1

INTRODUCTION TO VIDEO DATABASES

Oge Marques and Borko Furht
Department of Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL, USA
omarques@acm.org, borko@cse.fau.edu

1. INTRODUCTION

The field of distributed multimedia systems has experienced an extraordinary growth during the last decade. Among the many visible aspects of the increasing interest in this area is the creation of huge digital libraries accessible to users worldwide. These large and complex multimedia databases must store all types of multimedia data, e.g., text, images, animations, graphs, drawings, audio, and video clips. Video information plays a central role in such systems, and consequently, the design and implementation of video database systems has become a major topic of interest.

The amount of video information stored in archives worldwide is huge. Conservative estimates state that there are more than 6 million hours of video already stored and this number grows at a rate of about 10 percent a year [1]. Projections estimate that by the end of 2010, 50 percent of the total digital data stored worldwide will be video and rich media [5]. Significant efforts have been spent in recent years to make the process of video archiving and retrieval faster, safer, more reliable and accessible to users anywhere in the world. Progress in video digitization and compression, together with advances in storage media, have made the task of storing and retrieving raw video data much easier. Evolution of computer networks and the growth and popularity of the Internet have made it possible to access these data from remote locations.

However, raw video data by itself has limited usefulness, since it takes far too long to search for the desired piece of information within a videotape repository or a digital video archive. Attempts to improve the efficiency of the search process by adding extra data (henceforth called *metadata*) to the video contents do little more than transferring the burden of performing inefficient, tedious, and time-consuming tasks to the cataloguing stage. The challenging goal is to devise better ways to automatically store, catalog, and retrieve video information with greater understanding of its contents. Researchers from various disciplines have acknowledged such challenge and provided a vast number of algorithms, systems, and papers on this topic during recent years. In addition to these localized efforts, standardization groups have been working on new standards, such as MPEG-7, which provide a framework for multimedia content description.

The combination of the growing number of applications for video-intensive products and solutions – from personal video recorders to multimedia collaborative systems – with the many technical challenges behind the design of contemporary video database systems yet to be overcome makes the topics discussed in this Handbook of extreme interest to researchers and practitioners in the fields of image and video processing, computer vision, multimedia systems, database systems, information retrieval, data mining, machine learning, and visualization, to name just a few (Figure 1.1).

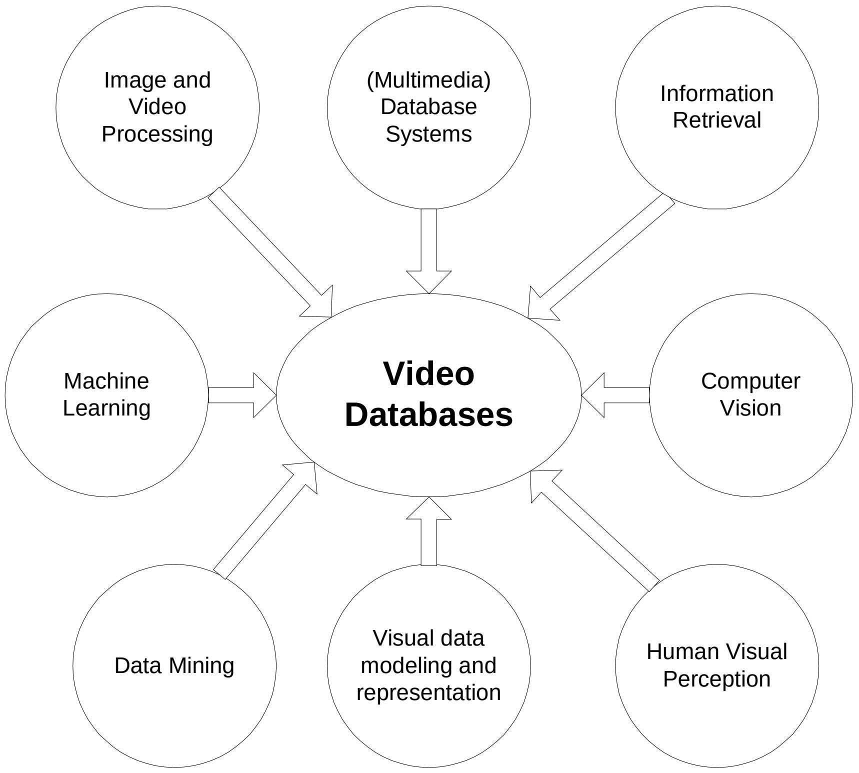

Figure 1.1 Visual Information Retrieval blends together many research disciplines.

In this chapter we present a general overview of the central topic in this Handbook: video databases. Its main goal is to introduce basic concepts behind general-purpose database systems and their extension to multimedia database systems, particularly image and video database systems. Section 2 introduces basic database concepts and terminology. In Section 3 we briefly outline the main steps of the database design process. Section 4 extends the discussion from general databases to multimedia databases and their own particular requirements and characteristics. Section 5 narrows down the field even further, focusing on the aspects that are specific to image and video databases. The main goals of this Handbook are outlined in Section 6. Finally, Section 7 provides the reader with an overview of the other chapters of this Handbook and how they have been organized.

2. BASIC DATABASE CONCEPTS

A *database* is a logically coherent collection of related data, where *data* refers to known facts that can be recorded and that have implicit meaning [2]. A *database management system* (DBMS) is a collection of programs that enables users to create and maintain a database. Together, the database and DBMS software are called a *database system*. Contemporary DBMS packages have a modular design and adopt a client-server architecture. A *client* module, running in the user's workstation, handles user interaction and provides friendly GUIs (graphical user interfaces). The *server* module is responsible for data storage, access, search, and other operations.

Database systems should be designed and built in a way as to hide details of data storage from its end users. Such abstraction can be achieved using a *data model*, normally defined as a collection of concepts that can be used to describe the structure of a database, i.e., its data types, relationships, and constraints imposed on data. Most data models also include a set of basic operations for specifying retrievals and updates on the database. An *implementation* of a given data model is a physical realization on a real machine of the components of the abstract machine that together constitute that model [3].

Data models can be categorized in three main groups:

(1) **High-level or conceptual data models**: use concepts such as entities, attributes, and relationships, which are close to the way many users perceive data.

(2) **Low-level or physical data models**: describe details on how data is stored in a computer, which make them meaningful to computer specialists, not to end users.

(3) **Representational (or implementation) data models**: widely used intermediate category, which aims at concepts that may be understood by end users but that are not far from the way data is organized in a computer. The most widely used representational data models are the *relational data model* and the *object data model.*

The description of a database is called the *database schema*, typically represented in a diagram (usually referred to as *schema diagram*) where each object is called a *schema construct.* The database schema is specified during database design and is expected to remain unchanged unless the requirements of the database applications change.

A typical architecture for a database system, proposed by the ANSI/SPARC Study Group on Database Management Systems and known as the ANSI/SPARC architecture, is shown in Figure 1.2. This architecture is divided into three levels, each of which has its own schema:

(1) The **internal level** (also known as the *physical level*) has an internal schema, which describes the physical storage structure of the database.

(2) The **conceptual level** (also known as the *community logical level*) has a conceptual schema, which describes the structure of the whole database for a community of users.

(3) The **external level** (also known as the *user logical level* or *view level*) has a number of external schemas or user views. Each external schema describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group.

The three-schema architecture involves mappings – represented by the double-pointed arrows – that can be updated any time a partial change in some of the database's schemas take place. The ability of changing the schema at one level of a database system without having to change the schema at the next higher level is known as *data independence*.

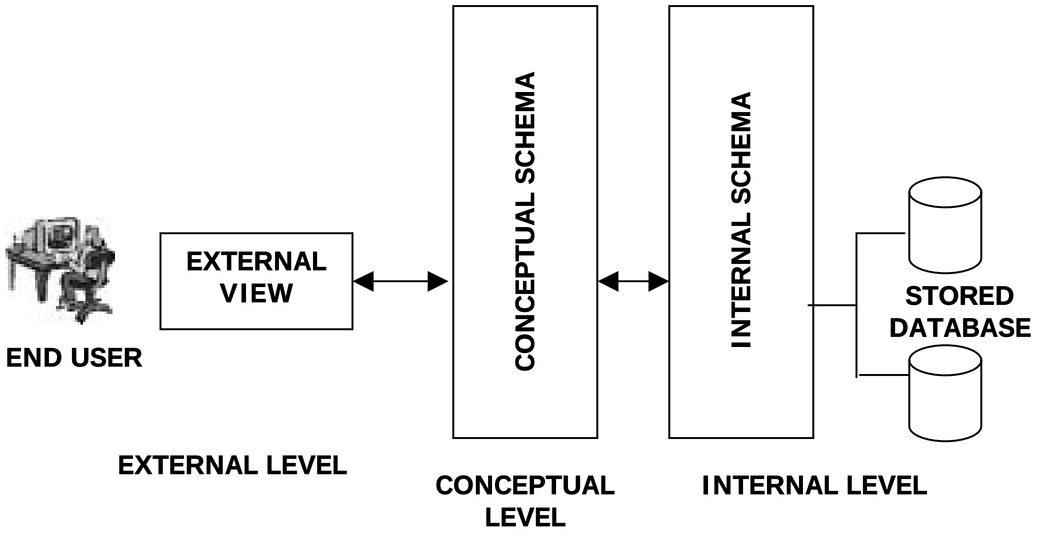

Figure 1.2 The three levels of the ANSI/SPARC architecture.

A DBMS must provide appropriate languages and interfaces for each category of users. The type of language that allows an end user to manipulate, retrieve, insert, delete, and modify data in the database is known as *data manipulation language* (DML). There are two main types of DMLs: high-level DMLs (such as Structured Query Language – SQL) can work on a set of records, while low-level DMLs retrieve and process one record at a time. A high-level DML used in a stand-alone interactive manner is also called a *query language*.

3. THE DATABASE DESIGN PROCESS

The problem of database design can be summarized in a question: Given some body of data to be represented in one or more databases, how do we decide on a suitable logical structure for that data such that the information needs of the users are properly accommodated?

Different authors suggest slightly different procedures for database design. In essence, all procedures contain the following stages:

(1) **Requirements collection and analysis**: the process of collecting and analyzing information about the part of the organization that is to be supported by the database application, and using this information to identify the users' requirements of the new system [4]. Requirement specification techniques include OOA (object-oriented analysis) and DFDs (data flow diagrams),

(2) **Conceptual database design**: this phase involves two parallel activities: (a) *conceptual schema design*, which produces a conceptual database schema based on the requirements outlined in phase 1; and (b)

transaction and application design, which produces high-level specifications for the applications analyzed in phase 1. Complex databases are normally designed using a top-down approach and use the terminology of the Entity-Relationship (ER) model or one of its variations.

(3) **Logical design**: at this stage the internal schemas produced in phase 2(a) are mapped into conceptual and external schemas. Ideally the resulting model should be independent of a particular DBMS or any physical consideration.

(4) **DBMS selection**: the selection of a commercially available DBMS to support the database application is governed by technical, economic, and sometimes even political factors. Some of the most relevant technical factors include: the type of data model used (e.g., relational or object), the supported storage structures and access paths, the types of high-level query languages, availability of development tools, and utilities, among many others.

(5) **Physical design**: the process of choosing specific storage structures and access methods used to achieve efficient access to data. Typical activities included in this phase are: choice of file organization (heap, hash, Indexed Sequential Access Method – ISAM, B^+-tree, and so on), choice of indexes and indexing strategies, and estimation of disk requirements (e.g., access time, total capacity, buffering strategies).

(6) **Implementation and testing**: the designed database is finally put to work and many unanticipated problems are fixed and the overall performance of the database system is fine-tuned.

4. MULTIMEDIA DATABASES

The design and implementation of multimedia databases create additional challenges due to the nature of multimedia data and the requirements of possible applications. Multimedia applications can be categorized in three main groups, each of which poses different data management challenges [2]:

(a) **Repository applications**: Large amounts of multimedia data and associated metadata are stored for retrieval purposes. These repositories may be distributed or centralized and can be managed using conventional DBMS. Examples of multimedia data stored in such repositories include medical and satellite images, and engineering drawings.

(b) **Presentation applications**: Applications that involve delivery of multimedia content to a possibly remote location, subject to temporal constraints. In these applications, data is consumed as it is delivered, as opposed to being stored for later processing. A number of new potential problems, such as jitter, latency, and the corresponding need to maintain and guarantee "quality of service" (QoS), come into play. Examples of such applications include audio and video broadcasting over the Internet.

(c) **Collaborative work using multimedia information**: a new breed of multimedia applications in which a geographically dispersed group of professionals (e.g., engineers or medical doctors) work together on a common, multimedia-intensive, task.

Accommodating massive amounts of text, graphics, images, animations, audio, and video streams into a database system is far from trivial and the popularization of multimedia databases has raised a number of complex issues for database designers. Some of these issues are [2]:

- *Modeling*: Multimedia information includes media objects, associated metadata, and the objects' temporal and spatial characteristics. This information is continuously manipulated and modified by applications. Some of the different techniques used for modeling multimedia data are [6]:
  - (a) **object-oriented modeling**: inspired by the object-oriented paradigm, it organizes multimedia information into hierarchical structures in which each multimedia object (e.g., text, audio, video, image) has its set of variables, methods, and messages to which it responds. Such a model can enforce concepts such as encapsulation, data hiding, and multiple inheritance, and can handle the metadata as well. Some of its drawbacks include the difficulty of accessing objects in a collective – rather than individual – manner, the need to handle the database schema independently from the class hierarchy, and the impossibility of creating new objects that are based on portions of existing objects and need only to inherit part of their attributes.
  - (b) **temporal models**: multimedia objects have associated temporal characteristics, which are particularly important in presentation-type applications. These characteristics specify parameters such as: time instant of an object presentation, duration of presentation, and synchronization among objects in the presentation. Temporal models are called *hard* when the temporal relationships are specified in a precise manner with exact values for time instants and duration of presentations, or *flexible* if they allow a range of values to be specified for each time-related parameter.
  - (c) **spatial models**: multimedia applications are constrained by the size of each window and the window layout. These constraints must be taken into account either in a hard way (assigning specific values for the x and y coordinates of each window corner), or in a flexible way, using difference constraints and specifying relative positions among the various windows in a presentation.
- *Design*: The conceptual, logical, and physical design of multimedia databases remains an area of active research. The general design methodology summarized in Section 3 can still be used as a starting point, but performance and fine-tuning issues are more complex than in conventional databases.
- *Storage*: Storage of multimedia information in conventional magnetic media brings new problems, such as representation, compression/ decompression, mapping to device hierarchies, archiving, and buffering during the I/O operations.
- *Queries and retrieval*: Efficient query formulation, query execution, and optimization for multimedia data is still an open problem and neither query languages nor keyword-based queries have proven to be completely satisfactory.
- *Performance*: While some multimedia applications can tolerate less strict performance constraints (e.g., the maximum time to perform a content-

based query on a remote image repository), others are inherently more critical, such as the minimum acceptable frame rate for video playback.

Recent developments in Multimedia Database Systems are expected to bring together two disciplines that have historically been separate: *database management* and *information retrieval*. The former assumes a rigid structure for data and derives the meaning of a data instance from the database schema, while the latter is more concerned with modeling the data content, without paying much attention to its structure [2].

There are very few commercial multimedia database management solutions currently available, e.g., MediaWay's Chuckwalla Broadband Media Management System. Nonetheless, many well-known DBMSs support multimedia data types; examples include Oracle 8.0, Sybase, Informix, ODB II, and CA-JASMINE. The way multimedia extensions are handled by each of these systems is ad hoc, and does not take into account interoperability with other products and solutions. It is expected that the MPEG-7 standard will promote the necessary standardization.

5. IMAGE AND VIDEO DATABASES

Image and video databases have particular requirements and characteristics, the most important of which will be outlined in this Section and described in much more detail along other chapters in this Handbook. Some of the technical challenges are common to both types of media, such as the realization that raw contents alone are not useful, unless they are indexed for further query and retrieval, which makes the effective indexing of images and video clips an ongoing research topic.

After having been catalogued, it should be possible to query and retrieve images and video clips based on their semantic meaning, using a measure of similarity between the query terms (textual or otherwise) and the database contents. Since in most cases there is no exact matching between the query and its expected result, there is a need to extend the information retrieval approaches to search based on similarity to the case where the mapping between visual and semantic similarity is not straightforward. These issues will be explored within the realm of image databases in Section 5.1. An extension to video and its specific challenges and complexities will be presented in Section 5.2.

5.1 IMAGE DATABASES

The design of image databases brings about the need for new abstractions and techniques to implement these abstractions.

Since raw images alone have very limited usefulness, some technique must be devised to extract and encode the image properties into an alphanumerical format that is amenable to indexing, similarity calculations, and ranking of best results. These properties can be extracted using state-of-the-art computer vision algorithms. Examples of such properties include: shape, color, and texture. These descriptors are recognizably limited and very often fail to capture the semantic meaning of the image, giving rise to the well-known *semantic gap* problem. Moreover, some of these descriptors are only useful if applied to the relevant objects within an image, which calls for some sort of segmentation to occur. Automatic segmentation of a scene into its relevant objects is still an unresolved problem, and many attempts in this field get around this problem by including the user in the loop and performing a semi-automatic segmentation instead.

Storing a set of images so as to support image retrieval operations is usually done with spatial data structures, such as R-trees and a number of variants (e.g., R^+ trees, R* trees, SS-trees, TV-trees, X-trees, M-trees) proposed in the literature.

Indexing of images is another open research problem. Raw images must have their contents extracted and described, either manually or automatically. Automatic techniques usually rely on image processing algorithms, such as shape-, color-, and texture-based descriptors. Current techniques for content-based indexing only allow the indexing of simple patterns and images, which hardly match the semantic notion of relevant objects in a scene. The alternative to content-based indexing is to assign index terms and phrases through manual or semi-automatic indexing using textual information (usually referred to as *metadata*), which will then be used whenever the user searches for an image using a query by keyword.

The information-retrieval approach to image indexing is based on one of the three indexing schemes [2]:

1. *Classificatory systems*: images are classified hierarchically according to the category to which they belong.
2. *Keyword-based systems*: images are indexed based on the textual information associated with them.
3. *Entity-attribute-relationship systems*: all objects in the picture and the relationships between objects and the attributes of the objects are identified.

Querying an image database is fundamentally different from querying textual databases. While a query by keyword usually suffices in the case of textual databases, image databases users normally prefer to search for images based on their visual contents, typically under the *query by example* paradigm, through which the user provides an example image to the system and (sometimes implicitly) asks the question: "Can you find and retrieve other pictures that look like this (and the associated database information)?" Satisfying such a query is a much more complex task than its text-based counterpart for a number of reasons, two of which are: the inclusion of a picture as part of a query and the notion of "imprecise match" and how it will be translated into criteria and rules for similarity measurements. The most critical requirement is that a similarity measure must behave well and mimic the human notion of similarity for any pair of images, no matter how different they are, in contrast with what would typically be required from a matching technique, which only has to behave well when there is relatively little difference between a database image and the query [7].

There are two main approaches to similarity-based retrieval of images [8]:

1. *Metric approach*: assumes the existence of a distance metric d that can be used to compare any two image objects. The smaller the distance between the objects, the more similar they are considered to be. Examples of widely used distance metrics include: Euclidean, Manhattan, and Mahalanobis distance.

2. *Transformation approach*: questions the claim that a given body of data (in this case, an image) has a single associated notion of similarity. In this model, there is a set of operators (e.g., translation, rotation, scaling) and cost functions that can be optionally associated with each operator. Since it allows users to personalize the notion of similarity to their needs, it is more flexible than the metric approach; however, it is less computationally efficient and its extension to other similar queries is not straightforward.

There are many (content-based) image retrieval systems currently available, in the form of commercial products and research prototypes. For the interested reader, Veltkamp and Tanase [9] provide a comprehensive review of existing systems, their technical aspects, and intended applications.

5.2 VIDEO DATABASES

The challenges faced by researchers when implementing image databases increase even further when one moves from images to image sequences, or video clips, mostly because of the following factors:

- increased size and complexity of the raw media (video, audio, and text);
- wide variety of video programs, each with its own rules and formats;
- video understanding is very much context-dependent;
- the need to accommodate different users, with varying needs, running different applications on heterogeneous platforms.

As with its image counterpart, raw video data alone has limited usefulness and requires some type of annotation before it is catalogued for later retrieval. Manual annotation of video contents is a tedious, time consuming, subjective, inaccurate, incomplete, and – perhaps more importantly – costly process. Over the past decade, a growing number of researchers have been attempting to fulfill the need for creative algorithms and systems that allow (semi-) automatic ways to describe, organize, and manage video data with greater understanding of its semantic contents.

The primary goal of a Video Database Management System (VDBMS) is to provide pseudo-random access to sequential video data. This goal is normally achieved by dividing a video clip into segments, indexing these segments, and representing the indexes in a way that allows easy browsing and retrieval. Therefore, it can be said that a VDBMS is basically a database of indexes (pointers) to a video recording [10].

Similarly to image databases, much of the research effort in this field has been focused on modeling, indexing, and structuring of raw video data, as well as finding suitable similarity-based retrieval measures. Another extremely important aspect of a VDBMS is the design of its graphical user interface (GUI). These aspects will be explored in a bit more detail below.

5.2.1 The main components of a VDBMS

Figure 1.3 presents a simplified block diagram of a typical VDBMS. Its main blocks are:

- *Digitization and compression*: hardware and software necessary to convert the video information into digital compressed format.

- *Cataloguing*: process of extracting meaningful story units from the raw video data and building the corresponding indexes.
- *Query / search engine*: responsible for searching the database according to the parameters provided by the user.
- *Digital video archive*: repository of digitized, compressed video data.
- *Visual summaries*: representation of video contents in a concise, typically hierarchical, way.
- *Indexes*: pointers to video segments or story units.
- *User interface*: friendly, visually rich interface that allows the user to interactively query the database, browse the results, and view the selected video clips.

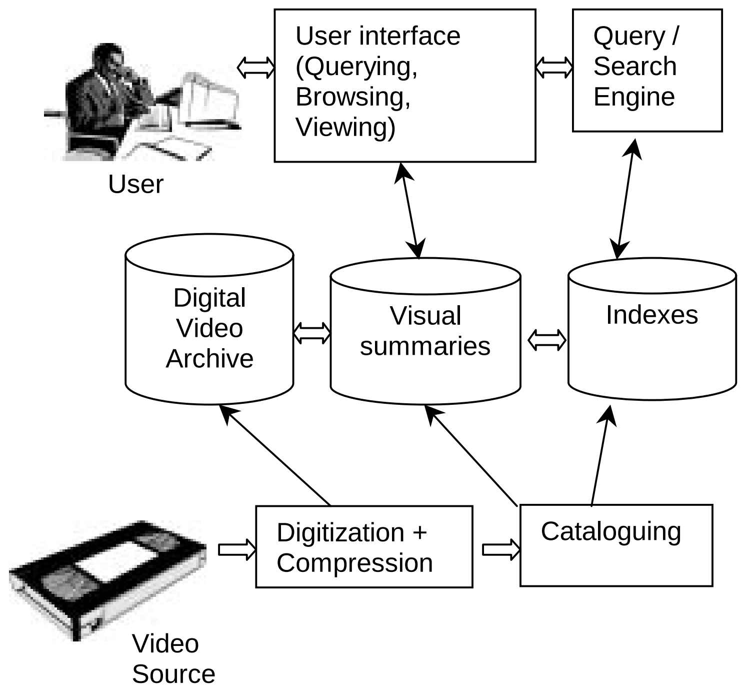

Figure 1.3 Block diagram of a VDBMS.

5.2.2 Organization of video content

Because video is a structured medium in which actions and events in time and space convey stories, a video program must not be viewed as a non-structured sequence of frames, but instead it must be seen as a document. The process of converting raw video into structured units, which can be used to build a visual table of contents (ToC) of a video program, is also referred to as *video abstraction*. We will divide it into two parts:

1. Video modeling and representation
2. Video segmentation (parsing) and summarization

Video modeling and representation

Video modeling can be defined as the process of designing the representation for the video data based on its characteristics, the information content, and the applications it is intended for. Video modeling plays a key role in the design of VDBMSs, because all other functions are more or less dependent on it.

The process of modeling video contents can be a challenging task, because of the following factors:

- video data carry much more information than textual data;
- interpretation is usually ambiguous and depends on the viewer and the application;
- the high dimensionality of video data objects;
- lack of a clear underlying structure;
- massive volume (bytes);
- relationships between video data segments are complex and ill-defined.

When referring to contents of video data, the following distinctions should be made, according to their type and level [11]:

- *Semantic content*: the idea or knowledge that it conveys to the user, which is usually ambiguous, subjective, and context-dependent.
- *Audiovisual content*: low-level information that can be extracted from the raw video program, usually consisting of color, texture, shape, object motion, object relationships, camera operation, audio track, etc.
- *Textual content*: additional information that may be available within the video stream in the form of captions, subtitles, etc.

Some of the requirements for a video data model are [11]:

- Support video data as one of its data types, just like textual or numeric data.
- Integrate content attributes of the video program with its semantic structure.
- Associate audio with visual information.
- Express structural and temporal relationships between segments.
- Automatically extract low-level features (color, texture, shape, motion), and use them as attributes.

Most of the video modeling techniques discussed in the literature adopt a hierarchical video stream abstraction, with the following levels, in decreasing degree of granularity:

- *Key-frame*: most representative frame of a shot.
- *Shot*: sequence of frames recorded contiguously and representing a continuous action in time or space.
- *Group*: intermediate entity between the physical shots and semantic scenes that serves as a bridge between the two.
- *Scene* or *Sequence*: collection of semantically related and temporally adjacent shots, depicting and conveying a high-level concept or story.
- *Video program*: the complete video clip.

A video model should identify physical objects and their relationships in time and space. Temporal relationships should be expressed by: before, during, starts, overlaps, etc., while spatial relationships are based on projecting objects on a 2-D or 3-D coordinate system.

A video model should also support annotation of the video program, in other words the addition of metadata to a video clip. For the sake of this discussion, we consider three categories of metadata:

- *content-dependent metadata* (e.g., facial features of a news anchorperson);
- *content-descriptive metadata* (e.g., the impression of anger or happiness based on facial expression);
- *content-independent metadata* (e.g., name of the cameraman).

Video data models can usually be classified into the following categories [11] (Figure 1.4):

- *Models based on video segmentation*: adopt a two-step approach, first segmenting the video stream into a set of temporally ordered basic units (shots), and then building domain-dependent models (either hierarchy or finite automata) upon the basic units.
- *Models based on annotation layering* (also known as *stratification models*): segment contextual information of the video and approximate the movie editor's perspective on a movie, based on the assumption that if the annotation is performed at the finest grain (by a *data camera*), any coarser grain of information may be reconstructed easily.
- *Video object models*: extend object-oriented data models to video. Their main advantages include the ability to represent and manage complex objects, handle object identities, encapsulate data and associated methods into objects, and inherit attribute structures and methods based on class hierarchy.

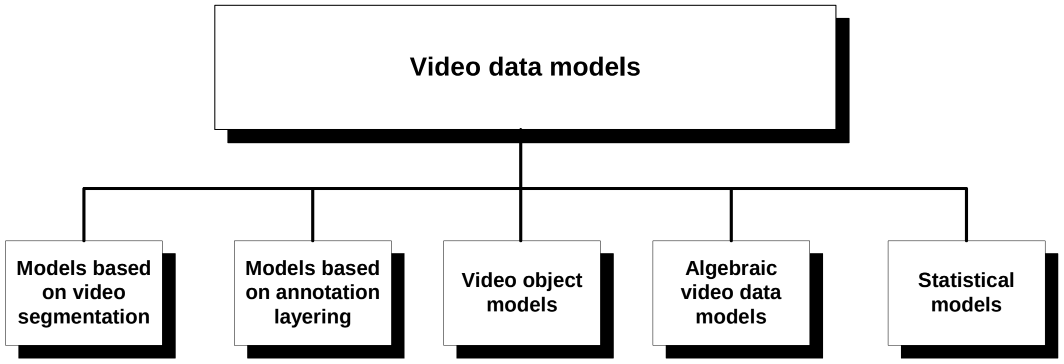
Video data models
Models based on video segmentation
Models based on annotation layering
Video object models
Algebraic video data models
Statistical models

Figure 1.4 Classification of video data models.

- *Algebraic video data models*: define a video stream by recursively applying a set of algebraic operations on the raw video segment. Their fundamental entity is a *presentation* (multi-window, spatial, temporal, and content combination of video segments). Presentations are described by video expressions, constructed from raw segments using video algebraic operations. As an example of the algebraic approach to video manipulation, the reader is referred to Chapter 19 of this Handbook, where Picariello, Sapino, and Subrahmanian introduce AVE! (Algebraic Video Environment), the first algebra for querying video.

- *Statistical models*: exploit knowledge of video structure as a means to enable the principled design of computational models for video semantics, and use machine learning techniques (e.g., Bayesian inference) to learn the semantics from collections of training examples, without having to rely on lower level attributes such as texture, color, or optical flow. Chapter 3 of this Handbook contains an example of statistical modeling of video programs, the Bayesian Modeling of Video Editing and Structure (BMoViES) system for video characterization, developed by Nuno Vasconcelos.

Video segmentation

Video segmentation (also referred to as *video parsing*) is the process of partitioning video sequences into smaller units. Video parsing techniques extract structural information from the video program by detecting temporal boundaries and identifying meaningful segments, usually called *shots*. The shot ("a continuous action on screen resulting from what appears to be a single run of the camera") is usually the smallest object of interest. Shots are detected automatically and typically represented by key-frames.

Video segmentation can occur either at a shot level or at a scene level. The former is more often used and sometimes referred to as shot detection. Shot detection can be defined as the process of detecting transitions between two consecutive shots, so that a sequence of frames belonging to a shot will be grouped together. There are two types of shot transitions: abrupt transitions (or *cuts*) and gradual transitions (e.g., fade-in, fade-out, dissolve). Earlier work on shot detection addressed only the detection of cuts, while more recent research results report successful techniques for detection of gradual transitions as well.

An alternative to shot detection, scene-based video segmentation consists of the automatic detection of semantic boundaries (as opposed to physical boundaries) within a video program. It is a much more challenging task, whose solution requires a higher level of content analysis, and the subject of ongoing research. Three main strategies have been attempted to solve the problem:

- Segmentation based on film production rules (e.g., transition effects, shot repetition, appearance of music in the soundtrack) to detect local (temporal) clues of macroscopic change;
- Time-constrained clustering, which works under the rationale that semantically related contents tend to be localized in time;
- *A priori* model-based algorithms, which rely on specific structural models for programs whose temporal structures are usually very rigid and predictable, such as news and sports.

Video segmentation can occur either in the uncompressed or compressed domain. In the uncompressed domain, the basic idea upon which the first algorithms were conceived involved the definition of a similarity measure between successive images and the comparison of successive frames according to that measure: whenever two frames are found to be sufficiently dissimilar, there may be a cut. Gradual transitions are found by using cumulative difference measures and more sophisticated thresholding schemes. Temporal video segmentation techniques that work directly on the compressed video streams were motivated by the computational savings resulting from not having to perform decoding/re-encoding, and the possibility of exploiting pre-computed features, such as motion vectors (MVs) and block averages (DC coefficients), that are suitable for temporal video segmentation.

Temporal video segmentation has been an active area of research for more than 10 years, which has resulted in a great variety of approaches. Early work focused on cut detection, while more recent techniques deal with gradual transition detection. Despite the great evolution, most current algorithms exhibit the following limitations [12]:

- They process unrealistically short gradual transitions and are unable to recognize the different types of gradual transitions;
- They involve many adjustable thresholds;
- They do not handle false positives due to camera operations.

For a comprehensive review of temporal video segmentation, the interested reader is referred to a recent survey by Koprinska and Carrato [12].

Video summarization

Video summarization is the process by which a pictorial summary of an underlying video sequence is presented in a more compact form, eliminating – or greatly reducing – redundancy. Video summarization focuses on finding a smaller set of images (key-frames) to represent the visual content, and presenting these key-frames to the user.

A still-image abstract, also known as a static storyboard, is a collection of salient still images or key-frames generated from the underlying video. Most summarization research involves extracting key-frames and developing a browser-based interface that best represents the original video. The advantages of a still-image representation include:

- still-image abstracts can be created much faster than moving image abstracts, since no manipulation of the audio or text information is necessary;
- the temporal order of the representative frames can be displayed so that users can grasp concepts more quickly;
- extracted still images are available for printing, if desired.

An alternative to still-image representation is the use of video skims, which can be defined as short video clips consisting of a collection of image sequences and the corresponding audio, extracted from the original longer video sequence. Video skims represent a temporal multimedia abstraction that is played rather than viewed statically. They are comprised of the most relevant phrases, sentences, and image sequences and their goal is to present the original video sequence in an order of magnitude less time. There are two basic types of video skimming:

- Summary sequences: used to provide a user with an impression of the video sequence.
- Highlights: contain only the most interesting parts of a video sequence.

Since the selection of highlights from a video sequence is a subjective process, most existing video-skimming work focuses on the generation of summary sequences.

A very important aspect of video summarization is the development of user interfaces that best represent the original video sequence, which usually translates into a trade-off between the different levels and types of abstractions presented to the user: the more condense the abstraction, the easier it is for a potential user to browse through, but maybe the amount of information is not enough to obtain the overall meaning and understanding of the video; a more detailed abstraction may present the user with enough information to comprehend the video sequence, which may take too long to browse.

Emerging research topics within this field include adaptive segmentation and summarization (see Chapters 11 and 12) and summarization for delivery to mobile users (see Chapter 14).

5.2.3 Video indexing, querying, and retrieval

Video indexing is far more difficult and complex than its text-based counterpart. While on traditional DBMS, data are usually selected based on one or more unique attributes (key fields), it is neither clear nor easy to determine what to index a video data on. Therefore, unlike textual data, generating content-based video data indexes automatically is much harder.

The process of building indexes for video programs can be divided into three main steps:

1. *Parsing*: temporal segmentation of the video contents into smaller units.
2. *Abstraction*: extracting or building a representative subset of video data from the original video.
3. *Content analysis*: extracting visual features from representative video frames.

Existing work on video indexing can be classified in three categories [11]:

1. **Annotation-based indexing**

   Annotation is usually a manual process performed by an experienced user, and subject to problems, such as: time, cost, specificity, ambiguity, and bias, among several others. A commonly used technique consists of assigning keyword(s) to video segments (shots). Annotation-based indexing techniques are primarily concerned with the selection of keywords, data structures, and interfaces, to facilitate the user's effort. But even with additional help, keyword-based annotation is inherently poor, because keywords:

   - Do not express spatial and temporal relationships;
   - Cannot fully represent semantic information and do not support inheritance, similarity, or inference between descriptors;
   - Do not describe relations between descriptions.

   Several alternatives to keyword-based annotation have been proposed in the literature, such as the multi-layer, iconic language, Media Streams [14].

2. **Feature-based indexing**

Feature-based indexing techniques have been extensively researched over the past decade. Their goal is to enable fully automated indexing of a video program based on its contents. They usually rely on image processing techniques to extract key visual features (color, texture, object motion, etc.) from the video data and use these features to build indexes. The main open problem with these techniques is the semantic gap between the extracted features and the human interpretation of the visual scene.

3. **Domain-specific indexing**

   Techniques that use logical (high-level) video structure models (*a priori* knowledge) to further process the results of the low-level video feature extraction and analysis stage. Some of the most prominent examples of using this type of indexing technique have been found in the area of summarization of sports events (e.g., soccer), such as the work described in Chapters 5 and 6 of this Handbook.

The video data retrieval process consists of four main steps [11]:

1. User specifies a query using the GUI resources.
2. Query is processed and evaluated.
3. The value or feature obtained is used to match and retrieve the video data stored in the VDB.
4. The resulting video data is displayed on the user's screen for browsing, viewing, and (optionally) query refining (relevance feedback).

Queries to a VDBMS can be classified in a number of ways, according to their content type, matching type, granularity, behavior, and specification [11], as illustrated in Figure 1.5. The semantic information query is the most difficult type of query, because it requires understanding of the semantic content of the video data. The meta information query relies on metadata that has been produced as a result of the annotation process, and therefore, is similar to conventional database queries. The audiovisual query is based on the low-level properties of the video program and can be further subdivided into: spatial, temporal, and spatio-temporal. In the case of deterministic query, the user has a clear idea of what she expects as a result, whereas in the case of browsing query, the user may be vague about his retrieval needs or unfamiliar with the structures and types of information available in the video database.

Video database queries can be specified using extensions of SQL for video data, such as TSQL2, STL (Spatial Temporal Logic), and VideoSQL. However, query by example, query by sketch, and interactive querying/browsing/viewing (with possible relevance feedback) are more often used than SQL-like queries.

Query processing usually involves four steps [11]:

1. Query parsing: where the query condition or assertion is usually decomposed into the basic unit and then evaluated.
2. Query evaluation: uses pre-extracted (low-level) visual features of the video data.
3. Database index search.

4. Returning of results: the video data are retrieved if the assertion or the similarity measurement is satisfied.

As a final comment, it should be noted that the user interface plays a crucial role in the overall usability of a VDBMS. All interfaces must be graphical and should ideally combine querying, browsing summaries of results, viewing (playing back) individual results, and providing relevance feedback and/or query refinement information to the system. Video browsing tools can be classified in two main types:

- *Time-line display of video frames and/or icons*: video units are organized in chronological sequence.
- *Hierarchical or graph-based story board*: representation that attempts to present the structure of video in an abstract and summarized manner.

This is another very active research area. In a recent survey, Lee, Smeaton, and Furner [13] categorized the user-interfaces of various video browsing tools and identified a superset of features and functions provided by those systems.

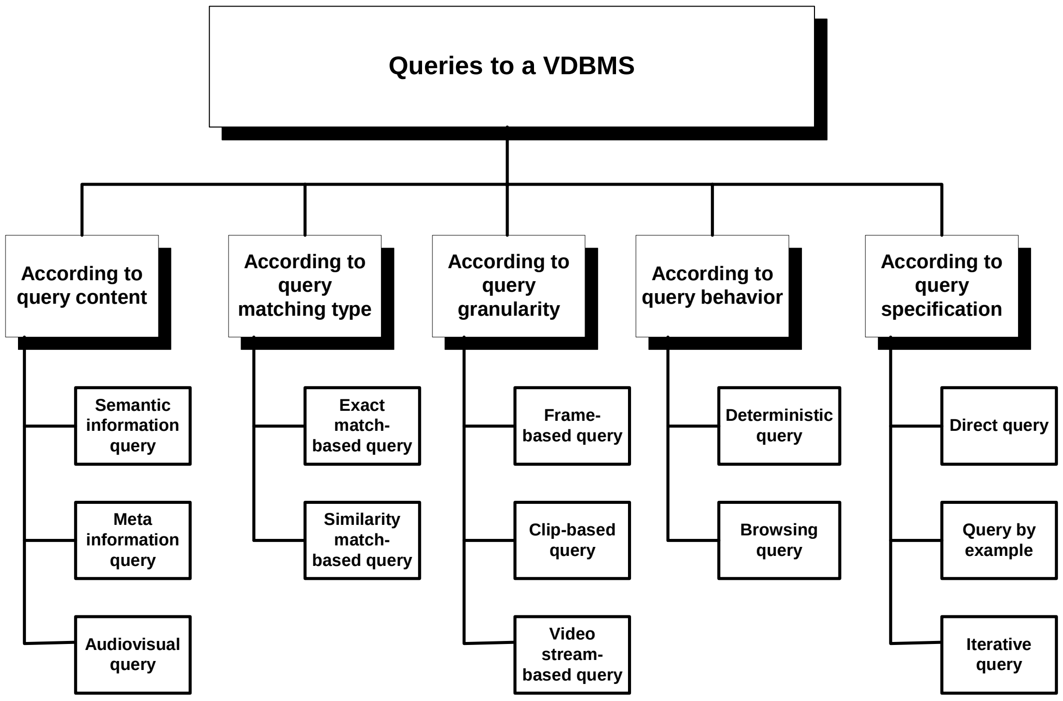

Figure 1.5 Classification of queries to a VDBMS.

6. OBJECTIVES OF THIS HANDBOOK

This Handbook was written to serve the needs of a growing community of researchers and practitioners in the fields of database systems, information retrieval, image and video processing, machine learning, data mining, human-computer interaction, among many others, and provide them with a comprehensive overview of the state of the art in this exciting area of video databases and applications. With contributions from more than 100 recognized world experts in the subject, it is an authoritative reference for most of the relevant research that is being carried out in this field. In addition to chapters that provide in-depth coverage of many topics introduced in this chapter, it showcases many novel applications and provides pointers to hundreds of additional references, becoming a one-stop reference for all the relevant issues related to video databases.

7. ORGANIZATION OF THE HANDBOOK

In the remainder of the Handbook, some of the world leading experts in this field examine the state of the art, ongoing research, and open issues in designing video database systems.

Section II presents concepts and techniques for video modeling and representation. In Chapter 2, Garg, Naphade, and Huang discuss the need for a semantic index associated with a video program and the difficulties in bridging the gap that exists between low-level media features and high-level semantics. They view the problem of semantic video indexing as a multimedia understanding problem, in which there is always a context to the co-occurrence of semantic concepts in a video scene and propose a novel learning architecture and algorithm, describing its application to the specific problem of detecting complex audiovisual events. In Chapter 3, Vasconcelos reviews ongoing efforts for the development of statistical models for characterizing semantically relevant aspects of video and presents a system that relies on such models to achieve the goal of semantic characterization. In Chapter 4, Eleftheriadis and Hong introduce *Flavor* (Formal Language for Audio-Visual Object Representation), an object-oriented language for bitstream-based media representation. The very active field of summarization and understanding of sports videos is the central topic of Chapter 5, where Assfalg, Bertini, Colombo, and Del Bimbo report their work on automatic semantic video annotation of generic sports videos, and Chapter 6, where Ekin and Tekalp propose a generic integrated semantic-syntactic event model to describe sports video, particularly soccer video, for search applications.

Section III describes techniques and algorithms used for video segmentation and summarization. In Chapter 7, Ardizzone and La Cascia review existing shot boundary detection techniques and propose a new neural network-based segmentation technique, which does not require explicit threshold values for detection of both abrupt and gradual transitions. In Chapter 8, Chua, Chandrashekhara, and Feng provide a temporal multi-resolution analysis (TMRA) framework for video shot segmentation. In Chapter 9, Smith, Watclar, and Christel describe the creation of video summaries and visualization systems through multimodal feature analysis, combining multiple forms of image, audio, and language information, and show results of evaluations and user studies under the scope of the Informedia Project at Carnegie Mellon University. In Chapter 10, Gong presents three video content summarization systems developed by multidisciplinary researchers in NEC USA, C&C Research Laboratories. These summarization systems are able to produce three kinds of motion video summaries: (1) audio-centric summary, (2) image-centric summary, and (3) audio-visual content summary. In Chapter 11, Mulhem, Gensel, and Martin present then their work on the VISU model, which allows both to annotate videos with high level semantic descriptions and to query these descriptions for generating video summaries. This discussion is followed by a broad overview of adaptive video segmentation and summarization approaches, presented by Owen and Dixon in Chapter 12. In Chapter 13, Owen, Zhou, Tang, and Xiao describe augmented imagery and its applications as a powerful tool for enhancing or repurposing content in video databases, including a number of interesting case studies. In Chapter 14, Ahmed and Karmouch present a new algorithm for video indexing, segmentation and key framing, called the binary penetration algorithm, and show its extension to a video web service over the World Wide Web for multiple video formats. Concluding the Section, in Chapter 15, Zhao and Grosky revisit the *semantic gap* problem and introduce a novel technique for spatial color indexing, *color anglogram*, and its use in conjunction with a dimension reduction technique, latent semantic indexing (LSI), to uncover the semantic correlation between video frames.

Section IV examines tools and techniques for designing and interacting with video databases. In Chapter 16, Hjelsvold and Vdaygiri present the result of their work while developing two interactive video database applications: HotStreams™ – a system for delivering and managing personalized video content – and TEMA (Telephony Enabled Multimedia Applications) – a platform for developing Internet-based multimedia applications for Next Generation Networks (NGN). In Chapter 17, Huang, Chokkareddy, and Prabhakaran introduce the topic of animation databases and present a toolkit for animation creation and editing. In Chapter 18, Djeraba, Hafri, and Bachimont explore different video exploration strategies adapted to user requirements and profiles, and introduce the notion of probabilistic prediction and path analysis using Markov models. Concluding the Section, in Chapter 19, Picariello, Sapino, and Subrahmanian introduce AVE! (Algebraic Video Environment), the first algebra for querying video.

The challenges behind audio and video indexing and retrieval are discussed in Section V. It starts with a survey of the state of the art in the area of audio content indexing and retrieval by Liu and Wang (Chapter 20). In Chapter 21, Ortega-Binderberger and Mehrotra discuss the important concept of relevance feedback and some of the techniques that have been successfully applied to multimedia search and retrieval. In Chapter 22, Santini proposes a novel approach to structuring and organizing video data using *experience units* and discusses some of its philosophical implications. Farin, Haenselmann, Kopf, Kühne, and Effelsberg describe their work on a system for video object classification in Chapter 23. In Chapter 24, Li, Tang, Ip, and Chan advocate a web-based hybrid approach to video retrieval by integrating the query-based (database) approach with the content-based retrieval paradigm and discuss the main issues involved in developing such a web-based video database management system supporting hybrid retrieval, using their VideoMAP* project as an example. In Chapter 25, Zhang and Chen examine the *semantic gap* problem in depth. The emergence of MPEG-7 and its impact on the design of video databases is the central topic of Chapter 26, where Smith discusses the topic in great technical detail and provides examples of MPEG-7-compatible descriptions. In Chapter 27, Satoh and Katayama discuss issues and approaches for indexing of large-scale (tera- to peta-byte order) video archives, and report their work on Name-It, a system that associate faces and names in news videos in an automated way by integration of image understanding, natural language processing, and artificial intelligence technologies. The next two chapters cover the important problem of similarity measures in video database systems. In Chapter 28, Cheung and Zakhor discuss the problem of video similarity measurement and propose a randomized first-order video summarization technique called the Video Signature (ViSig) method, whereas in Chapter 29, Traina and Traina Jr. discuss techniques for searching multimedia data types by similarity in databases storing large sets of multimedia data and present a flexible architecture to build content-based image retrieval in relational databases. At the end of the Section, in Chapter 30, Zhou and Huang review existing relevance feedback techniques and present a variant of discriminant analysis that is suited for small sample learning problems.

In Section VI we focus on video communications, particularly streaming, and the technological challenges behind the transmission of video across communication networks and the role played by emerging video compression algorithms. In Chapter 31, Hua and Tantaoui present several cost-effective techniques to achieve scalable video streaming, particularly for video-on-demand (VoD) systems. In Chapter 32, Shahabi and Zimmermann report their work designing, implementing, and evaluating a scalable real-time streaming architecture, Yima. In Chapter 33, Zhang, Aygün, and Song present the design strategies of a middleware for client-server distributed multimedia applications, termed *NetMedia*, which provides services to support synchronized presentations of multimedia data to higher level applications. Apostolopoulos, Tan, and Wee examine the challenges that make simultaneous delivery and playback of video difficult, and explore algorithms and systems that enable streaming of pre-encoded or live video over packet networks such as the Internet in Chapter 34. They provide a comprehensive tutorial and overview of video streaming and communication applications, challenges, problems and possible solutions, and protocols. In Chapter 35, Ghandeharizadeh and Kim discuss the continuous display of video objects using heterogeneous disk subsystems and quantify the tradeoff associated with alternative multi-zone techniques when extended to a configuration consisting of heterogeneous disk drives. In Chapter 36, Vetro and Kalva discuss the technologies, standards, and challenges that define and drive universal multimedia access (UMA). In Chapter 37, Basu, Little, Ke, and Krishnan look at the dynamic stream clustering problem, and present the results of simulations of heuristic and approximate algorithms for clustering in interactive VoD systems. In Chapter 38, Lienhart, Kozintsev, Chen, Holliman, Yeung, Zaccarin, and Puri offer an overview of the key questions in distributed video management, storage and retrieval, and delivery and analyze the technical challenges, some current solutions, and future directions. Concluding the Section, Torres and Delp provide a summary of the state of the art and trends in the fields of video coding and compression in Chapter 39.

Section VII provides the reader with the necessary background to understand video processing techniques and how they relate to the design and implementation of video databases. In Chapter 40, Wee, Shen, and Apostolopoulos present several compressed-domain image and video processing algorithms designed with the goal of achieving high performance with computational efficiency, with emphasis on transcoding algorithms for bitstreams that are based on video compression algorithms that rely on the block discrete cosine transform (DCT) and motion-compensated prediction, such as the ones resulting from predominant image and video coding standards in use today. In Chapter 41, Wang, Sheikh, and Bovik discuss the very important, and yet largely unexplored, topic of image and video quality assessment. Concluding the Section, in Chapter 42, Doërr and Dugelay discuss the challenges behind extending digital watermarking, the art of hiding information in a robust and invisible manner, to video data.

In addition to several projects, prototypes, and commercial products mentioned throughout the Handbook, Section VIII presents detailed accounts of three projects in this field, namely: an electronic clipping service under development at At&T Labs, described by Gibbon, Begeja, Liu, Renger, and Shahraray in Chapter 43; a multi-modal two-level classification framework for story segmentation in news videos, presented by Chaisorn, Chua, and Lee in Chapter 44; and the Video Scout system, developed at Philips Research Labs and presented by Dimitrova, Jasinschi, Agnihotri, Zimmerman, McGee, and Li in Chapter 45.

Finally, Section IX assembles the answers from some of the best-known researchers in the field to questions about the state of the art and future research directions in this dynamic and exciting field.

REFERENCES

[1] R. Hjelsvold, VideoSTAR – A database for video information sharing, Dr. Ing. thesis, Norwegian Institute of Technology, November 1995.

[2] R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems – 3rd edition*, Addison-Wesley, Reading, MA, 2000.

[3] C.J. Date, *An Introduction to Database Systems – 7th edition*, Addison-Wesley, Reading, MA, 2000.

[4] T. Connolly, C. Begg, and A. Strachan, *Database Systems – 2nd ed.*, Addison-Wesley, Harlow, England, 1999.

[5] K. Brown, A rich diet: Data-rich multimedia has a lot in store for archiving and storage companies, *Broadband Week*, March 5, 2001.

[6] B. Prabhakaran, *Multimedia Database Management Systems*, Kluwer, Norwell, MA, 1997.

[7] S. Santini, and R. Jain, Image databases are not databases with images", Proc. 9th International Conference on Image Analysis and Processing (ICIAP '97), Florence, Italy, September 17-19, 1997.

[8] V.S. Subrahmanian, *Principles of Multimedia Database Systems*, Morgan Kaufmann, San Francisco, CA, 1998.

[9] R.C. Veltkamp, and M. Tanase, A survey of content-based image retrieval systems, in *Content-Based Image and Video Retrieval*, O. Marques, and B. Furht, Eds., Kluwer, Norwell, MA, 2002.

[10] R. Bryll, A practical video database system, Master Thesis, University of Illinois at Chicago, 1998.

[11] A.K. Elmagarmid, H. Jiang, A.A. Helal, A. Joshi, and M. Ahmed, *Video Database Systems*, Kluwer, Norwell, MA, 1997.

[12] I. Koprinska and S. Carrato. Video segmentation: A survey", *Signal Processing: Image Communication*, 16(5), pp. 477-500, Elsevier Science, 2001.

[13] H. Lee, A.F. Smeaton, and J. Furner, User-interface issues for browsing digital video, in Proc. 21st Annual Colloquium on IR Research (IRSG 99), Glasgow, UK, 19-20 Apr. 1999.

[14] M. Davis, Media Streams: An iconic visual language for video representation, in *Readings in Human-Computer Interaction: Toward the Year 2000*, 2nd ed., R. M. Baecker, J. Grudin, W.A.S. Buxton, and S. Greenberg, Eds., Morgan Kaufmann, San Francisco, CA, 1995.