SEMCOG: A Hybrid Object-based Image Database System and Its Modeling, Language, and Query Processing

Wen-Syan Li

C&C Research Laboratories
NEC USA, Inc., MS/SJ100
San Jose, CA 95134
wen@ccrl.sj.nec.com

K. Selçuk Candan*

Computer Science and Engineering Dept.
Arizona State University, Box 875406
Tempe, AZ 85287-5406
candan@asu.edu

Abstract

Image data is structurally more complex than traditional types of data. An image can be viewed as a compound object containing many sub-objects. Each sub-object corresponds to image regions that are visually and semantically meaningful (e.g. car, man, and etc). We introduce a hierarchical structure for image modeling that supports image retrieval, at both whole-image and object levels, using combinations of semantics expressions and visual examples. We present formal definition of a multimedia query language, we give details of the implementation and query processing, and we discuss our methods for merging similarities from different types of query criteria.

1 Introduction

We categorize existing approaches to image retrieval as follows: (1) Browsing and navigation approach: users are presented with directories where each directory contains a single category of images (e.g. business, people, and sports); (2) Syntactic keyword-based approach: users input keywords and are presented with the images whose captions or keyword lists contain specified keywords; (3) Descriptive semantics-based approach: users specify semantics of the image for retrieval. The weakness of this approach is its low visual expressive capability; images are visual and hard to describe using text alone; and (4) Cognition-based approach: Users pose queries by providing drawings or image examples. The system retrieves images which are visually similar to the example provided by the user. One disadvantage of this approach is its lower precision due to the lower precision of users' drawings and image recognition techniques. Another weakness of this approach is that it cannot support queries on generalized concepts, such as transportation.

An image can be viewed as a compound object containing many components (sub-objects). Each component corresponds to a region which is visually and semantically meaningful (e.g. car, man). Since images have both visual and semantical properties, image retrieval using either one of them alone is not sufficient. Since images have hierarchical structures, both whole image and object-based image matching are essential to provide more flexible image retrieval. However, most existing systems only support image retrieval based on visual characteristics and/or semantic meanings of *whole* images. As a result, users cannot query objects within an image nor have flexible query specification means.

The rest of this paper is organized as follows: We first present our image model and give an overview of our approach. Section 3 presents a system architecture for implementing this image modeling. In Sections 4, we give the syntax of an image query language based on SQL3, used in our system. Section 5 discusses our query processing techniques. In Section 6 we summarize related work in this area and provide comparison with SEMCOG. In Section 7, we give our conclusions.

2 Hierarchical Image Modeling for Object-based Media Retrieval

Most existing approaches treat an image as a whole entity for image matching.

Our approach, on the other hand, is to model images using a hierarchical structure that supports queries at both image and object levels. Figure 1 shows an image *Boy_bike.gif*, whose structure is viewed as a compound image with two image objects (i.e. boy and bicycle) based on color/shape region division and their spatial relationship. After image processing

*This work was performed when the author visited NEC, CCRL.

0-8186-8289-2/98 $10.00

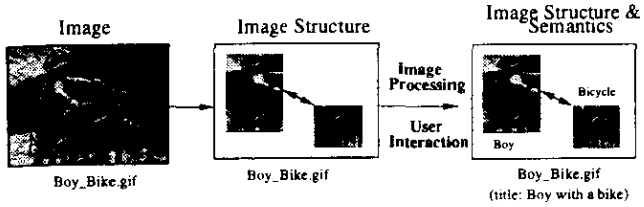


Figure 1: Image Structure and Semantics

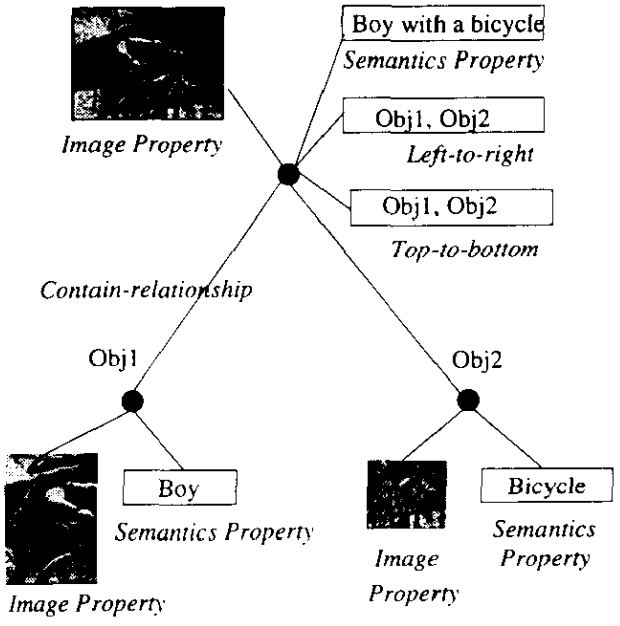


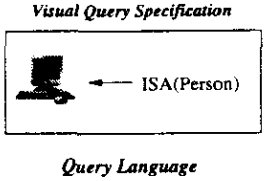
Figure 2: Hierarchical Image Modeling

and/or user interaction, semantics for the objects are assigned.

To represent the image structure, we model an image as a compound object with (1) image property, (2) semantics property, and (3) spatial relationships (left-to-right and top-to-bottom) as shown on the top of Figure 2. Image objects also have both image properties and semantics properties. Semantics of the image and objects are generated or specified after image processing and/or user interaction.

In this paper, we introduce an image database system, SEMCOG (SEMantics and COGnition-based image retrieval). SEMCOG aims at integrating semantics and cognition-based approaches and allows queries based on object-level information. For instance, a query of the form "retrieve all images in which there is a person who is to the right of an object that looks like image X" can be specified as shown on Figure 3. In this query, three types of sub-queries are involved: semantics-based, cognition-based, and scene-based. The user's query can be visualized as a graph as shown on the top left hand side of Figure 3.

Figure 3 shows two candidate images. The arrows



Select image P

where P contains X and P contains Y

(1) X isa person (Semantics-based Query)

(2) Y i_like (Cognition-based Query)

(3) X to_the_right_of Y (Scene-based Query)

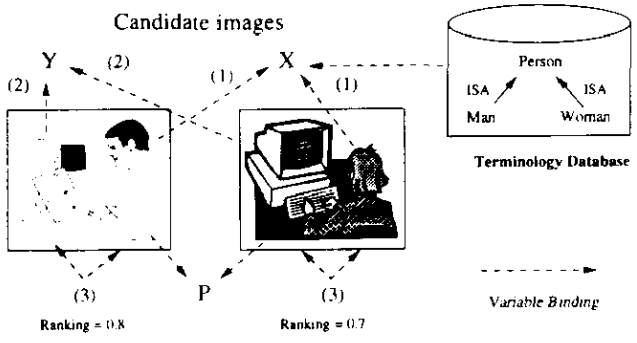


Figure 3: Example of Query Processing in SEMCOG

indicate how the variables X, Y, and P are bound with images and component objects. In this case, (X, Y) is bound based on spatial relationship comparisons. The candidate images are presented to users as a set of images ranked by their *relevancy*. The relevancy is calculated by matching both semantical and visual characteristics of component objects and their spatial relationships (layout) with user's query. In Section 5.2 we describe how we compute the relevancy values.

Note that, in the above example, the *ISA* predicate is relaxed: the two candidate images contain a man and a woman respectively, rather than "person". Our system features a terminology database to store hierarchical structure of concepts, shown on the right of Figure 3, for query relaxation.

3 System Architecture

Figure 4 shows the operation and data flows in SEMCOG. Three layers of functions are provided. The client functions are supported by IFQ visual query interface, including query specification, result display, and query generating. The server functions are supported by five modules that augment the underlying database system by media manipulation and media search capabilities. SEMCOG also maintains

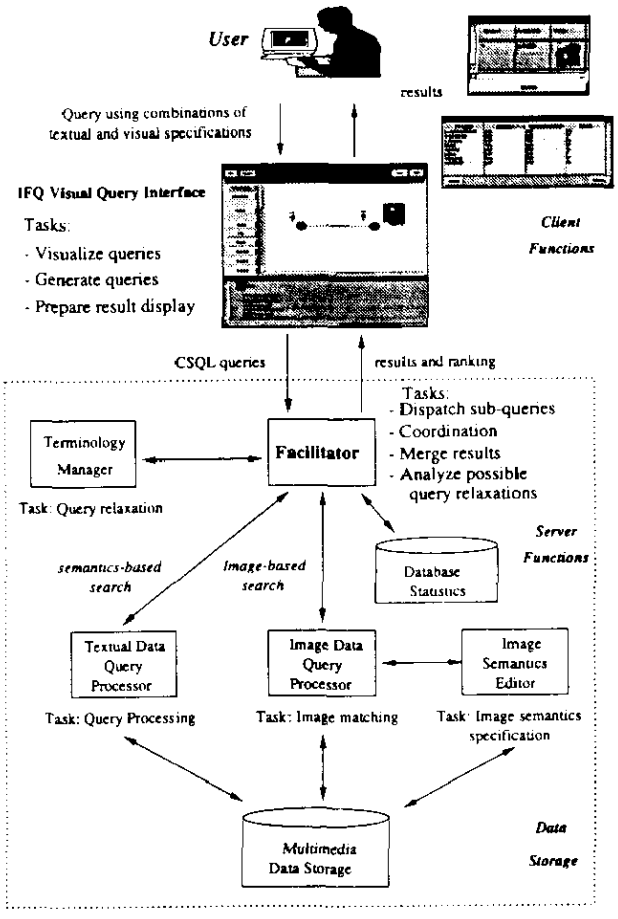


Figure 4: System Architecture of SEMCOG

database statistics for query optimization and query reformulation facilitation. The components of SEMCOG and their functionalities are as follows:

Image Data Query Processor is used to perform image matching task. COIR (Content-Oriented Image Retrieval)[6] is the image processing module employed in SEMCOG. It performs image matching at the object level by identifying regions defined as homogeneous (in terms of colors) and continuous segments identified in an image. The main task of COIR is to identify image regions. To identify objects within an image and their semantics, COIR needs to consult *Image Component Catalog* to match with existing image object samples to determine image semantics.

When an image is *registered* at SEMCOG, some image descriptions, such as the image size, image format, registration date, are automatically extracted; whereas some metadata, such as semantics of images, cannot be extracted automatically. *Image Semantics Editor* interacts with COIR to specify the semantics of an image.

Note that SEMCOG stores both locations of objects and their *relative spatial relationships* (horizontal and vertical) as described in 2-D strings[10]. With 2-D strings, we convert relative-spatial-relationship-based queries into text-based queries without computing objects' spatial relationship during the query processing.

Terminology Manager maintains a terminology base that is used for query term relaxation. Consulting with *Terminology Manager*, SEMCOG can find synonyms for a given term and access IS_A hierarchies. For predicates, such as "*is_a transportation*", the *Terminology Manager* returns multiple resulting terms ranked according to their relevancy values. We do not need to build a terminology database. Many existing lexical reference systems, such as WordNet[8], can be used.

Textual Data Query Processor processes queries concerning image or object semantics, which are "non-image-matching operations". All query statements except those containing *i_like* (image like) and *contains* are semantics-based. The query processing performed at this module is the same as that in traditional relational DBMSs, except the additional task associated with query relaxation predicates, such as *is_a*.

We design SEMCOG as a hybrid architecture so that many modules can be plugged in as components to handle multiple media. *Facilitator* coordinates the interactions between components. It performs the following tasks: (1) Query relaxation: It reformulates query statements containing *is_a* or *s_like* predicates by consulting the *terminology manager*; (2) Query forwarding: It forwards query statements containing *i_like* or *contains* (visually) predicates to the *Image Data Query Processor* to perform image matching while forwarding other non-image-matching operation statements to *Textual Data Query Processor*; and (3) Result integration: It merges results of image-matching query statements and non-image-matching operation statements. This task includes computing overall similarity values and eliminating images based on specified filtering constraints. Details on query processing technique in SEMCOG is given in Section 5.

In addition to these components, SEMCOG provides a visual query interface, IFQ[7], for query specification, query generation, result display, and feedback visualization. With IFQ, queries are posed by specifying image objects and their layouts using the interface using icons and menus in a drag-and-drop fashion,

4 Query Language

In this section, we introduce the syntax of CSQL (Cognition and Semantics-based Query Language). Traditionally SQL is used to manipulate only textual

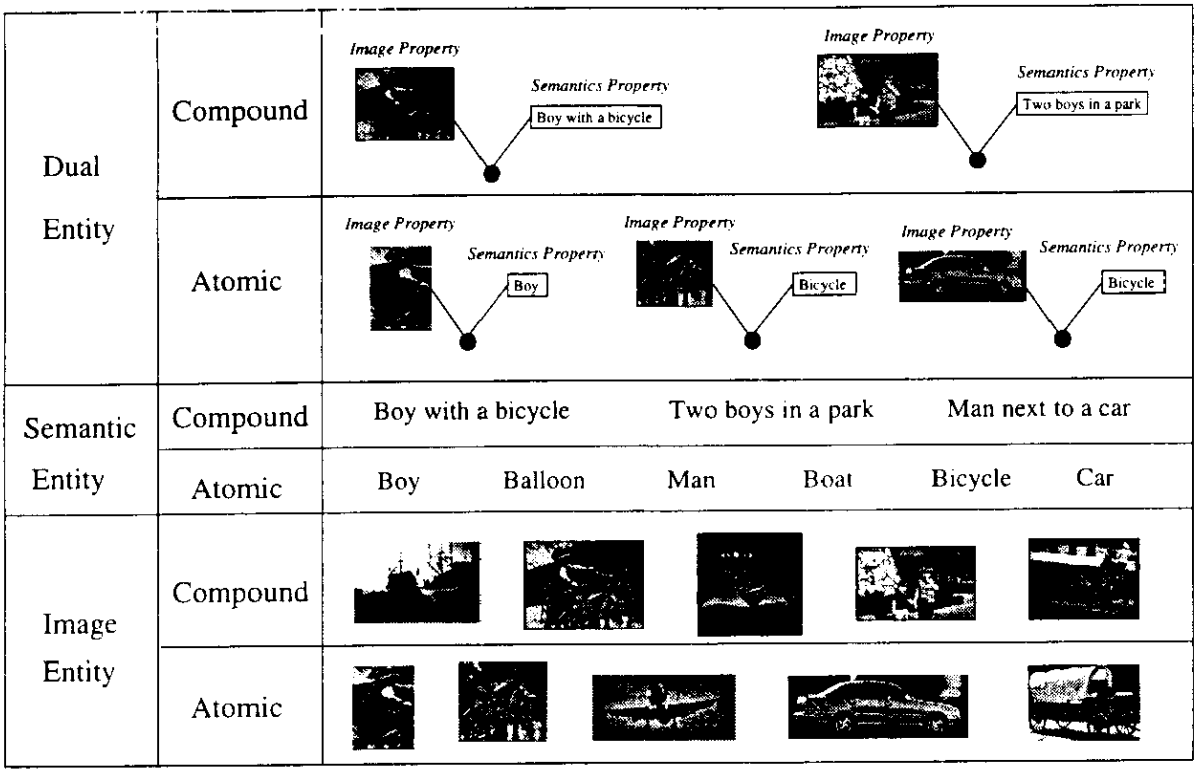


Figure 5: Classification of Entities and Examples

data stored as tables. SQL3 extends SQL to support user defined data types and user defined functions to manipulate user defined data types. We have implemented CSQL on SQL3. We start with giving definitions of terminology used in CSQL.

4.1 Property Classification

An image or an object can be interpreted based on its semantics and/or image properties. A semantic interpretation of an image is the image's *real world meaning*. A visual interpretation of an image is how it looks like based on human perception or image matching engines. Ideally the semantic interpretation and visual interpretation of an image should be consistent. However they may be inconsistent because of the limitations of the image recognition technique used.

4.2 Entity Classification

There are three types of entities used in CSQL (each entity can be further classified into *compound* and *atomic*):

Semantic entities: A semantic entity is an entity with only *semantic property*. Semantic descriptions of images/objects, or a semantic term are semantic entities. A *compound semantics entity* is defined as a semantics entity which contains more than one semantics terms. An example compound semantics entity is *a boy with a bicycle*.

Image entities: An image entity is an entity with only *image property*. An atomic image entity is an image entity with only an image object, while a compound image entity is as an image entity with more than one image objects. Other than the semantic descriptions, users can also provide images or sketches as examples to retrieve visually similar images. These input images and sketches are also image entities. Whether or not they are treated as atomic or compound entities depends on the predicate associated with (i.e. being used for whole image matching or object level image matching).

Dual entities: Dual entities are polymorphic entities with both image property and semantics property. Dual entities can be classified in two categories: *atomic dual entities* and *compound dual entities*. Images are stored as compound dual entities.

Examples of these entities are shown in Figure 5.

4.3 Selection Criteria

In this section, we introduce the additional predicates (i.e. user defined functions) to extend SQL for object-based image retrieval. We categorize query selection criteria based on the types of entity and operators involved as follows:

Predicate	1^{st} ar.	2^{nd} ar.	3^{rd} ar.
is	s/d/i	s/d/i	
is_a	s/d/i	s/d/i	
s_like	s/d/i	s/d/i	
i_like	d/i	d/i	
contains	i	d	
to_the_right_of in	s/d	s/d	i
to_the_left_of in	s/d	s/d	i
above_of in	s/d	s/d	i
below_of in	s/d	s/d	i

Table 1: Predicates and Their Usage

Semantics-based selection criteria: Users can specify semantic conditions to retrieve images at different levels of strictness:

- *is*: returns true if and only if both of the arguments have identical semantics (man vs. man).
- *is_a*: returns true if and only if the second argument is a generalization of the first one (car vs. transportation).
- *s_like*: (semantically like) predicate returns true if and only if both arguments are semantically similar (man vs. woman).

Note that the arguments of the above predicates can be semantics, image, or dual entities. In all cases, the comparison is based on their semantic property. If the argument is an image entity, its semantics property must be extracted before the predicate can be executed.

Cognition-based selection criteria: The cognition-based predicate introduced in CSQL is *i_like* (i.e. image like). The two arguments to the *i_like* predicate are either image entities or dual entities. In either case, comparisons are based on their image properties. The input images of the first parameter can be either a whole image or an image object.

Spatial relationship-based selection criteria: CSQL supports the following spatial predicates: *to_the_right_of in*, *to_the_left_of in*, *above_of in*, and *below_of in*. These predicates take two entities (semantic or dual) and an image entity, and they check the spatial relationship between two entities in the given image. In the case user provides a dual entity as one of the first two predicates, the system uses its semantic identity unless the dual entity is extracted from the given image entity. Please note that these spatial relationships are those that can be *automatically identified by COIR*.

Containment selection criteria: The first argument of this predicate must be an image entity. The binary *contains* predicate returns true if the second argument, which is a dual entity, is contained in the first argument, which is an image entity.

The predicate usage is summarized in Table1 (s, d, and i stand for *semantic entity*, *dual entity*, and *image entity* respectively).

5 Query Processing issues

CSQL can be viewed as a logic-based language. An CSQL query is of the form

$$Q(Y_1, \ldots, Y_n) \leftarrow P_1 \wedge P_2 \wedge \ldots \wedge P_p.$$

where $Y_1, \ldots, Y_n$ are variables and $P_1, .., P_p$ are the predicates defined in Table 1. For the query shown in Figure 6, there are five predicates: three image matching predicates and two spatial relationship predicates.

In CSQL, the solution of a query Q is defined as an ordered set S of n-tuples of the form $< X_1, X_2, \ldots, X_n >$, where X_i corresponds to a variable, Y_i, in Q, each X_i satisfies the type constraints of the predicates in Q, and when X_i's are simultaneously substituted to the places of Y_i's on the right hand side of the query rule, each and every predicate in Q are satisfied. The order of the set S denotes the relevance ranking of the solutions. Integration of semantics-, cognition-, and spatial relation-based selection criteria introduce the following new challenges for query processing:

- Traditional query execution methods need to be adapted to handle similarity values generated from image matching and semantics-based comparisons. The $\wedge$ operator in $Q(Y_1, \ldots, Y_n)$ has different meaning when the values of $P_1, .., P_p$ are not either *true/false* or *probability*. Instead they are similarity and relevance values.
- Need to rank the relevancy of candidate images to a query based on both object matching and image structure matching above in such a way that users are most satisfied.
- Need to support query relaxations for image matching, semantics comparisons, and spatial relationships. Because there may be many partial matches, methods to handle large intermediate results are required.

5.1 Resolution Methods

In standard logic a predicate is analogous to a propositional function. Since propositions are either true or false, predicates have one of the two values: *true* or *false*. However, in CSQL, predicates correspond to similarity functions which return values between 0.0 and 1.0. Such a deviation in the semantics of

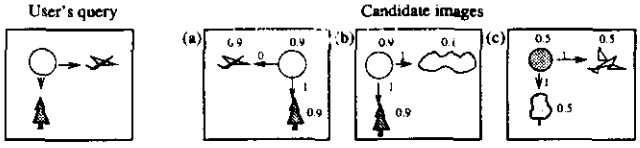


Figure 6: Examples of Matched Images

the predicate concept causes many challenges. When we substitute the values in an n-tuple t, of the form $< X_1, X_2, \ldots, X_n >$, into the corresponding variables in predicates $P_1, \ldots, P_p$ of Q, these predicates return similarity values $v_1, \ldots, v_p$ where $0.0 \leq v_j \leq 1.0$. One question is how t forms a solution to Q. We see two possible solutions as follows:

The first solution is transforming similarity functions into propositional functions by choosing a cut-off point, r_{true} and by mapping all real numbers in $[0.0, r_{true})$ to false and real numbers in $[r_{true}, 1.0]$ to *true*. Hence, the output of similarity functions can be mapped into either *true* or to *false*. Such a cut-off point will correspond to a similarity degree which denotes *dissimilarity*.

The second solution is leaving the final decision not to the constituent predicates but to the n-tuple as a whole. We can view this strategy as relaxing *conjunction* constraints *during* the resolution process. This approach requires a function, $\mathcal{R}_Q$ for evaluating overall similarity of candidate images based on the values of $P_1, .., P_p$. A solution (image) is accepted if its value of $\mathcal{R}_Q$ is is greater than a given threshold.

The choice of the solution method is critical. In the first approach, predicates can refute solutions as soon as they are evaluated. If users only consider full matching, this method is preferred because it allows query optimization by early elimination of the search space. If, however, users are interested in both partial matches and full matches, the second method is more suitable. SEMCOG currently uses the *second* method since it supports both complete and partial matching by relaxing image matching, semantics comparisons, and spatial constraints.

5.2 Image Relevance Ranking

The order of the set S denotes the order of relevance of the solutions in S based on their *overall* similarity. A relevance value of a solution to a user query must be calculated based on both object matching and image structure matching. We argue that the approach of simply taking an average, used by most systems, is not desirable for relevance value calculations. One reason is that similarity values are relative, not absolute. The similarity values from image matching, semantics-based comparisons, and spatial relationship comparisons are on different scales. We have explored the following approaches:

The first approach to relevance ranking is to allow users to specify how to combine these similarity values - weight of each aspect comparison. SEMCOG allows users to specify ratio of color importance to shape importance and ratio of object-matching importance to structure-matching importance. The ranking function for a result for query Q, $\mathcal{R}_Q(\alpha_1, \alpha_2)$, can be specified as $\alpha_1 \times Object\_Matching\_Ranking + \alpha_2 \times Structure\_Matching\_Ranking$, where α_1 and α_2 are between 0.0 and 1.0 and $\alpha_1 + \alpha_2 = 1.0$.

Object_Matching_Ranking is calculated based on object semantics matching, color matching, and shape matching. *Structure_Matching_Ranking* is defined as the number of edge (representing spatial relationships) matches between query specification and image layout.

Based on our initial study and experiments, the relevance function that we use is $\mathcal{R}_Q(t, r_{true}, \alpha_1, \alpha_2, \beta) = \alpha_1 \times \frac{((\prod_{v_{t,j} \geq r_{true}} v_{t,j}) \times (\prod_{v_{t,j} < r_{true}} \beta))^{1/p} - \beta}{1-\beta} + \alpha_2 \times \frac{\sum_{v_{t,j} \geq r_{true}} 1}{p}$ where p is the number of predicates in Q, α_1 and α_2 are between 0.0 and 1.0 such that $\alpha_1 + \alpha_2 = 1.0$, β is a value greater than 0 and less than r_{true}, r_{true} is the similarity cutoff point, and $v_{t,j}$ is the similarity value returned by predicate P_j for n-tuple t.

The first term corresponds to the calculation of relevancy of object matching based on semantics and visual signatures while the second term corresponds to the relevancy of structural matching. Here, the term $(\prod_{v_{t,j} \geq r_{true}} v_{t,j}) \times (\prod_{v_{t,j} < r_{true}} \beta))^{1/p}$ returns a value between β and 1.0 where 1.0 corresponds to the highest relevance and r_{true} corresponds to the minimum predicate relevance. β is a offset for preventing the final relevance value to be 0 when one of the predicates has a similarity value of 0, and the use of power of $1/p$ is to prevent the multiplicative term to decrease fast. The subtraction of β from $(\prod_{v_{t,j} \geq r_{true}} v_{t,j}) \times (\prod_{v_{t,j} < r_{true}} \beta))^{1/p}$ and the division of the result by $1 - \beta$ is to normalize the result back to between 0.0 and 1.0 from between β and 1.0. The second major component of the function, i.e. $(\sum_{v_{t,j} \geq r_{true}} 1)/p$ returns the fraction of the predicates which are matched. We use this term to evaluate results of structural matching.

The second approach is to design the system which is capable of learning users' preference and then adapting the values of α_1 and α_2, the importance of object matching versus structure matching. Learning is performed through interactions between users and

SEMCOG. In Figure 6, we have three types of partial match. Images that match user query perfectly are categorized as "complete match". When SEMCOG could not find complete match images, SEMCOG presents candidate images instead and it asks for user feedback. If the users tend to choose Figure 6(a) type of images, the system will give a heavier weight for object match similarity values. On the other hand, if users tend to choose Figure 6(b) type of images, the system will give a heavier weight for structure match similarity values.

5.3 Performance Considerations

The highlight of our architecture is its modularity by assigning media-based and semantics-based tasks to different modules, so that different types of tasks are executed by corresponding specialized modules.

Media-based tasks, such as image matching, are usually more expensive and it is harder to estimate their processing costs. On the other hand, Semantics-based tasks require mainly traditional database query processing and many existing indexing and query optimization techniques can be applied to gain better performance. We assign these two types of tasks to separate components, the *Image Data Query Processor* and the *Textual Data Query Processor* respectively and employ a facilitator module to coordinate the interactions between modules.

We assign the tasks of coordination, query reformulation, forwarding, and result integration to the *Facilitator*, because it has a more complete knowledge of the query execution statistics compared to the other individual modules in the architecture, the facilitator can provide a better query execution plan. Another reason for our choice is that the *Facilitator* provides a uniform way of handling all different types of rewrites, such as term relaxations, adjective relaxations, and query rewritings (query execution sequence re-arrangement). As a result, the *Facilitator* can generate a more efficient execution plan.

Like most query optimizers in traditional database systems, the *Facilitator* also periodically collect database statistics to improve query processing performance; especially using selectivity for query statement execution sequence arrangement. One additional unique function provided by the *Facilitator* is to recommend users query reformulation alternatives and analyze possible outcomes so that users can visualize distributions of related images. This functionality is described in [7]. The query reformulation functionality is also used for automated query relaxation as examples shown in Figure 6.

6 Related Work

Virage[2] is a system for image retrieval based on visual features (such as *image primitives* color, shape, and texture) and other domain specific features. Virage's has an SQL-like query language extended by user-defined data types and functions. Virage also provides users a form-based query interface called VIV. QBIC[3], developed at IBM, is another system that supports image retrieval using visual examples. The image matching is based on image features, such as colors, textures, shapes, locations, and layout. Both Virage and QBIC support image matching and keyword-based retrieval functionality on the *whole* image level provide semantics-based access to objects in images. SEMCOG supports this functionality at the whole image level. Additionally, SEMCOG supports object-based image retrieval and allows users to specify descriptions of objects in the images using semantical terms and/or visual examples. SEMCOG also allows users to specify more relaxed terms using the ISA predicate.

SCORE[1, 11] is a similarity-based image retrieval system developed at University of Illinois at Chicago. This work focuses on the use of a refined ER model to represent the contents of pictures and the calculation of similarity values based between E-R representations of images stored and query specifications. However, SCORE does not support image-matching.

VisualSeek [4] is a content-based image query system developed at Columbia University. VisualSeek uses color distributions to retrieve images. Although VisualSeek is not object-based, it provides region-based image retrieval: users can specify how color regions shall be placed with respect top each other. VisualSeek also provide image comparisons and sketches for image retrieval. However, VisualSeek is designed for image matching, it does not support retrieval based on semantics at either the image level or the object level.

Chabot project[9] at UC Berkeley is initiated to study storage and retrieval of a vast collection of digitized images from the State of California Department of Water Resources. Chabot provides a form based browser where users can either provide metadata, keywords, concepts, or color-distributions to retrieve images. Chabot allows users to define a range of a color spectrum using pre-defined keywords, such as *Rose Red*. SEMCOG also supports this functionality as described in [7].

MQL[5] is a multimedia query language. The syntax of the MQL is *select* $< A >< V >$ *from* $< R >$ *where* $< C >$ in which $< A >$ is a list of attributes

to be retrieved, $< V >$ is the result of version, $< R >$ is the domain class, and $< C >$ is a condition. [5] claims MQL can support complex object queries, version queries, and nested queries (e.g. IN). MQL also supports a *Contain* predicate through pattern matching on images, voice, or text. MQL is very similar to our query language in modeling and query language design. CSQL provides additional predicates, such as ISA and S-LIKE, to allow users to control the relaxation of query processing. Compared with MQL, SEMCOG is a prototype system rather than a theoretical contribution. SEMCOG also provides a visual query interface and query generator to assist users in posing such complicated queries.

7 Conclusions

In this paper, we introduce an image model in which images are viewed as hierarchical structured complex objects with both semantics and visual properties. Based on this model, we present a novel approach to image retrieval which supports queries, both at image and object levels, using combinations of semantics, visual, and spatial conditions. The contributions of our work include (1) hierarchical structure for image modeling based on both semantics and visual property and the object level spatial relationships; (2) system architecture and implementation for the proposed image modeling; and (3) formal definition and implementation of a multimedia language for the image modeling.

Acknowledgments

We would like to express our sincere appreciation to Kyoji Hirata, Yoshinori Hara, Peter Scheuermann, Chris Clifton, and anonymous reviewers for their invaluable comments and suggestions on this work.

References

[1] Y. Alp Aslandogan, Chuck Thier Clement Yu, Chengwen Liu, and Krishnakumar R. Nair. Design, Implementation and Evaluation of SCORE. In *Proceedings of the 11th International Conference on Data Engineering*, Taipei, Taiwan, March 1995. IEEE.

[2] J. R. Bach, C. Fuller, A. Gupta, A. Hampapur, B. Horowitz, R. Jain, and C.-F. Shu. The Virage Image Search Engine: An Open Framework for Image Management. In *Proceedings of the SPIE - The International Society for Optical Engineering: Storage and Retrieval for Still Image and Video Databases IV*, San Jose, CA, USA, February 1996.

[3] Myron Flickner, Harpreet Sawhney, Wayne Niblack, Jonathan Ashley Qian Huang, Byron Dom, Monika Gorkani, Jim Hafner, Denis Lee, Dragutin Petkovic, David Steele, and Peter Yanker. Query by Image and Video Content: The QBIC System. *IEEE Computer*, 28(9):23–32, September 1995.

[4] John R. Smith and Shin-Fu Chang. VisualSEEk: a Fully Automated Content-based Image Query System. In *Proceedings of the 1996 ACM Multimedia Conference*, pages 87–98, Boston, MA, 1996.

[5] S.C. Kau and J. Tseng. MQL - A Query Language for Multimedia Databases. In *Proceedings of 1994 ACM Multimedia Conference*, pages 511–516, 1994.

[6] Kyoji Hirata and Yoshinori Hara and H. Takano and S. Kawasaki. Content-Oriented Integration in Hypermedia Systems. In *Proceedings of 1996 ACM Conference on Hypertext*, pages 11 – 21, Washington DC, USA, March 1996.

[7] Wen-Syan Li, K. Selçuk Candan, Kyoji Hirata, and Yoshinori Hara. Facilitating Multimedia Database Exploration through Visual Interfaces and Perpetual Query Reformulations. In *Proceedings of the 23th International Conference on Very Large Data Bases*, pages 538–547, Athens, Greece, August 1997. VLDB.

[8] G. A. Miller. WordNet: A Lexical Databases for English. *Communications of the ACM*, pages 39–41, November 1995.

[9] Virginia E. Ogle and Michael Stonebraker. Chabot: Retrieval from a Relational Database of Images. *IEEE Computer*, 28(9):40–48, September 1995.

[10] S. K. Chang and C. W. Yan and Donald C. Dimitroff and Timothy Arndt. An Intelligent Image Database System. *IEEE Transaction on software Engineering*, 14(5):681–688, 1988.

[11] A. Prasad Sistla, Clement Yu, Chengwen Liu, and King Liu. Similarity based Retrieval of Pictures Using indices on Spatial Relationships. In *Proceedings of the 1995 VLDB Conference*, Zurich, Switzerland, September23-25 1995.