SEMCOG: A Hybrid Object-based Image and Video 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, USA.

K. Selçuk Candan*
Computer Science and Engineering Dept.
Arizona State University, Box 875406
Tempe, AZ 85287-5406, USA.

Multimedia data types, such as image and video, are structurally more complex than traditional data types. We view an image 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). In this paper, we first present a framework for modeling images and videos using object-based hierarchical structures. The proposed structure supports image retrieval at both whole-image and object levels. We show how to extent the object-based image modeling framework to video retrieval at object, frame, action, and composite action levels. The advantages of our approach are as follows: (1) hierarchical model allows translation of video retrieval queries to image retrieval queries with additional temporal relationships, (2) concept definitions release users from learning the system and implementation details and the language syntax, and (3) IFQ, a visual query interface, specifically designed for object-based image retrieval is extended for video retrieval query interfaces.

1. Introduction

Multimedia data, such as images and video, are very rich in terms of their visual and semantic contents. Especially video is can be considered structurally the most complex media as it consists of objects as well as frames and frame sequences linked with temporal relationships.

Image and video retrieval are key issues in many application domains, including TV stations, news agencies, and advertisement agencies. Content-based image and video retrieval is also important in digital library development. In [29], we presented the techniques we developed for modeling, querying, and retrieving image data. In this paper, we first provide an overview of the image model that captures image contents at both whole-image and object levels. We, then, extend this framework to handle video data. We give a formal definition of a video query language suitable for the proposed framework. We finally present the processing techniques we utilize for video queries.

1.1. Images

We categorize existing approaches to image retrieval as follows:

- *Browsing and navigation approach.* Users are presented with directories, each containing a single category of images.
- *Syntactic keyword-based approach.* Users provide keywords. The image retrieval system presents images whose captions or keyword lists contain the specified keywords.
- *Descriptive semantics-based approach.* Users specify *semantics* of the image for retrieval. This approach requires the semantics of the query and images to be explicit so that the query processor can determine the matching. The weakness of this approach is that images may be hard to describe using textual descriptions.
- *Cognition-based approach.* Users pose queries by providing sketch or image examples. The retrieval system finds images which are visually similar to these examples. One major disadvantage of this approach is the low precision and recall due to fuzziness of users' drawings and limitations of the image recognition techniques. Another weakness is that it cannot support queries on generalized concepts, such as *transportation*.

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

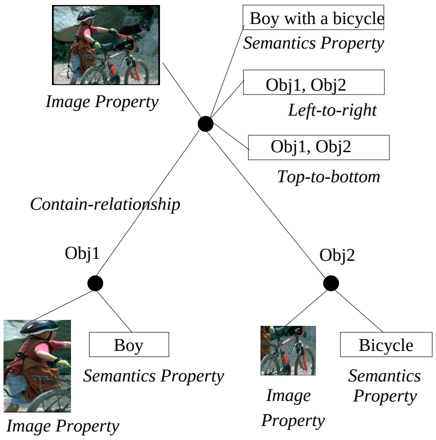

FIG. 1. Hierarchical image model

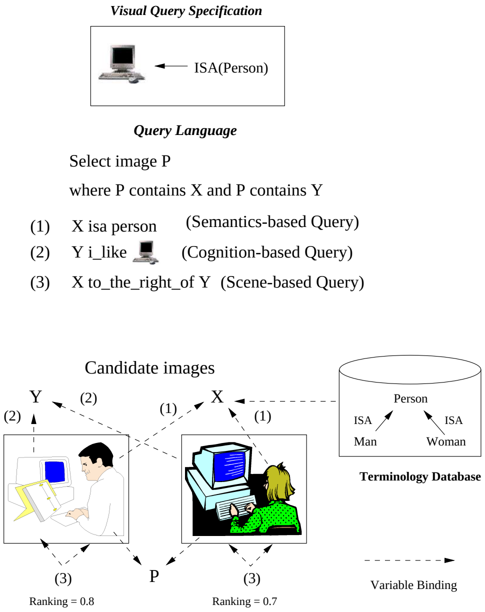

FIG. 2. Example of query processing

Our approach is to model images as hierarchical structures and, consequently, support queries based on combinations of visual characteristics and semantics at both the image and object levels. To represent the image contents and structures, we model each image as a compound object with **(1)** an image property (or a set relevant visual features), **(2)** a set of (fuzzy) semantic properties, and **(3)** spatial relationships (Figure 1). A compound object may contain multiple components. These component objects also have both image and semantics properties. This modeling schemes integrate semantics and cognition-based approaches to achieve greater flexibility. Hence, we named our system as SEMCOG (SEMantics and COGnition-based Approaches).

This model allows users to specify queries of the form "*retrieve all images in which there is a person who is to the right of an object that looks like the image X*" (Figure 2). This query contains three types of subqueries: **(1)** semantics-based, **(2)** cognition-based, and **(3)** scene-based. We will explain these corresponding predicates in greater detail in Section 3.

Figure 2 also shows two candidate images for the query. The arrows in the figure indicate the bindings of the variables X, Y, and P with objects and images. SEMCOG sorts the retrieved images according to their relevance to the query. The relevance value of an image is the degree of matching between the image and the query with respect to the semantic, visual, and spatial criteria. In Section 6.1, we describe how we compute such relevance values.

1.2. Video

Most content-based video retrieval systems consider video as a static, hierarchical structure composed of scenes and shots. Each shot corresponds to a visual change in the content of the stream. Scenes are generally contiguous sequences of shots that are semantically connected. Consequently, this structure is mainly visual. However, video retrieval systems based on a static hierarchy also use it for representing the information hierarchy. They associate shots or scenes with semantic descriptions (i.e. keywords or natural languages). Video can then be retrieved using these textual descriptions. This approach, however, has some weakness:

- It assumes that the information of users' interest are known in advance and can be pre-annotated on the shots and scenes. However, except in some applications, this assumption is not true. Consequently, out work does not rely on shot and scene annotations.
- Since, each pre-extracted shot or scene semantics require extra storage, the number of semantics that can be pre-extracted in this way is limited. Instead, we pre-extract information from frames and use temporal information to dynamically construct scenes that users are interested in.
- The information that users are interested in may be distributed in a series of non-contiguous scenes and shots. A scene-based hierarchy may not be able to capture this. Our approach does not rely on such a static hierarchy for retrieval.

Retrieval of video by providing examples may also not be adequate. Since there may be multiple entities (including the properties of the camera, such as the position, angle, and motion) of interest in a given video sequence, it is hard to describe the content of interest a video sequence by examples.

Therefore, we argue that most data models developed for video information are not adequate. They fail to capture essential aspects of the video simultaneously. To overcome the shortcomings of the existing models we propose a model which uniformly captures the semantic, visual, spatial, and temporal information contained in video, as well as the inherent fuzziness and subjectivity in media-based systems.

The video model we propose is based on *objects* and *actions*. An action is primarily defined in terms of the objects involved and the temporal information describing the (possibly non-linear) order in which these objects appear. Consequently, unlike the scene/shot based models, our proposed model is based on the logical structures of the video. Although a video has only one physical structure, it may have many logical structures, each with a different importance for different users. The proposed model provides higher flexibility.

Since, it is not always possible (and not desirable) to pre-extract all the logical structures that may be contained in a video stream, we allow users to specify (through queries) logical structures that the system will construct on-the-fly. Semantic and visual signatures of the *objects*, spatial and temporal constraints relating objects, and actions, can be used to specify the content-based video information. This flexibility is one of the primary advantages of SEMCOG video retrieval system. In Section 3, we describe the object- and action-based language that enables users to benefit from the proposed flexible framework.

In order to facilitate queries in such a highly expressive environment, the visual user interface model must match the language very well. It also must be simple enough. For this purpose, we propose the IFQ (In Frame Query) query interface which follows the hierarchical structure of the underlying model.

2. Image and Video Models

In this section, we first describe the image model in greater detail. We, then, discuss the video model and describe how it is built on top of the image model.

2.1. Image Model

An experimental study conducted by H. Nishiyama *et al.* [38] shows that users are viewing paintings or images using two primary patterns. The first pattern consists of viewing the whole image roughly, focusing only on the layout of the images of particular interest. The second pattern, on the other hand, consists of concentrating on specific objects within the image. However, many existing approaches treat an image as an atomic entity. In such systems the image retrieval is usually based on feature vectors extracted from the whole of the image. We argue that both whole image- and object-based image matching are essential. Also, since images and the component objects have both semantics and visual attributes, we also argue that image retrieval using either semantics and visual matching *alone* is not sufficient.

Consequently, we interpret an image or an object in it based on both semantic and/or visual 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. We refer to the semantic interpretation and visual interpretation of images as *semantic property* and *image property*, respectively.

Figure 1 in Section 1 shows an image, *Boy_bike.gif*, whose structure is viewed as a hierarchy with two image components (i.e. boy and bicycle). These components are identified based on color/shape region division (visual interpretation) and their real world meanings (semantic interpretation).

The relationship between an image and its components is *contains*. For the example shown in Figure 1, the image *Boy_Bike.gif* contains two component objects, namely, *Obj1* and *Obj2*. *Obj1* has *left-to-right* and *top-to-bottom* spatial relationships with *Obj2*. The spatial relationships described here are similar to the 2D-String representation described in [27]. At the whole-image level, the image property is *Boy_Bike.gif* itself. The corresponding semantic property, on the other hand, is the description of the image, i.e. "*boy with a bicycle*." At the object level, the *Obj1* and *Obj2* both *image* and *semantic properties* (Figure 1). Each object can be further modeled with additional sub-objects. The number of levels in the object hierarchy depends on users' need or the image resolution.

Ideally, the semantic and visual interpretation of an image should be consistent. However, since the current image recognition techniques are usually limited, these two interpretations may be inconsistent. For example, the semantic interpretation of an image of a man may be *man*, whereas its visual interpretation of this image, may be only *human*. In this example, the visual interpretation is not as definite as the the actual semantics. Furthermore, the same image may be identified as a *woman* by another engine. In this case, the result is as definite, but incorrect, as the actual meaning of the image. Yet, in another case, the image processing engine may identify the image as a *man*, but with a low confidence value. In this case, the image and semantic interpretations are consistent, but the semantic interpretation is not definite.

2.2. Video Model

In this section, we present the video model that SEMCOG uses for video information retrieval. The video model is based on the image model presented in the previous section. Consequently, it also consists of an hierarchy of information associated by additional temporal relationship.

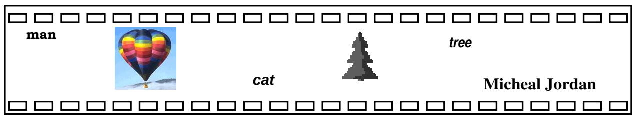

FIG. 3. Object Level

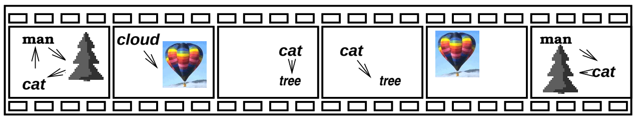

FIG. 4. Frame Level

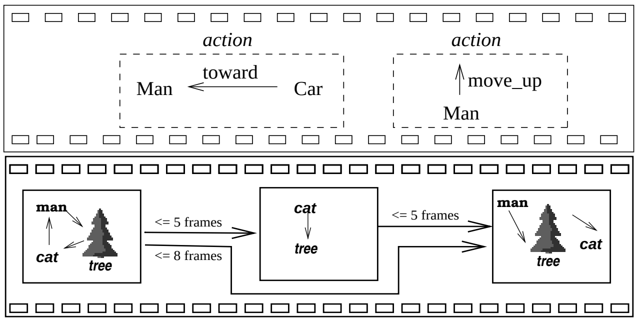

FIG. 5. Simple Action Level

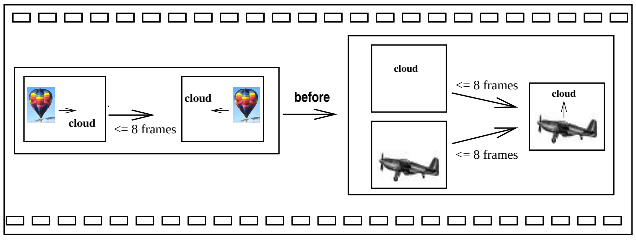

FIG. 6. Composite Action Level

Each level in the video model hierarchy provides a different granularity of information to answer different types of video retrieval query. For instance, a simple query, such as "*retrieve all video clips containing a submarine*", that are simply concerned with video clips which contain a particular object. A more complicated query, such as "*retrieve video clips containing a submarine combat scene*", need different levels of information. The hierarchical model provide a modular and flexible structure that can answer various types of queries.

The SEMCOG video model describes different types of information, including object-based, spatial, instance-based temporal and interval-based temporal information. The object-based and spatial information are similar to the image model we presented in the previous subsection. The instant- and interval-based temporal information, on the other hand, are introduced only for the video model. We will elaborate on the temporal information in Sections 2.2.3 and 2.2.4.

The hierarchical video model consists of four levels (Figures 3-6), i.e., the object, single frame, simple action, and composite action levels.

Object level. At the lowest level of the hierarchy, the information modeled is the semantics and image contents of the objects in the video. Example queries that can be answered by the information at this level include "*Retrieve all video clips that contain an object similar to the* `example image`" and "*retrieve all video clips that contain a submarine.*"

Frame level. At the second level of the hierarchy, the concept of a video frame is introduced. The additional information maintained at this level is spatial relationships among objects within a frame and other meta-information related to frames, such as "being a representative frame for a shot or a pre-extracted action" and frame numbers (a representative or key frame of a shot is the frame which describes the content of a shot the best). An example query that can be answered by the information at this level is "*retrieve all video clips that contain a frame in which there are a man and a car and the man is to the right of the car.*"

Simple action level. The next level of the hierarchy introduces the concept of time, i.e., the temporal relationships between individual frames. The temporal relationships are added to the model through implication of frame numbers. Since each frame corresponds to distinct time points within the video, the temporal relationships introduced at this level are instant based. Multiple frames with temporal relationships construct actions. For example, an action of "*a torpedo launch from a submarine*" can be defined as a frame sequence as a frame with a submarine followed by a frame with a submarine and a torpedo and a frame with only a torpedo. Another example of an action "*a man moving to the right*" can be defined as a frame in which there is a man on left followed by a frame with a man on the right side. Actions are defined as video frame sequences that have associated action semantics. The sequence of frames associated with an action definition is called an *extent*. We will formally define the concept of an extent later in Section 2.2.3.

An example query that can be answered by the information at this level is "*retrieve all video clips that contains two frames where the first frame contains a submarine and a torpedo followed by a frame contains an explosion and these two frames are at most 10 seconds apart.*" Two more complicated queries that can be answered by the information modeled at this level are "*retrieve all video clips that contain an action of torpedo launch from a submarine*" and "*retrieve all video clips that contain an extent of that a man is moving the right.*"

Composite action level. This level introduces the concept of *composite actions*. A composite action is a combination of multiple actions with instant- or interval-based time constraints. For example, a composite

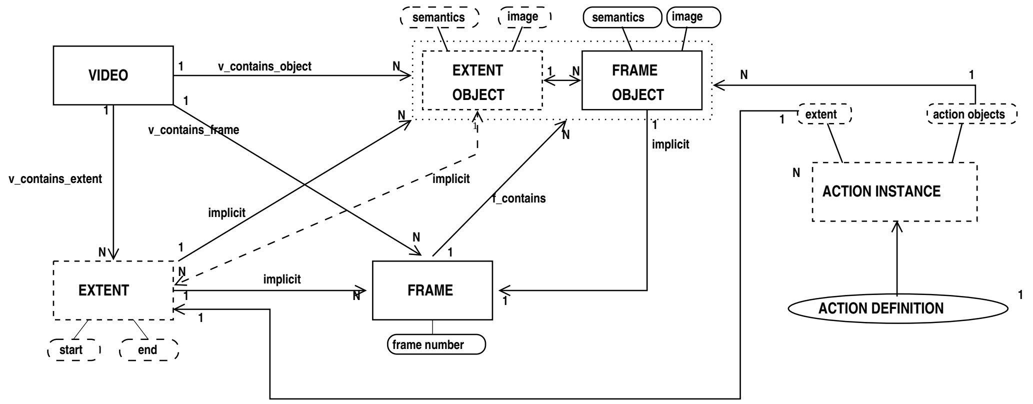

FIG. 7. High level relationships between the video entities

action of "*a submarine combat*" can be represented with combinations of actions "*submarine moving to the right*," "*submarine moving to the left*," "*a torpedo launch from a submarine*," "*explosion*," and interval-based time constraints associated with these actions. The interval-based time management was introduced by Allen in [3] and studied by many researchers, including [33].

Note that the frames discussed above may correspond to the actual video frames or the representative frames (or key frames) of the shots. Since, it results smaller number of frames, the key frame based approach is used by many video retrieval systems. However, since representing a shot with a single key frame results in the loss of all the temporal information within the shot, the information granularity also decreases. One solution to this problem is maintaining one key frame per shot as the visual representative along with the object trajectories that describe the motion of the objects within the shot. Consequently, both the storage and visual indexing requirements can be reduced and the temporal and spatial information can be maintained. Since, theoretically, there is no difference between the actual frame or key frame based approaches, in this paper, we view a video stream v as a sequence of frames $\langle f_1, \ldots, f_n \rangle$, where n denotes the length of the video. We also define a *frame sequence* $\langle f_s, \ldots, f_e \rangle$ as a consecutive set of frames identified primarily by its start (f_s) and end (f_e) frames.

Figure 7 depicts the relationships between video, frame, object, extent, and action concepts. Next, we discuss the introduced levels and these concepts in greater detail.

2.2.1. Object Level

At the lowest level of the hierarchy, we have information related with the objects within a video clip. Figure 8 shows that the video clip 20 contains two visible objects: an *airplane* and a *balloon.* This information is usually extracted in advance and stored so that the queries on the objects in video clips can be treated as as semantics- or cognition-based matching of image objects.

2.2.2. Frame Level

At the frame level of the hierarchy, we have the *spatial relationships* of the objects within a single frame. Hence, at this level, the information modeled is not objects but properties of the frames. As we informally introduced earlier, a video frame is an atomic piece of video. The closest analogy of a video frame is a static image. The information of a frame can be modeled as shown in Figure 9. Figure 9 shows that frame 20 contains two visible objects: *man/woman* and *computer/television.*

Note that one can define a spatial relationship between objects which are not in the same frame as well. Since there is no time information at the frame level, such spatial relationships can be described using the information modeled in this level.

2.2.3. Simple Action Level The concept of time and instant-based relationships between frames are introduced at the third level of the hierarchy. These relationships include *before, after*, and *equal.* Instant based relationships described using difference constraints [7, 8]: let f_i and f_j be two frames, then $i - j \leq \delta_1$ and $j - i \leq \delta_2$ is enough to describe all (non-disjunctive) instant-based relationships between two frames. For instance, the statement "*f_i and f_j are the same frame*" can be described as $((i - j \leq 0)\, and\, (j - i \leq 0))$. Similarly, "*$f_i$ occurs more than 5 frames before f_j*" can be described as $(i - j \leq -5)$.

The followings are example queries that can be posed at this level:

- "*Retrieve all video clips which contain two video frames f_1 and f_2, where f_1 contains an object that looks like* `car.gif`*, f_2 contains an object whose semantics is similar to* `man`*, and f_1 is within 10 frames of f_2.*"

- "*Retrieve all video clips and the frame number f_2 where the video clips contain two video frames f_1 and f_2, where f_1 contains a* `man` *and f_2 contains an object whose semantics is a kind of* `transportation`*, and f_1 is within 10 frames of f_2.*"

Before moving to the next level in our hierarchy, we introduce some new concepts that are related to the information modeled at the simple action level:

Smallest extent of a set of frames. Given a set S of video frames $\{f_{i_1}, f_{i_2}, \ldots, f_{i_k}\}$, we can identify a frame number s which is equal to the smallest of the frame indices i_1 through i_k, and a frame number l which is equal to the largest of the frame indices i_1 through i_k. We call the frame sequence defined as $\langle f_s, \ldots, f_l \rangle$ the *smallest extent* (se_S) of the set S. We also call f_s as *the first frame*, and f_l as *the last frame* of se_S. Please note that smallest extent of a set of frames are not necessarily consecutive.

Equivalence of frame objects. Let f_i and f_k be two video frames. Let also o_i be an object identified in f_i and o_k be an object identified in f_k. The object o_i is said to be *equal to* o_k if their respective visual features are similar to each other above a threshold value (Figure 10). If in f_k there are two alternative objects o_{k_1} and o_{k_2} that are both similar to o_i above a threshold, then the one whose spatial location in f_k is closest to the spatial location of o_i in f_i is said to be *equal* to o_i. A set E of equal objects define an equivalence class which is identified by a composite object $o_{com\{E\}}$.

Extent objects. Given an extent se_S, and a set of object equivalence classes defined within the extent se_S, we call the composite objects that identify the equivalence classes the *extent objects*. Given an extent se_S, and a composite object o_{com} defined within this extent, $first(o_{com})$ denotes the first frame in se_S that o_{com} was visible, and $first(o_{com})$ denotes the last frame in se_S that o_{com} was identified. (Figure 11).

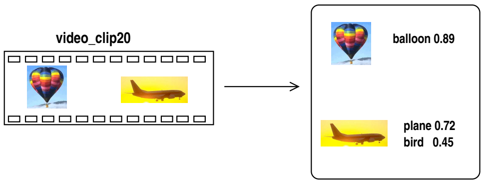

FIG. 8. Video objects

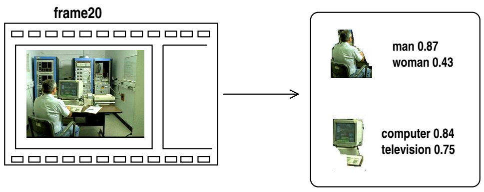

FIG. 9. Frame objects

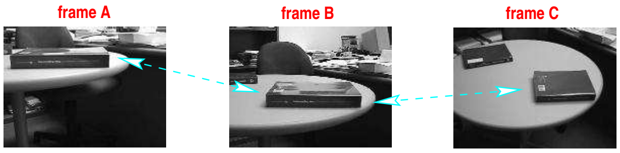

FIG. 10. Object equivalence

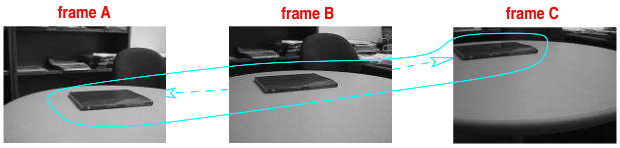

FIG. 11. Extent object

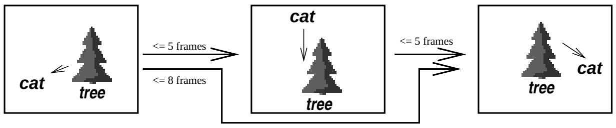

FIG. 12. Action definition

Actions, action instances, and action objects. Actions, such as "*a man is walking to the right,*" "*a man is getting closer to a car,*" or "*Michael Jordan is making a slam dunk,*" can be expressed as queries at the simple action level. Each action definition may consist of two or more frames, the objects that are of interest within these frames, the semantics or image contents and layouts of these objects, and the instant based temporal relationships among the frames (Figure 12). Each answer to such a query (or action definition) is an *action instance.*

Each action instance has an associated extent, and hence, it also has extent objects that can be queried. The extent objects that are primarily involved in the actions are called *action objects.* For instance, in the extent described by the action "*Michael Jordan is making a slam dunk,*" Michael Jordan is an action object. However, an object which is identified as an other player within the corresponding extent is not an action object for this particular action.

Action instances can be defined either through description of action objects and spatial and temporal relationships or manually by the action authors. A manual definition involves providing an extent for the action and the corresponding action objects. Similarly, the action instances that have been extracted as answers to the queries can be stored as extents and sets of action objects to provide faster access in further queries that may involve the same action instance.

2.2.4. Composite Action Level At the fourth level of the hierarchy, we have interval based temporal relationships among extents. There are 13 qualitative interval based relationships as described by Allen in [3]. These relationships are only qualitative, i.e., they do not have any temporal values, such as the length of an interval, the distance between the start times of two in-

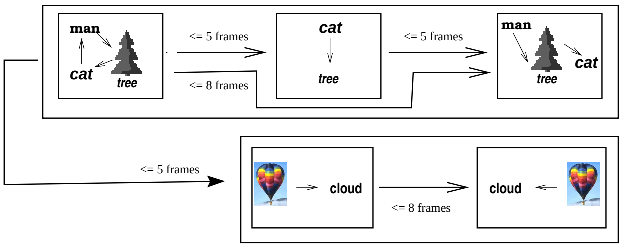

FIG. 13. Composite action definition

tervals and so on. We use these 13 relationships along with distance constraints we introduced earlier (we apply these constraints to the start and end times of the extents) to increase the expressive power.

The extents that are involved in interval based relationships at this level can be specified by

- specifying the start and end frames of the extents to denote an interval of interest,
- providing a query which defines an action (hence an extent),
- providing an action name which corresponds to the action definition query, or
- providing a canned action instance(i.e., an extent and the corresponding action objects that were processed and stored earlier).

Composite actions are defined using two or more action definitions, action names, action instances, extents, and the interval based temporal relationships among them (Figure 13). Each answer to a composite-action definition query is a *composite-action instance*. The action objects of the involved actions and the other frame or extent objects used in the definition are called *composite-action objects*. Similar to the actions, composite actions also have extents.

Table 1 summarizes the levels of video information in the proposed model.

In the next section, we will describe the image and video retrieval query languages, CSQL and VCSQL, respectively.

3. Query Languages: CSQL and VCSQL

In order to maximize the benefit of the rich data model presented earlier, we need to have a query language that reflects the same flexibility and richness. In this section, we introduce the syntax of CSQL (Cognition and Semantics-based Query Language) and VCSQL (Video Cognition and Semantics-based Query Language). For the video queries related to only the two lower levels (object and frame levels), the video retrieval queries are similar to image retrieval queries.

TABLE 1. Levels of video information

Level	Information represented
Object	Visual and semantic contents of the objects
Frame	Intra-frame spatial relationships of objects
Simple Action	Inter-frame spatial relationships of objects, instant-based temporal relationships between frames, and actions
Composite Action	Interval-based temporal relationships between time-intervals, extents, and actions.

3.1. Cognition and Semantics-based Query Language - CSQL

There are three types of entities used in CSQL: *semantic entities*, *image entities*, and *dual entities*. Each entity can be further classified into *compound* and *atomic* as shown in Figure 1:

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.

3.1.1. Selection Criteria

In this section, we introduce the predicates CSQL for object-based image retrieval. We categorize query selection criteria based on the types of entity and operators involved:

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 an image processing engine, such as COIR [22].

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 following examples show how entities and above predicates are used:

- *Object1 is man*: In this sentence (query component), *man* is a *semantic entity* and *Object1* is a *dual entity.* However, only *semantic interpretation* of *Object1* is compared with *man*.
- *Object1 i_like* `sketch`: In this query component, *Object1* is a *dual entity.* `sketch` is a sketch provided by a user, thus, an *image entity.* However, *i_like* (i.e. image like) specifies comparison of two arguments' visual identities.
- (*Object1 s_like man*) *and* (*Object1 i_like* `sketch`): In this query component, *Object1* is a *dual entity.* In the first comparison the *semantic identity*, whereas in the second comparison, the *image identity* of *Object1* is used.

TABLE 2. Predicates and their usage

Predicate	1^{st} arg.	2^{nd} arg.	3^{rd} arg.
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

The predicate utilization of is explained in Table 2 (s, d, and i stand for *semantic entity*, *dual entity*, and *image entity* respectively).

3.2. Video Cognition and Semantics-based Query Language - VCSQL

In this subsection, we introduce the predicates and other constructs used by the VCSQL (Video Cognition and Semantics-based Query Language). We also provide examples to illustrate its capabilities and expressive power.

3.2.1. Frame Level Image or video objects can be interpreted either semantically or visually. Hence, at the object level, VCSQL has predicates similar to those introduced by CSQL:

- *Semantics-based selection predicates*: *is*, *is_a*, and *s_like*.
- *Cognition-based selection predicate*: *i_like*.

Unless specified by the user, the system uses the extent objects, instead of the frame objects, for matching. This means that, unless the user explicitly refers to a frame, the semantic and image similarities are measured using the extent characteristics of the objects. Theoretically, the semantics and the visual characteristics of an extent object should be equal to those of the constituent frame objects. However, in practice, there may be differences due to imprecise processing capabilities of the image processing and semantics identification engines.

Video frames are primarily identified by their frame numbers. Frames can also be specified by the objects they contain and the spatial relationships of those objects:

- *f_contains*: The first argument of this predicate is a frame and the second argument is an object. This predicate returns true if the object is contained in the frame which corresponds to the first argument.
- *Spatial relationship-based selection criteria*: VCSQL supports the following spatial predicates: *to_the_right_of*, *to_the_left_of*, *above_of*, *below_of*. These predicates take two objects and a frame, and they check if the specified spatial relationships between two objects holds in that frame.
- *Location-based selection criteria* (*at*): This predicate takes an object, a pair of integers (a coordinate), and a frame, and returns true if the specified object is at the specified coordinates in the corresponding frame.
- *Distance-based selection criteria* (*distance*): This predicate takes two objects, a frame, and an integer, and returns true if the distance of the two ob-

jects in the corresponding frame matches the given value.

VCSQL allows users to substitute a point coordinate, i.e, an $\langle x,y\rangle$ pair, in the place of an object. In such cases, the predicate will check if the relationship holds for the other input object and the provided coordinates. Given a query of the form

$to\_the\_left\_of(O, \langle 50,70\rangle, f)$,

assuming that the uppercase letters denote variables, and the lowercase letters denote constants, the system will return all the objects that are to left of the point $\langle 50,70\rangle$ in frame f.

As described in the previous subsection, the objects referred to are extent objects unless a frame is specified. For instance, the query

$s\_like(O_1, man)$ *and*
$i\_like(O_2,$ `man.gif`$)$ *and*
$i\_like(O_1, O_2)$

would return all frames in which an extent object, O_1, is to the left of an extent object, O_2. The semantics of the extent object O_1 is like *man* and the extent object O_2 is visually similar to `car.gif`. In the following query, however, the semantics of two frame objects are compared to each other:

$f\_contains(f_1, O_1)$ *and*
$f\_contains(f_2, O_2)$ *and*
$s\_like(O_1, O_2)$

The last predicate defined at this level checks if a given frame is similar to a given example image. This allows users to access video frames without paying attention to the individual objects contained within the frames:

- *f_i_like*: The first argument of the *f_i_like* is a frame and the second attribute is either a frame or an image. The predicate returns true if the visual contents of its two inputs match.

3.2.2. Simple Action Level An extent is a frame sequence primarily identified using a *start frame number* and a *last frame number*. However, extents can also be specified by a set of frames and instant based temporal relationships among them:

- *Instant-based temporal selection criteria*($before_{\leq}$, $before_{\geq}$, $after_{\leq}$, $after_{\geq}$, *within*): These predicates take two frames and an integer and check if the specified instant-based temporal relationship between those frames holds. For instance $before_{\leq}(F_1, F_2, 5)$ is true if F_1 is before F_2 and the temporal distance between them is less than or equal to 5 frames.

For instance, a query of the form "*find all the frame sequences less then or equal to 5 frames, in which a man moves to the right*" can be formulated as follows:

$is(O, man)\wedge$
$at(O, Coord_1, F_1)\wedge$
$at(O, Coord_2, F_2)\wedge$
$to\_the\_right\_of(Coord_2, Coord_1, F)\wedge$
$after_{\leq}(F_2, F_1, 5)$.

We use a special construct, called *extends*, to create extents from frames. The extents can then be stored or cached independently or used in other action definitions:

- *extends*: This construct takes an extent and a frame as its inputs, and it adds the provided frame into the extent, i.e. it modifies the start and end frames of the extent to include the new frame.

In the previous example, frames F_1 and F_2 can be converted into an extent by adding the following two lines to the above query:

$extends(E, F_1) \wedge extends(E, F_2)$.

When the above query is executed, E will contain the frame sequences that the user requested. The following predicates check for containment by extent:

- *Extent-based containment criteria*($e\_contains\_object$ and $e\_contains\_frame$): These predicates take an extent and an object or a frame, respectively, and they check if the later is contained in the given extent.

VCSQL also allows users to check the first frame and the last frame of an extent using the following predicates:

- *first*, *last*: These predicates take an extent and a frame as their inputs, and return true if the frame is respectively the first or the last frame of the given extent.

Queries that involve multiple frames can be converted into action definitions. An action definition must describe the action objects and the extents. For instance, in previous example, the user can convert the above query into an action definition by storing it as an action $man\_moves\_to\_the\_right$. In this case, O is the action object of this action and E is its extent. We represent this action definition by a new predicate of the form $man\_moves\_to\_the\_right(O, E)$.

3.2.3. Composite Action Level The temporal relationships between different extents or actions can also be used in querying. We use the following interval based relationships introduced by Allen [3]: *before, meets, overlaps, during, starts, finishes, equal*. These predicates take two extents and checks whether these extents satisfy the interval based relationship suggested by the predicate name.

Queries that involve interval based relationship between extents can also be converted into action definitions as described in Section 3.2.2. These action definitions can then be used in the definition of further action definitions.

Users can also describe the instant-based relationships between the *first* and *last* frames of action extents to create composite-actions.

3.2.4. Video Finally VCSQL has a set of predicates to retrieve videos that satisfy certain criteria:

- *v_contains_object*, *v_contains_frame*, and *v_contains_extent*: These predicates take a video and an object, a frame, or an extent, respectively, and they check if the later is contained in the given video.

For instance, a user may ask for the extents, contained in a video named $\texttt{video}_1$, that contain a man moving to the right:

$$v\_contains\_extent(\texttt{video}_1, E) \wedge is(O, \text{“man”}) \wedge$$
$$man\_moves\_to\_the\_right(O, E)$$

In this query, *man_moves_to_the_right* is an action definition specified earlier.

In this section we described the predicates CSQL and VCSQL provide for querying images and video, respectively. In the next section, we describe the IFQ visual query interface that facilitates query specification in SEMCOG.

4. IFQ: User Interface for Image and Video Retrieval

IFQ (In Frame Query) is shown in Figure 14. IFQ uses Tcl/Tk libraries and runs on X window systems. It is a visually rich query interface which allows users to pose queries using combinations of keywords, concepts, semantics, image examples, sketches, and spatial and temporal relationships in *frames*. As IFQ's name shows, the query specification in the IFQ window (or frame) visualizes the target images during the query specification process.

In the next subsection we give the details of the theory, design, and CSQL functionalities of IFQ. We describe the extensions to IFQ that facilitate VCSQL queries.

4.1. Query Specification for Image Retrieval

The query specification for image retrieval 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 smaller bullets.

While the user specifies the above query using IFQ, the corresponding CSQL query is automatically generated in the CSQL window. IFQ can check the consistency between the spatial relationship specifications and the actual layout on the screen. If there is a mismatch, IFQ rearranges the query objects on the screen according to the specifications. If there is a conflict in the layout specifications IFQ informs the user by highlighting the conflicting specifications.

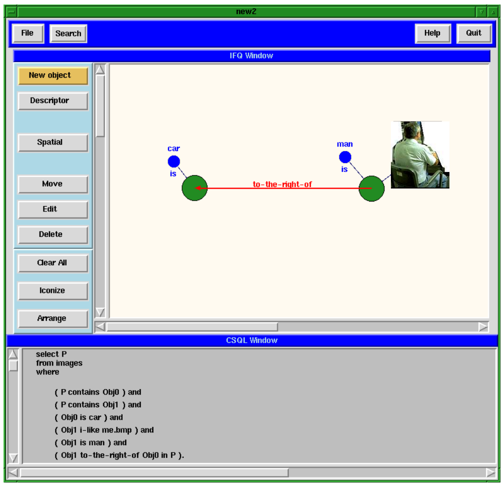

FIG. 14. IFQ specification window (top) and CSQL generating window (bottom)

4.1.1. Query Modeling and Query 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. Specification of an object is achieved by placing a large circle within the frame boundaries.

Object description. Users can further specify visual and semantic descriptions of an object by attaching to it smaller circles labeled with predicates: *is*, *is_a*, *s_like*, and *i_like*.

Spatial relationship. Users can describe the spatial relationships between the objects in the required images by attaching the circles corresponding to objects with edges labeled with the following predicates: *to_the_right_of*, *to_the_left_of*, *above_of*, and *below_of*.

Concept definitions for image descriptors. IFQ supports user defined concepts through combinations of visual examples, terms, predicates, and other concept definitions [31]. For example, users may define the concept of *transportation* as follows:

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

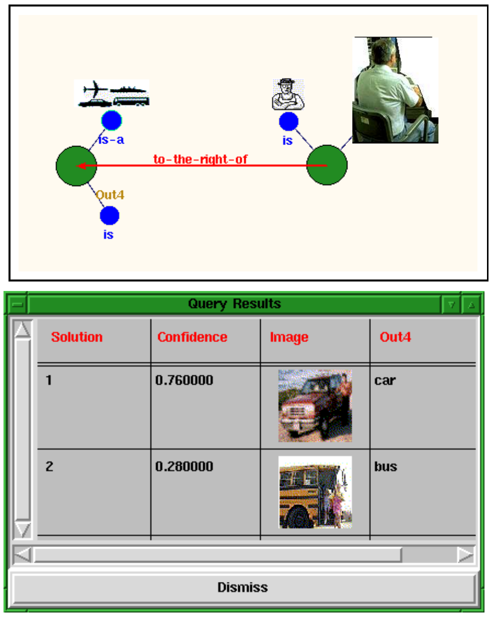

FIG. 15. Relaxed query for extracting semantics and its result

Semantics extraction and query relaxation. IFQ allows users to extract the semantics of the object. Figure 14 depicts a query that asks for an object which is "semantically similar to a man." The unbound descriptor in the figure means that the user wants to extract the semantics of the matching objects. Such a semantic may be *man*, *woman*, or *child*.

In Figure 15, the user relaxes the condition of "being a *car*" in Figure 14 to "being a kind of transportation." The user also introduces an unbound descriptor *out4* to check the actual semantics of the matching object. The corresponding CSQL query, generated by the IFQ interface, 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 15 shows two candidate images. One image contains a car and the other contains a bus. The results for *X.semantics* (*out4* as its label) are *car* and *bus*, respectively. In this example, the second candidate image has a lower confidence, because the human in the image can not be identified as a man.

Using collected statistics, IFQ provides feedbacks to the users to guide them in the query relaxation process.

4.2. Query Specification for Video Retrieval

(a) Object definition (b) Information extraction

FIG. 16. Object definition examples

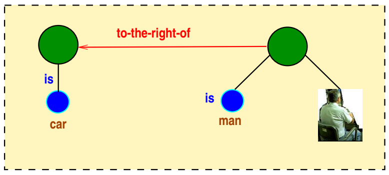

FIG. 17. Frame definition example

For the first two levels of VCSQL, the IFQ specifications are similar to those for the images. Figures 16 and 17 show the object and frame level query formulation, respectively. In Figure 17, the dashed line around the objects and spatial relationships denotes a frame.

At the frame level, unless specified by the user otherwise, an object's location is assumed to be *free*. Hence, a specified object can be found at any location within the frame. However, the user can *anchor* an object to a specific location within the frame. In this case, only those frames that have the given object at the specified location (or at its neighborhood) will be retrieved. The anchor corresponds to the *at* predicate in VCSQL.

4.2.1. Simple Action Level

As we introduced earlier, at this level, users can specify instant-based relationships among frames. The interpretation of the two frames and a line, annotated with $\leq$ ($\geq$) δ_f, leaving one of the frames and going into the other, is "*the later must occur after the former, but the temporal distance between them must be less than or equal to (greater than equal to)* δ_f *frames*" (Figure 18).

One problem in user interface modeling at the frame level is to find which frame objects in different frame definitions are intended to be equivalent, i.e. they correspond to the same extent object. In IFQ, instead of making the user explicitly state which objects should be accepted equivalent, all the frame object definitions that have the same (or similar above a threshold) structure are assumed to correspond to the same extent object. In some cases, this may lead into ambiguities in the query. Hence, the user also has the capability of overwriting system chosen equivalences.

The example in Figure 18 can be read as "*find all videos which contain two frames such that the first frame contains a man to the right of a car, the second frame contains the same car to the right of the same man, the second frame is after the first frame, and the distance between these two frames is less than or equal to 5 frames.*"

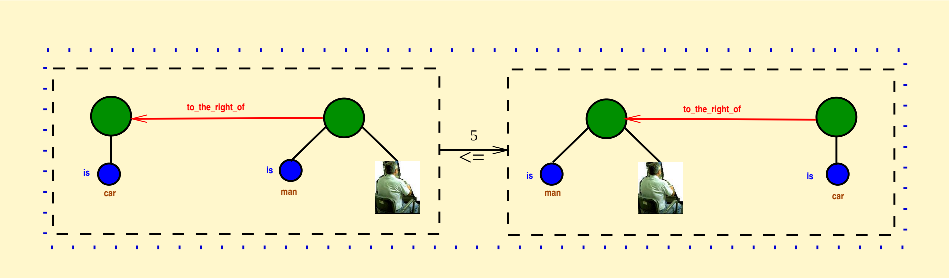

FIG. 18. Instant based temporal definition example

(a) Single action object

(b) Multiple action objects

FIG. 19. Action definition examples

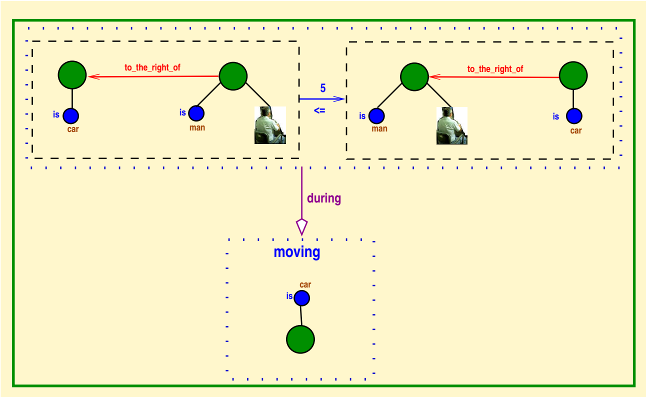

FIG. 20. Interval based temporal definition example and a complete video query

The dotted line around the frames in Figure 18 is the extent definition. The extent definition is necessary when the user is interested in learning about the start and end times of the corresponding frame sequence. In addition, to store a given instant based temporal query as an action definition, the user has to first formulate the query as am extent definition. For instance, the extent defined in Figure 18 can be stored in one of the forms shown in Figure 19. The action definition in Figure 19(a) contains one action object, whereas the action definition in Figure 19(b) contains two action objects.

Note that, in several aspects, the action definition for video is similar to the concept definition for image retrieval. First, describe higher levels of semantics using the already extracted semantics and visual information. Second, both of them increases the usability of the system by reducing the amount of work one has to spent in expressing complex queries. Third, both allows alternative definitions of the concepts and actions and they are open to improvement as more matches are accepted by the user.

4.2.2. Composite Action Level

At this highest level of the IFQ interface model, users can define interval-based relationships among extents. In additions, users can define instant-based relationships between the start and end times of different extents.

The interval-based temporal definitions are performed by linking two extents by arrowed lines annotated with *during*, *meets*, *starts*, *overlaps*, *finishes*, *before*, or *equal*. For instance, the interpretation of two extents and a line, annotated with *during*, leaving one of the extents and going into the other, is "*the former extent is during the later.*" The other relations also have similar interpretations.

In order to describe instant based relationships, on the other hand, user can treat the left-most vertical boundary as the *first* frame and the right-most vertical boundary as the *last* frame of the extent.

Figure 20 shows a composite action definition. It contains two extents. The first extent contains a car and a man switching their positions. The second extent contains a car which is moving. In addition, the first extent occurs sometime *during* the lifetime of the second extent. In this example, the action *moving* is a canned action, that is, it already has been defined and stored as an action definition containing one action object. In Figure 20, the user further describes this action object as a *car*. Hence, only those canned *moving* actions which has a man as the action object match the query.

IFQ also allows multiple canned actions to be kept under one name. For instance, two canned action definitions, *moving to the right* and *moving to the left*, may be

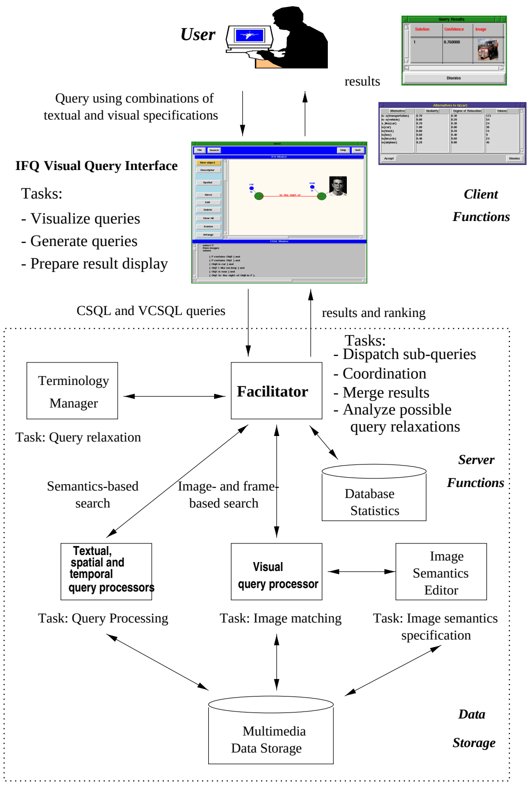

FIG. 21. System architecture of SEMCOG

put together under the action definition *moving*. When the system evaluates the action definition, it replaces this definition with the appropriate atomic action definitions.

Composite action definitions can also be stored as canned actions by drawing dotted rectangles around the definitions.

Figure 20 also shows a video query. The non-dashed rectangle around the definitions denotes a video. Hence, this query asks, unlike the previous queries which ask for extents, for a video.

5. System Architecture

Figure 21 shows the operation and data flows in SEMCOG. The individual components and their functionalities are as follows:

Image/Frame Data Query Processor is used to perform image and frame matching task. COIR (Content-Oriented Image Retrieval)[22] 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. To identify the object semantics, COIR needs to consult *Image Component Catalog* to match with existing image object samples to determine image semantics. *Image Semantics Editor* interacts with COIR to specify the semantics of an image.

SEMCOG stores both locations of objects and their *relative spatial relationships* (horizontal and vertical) as described in 2-D strings[10]. SEMCOG also stores *instant-* and *interval-based temporal relationships* and uses these to answer temporal queries.

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. We use a similarity-capable extension of WordNet[35] as the terminology manager.

Textual Data Query Processor processes queries concerning image or object semantics. All query statements except those containing *i_like* (image like) and *contains* are semantics-based.

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 6.

In addition to these, SEMCOG provides a visual query interface, IFQ[28], for query specification, query generation, result display, and feedback visualization.

6. Query Processing

A CSQL query, $Q(Y_1, \ldots, Y_n)$, consists of a set of fuzzy and crisp predicates, constants, variables, and conjunction, disjunction, and negation operators. The fuzzy predicates are based on the evaluation on different media criteria, such as color, shape, layout, keyword, etc., and they yield similarity values. For instance, for the query shown in Figure 22, there are five predicates: three image matching predicates and two spatial relationship predicates. The results to such a query is defined as an ordered set S_Q of n-tuples of the form $\langle X_1, X_2, \ldots, X_n \rangle$. The order of the set denotes the relevance ranking of the solutions. Integration of semantics-, cognition-, and spatial relation-based selection criteria introduce many new challenges:

- Traditional query execution methods need to be adapted to handle similarity values generated from image matching and semantics-based comparisons.

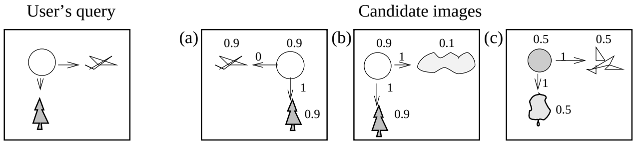

FIG. 22. Examples of matched images

- The system needs 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.
- The system needs to support query relaxations for image matching, semantics comparisons, and spatial relationships.

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 the predicate concept causes many challenges. When we substitute the values in an n-tuple t, of the form $\langle X_1, X_2, \ldots, X_n \rangle$, 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 cutoff 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 cutoff 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.

6.1. Relevance Ranking

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.

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.

The scoring function that we use for conjunctive queries 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.

6.2. Performance Considerations

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). Like most query optimizers in traditional and non-traditional database systems, the *Facilitator* periodically collects database statistics to improve query processing performance [1]. The statistics collected by the facilitator include, selectivities, predicate execution costs, and qualities (average relevances). One additional unique function provided by the *Facilitator* is to recommend users query reformulation alternatives and analyze possible query relaxation outcomes so that users can visualize distributions of related images. This functionality is described in [28].

A query predicate may have overloaded implementations each corresponding to a different binding pattern. Different implementations of a predicate may have drastically different execution costs. For instance, since it may require a pass over all images in the database to check if any of them contains an object matching the object O, a predicate of the form `image_contains_object_at (Image *I, Object O, Coord *C)` may be a very expensive operation (those arguments thatare marked with * are outputs). On the other hand, `image_contains_object_at(Image I, Object *O, Coord C)` may be significantly cheaper as it only needs to extract an object from a single image.

Different implementations of a given user-defined function may also have different *result qualities* [29, 30] or information contents. For instance, if we take the two implementations of `image_contains_object_at` discussed above, `image_contains_object_at(Image *I, Object O, Coord *C)` is likely to have a quality less than 1.0, as it may allow non-exact matches between given object and image objects. The user defined function,`image_contains_object_at(Image *I, Object O, Coord *C)`, on the other hand, is likely to return a quality value equal to 1.0 as it is extracting, very deterministically, an object from a given image at a given coordinates. Note that this implies that we may have cost/quality trade-offs.

Relational query optimizers generally assume that join operations that merge information from multiple tables are the most expensive operations. In database systems which allow external predicates, however, the query optimizer has to learn about the properties of the corresponding user defined functions. User-defined functions may be more expensive than most join operations. The cost- and quality-based query optimization techniques were described elsewhere [9].

6.3. Processing Video Queries

Video retrieval in SEMCOG is analogous to image retrieval. Most of the video queries, especially those that correspond to the frame level, can be converted into image-based queries and processed as such.

Since object-level queries do not have associated frames, they are processed using the aggregated semantics and visual information contained within extent objects. For instance, the following query, which asks for videos which contain an object that looks like `car.gif`, uses the extent objects for matching not the frame objects.

select video V
where (V v_contains_object O) and
(O i_like `car.gif`*).*

Hence, it is important to choose an aggregation method that extracts the dominant features of all frame objects that correspond to a given extent object. This process can be explained as follows: Let e_i be an extent object which corresponds to n frame objects, $o_i = \{o_{i,1}, o_{i,2}, \ldots o_{i,n}\}$. Let v_j, $1 \leq j \leq n$, denote the m dimensional visual feature vector corresponding to object $o_{i,j}$. An m dimensional feature vector consists of shape and color descriptions. Note that, since each frame object, $o_{i,j} \in o_i$, corresponds to the same real-life object, their visual characteristics will be similar to each other; i.e., in an m dimensional feature space, the feature vectors, $v_1, \ldots, v_n$, will be clustered. The feature vector that represents the cluster the best is assigned as the visual characteristics of e_i. A similar process is also applied to the semantics of the extent objects.

The use of extent objects, instead of the frame objects, result in considerable savings in the query processing time.

When, the user asks frame-level queries, however, using extent objects may not be enough. For instance, the following query asks for videos which contain a frame which contain an object that looks like `car.gif`.

select video V
where (V v_contains_frame F) and
(F f_contains O) and
(O i_like `car.gif`*).*

There are various ways to process this query, each with different associated costs. In order to reduce the query processing cost, we process this query as follows:

1. Find all extent objects that looks like `car.gif`. Let E be the set of extent objects that passed the test.
2. For every extent object, $e_i \in E$, compare each corresponding frame object, $o_{i,j} \in o_i = \{o_{i,1}, o_{i,2}, \ldots o_{i,n}\}$, with `car.gif`. Let F be the set of frame objects returned at the end of this process.
3. For every frame object $f \in F$, return the corresponding frame and video numbers.

The first step in the above procedure filters out irrelevant extent objects, thereby it reduces the number of actual frame objects that need to be matched against `car.gif`.

Processing simple and composite action based queries require reasoning about temporal relationships between frames. One way to process such queries is to process the object and frame level subqueries first and, then, to use the temporal query conditions to filter out irrelevant answers.

select video V, F_1, F_2
where (V v_contains_frame F_1, F_2) and
(F_1 f_contains O_1) and
(F_2 f_contains O_2) and
(O_1 i_like `car.gif`*) and*
(O_2 s_like man) and
(F_1 - $F_2 \leq 5$).

The above query is processed as follows:

1. Find all extent objects that looks like `car.gif`. Let E_1 be the set of extent objects that passed the test.
2. Find all extent objects that are semantically similar to man. Let E_2 be the set of extent objects that passed the test.
3. For every extent object, $e_i \in E_2$, compare each corresponding frame object, $o_{i,j} \in o_i = \{o_{i,1}, o_{i,2}, \ldots o_{i,n}\}$, with *man*. Let Fr_2 be the set of frame objects returned at the end of this process. Let also $Frames$ be the set of frames corresponding to the frame objects in Fr_2.
4. For every extent object, $e_i \in E_1$, let $o_i^* = \{o_{i,1}, o_{i,2}, \ldots o_{i,n}\}$ be those constituent frame objects that satisfy temporal condition. Compare each corresponding frame object, $o_{i,j} \in o_i^*$ with `car.gif`. Let Fr_2 be the set of frame objects returned at the end of this process.
5. For every pair, $\langle f_1, f_2 \rangle$, of frame objects, such that $f_1 \in Fr_1, f_2 \in Fr_2$, and the corresponding frame numbers F_1 and F_2, respectively, satisfy the condition, F_1 - $F_2 \leq 5$, return F_1 and F_2.

Clearly, this query can be processed in various other ways. However, the above scheme minimizes the costly image matching process as much as possible.

Composite action queries can also be processed in a similar way, by dividing the query into its subqueries and by using the temporal conditions to filter out irrelevant frames before the image matching operations are performed. However, since once identified, actions have associated extents, we utilize a data structure suitable for indexing interval-based information [2] in order to speed up interval-based joins and checking of interval-based query conditions. Whenever they are available, canned action objects and the corresponding extents are used to limit the search space, thereby reducing the query execution cost significantly. Let us assume that the query

select video V, E_1, E_2
(V v_contains_extent E_1, E_2) and
(O_1 i_like `car.gif`*) and*
(O_2 s_like man) and
car_moves_to_the_left(O_1,E_1) and
man_moves_to_the_right(O_2,E_2) and
overlaps(E_1,E_2).

contains two canned actions whose extents have been identified and stored for instantaneous access. To process this query SEMCOG would either (1) identify those canned actions that satisfy the object similarities and return those that overlap, or (2) find *car_moves_to_the_left* and *car_moves_to_the_left* extents that overlap and return those extents which satisfy the action object similarities. The choice between the two options depends on the selectivities, costs, and processing qualities of the predicates and the corresponding user defined functions.

7. Related Work

Virage[45] is an image retrieval system based on visual (such as *image primitives* color, shape, and texture) features. Virage's has an SQL-like query language extended by user-defined data types and functions. QBIC[36, 37] supports image retrieval using visual examples. The image matching is based on image features. Both Virage and QBIC support image matching and keyword-based retrieval functionalities at the *whole* image level. SEMCOG supports these functionalities also at the whole image level and it allows users to relax queries.

SCORE[6, 41] is a similarity-based image retrieval system. It uses a refined ER model to represent the contents of pictures and calculates similarity based on the E-R representations of images stored and the query specifications. SCORE does not support image-matching.

VisualSeek [42] 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 and sketch comparisons for image retrieval. However, VisualSeek does not support retrieval based on semantics at either the image level or the object level.

Chabot project[39] 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.*

MQL[25] is a multimedia query language that 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. CSQL provides additional predicates, such as

ISA and S-LIKE, to allow users to control the relaxation of query processing.

In [21], Gibbs *em al.* study how temporal multimedia data may be modeled using object based methods. Hjelsvold and Midtstraum [23] develop a data model for capturing video content and structure. Their approach is to extend relational databases to handle video data as well. In their model, video data is viewed as an hierarchy of structural elements (shots, scenes, etc.) based on video creation. However, our model approaches to the video data from a different perspective: it bases the hierarchy on the information contained in the video (i.e. the content of the video itself). Weiss *et al.* develop a video algebra which allows temporal and spatial combination of video segments and content-based access to video information ([46]). They use the close-captions and the user inserted attributes to identify segments of interest. In [12, 13, 20], Day *et al.* describe a spatial-temporal framework for accessing salient objects of a video. Their work, again, is based on the video structure. In [40], Oomoto and Tanaka proposes a language (VideoSQL) for accessing video data. [17] and [43] provide overviews of the state of the art video indexing and retrieval techniques. [14] proposes a spatial-temporal language which allows qualitative description of sequences of images.

A more related work is described in [5]. In this paper, Arisawa *et al.* propose a data model and a query language based on the identification of the objects in video frames. In this work, authors introduce the concept of *logical cuts* which is a step towards a logical structure of video. In [26], authors propose a video query language (CVQL) for content-based access to video databases. Their framework handles spatial and temporal relationships among content-objects. Similarly, in [2], we *et al.* propose a formal video model based on video objects, activities, and events, and describe a prototype system (AVIS). In a related effort, Hwang and Subrahmanian [24] develop an SQL-like language (VIQS) for querying video databases.

Recently, Liu *et al.* discussed temporal query processing issues in video databases [34]. Also, in [44], Yu and Meng discussed query processing issues in multimedia databases. Zadeh [47] and Fagin [15, 16] discuss scoring functions for fuzzy logical operators. [11, 18, 19] approaches for query processing in the presence of external predicates, or user defined functions which are very common in multimedia systems.

Most of the above temporal works are based on the interval based formalism proposed by James in F. Allen in [3, 4]. Allen differentiates between *properties*, *processes*, and *events*, but describes each of them using time intervals. He also provides thirteen qualitative temporal relationships (*before, meets, overlaps, during, starts, finishes, equal,* $before^{-1}$, $meets^{-1}$, $overlaps^{-1}$, $during^{-1}$, $starts^{-1}$, and $finishes^{-1}$) that can hold between two intervals. In [32], Little and Ghafoor propose an interval based model using object composition petri nets (OCPN), a modification of timed petri net model. Due to its higher flexibility and simplicity, this model is very useful in multimedia authoring and multimedia simulating applications. In [33], Little and Ghafoor introduce another interval based conceptual model which can handle n-ary relationships among intervals.

8. Conclusions

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

Acknowledgments

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

References

[1] S. Adalı, K.S. Candan, Y. Papakonstantinou, and V.S. Subrahmanian. (1996) *Query Caching and Optimization in Distributed Mediator Systems.* ACM SIGMOD'96, June 96.

[2] S. Adalı, K.S. Candan, S.-S. Chen, K. Erol, and V.S. Subrahmanian. (1996) *Advanced Video Information System: Data Structures and Query Processing.* ACM-Springer Multimedia Systems Journal, Vol. 4, pp 172-186, August 96.

[3] J.F. Allen (1983) *Maintaining Knowledge about Temporal Intervals*, Comm. of the ACM, 26(11), pp. 832-843, Nov. 1983.

[4] J.F. Allen (1984) *Towards a General Theory of Time and Action*, Artificial Intelligence, 23, pp. 123-154, 1984.

[5] H. Arisawa, T. Tomii and K. Salev. (1996) *Design of Multimedia Database and a Query Language for Video Image Data.* IEEE ICMCS, pp. 462-467, Japan, June 1996.

[6] Y.A. Aslandogan, C. Yu, C. Liu, and K.R. Nair. (1995) *Design, Implementation and Evaluation of* SCORE. In *ICDE'95*, Taiwan, March 1995.

[7] K.S. Candan, B. Prabhakaran, and V.S. Subrahmanian (1996) *A Framework for Supporting Multimedia Document Authoring and Presentation,* ACM Multimedia Conference'96, pp. 329-339, Boston, Nov. 1996.

[8] K.S. Candan, B. Prabhakaran, and V.S. Subrahmanian. (1998) *Collaborative Multimedia Documents: Authoring and*

Presentation. Special Issue of the Int. Journal of Intelligent Systems on Multimedia Computing Systems, 13(12), 1998.

[9] K.S. Candan and Y. Yang. (1997) *Cost- and Similarity-driven Query Optimization in the Presence of User-defined Functions,* CSE Technical Report, Arizona State University, TR-97-041, 97.

[10] S.K. Chang, C.W. Yan, D.C. Dimitroff, and T. Arndt. (1988) *An Intelligent Image Database System. IEEE Transaction on Software Engineering*, 14(5). pp. 681-688, 1988.

[11] S. Chaudhuri and K. Shim. (1996) *Optimization of Queries with User-defined Predicates.* VLDB'96, pp. 87-98, 1996.

[12] Y.F. Day, S. Dagtas, M. Iino, A. Khokar, A. Ghafoor. (1995) *Spatial-temporal Modeling of Video Data for On-Line Object-Oriented Query Processing.* IEEE ICMCS'95, pp. 98-105, Washington, DC, May 1995.

[13] Y.F. Day, S. Dagtas, M. Iino, A. Khokhar and A. Ghafoor. (1995) *An Object-Oriented Conceptual Modeling of Video Data.* ICDE'95, pp. 401-408, 1995.

[14] A. Del Bimbo, E. Vicario, and D. Zingoni. (1995) *Symbolic Description and Visual Querying of Image Sequences Using Spatio-Temporal Logic* TKDE, 7(4), pp. 609-622, 1995.

[15] R. Fagin. (1998) *Fuzzy Queries in Multimedia Database Systems.* ACM PODS'98, pp. 1-10, June 1998.

[16] R. Fagin. (1996) *Combining Fuzzy Information from Multiple Systems.* ACM PODS'96 , pp. 216-226, June 1996.

[17] B. Furht, S. W. Smoliar and H. Zhang. (1995) *Video and Image Processing in Multimedia Systems*, Kluwer Academic Publishers, 1995.

[18] J.M. Hellerstein and M. Stonebraker. (1993) *Predicate Migration: Optimizing Queries with Expensive Predicates.* SIGMOD'93, pp. 267-276, 1993.

[19] J.M. Hellerstein. (1999) *optimization Techniques for Queries with Expensive Methods.* ACM Trans. on Database Systems, 23(2) pp. 113-157, June 1998.

[20] M. Iino, Y.F. Day and A. Ghafoor. (1994) *An Object-Oriented Model for Spatio-Temporal Synchronization of Multimedia Information*, IEEE ICMCS'94, pp. 110-120, 1994.

[21] S. Gibbs, C. Breiteneder and D. Tsichritzis. (1994) *Data Modeling of Time-Based Media*, ACM SIGMOD'94, pp. 91-102, 1994.

[22] K. Hirata, Y. Hara, H. Takano, and S. Kawasaki. (1996) *Content-Oriented Integration in Hypermedia Systems*, ACM Conf. on Hypertext, pp. 11-21, Washington DC, Mar. 96.

[23] R. Hjelsvold and R. Midtstraum. (1994) *Modeling and Querying Video Data*, Proc. 1994 Intl. Conf. on Very Large Databases, pp. 686-694, Santiago, Chile.

[24] E. Hwang and V.S. Subrahmanian. (1996) *Querying Video Libraries.* Journal of Visual Communication and Image Representation, 7(1), pp. 44-60, 1996.

[25] S.C. Kau and J. Tseng. (1994) *MQL - A Query Language for Multimedia Databases.* In *Proceedings of 1994 ACM Multimedia Conference*, pp. 511-516, 1994.

[26] T.C.T. Kuo and L. P. Chen. (1996) *A Content-Based Query Language for Video Databases.* IEEE ICMCS'96, pp. 209-214, Japan, pp. 209-214, 1996.

[27] S.Y. Lee, M.K. Shan, and W.P. Yang. (1989) *Similarity Retrieval of ICONIC Image Database Systems*, Pattern Recognition, 22(6), pp. 675-682, 1989.

[28] W.-S. Li, K.S. Candan, K. Hirata, and Y. Hara. (1997) *Facilitating Multimedia Database Exploration through Visual Interfaces and Perpetual Query Reformulation,* VLDB'97, pp. 538-547, Greece, Aug. 1997.

[29] W.-S. Li and K.S. Candan. (1998) *SEMCOG: A Hybrid Object-based Image Database System and Its Modeling, Language, and Query Processing,* IEEE ICDE, Feb. 1998.

[30] W.-S. Li, K.S. Candan, K. Hirata, and Y. Hara. (1998) *Hierarchical Image Modeling for Object-based Media Retrieval*, Data and Knowledge Engineering, Elsvier Science, 27(2), pp. 139-176, July 1998.

[31] W.-S. Li, K.S. Candan, K. Hirata, and Y. Hara. (1997) *A Visual Query Interface and Query Generator for Object-based Media Retrieval.* IEEE Multimedia Systems Conference'97, pp. 353-361, Ottawa, Ontario, Canada, June 1997.

[32] T.D.C. Little and A. Ghafoor (1990) *Synchronization and Storage Models for Multimedia Objects*, IEEE J. on Selected Areas of Communications, 8 (3), pp. 413-427, April 1990.

[33] T.D.C. Little and A. Ghafoor (1993) *Interval-Based Conceptual Models for Time-Dependent Multimedia Data* IEEE TKDE, 5(4), pp. 551-563, Aug 1993.

[34] K.L. Liu, P. Sistla, C. Yu, and N. Rishe. (1998) *Query Processing in a Video Retrieval System*, ICDE, pp. 276-283, 1998.

[35] G. A. Miller. (1995) *WordNet: A Lexical Databases for English. Communications of the ACM*, pages 39-41, Nov.1995.

[36] W. Niblack *et al.*. (1993) *The QBIC Project: Querying Images by Content Using Color, Texture and Shape,* IBM Research Report, Feb. 1993.

[37] M. Flickner *et al.* (1995) *Query by Image and Video Content: The QBIC System*, IEEE Computer, 28(9), pp. 23-32, September 1995.

[38] H. Nishiyama, S. Kin, T. Yokoyoma, and Y. Matsushita. (1994) *An Image Retrieval System Considering Subjective Perception*, SIGCHI'94, pp. 30-36, Boston, MA, April 1994.

[39] Virginia E. Ogle and Michael Stonebraker. (1995) *Chabot: Retrieval from a Relational Database of Images,* IEEE Computer, Vol. 28, No. 9, September 1995.

[40] E. Oomoto and K. Tanaka. (1993) *OVID: Design and Implementation of a Video-Object Database System*, IEEE TKDE, 5(4), pp. 629-643, 1993.

[41] A. Prasad Sistla, Clement Yu, Chengwen Liu, and King Liu. (1995) Similarity based Retrieval of Pictures Using indices on Spatial Relationships, VLDB'95, Switzerland, Sept. 1995.

[42] J.R. Smith and S.-F. Chang. (1996) *VisualSEEk: A Fully Automated Content-based Image Query System*, ACM Multimedia Conference, pp. 87-98, Boston, MA, 1996.

[43] S.W. Smoliar and H. Zhang. (1994) *Content-Based Video Indexing and Retrieval,* IEEE Multimedia, 1, pp. 62-72, 1994.

[44] C.T. Yu and W. Meng. (1998) *Principles of Database Query Processing for Advanced Applications.* Morgan Kauffman Publishers, 1998.

[45] J.R. Bach, C. Fuller, A. Gupta, A. Hampapur, B. Horowitz, R. Jain, and C.-F. Shu. (1996) *The Virage Search Engine: An Open Framework for Image Management*, SPIE- The International Society for Optical Engineering: Storage and Retrieval for Still Image and Video Databases IV, CA, Feb. 1996.

[46] R. Weiss, A. Duda, and D.K. Gifford. (1994) *Content-based Access to Algebraic Video.* IEEE ICMCS, pp. 140-151, 1994.

[47] L. Zadeh. (1965) *Fuzzy Sets.* Information and Control, 9, pp. 338-353, 1965.