Multimedia Tools and Applications 6, 289–312 (1998)

A Language for Content-Based Video Retrieval

FOROUZAN GOLSHANI golshani@asu.edu
Department of Computer Science, Arizona State University, Tempe, Arizona 85287-5406

NEVENKA DIMITROVA nvd@philabs.research.philips.com
Philips Research, 345 Scarborough Rd., Briarcliff Manor, NY 10510

Abstract. We present an effective technique for automatic extraction, representation, and classification of digital video, and a visual language for formulation of queries to access the semantic information contained in digital video. We have devised an algorithm that extracts motion information from a video sequence. This algorithm provides a low-cost extension to the motion compensation component of the MPEG compression algorithm. In this paper, we present a visual language called VEVA for querying multimedia information in general, and video semantic information in particular. Unlike many other proposals that concentrate on browsing the data, VEVA offers a complete set of capabilities for specifying relationships between the image components and formulating queries that search for objects, their motions and their other associated characteristics. VEVA has been shown to be very expressive in this context mainly due to the fact that many types of multimedia information are inherently visual in nature.

Keywords: visual languages, content-based retrieval, digital video

1. Introduction

It is generally believed that the human mind is visually oriented. For those concepts that can be presented visually, people acquire information at a higher rate by discovering graphical relationships in complex pictures rather than plain text. Analogous to this human characteristic, graphical/visual interfaces attempt to augment the traditional human-computer interactive mechanisms with visual aides [25].

Human and computer vision refer to the ability of seeing an image and understanding its details—generally known as image contents. This ability is often summarized as: to observe and to evaluate. Visual information is what humans can extract and understand from images and video. Making inferences is much easier in visual systems. For example, given two objects—one large and one small—a visual system can immediately recognize the disparity of size. Such an inference would be much more difficult to make using a text annotations.

Spatial and motion characteristics of objects derived from images and video sequences are inherently visual. This visual aspect behooves us to find best suited visual paradigms for content retrieval of images and video sequences.

In a previous paper, we have presented a method for extracting object motion during video encoding by the MPEG method. Since a large part of the low level motion analysis is already performed by the video encoder for the encoding of the blocks in the predicted and bidirectional frames, our algorithms perform very well in creating object trajectories based on these coarse-grained representations of the optical flow. Simply put, we extract macroblock trajectories which are spatiotemporal representations of macroblock motion and use them for object motion recovery. By describing the movements that we derive from the process of motion analysis, we introduce a dual hierarchy consisting of spatial and temporal parts for video sequence representation. This gives us the flexibility to examine arbitrary frames at various levels of abstraction, and to retrieve the associated temporal information (say, object trajectories) in addition to the spatial representation. See [12] for details.

Motion in a video segment may be a result of a number of different phenomena. Our work does not cover all types. For example, camera motion cannot be separated from object motion. Basic camera movements include: vertical rotation (called tilting), vertical transverse movement (or booming), horizontal rotation (known as panning), horizontal transverse movement (called tracking), variations in focus distance (or zooming), and horizontal lateral movement (commonly known as dollying). These six along with no movement, i.e., fixed, constitute the seven basic camera operations. Obviously, camera operations are the cause of significant characteristics of the video data and should be modeled properly. Optical flow techniques for motion extraction work much better for detecting this kind of change. See Akutsu et al. [1] for more detail.

There are certain other types of motion that present a problem for our schemes. Our algorithms are primarily geared to recognizing motion in 2D, i.e., the (x, y) direction. When an object is moving directly toward the camera, similar to the zooming operation of the camera, we cannot distinguish and accurately formulate the motion. Again, to a certain degree, schemes based on optical flow analysis perform better in this regard.

In this paper, we outline the design of a multimedia database language which has well defined semantics in both the icon-based and character-based paradigms. The reason for supporting both paradigms is to have the best of both worlds: intuitive visual description of visual languages and the scalability of the character based languages. This paper first surveys the visual metaphors for content based retrieval for digital video in Section 2. The video information model are then presented in Section 3. In Section 4, we present the formal foundation of the language Visual Extension to VArqa (VEVA) which embodies the representation of the visual model into a common iconic and character-based language. The grammar of a language based on VEVA is given in Section 4.1. The execution model is described in Section 4.2. Section 4.3 illustrates the usage of the VEVA language through example queries and results. Some final observations are presented in the conclusions section.

2. Visual metaphors for content retrieval

The basic concept of visual interaction is the *visual metaphor*. The quality, in terms of understandability, clarity and effectiveness (i.e., having intuitive connotations with the concept they represent), is very important for raising the level of communication and increasing the potential number of users. A visual metaphor may be characterized as some symbol that would remind the user of an object or concept. Such a symbol may be a part of the object or an exaggeration of one of its significant attributes. In the case of an iconic language, the visual metaphor is the icon. An icon is usually a simplified image of the function of the system or an entity, and is used for manipulation of the system. The user recognizes the objectives from the icon images and their spatial arrangements. In a flexible setting, the shape and other characteristics of the icons are tailored by the user to suit her/his own mental representation of the tasks to be performed. The specification of icons is a whole new field of research.

Visual languages are based on the direct manipulation of graphical objects. Several languages cater to graphical description of a wide variety of applications, and allow the specification of complex computing environments. We will discuss a few examples. Media Streams is a visual language that enables users to create multilayered, iconic annotations of video content [10]. The objects denoted by icons are organized into hierarchies. The icons are used to annotate video streams in what the author calls a “Media Time Line”. While appealing with the simplicity and the abundance of iconic expressions, Media Streams uses a fixed vocabulary for the video annotation process.

A system proposed by Little et al. supports content based retrieval and playback [20]. A specific schema is composed of movie, scene and actor relations with a fixed set of attributes. The system requires manual feature extraction. The features are then inserted into the schema. Querying may involve reference to the attributes of movie, scene or actor. Once a movie is selected, the user can browse from scene to scene beginning with the initial selection. Querying in this system is achieved by browsing the predetermined attributes of the movies. Browsing as a visual interaction metaphor in digital video is useful for applications which have a predefined set of attributes and where users are not expected to learn a new interaction language.

An algebraic approach to content-based access to video is presented in [30]. Video presentations are composed of video segments using a *Video Algebraic Model*. The algebra contains methods for combining video segments both temporally and spatially, as well as methods for navigation and querying. This model leads to a framework for efficient access and management of video collections. However, the search process is based on the attribute information of the algebraic video nodes which are textually represented by human readable, semi-structured, algebraic video files. This approach ties together the attribute-based access and content browsing of the video nodes.

One premise of our work is that visual languages and browsing are extremely important for interactive capabilities in digital video applications. The language presented here, VEVA, has much in common with the above visual languages. It is an iconic language which has a formal mathematical model. Video sequences can be queried based on the motion as well as spatial aspects of the objects. The iconic representations of all the concepts can be queried based on their temporal appearance in a way similar to that in Media Streams. The language closely follows a model for motion recovery and representation of objects in digital video [12]. The advantage of VEVA is that it is based on the same model which is used for extraction of semantics of video sequences. As part of the language, a suite of operators can be used for automatic feature extraction and matching of objects.

3. Video information model and language

3.1. The model

In this section we present a formal foundation for our video information model. It is based on an algebraic framework [13, 31] which has been used in many other areas including programming languages, software and hardware specifications, and object oriented modeling. Algebraic specifications are developed in the following manner. Given an alphabet consisting of several classes of symbols for *types* and their associated *operators* (functions), a schema is specified. (Formally, the schema corresponds closely to the notion of signature in the algebraic framework.) The schema has all the necessary syntactic information along with typing rules, i.e., the rules that determine what type of object(s) can be given to each operator and what type of object(s) it will return. In our system, the domain-dependent information, provided by the system developer, is combined with the application-independent constructs, provided by the system itself, in order to create the schema. In essence, the schema is a formal specification of all objects of interest, real or conceptual, and the relationships between them. Given a schema, the set of well-formed expressions is defined. These expressions are constructed by using the numerous powerful operators that are provided for each type. We will see that, despite a formalized underpinning, the language presented here is extremely simple. Strongly resembling conventional set theory, our language has a functional flavor, similar to Lucid [29] and others. In Section 3.2 we will expand the treatment of video data type. This is based on our work on specification of object recognition in images and motion recovery in video [11].

The basic constructs of the model are *data types* and *functions* that operate on the data types. Two main kinds of data types that are used in this discussion are:

- System-defined, fixed data types, called "deliverable types", are: string, integer, boolean, text, image, audio and video. These are the application-independent constituents of the system and are present in any specification. The traditional data types integer, string and boolean are known as printable objects. Analogously, we call audio and video deliverable (presentable) types, since they inherently contain a time component.
- User-defined data types, called "entity types", such as "PERSON", "STUDENT", are those that represent objects or concepts in the real world. These, generally, have properties that are embodied in deliverable types.

Each data type has a number of operators associated with it. Operators that are associated with user-defined types are called *user-defined functions*. User-defined functions describe the domain related relationships, cross references between entity types, and the attributes of objects. Cross references typically represent multivalued relationships. Function types have the general form of:

$$\phi : \alpha_0 \times \cdots \times \alpha_{n-1} \to \alpha_n$$

where every α_i, called a type expression, is inductively defined as:

— a data type
— $\alpha_1 \cup \alpha_2$, $\alpha_1 \times \alpha_2$ or $\mathcal{P}(\alpha_1)$ where α_1 and α_2 are type expressions.

Note that the above definition will allow the functions to take as arguments: object types, their unions, cartesian products and powersets.

In addition to the user-defined functions, we need a collection of operators that are independent of the application domain, and as such operate on system defined data types. Each data type, such as text, graphics, scanned images, audio, and video, has a rich selection of operators associated with it. For example, the operator *appendPar* performs concatenation operation for text paragaraphs for the type *Text*. All set theoretical, boolean, and arithmetic operators are included. In addition, there are a number of variable binding operators. The main characteristic of these operators is that they cause a variable to range over the elements of its domain. An example is the set construction operator that has the form $\{f(x) \mid P(x)\}$, where $f(x)$ denotes the desired output objects, and $P(x)$ denotes the retrieval predicate that must hold for those objects. While x ranges over its domain, whenever $P(x)$ is satisfied, $f(x)$ is added to the set. There are a number of other operators like this, including the logical quantifiers. We will see some examples when we introduce video operators.

A list of basic multimedia operators and their semantics are presented in [15]. These operators are categorized into: set operators, logical operators, operators for temporal synchronization and spatial composition, arithmetic operations, and media specific operators for text, graphics, audio, image, and video manipulation. The basic data types and their associated operators are part of the schema (signature) of the multimedia information system. The syntax of the language is developed on the basis of this schema.

In the algebraic framework, the algebra associates with each type a set of objects that behave as mandated by the specification of that type. Thus, the set associated with the type *Integer* is, obviously, the set of integer numbers. Other data types denote objects that have the predefined properties. Objects of type *Text* are paragraphs. Data type *Audio* denotes signals of one dimension, and can be thought of as a function of time. The type *Image* contains signals of two dimensions. Finally, *Video* is a signal of three dimensions $F(x, y, i)$ represented as $F_i(x, y)$ where i represents the frame counter, and x and y are pixel coordinates. When no confusion is expected, we may omit the references to the coordinates x and y or to the frame counter i. When needed we will use superscripts to distinguish between different video streams, e.g., $F_i^1(x, y)$ and $F_i^2(x, y)$. The following notation will be used:

— $F_b(x, y)$ for the first frame of the video signal
— $F_e(x, y)$ for the last frame of the video signal
— $F_c(x, y)$ for the current frame of the video signal

The algebra allows partial functions. Whenever the function is not defined over a certain part of the domain we use the symbol θ to denote the undefined values.

In this paper we focus specifically on the *Video* data type. In the next sections we describe operators for the video data type in detail.

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | sales@docketalarm.com | 1-866-77-FASTCASE

fastcase® Smarter legal research.