

- Although with traditional DB's and a few attributes this might be a reasonable assumption, when we consider MM DBs with *many* attributes/features it is not clear *how a user might guess the right query and the right combination of weights*
  - E.g., how can you define the 64 weights of a color-based search using the weighted Euclidean distance?

The idea of relevance feedback

- Shift the burden of finding the "right query formulation" from the user to the system [RHO+98]
- For this being possible, the user has to provide the system with some information about "how well" the system has performed in answering the original query
- This user feedback typically takes the form of *relevance judgments* expressed over the answer set
- The "feedback loop" can then be iterated multiple times, until the user gets satisfied with the answers



Relevance judgments

- The most common way to evaluate the results is based on a 3-valued assessment:

Relevant: the object is relevant to the user

Non-relevant: the object is definitely not relevant (false drop)

Don't care: the user does not say anything about the object

- Information provided by the relevant objects constitutes the so-called "positive feedback", whereas non-relevant objects provide the so-called "negative feedback"
  - It's common the case of *systems that only allow for positive feedback*
- "Don't care" is needed also to avoid the user the task of assessing the relevance of **all** the results
- Models that allow a finer assessment of results (e.g., relevant, very relevant, etc.) have also been developed

A practical example (1)

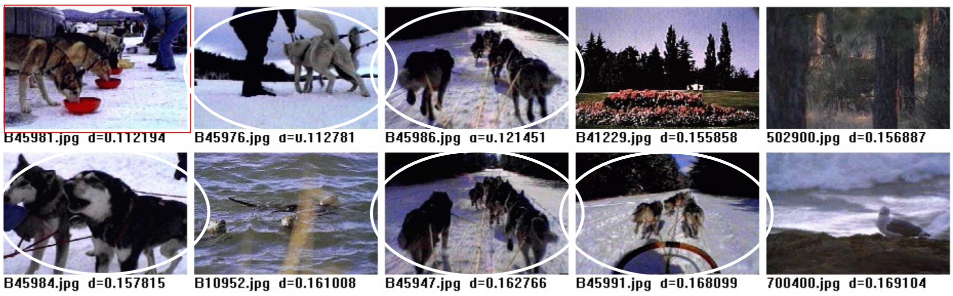
QueryImage Euclidean distance 32-D HSV histograms

This is the initial query, for which 2 object are assessed as relevant by the user

Precision = 0.3 (including the query image)

A practical example (2)

QueryImage



These are the results of the "refined" (new) query, generated using the 1st strategy we will see

Precision = 0.6 (including the query image)

A practical example (3)

QueryImage

These are the results of the "refined" (new) query, generated using the 2nd strategy we will see

Precision = 0.8 (including the query image)

A practical example (4)

And these are the results obtained by combining the 2 strategies…

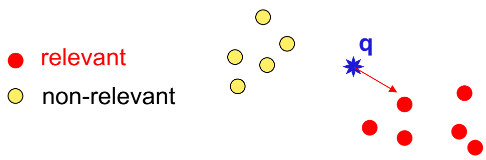
Precision = 0.9 (including the query image)

Basic query refinement strategies

- When the feature values are vectors, two basic strategies for obtaining a refined query from the previous one and from the user feedback are:

Query point movement:

the idea is simply to move the query point so as to get closer to relevant objects



Re-weighting:

the idea is to change the weights of the features so as to give more importance to those features that better capture, for the given query at hand, the notion of relevance

Query point movement

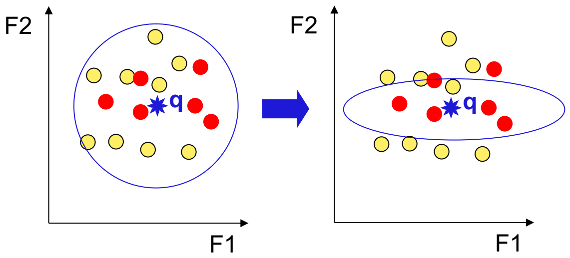
- The 1st formulation of the query point movement (QPM) strategy dates back to 70's, when it was proposed by J.J. Rocchio in the context of text retrieval systems based on the Vector Space model
- Rocchio's formula is:

$$\boxed{q_{new} = q_{old} + \beta \times \frac{\sum_{p_j \in Rel} (p_j - q_{old})}{|Rel|} - \gamma \times \frac{\sum_{p_j \in NonRel} (p_j - q_{old})}{|NonRel|}}$$

where:
- q_{old} is the previous query point
- Rel is the set of relevant objects that have been retrieved by q_{old},
- NonRel is the set of non-relevant objects that have been retrieved by q_{old},
- β and γ are non-negative parameters that control at which speed the query point moves towards relevant objects and far from non-relevant objects

Re-weighting

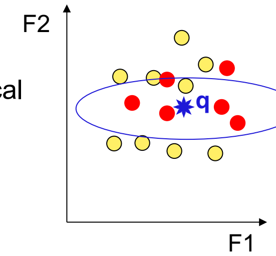
- The idea of the re-weighting strategy is to analyze the relevant objects in order to understand if some feature (dimension) is more important than others in determining "what makes an object relevant"



- The feature F2 allows a better discrimination than F1 of relevant and non-relevant objects

Variance-based re-weighting

- For the relevant case of weighted Euclidean distances, the re-weighting strategy is easily implemented as follows:
  - Let Rel = $\{p_1,\ldots,p_{|Rel|}\}$ be the set of relevant objects retrieved by q_{old}
  - Let $p_{i,j}$ be the feature value of p_j for the i-th feature (i=1,…,D)
- The weight w_i of the i-th feature is estimated as $\mathbf{w_i \propto 1/\sigma_i^2}$, that is, the inverse of the variance of feature values along the i-th coordinate
  - In the figure $w_2 > w_1$ since the variance on F2 is less than the variance on F1
- Besides the intuition, this strategy has a theoretical justification, which relies on the minimization of distances from the relevant objects [RH00]



Enjoy some demo applications ☺

SCENIQUE: *Semantic and ContENt-based Image QUErying* [BC08b, Bar09]

SHIATSU: *Semantic-based HIerarchcal Automatic Tagging of videos by Segmentation Using cuts* [BPR10, BR10a, BR10b, BPR13]

`http://www-db.disi.unibo.it/ibartolini/publications.html`