ELECTRONIC **WORKSHOPS IN COMPUTING**

Series edited by Professor C.J. van Rijsbergen

Rainer Manthey and Viacheslav Wolfengagen (Eds)

Advances in Databases and Information Systems 1997

Proceedings of the First East-European Symposium on Advances in Databases and Information Systems, (ADBIS'97), St Petersburg, 2-5 September 1997

A Query Language for Similarity-based Retrieval of Multimedia Data

Giuseppe Amato, Gianni Mainetto, and Pasquale Savino

Published in collaboration with the
British Computer Society

BCS

Springer

ISBN: 3-540-76227-2

A Query Language for Similarity-Based Retrieval of Multimedia Data

Giuseppe Amato
IEI-CNR
Pisa, Italy
E-mail: amato@iei.pi.cnr.it

Gianni Mainetto
CNUCE-CNR
Pisa, Italy
E-mail: G.Mainetto@cnuce.cnr.it

Pasquale Savino
IEI-CNR
Pisa, Italy
E-mail: savino@iei.pi.cnr.it

October 18, 1997

Abstract

This paper presents the main features of a Multimedia Query Language tailored for content-based similarity retrieval of multimedia objects. The Query Language processor is a component of a multimedia database system that adopts a model that permits both a structural representation of raw multimedia data and an automatically computed description of the multimedia data content. The Query Language is an extension of a traditional object-oriented query language. It allows to express restrictions on features, concepts and structural aspects of the multimedia database objects. In addition, the language supports the formulation of queries with imprecise conditions. The outcome of a query execution is an ordered set of pairs, each one consisting of an object and a measure of the similarity of the object with the criteria specified in the query.

1 Introduction.

Nowadays, content-based searching of multimedia objects is mainly provided for visual features such as color, texture, shape and spatial information contained within visual scenes. Existing systems (e.g. [7, 2]) support content-based retrieval of images and (possibly) videos using these extracted features. These features are used to compute the similarities between different images and videos.

In terms of the information used to represent the content of multimedia data, we can broadly classify the different approaches to content-based retrieval into three categories [8]:

keyword based, where the content of the multimedia data is described through annotations provided by users as for example free text or keywords taken from a controlled vocabulary;

feature based, where a set of features is directly *extracted*, i.e. computed, from the machine readable representation of multimedia data and used for the retrieval. Typical features are values that represent either generic information about multimedia data, such as *color, texture, shape, speed, position, motion, etc.*, or specific information needed by a particular application, such as *face recognition*, *trademarks* [12], *medical image analysis* [9]. Feature extraction is either performed through the supervision and support of the user or automatically, which is in some cases computationally expensive and usually application domain specific.

concept based, where application domain knowledge is used to perform an *interpretation* of object's content. This interpretation leads to the recognition of *concepts* which are used to retrieve the object itself. Usually this process is application domain specific and may require the user intervention.

In our opinion, the abstraction mechanisms of a model for multimedia data should be able to represent the features and concepts which are intrinsically present in every multimedia data item. Furthermore, the data

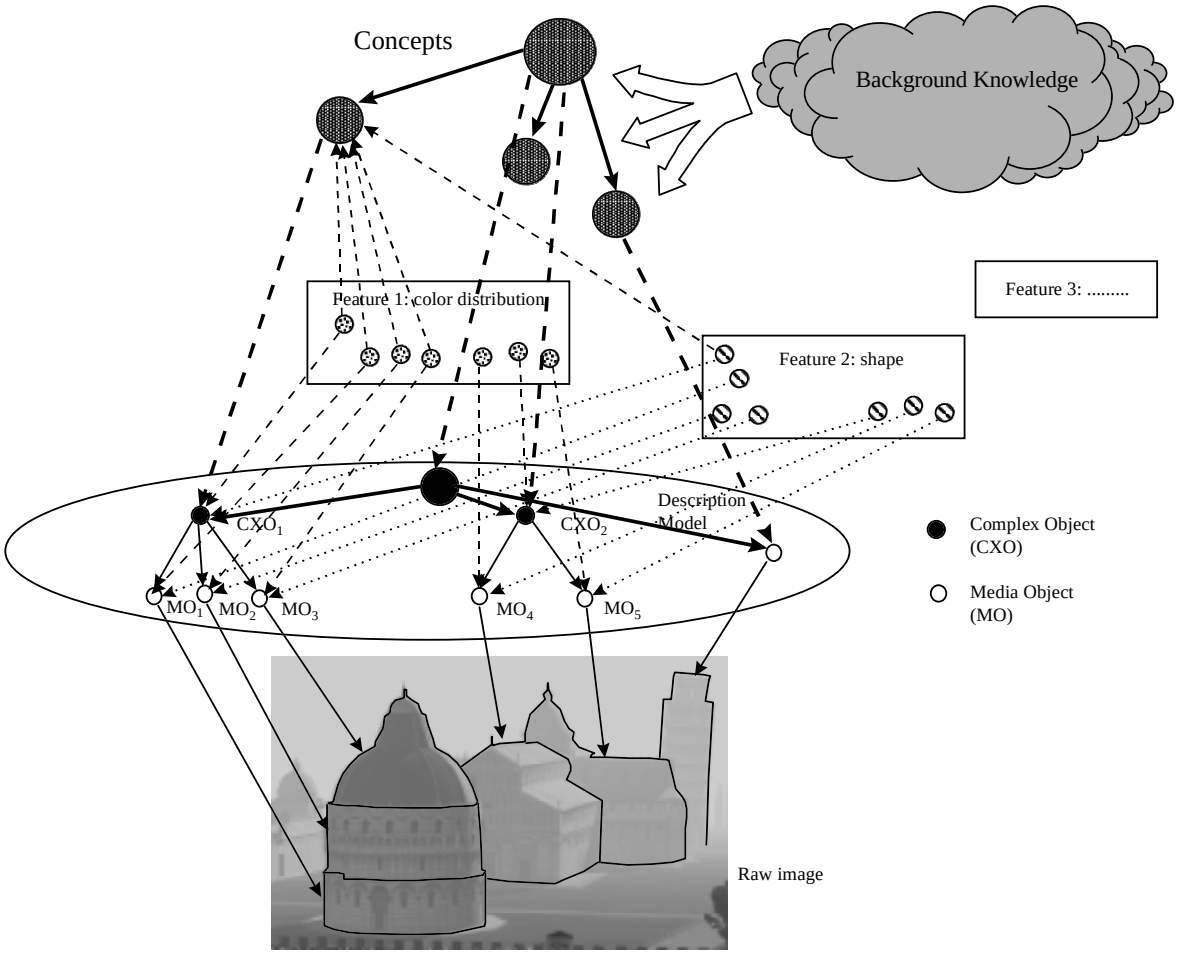

Figure 1: Example of the mappings from raw level, description level and interpretation level

model should allow to describe the structure of such data. In a recent paper [1], we have described the main characteristics of a model like that and its supporting system. The model is Object-Oriented: indeed the OO model is closer to the complexity of the real world. Since multimedia data are a raw machine readable representation of the reality, the OO model should better allow to structure the knowledge about such representation.

In this paper we will mainly describe how queries can be formulated and executed. The Multi Media Query Language (MMQL) has the usual expressive power of an O-O query language (such as, for example OQL [4]) and it has been extended to allow the formulation of interrogations that can at the same time consider restrictions on features, concepts and the structural aspects of MM objects. Furthermore, the language supports the formulation of queries with imprecise conditions. The outcome of an interrogation is an ordered set of pairs composed of an object together with a degree of matching with respect to the formulated query.

The paper is organized as follows. Section 2 provides an overview of the Multimedia Data Model. The Query Language, which exploits the features offered by the model, is illustrated in Section 3. Section 4 contains examples of queries while section 5 summarizes the paper and outlines some open issues and areas for future research.

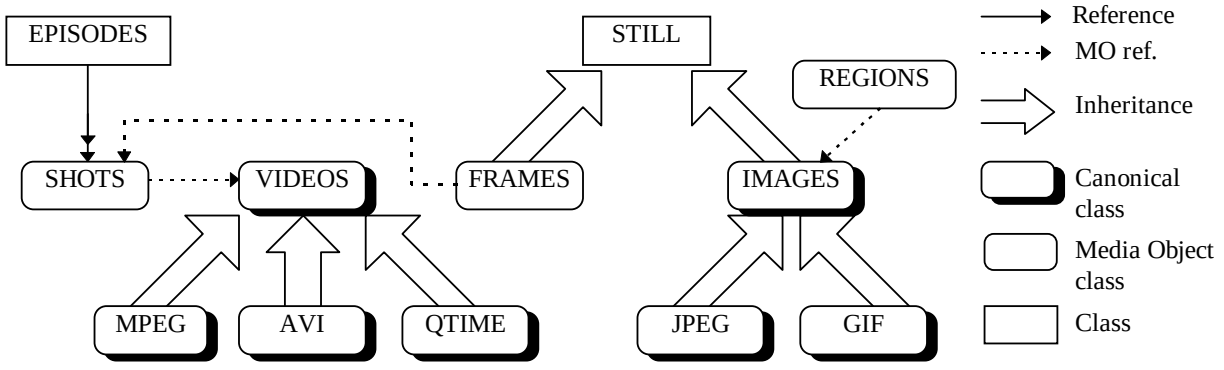

Figure 2: Description model schema

2 An overview of the Multimedia Data Model

At the lowest level of representation, a multimedia document is viewed as an unstructured piece of information, of type video, audio, image, or audio/video. A structural view of raw multimedia data is provided by the *Multimedia Description Model* (MDM), while the *Multimedia Presentation Model* (MPM), describes the temporal and spatial relationships among different structured multimedia data; the *Multimedia Interpretation Model* (MIM) allows semantic interpretations to be associated with structured multimedia data.

Figure 1 shows how the structure and the content of an image can be represented by using the model.

At the physical level any digital object can be stored; objects acquired through an acquisition device (e.g. a video camera, a scanner, etc.) as well as objects taken from a multimedia repository. At this level, any piece of information is a *raw object* (RO) which does not contain any description about its semantic content. The RO contains information about the physical encoding of the multimedia data and about the strategy used to store it. This can indicate where the object is located as well as an information that allows to achieve a certain quality when the object is presented. For example a RO may represent an image with a GIF encoding while the storage strategy may indicate that the object can be fetched by following a particular web URL. Another RO may represent an MPEG encoded video and its storage strategy specifies that it is stored striped across a disk array.

The *Multimedia Description Model* provides the mechanism for defining and manipulating a structured representation of the information contained in raw objects. The components of the raw object that are represented at the description level depend from the application purposes. Let us suppose that the main purpose is to provide a description of ROs in order to provide support to content-based retrieval of images describing historical places. Let us consider the image of Figure 1; the objects representing the tower, the Baptistery and the Duomo will be considered as the most relevant and are represented at the Description Level.

At the Multimedia Description Level, each RO is first represented by a *Canonical Object* (CO), which represent the entire multimedia document. Relevant portions of COs can be successively represented by *Media objects* (MOs). COs represent entire multimedia documents. For instance an image or a video are represented by a CO. The region of an image that contains a person can be represented with an MO. Other examples of Media Objects are a shot and a frame of a video. COs and MOs can also be composed into *complex objects* (CXOs). For instance an HTML page containing in-line images and embedded videos can be represented by a CXO that points to an CO that contains the HTML source, to a set of COs that contain the in-line images and to a set of COs that contain the embedded videos. An example of a description model schema is given in Figure 2.

The *Multimedia Interpretation Model* allows to represent the content of COs, MOs and CXOs. Two levels of

representation are considered in the interpretation model: the *feature level* and the *concept level*.

The *feature level* provides an interpretation of object's content by measuring physical properties of objects at the description level. These properties are called *features*: examples of features are colour distribution, shape, texture, motion vectors, etc. For each object of the description level, several features can be computed; at the same time, each feature can be applied to several objects of the description level (see Figure 1).

A feature consists of a quintuple <*fname, dclass, type, extrf, simf*>. *fname* is the name of the feature (e.g. *colour*, *motion vector*, etc.); the feature *fname* can be calculated for objects of the description level belonging to class *dclass*; the calculated values are stored using a data structure of type *type*; *extrf* is the function used to calculate the feature value and *simf* is the binary function that, given two feature values, allows to measure the similarity between them. Each class of objects of the description level may be associated with several features. The similarity function is necessary because the retrieval process, as described in next section, is imprecise: retrieved objects will not correspond exactly to the query specification. The similarity function allows to measure the degree of match between the feature value specified in the query and the values extracted from the objects in the database. Indexes can be created on feature values in order to speed up the retrieval process. The model provides the mechanisms to define dynamically new features, offering a level of flexibility which allows the adaptation to changes of user's needs.

The *concept level* describes the semantic content of objects of the description model. It may describe and represent the conceptual entities contained in an object and the relations among entities. This is obtained using an object oriented model extended to cope with the issues of multimedia document description. In particular objects and classes of the concept level represent *concepts*. A concept is recognized in an object of the description level, according to the values of the features extracted from the object and (possibly) other concepts already recognized for the same object. Each class and each object of the concept level is associated with a set of membership functions. Each membership function, given an object of the description model returns a value that represent the recognition degree of the concept. Membership functions may use various strategies to infer recognition degrees. The recognition degree may be for instance obtained comparing description model objects with a prototype document of a concept; a concept may be identified considering a particular combination of features; a concept may be recognised because other concepts have already been recognised. The definition of a new concept requires to provide a set of membership functions in order to recognise the concept in objects of a class of the description model. At the concept level traditional object-oriented information can be merged with multimedia information using the above described membership functions. An example of a schema of the concept level is given in Figure 3.

The *Presentation Model* allows to define complex spatial and temporal relationship among description model objects to create multimedia presentations. Its description is outside the scope of this paper.

3 The MM Query Language

The retrieval process in Multimedia Database Systems is inherently different from the retrieval process in traditional DataBase Systems (DBMS). Retrieval in DBMSs is based on the exact evaluation of a boolean combination of predicates on attributes. Each attribute has a well defined domain and predicates which can be applied to it. It is possible to exactly determine in a DBMS when the query is satisfied or not. The answer to a query is the set of database records for which the query condition (i.e. the boolean combination of predicates on record attributes) evaluate to "true".

The retrieval of Multimedia data consists in determining all documents whose properties are similar [12, 9, 10, 3, 5, 6] to those present in the query.

In general, a similarity-based retrieval is needed when:

- exact comparison is not possible, it is too restrictive or it may even lead to empty results-the data is vague and/or the user is not able to express queries in a precise way;
- ranking of retrieved objects is needed so that the set of retrieved objects can be restricted and/or qualifying objects shown to the user in decreasing order of relevance.

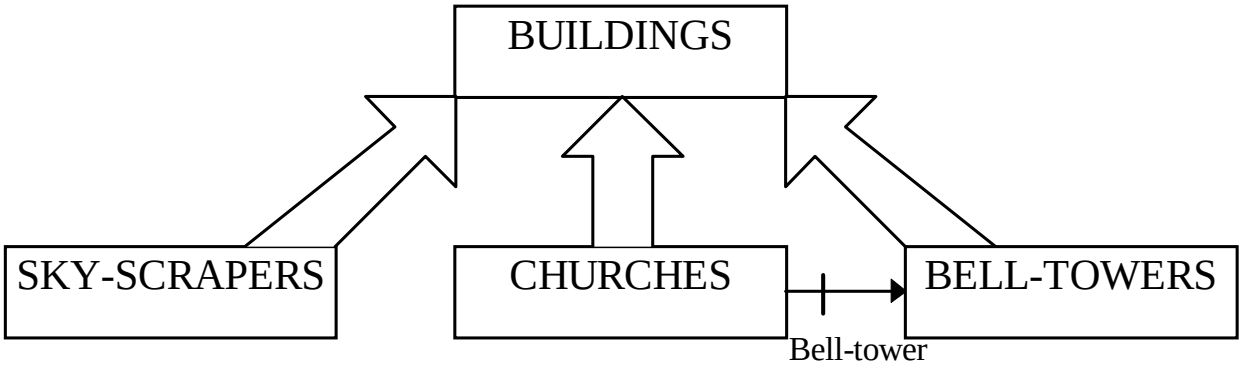

Figure 3: Conceptual level schema

Similarity based queries can be processed if a measure of the *similarity* between each retrieved object and the query (the *matching degree*) can be calculated. The result of a query execution is a set of objects ranked according to the value of the matching degree of each object. The imprecision of the match between the query and the retrieved objects is due to the fact that the interpretation of multimedia documents is intrinsically imprecise: the extraction of features and the recognition of concepts may be affected by errors; the result of the interpretation is the value of the feature or the name of the concept associated with the object, plus a *recognition degree* which gives a measure of the quality of the recognition. The most frequently used types of similarity queries are the *range* and the *nearest neighbor* queries. A *Range query* has the form: *find all objects that are similar to the desirable query object within a specific distance*; the *Nearest neighbor queries* have the form: *find the first k closest objects to the given query object*.

Traditional query languages does not provide an adequate support to express similarity based queries. Furthermore, the user usually has only an imprecise knowledge of the characteristics of documents he is seeking and it is also difficult, with features offered by available query languages, to express queries which help in discriminating between relevant and non relevant documents.

Here we propose a query language that has the traditional *select-from-where* syntax and has constructs specialised for multimedia data retrieval. Of course, because of the nature of queries that users would like to express in order to retrieve multimedia documents, the best interface to specify queries would be a graphical interface. The described query language can be considered as the target language of a *query formulation tool* which provides such a graphical interface.

The proposed language allows to express the following types of restrictions:

Features: The user may express restrictions on the values of *object's features*. An example of a query on features, provided that the features color, spatial position and motion have been defined, is "Retrieve all videos that have a *red* spot on this *position* that is *moving toward left*".

Concepts: Queries that check the presence of a concept in a multimedia document can be expressed using objects of the concept level. As mentioned above, objects of the concept level are associated with a set of membership functions used to check the presence of a concept in objects of the description level. An example of a query that uses concepts is as follows: "Retrieve all images that contain a *person*"; we suppose that the concept person has been defined and that this concept is associated with a membership function for the class *images* of the description model.

Object Structure: Single media objects as well as multimedia objects are structured, as illustrated in previous section. The query language will allow the user to express restrictions on the structure of the multimedia objects to be retrieved. An example of query that uses the structure is: "Retrieve all *shots* of this *episode* of this *video*."

Spatio-temporal aspects: An important feature of multimedia data is related to the spatial and temporal relationships among different objects. The user should have the possibility to formulate restrictions on the spatial and temporal relationships of the objects to be retrieved.

This is possible if specific features have been defined to express object spatial and temporal position. These features also entail defining operations to measure the relative position of two (or more) objects.

Furthermore the proposed query language allows to handle the following aspects:

Uncertainty: The query language will allow to express the uncertainty the user has on some of the restriction formulated and on the preference the user may have on some conditions with respect to others. For example the user may not be certain of the color of an object, while he is sure of the presence of an object. The values of preference and importance will be used to measure the degree of matching between the query and the retrieved objects.

Imprecision: Similarity and classification operations are intrinsically imprecise. Similarity based retrieval is needed since often exact comparison is not possible, it is too restrictive or it may lead to empty results. Ranking of the retrieved elements is needed in these cases so that the set of the retrieved elements can be restricted and/or qualifying elements shown to the user in decreasing order of relevance. To cope with these issues, the logical operators of the query language should deal with *recognition degrees* instead of *booleans*. Query executions should always return an ordered set of objects. In the ordered set each element is associated with a value that represents a measure of the degree of relevance of the element and the ordering is made with respect to that value.

3.1 The Query Language definition

The *Multi Media Query Language (MMQL)* described in the following offers both standard functionality of an O-O query language (such as, for example OQL [4]) and constructs that correspond to the above described characteristics.

A query has the typical `select-from-where` structure:

```
Q ::= select <select-stmnt>
      from   <from-stmnt>
      where  <condition>
```

where

- the `<from-stmnt>` specifies the set of objects that are candidates to be returned by the query. It may be one of the following elements: a *concept level class*, a *description level class*, a *query*, the *union* **U**, *intersection* **I**, or *Cartesian product* **X** of two queries:

  `<from-stmnt>` ::= CLASS | DCLASS | Q | $\mathbf{U}(fs_1, w_1, fs_2, w_2)$ | $\mathbf{I}(fs_1, w_1, fs_2, w_2)$ | $\mathbf{X}(fs_1, w_1, fs_2, w_2)$

  where $fs_1, fs_2 \in$ `<from-stmnt>`

  The union, intersection and Cartesian product operations do not follow the *boolean logic*. For example, $\mathbf{U}(Q_1, w_1, Q_2, w_2)$ represents the union of queries Q_1 and Q_2 with a relative importance w_1 and w_2 respectively. This operation will perform the union of the result of Q_1 and Q_2; the degree of matching of each object will depend on its degree of matching in Q_1 (resp. Q_2) and the relative importance w_1 (resp. w_2).

- `<select-stmnt>` has the form:

  `<select-stmnt>` ::= [*n-hits*] `<select-list>`

  where *n-hits* indicates that only the first *n-hits* should be put in the result and `<select-list>` is an expression that specifies what a query should return for each element of the from statement that has been validated in the query condition. It represents the projection of the query. The query projection is a valid expression based on constants and on paths rooted at `<from-stmnt>`.
  For example, if the from statement of the query contains the *class Persons* that as an attribute *Name*, then a valid expression for the select-list could be *Person.Name*.

- `<condition>` is the query condition. It can be a simple condition or a complex condition. A simple condition can be a precise or imprecise comparison. A complex condition is an expression that may involve simple conditions and other complex conditions:

  `<condition>` ::= `<simple-cond>` |
  `<complex-cond>`

  `<simple-cond>` ::= `<precise-compar>` |
  `<imprecise-compar>`

The precise comparison expressions are the usual comparison expressions such as equal, less-than, greater-than. In the following we will describe the imprecise comparisons and the complex expressions.

We indicate with *v* expressions that return a generic value, with *o* expressions that return a generic object, with *do* expressions that return an object of the description level, with *mo* expressions that return a media object, with *c* expressions that return a concept, with *e* condition expressions, with *w* weights to be associated to conditions when complex condition are specified, with *fv* expressions that return feature values.

Imprecise comparison

Imprecise comparition, as previously stated, returns recognition degree instead of boolean values. These recognition degrees are real values that range from 0 to 1.

The proposed language basically provides three operators that perform imprecise comparitions. The first checks the similarity between two feature values. The second checks recognition degree of a concept in a multimedia document. The last allows to infer to what degree a value belongs to a ranked set. The syntax for these operators is the following:

fv_1 `sim` fv_2	similarity between feature values
do `match` *c*	tests if a descr. obj. matches a concept
v `in` *Q*	tests if a value belongs to a ranked set

The **sim** operator is used to evaluate the similarity between two feature values. Since the set of features that can be used is not fixed but users can define new features, as we said previously, the behavior of this operator cannot be determined statically. It depends on how the feature, which the specified feature values belong to, has been defined. Actually the evaluation of the **sim** operator corresponds to evaluate the similarity function associated with the feature which the feature values belong to. The **sim** operator will behave in different way if we check the similarity between two colours, between two shapes, between two motion vectors or between two strings.
The **match** operator measures the recognition degree of a concept `c` in a description object `do`. Also in this case its behavior cannot defined statically. Evaluating the match operator corresponds to evaluating the membership function of the concept *c* corresponding to the description model class which the description model object *do*

belongs to. As we said previously, the membership functions should be provided by users when they define new concepts. Two different concepts may be associated with different membership functions.
The **in** operator tests to what degree a value belongs to a ranked set. It actually returns the rank associated with the specified value in the ranked set. It corresponds to the traditional SQL **in** operator with the difference that it works whit ranked sets instead of traditional sets and returns recognition degrees (i.e. ranks) instead of booleans.

Complex conditions

Simple conditions can be combined into complex conditions through the use of **and**, **or** and **not** operators. In the following we consider that two expressions, e_1 and e_2 have to be combined. Expression e_1 (resp. e_2) has a relevance w_1 (w_2). The relevance allows one to specify the weight to be assigned to each expression. It takes into account the uncertainty that users may have regarding some of the conditions they are expressing (for example they want to express that it is much more important that the retrieved images contain a church rather than of a bell tower). The proposed operators are the following:

$e_1[, w_1]$ `and` $e_2[, w_2]$
$e_1[, w_1]$ `or` $e_2[, w_2]$
`not` e

The query language does not impose any constraints on the method to be used to compute the recognition degree of the complex expressions. This is a task of the query processor, which is not described in the paper. However, various approaches can be followed, such as the adoption of fuzzy logic, probabilistic logic, or the probabilistic model of Information Retrieval [11].

The language is not tied to a specific approach whose validity has still not been demonstrated. The query language uses constructs that deal with imprecision and it manages recognition degrees, but we do not restrict it to a specific approach. However, a specific approach must be defined when the language is implemented, since its choice affects implementation and optimisation techniques.

The intuitive meaning of the **and** and **or** operators should be preserved, irrespective of the implementation: the use of the **and** operator means that both recognition degrees of the terms should be high; the use of the **or** operator means that at least one of the recognition degrees should be high. The difference between the actual behavior and the intuitive behavior of the language can be considered as a matter of precision and it is often subjective. Since the information is also intrinsically imprecise, the user may also use the language without knowing the actual implementation of constructs. Users may only use their intuitive ideas of the behavior knowing that the results are affected by a certain degree of imprecision.

Selectors

The proposed language supports all classical expression of an object oriented query language and constructs specific for the adopted multimedia model. In particular, three specific selectors are offered to cope with feature extraction, recognition degrees and structure of multimedia documents in addition to classical selectors for accessing fields of structured values and for evaluating objects' methods. The first allows to access the value of a particular feature in a multimedia document. The second allows to access the recognition degree of a candidate value of the query, the third allow to access the object which a media object has been extracted from:

o.`feature`*(fid)*	feature value of a descr. obj.
v.`rec-degree`	recognition degree of a value
mo.`part-of`	original object of a media obj.
v.attribute	field of a structured value
o.method(args)	method evaluation of an object

Given an object of the description level, it is possible to access its feature value corresponding to the feature *fid* using the **feature** selector. This would correspond to evaluating the feature extraction function associated with the specified feature. However notice that generally it is not needed to evaluate that function since feature extraction is done when objects are inserted into the database, in order to index them, and the feature values can be successively obtained simply accessing some indexing structure.

The **rec-degree** selector is used to access the recognition degree of values. During the query processing, the values specified in the `<from-stmnt>` are validated using the `<condition>` specified for the query. This validation process corresponds to check the `<condition>` for each value of the `<from-stmnt>`. At the end each value is associated with its recognition degree. The **rec-degree** selector allows to access this recognition degree. For instance the following query:

```
select I
from IMAGES as I
where
   I.feature(color) sim red
   and
   I.rec-degree > 0.7;
```

returns all images whose color is similar to red with a degree greater than 0.7.
Since the condition can be a complex condition, the **rec-degree** selector returns the recognition degree of a value limited to the context in which the selector is used. For instance the following query:

```
select I
from IMAGES as I
where
   (
      I.feature(color) sim red
      and
      I.rec-degree > 0.7
   )
   and
   (
      I.feature(shape) sim circle
      and
      I.rec-degree > 0.5
   );
```

returns all images whose color is similar to red with a degree greater than 0.7 and whose shape is similar to a circle with a degree greater than 0.5. On the other hand the following query

```
select I
from IMAGES as I
where
   (
      I.feature(color) sim red
      and
      I.feature(shape) sim circle
   )
   and
   I.rec-degree > 0.6;
```

returns all images whose color is similar to red and shape is similar to circle with a joint recognition degree greater of 0.6.

The **part-of** selector is used to access the object of the description model which the specified media object refers to. This can be useful for instance, given a media object that represent a shot or a frame, to access the video which it belongs to.

4 Examples

In the following we present some examples of queries that give the flavor of the proposed query language. In the examples the description model schema and concept level schema of Figure 2 and Figure 3 are used.

At the description level we consider videos and images. Videos can be encoded in MPEG, AVI and Quick time. Images can be encoded in JPEG and GIF. We represent the structure of videos in terms of frames shots and episodes. We represent regions of images that contain relevant entities. This is expressed in the description model schema by defining two class of canonical objects: the VIDEOS and the IMAGES classes. The class VIDEOS has three subclasses of canonical objects: MPEG, AVI and QTIME classes, each one corresponding to a different video encoding. The class IMAGES has two subclasses of canonical objects: the JPEG and GIF classes. We represent shots and frames as media objects. These media objects are grouped in the SHOTS and FRAMES classes of media objects. Shots are extracted form videos. Frames are extracted from shots. Set of shots that form an episodes are represented by objects of the EPISODES class. Relevant regions of images are represented by media objects of the REGIONS class. Frames and images are both considered as still images so the FRAMES and IMAGES classes are subclasses of the STILL class.

At the concept level we suppose to model important buildings. Actually we want to manage information about sky scrapers, churches and bell towers. In the concept level schema the BUILDINGS class has three subclasses: the SKY-SCRAPERS, CHURCHES and BELL-TOWERS classes. The objects of the CHURCHES class that represents churches that have an important bell-tower are associated with the object of the BELL-TOWER class that represents such a bell-tower. We suppose that classes and objects are all associated with membership functions that allow to test if objects of the description level contain the concept they represent.

We suppose that the feature *shape*, the feature *color* and the feature *position* are defined. They are extracted from objects of the class REGIONS of the description level. The functions *left-to* and *right-to* have been defined to compare positions.

Query 1

Features can be used to retrieve multimedia documents that contain concepts that have not been defined. For instance in our example the concept *sun* has not been defined. However we can search for regions that contain the sun searching for regions that are circular and red. Therefore the query that we should express is "*retrieve the 10 best fitting regions that are circular and mostly red*". Let's suppose that we are more confident on the fact that the region we are looking for is red than it is circular. In the query we use the two constants *circle* and *red*, that we suppose have already been defined. They represent the feature values corresponding to a circle and to the red color. The following query accomplishes to this task:

```
select 10 R.oid
from REGIONS as R
where
   R.feature(color) sim red, 0.7  and
   R.feature(shape) sim circle, 0.3;
```

The user's uncertainty has been expressed by associating different weights to the two clauses of the condition. We used 0.7 for the similarity to the *red* color and 0.3 for the similarity to the *circle* shape. Notice that, if we wanted to define the sun concept, the previous query could be used to build a membership function for such a concept.

Query 2

Let us suppose that we want to "*retrieve all images that contain churches with their bell tower*". We want to privilege the fact that in the image there is the church instead of the fact that there is the bell tower.

The query performs a Cartesian product between the IMAGES and the CHURCHES classes then it keeps just the rows in which the image matches both the church and its bell tower:

```
select I.oid
from CHURCHES as C, IMAGES as I
where
   I.oid match C.oid, 0.6
   and
   I.oid match C.bell_tower, 0.4;
```

The previous query mixes information extracted from multimedia data and conceptual information: the association between a church and its bell tower is a concept level information; the fact that both appear in the same images is inferred from multimedia data.

Query 3

Let's suppose that the concept *churches* is associated with a membership function that allows to recognise it in video frames. We want to "*search for all shots that contain a church*". In order to retrieve shots that contain a church we can use the *churches* concept to retrieve frames that contain a church, then we can use the information on the structure of videos to retrieve the corresponding shots. In addition let's suppose that we want to be quite accurate so we want that the recognition degree should be greater than 0.7. The following query returns what we need:

```
select S.oid
from SHOTS
where
   S.oid in (
      select F.part-of
      from FRAMES as F
      where
         F.oid match CHURCHES)
   and S.rec-degree > 0.7;
```

Query 4

Spatial queries can be expressed by using the feature *position*. Let us suppose that we want to "*retrieve the 10 best fitting images that contain a circular region on the left of a rectangular region*". Let us suppose that we have already defined the two constants *circle* and *rectangle* that respectively represent the feature values corresponding to a circle and to a rectangle. This query can be expressed as follows:

```
select 10 I.oid
from IMAGES as I, REGIONS as C, REGIONS as R
where
   C.part-of = I.oid and
   R.part-of = I.oid and
   C.feature(shape) sim circle and
   R.feature(shape) sim rectangle and
   left-to(C.feature(position),
       R.feature(position));
```

In the query the uncertainty is introduced when the shapes of the regions are compared with that of circles and rectangles. The other clauses of the condition are in fact boolean conditions.

Query 5

The following query uses at the same time concept level, feature level and description level information. Let us suppose that we want to "*retrieve all buildings whose shape is similar to that of the Empire State Building*". Let us suppose that objects of the concept level class BUILDINGS have a field *name* that contains the name of the building. We want to privilege the fact that a region is a building instead of the fact that it is similar to the Empire state building.

The query should first retrieve all regions that match the Empires State Building. Then all regions that match some building and whose shape is similar to that of the regions containing the Empire State Building are retrieved. Since we want to give more importance to the fact that a region is a building then it is similar to the Empire state building, we use the weight 0.7 for the first clause and 0.3 for the second:

```
select B.oid
from REGIONS as R1, BUILDINGS as B
where
  R1.feature(shape) sim any(
    select R.feature(shape)
    from REGIONS as R
    where
      R.oid match (
        select unique SS.oid
        from  SKY_SCRAPESR as SS
        where SS.name='Emp. St. Build.')), 0.3
  and
  R1.oid match B.oid, 0.7;
```

5 Conclusions and future work

In this paper we presented a Query Language that supports (i) partial match retrieval, i.e. there are retrieved all objects which are similar to the query at least with a certain degree; (ii) restrictions on the values of features, the presence of concepts and the structure of objects; (iii) the possibility to take into account user uncertainty on some parts of the query (iv) the possibility to take into account the imprecision of the interpretation of the content of the multimedia object.
The result of a query is a ranked set, that is a set of pairs (object, recognition degree). The recognition degree is a measure of the degree of match between the query and the object.

We consider our future work to evolve along the following main directions:

- Study how to combine similarity degrees deriving from different features and concepts.
- The implications of these models with the storage and access of multimedia objects will be studied. Indeed, real application environments will require the storage of many trillions of bytes of data, requiring the use of a storage hierarchy consisting of different layers. This implies that data placement is crucial for effective manipulation of the data and for an efficient retrieval.
- Study of an effective and efficient query processing algorithm.
- Completion of the implementation of the system that supports the proposed model.

Acknowledgment

This work has been partly funded by European Union under ESPRIT LTR Project HERMES, No 9141

References

[1] G. Amato, G. Mainetto, and P. Savino. A Model for Content-Based Retrieval of Multimedia Data. In *Proc. of 2nd Multimedia Information Systems Workshop*, West Point, USA, September 1996.

[2] J. R. Bach, C. Fuller, A. Gupta, A. Hampapur, B. Horowitz, R. Humphrey, R. Jain, and C.-F. Shu. The Virage image search engine: An open framework for image management. In *Proceedings of the SPIE 96*, 1996.

[3] A. Cardenas, I. Ieong, R. Taira, R. Barker, and C. Breant. The Knowledge-Based Object-Oriented PICQUERY+ Language. *IEEE Transactions on Knowledge and Data Engineering*, 5(4):644–657, Aug. 1993.

[4] R. Cattel. *The Object Database Standard: ODMG-93, Release 1.2*. Norwell, MA, 1996.

[5] Y. Day, S. Dagtas, M. Iino, A. Khokhar, and A. Ghafoor. Object-Oriented Conceptual Modeling of Video Data. In *Proc. of the 11th Int. Conf. on Data Engineering, Taiwan*, pages 401–408, 1995.

[6] N. Dimitrova and F. Golshani. Rx for Semantic Video Database Retrieval. In *Proceedings of the ACM Multimedia '94*, 1994.

[7] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content: The QBIC system. *IEEE Computer*, 28(9):23–32, September 1995.

[8] V. N. Gudivada and V. V. Raghavan. Content-based Image Retrieval Systems: Guest Editors' Introduction. *IEEE Computer*, pages 18–22, September 1995.

[9] E. Petrakis and C. Faloutsos. Similarity Searching in Large Image Databases. *IEEE Transactions on Knowledge and Data Engineering*, 1996.

[10] E. Petrakis and S. Orphanoudakis. Methodology for the Representation, Indexing and Retrieval of Images by Content. *Image and Vision Computing*, 11(8):504–521, Oct. 1993.

[11] G. Salton. *Automatic Text Processing - the Transformation, Analysis and Retrieval of Information by Computer*. Addison-Wesley, Readings, MA, 1989.

[12] J. Wu, A. Narasimhalu, B. Mehtre, C. Lam, and Y. Gao. CORE: A Content-based Retrieval Engine for Multimedia Information Systems. *Multimedia Systems*, 3(1):25–41, Feb. 1995.