IFQ: A Visual Query Interface and Query Generator for Object-based Media Retrieval

Wen-Syan Li K. Selçuk Candan* Kyoji Hirata Yoshinori Hara

C&C Research Laboratories, NEC USA, Inc., 110 Rio Robles, San Jose, CA 95134

Email: {wen,hirata,hara}@ccrl.sj.nec.com, candan@cs.umd.edu

Abstract

There are two major directions for image retrieval system development. The first direction is the direct manipulation of information using query languages, which are precise to computers but not user friendly. The second direction is the development of natural interfaces, such as *query by image example.* The latter approach has the advantage of better query visualization, but it usually yields a low precision because users' drawings may not be precise. IFQ (In Frame Query) is a visual user query interface for object-based media retrieval systems. It aims at not only providing a natural visual query interface but also supporting precise direct manipulation through automated query generating. Furthermore, IFQ gives users flexibility of using combinations of semantics expressions, conceptual definitions, sketch, and image examples to pose queries.

Keywords. Object-based media retrieval, multimedia databases, query language, visual query interface.

1 Introduction

With the increasing interest in multimedia systems, content-based image retrieval attracted attention of researchers. We concentrate our efforts on two issues: *content-based image retrieval methods* and *query specifications.*

Content-based retrieval methods includes two approaches[7]. In the first approach, image contents are modeled as a set of attributes extracted manually or semi-automatically and are managed in traditional relational DBMSs. Queries are posed using these attributes. The accuracy of this attribute-based (semantics-based) approach depends on levels of abstraction. This approach is good at retrieval based on image semantics. However, it has a weakness of a low visual expressive capability since images are visual and hard to describe in detail using text. Another weakness of this approach is that the attributes/semantics of images must be specified explicitly in advance if the semantics can not be extracted automatically.

The second approach to content-based image retrieval employs feature-extraction and object recognition techniques for image matching. This image matching-based (cognition-based) approach is usually computational expensive and difficult. As a result, this approach primarily aims at domain-specific applications, such as face and finger print recognition and tumor identification in medical applications. This approach has the advantage of using visual examples. However, one disadvantage of using the image matching-based approach alone is its lower precision because users' drawings are usually not precise enough. Another weakness of this approach is that it can not support queries on generalized concepts, such as transportation and appliances.

As to *query specification mechanisms,* there are two major development directions. One direction is manipulation of the information through query languages which are precise to computers but not natural to users since they do not support visualization of queries. Approaches include (1) SQL-like query languages, such as the Multimedia Query Language MQL[10], and (2) various keyword-based interfaces.

Another direction is the use of a more natural specification method, such as querying by image examples. Related approaches include (1) cognition-based interfaces that support queries by providing sketches and image examples, and (2) descriptive natural language-based systems where users pose queries by describing target image semantics using a somewhat "natural language." These two types of interfaces match users' cognitive representation and mental models. However, their results usually yield low precision because queries are usually ambiguous to computers.

We see that there is a gap between existing ap-

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

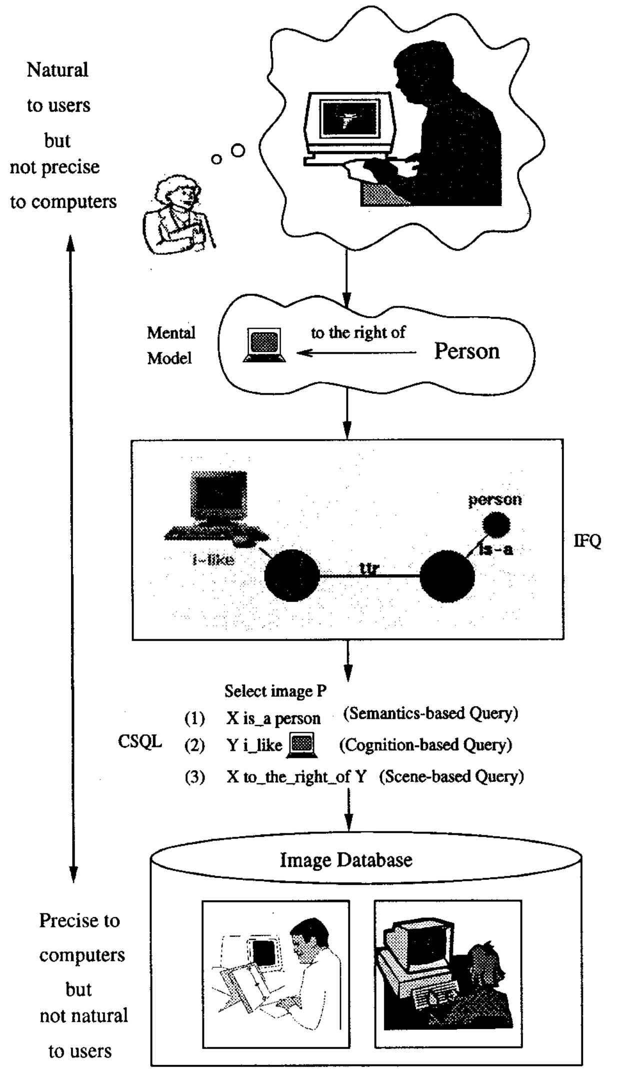


Figure 1: Visual Query Interface for Media Retrieval

proaches to *content-based image retrieval methods* and *query specification mechanisms.* We believe the integration of both approaches of content-based image retrieval methods would give users a high flexibility to query image databases based on combinations of semantics and visual examples. We also argue that a query interface that can match users' mental models through visualizing queries and also support precise information manipulation is essential for effective image retrieval.

IFQ (In Frame Querying) is a visual interface for *object-based* media retrieval systems, such as the SEMCOG[11] (SEMantics and COGnition based image retrieval system). IFQ provides functionalities of *object-based image retrieval, flexible visual query interface*, and *query specification assistance.* IFQ queries are posed by specifying image objects, their semantics, image contents, and layouts. We have integrated semantics and cognition-based approaches to give users a higher flexibility to pose queries. As IFQ's name states, queries themselves specified in the IFQ window (frame) actually visualize target images as the query specification progresses.

Figure 1 shows an example in which a user wants to retrieve an image containing a person and a computer where the person is to the right of the computer. Note that as shown in the figure, there is a gap between the user's mental model and the actual images stored in the database system. An actual window dump of IFQ for this query is shown in the middle of Figure 1. The user specifies the query by interacting with IFQ while the corresponding query is automatically generated by IFQ. Figure 1 shows the corresponding query in CSQL (Cognition and Semantics-based Query Language), a SQL-like query language used in our system.

With our system, users can use the combination of semantics and visual examples. Our system provides higher flexibility by integrating the approaches of semantics-based and cognition-based image retrieval. The result of the query is a set of images ranked by the matchings of the image objects and their layouts with the users' query. The results for our example above contain two images. Please note that in the results the concept *person* has been relaxed to the concepts *man* and *woman* accordingly.

The rest of this paper is organized as follows: We first review existing work related to this paper. We then give an overview of the architecture of a multimedia database system, SEMCOG, that IFQ is built on top of. In Section 4, we introduce our image database query language (CSQL), an extension of SQL. Section 5 presents the theory, design, and functionalities of IFQ. In Section 6, we discuss the extendibility of IFQ to video modeling and retrieval. Finally we offer our conclusions.

2 Related Work

H. Nishiyama et. al[12] pointed out that there are two patterns of end-users in their visual memory when they view paintings or images. The first pattern consist of roughly the whole image, whereas the second pattern concentrates on specific objects within the image, such as a man or a desk. These arguments support the development of IFQ *visual*, which properly matches users' query model, which captures semantical, cognitive, and structural query content, with media models in multimedia databases.

In [1], Amato et. al suggests an object-based multimedia data model which also addresses the structural properties of the images. They also discuss how the model can capture temporal information. However, they do not present a language or a user interface to support their data model.

V. N. Gudivada et. al[6] categorizes one type of image retrieval as Retrieval by Spatial Constraints (RSC) that facilitates a class of queries that are based on relative spatial relationships among objects in an image. RSC queries are further categorized into *relaxed* RSC queries and *strict* RSC queries. We take a similar approach to compare spatial structures of users' query specifications with stored images.

Virage[3] is a system for image retrieval based on visual features, such as *image primitives*, such as color, shape, or texture and other domain specific features. Virage's has an SQL-like query language extended by user-defined data types and functions. Virage provides users a form-based query interface called VIV, which is not a visual query interface like IFQ.

QBIC[5] is a system that supports image retrieval using visual examples. The image matching is based on features images, such as colors, textures, shapes, locations, and layout of images. QBIC does not provide semantics-based access to objects in images.

SCORE[2] is a similarity-based image retrieval system developed at UIC. 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 ER representations of images stored and query specifications. However, SCORE does not support image-matching.

VisualSeek [9] is a content-based image query system developed at the 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.

Chabot project[13] at UC Berkeley is initiated to study storage and retrieval of a vast collection of digitized images. Chabot provides a form based browser where users can either provide metadata, keywords, concepts, or color-distributions to retrieve images. Chabot also supports concept definition functionalities. However, Chabot is not object-based, it does not provide facilities for spatial queries and semantic-based object retrieval.

VISUAL[4] is an object-oriented graphical query language designed for scientific databases where the data has spatial properties and exploratory queries are common. VISUAL has a visual interface to allow users to pose queries based on an object-oriented query specification model. VISUAL's interface is more like a graphical query interface.

MQL[10] is a multimedia query language. MQL also supports a *Contain* predicate through pattern matching on images, voice, or text. However, all users' input must be text rather than both using visual interfaces.

3 System Architecture

SEMCOG architecture contains five components as shown in Figure 2. For more detail of SEMCOG, please see [11] We summarize the functionality of each component as follows:

The *Facilitator* coordinates the interactions between components of SEMCOG. It forwards image matching related tasks to the *Cognition-based Query Processor* and non-image matching tasks to the *Semantics-based Query Processor.* One advantage of assigning these tasks to the *Facilitator* because the *Facilitator* has more complete knowledge of the query execution statistics and it can provide a globally optimum query processing.

COIR (Content-Oriented Image Retrieval)[8] is an object-based image retrieval engine based on colors and shapes. We use it as the *Cognition-based Query Processor* in SEMCOG. The main task of the COIR is to identify image regions based on pre-extracted image metadata, colors and shapes. Since an object may consist of multiple image regions, COIR consults the *image component catalog* for matching image objects.

When an image is "registered" at SEMCOG, the *Image Semantics Editor* interacts with COIR to edit the semantics of an image and objects in the image. The *Image Semantics Editor* then stores the image, semantics, and image metadata to the database.

The *Terminology Manager* maintains a terminology base which is used for query relaxation. For example, a user may submit a query as "Retrieve all images containing an *appliance.*" Since *appliance* is a generalized concept rather than an atomic term, the *facilitator* consults the *Terminology Manager* to reformulate the query. Existing dictionaries, such as Wordnet, can be employed to build a terminology base.

The *Semantics-based Query Processor* performs queries concerning image semantics. The image semantics required for query processing is generated during the image registration. The semantics-based query processing is the same as traditional query processing on relational DBMSs.

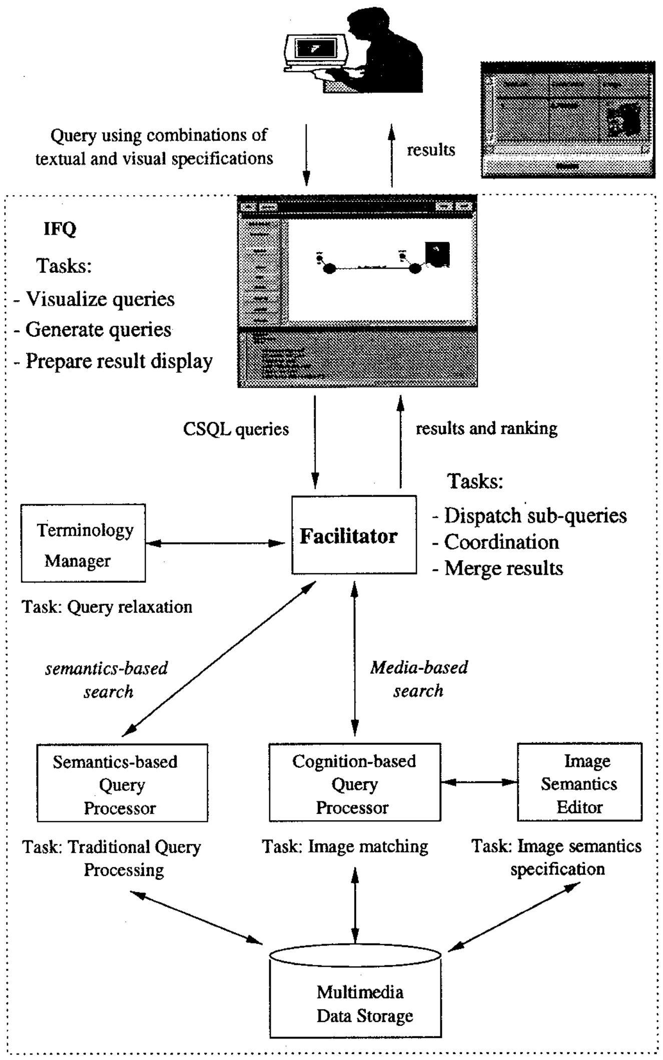


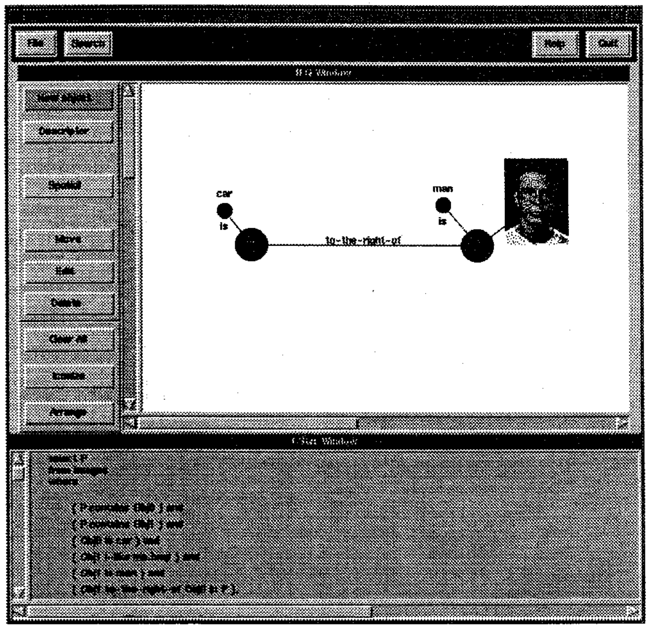
Figure 2: System Architecture of SEMCOG

4 Query Language

CSQL is the underlying query language used in SEMCOG by augmenting SQL with additional predicates to handle multimedia data. These predicates extend the underlying database system to a multimedia database system. These predicates defined in CSQL include: (1) *Semantics-based*: *is* (e.g. man vs. man), *is_a* (e.g. car vs. transportation, man vs. human), and *s_like* for "semantics like" (e.g. car vs. truck); (2) *Cognition-based*: *i_like* for "image like" that compares visual signatures of two arguments and *contains*; (3) *spatial relationship-based* such as *above*, *below* and etc.

5 IFQ Query Interface

The IFQ (In Frame Query) interface is shown in Figure 3. IFQ, implemented using Tcl/Tk and running on X window systems, is a visually rich query



```
select P
from Images
where
    ( P contains Obj0 ) and
    ( P contains Obj1 ) and
    ( Obj0 is car ) and
    ( Obj1 [illegible] ) and
    ( Obj1 is man ) and
    ( Obj1 [illegible] Obj0 in P ).
```

Figure 3: IFQ Specification Window (top) and CSQL Generating Window (button)

interface which allows users to pose queries using combinations of keywords, concepts, semantics, image examples, sketches, and spatial relationships in a single "frame". As IFQ's name shows, the query specification in the IFQ window (frame) *itself* visualizes the target images as query specification process progresses. IFQ allows users to specify queries in a more natural manner: users can graphically describe objects contained in the target image and their layout. The corresponding CSQL query is generated by the interface. As a result, users are not required to be aware of the schema and implementation details. In this section we give details of the theory, design, and functionalities of IFQ.

5.1 Query Specification

The query specification process in IFQ consists of three steps: (1) introducing objects in the target image, (2) describing objects, and (3) specifying objects' spatial relationships. In IFQ, objects are represented with bullets, and *descriptors*, which describe the properties of the objects they are attached to, are represented as small bullets. Now we show, step by step, how the query "Retrieve all images in which there is a man to the right of a car and he looks like this image" can be posed using IFQ:

- In step 1, user introduces the first object in the image.
- In step 2, s/he further describes the object by attaching "i_like < *image* >" and "is *man*" de-

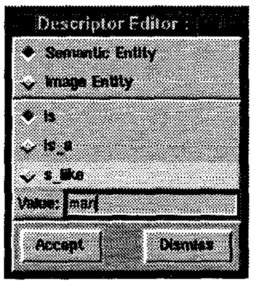


Figure 4: Semantics Input Windows

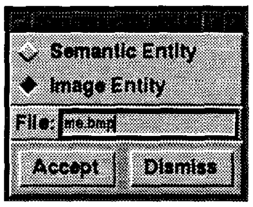


Figure 5: Image Path Input Windows

scriptors. Figure 4 shows the interface for specifying semantics of entity descriptors. Figure 5 shows the interface for input images by specifying the file path. After a user specifies an image path, the system automatically replaces the descriptor with the thumbnail size image the user specifies. Users can also draw sketches to provide visual examples as part of queries using the interface shown in Figure 6.

- In step 3, user introduces another object and describes it using the "is *car*" descriptor.
- Then, in step 4 user describes the spatial relationship between these two objects by drawing a line, labeled by *to-the-right-of*, from the man object to the car object. Figure 7 shows the interface for specifying spatial relationships.

Please note that while user is specifying the query using IFQ as shown in Figure 3, the corresponding CSQL query is automatically generated in the CSQL window. Users do not need not be aware of the CSQL syntax and variable specifications since these are handled by the IFQ interface. Users pose queries by simply clicking buttons and dragging and dropping icons representing entities and descriptors.

5.1.1 Checking and Arrangement

IFQ also has two optional functionalities to increase the perceptual quality and correctness of the query specifications. The first one is the *arrange* option. IFQ can check the matching between the spatial relationship specifications and the actual layout on the screen. If there is a mismatch, IFQ rearranges the

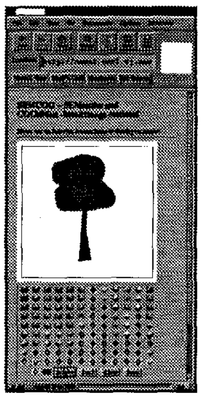


Figure 6: Sketch Input Windows

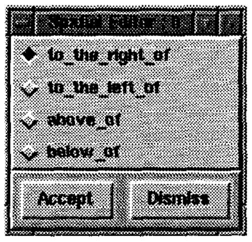


Figure 7: Spatial Relationship Input Windows

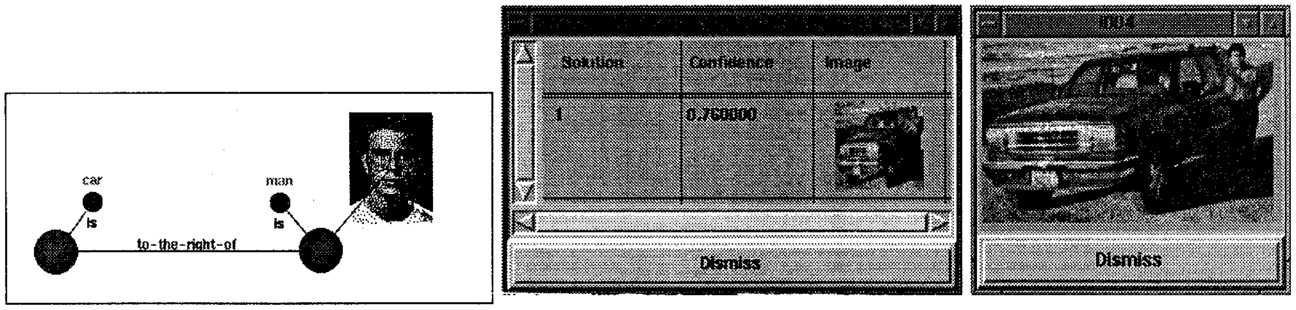
query objects on the screen according to the query specifications. Furthermore, if there is a conflict in the layout specifications provided by the user (e.g. a man is specified to be above a tree and the same tree is specified to be above the man), IFQ informs the user by highlighting such conflict specifications.

5.1.2 Viewing the Results

After the query is specified, the user can submit it. The query generated in the CSQL window below the IFQ window will then be executed. Figure 8 shows a query "retrieve images in which there is a man and a car, the man is to the right of the car, and the man looks like the image *me.gif*" and its result. The result contains thumbnail size images ranked by degrees of confidence. Users can click on any thumbnail image to see the real image as shown on the right side.

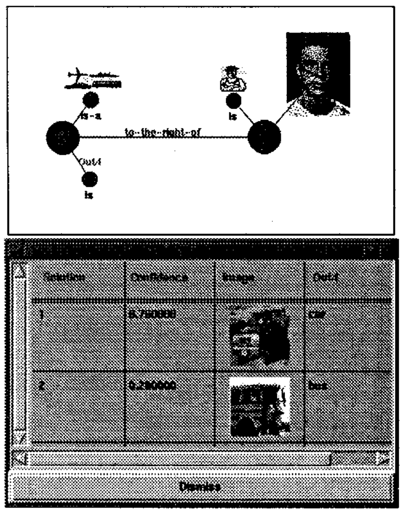
5.1.3 Iconization

The second optional functionality of IFQ is the *iconize* option which replaces the semantic terms of descriptors in the IFQ window with the corresponding icons to improve the perceptual quality of the query. In Figure 9, we show a window dump in which *man* and *transportation* are replaced by icons.



Solution	Confidence	Image
1	0.760000	

Figure 8: Query Results and Image Retrieved



Solution	Confidence	Image	Out4
1	0.760000		car
2	0.290000		bus

Figure 9: Semantics Extraction and Its Results

5.2 Semantics Extraction/Relaxation

It is often that users want to learn more on image contents or relax query conditions for initial queries since they may not have specific target images in their mind. When the users retrieves initial sets of images, they would have better ideas on how to refine the queries. IFQ supports these functionalities through IFQ and user interactions.

Figure 9 shows an example that the user relaxes the condition in Figure 8 for "being a car" to "being a kind of transportation" using *is_a transportation* and the user also wants to extract the semantics of the corresponding objects in the retrieved candidate images. This form of interaction for semantics extraction is performed by specifying unbound descriptors. In this example, the user attached an unbound descriptor to the object and IFQ assigned a name (*out4*) to it to check the actual matched semantics. The corresponding CSQL query generated by IFQ is as follows:

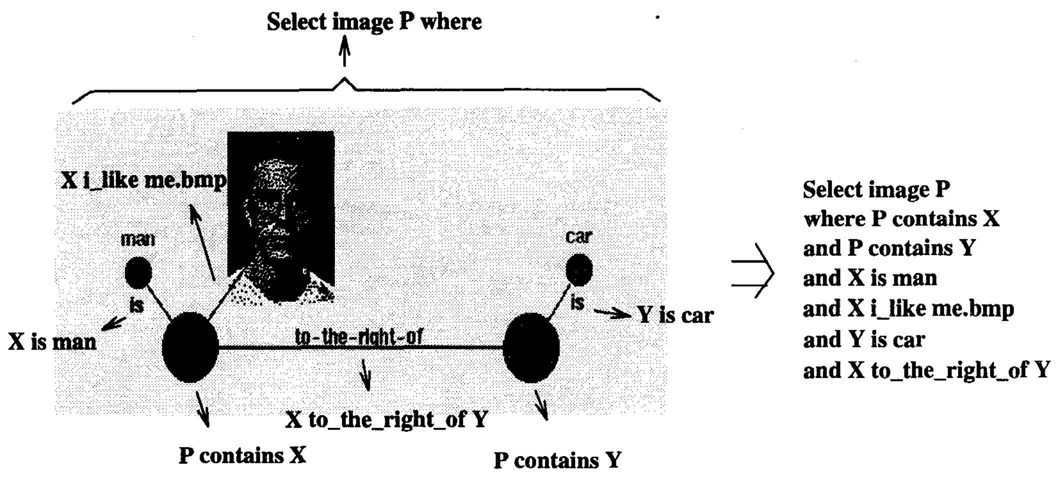
select image P, X.semantics
where P *contains* X
  and P *contains* Y
  and X *is_a* transportation
  and Y *is* man
  and Y *i_like* me.gif
  and Y *to_the_right_of* X

The result given in Figure 9 shows two candidate images. One image contains a car and the other contains a bus as the results for *X.semantics* (*out4* as its label) are *car* and *bus*. In this example, the second candidate image has a lower confidence, because the human in the image can not be identified as a man. We next show how CSQL queries are generated.

5.3 Query Modeling and Generating

In this section, we describe how we model CSQL queries using IFQ visual specifications. The predicates in CSQL can be grouped into three categories:

- *Containment:* Since SEMCOG is an object-based image database system, users can query image objects with finer granularity than a whole image. When a user specifies a query to retrieve images with some objects in it, *contain* is the default condition that must be satisfied. An example of this type of query, "retrieve all images that contain one object", can be posed as *select image P where P contains X.*
- *Object description:* Users can further specify visual and semantic descriptions of objects using the following predicates: *is*, *is_a*, *s_like*, and *i_like.* An example of this type of query, "retrieve all images that contain a man", can be posed as *select image P where (P contains X) and (X is man).*
- *Spatial relationship:* Users can describe the spatial relationships between the objects in the required images using the following predicates: *to_the_right_of*, *to_the_left_of*, *above_of*, and *below_of.* An example query can be posed as: *select image p where (p contains x) and (p*



```
Select image P
where P contains X
and P contains Y
and X is man
and X i_like me.bmp
and Y is car
and X to_the_right_of Y
```

Figure 10: Query Modeling Using IFQ Specifications

contains y) and (x is <u>*man*</u>*) and (y is_a* <u>*transportation*</u>*) and (y to_the_right_of x).*

The way that we map these three types of query criteria to visual specifications in IFQ is similar to the ER (Entity-Relational) data model. The entities and relationships in the ER model correspond to the objects and spatial relationships in IFQ. The attributes in the ER model correspond to the descriptors in IFQ with the following differences: (1) In IFQ, the number of descriptors (attributes) may vary for different objects, (2) the types of descriptors (attributes) may vary; types of descriptors can be used to describe visual features (such as *i_like*) or semantics, such as *s_like* or *is_a*, and (3) the descriptors (attributes) can be unbound as described later in Section 5.2.

The guide lines of modeling CSQL queries to IFQ visual specifications are as follows:

1. "Select image P where" is a default CSQL statement generated when an IFQ window is initialized.

2. The *Containment* type of CSQL specifications are modeled by adding objects in the IFQ window. That is, by adding an object in the IFQ window, a CSQL statement "*P contains object_variable*" is generated. The corresponding *object_variable* is assigned by the query generator.

3. *Object description* type of CSQL specifications are modeled by adding *object descriptors* and attaching *object descriptors* to objects. The *object descriptors* can be "*is* <u>*man*</u>", "*is_a* <u>*transportation*</u>", "*i_like* <u>*tree.bmp*</u>", "*s_like* <u>*man*</u>, and so on. When the user attaches an object descriptor, say "*is* <u>*man*</u>", to an object, say X, a CSQL statement "*X is* <u>*man*</u>" is generated.

4. *Spatial relationship* type of CSQL specifications are modeled by adding lines between two objects and specifying spatial meanings of the lines. When the user draws a line between two objects, say *object_variable*1 and *object_variable*2, and specifies the line as *to_the_right_of*, a CSQL statement *object_variable1 to_the_right_of object_variable2* is generated.

On the left side of Figure 10, we illustrate how each IFQ specification corresponds to the CSQL query statements. The right side of Figure 10 shows the CSQL query generated for the given IFQ specification.

5.4 Concept Definitions

In many cases, users may want to define their own concepts. IFQ also supports user defined concepts through combinations of visual examples, semantics and predicates defined in CSQL, such as *i_like* and *is*.

For example, users may define the concept of *transportation* as follows:

Define concept transportation *as*
select P.semantics
where P *is* <u>car</u> *or* P *is* <u>airplane</u> *or* P *is* <u>ship</u> *or* P *is* <u>bicycle</u>

Users can also define concepts using visual examples. For example, a user may define the concepts of *Fuji_Mountain* and *Lake* as follows:

Define concept Fuji_Mountain *as*
select image P
where P *i_like* <u>fuji_mountain.gif</u>

Define concept Lake *as*
select image P
where P *i_like* <u>lake1.gif</u>
   or P *i_like* <u>lake2.gif</u>
   or P *i_like* <u>lake3.gif</u>

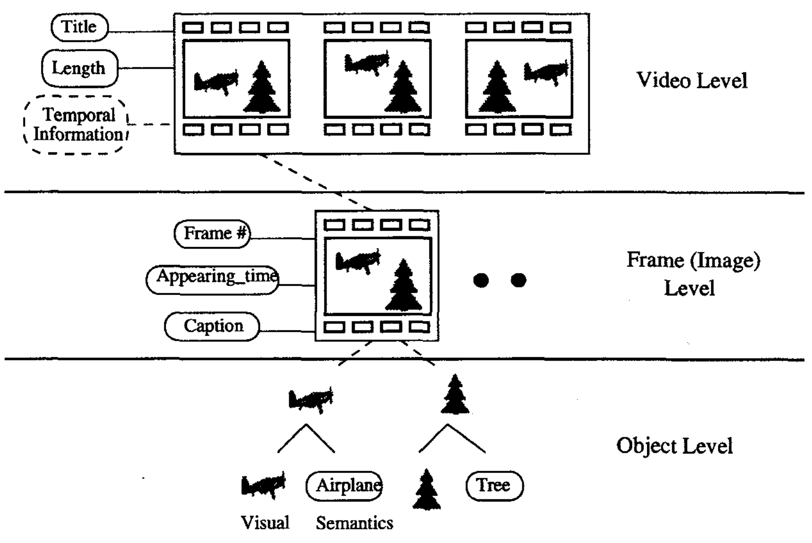


Figure 11: Video Modeling in IFQ

A concept can be also defined using other defined concepts or combinations of semantics and visual examples as a regular CSQL query. A concept definition *Fuji_Mountain_by_a_lake* can be posed as follows:

Define concept Fuji_Mountain_by_a_lake *as*
select image P
where P *contains* X
  and P *contains* Y
  and X *i_like* Fuji_Mountain
  and Y *i_like* Lake
  and X *on_the_top_of* Y

6 Extending IFQ to Video Retrieval

IFQ can be extended for querying video data. We are currently working on a method of modeling video data based on the image modeling discussed in Section 5.3. Our goal is to translate the video retrieval process into object-based image retrieval, and hence to build a video retrieval system on top of SEMCOG by extending existing CSQL predicates and IFQ.

Figure 11 shows a three-level model of video representation. At the first level (object level), we model the video objects. An object consists of two parts: semantics and visual identity. Objects, along with the corresponding spatial information, form an image. An image with additional information, such as *Frame #*, *Appearing_time*, and *Caption*, forms a video Frame. A sequence of video frames along with additional information, such as *Title*, *Length*, and temporal information, forms a video clip. Please note the temporal information available in the video level because the temporal information is *derived* from *Frame #* and *Appearing_time* information in the frame level.

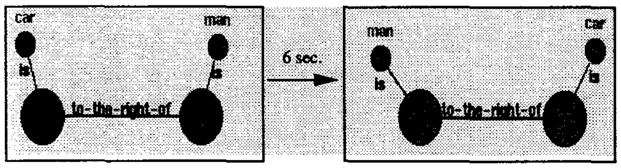


Figure 12: Example IFQ Query for Video Retrieval

For example, a query of the form "retrieve video clips in which there are a car and a man and the car which is moving to the right passes the man" can be posed using IFQ as shown in Figure12. Figure 12 shows two still images representing this query by two frames in the video separated with at most 6 seconds. In our video modeling, this video retrieval query can be translated into two image retrieval queries for the two still images with temporal relationship constraint (< 6 seconds apart) at the video level. The IFQ query shown in Figure 12 can be translated into a Video CSQL (VCSQL) query as follows:

select video V
where V *contains* Frame1
  and V *contains* Frame2
  and (Frame2.Appearing_time - Frame1.Appearing_time) < 6
  and (Frame2.Appearing_time - Frame1.Appearing_time) > 0
  and Frame1 *contains* X
  and Frame1 *contains* Y
  and X *is* car
  and Y *is* man
  and Y *to_the_right_of* X *in* Frame1
  and Frame2 *contains* X
  and Frame2 *contains* Y
  and X *is* car
  and Y *is* man
  and X *to_the_right_of* Y *in* Frame2

One way to answer this query is to allocate two frames that match the query specifications and then check their temporal relationships which can be specified in seconds or numbers of frames. Of course, we can increase the efficiency of this process by using temporal indices.

7 Conclusions

This paper has presented design, theory, and implementation of IFQ in this paper. We start with the motivation of IFQ through analysis of existing approaches of content-based image retrieval methods and query specification mechanisms. Then, we present our problem statement as the investigation and development of techniques that address three essential features of media retrieval: (1) Object-based image retrieval, (2) flexible visual query interface, and (3) query specification assistance. The contributions of our work include integrating various approaches and techniques to support the above features. IFQ (In Frame Query) is developed to support object-based media retrieval. It serves as a visual user query interface and query generator and can run on various DBMSs that support user defined functions and data types (part of the SQL3 standard).

References

[1] G. Amotoa, G. Mainetto, and P. Savino. A model for content-based retrieval of multimedia data. In *Proceedings of the Second International Workshop on Multimedia Information Systems*, West-Point, New York, September 1996.

[2] 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.

[3] 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.

[4] N. H. Balkir, E. Sukan, G. Ozsoyoglu, and Z.M. Ozsoyoglu. Visual: A Graphical Icon-Based Query Language. In *Proceedings of 12th International Conference on Data Engineering*, March 1996.

[5] 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.

[6] Venkat N. Gudivada, Vijay Raghavan, and Kanonluk Vanapipat. *Multimedia Database Systems: Issues and Research Directions*, chapter A Unified Approach to Data Modeling and Retrieval for a Class of Image Database Applications, pages 37–78. Springer-Verlag, New York, 1996.

[7] Venkat N. Gudivada and Vijay V. Raghavan. Content-Based Image Retrieval Systems. *IEEE Computer*, 28(9):18–22, September 1995.

[8] Kyoji Hirata, Yoshinori Hara, H. Takano, and S. Kawasaki. Content-Oriented Integration in Hypermedia Systems. In *Proceedings of 1996 ACM Conference on Hypertext*, March 1996.

[9] 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.

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

[11] Wen-Syan Li, K. Selçuk Candan, and K. Hirata. SEMCOG: An Integration of SEMantics and COGnition-based Approaches for Image Retrieval. In *Proceedings of 1997 ACM Symposium on Applied Computing Special Track on Database Technology*, San Jose, CA, USA, March 1997.

[12] Haruhiko Nishiyama, Sumi Kin, Teruo Yokoyama, and Yutaka Matsushita. An Image Retrieval System Considering Subjective Perception. In *Proceedings of the 1994 ACM SIGCHI Conference*, pages 30–36, Boston, MA, April 1994.

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