An Information Retrieval Approach for Image Databases

F. Rabitti and P. Savino

Istituto di Elaborazione dell' Informazione - CNR
Via S. Maria 46, 56100 Pisa (Italy)

Abstract

The retrieval process in image database systems is inherently different from the retrieval process in traditional (record oriented) database systems. While the latter can be considered an exact process (records either satisfy the query or do not so), the former is not an exact process and the system must take into account the uncertainty factor (i.e. the answer is not only "true" or "false" but is often in between them). Uncertainty is mainly introduced during the image analysis process (i.e. when semantic objects are associated, with a certain recognition degree, to image components, and multiple image interpretations are generated) and in the evaluation of how complex objects in images are relevant to user's queries. Moreover, it may be useful to give the user some flexibility in specifying the query on images (i.e. the "importance" of different parts of the query). The possibility to specify imprecise queries in a system can significantly increase the effectiveness in the retrieval process on image databases.

1. Introduction.

The continuing advances in computer technology have made feasible to electronically generate and process increasingly larger quantities of digital images (both pictorial and graphical images). These images must be stored, and sometimes retrieved and processed. This is determining the need of systems able to efficiently manage large database of images.

**Proceedings of the 18th VLDB Conference
Vancouver, British Columbia, Canada 1992**

This situation is encouraging new research efforts for image databases (see [IEEE88] and [Kuni89], for example) and requires to approach the problem of image retrieval with the definition of suitable retrieval models, query languages and access structures. Different query languages and retrieval approaches have been described in literature (for example, [Chan81] for pictorial images, [Rose84] for CAD/CAM images, in [Rabi89] for graphical image retrieval).

In this paper, we are particularly interested in analyzing the problem of *efficient* and *effective* retrieval of images from large image databases.

Two different retrieval approaches can be attempted, i.e. the Database approach and the Information Retrieval approach. Retrieval in Data Base Management Systems 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 on it. It is easy to determine in a DBMS if 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 Information Retrieval approach to document retrieval consists of retrieving all documents whose properties are similar to those present in the query. Historically, Information Retrieval research has focused on the problem of retrieving unstructured text document from large document archives [Rijs79] [Salt83]. Text retrieval techniques can be classified in two broad classes: *exact match* techniques and *partial match* techniques. The *exact match* retrieval techniques provide a basic token matching capability in that only the documents that exactly match with the specified query can be retrieved. The *partial match* retrieval techniques allow to retrieve the documents that match only partially with the query. However the exact match retrieval techniques have some disadvantages [Belk87] since:

a) documents whose representation matches the query only partially are missed

b) retrieved documents cannot be ranked in relevance order

c) the relative importance of concepts either within the query or within the text cannot be taken into account.

These problems can be solved if partial match techniques are used; indeed these techniques are more powerful of the exact match techniques for what concerns the effectiveness of query results. Document retrieval based on partial matching may be viewed, in general, as a process of plausible inference [Rijs86] [Crof89] where the documents that can be plausibly implied by the query are retrieved.

The Information Retrieval approach seems more suitable than the database approach when applied not only to unstructured text documents, but also to structured images.

In case of image databases, queries will be more complex since the image structure can be used in query formulation. The measure of the plausibility of the inference between an image and a query is dependent on the uncertainty of the query representation, on the matching degree between image structure and content and query restriction and also on the uncertainty of the image representation, as resulting from the image analysis process [Rabi91a]. A ranking function, which includes all these sources of evidence may be defined, so that retrieved images can be ranked in decreasing relevance order.

In [Rabi91b] and [Rabi91c] we presented an approach to the retrieval of images from semantic image databases which takes into account different interpretation of images, different levels of recognition of image components, etc. This approach is an extension of the approach used in the implementation of the MULTOS prototype [Cont90]. However, this is essentially a database approach since it adopts a boolean query language, the query processing is based on an exact match between the query and the retrieved images (a cutoff technique is used to determine the "yes or no" answer) and it is not possible to measure the relevance of retrieved images (no ranking is possible).

In this paper we present an approach to image retrieval that allows to take into account the imprecision assigned to the different parts of the query and to rank the retrieved images on the basis of this imprecision and the level of recognition of the corresponding components of the retrieved images.

The paper is so structured: section 2 describes our previous approach in image retrieval, section 3 presents the new approach based on IR techniques, section 4 contains the final remarks.

2. The database approach to image retrieval.

In general, when dealing with images, the retrieval capabilities are strictly related to the possibility to analyze (i.e. *classify*) each image in the database, on the ground of definitional information (i.e. the *schema*) that is dependent on the specific application domain.

An image analysis process, based on a given application domain definition, tries to recognize the objects contained in the images, recording different interpretations, the associated degree of recognition and their position in the image space. Images may contain simple objects (hereafter called *basic objects*) and *complex objects*, which are composed of basic and complex objects. The analysis process tries to determine the composition of the complex objects in terms of simpler objects. (Details on the automatic image analysis process can be found in [Rabi91a] and [Rabi92].)

The symbolic representation of an image *I* is one ISR-DB (Image Symbolic Representation at Database Level, according to the format described in [Rabi91a]) for each application domain selected. The representation of *I* is composed of several image interpretations. Each image interpretation is composed of a set of contexts, each one having several possible interpretations. Each context interpretation contains the basic and complex objects which have been recognized in the image. For each object is given the recognition degree and the positional information and its recursive composition.

In the following discussion we refer to a simple real-world example (Figure 1). The example belongs to an application domain composed of apartment's plants with their furnitures (*ApartmentDesign*). The image analysis allows the recognition of furnitures and types of rooms (e.g. dining room, bathroom, etc.) contained in the images. Furnitures are either basic objects (e.g. table, chair, etc.) or complex objects (e.g. double bed). Rooms are complex objects.

The image query language (described in detail in

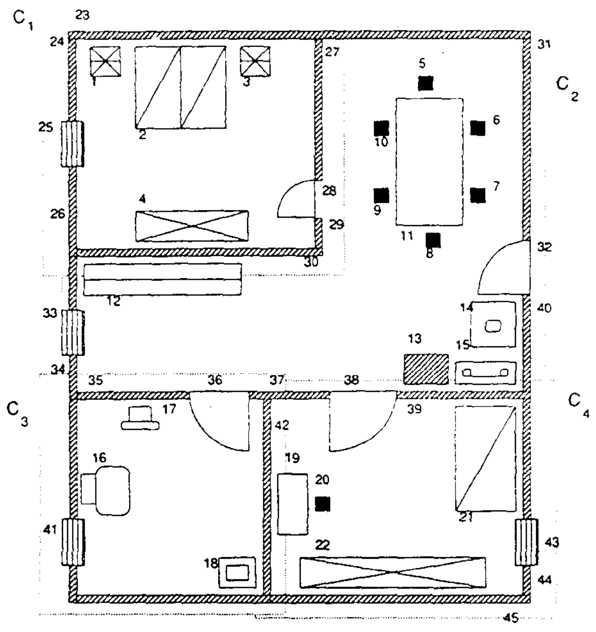

Figure 1: Image example

[Rabi91a] and [Rabi91d]) allows to restrict the query to one or more application domains. Only the images that belong to the specified domain (or domains) will be considered. It also allows to express a Boolean combination (i.e. using AND, OR, NOT operators) of conditions (*object clauses*) on objects to be found in the various image interpretations. An object clause is expressed in *Conjunctive Normal Form*. Quantifiers (i.e. AT MOST, AT LEAST, EXACTLY) can be associated to an object specification. They serve to pose conditions on the number of object instances of the same object type to be found in the same image interpretation. Absolute positional constraints can be associated to an object specification and relative positional constraints can be specified between couples of object conjuncts.

Furthermore, an object specification can be nested, i.e. conditions can be given on the objects composing the required object (i.e. WITH followed by an object clause). This poses conditions on the rules exploited in the recognition of a specific complex object in the image. We must remember that in the application domain definition, several recognition rules can be associated to a specific complex object, leading to alternative object recognitions. The WITH clause allows the user to select one or more specific composition for a complex object in the image.

The main characteristics of this query language are illustrated in the following example:

Example 1:

```
FIND IMAGE IN DOMAIN ApartmentDesign
CONTAINING
  OBJECTS
  (DiningRoom RECOGN 0.5
   AND DoubleBedroom RECOGN 0.8
   AND Bathroom RECOGN 0.7)
  OR
  OBJECTS
  (DiningRoom RECOGN 0.7
      WITH (Kitchenette
            AND Table
            AND AT LEAST 4 Chair
            SUCH THAT
              ((OBJ(1), OBJ(2) ARE S),
               (OBJ(1), OBJ(3) ARE CLOSE)))
   AND EXACTLY 1 SingleBedroom
         RECOGN 0.7 POSITION (0.4, 0.7), (1, 1)
  );
```

The query clause is composed of two object clauses in logical disjunction. The first object clause is composed of three object conjuncts. These object conjuncts require that in one image interpretation at least one complex object ("AT LEAST 1" is assumed if not otherwise specified) is recognized as a *Dining Room* (O_{DIR}), with minimum recognition degree of 0.5, another complex object is recognized as a *Double Bedroom* (O_{DR}), with minimum recognition degree of 0.8, and another complex object is recognized as a *Bathroom* (O_{BR}), with minimum recognition degree of 0.7.

The second object clause is composed of two object conjuncts. The first object conjunct requires that in one interpretation at least one object must be recognized as a *Dining Room* (O_{DIR}), with minimum recognition degree of 0.7, and the second conjunct requires that exactly one object must be recognized as a *Single Bedroom* (O_{SBR}) with minimum recognition degree of 0.7. This means that, while more objects O_{DIR} may be present in the image interpretation, only one object O_{SBR} must be present in the image interpretation to satisfy the object clause. Notice also the absolute positional constraint on the object O_{SBR}, requiring that the object be fully contained in the lower-right part of the image (i.e. within the rectangle determined by the points (0.4,0.7) and (1,1).

Example 1 also shows that the object clauses can be recursive. In general, a condition on the presence of an object (in an object conjunct or a disjunct) may contain another complete object clause (following the keyword WITH). This serves to pose further conditions on the composition of an object, in terms of simpler objects (this also implies restrictions on the rule used to recognize an object). In Example 1, the first conjunct, requiring the presence of a *Dining Room* (O_{DIR}) has a further object clause associated. It means that, in order to satisfy this conjunct, it is not enough that any object O_{DIR} is recognized in the image interpretation, but it is necessary that O_{DIR} satisfies further conditions (i.e. the associated object clause). In particular, it requires that O_{DIR} must be composed, among the other objects, of (at least) one complex object *Kitchenette* (O_{KI}), (at least) one basic object *table* (O_t) and at least four basic objects *chair* (O_c). Moreover, two positional constraints are defined on the objects composing O_{DIR}: (1) OBJ(1), i.e. the object O_{KI}, must be positioned South (this is a way to specify directions in 2D images) with respect to OBJ(2), i.e. the object O_t; (2) OBJ(1), i.e. the object O_{KI}, must be *close* to OBJ(3), i.e. each of the objects O_c. Note that in our example application domain, we define two objects as *close* if the distance of their enclosing rectangles (on either X or Y axis) is lower than 1/4 of the image dimension (on the corresponding axis).

Table 1. Summary of symbols used.

Symbol	Definition
I	Image
T	Image interpretation
C	Context
R	Context interpretation
Q	Query
OC	object clause
dom	Domain
SD(Q)	Set of application domains mentioned in Q
SO(Q)	Objects (basic and complex) in Q
L(dom)	Objects (basic and complex) of the domain
RDB	Set of all ISR-DB in the Image Database
RDB(dom)	Set of ISR-DB belonging to dom

The query Q is executed according to the following steps:

1) Parse query Q (a corresponding *parse tree* is generated).
2) The set of all domains specified in the query (D') is determined.
   $D' = SD(Q)$
3) From the set D' all domains that are not compatible with the query, are removed. All domains that do not contain some of the objects mentioned in the query are eliminated. The resulting set D'' is as follows:
$$D'' = \left\{ dom \in D' \text{ such that } SO(Q) \subseteq L(dom) \right\}$$
4) Use the filtering technique, based on multi-level image signatures [Rabi91c], in order to determine the query answer set
   a) $filter\text{-}answer = \varnothing$
   filter-answer is the query answer set determined on the base of the access methods adopted to speed-up the query processing. It contains a set of ISR-DB. It must be observed that each ISR-DB contained in *filter-answer* can be composed of a subpart of the corresponding ISR-DB in the image DB.
   b) *for each* **dom** $\in D''$
$$filter\text{-}answer = filter\text{-}answer \cup QueryFilter(Q, dom)$$
   where QueryFilter(Q,dom) is a procedure that returns the set of ISR-DB $\in$ RDB(dom) passing the filter corresponding to query Q, when executed on the image access structures adopted in this image retrieval approach. The procedure is described in [Rabi91c] and [Rabi91d].
5) Remove the false drops from the images contained in *filter-answer* and check for the positional constraints.
   a) $answer = \varnothing$
   answer is the query answer set
   b) *for each* **ISR-DB** $\in$ *filter−answer*
      if $QueryProc(Q, ISR\text{-}DB)$ *then*
         $ISR\text{-}DB \rightarrow answer$
   where QueryProc(Q,ISR-DB) is a procedure that returns a boolean value. It returns **True** if Q is satisfied on ISR-DB, otherwise it returns **False**. The procedure QueryProc removes the false drops, verifies the positional and cardinal constraints and verifies again all query conjuncts. This procedure is described in [Rabi91d].

3. The IR approach to image retrieval

Retrieval of images is more complex than retrieval of formatted data in a DBMS. This is because images are more complex than traditional database objects, and they are subject to different, and sometimes conflicting, interpretations. Moreover, the user has usually only an imprecise knowledge of the characteristics of images he/she is seeking. It is also difficult, with the features offered by the available query languages, to express queries that discriminate precisely between relevant and non relevant images.

It is however possible to improve the quality of retrieval: indeed, it has been observed that the quality of the output of the retrieval process is strictly dependent from the quality of the input, i.e. the query. This means that a query language should offer to the user the possibility of expressing as much as possible the knowledge he/she has on the characteristics of the documents he/she is seeking [Crof88] [Morr87] [Crof90] [Morr90].

In section 2 we have presented a query language for semantic image databases, which has been implemented in a prototype image analysis and retrieval system. This image query language addresses important aspects of the image interpretations resulting from image analysis, by defining partial conditions on the composition of the complex objects, requirements on their degree of recognition, and requirements on their position in the image interpretation. However, in this image query language it is not possible to express uncertain knowledge the user has about the images to be retrieved. Furthermore, the query language assumes an exact match between the query and the retrieved images. The expressive power of this query language can be enhanced by allowing to express user uncertainty explicitly in the query and allowing the retrieval of images even if a partial match between the query and the image exists.

Query components may have different *importance* for the user (e.g., with reference to Example 1, it could be more important that retrieved documents satisfy the clause requiring the presence of a *DiningRoom, DoubleBedroom and BathRoom* than the clause requiring the presence of a *DiningRoom* composed of a *Kitchenette*, etc.) and the positional constraint (absolute or relative) associated to a query component could have different *preference* (e.g., again with reference to Example 1, the user may remember that the *Kitchenette and the Table are South* or that they are *South-Est*; however, he/she may prefer the first possibility). The *preference* values and the *importance* values, combined with the recognition degree of the corresponding objects in the various image interpretations, can then be used for ranking the retrieved images.

After the retrieval operation, the set of images may then be presented to the user as an ordered list. The ordering is given by a "score" associated to each image. This score gives a measure of the matching degree between each image and the query. It is obtained as a combination of preference and importance values associated to each predicate in the specified query [Rijs79] [Crof81].

3.1. Query Language

We now present more in detail the extensions necessary to the query language that allows for the formulation of imprecise queries. The complete syntax of the new query language is given in Appendix A.

The extended image query language borrows most of the features of the language described in section 2 (and in [Rabi91a] and [Rabi91d]). A query can be expressed as follows:

```
FIND <number-of-images> IMAGE
     <domain-clause>
CONTAINING <query-clause>
```

The possibility of limiting the number of retrieved images through the <number-of-images> value, is useful to limit the image overloading, typical of classical boolean query languages (it must be noted that this is possible because the retrieved images are ranked in decreasing relevance order).

A `<query-clause>` is a list of `<query-term>` each one having the form `OBJECTS <object-term> <importance>`, where `<importance>` can assume a real value in the range [0,1] or one of the symbolic values HIGH, MEDIUM, LOW (these values are associated to real numbers in the range [0,1]) and `<object-term>` is a list of `<object>` with, possibly, a list of `<constraint>` among the objects in the list. For each `<object>` in the list it is possible to specify the minimum recognition degree it must have in order to be acceptable, and/or an absolute positional constraint. Furthermore, in case of complex objects, an object specification can be nested, i.e. conditions can be given on the objects composing the required object. For each `<constraint>`, it is possible to specify a `<preference>`, that can assume a real value in the range [0,1] or one of the symbolic values PREFERRED, ACCEPTABLE (these values are associated to real numbers in the range [0,1]).

The main differences between the query language adopted for the database approach (QL-DB) to image retrieval and that adopted for the IR approach (QL-IR), are as follows:

- The QL-IR is not based on a boolean logic.
- The QL-IR allows to limit the number of retrieved images. Furthermore, it fits better that the QL-DB for a partial match between the query and the retrieved images.
- The QL-IR allows to associate an importance value to the different object clauses.
- The QL-IR allows to associate a preference value to the different positional constraints.

The query used as example in section 2 can now be expressed as in the following:

Example 2:

```
FIND 30 IMAGE IN DOMAIN ApartmentDesign
CONTAINING
  OBJECTS
  (DiningRoom RECOGN 0.5 ,
   DoubleBedroom RECOGN 0.8 ,
   Bathroom RECOGN 0.7) IMPORTANCE HIGH
  OBJECTS
  (DiningRoom RECOGN 0.7
     WITH (Kitchenette ,
           Table ,
           Chair ,
           SUCH THAT
             ((OBJ(1), OBJ(2) ARE S)
                       PREFERENCE PREFERRED,
              (OBJ(1), OBJ(2) ARE SE)
                       PREFERENCE ACCEPTABLE,
              (OBJ(1), OBJ(3) ARE CLOSE)))
   SingleBedroom ,
          RECOGN 0.7 POSITION (0.4, 0.7), (1, 1)
  ) IMPORTANCE MEDIUM;
```

It can be observed that the query language allows to express the uncertainty the user has about the composition of the images he/she is searching: are they composed of a *DiningRoom, a DoubleBedroom and a Bathroom* or are they composed of a *DiningRoom* (containing a *Kitchenette, a Table and 4 Chair*) and a *SingleBedroom*? Both solutions are possible, but the user specified that it is more important that the retrieved images satisfy the second one. Furthermore, the user is not sure about the relative position of the *Kitchenette* and the *Table*: is the first positioned South with respect to the second or it is positioned South-Est? The user is only able to express the fact that the first possibility is preferred.

3.2. Query Processing

In the query language defined in the previous subsection, a query Q can be expressed as a sequence of n *object-clauses* (*OC*):

$$Q = OC_1, \dots, OC_n \tag{1}$$

Each *object-clause* is composed of an *object-term* and an *importance* value, so that it can be expressed as follows:

$$OC = OT\ im \tag{2}$$

where OT is the *object-term* and im is the importance value.

The *object-term* is composed of a list of objects (*OBJ*) either basic or complex, plus a set of *positional constraints* (*PC*) among the objects.

$$OT = OBJ_1, \dots, OBJ_m \text{ } SUCH\ THAT\ PC_1, \dots, PC_r \tag{3}$$

where m is the number of objects present in the object term and r is the number of positional constraints (these numbers are different for each object-term).

The object *OBJ* can be expressed in one of the following two forms:

$$OBJ = \begin{cases} case\ 1: & O \\ case\ 2: & O\ WITH\ OC \end{cases} \tag{4}$$

where O is the name of an object. The (4) specifies that an object specification can be recursive.

As underlined in the introduction, multiple sources of evidence can be used to measure the relevance of an image for the query. The language we are considering allows to take into account two different sources of evidence: the first one is directly dependent on the uncertainty of query representation and it is provided by the user through the importance and preference values; the second one depends on the uncertainty of image representation, with objects recognized with a certain recognition degree. Furthermore, the semantics associated to the query language implies a third source of evidence which is given by the partial match between the query and the retrieved images. Then, the ranking function, which measures the relevance of each retrieved image, is dependent from the recognition degree of each object present in the image, the importance of each query term and the preference assigned by the user to each positional constraint that is verified. This function (g) is reported in the following; it has a form which is similar to the ranking functions used in the retrieval of text documents.

$$g = \sum_{i=1 \mid OC_i \text{ is true}}^{n} c_i \times im_i \tag{5}$$

In the equation (5), only the object clauses that evaluated to true (i.e. that are verified) contribute to the sum. An object clause OC is verified in an image I, if at least one of the objects contained in OC is also present in I and at least one of the positional constraints (eventually present in OC) is verified. The ranking function depends from the importance values of each object clause in the upper level of the query (those present in relation (1)). However, we must remember that each object clause is composed of a list of objects, with a list of possible positional constraints. The composition of each object can be specified through an object clause (O WITH OC). c_i takes into account all these aspects. It is expressed with the following formula:

$$c = pr \times \sum_{i=1 \mid OBJ_i \text{ is true}}^{m} RD_i \times d_i \tag{6}$$

where

- *pr* is the preference value associated to the positional constraint that has been verified ($pr=1$ if no cardinal constraint is present in the object clause or if no preference value is specified).
- RD_i is the recognition degree associated to the object OBJ_i in the image.
- d_i takes into account the possibility that a WITH condition has been specified for OBJ_i (case 2 of relation (4)).

$$d_i = \begin{cases} case\ 1: & 1 \\ case\ 2: & c \times im \end{cases} \tag{7}$$

Equation (7) requires that equation (6) must be used to calculate c, since c is the measure of the relevance on an object term. This value must be multiplied by the importance value of the object clause.

The query execution is based on an algorithm which is similar to that described in section 2. The differences between the two algorithms are in steps 4 and 5.

Step 4 is modified since the procedure *QueryFilter* determine all images that may verify (apart the false drops and positional constraints) at least one condition on an object *OBJ* (*OBJ* has been defined in relation (4)). *QueryFilter* returns the *filter-answer*, which contains a set of ISR-DB', where a ISR-DB' consists in a reduced form of each ISR-DB satisfying the query filter. A ISR-DB' is obtained from ISR-DB pruning the branches of its definition tree which do not pass the filter, and therefore do not satisfy the query. The new version of the procedure *QueryFilter* is described in the next subsection.

Step 5 is modified as follows.

5) Remove the false drops from the images (i.e., their reduced representation ISR-DB') contained in *filter-answer*, check for the positional constraints and evaluate the ranking function g.

   a) $answer = \emptyset$
      answer is the query answer set

   b) *for each* **ISR-DB'** $\in$ *filter-answer*
      if $QueryProc\,(Q, ISR\text{-}DB') > 0$ *then*
      $ISR\text{-}DB \rightarrow answer$

   where QueryProc(Q,ISR-DB') is a procedure that returns a real value. This is equal to 0 if Q is not satisfied on ISR-DB', otherwise it returns the value of the ranking function g calculated for the ISR-DB'. The procedure QueryProc removes the false drops and verifies the positional constraints.

3.3. Filtering process

The access structure for the image retrieval based on exact match has been described in [Rabi91]. It must be observed that the adoption of the IR approach to image retrieval implies that the query Q is verified even if only a single object clause is verified. Furthermore, an object clause is verified even if a single object *OBJ* that composes the object clause, is present in the image. We will also assume that if *OBJ* has the form *O WITH OC* then *OBJ* is verified for an image *I* only when *O* is contained in *I* and *OC* is verified in the image. This discussion implies that the generation of the query level signature and the object-clause level signature using the method described in [Rabi91c], would require to perform the intersection of the signatures of the single objects (*OBJ*) composing the query. This may produce signatures with a poor selectivity, since the intersection tends to reduce the number of bits set to "1" in the signature record. In the following we describe a different method that allows to overcome this problem.

The approach is still based on the *signature technique* [Chri84] and on the method, described in [Rabi91c], for the generation of the signature of a single object. For each application domain, the signature of an object (simple or complex) is fixed as *M* specific bit positions in the complete signature block (*F* bits). The codes of all possible objects in the domain are specified in a look-up table. A simple object (e.g. O_t in the example image) will set only *M* bits in the image signature, as specified in the application look-up table. A complex object, instead, will set its *M* bit position and the bit positions associated with all the simpler objects which compose it in the specific interpretation in the symbolic representation of the image. For example, the signature of O_{DIR} in the image example of Figure 1, is obtained superimposing the codes, obtained from the look-up table, of O_t, O_c, O_{KJ}, O_{gs}, O_{kt}, O_{wb} and O_{sb}.

In the generation of the query signature we will view a query as composed of a list of *r* objects (*OBJ*)

$$Q = OBJ_1, \ldots, OBJ_r$$

where each *OBJ* is defined as in (4).

A signature is generated for each object *OBJ* that is present in Q. In (4) two forms have been considered for *OBJ*; they will produce the following values for the signature of *OBJ* (*SIGN*(*OBJ*)).

$$SIGN(OBJ) = \begin{cases} 1: & code(O) \\ 2: & List_{i=1,\ldots,s} Sup\left[code(O), SIGN(OBJ_i)\right] \end{cases} \tag{8}$$

where $code(O)$ is the code associated to the object O in the lookup table and we consider that the object clause OC is composed of s objects OBJ. Note that $Sup\left\{code(O), SIGN(OBJ_i)\right\}$ is defined as the superimposition (i.e. bitwise OR) of the $code(O)$ and $SIGN(OBJ_i)$ signatures. In case 2 (i.e. when $OBJ=O$ $WITH$ OC) $SIGN(OBJ)$ is given as a list of s signatures, each one obtained as the superimposition of $code(O)$ and the signature of one of the objects OBJ_i composing OC. This produces a recursive definition of $SIGN(OBJ)$.

Let use consider an example that clarifies the method used for the generation of the signature.

We consider the following query:

$$Q = O_1, O_2 WITH(O_3, O_4 WITH(O_5, O_6), O_7)$$

The signature of the query is composed of the following list of signatures:
(a) $code(O_1)$
(b) $Sup(code(O_2), code(O_3))$
(c) $Sup(code(O_2), code(O_4), code(O_5))$
(d) $Sup(code(O_2), code(O_4), code(O_6))$
(e) $Sup(code(O_2), code(O_7))$

The image signature is composed of four levels, as described in [Rabi91c]. An image I is composed of several image interpretations T, each image interpretation is composed of several contexts C and each context contains several context interpretations R. The four level signatures are represented as follows: $SIGN(I)$ is the *image-level signature*, $SIGN(I,T)$ is the *image-interpretation-level signature*, $SIGN(I,T,C)$ is the *context-level signature* and $SIGN(I,T,C,R)$ is the *context-interpretation-level signature*.

The filtering process is executed in the following four steps:

Step 1: The image level signature file $SIGN(I)$ is scanned. Each signature generated from the query is matched against the image-level signatures $SIGN(I)$ for all images I in the domain *dom*. The set S_1 is determined, defined as the set of all images whose $SIGN(I)$ matches at least one of the signatures generated for Q. A query signature matches a data signature if all "one" positions in the query signature correspond to "one" positions in the data signature [Chri84]. S_1 is the first restriction of the image database.

Step 2: For each image I in S_1, determine the set of image-interpretation-level signatures $SIGN(I,T)$. If at least one of the query level signatures matches at least one image-interpretation-level signature $SIGN(I,T)$ then I and the matching image interpretation T are inserted in the set S_2. S_2 is composed of couples (I,T).

Step 3: For each couple (I,T) in S_2, determine the set of context-level signatures $SIGN(I,T,C)$, corresponding to all contexts C of (I,T). If at least one of the query level signatures matches at least one image-interpretation-level signature $SIGN(I,T,C)$ then (I,T,C) is inserted in S_3. S_3 is the third restriction of the image database. (I,T,C) is a triple that represents the context C of image interpretation T of image I.

Step 4: For each context (I,T,C) in S_3, determine the set of context-interpretation-level signatures $SIGN(I,T,C,R)$, corresponding to all context interpretations R of context C. If at least one of the query level signatures matches at least one image-interpretation-level signature $SIGN(I,T,C,R)$ then (I,T,C,R) is inserted in $\bar{S}$. (I,T,C,R) is the context interpretation R of context C of image interpretation T of image I.

$\bar{S}$ is the result of the procedure *QueryFilter*. From $\bar{S}$, it is possible to derive the reduced form of each ISR-DB satisfying the filter, i.e., ISR-DB'. Remember that ISR-DB' is obtained from ISR-DB pruning the branches of its definition tree which do not pass the filter, and therefore do not satisfy the query. The algorithm to construct ISR-DB' for an image $\bar{I}$ is the following:

a) if at least one $SIGN(\bar{I},T,C,R)$ is in $\bar{S}$, then ISR-DB' is defined for $\bar{I}$ (otherwise all $\bar{I}$ has been filtered out);

b) add to ISR-DB' of $\bar{I}$ all image interpretations $\bar{T}$ such that at least one $SIGN(\bar{I},\bar{T},C,R)$ is in $\bar{S}$;

c) add to ISR-DB' of $\bar{I}$, to the corresponding image interpretation $\bar{T}$, all image contexts $\bar{C}$ such that at least one $SIGN(\bar{I},\bar{T},\bar{C},R)$ is in $\bar{S}$;

d) add to ISR-DB' of $\bar{I}$, to the corresponding image interpretation $\bar{T}$ and image context $\bar{C}$, all image context interpretations $\bar{R}$ such that $SIGN(\bar{I},\bar{T},\bar{C},\bar{R})$ is in $\bar{S}$;

The final image query processing (i.e., QueryProc) will then be limited to ISR-DB', saving processing time with respect to the original ISR-DB.

The following general observations can be made when comparing the query filtering described in [Rabi91c] (referred as *method A*) and the query filtering proposed here (referred as *method B*).

- The storage overhead of method A and method B is the same.
- Method A requires the generation of two query signatures, while method B requires the generation of a

variable number of query signature; this number is, in general larger than two.

- The number of I/O operations, needed for reading the image signature files, is the same in both methods.
- Method B is slightly less efficient than method A, since a larger number of signature records must be compared with the image signatures.

3.3.1. Query execution example

The query execution process can be better understood through the use of a running example.

Let us consider the query of Example 2. We suppose that the importance levels HIGH, MEDIUM and LOW have, respectively, the following associated values (0.9,0.6,0.3), while the preference values are 1 for PREFERRED and 0.6 for ACCEPTABLE. The tuning of these values should be taken into account in order to improve the effectiveness of retrieval (this tuning could be also done automatically, with a relevance feedback process).

The query processing is as follows. After the parsing of the query, steps 2) and 3) are executed. The result is $D'' = ApartmentDesign$. Then the signature of the query is generated. It is composed of the following signature records:
code (DiningRoom)
code (DoubleBedroom)
code (Bathroom)
Sup (code (DiningRoom), code (Kitchenette))
Sup (code (DiningRoom), code (Table))
Sup (code (DiningRoom), code (Chair))
code (SingleBedroom)
The four level signatures are scanned in order to remove most of the images that does not match with the query. The remaining images compose the *filter-answer*.

For each image belonging to the *filter-answer* the procedure *QueryProc* is executed.

QuerProc accesses directly the image and compares it with the query in order to remove the false drops and to verify the positional constraints. For each remaining image the ranking function g is evaluated.

Let us evaluate the value of g for the following four images.

Image 1

I_1 `= DiningRoom (RD=0.6),`
`DoubleRoom (RD=0.9)`

$$g(I_1) = 1\times(0.6 + 0.9)\times0.9 = 0.9$$

Image 2

I_2 `= DiningRoom (RD=0.6)`

$$g(I_2) = 1\times0.6\times0.9 = 0.54$$

Image 3

I_3 `= DiningRoom (RD=0.7) COMPOSED-OF`
`Kitchenette (RD=0.6),`
`Table (RD=0.5)`

where *(Kitchenette,Table) ARE South*

$$g(I_3) = 1\times0.7\times0.9 + (1\times(0.7\times(1\times(0.6 + 0.5))))\times0.6 = 1.092$$

Image 4

I_4 `= DiningRoom (RD=0.7) COMPOSED-OF`
`Kitchenette (RD=0.6),`
`Table (RD=0.5),`
`Chair (RD=0.7)`

where *(Kitchenette,Table) ARE South-Est.*

$$g(I_4) = 1\times0.7\times0.9 + (1\times(0.7\times(0.6\times(0.6 + 0.5 + 0.7))))\times0.6 = 1.086$$

The following general observations can be made when comparing the values of g for the four images.

- $g(I_1) > g(I_2)$ as we expected, since I_1 contains two objects present in the query clause, while I_2 contains only one.
- $g(I_3) \approx g(I_4)$. Indeed, even if I_3 verifies the preferred positional constraint it has less objects that match the objects in the query than I_4.

4. Final Remarks

The problem of managing uncertainty for attaining better results in retrieving images from semantic image databases is discussed in this paper. The user can specify imprecise queries to express his/her uncertain knowledge of content of images he/she is seeking. We presented also the techniques used for evaluating the relevance of image components in relation to the user's query. The combined use of these techniques allows the ranking of the retrieved images in order of relevance to the query. This leads to a better understanding by the system of the user needs.

A prototype system, based on this IR approach, is being implemented. It is based on a previous image analysis and retrieval prototype, which is an extension of the image handling component of MULTOS [Cont90}. The aspects regarding the image semantic analysis are described in [Rabi91a] and [Rabi92], while the aspects regarding the query processing and retrieval are described in [Rabi91c] and [Rabi91d]. In this prototype, query processing was based on exact match. In the new prototype the image analysis function need not be modified, while, the image retrieval function is based on the query processing module described in this paper. It is based on a multi-level signature technique for fast content-based access to the database of images [Rabi91c]. In addition, a new module is being implemented according to the techniques presented in this paper. This module will perform the computation of the "score" associated to each image interpretation, obtained from the previous query processing module, and will rank the images in the query answer according to the computed score, limiting the answer size according to the user request.

Acknowledgments

This work has been funded in part by the ESPRIT Basic Research Action Project No. 3070, FIDE (Formally Integrated Data Environment) and in part by Progetto Finalizzato Sistemi Informatici e Calcolo Parallelo, Sottoprogetto 5.

REFERENCES

[Belk87] Belkin N.J. and Croft W.B., "Retrieval Techniques", Annual Review of Information Science and Technology (ARIST), Vol. 22, 1987, pp. 109-145.

[Chan81] Chang N., Fu K., "Picture Query Languages for Pictorial Data-Base System", IEEE Computer Vol.14 (Nov. 1981)

[Chri84] Christodoulakis S. and Faloutsos C., "Signature Files: An Access Methods for Documents and its Analytical Performance Evaluation", ACM Transactions on Office Information Systems, Vol. 2, N. 4, pp. 267-288 (1984).

[Cont90] Conti P., Rabitti F., "Image Retrieval by Semantic Content", in "Multimedia Office Filing and Retrieval: The MULTOS Approach", edited by C. Thanos, North-Holland Series in Human Factors in Information Technology, North-Holland, 1990.

[Crof81] Croft W. B., "Document Representation in Probabilistic Models of Information Retrieval", Journal of the American Society of Information Science, Vol. 32, 1981, pp. 451-457.

[Crof88] Croft W. B. and Krovetz R., "Interactive retrieval of office documents", Proc. Conf. on Office Information Systems, 1988, pp. 228-235.

[Crof89] Croft W.B., Lucia T.J., Cringean J and Willett P., "Retrieving documents by plausible inference: an experimental study", Information Processing & Management, Vol. 25, No. 6, pp. 599-614, 1989

[Crof90] Croft W.B., Krovetz R. and Turtle H., "Interactive retrieval of complex documents", Information Processing & Management, Vol. 26, No. 5, pp. 593-61, 1990

[Kuni89] "Visual Database Systems", edited by T. L. Kunii, North-Holland (1989).

[IEEE88] Special Issue on Image Databases, IEEE Trans. on Software Engineering, Vol. 14, No. 5, pp. 630-638 (May 1988)

[Morr87] Morrissey J.M. and Van Rijsbergen C. J., "Formal Treatment of Missing and Imprecise Information", Proc. of the 10th International ACM-SIGIR Conference on Research and Development in Information Retrieval, 1987, pp. 149-156.

[Morr90] Morrissey J. M., "Imprecise Information and Uncertainty in Information Systems", ACM Transactions on Information Systems, Vol. 8, No. 2, pp. 159-180, April 1990

[Rabi89] Rabitti F. and Stanchev P., "GRIM_DBMS: a GRaphical IMage Data Base Management System" Proc. IFIP TC-2 Working Conference on Visual Database Systems, Tokyo (April 1989), in "Visual Database Systems", edited by T. L. Kunii, North-Holland, pp. 415-430 (1989).

[Rabi91a] Rabitti F. and Savino P., "Automatic Image Indexation and Retrieval", Proc. RIAO'91, Barcelona, Spain, April 2-5, 1991.

[Rabi91b] Rabitti F., Savino P., "Query Processing on Image Databases", Proceedings of 2nd Working Conference on Visual Database Systems, IFIP WG 2.6, Budapest, Hungary, Sept. 30 - Oct.3, 1991, pp. 174-188.

[Rabi91c] Rabitti F., Savino P., "Image Query Processing Based on Multi-level Signatures", Proceedings of ACM SIGIR'91, International Conference on Research and Development in Information Retrieval, Chicago, Illinois, October 13-16, 1991, pp. 305-314.

[Rabi91d] Rabitti F., Savino P., "Content Based Retrieval in Semantic Image Databases", IEI-CNR Tech. Rep. B4-49, Pisa, Dec. 1991.

[Rabi92] Rabitti F., Savino P., "Automatic Image Indexation to Support Content Based Retrieval", Information

Processing & Management, Vol. 28, No. 4. Pergamon Press, New York, 1992.

[Rijs79] Van Rijsbergen C.J., "Information Retrieval", McGraw-Hill, London, 1979.

[Rijs86] Van Rijsbergen C.J., "A Non-Classical Logic for Information Retrieval", Computer Journal, Vol. 29, 1986, pp. 481-485.

[Rose84] Rosenthal A., Heiler S., Manola F., "An Example of Knowledge-Based Query Processing in a CAD/CAM DBMS", Proc. Tenth Int. Conf. on VLDB, Singapore, pp. 363-370 (1984).

[Salt83] Salton G. and McGill M., "Introduction to Modern Information Retrieval", McGraw-Hill, London, 1983.

APPENDIX A
Query Language

```
<query> := FIND <number-of-images>
             IMAGE <domain-clause>
             CONTAINING <query-clause> ;
<number-of-images> :=
                     | Integer
<domain-clause> :=
           | IN ALL DOMAINS
           | IN DOMAIN <domain-list>
<domain-list> := <domain-name>
       | <domain-list> , <domain-name>
<domain-name> := Identifier
<query-clause> := <query-term-list>
<query-term-list> := <query-term-list> ,
                     <query-term>
                   | <query-term>
<query-term> := OBJECTS <object-clause>
<object-clause> := <object-term>
                   <importance>
<importance> := | IMPORTANCE <level>
                | IMPORTANCE VALUE Real01
<level> := HIGH
         | MEDIUM
         | LOW
<object-term> := ( <object-list> )
       | ( <object-list>
         SUCH THAT ( <constraint-list> ) )
<object-list> := <object-list> , <object>
               | <object>
<object> := <obj-condition>
          | <obj-condition>
            WITH <object-clause>
<obj-condition> := <object-name>
           <obj-requirement>
                | <object-name>
<object-name> := Identifier
<obj-requirement> := <min-recognition>
           ( <abs-position-list> )
                  | <min-recognition>
                  | ( <abs-position-list> )
<min-recognition> := RECOGN Real01
<abs-position-list> := <abs-position-list> ,
        <abs-position-pred> <preference>
          | <abs-position-pred> <preference>
<preference> := | PREFERENCE <pref-level>
                | PREFERENCE VALUE Real01
<pref-level> := ACCEPTABLE
              | PREFERRED
<constraint-list> := <constraint>
          | <constraint-list> , <constraint>
<constraint> := ( <binary-constr> )
          <preference>
        | ( <group-constr> ) <preference>
<binary-constr> := <obj-var> , <obj-var>
         ARE <bin-position-pred>
<group-constr> := <obj-var> , <obj-var-list>
         ARE <group-position-pred>
<obj-var-list> := <obj-var>
                | <obj-var-list> , <obj-var>
<obj-var> := OBJ ( Integer )

NOTE: <abs-position-pred>, <bin-position-pred> and
<group-position-pred> are dependent on the
particular image application domain.
For 2D images, they could be:

<abs-position-pred> := BC POSITION <encl-rect>
                     | POSITION <encl-rect>
<encl-rect> := <left-high-corner>
          <right-low-corner>
<left-high-corner> := ( Real01, Real01 )
<right-high-corner> := ( Real01, Real01 )
<bin-position-pred> := <direction> <distance>
                     | <direction>
                     | <distance>
<direction> := N
             | S
             | E
             | W
             | NE
             | NW
             | SE
             | SW
<distance> := CONTIG
            | CLOSE
            | FAR
<group-position-pred> := <distance>
```