1

Introduction

Henk Blanken, Ling Feng, Maurice van Keulen, and Henk Ernst Blok

University of Twente

1.1 Introduction

People interact with multimedia every day: reading books, watching television, listening to music, etc. For quite some time we have faced astonishing technological developments causing an explosion of digital multimedia information. Large amounts of text, images, speech, and video are converted to digital form. Think of catalog information of libraries, information about museums with nice pictures of paintings or famous speeches that are available on DVD. Moreover, much information is produced directly in digital form: TV programs, audio-visual data from surveillance cameras, photos. Major advantages of digitized data over analog data are easy storage, processing and sharing of data. Multimedia applications influence our daily life. Consider for example the following scenarios.

1.1.1 Journalism

This scenario is based on a field study by Markkula and Sormunen [12]. A journalist has to write an article about the influence of drinking alcohol on driving. Of course, she does some investigations. She collects news paper articles about accidents, scientific reports, television commercials broadcasted on behalf of the government, and interviews with policemen and medical experts.

After the article has been written, she has to illustrate it with one or more photos. She searches in the publishers' photo archives, and probably tries the archives of some stock footage companies as well. The selection of "good" photos from the candidate set is very subjective, and depends mainly on visual and emotional attributes like "shocking", "funny", and so on.

1.1.2 Watching a TV program

Large digital video libraries will become more and more publicly available as a result of recent technology developments in digital video, Internet, and computers. The presentation of already recorded TV programs is possible. This implies the storage and use of huge collections of TV programs in digital

libraries. Consider a viewer who wants to see a movie. Sometimes she may be able to identify the movie precisely by providing the title and the name of the director. Another time the viewer may not be so sure. She is, however, able to define the type of movie (e.g., adventure). If the library "knows" her taste, "knows" the movies she has already seen, then some sensible suggestions can be made to her. Still another time the viewer intends to explore science fiction, a type of movie unfamiliar to her. However, her friend Cathy is very familiar with science fiction. So, she states that she would like to see a science fiction movie her friend Cathy favors.

1.1.3 Searching on the Web

Finding relevant information on the World-Wide Web is often a frustrating task, partly due to the unstructured character of the data involved. Consider the Australian Open web-site (see http://www.ausopen.org/) that contains multimedia objects like text-fragments, images, and videos. These multimedia data show the latest results of players competing at this international tennis tournament. We would like to access the Australian Open site in a user friendly way using terms like player, match, and profile. Related to these entities are attributes, like a player's name, photo, age, and history, etc. Furthermore, interesting events can be extracted from the multimedia data. Think of players approaching the net or smashing the tennis ball. The integration of conceptual terms and interesting events delivers the necessary ingredients to effectively answer a content based query such as "*Give information about female American tennis players and include video-segments showing the player going to the net*".

1.2 Retrieval problem

Retrieval of multimedia data is different from retrieval of structured data. Retrieving data from a (relational) database is rather "easy". The database structure is given and using a language like SQL a user can specify which data she requires. Suppose that there is a relation which keeps information about employees:

```
EMPLOYEE (Name: char(20), City: char(20), Photo: image)
```

Selecting employees living in the city of Amsterdam results in a query in which the condition `WHERE City = "Amsterdam"` appears. Use is made of the fact that the city of Amsterdam is identified by the text "Amsterdam". A problem arises when you want to select employees having a certain value for the attribute Photo: specifying an image value is practically impossible. So, for attributes with data type image, sound, video, etc. the "normal" way of retrieving data does not work. *This book concentrates on techniques that enable retrieval of multimedia data.* We start with taking a more detailed look at various types of multimedia data.

1.3 Characteristics of Media Data

What are the characteristics of text, audio, image, and video? It would take a lot of space to describe these media. In this introduction we confine ourselves to briefly giving some basic characteristics.

1.3.1 Medium

The general meaning of medium is a means to communicate. Here we define a medium to be a type of information representation, like alphanumeric data, audio, images, and video. Alphanumeric data are the data normally occurring in SQL-like databases containing numeric and alphabetic (text/string) data. Media types like audio, video, and image have traditionally analog representations. We are interested in digital representations and further on we pay some attention to conversion of analog to digital data.

1.3.2 Static versus Dynamic Media

Media types can be classified according to the relationship with time, which leads to two classes of media types: *static* and *dynamic* (or *time continuous*).

Static media do not have a time dimension, dynamic media do. Examples of static types are alphanumeric data, images, and graphics. Audio, animation, and video are examples of dynamic types. Presenting dynamic media puts time requirements. When the human eye sees a video played at a rate of at least 25 frames a second, then it perceives a smooth movement. Playing back music puts even stronger requirements: only playback rates in a very strict region make sense. Other rates are perceived as unnatural.

1.3.3 Multimedia

Multimedia refers to a collection of media types used together. (Notice that a collection may have only one member.) At least one of the media types must be non-alphanumeric. As stated before we are not interested in analog audio or video representation: we deal with digitized, computer readable representation. In this book we concentrate on images, video, and speech, but we also devote a chapter to text only. Text may contain alphanumeric data only. The term "multimedia" is mostly used as an adjective in phrases like multimedia information, multimedia system, and multimedia applications. We use the term *multimedia object* to refer to multimedia data to which a certain meaning has been attached, like a video of a football match Ajax–Arsenal, or an image of a Van Gogh painting.

1.3.4 Representation of Multimedia Data

Multimedia data are often represented in an analog way. Below we briefly discuss the problem of obtaining a digital representation.

Text

Plain text consists of alphanumeric characters. Optical character recognition (OCR) techniques are applied to convert analog text to digital text. The most common digital representation of characters is the American Standard Code for Information Interchange (ASCII). For each character seven bits are needed (often eight bits are used, where the eighth bit is reserved for a special purpose). Notice that Chinese characters need more space. The required storage space for a text document is equal to the number of characters. A text document of say 15 pages of about 4000 characters requires 60 kilobytes. This is quite moderate.

Structured text documents are becoming more and more popular. Such a document consists of titles, chapters, sections, paragraphs, and so on. A title may be presented to the user in a format different from a paragraph or a sentence. Standards like HTML [20] and XML [19] are used to encode structured information.

There are some techniques, e.g., Huffman and arithmetic coding [23], to compress text, but as storage requirements are not too high, the compression techniques are in general less important for text than for multimedia data.

Audio

Audio is caused by air pressure waves having a frequency and amplitude. When the frequency of the waves is between 20 to 20,000 Hertz a human hears a sound. For example, elephants are able to hear wider ranges. Besides frequency, also the amplitude of a wave is important, see Figure 1.1a (Figure 1.1 has been taken from Lu [11]). A low amplitude causes the sound to be soft.

How to digitize these pressure waveforms? First, the air wave is transformed into an electrical signal (by a microphone). This signal is converted into discrete values by processes called *sampling* and *quantization*. Sampling causes the continuous time axis to be divided into small, fixed intervals, see Figure 1.1b. The number of intervals per second is called the sampling rate. The determination of the amplitude of the audio signal at the beginning of a time interval is called quantization. So the continuous audio signal is approximated by a sequence of values, see Figure 1.1c. If the sampling rate is high enough and the quantization is precise enough the human ear will not notice any difference between the analog and digital audio signal. The process just described is called analog-to-digital conversion (ADC); the other way around is called digital-to-analog conversion (DAC).

To give an indication of storage requirements, consider a CD-audio using 16 bits per sample, having 44,000 samples per second and two (stereo) channels. This gives rise to about 1.4 Mbit per second required storage capacity. This is a lot, so compression techniques are welcome. To compress sound within the entire audible range of 20 kHz a masking technique is often used. The idea is that one sound can make it impossible to hear another as is the

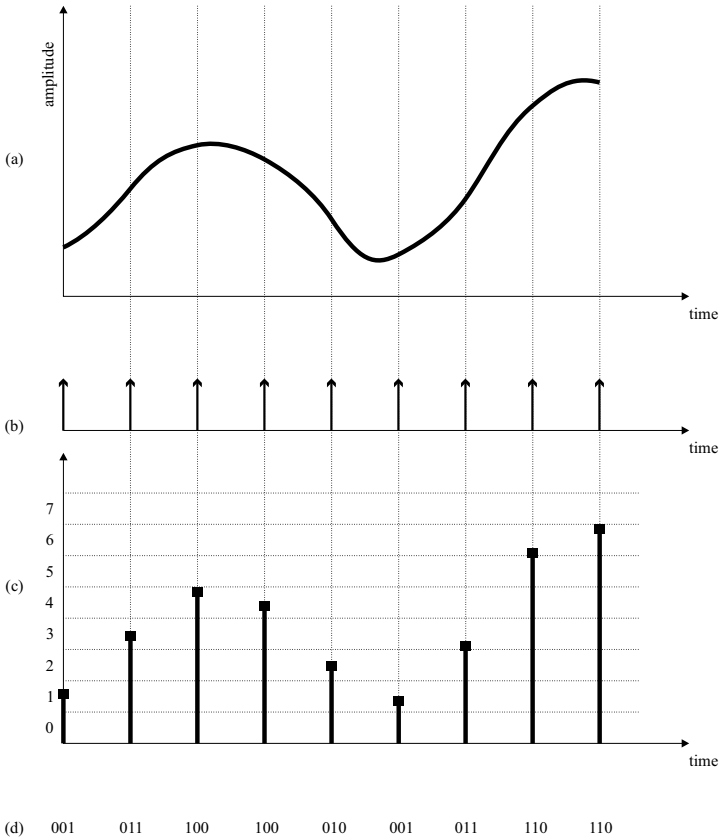

Fig. 1.1. Analog-to-digital conversion: (**a**) Original analog signal; (**b**) sampling pulses; (**c**) quantization; (d) digitized values.

case with a loud and a soft sound. The soft sound is supposed to be masked. Because masked sounds are not audible, they can safely be discarded. For speech (frequency lower than 7 kHz) other techniques are available. One technique uses the fact that low frequency speech is more important than high frequency. The coding accuracy has to be accordingly.

Moving Pictures Expert Group, abbreviated as MPEG, is a standardization group of ISO aimed to develop standards for coded representation of moving pictures, associated audio, and their combination for storage on devices (e.g., CD-ROMs) as well as telecommunication channels, e.g., LANs. MPEG has defined many standards, among others MPEG-Audio [18]. MPEG-audio is a general audio compression standard that exploits among others masking.

Image

Digital images can be obtained by scanning (analog) photos and pictures using a scanner. A scanner works according to the same principles as the ADC for

audio: the analog image is approximated by a rectangle of small dots. In digital cameras the ADC is built in. Another source of digital images is formed by the frames of a digitized or digital video.

Images can be in gray-scale or in color. An image displayed on a screen consists of many small dots or picture elements (pixels). To describe the gray-scale of a pixel we need say one byte of eight bits. For a color pixel we need three colors (e.g., Red, Green, and Blue) of one byte each. So, for a rectangular screen we can compute the amount of data required for the image using the formula

$$A = xyb$$

where A is the number of bytes needed, x is the number of pixels per horizontal line, y is the number of horizontal lines, and b is the number of bytes per pixel. For a screen with $x = 800$, $y = 600$, and $b = 3$ we get $A = 1.44$ Mbyte.

This amount of data is quite substantial, so compression is needed. Image compression is based on exploiting redundancy in images and properties of the human perception. It appears that pixels in a certain area are often similar; this is called *spatial redundancy*. Think of the yellow sand on a beach or the blue of a sky on a sunny day. Humans looking at images are tolerant regarding some information error or loss meaning that the compressed image does not need to exactly represent the original image. A compressed image with some error may still allow effective communication. Notice that this does not hold for alphanumeric data, where exact match is used for selecting data.

Several compression techniques are available, among others transform coding [5], and fractal image coding [10].

Video

A digital video consists of a sequence of frames or images that have to be presented at a fixed rate. Digital videos can be obtained by digitizing analog videos or directly by digital cameras. Playing a video at a rate of 25 frames per second gives the user the illusion of a continuous view. It takes a huge amount of data to represent a video. For example a one second video with image size of 512 lines and 512 pixels per line, 24 bits per pixel and 25 frames per second amounts to $512 \times 512 \times 3 \times 25 = 19$ Mbytes of storage. Imagine what it takes for a movie of one hour. So, compression techniques are a must!

In general the image compression techniques can also be applied to the frames of videos. The same principles as with images are used: reducing redundancies and exploiting human perception properties.

Besides spatial redundancy we also have *temporal redundancy*. This means that neighboring frames are normally similar. This redundancy can be removed by applying the motion estimation and compensation where each image is divided into fixed-size blocks. For each block in the current image the most similar block in the previous image is determined and the *pixel difference* is computed. Together with the *displacement* between the two blocks, this difference is stored and if needed transmitted. This is in general more efficient

than manipulating the current block by itself. (By the way: shot detection in a video uses the same pixel difference: as soon as this pixel difference is high a shot change is supposed to be detected.) Using compression a gain of a factor 10–100 can be achieved, which is quite substantial.

MPEG-1 [3], MPEG-2 [4], and MPEG-4 [9] deal with coding of video data of up to a speed of 1.5, 10, and 40 Megabits per second respectively. The intention of MPEG-1 is to code VHS quality video; MPEG-2 is an extension of MPEG-1 and provides high quality audio-visual encoding. MPEG-4 covers storage, transmission, and manipulation of multimedia data. It provides tools for decoding and representing atomic units of image and video objects, called "video objects", like a person in an image. While MPEG-1 and MPEG-2 perform pixel-based coding, MPEG-4 bridges the gap to object-based coding. Object-based coding makes content-based indexing and retrieval of multimedia data achievable.

Summary of Storage Requirements
The storage requirements (without compression) of some media can be roughly estimated in the following way:

a book of 500 pages	2 Mbytes
100 color images	144 Mbytes
1 hour of CD audio	635 Mbytes
1 hour of video	68.4 Gbytes .

1.3.5 Language for Composite Multimedia Documents

Multimedia documents contain data of different types. A scientific paper may consist of text, figures, and tables. A financial document on the Internet may have, besides text, also a spoken message and/or a video. There have been developed languages to describe "real" multimedia objects, e.g., SGML [1] together with HyTime [14]. We briefly describe the above mentioned language XML.

XML is an international standard and derived from SGML. The language enables the description of structured information; the description is independent of the presentation of the information. With the help of XML the type of a text document can be defined in a Document Type Definition (DTD). A document is "marked up" according to a DTD and an XML processor can check whether a document satisfies the type definition of the corresponding DTD. If true, the document is said to conform to the DTD. The DTD deals only with the structure of documents. To govern the presentation of documents that conform to a DTD, we relate structured elements of a DTD to output specifications. This happens in a separate specification expressed in the language DSSSL [6]. Roughly speaking the specification can be compared with the style sheet of an MS-Word document.

1.4 Metadata of Multimedia Objects

Let us return to the example in which we tried to find all employees having a certain value for attribute Photo. The type of attribute Photo is image. The question arises: how to search for images, or, in general, for multimedia objects? Analyzing all employee photos one by one is mostly no option as it takes too long. The standard way is to add information that describes in one way or another the multimedia object; we call this information *metadata*. Of course, these metadata have to be stored somewhere. So, instead of searching for a multimedia object directly, we search for the metadata that have been added to it. To be valuable in searching for multimedia objects, metadata have to satisfy certain requirements:

- a description of a multimedia object should be as complete as possible;
- storage of the metadata must not take too much overhead; and
- comparison of two metadata values has to be fast.

Below we distinguish several kinds of metadata.

1.4.1 Descriptive Data

Descriptive data give some format or factual information about the multimedia object. Think of author name, creation date, length of the multimedia object, used representation technique, and so on. There is a standard for descriptive data called Dublin Core [8] that gives many possibilities to describe a multimedia object. Let us assume we are looking for a certain movie and we know the name of the director and the year of release. Then we can address the metadata and formulate an SQL query in which this knowledge is represented as conditions in the WHERE-clause.

1.4.2 Annotations

Text annotation is a textual description of the contents of the object. Think of text added to photos in an album. Annotation can be a free format description or a sequence of keywords. Text is added mostly manually making it time consuming and expensive. Another disadvantage of annotation is its subjectivity and incompleteness. Later on we elaborate on so-called information retrieval techniques. These techniques can be used to find multimedia objects based on text annotation. The techniques are powerful and much used, especially in combination with techniques addressing the content of multimedia objects directly.

Besides text, also structured concepts can be used to describe the contents of multimedia objects [24]. This results in a kind of Entity-Relationship schema, which gives concepts, their relationships to each other and to multimedia objects. The advantage is that use can be made of query languages with an SQL like power. This can be a useful approach, especially in the realm of a certain company where a tight control of the schema can be realized. Conceptual annotation is manual, so, again, slow and expensive.

1.4.3 Features

Now we turn our attention to approaches which try to derive characteristics from the multimedia object itself. These derived characteristics are called *features*. To describe features a kind of language is needed. MPEG-7 [13] is by now the most important standard. In Chapter 2 we treat MPEG-7 in some detail. The process to capture features from a multimedia object is called feature extraction. This process is often performed automatically, sometimes however with human support. Two feature classes are distinguished, namely low-level and high-level features.

Low-Level Features

Low-level features grasp *data patterns and statistics* of a multimedia object and depend strongly on the medium. Low-level feature extraction is performed automatically. Let us start with the well-known text documents. Which "content" can be derived automatically? During the indexing process, words like "the", "it", "a", etc. are neglected: they do not bear any relevance for the meaning of the document. In Chapter 4 the indexing process will be explained. The result is essentially a list of keywords with frequency indicators, which is supposed to describe the content of the text document.

An audio signal can be represented by an amplitude–time sequence: for each sample value the air pressure is quantified. The amplitude belonging to the air pressure of silence is represented as 0, an air pressure higher than the silence pressure means a positive amplitude, and a lower air pressure implies a negative amplitude. We derive some low-level features from this amplitude–time sequence, among others average energy (indicates loudness of signal), zero crossing rate ZCR (indicates the frequency of sign change in signal amplitude), and silence ratio. A low silence ratio often indicates music, while a high ZCR variability often indicates speech. The Discrete Fourier Transform (DFT) of the amplitude–time representation is also used to derive low-level features. In Chapter 7 more attention is paid to these features.

What about images? When processing images you can count the number of pixels that have a color in a certain color range, giving rise to so-called *color histograms*. We can use color histograms to distinguish images. In an image also *spatial relations* may hold: we observe a spatial relation when, e.g., a blue area appears above a yellow area in a beach photo. If an image has many dark spots neighboring light spots, then it has a high score regarding the feature *contrast*. Many other features (e.g., shape, circularity) have been defined [7]. Chapter 5 deals with images.

Videos are sequences of images, so low-level features of images also apply to video. Video is a continuous medium and has as such a *temporal dimension*. Let us define a shot as a sequence of images taken with the same camera position. The end of a shot can be determined by computing the pixel difference between subsequent images. As soon as the pixel difference between two images is higher than a certain threshold we assume to have observed a shot change. Chapters 8, 9, and 10 cover video.

High-Level Features

Low-level features often have not too much meaning for the end user. Consider for instance images. What does a color histogram really mean? Much green may indicate many things: a golf course, or maybe a forest? With high-level features (or high-level concepts) we mean features, which are meaningful for the end user, like forest or golf course. There is obviously a gap between low and high level features. This gap is called the *semantic gap*. High-level feature extraction attempts to bridge this gap and tries to recognize concepts that are meaningful for the user.

An important part of this book tries to answer the question: How can we derive high-level features from low-level features? Low-level features in a text document are keywords. Luckily enough, keywords have a strong relationship with concepts in the human mind. When a document contains words like "football", and "referee" then this gives an indication of the content of the document.

It also appears that in speech recognition much progress has been made. For many languages reasonable speech to text translators have been built. Suppose we know that a certain data source contains speech. Extractors can automatically derive low-level features from speech and apparently translators successfully bridge the gap between low-level features on one hand and words and sentences on the other.

In areas like images, non-speech audio, and video deriving concepts from low-level features is in general not possible. Focusing on a special application domain fosters progress. For instance, consider videos of football matches. Observing a loud sound coming from the crowd and a round object passing a white line, followed by a sharp whistle, often indicates a semantically interesting concept: a goal. A combination of low-level features may imply a high-level feature (see also Chapters 9 and 10).

1.5 Schematic Overview of MIRS

We deal with a system that stores and retrieves multimedia objects. In this book we concentrate on the retrieval part, which is the reason that we call such a system a multimedia information retrieval system (MIRS). In Figure 1.2 we give a schematic overview of a MIRS. It shows that arriving multimedia objects are archived while metadata are extracted and stored in a so-called metadata server or index. A user poses queries in a certain way and with the help of metadata from the index an answer set is composed. But, things are not always that simple: querying has *vague* aspects. Sometimes, the user knows what she is looking for but is not able to formulate it. It may happen that a user does not know exactly what she is looking for, but she will know when she sees the right result. The user operates more or less like a woman looking for a dress in a shop. This kind of querying multimedia data is supported by a blend of browsing and searching steps. Searching and browsing result in a

list of multimedia objects that are sent to the user where it can be viewed or played.

Below we treat the parts of a MIRS in more detail.

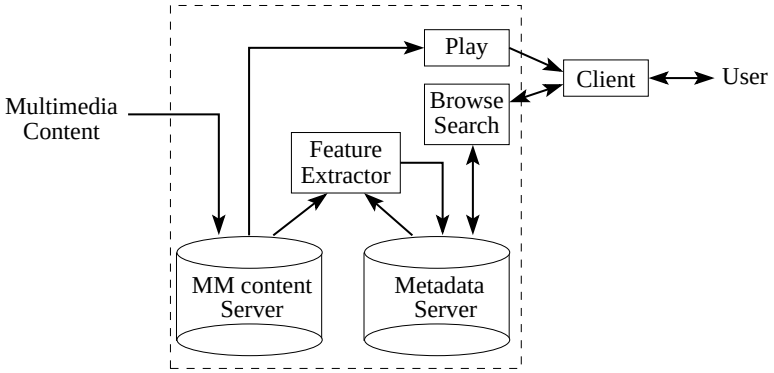

Fig. 1.2. Schematic view of a MIRS.

1.5.1 Archiving

The main characteristics of multimedia data that influence the architecture of a MIRS are that they are voluminous and that visible or audible delays in their playback are unacceptable. Hence, accessing multimedia content poses fundamentally different requirements in comparison to ordinary data. Therefore, there often is a strict separation between the "raw" multimedia content, be it audio, video, or other documents, and the metadata describing it. The multimedia content is managed separately in a special multimedia content server. At storage time a multimedia object gets an identification that can be used in other parts of the MIRS. Topics like compression and protection have to be dealt with. In this book almost no attention will be paid to compression.

1.5.2 Feature Extraction (Indexing)

Metadata are extracted from an incoming multimedia object. Metadata contain annotations and descriptions of the multimedia content, and features extracted from content.

The extraction process is also sometimes called *indexing*. Metadata play a dominant role in retrieval of multimedia objects, so it is important that retrieval of metadata is efficient. Moreover, as the metadata can be voluminous by itself, compact storage is also required. For example, the content of text documents may be indexed by a sorted list of keywords. These keywords together may require a lot of space.

Extraction Dependencies

Metadata, especially features, not only depend on the raw data of a multimedia object, but also on each other. Consider for example a content-based video search system that allows the user to search for net playing or shortest/longest rallies in tennis matches [2]. One can imagine the following steps to be taken for the required annotation of the video objects with start and end times of net playing and games:

- The video object is segmented and for each segment a key frame is chosen.
- Dominant color and other low-level features are extracted for the key frames.
- Based on the low-level features of the key frames, the segments are classified into shots of the playing field, audience and close-ups.
- From the playing field shots, the position of the player is detected in each frame.
- Another detector then extracts body-related features for the player like eccentricity, orientation, positions of the arms relative to the body, etc.
- It is then determined where rallies and net playing starts and ends.

In this example, each step presupposes the results of one or more previous steps to be available. The subsystem which controls the automatic feature extraction has to take care of these dependencies and use them to call algorithms and evaluate rules in the right order.

Incremental Maintenance

Complicating the task of the feature extraction subsystem further, the metadata stored in the database may reflect only the current status in an evolving environment. There are several possible sources of change.

- Multimedia objects may be modified. Upon modification of a multimedia object, features previously extracted from this object need to be invalidated and re-extracted. Note that this is a recursive process, because features that depend on other invalidated features, need to be invalidated and re-extracted as well.
- Detectors (feature extractors) may be modified. If an algorithm is improved (or a bug fixed), features generated by the previous version need to be invalidated and re-extracted (again recursively).
- The output/input dependencies between features may be modified. Any feature that is generated based on dependencies that have changed, needs to be invalidated and re-extracted.

Since feature extraction is an expensive process and the number of multimedia objects and features handled may be huge, it is often not feasible to simply do the feature extraction all over again when a number of such changes have occurred. A feature extraction architecture like ACOI [22] as shown in Figure 1.3 analyzes the dependencies and reruns only those extractions which

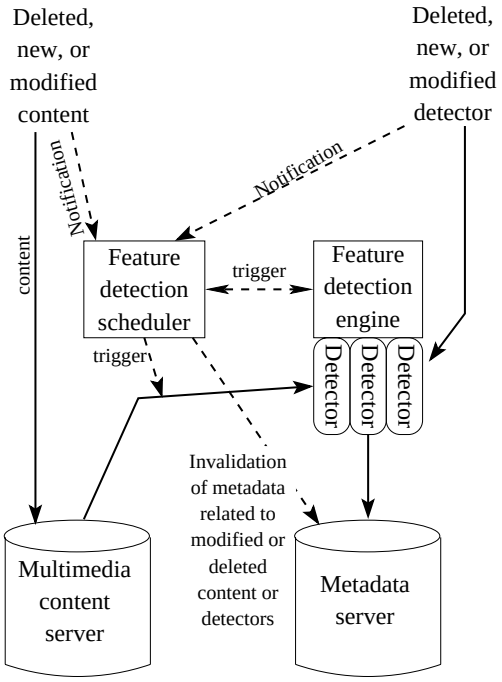

Fig. 1.3. ACOI feature extraction architecture.

are affected by the change. Any insert, delete, or modification of multimedia content or detector results in a notification to the feature detector scheduler. It invalidates the appropriate metadata and triggers the various detectors to (re-)extract features for multimedia objects in the right order. In this way, a complete rerun, including unnecessary reruns of expensive algorithms, is prevented and the database is maintained incrementally.

1.5.3 Searching

An SQL query on a relational DBMS precisely describes the information need of a user. The result of processing the query exactly satisfies the description. In other words, there is an *exact match* between specification by the user and result issued by the SQL system. In a multimedia environment this is normally not the case. How to describe a multimedia object, e.g., a photo, in a search condition? We are unable to specify the bits defining the photo. Maybe we can use low or high level features or some annotation that is added to the photo? Multimedia queries are *diverse*: the user can specify queries in many different ways. In Figure 1.2 the arrow from the client into the system indicates the issued query.

We distinguish two cases of specifying an information need, a direct and an indirect case. In the direct case the user specifies the information need by

herself. In the indirect case she relies on other users. These cases are sometimes called pull and push respectively. See Figure 1.4 for an overview. Below we discuss the possible querying scenarios.

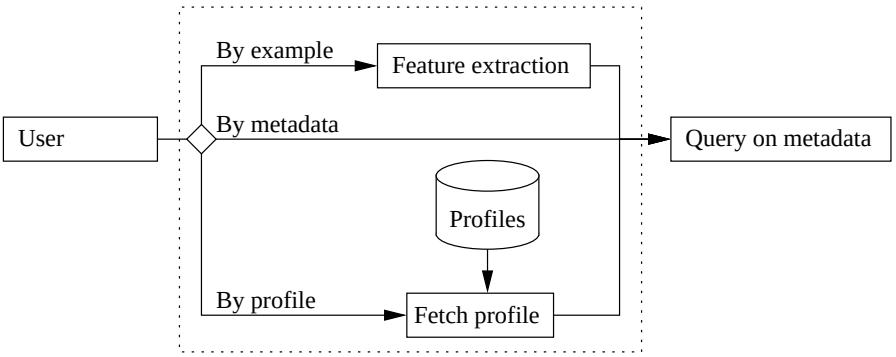

Fig. 1.4. Composition of query.

Queries Based on Profile
Let us start with the *push case*, which is also called collaborative searching. In the push case users expose their preferences in one way or another. These preferences are stored in the MIRS as a *user profile*. Suppose a user wants to buy a New Age CD, but is not certain about her taste. She trusts, however, the taste of a friend. Using the profile of the friend, the MIRS searches for CDs and returns results to the user who makes her choice.

In the *pull case* the user may exploit each of the four kinds of metadata described before, and also combinations of these. Indexes allow the MIRS to find corresponding multimedia objects. Below we describe briefly the four query types including the so-called query by example.

Queries Based on Descriptive Data
Queries can be based on format and factual information about a multimedia object. As an example consider a query about all movies with `DIRECTOR = "Steven Spielberg"`.

Queries Based on Annotations
As shown before, annotations can be text based, but also based on concepts. Let us address text based annotations. Queries can be a set of keywords or a text in natural language. An example query can be: "Show me the movie in which Tom Cruise marries Jennifer Lopez". In this case a set of keywords is derived from the query. The set of keywords is compared with the keywords occurring in text annotations of movies and similarity is computed according to information retrieval techniques (see later on).

Queries Based on Features
These queries are also called content based queries as the features are derived (semi-)automatically from the content of the multimedia object. Low and high level features can be used. We are all familiar with querying text documents by issuing a few keywords which characterizes the documents we are looking for. Another example query might be: "Give all photos with a color distribution like THIS" (THIS indicates a given photo!). Or "Give me all football videos in which a goal occurs between the 75^{th} and 90^{th} minute". In the latter query the high-level feature "goal" must be known to the MIRS.

Query by Example (Search)
Suppose you want to see photos of beach scenes. A way to indicate that is showing an example photo of a beach scene hoping that the MIRS is able to find other ones. The MIRS extracts all kinds of features from the example object. In fact, the resulting query is a query based on features.

Similarity
In multimedia retrieval the concept of similarity is important. *Similarity* describes the degree to which a query and a multimedia object of the MIRS are similar. Similarity is calculated by the MIRS and is based on metadata of the multimedia object and the query. Similarity tries to approximate the value or relevance of a multimedia object for the user. In Figure 1.5 we briefly describe the retrieval process.

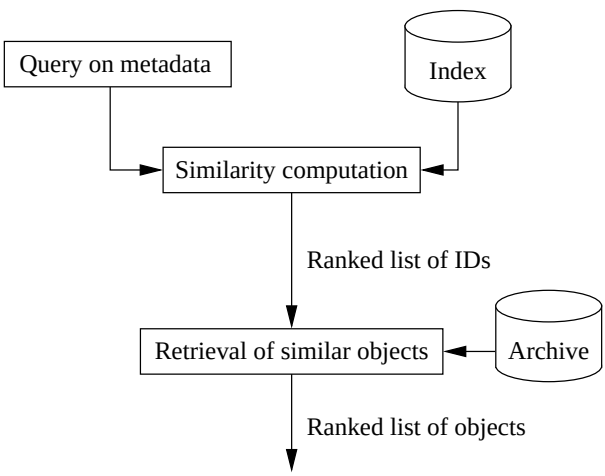

Fig. 1.5. General retrieval model.

The output is a list of multimedia objects. Normally it is a *descending ordered list*, the ordering criterion being the similarity value computed by the MIRS.

The list can be very long. As users normally do not have much patience it is very important that the most relevant objects are high on the output list. Otherwise the users might miss them. The final judgment (relevant or not) belongs always to the user. A problem is, however, that human judgment is subjective and even context dependent. So, it may happen that the computed similarity is high, while the user does not consider the object relevant.

Relevance Feedback
Users often do not know exactly what they are looking for, which causes a problem with respect to query formulation. An interactive approach can help to alleviate this problem. By specification or by issuing an example, the user formulates a starting query. The MIRS composes a result set and the user judges the output by saying relevant/not relevant to objects. The MIRS uses this *relevance feedback* to improve the retrieval process. In Figure 1.6 relevance feedback is pictured.

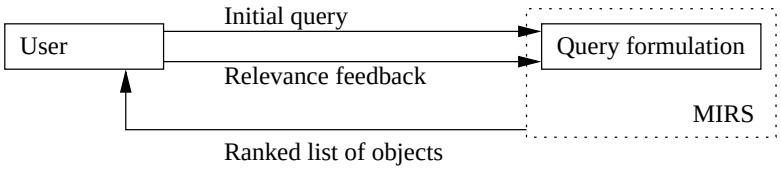

Fig. 1.6. Relevance feedback.

In a query you can of course combine the types mentioned here. If videos have text annotations, then queries based on text annotation and video features can be formulated.

1.5.4 Browsing
As stated before users often may not be able to precisely specify what they want. They can, however, recognize what they want when they see it. This phenomenon underlies relevance feedback, but also allows a process called browsing. Browsing multimedia objects means scanning through those objects. Browsing often exploits hyperlinks which lead the user from one object to the other. As soon as the MIRS shows an object, the user can judge its relevance and proceed accordingly. As objects tend to be huge often small representative "icons" are displayed in browsing mode. See also the next subsection.

There are several ways to obtain a starting point for browsing. One way is to start with a query that describes the information need as good as possible. Another way is to ask the system for a starting point. When browsing does not end up in a satisfying result, the user asks for another starting point, and so on. Finally, a third way is to classify objects in one way or another, e.g., on topic. Topics may have subtopics, and so on. This classification can be exploited by the user to obtain a sensible starting point.

1.5.5 Play: Output Presentation

A MIRS returns an (ordered) list of multimedia objects. First, the system has to determine whether the user has the right to see the objects. Now we enter the realm of digital rights management DRM (see also Chapter 12). Let us assume that the user has the necessary rights, so presentation may start. In general, the objects are huge. Moreover, the number of objects in a result set is often large. So the question arises how to service the user as good as possible.

Of course, the user interface should be able to present all kinds of multimedia data. Besides text, this means also audio, video, and images. To deal with the size of objects, the MIRS has to take measures. Presenting the whole object is out of the question, but the user still has to get a perception of the content of the object. The problem is to extract and present essential information for users in order to continue browsing and selecting objects. The familiar way for text documents is to give a title of the document with some additional information about the content: a kind of summary, or the places in the document where the query keywords occur. Summarizing audio is not that easy. From a song you can select a tune or the start. For images this means that the MIRS constructs and presents small summaries of images (called thumbnails). But how to summarize a video? We can cut a video into scenes and we can choose from each scene a prime image. Together these images give an impression of the video. Other techniques are available as well.

In many settings, multimedia content is accessed by many users through a network. Especially for audio and video, there are special requirements regarding smooth real-time playing. It is not acceptable when parts of the screen do not correspond with the rest, and that audio and video do not synchronize or that the screen is blank for short periods. Furthermore, it is usually not acceptable that a user has to wait for an audio or video file to be completely downloaded before it can be played. Therefore, a multimedia content server is often equipped with support for *streaming*. The content is sent to the client at a specific rate and, except for buffering, played directly. With current network technology, glitch-free streaming data is hard to deliver, because:

- An audio or video stream is in fact a stream of packets. This stream cannot, however, be delivered as a continuous stream of packets, but only as many individual packets that compete equally with other applications for network resources.
- The large number of packets needed to deliver audio or video consumes a huge amount of disk, bus and network resources.
- A shared Ethernet network card can drop packets due to transmission timeouts in heavy network traffic.

Measures to deal with such problems are:

- Using switched Ethernet instead of shared Ethernet establishes a dedicated link to each client.

- Disk and bus problems in the content server itself can be dealt with by using striping. An audio or video object is stored on multiple disks in a fragmented way such that reading consecutive fragment means accessing many disks at the same time.
- For glitch-free enjoyment of audio or video, it is necessary that all available frames are actually played. Skipping frames during play-back is a common technique for graceful degradation in quality when resources become scarce like insufficient network bandwidth or temporary congestion.

When the amount of multimedia content and/or number of client requests grows larger than can be handled by one server system, scalability problems come into play. The multimedia content can be fragmented over several content servers. A logical component in between client and server is needed to divert client requests to the corresponding server. To deal with the problem of too many clients, video content can be replicated over different servers. A similar intermediary component is needed for load balancing.

Sometimes presentation of multimedia objects is accompanied by advertisements, banners, and the like. A user may specify that she is not interested in those data. Of course, these wishes have to be met. Chapter 11 concentrates on user interaction meaning querying, browsing, and output presentation.

1.6 Quality of an MIRS

To compare relational database management systems, efficiency is often taken as a criterion. How fast are certain important queries answered? The underlying assumption is that correct results are obtained by all competing systems. Matters are more complicated for MIRSs as in general the results to a certain query are not equal. In other words, different MIRSs compute different result sets. The users have to decide how relevant these results are. Remember, the user determines the relevance and not the system! So, how to compare MIRSs?

The effectiveness of a system is mostly expressed in *recall* and *precision*. Let r be the number of relevant documents retrieved by the system, n the number of documents retrieved, and R the number of relevant documents in the considered collection. Then recall is defined as the fraction of the relevant objects that is actually retrieved, which is r/R. Precision is the fraction of the objects retrieved that appear to be relevant, which is r/n.

The concepts of recall and precision come from the information retrieval society. Based on these concepts, this society defined an approach to evaluate Information Retrieval systems (IR systems). The Text REtrieval Conference (TREC) [15] is a conference where IR systems are evaluated. First of all, a set of text objects is selected. Then a set of queries is defined that has to be applied to the object set. All IR systems are requested to process the queries. Often, each IR system also indicates the degree to which this object is similar to the user request. The collective results are judged by humans whether they are relevant for the queries or not. Finally, TREC determines a measure to

which the IR systems are compared and a judgment of the IR systems is computed.

In the meantime TREC has tracks involved in evaluating MIRSs [16, 17]. So now queries involving audio, video, and other media types are defined, a collection of multimedia objects is obtained, and so on. Chapter 13 addresses the problem of evaluating MIRSs.

1.7 Role of DBMSs

Multimedia data strongly differ from structured alphanumeric data. We have proposed an MIRS to deal with multimedia objects. As is known, (relational) DBMSs have been defined to handle large amounts of structured, alphanumeric data. Can a DBMS be helpful to implement an MIRS? Before answering that question we first summarize some differences between multimedia objects and structured, alphanumeric objects:

- Multimedia objects are in general huge in size.
- Continuous, multimedia objects have a temporal dimension.
- The meaning of multimedia objects is often unclear and at least subjective.
- Multimedia objects lack obvious semantic structure.
- To capture the meaning of multimedia objects, metadata have to be derived.

A DBMS has a schema that gives the structure to which the data in the database have to conform. For collections of multimedia objects it is mostly impossible to define such a schema. Moreover, a DBMS is based on the exact match paradigm, which does not apply to an MIRS. So, query formulation and query processing is essentially different. To make things worse, a DBMS is suitable to present structured, alphanumeric results, but not to present multimedia data with continuous aspects. So, a MIRS cannot be mapped simply onto a DBMS.

A DBMS can, however, be useful to manage a part of a MIRS, namely the metadata that play an important part in capturing the meaning of multimedia objects. Brief textual annotations, a color histogram, the average pitch level of an audio signal, all these data can be described by structured, alphanumeric data. So a DBMS could be used to manage metadata. This does not go, however, without problems: current DBMSs have to be extended. For example, to store and access color histograms efficiently, a DBMS must offer access and storage structures for multidimensional data. Today, many DBMSs do not offer those structures.

1.8 Role of Information Retrieval Systems

Information retrieval systems have a long history. These systems have been designed and used to allow storage and retrieval of text documents. Text documents form an important information source in most organizations. Think

of libraries containing books, letters, manuals, and so on. IR techniques can be used to query such libraries. Text can, however, also be used to annotate multimedia objects like images and videos. Therefore the same IR techniques can be used to retrieve multimedia objects. Text also results when a speech recognizer processes speech. Consider a video in which speech is transformed to text. These videos can be searched for later on using conditions applied to spoken text. So, IR techniques can be useful to support multimedia retrieval. Think, e.g., of query formulation, similarity computations, and evaluation of systems.

1.9 Organization of the Book

The book consists of 13 chapters.

In this introductory chapter, we have given a schematic description of an MIRS. In Chapter 2 we sketch languages to describe various types of metadata, namely Dublin Core, RDF, and MPEG-7/MPEG-21 [21]. The descriptions will be a starting point for mapping of metadata on database structures. During indexing metadata are stored in a DBMS. At query time a MIRS exploits these metadata to compose the result set.

Low-level features are not suitable for searching directly. Pattern recognition tries to derive a high-level description of the multimedia data. Chapter 3 introduces various methods of pattern recognition and shows what role these methods play in multimedia content analysis. As such this chapter plays a central place in the book. For example, hidden Markov models, unsupervised learning and pattern clustering, and dimension reduction are nicely described.

Chapter 4 deals with retrieval of text documents. The indexing process is described briefly. Mathematical formulas to compute similarity between a text document and a query are explained and formulas are compared.

Chapter 5 concentrates on analysis of static images. This results in a description of the content of the image. A hierarchy of data representation (pixels, points, line segments, and so on until objects) is used as a guideline. Not only image analysis, but also the other way around is covered: find images based on a description which may be abstract or based on an example. We also cover some underlying mathematics.

Chapter 6 provides generative probabilistic models for text and image retrievals. Such kinds of models offer a concise description of the (visual) characteristics of the document, which is useful in a retrieval setting.

Speech indexing is the topic of Chapter 7. Speech recognition is a pattern matching problem and the solution heavily rests on training. An architecture of speech recognizers will be given. The mathematics behind a recognition process get significant attention.

Chapter 8 proposes a generic approach for semantic video indexing. It combines some successful methods into a semantic value chain. The approach is based on the video production process covering notions of content (e.g.,

words in audio modality), style (e.g., camera distance), and context (e.g., news setting).

High-level features of video play an important part in Chapter 9. The chapter deals with a tennis case and shows how low-level features are used to automatically derive high-level features like smash, service, etc. A spatial-temporal and a probabilistic approach are described and even combined. Also, some experiments give an indication of retrieval effectiveness.

A common way to add semantics to multimedia data is to annotate with text. Chapter 10 discusses, again, an automatic approach for high-level feature derivation. High-level concepts (e.g., start of race, passing) which occur in Formula-1 car races are introduced. Low-level features are derived from videos and fed into a (dynamic) Bayesian Network to derive concepts. Parameters, i.e., conditional probabilities, of the network are obtained by a training process. Various networks are used in experiments and results are given.

Chapter 11 focuses on interaction. How to input a query and how to issue relevance feedback? How does a user keep control during browsing? Which techniques can a MIRS use to implement collaborative searching? These and other questions are covered together with issues regarding the presentation of output results.

Users may not freely retrieve all kinds of multimedia objects. Most objects are protected against unauthorized access. This is the area of digital rights management (DRM). When a user accesses a certain object, however, her privacy must be guaranteed as well. In Chapter 12 various techniques regarding protection of content and privacy are described. Attention is paid to the balance between protection of content and privacy of user.

Finally, Chapter 13 elaborates on the important topic of evaluating multimedia retrieval systems. Efficiency is of minor importance in this chapter. Most attention is devoted to effectiveness. As an MIRS deals with many media types, evaluation becomes much more complicated than in the context of traditional text documents. Questions to be answered are: which media to include, which queries to be selected, which collections of multimedia objects to choose. Notice that a query may address many media. The TREC conferences, well-known in the information retrieval community play an important part. TREC has extended its task to include also multimedia data.

References

1. ISO/IEC JTC 1/SC34. JTC 1/SC 34 – Document Description and Processing Languages. `http://www.jtc1sc34.org/`.
2. Henk Ernst Blok, Menzo Windhouwer, Roelof van Zwol, Milan Petkovic, Peter M. G. Apers, Martin L. Kersten, and Willem Jonker. Flexible and scalable digital library search. In *Proceedings of 27th International Conference on Very Large Data Bases (VLDB), September 11–14, 2001, Roma, Italy*, pages 705–706. Morgan Kaufmann, September 2001.
3. L. Chiariglione. Short MPEG-1 description. `http://www.chiariglione.org/mpeg/`, June 1996. ISO/IECJTC1/SC29/WG11 N MPEG96, Final.

4. L. Chiariglione. Short MPEG-2 description. http://www.chiariglione.org/mpeg/, October 2000. ISO/IECJTC1/SC29/WG11 N MPEG00, Final.
5. R.J. Clarke. *Transform Coding of Image*. Academic Press, 1984.
6. J. Farreres. *The DSSSL Book: An XML/SGML Programming Language*. Springer, September 2003.
7. Myron Flickner, Harpreet Sawhney, Wayne Niblack, Janathan Ashley, Qian Huang, Byron Dom, Monika Gorkani, Jim Hafner, Denis Lee, Dragutin Petkovic, David Steele, and Peter Yanker. Query by Image and Video Content: The QBIC System. *IEEE Computer*, 28(9):23–32, September 1995.
8. Dublin Core Metadata Initiative. Dublin core metadata element set, version 1.1: Reference description. http://www.dublincore.org/documents/desc/, 1997.
9. R. Koenen. MPEG-4 overview. http://www.chiariglione.org/mpeg/, March 2002. ISO/IECJTC1/SC29/WG11 N4668, Jeju meeting.
10. G. Lu. Fractal Image Compression. *Signal Processing: Image Communication*, 4(4):327–343, October 1993.
11. G. Lu. *Multimedia Database Management Systems*. Artech House, Inc., 1999.
12. M. Markkula and E. Sormunen. Searching for photos – journalists'practices in pictorial IR. In *The challenge of image retrieval*, Newcastle upon Tyne, UK, 1998. University of Northumbria.
13. J.M. Martinez. Overview of the MPEG-7 standard. http://www.chiariglione.org/mpeg/, July 2002. ISO/IEC JTC1/SC29/WG11 N4980, Klagenfurt meeting.
14. S. Newcomb, N. Kipp, and V. Newcomb. The HYTIME, Hypermedia Time based Document Structuring Language. *Communications of the ACM*, 34(11):67–83, 1991.
15. National Institute of Standards and Technology (NIST). TREC Overview. http://trec.nist.gov/overview.html.
16. National Institute of Standards and Technology (NIST). TREC Tracks. http://trec.nist.gov/tracks.html.
17. National Institute of Standards and Technology (NIST). TREC Video Retrieval Evaluation. http://www.itl.nist.gov/iaui/894.02/projects/trecvid.
18. D. Pan. A tutorial on MPEG/Audio Compression. *IEEE Multimedia*, 2(2):60–74, 1995.
19. W3C. Extensible Markup Language (XML). http://www.w3.org/XML/.
20. W3C. HyperText Markup Language (HTML). http://www.w3.org/MarkUp/.
21. W3C. Resource Description Framework (RDF). http://www.w3.org/RDF/.
22. M.A. Windhouwer. *Feature Grammar Systems – Incremental Maintenance of Indexes to Digital Media Warehouses*. PhD thesis, Universiteit van Amsterdam, Amsterdam, The Netherlands, November 2003.
23. I.H. Witten, A. Moffat, and T.C. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann Publishing, San Francisco, 2nd edition, May 1999. ISBN 1-55860-570-3.
24. R. van Zwol. *Modelling and searching web-based document collections*. Ph.D. thesis, University of Twente, Enschede, The Netherlands, April 2002.