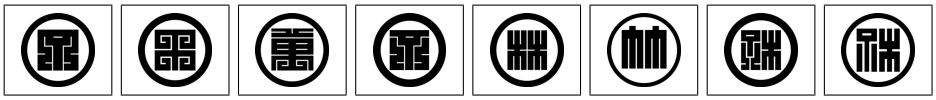
part iii. multimedia information retrieval

.. my history might well be your future ...
ted nelson

5. information retrieval
6. content annotation
7. information system architecture



2

reading directives In the following chapters we will discuss how we can make the various media formats, including text, images, audio and video amenable to search, either by analyzing content or by providing explicit meta information. For video, in particular, we develop a simple annotation logic that captures both the story line and the actors, that is persons and objects, that figure in it.

Essential sections are section 5.1, that characterizes scenarios for information retrieval, section 5.3, that introduces standard information retrieval concepts stemming from text search, section 6.4, that defines the aforementioned annotation logic, and section 7.2, that gives an outline of an abstract multimedia data format.

Section 6.3 is rather technical and may safely be skipped. Also sections 5.2, 6.1 and 7.3 may be skipped on first reading.

perspectives Apart from the many technical issues in information retrieval, perhaps the human interaction issues are the most urgent. As possible perspectives to look at these issues. consider:

perspectives – multimedia information retrieval

- application(s) – digital dossier
- psychological – focus
- experimental – user interaction
- algorithmic – (information) access
- system – unified presentation space
- presentation – embodied agents
- search – semantic annotation
- commercial – future systems

As you will see in the *research directions* given for each section, there are many proposals to improve interaction, for example the use of 3D virtual environments as an alternative way of presenting information.

essay topics For further study you may want to look at algorithms for analyzing content, annotation schemes for pareticular application domains, or the presentation issues mentioned before. Possible essay titles are:

- searching the web – searching for images, video and sound
- finding a tune – mobile music search services

Since the retrieval problem seems to be rather intractable in a general fashion, you should limit your discussion to a specific domain, for example retrieval in the domain of cultural heritage, and relate technical issues to the requirements of users in that particular domain.



the artwork

1. kata – japanese martial arts picture.
2. signs – japanese coats of arms, Signs, p. 140, 141.
3. photographs – Jaap Stahlie[1], two early experiments (left, and right)

[1] www.jaapstahlie.com

5. information retrieval

information retrieval is usually an afterthought

learning objectives

After reading this chapter you should be able to describe scenarios for information retrieval, to explain how content analysis for images can be done, to characterize similarity metrics, to define the notions of recall and precision, and to give an example of frequence tables, as used in text search.

Searching for information on the web is cumbersome. Given our experiences today, we may not even want to think about searching for multimedia information on the (multimedia) web. Nevertheless, in this chapter we will briefly sketch one of the possible scenarios indicating the need for multimedia search. In fact, once we have the ability to search for multimedia information, many scenarios could be thought of. As a start, we will look at two media types, images and documents. We will study search for images, because it teaches us important lessons about content analysis of media objects and what we may consider as *being similar*. Perhaps surprisingly, we will study text documents because, due to our familiarity with this media type, text documents allow us to determine what we may understand by effective search.

1

5.1 scenarios

Multimedia is not only for entertainment. Many human activities, for example medical diagnosis or scientific research, make use of multimedia information. To get an idea about what is involved in multimedia information retrieval look at the following scenario, adapted from Subrahmanian (1998),

Amsterdam Drugport

> *Amsterdam is an international centre of traffic and trade. It is renowned for its culture and liberal attitude, and attracts tourists from various ages, including young tourists that are attracted by the availability of soft drugs. Soft drugs may be obtained at so-called coffeeshops, and the possession of limited amounts of soft drugs is being tolerated by the authories.*
>
> *The European Community, however, has expressed their concern that Amsterdam is the centre of an international criminal drug operation. Combining national and international police units, a team is formed to start an exhaustive investigation, under the code name* Amsterdam Drugport.

Now, without bothering ourselves with all the logistics of such an operation, we may establish what sorts of information will be gathered during the investigation, and what support for (multimedia) storage and (multimedia) information retrieval must be available.

Information can come from a variety of sources. Some types of information may be gathered continuously, for example by video cameras monitoring parking lots, or banks. Some information is already available, for example photographs in a (legacy database) police archive. Also of relevance may be information about financial transactions, as stored in the database of a bank, or geographic information, to get insight in possible dug traffic routes.

From a perspective of information storage our informatio (data) include the following media types: images, from photos; video, from surveillance; audio, from interviews and phone tracks; documents,from forensic research and reports; handwriting, from notes and sketches; and structured data, from for example bank transactions.

We have to find a way to store all these data by developing a suitable multimedia information system architecture, as discussed in chapter 6. More importantly, however, we must provide access to the data (or the information space, if you will) so that the actual police investigation is effectively supported. So, what kind of queries can we expect? For example, to find out more about a murder which seems to be related to the drugs operation.

retrieval

- *image query* – all images with this person
- *audio query* – identity of speaker
- *text query* – all transactions with BANK Inc.
- *video query* – all segments with victim
- *complex queries* – convicted murderers with BANK transactions
- *heterogeneous queries* – photograph + murderer + transaction
- *complex heterogeneous queries* – *in contact with* + murderer + transaction

Apparently, we might have simple queries on each of the media types, for example to detect the identity of a voice on a telephone wiretap. But we may also have more complex queries, establishing for example the likelihood that a murderer known by the police is involved, or even *heterogeneous* queries (as they are called in Subrahmanian (1998)), that establish a relation between information coming

from multiple information sources. An example of the latter could be, *did the person on this photo have any transactions with that bank in the last three months*, or nore complex, *give me all the persons that have been in contact with the victim (as recorded on audio phonetaps, photographs, and video surveillance tapes) that have had transactions with that particular bank*.

I believe you'll have the picture by now. So what we are about to do is to investigate how querying on this variety of media types, that is images, text, audio and video, might be realized.

2

research directions– *information retrieval models*

Information retrieval research has quite a long history, with a focus on indexing text and developing efficient search algorithms. Nowadays, partly due to the wide-spread use of the web, research in information retrieval includes modeling, classification and clustering, system architectures, user interfaces, information visualisation, filtering, descriptive languages, etcetera. See Baeza-Yates and Ribeiro-Neto (1999).

Information retrieval, according to Baeza-Yates and Ribeiro-Neto (1999), deals with the representation, storage, organisation of, and access to information items. To see what is involved, imagine that we have a (user) query like:

find me the pages containg information on ...

Then the goal of the information retrieval system is to retrieve information that is useful or relevant to the user, in other words: *information that satisfies the user's information need.*

Given an information repository, which may consist of web pages but also multimedia objects, the information retrieval system must extract syntactic and semantic information from these (information) items and use this to match the user's information need.

Effective information retrieval is determined by, on the one hand, the *user task* and, on the other hand, the *logical view* of the documents or media objects that constitute the information repository. As user tasks, we may distinguish between *retrieval* (by query) and *browsing* (by navigation). To obtain the relevant information in retrieval we generally apply *filtering*, which may also be regarded as a ranking based on the attributes considered most relevant.

The logical view of text documents generally amounts to a set of index terms characterizing theb document. To find relevant index terms, we may apply operations to the document, such as the elimination of stop words or text stemming. As you may easily see, full text provides the most complete logical view, whereas a small set of categories provides the most concise logical view. Generally, the user task will determine whether semantic richness or efficiency of search will be considered as more important when deciding on the obvious tradeoffs involved.

information retrieval models In Baeza-Yates and Ribeiro-Neto (1999), a great variety of information retrieval models is described. For your understanding, an information retrieval model makes explicit how index terms are represented and how the index terms characterizing an information item are matched with a query.

When we limit ourselves to the classic models for search and filtering, we may distinguish between:

information retrieval models

- boolean or set-theoretic models
- vector or algebraic models
- probabilistic models

Boolean models typically allow for *yes/no* answers only. The have a set-theoretic basis, and include models based on fuzzy logic, which allow for somewhat more refined answers.

Vector models use algebraic operations on vectors of attribute terms to determine possible matches. The attributes that make up a vector must in principle be orthogonal. Attributes may be given a weight, or even be ignored. Much research has been done on how to find an optimal selection of attributes for a given information repository.

Probabilistic models include general inference networks, and belief networks based on Bayesan logic.

Although it is somewhat premature to compare these models with respect to their effectiveness in actual information etrieval tasks, there is, according to Baeza-Yates and Ribeiro-Neto (1999), a general consensus that vector models

will outperform the probabilistic models on general collections of text documents. How they will perform for arbitrary collections of multimedia objects might be an altogether different question!

Nevertheless, in the sections to follow we will focus primarily on generalized vector representations of multimedia objects. So, let's conclude with listing the advantages of vector models.

vector models

- attribute term weighting scheme improves performance
- partial matching strategy allows retrieval of approximate material
- metric distance allows for sorting according to degree of similarity

Reading the following sections, you will come to understand how to adopt an attribute weighting scheme, how to apply partial matching and how to define a suitable distance metric.

So, let me finish with posing a research issue: *How can you improve a particular information retrieval model or matching scheme by using a suitable method of knowledge representation and reasoning?* To give you a point of departure, look at the logic-based multimedia information retrieval system proposed in Fuhr et al. (1998).

5.2 images

An image may tell you more than 1000 words. Well, whether images are indeed a more powerful medium of expression is an issue I'd rather leave aside. The problem how to get information out of an image, or more generally how to query image databases is, in the context of our *Amsterdam Drugport* operation more relevant. There are two issues here

- obtaining descriptive information
- establishing similarity

These issues are quite distinct, although descriptive information may be used to establish similarity.

descriptive information

When we want to find, for example, all images that contain a person with say sunglasses, we need to have of the images in our database that includes this information one way or another. One way would be to annotate all images with (meta) information and describe the objects in the picture to some degree of detail. More challenging would be to extract image content by image analysis, and produce the description (semi) automatically.

According to Subrahmanian (1998), content-based description of images involves the identification of objects, as well as an indication of where these objects are located in the image, by using a *shape descriptor* and possibly *property descriptors* indicating the pictorial properties of a particular region of the object or image.

Shape and property descriptors may take a form as indicated below.

shape

- bounding box – (XLB,XUB,YLB,YUB)

property

- property – name=value

As an example of applying these descriptors.

example

shape descriptor: XLB=10; XUB=60; YLB=3; YUB=50
property descriptor: pixel(14,7): R=5; G=1; B=3

Now, instead of taking raw pixels as the unit of analysis, we may subdivide an image in a grid of cells and establish properties of cells, by some suitable algorithm.

definitions

- image grid: $(m * n)$ cells of equal size
- cell property: (Name, Value, Method)

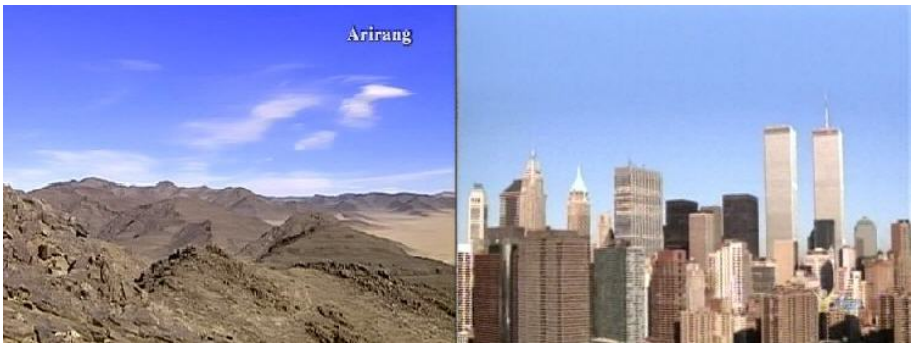
As an example, we can define a property that indicates whether a particular cell if black or white.

example

property: (bwcolor,{b,w},bwalgo)

The actual algorithm used to establish such a property might be a matter of choice. So, in the example it is given as an explicit parameter.

From here to automatic content description is, admittedly, still a long way. We will indicate some research directions at the end of this section.



similarity-based retrieval

We need not necessarily know what an image (or segment of it) depicts to establish whether there are other images that contain that same thing, or something similar to it. We may, following Subrahmanian (1998), formulate the problem of similarity-based retrieval as follows:

> *How do we determine whether the content of a segment (of a segmented image) is similar to another image (or set of images)?*

Think of, for example, the problem of finding all photos that match a particular face.

According to Subrahmanian (1998), there are two solutions:

- *metric approach* – distance between two image objects
- *transformation approach* – relative to specification

As we will see later, the transformation approach in some way subsumes the metric approach, since we can formulate a distance measure for the transformation approach as well.

metric approach What does it mean when we say, the distance between two images is less than the distance between this image and that one. What we want to express is that the first two images (or faces) are more alike, or maybe even identical.

Abstractly, something is a distance measure if it satisfies certain criteria.

metric approach

distance $d : X \to [0, 1]$ is distance measure if:

$$\begin{aligned} &\mathrm{d(x,y)} = \mathrm{d(y,x)} \\ &\mathrm{d(x,y)} \leqslant \mathrm{d(x,z)} + \mathrm{d(z,y)} \\ &\mathrm{d(x,x)} = 0 \end{aligned}$$

For your intuition, it is enough when you limit yourself to what you are familiar with, that is measuring distance in ordinary (Euclidian) space.

Now, in measuring the distance between two images, or segments of images, we may go back to the level of pixels, and establish a distance metric on pixel properties, by comparing all properties pixel-wise and establishing a distance.

pixel properties

- objects with pixel properties $p_1, \ldots, p_n$
- pixels: $(x, y, v1, \ldots, v_n)$
- object contains w x h (n+2)-tuples

Leaving the details for your further research, it is not hard to see that even if the absolute value of a distance has no meaning, relative distances do. So, when an image contains a face with dark sunglasses, it will be closer to (an image of) a face with dark sunglasses than a face without sunglasses, other things being

equal. It is also not hard to see that a pixel-wise approach is, computationally, quite complex. An object is considered as

complexity

> a set of points in k-dimensional space for k = n + 2

In other words, to establish similarity between two images (that is, calculate the distance) requires n+2 times the number of pixels comparisons.

feature extraction Obviously,we can do better than that by restricting ourselves to a pre-defined set of properties or features.

feature extraction

- maps object into s-dimensional space

For example, one of the features could indicate whether or not it was a face with dark sunglasses. So, instead of calculating the distance by establishing color differences of between regions of the images where sunglasses may be found, we may limit ourselves to considering a binary value, yes or no, to see whether the face has sunglasses.

Once we have determined a suitable set of features that allow us to establish similarity between images, we no longer need to store the images themselves, and can build an index based on feature vectors only, that is the combined value on the selected properties.

Feature vectors and extensive comparison are not exclusive, and may be combined to get more precise results. Whatever way we choose, when we present an image we may search in our image database and present all those objects that fall within a suitable *similarity range*, that is the images (or segments of images) that are close enough according to the distance metric we have chosen.

4

transformation approach Instead of measuring the distance between two images (objects) directly, we can take one image and start modifying that until it exactly equals the target image. In other words, as phrased in Subrahmanian (1998), the principle underlying the transformation approach is:

transformation approach

> *Given two objects* o1 *and* o2*, the level of* dissimilarity *is proportional to the (minimum) cost of transforming object* o1 *into object* o2 *or vice versa*

Now, this principle might be applied to any representation of an object or image, including feature vectors. Yet, on the level of images, we may think of the following operations:

> $to_1, \ldots, to_r$ – translation, rotation, scaling

Moreover, we can attach a cost to each of these operations and calculate the cost of a transformation sequence TSby summing the costs of the individual operations. Based on the cost function we can define a distance metric, which we call for obvious reasons the *edit distance*, to establish similarity between objects.

cost

- $cost(TS) = \Sigma_{i=1}^{r} cost(to_i)$

distance

- d(o,o') = min { cost(TS) | TS in TSeq(o,o') }

An obvious advantage of the *edit distance* over the pixel-wise distance metric is thatwe may have a rich choice of transformation operators that we can attach (user-defined) cost to at will.

For example, we could define low costs for normalization operations, such as scaling and rotation, and attach more weight tooperations that modify color values or add shapes. For face recognition, for example, we could attribute low cost to adding sunglasses but high cost to changing the sex.

To support the *transformation approach* at the image level, our image database needs to include suitable operations. See Subrahmanian (1998).

operations

> rotate(image-id,dir,angle)
> segment(image-id, predicate)
> edit(image-id, edit-op)

We might even think of storing images, not as a collection of pixels, but as a sequence of operations on any one of a given set of base images. This is not such a strange idea as it may seem. For example, to store information about faces we may take a base collection of prototype faces and define an individual face by selecting a suitable prototype and a limited number of operations or additional properties.

5

example(s) – *match of the day*

The images in this section present a *match of the day*, which is is part of the project *split rpresentation* by the Dutch media artist Geert Mul. As explain in the email sending the images, about once a week, *Television images are recorded at random from satellite television and compared with each other. Some 1000.000.000 (one billion) equations are done every day.*

. The *split representation* project uses the image analyses and image composition software *NOTATION*[2], which was developed by Geert Mul (concept) and Carlo Preize (programming ∓ software design).

research directions – *multimedia repositories*

What would be the proper format to store multimedia information? In other words, what is the shape multimedia repositories should take? Some of the issues involved are discussed in chapter , which deals with information system architectures. With respect to image repositories, we may rephrase the question into *what support must an image repository provide, minimally, to allow for efficient access and search?*. In Subrahmanian (1998), we find the following answer:

image repository

- *storage* – unsegmented images
- *description* – limited set of features
- *index* – feature-based index
- *retrieval* – distance between feature vectors

And, indeed, this seems to be what most image databases provide. Note that the actual encoding is not of importance. The same type of information can be encoded using either XML, relational tables or object databases. What is of importance is the functionality that is offered to the user, in terms of storage and retrieval as well as presentation facilities.

[2] homepage.mac.com/geertmul2

What is the relation between presentation facilities and the functionality of multimedia repositories? Consider the following mission statement, which is taken from my research and projects page.

mission

> *Our goal is to study aspects of the deployment and architecture of virtual environments as an interface to (intelligent) multimedia information systems <black>...*

Obviously, the underlying multimedia repository must provide adequate retrieval facilities and must also be able to deliver the desired objects in a format suitable for the representation and possibly incorporation in such an environment. Actually, at this stage, I have only some vague ideas about how to make this vision come through. Look, however, at chapter and appendix **??** for some initial ideas.

6

5.3 documents

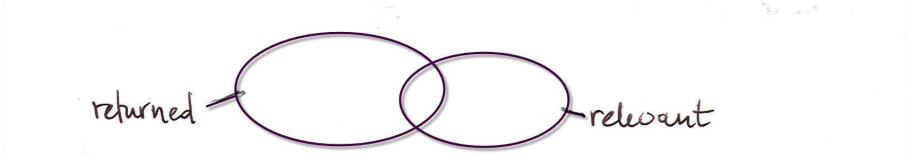
Even in the presence of audiovisual media, text will remain an important vehicel for human communication. In this section, we will look at the issues that arise in querying a text or document database. First we will characterize more precisely what we mean by effective search, and then we will study techniques to realize effective search for document databases.

Basically, answering a query to a document database comes down to string matching. However, some problems may occur such as synonymy and polysemy.

problems

- synonymy – topic T does not occur literally in document D
- polysemy – some words may have many meanings

As an example, *church* and *house of prayer* have more or less the same meaning. So documents about churches and cathedrals should be returned when you ask for information about 'houses of prayer'. As an exampleof polysemy, think of the word *drum*, which has quite a different meaning when taken from a musical perspective than from a transport logistics perspective.



precision and recall

Suppose that, when you pose a query, everything that is in the database is returned. You would probably not be satisfied, although every relevant document will be included, that is for sure. On the other hand, when nothing is returned, at least you cannot complain about non-relevant documents that are returned, or can you?

In Subrahmanian (1998), the notions of *precision* and *recall* are proposed to measure the effectiveness of search over a document database. In general, precision and recall can be defined as follows.

effective search

- precision – how many answers are correct
- recall – how many of the right documents are returned

For your intuition, just imagine that you have a database of documents. With full knowledge of the database you can delineate a set of documentsthatare of relevance to a particular query. Also, you can delineate a set that will be returned by some given search algorithm. Then, *precision* is the intersection of the two sets in relation to whatthe search algorithm returns, and *recall* that same intersection in relation to what is relevant. In pseudo-formulas, we can express this as follows:

precision and recall

```
precision = ( returned and relevant ) / returned
recall = ( returned and relevant ) / relevant
```

Now, as indicated in the beginning, it is not to difficult to get either perfect recall (by returning all documents) or perfect precision (by returning almost nothing). But these must be considered anomalies (that is, sick cases), and so the problem is to find an algorithm that performs optimally with respect to both precision and recall.

For the total database we can extend these measures by taking the averages of precision and recall for all topics that the database may be queried about.

Can these measures only be applied to document databases? Of course not, these are general measures that can be applied to search over any media type!

frequency tables

A *frequency table* is an example of a way to improve search. Frequency tables, as discussed in Subrahmanian (1998), are useful for documents only. Let's look at an example first.

example

term/document	d0	d1	d2
snacks	1	0	0
drinks	1	0	3
rock-roll	0	1	1

Basically, what a frequency table does is, as the name implies, give a frequency count for particular words or phrases for a number of documents. In effect, a

complete document database may be summarized in a frequency table. In other words, the frequency table may be considered as an index to facilitate the search for similar documents.

To find a similar document, we can simply make a word frequency count for the query, and compare that with the colums in the table. As with images, we can apply a simpledistance metric to find the nearest (matching) documents. (In effect, we may take the square root for the sum of the squared differences between the entries in the frequence count as our distance measure.)

The complexity of this algorithm may be characterized as follows:

complextity

compare term frequencies per document – O(M*N)

where M is the number of terms and N is the number of documents. Since both M and N can become very large we need to make an effort to reduce the size of the frequency table.

reduction

- stop list – irrelevant words
- word stems – reduce different words to relevant part

We can, for example, introduce a *stop list* to prevent irrelevant words to enter the table, and we may restrict ourselves to including *word stems* only, to bring back multiple entries to one canonical form. With some additional effort we could even deal with synonymy and polysemy by introducing, respectively equivalence classes, and alternatives (although we then need a suitable way for ambiguation). By the way, did you notice that frequency tables may be regarded as feature vectors for documents?

7

research directions– *user-oriented measures*

Even though the reductions proposed may result in limiting the size of the frequency tables, we may still be faced with frequency tables of considerable size. One way to reduce the size further, as discussed in Subrahmanian (1998), is

to apply *latent sematic indexing* which comes down to clustering the document database, and limiting ourselves to the most relevant words only, where relevance is determined by the ratio of occurrence over the total number of words. In effect, the less the word occurs, the more discriminating it might be. Alternatively,the choice of what words are considered relevant may be determined by taking into account the area of application or the interest of a particular group of users.

8

user-oriented measures Observe that, when evaluating a particular information retrieval system, the notions of precision and recall as introduced before are rather system-oriented measures, based on the assumption of a user-independent notion of relevance. However, as stated in Baeza-Yates and Ribeiro-Neto (1999), different users might have a different interpretation on which document is relevant. In Baeza-Yates and Ribeiro-Neto (1999), some user-oriented measures are briefly discussed, that to some extent cope with this problem.

user-oriented measures

- *coverage ratio* – fraction of known documents
- *novelty ratio* – fraction of new (relevant) documents
- *relative recall* – fraction of expected documents
- *recall effort* – fraction of examined documents

Consider a reference collection, an example information request and a retrieval strategy to be evaluated. Then the *coverage ratio* may be defined as the fraction of the documents known to be relevant, or more precisely the number of (known) relevant documents retrieved divided by the total number of documents known to be relevant by the user.

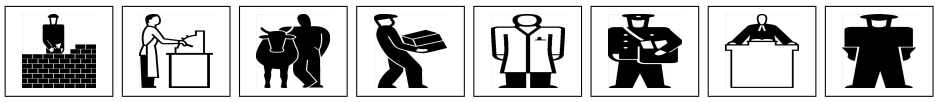
The *novelty ratio* may then be defined as the fraction of the documents retrieved which were not known to be relevant by the user, or more precisely the number of relevent documents that were not known by the user divided by the total number of relevant documents retrieved.

The *relative recall* is obtained by dividing the number of relevant documents found by the number of relevant documents the user expected to be found.

Finally, *recall effort*may be characterized as the ratio of the number of relevant documents expected and the total number of documents that has to be examined to retrieve these documents.

Notice that these measures all have a clearly 'subjective' element, in that, although they may be generalized to a particular group of users, they will very likely not generalize to all groups of users. In effect, this may lead to different retrieval strategies for different categories of users, taking into account levelof expertise and familiarity with the information repository.

5.4 development(s) – tags, labels and descriptors



9

questions

information retrieval

1. (*) What is meant by the *complementarity of authoring and retrieval*? Sketch a possible scenario of (multimedia) information retrieval and indicate how this may be implemented. Discuss the issues that arise in accessing multimedia information and how content annotation may be deployed.

concepts

2. How would you approach *content-based description* of *images*?
3. What is the difference between a *metric* approach and the *transformational* approach to establishing similarity between images?
4. What problems may occur when searching in text or document databases?

technology

5. Give a definition of: *shape descriptor* and *property descriptor*. Give an example of each.
6. How would you define *edit distance*?
7. Characterize the notions *precision* and *recall*.
8. Give an example (with explanation) of a *frequency table*.

projects & further reading As a project, you may implement simples image analysis algorithms that, for example, extract a color histogram, or detect the presence of a horizon-like edge.

You may further explore scenarios for information retrieval in the cultural heritage domain. and compare this with other applications of multimedia information retrieval, for example monitoring in hospitals.

For further reading I suggest to make yourself familiar with common techniques in information retrieval as described in Baeza-Yates and Ribeiro-Neto (1999), and perhaps devote some time to studying image analisis, Gonzales and Wintz (1987).

the artwork

1. artworks – ..., Miro, Dali, photographed from Kunstsammlung Nordrhein-Westfalen, see artwork 2.
2. left Miro from Kunst, right: Karel Appel
3. *match of the day* (1) – Geert Mul
4. *match of the day* (2) – Geert Mul
5. *match of the day* (3) – Geert Mul
6. *mario ware* – taken from gammo/veronica[3].
7. *baten kaitos – eternal ways and the lost ocean*, taken from gammo/veronica.
8. idem.
9. signs – people, van Rooijen (2003), p. 252, 253.

The art opening this chapter belongs to the tradition of 20th century art. It is playful, experimental, with strong existential implications, and it shows an amazing *variety of styles*.

The examples of *match of the day* by Geert Mul serve to illustrate the interplay between *technology and art*, and may also start you to think about what *similarity* is. Some illustrations from games are added to show the difference in styles.

[3] www.gammo.nl

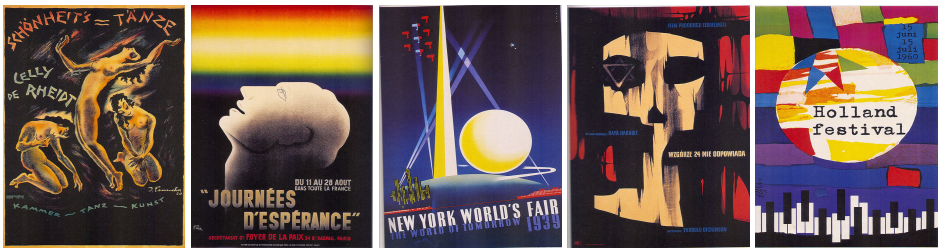
6. content annotation

video annotation requires a logical approach to story telling

learning objectives

After reading this chapter you should be able to explain the difference between content and meta information, to mention relevant content parameters for audio, to characterize the requirements for video libraries, to define an annotation logic for video, and to discuss feature extraction in samples of musical material.

Current technology does not allow us to extract information automatically from arbitrary media objects. In these cases, at least for the time being, we need to assist search by annotating content with what is commonly referred to as meta-information. In this chapter, we will look at two more media types, in particular audio and video. Studying audio, we will learn how we may combine feature extraction and meta-information to define a data model that allows for search. Studying video, on the other hand, will indicate the complexity of devising a knowledge representation scheme that captures the content of video fragments. Concluding this chapter, we will discuss an architecture for feature extraction for arbitrary media objects.



1

6.1 audio

The audio media type covers both spoken voice and musical material. In this section we will discuss audio signal, stored in a raw or compressed (digital) format, as well as similarity-based retrieval for musical patterns.

In general, for providing search access to audio material we need, following Subrahmanian (1998), a data model that allows for both mdeta-data (that is information about the media object) and additional attributes of features, that we in prinicple obtain from the media object itself, using feature extraction.

audio data model

- *meta-data* – describing content
- *features* – using feature extraction

As an example of audi meta-data, consider the (meta-data) characterization that may be given for opera librettos.

example

singers – (Opera,Role,Person)
score – ...
transcript – ...

For signal-based audio content, we have to perform an analysis of the audio signal for which we may take parameters such as frequency, velocity and amplitude. For the actual analysis we may have to break up the signal in small windows, along the time-axis. Using feature extraction, we may characterize (signal-based) properties such as indicated below.

feature extraction

- *intensity* – watts/m^2
- *loudness* – in decibels
- *pitch* – from frequency and amplitude
- *brightness* – amount of distortion

For a more detailed treatment of signal-based audio content description, consult Subrahmanian (1998).

In the following we will first give an overview of musical search facilities on the web and then we will discuss similarity-based retrieval of musical patterns in somewhat more depth in the section on *research directions*, In section 6.3, we will have a closer look at feature extraction for arbitrary media types.

musical similarity

Although intuitively obvious, how can we characterize musical simularity? And perhaps more importantly, how can we compute the extent to which one piece of music or a melody line is similar to another piece of music or melody line. As concerns musical content, at least for most genres, it appears that

According to Selfridge (1998), we should focus primarily on *melody*, since

> *"It is melody that makes music memorable: we are likely to recall a tune long after we have forgotten its text."*

Other features, content-based as well as descriptive, may however be used as additional filters in the proces of retrieval.

Melodic searching and matching has been explored mainly in the context of bibliographic tools and for the analysis of (monophonic) repertories Hewlett and Selfridge-Field (1998). As described in section , many of these efforts have been made available to the general public through the Web. Challenges for the near future are, however, to provide for melodic similarity matching on polyphonic works, and retrieval over very large databases of musical fragments.

In this section we will look in somewhat more detail at the problem of melodic similarity matching. In particular, we will discuss representational issues, matching algorithms and additional analysis tools that may be used for musical information retrieval.

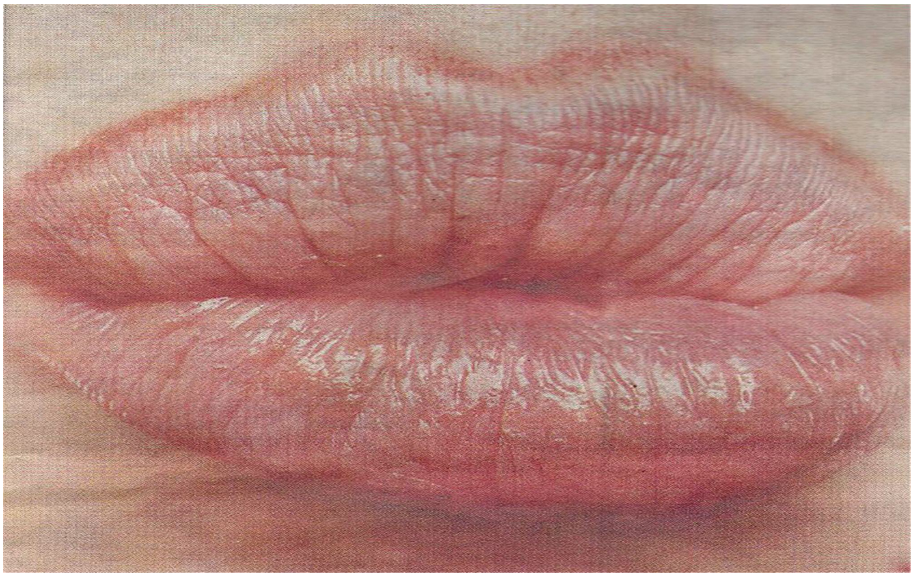
melodic similarity Consider the musical fragment *Twinkle, twinkle little star* (known in the Dutch tradition as *"Altijd is Kortjakje ziek"*), which has been used by Mozart for a series of variations Mozart (1787). Now, imagine how you would approach establishing the similarity between the original theme and these variations. As a matter of fact, we discovered that exactly this problem had been tackled in the study reported in Mongeau and Sankoff (1990), which we will discuss later. Before that, we may reflect on what we mean by the concept of a *melody*. In the aforementioned variations the original melody is disguised by, for example, decorations and accompaniments. In some variations, the melody is distributed among the various parts (the left and right hand). In other variations, the melody is only implied by the harmonic structure. Nevertheless, for the human ear there seems to be, as it is called in Selfridge (1998), a *'prototypical'* melody that is present in each of the variations.

When we restrict ourselves to pitch-based comparisons, melodic similarity may be established by comparing profiles of pitch-direction (up, down, repeat) or pitch contours (which may be depicted graphically). Also, given a suitable representation, we may compare pitch-event strings (assuming a normalized pitch representation such as position within a scale) or intervallic contours (which gives the distance between notes in for example semitones). Following Selfridge (1998), we may observe however that the more general the system of representation, the longer the (query) *string* will need to be to produce meaningful discriminations. As further discussed in Selfridge (1998), recent studies in musical perception indicate that pitch-information without durational values does not suffice.

representational issues Given a set of musical fragments, we may envisage several reductions to arrive at the (hypothetical) prototypical melody. Such reductions must provide for the elimination of confounds such as rests, repeated notes and grace notes, and result in, for example, a pitch-string (in a suitable representation), a duration profile, and (possibly) accented note profiles and harmonic reinforcement profiles (which capture notes that are emphasized by harmonic changes). Unfortunately, as observed in Selfridge (1998), the problem of which reductions to apply is rather elusive, since it depends to a great extent on the goals of the query and the repertory at hand.

As concerns the representation of pitch information, there is a choice between a base-7 representation, which corresponds with the position relative to the tonic in the major or minor scales, a base-12 representation, which corresponds with a division in twelve semitones as in the chromatic scale, and more elaborate encodings, which also reflect notational differences in identical notes that arise through the use of accidentals. For MIDI applications, a base-12 notation is most suitable, since the MIDI note information is given in semitone steps. In addition to relative pitch information, octave information is also important, to establish the rising and falling of melodic contour.

When we restrict ourselves to directional profiles (up, down, repeat), we may include information concerning the slope, or degree of change, the relation of the current pitch to the original pitch, possible repetitions, recurrence of pitches after intervening pitches, and possible segmentations in the melody. In addition, however, to support relevant comparisons it seems important to have information on the rhythmic and harmonic structure as well.



example(s) – *napster*

Wasn't it always your dream to have all your music free? Napster[4] was the answer. (But not for long.) Napster is, as we learn in the WikiPedia[5], *an online music service which was originally a file sharing service created by Shawn Fanning. Napster was the first widely-used peer-to-peer music sharing service, and it made a major impact on how people, especially college students, used the Internet. Its technology allowed music fans to easily share MP3 format song files with each other, thus leading to the music industry's accusations of massive copyright violations. The service was named Napster after Fanning's nickname.* However, Napster has been forced to become commercial. So the question is: is there life after napster? Well, there is at least open source!

research directions – *musical similarity matching*

An altogether different approach at establishing melodic similarity is proposed in Mongeau and Sankoff (1990). This approach has been followed in the Meldex system McNab et al. (1997), discussed in section . This is a rather technical section, that may be skipped on first reading. The approach is different in that it relies on a (computer science) theory of finite sequence comparison, instead of musical considerations. The general approach is, as explained in Mongeau and Sankoff (1990), to search for an optimal correspondence between elements of two sequences, based on a distance metric or measure of dissimilarity, also known more informally as the *edit-distance*, which amounts to the (minimal) number of transformations that need to be applied to the first sequence in order to obtain the second one. Typical transformations include *deletion*, *insertion* and *replacement*. In the musical domain, we may also apply transformations such as *consolidation* (the replacement of several elements by one element) and *fragmentation* (which is the reverse of consolidation). The metric is even more generally applicable by associating a weight with each of the transformations. Elements of the musical sequences used in Mongeau and Sankoff (1990) are pitch-duration pairs, encoded in base-12 pitch information and durations as multiples of 1/16th notes.

The matching algorithm can be summarized by the following recurrence relation for the dissimilarity metric. Given two sequences $A = a_1, \ldots, a_m$ and $B = b_1, \ldots, b_n$ and $d_{ij} = d(a_i, b_j)$, we define the distance as

$$d_{ij} = min \begin{cases} d_{i-1,j} + w(a_i, 0) & \text{deletion} \\ d_{i,j-1} + w(0, b_j) & \text{insertion} \\ d_{i-1,j-1} + w(a_i, b_j) & \text{replacement} \\ d_{i-k,j-1} + w(a_{i-k+1}, \ldots, a_i, b_j).\ 2 \leqslant k \leqslant i & \text{consolidation} \\ d\_i - 1, j - k + 1 + w(a\_i, b\_j - k + 1, \ldots b\_j)\ 2 <= k <= j & \text{fragmentation} \end{cases}$$

with

$$d_{i0} = d_{i-1,0} + w(a_i, 0),\ i \geqslant 1 \qquad \textit{deletion}$$

[4] www.napster.com

[5] en.wikipedia.org/wiki/Napster

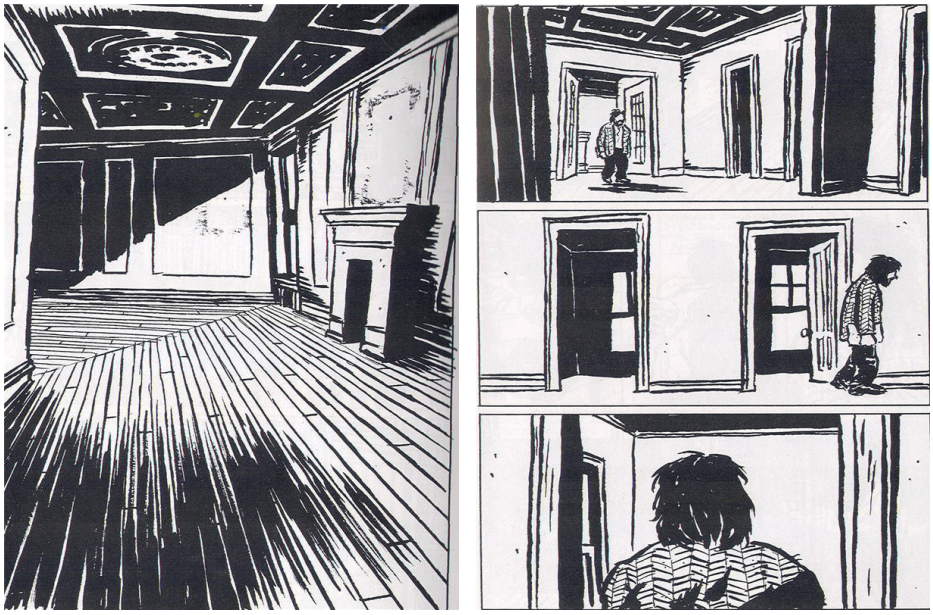
$$d_{0j} = d_{0,j-1} + w(0, b_i),\ j \geqslant 1 \qquad \textit{insertion}$$

and $d_{00} = 0$. The weigths $w(\_, \_)$ are determined by the degree of dissonance and the length of the notes involved.

The actual algorithms for determining the dissimilarity between two sequences uses dynamic programming techniques. The algorithm has been generalized to look for matching phrases, or subsequences, within a sequence. The complexity of the algorithm is $O(mn)$, provided that a limit is imposed on the number of notes involved in consolidation and fragmentation.

Nevertheless, as indicated in experiments for the Meldex database, the resulting complexity is still forbidding when large databases are involved. The Meldex system offers apart from the (approximate) dynamic programming algorithm also a state matching algorithm that is less flexible, but significantly faster. The Meldex experiments involved a database of 9400 songs, that were used to investigate six musical search criteria: (1) exact interval and rhythm, (2) exact contour and rhythm, (3) exact interval, (4) exact contour, (5) approximate interval and rhythm, and (6) approximate contour and rhythm. Their results indicate that the number of notes needed to return a reasonable number of songs scales logarithmically with database size McNab et al. (1997). It must be noted that the Meldex database contained a full (monophonic) transcription of the songs. An obvious solution to manage the complexity of searching over a large database would seem to be the storage of prototypical themes or melodies instead of complete songs.

indexing and analysis There are several tools available that may assist us in creating a proper index of musical information. One of these tools is the Humdrum system, which offers facilities for metric and harmonic analysis, that have proven their worth in several musicological investigations Huron (1997). Another tool that seems to be suitable for our purposes, moreover since it uses a simple pitch-duration, or *piano-roll*, encoding of musical material, is the system for metric and harmonic analysis described in Temperley and Sleator (1999). Their system derives a metrical structure, encoded as hierarchical levels of equally spaced beats, based on preference-rules which determine the overall likelihood of the resulting metrical structure. Harmonic analysis further results in (another level of) *chord spans* labelled with roots, which is also determined by preference rules that take into account the previously derived metrical structure. As we have observed before, metrical and harmonic analysis may be used to eliminate confounding information with regard to the 'prototypical' melodic structure.



4

6.2 video

Automatic content description is no doubt much harder for video than for any other media type. Given the current state of the art, it is not realistic to expect content description by feature extraction for video to be feasible. Therefore,to realize content-based search for video, we have rely on some knowledge representation schema that may adequately describe the (dynamic) properties of video fragments.

In fact, the description of video content may reflect the story-board, that after all is intended to capture both time-independent and dynamically changing properties of the objects (and persons) that play a role in the video.

In developing a suitable annotation for a particular video fragment, two questions need to be answered:

video annotation

- what are the interesting aspects?
- how do we represent this information?

Which aspects are of interest is something you have to decide for yourself. Let's see whether we can define a suitable knowledge representation scheme.

One possible knowledge representation scheme for annotating video content is proposed in Subrahmanian (1998). The scheme proposed has been inspired by knowledge representation techniques in Artificial Intelligence. It captures both static and dynamic properties.

video content

video v, frame f
f has associated objects and activities
objects and activities have properties

First of all, we must be able to talk about a particular video fragment v, and frame f that occurs in it. Each frame may contain objects that play a role in some activity. Both objects and activities may have properties, that is attributes that have some value.

property

property: name = value

As we will see in the examples, properties may also be characterized using predicates.

Some properties depend on the actual frame the object is in. Other properties (for example sex and age) are not likely to change and may considered to be frame-independent.

object schema

(fd,fi) – frame-dependent and frame-independent properties

Finally, in order to identify objects we need an object identifier for each object. Summing up, for each object in a video fragment we can define an *object instance*, that characterizes both frame-independent and frame-dependent properties of the object.

object instance: (oid,os,ip)

- *object-id* – oid
- *object-schema* – os = (fd,fi)
- *set of statements* – ip: name = v and name = v IN f

Now, with a collection of object instances we can characterize the contents of an entire video fragment, by identifying the frame-dependent and frame-independent properties of the objects.

Look at the following example, borrowed from Subrahmanian (1998) for the *Amsterdam Drugport* scenario.

frame	objects	*frame-dependent properties*
1	Jane	has(briefcase), at(path)
-	house	door(closed)
-	briefcase	
2	Jane	has(briefcase), at(door)
-	Dennis	at(door)
-	house	door(open)
-	briefcase	

In the first frame Jane is near the house, at the path that leads to the door. The door is closed. In the next frame, the door is open. Jane is at the door, holding a briefcase. Dennis is also at the door. What will happen next?

Observe thatwe are using predicates to represent the state of affairs. We do this, simply because the predicate form *has(briefcase)* looks more natural than the other form, which would be *has* = *briefcase*. There is no essential difference between the two forms.

Now, to complete our description we can simply list the frame-independent properties, as illustrated below.

object	*frame-independent properties*	value
Jane	age	35
	height	170cm
house	address	...
	color	brown
briefcase	color	black
	size	40 x 31

How to go from the tabular format to sets of statements that comprise the object schemas is left as an (easy) exercise for the student.

Let's go back to our *Amsterdam Drugport* scenario and see what this information might do for us, in finding possible suspects. Based on the information given in the example, we can determine that there is a person with a briefcase, and another person to which that briefcase may possibly be handed. Whether this is the case or not should be disclosed in frame 3. Now, what we are actually looking for is the possible exchange of a briefcase, which may indicate a drug transaction. So why not, following Subrahmanian (1998), introduce another somewhat more abstract level of description that deals with *activities*.

activity

- activity name – id
- statements – *role* = *v*

An activity has a name, and consists further simply of a set of statements describing the *roles* that take part in the activity.

example

{ giver : Person, receiver : Person, item : Object }
giver = Jane, receiver = Dennis, object = briefcase

For example, an *exchange* activity may be characterized by identifying the *giver*, *receiver* and *object* roles. So, instead of looking for persons and objects in a video fragment, you'd better look for activities that may have taken place, by finding a matching set of objects for the particular roles of an activity. Consult Subrahmanian (1998) if you are interested in a further formalization of these notions.

5

video libraries

Assuming a knowledge representation scheme as the one treated above, how can we support search over a collection of videos or video fragments in a video library.

What we are interested in may roughly be summarized as

video libraries

> which videos are in the library
> what constitutes the content of each video
> what is the location of a particular video

Take note that all the information about the videos or video fragments must be provided as meta-information by a (human) librarian. Just imagine for a moment how laborious and painstaking this must be, and whata relief video feature extraction would be for an operation like *Amsterdam Drugport*.

To query the collection of video fragments, we need a query language with access to our knowledge representation. It must support a variety of retrieval operations, including the retrieval of segments, objects and activities, and also property-based retrievals as indicated below.

query language for video libraries

- *segment retrievals* – *exchange of briefcase*
- *object retrievals* – all people in v:[s,e]
- *activity retrieval* – all activities in v:[s,e]
- *property-based* – find all videos with object oid

Subrahmanian (1998) lists a collection of video functions that may be used to extend SQL into what we may call VideoSQL. Abstractly, VideoSQL may be characterized by the following schema:

VideoSQL

SELECT – v:[s,e]
FROM – video:<source><V>
WHERE – term IN funcall

where *v:[s,e]* denotes the fragment of video *v*, starting at frame *s* and ending at frame *e*, and *term IN funcall* one of the video functions giving access to the information about that particular video. As an example, look at the following VideoSQL snippet:

example

```
SELECT  vid:[s,e]
FROM video:VidLib
WHERE (vid,s,e) IN VideoWithObject(Dennis) AND
      object IN ObjectsInVideo(vid,s,e) AND
      object != Dennis AND
      typeof(object) = Person
```

Notice that apart from calling video functions also constraints can be added with respect to the identity and type of the objects involved.

6

example(s) – *video retrieval evaluation*

The goal of the TREC[6] conference series is to encourage research in information retrieval by providing a large test collection, uniform scoring procedures, and a forum for organizations interested in comparing their results. Since 2003 their is an independent *video* track devoted to research in automatic segmentation,

[6] trec.nist.gov

indexing, and content-based retrieval of digital video. In the TRECVID[7] 2004 workshop, thirty-three teams from Europe, the Americas, Asia, and Australia participated. Check it out!

7

research directions– *presentation and context*

Let's consider an example. Suppose you have a database with (video) fragments of news and documentary items. How would you give access to that database? And, how would you present its contents? Naturally, to answer the first question, you need to provide search facilities. Now, with regard to the second question, for a small database, of say 100 items, you could present a list of videos thatb matches the query. But with a database of over 10.000 items this will become problematic, not to speak about databases with over a million of video fragments. For large databases, obviously, you need some way of visualizing the results, so that the user can quickly browse through the candidate set(s) of items.

Pesce (2003) provide an interesting account on how *interactive maps* may be used to improve search and discovery in a (digital) video library. As they explain in the abstract:

> *To improve library access, the Informedia Digital Video Library uses automatic processing to derive descriptors for video. A new extension to the video processing extracts geographic references from these descriptors.*
>
> *The operational library interface shows the geographic entities addressed in a story, highlighting the regions discussed in the video through a map display synchronized with the video display.*

[7] www-nlpir.nist.gov/projects/trecvid

So, the idea is to use geographical information (that is somehow available in the video fragments themselves) as an additional descriptor, and to use that information to enhance the presentation of a particular video. For presenting the results of a query, candidate items may be displayed as icons in a particular region on a map, so that the user can make a choice.

Obviously, having such geographical information:

> *The map can also serve as a query mechanism, allowing users to search the terabyte library for stories taking place in a selected area of interest.*

The approach to extracting descriptors for video fragments is interesting in itself. The two primary sources of information are, respectively, the spoken text and graphic text overlays (which are common in news items to emphasize particular aspects of the news, such as the area where an accident occurs). Both speech recognition and image processing are needed to extract information terms, and in addition natural language processing, to do the actual 'geocoding', that is translating this information to geographical locations related to the story in the video.

Leaving technical details aside, it will be evident that this approach works since news items may relevantly be grouped and accessed from a geographical perspective. For this type of information we may search, in other words, with three kinds of questions:

- *what* – content-related
- *when* – position on time-continuum
- *where* – geographic location

and we may, evidently, use the geographic location both as a search criterium and to enhance the presentation of query results.

mapping information spaces Now, can we generalize this approach to other type of items as well. More specifically, can we use maps or some spatial layout to display the results of a query in a meaningful way and so give better access to large databases of multimedia objects. According to Dodge and Kitchin (2002), we are very likely able to do so:

> *More recently, it has been recognized that the process of spatialization – where a spatial map-like structure is applied to data where no inherent or obvious one does exist – can provide an interpretable structure to other types of data.*

Actually, we are taking up the theme of *visualization*, again. In Dodge and Kitchin (2002) visualizations are presented that (together) may be regarded as an *atlas of cyberspace*.

atlas of cyberspace

> *We present a wide range of spatializations that have employed a variety of graphical techniques and visual metaphors so as to provide striking and powerful images that extend from two dimension 'maps' to three-dimensional immersive landscapes.*

As you may gather from chapter 7 and the *afterthoughts*, I take a personal interest in the (research) theme of *virtual reality interfaces for multimedia information systems*. But I am well aware of the difficulties involved. It is an area that is just beginning to be explored!

8

6.3 feature extraction

Manual content annotation is laborious, and hence costly. As a consequence, content annotation will often not be done and search access to multimedia object willnot be optimal, if it is provided for at all. An alternative to manual content annotation is (semi) automatic feature extraction, which allows for obtaining a description of a particular media object using media specific analysis techniques.

The Multimedia Database Research group at CWI has developed a framework for feature extraction to support the *Amsterdam Catalogue of Images* (ACOI). The resulting framework for feature extraction is known as the ACOI framework, Kersten et al. (1998).

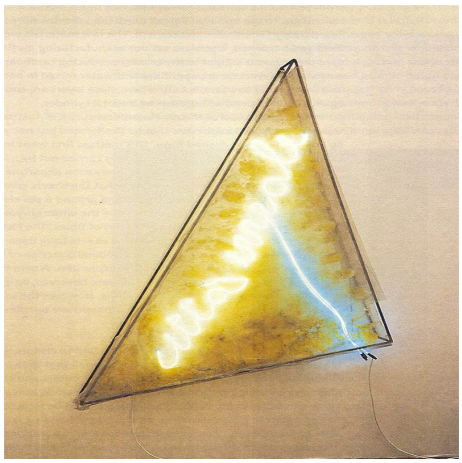
The ACOI framework is intended to accomodate a broad spectrum of classification schemes, manual as well as (semi) automatic, for the indexing and retrieval of arbitrary multimedia objects. What is stored are not the actual multimedia objects themselves, but structural descriptions of these objects (including their location) that may be used for retrieval.

The ACOI model is based on the assumption that indexing an arbitrary multimedia object is equivalent to deriving a grammatical structure that provides a namespace to reason about the object and to access its components. However there is an important difference with ordinary parsing in that the lexical and grammatical items corresponding to the components of the multimedia object must be created dynamically by inspecting the actual object. Moreover, in general, there is not a fixed sequence of lexicals as in the case of natural or formal languages. To allow for the dynamic creation of lexical and grammatical items the ACOI framework supports both *black-box* and *white-box* (feature) detectors. Black-box detectors are algorithms, usually developed by a specialist in the media domain,

that extract properties from the media object by some form of analysis. White-box detectors, on the other hand, are created by defining logical or mathematical expressions over the grammar itself. Here we will focus on black-box detectors only.

The information obtained from parsing a multimedia object is stored in a database. The feature grammar and its associated detector further result in updating the data schemas stored in the database.

formal specification Formally, a feature grammar G may be defined as $G = (V, T, P, S)$, where V is a collection of variables or non-terminals, T a collection of terminals, P a collection of productions of the form $V \rightarrow (V \cup T)$ and S a start symbol. A token sequence ts belongs to the language $L(G)$ if $S \xrightarrow{*} ts$. Sentential token sequences, those belonging to $L(G)$ or its sublanguages $L(G_v) = (V_v, T_v, P_v, v)$ for $v \in (T \cup V)$, correspond to a complex object C_v, which is the object corresponding to the parse tree for v. The parse tree defines a hierarchical structure that may be used to access and manipulate the components of the multimedia object subjected to the detector. See Schmidt et al. (1999) for further details.



9

anatomy of a feature detector

As an example of a feature detector, we will look at a simple feature detector for (MIDI encoded) musical data. A special feature of this particular detector,that I developed while being a guest at CWI, is that it uses an intermediate representation in a logic programming language (Prolog) to facilitate reasoning about features.

The hierarchical information structure that we consider is defined in the grammar below. It contains only a limited number of basic properties and must be

extended with information along the lines of some musical ontology, see Zimmerman (1998).

feature grammar

```
detector song; # # to get the filename
detector lyrics; # # extracts lyrics
detector melody; # # extracts melody
detector check;  # # to walk the tree

atom str name;
atom str text;
atom str note;

midi: song;

song: file lyrics melody check;

file: name;

lyrics: text*;
melody: note*;
```

The start symbol is a *song*. The detector that is associated with *song* reads in a MIDI file. The musical information contained in the MIDI file is then stored as a collection of Prolog facts. This translation is very direct. In effect the MIDI file header information is stored, and events are recorded as facts, as illustrated below for a *note_on* and *note_off* event.

```
event('twinkle',2,time=384, note_on:[chan=2,pitch=72,vol=111]).
event('twinkle',2,time=768, note_off:[chan=2,pitch=72,vol=100]).
```

After translating the MIDI file into a Prolog format, the other detectors will be invoked, that is the *composer*, *lyrics* and *melody* detector, to extract the information related to these properties.

To extract relevant fragments of the melody we use the melody detector, of which a partial listing is given below.

melody detector

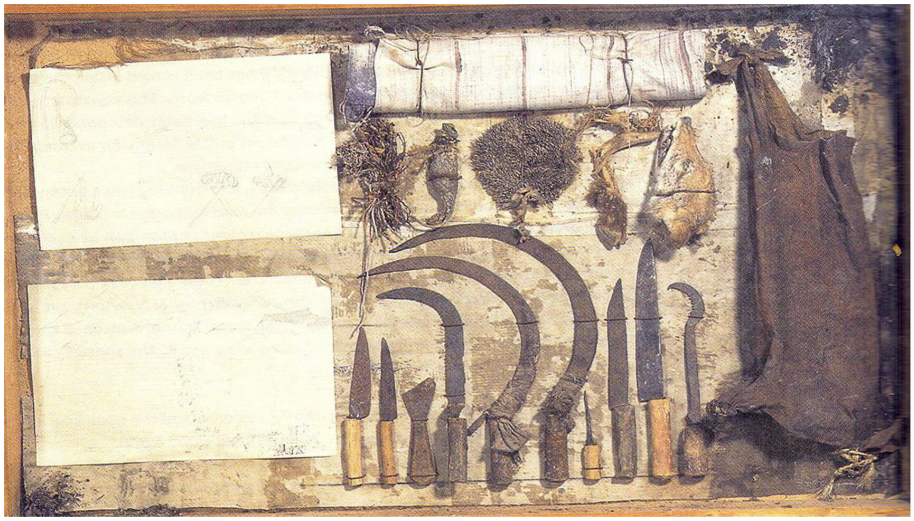
```
int melodyDetector(tree *pt, list *tks ){
char buf[1024]; char* _result;
void* q = _query;
int idq = 0;

  idq = query_eval(q,"X:melody(X)");
  while ((_result = query_result(q,idq)) ) {
        putAtom(tks,"note",_result);
        }
```

```
    return SUCCESS;
}
```

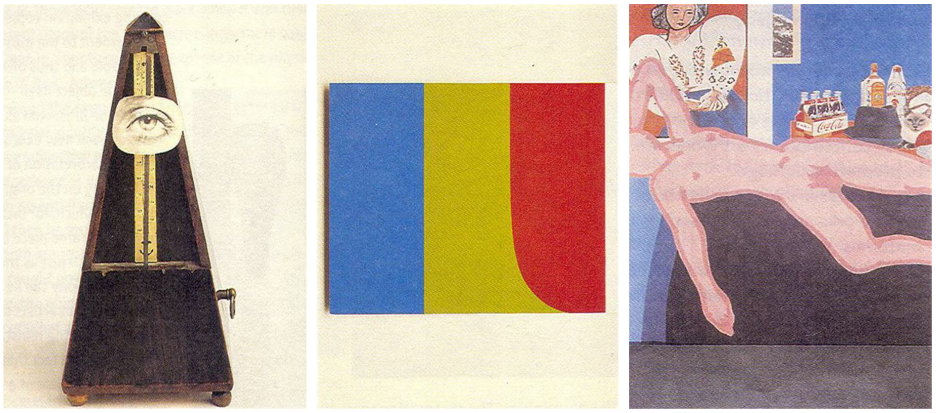
The embedded logic component is given the query `X:melody(X)`, which results in the notes that constitute the (relevant fragment of the) melody. These notes are then added to the tokenstream. A similar detector is available for the lyrics.

Parsing a given MIDI file, for example *twinkle.mid*, results in updating the database.



10

implementation The embedded logic component is part of the *hush* framework, Eliens (2000). It uses an object extension of Prolog that allows for the definition of native objects to interface with the MIDI processing software written in C++. The logic component allows for the definition of arbitrary predicates to extract the musical information, such as the melody and the lyrics. It also allows for further analysis of these features to check for, for example, particular patterns in the melody.



11

example(s) – *modern art: who cares?*

The artworks shown above are taken from Hummelen and Sillé (1999), which bundles the experiences and insights resulting from studying the preservation of contemprary art, under the title: *modern art, who cares?* This project was a precursor to the INCCA[8] that provided the input to our *multimedia casus*, which is introduced in chapter 10.

Both the INCCA project and the related *Open Archives Initiative*[9], focus on making meta-information available on existing resources for the preservation of contemporary art and cultural heritage in general, including reports, case studies and recordings of artworks, that is images, videos and artists interviews.

12

research directions– *media search*

There is a wealth of powerful search engines on the Web. Technically, search engines rely either on classification schemes (as for example Yahoo) or content-based (keyword) indexing (as for example Excite or AltaVista). Searching on the Web, nowadays, is moderately effective when text-based documents are considered. For multimedia objects (such as images or music) existing search facilities are far less effective, simply because indexing on category or keywords can not so easily be done automatically. In the following we will explore what search facilities there are for music (on the web). We will first give some examples of search based on keywords and categories, then some examples of content-based search and finally we will discuss a more exhaustive list of musical databases and search facilities on the Web. All search facilities mentioned are listed online under *musical resources*.

[8] www.incca.org
[9] www.openarchives.org

keywords and categories For musical material, in particular MIDI, there are a number of sites that offer search over a body of collected works. One example is the Aria Database, that allows to search for an aria part of an opera based on title, category and even voice part. Another example is the MIDI Farm, which provides many MIDI-related resources, and also allows for searching for MIDI material by filename, author, artist and ratings. A category can be selected to limit the search. The MIDI Farm employs voting to achieve collaborative filtering on the results for a query. Search indexes for sites based on categories and keywords are usually created by hand, sometimes erreonously. For example, when searching for a Twinkle fragment, Bach's variations for Twinkle were found, whereas to the best of our knowledge there exist only Twinkle variations by Mozart Mozart (1787). The Digital Tradition Folksong Database provides in addition a powerful lyrics (free text) search facility based on the AskSam search engine. An alternative way of searching is to employ a meta-search engine. Meta-search engines assist the user in formulating an appropriate query, while leaving the actual search to (possibly multiple) search engines. Searching for musical content is generally restricted to the lyrics, but see below (and section).

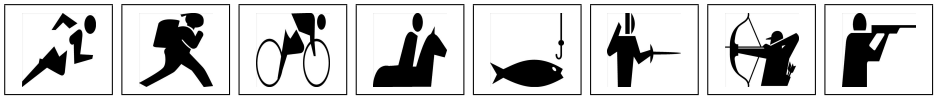
content-based search Although content-based search for images and sound have been a topic of interest for over a decade, few results have been made available to the public. As an example, the MuscleFish Datablade for Informix, allows for obtaining information from audio based on a content analysis of the audio object. As far as content-based musical search facilities for the Web are concerned, we have for example, the Meldex system of the New Zealand Digital Library initiative, an experimental system that allows for searching tunes in a folksong database with approximately 1000 records, McNab et al. (1997). Querying facilities for Meldex include queries based on transcriptions from audio input, that is humming a tune! We will discuss the approach taken for the Meldex system in more detail in research directions section, to assess its viability for retrieving musical fragments in a large database.

music databases In addition to the sites previously mentioned, there exist several databases with musical information on the Web. We observe that these databases do not rely on DBMS technology at all. This obviously leads to a plethora of file formats and re-invention of typical DBMS facilities. Without aiming for completeness, we have for example the *MIDI Universe*, which offers over a million MIDI file references, indexed primarily by composer and file length. It moreover keeps relevant statistics on popular tunes, as well as a hot set of MIDI tunes. It further offers access to a list of related smaller MIDI databases. Another example is the aforementioned Meldex system that offers a large collection of tunes (more than 100.000), of which a part is accessible by humming-based retrieval. In addition text-based search is possible against file names, song titles, track names and (where available) lyrics. The Classical MIDI Archive is an example of a database allowing text-based search on titles only. Results are annotated with an indication of "goodness" and recency. The Classical Themefinder Database allows extensive support for retrieval based on (optional) indications of meter,

pitch, pitch-class, interval, semi-tone interval and melodic contour, within a fixed collection of works arranged according to composer and category. The index is clearly created and maintained manually. The resulting work is delivered in the MuseData format, which is a rich (research-based) file format from which MIDI files can be generated, Selfridge (1997). A site which collects librarian information concerning music resources is the International Inventory of Music Resources (RISM), which offers search facilities over bibliographic records for music manuscripts, librettos and secondary sources for music written after c.a. 1600. It also allows to search for libraries related to the RISM site. Tune recognition is apparently offered by the Tune Server. The user may search by offering a WAV file with a fragment of the melody. However, the actual matching occurs against a melodic outline, that is indications of rising or falling in pitch. The database contains approx. 15.000 records with such pitch contours, of which one third are popular tunes and the rest classical themes. The output is a ranked list of titles about which the user is asked to give feedback.

discussion There is great divergence in the scope and aims of music databases on the Web. Some, such as the RISM database, are the result of musicological investigations, whereas others, such as the MIDI Farm, are meant to serve an audience looking for popular tunes. With regard to the actual search facilities offered, we observe that, with the exception of Meldex and the Tune Server, the query facilities are usually text-based, although for example the Classical Themefinder allows for encoding melodic contour in a text-based fashion.

6.4 development(s) – expert recommendations



13

questions

content annotation

1. (*) How can video information be made accessible? Discuss the requirements for supporting video queries.

concepts

2. What are the ingredients of an *audio data model*
3. What information must be stored to enable search for video content?
4. What is *feature extraction*? Indicate how feature extraction can be deployed for arbitrary media formats.

technology

5. What are the parameters for *signal-based (audio) content*?
6. Give an example of the representation of *frame-dependent* en *frame-independent* properties of a video fragment.
7. What are the elements of a query language for searching in video libraries?
8. Give an example (with explanation) of the use of *VideoSQL*.

projects & further reading As a project, think of implementing musical similarity matching, or developing an application retrieving video fragments using a simple annotation logic.

You may further explore the construction of media repositories, and finding a balance between automatic indexing, content search and meta information.

For further reading I advice you to *google* recent research on video analysis, and the online material on search engines[10].

the artwork

1. works from Weishar (1998)
2. faces – from www.alterfin.org, an interesting site with many surprising interactive toys in *flash*, javascript and html.
3. mouth – Annika Karlson Rixon, entitled *A slight Acquaintance*, taken from a theme article about the body in art and science, the Volkskrant, 24/03/05.
4. story – page from the comic book version of *City of Glass*, Auster (2004), drawn in an almost tradional style.
5. story – frame from Auster (2004).
6. story – frame from Auster (2004).
7. story – frame from Auster (2004).
8. *white on white* – typographical joke.
9. modern art – *city of light* (1968-69), Mario Merz, taken from Hummelen and Sillé (1999).
10. modern art – *Marocco* (1972), Krijn Griezen, taken from Hummelen and Sillé (1999).
11. modern art – *Indestructable Object* (1958), Man Ray, *Blue, Green, Red I* (1964-65), Ellsworth Kelly, *Great American Nude* (1960), T. Wesselman, taken from Hummelen and Sillé (1999).
12. signs – sports, van Rooijen (2003), p. 272, 273.

Opening this chapter are examples of design of the 20th century, posters to announce a public event like a theatre play, a world fair, or a festival. In comparison to the art works of the previous chapter, these designs are more strongly *expressive* and more simple and clear in their *message*. Yet, they also show a wide variety of styles and rethorics to attract the attention of the audience. Both the faces and the mouth are examples of using body parts in contemporary

[10] www.searchtools.com/tools/tools-opensource.html

art. The page of the comic book version of *City of Glass*, illustrates how the 'logic' of a story can be visualised. As an exercise, try to annoyaye the sequence of frames from the *City of Glass* can be described using the annotation logic you learned in this chapter. The modern art examples should interesting by themselves.

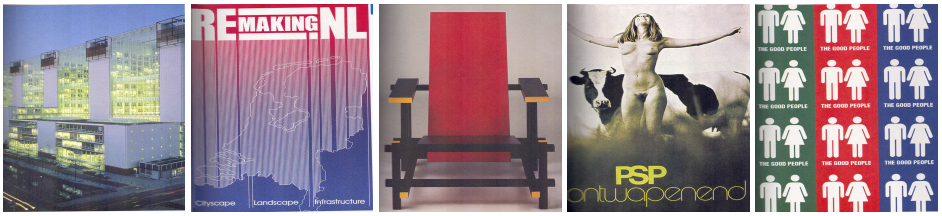
7. information system architecture

effective retrieval requires visual interfaces

learning objectives

After reading this chapter you should be able to dicuss the considerations that play a role in developing a multimedia information system, characterize an abstract multimedia data format, give examples of multimedia content queries, define the notion of virtual resources, and discuss the requirements for networked virtual environments.

From a system development perspective, a multimedia information system may be considered as a multimedia database, providing storage and retrieval facilities for media objects. Yet, rather than a solution this presents us with a problem, since there are many options to provide such storage facilities and equally many to support retrieval. In this chapter, we will study the architectural issues involved in developing multimedia information systems, and we will introduce the notion of media abstraction to provide for a uniform approach to arbitrary media objects. Finally, we will discuss the additional problems that networked multimedia confront us with.



1

7.1 architectural issues

The notion of *multimedia information system* is sufficiently generic to allow for a variety of realizations. Let's have a look at the issues involved.

As concerns the database (that is the storage and rerieval facilities), we may have to deal with homegrown solution, commercial third party databases or (even) legacy sources. To make things worse, we will usually want to deploy a combination of these.

With respect to the information architecture, we may wish for a common format (which unifies the various media types), but in practice we will often have to work with the native formats or be satisfied with a hybrid information architecture that uses both media abstractions and native media types such as images and video.

The notion of media abstraction, introduced in Subrahmanian (1998), allows for uniform indexes over the multimedia information stored, and (as we will discuss in the next section) for query relaxation by employing hierarchical and equivalence relations.

Summarizing, for content organisation (which basically is the information architecture) we have the following options:

content organisation

- *autonomy* – index per media type
- *uniformity* – unified index
- *hybrid* – media indexes + unified index

In Subrahmanian (1998), a clear preference is stated for a uniform approach, as expressed in the *Principle of Uniformity*:

Principle of Uniformity

> *... from a semantical point of view the* content *of a multimedia source is independent of the source itself, so we may use statements as meta data to provide a description of media objects.*

Naturally, there are some tradeoffs. In summary, Subrahmanian (1998) claims that: metadata can be stored using standard relational and OO structures, and that manipulating metadata is easy, and moreover that feature extraction is straightforward. </lp> Now consider, is feature extraction really so straightforward as suggested here? I would believe not. Certainly, media types can be processed and analysis algorithms can be executed. But will this result in meaningful annotations? Given the current state of the art, hardly so!

research directions– *the information retrieval cycle*

When considering an information system, we may proceed from a simple generic software architecture, consisting of:

software architecture

- a database of media object, supporting
- operations on media objects, and offering
- logical views on media objects

However, such a database-centered notion of information system seems not to do justice to the actual support and information system must provide when considering the full information retrieval cycle:

information retrieval cycle

1. specification of the user's information need
2. translation into query operations
3. search and retrieval of media objects
4. ranking according to likelihood or relevance
5. presentation of results and user feedback
6. resulting in a possibly modified query

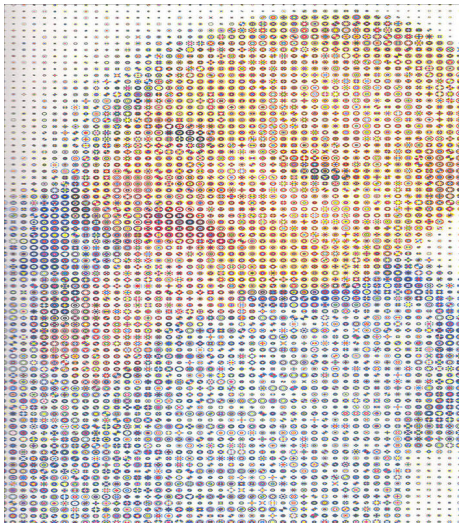
When we look at older day information retrieval applications in libraries, we see more or less the automation of card catalogs, with search functionality for keywords and headings. Modern day versions of these systems, however, offer graphical userinterfaces, electronic forms and hypertext features.

When we look at the web and how it may support digital libraries, we see some dramatic changes with respect to the card catalogue type of applications. We can now have access to a variety of sources of information, at low cost, including geographically distributed resources, due to improved networking. And, everybody is free to make information available, and what is worse, everybody seems to be doing so. Hence, the web is a continuously growing repository of information of a (very) heterogeneous kind.

Considering the web as an information retrieval system we may observe, following Baeza-Yates and Ribeiro-Neto (1999), that:

- despite high interactivity, access is difficult;
- quick response is and will remain important!

So, we need better (user-centered) retrieval strategies to support the full information retrieval cycle. Let me (again) mention someof the relevant (research) topics:*user interfaces, information visualisation, user-profiling and navigation.*



7.2 media abstractions

Let's have a closer look at media abstractions. How can we capture the characterization of a variety of media types in one common media abstraction. A definition of such a media abstraction is proposedin Subrahmanian (1998). Leaving the formal details aside, a media abstraction has the following components:

media abstraction

- *state* – smallest chunk of media data
- *feature* – any object in a state
- *attributes* – characteristics of objects
- *feature extraction map* – to identify content
- relations – to capture state-dependent information
- (inter)relations between 'states' or chunks

Now, that characterization is sufficiently abstract, and you may wonder how on earth to apply this to an actual media database.

However, before giving some examples, we must note that the *feature extraction map* does not need to provide information about the content of a chunk of media data automatically. It may well be a hand-coded annotation.

Our first example is an image database.

example – image database

states: { pic1.gif,...,picn.gif }
features: names of people
extraction: find people in pictures
relations: left-of, ...

In an image database it does not make much sense to speak about relations between 'states' or chunks of media data, that is the images.

For our next example though, video databases, it does make sense to speak about such relations, since it allows us to talk about scenes as sequences of frames.

example – video database

states: set of frames
features: persons and objects
extraction: gives features per frame
relations: frame-dependent and frame-independent information
inter-state relation: specifies sequences of frames

Now, with this definition of media abstractions, we can define a simple multimedia database, simply as

simple multimedia database

- a finite set M of media abstractions

But, following Subrahmanian (1998), we can do better than that. In order to deal with the problems of *synonymy* and *inheritance*, we can define a structured multimedia database that supports:

structured multimedia database

- *equivalence relations* –to deal with synonymy
- *partial ordering* – to deal with inheritance
- *query relaxation* – to please the user

Recall that we have discussed the relation between a 'house of prayer' and 'church' as an example of synonymy in section 4.3. As an example of inheritance we may think of the relation between 'church' and 'cathedral'. Naturally, every cathedral is a church. But the reverse does not necessarily hold. Having this information about possible equivalence and inheritance relationships, we can relax queries in order to obtain better results. For example, when a user asks for cathedral in a particular region, we could even notify the user of the fact that although there are no cathedrals there, there are a number of churches that may be of interest. (For a mathemathical characterization of structured multimedia databases, study Subrahmanian (1998).)

3

query languages Having media abstractions, what would a query language for such a database look like? Again, following Subrahmanian (1998), wemay extend SQL with special functions as indicated below:

SMDS – functions

Type: object $\mapsto$ type
ObjectWithFeatures: $f \mapsto \{o \mid \text{object } o \text{ contains } f\}$
ObjectWithFeaturesAndAttributes: $(f, a, v) \mapsto \{o \mid o \text{ contains } f \text{ with } a = v\}$
FeaturesInObject: $o \mapsto \{f \mid o \text{ contains } f\}$
FeaturesAndAttributesInObject: $o \mapsto \{(f, a, v) \mid o \text{ contains } f \text{ with } a = v\}$

Having such functionswe can characterize an extension of SQL, which has been dubbed SMDS-SQL in Subrahmanian (1998), as follows.

SMDS-SQL

SELECT – media entities

- m – if m is not a continuous media object

- $m : [i, j]$ – m is continuous, i, j integers (segments)
- $m.a$ – m is media entity, a is attribute

FROM

- <media><source><M>

WHERE

- term IN funcall

As an example, look at the following SMDS-SQL snippet.

example

```
SELECT M
FROM   smds source1 M
WHERE  Type(M) = Image AND
       M IN ObjectWithFeature("Dennis") AND
       M IN ObjectWithFeature("Jane") AND
       left("Jane","Dennis",M)
```

Note that M is a relation in the image database media abstraction, which contains one or more images that depict Jane to the left of Dennis. Now, did they exchange the briefcase, or did they not?

When we do not have a uniform representation, but a hybrid representation for our multimedia data instead, we need to be able to: express queries in specialized language, and to perform operations (joins) between SMDS and non-SMDS data.

Our variant of SQL, dubbed HM-SQL, differs from SMDS-SQL in two respects: function calls are annotated with media source, and queries to non-SMDS data may be embedded.

As a final example, look at the following snippet:

example HM-SQL

```
SELECT M
FROM smds video1, videodb video2
WHERE M IN smds:ObjectWithFeature("Dennis") AND
    M IN videodb:VideoWithObject("Dennis")
```

In this example, we are collecting all video fragments with Dennis in it, irrespective of where that fragment comes from, an (smds) database or another (video) database.

research directions– *digital libraries*

Where *media abstractions*, as discussed above, aremeant to be technical abstractions needed for uniform access to media items, we need quite a different set of abstraction to cope with one of the major applications of multimedia information storage and retrieval: digital libraries.

According to Baeza-Yates and Ribeiro-Neto (1999), digital libraries will need a long time to evolve, not only because there are many technical hurdles to be

overcome, but also because effective digital libraries are dependent on an active community of users:

digital libraries

> *Digital libraries are constructed – collected and organized – by a community of users. Their functional capabilities support the information needs and users of this community. Digital libraries are an extension, enhancement and integration of a variety of information institutions as physicalplaces where resources are selected, collected, organized, preserved and accessed in support of a user community.*

The occurrence of digital libraries on the web is partly a response to advances in technology, and partly due to an increased appreciation of the facilities the internet can provide. From a development perspective, digital libraries may be regarded as:

> *... federated structures that provide humans both intellectual and physical access to the huge and growing worldwide networks of information encoded in multimedia digital formats.*

Early research in digital libraries has focussed on the digitization of existing material, for the preservation of our cultural heritage, as well as on architectural issues for the 'electronic preservation', so to speak, of digital libraries themselves, to make them "immune to degradation and technological obsolescence", Baeza-Yates and Ribeiro-Neto (1999).

To bring order in the variety of research issues related to digital libraries, Baeza-Yates and Ribeiro-Neto (1999) introduces a set of abstractions that is know as the 5S model:

digital libraries (5S)

- *streams*: (content) – from text to multimedia content
- *structures*: (data) – from database to hypertext networks
- *spaces*: (information) – from vector space to virtual reality
- *scenarios*: (procedures) – from service to stories
- *societies*: (stakeholders) – from authors to libraries

These abstractions act as "a framework for providing theoretical and practical unification of digital libraries". More concretely, observe that the framework encompasses three technical notions (streams, structures and spaces; which correspond more or less with data, content and information) and two notions related to the social context of digital libraries (scenarios and societies; which range over possible uses and users, respectively).

For further research you may look at the following resources:

D-Lib Forum – http://*www.dlib.org*
Informedia – http://*www.informedia.cs.cmu.edu*

The D-Lib Forum site gives access to a variety of resources, including a magazine with background articles as well as a test-suite that may help you in developing digital library technology. The Informedia site provides an example of a digital library project, with research on, among others, video content analysis, summarization and in-context result presentation.

4

7.3 networked multimedia

For the end user there should not be much difference between a stand-alone media presentation and a networked media presentation. But what goes on *behind the scenes* will be totally different. In this section, we will study, or rather have a glance at, the issues that play a role in realizing effective multimedia presentations. These issues concern the management of resources by the underlying network infrastructure, but may also concern authoring to the extent that the choice of which media objects to present may affect the demands on resources.

To begin, let's try to establish, following Fluckiger (1995), in what sense networked multimedia applications might differ from other network applications:

networked multimedia

- real-time transmission of continuous media information (audio, video)
- substantial volumes of data (despite compression)
- distribution-oriented – e.g. audio/video broadcast

Naturally, the extent to which network resource demands are made depends heavily on the application at hand. But as an example, you might think of the retransmission of television news items on demand, as nowadays provided via both cable and DSL.

For any network to satisfy such demands, a number of criteria must be met, that may be summarized as: throughput, in terms of bitrates and burstiness; transmission delay, including signal propagation time; delay variation, also known as jitter; and error rate, that is data alteration and loss.

For a detailed discussion of criteria, consult Fluckiger (1995), or any other book on networks and distributed systems. With respect to distribution-oriented multimedia, that is audio and video broadcasts, two additional criteria play a role,

in particular: multicasting and broadcasting capabilities and document caching. Especially caching strategies are of utmost importance if large volumes of data need to be (re)transmitted.

Now, how do we guarantee that our (networked) multimedia presentations will come across with the right quality, that is free of annoying jitter, without loss or distortion, without long periods of waiting. For this, the somewhat magical notion of *Quality of Service* has been invented. Quoting Fluckiger (1995):

Quality of Service

> Quality of Service *is a concept based on the statement that not all applications need the same performance from the network over which they run. Thus, applications may indicate their specific requirements to the network, before they actually start transmitting information data.*

Quality of Service (*QoS*) is one of these notions that gets delegated to the other parties, all the time. For example, in the MPEG-4 standard proposal interfaces are provided to determine *QoS* parameters, but the actual realization of it is left to the network providers. According to Fluckiger (1995) it is not entirely clear how *QoS* requirements should be interpreted. We have the following options: we might consider them as hard requirements, or alternatively as guidance for optimizing internal resources, or even more simply as criteria for the acceptance of a request.

At present, one thing is certain. The current web does not offer *Quality of Service*. And what is worse, presentation formats (such as for example *flash*) do not cope well with the variability of resources. More specifically, you may get quite different results when you switch to another display platform

5

virtual objects

Ideally, it should not make any difference to the author at what display platform a presentation is viewed, nor should the author have to worry about low-quality or ill-functioning networks. In practice, however, it seems not to be realistic to hide all this variability from the author and delegate it entirely to the 'lower layers' as in the MPEG-4 proposal.

Both in the SMIL and RM3D standards, provisions are made for the author to provide a range of options from which one will be chosen, dependent on for example availability, platform characteristics, and network capabilities.

A formal characterization of such an approach is given in Subrahmanian (1998), by defining *virtual objects*.

virtual objects

- $VO = \{(O_i, Q_i, C_i) \mid 1 \leqslant i \leqslant k\}$

where

- $C_1, \ldots, C_k$ – mutually exclusive conditions
- $Q_1, \ldots, Q_k$ – queries
- $O_1, \ldots, O_k$ – objects

In general, a virtual object is a media object that consists of multiple objects, that may be obtained by executing a query, having mutually exclusive conditions to determine which object will be selected. Actually, the requirement that the conditions are mutually exclusive is overly strict. A more pragmatic approach would be to regard the objects as an ordered sequence, from which the first eligible one will be chosen, that is provided that its associated conditions are satisfied.

As an example, you may look at the Universal Media proposal from the Web3D Consortium, that allows for providing multiple URNs or URLs, of which the first one that is available is chosen. In this way, for instance, a texture may be loaded from the local hard disk, or if it is not available there from some site that replicates the Universal Media textures.

6

networked virtual environments

It does seem to be an exageration to declare *networked virtual environments* to be the ultimate challenge for networked multimedia, considering that such environments may contain all types of (streaming) media, including video and 3D graphics, in addition to rich interaction facilities. (if you have no idea what I am talking about, just think of, for example, Quake or DOOM, and read on.) To be somewhat more precise, we may list a number of essential characteristics of networked virtual environments, taken from Singhal and Zyda (1999):

networked virtual environments

- *shared sense of space* – room, building, terrain
- *shared sense of presence* – avatar (body and motion)
- *shared sense of time* – real-time interaction and behavior

In addition, networked virtual environments offer

- *a way to communicate* – by gesture, voice or text
- *a way to share ...* – interaction through objects

Dependent on the visual realism, resolution and interaction modes such an environment may be more or less 'immersive'. In a truly immersive environment, for example one with a haptic interface and force feedback, interaction through objects may become even threathening. In desktop VEs, sharing may be limited to the shoot-em-up type of interaction, that is in effect the exchange of bullets.

Networked virtual environments have a relatively long history. An early example is SIMNET (dating from 1984), a distributed command and control simulation developed for the US Department of Defense, Singhal and Zyda (1999). Although commercial multi-user virtual communities, such as the *blaxxun* Community server, may also be ranked under networked virtual environments, the volume of data exchange needed for maintaining an up-to-date state is far less for those environments than for game-like simulation environments from the military tradition. Consider, as an example, a command and control strategy game which contains a variety of vehicles, each of which must send out a so-called *Protocol Data Unit* (PDU), to update the other participants as to their actual location and speed. When the delivery of PDUs is delayed (due to for example geographic dispersion, the number of participants, and the size of the PDU), other strategies, such as *dead reckoning*, must be used to perform collision detection and determine possible hits.

To conclude, let's establish what challenges networked virtual environments offers with respect to software design and network performance.

challenges

- *network bandwidth* – limited resource
- *heterogeneity* – multiple platforms
- *distributed interaction* – network delays
- *resource management* – real-time interaction and shared objects
- *failure management* – stop, ..., degradation
- *scalability* – wrt. number of participants

Now it would be too easy to delegate this all back to the network provider. Simply requiring more bandwidth would not solve the scalability problem and even though adding bandwidth might allow for adding another hundred of entities, smart updates and caching is probably needed to cope with large numbers of participants.

The distinguishing feature of networked virtual environments, in this respect, is the need to

manage dynamic shared state

to allow for real-time interaction between the participants. Failing to do so would result in poor performance which would cause immersion, if present at all, to be lost immediately.

7

example(s) – *unreal*

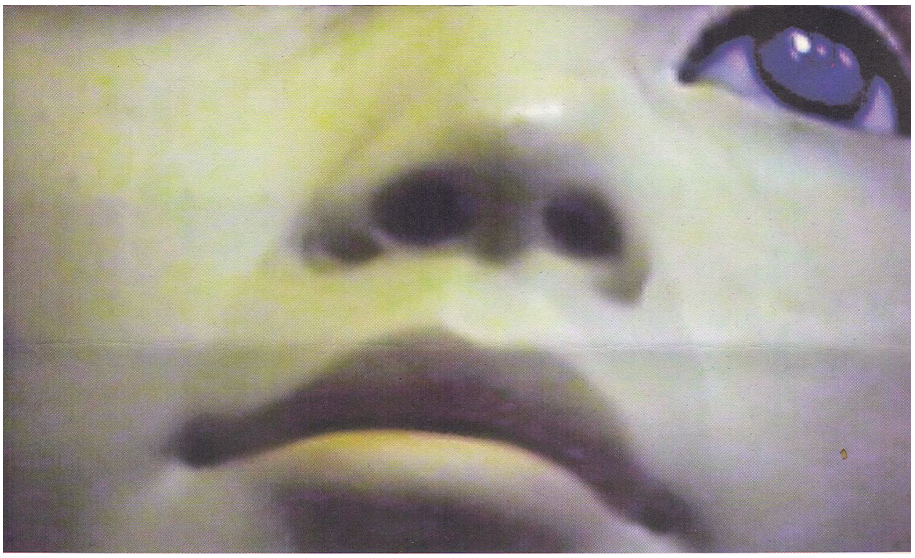
Unreal Tournament[11] is a highly popular multiplayer game. The storyline is simple, but effective: *It's the year 2362. The most anticipated Tournament ever is about to take place, dwarfing the spectacle and drama of previous events. The finest competitors ever assembled prepare to lay waste to their opponents and claim the Tournament Trophy for themselves.*

There a a number of roles you can associate with:

> the corrupt, thunder crash, iron guard, juggernauts, iron skull, sun blade, super nova, black legion, fire storm, hellions, bloof fist, goliath

An interesting feature of the Unreal Tournament games is that they can be andapted and even be re-programmed[12] by the users themselves, has has been done for example for the *Mission Rehearsal Exercise* discussed in section 9.2.

scripting: www.gamedev.net/reference/list.asp?categoryid=76



[11] www.unrealtournament.com
[12] www.unrealtournament.com/ut2004/screenshots.php

research directions– *architectural patterns*

Facing the task of developing a multimedia information system, there are many options. Currently, the web seems to be the dominant infrastructure upon which to build a multimedia system. Now, assuming that we chose the web as our vehicle, how should we approach building such a system or, in other words, what architectural patterns can we deploy to build an actual multimedia information system? As you undoubtly know, the web is a document system that makes a clear distinction between *servers* that deliver documents and *clients* that display documents. See Eliens (2000), section 12.1. At the server-side you are free to do almost anything, as long as the document is delivered in the proper format. At the client-side, we have a generic document viewer that is suitable for HTML with images and sound. Dependent on the actual browser, a number of other formats may be allowed. However, in general, extensions with additional formats are realized by so-called *plugins* that are loaded by the browser to enable a particular format, such as *shockwave*, *flash* or *VRML*. Nowadays, there is an overwhelming number of formats including, apart from the formats mentioned, audio and video formats as well as a number of XML-based formats as for example SMIL and SVG. For each of these formats the user (client) has to download a plugin. An alternative to plugins (at the client-side) is provided by Java *applets*. For Java applets the user does not need to download any code, since the Java platform takes care of downloading the necessary classes. However, since applets may be of arbitrary complexity, downloading the classes needed by an application may take prohibitively long.

The actual situation at the client-side may be even more complex. In many cases a media format does not only require a plugin, but also an applet. The plugin and applet can communicate with eachother through a mechanism (introduced by Netscape under the name LiveConnect) which allows for exchanging messages using the built-in DOM (Document Object Model) of the browser. In addition, the plugin and applet may be controlled through Javascript (or VBscript). A little dazzling at first perhaps, but usually not to difficult to deal with in practice.

Despite the fact that the web provides a general infrasructure for both (multimedia) servers and clients, it might be worthwhile to explore other options, at the client-side as well as the server-side. In the following, we will look briefly at:

- the Java Media Framework, and
- the DLP+X3D platform

as examples of, respectively, a framework for creating dedicated multimedia applications at the client-side and a framework for developing intelligent multimedia systems, with client-side (rich media 3D) components as well as additional server-side (agent) components.

Java Media Framework The Java platform offers rich means to create (distributed) systems. Also included are powerful GUI libraries (in particular, Swing),

3D libraries (Java3D) and libraries that allow the use and manipulation of images, audio and video (the Java Media Framework). Or, in the words of the SUN web site:

java Media Framework[13]

> *The JavaTM Media APIs meet the increasing demand for multimedia in the enterprise by providing a unified, non-proprietary, platform-neutral solution. This set of APIs supports the integration of audio and video clips, animated presentations, 2D fonts, graphics, and images, as well as speech input/output and 3D models. By providing standard players and integrating these supporting technologies, the Java Media APIs enable developers to produce and distribute compelling, media-rich content.*

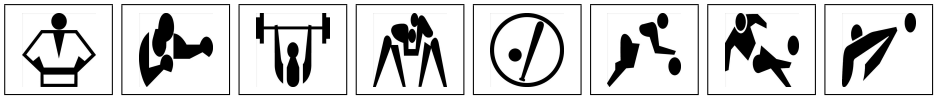
However, although Java was once introduced as the *dial tone of the Internet* (see Eliens (2000), section 6.3), due to security restrictions on applets it is not always possible to deploy media-rich applets, without taking recourse to the Java plugin to circumvent these restrictions.

DLP+X3D In our DLP+X3D platform, that is introduced in section **??** and described in more detail in appendix **??**, we adopted a different approach by assuming the availability of a generic X3D/VRML plugin with a Java-based External Authoring Interface (EAI). In addition, we deploy a high-level ditributed logic programming language (DLP) to control the content and behavior of the plugin. Moreover, DLP may also be used for creating dedicated (intelligent) servers to allow for multi-user applications.

The DLP language is Java-based and is loaded using an applet. (The DLP jar file is of medium size, about 800 K, and does not require the download of any additional code.) Dua, again, to the security restrictions on applets, additional DLP servers must reside on the site from where the applet was downloaded.

Our plugin, which is currently the *blaxxun* VRML plugin, allows for incorporating a fairly large number of rich media formats, including (real) audio and (real) video., thus allowing for an integrated presentation environment where rich media can be displayed in 3D space in a unified manner. A disadvantage of such a unified presentation format, however, is that additional authoring effort is required to realize the integration of the various formats.

7.4 development(s) – clients versus servers



[13] java.sun.com/products/java-media/jmf/reference/api

questions

information system architecture

1. (*) What are the issues in designing a *(multimedia) information system architecture*. Discuss the tradeoffs involved.

concepts

2. What considerations would you have when designing an architecture for a multimedia information system.
3. Characterize the notion of *media abstraction*.
4. What are the issues in *networked multimedia*.

technology

5. Describe (the structure of) a video database, using *media abstractions*.
6. Give a definition of the notion of a *structured multimedia database*.
7. Give an example (with explanation) of querying a *hybrid multimedia database*.
8. Define (and explain) the notion of *virtual objects* in *networked multimedia*.

projects & further reading As a project, you may implement a multi-player game in which you may exchange pictures and videos, for example pictures and videos of celebrities.

Further you may explore the development of a data format for text, images and video with appropriate presentation parameters, including postioning on the screen and intermediate transitions.

For further reading you may study information system architecture patterns[14], nd explore the technical issues of constructing server based advanced multimedia applications in Li and Drew (2004).

the artwork

1. examples of dutch design, from Betsky (2004).
2. idem.
3. sceenshots – from *splinter cell: chaos theory*, taken from Veronica/Gammo[15], a television program about games.
4. screenshots – respectively *Sekken 5*, *Sims 2*, and *Super Monkey Ball*, taken from insidegamer.nl[16].
5. screenshots – from Unreal Tournament[17], see section 7.3.
6. idem.
7. idem.
8. *resonance* – exhibition and performances, Montevideo[18], april 2005.

[14] www.opengroup.org/architecture/togaf8-doc/arch/p4/patterns/patterns.htm
[15] www.gammo.nl
[16] http://www.insidegamer.nl
[17] www.unrealtournament.com/ut2004/screenshots.php
[18] www.montevideo.nl

9. signs – sports, van Rooijen (2003), p. 274, 275.

Opening this chapter are examples of *dutch design*, taken from the book *False Flat*, with the somewhat arrogant subtitle *why is dutch design so good?*. It is often noted that dutch design is original, functional and free from false traditionalism. Well, judge for yourself.

The screenshots from the various games are included as a preparation for chapter 9, where we discuss realism and immersion in games, and also because multiplayer games like *Unreal Tournament* have all the functionality a serious application would ever need.