Habilitationsschrift

Visual Information Retrieval

eingereicht an der

Technischen Universität Wien
Fakultät für Informatik

von

Dr. Horst Eidenberger

Erdbergstrasse 103/2/12a, 1030 Wien

Wien, am 20. September 2004

Table of Contents

Introduction: Visual Information Retrieval

This section gives a general introduction to the field of Visual Information Retrieval (VIR). The first subsection defines the scope of research and discusses examples for research questions. Then, potential application areas are described and problem areas are sketched. In the following subsection, VIR is structured by identifying and listing the major research areas. Borders to neighbouring research disciplines are drawn by a brief discussion of the techniques employed. Finally, a short outlook on likely future developments is given and the author's research contributions, i.e. the papers collected in this thesis, are fit in the sketched categorisation of research areas.

What is Visual Information Retrieval?

Despite hundreds of research papers per year (see [21] for publication statistics of recent years) and detailed surveys of the field (e.g. [42, 17, 69, 58, 74, 52]), hardly any definitions of the term "Visual Information Retrieval" do exist. This may have to do with the distinction of Content-based Image Retrieval (CBIR) and Content-based Video Retrieval (CBVR) that has been made in the early years of VIR (approximately early 1990s). Though applying mostly the same models and algorithms (apart from media data extraction and media visualisation), systems have either been intended for CBIR or CBVR application purposes. Developments starting in the mid 1990s (especially, the work on the visual part of the MPEG-7 standard [57, 15, 56]) have led to the fusion of CBIR and CBVR in Visual Information Retrieval.

For our purpose we define Visual Information Retrieval as *all computer-based methods that try to approximate the similarity of visually perceivable media objects exclusively from their visual content*. This definition contains several elements: Firstly, it limits VIR to computer-based methods. This is unavoidable, since – as we will see in the discussion of applied methods below – VIR methods can only be applied on media objects that consist of digitised, uniform, computable media chunks. Visually perceivable media are mainly image and video. The distinction of visual media into temporal and non-temporal originates from the different methods (and levels of sophistication) required for manipulation in the early years of VIR. Nowadays, almost any CBIR technology can also be applied on video content, and vice versa. Of course, for some media-related methods this interchangeability would not make sense (e.g. application of high resolution image analysis on video frames, or application of time-dependent algorithms on images). Thirdly, the definition defines similarity measurement as the major focus of VIR. VIR is neither pattern recognition nor computer vision: The purpose is not to match or recognise *selected* elements of media content. Quite the contrary, VIR aims at judging the *general* similarity of media objects. Furthermore, following this definition similarity measurement should exclusively be based on the visual content of media objects. Text metadata (e.g. media-inserted tags, annotations) and other media types (e.g. audio streams of video clips) are not of primary concern. Although, they might increase precision in various practical applications.

This definition of Visual Information Retrieval opens the door for a wide range of applications. Before we discuss some of the most important application areas in the next subsection, we briefly describe in more detail similarity perception and similarity measurement as the central topics of VIR. VIR methods intend to tackle a large number of similarity-related problems. Maybe the most basic question is, whether or not two images are similar. Figure 1 illustrates a very simple example. Are these images similar? From a biological (content-aware) point of view they are not: A blue hare is more similar to a seal than to many other species, but they are not exactly related. Images of a brown European hare and a blue hare would be more similar. On the other hand, both animals are mammals, have a white fur and are positioned in a similar environment, a wintry scene. Even more radical, from an abstract (content-unaware) point of view one could argue that something almost elliptical bright white is located in the middle of something dark white. From this point of view, the images would be highly similar.

This difference in similarity assessment comes from different levels of perception, for example caused by a different level of expertise or a different focus of judgement. For our purpose it is interesting to notice that almost any existing VIR system would judge these images as highly similar, because their content is similar in colours, textures and shapes of foreground objects.

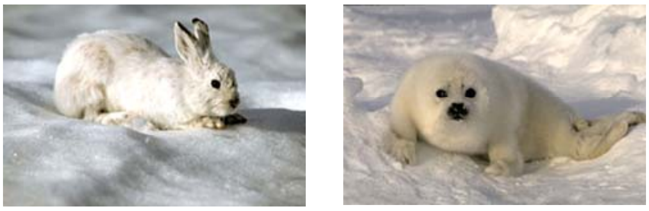


Figure 1: Are these images similar?

Figure 2 shows a second example. Obviously, both images depict flowers, but flowers of different kinds, different colours and numbers. Still, many users would judge the images as similar, because they show just the flowers and no other objects or scene elements. For a VIR system, judging the flower images as similar is almost impossible, because they use different colours, textures and contours. (Moreover, recognition from low-level features of the image content as flowers is – without additional knowledge – hardly possible.)

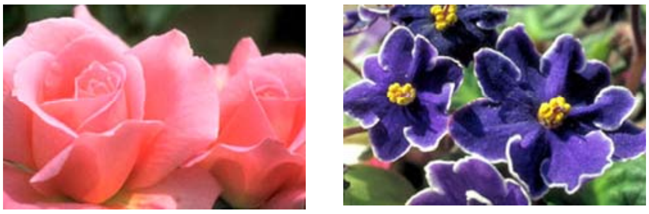


Figure 2: Are these images similar?

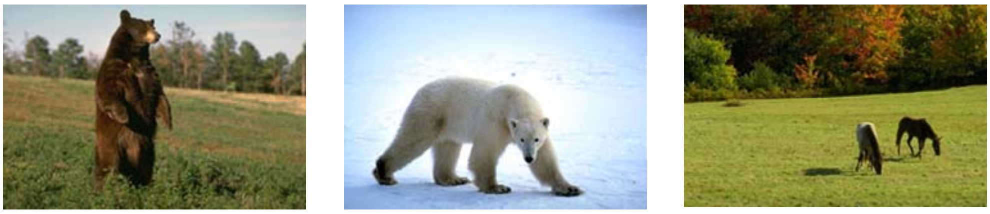


Figure 3: Is the second or the third image more similar to the first?

Figure 3 gives an example for a second type of question. While the first type of question is mostly relevant for recognition tasks, this question is important for the selection of those media objects from a collection that fit a given example. Again, following a "biological argumentation" the second image is more similar to the first image. On the other hand, the scene shown in the background of the first image is more similar to the scene in the third image. If a VIR system is used to decide this question, the decision depends on whether the system is able to recognise the shapes of the two bears (if shape properties are taken into account in similarity comparison). Most current VIR systems would select the second candidate as more similar.

A third type of question is illustrated in Figure 4. Here, the first two images are used to define an *idea* of content that can be described as "sunset in the desert". The question is, whether the third or the fourth image fits better to this idea. The third image shows a cactus in a desert. The fourth image shows sunset, but obviously, the foreground object is not a cactus. Again, it is almost impossible to select one image, since both can be argued for. A human selector would ask for further information ("Do you want sunsets or deserts?") to solve the problem. Most VIR systems would select the fourth image due to similar colours and similar shapes of the foreground objects.

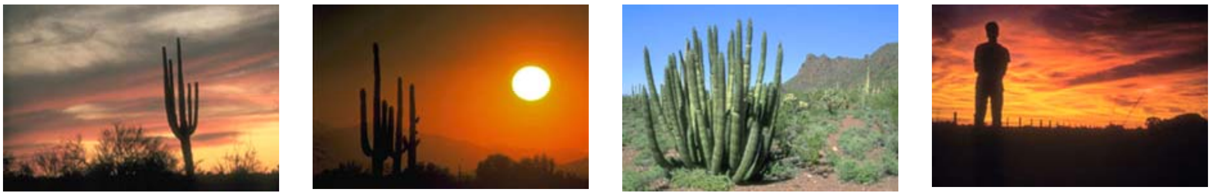


Figure 4: Which image fits better to the first two: the third or the fourth?

These examples should provide the understanding that the imitation of human visual similarity perception is a non-trivial task. VIR is a research discipline that has to deal with a considerable amount of ambiguity. In consequence, results can almost always be argued for or against. This thesis discusses selected approaches to overcome the problem of ambiguity and to reduce the gap between human and machine perception. Final remark: Only image examples were given in this subsection (due to the usage of paper). Of course, the same types of questions could be asked for video objects/collections.

Application areas

The digital imaging revolution has lead to an enormous increase in size and numbers of image and video collections. Amateur photography is just one aspect of this development. Today, imaging is used for a wide range of applications in industry, medical services, libraries, etc. Moreover, most created visual media are archived in databases for reuse or later analysis. The growth of media databases goes hand in hand with an increasing demand for tools for media organisation and retrieval. Since photographing is much easier to perform than image analysis and annotation (additionally, human annotation is a faulty and high-cost process), tools for the retrieval of media objects with similar content to given examples are highly desirable.

Since the beginning of VIR, systems have been developed for VIR applications in *digital libraries*. For example, the classic IBM QBIC system [39, 2] has been used to query collections of classic stamps [44] and collections of paintings [63]. Today, e.g., the Network of Excellence DELOS explores applications for VIR in digital libraries and develops new methods for general-purpose content-based retrieval [18].

Medical databases are a second classic application area of VIR [47, 80]. VIR methods are, for example, used to retrieve X-ray images from databases (e.g. [54]). Thus, VIR can support the diagnosis process by identifying images showing anomalies that are similar to those of a particular patient. Here, the major strength of VIR over other technologies is that it does not make any assumptions on the characteristics of image features.

VIR is especially suitable for the *recognition of trademark images* [19, 76]. If registration is requested for a new trademark, it has to be verified that it is sufficiently unsimilar to existing trademarks. Since today, hundreds of thousands of trademarks are registered worldwide, this is an almost ideal application for VIR systems. Moreover, it is often required to give a measure for the unsimilarity of trademark images. As we will see below, any VIR system fulfils this requirement by expressing similarity in numerical terms.

Face recognition is another application domain for VIR. Today, video cameras are widely used: for identification, for surveillance of public spaces, etc. For online use and exploitation of archived material, VIR technologies are often used for face recognition (e.g. [16, 78]). For example, the visual part of the MPEG-7 standard provides a set of content-based descriptions of faces [1]. Using these descriptions, faces should be reliably distinguishable.

In recent years, several proposals have been made for VIR systems for *news video analysis* (e.g. [77, 55, 5]). Today, news video analysis is (in numbers of sold systems) the most successful application area of VIR. Temporal segmentation is used to identify shot boundaries of anchor person shots and news item clips. Text recognition from video is employed to extract headlines and context information. Furthermore, face recognition is used for speaker identification. The extracted metadata is fed into classification algorithms in order to organise news video clips in predefined groups.

Inspection of metal surfaces is a new field of industrial application of VIR [45]. Since human-based quality assurance has been replaced by computers and cameras, pattern recognition is used to identify defects in metal surfaces. If defects vary widely in shape and size, then VIR is – because of the generality of the approach – a more error-robust alternative.

Finally, in prior work the author has developed a VIR system for retrieval of *coats of arms images* [11, 8]. Identification of coats of arms is a service scientific libraries offer to historians but also to interested private citizens. Identifying coats of arms (often from seal prints) helps to find out, when documents where written, by whom and in which historic context. Without a VIR system, a human expert has to go through ten thousands of images to identify the bearer of particular coats of arms.

Problem areas

The essential problem of Visual Information Retrieval is measuring visually perceivable similarity. This induces firstly, that properties of visual media have to be recognised by the VIR system and secondly, that the similarity of stimuli present/absent in two media objects has to be computed. In [21] we have laid down that human perception is based on three types of stimuli: generally perceived stimuli (also called low-level features, e.g. colour distributions), specifically perceived stimuli (recognised elements, e.g. foreground objects) and pseudo-random stimuli (e.g. perception habits related to upbringing and culture). Obviously, at most only the first two types of stimuli can be handled in VIR systems. Unfortunately, usually (e.g. for most types of media content), only generally perceived stimuli are available as basis for the similarity measurement process. Image understanding and object recognition are mostly beyond the capabilities of state of the art VIR algorithms.

This shortcoming is the major reason why general-purpose VIR systems are rarely utilised today. Humans have high-level concepts in mind (at least specifically perceived stimuli, but due to lacking awareness of personal/cultural particularities most times also pseudo-random stimuli). The VIR system, on the other hand, does similarity measurement based on low-level features. Apparently, in many cases such results must be unsatisfactory for human users. The discrepancy of high-level concepts and low-level features is regarded as the *semantic gap* [67, 17]. Figures 5 and 6 illustrate the semantic gap in two examples. Figure 5 shows two images that are rated as highly similar by most VIR systems, even though their content is completely different (a parade in a stadium and flowers between rocks). The reason is simply, that both images contain structures with highly similar colours, textures and (because of the textures) hardly recognisable objects.

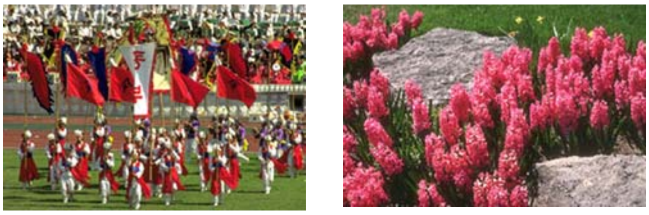


Figure 5: Semantic gap: Two images that are usually rated as highly similar by VIR systems.

The images in Figure 6 are rated as highly unsimilar by most VIR systems. Again, the reason is that VIR methods fail in recognising that both images show wildcats. Since the first one shows a lynx in snow (light background) while the second shows a tiger in a wood (dark background), they cannot be retrieved as biologically similar images. Moreover, the tiger is – because of its natural camouflage – hardly recognisable at all.

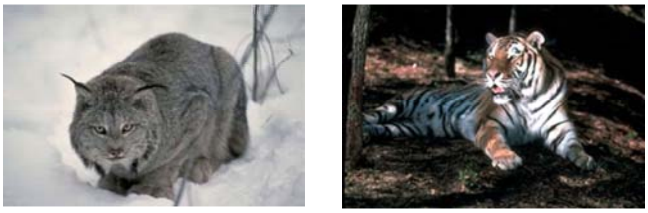


Figure 6: Semantic gap: Two images that are usually rated as not similar by VIR systems. Are they similar?

The semantic gap is mainly responsible for the rare successful utilisation of VIR. Still, a second problem plays an essential role in situations where specifically perceived stimuli can be identified. *Polysemy* denotes the phenomenon that one image can express various meanings. In fact, a single image can tell entire stories. Figure 7 illustrates examples for different levels of polysemy. The first image contains various meanings: Firstly, it shows runners running a *race*. From the background we can see that people are watching this race. The shirts of the runners inform us about the fact, that they run for different countries. So, *championship* is another meaning of the image. Further meanings may by *Marathon*, *Olympic Games*, etc. The second image contains less polysemy (mountain hike, sunny day, Alps, etc.) but still more than the third (flowers).

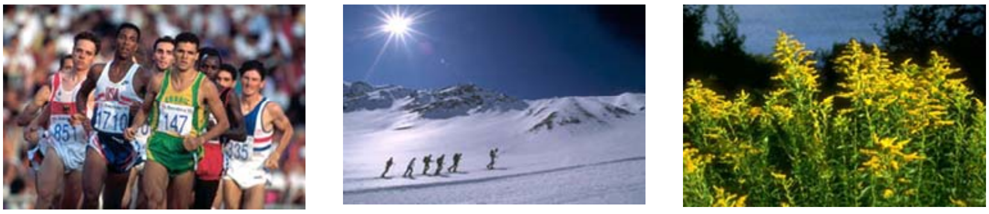


Figure 7: Examples for polysemy. First image: high polysemy, second image: medium polysemy, third image: low polysemy.

The problem of dealing with polysemy is to identify which interpretation is correct in a particular retrieval situation. To complete this task successfully, additional information on the semantic background of the user's query would be required. Such information is – in terms of computer-understandable models – hardly expressible. Hence, polysemy is an almost unsolvable problem. Practically, querying modules of VIR systems are constructed with enough degrees of freedom for the user to express his particular view by refining queries into particular directions.

A third, yet unsolved problem is *modelling of similarity perception*. Traditionally, the development of mathematical and computational models for the imitation of human (visual) similarity perception is a field of psychological research (e.g. [3, 51]). For example, in [71, 72] the authors falsify the commonly used model for visual similarity perception: The metric model assumes that properties of media objects (stimuli) are described by number vectors. These number vectors are interpreted as points in a vector space of Euclidean geometry. Since the metric axioms hold for such spaces, distance functions (e.g. Minkowski distances) can be utilised for measurement of unsimilarity. In the mentioned papers the authors show clearly that none of the metric axioms hold for human visual similarity perception. Instead, they propose a model based on predicate-like stimuli. This model has also been adapted for usage in VIR [67, 66]. Unfortunately, due to serious shortcomings in the model, it was not practically applicable [35]. In [23], we propose a similarity measurement model that allows the application of almost any similarity measure in VIR. Still, the similarity measure that would judge similarity as humans do, has not yet been identified.

A last problem of VIR research that is practically highly relevant is *querying performance*. Producing fast results is a challenge for every system that has to deal with large datasets, especially retrieval systems. In VIR, an additional degree of complexity is that (often, very sophisticated and computationally complex) functions are used to judge the similarity of media objects. In the worst case, if a ranked list of media objects is required, these functions have to be applied to every pair of media objects. Already being a problem of second order complexity, usage of complex similarity measurement functions may lead to unbearably slow replies. To overcome this problem, various indexing structures, heuristics, etc. have been proposed. Quite often, the problem is postponed for the sake of research efforts to narrow the semantic gap. Still, for practical applications, bad performance is a serious problem.

In the next subsection we sketch the components required to build a VIR system and technologies used to build these components. Especially, we indicate approaches to overcome the mentioned problem areas.

Areas of research and employed technologies

Overview

Every VIR systems consists of two major components: a component for *media access and the extraction of media descriptions* (features), and a component for *retrieval and similarity measurement*. The development of novel technologies and improvement of existing approaches employed in these two areas are the two major focal points of VIR research. Furthermore, *evaluation* of feature extraction and retrieval methods, *system design* considerations (improvement of performance, usability, etc.) and development of *application domain-specific solutions* are other major research areas.

Figure 8 illustrates the major elements of the feature extraction and retrieval components as well as their relationships. For media access, a middleware is required, because access of image and video data require different software paradigms. Feature extraction algorithms extract media descriptions (stimuli, e.g. generally perceived stimuli) from the media objects and store them in a media description database. The user interacts with the VIR system through appropriate user interfaces. The user interfaces make use of the media access API to visualise media content. Retrieval operations are executed by a query engine that is based on retrieval models and procedures for the iterative refinement by the user's relevance feedback (e.g. kernel-based learning [59]).

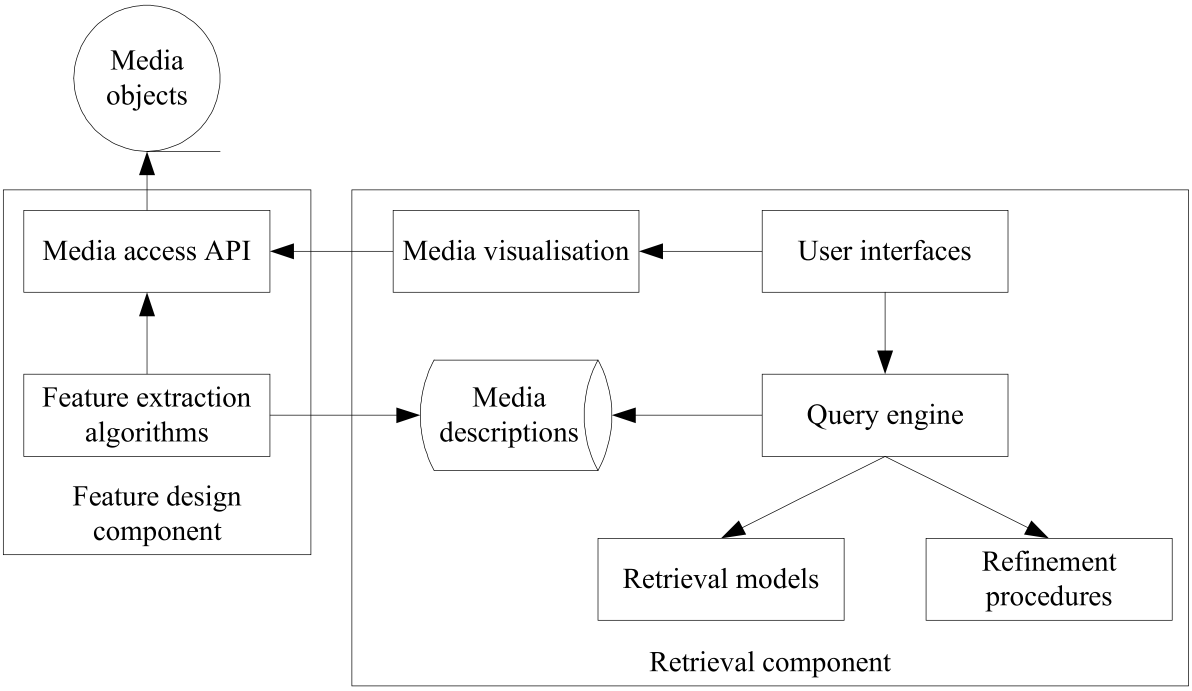


Figure 8: VIR system components.

In the next subsections, we sketch the scientific scope of the first four mentioned areas of research. Moreover, we give a general introduction to the major technologies used. In this introduction, we concentrate on the big picture. Technically more detailed descriptions and discussions are given in the first paper of this thesis [21].

Media access and feature design

Purpose of this research area is the development of algorithms for the extraction of meaningful descriptions of visual media objects. Traditionally, four groups of feature extraction algorithms (often, just "features") are distinguished: colour features, texture features, shape features and motion features (e.g. in [17, 58, 42]). Colour features extract colour properties (histograms of colour usage, dominant colours, etc.). Texture features try to characterise the textures occurring in images and video frames (regularity, direction, coarseness, etc.). Classic texture description methods are space to frequency transformations (e.g. Fourier transformation, Cosine transformation). Today, wavelets are used for texture extraction and description [73, 64, 56]. For VIR, especially Gabor wavelets have proven to imitate human texture perception adequately [65, 67]. Both, colour and texture features extract generally perceived media properties.

Shape features aim at extracting objects from pixel data. Segmented objects are either described by their border (contour features) or by their area (region features, e.g. moments). For shape extraction, computer vision techniques are used (e.g. edge detection, split and merge, region growing, Markov Random Fields, Hough transformation, etc. [70]). Generally, the goal for shape feature application is the extraction of specifically perceived properties. This requires an object recognition step after object description. Classifiers and graph-based methods are used for object recognition (e.g. Bayesian classifiers, Hidden Markov Models, etc. [7]). Unfortunately, object segmentation and object recognition are ill-posed problems. Therefore, the results for applications not narrowed to specific application domains are often unsatisfactory.

The motion features commonly used in VIR are a heterogeneous set of methods for temporal description. Temporal segmentation based on frame differences is used to detect shot boundaries (e.g. [17, 43]). Frame difference maps (optical flow) are used to detect likely camera motions (e.g. in the MPEG-7 standard [46, 57]). Segmented objects are tracked through video clips to detect their motion trajectories. Eventually, the overall motion in a video clip can be described by summarising the motion vectors of compressed video data.

It is obviously not to the advantage of VIR results that most features extract generally perceived properties. Today, additional sources of information and methods for semantic enrichment are used to narrow the semantic gap. For example, relevance feedback data and kernel-based learning methods are used to distinguish groups of media objects, information on the media content and classifiers are employed to cluster media objects, etc. In the next section we investigate, how similarity measurement grounded on these low-level and high-level features can be performed.

Similarity modelling and retrieval processes

VIR systems are often distinguished by their querying paradigm. Frequently used methods are querying by example, querying by sketch and querying by a group of examples. Though being important for the user, for the actual retrieval process the selected querying paradigm makes no difference. Media comparison is always based on descriptions. If not available beforehand, descriptions have to be extracted during the querying process (e.g. of sketches).

In most VIR systems, one of two retrieval processes is employed: retrieval based on the *vector space model*, or retrieval based on *probabilistic inference*. The vector space model assumes that media descriptions are points in a vector space and that this vector space has a geometry (mostly, Euclidean geometry) [41]. Then, unsimilarity of media objects can be measured as distance of media descriptions. The vector space model is successfully used in text information retrieval. Unfortunately, applied to VIR, two problems arise: Firstly, it is not clear what type of geometry (distance function) fits to human similarity perception (as pointed out above). Secondly often, differently extracted media descriptions require different distance measures. Selection of features for retrieval and usage of multiple distance measures are non-trivial, still open research problems (see, for example, [32, 35]).

Probabilistic inference models use media descriptions and a priori probabilities (computed from statistics, e.g. human relevance information) to compute differentiated a posteriori probabilities that can be used for retrieval [61, 41]. Employed models are mostly based on Bayesian networks. The major advantage of probabilistic inference over the vector space model is that it avoids the problem of explicitly defining similarity measures. The main disadvantages are that random sample data is required and that fast-learning relevance feedback algorithms are – compared to kernel-based learning – harder to define.

Severe problems as the semantic gap and polysemy have lead to the insight that VIR retrieval should be an iterative process. Retrieval steps must be based on and directed by the user's relevance feedback. Today, one refinement technology outperforms all other approaches: kernel-based learning with support vector machines (SVM) [79, 59]. The advantages of SVM are two-fold: easy application and high performance. SVM are easy to apply, since only two groups (relevant and irrelevant media objects) have to be distinguished by the user. Moreover, even though segmenting relevant and irrelevant samples is a problem of second order, thanks to kernel functions it can be solved in linear time.

Evaluation

Traditional VIR system evaluation is exclusively based on the quality of retrieval operations. Recall, precision and retrieval ranks are used to express retrieval appropriateness in numbers [17, 69, 41]. The obvious drawback of this approach is that retrieval components and media description components cannot be evaluated independently. Surprisingly, this problem has been almost ignored in VIR research. One reason may be that feature extraction algorithms are often seen as "something magical" that cannot be measured. We were the first to introduce an evaluation procedure exclusively for the assessment of media descriptions and applied it, for example, to the content-based MPEG-7 descriptors [20, 25, 28].

System design

System design comprises various aspects relevant for the implementation of successful VIR prototypes: performance improvement, querying parallelisation, loose coupling of system components, etc. A number of different approaches have been suggested to increase the performance of VIR systems (bad performance is mostly caused by slow distance measures), e.g. indexing structures for feature vectors, and complexity reduction algorithms for descriptions. Proposed indexing structures include R-trees, S-trees, etc. [6, 40]. We outlined their major shortcomings for VIR application in [21]. Principal component analysis [53] and singular value decomposition [48] are frequently used complexity reduction methods. Query parallelisation can be achieved through peer to peer networking strategies [27]. Loose coupling of VIR components (e.g. user interfaces and query engines [31]) can be based on web services and XML communication protocols (e.g. the Multimedia Retrieval Markup Language [60]).

Ongoing and future developments

The basic approach to VIR has not changed much since the beginning: Feature extraction algorithms are still applied to compute media descriptions. Retrieval algorithms are still based on low-level descriptions, use query by example interfaces for query definition and relevance feedback for iterative refinement. Moreover, almost the same features are used for media description as ten years ago. The MPEG-7 standard norms colour histograms, wavelet-based texture descriptions, shape moments and motion descriptors. Major past achievements are the introduction of wavelets for space to frequency transformation, the application of kernel-based learning for query refinement and the unified MPEG-7 framework for description extraction from image and video content.

Currently ongoing activities that will continue and expand in the future are reducing the semantic gap by enrichment of low-level features (e.g. by latent semantic indexing [14]), definition of standardised environments for evaluation (e.g. TREC video [62], Benchathlon [4]), development of more sophisticated user interfaces for iterative retrieval, and attempts to assemble tailor-made prototypes for specific application domains (e.g. the Networks of Excellence DELOS [18] and SCHEMA [68]). For future progress in VIR, more powerful methods for image understanding are required. Semantic enrichment of features improves querying results only up to a certain point. Really meaningful object features can only be thought of, if algorithms are able to understand (or at least classify) the presented content correctly. It is questionable if such advances can be achieved in the near future.

Summarising the evolution of Visual Information Retrieval, it can be observed that the major research effort has shifted from the signal processing aspect (feature extraction) to the statistical data mining aspect (classification, recognition, etc.). Today, hardly any new features are introduced, but almost every day new methods for semantic understanding of media descriptions are proposed.

My research contribution

Selection of papers

In my scientific work, together with my co-authors I have published papers in each of the five mentioned key areas of VIR research. In early work we have proposed novel feature extraction methods (both general-purpose and for specific application domains) [11, 8, 10]. Very soon, we started working on the query definition and similarity measurement problem (e.g. [13, 32, 9, 29]). Later, we proposed strategies for query acceleration [33, 30] and on VIR system design (e.g. [37, 31, 26, 12]). In [20, 25, 28] we proposed a novel method for evaluation of feature extraction methods.

For this thesis, I have selected representative papers from each area of research. "A new perspective on Visual Information Retrieval" [21] provides a technical overview over the state of the art and gives indication on ongoing paradigm shifts in VIR. In "Semantic Feature Layers in Content-based Image Retrieval: Implementation of Human World Features" [34], a novel approach for the implementation of semantic feature classes is introduced and an example for symmetry-based features is given. The papers "Visual Similarity Measurement with the Feature Contrast Model" [35] and "Distance measures for MPEG-7-based retrieval" [23] introduce new ideas for similarity measurement: the first paper proposes an iterative retrieval paradigm and a model for the integration of a powerful psychological similarity model in VIR systems. The second paper generalises this idea and compares the most relevant similarity measures proposed over the last century for their performance.

"Statistical analysis of MPEG-7 image descriptions" [28] introduces a novel method for visual descriptor evaluation and shows results for the content-based MPEG-7 descriptors. The three papers "An Experimental Study on the Performance of Visual Information Retrieval Similarity Models" [30], "VizIR - A Framework for Visual Information Retrieval" [36] and "A Data Management Layer for Visual Information Retrieval" [38] propose new system design ideas. The first suggests and compares query acceleration methods, the second introduces a framework for VIR system design, and the third describes a generic database layer for VIR. Finally, "A Video Browsing Application based on visual MPEG-7 Descriptors and Self-Organising Maps" [22] describes an application for video browsing by content.

All ideas presented in these papers were developed by Horst Eidenberger. Furthermore, all papers were written and the evaluation work for all papers except [38] was exclusively performed by myself. In the following subsections we discuss the major contributions of these publications.

A new perspective on Visual Information Retrieval

This paper summarises the state of the art of VIR research. It gives a brief overview over important survey publications in the field, summarises major past advances and describes selected ongoing activities that may have a significant impact on future VIR usage. The paper is organised in four research areas: feature design, similarity modelling, evaluation and system design. Additionally, important media-related aspects are described. The major contribution of the paper lies in integrating image- and video-related activities, in analysing the key components of the similarity modeling problem, and in proposing a software structure for integrated image and video VIR research.

Semantic Feature Layers in Content-based Image Retrieval: Implementation of Human World Features

This paper introduces a novel concept for hierarchical modelling of semantic feature classes. Features on higher levels are exclusively based on features on lower levels. As an example, a feature layer for the distinction of man-made from natural scenes is described. These features recognise "human world properties" from symmetry, geometric and harmonic properties in low-level descriptions. MPEG-7 descriptors are used for low-level description of media objects. For the evaluation, a new test dataset is introduced, XML descriptions are proposed for the high-level features and distance measures are defined. The evaluation results show that the proposed features succeed in understanding the origin of the media objects investigated.

Visual Similarity Measurement with the Feature Contrast Model

In this paper, we propose a model that allows to use the Feature Contrast Model (FCM, proposed for similarity measurement by Tversky [71]) in VIR systems. Following Tversky's argumentation, the FCM has properties that make it superior over other similarity models. Unfortunately, it is based on predicates, i.e. media objects are described by vectors of binary values. Obviously, this is not the case for most VIR features. Hence, we propose a set of statistical operators that allow for using the FCM with numerical features. Furthermore, the paper proposes a similarity measurement process that allows for using multiple features and distance measures in one query simultaneously. The evaluation results show that the proposed model is a successful similarity measure.

Distance measures for MPEG-7-based retrieval

This paper generalises the ideas of the last paper [35]. A unified model for the application of arbitrary distance measures in VIR systems is derived and evaluated. Based on this model, any predicate-based distance measure and any quantitative distance measure (e.g. Euclidean distance) can be used to measure distance between feature vectors. Analytical evaluation shows that the model is successful in transforming predicate-based distance measures in continuous measures and back (e.g. the equivalent Hamming distance and city block distance). Quantitative evaluation (based on MPEG-7 descriptors, test queries and recall and precision) reveals, which distance measures work well on which types of features. Eventually, quantitative analysis shows that some predicate-based measures clearly outperform traditional VIR distance measures (Minkowski distances, etc.). A revised and extended version of this paper is currently under review for journal publication (available from [24]).

Statistical analysis of MPEG-7 image descriptions

In this paper we propose an evaluation procedure for feature data. For illustration, the proposed methods are applied on the visual MPEG-7 descriptors. The evaluation process is based on statistical procedures: among others, cluster analysis techniques (e.g. Self-Organising Maps [49, 50]) and factor analysis methods (e.g. principal component analysis) are applied to identify redundancies, robustness and completeness of media descriptions. The main advantage of this method is that feature extraction methods are analysed by their outcome. Evaluation results are not biased by a retrieval system or human similarity judgement. Applying statistical analysis to the MPEG-7 descriptors revealed that some descriptors have serious shortcomings while others perform excellently for varying content.

An Experimental Study on the Performance of Visual Information Retrieval Similarity Models

This short paper compares the similarity measurement process proposed in [35] to the traditional approach (*k*-nearest neighbour querying) in terms of query execution performance. Every algorithm is employed in its default version and in one optimised version. The results show that the newly introduced querying paradigm clearly outperforms the traditional approach.

VizIR - A Framework for Visual Information Retrieval

VizIR is a software framework for the implementation of VIR systems. This paper describes the intention of the VizIR project as well as the architecture of the VizIR class framework. VizIR consists of two major components: a framework for querying (based on low-level features and the similarity measurement process proposed in [35, 23]) and a framework of user interface classes (based on a 3D interaction paradigm [31, 12]). VizIR makes use of the Multimedia Retrieval Markup Language [60] for loose coupling of query engines and user interfaces, and of the visual part of the MPEG-7 standard for media description. The VizIR project was also accepted for funding by the Austrian Scientific Research Fund (FWF) and is currently implemented at the Institute of Software Technology and Interactive Systems at the Vienna University of Technology. Furthermore, the VizIR group is involved in the SCHEMA and DELOS Networks of Excellence funded by the European Union [68, 18]. Project results are released under GNU General Public License and can be downloaded from [75].

A Data Management Layer for Visual Information Retrieval

This short paper describes the data management layer of the VizIR software framework. This component can be use to serialise arbitrary hierarchies of media objects and content-based media descriptions to arbitrary databases. Design and implementation are based on modern software engineering paradigms. The implemented approach is highly efficient and flexible enough to store any type of media and media metadata (including audio and text). Like the entire VizIR framework it is freely available from [75].

A Video Browsing Application based on visual MPEG-7 Descriptors and Self-Organising Maps

In this paper we propose an application of VIR technology for integrated browsing and retrieval of video data by content and time. Video content can be browsed hierarchically. Relationships between time and content can be defined and the user can switch between these two views at any point in time. Media representation is based on visual MPEG-7 features.

Self-Organising Maps are used for similarity-based organisation of media descriptions. The implementation is based on web protocols and light-weight interaction. First evaluation results show that the proposed approach is successful in making the temporally bound content of video clips transparent in static user interfaces.

References

1. Abdel-Mottaleb, M., Dimitrova, N., Agnihori, L., Dagtas, S., Jeannin, S., Krishnamachari, S., McGee, T., and Vaithilingam, G., "MPEG-7: A Content Description Standard beyond Compression," *Proceedings of IEEE Symposium on Circuits and Systems*, pp. 770-777, Las Cruces, NM, 1999.

2. Ashley, J., Flickner, M., Hafner, J., Lee, D., Niblack, W., and Petkovic, D., "The Query by Image Content (QBIC) System," *Proceedings of ACM SIGMOD International Conference on Management of Data*, p. 475, San Jose, CA, 1995.

3. Attneave, F., "Dimensions of Similarity," *American Journal of Psychology*, vol. 63, pp. 516-556, 1950.

4. Benchathlon Network, Project Website, http://www.benchathlon.net, last visited 2004-09-17.

5. Bertini, M., Del Bimbo, A., and Pala, P., "Content Based Annotation and Retrieval of News Videos," *Proceedings of IEEE Multimedia Conference*, pp. 483-486, New York, NY, 2000.

6. Böhm, C., Berchtold, S., and Keim, D.A., "Searching in High-Dimensional Spaces: Index Structures for Improving the Performance of Multimedia Databases," *ACM Computing Surveys*, vol. 33, no. 3, pp. 322-373, 2001.

7. Bozdogan, H., *Statistical Data Mining & Knowledge Discovery*, Chapman and Hall, Boca Raton, FL, 2003.

8. Breiteneder, C., and Eidenberger, H., "A Retrieval System for Coats of Arms," *Proceedings of International Symposium on Intelligent Multimedia and Distance Education*, Baden-Baden, Germany, 1999 (CD publication, available from http://www.ims.tuwien.ac.at/~hme/papers/isimade1999.pdf).

9. Breiteneder, C., and Eidenberger, H., "Automatic Query Generation for Content-based Image Retrieval," *Proceedings of IEEE Multimedia Conference*, pp. 705-708, New York, NY, 2000.

10. Breiteneder, C., and Eidenberger, H., "Content-based Image Retrieval in Digital Libraries," *Proceedings of IEEE International Conference on Digital Libraries*, pp. 288-295, Kyoto, Japan, 2000.

11. Breiteneder, C., and Eidenberger, H., "Content-based Image Retrieval of Coats of Arms," *Proceedings of IEEE International Workshop on Multimedia Signal Processing*, Helsingör, Denmark, pp. 91-96, 1999.

12. Breiteneder, C., Eidenberger, H., Fiedler, G., and Raab, M., "Lookmark: A 2.5D Web Information Visualization System," *Proceedings of Eurasian Conference on Advances in Information and Communication Technology*, Springer LNCS 2510, pp. 93-101, Teheran, Iran, 2002.

13. Breiteneder, C., Merkl, D., and Eidenberger, H., "Merging Image Features by Self-organizing Maps in Content-based Image Retrieval," *Proceedings of European Conference on Electronic Imaging and the Visual Arts*, pp. 131-138, Berlin, Germany, 1999.

14. Carcassoni, M., Ribeiro, E., and Hancock, E.R., "Texture Recognition through Modal Analysis of Spectral Peak Patterns," *Proceedings of IEEE International Conference on Pattern Recognition*, pp. 243-246, Quebec, Canada, 2002.

15. Chang, S.F., Sikora, T., and Puri, A., "Overview of the MPEG-7 Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 6, pp. 688-695, 2001.

16. Chellappa, R., Wilson, C.L., and Sirohey, S., "Human and Machine Recognition of Faces: A Survey," *Proceedings of the IEEE*, vol. 83, no. 5, pp. 705-741, 1995.

17. Del Bimbo, A., *Visual Information Retrieval*, Morgan Kaufmann, San Francisco, CA, 1999.

18. DELOS Network of Excellence, Project Website, http://www.delos.info/, last visited 2004-09-17.

19. Eakins, J.P., "Trademark Image Retrieval," in Lew, M.S. (ed.), *Principles of Visual Information Retrieval*, Heidelberg, Springer, Heidelberg, Germany, 2001, pp. 319-350.

20. Eidenberger, H., "A New Method for Visual Descriptor Evaluation," *Proceedings of SPIE IS&T Storage and Retrieval Methods and Applications for Multimedia Conference*, SPIE vol. 5307, pp. 145-157, San Jose, USA, 2004.

21. Eidenberger, H., "A New Perspective on Visual Information Retrieval," *Proceedings of SPIE IS&T Storage and Retrieval Methods and Applications for Multimedia Conference*, vol. 5307, pp. 133-144, San Jose, USA, 2004.

22. Eidenberger, H., "A Video Browsing Application based on Visual MPEG-7 Descriptors and Self-Organising Maps," *International Journal of Fuzzy Systems*, 2004 (accepted for publication).

23. Eidenberger, H., "Distance Measures for MPEG-7-based Retrieval," *Proceedings of ACM SIGMM Multimedia Information Retrieval Workshop*, pp. 130-137, Berkeley, USA, 2003.

24. Eidenberger, H., "Evaluation and Analysis of Similarity Measures based on Visual MPEG-7 Descriptors," *Technical Report*, Vienna University of Technology, 2004 (available from http://www.ims.tuwien.ac.at/~hme/papers/tr-ims-2004-08.pdf).

25. Eidenberger, H., "How Good are the Visual MPEG-7 Features?," *Proceedings of SPIE Visual Communications and Image Processing Conference*, SPIE vol. 5150, pp. 476-488, Lugano, Switzerland, 2003.

26. Eidenberger, H., "Media Handling for Visual Information Retrieval in VizIR," *Proceedings of SPIE Visual Communications and Image Processing Conference*, SPIE vol. 5150, pp. 1078-1088, 2003.

27. Eidenberger, H., "Parallel Visual Information Retrieval in VizIR," *Proceedings of SPIE Information Technology and Communication Symposium*, Philadelphia, PA, 2004 (accepted for publication, available from http://www.ims.tuwien.ac.at/~hme/papers/itcom2004-parallel.pdf).

28. Eidenberger, H., "Statistical Analysis of MPEG-7 Image Descriptions," *ACM Multimedia Systems Journal,* Springer, vol. 10, no. 2, 2004 (accepted for publication).

29. Eidenberger, H., "Query Model-Based Content-based Image Retrieval," *Proceedings of Doctoral Symposium of ACM Multimedia Conference*, pp. 513-514, Los Angeles, USA, 2000.

30. Eidenberger, H., and Breiteneder, C., "An Experimental Study on the Performance of Visual Information Retrieval Similarity Models," *Proceedings of IEEE International Workshop on Multimedia Signal Processing*, pp. 233-236, Virgin Islands, USA, 2002.

31. Eidenberger, H., and Breiteneder, C., "A Framework for User Interface Design in Visual Information Retrieval," *Proceedings of IEEE Multimedia Software Engineering Symposium*, pp. 255-262, Newport Beach, USA, 2002.

32. Eidenberger, H., and Breiteneder, C., "Macro-Level Similarity Measurement in VizIR," *Proceedings of IEEE Multimedia Conference*, pp. 721-724, Lausanne, Switzerland, 2002.

33. Eidenberger, H., and Breiteneder, C., "Performance-Optimized Feature Ordering for Content-based Image Retrieval," *Proceedings of European Signal Processing Conference*, Tampere, Finland, 2000 (CD proceedings, available from http://www.ims.tuwien.ac.at/~hme/papers/eusipco2000.pdf).

34. Eidenberger, H., and Breiteneder, C., "Semantic Feature Layers in Content-based Image Retrieval: Implementation of Human World Features," *Proceedings of IEEE International Conference on Control, Automation, Robotic and Vision*, pp. 174-179, Singapore, 2002 (invited paper).

35. Eidenberger, H., and Breiteneder, C., "Visual Similarity Measurement with the Feature Contrast Model," *Proceedings of SPIE IS&T Storage and Retrieval for Media Databases Conference*, vol. 5021, pp. 64-76, Santa Clara, USA, 2003.

36. Eidenberger, H., and Breiteneder, C., "VizIR - A Framework for Visual Information Retrieval," *Journal of Visual Languages and Computing*, Elsevier, vol. 14, no. 5, pp. 443-469, 2003.

37. Eidenberger, H., Breiteneder, C., and Hitz, M., "A Framework for Visual Information Retrieval," *Proceedings of Visual Information Systems Conference*, Springer LNCS 2314, pp. 105-116, HSinChu, Taiwan, 2002.

38. Eidenberger, H., and Divotkey, R., "A Data Management Layer for Visual Information Retrieval," *Proceedings of ACM SIGKDD Multimedia Data Mining Workshop*, pp. 48-51, Seattle, USA, 2004.

39. Flickner, M., Sawhney, H., Niblack, W., Ashley, J., Huang, Q., Dom, B., Gorkani, M., Hafner, J., Lee, D., Petkovic, D., Steele, D., and Yanker, P., "Query by Image and Video Content: The QBIC System," *IEEE Computer*, vol. 28, no. 9, 23-32, 1995.

40. Fonseca, M.J., and Jorge, J.A., "Indexing High-Dimensional Data for Content-Based Retrieval in Large Databases," *Proceedings of International Conference on Database Systems for Advanced Applications*, pp. 267-274, Kyoto, Japan, 2003.

41. Fuhr, N., "Information Retrieval Methods for Multimedia Objects," in Veltkamp, R.C., Burkhardt, H., and Kriegel, H.P. (eds.), *State-of-the-Art in Content-Based Image and Video Retrieval*, Kluwer, Boston, MA, 2001, pp. 191-212.

42. Furht, B., Smoliar, S.W., and Zhang, H., *Video and Image Processing in Multimedia Systems*, Kluwer, Boston, MA, 1996.

43. Hampapur, A., Jain, R., and Weymouth, T., "Production Model-based Digital Video Segmentation," *Multimedia Tools and Applications*, vol. 1, no. 1, pp. 9-46, 1995.

44. IBM, QBIC Imagebase of US Stamps before 1995, http://wwwqbic.almaden.ibm.com/cgi-bin/stamps-demo/, currently not available.

45. Iivarinen, J., Pakkanen, J., and Rauhamaa, J., "A SOM-based System for Web Surface Inspection," *Proceedings of SPIE Conference on Machine Vision Applications in Industrial Inspection*, SPIE vol. 5303, pp. 178-187, San Jose, CA, 2004.

46. Jeannin, S., and Divakaran, A., "MPEG-7 Motion Descriptors," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 6, pp. 720-724, 2001.

47. Kak, A., and Pavlopoulou, C., "Content-based Image Retrieval from Large Medical Databases," *Proceedings of International Symposium on 3D Data Processing Visualization and Transmission*, pp. 138-147, Padova, Italy, 2002.

48. Klema, V., and Laub, A., "The Singular Value Decomposition: Its Computation and Some Applications," *IEEE Transactions on Automatic Control*, vol. 25, no. 2, 164-176, 1980.

49. Kohonen, T., "The Self-Organising Map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464-1480, 1990.

50. Kohonen, T., Oja, E., Simula, O., Visa, A., and Kangas, J., "Engineering Applications of the Self-Organising Map," *Proceedings of IEEE*, vol. 84 , no. 10, pp. 1358-1384, 1996.

51. Krumhansl, C.L., "Concerning the Applicability of Geometric Models to Similarity Data: The Interrelationship Between Similarity and Spatial Density," *Psychological Review*, vol. 85, no. 5, pp. 445-463, 1978.

52. Lew, M.S. (ed.), *Principles of Visual Information Retrieval*, Springer, Heidelberg, Germany, 2001.

53. Loehlin, J.C., *Latent variable models: An introduction to Factor, Path, and Structural Analysis*, Lawrence Erlbaum Associates, Mahwah, NJ, 1998.

54. Long, L.R., and Thoma, G.R., "Computer Assisted Retrieval of Biomedical Image Features from Spine X-rays: Progress and Prospects," *Proceedings of IEEE Symposium on Computer-Based Medical Systems*, Bethesda, MD, pp. 46-50, 2001.

55. Low, C.Y., Tian, Q., and Zhang, H., "An Automatic News Video Parsing, Indexing and Browsing System," *Proceedings of ACM Multimedia Conference*, pp. 425-426, Boston, MA, 1997.

56. Manjunath, B.S., Ohm, J.R., Vasudevan, V.V., and Yamada, A., "Color and Texture Descriptors," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 6, pp. 703-715, 2001.

57. Manjunath, B.S., Salembier, P., and Sikora, T., *Introduction to MPEG-7*, Wiley, San Francisco, CA, 2002.

58. Marques, O., and Furht, B., *Content-Based Image and Video Retrieval*, Kluwer, Boston, MA, 2002.

59. Müller, K.R., Mika, S., Rätsch, G., Tsuda, K., and Schölkopf, B., "An Introduction to Kernel-based Learning Algorithms," *IEEE Transactions on Neural Networks*, vol. 12, no. 2, pp. 181-202, 2001.

60. Müller, W., Müller, H., Marchand-Maillet, S., Pun, T., Squire, D.M., Pecenovic, Z., Giess, C., and de Vries, A.P., "MRML: A Communication Protocol for Content-Based Image Retrieval," *Proceedings of International Conference on Visual Information Systems*, pp. 300-311, Lyon, France, 2000.

61. Naphide, H.R., and Huang, T.S., "A Probabilistic Framework for Semantic Video Indexing, Filtering, and Retrieval," *IEEE Transactions on Multimedia*, vol. 3, no. 1, pp. 141-151, 2001.

62. Over, P., Leung, C., Ip, H., and Grubinger, M., "Multimedia Retrieval Benchmarks," *IEEE Multimedia*, vol. 11, no. 2, pp. 80-84, 2004.

63. Petkovic, D., Dow, R., Gee, M., Vo, M., Vo, P., Holt, B., Hethorn, J., Weiss, K., Elliott, P., Bird, C., Niblack, W., Flickner, M., Steele, D., Lee, D., Yin, J., Hafner, J., Tung, F., and Treat, H., "Recent Applications of IBM's Query by Image Content (QBIC)," *Proceedings of ACM Symposium on Applied Computing*, pp. 2-6, Philadelphia, PA, 1996.

64. Saha, S., and Vemuri, R., "Adaptive Wavelet Coding of Multimedia Images," *Proceedings of ACM International Conference on Multimedia*, pp. 71-74, Orlando, FL, 1999.

65. Santini, S., and Jain, R., "Gabor Space and the Development of Pre-attentive Similarity," *Proceedings of IEEE International Conference on Pattern Recognition*, pp. 40-44, Vienna, Austria, 1996.

66. Santini, S., and Jain, R., "Similarity is a Geometer," *Multimedia Tools and Applications*, vol. 5, no. 3, pp. 277-306, 1997.

67. Santini, S., and Jain, R., "Similarity Measures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 9, pp. 871-883, 1999.

68. SCHEMA Network of Excellence, Project Website, http://www.iti.gr/SCHEMA/, last visited 2004-09-17.

69. Smeulders, A.W.M., Worring, M., Santini, S., Gupta, A., and Jain, R., "Content-Based Image Retrieval at the End of the Early Years," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1349-1380, 2000.

70. Sonka, M., Hlavac, V., and Boyle, R., *Image Processing: Analysis and Machine Vision*, Thomson-Engineering, Toronto, Canada, 1998.

71. Tversky, A., "Features of Similarity," *Psychological Review*, vol. 84, no. 4, pp. 327-351, 1977.

72. Tversky, A., and Gati, I., "Similarity, Separability, and the Triangle Inequality," *Psychological Review*, vol. 89, no. 2, pp. 123-154, 1982.

73. Unser, M., "Texture Classification and Segmentation using Wavelet Frames," *IEEE Transactions on Image Processing*, vol. 4, no. 11, pp. 1549-1560, 1995.

74. Veltkamp, R.C., Burkhardt, and H., Kriegel, H.P. (eds.), *State-of-the-Art in Content-Based Image and Video Retrieval*, Kluwer, Boston, MA, 2001.

75. Vienna University of Technology, VizIR Project Website, http://vizir.ims.tuwien.ac.at/, last visited 2004-09-17.

76. Wu, J.K., Lam, C., Mehtre, B.M., Gao, Y.J., and Desai Narasimhalu, A., "Content-Based Retrieval for Trademark Registration," *Multimedia Tools and Applications*, vol. 3, no. 3, pp. 245-267, 1996.

77. Yang, H., Chaisorn, L., Zhao, Y.L., Neo, S.Y., and Chua, T.S., "VideoQA: Question Answering on News Video," *Proceedings of ACM Multimedia Conference*, pp. 632-641, Berkeley, CA, 2003.

78. Yang, M.H., Kriegman, D.J., and Ahuja, N., "Detecting Faces in Images: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 1, pp. 34-58, 2002.

79. Zhang, L., Lin, F., and Zhang, B., "Support Vector Machine Learning for Image Retrieval," *Proceedings of International Conference on Image Processing*, pp. 721-724, Thessaloniki Greece, 2001.

80. Zrimec, T., "A Content-Based Retrieval System for Medical Images," *Proceedings of IEEE International Conference on Control, Automation, Robotics and Vision*, pp. 180-185, Singapore, 2002.

A new perspective on visual information retrieval

Horst Eidenberger[*]
Vienna University of Technology, Institute of Software Technology and Interactive Systems,
Favoritenstrasse 9-11, 1040 Vienna, Austria

ABSTRACT

Visual information retrieval (VIR) is a research area with more than 300 scientific publications every year. Technological progress lets surveys become out of date within a short duration. This paper intends to shortly describe selected important advances in VIR in recent years and point out promising directions for future research. A software architecture for visual media handling is proposed that allows handling image and video content equally. This allows to integrate both types of media in a singe system. The major advances in feature design are sketched and new methods for semantic enrichment are proposed. Guidelines are formulated for further development of feature extraction methods. The most relevant retrieval processes are described and an interactive method for visual mining is suggested that really puts "the human in the loop". For evaluation, the classic recall- and precision-based approach is discussed as well as a new procedure based on MPEG-7 and statistical data analysis. Finally, an "ideal" architecture for VIR systems is outlined. The selection of VIR topics is subjective and represents the author's point of view. The intention is to provide a short but substantial introduction to the field of VIR.

Keywords: Visual Information Retrieval, Content-based Image Retrieval, Content-based Video Retrieval, Survey, Media Representation, Feature Extraction, Similarity Definition, Evaluation

1. INTRODUCTION

This is a paper on retrieval of visual objects (images and videos) by content. In the year 2003 it is probably one of more than thousand papers in this area of research. In 2002 the IEEE alone has published more than 700 retrieval papers. Figure 1 depicts the increase in visual retrieval publications since 1981 (on basis of the IEEE digital library). Due to the increase of cheaply available (digital) image and video cameras and the increasing power of affordable computer systems visual information retrieval becomes more and more popular as a research discipline. Since 1994 more than hundred papers (=new ideas?) have been published every year.

In this paper we try to fence off important areas of visual information retrieval (VIR). For each area we will shortly describe important past advances and point out relevant, currently ongoing activities. The main focus of the paper is on arguing for new perspectives on selected VIR problem areas. In our opinion, the basic building blocks of VIR research are media management, feature design, querying, evaluation and system design. Each of these areas will be discussed in one section.

Our motivation is that, even though significant advances have been achieved and, by now, a large number of freely available mature VIR systems exists, VIR techniques are not adopted to an adequate extent in relevant application domains (e.g. digital libraries). One major reason may be the discrepancy of hopes associated with VIR (querying by *semantic* similarity) and the reality implemented in most prototypes (querying by low-level features). For example, it is annoying trying to retrieve Hollywood kisses in a movie database by colour, texture and shape features. On the technical level this fact is called "semantic gap"[19].

Even though in recent years a large number of approaches have been proposed to close – or at least narrow – the semantic gap (e.g. semantic enrichment of features, kernel-based learning to find relevant media objects) the potential of VIR still seems to be judged from the performance of the classic prototype systems. Clarifying the state of the art as well as future potentials is certainly an important task if VIR should have a future as a *practically* relevant addition to existing media management and retrieval tools (based on text). From the author's experience, one point should be stressed as most important: VIR technology is able to fulfil sophisticated semantic retrieval tasks but it is *not* able to replace human perception.

[*] eidenberger@ims.tuwien.ac.at; phone 43 1 58801-18853; fax 43 1 58801-18898; www.ims.tuwien.ac.at

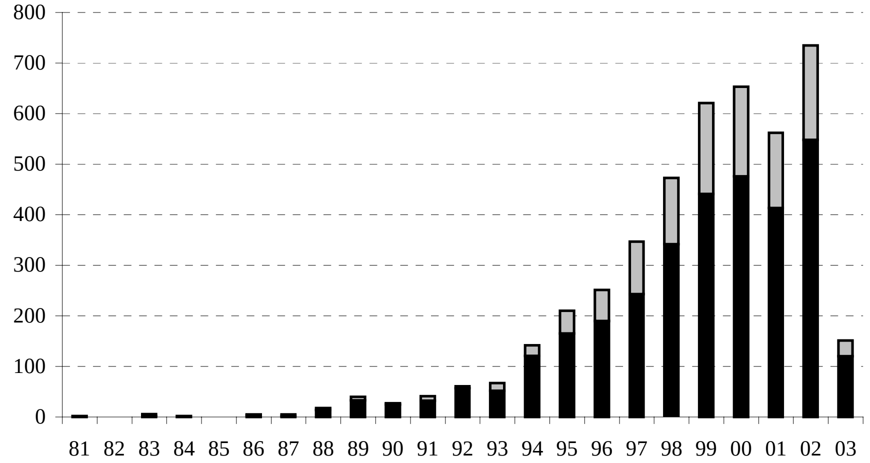


Figure 1: Number of papers in IEEE digital library containing "image retrieval" (black) or "video retrieval" (grey) in bibliographic data. (year 2003: status of 1st October 2003).

This paper reviews VIR from a subjective point of view: It reflects the author's opinion. The organisation is as follows. Section 2 points out relevant related work. The basic VIR building blocks are discussed in consecutive sections: Section 3 visual media, Section 4 visual feature design, Section 5 the retrieval process (similarity definition, interaction), Section 6 evaluation and, finally, Section 7 aspects of VIR system design.

2. BACKGROUND: VIR STATE OF THE ART REPORTS

A handful of VIR publications exists that survey the state of the art. Most of them reflect in organisation and content the perception of VIR of the time when they were written. Below, firstly, we will name a few outstanding representatives and try to sketch their view of VIR. The section will be concluded with remarks on ongoing activities to summarise recent findings in this area of research.

In the book "Image and Video Processing in Multimedia Systems"[14] by Furht, Smoliar and Zhang the state of the art of VIR up to the publication date (1996) is described. The authors start with a system model of content-based image retrieval (CBIR), describe image features (distinguished classically in colour, texture and shape features) and video features (shot detection and camera operation detection), indexing approaches for high-dimensional feature vectors, methods for interactive querying and evaluation based on ground truth information and retrieval quality indicators (recall and precision). Additionally, promising application domains are described and case studies for video visualisation are given.

"Image Retrieval: Past, Present and Future"[18] by Rui, Huang and Chang (1997) concentrates on CBIR. Again, the organisation is classic. Features are split into colour, texture and shape and high-dimensional indexing as well as dimension reduction (e.g. by principal component analysis) are important topics. Well-known CBIR prototype systems (QBIC, Virage, Retrievalware, Photobook, VisualSEEk, MARS) are described in detail. Additionally, this paper was the first survey that described Gabor wavelets as the best suited (in terms of perception) for time to frequency transformation. It led the way for future research as it stressed the importance of putting the "human in the loop" of interactive querying (relevance feedback) and of semantic enrichment of low-level features by artificial intelligence methods. Also, it stated the evident demand for benchmarking initiatives for CBIR systems and gave a first outlook on the MPEG-7 project.

The book "Visual Information Retrieval"[2] by Del Bimbo (published in 1999) is organised by feature groups. As in all other VIR surveys up to now, image and video retrieval are treated separately. For each group of features (colour, texture, shape, motion (shot segmentation only)) extraction methods, distance measures and application examples are described. Classic topics like indexing, evaluation and system design are briefly described. To the author's knowledge this book introduces the terms "semantic gap" and "multi-resolution analysis" for the first time in a survey. The hypothesis of multi-resolution analysis is that using iteratively computed 2D wavelet coefficient matrices as features is sufficient for retrieval. Additionally, the author describes in detail the usage of image features in (spatial) combinations.

The journal paper "Content-based Image Retrieval at the End of the early Years"[22] by Smeulders, Worring, Santini, Gupta and Jain (2000) gives a broad view on CBIR. For the first time selected features are not described in detail but the characteristics of features classes (mainly shape features) are abstracted. Similarity measurement is treated as a topic independently of feature extraction, and distance measures and their geometric foundations are discussed in detail. The importance of learning methods for iterative query optimisation is stressed. Additionally, system aspects (indexing, evaluation, etc.) and techniques of related fields (e.g. edge detection, shape description) are discussed.

Finally, "Content-based Image and Video Retrieval"[16] by Marques and Furht gives only a short overview over the various building blocks of VIR systems and concentrates on conservative techniques. Its major strength lies in the description of a vast number of prototypes for both image and video retrieval. Additionally, design issues of image and video retrieval systems are discussed and case studies are given.

Since hundreds of new ideas are introduced in VIR every year, every survey can only stay up to date for a very short duration. Among the recent publications, the papers on the visual MPEG-7 descriptors[3] can be seen as surveys on feature design, because these features were selected on careful design and comparison to other feature proposals. The currently ongoing SCHEMA project[20] of the European Union intends to provide state of the art reports on content-based media retrieval. At the point in time when this paper is written, deliveries on retrieval concepts, feature extraction and system evaluation are available from the SCHEMA website.

3. THE VISUAL MEDIA

The two types of visual media we are going to consider (image and video) have two major properties that have been examined in VIR research. The first is the colour model used for colour representation and the second is the spatio-temporal resolution of visual media. Colour models have been investigated, for example, by Del Bimbo[2]. Generally, colour models that take human perception into account have been preferred for colour feature extraction. An example is the CIE XYZ space: its unbalanced representation of colours (e.g. more green than red shades) reflects the evolutionary development of the human eye and perception system. For texture and shape analysis, colour models with a luminance channel (originating in TV broadcasting) have been preferred, because, essentially, colour information is irrelevant for this type of analysis. Additionally, a new colour model (HMMD[3]) has been proposed for the MPEG-7 standard. The MPEG-7 authors are arguing that HMMD has properties that make it superior over other colour models. In the author's opinion, since colour values can easily be transformed from any colour model to any other, the selection of colour models is only of minor importance for successful retrieval applications.

Next we will discuss if image and video are similar enough to be handled in one VIR system. The visual media differ significantly in their spatio-temporal resolution. Normally, images have a higher spatial resolution than video. Even though images do usually not contain more information than video frames, due to the different capturing process more scene details are available. The temporal resolution of video is regionally bound and originally derived from TV standards. Images do not have a temporal dimension. Still, a tendency in VIR can be observed to apply features on media objects independently of the availability of a temporal dimension (motion). The authors of the visual part of the MPEG-7 standard stress that their features can be applied reasonably well to both image and video data. They provide structures and models for spatio-temporal localisation and aggregation that allow the application of image features on video content.

We think that in future VIR research the distinction between image and video will become irrelevant. Our argumentation is threefold: Firstly, human vision is a temporal process. The eye scans images and videos by the same saccadic eye movements (to put it simple: close circles in complex areas, larger circles in uniform areas). Therefore, the visual media stream that is sent from the eye to the perception system is always a stream of patterns that has a temporal dimension. Secondly, the result of visual analysis (feature extraction) in VIR is always a number vector of finite length (for technical reasons, etc.). Therefore, image and video are represented by the same type of data. Thirdly, even though some motion features are meaningless for image data, they can at least be used to distinguish the media type by feature vectors. Uniform application of features on media objects is a resource-consuming approach. However, neither

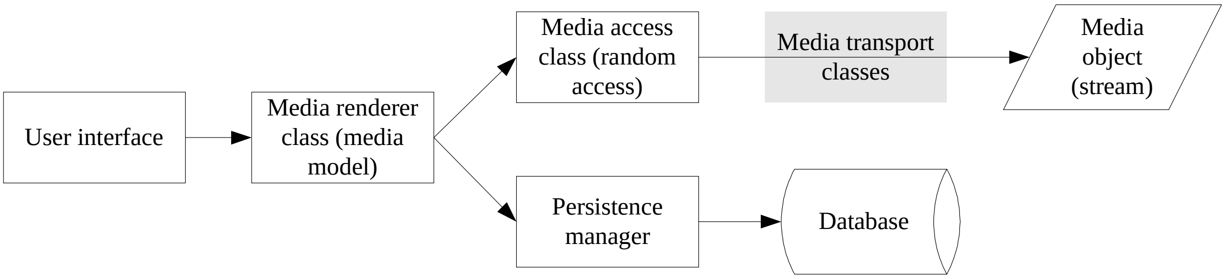


Figure 2: Media encapsulation in VIR.

computation power nor storage is scarce in modern computer systems.

Technically, past VIR prototypes worked either on image data or video data. Mainly, technical shortcomings caused this development. For the future it would be desirable to have VIR prototypes available that support image and video retrieval in a common framework and hide technical media access from VIR-specific tasks (feature extraction, etc.). The author has proposed a VIR framework (called VizIR) that implements this goal[8]. Basically, media access is needed for two functions of VIR systems: feature extraction and media visualisation (e.g. for querying).

VIR video access differs significantly from other media processing applications. Real-time processing is no required. Therefore, video does not have to be considered as a stream but can be accessed like any other pooled data. In the VizIR framework one class is responsible for access of any type of media content. It offers methods for random access of *views*. It is possible to access the view of a media object at any point in time (independent if it is image or video). Additionally, this class is responsible for media content representation and colour space conversion. In a further developed version of this class media objects are abstracted as "visual cubes" (two spatial and one temporal dimension). Transformations (stretching, cutting, etc.) can be applied to manipulate visual cubes.

Media visualisation is (in terms of needed software components) more difficult to perform. The main problem is to visualise the motion in videos in static user interfaces (for querying, result display, etc.). First of all, since user interfaces are normally located on a client while querying components mostly run on a server, media transportation classes are needed that stream the media from server to client. In the VizIR framework, these classes can transparently be attached to the media access class. Media renderer classes are responsible for temporal media visualisation. They make use of the media access interface and construct models of the visualisation that can be used for graphical rendering (e.g. by OpenGL) and be kept persistent in a database. A number of methods have been proposed for video visualisation (e.g. Micons[14]). In VizIR, each method is implemented in a separate media renderer class. Figure 2 summarises the media access components in VizIR.

In conclusion, media-independent availability of visual data in VIR frameworks is a desirable goal. To reach it, making use of software patterns is an important issue (see Section 7). The VizIR framework implements methods for media-independent access. For the future in addition to visual cubes, computing pseudo-saccadic representations of media objects may be worth considering. Completely new features could be designed on the basis of visual pattern streams.

4. FEATURE DESIGN

Since the early days of VIR research, one major focus was on visual feature extraction. The idea of feature transformations is as follows: Since (digital representations of) visual media cannot be easily compared in computer systems (pixel comparison is computational expensive and inadequate to measure *similarity*), there is a need to represent visual content in a form that allows simple but effective (in comparison to human judgement) similarity measurement. In VIR, this is performed by extracting visual media properties as number vectors that can be seen as points in a vector space. If a form of geometry is considered for this space, it is possible to measure dis-similarity as distance. This model is an application of the vector space model of text information retrieval[13].

Since human perception is based on three stimuli: generally perceived (not recognized) stimuli, specifically perceived (recognized) stimuli and pseudo-random (psychological, sociological, etc.) stimuli, two types of features can be distinguished in VIR: *quantitative* (low-level) features and *qualitative* (high-level) features. Unfortunately, only those of the first type can be extracted easily. For the second group semantic understanding would be needed and at the point when this paper is written, software is still far from being able to reason semantically. Therefore, semantic enrichment of low-level features is the mostly adopted course to compute high-level features.

Low-level features are, as pointed out in Section 2, traditionally organised in three groups: (1) colour-related features, (2) texture- and shape-related features and (3) motion-related features. Most colour features (e.g. those in the MPEG-7 standard) extract histograms of pre-defined regions (globally or locally). Only a few approaches exist that make use of colour for other purposes (for example, object segmentation). Texture and shape features can be grouped together, because they make use of the same techniques for feature representation. Both types of features work on the distribution of brightness in visual objects. Texture features aim at detecting statistical edge properties while shape features aim at deriving semantic edge properties (object boundaries). For both types of features it is essential that derived feature representations are invariant against geometric transformations (rotation, scaling, etc.). Motion features include shot detection, camera operation detection and activity detection. Since these features aim at finding features over time, they are mostly built around a core of gradient methods (optical flow, motion trajectories). Usually, low-level feature design results in a cookbook: Building blocks from signal processing (Fourier, Radon transformation, etc.) and other research areas are combined to a new feature. This development has reached a peak in the visual part of the MPEG-7 standard where several cookbooks for low-level features are defined.

One of the most relevant present activities in feature design is semantic enrichment/interpretation of low-level features to narrow the semantic gap. Since as humans we are used to base our similarity judgement on all three groups of stimuli mentioned above, retrieving features just by generally perceived properties is unsatisfactory for us. Generally, three sources of information can be used to enhance features: (1) information on the application domain, (2) information on the user and (3) information on the characteristics of the feature. Additional knowledge can be induced with methods from statistics, artificial intelligence, etc. For example, domain knowledge on football could be used to identify ball and players from shape features (e.g. circularity).

As an example for feature enrichment, in our earlier work we have proposed a semantic feature approach that is based on human perception[9]. Low-level features are used to detect high-level properties that usually play an important role in visual perception. For example, edge and texture features are used to detect symmetries in images. Symmetries are very important for humans. Objects originating from natural processes can easily be distinguished from human-originating objects by their symmetries: Symmetry in nature is never as strict as it is for man-made objects. Probably, it is even possible to distinguish cultures by the symmetries in pictures of their living world. In conclusion, practically, the applicability of semantic enrichment is – at the current point in time – still very limited and for application-independent VIR prototypes no common solution exists.

Another important activity is the ongoing search for 2D segmentation and shape description features. Visual segmentation is the inverse process of rendering. Rendering is a well-posed problem. Therefore, segmentation has to be an ill-posed problem. Nevertheless, the problem is partially solvable, if additional information (on application domain, etc.) is available or if the user helps (for example, by giving a segmentation path). Unfortunately, especially in VIR systems the required additional knowledge (very specific, spatial) is hardly ever present. If we consider, how many different 2D views even a simple object like an apple can have, it becomes unlikely that robust segmentation tools for VIR are possible. However, it will be exciting to see future advances for (narrowly defined) application domains (e.g. salient objects in video).

If we consider the past flood of features, one problem of feature design is obviously answering the question, how many *meaningful* visual features do exist? In other words, which features should be used and which not, because they are outperformed by others? And, on which spatio-temporal regions of media objects should the selected features be applied on? The classic answer to these questions is Multi-Resolution Analysis (MRA). MRA originates in wavelet decomposition. The idea is to make use of a wavelet transformation for computation of wavelet coefficient representations of visual media with decreasing complexity. Either the coefficients themselves or features extracted from the coefficients are used as features (see Figure 3). Unfortunately, it is not clear and could not yet be proven *why* MRA should guarantee that all relevant media parts are properly considered in the feature extraction process.

Our proposal differs from the MRA view: Everything can be a feature, if it fulfils two conditions. Firstly, it has to

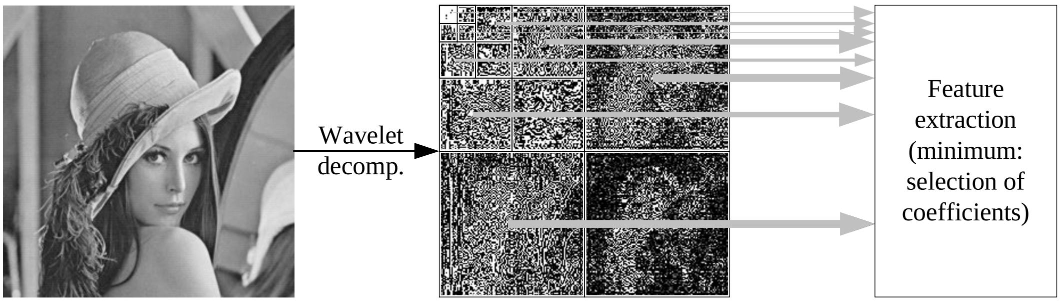


Figure 3: Multi-resolution analysis.

represent a visual property and secondly, it has to be *statistically* independent of existing features. If a feature is statistically independent it is obviously a valuable contribution to a feature set. Independence can be measured by cluster analysis, factor analysis and other methods of statistical data analysis. In previous work we have developed a statistical evaluation procedure and tested the visual MPEG-7 features on these criteria[7, 5] (see also Section 6). Based on this view it is possible to argue for a large number of features to be reasonable. The feature problem is shifted from designing well-performing features to estimating the *relevance* of a feature for a particular querying situation. Essentially, this is up to the user and should be implemented in an iterative retrieval process that makes use of visualisation tools for feature vectors[8].

5. RETRIEVAL PROCESS

Generally, the visual retrieval process aims at finding media objects that are *similar* to given examples. "Similarity" is a weakly defined term and, consequently, difficult to implement in computer systems. Matching by similarity should definitely be less strict than hard pattern matching but still result in comprehensible results. A handful of retrieval processes exists for implementing similarity matching in VIR. Two requirements have to be fulfilled by a model: Similarity matching has to be performed on media objects represented by feature vectors and the user (his feedback) has to be integrated in the retrieval process. Therefore, retrieval is necessarily an *iterative* communication process between man and machine.

Since the actual retrieval process is always based on feature vectors, distinguishing different querying paradigms is irrelevant for the type of retrieval process used. Independently of whether querying by example, sketch, etc. is implemented in the user interface, eventually, the input used for retrieval is always converted to a feature vector (as in text retrieval, where queries are regarded as sets of terms[13]). In consequence we will not refer to different querying paradigms below.

A number of retrieval processes has been introduced to VIR. They are mostly derived from text retrieval concepts. We will consider the four most important models: (1) Distance measurement & indexing, (2) distance measurement and linear merging, (3) distance measurement and non-linear merging and (4) probabilistic retrieval. Except for the last approach, the first step is always distance measurement between the elements of feature space and the given reference point(s).

Distance measurement can be done in two ways: Firstly, a particular type of geometry can be assumed for feature space and metrics can be applied to measure distance. For example, feature space can be assumed to be of Euclidean geometry. Then, the metric axioms hold and any distance measure fulfilling the axioms can be used for distance measurement (e.g. Euclidean distance, city block distance, any Minkowski distance, etc.). Secondly, feature properties (vector elements) can be interpreted as being binary (for example, by fuzzy or probabilistic interpretation). In a binary feature space (populated by binary vectors) predicate-based methods can be used for distance measurement instead of geometric distance measures (e.g. Tversky's well-known Feature Contrast Model[24], Hamming distance, pattern difference).

In recent work we introduced a model that allows for unifying geometric (continuous) and predicate (binary) distance measures[6]. The model allows for using any type of measure on any type of feature data. In experiments on MPEG-7 descriptors we could show that predicate-based measures using the model are often superior over geometric distance measures. The results in the mentioned paper suggest that distance measures should not be designed (derived of feature properties, qualitative arguments) but selected on the basis of quantitative results (e.g. retrieval tests). Generally, the tailor-made distance measure for a feature seldom exists. Optimality depends of the retrieval situation. Therefore, distance measure selection should be automated and derived from given query examples.

Indexing is the art of clever organising data in order to locate them quickly. Since VIR retrieval is based on distance measurement for *all* elements of feature space, indexing as an acceleration technique is irrelevant for querying. But indexing can be used as a querying method itself. In high-dimensional index structures those regions can be selected as positive retrieval results that lie in proximity to the given examples. Unfortunately, hardly any indexing methods do exist that could deal with multiple distance measures and variable (in terms of query examples) data organisation. Therefore, the applicability of indexing methods for VIR is relatively limited.

Linear and non-linear merging approaches are addressing the problem of how to use multiple features (and distance measures) in one query and to retrieve single result set. Linear merging solves the problem by weighting the distance values and summing them up for each media object. Next to weights, transformations are used as well. The resulting value is used to rank media objects and select the first ones as similar. Two problems are connected to linear merging: the weights and the size of the result set have to be provided by the user and some features cannot be combined linearly. Non-linear merging tries to overcome these problems. Often, neural network techniques are used to combine individual distance values to a rank. For example, a multi-layer feed-forward net can be trained on basis of ground truth information. Unfortunately, non-linear methods are – as any other retrieval method – not able to satisfy all user needs and are hardly configurable because of their inflexible architecture.

Using probabilistic approaches (for example, the Binary Independence Model[13]) for retrieval results in two major problems. Firstly, since most models where developed for text retrieval they require binary input that is seldom available in VIR. Again the same methods as for predicate-based distance measurement can be used to convert continuous values to predicates but every additional interpretation step reduces the quality of the results. Secondly, probabilistic models judge general relevance (similarity) on basis of elementary (feature-wise) relevance information. This relevance information has to be provided in form of examples. Already difficult for text retrieval this is nearly impossible for visual data, because the number of possible features and feature values (representing all types of visual cues) is nearly indefinite. Therefore, if probabilistic model are used, then mostly in elementary form (e.g. simple applications of Bayes' theorem).

One major advance in VIR in recent years was achieved in iterative refinement by relevance feedback. Clearly, retrieval should be centered around the user but the question arises of how to apply his feedback in the retrieval process. Here, kernel-based learning techniques[17] mark a significant advance. Using results of previous queries that are enriched by elementary user feedback ("highly relevant", "irrelevant", etc.) as reference points and training a kernel function to segment feature space optimally improves results dramatically. After all, finding a dichotomy of relevant/irrelevant media objects is all that is required of a VIR system. Often used kernel-based learning methods include support vector machines and kernel principal component analysis. The main problem of applying kernel-based learning to VIR is finding a kernel functions that neither over-fits (too complex, too high dimensionality) nor under-fits (too simple, bad segmentation) the retrieval problem.

Unfortunately, even the most sophisticated retrieval and refinement algorithms are still not able to satisfy the user's desire for similarity-based retrieval sufficiently. Therefore, we have designed a retrieval process (called *visual mining*, VM) that is user-centered from the first to the last querying iteration and makes use of 3D perception. Figure 4 shows the retrieval process schematically. Media objects are visualised on the image plane while on the floor dimensions their relative location (distance) is visualised for two features. The features selected for the floor dimensions can be changed at any time implying changes in the organisation of the media objects. This form of visualisation allows the user to visually perceive the retrieval process. Queries are defined by labelling media objects as positive or negative examples. Implicitly, the labelling defines hyper-clusters. The query engine tries to fill the defined clusters with similar objects. For this purpose it makes use of distance functions and data segmentation methods.

Visual mining aims at really putting "the human in the loop"[18]. In Figure 4 image and video objects (represented as Micons[14]) are used in the same query. In a typical querying situation multiple instances of the shown panel are used. For example, one for query definition, one that shows the last result set, one that gives a general overview over feature space, etc. The VM process and the user interfaces are explained in more detail in recent publications[11, 10].

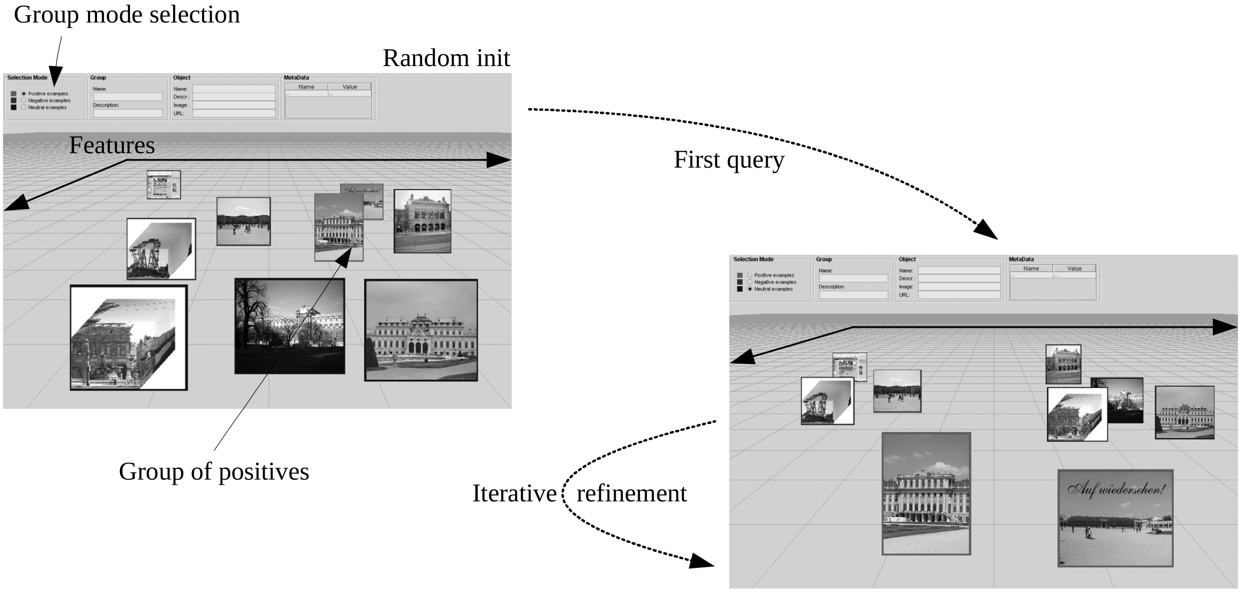


Figure 4: Iterative group querying process.

In conclusion of Section 4 and 5, a great variety of feature design and VIR retrieval methods exists that all have their advantages and disadvantages. To be useful for practical application it is necessary to be able to judge the specific qualities of querying prototypes. In the next section, the methods mostly used for VIR evaluation are shortly sketched and new methods that could supplement existing ones are proposed.

6. EVALUATION

Evaluation of VIR systems is needed for various purposes: It has to be possible to judge the quality of new feature extraction methods in relation to existing ones, to compare the quality of novel querying paradigms, to judge the usability of user interfaces for retrieval, etc. The most interesting problem is measuring the quality of similarity measurement compared to human visual similarity perception. For this purpose, the recall and precision quality indicators of text information retrieval evaluation were adopted[13]. Recall and precision are usually defined as follows:

$$recall = \frac{|retrieved \cap relevant|}{|relevant\ objects|}, precision = \frac{|retrieved \cap relevant|}{|retrieved\ objects|} \qquad (1)$$

In case of VIR, objects are media objects represented by feature vectors. Recall and precision are inter-dependent. It is easily possible to optimise one indicator, if the other is not considered. Meaningful results can only be derived if both indicators are considered. In addition to recall and precision other measures exist (for example, ANMRR, used for evaluation of visual MPEG-7 descriptors[15]).

VIR evaluation based on recall and precision is a four-step process (see Figure 5): (1) Definition of a media set. The media set should be appropriate for the evaluation goal and contain a reasonably large number of items. Often, collections of thousand and more media objects are used. (2) Derivation of ground truth information. The ground truth says, which objects in a media set are similar (and sometimes, *how* similar they are). Ideally, it should be invariant against cultural, sociological and other human-related influence factors. In practice, deriving such a ground truth is impossible. Usually, groups of more than average similarity are defined by a few test users. (3) Execution of test queries. This step requires automatic selection of query examples and a sufficiently large number of test queries. For guaranteeing statistical correctness, the number of test queries should be hundred or larger. (4) Computation of retrieval indicators. Recall and precision can, for example, be averaged over all test queries and visualised in a recall-precision-graph. This evaluation procedure has several shortcomings: Firstly, it is subjective and culture-dependent (media

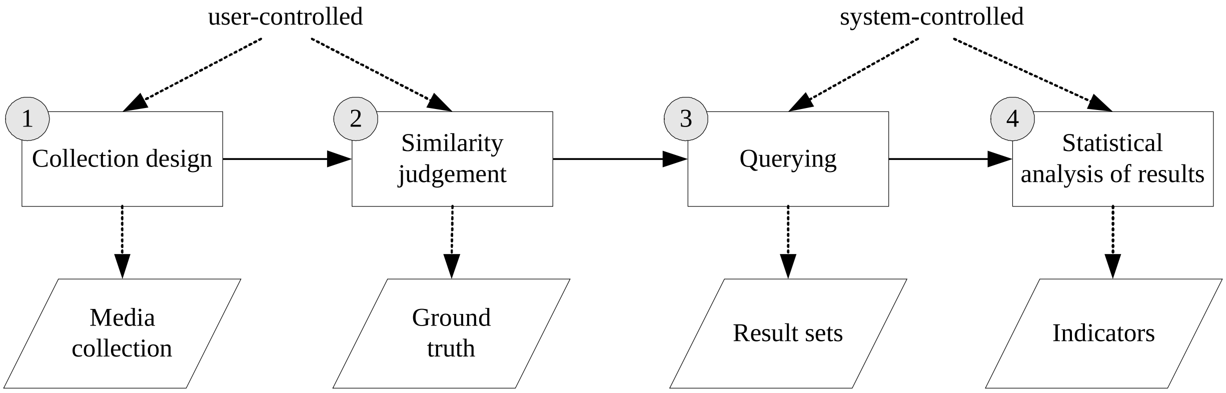


Figure 5: VIR evaluation process.

collection, ground truth). Secondly, it cannot be used to evaluate interactive retrieval processes. Thirdly, it is a heavy-weight process that adds a lot of influence factors that may bias the evaluation results. For example, this may be the case if a new feature should be evaluated.

Present evaluation activities include gathering free media objects in public collections (e.g. the Benchathlon project[1]) and events for comparative system evaluation. One example for the second is the annual TREC video retrieval competition[21]. VIR groups can attend in a number of competitions (e.g. shot segmentation) and see how good their methods are in comparison to other approaches. Additionally, a new (very large) set of video clips is created each year that can be used for other purposes as well. This is especially positive since most freely available visual media collections are image collections.

In our recent work we have proposed an evaluation procedure for features that is based on statistical data analysis and the visual MPEG-7 features[5, 7]. The procedure makes use of factor analysis and cluster analysis techniques. In contrast to the standard procedure it does not suffer from the three mentioned disadvantages. Essentially, feature vectors are calculated for arbitrary media collections and compared to the MPEG-7 feature vectors by statistical methods. The results can be used to judge the feature type (colour, texture, etc.), redundancies with existing approaches, etc. It is intended to be used as a supplement to recall- and precision-based evaluation.

7. SYSTEM DESIGN

Good, professional system design is not a VIR-specific issue; it is desired for any type of information system. What makes system design especially important in VIR is the fact that acceptance of VIR methods is strongly bound to their appearance. Since VIR systems actually fail to fulfil the promise of human-like similarity retrieval, it is even more important that they are at least fast and easy to use tools for visual media mining (pre-selection of likely hits). Below, we point out the design of classic systems, currently ongoing design activities and our ideas for ideal VIR system design.

Past VIR prototypes were mostly monolithic systems that ran on server side and were limited to one type of media. Most VIR systems implemented image retrieval: a few features (colour histogram, texture moments, etc.), query by example and retrieval by linear merging. Most of them were general-purpose, some application-specific (e.g. for trademark retrieval). Video retrieval systems were mostly intended for specific applications (e.g. news analysis) and often concentrated on the user interface aspect (visualisation of temporal media in static user interfaces). Well-known VIR prototypes include QBIC, Virage, RetrievalWare, Photobook, VisualSEEk, MARS, OVID and CueVideo. Surveys exist that evaluate these and other prototypes and compare them by their advantages and disadvantages[16, 27].

IBM's Query by Image Content system[12] (QBIC) may stand as a representative for these prototypes. QBIC is a classic system that introduced many of the concepts that are implemented today in a wide range of VIR prototypes. QBIC is based on the C++ programming language and organised in components. The architecture is extendible: new features and query engines can be defined and added. Querying components are separated from the user interface and communicated over HTTP. Image data is encapsulated in a data class that is also responsible for converting various image formats to

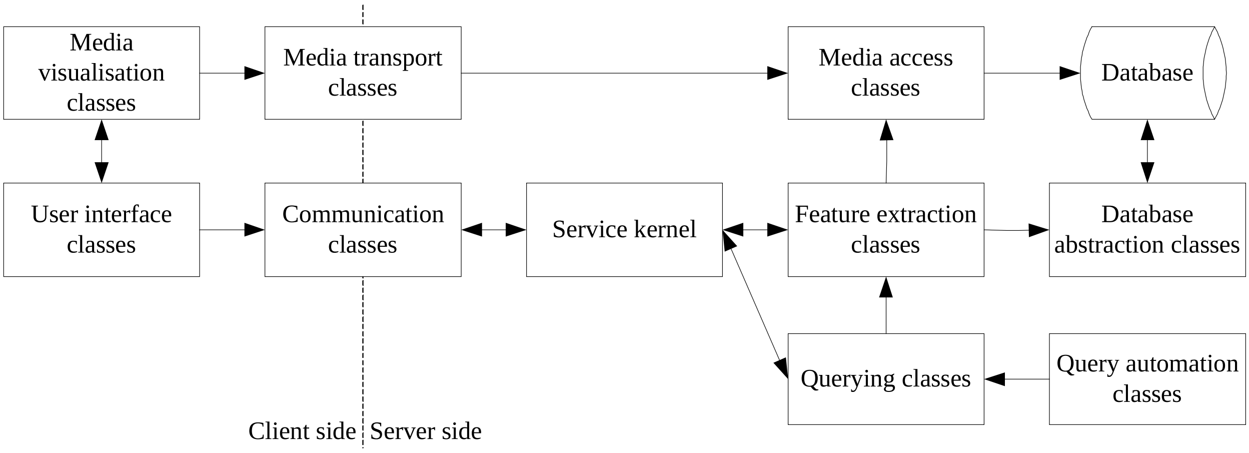


Figure 6: Ideal VIR system design. Arrows show "make use" dependencies.

raw RGB pixel maps. Those source code elements needed for the extension mechanism are shipped with the binary distributions for various operating systems. QBIC contains a number of state-of-the-art feature classes and used linear merging for retrieval. Additionally, it is based on a simple file database for feature storage.

At present, these concepts are imitated in a number of prototypes. For example, the GNU Image Finding Tool[25] (GIFT) makes use of the Multimedia Retrieval Markup Language[26] (MRML, based on XML) for loose coupling of server and client components. GIFT is open source and based on other GNU components that allow using a large number of data formats for image querying. Since the communication language for server and client components is standardised, different user interfaces can be used to access the query engine.

The MPEG-7 experimentation model[23] (XM) goes one further step ahead, as it allows querying in image and video collections. Like for QBIC and GIFT, the XM classes are split in server components (for querying) and client components. It allows extension with new descriptors and is available as open source. Unfortunately, the practical use of the XM is limited, because only a very small number of video formats are supported and hardly any documentation exists for architecture and application programming interfaces. Still, the XM is used as basis for a number of VIR projects. For example, the SCHEMA project of the European Union[20] develops new VIR solutions on basis of the MPEG-7 XM. Other projects (e.g. of the DELOS Network of Excellence of the European Union[4]) are following different, individual approaches.

In recent publications we have proposed an "ideal" architecture for VIR systems. This architecture is currently under development in the VizIR project[11]. One major goal of the VizIR project is providing a framework of VIR tools that are media-independent. Another is encapsulating visual media in a way that most common image and video formats are supported and that media content can be accessed with exactly the same methods. VizIR is an open source project that is based on the Java programming language. It implements all of the proposals for feature design, retrieval and evaluation made in this paper.

Figure 6 shows the VizIR system design. Components are split into typical client components (user interfaces) and server components. Client components are the user interface presented in Section 5 and the classes for visual media representation presented in Section 3. On the server side a service kernel is responsible for dispatching server calls (e.g. query execution, media management). This service kernel can, for example, be implemented as a web service using SOAP, WSDL and UDDI. It organises the classes for querying and feature extraction that are derived from general interfaces. Therefore, it is easily possible to extend the VizIR framework with new features and querying paradigms. Database storage and additional functionalities for query acceleration (feature vector indexing, querying heuristics, etc.) are encapsulated in an object-oriented persistence manager that hides the database (for feature storage, etc.) from the VIR-specific classes. The same purpose is fulfilled by the media access classes for the media objects. Query automation classes are used for evaluation purposes.

Communication between server and client side is performed by communication classes that make use of XML messaging and are fully compatible with the service kernel. For media transport individual classes are implemented that fulfil their job in separate threads in the background. It is important to notice that all VizIR framework components are designed to be applicable independently of the type of media used and of the location from where they are used. It is possible to build arbitrary VIR applications by using existing building blocks. New ones can be added easily. In order to guarantee that every component can communicate with any other, event-based messaging is used and implemented following established design patterns (e.g. SUN's delegation event model). Generally, design patterns are used wherever possible (e.g. factories for media access).

8. CONCLUSIONS & OUTLOOK

This paper summarises selected advances in visual information retrieval. We try to sketch important advances in visual media representation, feature extraction, retrieval (including query definition, similarity measurement and query refinement). Additionally, we propose problem areas and possible solutions for future visual information retrieval research. The selection is subjective: it represents the author's point of view on image and video retrieval.

The major problem of visual information retrieval is its failure to imitate human visual perception and human similarity judgement properly. The goal is to automatically find visual media in, usually very large, collections by imitating human visual similarity perception. Clearly, since computers are still unable to do visual reasoning and recognise the real world objects behind two-dimensional views, they are condemned to fail. What they can do is to extract visual features on a low syntactical level and to measure dis-similarity as distance. Even though this service can be of great value (e.g. as a pre-selection step when mining large media collections), the unsatisfactory results are a major reason why content-based retrieval techniques are still hardly used in digital library systems and other applications.

In consequence, the key question is: does visual information retrieval have a perspective for practical application? To the author's belief, this question can be answered by "yes" if research and implementation focus are laid on issues different from the currently most investigated. Visual information retrieval is a mining tool that should be centered around the user and have its major strength in the user interface components used for media and query visualisation. Systems have to be designed in an easy to use way and it has to be made clear that visual information retrieval systems are not intended to replace but to supplement human beings and their visual perception system.

ACKNOWLEDGEMENTS

The author would like to thank Christian Breiteneder for his valuable comments and suggestions for improvement. This work is part of the VizIR research project[11] that is supported by the Austrian Scientific Research Fund (FWF) under grant no. P16111-N05.

REFERENCES

1. Benchathlon network website, http://www.benchathlon.net/ (last visited 2003-10-25).

2. A. Del Bimbo, *Visual Information Retrieval*, Morgan Kaufmann Publishers, San Francisco, 1999.

3. S.F. Chang, T. Sikora, A. Puri, "Overview of the MPEG-7 standard", *Special Issue on MPEG-7, IEEE Transactions on Circuits and Systems for Video Technology*, **11/6**, 688-695, 2001.

4. DELOS EU Network of Excellence website, http://delos-noe.iei.pi.cnr.it/ (last visited: 2003-10-25).

5. H. Eidenberger, "A new method for visual descriptor evaluation", *Proceedings SPIE Electronic Imaging Symposium*, SPIE, San Jose, 2004 (to appear).

6. H. Eidenberger, "Distance Measures for MPEG-7-based Retrieval", *Proceedings ACM Multimedia Information Retrieval Workshop*, ACM Multimedia Conference Proceedings, Berkeley, 2003 (to appear).

7. H. Eidenberger, "How good are the visual MPEG-7 features?", *Proceedings SPIE Visual Communications and Image Processing Conference*, vol. 5150, 476-488, SPIE, Lugano, 2003.

8. H. Eidenberger, "Media Handling for Visual Information Retrieval in VizIR", *Proceedings SPIE Visual Communications and Image Processing Conference*, vol. 5150, 1078-1088, SPIE, Lugano, 2003.

9. H. Eidenberger, C. Breiteneder, "Semantic Feature Layers in Content-based Image Retrieval: Implementation of Human World Features", *Proceedings IEEE International Conference on Control, Automation, Robotic and Vision*, Singapore, 2002 (published on CD, available from: http://www.ims.tuwien.ac.at/~hme/papers/icarcv2002.pdf, last visited: 2003-10-25).

10. H. Eidenberger, C. Breiteneder, "Visual similarity measurement with the Feature Contrast Model", *Proceedings SPIE Storage and Retrieval for Media Databases Conference*, vol. 5021, 64-76, SPIE, Santa Clara, 2003.

11. H. Eidenberger, C. Breiteneder, "VizIR – A Framework for Visual Information Retrieval", *Journal of Visual Languages and Computing*, **14**, 443-469, 2003.

12. M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, P. Yanker, "Query by Image and Video Content: The QBIC System", *IEEE Computer*, **28/9**, 23-32, 1995.

13. N. Fuhr, "Information Retrieval Methods for Multimedia Objects", *State-of-the-Art in Content-Based Image and Video Retrieval*, R.C. Veltkamp, H. Burkhardt, H.P. Kriegel, 191-212, Kluwer, Boston, 2001.

14. B. Furht, S.W. Smoliar, H. Zhang, *Video and Image Processing in Multimedia Systems*, Kluwer, Boston, 1996.

15. B.S. Manjunath, J.R. Ohm, V.V. Vasudevan, A. Yamada, "Color and texture descriptors", *Special Issue on MPEG-7, IEEE Transactions on Circuits and Systems for Video Technology*, **11/6**, 703-715, 2001.

16. O. Marques, B. Furht, *Content-Based Image and Video Retrieval*, Kluwer, Boston, 2002.

17. K.R. Müller, S. Mika, G. Rätsch, K. Tsuda, B. Schölkopf, "An Introduction to Kernel-based Learning Algorithms", *IEEE Transactions on Neural Networks*, 12/2, 181-202, 2001.

18. Y. Rui, T.S. Huang, S.F. Chang, "Image Retrieval: Past, Present, And Future", *Proceedings International Symposium on Multimedia Information Processing*, 1997.

19. S. Santini, R. Jain, "Similarity Matching", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **21/9**, 871-883, 1999.

20. SCHEMA EU project website, Delivery on visual information retrieval techniques, available from http://www.iti.gr/SCHEMA/preview.html?file_id=67/ (last visited 2003-10-25).

21. A.F. Smeaton, P. Over, "The TREC-2002 video track report", *NIST Special Publication*, SP 500-251, 2003 (available from: http://trec.nist.gov/pubs/trec11/papers/ VIDEO.OVER.pdf, last visited: 2003-10-25).

22. A.W.M. Smeulders, M. Worring, S. Santini, A. Gupta, R. Jain, "Content-based image retrieval at the end of the early years", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22/12**, 1349-1380, 2000.

23. TU Munich, MPEG-7 experimentation model website, http://www.lis.e-technik.tu-muenchen.de/research/bv/topics/ mmdb/e_mpeg7.html (last visited: 2003-10-25).

24. A. Tversky, "Features of Similarity", *Psychological Review*, 84/4, 327-352, 1977.

25. University of Geneva, GNU Image Finding Tool website, http://www.gnu.org/software/gift/ (last visited: 2003-10-25).

26. University of Geneva, Multimedia Retrieval Markup Language website, http://www.mrml.net/ (last visited: 2003-10-25).

27. R. Veltkamp, M. Tanase, D. Sent, "Features in Content-based Image Retrieval Systems", *State-of-the-Art in Content-Based Image and Video Retrieval*, R.C. Veltkamp, H. Burkhardt, H.P. Kriegel, 97-124, Kluwer, Boston, 2001.

Semantic Feature Layers in Content-based Image Retrieval: Implementation of Human World Features

Horst Eidenberger and Christian Breiteneder

Vienna University of Technology, Institute of Software Technology and Interactive Systems,
Favoritenstrasse 9-11 – 188/2, A-1040 Vienna, Austria
{eidenberger, breiteneder}@ims.tuwien.ac.at

ABSTRACT

The major problem of most CBIR approaches is bad quality in terms of recall and precision. As a major reason for this, the semantic gap between high-level concepts and low-level features has been identified. In this paper we describe an approach to reduce the impact of the semantic gap by deriving high-level (semantic) from low-level features and using these features to improve the quality of CBIR queries. This concept is implemented for a high-level feature class that describes human world properties and evaluated in 300 queries. Results show that using those high-level features improves the quality of result sets by balancing recall and precision.

1 INTRODUCTION

Content-based Image Retrieval (CBIR, [2]) approaches aim at finding images that are *semantically similar* to a given query (often a single example image). In this definition, 'semantically similar' is meant in the sense of human visual similarity perception (in CBIR publications mostly just called 'high-level'). The methods used to satisfy this demand are based on numerical feature extraction (e.g. with signal processing and computer vision techniques) and (metric-based) distance measurement. This approach is usually referred to as 'low-level'. Now the problem of most (general-purpose) CBIR approaches is bad quality in terms of recall and precision. As a major reason for this, the semantic gap has been identified ([9]). This is the gap between the high-level requirements of CBIR and the low-level implementation.

In this paper we describe a novel approach to reduce the impact of this semantic gap. Usually, iterative refinement by relevance feedback is used to minimize the semantic gap in CBIR systems ([7], [12]). We follow a different path by deriving high-level (semantic) from low-level features and using these features to improve the quality of CBIR queries. We show by an example prototype implementation and evaluation the idea of the approach.

The results of this paper are part of the Visual Information Retrieval project VizIR. The VizIR project aims at the following major goals:

- Implementation of a modern, open class framework for content-based retrieval of visual information as basis for further research on successful methods for automated information extraction from visual media, definition of similarity measures and new, better concepts for the user interface aspect of visual information retrieval.
- Implementation of a working prototype system that is fully based on the visual part of the MPEG-7 standard. Obtaining this goal requires seeking for suitable extensions and supplementations of the MPEG-7 standard.
- Development of integrated, general-purpose user interfaces for visual information retrieval.
- Support of methods for distributed querying, storage and replication of visual information and features and methods for query acceleration.

To achieve these goals state-of-the-art software development is necessary. VizIR is based on reverse engineering and the Rational Unified Process ([6]). The output of VizIR will be available to the public. The overall goal of VizIR is providing the research community with a flexible tool for experiments. See [3] for more information on VizIR.

The rest of the paper is organized as follows. Section 2 points out relevant related work, Section 3 describes the idea of semantic features, in Section 4 we outline the design of the Human World Feature class (HWF), Section 5 describes the implementation of HWF in our test environment, Section 6 discusses experimental results and finally, Section 7 sketches our next activities in the context of this paper.

2 RELATED WORK

Subsequently, we will review the semantic gap problem, point out a second current approach for semantic feature extraction and briefly describe the descriptor definition language (DDL) of the MPEG-7 standard, that will be used to describe HWF.

According to [9], the semantic gap can be defined as the space of disappointment between the high-level intentions of CBIR and the low-level features that are used for querying. The size of the gap in current general-purpose systems ranges from 60% to 80% of querying performance (recall and precision, e.g. in [10]). In his keynote speech at the Visual Information Systems conference 2002, William Grosky described a semantic feature extraction method related to the Semantic Web project ([8]) that should help to reduce the semantic gap. Basically, the idea is to integrate close distant information into the feature extraction process. For example, on a webpage, image features are not just derived for the area of each image but for an area that includes the image and the text around it. Thus, semantic information is integrated in the feature vectors. The

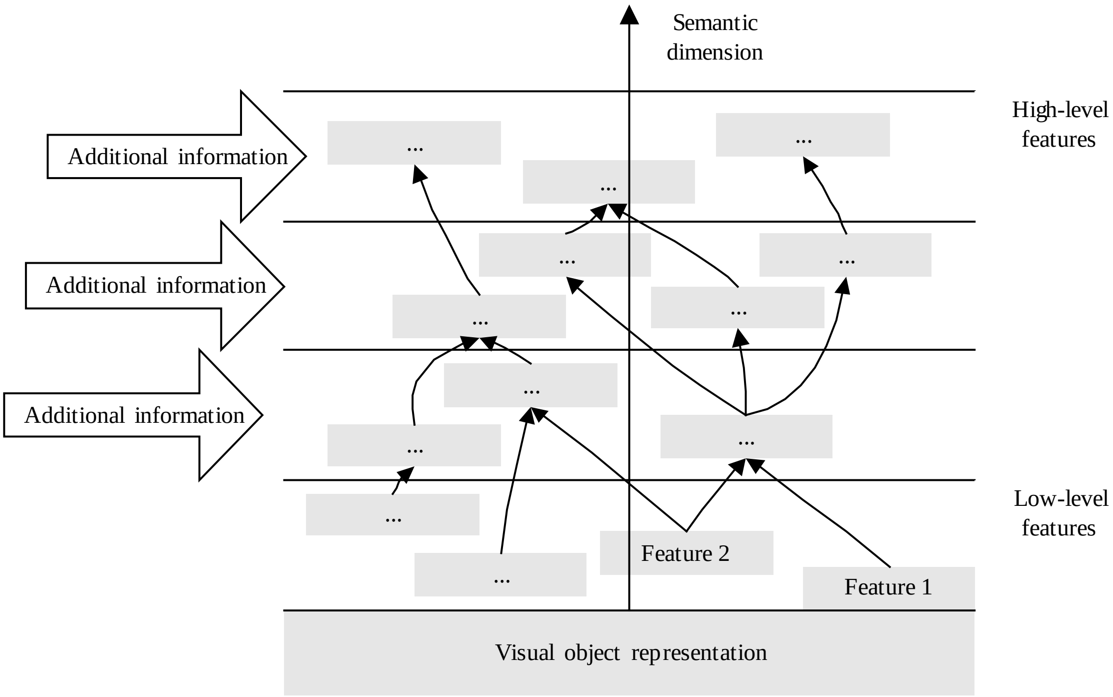


Figure 1. Semantic feature layers. Features on higher levels are based on the Descriptors of features on lower levels. Together with additional information they derive new Descriptors on higher semantic levels. Additional information includes modeling information, statistical information and domain knowledge. This model should help to narrow the semantic gap.

problem of this method – from our point of view – is that it is difficult to argue, *why* adding image rendered text information to the image feature extraction process should improve the quality of retrieval results.

The semantic features introduced below are defined on the basis of the MPEG-7 Descriptor Definition Language (DDL). MPEG-7 defines Descriptors (D), Descriptor Schemes (DS) and the DDL. DS are containers of D and DS. The DDL is a uniform method for the description of D and DS. Essentially, the DDL is the XML Schema Language, extended by a few custom data types (like matrices, histograms, etc.). As the authors of [5] state, 'the DDL is not a modeling language such as Unified Modeling Language (UML, [11]) but a schema language to represent the *results* of modeling audiovisual data.'. Thus it is impossible to model the usage of additional knowledge in D and DS.

3 SEMANTIC FEATURE LAYERS

The idea of semantic feature layers (SFL) is the design of semantically related feature classes that are based on features of lower levels and include additional knowledge (see Figure 1). Additional knowledge can be comprised of modeling information, domain knowledge, statistical information, etc. and be expressed as data (e.g. a color covariance matrix) or as algorithms (e.g. a sophisticated distance measurement algorithm). SFL should help to reduce the size of the semantic gap.

SFL are more than DS. DS define hierarchical relationships of static Descriptors and other DS. In SFL, Descriptors do not remain static on higher levels but are transformed by additional knowledge to more specific (semantic) representations. Using SFL in addition or instead of low-level features has two major advantages:

1. It is possible – in the context of the SFL – to perform high-level queries without the need to translate them to queries on low-level features. This should lead to better results.
2. Queries are much faster, because of simpler feature vectors and simpler querying methods. The integration of additional knowledge on the basis of low-level features will in most cases lead to a compression of the high-level feature vectors. This process is performed offline during the feature extraction process. Querying methods can be simpler because no mapping is necessary and feature vectors are simpler.

SFL are an abstraction of low-level features. In the next section we will introduce an example of a SFL for the description of human world properties in images.

4 HUMAN WORLD FEATURES

The world of visual objects (from the human point of view) can be split into two groups: nature-originated objects (e.g. landscapes, trees, etc.) and human-originated objects (e.g. equipment, houses, etc.). The idea of the human world properties SFL (HWL) is the definition of features that describe typical properties of human-originated objects and scenes. This is useful, because most images consist of both types of objects and the relationship of them is often typical for a certain image group (cluster, application domain, etc.). For

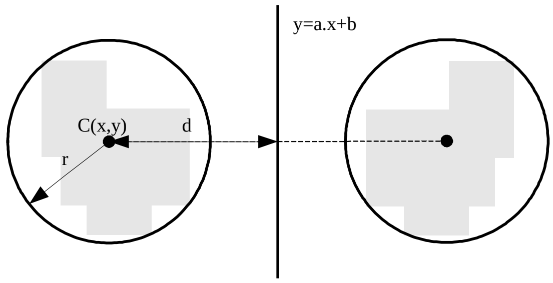


Figure 2. Semantic symmetry feature: symmetry axis detection. The symmetry axis for two objects is derived from the circles around them.

example, with HWL features images of family photos at Christmas can easily be distinguished from family photos in summer (at least in the colder regions of the world), because Christmas photos are usually made indoors (with a lot of human-originated objects in the background) while summer photos are usually made outdoors (with significantly more nature-originated objects in the background).

We have identified three major properties of human-originated objects that can be relatively easy described with numerical feature vectors:

1. Geometry. Humans love to create objects with the major properties of Euclidean geometry: straight lines and right angles. These properties are hardly present in natural objects.
2. Harmony. This includes human characteristics like the harmonic application of colors (matching colors and color shades), harmonic textures and the regular arrangement of objects in scenes. Even though the human preference for harmony is presumably originated in natural characterization it furthermore has a cultural component that makes it different from the harmony appearing in natural scenes.
3. Symmetry. This does not refer to the mathematical symmetry term (concerning symmetric objects, this symmetry exists in nature as well) but to the *symmetric arrangement* of objects (represented by more or less coarse object representations) that can be symmetric (e.g. a row of windows), mirrored (e.g. semidetached houses) or repetitive (e.g. a row of computers).

These properties are employed to judge whether an object appearing in a scene is human-originated. They can be represented by feature classes that can be based on arbitrary low-level features that include spatial or geometric information (e.g. localized color histograms, object descriptions, edge histograms, etc.). As an example, let us detail the algorithm for an implementation of the symmetry feature. It consists of three steps (see Figure 2):

1. Extraction of all occurrences of the underlying feature in the visual object. The underlying feature can be every feature not invariant against mirroring and that may be contained multiple times in a visual object (e.g. spatial color distribution, texture moments, etc.).

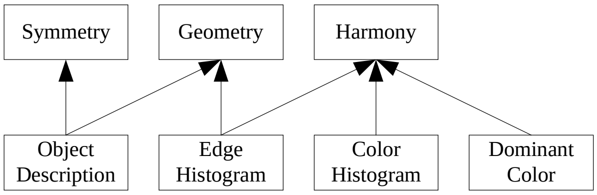


Figure 3. Human world feature layer implementation. The three high-level features are based on four low-level features.

2. Extraction of all mirrored occurrences of the underlying feature in the visual object. Each found object is represented by the radius and center of the circle around it.
3. Detection of the parameters of the symmetry axis for found pairs.

The Descriptor of this symmetry feature (according to a specific underlying feature) could be the following vector:

$$\left(C(x,y),r,a,b,d\right)^T$$

where *C(x,y)* and *r* are defined as above (for a not mirrored object), *a* and *b* are the parameters of the symmetry axis and *d* is half of the shortest distance from *C(x,y)* to the symmetry axis. For our tests we used an even simpler implementation of the symmetry feature. The next section is dedicated to this matter.

5 IMPLEMENTATION

For experimental evaluation (see Section 6 for results) we have implemented a simple version of the HWL. It consists of three features, one for each of the properties above. These features are based on four low-level features. Figure 3 shows the dependencies.

The first low-level feature derives a simple object description that includes the object size (in macroblocks), the circularity of the border (as defined in [4]) and the position in the image for the first five objects. A macroblock has one 64th of the width and height of the image. The edge histogram has four bins for all edges in an image with length smaller than one macroblock, one to two macroblocks, two to four macroblocks and more than four macroblocks. Additionally, we use a global color histogram with nine bins and the MPEG-7 dominant color feature with two bins. The first three low-level features use Euclidean distance functions for dissimilarity measurement. The dominant color feature uses the following function to compare two objects *A=(c1$_A$,c2$_A$)* and *B*.

$$d(A,B)=0.5u(c1_A,c1_B)+0.2u(c1_A,c2_B)+\\0.2u(c2_A,c1_B)+0.1u(c2_A,c2_B)$$

where *u(x,y)* is defined as follows:

$$u(x,y)=\begin{cases}0...if\ x=y\\1...otherwise\end{cases}$$

The weights were set based on heuristics. The output of

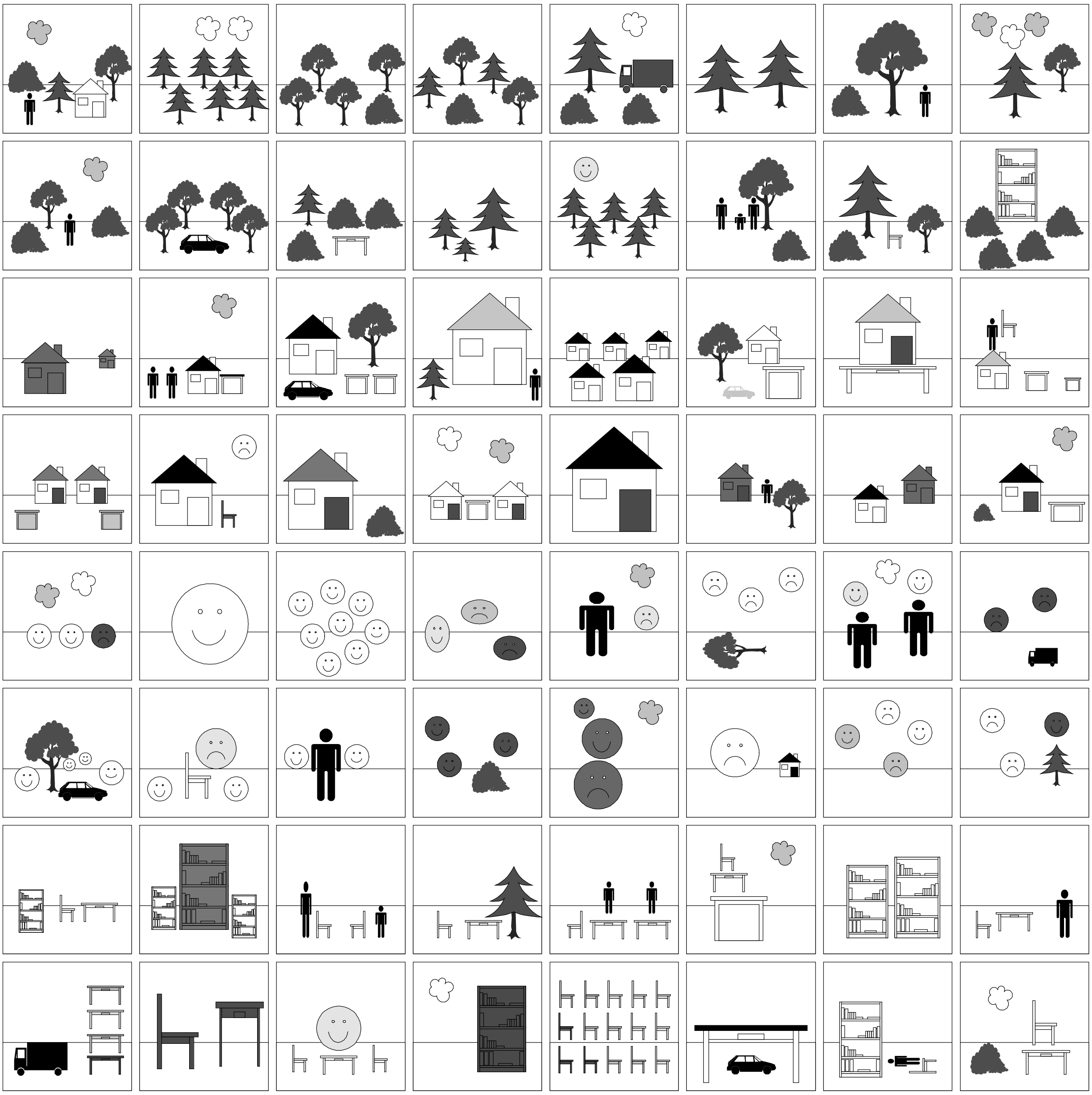


Figure 4. Test images and ground truth. The collection consists of four groups with 16 images each. The four groups are: images of forests (first and second row), images of houses (third and fourth row), images of faces (fifth and sixth row) and images of equipment (seventh and eighth row).

all distance measures is normalized to the interval *[0,1]*.

The geometry feature is based on the object description and the edge histogram. It measures the number of straight lines with significant length (longer than two macroblocks; derived from the edge histogram) and the number of right angles in an image (derived from the circularity values). We define the following MPEG-7 descriptor:

```
<complexType name=”GeometryFeature”>
   <element name=”StraightLines”
    type=”unsignedInt” use=”required”/>
   <element name=”RightAngles”
    type=”unsignedInt” use=”required”/>
</complexType>
```

The distance of two descriptors $A=(sl_A,ra_A)$ and B is measured with the following distance function.

$$d(A,B)=\sqrt{\frac{(sl_A-sl_B)^2+(ra_A-ra_B)^2}{2}}$$

This is basically an Euclidean distance.

The harmony feature is based on the edge length histogram, the color histogram and the dominant color feature. It has three bins for the amount of activity in an image, the number of color gradations and the color type (warm, cold, grey-scale). The activity in an image is measured as the variance of edge lengths. The MPEG-7 descriptor for the harmony feature is defined as follows.

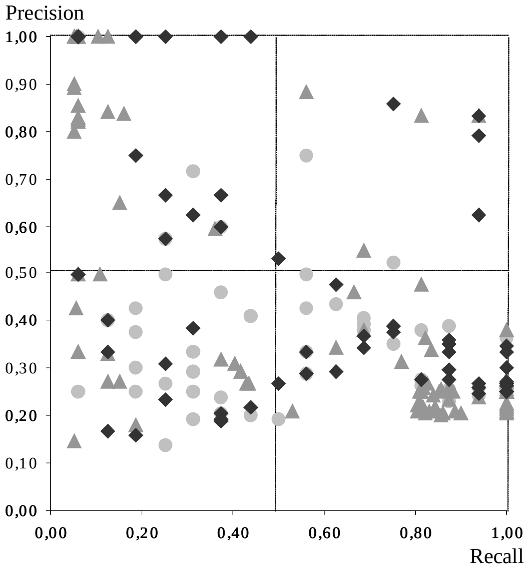


Figure 5. Experimental results for 300 queries. Triangles represent the query results for the low-level features, circles the results for the HWF and rhombs the results for queries on all feature classes.

```
<complexType name=”HarmonyFeature”>
   <element name=”Activity”
    type=”unsignedInt” use=”required”/>
   <element name=”ColorShades”
    type=”unsignedInt” use=”required”/>
   <element name=”ColorType”
    type=”unsignedByte” use=”required”/>
</complexType>
```

The distance of two descriptors *A=(act$_A$,cg$_A$,ct$_A$)* and *B* is measured with the following distance function.

$$d(A,B)=\sqrt{\frac{(act_A-act_B)^2+(cg_A-cg_B)^2+u(ct_A,ct_B)}{3}}$$

where *u()* is defined as for the distance function of the dominant color feature.

The symmetry feature is based on the object description feature. It counts the number of symmetric objects (equal descriptions with equal size) and the number of repeated objects (equal descriptions with different size). We define the following Descriptor:

```
<complexType name=”SymmetryFeature”>
   <element name=”Symmetries”
    type=”unsignedInt” use=”required”/>
   <element name=”Repetitions”
    type=”unsignedInt” use=”required”/>
</complexType>
```

The distance of two descriptors *A* and *B* is measured with the same function as for the geometry feature.

All features (low-level and HWF) and a querying engine that is based on our Query Model concept ([1]) were implemented as Perl objects in our test environment. Perl was chosen because it allows rapid prototyping. The next section explains how we tested the HWL features and the results we got.

6 EXPERIMENTAL RESULTS

All experiments were done on a collection of 64 synthetic images. This collection consists of four groups with 16 similar images each. Figure 4 depicts the test database. Each group consists of two rows. The four groups (ground truth) are: images of forests (first and second row), images of houses (third and fourth row), images of faces (fifth and sixth row) and images of equipment (seventh and eighth row). Each image was constructed from a stencil with 14 basic icons in Microsoft Visio (the image collection and the icon stencil can be obtained from the authors). We chose this image collection because it is – although the images are synthetic – a hard test for the SFL concept and the HWL implementation. It is a hard test because these images do not contain much information and it is difficult to derive more information with high-level features than the powerful low-level features (color histogram, object description) already do.

The hypothesis of our experiments was that *using SFL reduces the impact of the semantic gap*. This was tested in the following way:

- The HWF defined above were used as an example of an SFL. We did 300 valid queries: 100 with the low-level features, 100 with the HWF features and 100 with all features. The parameters of these queries were selected automatically (query example, threshold parameters, see [1]).
- A query was defined as valid, if the result set was not empty. This was the only restriction in the automatic evaluation process.
- The reduction of the semantic gap was measured by the change in the quality of result sets. Quality was measured with recall and precision. The ground truth from above was used for evaluation.

Querying was done by selecting an example image from the given collection and setting threshold values for the used features. The thresholds are upper limits for the distance from an image to the query example. If an image exceeds the threshold for a certain feature, it is discarded from the querying process. The result set contains only the images with a distance (for every feature) to the query example that is not greater than the feature-specific threshold.

Figure 5 shows the results of all queries. Triangles represent the query results for the low-level features, circles the results for the HWF and rhombs the results for all features. We have split the diagram in four areas: *excellent* (recall and precision *>50%*), *precise* (recall *<=50%*, precision *>50%*), *complete* (recall *>50%*, precision *<=50%*) and *poor* (recall and precision *<=50%*). Only *5%* of all results lie in the excellent area, *10%* are precise, *15%* are poor and about *70%* are complete. That means, our system tends to optimize the recall.

Looking at the distribution of results reveals that the triangles form two clusters with *(recall, precision)* at *(80%,20%)* and *(10%,85%)*. That means, the low-level features produce extreme results with either high recall or high precision. The HWF results (circles) are about

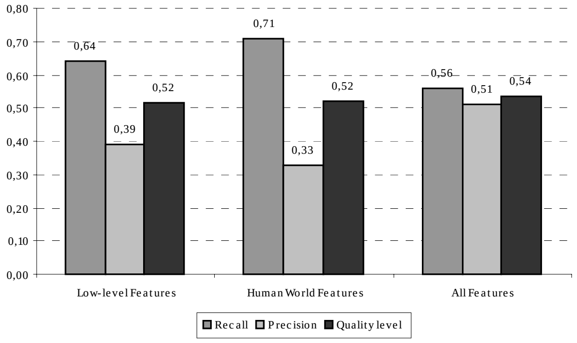


Figure 6. Quality comparison of evaluated methods. Using all features optimizes the quality level.

equally distributed in the poor and precise area. Most rhombs lie in the precise area with slightly better precision than the triangles. That means, using low-level features and HWF features together leads to more balanced results. Most results in the excellent area are rhombs.

Figure 6 summarizes the overall recall and precision (mean values over 100 tests each). The high-level features produce an excellent recall of *64%* with a poor precision of *39%*. The HWF features alone result in even more unbalanced results (*71%* and *34%*). Using all features reduces the recall but improves the precision.

The quality level in Figure 6 is the sum of recall and precision (for visualization it is divided by 2). It is a measure for the *maximum level* recall and precision can reach for a specific querying method and ground truth *independent from the query parameters*. The quality level for the method with all features is *54%*. This is a slight improvement of *2%* over the two basic methods. These results suggest that using HWF features refines the results of low-level features and balances the result set quality.

7 CURRENT AND FUTURE WORK

Next work on the HWF will include the development of more sophisticated versions of the descriptors and distance measures as well as additional tests on other image collections. In the future, we will try to base all HWF features on MPEG-7 image descriptors.

Additionally, we will define and investigate two further semantic feature layers: image creation artifacts (ICA) and chaotic image properties (CIP). ICA try to extract typical image errors that are originated in the photographing technology (digitized photos, video frames, etc.) or in the photographing task (shooting portrait photos, film scenes, etc.). For example, such a property could be color errors (derived from color histograms). These could be used to guess the age of an image. CIP extract chaotic elements of images (e.g. trees, flowers, etc.). They will be based on fractal theory and can be used to distinguish images of natural scenes.

8 CONCLUSION

In this paper we describe a novel approach to reduce the semantic gap problem of CBIR system. The basic idea is enhancing queries with high-level features that are based on low-level features. We have implemented a prototype for a feature class that describes human world properties. This feature class was tested in our test environment in 300 queries. The result was: using high-level features improves the quality of result sets by balancing recall and precision.

Our conclusion is that using semantic feature layers is reasonable when the used feature class suits the given querying problem (application domain). Otherwise it may even lead to a reduction of the querying performance. The semantic feature layer concept will be incorporated in the open VizIR project. Interested researchers are invited to join this project or use its results and deliveries for further CBIR research.

9 REFERENCES

[1] Breiteneder, C., and Eidenberger, H. A Retrieval System for Coats of Arms in Proceedings International Symposium on Multimedia Application and Distance Education (Baden-Baden Germany, 1999).

[2] Del Bimbo, A. Visual Information Retrieval. Morgan Kaufmann Publ., San Francisco CA, 1999.

[3] Eidenberger, H., and Breiteneder, C. A Framework for Visual Information Retrieval in Proceedings Visual Information Systems Conference (HSinChu Taiwan, March 2002), LNCS, Springer Verlag, 105-116.

[4] Furht, B., Smoliar, S.W., Zhang, H.: Video and Image Processing in Multimedia Systems. 2nd edn., Kluwer, Boston MA (1996).

[5] MPEG-7 standard documents Website. http://mpeg.telecomitalialab.com/standards/ mpeg-7/mpeg-7.htm

[6] Rational Unified Process Website. http://www.rational.com/products/rup/index.jsp

[7] Rui, Y., Huang, T., Ortega, M. and Mehrotra, S. Relevance Feedback: A Power Tool for Interactive Content-Based Image Retrieval. IEEE Transactions on Circuits and Systems for Video Technology, 8/5 (1998), 644-655.

[8] Semantic Web Website. http://www.semanticweb.org

[9] Smeulders, A.W.M., Worring, M., Santini, S., Gupta, A., and Jain, R. Content-Based Image Retrieval at the End of the Early Years. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22/12 (December 2000), 1349-1380.

[10] Smith, J.R., and Chang, S.F. VisualSEEk: a fully automated content-based image query system in Proceedings ACM Multimedia (Boston MA, 1996), ACM Press, 87-98.

[11] Unified Modeling Language Website. http://www.uml.org

[12] Wood, M., Campbell, N. and Thomas, B. Iterative Refinement by Relevance Feedback in Content-Based Digital Image Retrieval in Proceedings ACM Multimedia (Bristol UK 1998), 13-20.

Visual Similarity Measurement with the Feature Contrast Model

Horst Eidenberger*, Christian Breiteneder
Vienna University of Technology, Institute of Software Technology and Interactive Systems,
Favoritenstrasse 9-11, 1040 Vienna, Austria

ABSTRACT

The focus of this paper is on similarity modeling. In the first part we revisit underlying concepts of similarity modeling and sketch the currently most used VIR similarity model (Linear Weighted Merging, LWM). Motivated by its drawbacks we introduce a new general similarity model called Logical Retrieval (LR) that offers more flexibility than LWM. In the second part we integrate the Feature Contrast Model (FCM) in this environment, developed by psychologists to explain human peculiarities in similarity perception. FCM is integrated as a general method for distance measurement. The results show that FCM performs (in the LR context) better than metric-based distance measurement. Euclidean distance is used for comparison because it is used in many VIR systems and is based on the questionable metric axioms. FCM minimizes the number of clusters in distance space. Therefore it is the ideal distance measure for LR. FCM allows a number of different parameterizations. The tests reveal that in average a symmetric, non-subtractive configuration that emphasizes common properties of visual objects performs best. Its major drawback in comparison to Euclidean distance is its worse performance (in terms of query execution time).

Keywords: Visual Information Retrieval, Content-based Image Retrieval, Content-based Video Retrieval, Visual Similarity Measurement, Similarity Modeling, Linear Weighted Merging, Logical Retrieval, Feature Contrast Model, Boolean Retrieval, Vector Space Model

1 INTRODUCTION

Content-based Image Retrieval (CBIR) and Content-based Video Retrieval (CBVR) are two research directions of Multimedia (or Media Processing) systems that have been very active in the last couple of years. There is a trend to unify concepts and methods from CBIR and CBVR under one umbrella and the MPEG-7 standard is a first step in this direction. In this paper we will refer to both CBIR and CBVR as Visual Information Retrieval[4] (VIR). Essentially, VIR research has undergone three phases with different focus: (1) feature design and indexing methods, (2) user-interfaces and iterative refinement and (3) benchmarking (now). To overcome the big open problems of VIR[19] we think it being necessary to emphasize careful feature and similarity modeling.

The focus of this paper is on similarity modeling, which is probably the most neglected area of VIR research. First we sketch the currently most used VIR similarity model (Linear Weighted Merging, LWM) and point out its major weaknesses. Then we describe a new general similarity model called Logical Retrieval (LR) that offers more flexibility than LWM without suffering from its drawbacks. In the third step we integrate the Feature Contrast Model (FCM), developed by Tversky[20] into this environment. The FCM is based on psychological observations of human similarity perception. We integrate it in LR as a general method for feature-based distance measurement.

Utilizing FCM for VIR is not a new idea: it was first performed by Santini and Jain[15, 16], who built a unified similarity theory integrating geometric and set theoretic approaches. The FCM defines a family of distance measures that are very attractive for VIR: It is possible to define asymmetric similarity (e.g. "how similar is image A to B?" instead of "how similar are A and B?") and the model is generally less restrictive than distance functions based on the metric axioms (e.g. Minkowski distances).

The following Section 2 offers background information on similarity structures and distance measurement. Section 3 reviews similarity modeling in VIR, the LWM approach and describes LR as a more flexible model than LWM. Section 4 integrates the FCM in LR and investigates whether the integrated model still preserves the idea of Tversky's FCM. In Section 5 the FCM is implemented in a prototype as a general-purpose distance measure and tested on image data. Performance results are analyzed in comparison to Euclidean distance as a standard VIR distance measure.

* eidenberger@ims.tuwien.ac.at; phone 43 1 58801-18853; fax 43 1 58801-18898; www.ims.tuwien.ac.at

2 BACKGROUND

Subsequently, we define the term similarity structure as the fundamental concept for distance respective similarity measurement. Subsection 2.2 sketches distance measurement based on the metric axioms as it is usually used in VIR systems. Finally, Subsection 2.3 briefly describes alternative axiomatic systems for similarity structures including the FCM.

2.1 Similarity structures

Let *E* be an arbitrary set of objects. According to Sint[17] a similarity structure (or: a similarity measure) for the elements e_i of *E* is defined as a relation respective function over a set of pairs *ExE* of objects (represented as numerical feature vectors). The given measurements have to be somehow transformed into this relation. The list of possible similarity structures *S* over *ExE* includes[17]:

- S_1: *S* is an Euclidean distance over *ExE*. This measure assumes that feature space has Euclidean geometry (fulfills the metric axioms, see below).

- S_2: *S* is a metric over *ExE*. This measure makes no assumption on the geometric shape of feature space. S_2 is a generalization of S_1.

- S_3: *S* is symmetric and rational over *ExE*.

- S_4: *S* is a total or partial order of *E*.

These four are the most common similarity structures but of course, many more do exist. This definition spans an umbrella over a wide range of similarity understandings (visual, mathematical, psychological, etc.). In this paper we will investigate another method: the generation of a dichotomy of similar and not similar objects with dynamic borders over *E*.

2.2 Distance measurement based on the metric axioms

Usually, VIR similarity measurement follows the vector space model from information retrieval theory (e.g. in LWM, see Section 3.1). It is done by measuring the distances of feature vectors with distance functions and interpreting similarity as a point in an n-dimensional distance space. The vector space model is an applied similarity structure of type S_2. That means, it strongly relies on metric-based distance measurement. For distance measurement in (feature) vector spaces a certain type of geometry has to be considered. In VIR the feature space is usually considered to be of Euclidean shape. That means, distance measures *d()* fulfil four conditions (metric axioms)[16]:

1. Constancy of self-similarity:

$$d(f_A, f_A) = d(f_B, f_B) \tag{1}$$

for the feature vectors f_A and f_B of two stimuli *A* and *B* (in VIR: media objects). Psychological experiments have show that self-similarity is not always the case for human similarity perception[16].

2. Minimality:

$$d(f_A, f_B) \geq d(f_A, f_A) \tag{2}$$

3. Symmetry:

$$d(f_A, f_B) = d(f_B, f_A) \tag{3}$$

Like for the constancy of self-similarity, psychological experiments have turned out that humans do not always have a symmetric similarity perception.

4. Triangle inequality:

$$d(f_A, f_B) + d(f_B, f_C) \geq d(f_A, f_c) \tag{4}$$

Distance measures that fulfil the metric axioms are Minkowski distances, the Euclidean distance and the city block measure[16]. Experimental investigations during the last fifty years have turned out that metric axioms may be too restrictive for human similarity perception. The triangle inequality (in CBIR sometimes used for query acceleration) was even falsified[16, 20]. Newer theories as the ones sketched in the next subsection suggest a better representation of human similarity perception.

2.3 Alternatives for the metric axioms

According to Santini and Jain, Monotone Proximity Structures (MPS, a system of three distance axioms) could be used to replace the metric axioms with a less rigid system[16]. As can be easily shown, MPS suffers from severe inconsistencies. One of the axioms is the dominance axiom:

$$d(x_1 y_1, x_2 y_2) > \max\{d(x_1 y_1, x_1 y_2), d(x_1 y_1, x_2 y_1)\} \tag{5}$$

Here, two stimuli *A* and *B* are compared by distance function *d()* where *A* and *B* are represented by two-dimensional feature vectors *(x_1,y_1)* and *(x_2,y_2)*. For example, if the two features are the following predicates: (1) "*X* is a color image" and (2) "*X* has landscape spatial layout" (where *X* is an arbitrary stimulus) then a greyscale landscape media object can be represented by $x_1=0$, $y_1=1$ and a color landscape media object can be represented by $x_2=1$ and $y_2=1$. For this example, the dominance axiom has to be written as:

$$d((0,1),(1,1)) > \max\{d((0,1),(1,1)), d((0,1),(1,1))\} \equiv d((0,1),(1,1)) > d((0,1),(1,1)) \tag{6}$$

Obviously, no distance function *d()* exists for which equation 6 holds.

In comparison to the metric axioms and MPS, FCM is not a geometric but set-theoretic approach[20]. Basically, the idea is measuring the similarity of two stimuli (represented by feature vectors *X* and *Y*) with the formula in equation 7 (*f()* is a monotone increasing function and the non-negative parameters α, β determine, whether *s()* is symmetric (α=β) or asymmetric (else) and subtractive (α>*0* or β>*0*) or non-subtractive (else)).

$$s(X,Y) = f(X \cap Y) - \alpha\, f(X - Y) - \beta\, f(Y - X) \tag{7}$$

The FCM is very successful in representing the properties of human similarity measurement because it allows to distinguish between symmetric and asymmetric similarity perception and accounts for non-constant self-similarity. On the other hand it does not allow measurement with constant self-similarity and can only be applied to qualitative feature vectors (predicates). The latter is because of the logical operators used in *f()*.

To overcome the second drawback, Santini and Jain developed the Fuzzy FCM (FFCM) where the numerical elements of feature vectors are transformed to truth values and the logical operators (intersection and subtraction) are replaced by fuzzy equivalents[15, 16]. In addition, they developed a geometric equivalent for FFCM by replacing the fuzzy set operators by continuous functions. This formula somehow integrates geometric and set-theoretic similarity approaches. Santini and Jain[16] present a solution for the problematic fact that in FFCM feature vector elements are considered to be independent, which is not the case in reality. Unfortunately, this approach has not been integrated with the continuous FFCM formula. Additionally, these approaches suffer from the drawback, that the major degree of freedom (the selection of the formula *f()*) had to be abandoned in favour of unification. *f()* is always the fuzzy cardinality of the given truth values.

Maybe because of these problems, it seems that using FCM for VIR is not further investigated. For example, in Smeulders et al[18] it is not mentioned any more. We think this being regrettable, because the ideas of FCM are valuable for VIR distance measurement. In Section 4 we describe a new approach to incorporate FCM in a process-oriented environment for similarity measurement (LR). Next, we design a general model for the representation of human similarity perception.

3 SIMILARITY MODELING

In philosophy, similarity defines the relation between an object and its representation (Plato's 'image'). In VIR the philosophical similarity problem is usually subsumed under the term sensory gap[18]. Essentially, this term describes the loss of information in the (repeated) photographing process. This problem is accepted and not treated in VIR. Computer vision people are plowing this field and often, their plow is a 3D model (e.g. an active contour based on a deformable template, etc.).

Quite differently, 'similarity' in VIR is the relation of two images (stimuli). These may, but need not be representations of two objects (scenes, etc.). Thus, the VIR similarity problem is modeling the *real* similarity perception, humans have developed since they are able to use sticks for drawing in the dust. This problem has been investigated mostly by psychologists (perception theory, Gestalt theory). Researchers in other areas of work use the term 'similarity' as well but – as pointed out above – mean something different than (human perception of) visual similarity (e.g. mechanics: similarity of machines), have a more strict similarity concept (e.g. mathematics: similarity of triangles) or investigate the similarity of abstract representations of objects or images (e.g. physics: similarity in thermo-dynamics, medicine: homeopathy theory). Somehow logical, the most similar meaning of 'similarity' is used in biology for categorization of species. In this section we examine the standard VIR similarity model and develop a new one (Subsection 3.2) that is more suitable for human similarity perception.

3.1 Standard VIR similarity model

The usual approach for VIR similarity measurement is called Linear Weighted Merging (LWM) and has the following form[18] (generalized):

$$s_A(F,E)=\frac{\sum_{i=1}^{|F|}\left(w_i\sum_{j=1}^{|E|}u_{i,j}h_i\left(d_i\left(f_{i,A},f_{i,j}\right)\right)\right)}{\sum_{i=1}^{|F|}w_i\sum_{j=1}^{|E|}u_{i,j}} \qquad (8)$$

$s_A(F,E)$ is the average dissimilarity of stimulus *A* related to the used set of features *F* and the query examples in set *E*. The w_i are the weights for the features, $u_{i,j}$ is a binary matrix of size $|F|x|E|$ that contains a '1' for each combination of feature and query example that is used in the query. $h_i()$ can be any linear or exponential transformation of the distance values $d_i(f_{i,A}, f_{i,j})$ for feature *i*, stimulus *A* and query example *j*. Typical (accepted) transformations are identity and negative exponential transformation[18]:

$$h_i(d)=e^{-d} \qquad (9)$$

This numerator is standardized by the denominator: the number of dimensions of distance space. Because the denominator represents a linear transformation it is often omitted. In this case we do not call $s_A()$ a similarity but a position value, because the $s_A()$ for all objects A are a partial order over the given object collection. Like the distance functions, $s_A()$ is a similarity structure (in this case of type S_4, see Subsection 2.1). Usually, the most similar objects have the smallest position values. If the second transformation (equation 9) is used, the most similar stimuli have the highest position values. The linear weighted merging formula implies that all distance measures are standardized to the same interval (usually [0,1]). Ideally, they should have the same distribution as well. The weights are usually provided by the user and usually sum up to 1.

The LWM formula does not measure the distance of an object to the origin of distance space! It is just a linear combination of distance values. One argument against this formula is that most features are (of course) not linearly related. The fundamental law of Gestalt Theory is a generalization of this fact: the whole is more than its elements. In the formula above, the maximum of derivable information is always the sum of the elements.

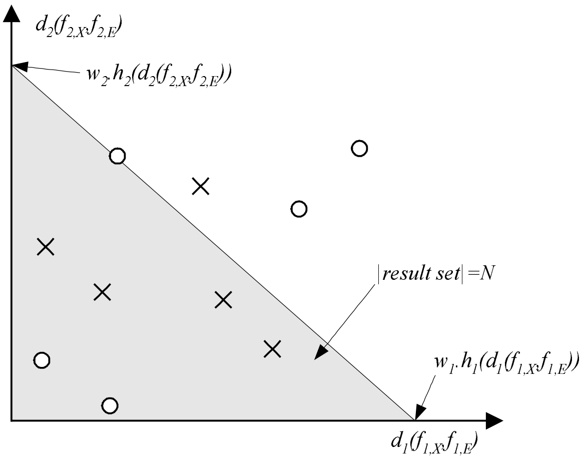

$d_2(f_{2,X},f_{2,E})$
$w_2 \cdot h_2(d_2(f_{2,X},f_{2,E}))$
$|result\ set| = N$
$w_1 \cdot h_1(d_1(f_{1,X},f_{1,E}))$
$d_1(f_{1,X},f_{1,E})$

Figure 1: Similarity definition in distance space by LWM. The selected subspace is always of triangular shape. Its size depends the result set size and the elements in the queried collection. The slope of the border is determined by the weights. Thus it is impossible to sort out all irrelevant elements in iterative refinement.

One could argue, that for testing it is sufficient to know the ground truth for the query examples and use this formula to find a lower bound for the quality of a retrieval method. The ground truth according to one element is the number of similar elements in a collection. For obvious reasons this is not true. LWM just generates a partial order over all elements of a collection. To be usable for retrieval the user has to specify a number *N* for the number of similar elements in the result set. Knowing the ground truth *G* (for the given examples) per se influences the selection of *N*. If *G* is selected greater than *N* the recall improves and if *G* equals *N* the precision improves. Additionally, it is not clear, how the ground truth can be found for tests, where using multiple example objects is necessary. Finally, knowing the ground truth and using LWM is not enough to judge the quality of a method (e.g. for feature extraction). In this case it would be necessary to know the distribution of objects in distance space as well.

The next argument follows a similar direction. Usually, iterative refinement by relevance feedback is used to reduce the semantic gap[18]. Consequently, in the past a lot of research effort has been invested into relevance feedback algorithms. Such algorithms can only be successful if the similarity model of the underlying query engine is flexible enough to represent the user's intention. For example, we use two features f_1 and f_2 and assume a media collection with ten elements. The distribution of the elements in distance space is shown in Figure 1. The query example(s) define(s) the origin. The ground truth of this collection is that 5 elements are similar (shown as *o*) and the 5 other are not (depicted as *x*).

If we use the formula above, we have two parameters that can be manipulated during relevance feedback: the query examples and the weight vector. Anyway, in distance space the result space is always the simplest simplex (a triangle in 2D, a tetrahedron in 3D, etc.), defined by the weights. Thus in the situation above it is impossible to retrieve all similar elements without retrieving the non similar as well. This allows two possible conclusions: (1) the situation above can never occur. Human similarity judgment can never result in such a ground truth with this element distribution or (2) the LWM formula is – as a similarity model – not suitable for VIR. We tend to the second explanation. In the next subsection we will introduce a more flexible model.

3.2 Logical Retrieval

McLuhan writes that images are just illusions while only film (or video) is able to transport visual content appropriately[12]. His statement covers the simple observation that nothing exists without time and that *time means change*. We think that there is deep truth in this statement and derive that similarity measurement based on visual information should not be static but a dynamic process – as it is for human beings.

Logical Retrieval (LR) is based on observation of human behavior. When people are arguing their visual similarity perception they do not do this by making general comments but by pointing out certain aspects and details and stressing their remarkable analogy. These aspects are the features in the querying process. This view of similarity is very old. It

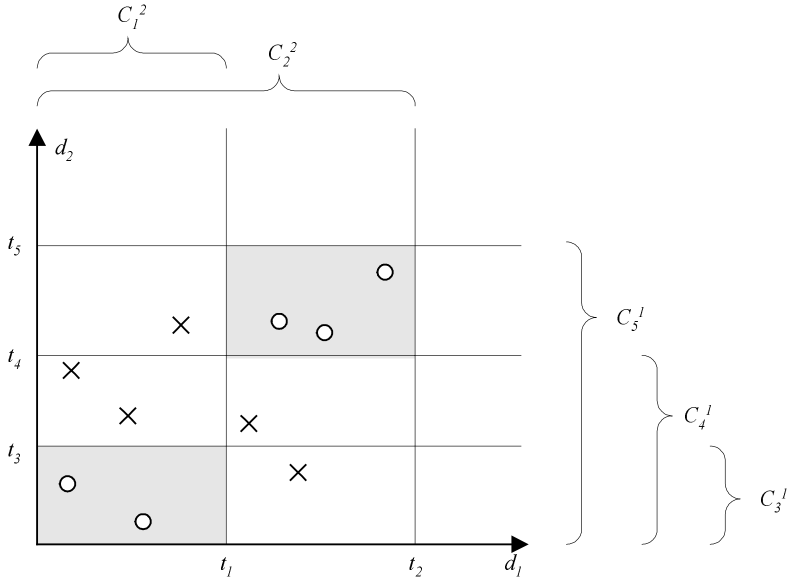


Figure 2: Similarity definition in distance space with logical expressions. The LR approach is a suitable similarity model because it allows every combination of elements. Thus it can represent each possible similarity perception.

was introduced by Aristotle, who saw similarity, when two objects had most or the most important properties, in which they could differ, in common[3]. Derived from this, similarity can be formally defined as the correspondence of objective measurable elements of complex objects or of their physical neighborhood[3]. For example, imagine cartoon figures. The perceived similarity of cartoon figures with real persons comes from extracting and imitating their major features (physique, motion, etc.).

The idea behind LR is simple. It should be a vehicle (model) that allows the selection of every possible combination of elements from a given collection. Thus it does not make any assumption, gives the user full control over the retrieval process and supports every thinkable similarity perception. The standard argument against this technique is: how should the average user be able to handle such a system? To the authors' belief (and experiences) this argument makes little sense at this point, because this is just a user-interface problem, while we are searching for a suitable model for similarity definition. We think that persisting on a very limited model that is easy to handle does not make much sense, if the overall problem of VIR are recall and precision rates of less than 40%.

We define LR similarity measurement as a two-step process[8]. In the first step (micro-level) feature vectors are mapped to points in distance space. Distance space is defined as the vector space that is derived by measuring the distance of media objects to given query examples with distance functions (micro-level similarity measurement). It has one dimension for each unique combination of distance measure and reference stimulus. In the second step (macro-level) the user defines his similarity perception as a logical expression. The logical expression consists of conditions C_i^j of the form given in equation 10. The parameter t_i is a threshold for the maximum distance of a media object for distance space dimension d_j.

$$d_j \le t_i \tag{10}$$

A media object is added to the result set, if the query expression evaluates to *true* for its distance values. This expression is then refined in an iterative process. We have developed GUI methods where the user need not define the expression directly but implicitly by selecting and moving media objects in a 3D user interface[5]. The set of similar objects in the example above (see Figure 1 in Subsection 3.1) could be described by the following expression (see Figure 2):

$$Query = C_1^2 \wedge C_3^1 \vee \left(C_2^2 \wedge \neg C_1^2\right) \wedge \left(C_5^1 \wedge \neg C_4^1\right) \tag{11}$$

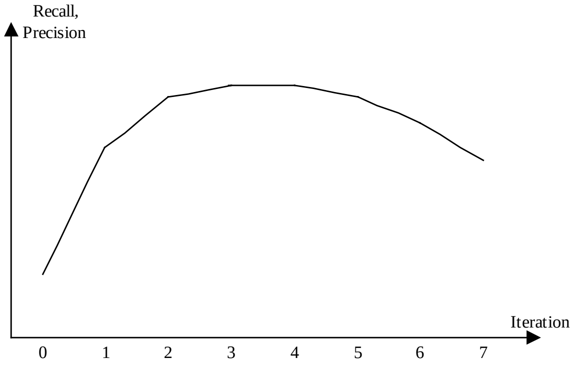


Figure 3: Typical development of recall and precision in iterative refinement by relevance feedback[11]. We propose that – at least partially – this characteristic comes from the user's limited influence on the LWM querying process.

The example makes clear that LR expressions can describe each possible selection in distance space and therefore represent each type of similarity perception. LR is not a new invention. The idea is partially based on logical expressions as in Boolean Retrieval, although it has nothing to do with the Boolean Retrieval method in information retrieval. It is the integration of logical expressions with vector spaces, optimized for visual similarity perception. Our approach to Logical Retrieval is a generalization of our earlier Query Model concept[1]. In addition, we are aware of a second approach that results in a similar concept from a different starting point[13]. Next we will investigate the structure of LR expressions in VIR.

Every query expression can be defined in disjunctive normal form (*OR*-connected terms as in the example above). In this style each *AND*-connected term of an expression describes a n-dimensional cube of elements and should contain at most one condition for each dimension of distance space. Each such n-dimensional cube can be interpreted as a cluster (these clusters correspond to the Query Models we introduced in earlier publications[1]). The Logical Retrieval process is essentially describing clusters of similar elements. According to information retrieval theory, such clusters exist for each collection of elements with reasonable size.

If we interpret LR as disjunctive concatenating of cluster expressions we can do the following simplifications:

1. The *NOT* can be integrated into the cluster terms. Instead of $NOT\ C_i^j$ we can write:

$$NOT C_i^j \equiv \neg\left(d_j \leq t_i\right) \equiv d_j > t_i \equiv \overline{C_i^j} \tag{12}$$

2. Two disjunctive conditions on a distance space dimension x in a single cluster term can be integrated into a single expression and written as follows:

$$C_i^x \wedge \overline{C_j^x} = d_x \leq t_i \wedge d_x > t_j \equiv t_j < d_x \leq t_i \equiv C_{i,j}^x \tag{13}$$

We call $C_{i,j}{}^x$ a cluster restriction. Each *AND*-connected expression of cluster restrictions describes an either finite (one cluster restriction for each dimension of distance space) or (ideally) infinite cluster (less cluster restrictions than dimensions).

Next we will point out the major advantages and drawbacks of LR. In opposition to the arguments above, LR has a very positive consequence for user-interface design in VIR. It is intuitive and easy to visualize. Queries are usually represented by selecting example elements. In LR this selection can be made by dragging rectangles around two-dimensional views of examples in feature space or distance space. Views can be created by selecting arbitrary features for the X- and Y-axis. Obviously, these rectangles can be directly transformed into an LR expression. Iterative refinement can be performed in the same way. Abstract annotations like 'very similar' are not needed[5].

By now, it is accepted that iterative refinement based on LWM is limited. It leads to improvements in recall and precision in the first four to five cycles[11]. Then the quality of the results begins to decrease. Figure 3 describes the typical development of recall and precision over multiple refinement iterations. We think that that the reason for this is the limited influence of the user in a querying process based on LWM. Four to five cycles is exactly the time it takes to adjust the weights for a few features to the optimum values. In LR iterative refinement means finding additional clusters and optimizing their borders. This should at least stretch the typical refinement curve and lead to a higher quality peak in the refinement process.

In addition, LR has a nice side-effect on query execution time. Within each *AND*-connected cluster, the result set of a query is independent from the order of the cluster restrictions. An algorithm that sorts the conditions in a way that those, which sort out most elements and/or use the fastest distance functions, are used first in the querying process, would lead to significant query acceleration. We have presented the design and implementation of such an algorithm[7]. It reduces the average query execution time in our test environment by 66% (in comparison to a QBIC system[9, 10] with the same feature classes and distance functions).

4 INTEGRATION OF THE FEATURE CONTRAST MODEL

In the LR model, we would like to incorporate the ideas of the FCM: asymmetry and non-constant self-similarity. Even though the standard FCM works on binary predicates that are related to the conditions from above (equation 10), we think that that the distinction between symmetric and asymmetric queries belongs to the micro-level and therefore FCM should be incorporated as a (general-purpose) distance measure. To do this, we are not going to interpret numeric feature vector elements fuzzy or probabilistic, because we cannot give good reasons for such an interpretation. Instead, we use the following substitute for continuous data: the similarity function *s()* is defined as in equation 7 and the set operators are replaced by suitable continuous functions. The intersection operator is replaced by one of the two following functions:

$$\text{inter}_c(X,Y) = (a_i) \, where \, a_i = \begin{cases} \frac{x_i + y_i}{2} & if \; \max - \frac{x_i + y_i}{2} \le \varepsilon_1 \\ 0 & else \end{cases} \tag{14}$$

$$\text{inter}_d(X,Y) = (a_i) \, where \, a_i = \begin{cases} \max - |x_i - y_i| & if \; |x_i - y_i| \le \varepsilon_1 \wedge \max - x_i < \varepsilon_1 \\ 0 & else \end{cases} \tag{15}$$

max is the maximum distance of feature vector elements x_i and y_i and ε_1 approaches *0*. $inter_c$ emphasizes common properties of *X* and *Y* while $inter_d$ emphasizes their differences. In the tests in section 5 we will try to find out which formula performs better for continuous data. For the subtraction operator we use the function from equation 16:

$$\text{sub}(X,Y) = (a_i) \, where \, a_i = \begin{cases} x_i - y_i & if \; \max - (x_i - y_i) \le \varepsilon_2 \\ 0 & else \end{cases} \tag{16}$$

This model should preserve the idea of the FCM. The intersection operator selects properties that are present in both stimuli to a similar extent and the subtraction operator selects properties that are present just in *X*. For *f()* we suggest to use formula 17. The definition of *f()* is not part of Tversky's FCM model. Therefore we do not include it in the continuous model either.

$$f'(X) = \frac{\sum_i a_i}{i} \, where \, a_i = \begin{cases} val(t) & if \; x_i \ne 0 \\ 0 & else \end{cases} \tag{17}$$

t is the determining parameter of *f()*. *val(t)* returns the value of t: the constant's value if t is a constant (e.g. *val(2)=2*) or the variable's value if *t* is a variable (*val(x)=2* if *x=2*). Thus, if *t=1, f()* is the cardinality of relevant properties (equivalent to FCM). If $t=x_i$, *f()* measures the mean of the difference of all relevant properties of two stimuli (either both present or only one present).

Two problems are connected to this approach: how to choose ε_1 and ε_2, and how to set the parameters α and β that determine, if the FCM is symmetric or asymmetric. We suggest to base the selection of ε_1 and ε_2 on statistical analysis of the given feature data (with $\varepsilon_i << max$!) and to implement the setting of α and β by a switch in the user interface that allow the specification of symmetric ($\alpha=\beta=0$) and asymmetric queries as well as subtractive ($\alpha>0$ or $\beta>0$) and non-subtractive queries. Below, in Section 5 we will show how this continuous FCM model was implemented in a prototype.

Next we investigate the behaviour of the continuous FCM for binary predicates. Ideally, the continuous FCM should produce the same results for binary predicates as the original FCM. The following tables show all possible relations for two predicate vectors $X=(x_i)$ and $Y=(y_i)$. The intersection (*inter*) should be '1' only if predicate *i* is present both in *X* and *Y*. The subtraction (*sub*) should be '1' if a predicate is present just in *X*.

x_i	y_i	*inter*	$\max-\frac{x_i+y_i}{2}\le\varepsilon_1$	$inter_c$	$\lvert x_i-y_i\rvert\le\varepsilon_1\wedge\max-x_i<\varepsilon_1$	$inter_d$
1	1	1	*1-(1+1)/2=0 < ε_1... true*	1	*|1-1|=0 < ε_1 ... true, 1-1 < ε_1... true*	1
1	0	0	*1-(1+0)/2=0,5 < ε_1... false*	0	*|1-0|=1 < ε_1... false*	0
0	1	0	*1-(0+1)/2=0,5 < ε_1... false*	0	*|0-1|=1 < ε_1 ... false*	0
0	0	0	*1-(0+0)/2=1 < ε_1... false*	0	*|0-0|=0 < ε_1 ... true, 1-0=1 < ε_1... false*	0

Table 1. Evaluation of intersection operator for binary predicates.

x_i	y_i	*sub*	$\max-(x_i-y_i)\le\varepsilon_2$	x_i-y_i
1	1	0	*1-(1-1)=1 < ε_2... false*	0
1	0	1	*1-(1-0)=0 < ε_2... true*	1
0	1	0	*1-(0-1)=2 < ε_2... false*	0
0	0	0	*1-(0-0)=1 < ε_2... false*	0

Table 2. Evaluation of subtraction operator for binary predicates.

For binary predicates $max=1$. If we set $\varepsilon_1<0,5$ and $\varepsilon_2<1$ the tables show that all suggested operators perform as desired. That means, if the continuous operators are fed with binary predicates the behaviour of the model is exactly the same as for Tversky's model. This is independent from the selection of *f()*. In the next section we will investigate if the operators are suitable for practical use in VIR systems.

5 TESTS AND RESULTS

Goal of the tests is to measure the performance of the continuous FCM as a *general-purpose* distance measure in comparison to another standard distance measure: the Euclidean distance. The principal superiority of the LR approach over LWM has already been shown in other publications[1, 8, 13]. We have implemented the FCM models from section 4 in a Perl prototype. Perl was chosen because it offers powerful data processing capabilities and allows rapid prototyping. Additionally, by now powerful image analysis libraries exist for Perl.

The selection of the test procedure was problematic. Normally, new VIR methods are tested by selecting a large image library (e.g. the Corel-library), defining a ground truth based on semantic image properties (e.g. images of flowers, images of cars) and evaluating the new method by a reasonably large number of queries with the recall and precision measures[18]. The general problem with this procedure is the following: based on the LR model as a flexible similarity measurement process it is always possible to maximize recall and precision at the same time. Additionally, here we want to measure the performance of FCM as a distance measure on the micro-level. If the performance was weak, the overall system performance in terms of recall and precision could still be good because of LR's flexibility. Especially, the characteristic advantages of FCM cannot be measured with such a procedure.

Because of these considerations we gave up the idea of an evaluation based on recall and precision and developed the following test pattern. We compare the cluster structure of the distance spaces created on the micro-level by the used distance measures. A cluster is defined as a group of objects that belong to the same semantic group as the query example (defined by the ground truth). In detail we are doing the following (in a reasonable number of repetitions).

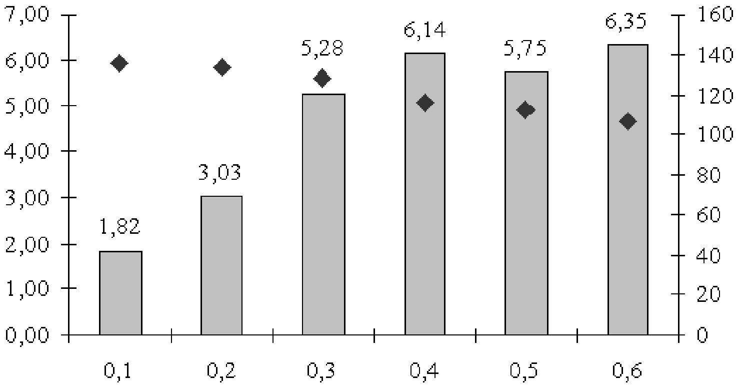


Figure 4: Average cluster size (bars, left Y-axis) and cluster pollution (rhombs, right Y-axis, in percent of correct cluster members) for the symmetric non-subtractive FCM (α=β =*0*) depending on ε_1 (X-axis).

Based on a given set of feature vectors and a randomly selected query example we generate two distance spaces: one with FCM and the other with Euclidean distance. In these spaces we identify all clusters of objects that belong to the group of the query example and calculate the average group size and the group size variance. Average and variance of the group size are meaningful measures for the quality of a distance function in LR, because a good distance measure should generate as large as possible (and therefore as less as possible) clusters that can then be easily tracked on the macro-level. The major problem of this approach is the identification of clusters in an n-dimensional distance space (derived from n features). This is non-trivial but can be avoided by taking all features together and measuring the distance as a *whole*. The generated distance space is then one-dimensional and the test for the general-purpose distance measures is even harder (especially for the FCM) because they have to integrate arbitrarily related features.

For the tests we use a collection of 444 images of coats of arms. The images are synthetic (painted, not scanned or photographed) and have been described in earlier publications[1, 2]. Like Santini and Jain[15] we think that computer-based similarity assessment should be pre-attentive and therefore VIR benchmarks should be based on pre-attentive similarity judgement as well. This can be achieved by using media collections with *abstract* content for evaluation. Unfortunately, we are not aware of any visual media collection with really abstract content. Therefore we think that using the coats of arms library instead is a good compromise, because coats of arms carry no inherent visual meaning. Even though the elements of arms have precisely defined semantics the visual image itself has no meaning at all (except some ordinaries like horses, crowns, etc.). To select query examples and identify clusters we need a ground truth. Based on the visual impression we built a pre-attentive ground truth of four groups of images with similar colors, layouts and textures. The group size varies from 18 to 24 images. Finally, for the distance calculation we need feature vectors. We use the features from our coats of arms CBIR system[1, 2]. These include color histograms (global and localized) and other color features, object features (contours, etc.), image symmetry features and application-specific features (coats of arms segmentation, etc.). Each of the 444 images is represented by a feature vector with 58 elements.

Testing FCM as a general-purpose distance measure we want to clarify the following question: are the characteristics of the FCM as a tailor-made similarity measure still relevant in the LR model? Our hypothesis is: yes. We try to answer this question with three tests: (1) Performance comparison of FCM with the $inter_c$ intersection operator to FCM with the $inter_d$ operator, (2) comparison of symmetric FCM without consideration of features that are only present in one stimulus (α=β=*0*, non-subtractive) to asymmetric and/or subtractive FCM, and (3) comparison of the best FCM model to the Euclidean distance. To optimize FCM the optimal values for the parameters *t*, α, β, ε_1 and ε_2 have to be found. In summary we run 217000 queries: 1000 on Euclidean distance (no parameters), 72000 on FCM with $inter_d$ (6 parameters) and 144000 on FCM with $inter_c$ (4 parameters, *f()* with *t=1* was not evaluated, see below).

The comparison of $inter_c$ and $inter_d$ lead to very clear results. FCM with $inter_c$ (emphasizes common features) was in every single test better than FCM with $inter_d$ (emphasizes differences) with equal parameters. While the average number of clusters for FCM with $inter_c$ is most times less than 9 elements it is nearly always higher than 9 for FCM with $inter_d$ intersection operator. That means, in average the objects of the query examples group fall in more than 9 clusters in distance space. Consequently, $inter_d$ was not considered in the rest of the evaluation. Next we tried to

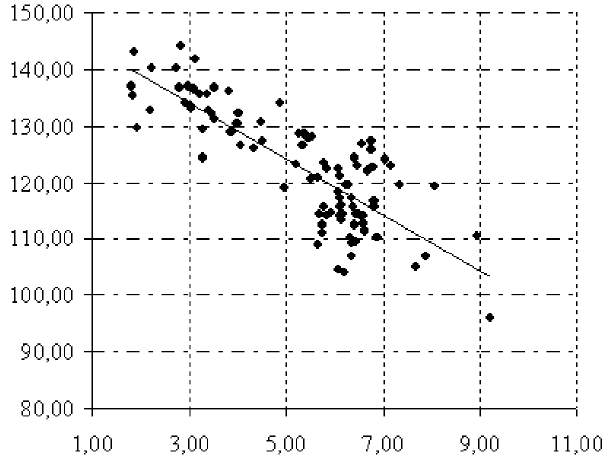


Figure 5: Correlation of average cluster size (X-axis) and average cluster pollution (Y-axis) for symmetric non-subtractive FCM. The correlation is not significant (81,9%).

optimize the parameter *t* of FCM and found out the following: $t=x_i$ clearly outperforms $t=1$. That means, function *f()* with continuous predicate interpretation is always better than *f()* with binary interpretation (and FCM with equal parameters). Thus $t=1$ was not considered in the further tests as well.

To find out whether symmetric non-subtractive FCM is better than asymmetric and/or subtractive FCM we optimized the parameters for each of the four possible combinations. This revealed that in average symmetric non-subtractive FCM performs best. Figure 4 shows the results for $\alpha=\beta=0$. The bars show the average number of clusters in distance space depending on ε_1 while the rhombs show a new phenomenon that we call cluster pollution. First, we concentrate on the average number of clusters. We can see that it decreases with decreasing ε_1 from 6 (about constant for $\varepsilon_1 > 0,3$) to about 2 ($\varepsilon_1 = 0,1$). This is because if ε_1 is set smaller, less predicates are used to judge the similarity of objects. From the small number of clusters we can conclude that FCM has an inherent 'intelligence' to select the *right* properties and using lower epsilons results in a better cluster structure.

Cluster pollution means that in a cluster of adjacent objects from the queried group, false objects exist that have *exactly* the same distance value as one of the cluster members but do not belong to the clustered group (according to the ground truth). Such false objects cannot be identified with LR expressions and therefore have to be treated as cluster members. Generally, it should be very unlikely that two objects come out at exactly the same point in distance space but because of the nature of FCM (only some predicates are used, controlled by ε_1) this can happen. In Figure 4 we see that cluster pollution decreases with increasing ε_1 from 140% to 100%. That means for $\varepsilon_1 = 0,6$ each cluster contains about the same number of correct and false members. Of course, to a large degree this can be explained by the one-dimensional distance space. If distance space had more dimensions the clusters would be less polluted. Still, cluster pollution is a consequence of using FCM. Clusters in an Euclidean distance space are not polluted (see below). The results in Figure 4 may suggest that a lower number of clusters (gained by lowering ε_1) corresponds with higher cluster pollution. For clarification we calculated the correlation of average cluster size and cluster pollution. Figure 5 shows the results. There is no significant correlation between the cluster size and cluster pollution. The correlation coefficient is lower than 82%.

The results for asymmetric and/or subtractive FCM can be seen in Figure 6. In this case, either α, β or both are greater 0 and therefore the results depend on ε_1 and ε_2. The left diagrams show the average number of clusters. Black areas (combinations of ε_1 and ε_2) mark results of average 2 clusters (1,5-2,5 clusters of correct objects in distance space), dark grey results of average 3 clusters (2,5-3,5), and so on. The right diagrams show the average cluster pollution. Black areas have a cluster pollution of average 135%, dark grey of 130%, et cetera. The first row of diagrams shows the results for FCM with $\alpha=1$ and $\beta=0$. In this case all features are taken into account that exist in both objects or only in the query example. The second row of diagrams shows the results for FCM with $\alpha=0$ and $\beta=1$. These two FCM configurations are asymmetric and subtractive. The third row of diagrams shows the results for FCM with $\alpha=1$ and $\beta=1$. This configuration is symmetric and subtractive. We have only investigated these cases but not linear combinations between them, because these configurations are extreme cases and the results of intrapolated

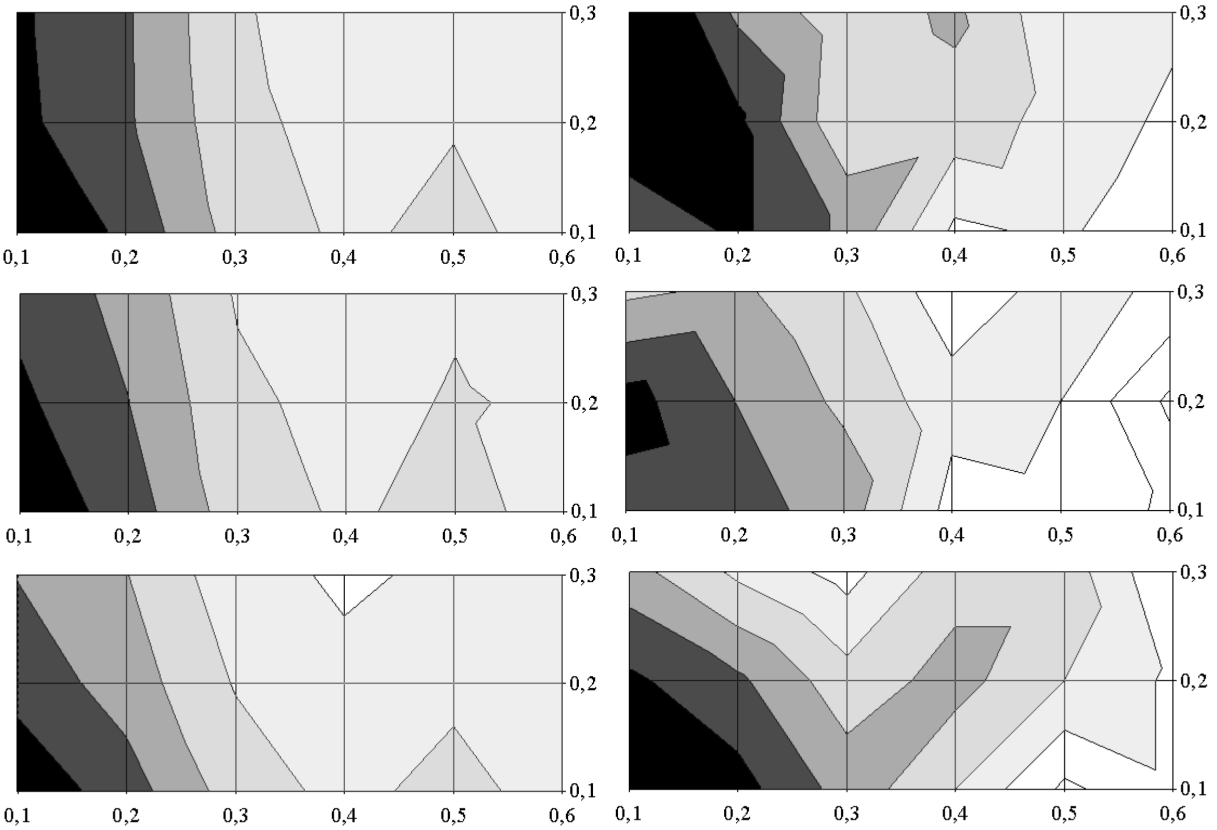


Figure 6: Cluster size (left column) and cluster pollution (right column) depending on ε_1 (X-axis) and ε_2 (Y-axis). First row: *α=1*, *β =0*, second row: *α=0*, *β =1*, third row: *α=1*, *β =1*.

configurations would be just intrapolations of the extreme results.

From the diagrams we can induce the following observations: the FCM with *β=0* is in the average better than the FCM with *β=1*. That means, using the information of features that are only present in the compared object *Y* does not improve the results. For visual information this is intuitive: non-similar objects can have a vast number of different features. Using them in a query is misleading and can hardly be supported by arguments. Additionally, we can see from the first row of images that for very low epsilons there is an area with a lower cluster pollution. This supports our argumentation that bigger clusters are not just a trade-off for higher cluster pollution.

In comparison to Euclidean distance (applied to the same test data) we see that FCM performs much better. Euclidean distance generates clusters of about two members. That means in average the queried image collection in distance space falls in 10 clusters. This is much worse than for FCM (2-8 clusters). On the other hand, Euclidean distance has two advantages: the resulting clusters have no pollution (because all features of an object are taken into account for distance measurement) and Euclidean distance is faster than FCM. The relationship in query execution time for the same test data on the same system is about 3:1. That means an FCM query takes about 3 times as long as an Euclidean query.

6 CONCLUSION

This paper reviewed similarity models for Visual Information Retrieval (VIR) and introduced the Feature Contrast Model (FCM) for VIR distance measurement. In the first part underlying concepts of similarity modeling were revisited and the standard VIR model was sketched. Motivated by the drawbacks of this model the more flexible Logical Retrieval model (LR) was introduced. According to this model, distance measurement is reduced from the central element of similarity measurement to a less important role. It should help to organize similar objects in distance space in a way that they are easy to find in an iterative querying process. In the second part the FCM, developed by psychologists to explain human peculiarities in similarity perception, was integrated in LR as a general-purpose distance measure. To do that a continuous model of FCM was developed. This model was tested on an abstract image library with a pre-attentive ground truth to judge its performance and find out the optimal parameterization.

The results show that FCM performs (in the LR context) better than Euclidean distance. Euclidean distance was used for comparison because it is used in many VIR systems and is based on the (questionable) metric axioms. FCM minimizes the number of clusters in distance space. Therefore it is the ideal distance measure for LR. FCM allows a number of different parameterizations. The tests revealed that in the average a symmetric, non-subtractive configuration that emphasizes common properties of visual objects performs best. Its major drawback in comparison to Euclidean distance is its worse performance (in terms of query execution time).

In future work we will try to improve the performance of FCM. Additionally, we will develop heuristics for FCM configuration for various kinds of feature data (setting ε_1, ε_2 and α). To do this, we will integrate FCM in the VIR project VizIR. VizIR aims at developing an open framework for VIR[6]. Interested researchers are invited to contact the authors for more information.

7 REFERENCES

1. C. Breiteneder, H. Eidenberger, "A Retrieval System for Coats of Arms", *Proceedings International Symposium on Multimedia Application and Distance Education,* Baden-Baden, 1999 (available from http://www.ims.tuwien.ac.at/~hme/papers/isimade1999.pdf).
2. C. Breiteneder, H. Eidenberger, "Content-based Image Retrieval of Coats of Arms", *Proceedings IEEE International Workshop on Multimedia Signal Processing*, 91-96, IEEE, Helsingör, 1999.
3. W. Butollo, *Subjective and Objective Similarity in Verbal Learning*, Notring Verlag, Vienna, 1968 (in German).
4. A. Del Bimbo, *Visual Information Retrieval*, Morgan Kaufmann Publishers, San Francisco , 1999.
5. H. Eidenberger, C. Breiteneder, "A Framework for User Interface Design in Visual Information Retrieval", *Proceedings IEEE International Symposium on Multimedia Software Engineering*, IEEE, Newport Beach, 2002.
6. H. Eidenberger, C. Breiteneder, "A Framework for Visual Information Retrieval", *Proceedings Visual Information Systems Conference*, 105-116, Springer Verlag, HSinChu 2002.
7. H. Eidenberger, C. Breiteneder, "Performance-optimized feature ordering for Content-based Image Retrieval", *Proceedings European Signal Processing Conference*, EUSIPCO, Tampere, 2000 (available from http://www.ims.tuwien.ac.at/~hme/papers/eusipco2000.pdf).
8. H. Eidenberger, C. Breiteneder, "Visual Similarity Measurement in VizIR", *Proceedings IEEE Multimedia Conference*, IEEE, Lausanne, 2002.
9. M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, P. Yanker, "Query by Image and Video Content: The QBIC System", *IEEE Computer,* **28/9**, 23-32, 1995.
10. IBM QBIC Website, http://wwwqbic.almaden.ibm.com/, last visited: 18th October 2002
11. C. Leung, "Visual Information Search and Benchmarking", *Visual Information Systems Conference*, Springer Verlag, HSinChu, 2002 (Plenary Talk).
12. M. McLuhan, *Understanding Media*, McGraw-Hill Publishers, New York, 1964.
13. M. Ortega, R. Yong, K. Chakrabarti, K. Porkaew, S. Mehrotra, T.S. Huang, "Supporting Ranked Boolean Similarity Queries in MARS", *IEEE Transactions on Knowledge and Data Engineering*, **10/6**, 905-925, 1998.
14. Y. Rui, T.S. Huang, M. Ortega, S. Mehrotra, "Relevance Feedback: A Power Tool for Interactive Content-Based Image Retrieval", *IEEE Transactions on Circuits and Systems for Video Technology,* **8/5**, 644-655, 1998.
15. S. Santini, R. Jain, "Similarity is a Geometer", *Multimedia Tools and Applications*, **5/3**, 277-306, 1997.
16. S. Santini, R. Jain, "Similarity Matching", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **21/9**, 871-883, 1999.
17. P.P. Sint, *Similarity Structures and Similarity Measures*, Austrian Academy of Sciences Press, Vienna, 1975 (in German).
18. A.W.M. Smeulders, M. Worring, S. Santini, A. Gupta, R. Jain, "Content-based image retrieval at the end of the early years", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22/12**, 1349-1380, 2000.
19. J.R. Smith, S.F. Chang, "VisualSEEk: a fully automated content-based image query system", *Proceedings ACM Multimedia*, 87-98, ACM Press, Boston, 1996.
20. A. Tversky, "Features of Similarity", *Psychological Review*, 84/4, 327-352, 1977.

Distance measures for MPEG-7-based retrieval

Horst Eidenberger
Vienna University of Technology, Institute of Software Technology and Interactive Systems
Favoritenstrasse 9-11 – A-1040 Vienna, Austria
Tel. + 43-1-58801-18853
eidenberger@ims.tuwien.ac.at

ABSTRACT

In visual information retrieval the careful choice of suitable proximity measures is a crucial success factor. The evaluation presented in this paper aims at showing that the distance measures suggested by the MPEG-7 group for the visual descriptors can be beaten by general-purpose measures. Eight visual MPEG-7 descriptors were selected and 38 distance measures implemented. Three media collections were created and assessed, performance indicators developed and more than 22500 tests performed. Additionally, a quantisation model was developed to be able to use predicate-based distance measures on continuous data as well. The evaluation shows that the distance measures recommended in the MPEG-7-standard are among the best but that other measures perform even better.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *Information filtering, Query formulation, Retrieval models.*

General Terms

Algorithms, Measurement, Experimentation, Performance, Theory.

Keywords

Visual Information Retrieval, Content-based Image Retrieval, Content-based Video Retrieval, Similarity Measurement, Distance Measurement, Similarity Perception, MPEG-7.

1. INTRODUCTION

The MPEG-7 standard defines – among others – a set of descriptors for visual media. Each descriptor consists of a feature extraction mechanism, a description (in binary and XML format) and guidelines that define how to apply the descriptor on different kinds of media (e.g. on temporal media). The MPEG-7 descriptors have been carefully designed to meet – partially complementary – requirements of different application domains: archival, browsing, retrieval, etc. [9]. In the following, we will exclusively deal with the *visual* MPEG-7 descriptors in the context of media *retrieval*.

The visual MPEG-7 descriptors fall in five groups: colour, texture, shape, motion and others (e.g. face description) and sum up to 16 basic descriptors. For retrieval applications, a rule for each descriptor is mandatory that defines how to measure the similarity of two descriptions. Common rules are distance functions, like the Euclidean distance and the Mahalanobis distance. Unfortunately, the MPEG-7 standard does not include distance measures in the normative part, because it was not designed to be (and should not exclusively understood to be) retrieval-specific. However, the MPEG-7 authors give recommendations, which distance measure to use on a particular descriptor. These recommendations are based on accurate knowledge of the descriptors' behaviour and the description structures.

In the present study a large number of successful distance measures from different areas (statistics, psychology, medicine, social and economic sciences, etc.) were implemented and applied on MPEG-7 data vectors to *verify* whether or not the recommended MPEG-7 distance measures are really the best for any reasonable class of media objects. From the MPEG-7 tests and the recommendations it does not become clear, how many and which distance measures have been tested on the visual descriptors and the MPEG-7 test datasets. The hypothesis is that analytically derived distance measures may be good in general but only a quantitative analysis is capable to identify the *best* distance measure for a specific feature extraction method.

The paper is organised as follows. Section 2 gives a minimum of background information on the MPEG-7 descriptors and distance measurement in visual information retrieval (VIR, see [3], [16]). Section 3 gives an overview over the implemented distance measures. Section 4 describes the test setup, including the test data and the implemented evaluation methods. Finally, Section 5 presents the results per descriptor and over all descriptors.

2. BACKGROUND

2.1 MPEG-7: visual descriptors

The visual part of the MPEG-7 standard defines several descriptors. Not all of them are really descriptors in the sense that they extract properties from visual media. Some of them are just structures for descriptor aggregation or localisation. The basic descriptors are Color Layout, Color Structure, Dominant Color, Scalable Color, Edge Histogram, Homogeneous Texture, Texture Browsing, Region-based Shape, Contour-based Shape, Camera Motion, Parametric Motion and Motion Activity.

Other descriptors are based on low-level descriptors or semantic information: Group-of-Frames/Group-of-Pictures Color (based on Scalable Color), Shape 3D (based on 3D mesh information), Motion Trajectory (based on object segmentation) and Face Recognition (based on face extraction).

Descriptors for spatiotemporal aggregation and localisation are:

MIR'03, November 7, 2003, Berkeley, California, USA.

Spatial 2D Coordinates, Grid Layout, Region Locator (spatial), Time Series, Temporal Interpolation (temporal) and SpatioTemporal Locator (combined). Finally, other structures exist for colour spaces, colour quantisation and multiple 2D views of 3D objects.

These additional structures allow combining the basic descriptors in multiple ways and on different levels. But they do not change the *characteristics* of the extracted information. Consequently, structures for aggregation and localisation were not considered in the work described in this paper.

2.2 Similarity measurement on visual data

Generally, similarity measurement on visual information aims at imitating human visual similarity perception. Unfortunately, human perception is much more complex than any of the existing similarity models (it includes perception, recognition and subjectivity).

The common approach in visual information retrieval is measuring *dis*-similarity as *distance*. Both, query object and candidate object are represented by their corresponding feature vectors. The distance between these objects is measured by computing the distance between the two vectors. Consequently, the process is independent of the employed querying paradigm (e.g. query by example). The query object may be natural (e.g. a real object) or artificial (e.g. properties of a group of objects).

Goal of the measurement process is to express a relationship between the two objects by their distance. Iteration for multiple candidates allows then to define a partial order over the candidates and to address those in a (to be defined) neighbourhood being *similar* to the query object. At this point, it has to be mentioned that in a multi-descriptor environment – especially in MPEG-7 – we are only half way towards a statement on similarity. If multiple descriptors are used (e.g. a descriptor scheme), a rule has to be defined how to combine all distances to a global value for each object. Still, distance measurement is the most important first step in similarity measurement.

Obviously, the main task of good distance measures is to *reorganise* descriptor space in a way that media objects with the highest similarity are nearest to the query object. If distance is defined minimal, the query object is always in the origin of distance space and similar candidates should form clusters around the origin that are as large as possible. Consequently, many well known distance measures are based on geometric assumptions of descriptor space (e.g. Euclidean distance is based on the metric axioms). Unfortunately, these measures do not fit ideally with human similarity perception (e.g. due to human subjectivity). To overcome this shortage, researchers from different areas have developed alternative models that are mostly predicate-based (descriptors are assumed to contain just binary elements, e.g. Tversky's Feature Contrast Model [17]) and fit better with human perception. In the following distance measures of both groups of approaches will be considered.

3. DISTANCE MEASURES

The distance measures used in this work have been collected from various areas (Subsection 3.1). Because they work on differently quantised data, Subsection 3.2 sketches a model for unification on the basis of quantitative descriptions. Finally, Subsection 3.3 introduces the distance measures as well as their origin and the idea they implement.

3.1 Sources

Distance measurement is used in many research areas such as psychology, sociology (e.g. comparing test results), medicine (e.g. comparing parameters of test persons), economics (e.g. comparing balance sheet ratios), etc. Naturally, the character of data available in these areas differs significantly. Essentially, there are two extreme cases of data vectors (and distance measures): predicate-based (all vector elements are binary, e.g. {0, 1}) and quantitative (all vector elements are continuous, e.g. [0, 1]).

Predicates express the *existence* of properties and represent high-level information while quantitative values can be used to measure and mostly represent low-level information. Predicates are often employed in psychology, sociology and other human-related sciences and most predicate-based distance measures were therefore developed in these areas. Descriptions in visual information retrieval are nearly ever (if they do not integrate semantic information) quantitative. Consequently, mostly quantitative distance measures are used in visual information retrieval.

The goal of this work is to compare the MPEG-7 distance measures with the most powerful distance measures developed in other areas. Since MPEG-7 descriptions are purely quantitative but some of the most sophisticated distance measures are defined exclusively on predicates, a model is mandatory that allows the application of predicate-based distance measures on quantitative data. The model developed for this purpose is presented in the next section.

3.2 Quantisation model

The goal of the quantisation model is to redefine the set operators that are usually used in predicate-based distance measures on continuous data. The first in visual information retrieval to follow this approach were Santini and Jain, who tried to apply Tversky's Feature Contrast Model [17] to content-based image retrieval [12], [13]. They interpreted continuous data as fuzzy predicates and used fuzzy set operators. Unfortunately, their model suffered from several shortcomings they described in [12], [13] (for example, the quantitative model worked only for one specific version of the original predicate-based measure).

The main idea of the presented quantisation model is that set operators are replaced by *statistical* functions. In [5] the authors could show that this interpretation of set operators is reasonable.

The model offers a solution for the descriptors considered in the evaluation. It is not specific to a certain distance measure, but can be applied to any predicate-based measure. In the following it will be shown that this model does not only work for predicate data but for quantitative data as well. Each measure implementing the model can be used as a substitute for the original predicate-based measure.

Generally, binary properties of two objects (e.g. media objects) can exist in both objects (denoted as *a*), in just one (*b*, *c*) or in none of them (*d*). The operator needed for these relationships are *UNION*, *MINUS* and *NOT*. In the quantisation model they are replaced as follows (see [5] for further details).

$$a = X_i \cap X_j = \sum_k s_k, \;\; s_k = \begin{cases} \dfrac{x_{ik} + x_{jk}}{2} & if\; M - \dfrac{x_{ik} + x_{jk}}{2} \le \varepsilon_1 \\ 0 & else \end{cases}$$

$$b = X_i - X_j = \sum_k s_k,\ s_k = \begin{cases} x_{ik} - x_{jk} & if\ M - \left(x_{ik} - x_{jk}\right) \le \varepsilon_2 \\ 0 & else \end{cases}$$

$$c = X_j - X_i = \sum_k s_k,\ s_k = \begin{cases} x_{jk} - x_{ik} & if\ M - \left(x_{jk} - x_{ik}\right) \le \varepsilon_2 \\ 0 & else \end{cases}$$

$$d = \neg X_i \cap \neg X_j = \sum_k s_k,\ s_k = \begin{cases} M - \dfrac{x_{ik} + x_{jk}}{2} & if\ \dfrac{x_{ik} + x_{jk}}{2} \le \varepsilon_1 \\ 0 & else \end{cases}$$

with:

$$X_i = \left(x_{ik}\right) with\, x_{ik} \in \left[x_{\min}, x_{\max}\right]$$

$$M = x_{\max} - x_{\min}$$

$$\varepsilon_1 = \begin{cases} M\left(1 - \dfrac{\mu}{p}\right) & if\ p \ge \mu \\ 0 & else \end{cases} \quad where\ \mu = \frac{\sum_i \sum_k x_{ik}}{i.k}$$

$$\varepsilon_2 = \begin{cases} M\left(1 - \dfrac{\sigma}{p}\right) & if\ p \ge \sigma \\ 0 & else \end{cases} \quad where\ \sigma = \sqrt{\frac{\sum_i \sum_k \left(\mu - x_{ik}\right)^2}{i.k}}$$

$$p \in R^+ \setminus \{0\}$$

a selects properties that are present in both data vectors (X_i, X_j representing media objects), *b* and *c* select properties that are present in just one of them and *d* selects properties that are present in neither of the two data vectors. Every property is selected by the *extent* to which it is present (*a* and *d*: mean, *b* and *c*: difference) and only if the amount to which it is present exceeds a certain threshold (depending on the mean and standard deviation over all elements of descriptor space).

The implementation of these operators is based on a single assumption. It is assumed that vector elements measure on an interval scale. That means, each element expresses that the measured property is "more or less" present ("*0*": not at all, "*M*": fully present). This is true for most visual descriptors and all MPEG-7 descriptors. A natural origin as it is assumed here ("*0*") is not needed.

Introducing *p* (called discriminance-defining parameter) for the thresholds $\varepsilon_1, \varepsilon_2$ has the positive consequence that *a, b, c, d* can then be controlled through a *single* parameter. *p* is an additional criterion for the behaviour of a distance measure and determines the thresholds used in the operators. It expresses how accurate data items are present (quantisation) and consequently, how accurate they should be investigated. *p* can be set by the user or automatically. Interesting are the limits:

$$1.\ p \to \infty \Rightarrow \varepsilon_1, \varepsilon_2 \to M$$

In this case, all elements (=properties) are assumed to be continuous (high quantisation). In consequence, all properties of a descriptor are used by the operators. Then, the distance measure is *not* discriminant for properties.

$$2.\ p \to 0 \Rightarrow \varepsilon_1, \varepsilon_2 \to 0$$

In this case, all properties are assumed to be predicates. In consequence, only binary elements (=predicates) are used by the operators (1-bit quantisation). The distance measure is then highly discriminant for properties.

Between these limits, a distance measure that uses the quantisation model is – depending on *p* – more or less discriminant for properties. This means, it selects a subset of all available description vector elements for distance measurement.

Table 1. Quantisation model on predicate vectors.

X_i	X_j	*a*	*b*	*c*	*d*
(1)	(1)	1	0	0	0
(1)	(0)	0	1	0	0
(0)	(1)	0	0	1	0
(0)	(0)	0	0	0	1

For both predicate data and quantitative data it can be shown that the quantisation model is reasonable. If description vectors consist of binary elements only, *p* should be used as follows (for example, *p* can easily be set automatically):

$$p \to 0 \Rightarrow \varepsilon_1, \varepsilon_2 = 0, e.g.\, p = \min(\mu, \sigma)$$

In this case, *a*, *b*, *c*, *d* measure like the set operators they replace. For example, Table 1 shows their behaviour for two one-dimensional feature vectors X_i and X_j. As can be seen, the statistical measures work like set operators. Actually, the quantisation model works accurate on predicate data for any $p \neq \infty$.

To show that the model is reasonable for quantitative data the following fact is used. It is well known (and easy to show) that for predicate data some quantitative distance measures degenerate to predicate-based measures. For example, the L^1 metric (Manhattan metric) degenerates to the Hamming distance (from [9], without weights):

$$L^1 = \sum_k \left|x_{ik} - x_{jk}\right| \equiv b + c = Hamming\, distance$$

If it can be shown that the quantisation model is able to *reconstruct* the quantitative measure from the degenerated predicate-based measure, the model is obviously able to *extend* predicate-based measures to the quantitative domain. This is easy to illustrate. For purely quantitative feature vectors, *p* should be used as follows (again, *p* can easily be set automatically):

$$p \to \infty \Rightarrow \varepsilon_1, \varepsilon_2 = 1$$

Then, *a* and *d* become continuous functions:

$$M - \frac{x_{ik} + x_{jk}}{2} \le M \equiv true \Rightarrow a = \sum_k s_k\ where\, s_k = \frac{x_{ik} + x_{jk}}{2}$$

$$\frac{x_{ik} + x_{jk}}{2} \le M \equiv true \Rightarrow d = \sum_k s_k\ where\, s_k = M - \frac{x_{ik} + x_{jk}}{2}$$

b and *c* can be made continuous for the following expressions:

$$M - \left(x_{ik} - x_{jk}\right) \le M \equiv x_{ik} - x_{jk} \ge 0$$

$$\Rightarrow b = \sum_k s_k\ where\, s_k = \begin{cases} x_{ik} - x_{jk} & if\ x_{ik} - x_{jk} \ge 0 \\ 0 & else \end{cases}$$

$$M - \left(x_{jk} - x_{ik}\right) \le M \equiv x_{jk} - x_{ik} \ge 0$$

$$\Rightarrow c = \sum_k s_k\ where\, s_k = \begin{cases} x_{jk} - x_{ik} & if\ x_{jk} - x_{ik} \ge 0 \\ 0 & else \end{cases}$$

$$\Rightarrow b + c = \sum_k s_k\ where\, s_k = \left|x_{ik} - x_{jk}\right|$$

Table 2. Predicate-based distance measures.

No.	Measure	Comment
P1	$a-\alpha.b-\beta.c$	Feature Contrast Model, Tversky 1977 [17]
P2	a	No. of co-occurrences
P3	$b+c$	Hamming distance
P4	$\frac{a}{K}$	Russel 1940 [14]
P5	$\frac{a}{b+c}$	Kulczvnski 1927 [14]
P6	$\frac{bc}{K^2}$	Pattern difference [14]
P7	$\frac{ad-bc}{\sqrt{(a+b)(a+c)(b+d)(c+d)}}$	Pearson 1926 [11]

$$b-c=\sum_k s_k \ where \ s_k = x_{ik}-x_{jk}$$

$$c-b=\sum_k s_k \ where \ s_k = x_{jk}-x_{ik}$$

This means, for sufficiently high *p* every predicate-based distance measure that is either not using *b* and *c* or just as *b+c*, *b-c* or *c-b*, can be transformed into a continuous quantitative distance measure. For example, the Hamming distance (again, without weights):

$$b+c=\sum_k s_k \ where \ s_k=\left|x_{ik}-x_{jk}\right|=\sum_k \left|x_{ik}-x_{jk}\right|=L^1$$

The quantisation model successfully reconstructs the L^1 metric and no distance measure-specific modification has to be made to the model. This demonstrates that the model is reasonable. In the following it will be used to extend successful predicate-based distance measures on the quantitative domain.

The major advantages of the quantisation model are: (1) it is application domain independent, (2) the implementation is straightforward, (3) the model is easy to use and finally, (4) the new parameter *p* allows to control the similarity measurement process in a new way (discriminance on property level).

3.3 Implemented measures

For the evaluation described in this work next to predicate-based (based on the quantisation model) and quantitative measures, the distance measures recommended in the MPEG-7 standard were implemented (all together 38 different distance measures).

Table 2 summarises those predicate-based measures that performed best in the evaluation (in sum 20 predicate-based measures were investigated). For these measures, *K* is the number of predicates in the data vectors X_i and X_j. In P1, the *sum* is used for Tversky's *f()* (as Tversky himself does in [17]) and *α, β* are weights for element *b* and *c*. In [5] the author's investigated Tversky's Feature Contrast Model and found *α=1, β=0* to be the optimum parameters.

Some of the predicate-based measures are very simple (e.g. P2, P4) but have been heavily exploited in psychological research. Pattern difference (P6) – a very powerful measure – is used in the statistics package SPSS for cluster analysis. P7 is a correlation coefficient for predicates developed by Pearson.

Table 3 shows the best quantitative distance measures that were used. Q1 and Q2 are metric-based and were implemented as representatives for the entire group of Minkowski distances. The w_i are weights. In Q5, μ_i, σ_i are mean and standard deviation for the elements of descriptor X_i. In Q6, *m* is $\frac{M}{2}$ (=0.5). Q3, the Canberra metric, is a normalised form of Q1. Similarly, Q4, Clark's divergence coefficient is a normalised version of Q2. Q6 is a further-developed correlation coefficient that is invariant against sign changes. This measure is used even though its particular properties are of minor importance for this application domain. Finally, Q8 is a measure that takes the differences between adjacent vector elements into account. This makes it structurally different from all other measures.

Obviously, one important distance measure is missing. The Mahalanobis distance was not considered, because different descriptors would require different covariance matrices and for some descriptors it is simply impossible to define a covariance matrix. If the identity matrix was used in this case, the Mahalanobis distance would degenerate to a Minkowski distance.

Additionally, the recommended MPEG-7 distances were implemented with the following parameters: In the distance measure of the Color Layout descriptor all weights were set to "1" (as in all other implemented measures). In the distance measure of the Dominant Color descriptor the following parameters were used: $w_1=0.7, w_2=0.3, \alpha=1, T_d=20$ (as recommended). In the Homogeneous Texture descriptor's distance all $\alpha(k)$ were set to "1" and matching was done rotation- and scale-invariant.

Important! Some of the measures presented in this section are *distance* measures while others are *similarity* measures. For the tests, it is important to notice, that all similarity measures were *inverted* to distance measures.

4. TEST SETUP

Subsection 4.1 describes the descriptors (including parameters) and the collections (including ground truth information) that were used in the evaluation. Subsection 4.2 discusses the evaluation method that was implemented and Subsection 4.3 sketches the test environment used for the evaluation process.

4.1 Test data

For the evaluation eight MPEG-7 descriptors were used. All colour descriptors: Color Layout, Color Structure, Dominant Color, Scalable Color, all texture descriptors: Edge Histogram, Homogeneous Texture, Texture Browsing and one shape descriptor: Region-based Shape. Texture Browsing was used even though the MPEG-7 standard suggests that it is not suitable for retrieval. The other basic shape descriptor, Contour-based Shape, was not used, because it produces structurally different descriptions that cannot be transformed to data vectors with elements measuring on interval-scales. The motion descriptors were not used, because they integrate the temporal dimension of visual media and would only be comparable, if the basic colour, texture and shape descriptors would be aggregated over time. This was not done. Finally, no high-level descriptors were used (Localisation, Face Recognition, etc., see Subsection 2.1), because – to the author's opinion – the behaviour of the basic descriptors on elementary media objects should be evaluated *before* conclusions on aggregated structures can be drawn.

Table 3. Quantitative distance measures.

No.	Measure	Comment	No.	Measure	Comment
Q1	$\sum_k w_i \left\| x_{ik} - x_{jk} \right\|$	City block distance (L^1)	Q2	$\sqrt{\sum_k w_i \left(x_{ik} - x_{jk}\right)^2}$	Euclidean distance (L^2)
Q3	$\sum_k \frac{x_{ik} - x_{jk}}{x_{ik} + x_{jk}}$	Canberra metric, Lance, Williams 1967 [8]	Q4	$\frac{1}{K}\sqrt{\sum_k \frac{\left(x_{ik} - x_{jk}\right)^2}{x_{ik} + x_{jk}}}$	Divergence coefficient, Clark 1952 [1]
Q5	$\frac{\sum_k \left(x_{ik} - \mu_i\right)\left(x_{jk} - \mu_j\right)}{\sqrt{\sum_k \left(x_{ik} - \mu_i\right)^2 \sum_k \left(x_{jk} - \mu_j\right)^2}}$	Correlation coefficient	Q6	$\frac{\sum_k x_{ik} x_{jk} - Km - m\left(\sum_k x_{ik} + \sum_k x_{jk}\right)}{\sqrt{\left(\sum_k x_{ik}^2 - Km^2 - 2m.x_{ik}\right)\left(\sum_k x_{jk}^2 + Km^2 - 2m\sum_k x_{ik}\right)}}$	Cohen 1969 [2]
Q7	$\frac{\sum_k x_{ik} x_{jk}}{\sum_k x_{ik}^2 \sum_k x_{jk}^2}$	Angular distance, Gower 1967 [7]	Q8	$\sum_k^{K-1} \left(\left(x_{ik} - x_{ik+1}\right) - \left(x_{jk} - x_{jk+1}\right)\right)^2$	Meehl Index [10]

The Texture Browsing descriptions had to be transformed from five bins to an eight bin representation in order that all elements of the descriptor measure on an interval scale. A Manhattan metric was used to measure proximity (see [6] for details).

Descriptor extraction was performed using the MPEG-7 reference implementation. In the extraction process each descriptor was applied on the entire content of each media object and the following extraction parameters were used. Colour in Color Structure was quantised to 32 bins. For Dominant Color colour space was set to YCrCb, 5-bit default quantisation was used and the default value for spatial coherency was used. Homogeneous Texture was quantised to 32 components. Scalable Color values were quantised to *sizeof(int)-3* bits and 64 bins were used. Finally, Texture Browsing was used with five components.

These descriptors were applied on three media collections with image content: the Brodatz dataset (112 images, 512x512 pixel), a subset of the Corel dataset (260 images, 460x300 pixel, portrait and landscape) and a dataset with coats-of-arms images (426 images, 200x200 pixel). Figure 1 shows examples from the three collections.

Designing appropriate test sets for a visual evaluation is a highly difficult task (for example, see the TREC video 2002 report [15]). Of course, for identifying the best distance measure for a descriptor, it should be tested on an infinite number of media objects. But this is not the aim of this study. It is just evaluated if – for likely image collections – better proximity measures than those suggested by the MPEG-7 group can be found. Collections of this relatively small size were used in the evaluation, because the applied evaluation methods are above a certain minimum size invariant against collection size and for smaller collections it is easier to define a high-quality ground truth. Still, the average ratio of ground truth size to collection size is at least 1:7. Especially, no collection from the MPEG-7 dataset was used in the evaluation because the evaluations should show, how well the descriptors and the recommended distance measures perform on "unknown" material.

When the descriptor extraction was finished, the resulting XML descriptions were transformed into a data matrix with 798 lines (media objects) and 314 columns (descriptor elements). To be usable with distance measures that do not integrate domain knowledge, the elements of this data matrix were normalised to [0, 1].

For the distance evaluation – next to the normalised data matrix – human similarity judgement is needed. In this work, the ground truth is built of twelve groups of similar images (four for each dataset). Group membership was rated by humans based on semantic criterions. Table 4 summarises the twelve groups and the underlying descriptions. It has to be noticed, that some of these groups (especially 5, 7 and 10) are much harder to find with low-level descriptors than others.

4.2 Evaluation method

Usually, retrieval evaluation is performed based on a ground truth with *recall* and *precision* (see, for example, [3], [16]). In multi-descriptor environments this leads to a problem, because the resulting recall and precision values are strongly influenced by the method used to merge the distance values for one media object. Even though it is nearly impossible to say, how big the influence of a single distance measure was on the resulting recall and precision values, this problem has been almost ignored so far.

In Subsection 2.2 it was stated that the major task of a distance measure is to bring the relevant media objects *as close* to the origin (where the query object lies) *as possible*. Even in a multi-descriptor environment it is then simple to identify the similar objects in a large distance space. Consequently, it was decided to

Table 4. Ground truth information.

Coll.	No	Images	Description
Brodatz	1	19	Regular, chequered patterns
	2	38	Dark white noise
	3	33	Moon-like surfaces
	4	35	Water-like surfaces
Corel	5	73	Humans in nature (difficult)
	6	17	Images with snow (mountains, skiing)
	7	76	Animals in nature (difficult)
	8	27	Large coloured flowers
Arms	9	12	Bavarian communal arms
	10	10	All Bavarian arms (difficult)
	11	18	Dark objects / light unsegmented shield
	12	14	Major charges on blue or red shield

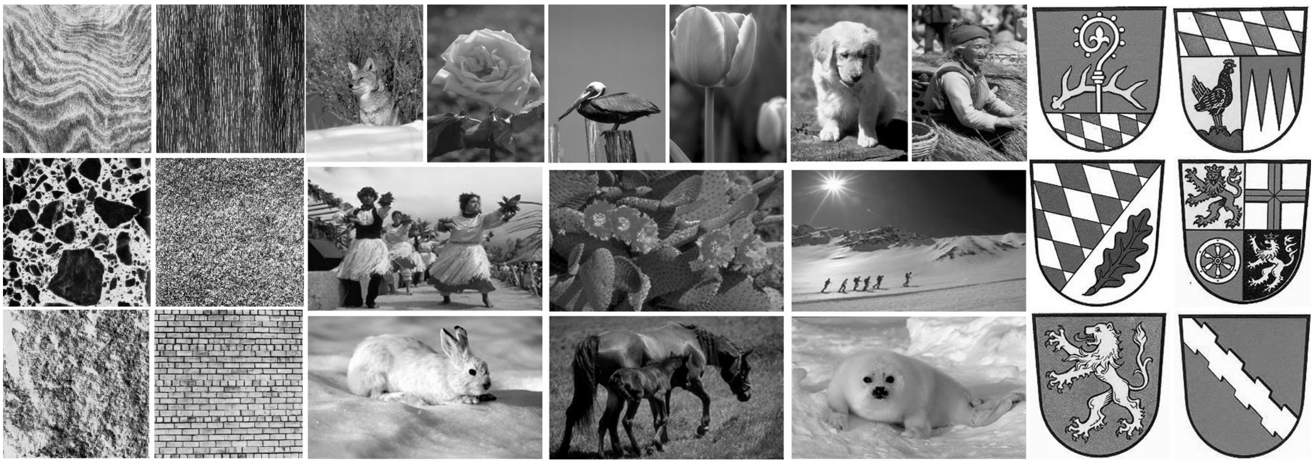


Figure 1. Test datasets. Left: Brodatz dataset, middle: Corel dataset, right: coats-of-arms dataset.

use indicators measuring the distribution in distance space of candidates similar to the query object for this evaluation instead of recall and precision. Identifying clusters of similar objects (based on the given ground truth) is relatively easy, because the resulting distance space for one descriptor and any distance measure is always *one-dimensional*. Clusters are found by searching from the *origin* of distance space to the first similar object, grouping all following similar objects in the cluster, breaking off the cluster with the first un-similar object and so forth.

For the evaluation two indicators were defined. The first measures the average distance of all cluster means to the origin:

$$\mu_d = \frac{\sum_i^{no\_clusters} \frac{\sum_j^{cluster\_size_i} distance_{ij}}{cluster\_size_i}}{no\_clusters.avg\_distance}$$

where *distance*$_{ij}$ is the distance value of the *j*-th element in the *i*-th cluster, $avg\_distance = \frac{\sum_i^{CLUSTERS} \sum_j^{cluster\_size_i} distance_{ij}}{\sum_i^{CLUSTERS} cluster\_size_i}$, *no_clusters* is the number of found clusters and *cluster_size*$_i$ is the size of the *i*-th cluster. The resulting indicator is normalised by the distribution characteristics of the distance measure (*avg_distance*). Additionally, the standard deviation is used. In the evaluation process this measure turned out to produce valuable results and to be relatively robust against parameter *p* of the quantisation model.

In Subsection 3.2 we noted that *p* affects the discriminance of a predicate-based distance measure: The smaller *p* is set the larger are the resulting clusters because the quantisation model is then more discriminant against properties and less elements of the data matrix are used. This causes a side-effect that is measured by the second indicator: more and more un-similar objects come out with *exactly* the same distance value as similar objects (a problem that does not exist for large *p*'s) and become *indiscernible* from similar objects. Consequently, they are (false) cluster members. This phenomenon (conceptually similar to the "false negatives" indicator) was named "cluster pollution" and the indicator measures the average cluster pollution over all clusters:

$$cp = \frac{\sum_i^{no\_clusters} \sum_j^{cluster\_size_i} no\_doubles_{ij}}{no\_clusters}$$

where *no_doubles*$_{ij}$ is the number of indiscernible un-similar objects associated with the *j*-th element of cluster *i*.

Remark: Even though there is a certain influence, it could be proven in [5] that no significant correlation exists between parameter *p* of the quantisation model and cluster pollution.

4.3 Test environment

As pointed out above, to generate the descriptors, the MPEG-7 reference implementation in version 5.6 was used (provided by TU Munich). Image processing was done with Adobe Photoshop and normalisation and all evaluations were done with Perl. The querying process was performed in the following steps: (1) random selection of a ground truth group, (2) random selection of a query object from this group, (3) distance comparison for all other objects in the dataset, (4) clustering of the resulting distance space based on the ground truth and finally, (5) evaluation.

For each combination of dataset and distance measure 250 queries were issued and evaluations were aggregated over all datasets and descriptors. The next section shows the – partially surprising – results.

5. RESULTS

In the results presented below the first indicator from Subsection 4.2 was used to evaluate distance measures. In a first step parameter *p* had to be set in a way that all measures are *equally* discriminant. Distance measurement is fair if the following condition holds true for any predicate-based measure d_P and any continuous measure d_C:

$$cp(d_P, p) \approx cp(d_C)$$

Then, it is guaranteed that predicate-based measures do not create larger clusters (with a higher number of similar objects) for the price of higher cluster pollution. In more than 1000 test queries the optimum value was found to be *p=1*.

Results are organised as follows: Subsection 5.1 summarises the

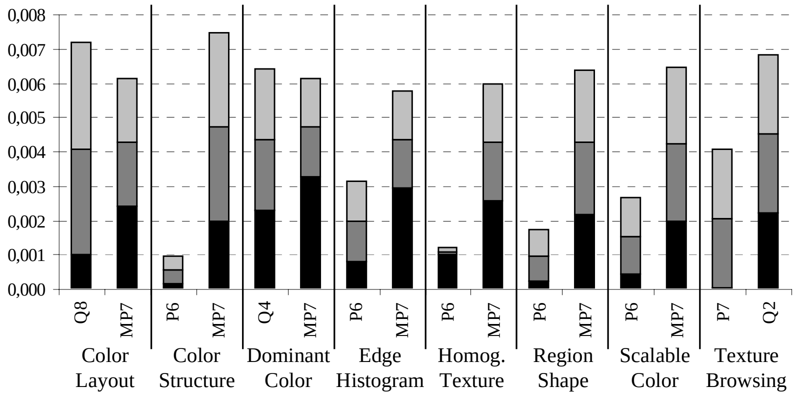


Figure 2. Results per measure and descriptor. The horizontal axis shows the best measure and the performance of the MPEG-7 recommendation for each descriptor. The vertical axis shows the values for the first indicator (smaller value = better cluster structure). Shades have the following meaning: black=μ-σ (good cases), black + dark grey=μ (average) and black + dark grey + light grey=$\mu+\sigma$ (bad).

best distance measures per descriptor, Section 5.2 shows the best overall distance measures and Section 5.3 points out other interesting results (for example, distance measures that work particularly good on specific ground truth groups).

5.1 Best measure per descriptor

Figure 2 shows the evaluation results for the first indicator. For each descriptor the best measure and the performance of the MPEG-7 recommendation are shown. The results are aggregated over the tested datasets.

On first sight, it becomes clear that the MPEG-7 recommendations are mostly relatively good but *never* the best. For Color Layout the difference between MP7 and the best measure, the Meehl index (Q8), is just 4% and the MPEG-7 measure has a smaller standard deviation. The reason why the Meehl index is better may be that this descriptors generates descriptions with elements that have very similar variance. Statistical analysis confirmed that (see [6]).

For Color Structure, Edge Histogram, Homogeneous Texture, Region-based Shape and Scalable Color by far the best measure is pattern difference (P6). Psychological research on human visual perception has revealed that in many situation differences between the query object and a candidate weigh much stronger than common properties. The pattern difference measure implements this insight in the most consequent way. In the author's opinion, the reason why pattern difference performs so extremely well on many descriptors is due to this fact. Additional advantages of pattern difference are that it usually has a very low variance and – because it is a predicate-based measure – its discriminance (and cluster structure) can be tuned with parameter *p*.

The best measure for Dominant Color turned out to be Clark's Divergence coefficient (Q4). This is a similar measure to pattern difference on the continuous domain. The Texture Browsing descriptor is a special problem. In the MPEG-7 standard it is recommended to use it exclusively for browsing. After testing it for retrieval on various distance measures the author supports this opinion. It is very difficult to find a good distance measure for Texture Browsing. The proposed Manhattan metric, for example, performs very bad. The best measure is predicate-based (P7). It works on common properties (*a*, *d*) but produces clusters with very high cluster pollution. For this descriptor the second indicator is up to eight times higher than for predicate-based measures on other descriptors.

5.2 Best overall measures

Figure 3 summarises the results over all descriptors and media collections. The diagram should give an indication on the *general potential* of the investigated distance measures for visual information retrieval.

It can be seen that the best overall measure is a predicate-based one. The top performance of pattern difference (P6) proves that the quantisation model is a reasonable method to extend predicate-based distance measures on the continuous domain. The second best group of measures are the MPEG-7 recommendations, which have a slightly higher mean but a lower standard deviation than pattern difference. The third best measure is the Meehl index (Q8), a measure developed for psychological applications but because of its characteristic properties tailor-made for certain (homogeneous) descriptors.

Minkowski metrics are also among the best measures: the average mean and variance of the Manhattan metric (Q1) and the Euclidean metric (Q2) are in the range of Q8. Of course, these measures do not perform particularly well for any of the descriptors. Remarkably for a predicate-based measure, Tversky's Feature Contrast Model (P1) is also in the group of very good measures (even though it is not among the best) that ends with Q5, the correlation coefficient. The other measures either have a significantly higher mean or a very large standard deviation.

5.3 Other interesting results

Distance measures that perform in average worse than others may in certain situations (e.g. on specific content) still perform better. For Color Layout, for example, Q7 is a very good measure on colour photos. It performs as good as Q8 and has a lower standard deviation. For artificial images the pattern difference and the Hamming distance produce comparable results as well.

If colour information is available in media objects, pattern difference performs well on Dominant Color (just 20% worse Q4) and in case of difficult ground truth (group 5, 7, 10) the Meehl index is as strong as P6.

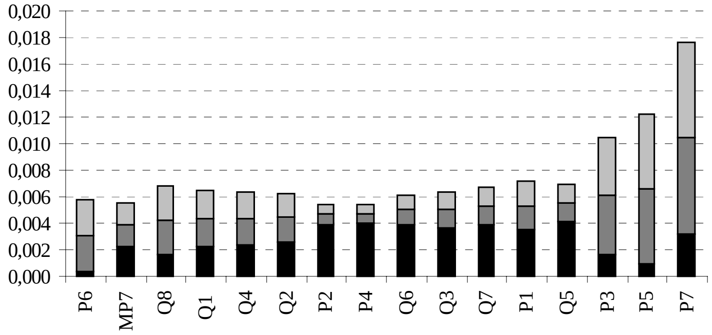


Figure 3. Overall results (ordered by the first indicator). The vertical axis shows the values for the first indicator (smaller value = better cluster structure). Shades have the following meaning: black=*μ*-*σ*, black + dark grey=*μ* and black + dark grey + light grey=*μ*+*σ*.

6. CONCLUSION

The evaluation presented in this paper aims at testing the recommended distance measures and finding better ones for the basic visual MPEG-7 descriptors. Eight descriptors were selected, 38 distance measures were implemented, media collections were created and assessed, performance indicators were defined and more than 22500 tests were performed. To be able to use predicate-based distance measures next to quantitative measures a quantisation model was defined that allows the application of predicate-based measures on continuous data.

In the evaluation the best overall distance measures for visual content – as extracted by the visual MPEG-7 descriptors – turned out to be the pattern difference measure and the Meehl index (for homogeneous descriptions). Since these two measures perform significantly better than the MPEG-7 recommendations they should be further tested on large collections of image and video content (e.g. from [15]).

The choice of the right distance function for similarity measurement depends on the descriptor, the queried media collection and the semantic level of the user's idea of similarity. This work offers suitable distance measures for various situations. In consequence, the distance measures identified as the best will be implemented in the open MPEG-7 based visual information retrieval framework VizIR [4].

ACKNOWLEDGEMENTS

The author would like to thank Christian Breiteneder for his valuable comments and suggestions for improvement. The work presented in this paper is part of the VizIR project funded by the Austrian Scientific Research Fund FWF under grant no. P16111.

REFERENCES

[1] Clark, P.S. An extension of the coefficient of divergence for use with multiple characters. Copeia, 2 (1952), 61-64.

[2] Cohen, J. A profile similarity coefficient invariant over variable reflection. Psychological Bulletin, 71 (1969), 281-284.

[3] Del Bimbo, A. Visual information retrieval. Morgan Kaufmann Publishers, San Francisco CA, 1999.

[4] Eidenberger, H., and Breiteneder, C. A framework for visual information retrieval. In Proceedings Visual Information Systems Conference (HSinChu Taiwan, March 2002), LNCS 2314, Springer Verlag, 105-116.

[5] Eidenberger, H., and Breiteneder, C. Visual similarity measurement with the Feature Contrast Model. In Proceedings SPIE Storage and Retrieval for Media Databases Conference (Santa Clara CA, January 2003), SPIE Vol. 5021, 64-76.

[6] Eidenberger, H., How good are the visual MPEG-7 features? In Proceedings SPIE Visual Communications and Image Processing Conference (Lugano Switzerland, July 2003), SPIE Vol. 5150, 476-488.

[7] Gower, J.G. Multivariate analysis and multidimensional geometry. The Statistician, 17 (1967),13-25.

[8] Lance, G.N., and Williams, W.T. Mixed data classificatory programs. Agglomerative Systems Australian Comp. Journal, 9 (1967), 373-380.

[9] Manjunath, B.S., Ohm, J.R., Vasudevan, V.V., and Yamada, A. Color and texture descriptors. In Special Issue on MPEG-7. IEEE Transactions on Circuits and Systems for Video Technology, 11/6 (June 2001), 703-715.

[10] Meehl, P. E. The problem is epistemology, not statistics: Replace significance tests by confidence intervals and quantify accuracy of risky numerical predictions. In Harlow, L.L., Mulaik, S.A., and Steiger, J.H. (Eds.). What if there were no significance tests? Erlbaum, Mahwah NJ, 393-425.

[11] Pearson, K. On the coefficients of racial likeness. Biometrica, 18 (1926), 105-117.

[12] Santini, S., and Jain, R. Similarity is a geometer. Multimedia Tools and Application, 5/3 (1997), 277-306.

[13] Santini, S., and Jain, R. Similarity measures. IEEE Transactions on Pattern Analysis and Machine Intelligence, 21/9 (September 1999), 871-883.

[14] Sint, P.P. Similarity structures and similarity measures. Austrian Academy of Sciences Press, Vienna Austria, 1975 (in German).

[15] Smeaton, A.F., and Over, P. The TREC-2002 video track report. NIST Special Publication SP 500-251 (March 2003), available from: http://trec.nist.gov/pubs/trec11/papers/VIDEO.OVER.pdf (last visited: 2003-07-29)

[16] Smeulders, A.W.M., Worring, M., Santini, S., Gupta, A., and Jain, R. Content-based image retrieval at the end of the early years. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22/12 (December 2000), 1349-1380.

[17] Tversky, A. Features of similarity. Psychological Review, 84/4 (July 1977), 327-351.

Statistical analysis of content-based MPEG-7 descriptors for image retrieval

HORST EIDENBERGER

Vienna University of Technology, Institute of Software Technology and Interactive Systems, Interactive Media Systems Group, Favoritenstrasse 9-11, A-1040 Vienna, Austria

Phone 43 1 58801-18853, Fax 43 1 58801-18898

eMail eidenberger@ims.tuwien.ac.at, Web www.ims.tuwien.ac.at

Abstract. The study presented in this paper analyses the visual MPEG-7 descriptors from a statistical point of view. A statistical analysis is able to reveal the properties and qualities of the used descriptors: redundancies, sensitivity on media content, etc. These aspects were not considered in the MPEG-7 design process where the major goal was optimising the retrieval rate. For the statistical analysis eight basic visual descriptors were applied to three media collections: the Brodatz dataset, a selection of the Corel photo dataset and a set of coats-of-arms images. The resulting feature vectors were analysed with four statistical methods: mean and variance of description elements, distribution of elements, cluster analysis (hierarchical and topological) and factor analysis. The analysis revealed that, for example, most MPEG-7 descriptions are highly redundant and sensitive to the presence of colour shades.

Keywords: *Visual Information Retrieval, MPEG-7, Cluster Analysis, Factor Analysis, Self-Organizing Map*

1. Introduction

The MPEG-7 standard defines – among others – a set of descriptors (semantics and syntax) for visual media content [16, 15, 1, 3]. During the design process these descriptors were tested with recall- and precision-like performance measures on large datasets and ground truth information (average normalised modified retrieval rank (ANMRR) [15]). These tests represented well the performance of the descriptors in the retrieval process, but cannot be used to judge their *application-independent* performance.

Descriptor algorithms extract feature vectors from media content. In order to be able to measure proximities between feature vectors, a fundamental operation in most applications as retrieval, browsing, etc., vectors are interpreted as points in a metric space [8]. An efficient (general-purpose) descriptor should provide a surjective mapping from media objects to points in feature space and meet several criteria: Ideally, a descriptor should be highly discriminant for any type of media content. The description extraction process should be robust against different levels of quality and detail. Additionally, all description elements should contain meaningful data for any type of media.

The work described in this paper aims at assessing the *efficiency* of MPEG-7 descriptors. Efficiency describes the extent to which descriptors are suitable for the intended application domains: for example, whether or not it makes sense to combine them with other descriptors. The efficiency of visual descriptors is derived from the statistical quality of descriptions extracted from pre-defined media collections. In particular, it is investigated if redundancies exist among the descriptors (for specific content and in general), how sensible the descriptors are for changes in the content and whether or not the descriptors cover the proposed media property (e.g. color layout) completely. Of course, not all relevant aspects can be covered by these three areas. However, redundancy, sensitivity and completeness are fundamental statistical properties for the judgement of the quality of descriptors. Surprisingly, this was not considered in the MPEG-7 design process.

The practical goals of the evaluation are the refinement of guidelines describing the usage of descriptors (e.g. for visual information retrieval [4, 18]) and suggestions describing the improvement of descriptors. In practice the criteria pointed out above can only partially be met. In the evaluation we found that, for example, *Homogeneous Texture* extracts highly redundant descriptions, that the *Group-of-Frames/Group-of-Pictures Color* descriptor should not be based on *Scalable Color*, because this descriptor is highly sensitive to the presence/absence of colour information and that one component of *Color Layout* explains almost the entire *Region-based Shape* descriptor.

The paper is organised as follows. Section 2 summarises background information on the visual MPEG-7 descriptors and on statistical methods for data analysis. Section 3 describes the evaluation framework. For the analysis descriptors are applied on pre-defined media collections (see 3.2) and the results are analysed with statistical methods (e.g. cluster analysis, factor analysis; see 2.2 for details). Section 4, 5 and 6 investigate various aspects of the visual MPEG-7 descriptions: redundancy, sensitivity on varying media content and completeness. Finally, Section 7 summarises the analysis results.

2. Background

2.1 MPEG-7 visual descriptors

The visual part of the MPEG-7 standard defines several descriptors [16, 15, 1]. Not all of them are actually descriptors in the sense that they extract properties of media content. Some of them are just structures for descriptor aggregation and localisation. The basic descriptors are *Color Layout*, *Color Structure*, *Dominant Color*, *Scalable Color* (colour), *Edge Histogram*, *Homogeneous Texture*, *Texture Browsing* (texture), *Region-based Shape*, *Contour-based Shape* (shape), *Camera Motion*, *Parametric Motion* and *Motion Activity* (motion).

Other descriptors are based on these low-level descriptors or on additional semantic information: *Group-of-Frames*/*Group-of-Pictures* (aggregation of *Scalable Color* descriptions), *Shape 3D* (based on 3D mesh information), *Motion Trajectory* (based on object segmentation) and *Face Recognition* (based on face extraction). Descriptors for spatiotemporal aggregation and localisation are: *Spatial 2D Coordinates*, *Grid Layout*, *Region Locator* (spatial), *Time Series*, *Temporal Interpolation* (temporal) and *SpatioTemporal Locator* (combined). Finally, supplementary (textual) structures exist for colour spaces, colour quantisation and multiple 2D views of 3D objects.

These additional structures allow for combining the basic descriptors in multiple ways and on different levels. But they do not change the *characteristics* of the extracted information. Consequently, structures for aggregation and localisation were not considered in the analysis described in this paper.

2.2 Statistical analysis of data vectors

Matrix 1 shows a fraction of a data matrix as it could be computed by feature extraction algorithms from media objects: two features f_1 (elements e_1 to e_m) and f_2 (elements e_{m+1} to e_n) and media objects o_1 , o_2 to o_l. The major quality indicators for feature extraction methods are the characteristics of the extracted *description elements* (e.g. the bins of a color histogram). The elements are given as vectors over the size of the test dataset (rows of the data matrix). The characteristics can, for example, be measured as moments (e.g. mean, variance) *of* vectors (rows), proximity and dependencies *between* vectors and distributions of quantised vector elements. These characteristics are of particular importance for visual information retrieval, because most querying paradigms follow the vector space model where feature vectors are interpreted as points in a metric vector space (feature space) [8]. Since the media collections used for feature extraction are well known, the characteristics of the population of feature space can be used to draw conclusions on the characteristics of the applied descriptors.

$$\begin{array}{c|ccccc|ccc|}
 & \multicolumn{5}{c}{\overbrace{\qquad\qquad\qquad\qquad}^{f_1}} & \multicolumn{3}{c}{\overbrace{\qquad\qquad\quad}^{f_2}} \\
 & e_1 & e_2 & e_3 & \cdots & e_m & e_{m+1} & \cdots & e_n \\
o_1 & x_{11} & x_{12} & x_{13} & \cdots & x_{1m} & x_{1m+1} & \cdots & x_{1n} \\
o_2 & x_{21} & x_{22} & x_{23} & \cdots & x_{2m} & x_{2m+1} & \cdots & x_{2n} \\
o_3 & x_{31} & x_{32} & x_{33} & \cdots & x_{3m} & x_{3m+1} & \cdots & x_{3n} \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
o_l & x_{l1} & x_{l2} & x_{l3} & \cdots & x_{lm} & x_{lm+1} & \cdots & x_{\mathrm{ln}}
\end{array} \tag{1}$$

In the described work, five statistical methods were used to analyse the data matrix generated with visual MPEG-7 descriptors: (1) extraction of statistical indicators (mean and standard deviation) of elements, (2) calculation of the distribution of quantised element values (the x_{ij} in the data matrix), (3) one-dimensional (hierarchical) cluster analysis of elements, (4) two-dimensional (topological) cluster analysis and (5) factor analysis. The methods are based on a single prerequisite: all description elements have to measure at least on interval scale, i.e. it must be possible to measure differences of description elements (a natural zero and computation of ratios are not required).

Mean and standard deviation represent a simple characterisation of a data vector. The mean points at the average location of the underlying extraction method and the standard deviation gives a first clue on its discriminance. If the standard deviation is almost zero, a description extraction method generates the same output for any type of given media content. Therefore, it may be characterised as being non-discriminant.

The distribution of the values of an element is calculated in a two step process: First, all coefficients (one row) are quantised to the same number of values (e.g. ten). Then, coefficients with equal value are summed up to a histogram bin. The result is a discrete density function expressing how often each value occurs. The histogram visualises the characteristics of the extraction methods (e.g. uniformly distributed, Gaussian-distributed). This method can be used to identify gaps within the measured range.

The cluster analysis methods derive groups of more than average similar description elements. Thus, for example, redundancies in descriptions can be found. For one-dimensional analysis a hierarchical method was employed [10] that displays analysis results in form of dendrograms. Next to clusters, the hierarchy of proximities can be used to assess the distribution of elements within the description extraction space. The two-dimensional method enriches the result of the hierarchical cluster analysis by identifying clusters on a two-dimensional map. Such a map shows the relationships of clusters and may be used to identify *holes* in the feature extraction process. For the discussed analysis, Self-Organizing Maps [13, 11] (SOMs, fully-connected two-layer neural networks with unsupervised feed-forward learning) were used, because they produce a more natural clustering than, for example, *k*-means clustering techniques. Finally, factor analysis [14] was applied as a method to eliminate redundancies in data vectors by identifying factors that cause the variance of the examined data. Additionally, factor analysis can be used to identify common properties of descriptors by finding elements that load high on the same factor.

3. Evaluation framework

This section describes all aspects of the framework used for the evaluation. The descriptors are described in subsection 3.1. Subsection 3.2 describes the media collections the descriptors were applied on. Subsection 3.3 sketches the test environment. Finally, in subsection 3.4 the parameters used for the statistical analysis methods are described.

3.1 Descriptors

Eight MPEG-7 descriptors were statistically analysed. All colour descriptors: *Color Layout* (CLD), *Color Structure* (CSD), *Dominant Color* (DCD), *Scalable Color* (SCD), all texture descriptors: *Edge Histogram* (EHD), *Homogeneous Texture* (HTD), *Texture Browsing* (TBD) and one shape descriptor: *Region-based Shape* (RSD). The other basic shape descriptor, *Contour-based Shape*, was not used, because it produces structurally different descriptions that cannot be transformed to data vectors measuring on interval scale. The motion descriptors were not considered, since they integrate the temporal domain of visual media and would only be comparable, if the basic colour, texture and shape descriptors would be aggregated over time. High-level descriptors (descriptors that are based on other descriptors instead of media data) were not used (*Localisation*, *Face Recognition*, etc.). In the author's opinion the behaviour of basic descriptors has to be evaluated *before* conclusions on aggregated structures can be drawn.

The *Texture Browsing* descriptor had to be transformed to be useable in the evaluation. In the MPEG-7 standard it is defined as follows [15]: *(regularity, direction$_1$, scale$_1$, direction$_2$, scale$_2$)* where *regularity* is element of *{not regular, slightly regular, regular, highly regular}*, *direction* (in degree) is element of *{no direction, 0, 30, 60, 90, 120, 150}* and *scale* is element of *{no scale, fine, medium, coarse, very coarse}*. Such a description is not suitable for the purpose of this paper. Therefore, the extracted descriptions were transformed to the following form: *(regularity, scale$_{no\ direction}$, scale$_0$, scale$_{30}$, scale$_{60}$, scale$_{90}$, scale$_{120}$, scale$_{150}$)* where *regularity* is element of *{0 (not regular), 1 (slightly regular), 2 (regular), 3 (highly regular)}* and the *scale* bins are element of *{0 (no scale), 1 (fine), 2 (medium), 3 (coarse), 4 (very coarse)}*. Defined like this, all elements of *Texture Browsing* measure on interval scale.

Description extraction was performed employing the MPEG-7 experimentation model (XM, [17]) of MPEG-7 Part 6: Reference Software. In the extraction process each descriptor was applied on the entire content of every media object. The following extraction parameters were used. Colour in *Color Structure* was quantised to 32 bins. For *Dominant Color*, colour space was set to YCrCb, 5-bit default quantisation was used and the default value for spatial coherency was used. *Homogeneous Texture* was quantised to 32 components. *Scalable Color* values were quantised to *sizeof(int)-3* bits and 64 bins were used. Finally, *Texture Browsing* was used with five components.

3.2 Media collections

The descriptors were applied on three media collections with image content: the Brodatz dataset [4] (112 monochrome images, 512x512 pixel), a subset of the Corel dataset [9] (260 colour photos, 460x300 pixel, portrait and landscape) and a dataset with coats-of-arms images [2] (426 synthetic images, 200x200 pixel). The Brodatz dataset is tailor-made for texture descriptors but a good test for colour and shape descriptors as well, because most colour descriptors are very sensitive for luminance and most shape descriptors use monochrome information for feature extraction. The Corel dataset (shipped with Corel Draw) is a widely applied set of colour photos showing humans, animals, flowers, landscapes, etc. One would suppose that colour and texture descriptors should work well on this set. For the evaluation a subset of images from all collections was randomly chosen. The dataset was not used in entirety, because most employed statistical algorithms would be overtaxed by such a large number of cases. The coats-of-arms dataset lies in-between these two collections: it consists of colour images with clear structures, few colour gradations and hardly any textures [2]. Therefore, colour and shape descriptors should work well on this dataset. Fig. 1 shows examples from the three collections.

Without doubt, other collections exists that could have

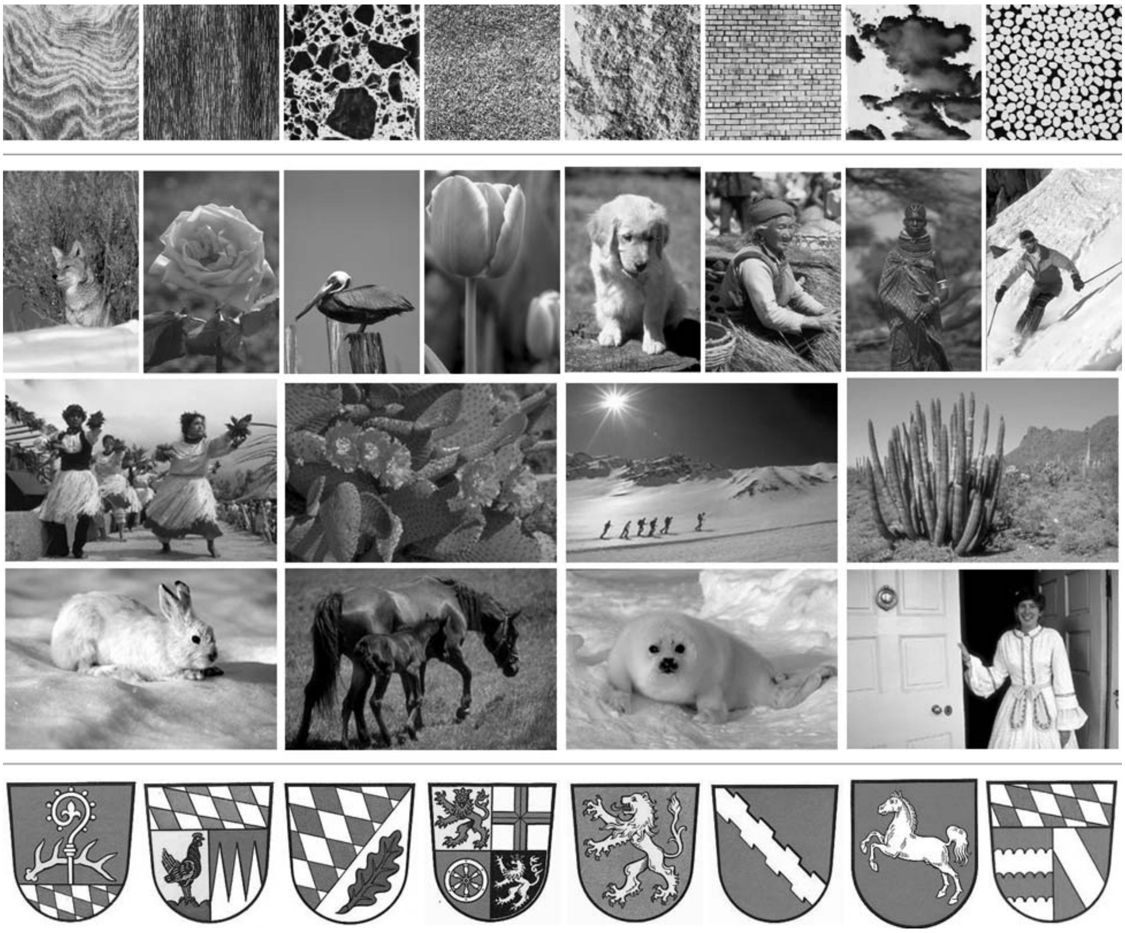


Fig. 1. Examples from the test datasets. First row: Brodatz dataset, second to fourth row: Corel dataset, last row: coats-of-arms dataset.

been used as well. The selected media sets have the advantage that they are carefully selected and highly independent from each other, since they are expressing entirely different properties. Especially, no collection from the MPEG-7 dataset was used in the evaluation, because the MPEG-7 descriptors were developed on the basis of these datasets. The evaluation should indicate, how the descriptors perform on "unknown" material.

3.3 Test environment

The evaluation was performed in the following steps: (1) description extraction, (2) transformation from XML to a tab-delimited format and normalisation, (3) extraction of statistical indicators, (4) quantisation and extraction of distributions, (5) hierarchical cluster analysis, (6) SOM calculation and (7) factor analysis. As pointed out above, the MPEG-7 experimentation model [17] in version 5.6 was used to generate the descriptions. After the description extraction, the resulting XML-descriptions were transformed into a data matrix with 798 lines (media objects) and 314 columns (description elements). To be usable for statistical analysis, the elements of this data matrix had to be normalised to a certain range. This was performed for every element with a simple min-max-normalisation:

$$x'_{ij} = \frac{x_{ij} - min_j}{max_j - min_j} \qquad (2)$$

where min_j is the minimum and max_j is the maximum of column *j*. The resulting value x'_{ij} is normalised to [0, 1]. This normalisation has the advantage that the relative distributions (variances) of both rows and columns of the data matrix are preserved. Normalisation and all other pre-processing steps (e.g. transformation of the *Texture Browsing* descriptor) were computed with Perl scripts. Hierarchical cluster analysis and factor analysis were calculated with SPSS and SOMs were calculated with SOM-PAK [12]. All other algorithms were implemented in Perl. Perl was chosen for its outstanding data processing capabilities and because it allows rapid prototyping.

3.4 Analysis parameters

Mean and standard deviation were used as primary indicators for description elements:

$$\mu_i = \frac{\sum_j^N x_{ij}}{N}, \sigma_i = \sqrt{\frac{\sum_j^N \left(x_{ij} - \mu_i\right)^2}{N}} \tag{3}$$

where x_{ij} are the extracted values of the i-th description element (column of the data matrix) and N is the number of investigated media objects. To identify the distribution of values of description elements over N media samples, the coefficients of the data matrix were quantised to ten bins. For the hierarchical cluster analysis a single-linkage algorithm with squared Euclidean distance measurement was used. The results were depicted as dendrograms on a relative scale from 0 (identical) to 25 (not similar).

SOMs were calculated with a hexagonal layout (every non-border cluster has six neighbours), 15 rows and 15 columns (225 clusters for 314 elements). For cluster adaptation, a Gaussian neighbourhood kernel was used. Maps were initialised randomly. Learning was done in two iterations. In the first iteration, 10000 learning steps were performed with learning rate $\alpha = 0.05$ and radius 10 (clusters). In the second iteration (fine tuning), 100000 learning steps were performed with learning rate $\alpha = 0.02$ and radius 3. For every dataset 15 separate SOMs were calculated and the best map was chosen by the minimum quantisation error (as defined in [13]). Since the capacity of the SOM-PAK implementation is very limited, only 200 (of 260) randomly chosen Corel images and 200 (of 426) coats-of-arms images could be used for training. See the SOM-PAK handbook [12] for more information on the learning parameters.

For factor extraction a principal component analysis (analysis of the coefficients of the correlation matrix) was used [14]. All Eigenvalues greater than one were selected as factors. To simplify interpretation, a Varimax-rotation was performed on the factor loadings matrix. Factor analysis can only be applied on elements with existing variance. Therefore, for the Brodatz dataset 225 elements could be used, for the Corel dataset 311 and for the coats-of-arms dataset 310. For the remaining elements, the description extraction algorithms came up with exactly the same values independent of the analysed media content.

4. Redundancy analysis

Section 4, 5 and 6 contain the results of the redundancy analysis, the sensitivity analysis and the completeness analysis. The first subsection of each section describes the goals of the analysis and names the methods used to extract the required information. Analysis results are described in the second subsection of each section. Readers that are mainly interested in the *interpretation* of the statistical results may jump directly from the first to the third subsection, where conclusions are drawn from the analysis results.

4.1 Scope

In this analysis we are trying to identify whether the description elements extracted from visual content are *unique* or not. Redundancy information is highly valuable for two major reasons. It may influence how descriptors are organised in description schemes (efficiency of application). It is obviously not desired to combine certain descriptors to a description scheme if it is well known that the descriptors are highly redundant for the concerned media class. Additionally, it can be used as a supplementary method to the MPEG-7 binary format (BiM [16]) for compression of descriptions (e.g. for specific classes of content). This helps to further reduce the amount of resources needed for storage and transmission in visual information retrieval systems (efficiency of representation).

Since the content-based algorithms used in the MPEG-7 descriptors were not evaluated with statistical methods, it is likely that they contain a considerable amount of redundancy. Our interest goes to four types of redundancy: firstly, the general redundancy of all elements, secondly, the redundancy of elements that belong to the same descriptor, thirdly, elements that are unique for all other elements and finally, complementary elements that show "inverse" redundancies (significant negative correlations) to other elements.

Three of the statistical methods named in Subsection 2.2 provide helpful information to identify redundancy: the hierarchical cluster analysis and the topological cluster analysis provide information on the redundancy of elements, the factors extracted in the factor analysis reveal the general redundancy of elements and the rotated factor loadings matrices allow to identify elements that are positively or negatively correlated (redundant) or independent from all other description elements. In general, factor analysis provides four outputs: the number of extracted factors, the factors, the amount of variance explained by each factor and the factor loadings matrix. The coefficients of this matrix express to which extent the factors influence the elements from which they are derived. Elements that load high on the same factor have a similar variance. If a factor loads high on two elements but with opposite signs (factor loadings are element of [-1, 1]), then these elements are highly un-similar and suitable for being used in combination. This holds also for elements explained by factors that do not load on any other element.

4.2 Analysis results

A first striking result revealed by the hierarchical cluster analysis (visualised in dendrograms) is the high self-similarity of the elements of the *Homogeneous Texture* descriptor for any type of media (see Table 1). For the Brodatz dataset (rich textures) and the coats-of-arms

Descriptor	*Media collection*	*No. of clusters*	*Maximum distance between clusters*
Homogeneous Texture	Brodatz, coats-of-arms	1	4%
	Corel	2	20%
Edge Histogram	any	5-10	12%-20%
other	any	>5	>20%

Table 1. Results of hierarchical cluster analysis: number of clusters and distance between clusters. The maximum distance is given in percent (where 100% would be the distance of a vector of "0" values to a vector of "1" values). Only *Homogeneous Texture* and *Edge Histogram* descriptions have unbalanced cluster structures.

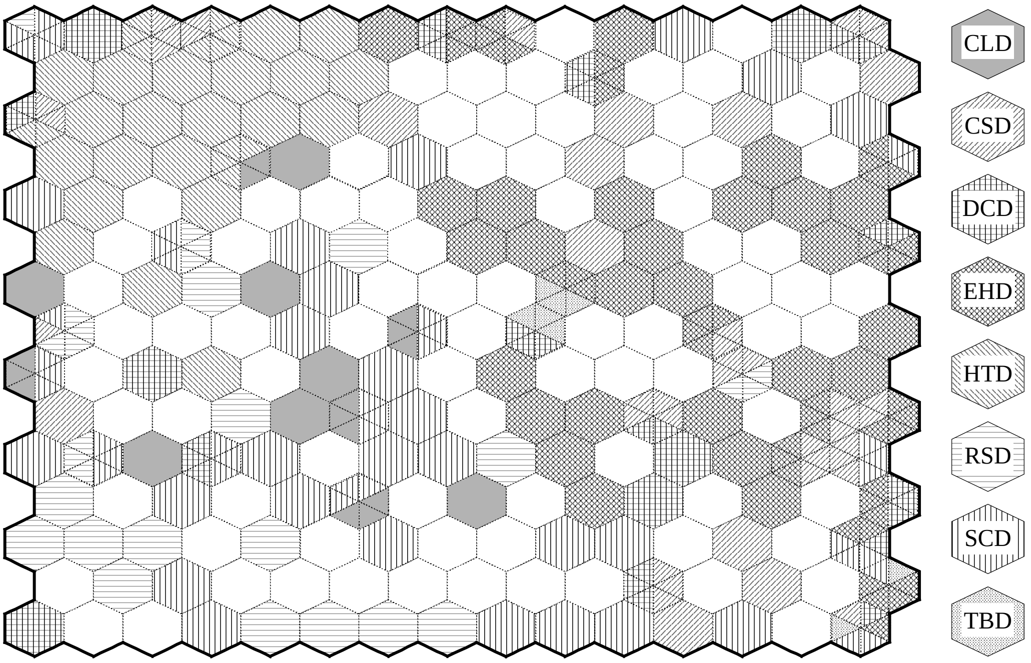


Fig. 2. Self-Organizing Map of MPEG-7 description elements for the coats-of-arms dataset. Neighbouring clusters contain similar description elements. Since every non-border cluster has six neighbours, clusters are shown as hexagons. Cluster populations are depicted as textures (CLD: *Color Layout*, CSD: *Color Structure*, DCD: *Dominant Color*, EHD: *Edge Histogram*, HTD: *Homogeneous Texture*, RSD: *Region Shape*, SCD: *Scalable Color*, TBD: *Texture Browsing*). If clusters are shared between descriptors, hexagons are split into triangular regions.

dataset (poor textures) all description elements form a single cluster with a maximum distance of 4%. For the Corel dataset the descriptor forms two clusters, where the larger one has the same characteristics as for the other two media sets. Only some energy values but no energy deviations fall apart. Interestingly, the *Edge Histogram* descriptor forms five to ten clusters with ten to 15 elements for any type of content. The elements of these clusters are self-similar but the distance between the clusters is relatively large. This descriptor (edge histograms with five bins for 16 predefined rectangular regions) seems to describe areas with similar texture (one or more regions) by sets of highly similar elements (edge bins). Additionally, some elements of *Region-based Shape* and *Scalable Color* form smaller clusters but most elements are – from the one-dimensional point of view – not very redundant.

Looking at the two-dimensional Self-Organizing Map (SOM) clustering gives a "topological" view of the data. Projecting the high-dimensional data vectors onto a map, the clustering algorithm has two degrees of freedom to arrange elements. Therefore, *relationships* between elements can more easily be visualised, and some elements may be grouped closer to each other than they would be in a one-dimensional cluster analysis.

Analysing the SOMs for the three media collections (Figs. 2, 3 and 4 show the maps for the coats-of-arms, Brodatz and Corel dataset, respectively) supports the first impression of the hierarchical analysis. *Homogeneous Texture* lays a fine-meshed net over the investigated media property. *Homogeneous Texture* consists of 15 to 20 clusters (independently of the media content). Each cluster contains three to five elements. These clusters form a homogeneous super-cluster within the map: most clusters are connected to at least one other cluster and the border of the super-cluster is nearly

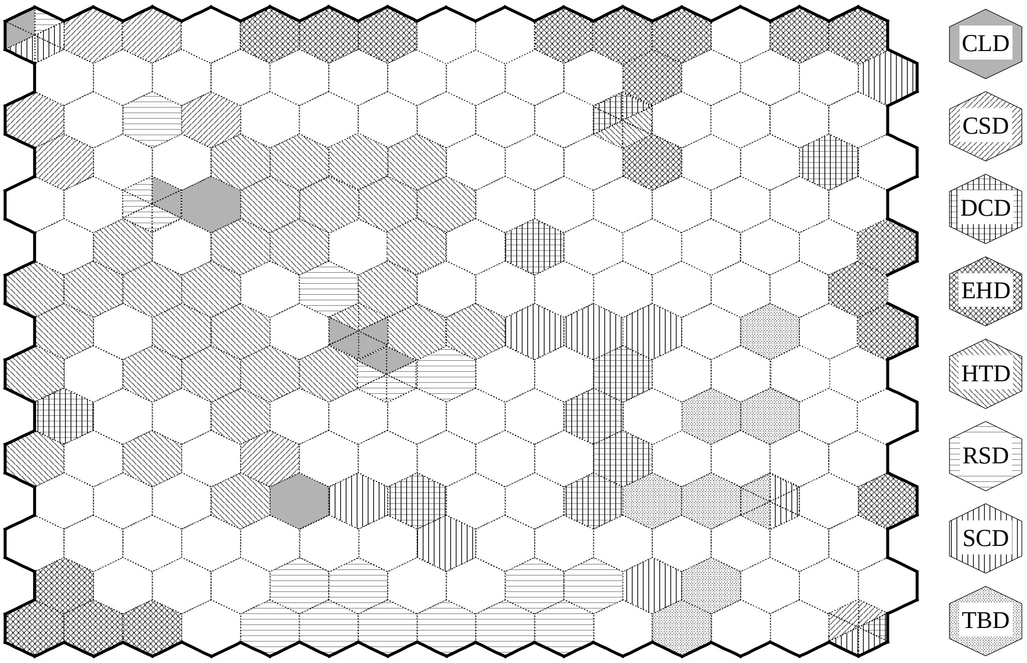


Fig. 3. Self-Organizing Map of MPEG-7 description elements for the Brodatz dataset (see Fig. 2 for description).

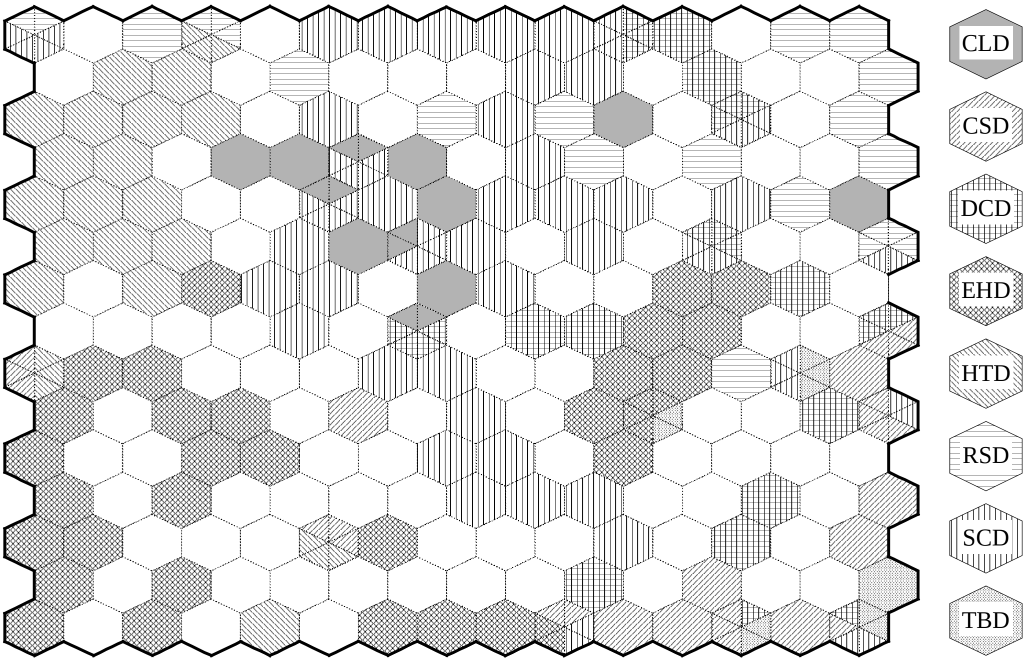


Fig. 4. Self-Organizing Map of MPEG-7 description elements for the Corel dataset (see Fig. 2 for description).

circular. The *Edge Histogram* descriptor forms clusters that contain slightly more elements than *Homogeneous Texture* clusters. These clusters are spread over large regions and only loosely connected. Therefore, the net of the *Edge Histogram* is wide-meshed but the descriptor covers a larger area of the variance in the media data. All other descriptors form rather small two-dimensional clusters for any type of content. These clusters are spread over the entire maps.

The results of the cluster analyses give a first indication on redundant descriptors. A more detailed view can be obtained from the factors extracted by factor analysis algorithms (see Table 2). For the Brodatz dataset 34 factors explain 225 description elements (the remaining elements have zero variance). This is a relationship of nearly 7:1. Applied on the Corel dataset, coloured content with rich details, the MPEG-7 descriptors are redundant with a ratio of about 9:2. This is surprising

Media collection	*Elements with existing variance*	*Factors*	*Explained variance (all)*	*Explained variance (first factor)*	*Redundancy relationship*
Brodatz	225	34	89%	15%	7:1
Corel	311	69	85%	12%	9:2
Coats-of-arms	310	71	80%	6.7%	9:2

Table 2. Results of factor analysis: number of extracted factors and explained variance. Only elements with existing variance are considered.

because the Corel photos include brilliant colours, significant edges and textures and – many of them – tailor-made object arrangements for the *Region-based Shape* descriptor. For the coats-of-arms dataset 71 factors explain 310 elements. The ratio of redundancy is again 9:2. It is surprising that the MPEG-7 descriptors perform slightly worse on the coats-of-arms dataset than on the Corel dataset. The Corel photos contain more details and, generally, descriptors should be less redundant on material with richer content. The reason for the contrary results may lie in the characteristics of the coats-of-arms dataset that is – with respect to its visual properties – positioned between the Brodatz and the Corel collection (few colours, no colour gradations, long sharp edges, hardly any textures, large regions of uniform colour).

The rotated factor loadings matrix expresses, to which (linear) extent factors influence description elements. Therefore, the coefficients of this matrix can be used to find redundant elements. Elements that are loaded by the same factor are redundant. Elements that are loaded by just one factor are independent.

For the Brodatz dataset the first factor loads high on the DC value for the luminance colour component (Y-DC) of *Color Layout*, the second colour bin of *Dominant Color* (the first dominant colour is white), half of the energy values and deviations of *Homogeneous Texture*, almost all bins of *Region-based Shape* and bin 7, 9, 13 of *Scalable Color*. The latter bins are responsible for greyscale pixel. These elements are highly redundant and, for example, the Y-DC could be used as a good indicator for them. Factors 2, 4-6 and 8 (explaining 9%, 8%, 7%, 5%, 4%) measure the five edge types of the *Edge Histogram*. The values for each edge type are highly similar and using a global edge histogram (as defined in the standard) could be a good idea on content comparable to the Brodatz dataset (monochrome, high contrast). The third factor (explains 8%) loads on the half of the elements of *Homogeneous Texture* that is not explained by the first factor. These bins (mainly 4-10, 19-24) are highly redundant.

For the Corel dataset the first factor loads high on non-directional edges (*Edge Histogram*) and almost all elements of *Homogeneous Texture*. This supports the impression that *Homogeneous Texture* is highly redundant. Similarity of non-directional edges can be easily explained by the applied extraction algorithm [15] that tends to classify complex textures as non-directional edges. The second factor (7%) loads high on the Y-DC coefficient of *Color Layout* and most elements of *Region-based Shape* (as for the Brodatz dataset). Surprisingly, the first colour bin and all *Region-based Shape* elements seem to be highly correlated *independently* of the complexity of the media content. Other factors do not show significant redundancies of elements. For example, for complex media content the edge bins of the *Edge Histogram* seem to be very different from each other. Therefore, using a global histogram on complex content may not be a good idea.

For the coats-of-arms dataset the first factor loads high on non-directional edges and half of the *Homogeneous Texture* bins. Even though these media objects hardly contain texture information, non-directional edges and energy bins are highly correlated. This allows the conclusion that these elements are highly redundant and non-directional edges may be used as a substitute for *Homogeneous Texture*. Looking at the cluster analysis results reveals that, indeed, these elements are clustered close to each other. Factor 2 and 3 (each explaining 4% variance) load high on various colour bins of the colour descriptors. This is not surprising as coats-of-arms images mainly consist of large coloured areas. A significant correlation of bins of the same edge type could not be identified.

Another interesting result of the factor analysis is that – for any type of content – the *Dominant Color* descriptor has the tendency to identify colours with identical colour component values. According to the used parameters (see Subsection 3.4) dominant colours are described in the YCrCb-colour space. Somehow, independent of the hue, the values of Y-, Cr-, and Cb-components of dominant colours are most times highly similar. Maybe certain characteristics (e.g. quantisation) in the extraction algorithm implemented in the MPEG-7 experimentation model cause this phenomenon.

4.3 Interpretation

Several observations can be made from these analysis results. Generally, the MPEG-7 descriptors generate results of high redundancy. The relationship of description elements to redundancy-free factors varies from 4:1 to 7:1. If the MPEG-7 descriptors are used in a situation, where storage capacity or network bandwidth is scarce, it may be a good idea, in addition to using the BiM format [16], to make use of transformations for data compression (e.g. Karhunen Loewe transformation [4]). Especially, all MPEG-7 descriptors are highly redundant for monochrome media content. This is not very surprising, because four of eight investigated descriptors are colour descriptors and some implemented extraction algorithms work inferiorly on luminance information alone (see Section 5). Of course, this is a problem if MPEG-7 colour descriptors should be applied on media objects with monochrome content (e.g. for archival of old movies or drawings). If enough colour information is present in the media content, *Color Layout*, *Color Structure* and *Scalable Color* are independent of each other and other descriptors. The *Dominant Color* descriptor is – for any type of content – absolutely independent of all other colour descriptors and shows no similarities to texture descriptors either.

Another interesting result is that all bins of *Color Layout* are highly un-similar for any type of media content and independent from all other elements. In every map and every factor loadings matrix a separate cluster/factor can be found for any element of *Color Layout*. For all types of media, the luminance DC coefficient of *Color Layout* determines most elements of *Region-based Shape*. This element seems to be a good indicator for global shape information even for complex scenes (as *Region-based Shape* should be).

The elements of the *Homogeneous Texture* descriptor are – independent of the media – highly self-similar and redundant. Therefore, the measured property could be expressed with much fewer description elements. For example, the non-directional edges of the *Edge Histogram* descriptor could be used instead of *Homogeneous Texture*. Similarly, *Edge Histogram* consists of clusters of redundant elements. *Texture Browsing* is independent of all other descriptors.

Color Layout, *Dominant Color*, *Edge Histogram* and *Texture Browsing* are the most independent descriptors. This is supported by the factor loadings. *Color Layout*, *Dominant Color*, *Edge Histogram* and *Texture Browsing* are mainly explained by unique factors (even though *Dominant Color* and *Edge Histogram* contain a certain amount of self-similarity). If relationships exist, they are highly negative (e.g. the AC coefficients of *Color Layout* and all elements of *Texture Browsing* are pair-wise highly negatively loaded). Most elements of the other descriptors depend on factors associated with these descriptors.

In conclusion, the ideal – content-independent – description scheme for visual content seems to be *Color Layout* (because of the luminance DC coefficient), *Dominant Color*, *Edge Histogram* and *Texture Browsing*. This description scheme covers most properties measured by the visual MPEG-7 descriptors with a minimum of redundancy. Still, the other descriptors may be meaningful in specific situations and application scenarios. Additionally, the results of the factor analysis suggest that the characteristics of the coats-of-arms dataset supplement the properties of the two other collections substantially.

5. Sensitivity analysis

5.1 Scope

The MPEG-7 standard has been defined for a wide range of applications on any kind of visual content. This analysis tries to give indication on the sensitivity of the descriptors on varying media content. In detail, three forms of sensitivity are investigated: firstly, sensitivity of colour descriptors for monochrome content, secondly, sensitivity of colour descriptors for content with few colour shades (e.g. animations) and finally, sensitivity of the texture descriptors and *Region-based Shape* for coarse, medium and fine structures in the content.

Ideally, the descriptors should provide surjective mappings from visual content to feature space. These mappings should be robust against variations in the quality of the content (e.g. presence of colour information, resolution). Analysing the sensitivity allows the judgement to which extent "bad" (e.g. bleached) input affects the data quality of the descriptions. Even more important, it can be judged whether or not the descriptors are really suitable for the proposed applications. For example, if the descriptions extracted with colour descriptors are meaningless for content with just one colour channel, then these descriptors are obviously not suitable for the archival of digitised old movies.

Two statistical methods provide valuable indicators to judge sensitivity. Mean and standard deviation of element values describe location and distribution of elements. If the extracted descriptions are distributed over a wide range of values, the associated media objects are easy to distinguish. If all objects of a specific content type fall into the same range, the extraction methods are obviously not sensitive to the properties that make the considered media objects distinguishable. Additionally, the distribution of clusters (computed for various types of content) can be used to judge the sensitivity of the descriptors. In the next subsections the results for all investigated descriptors and the three tested media collections are described.

5.2 Analysis results

The main indicators for sensitivity are mean and standard deviation. For a uniformly distributed element on the interval [0, 1] with a mean of 0.5, the maximum

Descriptor	*Media collection*	*Average mean*	*Average standard deviation*
Color Layout	Brodatz	0.7	0.1
	Corel	0.55	0.2
	Coats-of-arms	0.65	0.15-0.2
Color Structure	Brodatz	0.85-0.9	0.05-0.15
	Corel, coats-of-arms	0.5	0.25
Dominant Color	any	0.45-0.5	0.3
Scalable Color	Brodatz	0.4	0.3
	Corel	0.5	0.3
	Coats-of-arms	0.4-0.5	0.15

Table 3. Average mean and standard deviation of colour description elements. Only elements with existing variance are considered in the averaging process. The mean should be around 0.5 and the standard deviation should be 0.2 or higher (max. 0.346 for uniformly distributed elements).

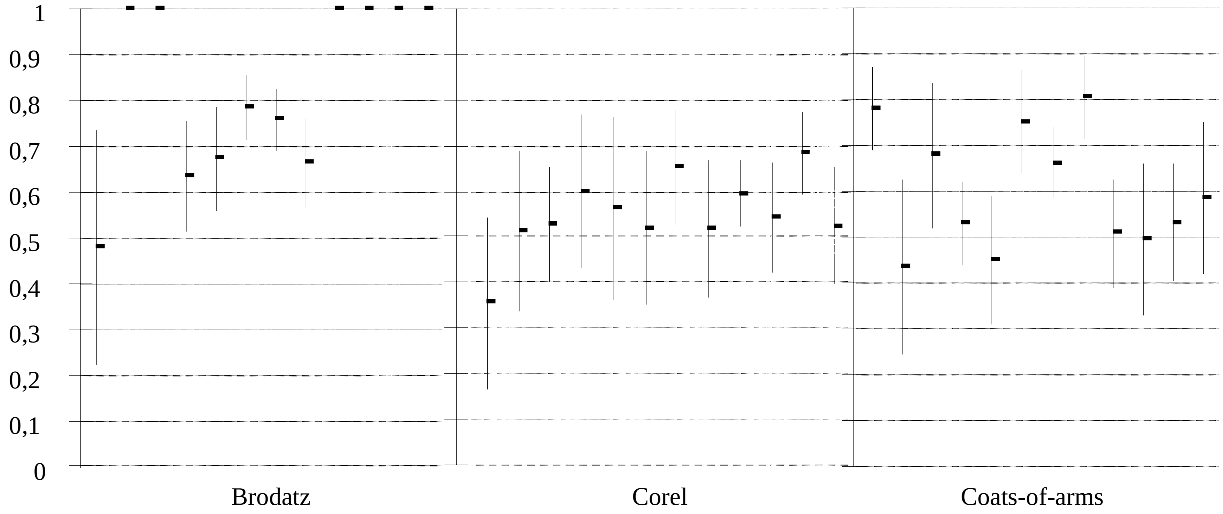


Fig. 5. Mean (depicted as "-") and standard deviation (depicted as vertical lines) of the elements of the *Color Layout* descriptor (twelve elements, shown on the horizontal axis) for the tested collections. The bin values are normalised to [0, 1].

standard deviation is 0.346. In the evaluation the standard deviation should be 0.2 or higher (using at least an interval of 40% of the data range for 66% of all media objects) to be acceptable. Then, the description element can be considered as being sufficiently discriminant to distinguish media objects independently of variations in the content.

Table 3 summarises the average means and standard deviations of the colour description elements. *Color Layout* performs badly on monochrome data (Brodatz dataset). Only six of twelve bins have a standard deviation greater than zero: the DC and AC coefficients of the luminance channel (Fig. 5). Even for these, the standard deviation is very low. For the datasets with colour information the descriptor works well: the standard deviation is about 0.2. Like *Color Layout*, *Color Structure* performs inferiorly on monochrome data: 24 of 32 colour bins have no variance, the other bins are apparently responsible for brightness (bins 37 to 44). Even these bins have a very low variance. Whenever colour is present – independent of the number of gradations – *Color Structure* performs excellently. The standard deviation is in average 0.25. Therefore, the element values are distributed over the entire range of possible values. As can be seen from Table 3, the *Dominant Color* descriptor performs equally well on any type of media content. The distribution of values is very similar for any type of media. *Scalable Color* performs exactly like *Color Layout* and *Color Structure*. For monochrome content, *Scalable Color* is not able to derive meaningful descriptions. Only eight bins have an existing variance (bins 1-3, 5, 9, 13, 33 and 49). For the

Descriptor	*Media collection*	*Average mean*	*Average standard deviation*
Edge Histogram	any	0.5	0.25-0.3
Homogeneous Texture	Brodatz	0.65-0.7	0.1
	Corel	0.75	0.1
	Coats-of-arms	0.75	0.05
Texture Browsing	Brodatz, Corel	0.2-0.3	0.2-0.25
	Coats-of-arms	0.1	0.05
Region-based Shape	any	0.5	0.2-0.25

Table 4. Average mean and standard deviation of texture and shape description elements.

Corel dataset, *Scalable Color* results are excellent: standard deviation of 0.3. For less colour gradations than in photos, many standard deviation values drop down below 0.15. This means that *Scalable Color* is hardly discriminant for synthetic content.

Edge Histogram performs excellently on any type of media (see Table 4 for details on texture and shape descriptors). Even on coats-of-arms images with hardly any textures present (but, of course, very sharp edges) the average standard deviation is above 0.25. For different content the statistical indicators are even better. The *Homogeneous Texture* descriptor works poorly on colour images, especially if they have few colour shades and textures in them. In this case, the standard deviation of most elements drops below 0.05. The descriptor performs slightly better on the Brodatz dataset. This means, *Homogeneous Texture* is not discriminant for colour media and still relatively poor for texture regions. *Texture Browsing* (in the form described in Subsection 3.1) performs well on the Brodatz dataset and the Corel dataset but poorly on the coats-of-arms dataset. Finally, the *Region-based Shape* descriptor measures excellently on any type of media. The standard deviations are in average 0.2 to 0.25.

These findings are supported by the cluster analysis results. Most clusters are on distance level lower than 20%. Hardly any clusters exist at average distance (20% to 60%). Cluster structure and clusters size varies widely for different content. An interesting phenomenon can be observed in the Self-Organizing Map of the Brodatz dataset (Fig. 3). On such content (sharp edges), the *Edge Histogram* descriptor should be highly discriminant, but in the map most clusters are border clusters. This suggests that descriptions generated by the *Edge Histogram* are extreme (in terms of variance) and not as discriminant as they should be.

5.3 Interpretation

The analysis results can be summarised as follows. All colour descriptors work excellently on photos but *Color Layout*, *Color Structure* and *Scalable Color* perform badly on artificial media objects with few colour gradations and very badly on monochrome content. *Dominant Color* is an exception. This descriptor works well on any type of content even though it is particularly sensitive to brightness. A solution for using colour descriptors on media objects with a single colour component could be storing, transporting and utilising only the bins that are sensitive for brightness (e.g. using special distance measures for retrieval that take only these elements into account).

The statistical properties of the colour descriptors cause one major side effect on other descriptors. The *Group-of-Frames/Group-of-Pictures Color* descriptor (*GoF/GoP*) is based on *Scalable Color*. It is intended to generate descriptions for short video clips and animations. The description is computed by taking the mean over all colour histograms of media objects in a group. Since the averaging process does not introduce new information, *GoF/GoP* descriptions cannot contain valuable information, if the *Scalable Color* descriptions for the frames do not contain valuable information. Therefore, obviously, this descriptor does not work if the given media does not consist of nicely shot pictures (high image sharpness, good lighting conditions, rich colours and shades). For example, it cannot be used on animations, old monochrome movie clips, cartoons, etc. The same would be true if *Color Layout* or *Color Structure* was used. From the statistical results it has to be concluded that *GoF/GoP* should be based on *Dominant Color* instead of *Scalable Color*. For monochrome content an implementation of this descriptor would be straightforward, because colours based on one component can easily be averaged. For colours with three components the implementation would be a bit more complicated, because it would be necessary to identify corresponding colours. Still, the problem would be solvable, if domain knowledge on colour models would be integrated in the averaging process.

Additionally, for retrieval applications the distance measures recommended in the MPEG-7 standard for colour descriptors should be refined. For monochrome

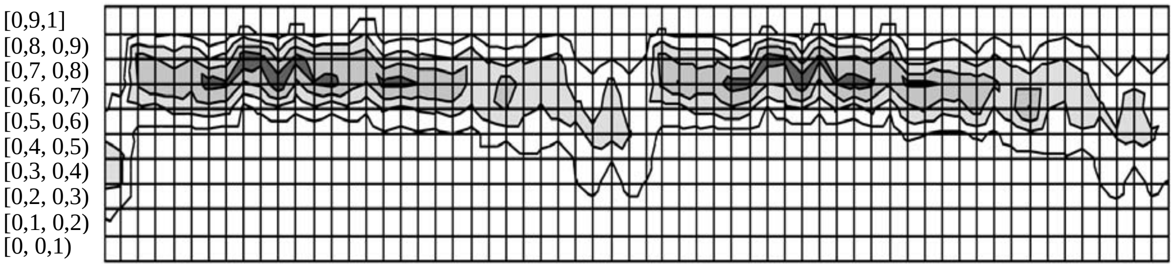


Fig. 6. Topological visualisation of the discrete density function of the *Homogeneous Texture* descriptor applied on the Corel dataset (ten bins on the vertical axis). Description elements (energy values and distributions) are depicted on the horizontal axis. White: 0-10 percent of examples in this density histogram bin, 20% grey: 10-20 percent, 40% grey: 20-30 percent, 60% grey: 30-40 percent, black: 40-50 percent.

content they should exclusively measure bins with existing variance (see Subsection 5.2). Otherwise, media objects with one colour channel would be rated more similar to query examples than the same media objects with three colour channels.

In conclusion, from the statistical point of view, *Edge Histogram* is by far the best texture descriptor (high sensitivity, low redundancy). *Homogeneous Texture* is highly sensitive to the analysed media content and the variance of results is small. *Texture Browsing* produces partially ambiguous results (because of poor variances, few elements) that are, indeed, suitable for browsing but not for other MPEG-7 applications (e.g. retrieval). *Region-based Shape* is a good descriptor in any situation that can be applied to any type of media content.

6. Completeness analysis

6.1 Scope

Ideally, for sufficiently large media collections with varying content, the values of each description element should be uniformly distributed over the available data range (in our case [0, 1]). Such description elements would utilise the data space optimally and discriminate media objects to the largest possible number of descriptions. Practically, most descriptors have peaks in the distribution and "holes": ranges of data values that are not utilised at all. This analysis aims at finding holes in the descriptions that are not covered by any descriptor and therefore, cannot be filled with description schemes either. In detail, three properties are investigated: firstly, to which extent the available data range is used, secondly, if the element values are uniformly distributed or whether or not peaks exist and finally, whether the structure of all description elements contains holes or not.

Identifying shortcomings in the completeness of the visual MPEG-7 descriptors has three advantages. The descriptions can be compressed by quantising elements that do not make use of the available data range to smaller data types. If peaks exist (e.g. Gaussian-distributed elements), transformations can be applied on the data to increase the discriminance of the descriptors. Finally, by identifying holes between elements suggestions can be made for new descriptors that could supplement the MPEG-7 standard.

Three statistical methods are used to judge completeness. The distribution of element values is used to identify holes and the distribution type (extraction is explained in subsection 2.2). Four distribution types are distinguished: uniform, Gaussian, exponential and irregular (else). Additionally, mean values and standard deviations can give hints on holes in the data distribution. Finally, cluster analysis algorithms are used to detect holes between element clusters.

6.2 Analysis results

From the distribution of element values, it can be seen that the *Color Layout* descriptor tends to cover just 90% of all possible values. The values in the interval [0, 0.1] are hardly used (values are normalised to [0, 1]). For the Brodatz dataset, all values are higher than 0.3. For the Corel dataset and the coats-of-arms dataset the situation is less bad but still not optimal. Values below 0.1 hardly ever occur. All other values are sufficiently utilised. Most energy values and deviations of *Homogeneous Texture* measure exclusively on the range [0.5, 1] (Fig. 6). Except for a few bins for colour photos the first half of values is not utilised.

An interesting phenomenon can be observed for the *Edge Histogram* descriptor (Fig. 7). Independent of the type of media this descriptor does hardly measure on the ranges [0.26, 0.35], [0.55, 0.65] and [0.85, 0.95]. Since this descriptor simply counts edges of certain orientations in pre-defined spatial regions, this behaviour must be caused by the (non-normative) extraction process implemented in the MPEG-7 experimentation model. A possible explanation could be the simple edge operators (*2x2* matrices) that are used. In any case, this phenomenon allows extended compacting of the descriptor. About 30% of the available data range is not

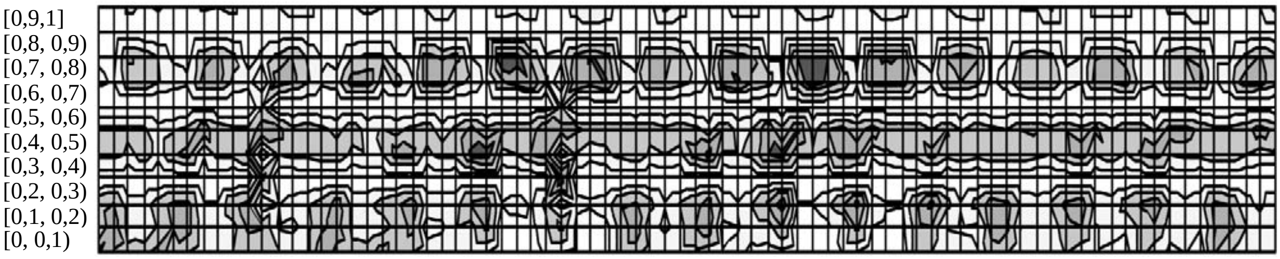


Fig. 7. Topological visualisation of the discrete density function of the *Edge Histogram* descriptor applied on the Corel dataset. Description elements (edge bins) are depicted on the horizontal axis. White: 0-6 percent of examples in this density histogram bin, 20% grey: 6-12 percent, 40% grey: 12-18 percent, 60% grey: 18-24 percent, black: 24-30 percent.

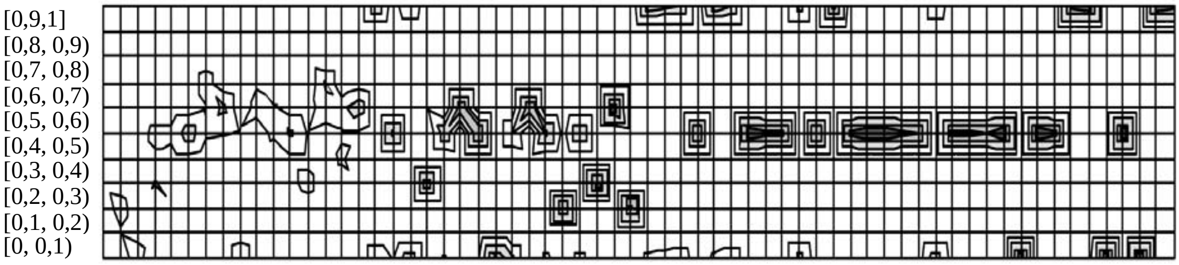


Fig. 8. Topological visualisation of the discrete density function of the *Scalable Color* descriptor applied on the Corel dataset. Description elements (colour bins) are depicted on the horizontal axis. White: 0-20 percent of examples in this density histogram bin, 20% grey: 20-40 percent, 40% grey: 40-60 percent, 60% grey: 60-80 percent, black: 80-100 percent.

used. The same phenomenon as for *Edge Histogram* can also be observed for *Scalable Color* (Fig. 8). For any type of media (except for a few of the first bins) the ranges [0.1, 0.2] and [0.35, 0.45] are not used at all; [0.7, 0.9] is hardly used. This may have to do with the application of the Haar transformation [15]. All other investigated descriptors do not show significant holes.

One possible reason for holes in the data range could be the distribution type of data. If the data range is non-uniformly distributed, peaks exist and not all bins of the distribution are utilised to the same amount. For any type of media, all elements of *Color Layout* (with existing variance) follow a Gaussian-like distribution. The distribution of most *Color Structure* elements is irregular (neither uniform nor Gaussian nor exponential). Only a few elements are distributed in Gaussian-like way. Most bins of *Dominant Color* with percentage values are Gaussian-distributed while the colour bins are mostly irregularly distributed. This may have to do with the phenomena described in subsection 4.2 that the colour component values seem to be coupled. Only a few elements of the descriptor are uniformly distributed. Nearly all elements of *Edge Histogram* and *Scalable Color* have an irregular distribution. Most elements of the *Edge Histogram* descriptor have two significant peaks in the intervals [0, 0.1] and [0.8, 0.9]. Nearly all description elements of *Homogeneous Texture* seem to be Gaussian-distributed for any type of content, and because of the low standard deviations only a few bins are utilised. The elements of *Texture Browsing* are irregularly distributed. Finally, most elements of *Region-based Shape* are uniformly distributed. The others are Gaussian-distributed.

Looking at the hierarchical cluster structure reveals that hardly any clusters exist on average distance (20% to 60%) for any type of media. Most clusters are on distances lower than 20%. The size of these clusters varies widely. This supports the conclusion that the elements of most descriptors are far from being uniformly distributed. If they were, a lot of clusters with similar sizes should exist on average level and the dendrograms should look like well-balanced trees. Generally, descriptors consisting of uniformly distributed elements should be more efficient for retrieval and browsing than descriptors consisting of non-uniformly distributed elements (see Section 8).

Looking at the Self-Organizing Maps shows that – as described in section 4.2 – *Homogeneous Texture* spans a fine-meshed net, but only over a small area of the available variance (and therefore, for the price of high redundancy). *Edge Histogram* spans a large wide-meshed net with large clusters. This bears the risk that media objects that are only slightly different may be judged equal in one case and completely different in another. On the other hand *Edge Histogram* is generally less redundant. The topology of the Brodatz cluster map shows large holes between the *Homogeneous Texture* descriptor and the *Edge Histogram* descriptor. One obvious reason for the large holes are the colour histogram features that fill these holes for other content but do not work on monochrome content.

For the coats-of-arms dataset the same problem occurs as for the Brodatz dataset (Figs. 2, 3). Large holes can be found between *Edge Histogram* and *Homogeneous Texture*, because *Scalable Color* is missing: *Scalable Color* is not discriminant for content with few color shades. The extraction algorithm of this descriptor should be more sensitive, if only few colour gradations are present. *Scalable Color* generates a mesh that is very similar to *Edge Histogram*: large clusters and widely meshed. To the author's opinion, using this descriptor with more bins (e.g. 256) would not improve the cluster structure because of the large clusters. For the Corel dataset only comparably small holes do exist (Fig. 4). The large holes are filled by elements of *Scalable Color*. The outcome of *Region-based Shape* is – for any type of content – mostly widely meshed with large clusters. Finally, *Color Structure* has smaller clusters but very large holes in the cluster distribution. For this descriptor, using more colour bins could improve the cluster structure and reduce the size of holes (or increase the size of clusters). The other descriptors have too few elements to allow conclusions on the cluster structure.

6.3 Interpretation

In addition to using the binary MPEG-7 format, most descriptor values could be transformed and quantised to smaller data types. For the *Color Layout* descriptor this could save (depending on the content) at least 10% of storage space. For the *Homogeneous Texture* descriptor the amount of required storage could be reduced to 50% using an appropriate transformation and quantisation, for the *Edge Histogram* descriptor to 70% and for the *Scalable Color* descriptor to about 45%. In conclusion, for example, if any element of a complete MPEG-7 description would be represented by a floating point variable with double precision (eight bytes), the amount of needed storage could be reduced from 2512 bytes to 1832 bytes (73%) *without* losing precision in the data.

Nearly no holes exist between clusters of colour descriptors. If they work, these descriptors are very independent from each other and generate a fine-meshed structure with clusters of – at most – average size. For the texture descriptors large holes exist between *Edge Histogram* and *Homogeneous Texture,* if *Scalable Color* is not able to discriminate colour shades. *Texture Browsing* is – because of its poor variance – unable to close these holes. Two approaches look promising to close the holes between *Edge Histogram* and *Homogeneous Texture* for content with poor colour information: changing the algorithms of the colour histograms to make them sensitive for greyscale media objects, and modifying the "neighbouring" descriptors, especially *Edge Histogram*. *Edge Histogram* should produce a more homogeneous cluster structure with smaller clusters. This could be achieved by making the extraction mechanism more sensitive for small differences in the content (e.g. by using less-coarse edge operators). Since these changes would not affect the normative part of the MPEG-7 standard, both solutions could easily be implemented. Finally, if the *Scalable Color* descriptor is used in combination with the best other descriptors on content with rich colours, then the resulting descriptions have a fine-meshed cluster structure and even small differences in the content are mirrored by the descriptions.

7. Summary

Table 5 summarises the results presented in sections 4, 5 and 6. The three datasets were selected as representatives for their respective media classes:

- Brodatz dataset: monochrome images with sharp edges and rich textures.
- Corel dataset: colour photos with rich content, detailed textures and many colour shades.
- Coats-of-arms dataset: artificial colour images with clear structures and few colour gradations.

8. Conclusions

The study presented in this paper analyses descriptions extracted from visual content by MPEG-7 descriptors from a statistical point of view. Good descriptors should be invariant against the analysed media content and generate descriptions with high variance and a well-balanced cluster structure. Such descriptions would be highly discriminant and be usable to distinguish different media content. Statistical analysis reveals the quality of the used descriptors. This was not considered in the MPEG-7 design process where optimising the recall (retrieval rate) was the major goal. For the analysis eight basic visual descriptors were applied on three media collections: the Brodatz dataset (monochrome textures), a selection of the Corel dataset (colour photos) and a set of coats-of-arms images (synthetic colour images with few colour shades). The results were analysed with four statistical methods: mean and standard deviation of description elements, distribution of element values, cluster analysis (hierarchical and topological) and factor analysis.

The main results are: The best descriptors for combination are *Color Layout*, *Dominant Color*, *Edge Histogram* and *Texture Browsing*. The others are highly dependent on these. The colour histograms (*Color Structure* and *Scalable Color*) perform badly on monochrome input. Therefore, *Dominant Color* should be used for the *Group-of-Frames/Group-of-Pictures* descriptor instead of *Scalable Color*. Generally, all

Descriptor	*Brodatz dataset*	*Corel dataset*	*Coats-of-arms dataset*
Color Layout	Description bins are independent from each other, luminance DC coefficient explains other descriptors, sensitive to missing colour information, large holes in utilisation of data range	Description bins are independent from each other, luminance DC coefficient explains other descriptors, high variance, small holes in utilisation of data range	Description bins are independent from each other, luminance DC coefficient explains other descriptors, high variance, small holes in utilisation of data range
Color Structure	Highly redundant elements, sensitive to missing colour information	High variance	High variance
Dominant Color	Independent of other descriptors, robust against missing colour information	Independent of other descriptors, high variance	Independent of other descriptors, high variance
Edge Histogram	Edge type bins with equal orientation have very similar variance, wide-meshed cluster structure, high variance, large holes in utilisation of data range	Wide-meshed cluster structure, high variance, large holes in utilisation of data range	Wide-meshed cluster structure, high variance, large holes in utilisation of data range
Homogeneous Texture	Highly redundant elements, fine-meshed cluster structure, average variance, large holes in utilisation of data range	Highly redundant elements, fine-meshed cluster structure, sensitive for present colour information, large holes in utilisation of data range	Highly redundant elements, fine-meshed cluster structure, sensitive for present colour information and missing texture information, large holes in utilisation of data range
Region-based Shape	Highly dependent on luminance DC coefficient of *Color Layout*, high variance	Highly dependent on luminance DC coefficient of *Color Layout*, high variance	Highly dependent on luminance DC coefficient of *Color Layout*, high variance
Scalable Color	Highly redundant elements, highly sensitive to missing colour information, large holes in utilisation of data range	High variance, large holes in utilisation of data range	Sensitive to missing colour shades, large holes in utilisation of data range
Texture Browsing	High variance	High variance	Sensitive to missing texture information

Table 5. Summary of analysis results.

descriptors are highly redundant and applying compression, quantisation and transformation algorithms in addition to using the binary MPEG-7 (BiM) format could save up to 80% of storage and transmission resources. Finally, analysis shows that some aspects of visual media objects are not captured by any of the MPEG-7 descriptors. For content-based retrieval and browsing applications, MPEG-7 descriptions should be augmented by additional descriptors. For example, a histogram of brightness values could be valuable to describe monochrome images. Shape moments could be used to describe the properties of image regions. Additional texture features could supplement MPEG-7 texture descriptors. A colour histogram that is more robust than *Color Structure* and *Scalable Color* could be used in combination with the *Dominant Color* descriptor. Since MPEG-7 offers a descriptor definition language and allows definition of description schemes these and other suggestions could easily be implemented.

The study was prepared for the visual information retrieval project VizIR [5, 6]. VizIR is based on the visual MPEG-7 descriptors and, as a consequence of the presented results, the future implementation focus will lie on *Color Layout*, *Dominant Color*, *Edge Histogram* and *Texture Browsing*.

Acknowledgements

The author would like to thank Christian Breiteneder for his valuable comments and suggestions for improvement. This work and the VizIR project are supported by the Austrian Scientific Research Fund (FWF) under grant number P16111-N05.

References

1. Bober M (2001) MPEG-7 visual shape descriptors. Special issue on MPEG-7, IEEE Transactions on Circuits and Systems for Video Technology 11/6 : 716-719
2. Breiteneder C, Eidenberger H (1999) Content-based image retrieval of coats of arms. Proc IEEE International Workshop on Multimedia Signal Processing, Helsingör : 91-96
3. Chang SF, Sikora T, Puri A (2001) Overview of the MPEG-7 standard. Special issue on MPEG-7, IEEE Transactions on Circuits and Systems for Video Technology 11/6 : 688-695
4. Del Bimbo A (1999) Visual information retrieval. Morgan Kaufmann, San Francisco CA
5. Eidenberger H, Breiteneder C, Hitz M (2002) A framework for visual information retrieval. Proc Visual Information Systems Conference, HSinChu Springer LNCS 2314 : 105-116
6. Eidenberger H, Breiteneder C (2003) VizIR - a framework for visual information retrieval. Journal of Visual Languages and Computing 14 : 443-469
7. Eidenberger H (2003) How good are the visual MPEG-7 features?, Proc SPIE Visual Communications and Image Processing Conference, Lugano SPIE Vol. 5150 : 476-488 (available from http://www.ims.tuwien.ac.at/~hme/papers/vcip2003-mpeg7.pdf, last visited 2004-04-05)
8. Fuhr N (2001) Information Retrieval Methods for Multimedia Objects. In: Veltkamp RC, Burkhardt H, Kriegel HP (ed) State-of-the-Art in Content-Based Image and Video Retrieval. Kluwer, Boston, pp 191-212
9. Guo J, Kuo JCC (2001) Semantic video object segmentation for content-based multimedia applications. Kluwer, Boston MA
10. Jain AK, Murty MN, Flynn PJ (1999) Data clustering: a review. ACM Computing Surveys 31/3 : 264-323
11. Kohonen T (1990) The Self-Organizing Map. Proc IEEE 78/9 : 1464-1480
12. Kohonen T, Hynninen J, Kangas J, Laaksonen J (1995) SOM-PAK: The Self-Organizing Map program package. Technical report, Helsinki University of Technology
13. Kohonen T, Oja E, Simula O, Visa A (1996) Engineering applications of the Self-Organizing Map. Proc IEEE 84/10 : 1358-1384
14. Loehlin JC (1998) Latent variable models: An introduction to factor, path, and structural analysis (3rd edition). Lawrence Erlbaum Assoc, Mahwah NJ
15. Manjunath BS, Ohm JR, Vasudevan VV, Yamada A (2001) Color and texture descriptors. Special issue on MPEG-7, IEEE Transactions on Circuits and Systems for Video Technology 11/6 : 703-715
16. Manjunath BS, Salembier P, Sikora T (2002) Introduction to MPEG-7. Wiley, San Francisco CA
17. MPEG-7 experimentation model website. http://www.lis.e-technik.tu-muenchen.de/research/bv/topics/mmdb/e_mpeg7.html (hosted by TU Munich, last visited 2004-04-05)
18. Smeulders AWM, Worring M, Santini S, Gupta A, Jain R (2000) Content-based image retrieval at the end of the early years. IEEE Transactions on Pattern Analysis and Machine Intelligence 22/12 : 1349-1380

An Experimental Study on the Performance of Visual Information Retrieval Similarity Models

Horst Eidenberger and Christian Breiteneder
Institute of Software Technology and Interactive Systems
Vienna University of Technology
Vienna, Austria
{eidenberger, breiteneder}@ims.tuwien.ac.at

Abstract–**This paper is an experimental study on the performance of the two major methods for macro-level similarity measurement: linear weighted merging and logical retrieval. Performance is measured as the average query execution time for a significant number of tests. The two models were implemented in the standard version (as they are applied in a number of prototypes) and in an optimized version. The results show, that optimized logical retrieval clearly outperforms optimized linear weighted merging.**

Keywords–content-based image retrieval; content-based visual retrieval; visual information retrieval; query optimization; similarity measures; triangle inequality; experimental study

I. Introduction

Content-based retrieval of information from visual media (images and video; CBIR) has been an area of increasing interest and research in the past years ([6]). Up to now, one of the major problems of most CBIR approaches has been bad performance in terms of query execution time. Similarity measurement in CBIR systems is essentially based on distance measurement of feature vectors that have been previously extracted from visual media. Most used distance functions are L_1 and L_2 metrics, e.g. the city block distance and the Euclidean distance. These distance functions have a complexity of at least $O(n)=n$, n being the size of the feature vectors. Most query acceleration approaches follow one of three directions:

1. Indexing of feature data. Indexing method include tree techniques (quadtree, R-tree, etc.) and gridfiles. They suffer from the drawback that most of them support only one inherent distance measure (mostly Euclidean distance) and therefore have to be implemented fore each group of features with common distance measure separately. Additionally, most of them become increasingly ineffective for high-dimensional data.
2. Complexity reduction of feature vectors prior or after the feature extraction process. This includes coarse representation of features (reduced scales or number of histogram-bins, etc.) and redundancy reduction (e.g. factor analysis).
3. Occlusion of media objects to minimize the number of distance comparisons. The most well-known approach from this area is using the triangle inequality (the fourth metric axiom) to exclude dominated media objects (see [1]).

In the paper we investigate methods from the third area: occlusion of media objects that are based on the similarity models used in most CBIR systems. We will compare the performance of the linear weighted merging model (LWM) and the logical retrieval model (LR) in form of a simple and an optimized algorithm (see Section II for details on LWM and LR). The rest of the paper is organized as follows. Section II points out relevant related work, Section III describes the algorithms we used in our experiments, Section IV is a brief sketch on the test environment we used and Section V describes the experimental results.

II. Related work

Subsequently, we will outline the CBIR macro-level similarity measurement process and earlier work on CBIR query acceleration. In [3] we define macro-level similarity measurement as the process that extracts a result set for a query from a given distance space. Distance space is defined as the vector space that is derived from feature space by measuring the distance of media objects to given query examples with distance functions (micro-level similarity measurement). In feature space, media objects are represented as numerical feature vectors.

The two most widely applied methods for macro-level similarity measurement are: (1) linear weighted merging (LWM) and (2) logical retrieval (LR). LWM is a two step process. In the first step, a position value is calculated for each media object (according to equation 1) and in the second, the media objects are ordered by this position value and the first n elements are selected as the result set.

$$Position\,value_{Object} = \sum_{i=1}^{F} w_i d_i \tag{1}$$

In equation 1 d_i and w_i are the distance value and weight for feature i (of F) and the given media object. This equation is a simplified version of the formula given in [6]. It is implemented by most CBIR prototypes (e.g. QBIC, [4]). In opposition to LWM, the result set size in LR is not constant and depends on the given media collection. In LR, each query is a logical expression of terms c_i of the form given in equation 2. The parameters t_{i1}, t_{i2} are thresholds for the minimum

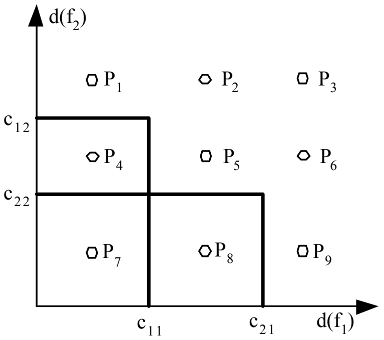


Figure 1. LR conditions and possible locations of media objects P_i.

respective maximum distance of a media object for a certain feature. A media object is added to the result set, if the query expression evaluates to *true* for its distance values. For example, this method is implemented in MARS ([5]).

$$t_{i1} \le d_i \le t_{i2} \qquad (2)$$

In our earlier work, we implemented a simplified version of LR (called QueryModels), where conditions may only be *and*-connected. For this approach we implemented a heuristic optimization technique that tries to order the conditions of an expression in a way, that those conditions (distance functions) are evaluated first that cut off most non-similar media objects and/or use the fastest distance functions. Because of the *and*-connection, consecutive conditions have to take only those media object into account that have not been cut off by prior conditions. This method lead to a reduction of the average query execution time of *66%* (see [2] for details). Subsequently, we will introduce a simple optimization technique for LR expressions that contain the logical operators *and*, *or* and *not*.

III. Used Algorithms

We tested four different algorithms for macro-level similarity measurement: (1) LWM with simple optimizations (referred to as LWM), (2) LWM with triangle inequality optimization (LWM+), (3) LR with no optimization (LR) and (4) LR with a simple optimization technique (LR+). In the tests we used no indexing or complexity reduction techniques, because these methods are optimizations on the micro-level and can be applied to any of the four algorithms. The test plan was as follows:

1. Select query parameters (query example, features, weights, result set size, etc.). The details concerning this step will be described in Section IV.
2. Calculate the result sets for LMW and LWM+.
3. Derive an LR expression from the LWM result set that represents exactly the same result set as the LWM algorithm.
4. Calculate the result sets for LR and LR+.

The used LWM algorithm has the following form (pseudo-code):

```
FOR EACH mo {
   pv:= 0;
   FOR EACH feature {
      dist:= CALC DISTANCE FROM qe TO mo;
      pv:= pv + dist*weight(feature);
      IF pv > distanceSum(n) THEN {
         GOTO break;
      }
   }
   ADD pv to distanceSum;
break:
}
rs:= FIRST n ELEMENTS BY distanceSum;
```

In this algorithm, *qe* is the query example, *mo* is a media object, *pv* is the (partial) position value of *mo* and *distanceSum* is a vector of media object position values (always sorted in ascending order). This algorithm differs from the standard LWM in one point: whenever the partially calculated position value exceeds the position value of the n-th element (result set border), the calculation for this media object is aborted and calculation continues with the next media object.

The LWM+ algorithm uses the triangle inequality technique (TRIQ) for query optimization. The TRIQ is an occlusion technique that can only be applied on distance functions that fulfill the metric axioms (see [1] for details on TRIQ). We use two distance measures that are both metric: city block distance and Euclidean distance. Based on [1] we use equation 3 (joint cutoff criterion) to occlude media objects. In this equation, *r* is a reference object, *q* is the query example, *x* is an arbitrary media object and min_d is the distance value of the n-th element in the result set. All *d(x,r)* values can be calculated *before* the querying process.

$$\left|d(x,r) - d(q,r)\right| > \min_d \qquad (3)$$

Because we are using multiple features and want to retrieve more than one media object, we use an adapted version of TRIQ. The final LWM+ algorithm looks as follows:

```
FOR EACH mo {
   pv:= 0;
   FOR EACH feature {
      IF |refDist(mo)-refDist(qe)| *
         weight(feature) >
         (distanceSum(n)-pv) THEN {
         GOTO break;
      }
      dist:= CALC DISTANCE FROM qe TO mo;
      pv:= pv + dist*weight(feature);
      IF pv > distanceSum(n) THEN {
         GOTO break;
      }
   }
   ADD pv to distanceSum;
break:
}
rs:= FIRST n ELEMENTS BY distanceSum;
```

The similarity measurement for a media object is terminated as soon as it becomes clear that the position value will exceed the position value of the n-th element in *distanceSum*. Using the TRIQ, this can be done prior to the distance calculation.

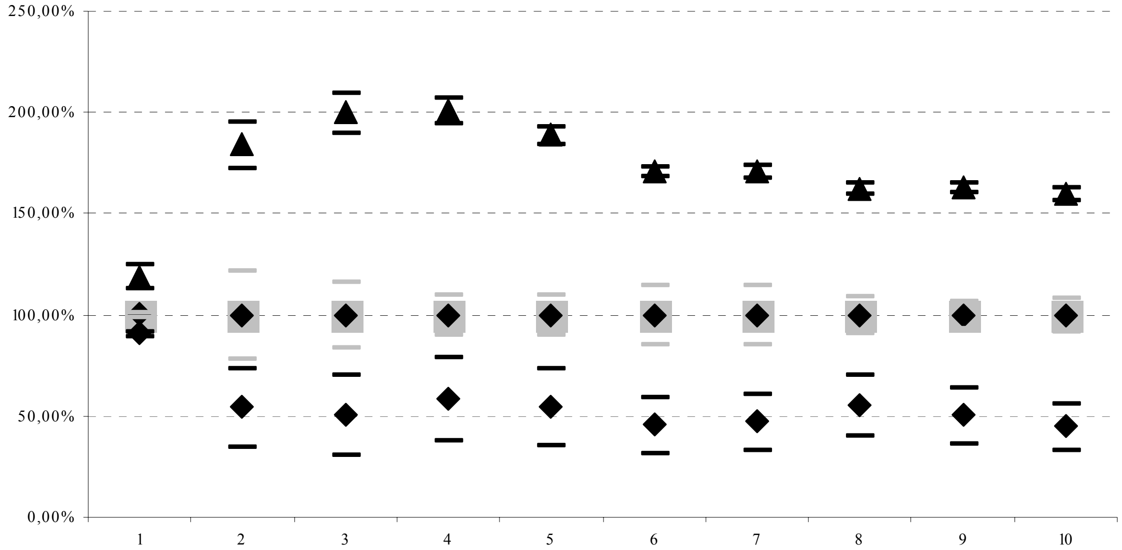


Figure 2. Results for varying number of queried features. The triangles show the average performance of LR, the diamonds stand for the LR+ performance and the gray squares and the diamonds within them depict the performance of LWM resp. LWM+. The lines above and below the icons show the standard deviations.

In our earlier work we have shown that LR is a more flexible model for macro-level similarity measurement than LWM. It is possible to derive an LR expression for arbitrary LWM result sets with the following algorithm:

```
expression:=();
FOR i:=n to 1 DO {
   FOR j:=i-1 to 1 DO {
      IF distVector(i) NOT EXCEEDS
         distVector(j) THEN {
         GOTO break;
      }
      IF distVector(i) EXCEEDS
         distVector(j) THEN {
         DEL distVector(j) FROM expression;
      }
   }
   ADD distVector(i) TO expression;
break:
}
```

Here, *expression* is a vector of all conditions and *distVector(i)* is the distance vector for media object *i*. The idea of the algorithm is to take each LWM result set element *i*, check, if it is included in the expression derived so far and – if not – add *i or*-connected to *expression*. Each added distance vector defines an *f*-dimensional *and*-connected **cube** (*f* features, a cube consists of one LR condition for each feature) where the distance values are the t_{f2} thresholds of LR conditions (see Section II) and the t_{f1} are all *0*. Figure 1 shows an example for two features: (c_{11},c_{12}) and (c_{21},c_{22}) are cubes of conditions and the result set consists of $\{P_4, P_7, P_8\}$. Additionally, the LWM to LR conversion algorithm checks, if new conditions dominate existing ones and – if yes – eliminates the dominated ones. LR querying based on the derived expression is done with the following algorithm:

```
FOR EACH mo {
   distVector:=();
   FOR EACH feature {
      dist:= CALC DISTANCE FROM qe TO mo;
      ADD dist to distVector;
   }
   FOR EACH condition {
      IF distVector EXCEEDS expression THEN {
         GOTO break;
      }
   }
   ADD mo TO rs;
break:
}
```

The optimized LR+ algorithm uses the same algorithm but adds an additional *and*-connected cube of conditions to *expression*. This cube consists of one condition for each feature, where t_{f2} is the maximum value of all threshold values for this feature in *expression* and t_{f1} is always *0*. For the example in Figure 1 the cube of conditions (c_{21},c_{12}) is added. Thus the media objects P_1, P_2, P_3, P_6 and P_9 can be occluded very fast. The next section describes the test environment and test data for these algorithms.

IV. Test environment

The querying algorithms were implemented in Perl and evaluated on a DOS computer. Perl was chosen, because it allows rapid prototyping and effective statistical analysis. DOS was chosen, because querying performance was tested by the average query execution time and therefore using a single user, single task operating system was the proper choice.

We did about 50000 tests on one to ten features (equally distributed) and up to 10000 artificial feature vectors with length between one and 32 elements (equally distributed). The artificial feature vectors were normalized to *[0,1]* and consisted of equally distributed (45%), normally distributed (50%) and negative exponentially distributed (5%) columns of random numbers. The reference values for LWM+ were calculated prior to the tests. Two distance functions were used: city block distance and Euclidean distance. Each artificial feature was bound to one distance function (equally distributed).

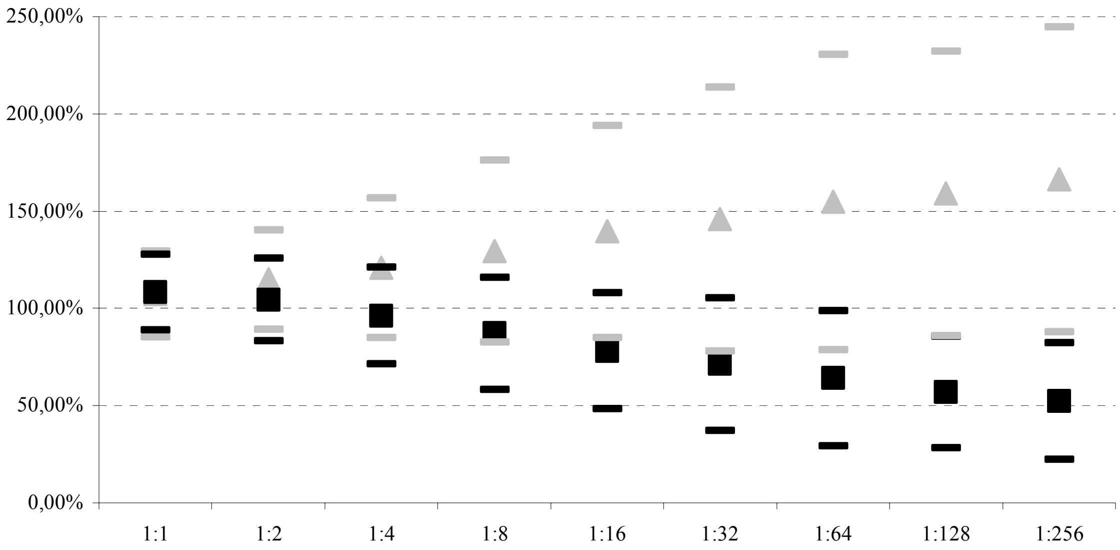


Figure 3. Results for varying size of result set. The grey triangles show the average performance of LR and the black squares depict the LR+ performance.

Each query was done with one random selected query example and random selected weights. The result set size was fixed to 32 elements for all tests. The next section describes the results for our experiments.

V. Experiments and results

First we tested the four algorithms performance for a varying number of features. We did 40000 queries on one to ten features. Figure 2 shows the results. The triangles show the average performance of LR, the diamonds stand for the LR+ performance and the gray squares and the diamonds within them depict the performance of LWM respective LWM+. All performance values are relative to LWM. The lines above and below the icons show the standard deviations relative to the average values.

This test revealed that using the TRIQ has hardly any effect on query execution time (smaller than 1%). Additionally, it showed that the performance of LWM is always better than LR and that LR has a very small standard deviation. Using the simple optimization in LR+ reduces the query execution time to about 50% of LWM. Because this is a heuristic approach, the standard deviation of LR+ is bigger than of LWM. The better performance of LR+ seems to be independent from the number of queried features.

In the second experiment, we tested the algorithms behavior for a varying relation of result set size and queried collection size. This is interesting because at least the performance of LR+ could be dependent on this relation. We did 7200 queries with a varying number of features and relations from 1:1 to 1:256. Figure 3 shows the results. This time, LWM and LWM+ were omitted. Still, the performance values are relative to LWM (100%). This test showed that only for relations of result set size to queried collection size of bigger than 1:4, the performance of LR+ is worse than LWM (above 100%). For relations lower than 1:4 LR+ outperforms LWM and reaches an average query execution time of about 50% at a relation of 1:256. In this test, the standard deviation for LR was significantly worse than in the first test. This is because queries on varying numbers of features were mixed. The overall performance of the tested algorithms (compared to LWM, 100%) is: LWM+: 99.9%, LR: 172% and LR+: 55%.

VI. Conclusion

In this paper we compared the query execution performance of two methods for macro-level similarity measurement: linear weighted merging (LWM) and logical retrieval (LR). We implemented each algorithm in a standard and an optimized version. Additionally, we implemented a conversion algorithm that generates LR expressions from LWM result sets. About 50000 tests were performed.

The major result of this study is, that optimized LR clearly outperforms LWM in terms of query execution time. In our earlier work we showed that this is true for the quality of retrieval results as well. Thus, there is – from our point of view – no reason to use LWM in CBIR systems any longer.

VII. References

[1] J. Barros, J. French, and W. Martin, "Using the triangle inequality to reduce the number of comparisons required for similarity based retrieval," Proc. SPIE Conf. on Storage and Retrieval for Image and Video Databases, San Jose CA, USA, pp. 392-403, 1996.

[2] C. Breiteneder, and H. Eidenberger, “Performance-optimized feature ordering for content-based image retrieval,” Proc. European Signal Processing Conference, Tampere, Finland, 2000.

[3] H. Eidenberger, and C. Breiteneder, "Macro-level similarity measurement in VizIR," Proc. IEEE Multimedia Conf. & Expo, Lausanne, Switzerland, 2002.

[4] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker, “Query by image and video content: the QBIC system,” *IEEE Computer*, vol. 28, no. 9, pp. 23-32, 1995.

[5] M. Ortega, R. Yong, K. Chakrabarti, K. Porkaew, S. Mehrotra, and T.S. Huang, “Supporting ranked boolean similarity queries in MARS,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 10, no. 6, pp. 905-925, November 1998.

[6] A.W.M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, “Content-based image retrieval at the end of the early years,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1349-1380, December 2000.

VizIR—a framework for visual information retrieval

Horst Eidenberger*, Christian Breiteneder

Interactive Media Systems Group, Institute of Software Technology and Interactive Systems, Vienna University of Technology, Favoritenstrasse 9-11-188/2, A-1040 Vienna, Austria

Received 31 May 2002; received in revised form 4 March 2003; accepted 28 April 2003

Abstract

In this paper the visual information retrieval project VizIR is presented. The goal of the project is the implementation of an open visual information retrieval (VIR) prototype as basis for further research on major problems of VIR. The motivation behind VizIR is the implementation of an open platform for supporting and facilitating research, teaching, the exchange of research results and research cooperation in the field in general. The availability of this platform could make cooperation and such research (especially for smaller institutions) easier. The intention of this paper is to inform interested researchers about the VizIR project and its design and to invite people to participate in the design and implementation process. We describe the goals of the VizIR project, the intended design of the querying framework, the user interface design and major implementation issues. The querying framework consists of classes for feature extraction, similarity measurement, media handling and database access. User interface design includes a description of visual components and their class structure, the communication between panels and the communication between visual components and query engines. The latter is based on the multimedia retrieval markup language (MRML, Website. http://www.mrml.net (last visited: 2003–03–20)). To be compatible with our querying paradigm, we extend MRML with additional elements. Implementation issues include a sketch on advantages and drawbacks of existing cross-platform media processing frameworks: Java Media Framework, OpenML and DirectX/DirectShow and details on the Java components used for user interface implementation, 3D graphics with Java and Java XML parsing.

Keywords: Visual information retrieval; Content-based image retrieval; Content-based video retrieval; Media processing

*Corresponding author. Tel.: +43-158801-18853; fax: +43-158801-18898.

E-mail addresses: eidenberger@ims.tuwien.ac.at (H. Eidenberger), breiteneder@ims.tuwien.ac.at (C. Breiteneder).

1. Introduction

The global integration of information systems with the ability of easy creation and digitization of visual content have risen the problem of how these vast amounts of data in collections or databases are managed. One of the crucial success factors of all approaches to solve this problem is apparently the implementation of effective but still easy to handle retrieval methods. Visual information retrieval (VIR) is still a rather new approach to overcome these problems by deriving features (or: descriptors; like color histograms, etc.) from the visual content and comparing visual objects by measuring the distance of features with distance functions. VIR is usually divided in two directions: Content-based image retrieval (CBIR) and content-based video retrieval (CBVR). The major advantages are fully automated indexing and the description of visual content by visual features. On the other hand, the fundamental drawbacks of VIR are the *semantic gap* between high level concepts presented to a user and the low level features that are actually used for querying [1] and the *subjectivity of human perception*. The latter means that different persons or the same person in different situations may judge visual content differently. This occurs in various situations: different persons may judge features (color, texture, etc.) differently, or if they judge them in the same way they still may perceive them differently [2].

Partly due to these principle drawbacks four major problems of VIR approaches can be identified:

- Low result quality.
- Complicated user interfaces.
- Unsatisfactory querying performance.
- Lack of assessment methods.

Retrieval results of low quality are—next to semantic gap and subjectivity of human perception—often the consequence of working only with general features for all types of visual content and asking the user to choose the features, he would like to use. Complicated user interfaces overtax the casual user if they demand for a precise opinion on similarity, the selection of features, and especially, the provision of feature weights. Many users would not even try a classic VIR interface, if they had the opportunity to use it. Simpler, but still effective user interfaces are needed to improve the acceptance of VIR systems.

Unsatisfactory querying performance—especially for large media collections—is a result of using distance functions in VIR systems to calculate the dissimilarity between visual objects. This process is often very slow and unbearable reply times may occur for large databases. Query acceleration methods include (1) indexing techniques (e.g. R*-trees), (2) complexity reduction techniques (e.g. coarse feature vector representation or suitable transformations) and (3) media object occlusion techniques (e.g. using the triangle inequality in [3]).

Finally, despite reasonable efforts in the last 3 years, very few standardized methods exist for assessing new querying paradigms. One exception is the Brodatz sample collection, which represents some kind of de-facto standard for the evaluation of texture querying. Promising approaches to overcome this situation are the Benchathlon project [4] that tries to collect and compare the performance of CBIR benchmarks and the annual TREC video retrieval competition [5] that defines evaluation procedures for CBVR.

In this paper we present the visual information retrieval project VizIR. The goal of the project is an open VIR framework as a basis for further research in order to overcome the problems pointed out above. The basic structure of VizIR was first laid down in [6,7]. VizIR was initiated in summer 2001 as a consequence of our experiences gained in earlier VIR projects and is supported by the Austrian research fund since December 2002. The motivation behind VizIR is: an open VIR platform would make research (especially for smaller institutions) easier and more efficient (because of standardized evaluation sets and measures, etc.). The intention of the paper is to let interested researchers know about the VizIR project and its design and to invite them to participate in the design and implementation process.

The goal of VizIR is not the development of a monolithic system but of a system-independent class framework of querying and user interface components (interaction panels, event model, etc.) based on the Java programming language. An important issue of VIR is the communication of user interfaces and query engines. This communication should be standardized in order to combine arbitrary user interfaces and querying systems and be based on modern communication paradigms (XML, etc.).

The paper is organized as follows: the following section points out relevant related work, Section 3 is dedicated to the VizIR project goals and Section 4 to the querying and user interface framework design. Section 5 discusses major implementation issues and finally, Section 6 gives an overview over past, current and next activities in the VizIR implementation process. The paper is supplemented by an appendix with an extension of the MRML [8].

2. Related work

In this section we discuss the architectural properties and shortcomings of earlier CBIR and CBVR prototypes and the user interface approaches that were used.

2.1. Existing VIR prototypes

Past research efforts have lead to several general-purpose prototypes like QBIC [9], Virage [10], VisualSEEk [11], Photobook [12], MARS [13], El Niño [1,14] and GIFT [15] for CBIR as well as OVID [16] and VIQS for CBVR and some application-specific prototypes like image retrieval systems for trademarks [17] or CueVideo for news videos analysis (e.g. [18]). Most of these prototypes share a number of serious drawbacks. The first is that all of them implement only a small number of features and do not offer the developer an API for extensions. An exception is IBM's QBIC system for image querying, which has (in version 3) a well-documented API for feature programming.

Another problem is that none of these prototypes has an architecture that supports the MPEG-7 standard (see [19]). To our knowledge, at present no MPEG-7-compliant prototype for VIR exists or is under development. Part 6 of MPEG-7 contains a reference implementation of its visual descriptors and a simple querying application, which was developed for testing and simulation [19]. Unfortunately, this reference implementation does not contain a framework, a documentation of the VIR part, a modern user interface (though a simple web-interface for experts is available by now), a suitable database, optimized descriptor extraction functions and performance-optimized algorithms. That is why it cannot be used as a VIR prototype, although it is still a good starting point for developing one.

One prototype that should be mentioned here is the GNU image finding tool (GIFT). GIFT is an extendible CBIR system (developed at the University of Geneva) available under GNU public license [15]. Unfortunately, GIFT supports only image querying and because it is based on C++ and the Unix operating system it can not be extended to video retrieval easily. Currently, no standardized video processing environment with a C/C++-API is available for Unix operating systems (see Section 5.1). Still, GIFT introduced several valuable concepts to CBIR (including MRML, see Section 2.2).

Apart from the mentioned focal points of research and the implemented prototypes the following *key issues* of VIR systems have not yet been investigated to a sufficient extent:

- Similarity measurement in multi-feature environments.
- Media sets for assessment.
- Integration of computer vision methods.

With similarity measurement we mean the transformation of a distance space (the result of distance measurement for multiple features and distance functions) to a result set. The common way of similarity measurement in VIR systems is measuring distances with an L^1- or L^2-metric (e.g. city block distance and Euclidean distance), merging a single object's distance values for multiple features by the weighted sum and presenting the user the objects with the lowest distance sum as the most similar ones. We have shown in our earlier work that this approach is not the most effective one [20]. More sophisticated methods for similarity definition would result in higher quality results (e.g. [21]).

Additionally, as pointed out above, not enough effort has been undertaken so far to put together standardized rated image and video sets for the various groups of features. This has lead to vague, often worthless statements on the quality of VIR prototypes.

Finally, surprisingly few ideas and methods have been taken over from computer vision and other areas up to now. Neural networks have been used for feature clustering (e.g. self-organizing maps [43]), face detection and thresholding methods for segmentation but hardly any shaping techniques for 3D object reconstruction or sophisticated neural networks for scene analysis have been yet applied.

2.2. VIR user interfaces

This section overviews user interfaces of well-known VIR systems: first CBIR systems and then CBVR systems. The focus in CBIR will be on classic systems (including QBIC and Virage) and two promising more recent approaches (El Niño and ImageGrouper). The section ends with a short description of an approach to standardize the communication of VIR user interfaces and query engines.

In the past, the design of user interfaces of VIR systems was quite simple—in comparison to most other visual systems. Most systems (QBIC, Virage [10], Photobook, VisualSEEk) use a single 2D panel of images for query definition and result set display. Querying is done by selecting one or more query examples, one (e.g. QBIC), a few (e.g. MARS) or all features (Virage) and—in the latter two cases—weights for the importance of these features. Iterative Refinement by Relevance Feedback [44,45] can usually be performed by defining the importance of result set elements textually and iterating the query. This paradigm has several drawbacks: earlier result sets are thrown away, selecting features and weights overtaxes the casual user and after all, the static structure of such an interface is not very user-friendly and from today's point of view may be judged old-fashioned.

Therefore several research groups have been working on new user-centric interface approaches in the last years. Two of the most interesting are El Niño and ImageGrouper [22]. To our knowledge, El Niño is the first approach to define a query implicitly by the distance relations of objects in a 3D panel. This query definition process can be done intuitively and easily by drag-and-drop. The most interesting innovations in ImageGrouper are the usage of two panels for the active and the last query and a history over all refinement steps in a querying session. The central idea of ImageGrouper is the definition of queries by three groups: positive examples, negative examples and neutral examples. ImageGrouper's major drawback is that it has no standard interface to query engines and is bound to an engine with classic distance measurement and linear weighted merging.

Like El Niño, VizIR will contain 3D user interfaces for query formulation. Using 3D information visualization techniques instead of 2D methods has several advantages. Generally, each 3D view is just a 2D projection [23]. 3D views take advantage of human spatial memory and allow displaying more information without incurring additional cognitive load because of pre-attentive processing of perspective views. In general, they lead to better retrieval results in user studies in terms of reaction time, number of incorrect retrievals and failed trials [24]. Additionally, they allow the rendering of more information items because of scaling possibilities and a better global view. Finally, there is experimental evidence that 3D displays enhance subjects' spatial performances [23]. The major open problem of 3D systems in this context is the development of suitable 3D user interaction techniques [24,25].

Classic CBVR systems are OVID [16] and VQIS. One of the most interesting aspects concerning the user interfaces of CBVR systems is the handling of temporal media (video and animations) in a static user interface. In general, there are three principle solutions to present video information: (1) integration of the full video with player controls into the environment (CPU power and network bandwidth consuming), (2) creation and usage of animated icons (CPU power consuming) and (3) creation of still images that represent the video content. The third solution is the most widely applied one (in VIR). The simplest form of the third type is an image matrix of all keyframes in a video clip. Another approach is the Micon, a 3D cube showing the first frame of a video clip as well as the first line and the last column of all consecutive frames (see element A and B in Fig. 5 for examples). Another type is the Hierarchical Video Browser, a tree-structured view of a video clip. In [26] a general overview of different presentation styles for video is given.

The interoperability of VIR user interfaces and querying systems is an issue that is gaining more and more attention. Interoperability should be achieved by standardized interfaces. The most promising effort in this direction is the MRML (developed at the University of Geneva [8]). MRML is an XML-based standard. It is implemented in GIFT, the user interface Charmer and the basis of the Benchathlon project (see [8] for details). We try to incorporate MRML into the user interface components of VizIR.

3. VizIR project goals

This section gives an overview of the objectives of the VizIR project. VizIR aims at the following major goals:

- Implementation of an open VIR class framework.
- Integration of MPEG-7 visual.
- Implementation of a framework of user interface components for VIR.
- Support for distributed querying.

The overall goal is the implementation of a modern, *open class framework* for content-based retrieval of visual information as basis for further research on successful methods for automated information extraction from images and video streams, the *definition of similarity measures* that can be applied to approximate human similarity judgment and new, better *concepts for the user interface* aspect of visual information retrieval, particularly for human–machine interaction for query definition and refinement and video handling. On top of this framework *working prototypes* are implemented that are fully based on the visual part of the *MPEG-7 standard* for multimedia content description. Reaching this goal requires the careful design of the database structure and an extendible class framework as well as research on suitable extensions and *supplementations* of the MPEG-7 standard by additional descriptors and descriptor schemes. Mathematical and logical fitting distance measures have to be selected for all descriptors (distance measures are not defined in the standard) and an appropriate and flexible *model for similarity definition* has to be defined. MPEG-7 is not information retrieval specific. One goal of this project is to apply the definitions of the standard to visual information retrieval problems.

Another goal is the development of a *general-purpose user interface framework* for visual information retrieval. This framework has to include a great variety of different properties: methods for query definition from examples or sketches, similarity definition by positioning of visual examples in a 3D space, appropriate result display and refinement techniques and cognitively easy handling of visual content, especially video. User interfaces and querying methods both have to support methods for *distributed querying, storage and replication* of visual information and features as well as methods for query acceleration. The importance of this issue becomes apparent from the large amount of data that has to be handled and the computation power that is necessary for querying by—often quite complex—distance functions. Methods for distributed querying, storage and replication include the replication of feature information, client-server architectures and remote method invocation in the querying and indexing modules as well as compression of video representations for the transport over low bandwidth networks. Methods for query acceleration include indexing schemes, mathematical methods for complexity reduction of distance functions and the generation of querying heuristics [27].

An additional, however, implicit goal of the VizIR project is the development of a *multimedia-specific UML-based software development process*. Multimedia applications have special needs that have to be considered during the system design and implementation. This includes modeling of real-time media processing (multiplexing, conversion with codecs, rendering, etc.), more sophisticated modeling of users and use-cases (e.g. abstraction of users to user profiles, etc.), metadata modeling and modeling of multimedia restrictions (Quality of Service parameters, interaction, etc.). Developing tailor-made software development methods on the basis of the UML design process is just a natural consequence.

4. VizIR framework design

This section describes technical details of the VizIR objectives and the intended system architecture. The VizIR framework can be split into four areas of work: (1) querying framework, (2) user interface framework, (3) configuration and communication interfaces and (4) assessment methods. The querying framework contains all methods for feature extraction, similarity measurement, query refinement, media handling and database access. The user interface framework contains a class hierarchy of user interface elements (panels), events and event handling methods and media visualization classes. Configuration and communication concerns all classes and methods for standardized communication of framework elements with other elements (e.g. query engines and user interfaces) or the environment. Assessment methods include benchmarking techniques and media sets for evaluation. The next four subsections detail the relevant design issues for these areas of work.

4.1. Querying framework

The most important issue related to the design and implementation of the querying framework is the implementation of a technically sound class framework for the system components. Even though this is not a research but a software engineering problem, we have to stress that using a professional database and programming environment are crucial success factors for a modern VIR research prototype. As pointed out above, most past approaches have serious shortages in their system architecture.

VizIR uses a relational database for media and feature data storage. Fig. 1 gives an overview of its data model and indicates the relationships between media and feature storage. Visual media is stored in table *Media* and associated with a single *MediaType*. Each media may belong to n collections and each collection may contain m elements. Descriptors are described in table *FeatureClass* with the MPEG-7 descriptor definition language (DDL; based on XML schema). Feature data for a certain descriptor is stored in binary and/or XML format in table *FeatureData*. To allow the implementation of MPEG-7 descriptor schemes, descriptors are organized in collections in table *FeatureCollection*. A collection may consist of descriptors and other collections. Optionally, it may have a DDL-description. Based on this data

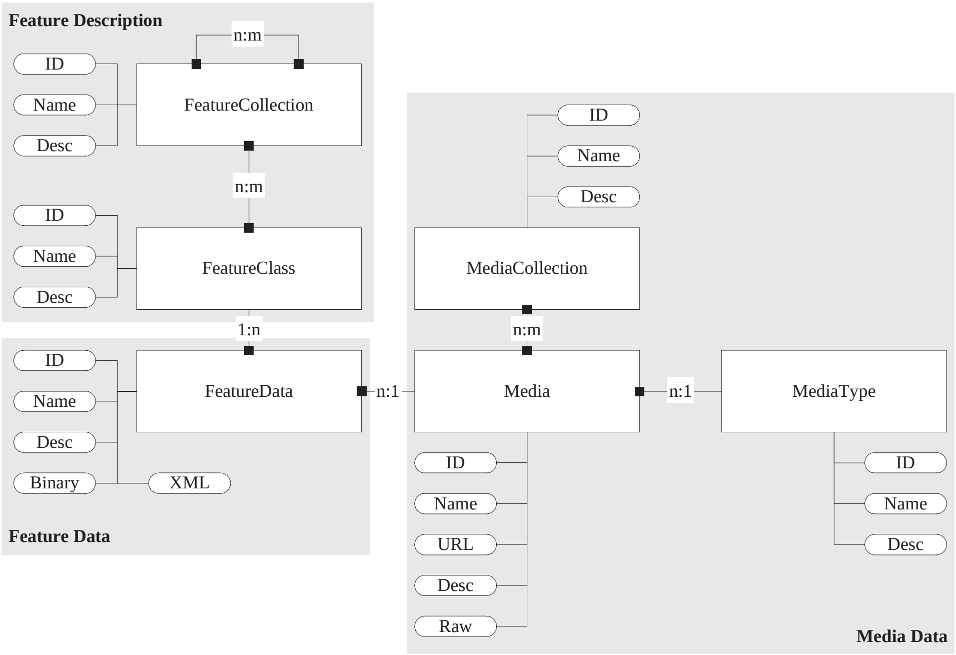


Fig. 1. EER database diagram. The framework contains a database manager that creates this structure during VizIR installation automatically.

model it is possible to use descriptor schemes in queries. If a certain feature collection is selected for a query, all referenced descriptors are selected and used in the query.

Fig. 2 outlines the class structure of the querying framework. To a certain extent this class framework follows the architecture of IBM's QBIC system [9], but largely differs from QBIC in its server/client independent classes. Similarly to QBIC, the database access is hidden from the feature programmer and the structure of all feature classes is predefined by an interface. Key element is class *QueryEngine*, which contains the methods for query generation and execution. Each query consists of a number of *QueryLayer* elements each of which implement exactly one feature. The result of each query is a set of media objects that is stored in a *Vector* object. Media objects are represented by objects of class *MediaContent*. *MediaContent* has an interface that hides the complexity of the actual media access from the framework programmer. For example, he can access the media data—independent whether it is image or video—with a method *getViewAtTime*(*Time*, *ColorModel*). For images, *Time* is irrelevant and for videos it is the position in the media stream. The *ColorModel* of the resulting image can be RGB, HMMD, etc. With the *MediaContent*-mechanism CBIR and CBVIR can be implemented in the same framework without having to introduce media-specific peculiarities in the architecture. Similarly, the methods for database access are encapsulated in the *DatabaseManager*.

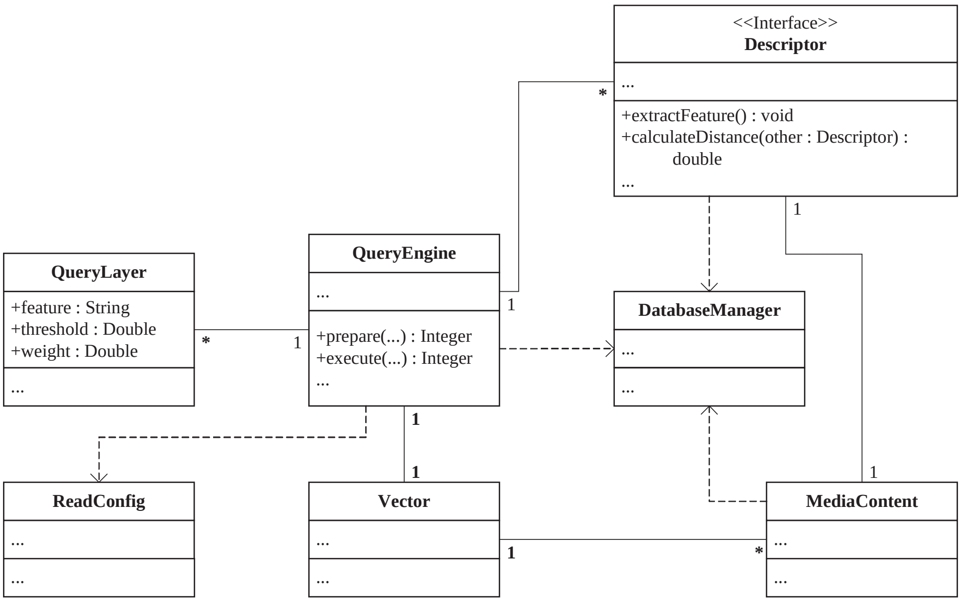


Fig. 2. UML class diagram for an ideal implementation of the VizIR class framework. Custom query engines can be added by sub-classing *QueryEngine*. The *DatabaseManager* offers a standard interface for accessing arbitrary relational databases. Similar to that, *MediaContent* offers methods for media access that hide the actually used media processing library.

All feature classes—MPEG-7 descriptors as well as all others—are derived from the interface *Descriptor* For the MPEG-7 descriptors it is intended to follow the reference implementation of part 6 of the standard. For the reasons given above and especially, because the algorithms of the reference implementation are not performance-optimized the redesign and implementation of the MPEG-7 descriptors is a time- and human resources-consuming task.

Descriptor contains methods for descriptor extraction (*extractFeature*()) and distance measurement (*calculateDistance*()). Unfortunately (for us), MPEG-7 is not a visual information retrieval-specific standard and in general does not include distance functions for the various descriptors. Neither does it give any recommendations for their selection. Therefore it is necessary to implement common distance metrics (like L^1-, L^2-metric, Mahalanobis distance, etc.; [2]), to associate them with descriptors and to find custom distance functions where these metrics are not applicable (e.g. object features, etc.).

The *extractFeature*()-method of *Descriptor* applies the actual feature extraction algorithm to the media considered (and accessible) as *MediaContent*. The MPEG-7 standard—although it is a major advance in multimedia content description—standardizes a number but not all useful features. It is necessary to implement additional descriptors and distance functions for texture description of images (wavelets, etc.; e.g. [28]), symmetry detection of objects (useful for face detection, detection of human-made objects, etc.), object description in video streams (structure recognition from motion, etc.), object representation (scene graphs, etc.) and video analysis (shot detection, etc.). Additionally, we plan to use fractal methods (iterated function systems; IFS) to describe the shape of objects effectively. So far IFS have been used for the compression of self-similar objects (e.g. [29]) but hardly for content-based retrieval (see [30]). We think, that IFS could be very effective for shape description, too.

The sequence diagram in Fig. 3 depicts the querying process. Methods for query definition and query refinement have to be flexible enough to satisfy different ways of how humans perceive and judge similarity and should still be applicable in a distributed querying environment. In VizIR each type of application (server, Servlet, client, applet, etc.) can initiate a query by instancing a *QueryEngine* object and calling the *prepare*() method. The *execute*() method of a query creates a feature class for each *QueryLayer* of a query and extracts a descriptor by calling *extractFeature*(). These objects of class *Descriptor* are then used for feature comparison with *Descriptor* objects of the images in the database by the method *calculateDistance*(). The images of the result set are returned via the *getElements*() method. To accelerate queries, indexing schemes and other query acceleration models will be implemented as part of VizIR. Next to classic index structures for visual content (e.g. R-tree, segment index tree, etc.) and query acceleration techniques (storage of the factorized terms of the Mahalanobis distance [31], etc.), experiments will be undertaken with new heuristic approaches like those we previously published [27].

Concluding this sketch of the VizIR querying framework architecture we outline several aspects of the application and data distribution. In a scalable framework it is simply necessary to implement tools for distributed and replicated visual content

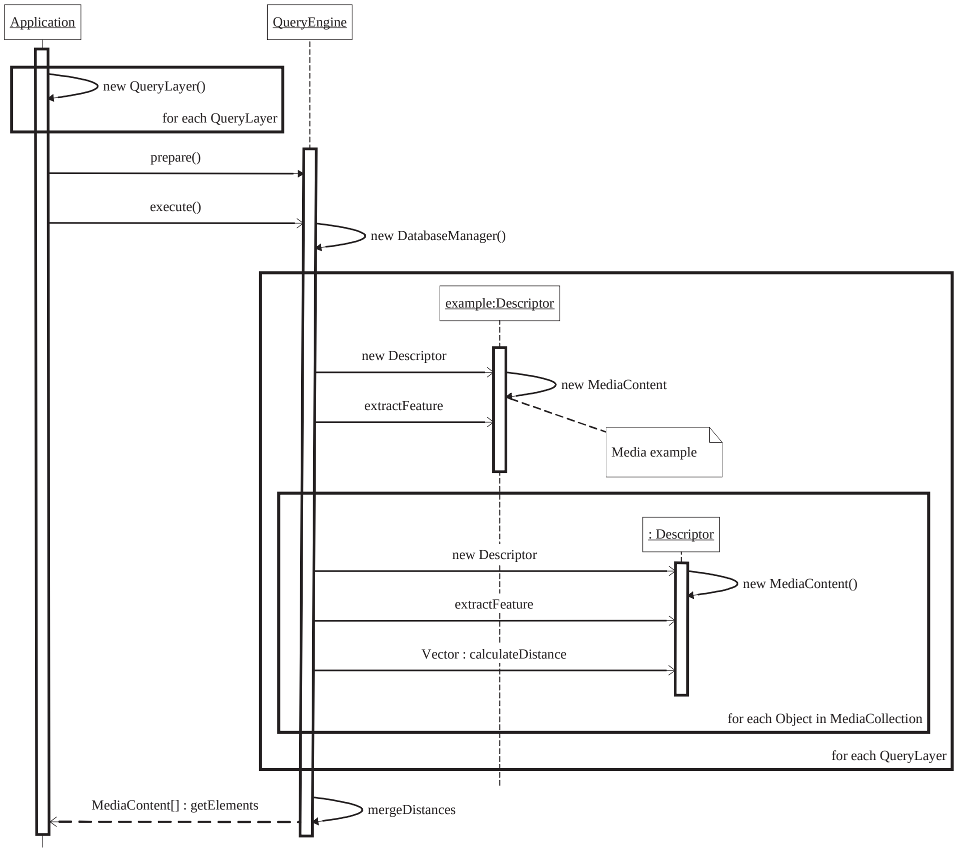


Fig. 3. Schematic UML sequence diagram of the querying process.

management as well as database management. Modern Web Service- or CORBA-based programming environments like the Java environment permit the network-independent distribution of applications, objects and methods (in Java through the Remote Method Invocation library) to increase the performance of an application by load balancing and multi-threading. VizIR is based on Java. Therefore the objects for querying can be implemented as JavaBeans, feature extraction functions with RMI, database management through Servlets and user interfaces as Applets. Database distribution is realized through standard replication mechanisms and database access through JDBC.

4.2. User interface framework

The VizIR user interface framework is a collection of components that can be combined arbitrarily. The major issue is the design of querying & query refinement interfaces that integrate image and video content, the implementation of methods for video content representation in static user interfaces and the support of multiple media-based querying paradigms. All user interface components have to be designed as intuitive and self-explanatory as possible to guarantee high usability and, as a consequence, increasing acceptance of VIR. In addition to user interface building blocks, methods have to be developed that allow their combination in application-specific user interfaces (fields of application in the future will be digital libraries, medical image search, TV broadcast archives, etc.).

Fig. 4 shows the static structure of the VizIR user interface framework that should satisfy these demands. Central element is the interface *UserInterfaceComponent* that is inherited by all classes having a visual panel. These are *MediaPanel* (the mother class of all panels that deal with media objects), *QueryEngine* (the mother class of all querying engines, the panel contains all elements necessary for query formulation), *Descriptor* (mother class of all implemented features, the panel contains a toolbox for sketch drawing), *MetadataPanel* and *LayerPanel* (a layer manager for multi-layer

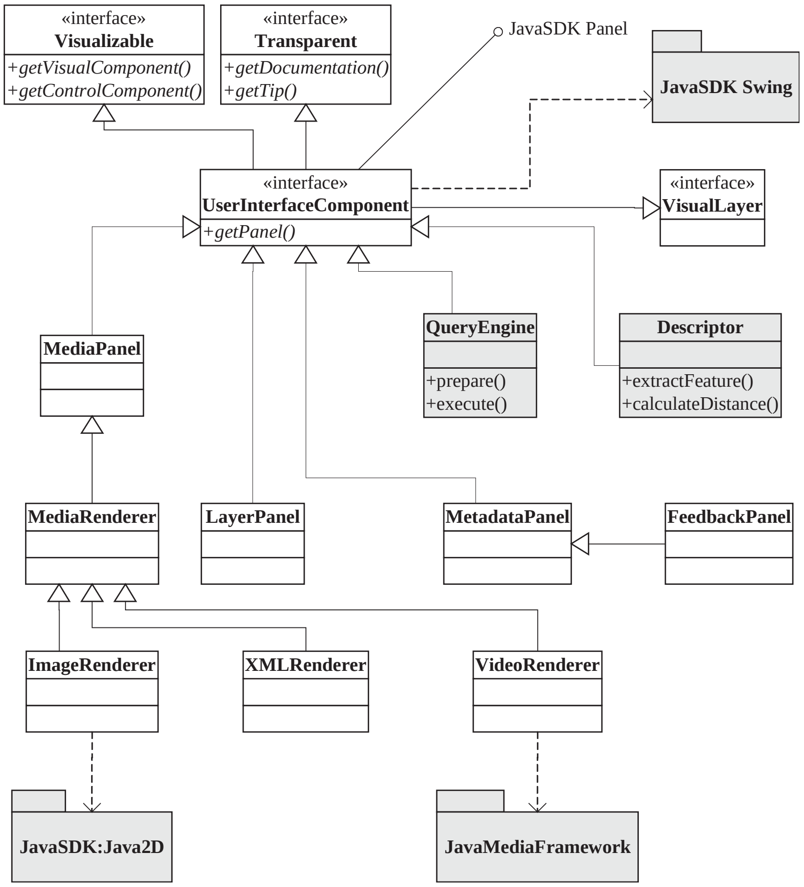
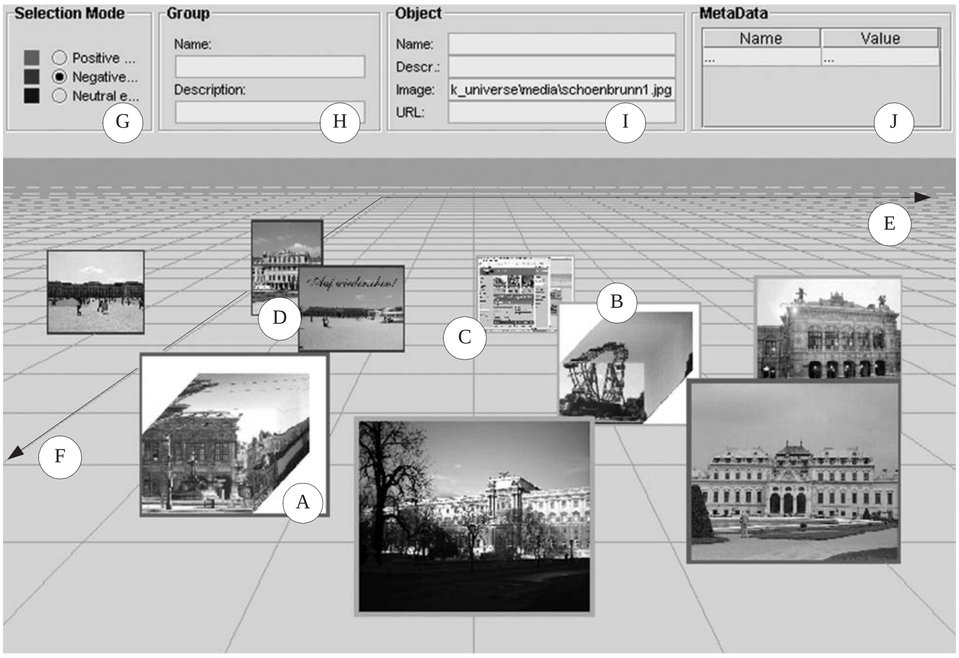


Fig. 4. Class diagram of the VizIR user interface framework.

image sketching as in Photoshop). VizIR user interface components are based on Java Swing. *UserInterfaceComponent* inherits methods from the interfaces *Visualizable* (methods for receiving a visual panel and a visual control component like in the Java Media Framework [32]), *Transparent* (methods for receiving visual documentation and help in the user interface) and *VisualLayer* (defines the structure of a layer of the sketching panel, basically a Java *Image* type).

MediaRenderer is a special type of *MediaPanel* for the visual rendering of media objects. *MediaRenderer* takes an arbitrary media object as input and generates a (2D or 3D) diagrammatic representation. Representing media objects in a static user interface is easy for images but difficult for (time-based) video content. Common approaches are index frames and Micons, which obviously are unsatisfactory. A more sophisticated approach would be an object viewer for all objects and their temporal trajectories in a video shot. Also, video cubism (allowing for interactively cutting an $X-Y$-time cube of video data along arbitrary oriented planes; [33]) should be considered as an alternative for presenting video results. So far, we have implemented three renderers for images (JPG, PNG, GIF, etc., based on Java2D), videos (generates Micons—see Section 2.2—for arbitrary video formats: MPG, AVI, MOV, etc., based on the Java Media Framework) and XML. *XMLRenderer* can render any XML-file that can be displayed in a web browser (see [7] for technical details). Fig. 5 shows examples: element A and B are Micons (representing videos of



Selection Mode

○ Positive ...
● Negative...
○ Neutral e...

Group

Name:

Description:

Object

Name:

Descr.:

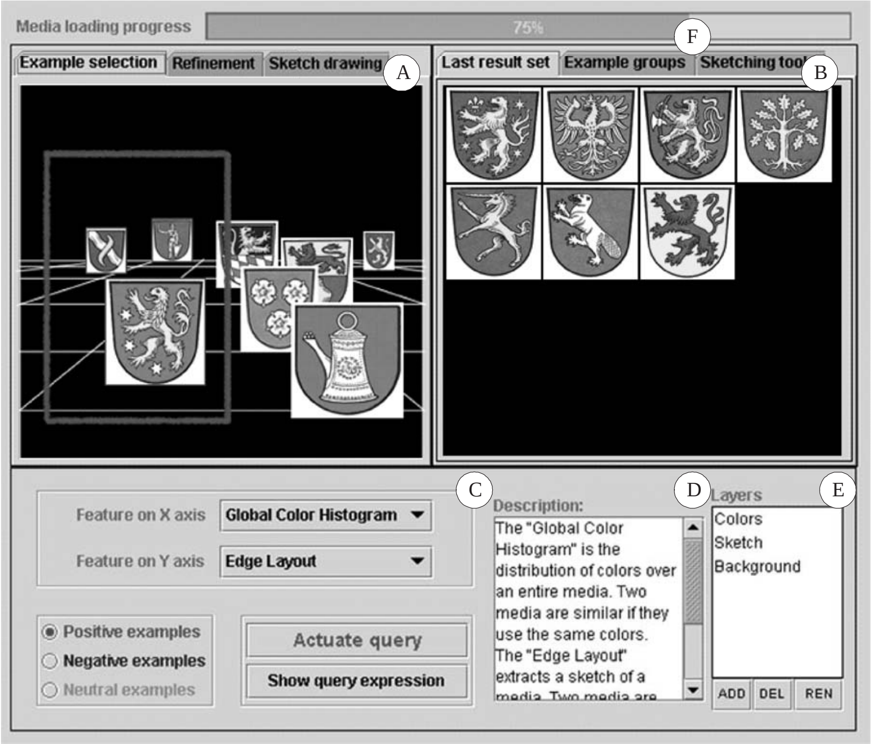
Image: k_universe\media\schoenbrunn1.jpg

URL:

MetaData

Name	Value
...	...

Fig. 5. Screenshot of the 2.5D panel (media objects are positioned at random).



Feature on X axis: Global Color Histogram

Feature on Y axis: Edge Layout

● Positive examples

○ Negative examples

○ Neutral examples

Description:
The "Global Color Histogram" is the distribution of colors over an entire media. Two media are similar if they use the same colors. The "Edge Layout" extracts a sketch of a media. Two media are

Fig. 6. Screenshot of a VizIR user interface prototype.

the Vienna opera house and the Prater Ferries wheel), element C is a webpage and all other media objects are standardized images. Like *MediaContent* for media access, the *MediaRenderer*-mechanism allows performing CBIR and CBVR in the same user interfaces and the implementation of unified APIs for both types of media.

The most important *MediaPanel* is the 2.5D media panel. For examples see Fig. 5 and element A of Fig. 6. The 2.5D panel is used for example selection, browsing, query formulation and the display of result sets. The rendered substitutes of media objects are displayed as images parallel to the image plane. It is possible to navigate in two dimensions (left–right, forward–back) and to zoom. Groups of objects can be selected, moved and associated with metadata (by communication with a *MetadataPanel*). The angle of the image plane and the $X-Y$-plane can be varied between 0° and 90°. The panel may have visual control components (elements G–J in Fig. 5 and element C in Fig. 6). Panel G in Fig. 5 (also shown in the lower left part of element C in Fig. 6) allows to set the selection mode for the cursor and panel H is for group definition. Panel I shows information on the currently selected object and panel J its metadata entries. The upper panel of element C of Fig. 6 is initialized with all dimensions of the media space to be displayed (in the VIR context: all implemented features). The view changes whenever new dimensions are chosen for the X- or Y-axis or the querying button in the lower right part of element C in Fig. 6 is pressed.

It is important to know—in rough terms—the querying process implemented in VizIR to understand the role of the 2.5D panel. Fig. 7 shows a State-Transition-Diagram of the underlying querying process. First the user interface components are initialized with media objects and query parameters (element F of Fig. 6 shows a progress bar panel for media loading). Then the user can define a first query by selecting example media objects. This sets the user interface in the defined state. Executing the query brings the user interface in the active state where refinement can be started or a new query can be defined. In active state the query is re-executed whenever the user presses the 'activate' button or the query engine control component detects substantial changes in the query definition.

Both panels for query definition and query refinement are 2.5D panels that have been initialized with MRML-documents. They can visualize any two-dimensional subspace of the distance space (for the selected features and examples) generated in the previous querying iteration. This is done by showing the media objects (or their representations) parallel to the image plane and, on the X- and Y-axis, arranged according to their relative distance (depicted in element E and F in Fig. 5). Similar objects are placed near to each other, un-similar objects far from each other. Element A of Fig. 6 shows the distance of images for a color histogram feature on the X-axis and the distance for an edge histogram on the Y-axis. The features (distance space dimensions) shown on the X- and Y-axis can be changed *interactively*. Queries are defined and refined in the same way by selecting media objects or groups and

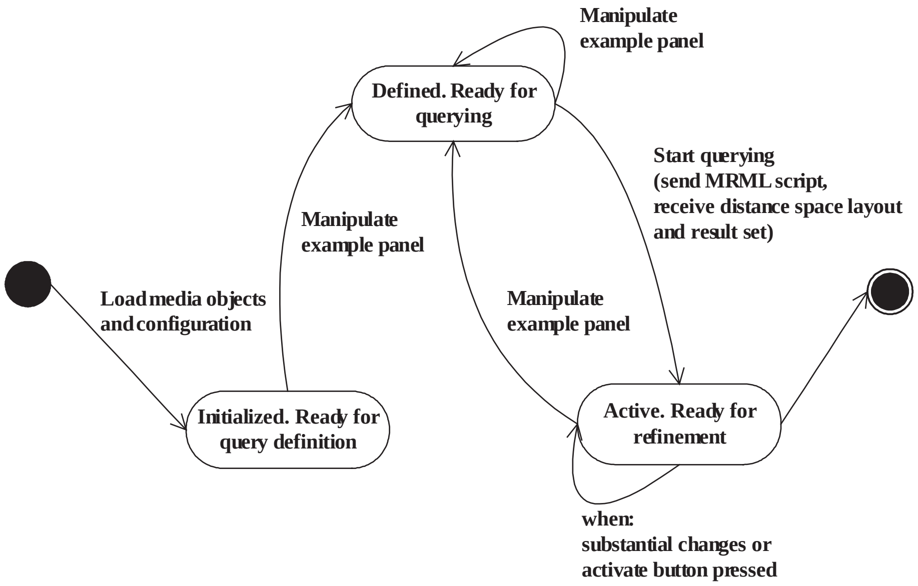


Fig. 7. State-transition-diagram of the querying process.

marking them as positive or negative examples. Thus, it is possible to define n-dimensional hyper-cubes (clusters) of (un-)similar media objects. The query engine tries to find all media objects that belong to the clusters with positive examples minus those with negative examples. We call this similarity measurement process Logical Retrieval (LR, see [35] for a more detailed description).

We are implementing two querying paradigms: query by example (QBE) and query by sketch (QBS), because they are media-based and intuitive. Even though for beginners text querying may be the easiest form of interaction, we are—at this point in time—not planning to implement a text interface, because implementing such an interface would not raise new VIR research questions nor help to solve the existing ones. QBE follows the querying process described above. Sketches for QBS can be drawn in the 'sketch drawing' panel in Fig. 6. This panel contains layers of type *VisualLayer* that are managed by the *LayerManager* (element E in Fig. 6) and allow drawing with the tools provided by the descriptor objects. These tools are collected in the 'sketching tools' panel (element B in Fig. 6). The 'last result set' panel contains the media objects of the last result set (similarity values are associated as metadata). It is just a special 2.5D example panel with an image-plane to $X-Y$-plane angle of 0°. The same is true for the 'example groups' panel in Fig. 6 that lists all query examples partitioned in three groups: positive, negative and neutral examples. (Neutral examples are explicitly excluded from the query. Their properties are marked as irrelevant for the query.) The 'description' panel (element D in Fig. 6) contains the information of the methods from the *Transparent* interface for the active user interface element.

The VizIR user interface class structure follows the paradigm that all components (methods, panels, etc.) are defined, where they are used. Thus, each query engine has a visual panel for query formulation and each descriptor has a panel with tools for sketching (e.g. line drawing tools for an edge layout descriptor). To guarantee the transparency of VizIR (defined in [6]), each visual component has to implement the *Transparent* interface with documentation and tips. The panels of the framework can be integrated into any visual Java container and organized arbitrarily. The layouts in the screenshots in Figs. 5 and 6 are just examples. Because the VizIR framework is based on Java and the Java SDK is possible to integrate the user interface components into any container (frame, applet, etc.) to perform distributed querying (with Web Services, CORBA, RMI, etc.) and querying in the background (in a separate thread).

The validity of arbitrary combinations is guaranteed by the communication mechanism of the framework. It follows the Delegation-Event-Model and is conceptually shown in Fig. 8. Each object of class *MediaPanel* (MediaPanel-1 and MediaPanel-2) may communicate with any other *MediaPanel* through *MediaPanel-Event* objects (e.g. the selection mode panel in element G in Fig. 5 with a 2.5D panel). Thus, all media panels have to implement listener classes that are defined in *UserInterfaceComponent* and flag the media panel events they fire. For easier user interface building the framework contains convenience classes with listener functions for standard communication operations (e.g. communication of query control panel and 2.5D panel when the example group selection is changed, etc.).

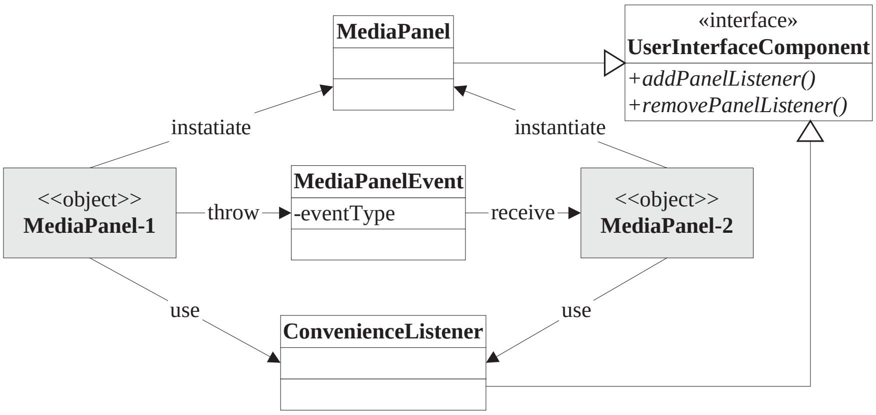


Fig. 8. Event model for panel communication. Media panels communicate through *MediaPanelEvent* objects.

4.3. Configuration and communication interfaces

Query engines in VizIR can be of arbitrary kind. We are implementing a query engine based on the querying process sketched in the previous section (see [35] for more details). In VizIR, the communication of user interfaces and query engines is loosely coupled based on MRML (see Section 2.2).

Each framework component that uses MRML for communication, uses instances of the classes *MRMLReader* and *MRMLWriter* (see Fig. 9). These classes are derived from *ReadConfig* (XML parser class) and *WriteConfig* (XML writer class). Communication classes for new XML languages can be implemented in the same way. In order to perform LR queries with MRML we had to extend its document type definition (see Appendix A for DTD code). We have defined elements for context-free media and media group definition (required for the implemented querying paradigm), descriptor definition and query definition. The following example illustrates how these extensions can be used:

```
<logicalQuery>
  <clusterDefinition>
    <clusterRestriction>
      <clusterDimension lowerBound=’’0.0’’
        upperBound=’’0.5’’>
          <mediaGroup id=’’qe1’’ type=’’positive’’>
            <mediaObject dataLocation=’’file:img1.gif’’
            iconLocation=’’file:thumb1.gif’’/>
          </mediaGroup>
        <descriptor name=’’ColorHistogram’’>
      </clusterDimension>
```

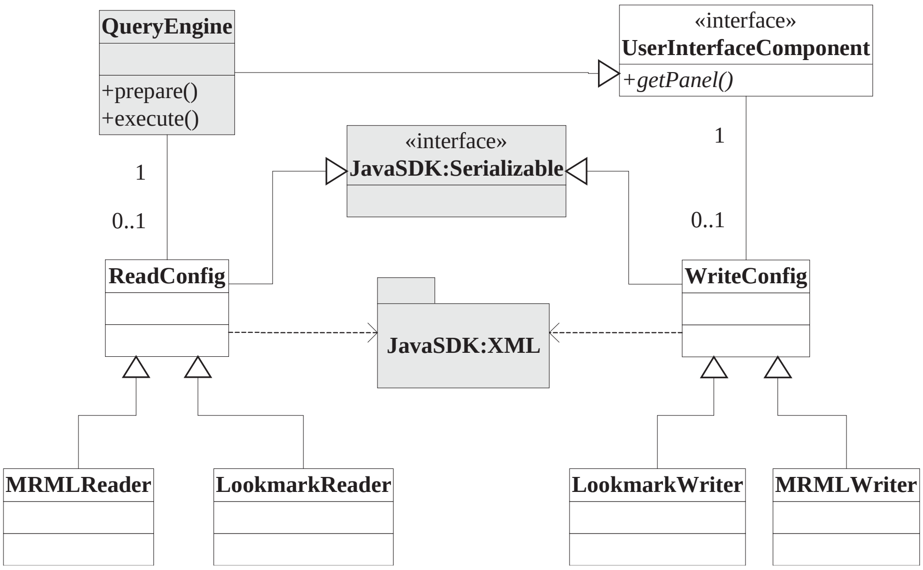


Fig. 9. Class diagram for MRML communication in VizIR. Query engines and user interface components use the classes *ReadConfig* and *WriteConfig* or their subclasses to read and write XML configuration files.

```
    </clusterRestriction>
  </clusterDefinition>
</logicalQuery>
```

This construct defines a query (on the *collection* defined elsewhere in the MRML script) with a single feature. A color histogram is used to find all media objects that have a distance to the positive query example 'img1.gif' (represented by the icon 'thumb1.gif') that is smaller than '0.5'. If we liked to retrieve all objects that fulfil this cluster-condition *and* a second one, we would put the second *clusterRestriction* in the same *clusterDefinition*. If we wanted to retrieve all media objects meeting the first *or* the second condition, we would put the second one in a new *clusterDefinition*. These constructs are flexible and can be used in various ways. They should not only support our LR concept but—according to the published querying paradigm—the one used in MARS as well [13].

4.4. Assessment methods

Concluding this description of the VizIR framework, we would like to point out issues that are related to VIR assessment methods. To our belief, a significant improvement of VIR research in the future will be the development of standardized quality assessment procedures (like in the Benchathlon project [4]). In the VizIR project the following assessment tasks will be undertaken:

- Formulation of standardized evaluation procedures.
- Collection and creation of media sets with ground truth.
- Evaluation of descriptors and querying methods.
- Evaluation of query acceleration methods.

Common evaluation models (recall, precision, etc. [36,37]) are analyzed to develop standardized evaluation procedures. The application of the standard measures in information retrieval, recall and precision, to VIR systems using linear weighted merging (see above) implies giving up at least 10% of recall, since a system with linear weighted merging returns the n 'most similar' available objects (independent of the question whether or not they are really similar), while the recall measures the ratio of really similar objects to all available objects. This has to be considered in the evaluation process. As a consequence, the feasibility of less well-known methods (systematic measures, etc.) will be investigated and methods from other research areas will be checked for applicability. This could be psychological methods, e.g. semantic differential techniques [38].

Evaluation sets with image and video content will be collected and—where not available—created for groups of descriptors and ground truth information will be derived from tests with volunteers (students, etc.). Such sets are obviously decisive for the quality judgment of VIR systems. Actually, however, only a few de-facto standards do exist, including the Brodatz database for texture images. Partially, these evaluation sets will be created by enriching and extending the image and video clip sets, used for the MPEG-7 evaluation. As well, different approaches—e.g. findings on the basis of gestalt laws—will be checked for their suitability to develop those test sets.

The extended evaluation of the MPEG-7 descriptors, descriptor schemes and other implemented descriptors with statistical methods will be performed in two steps: (1) Evaluation of their independent performance and their performance in combinations. From this information the overall performance of the visual part of MPEG-7 and VizIR can be judged. (2) Analysis of dependencies among descriptors with statistical methods (cluster analysis, factor analysis, etc.) to identify a base for the space of descriptors and to be able to normalize the visual part of the MPEG-7 standard and to extend it by new independent descriptors.

Finally, the performance optimization methods developed for VizIR will be compared to those developed for other comparable retrieval systems. In the past, we have implemented several performance optimization techniques and compared them by the reply time for queries (e.g. in [39]). This will be continued in VizIR.

5. Implementation

In this section, two major implementation decisions of VizIR are discussed: the choice of the programming environments for media handling and graphic i/o. When we made these decisions, we had not yet decided if we should base VizIR on C++ or Java.

5.1. Media programming environment

The major question concerning the implementation of the VizIR prototype is the programming environment. At this point in time, there are three major alternatives that support image and video processing to choose from:

- Java and the Java Media Framework (JMF; [32]).
- The Open Media Library standard (OpenML) of the Khronos group [40].
- Microsoft DirectX (namely DirectShow [41]).

All of these environments offer comprehensive video processing capabilities and are based on modern, object-oriented programming paradigms. DirectX is limited to Windows-operating systems and a commercial product. Therefore, in the following discussion we will concentrate on the first two alternatives: JMF and OpenML. JMF is a platform-dependent add-on to the Java SDK, which is currently available for SunOS, Windows, MacOS-X (implementation by SUN and IBM) as well as Linux (implementation by Blackdown) in a full version and in a Java version with less features for all other operating systems that have Java Virtual Machine implementations. JMF is free and extensible. OpenML is an initiative of the Khronos Group (a consortium of companies with expert knowledge in video processing, including Intel, SGI and SUN) that standardizes a C-interface for multimedia programming. OpenML includes OpenGL for 3D and 2D vector graphics, extensions to OpenGL for synchronization, the MLdc library for video and audio rendering and the 'OpenML core' for media processing (unfortunately, the media processing part of OpenML is named OpenML as well; therefore we will use the term 'OpenML-mp' for the media processing capabilities below). Lately, the first implementation of the OpenML SDK was announced for summer 2003 (for Irix).

Among the concepts that are implemented in a similar fashion in JMF and OpenML-mp are the following:

- Synchronization: a media object's time base (JMF: *TimeBase* object, OpenML-mp: Media Stream Counter) is derived from a single global time base (JMF: *SystemTimeBase* object, OpenML-mp: Unadjusted System Time).
- Streaming: both environments do not manipulate media data as a continuous stream, but instead as discrete segments in buffer elements.
- Processing control: JMF uses *Control* objects and OpenML-mp uses messages for this purpose.

Other important media processing concepts are implemented differently in JMF and OpenML-mp:

- Processing chains: in JMF real-time processing chains with parallel processing can be defined (one instance for one media track is called a Codec Chain). In OpenML-mp processing operations data always flow from the application to a single processor (called a Transcoder) through a pipe and back.

- Data flow: JMF distinguishes between data sources (including capture devices, RTP servers and files) and data sinks. OpenML-mp handles all I/O devices in the same way (called Jacks).

The major advantages of OpenML-mp are:

- Integration of OpenGL, the platform-independent open standard for 3D graphics.
- A low-level C API that will probably be supported by the decisive video hardware manufacturers and should have a superior processing performance.
- The rendering engine of OpenML (MLdc) seems to have a more elaborate design than the JMF renderer components. Especially, it can be expected that the genlocking-mechanism of MLdc will prevent lost-sync phenomena, usually occurring in JMF when rendering media content with audio and video tracks longer than 10 minutes.
- OpenML-mp defines more parameters for video formats and is closer related to professional video formats (DV, DVCPRO, D1, etc.) and television formats (NSTC, PAL, HDTV, etc.)

On the other hand the major disadvantages of OpenML are:

- It is not embedded in a CASE environment like Java for JMF. Therefore application development requires more resources and longer development cycles.
- OpenML is not object-oriented and does not include a mechanism for parallel media processing.

The major drawbacks of JMF are:

- Lower processing performance because of the high-level architecture of the Java Virtual Machine. This can be reduced by the integration of native C code with the Java Native Interface.
- Limited video hardware and video format support: JMF has problems with accessing certain video codecs, capture devices and with transcoding of some video formats.

The outstanding features of JMF are:

- Full Java integration. The Java SDK includes powerful methods for distributed and parallel programming, database access and I/O processing. Additionally, professional CASE tools exist for software engineering with Java.
- JMF is free software and reference implementations exist for a number of operating systems. JMF version 2.0 is a co-production of SUN and IBM. In version 1.0, Intel was involved as well.
- JMF is extensible. Additional codecs, multiplexers and other components can be added by the application programmer.

The major demands for the VizIR project are the need for a free and bug-free media processing environment that supports distributed software engineering and has a distinct and robust structure. Issues as processing performance and extended hardware support are secondary for the project. Therefore we think JMF currently being the best choice for the implementation.

Design and implementation follow an UML-based incremental design process and rely on prototyping. UML and prototyping are employed, because they both represent state-of-the-art in software engineering. Prototyping, in addition, shows invaluable positive effect on the motivation of the developers.

5.2. User interface and communication issues

One of the most important elements of the user interface class framework is the 2.5D panel. It is based on Gl4Java [34] instead of Java3D for the following reasons: (1) Gl4Java is based on OpenGL and much faster than Java3D, (2) event handling is easier and bug-free, (3) it is easier to install (e.g. less dependent on graphics hardware than Java3D) and (4) has less bugs than Java3D.

XML reader and writer classes are based on the Java XML package (JAXP). We use the JDOM parser for XML writing (because it allows the construction of an object tree in memory and does serialization automatically) and SAX for XML parsing (because it is more flexible and faster than JDOM).

A special communication problem of VIR user interfaces is the transportation of media objects to the client computer. We do media loading in the background through an RTP stream. The Java Media Framework contains a convenient RTP-based streaming component. The user interface is operational as soon as at least a certain quantity of the media objects has arrived at the client side. This is improved by first sending a subset of representative media objects through the stream.

6. Past, current and future work

Currently, we are working on the first release of the VizIR framework. Most components of the querying framework, the database manager, the basic user interface framework (including a video renderer and a webpage renderer for thumbnail creation), the 2.5D panel and the XML communication classes are finished since autumn 2002. Next, we will implement a general-purpose query engine, a unified media handler for images and video and some of the MPEG-7 visual descriptors. A first prototype of the full framework should be finished by autumn 2003. This first version (and all following) will be released under GNU Public License.

Next we will work on other methods for feature extraction, distance measurement and video representation. New feature extraction methods we are currently working on, are semantic feature classes that enrich existing descriptor data of low-level features (e.g. MPEG-7 descriptors) with additional knowledge (modeling information, statistical dependencies, etc.) to reduce the impact of the semantic gap (first results in [42]). Concerning video representation, we will follow two approaches. First, we will implement a renderer that produces animated icons of selected keyframes of a video. The keyframes will indicate scene changes. The second approach originates in 2D animation. Short sequences of keyframes will be overlaid with an alpha-channel and thus integrated into a video thumbnail. Another idea that we will follow in the future, is the implicit definition of features from the selection of media elements or media element regions and expert knowledge. In the past we have been working on a similar idea that resulted in the system presented in [20].

7. Conclusion

This paper describes the querying and user interface framework of the Visual Information Retrieval project VizIR. The framework consists of a class hierarchy of querying classes and user interface panels with event communication, communication and configuration methods based on XML and an extension of the MRML for communication of user interfaces and query engines. The intended major outcome of the VizIR project can be summarized as follows:

- An open class framework of methods for feature extraction, distance calculation, user interface components and querying.
- Evaluated user interface components and prototypes for content-based visual retrieval.
- System prototypes for the refinement of the basic methods and interface paradigms.
- Carefully selected evaluation sets for groups of features (color, texture, shape, motion, etc.) with human-rated co-similarity values.
- Evaluation results for the methods of the MPEG-7 standard, our earlier content-based retrieval projects and other promising methods.

VizIR is open, extendible and free. A first version of the user interface part (3D interaction panel, XML-communication classes) is available since autumn 2002, the first release of the full framework should be ready by autumn 2003 and will be available under GNU Public License. We would like to invite interested research institutions to join the discussion and participate in the design and implementation of the open VizIR project. Contact the authors to join the project and/or get a copy of the available pre-release software.

Acknowledgements

The VizIR project is supported by the Austrian Scientific Research Fund (FWF) under grant no. P16111–No 5.

Appendix A

This appendix contains the document type definition (DTD) for the essential part of our MRML extension. The extension includes elements for context-free media and media group definition, descriptor definition and query definition according to our querying paradigm. It is based on version 1.0 of the MRML definition presented in [8] (see Section 2.2 for details). The tags below can be easily integrated into MRML by adding *logicalQuery* and *mediaGroup* as sub-tags of the *mrml* tag.

A.1. Media and media group definition

In MRML media objects can be context-sensitively defined as *user–relevance–elements* (for querying) or as *query–result–elements*. For initialization we add a tag for general media definition:

```
< !ELEMENT mediaObject (descriptor*)>
< !ATTLIST mediaObject
      dataLocation CDATA #REQUIRED
      iconLocation CDATA #REQUIRED>
```

dataLocation and *iconLocation* are URLs. As far as we understand, the *collection* tag of MRML cannot be used for the definition of media groups (for querying, etc.). We define the following element for this purpose:

```
< !ELEMENT mediaGroup (mediaObject+)>
< !ATTLIST mediaGroup
      id CDATA #REQUIRED
      type (positive|negative|neutral|init|other) ‘positive’>
```

The first three types define querying groups. The fourth is for initialization. Neutral examples are explicitly excluded from the query. Their properties are marked as irrelevant for the querying process.

A.2. Descriptor definition

MRML uses the *algorithm*-construct for the definition of features. For extended use we define arbitrary descriptors as follows:

```
< !ELEMENT descriptor EMPTY>
< !ATTLIST descriptor
      name CDATA #REQUIRED
      value CDATA
      distanceValue CDATA>
```

distanceValue is a special field used only when media objects are grouped to describe the layout in distance space (related to the query examples) instead of feature space.

A.3. Logical retrieval query definition

According to our Logical Retrieval approach, a query can be defined by the following elements:

```
<!ELEMENT logicalQuery (clusterDefinition+)>
<!ELEMENT clusterDefinition (clusterRestriction+)>
<!ELEMENT clusterRestriction (clusterDimension+)>
<!ELEMENT clusterDimension (mediaGroup,descriptor)>
<!ATTLIST clusterDimension
     lowerBound CDATA #REQUIRED
     upperBound CDATA #REQUIRED>
```

See Section 4.3 for an example.

References

[1] S. Santini, R. Jain, Beyond query by example. ACM Multimedia, (1998) 345–350.

[2] S. Santini, R. Jain, Similarity Measures, IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (1999) 871–883.

[3] J. Barros, J. French, W. Martin, Using the triangle inequality to reduce the number of comparisons required for similarity based retrieval, in: Proceedings of SPIE Storage and Retrieval for Image and Video Databases, San Jose, USA, 1996, pp. 392–403.

[4] Benchathlon Network Website. http://www.benchathlon.net (last visited: 2003–03–20).

[5] TREC video retrieval competition website. http://www-nlpir.nist.gov/projects/trecvid/ (last visited: 2003–03–20).

[6] H. Eidenberger, C. Breiteneder A Framework for Visual Information Retrieval, in: Proceedings of Visual Information Systems Conference, HSinChu, Taiwan, 2002, pp. 105–116.

[7] H. Eidenberger, C. Breiteneder A Framework for user interfaced design in Visual Information Retrieval, in: Proceedings of IEEE Multimedia Software Engineering Symposium, Newport Beach, USA, 2002, pp. 255–262 (published on CD).

[8] Multimedia Retrieval Markup Language Website. http://www.mrml.net (last visited: 2003–03–20).

[9] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, P. Yanker, Query by image and video content: the QBIC system, IEEE Computer 28 (9) (1995) 23–31.

[10] J. Bach, C. Fuller, A. Gupta, A. Hampapur, B. Horowitz, R. Humphrey, R. Jain, C. Shu, The Virage image search engine: an open framework for image management, in: Proceedings of SPIE Storage and Retrieval for Image and Video Databases, San Jose, USA, 1996, pp. 76–87.

[11] J.R. Smith, S. Chang, VisualSEEk: a fully automated content-based image query system, in: Proceedings of ACM Multimedia Conference, Boston, USA, 1996, pp. 87–98.

[12] A. Pentland, R.W. Picard, S. Sclaroff, Photobook: tools for content-based manipulation of Image databases, in: Proceedings of SPIE Storage & Retrieval Image & Video Databases, San Jose, USA, 1994, pp. 34–47.

[13] M. Ortega, R. Yong, K. Chakrabarti, K. Porkaew, S. Mehrotra, T.S. Huang, Supporting ranked Boolean similarity queries in MARS, IEEE Transactions on Knowledge and Data Engineering 10 (6) (1998) 905–925.

[14] S. Santini, R. Jain, Integrated browsing and querying for image databases, IEEE Multimedia 3 (7) (2000) 26–39.

[15] GNU Image Finding Tool Website. http://www.gnu.org/software/gift/ (last visited: 2003–03–20).

[16] E. Oomoto, K. Tanaka, OVID: design and implementation of a video-object database system, IEEE Transactions on Knowledge and Data Engineering 5 (4) (1993) 629–643.

[17] J.K. Wu, C. Lam, B.M. Mehtre, Y.J. Gao, A. Desai Narasimhalu, Content-based retrieval for trademark registration, Multimedia Tools and Applications 3 (3) (1996) 245–267.

[18] T. Chua, L. Ruan, AVideo retrieval and sequencing system, ACM Transactions on Information Systems 13 (4) (1995) 373–407.

[19] MPEG-7 Documents Website. http://mpeg.telecomitalialab.com/working_documents.htm#MPEG-7 (last visited: 2003–03–20).

[20] C. Breiteneder, H. Eidenberger, Automatic query generation for content-based image retrieval, in: Proceedings of IEEE Multimedia Conference, New York, USA, 2000, pp. 705–708.

[21] G. Sheikholeslami, W. Chang, A. Zhang, Semantic clustering and querying on heterogeneous features for visual data, in: Proceedings of ACM Multimedia Conference, Bristol, UK, 1998, pp. 3–12.

[22] M. Nakazato, L. Manola, T.S. Huang, ImageGrouper: Search, Annotate and organize images by groups, in: Proceedings of Visual Information Systems Conference, HSinChu, Taiwan, 2002, pp. 129–142.

[23] M. Tavanti, M. Lind, 2D vs. 3D, implications on spatial memory, in: Proceedings IEEE Symposium on Information Visualization, San Diego, USA, 2001, pp. 139–145.

[24] G. Robertson, M. Czerwinski, K. Larson, Data mountain: using spatial memory for document management, in: Proceedings of ACM Symposium on User Interface Software and Technology, San Francisco, USA, 1997, pp. 153–162.

[25] D.A. Keim, Visual exploration of large data sets, Communications of the ACM 44 (8) (2001) 38–44.

[26] B. Furht, S.W. Smoliar, H. Zhang, Video and image processing in multimedia systems, Kluwer Publishers, Boston, 1996.

[27] C. Breiteneder, H. Eidenberger, Performance-optimized feature ordering for content-based image retrieval, in: Proceedings of European Signal Processing Conference, Tampere, Finland, 2000 (published on CD).

[28] F. Liu, R.W. Picard, Periodicity, directionality, and randomness: wold features for image modeling and retrieval, IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (7) (1996) 722–733.

[29] M.F. Barnsley, L.P. Hurd, M.A. Gustavus, Fractal video compression, in: Proceedings of IEEE Computer Society International Conference, USA, 1992, pp. 41–42.

[30] A. Lasfar, S. Mouline, D. Aboutajdine, H. Cherifi, Content-based retrieval in fractal coded image databases, in: Proceedings of Visual Information and Information Systems Conference, Amsterdam, Netherlands, 1999.

[31] Y. Rui, T. Huang, S. Chang, Image retrieval: past, present and future, Journal of Visual Communication and Image Representation 10 (1997) 1–23.

[32] Java Media Framework Website. http://java.sun.com/products/java-media/jmf/ (last visited: 2003–03–20).

[33] S. Fels, K. Mase, Interactive Video Cubism, in: Proceedings of ACM International Conference on Information and Knowledge Management, Kansas City, USA, 1999, pp. 78–82.

[34] GL4Java Website. http://www.jausoft.com/products/gl4java/gl4java_main.html (last visited: 2003–03–20).

[35] H. Eidenberger, C. Breiteneder, Visual similarity measurement with the feature contrast model, in: Proceedings of SPIE Storage and Retrieval for Media Databases, Santa Clara, USA, 2003 (published on CD).

[36] H. Frei, S. Meienberg, P. Schauble, The perils of interpreting recall and precision, in: N. Fuhr (Ed.), Information Retrieval, Springer, Berlin, 1991, pp. 1–10.

[37] J.S. Payne, L. Hepplewhite, T.J. Stonham, Evaluating content-based image retrieval techniques using perceptually based metrics, SPIE Transactions 3647 (1999) 122–133.

[38] C.E. Osgood, G.J. Suci, B.H. Tannenbaum, The Measurement of Meaning. University of Illinois Press, Urbana, 1971.

[39] H. Eidenberger, C. Breiteneder, An experimental study on the performance of visual information retrieval similarity models, in: Proceedings of IEEE Multimedia Signal Processing Workshop, St. Thomas, US Virgin Islands, 2002 (published on CD).

[40] OpenML Website. http://www.khronos.org/ (last visited: 2003–03–20).

[41] DirectX Website. http://msdn.microsoft.com/library/default.asp?url = /library/en-us/wcegmm/htm/dshow.asp (last visited: 2003–03–20).

[42] H. Eidenberger, C. Breiteneder, Semantic feature layers in content-based image retrieval: implementation of human world features, in: Proceedings of International Conference on Control, Automation, Robotics and Computer Vision, Singapore, 2002 (published on CD).

[43] T. Kohonen, J. Hynninen, J. Kangas, J. Laaksonen, SOM-PAK: The Self-organizing Map Program Package, HUT Technical Report, Helsinki, Finland, 1995.

[44] C. Nastar, M. Mitschke, C. Meilhac, Efficient Query Refinement for Image Retrieval, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Santa Barbara, USA, 1998, pp. 547–552.

[45] M. Wood, N. Campbell, B. Thomas, Iterative refinement by relevance feedback in content-based digital image retrieval, in: Proceedings of ACM Multimedia Conference, Bristol, UK, 1998, pp. 13–20.

A Data Management Layer for Visual Information Retrieval

Horst Eidenberger and Roman Divotkey
Vienna University of Technology
Institute of Software Technology and Interactive Systems
Favoritenstrasse 9-11, A-1040 Vienna, Austria
+43-1-58801-18853
{eidenberger, divotkey}@ims.tuwien.ac.at

ABSTRACT

This case study describes the data management layer of the VizIR visual information retrieval project. VizIR is an open source framework of software tools for visual retrieval research. In content-based multimedia retrieval media objects are described by high-dimensional feature vectors. These feature vectors have to be stored in an efficient way in order to accelerate the retrieval process. VizIR database management is based on object-oriented persistence management. The database interface has a three tier architecture: a pattern-based persistence system hides the underlying database, an object-relational mapping system maps classes to entities and a relational database provides state-of-the-art database features (transactions, integrity, recovery, etc.). The described database management prototype can be downloaded from the VizIR project website.

Categories and Subject Descriptors

H.2.4 [**Database Management**] Systems – *Multimedia databases, object-oriented databases, relational databases*.
H.2.8 [**Database Management**] Database Applications – *Data mining, image databases*.

General Terms

Management, Performance, Design, Reliability, Experimentation.

Keywords

Content-based Visual Information Retrieval, Video Retrieval, Image Retrieval, Object-oriented Database Design, Database Management, Persistence Management, High-dimensional Indexing, Multimedia Databases.

MDM/KDD'04, August 22, 2004, Seattle, WA, USA.

1. INTRODUCTION

Content-based visual information retrieval (VIR) is a field of multimedia research that aims at extracting meaningful (semantic) media information directly from the pixel level. Sophisticated algorithms (e.g. the MPEG-7 visual features [2, 7]) are used to locate relevant information (features, descriptors) in media objects. Usually, features are represented as high-dimensional data vectors. For example, if all visual MPEG-7 features are used to describe a media object, the data vector has more than 320 dimensions. Dis-similarity of media objects is measured as distance between feature vectors. See [3, 6, 8] for more information on content-based visual information retrieval.

The fundamental database problem of VIR is to establish the efficient storage of feature vectors in order to enable fast (but still flexible) content-based multimedia data mining. This case study describes the approach we implemented to solve this problem in the VizIR project [4]. VizIR aims at developing a software workbench of free tools for content-based image and video retrieval (see Section 2 for more information on VizIR). Below, we discuss general approaches for VIR database design, describe and argue for our design decision and give details on the concrete implementation in the VizIR framework (freely available from [10]).

The paper is organised as follows. Section 2 sketches the VizIR project. Section 3 points out principal data models for feature data. Section 4 describes the VizIR data management model. Finally, Section 5 describes selected implementation issues.

2. BACKGROUND: THE VIZIR PROJECT

Even though significant amounts of research on VIR have been conducted in recent years and a considerable number of research prototypes has been developed (see [8] for a quick overview), there is still no VIR software framework available that would satisfy the researchers' needs. Firstly, as similar methods are used for image and video retrieval, it would be desirable to support both media types in one environment. Furthermore, it would accelerate research work, if state-of-the-art VIR components (e.g. space to frequency transformations, kernel-based learning algorithms, user interfaces) would be readily available in an homogeneous environment.

With the VizIR project we are intending to satisfy these demands. VizIR is a framework of resources (mainly software components implemented in Java) that are needed to build VIR prototypes. The software components include classes for media access, transportation and visualisation in user interfaces, for feature extraction (including the content-based MPEG-7 descriptors), for querying and refinement based on a novel 3D retrieval and browsing panel, for user interface design, and for visualisation of media metadata, evaluation and benchmarking. As the framework itself and all elements have to be extendible, it is imperative that the underlying database system does not make any assumptions on the elements' structure in order to keep them persistent. This constraint drives the database design considerations presented in Section 4.

VizIR is an open project and all components are free under GNU General Public License. See [4] for a more detailed description on the VizIR project. All finished components (including the database layer presented in this paper) can be downloaded as source code from the project website [10].

3. RELATED WORK: DATABASE MANAGEMENT FOR FEATURE DATA

One scientific challenge of VIR is the high dimensionality of feature vectors. For example, if all content-based MPEG-7 descriptors are used to describe an image, the description has more than 300 dimensions. Solving the dimensionality problem adequately must be one of the first issues in designing a VIR system. Still, it is mandatory for the success of VIR in general and the VizIR project in particular that the database layer meets a number of software engineering requirements: Database access has to be simple, efficient, domain-independent and operating system-independent. Additionally, the database management system has to provide traditional features (integrity, recovery, etc.). Before we designed the VizIR database layer we surveyed approaches that were used in existing VIR systems or suggested for the future.

Classic RDBMS (e.g. DB2 in QBIC [8]) fulfil all software engineering requirements easily. If used, media objects are usually stored externally, feature vectors are stored as BLOBs (often in one table per feature) and indexed by context-free structures (e.g. B-trees). Therefore, the data can only be accessed sequentially (by ID). More sophisticated access methods (such as dis-similarity measurement by distance functions; for example, implemented as stored procedures) cannot be used. Fine-granular access would only be possible, if feature vector elements could be assigned to table attributes. This is usually impossible as many features have varying length.

In recent years, sophisticated indexing structures have been developed for multimedia RDBMS (see [1] for an overview). Various R-trees, SS-trees, etc. have been proposed to allow for efficient organisation and access to high-dimensional media data. Ideally, raw media data would be stored outside the database. Feature metadata should be stored in fine granulation in the database to enable context-specific indexing. If multimedia indexing structures do exist, feature data can be selected using distance functions. Unfortunately, a number of drawbacks are connected to this approach. Firstly, most indexing structures have the tendency to become inefficient for really high-dimensional data (in the MPEG-7 case: 320+ dimensions). Secondly, most indexing structures are unable to deal with multiple distance measures in one index (state-of-the-art in content-based retrieval). Thirdly, as for classic RDBMS it is mostly impossible to define a mapping from feature vector elements to entity attributes. Finally, multimedia indexing structures are hardly implemented in classic RDBMS and more specialised products are often not operating system-independent or do not provide traditional RDBMS features.

As it is very difficult to press polymorphic feature data in relational databases in fine-granular manner, we searched for alternative approaches of data representation. XML databases seem to provide ideal structures and properties for VIR data. Features can easily be mapped to XML documents (e.g. MPEG-7 defines an XML representation of its visual features). Media objects are per se separated from metadata and stored externally. All data points can easily be accessed by using document models and (simple) querying languages (e.g. W3C DOM and XPath).

One VIR-specific example for this group of systems is the PTDOM database [11]. PTDOM defines a document object model specific for the MPEG-7 features. All features (including those based on MPEG-7 types: vector, matrix) can be accessed on a fine-granular level and retrieved using XPath and database-internal user-defined functions (similar to stored procedures). Data elements can be indexed by B-trees. Of course, additionally, more sophisticated multimedia indexing structures could be implemented as well. The main drawback of PTDOM, in terms of practical application, is that the currently available implementation is strongly bound to commercial, operating system-dependent helper libraries.

The last VIR-specific approach that may become relevant in the future is the media mediator concept [9]. Media mediators are functions that are used to access media data live during a query. Conceptually, media mediators are defined on a semantic level and mapped to low-level features that extract information from the media samples. Theoretically, media mediators can be used to define arbitrary operations on media data but, as well, to implement distributed querying environments. The advantages of the media mediator concept are that everything is done on the fly and media objects are accessed in a fine-granular way. On the other hand, obviously, the comprehensive operations needed to implement mediators would be extremely resource-consuming. Additionally, it would be almost impossible to accelerate the querying process using indexing structures. These drawbacks make it unlikely that the media mediator concept can ever be implemented in its original form. Still, if particular operations could be identified as basic building blocks for media mediators, these operations could be computed prior to query execution. Hereby, the querying process could be dramatically accelerated while the flexibility of the concept would be largely preserved.

4. VIZIR DATA MANAGEMENT MODEL

Below, we describe the data management model we designed for the VizIR project from the described palette of approaches. Subsection 4.1 describes the design decision. Subsequent subsections describe all relevant aspects of the VizIR data management model.

4.1 Use case-driven design decision

Surveying principal VIR approaches showed that we could

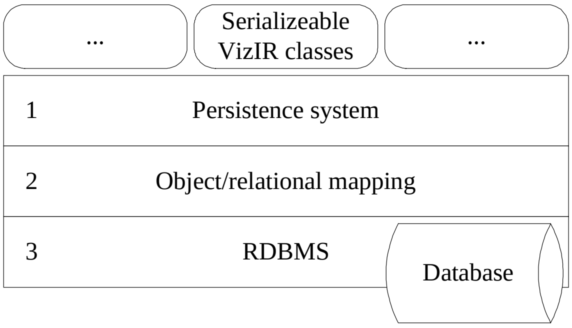


Figure 1. Layer structure of VizIR persistence system.

basically choose between a classic RDMBS (with self-implemented multimedia indexing structures) and an XML database. As VizIR is a software engineering project, we decided to follow a best practice and perform the database decision use case-driven.

VizIR is intended for general purpose VIR. For practical applicability it should provide reliable state-of-the-art persistence management. These requirements are best satisfied by classic RDBMS. An XML database would be a good choice, because the (implemented) visual MPEG-7 features are available as XML documents. Additionally, most feature structures can easily be represented in XML form. On the other hand, even professional XML databases have serious problems with handling large XML documents. Generally, implementing multimedia indexing structures would hardly make sense, since most features require variable distance measures. In this situation, an index would have to be defined for every distance measure used in the retrieval process. Obviously, following this approach would result in significant overload of indexing metadata. Furthermore, some distance measures used in VIR are not based on metrics and, in particular, do not meet the triangle inequality requirement. For these measures it would be even more difficult to define an index. Moreover, feature structures can be organised arbitrarily (e.g. as matrices). Additionally, in many retrieval situations, the query engine has to browse through the feature vectors sequentially anyway.

Therefore, we decided that VizIR should be grounded on a relational database and indexing structures should be implemented (if required) on the application level. Since VizIR is based on the query-by-example paradigm, low-level indexing in relation to a pre-defined origin (e.g. the zero vector of distance space) would not be feasible. An index would be required for every query example. However, variable indexing concepts on the application level (e.g. heuristics) may result in valuable query acceleration.

In order to guarantee application independence and framework extendibility we decided to employ object-oriented persistence management and to map serialised software objects to tables of a relational database. Figure 1 depicts the resulting three layer structure: The persistence system layer provides the methods needed to access the database (storage and retrieval), the mapping layer maps objects to entities and the database layer provides transactions, integrity and recovery. The advantages of this solution are that (1) any mapping tool and any database can be used behind the persistence system API, (2) any serialisable object can easily be made persistent and (3) database management is fully transparent to the rest of the VizIR framework.

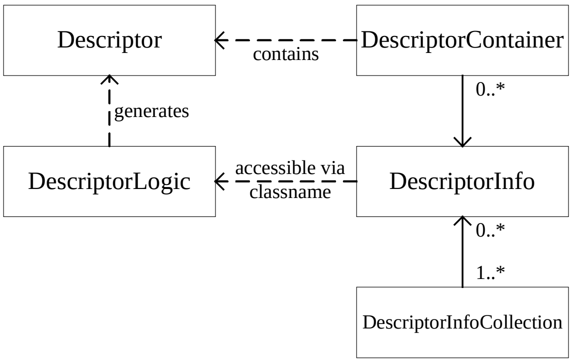


Figure 2. Descriptor-related entities (simplified).

4.2 VizIR entities

Generally, the VizIR persistence management system needs to store media-related and descriptor-related data. For media objects, just the visual data and some textual metadata are stored. The structure needed for descriptor-related data is shown in Figure 2 (in UML syntax). It is required both on the database level (as entities) and on the application level (as classes).

The main class is *DescriptorInfo*. This class holds the management methods for the other components. *DescriptorLogic* contains the extraction algorithms. *DescriptorLogic* may have an arbitrary structure: as it is stateless, it is not made persistent. The actual (XML) descriptor data are held in *Descriptor*. Since descriptors may have widely varying appearances, each *Descriptor* is encapsulated by a *DescriptorContainer*. As this class has a pre-defined, fixed structure, it can easily be made persistent (see Section 5). Additionally, every *Descriptor* may belong to a group (e.g. an MPEG-7 descriptor scheme). This relationship is implemented in *DescriptorInfo* and *DescriptorInfoCollection*.

Even though we did not have this generality in mind when we designed the VizIR persistence manager, the presented model is flexible enough to hold any type of feature data for any type of media. It could, for example, be employed to manage content-based features of audio streams or text features of arbitrary media objects.

4.3 Persistence management layer

The persistence management layer is responsible for offering all database-relevant methods to the VizIR framework while hiding the concrete implementation of the object-relational mapping and the database. Figure 3 illustrates the implemented model. The chosen design follows state-of-the-art software design patterns.

The main class *PersistenceSystem* is responsible for initialisation and the creation of all database-related entities (media objects and descriptors). Additionally, it contains a factory class for the creation of *PersistenceManager* classes (*PersistenceFactory*). *PersistenceManager* encapsulates all methods needed for database access and transaction management. This class is used

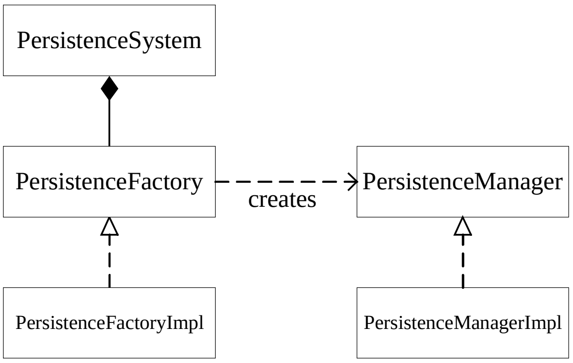


Figure 3. Persistence management classes.

to put VizIR objects under persistence control, reload objects from earlier instances and retrieve collections of objects by name. Currently, the persistence manager supports only direct queries by ID (e.g. descriptor class name). Joins can be used to retrieve, for example, all feature vectors for one media object or all media objects of a particular media collection. Generally, the level of sophistication of the querying components depends on the object-relational mapping tool.

In order to guarantee the exchangeability of the underlying mapping system, the persistence management classes implement the Bridge pattern: *PersistenceFactory* and *PersistenceManager* are just interfaces that define an API. The classes implementing these interfaces are dependent on the mapping layer. The factories *PersistenceSystem* and *PersistenceFactory* are responsible for instantiating the right implementing classes for a particular configuration of mapping layer and database.

5. IMPLEMENTATION

The Java implementation of the VizIR persistence management system makes use of the Hibernate system on the mapping layer [5]. Hibernate was selected, because it supports a wide range of commercial and open source database systems (including Oracle, DB2 and MySQL), provides powerful querying mechanisms and employs the Java Reflection API to analyse the structure of software classes. Furthermore, it is, like VizIR, an open source project that is published under GNU LGPL.

Classes that are made persistent using Hibernate have to meet a few requirements: A default constructor (without parameters, e.g. *newInstance()*) has to exist for each class and accessor methods (*get/set*) have to be available for every resource. These methods are used through the Reflection API. Optionally, every class should have an ID tag. Only two bits of information have to be provided externally: the mapping of resources to database data types and the primary/foreign key references in *1:n* and *n:m* relationships. This information is provided in simple XML documents. Even though it is possible to inform Hibernate about relationships of entities, the system leaves maintenance of referential integrity (at least of *n:m* relationships) to the user. Integrity can be achieved by implementing the *Lifecycle* interface and callback methods for data manipulation events (e.g. *onDelete()*).

We are making use of the properties of the Hibernate system to store arbitrarily shaped feature data in the database without the need to define mappings for every new *Descriptor* class: The mapping is defined for the resources of *DescriptorContainer*. Feature vectors (*Descriptor* objects) are properties of this class.

6. CONCLUSIONS AND FUTURE WORK

We tried to identify the most practicable database solution for a content-based visual information retrieval system that does neither make assumptions on features used nor on application domains. The VizIR framework is intended to be a modern, usable workbench for visual information retrieval research. Hence, grounding the system on a flexible and robust database layer was mandatory. It is interesting to notice that the best solution turned out to be a classic relational database in combination with an object-oriented persistence manager. Using the described design, VizIR can deal with arbitrary feature data and database systems. The programming effort for the VizIR user is reduced to a minimum. Actually, the VizIR persistence layer can be used to manage media objects and metadata (text or binary) of any kind. It is free software and can be downloaded from [10].

Future work will include performance tests with large MPEG-7 test datasets as well as architecture tests with mapping tools and database systems not considered so far.

7. ACKNOWLEDGMENTS

The authors would like to thank Christian Breiteneder for his valuable comments and suggestions for improvement. The VizIR project is supported by the Austrian Scientific Research Fund (FWF) under grant no. P16111-N05.

8. REFERENCES

[1] Böhm, C., Berchtold, S., and Keim D.A. Searching in High-Dimensional Spaces-Index Structures for Improving the Performance of Multimedia Databases. ACM Computing Surveys 33, 3 (2001), 322-373.

[2] Chang, S.F., Sikora, T., and Puri A. Overview of the MPEG-7 Standard. IEEE Transactions on Circuits and Systems for Video Technology 11, 6 (2001), 688-695.

[3] Del Bimbo, A. Visual information retrieval. Morgan Kaufmann, San Francisco CA, 1999.

[4] Eidenberger, H., and Breiteneder, C. VizIR – A Framework for Visual Information Retrieval. Journal of Visual Languages and Computing 14, 5 (2003), 443-469.

[5] Hibernate project website. http://www.hibernate.org/.

[6] Lew, M.S. (ed.) Principles of Visual Information Retrieval. Springer, Heidelberg, Germany, 2002.

[7] Manjunath, B.S., Salembier, P., Sikora T. Introduction to MPEG-7. Wiley, San Francisco CA, 2002.

[8] Marques, O., and Furht, B. Content-Based Image and Video Retrieval. Kluwer, Boston MA, 2002.

[9] Santini, S., and Gupta, A. Mediating Imaging Data in a Distributed System. in Proceedings of SPIE Electronic Imaging Symposium, Storage and Retrieval Methods and Applications for Multimedia (San Jose CA, January 2004), SPIE, 365-376.

[10] VizIR project website. http://vizir.ims.tuwien.ac.at/.

[11] Westermann, G.U., and Klas, W. A Typed DOM for the Management of MPEG-7 Media Descriptions. Multimedia Tools and Applications, to appear.

A Video Browsing Application
based on visual MPEG-7 Descriptors and Self-Organising Maps

Horst Eidenberger
Vienna University of Technology, Interactive Media Systems Group
Vienna, 1040 Austria

Abstract

The paper introduces a novel approach for interactive video browsing that makes video content fully transparent to the user. Video clips are analysed and indexed by two tree structures: a content index tree representing the content of automatically segmented video shots and a time index tree representing the temporal structure. The index top levels give an overview over the entire content. Subsequent levels illustrate content relationships more detailed. Every level of both trees is a two-dimensional self-organising map organising media objects by two degrees of freedom. Media objects are represented by content-based visual MPEG-7 descriptions. The implemented navigation scheme allows the user for switching between content index tree and time index tree without loosing the overview. Context information (position in the tree, content of next lower level, etc.) is permanently shown in auxiliary panels. The implementation is based on the scalable vector graphics standard (visualisation) and the MPEG-7 reference implementation. First evaluation results show that the proposed approach facilitates accessing video content in a novel way.

Keywords : Video Browsing, Video Segmentation, Self-Organising Map, MPEG-7.

1. INTRODUCTION

This paper describes a novel video browsing approach that is based on a neural network clustering technique. Interactive video browsing aims at making video content transparently accessible. Application scenarios include editing, post production and metadata annotation. Generally, video browsing problems are investigated in visual information retrieval research (VIR) [14, 17, 2]. Like the majority of VIR approaches, our approach is based on media representation by visual descriptions (e.g. colour histograms, edge maps). We employ the visual MPEG-7 descriptors [16, 15, 1, 8] to index video content and make it accessible for browsing in a web-based user interface. Indexing is performed using self-organising maps (see Subsection 2.2) [10, 9].

In our approach, video data is hierarchically indexed by two criteria: by shot content and by time. For the content index tree, video streams are segmented into shots (using automatic shot boundary detection). Shots are represented by average descriptions and visualised by representative key-frames (see Subsection 3.3 for details). Indexing is performed on multiple levels: from an overview level (coarse selection of representatives from all shots) to multiple detail levels (fine selection of representatives from similar shots). This is similarly true for time index tree. The difference is that for the time index tree, frames are selected at certain time intervals. On each level every *n*-th frame is used for indexing. *n*, the step width, is set to a large value for the overview level and to smaller values for the detail levels (see Subsection 3.2 for details). Hence, the time index tree represents a content-independent top-down view on video data while the content index is constructed bottom-up based on shot boundaries. Since content index tree and time index tree are based on the same data, the user is enabled to switch between the two views at any time during the browsing process.

Our video browsing approach differs from related approaches in the point that it employs both browsing and retrieval techniques: Visual descriptors are used to identify shot boundaries and to describe media objects. A similarity-based clustering algorithm is employed to cluster video segments. Similar video frames are located close to each other. Since we use a two-dimensional clustering technique, two degrees of freedom are available for clustering. Content-based and time-based selection and similarity-based clustering in hierarchically organised index trees result in a structured transparent view of video data. Technically, a major novelty is that the implementation is exclusively based on free software. For example, the user interfaces are based on the scalable vector graphics standard (SVG) [21]. Description extraction is based on the free reference implementation of the MPEG-7 standard. Shot detection is based on state-of-the-art VIR procedures.

The paper is organised as follows. Section 2 sketches relevant related work including the visual MPEG-7 descriptors, the clustering technique used, automatic video segmentation and recent video browsing approaches. Section 3 describes idea and design of the video browsing application and the implemented navigation paradigm. Section 4 deals with implementation issues: descriptor selection for video segmentation, description clustering and user interface implementation. Finally, Section 5 presents experimental evaluation results.

2. RELATED WORK

2.1. Visual MPEG-7 descriptors

In the video browsing application, we use visual MPEG-7 descriptors for media description and video segmentation. The

Corresponding Author: Horst Eidenberger is with the Institute of Software Technology and Interactive Systems, Vienna University of Technology, Favoritenstrasse 9-11, Vienna, 1040, Austria. FAX: +43-58801-18898
Email: eidenberger@ims.tuwien.ac.at

visual part of the MPEG-7 standard defines several descriptors [16, 15, 1, 8]. Not all of them are actually descriptors in the sense that they extract properties of media content. Some of them are just structures for descriptor aggregation and localisation. The basic colour descriptors are *Color Layout* (first DCT coefficients of YCrCb averages of major image/frame regions), *Color Structure* (histogram of colour usage in colour regions), *Dominant Color* (colour value and percentage of eight most used colours) and *Scalable Color* (classic, scalable colour histogram). Texture descriptors are *Edge Histogram* (edge orientation histograms for 4x4 sub-regions), *Homogeneous Texture* (energy values and distributions for 40 Gabor filters) and *Texture Browsing* (average coarseness and directionality of textures). Shape descriptors are *Region-based Shape* (35 ART coefficients for Y channel) and *Contour-based Shape* (contour descriptions of segmented objects). Motion descriptors are *Camera Motion* (based on optical flow), *Parametric Motion* (motion of predefined objects) and *Motion Activity* (motion vector-based frame by frame motion).

Other descriptors are based on these low-level descriptors or on additional semantic information: *Group-of-Frames*/*Group-of-Pictures* (aggregation of *Scalable Color* descriptions), *Shape 3D* (based on 3D mesh information), *Motion Trajectory* (based on object segmentation) and *Face Recognition* (major face parameters like eye to eye distance, etc.; based on face extraction). Finally, supplementary (textual) structures exist for colour spaces, colour quantisation and multiple 2D views of 3D objects. Since our application is dealing with individual key-frames, only the listed colour, texture and shape descriptors are considered below.

2.2. Self-organising maps

The self-organising map (SOM) [10, 9, 11] is a two-layer fully connected neural network that uses feed-forward learning. SOMs are mainly intended for clustering of high-dimensional data (see [7] for a survey). The input layer is interpreted as a one-dimensional data vector. The output layer is interpreted as a two-dimensional map of clusters. The clusters of the output map may have rectangular or hexagonal shape. Each cluster of the output map is described by a weight vector pointing to its center (codebook vector). In training and application, input data vectors are mapped to the codebook vector with minimum Euclidean distance (best matching unit, BMU). SOM learning is based on a predefined map size and randomly selected codebook vectors. The map is adapted by iteratively applying input vectors, selecting the codebook vector with minimum distance and changing its location by a fraction of the distance (weighted by learning rate α).

One major innovation of SOMs over other clustering methods is the introduction of neighbourhood kernels. These two-dimensional functions define the fraction, to which the BMU is adapted but also, to which extent neighbouring codebook vectors are adapted. Thus, SOM learning means learning of cluster neighbourhoods. A typical neighbourhood kernel is the two-dimensional Gaussian density function. Using neighbourhood kernels results in somewhat 'natural' cluster structures that intuitively fit with humans' similarity perception. This property is the major reason why we are using SOMs for clustering in the video browsing application.

The tree-structured SOM [12] is a further developed SOM that allows for constructing hierarchical cluster trees. Tree-structured SOMs are related to our approach. The major difference is that tree-structured SOMs cluster the entire data on every level while in our approach every SOM consists only of a small, carefully selected fraction of the entire data (video frames). Hence, it would not have been possible to achieve the effect desired by the proposed video browsing application by using tree-structured SOMs.

2.3. Temporal video segmentation

Automatic temporal video segmentation aims at identifying shot boundaries in video streams without user involvement. In recent years, a significant number of approaches have been proposed [2]. Today, state-of-the-art automatic video segmentation procedures identify more than 90% of all transitions (including fades and wipes) in video streams at minimal numbers of false positive detections. Generally, shot transitions can be distinguished in sharp cuts and effect transitions (fades and wipes). Sharp cuts are, for example, used in news videos. Effect transitions are regularly used in sports programs.

Methods for detection of sharp cuts are either based on uncompressed media data or compressed media data. The simplest approach that uses uncompressed data is the frame difference approach: Consecutive video frames are spatially pixel-wise compared. If the sum of differences exceeds a certain predefined threshold, a cut is assumed. This approach is easy to implement but has several drawbacks: it is computation power-demanding, not robust against global changes in the video data (e.g. changed lighting conditions) and sensitive for camera movement (e.g. zooming, panning, etc.). More sophisticated approaches use visual features to summarise frames. Examples are colour histograms (global features) or edge maps (local features). Shot boundaries are assumed where the distance of feature vectors exceeds a threshold. Obviously, feature-based approaches do not suffer from lacking robustness against photographing conditions and camera movement. Furthermore, if features can be computed in advance, the cut detection process is less computation power-demanding than the frame difference approach. Recently, since the visual part of the MPEG-7 standard for multimedia content description has been released, more and more feature-based approaches employ MPEG-7 descriptors for cut detection (e.g. *Scalable Color* in [5]). In Subsection 4.2 we try to identify the best MPEG-7 descriptors for cut detection.

Most methods for sharp cut detection that are based on compressed media data make use of motion vectors (e.g. [24]). If the optical flow changes significantly from frame to frame (again, significance implemented by a threshold), a shot boundary is assumed. The major advantage of compressed data-based approaches is that they require less computation power than approaches working on the uncompressed domain. Methods for detection of effects are usually based on feature-based approaches. Twin-comparison [23] employs two thresholds: All inter-frame distances exceeding a first threshold are summed up. If the sum exceeds the threshold for sharp cuts, an effect transitions is assumed. This approach works excellently for gradual transitions as fades and wipes. The production model-based approach [6] analyses effects top-down. Models for location (wipes) and intensity (fades) changes are derived. Frame sequences fitting to the models are assumed being effect transitions.

2.4. Video visualisation for browsing

The crucial user interface issue that has to be solved in video browsing systems is the visualisation of the temporal dimension of video. The spatial content of video changes over time. Since the view does not, there is no 'natural' way to visualise video content entirely on the spatial domain. In general, there are three solutions to present video information. Firstly, integration of the full video with player controls into the environment. This approach is CPU power- and network bandwidth-consuming. Secondly, creation and usage of animated iconic structures. Even though being less demanding in terms of network bandwidth, this approach is still computation power-consuming. Thirdly, creation of two- or three-dimensional models that represent the video content.

Examples for animated iconic structures are the hierarchical video magnifier [2] and the scene transition graph [22]. The approach followed by the hierarchical video magnifier is similar to the time index tree proposed in this paper. It provides a simple hierarchical structure of key-frames: Key-frames selected from the entire video content are shown on the top level. On subsequent levels, key-frames selected from parts of the video (but at smaller intervals) are shown. Layers are simply rows of key-frames ordered by time. The user can select detail views by clicking on key-frames on higher levels. Scene transition graphs give a graph representation of video content: Shots with similar content are clustered in nodes. Nodes are connected by arcs depending on their temporal relationships.

Model-based representation is the most widely applied video visualisation method. In the simplest form an image matrix of all key-frames in a video clip is used. A more sophisticated approach is the Micon [4], an image showing the first frame of a video clip as well as the first line and the last column of all consecutive frames in a cube-like view. Micons are easy to compute and give good indication on video motion for many types of content. The main shortcoming of Micons is that perspective cannot be changed easily. The video X-ray approach provides a fully three-dimensional model of video. Video X-rays are visualised as Micons but, since the model is three-dimensional, perspective can be adapted arbitrarily. Furthermore, video X-rays allow for editing of video clips (e.g. spatio-temporal cutting, compression, etc.). Another approach from a similar direction as the Micon is mosaic visualisation [2]. In a mosaic visualisation, the frames of a video clip are glued together to a panorama-like view. Stitching is based on object motion. See [4, 2] for comprehensive introductions to these and other video visualisation techniques.

3. VIDEO BROWSING APPLICATION

This section describes the design of the proposed video browsing application. Subsection 3.1 illustrates the novel ideas implemented in the approach. Subsections 3.2 and 3.3 describe, how the two types of browsing criteria (time and content) are used. Subsection 3.4 sketches navigation in the browsing process and switching between time index tree and content index tree.

3.1. Idea and motivation

In our video browsing application, video streams are organised in tree structures. The top level gives an overview over the entire video content. Subsequent levels show detail information (on groups of shots, shots, temporal fractions of the video stream). Leaves of the tree are shots (content index tree) or single frames (time index tree), respectively. The user browses through the tree structures from top to bottom. Selecting a cluster from a map causes him stepping one level down in the index tree and seeing more details on the selected fraction of the video stream. The route taken through the index tree is visualised in the user interface by auxiliary panels (see Subsection 3.4). Generally, this paradigm is similar to the hierarchical video browser (as described in [4]). Two aspects are responsible for making video perception through the proposed video browsing application a completely new experience:

- Two organisation criteria are used: time and content
- Tree layers are maps of elements clustered by content similarity

Mostly, existing video browsing approaches offer only a single view of video: a temporal view of key-frames selected at predefined intervals (independently of the content) or a content-based view of selected representative frames. In the author's opinion this is unsatisfactory, since many applications require having both types of index available simultaneously. For example, in video archival and browsing-based retrieval, the user might – depending on event characteristics – in some cases remember the time, when something happened and in other cases, in which context something happened: Regular viewers of soccer matches can easily remember *when* a goal was shot, if it was scored in overtime and decided the match, but hardly when a free kick was executed that did not have a major impact on the game. On the other hand, the free kick can easily be remembered, if it was the result of a brutal foul by a hated defender on a beloved star of the preferred team.

Our video browser offers both views in independently organised index structures. Key-frames for time-based and content-based indexing are selected independently and clustered hierarchically using the same procedure (see Subsection 4.3). Additionally, a matching procedure is provided that allows for switching between the two views. Subsections 3.2 and 3.3 describe the two index types and Subsection 3.4 sketches the matching procedure.

The second innovation in the proposed video browsing application is making use of content-based visual information retrieval for layer organisation to support the user's visual similarity perception. Key-frames are described by content-based visual MPEG-7 descriptors (see Subsection 2.1). These media descriptions (technically, high-dimensional data vectors) are clustered using self-organising maps (SOMs, see Subsection 2.2). The result is a two-dimensional map of clusters, in which similar media objects are located closely to each other. Since we use MPEG-7 descriptors, similarity is defined on the basis of generally perceived (un-recognised)

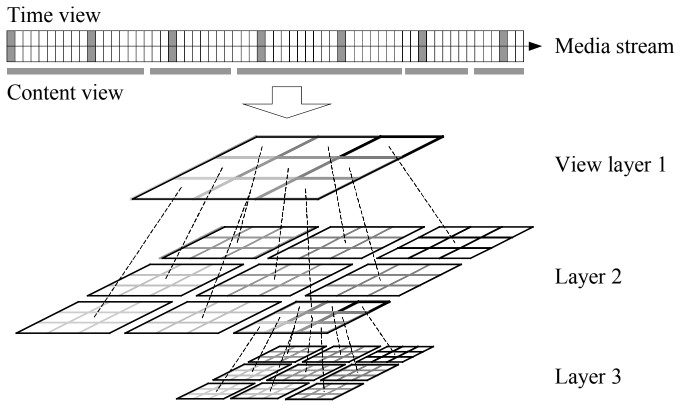


Figure 1. Video index trees. Every index tree consists of multiple layers. The number of layers depends on the media stream size. Indexes are constructed from the temporal video view (by selecting every *n*-th frame) and the content view (by selecting representative frames of shots).

image properties (e.g. colour distributions). The major advantage of this approach is that it supports human visual similarity perception. Similarity-based clustered key-frame images allow the user to judge the content of a particular layer more quickly and to uncover implicit similarities in the content of video streams. This allows, for example, to understand colour codes applied in advertisements better (e.g. bright colours for product properties that should bet perceived positively, etc.).

We use two-dimensional clustering in the video browser, because it supports human spatial perception. Additionally, it offers an additional degree of freedom in comparison to hierarchical clustering. Furthermore, maps can easily be illustrated in any type of user interface. In the past, we have also experimented with three-dimensional clustering based on Sammon mapping [19] and visualisation in virtual worlds using VRML [20]. We found that three-dimensional maps are more difficult to understand and navigation, overlapping and clipping can soon become confusing for the user. Therefore, we decided on two-dimensional clustering. SOMs were selected, because – as pointed out in Subsection 2.2 – by employing neighbourhood kernels for learning they provide a human perception-like cluster structure.

Figure 1 illustrates the resulting type of index: Layers are derived from the video stream by time and content criteria. Every layer is a two-dimensional map clustering elements by content-based features (e.g. colour, structure). Top levels give overview information. Subsequent layers give detail information. The entire video browsing application comprises two independently organised index structures inter-connected on the frame level. See Figures 8, 9 for examples.

3.2. Time index tree

For the time index tree, key-frames are selected from the video stream in a way that preserves the temporal order. Even though content-based access is an important issue in visual information browsing, the temporal structure must not be neglected. Humans have an excellent memory for the temporal order of events. The time index tree is responsible for providing a hierarchical temporal view on the media.

Key-frames are selected as follows (see Figure 1 for illustration): Every *n*-th frame of the video stream is selected. *n* (the step width) depends on *l*, the layer number (starting with '1' (top layer)). *n(l)* is defined by equation 1. Map dimensions are given by *r* (rows), *c* (columns). *round_up(X)* replaces *X* with the next higher cardinal number.

$$n(l) = round\_up\left(\frac{length(video\ clip)}{(rows.columns)^{l}}\right) \qquad (1)$$

Thus, the step width for a map on a particular layer *l* decreases proportionally to the position of the layer in the time index tree. At most, one frame per map entry (cluster) is selected. Maps on layers below the top level are mapped to clusters on the next higher level by an offset function *o(x, y)* (see equation 2): The offset function defines the starting offset for key-frame selection from the video stream.

$$o(x,y) = \left(y.r.c^{2} + x.r.c\right).n(l) + O_{l-1} \qquad (2)$$

x, *y* (cardinal numbers starting with zero) identify a cluster on layer *l-1* (that is elaborated on level *l*). O_{l-1} is the offset of the map on layer *l-1*. (Remark on navigation: It is important to notice that, since temporal order is lost in the content-based map clustering process, the pair *<x, y>* does not simply identify the *(y*rows+x)*-th cluster of the map on layer *l-1*. The corresponding cluster has to be located by establishing a link from map elements on layer *l* back to map elements on layer *l-1* using the input video stream.)

In conclusion, the content of maps of the time index tree is determined by *<n(l), o(x, y)>* pairs. On the top level, just one map exists. On subsequent levels, exactly one map exists for every cluster on the preceding level. Consequently, the time index tree is always a balanced tree. Leaves are single frames. If *rows=columns=2*, the time index tree is a quad-tree structured by visual content.

3.3. Content index tree

The content index tree is an iconic shot index. While the time index tree is constructed top-down, the content index tree is built bottom-up based on shots. Shot boundaries are detected using automatic video segmentation (see Subsections 2.3, 4.2). Even though automatic shot detection does not provide full accuracy, it is sufficiently good for our purpose.

In the indexing process, shots are represented by average media descriptions. Media descriptions are extracted from frames using the content-based visual MPEG-7 descriptors. These descriptions are averaged for the relatively coherent content of single shots. Generally, using a simple mean should be sufficient as an averaging method. The averaged descriptions are clustered using self-organising maps. In contrast to the time index tree, where only a fraction of frames are employed for clustering, all averaged descriptions are considered for clustering on the top level. Then, in a recursive process all clusters containing a number of elements that exceeds a predefined threshold are clustered again and mapped to clusters on the next higher level as detail levels. For practical reasons the threshold should be set larger than map size. Smaller threshold values would result in unnecessarily deep index structures.

Shots are the leaves of the content index tree. Since it is not

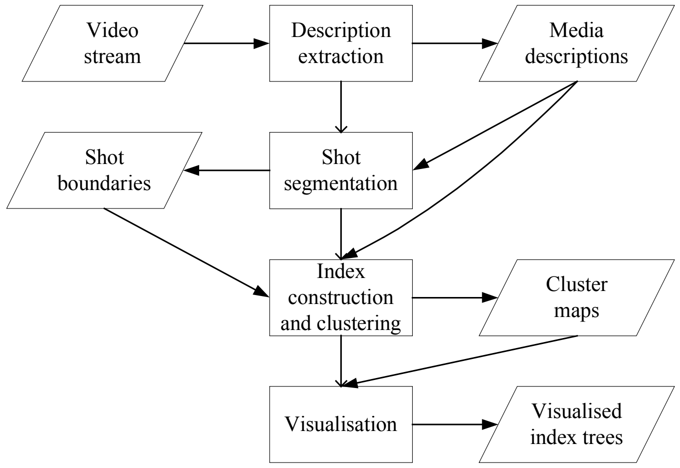


Figure 2. Workflow and data flow in the data preparation process of the video browsing application.

predictable, which content-based relations exist among shots, the content index tree is generally unbalanced containing deep, highly differentiated structures for frequently appearing content (e.g. shots of leading actors in movies) and less differentiated structures for less frequently appearing content (e.g. extras). This is desired by the approach as it supplements the context-free view provided by the time index tree elegantly.

The major design issue connected to the content index tree is selecting representative media objects for map clusters. Since, in contrast to the time index tree, clustered media descriptions are artificial, we cannot simply employ the cluster medians as representatives. A two-step procedure is required: Firstly, we identify the median average description vector (the one with minimum Euclidean distance to the codebook vector; see Subsection 2.2). Then, we identify the frame with the most similar description to the average vector (again, by Euclidean distance). This frame is selected as cluster representative and visualised in the map.

3.4. Tree matching and navigation

Above, it was mentioned that the video browsing application allows the user to switch from content index tree to time index tree and back during browsing. The implementation of this feature requires matching between the index trees. Starting from a selected cluster in one index tree, two parameters have to be determined for switching: map cluster correspondence and layer correspondence. A content index cluster and a time index cluster are defined as corresponding, if they use the same representative media object for cluster visualisation:

$$Median_{content\ index\ tree\ cluster} \equiv Best_{time\ index\ tree\ cluster} \qquad (3)$$

This is a one-to-many relationship: multiple clusters in the second tree may correspond to the selected cluster. In order to reduce the number of candidates to one, we use layer correspondence: The cluster is selected as switching target that is located on the layer with minimum hierarchical distance to the switching source. Formally:

$$\text{select } m_t \text{ with } d\left(\frac{layer(m_s)}{|source\ tree|},\frac{layer(m_t)}{|t\arg et\ tree|}\right)\rightarrow min \qquad (4)$$

where m_s, m_t are the maps in source and target index trees that contain the corresponding cluster representatives, *layer(X)* gives the layer number of map *X*, |*X*| is the span (number of layers) of index tree *X* and *d()* measures Euclidean distance. These two conditions define a unique mapping between content index tree and time index tree.

However, there is still one problem that needs to be solved. The time index tree contains all frames of an indexed video clip. Therefore, for any cluster representative in the content index tree it is possible to identify a corresponding cluster element in the time index tree. The other way around, this is not the case: In the content index tree entire shots are represented by a single frame. We suggest the following solution to overcome this problem: If, for a particular switching source in the time index tree, no corresponding frame exists in the content index tree, then the leaf map and cluster are chosen as switching target, that refer to the video shot containing the corresponding frame. In this case, the layer condition cannot be satisfied. To avoid a confusing effect on the user, she is notified by a system message.

Generally, browsing through and switching between trees can easily become confusing. We have implemented several user interface components to avoid such an effect. These components will be described in detail in Subsection 4.4. The major guidelines are: Trees are never shown entirely (information overload). Instead, we display the active layer, the preceding layer (with the selected cluster highlighted) and a preview of the next layer that corresponds to the cluster that is highlighted in the active layer. Moreover, in an additional panel the corresponding map in the non-active index tree is shown. Besides avoidance of information overload this scheme has the advantage that it can be implemented without complex and resource-consuming three-dimensional tree visualisations.

4. IMPLEMENTATION

Below, we describe relevant implementation issues of the video browsing application. Subsection 4.1 gives an overview over workflow and data flow in the index preparation process. Subsection 4.2 describes descriptor selection for automatic shot boundary detection. Subsection 4.3 gives details on the clustering process used. Finally, Subsection 4.4 sketches important visualisation and user interface aspects.

4.1. Overview

Figure 2 illustrates the index tree preparation workflow in the video browsing application. Starting from the input video stream, media descriptions are extracted. We apply the MPEG-7 image descriptors and describe each frame of a video clip by colour content, textures and general shape properties. Subsection 5.1 gives detailed information on the descriptors and parameters used. The media descriptions are the input for the automatic shot segmentation procedure. It employs description-based comparison (for sharp cuts) and twin comparison (for fades and wipes) on optimised MPEG-7 descriptions to identify shot boundaries (see Subsection 4.2 for details).

Shot boundary information and visual descriptions are fed into the index tree construction and clustering process. In the first step, averaged shot descriptions are computed. Then,

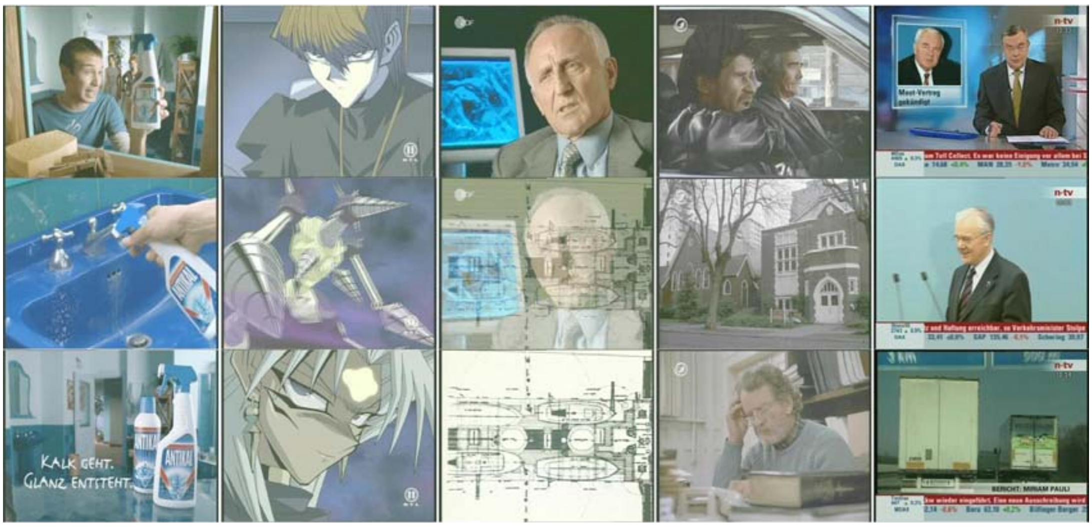


Figure 3. Example frames from advertisement, cartoon, documentary, movie and news clips employed in the evaluation (captured from German satellite television).

independently for time index tree and content index tree, frames are selected and the top views of both indexes are computed using self-organising maps (see Subsection 4.3 for details and Subsection 5.1 for parameters). Based on the top view clustering, SOM calculation is recursively repeated for the content index tree. The time index tree is computed by top-down selection of step widths and offsets (as described in Subsection 3.2). The resulting index trees are stored in a simple XML format that marks the endpoint of the pre-processing steps.

The XML document describing the two index trees is used as input for the visualisation process. Visualisation target is the web browser. Hence, we employ web-based standards for visualisation. Specifically, index tree components are visualised by scalable vector graphics (SVG). Visualisation is supplemented by event-based interaction: SVG supports ECMAScript, which is used for handling user requests. The visualisation part of the video browsing application is described in detail in Subsection 4.4.

4.2. Shot boundary detection

In the video browser we require a procedure for shot boundary detection. We use description-based cut detection in combination with the twin comparison approach for effect detection (see Subsection 2.3). Since we use visual MPEG-7 descriptors for media description, we want to use the same descriptions for cut detection. Below, we aim at identifying the best application domain-independent MPEG-7 description scheme for automatic cut detection. Optimising the performance of shot boundary detection is crucial for the quality of the video browser index trees. To reach this goal we apply the majority of content-based visual MPEG-7 descriptors on video clips of varying content and compare the results of automatic detection to ground truth information provided by human users.

Experimental Setup: We split the process of identifying the best description scheme for MPEG-7-based cut detection into two steps: First, we compute the individual performance of descriptors. Then, we try to identify combinations of descriptors that improve the individual results. This section describes the media sets used for evaluation, the visual descriptors we apply and the methods we apply for cut detection (including threshold optimisation), performance evaluation, descriptor combination and ranking.

The test data comprises media clips from five different genres: advertisements, cartoons, documentaries, movies and news (see also Subsection 5.1). Figure 3 shows example frames. The genres differ widely in cut rate and transition types used. In advertisements clips cuts occur after at least 2,5 percent of frames, in cartoons after about one percent and in documentaries, films and news after less than 0,5 percent. News programs and advertisements mostly apply sharp cuts while cartoons and documentaries often use transitions (mainly, fades) over twenty or more frames.

We apply the visual MPEG-7 descriptors (using the eXperimentation Model version 5.6) on all frames of the test videos. All colour descriptors are used: *Color Layout*, *Color Structure*, *Dominant Color*, *Scalable Color*, two texture descriptors: *Edge Histogram*, *Homogeneous Texture*, and the *Region-based Shape* descriptor. All descriptors are applied with maximum resolution. The feature vector elements are normalised to the interval [0, 1]. In total, every frame is described by a feature vector of 306 elements. For dissimilarity measurement we employ the distance measures and parameters suggested by the MPEG-7 authors (mostly, city block distance without weights).

For cut detection we define a threshold t_b (individually for each descriptor). Additionally, we use the twin comparison approach [2] to detect fades and wipes: A second threshold t_s is defined for gradual changes ($t_s << t_b$). Two indicators are computed to evaluate the performance of a descriptor: the number of correct hits and the number of false positives. Calculation of indicators is based on ground truth information provided by test users. Since we want to measure the best possible performance for each descriptor it is crucial guaranteeing that no descriptor is discriminated by false threshold values. Therefore, the thresholds are iteratively optimised in an automated procedure

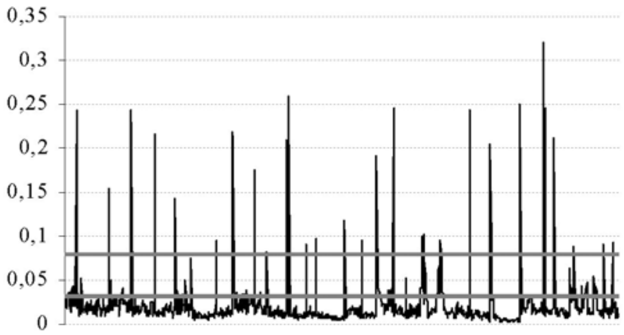


Figure 4. Frame difference signature of *Color Structure* descriptor applied to advertisement clips (X axis: time, Y axis: distance, grey lines: thresholds).

based on ground truth information. This optimisation is performed per genre. Then, the best identified thresholds are used to compute the hits and false positives indicators.

In this process we have to deal with two optimisation criteria. As the numbers of *hits* (correct / present) and *false positives* (false / present) cannot be easily combined, it is almost impossible to define a single goal function. Giving preference to one of them depends heavily on the considered type of application. Hence, we decided to base the ranking procedure on a superiority principle: One descriptor is considered being superior to another if it leads to better results for one indicator (more hits, less false positives) while being at least as good for the other indicator. Two descriptors, for which superiority cannot be clearly identified, are given the same rank (independently of the size of the performance gaps).

Based on the ranking of individual MPEG-7 descriptors we aim at identifying the best overall description scheme by combining descriptors using logical operators. Generally, cut detection results of two descriptors can be combined in two ways: Either, all cuts are assumed correct that are detected by both descriptors (*AND*) or all cuts that are detected by one of them (*OR*). An *AND* combination of two descriptors reduces the hits indicator to the value of the worse descriptor. The false positives indicator is reduced to a value in the interval [0, *min*(FP_1, FP_2)] where FP_1 and FP_2 are the false positive indicators for the first and second descriptor, respectively. In the best case, all false positives are eliminated. If two descriptors are combined by the *OR* operator the number of hits equals the hits indicator of the better descriptor. The false positives indicator becomes a value of [*max*(FP_1, FP_2), FP_1 + FP_2]. In the worst case, all false positives are part of the combined analysis. Obviously, *OR*-combined descriptors can never be superior to the involved descriptor with the higher correct hits rate. In consequence, the *OR* operator is not further considered in this study.

Results: In the first step the performance of individual MPEG-7 descriptors is analysed. For example, Figure 4 shows the distance signature of the *Color Structure* descriptor over time (frames). This feature is highly discriminant for sharp cuts and leaves enough space between cuts, fades and wipes, and object and camera movement to define the thresholds for twin comparison clearly. Actually, *Color Structure* showed the best performance of all evaluated MPEG-7 descriptors.

Descriptor	t_b	t_s	*Hits*	*FP*	*Rank*
Color Layout	0,465	0,027	95,4%	6,5%	1
Color Structure	0,096	0,036	97,2%	10,2%	1
Dominant Color	0,429	0,230	57,4%	61,1%	4
Edge Histogram	0,191	0,071	85,2%	1,9%	1
Homog. Texture	0,074	0,015	76,9%	5,6%	1
Region-based Shape	0,173	0,022	87,0%	15,7%	2
Scalable Color	0,078	0,015	68,5%	19,4%	3

Table 1. Shot detection thresholds and performance indicators for visual MPEG-7 descriptors.

Description Scheme	*Hits*	*FP*
Color Structure	97,2%	10,2%
Color Layout, Color Structure	95,4%	1,9%
Col. Layout, Col. Struct., Edge Hist.	85,2%	0,0%

Table 2. Shot boundary detection performance of best MPEG-7 description schemes.

Table 1 summarises optimal threshold values and performance indicators for all descriptors. *Color Structure* and *Color Layout* retrieve most cuts correctly, while the texture features *Edge Histogram* and *Homogeneous Texture* minimise the number of false positives. This may be the case because edge information is more robust against camera operation than colour information. Characteristics of descriptors and distance measures can be seen from the threshold values. For some descriptors (especially, *Scalable Color*) it is highly difficult to set the threshold for gradual transitions. In consequence, the hit rate is significantly less than for the best descriptors. The first rank (in terms of superiority) is shared between *Color Layout*, *Color Structure*, *Edge Histogram* and *Homogeneous Texture*. Only these descriptors were considered for combination.

Computing the performance for all *AND*-combined description schemes reveals three description schemes being superior over all others (see Table 2). The highest hit rate is achieved by the *Color Structure* descriptor alone. Using *Color Layout* and *Color Structure* in combination leads to a high hit rate and few false positives. If *Color Structure* is used in combination with *Color Layout* and *Edge Histogram*, the number of false positives drops to zero. This description scheme may be considered optimal for most applications. Consequently, it is used for shot boundary detection in the video browsing application.

4.3. Description clustering

The data clustering procedure of the video browsing application is responsible for visual similarity-based organisation of index tree layers. It takes its input from the key-frame selection procedure (as described in Subsections 3.2, 3.3). Key-frames are described by visual MPEG-7 descriptions, i.e., basically, high dimensional vectors of floating point numbers (in our case normalised to interval [0, 1]). The descriptions of key-frames are clustered by self-organising maps. Subsection 2.2 describes the learning process in self-organising maps and their specific advantages. SOMs have been used in visual information retrieval and browsing before: The PicSOM system of the Helsinki University of Technology

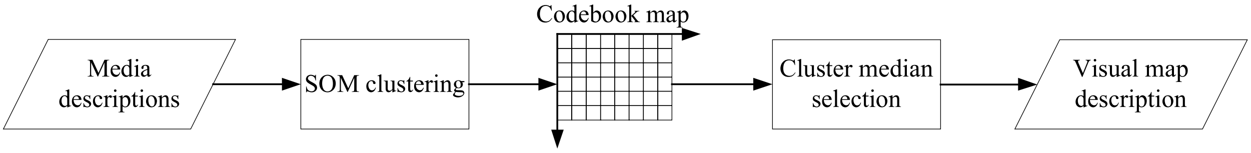


Figure 5. Workflow in MPEG-7 description clustering process. Every layer of the time index tree and the content index tree is clustered based on visual similarity criteria.

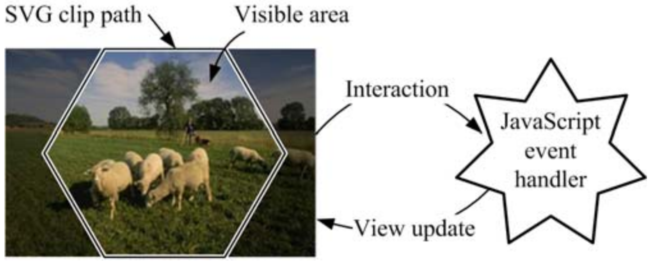


Figure 6. The SVG cell is the basic building block of the video browser user interface. A clip path is used to create the hexagonal shape. Event handling is implemented using W3C DOM event types and JavaScript listener procedures.

[13] is a successful content-based image retrieval system that employs SOMs for data clustering and incorporates iterative refinement by relevance feedback [18] into the retrieval process (based on tree-structured SOMs; see Subsection 2.2).

Self-organising maps offer two degrees of freedom for similarity-based media organisation. The SOM decides implicitly, which properties it selects for spatial organisation. Generally, the description elements with the highest variance have the most significant influence on the cluster structure. In visual information retrieval usually the strongest stimuli are colour and structure (textures, shapes) properties. Therefore, it is likely that SOMs trained from key-frames described by MPEG-7 descriptors are spatially organised by colour and structure appearance.

Figure 5 illustrates the workflow in the clustering process. Media descriptions are repeatedly fed into the SOM learning process. The output map has a predefined size. Every cluster is described by a vector pointing to the cluster center (so-called codebook vector). The codebook vectors are adapted in the learning process until the quantisation error is minimal. To compute the quantisation error, every input vector is fed into the SOM once and mapped to the codebook vector that has minimum Euclidean distance (best matching unit, BMU). The sum of distances over all vectors (normalised by the number of input vectors) defines the quantisation error: the average displacement, if input vectors would be replaced by their BMUs.

The set of codebook vectors completely defines a SOM but it does not explicitly express, to which clusters input vectors belong. Identifying the cluster structure requires locating the BMU for every input vector in an additional iteration. In some cases multiple input vectors are mapped to the same BMU and other BMUs are not associated with any input vectors. In our application, this behaviour is acceptable for the content index tree: similar shots are clustered together. Holes in the map may exist. See Figure 9 for examples. It is not acceptable for the time index tree. In the time index tree every cluster should consist of exactly one frame (time interval) that is detailed by a map on the subsequent layer. See Figure 8 for an example. To implement such a behaviour based on SOMs, we require an algorithm that identifies the best combination (e.g. in terms of quantisation error) of input vectors and codebook vectors. Since, generally this is a problem of order *O(n)=n!*, we use a simple heuristics to identify a sufficiently good *1:1* association of input and codebook vectors: For every randomly chosen codebook vector (map entry) we identify the best matching input vector (frame). Then, this input vector is removed and the procedure is repeated until all codebook vectors are mapped to input vectors. Experimental results show that this mapping procedure generates acceptable results.

After finished SOM training and identification of the BMU for every input key-frame, cluster coordinates and frame IDs of the key-frames representing clusters (see Subsections 3.2, 3.3 for the selection procedures) are stored in a simple XML document. The XML descriptions are used in the visualisation process described in the next subsection.

4.4. User interface design

User interface design for the video browsing application comprises two activities: visualisation of index tree layers and visualisation of the navigation system. As described in Subsection 3.4, we decided not to visualise entire index trees. Instead, the user interface displays the active map layer, the preceding layer and a preview of the subsequent layer (for the active cluster).

The basic building block of each layer is the cluster cell. Figure 6 describes its shape and functionality. Since we are using self-organising maps with hexagonal layout (every non-border codebook vector has six neighbours), the cluster cell is also of hexagonal shape. The cell is implemented by a scalable vector graphics (SVG) document. Every layer map consists of one cell per cluster. Hence, every map is a collection of SVGs that can easily be displayed and manipulated in a web browser window.

The SVG cell is based on the key-frame representing a cluster. A polygon of hexagonal shape is laid over this image. A copy of this hexagon is used as a clip-path to cut off those parts of the image that should not be visible in the cluster map. The resulting image is associated with an ECMAScript event listener for handling of mouse events. If the mouse cursor is moved over the cell, a listener method changes the border colour and triggers a user-defined event handler. This event handler can, for example, be responsible for displaying the preview of the map on the next lower level. The entire user interface of the video browsing application is based on this simple active SVG cell.

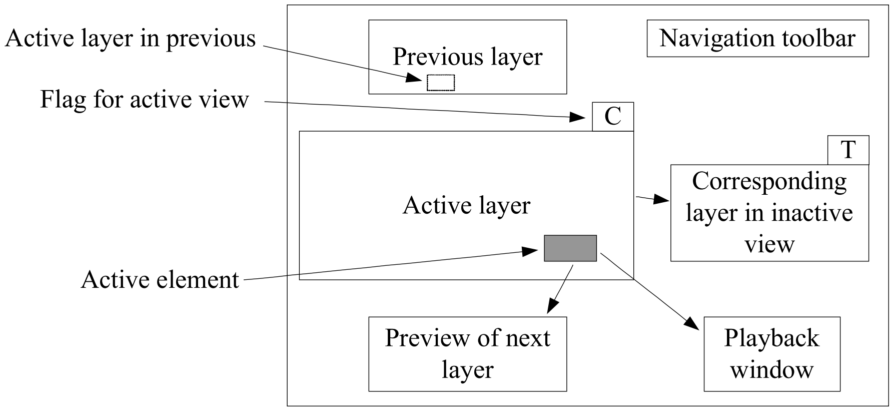


Figure 7. Navigation layout of the video browser user interface (see Subsection 4.4 for details).

Figure 7 illustrates the user interface layout. Central element is the selected layer of the active index tree ("C" for content index tree, "T" for time index tree). The selected cluster is shown highlighted. Above the active layer a smaller panel shows the next higher layer. In this window, the cluster is highlighted that is associated with the active layer. A window below the active layer shows a preview of the layer associated with the selected cluster in the active layer. If the active layer points to a leaf of the index tree (key-frames or shots, respectively), the associated video clip can be viewed in a playback window. Next to the three layers of the active index tree, the corresponding layer of the second index tree (see Subsection 3.4) is rendered in a smaller panel. Finally, on the top right a panel with navigation tools is shown (back button, history, etc.).

This user interface allows for browsing through the video content without having to visualise the entire three-dimensional index trees. In earlier experiments we found that two-dimensional user interfaces are easier to handle for non-expert users, if sufficient context information is given. Furthermore, this user interface can be implemented at a minimum demand of resources. All panels are based on the SVG cell. Interaction is exclusively based on ECMAScript and mostly executed locally. Remote access is only required if the user switches to a layer that has not been used before. The next section gives first evaluation results of the proposed video browsing application.

5. EVALUATION

5.1. Test environment

The following components were used for the prototype presented in this section. Firstly, clips with the following content were used: advertisements clips (short shots, fast changes, high quality images), cartoons (reduced colour palette, few colour gradations, slow scene changes, low motion activity), documentaries (alternating videos and animations, slow scene changes), movie clips (average image quality, average motion activity) and news clips (low motion activity, sometimes bad image quality). The media clips were captured from German satellite programs and stored in PAL format (720 by 576 pixels, 25 fps). Figure 3 shows examples.

Frames were described by seven visual MPEG-7 descriptors: *Color Layout*, *Color Structure*, *Dominant Color*, *Edge Histogram*, *Homogeneous Texture*, *Region-based Shape* and *Scalable Color*. Descriptor extraction was performed using the MPEG-7 eXperimentation model. After extraction, descriptions elements were normalised to identical intervals ([0, 1]).

Indexing was performed using self-organising maps (SOM; see Subsection 2.2). SOMs were computed with a hexagonal layout (every non-border cluster has six neighbours), six rows and eight columns. For learning, a Gaussian neighbourhood kernel was used. Maps were initialised randomly. Learning was performed in two iterations. In the first iteration 10000 learning steps were performed with learning rate α=0,05 and radius 5 (clusters). In the second iteration (fine tuning) 100000 learning steps were performed with learning rate α=0,02 and radius 3. For every dataset 15 separate SOMs were computed and the best map was chosen by the minimum quantisation error (as suggested in [11]).

The entire video browsing prototype is based on free software. Media access is implemented using Java and the Java Media Framework. Descriptions are extracted by the MPEG-7 reference implementation from the eXperimentation Model. SOMs are computed using the C-implementation provided by the Helsinki University of Technology [11]. Visualisation of maps is based on scalable vector graphics [21]. Visualisation of maps is implemented in Perl scripts and the SVG output is rendered by the Adobe SVG Viewer plug-in (tested for Netscape Navigator and Microsoft Internet Explorer). Finally, event-based interaction is implemented in ECMAScript scripts.

5.2. Experimental results

This subsection summarises our experiences with the video browser prototype. So far, we have not conducted a user study. Therefore, all presented results are preliminary based the authors' observations. In the first part of this section we will investigate the look-and-feel of the video browser. The second part discusses quantitative criteria, advantages and disadvantages as well as usage types.

Figures 8 and 9 illustrate hierarchical layer dependencies of time index tree and content index tree. The time index tree shows the top layer and two detail layers. Time-code values of key-frames depicted in clusters act as an additional source of information to the user. Since all elements are required for

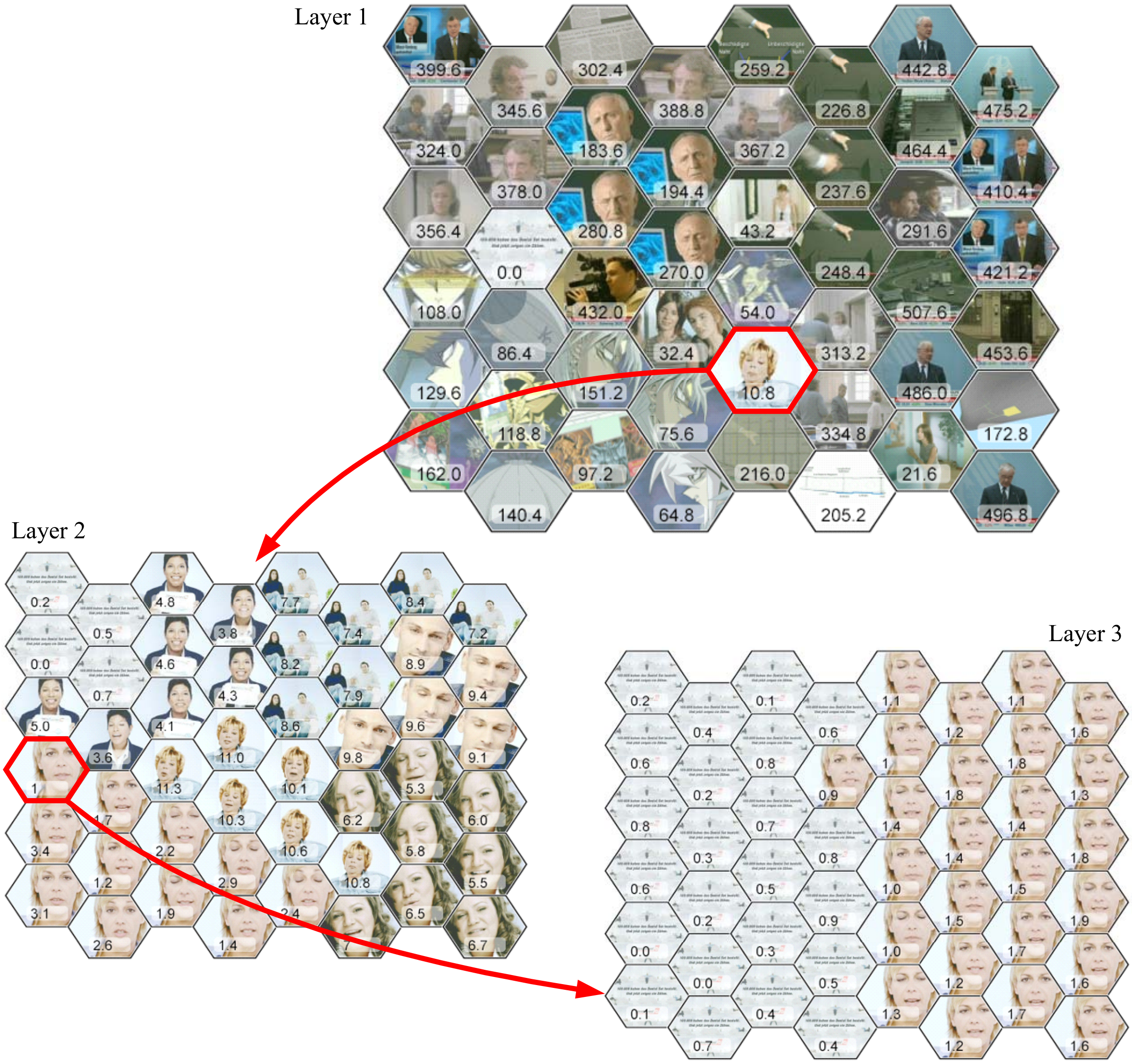


Figure 8. Example screenshot of time index view. The figure shows maps on three layers. Layer 1 is the top layer computed from the test videos used in the evaluation.

browsing, no holes are allowed in the SOMs. The algorithm describes in Subsection 4.3 solves this problem sufficiently. Some artefacts (e.g. some non-circular clusters) are due to its heuristic nature. Still, clustering of similar content is semantically understandable (especially on detail levels). The major clustering criteria seem to be colour distributions and edge layouts. This is similarly true for the content index tree (Figure 9). The figure illustrates the top layer for the test data used and one detail layer. If shots have similar content, they are clustered together. Hence, content index tree SOMs have holes and varying numbers of detail layers. Shot-content is visualised spatially. For example, Layer 2 organises the content of an animation sequence in a looped path (starting from bottom right; see time-code values). Interestingly, colour information is not the dominating clustering criterion. For example, the third and fourth cluster in the fifth row of Layer 1 of the content index tree have similar structures but different colours. In conclusion, since colour and structure are the two dominating clustering criteria, similarity is spatially perceivable in the two-dimensional SOMs.

Generally, the layer map size determines the capacity of the video browser index trees. For the example, we use maps with six rows and eight columns per row. Therefore, every map layer has 48 elements and a time index tree with three layers has a capacity of 48^3 = 110592 frames. For a frame rate of 25 frames per second (PAL, SECAM), this number equals to 73 minutes of video: Three layers are sufficient to browse through 73 minutes of content. A map size of 48 elements was chosen, because humans are able to perceive between 50 and 100 icons spatially by one look. Therefore, 48 is a very convenient number of items. Additionally, smaller maps can be computed faster and be visualised easier.

Next, we investigate major differences (in terms of practical usage) of content index tree and time index tree. The content index tree clusters dependencies in the content: Scenes that have no temporal relationship. Scenes with similar colour and

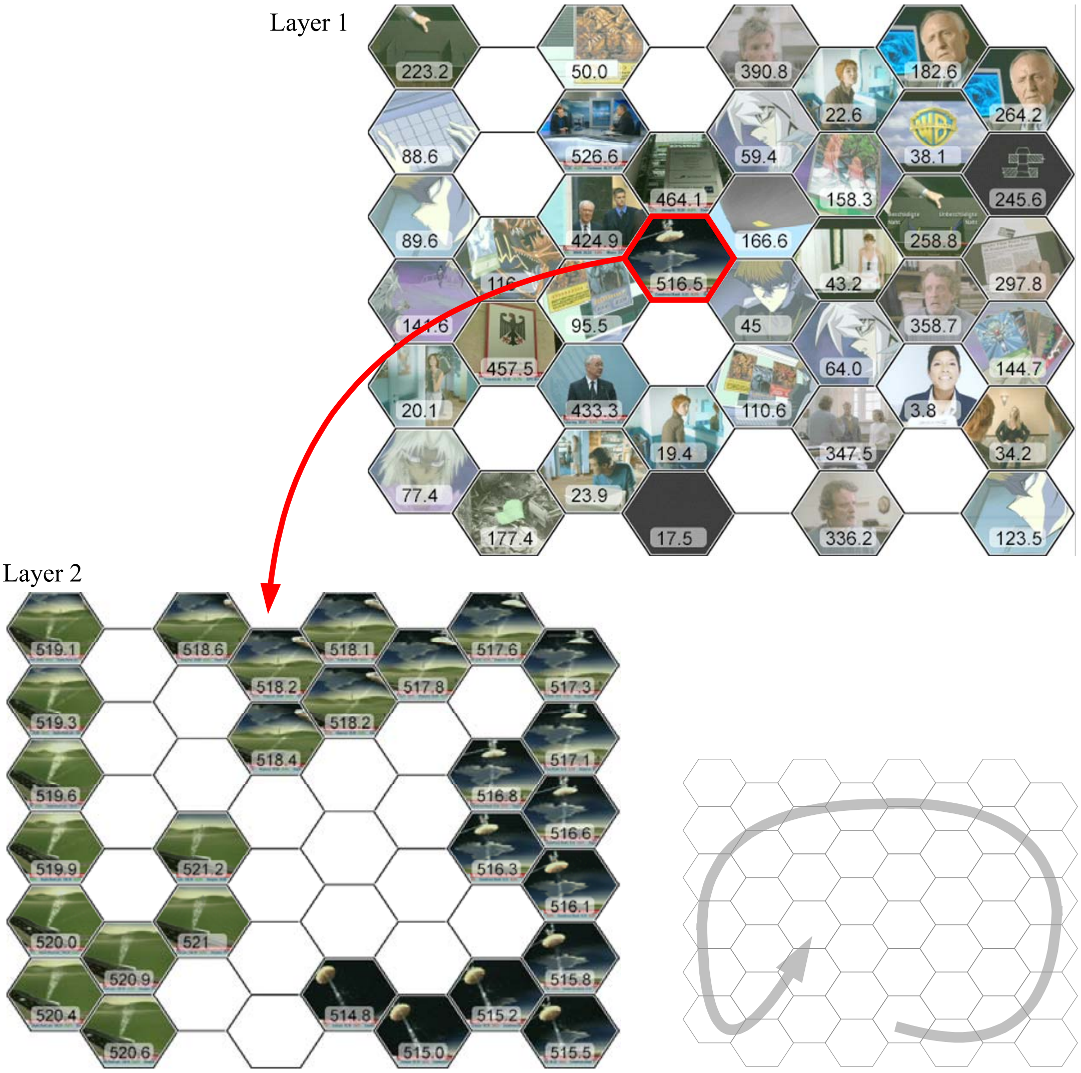


Figure 9. Example screenshot of content index view. The top layer visualises shots by representative frames. Layer 2 shows an animation shot in detail. Clustered spatially by similarity, the frames of the animation shot follow a loop.

structure properties are clustered together. For very similar scenes, one representative is chosen and the others are omitted. The content index tree shows the 'assets' of a video stream: it successfully selects prototypes of all appearing types of content and presents them to the user. Furthermore, the content index tree guarantees (on the top level) that the entire content is visualised in one view. In contrast, the time index tree clusters temporal transitions. To a certain extent it preserves the story and gives 'suggestions' for more detailed analysis in temporal order.

Technically, content index tree and time index tree are not that different: the frames selected as representatives for clusters are often located in close proximity in the video stream (of course, depending on the shot structure). If shots are short (as, for example, in advertisement clips) content-index tree and time index tree use mostly similar selections of key-frames. Additionally, since SOM clustering destroys the order in the set of selected key-frames anyway, content index tree and time index tree may appear highly similar (especially, on the top levels).

Usage experience shows that the content index tree is the main browsing tool. It is employed to identify interesting areas in the video content and analyse them in greater detail. The time index tree is mainly used in the starting phase to get a first impression of the video data, for orientation during a browsing session and as a tool for associative browsing. Since it preserves the temporal order (the story) of the video, it allows for semantic browsing through the content.

From our experiments, we draw the conclusion that the proposed video browsing approach is reasonable. Its major advantages are: Firstly, the video browser makes use of content analysis techniques and similarity-based clustering. This supports human visual perception and allows fast and effective browsing. Secondly, it summarises the assets of a video stream in an easy to overlook structure. The video browser allows real content-based random access of video data. Spatially, the video browser user interface makes use of human spatial memory. Since information overload is avoided by using small maps, the user can browse through the data quickly. Furthermore, the implemented navigation style is easy to understand. It does not implement revolutionary new interaction paradigms but is based on simple click operations. Finally, the spatial layout used in the layer maps fits to the users spatial expectations. In the video browser, video content is presented in a natural way. One major disadvantage of the proposed video browser is that temporal organisation of video is destroyed. The 'video feeling' is lost when analysing the content by the index trees. Even though illustrating the time-code together with cluster representatives allows the user to comprehend temporal organisation intellectually, the obvious visual temporal flow is lost (especially in the content index tree).

6. CONCLUSIONS

The paper describes a novel video browsing application that is based on two index structures. A time index tree visualises the temporal structure and a content index tree visualises the video stream content. The application is interactive: The user can browse through the trees and switch between the trees. Browsing is easy, because several additional panels visualise navigation-relevant context information. Furthermore, the index trees integrate visual information retrieval know-how as media objects used on index layers are clustered content-based. Media objects are described by visual MPEG-7 descriptions. Similarity-based clustering is performed using self-organising maps. From the implementation point of view, the video browser is novel as it is exclusively based on free software. Scalable vector graphics are used for index visualisation and the entire browsing application can be accessed through a web browser.

The major contribution of the video browsing application is allowing time and content-based access simultaneously. Moreover, it integrates ideas from information visualisation, information browsing and content-based information retrieval. The result is a powerful application that makes video content transparently accessible.

7. ACKNOWLEDGEMENTS

The author would like to thank Christian Breiteneder for his valuable comments and suggestions for improvement. The work presented in this paper is part of the VizIR project [3]. VizIR is funded by the Austrian Scientific Research Fund (FWF) under grant number P16111-N05.

8. REFERENCES

[1] M. Bober, "MPEG-7 Visual Shape Descriptors," *IEEE Trans. on Circuits and Systems for Video Technology*, Vol. 11, No. 6, pp. 716-719, 2001.

[2] A. Del Bimbo, *Visual Information Retrieval*, San Francisco, CA: Morgan Kaufmann, 1999.

[3] H. Eidenberger, and C. Breiteneder, "VizIR – A Framework for Visual Information Retrieval," *Journal of Visual Languages and Computing*, Vol. 14, No. 5, pp. 443-469, 2003.

[4] B. Furht, S.W. Smoliar, and H. Zhang, *Video and Image Processing in Multimedia Systems*, Boston MA: Kluwer 1996.

[5] M. Höynck, C. Mayer, and J.R. Ohm, "Application of MPEG-7 Descriptors for Temporal Video Segmentation," *SPIE Proceedings*, Vol. 4676, pp. 347-358, 2002.

[6] A. Hampapur, R. Jain, and T. Weymouth, "Production Model-based Digital Video Segmentation," *Multimedia Tools and Applications*, Vol. 1, No. 1, pp. 9-46, 1995.

[7] A.K. Jain, M.N. Murty, and P.J. Flynn, "Data Clustering: a Review," *ACM Computing Surveys*, Vol. 31, No. 3, pp. 264-323, 1999.

[8] S. Jeannin, and A. Divakaran, "MPEG-7 Motion Descriptors," *IEEE Trans. on Circuits and Systems for Video Technology*, Vol. 11, No. 6, pp. 720-724, 2001.

[9] T. Kohonen, E. Oja, O. Simula, A. Visa, and J. Kangas, "Engineering Applications of the Self-Organising Map," *Proceedings of IEEE*, Vol. 84 , No. 10, pp. 1358-1384, 1996.

[10] T. Kohonen, "The Self-Organising Map," *Proceedings of the IEEE*, Vol. 78, No. 9, pp. 1464-1480, 1990.

[11] T. Kohonen, J. Hynninen, J. Kangas, and J. Laaksonen, "SOM-PAK: The Self-Organizing Map Program Package," Helsinki University of Technology, Tech. Rep., 1995.

[12] P. Koikkalainen, and E. Oja, "Self-Organising Hierarchical Feature Maps," Proc. Neural Networks Conference, pp. 279-284, 1990.

[13] J. Laaksonen, M. Koskela, S. Laakso, and E. Oja, "PicSOM – Content-based Image Retrieval with Self-Organising Maps," *Pattern Recognition Letters*, Vol. 21, No. 13-14, pp. 1199-1207, 2001.

[14] M.S. Lew (ed.), *Principles of Visual Information Retrieval*, Heidelberg, Germany: Springer, 2001.

[15] B.S. Manjunath, J.R. Ohm, V.V. Vasudevan, and A. Yamada, "Color and Texture Descriptors," *IEEE Trans. on Circuits and Systems for Video Technology*, Vol. 11, No. 6, pp. 703-715, 2001.

[16] B.S. Manjunath, P. Salembier, and T. Sikora, *Introduction to MPEG-7*, San Francisco CA: Wiley, 2002.

[17] O. Marques, and B. Furht, *Content-Based Image and Video Retrieval*, Boston MA: Kluwer, 2002.

[18] Y. Rui, and T.S. Huang, "Relevance Feedback Techniques in Image Retrieval," in M.S. Lew (ed.), *Principles of Visual Information Retrieval*, Heidelberg, Germany: Springer, 2003).

[19] J.W. Sammon, "A nonlinear mapping for data structure analysis," *IEEE Trans. on Computers*, Vol. 18, No. 5, pp. 401-409, 1969.

[20] J.R. Vacca, *VRML* (second edition), Boston MA: Academic Press, 1998.

[21] World Wide Web Consortium, Scalable Vector Graphics standard website, http://www.w3c.org/Graphics/SVG/, last visited 2004-08-12.

[22] M. Yeung, B.L. Yeo, and B. Liu, "Extracting Story Units from long Programs for Video Browsing and Navigation," Proc. IEEE Multimedia Computing and Systems Conference, pp. 296-305, 1996.

[23] H.J. Zhang, A. Kankanhalli, and S. Smoliar, "Automatic Partitioning of Video," *ACM Springer Multimedia Systems*, Vol. 1, No. 1, pp. 10-28, 1993.

[24] H.J. Zhang, C.Y. Low, and S. Smoliar, "Video Parsing and Browsing using compressed Data," *Multimedia Tools and Applications*, Vol. 1, No. 1, pp. 89-111, 1995.