Project Proposal

Vienna University of Technology

Institute of Software Technology and Interactive Systems

VizIR

A Framework for Visual Information Retrieval

Contents

1 Problem and state of research

Global integration of information systems with the ability for easy creation and digitization of visual content (images and video) have lead to the problem of how these vast amounts of data in collections or databases can be managed. One of the crucial success factors of all approaches addressing this problem is apparently the implementation of effective but easy to handle retrieval methods. Text retrieval on indexing terms has turned out insufficient for two reasons:

- The indexing process is time consuming and expensive because it has to be done by humans
- Visual content is – naturally – often difficult or impossible to describe.

Content-based retrieval of images and video (CBVR) is 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 [10]. CBVR is a helpful addition to text retrieval systems. Its major advantages are fully automated indexing and description of visual content by visual features. However, this approach suffers from several disadvantages as well:

- The semantic gap between high level concepts presented to a user and the low level features that are actually used for querying [41].
- Subjectivity of human perception. Different persons or the same person in different situations may judge visual content differently. This problem occurs in various positions: different persons may judge features (color, texture, etc.) differently, or if they judge them in a similar way they still may perceive them in different ways [42].

Partly because of these two principle drawbacks four major problems of CBVR approaches can be identified [10]:

- Low result quality—Using only general features for all types of visual content and asking the user to choose features her- or himself leads to retrieval results of low quality.
- Complicated interfaces—Casual users are overtaxed by the demand for a definite opinion

on similarity, the selection of features and especially, by the often necessary provision of weights. Many users would not even try a typical CBVR interface, if they had the opportunity to use it. To improve the acceptance of CBVR systems simpler user interfaces are needed.

- Unsatisfactory querying performance—CBVR systems use distance functions to calculate the dissimilarity between visual objects. This process is often very slow and for large databases reply times in the range of minutes may occur.
- Lack of assessment methods—No standardized collections of images or videos exist for most types of features that could be used to assess new querying methods. One exception is the Brodatz database for textures, which is some sort of de-facto standard [23].

The most considerable past research efforts in can be observed in:

- Application of classic information retrieval methods to visual content. This includes approaches with image and video browsing techniques and semi-automatic indexing with key-words.
- Development of various kinds of features. This includes general purpose features to compute the similarity of colors, textures, shapes [31] in visual content as well as special feature types like rotation- and translation-invariant image signatures [48], shot-detection features for video clips [25], spatio-temporal motion trajectories of video objects [11] and neuronal features for face recognition [10].
- Development of user interface prototypes for video handling and browsing (e.g. micons, panoramas) and for similarity definition by spatial arrangements of images or video clips [43].
- Development of methods for query acceleration and indexing of visual content. The first include mathematical exclusion models as, for example, the triangle inequality [3] and heuristic methods, and the second methods comprise indexing structures as, for example, the R- and R*-tree family or the segment index tree for video [19].

These efforts have lead to several general-purpose prototypes like QBIC [20], Virage [1], VisualSEEk [45], Photobook [39], MARS [37], El Nino [43] and GIFT [24] for image querying and OVID [36] or VIQS for video indexing and retrieval. Some prototypes are more focussed on a particular application domain as e.g., image retrieval systems for trademarks

[50] or CueVideo for news videos analysis [9, 10]. Despite the large number of CBVR prototypes most of them are still far away from product status. After a more careful investigation they share a number of serious drawbacks:

- All of them implement only a small number of features and do not offer the developer an API for extension. An exception is IBM's QBIC system for image querying, which has (in version 3) a well-documented API for feature programming. This makes it even more regrettable that the QBIC project has been stopped.
- Due to a manifold of reasons most prototypes are not available for further research. Some of them, like the image querying engines QBIC and Virage or the video retrieval system OVID have been stopped and others, like Photobook or VisualSEEk were not released to the public.
- None of these prototypes have a structure that supports the MPEG-7 standard [32, 33]. At present, to the knowledge of the authors no MPEG-7 prototype for CBVR exists or is under development. MPEG-7 itself contains in part 6 a reference implementation of its visual descriptors and a simple querying application, which was developed for testing and simulation [33]. It does not contain a framework, a documentation of the CBVR part, a GUI, a suitable database, optimized descriptor extraction functions and performance optimized algorithms. For these reasons, unfortunately, this reference implementation cannot be used as a CBVR prototype, although it is still a good starting point for developing one.

Apart from the mentioned focal points of research and the implemented prototypes the following key issues of CBVR systems have hardly been discussed so far:

- Similarity definition—The standard approach of similarity definition in CBVR systems is first, the measurement of distances with an L1 or L2 metrics (city block distance and Euclidean distance), second, the merging of distance values for multiple features of a single object by the weighted sum and finally, the presentation of objects with the lowest distance sums as the most similar ones. In publications various authors have shown that this method is far from being the most effective one [6]. More sophisticated methods for similarity definition would result in a higher quality of retrieval results [44].
- Media sets for assessment and assessment methods—Until recently hardly any serious

effort has been undertaken to put together standardized rated image and video sets for the various groups of features and to implement suitable benchmarking algorithms. This has lead to vague, often worthless statements on the quality of various CBVR prototypes. A recent promising approach to overcome this situation is the Benchathlon project [4], that defines requirements for CBVR benchmarks and encourages the development of such benchmarks.

- Integration of computer vision methods—Surprisingly few ideas and methods have been taken over from the computer vision community up to now. Neural networks have been used for 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 applied [46].

To summarize, the most important current trends in CBVR are integration of the MPEG-7 standard for content description and the development of intuitive user interfaces for query specification. The major problem of CBVR research is the absence of any common basis for further research on features, similarity definition and the other fields of interest mentioned above.

The *VizIR* project (Visual Information Retrieval, started in Autumn 2001) *integrates the various directions of past and current research in an open-source, portable, extendible and well-documented class framework.* It was first presented to the research community at the Visual Information Systems Conference 2002 [13]. Providing the research community with this framework should help pushing CBVR research towards practical usefulness.

2 Project goals

VizIR aims at two major goals: (1) to collect various research efforts in the field of visual information retrieval under one umbrella (e.g. [6, 8, 16, 7, 18, 14, 17, 15, 12]) and (2) to provide the research community with an extendible framework of assets (classes, documentation, benchmarks, etc.). *VizIR does not intend to build a monolithic CBVR application.* It is work in progress. Goal is to maximize scientific progress. New versions are released as soon as the quality of the included components is in a mature state and well-

documented APIs do exist. The following two subsections list the project goals for research and framework development.

2.1 Research goals

The major CBVR research goals of VizIR can be organized in three groups: (1) feature design, (2) similarity measurement and (3) query acceleration.

Intended research on *feature design* includes various areas: (1) application-specific features like in [5], (2) semantic features like in [17], etc. An important goal is the implementation of the visual part of the MPEG-7 standard for multimedia content description. Obtaining this goal requires the identification of suitable extensions and supplementations of the MPEG-7 standard by additional descriptors and descriptor schemes, mathematically and logically fitting distance measures for all descriptors (distance measures are not defined in the standard) and the definition of an appropriate and flexible model for similarity definition. MPEG-7 is not information-retrieval-specific. In summary, one goal of this project is to apply the definitions of the standard to visual information retrieval problems.

CBVR *similarity measurement* is a major research focus of our group. We have developed a two-step process for distance measurement and the modeling of human visual similarity perception in [16, 18, 14]. The basic idea of this approach is that visual similarity is not just measuring distances of feature vectors. We have coupled the process model with intuitive user interface methods [12] that allow to formulate queries by arranging example objects and/or sketches of desired results. During the VizIR project we are implementing and refining the similarity model based on evaluation results.

Additionally, VizIR will support new methods for *query acceleration*. The importance of this issue becomes apparent from the large amount of data that has to be handled in such a system and the computation power that is necessary for querying by – often quite complex – distance functions. CBVR distance functions – as they are used in traditional systems and in the first step of our similarity process – have a complexity of at least $O(n)=n$, n being the size of the feature vectors. Most query acceleration approaches follow at least one of three directions:

1. Indexing of feature data—Indexing methods include tree techniques (quadtree, R-tree, etc.) and gridfiles. They suffer from the drawback that most methods support only one inherent

distance measure (most use Euclidean distance as the standard distance measure). Therefore index structures have to be implemented separately for each group of features that uses the same distance measure. Additionally, most of them become increasingly ineffective for high-dimensional data.

2. Complexity reduction—This includes coarse representation of features (reduced scales, number of histogram-bins, etc.) and redundancy reduction (e.g. factor analysis). Complexity reduction methods can be applied prior or after the feature extraction process.
3. Occlusion of media objects—Goal is 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.

In [14] we have show that – four our similarity model – approaches of the third area can be very successful. In VizIR we are continuing our work in this area to gain further improvements. Additionally, we will provide – as part of the framework – a standard API to access the media database. This allows to extend the framework by further query acceleration methods without having to do database programming. In the next subsection we point out project goals in the context of design and implementation of the asset framework.

2.2 Framework goals

Most important and as laid down in [13], *the VizIR framework has to be open, extendible, portable and well-documented*. Open means that everybody can download and use it for free. Additionally, we give away the complete source-code with documentation. With that we want to help the research community (especially smaller institutions) to accelerate CBVR research and maximize the scientific output. VizIR is extendible in various ways: researchers can add additional features, query engines, user interface components and query acceleration methods. This is guaranteed by the class structure and event-based interaction mechanisms. Communication between system components is XML-based. To be portable, *we use Java as the only platform for VizIR* development. System-dependent components (like database access and media handling) are hidden by classes with standard APIs. To guarantee that VizIR is well-documented we use Javadoc for API documentation, a standardized software-development process (Rational Unified Process) and state-of-the-art software development tools (TogetherSoft for round-trip-engineering).

VizIR implements a framework of assets. Assets include class hierarchies for querying (feature extraction, query engines, query acceleration, etc.) and user-interface-design (interaction panels, video handling methods, query refinement procedures, etc.), documentation, benchmarks with test sets, etc. Section 3 will give technical details on these objectives, the authors' approach to solve them, the intended system architecture and implementation issues.

3 Selected methodology

The first two subsections describe the methods to reach the intended project goals while the third sketches the project and time plan.

3.1 Research issues

3.1.1 Design of new features

As pointed out above, we are implementing the visual MPEG-7 descriptors in VizIR. The MPEG-7 standard – although it is a major advance in multimedia content description – standardizes only a number but not nearly all useful features. It is necessary to design and implement additional descriptors and distance functions for texture description of images (wavelets, etc.; e.g. [30]), symmetry detection of objects (useful for face detection, detection of human-made objects, etc.; [5]), object description in video streams (structure recognition from motion, etc.), object representation (scene graphs, etc.) and classic video analysis (shot detection, etc.) from uncompressed as well as compressed video streams. 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 [2] but hardly for content-based retrieval [29]. We think, that IFS should be very effective for shape description, too.

MPEG-7 allows descriptors (features) to be combined with aggregate descriptors (grid layout, time line, etc.) and grouped to descriptor schemes (DS). One task of the feature design part of the project is to discuss and identify, which combinations of descriptors make sense and how they can be implemented.

In [17] we have developed the idea of semantic feature layers (SFL) to bridge the semantic gap. This is the design of semantically related feature classes that are based on features on lower levels and include additional knowledge. 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 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:

- 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.
- 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.

Essentially, SFL are a general model for the abstraction and enrichment of low-level features.

3.1.2 Similarity measurement

The goal is the design of methods for query definition that are flexible enough to satisfy the different ways how humans can perceive and judge similarity and are still applicable in a distributed querying environment. In our earlier work we have developed the query model approach, which will be applied and extended for this task [6]. In [16] we have generalized this approach to a two-step process. 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 object. In the second step (macro-level similarity measurement) the user defines his similarity perception as a logical expression. A logical expression consists of conditions c_i of the form given in the following equation. The parameters t_{i1}, t_{i2} are thresholds for the minimum respective maximum distance of a media

object for feature *i*.

$$t_{i1} <= d_i <= t_{i2}$$

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 user interface methods that release the user from the burden to define expressions directly. Instead, he or she can define them implicitly by selecting and moving media objects in a 3D user interface (see 3.2.2).

MPEG-7 is not an information retrieval specific standard and does not include distance functions for the various descriptors. Neither does it give any recommendations for them. To be applicable with CBVR in general and our querying paradigm in particular, it is necessary to implement common distance metrics (like Minkowski metrics', Mahalanobis distance, etc.; [42]), to associate them with descriptors and to develop custom distance functions where these metrics are not applicable (e.g. object features, etc.).

Finally, we have developed a novel approach to integrate the Feature Contrast Model (FCM) in CBVR [15]. The FCM is a set-theoretic approach for similarity measurement developed by psychologists. Currently, FCM is the best model to represent human peculiarities in visual similarity perception. In VizIR, we will investigate how good our integration actually works and for which classes of features the FCM can be applied.

3.1.3 Query acceleration

In [14] we have implemented various methods for query acceleration and found that our similarity measurement process allows numerous optimizations. Still, the optimizations we used in the mentioned technical report were quite simple and straight-forward. In the future, we will implement more sophisticated query acceleration techniques in VizIR that integrate indexing, complexity reduction and heuristic approaches.

3.2 Framework design and implementation

The following subsections point out prominent aspects of the framework design. As the framework development process is iterative and based on round-trip-engineering, these remarks can only be preliminary.

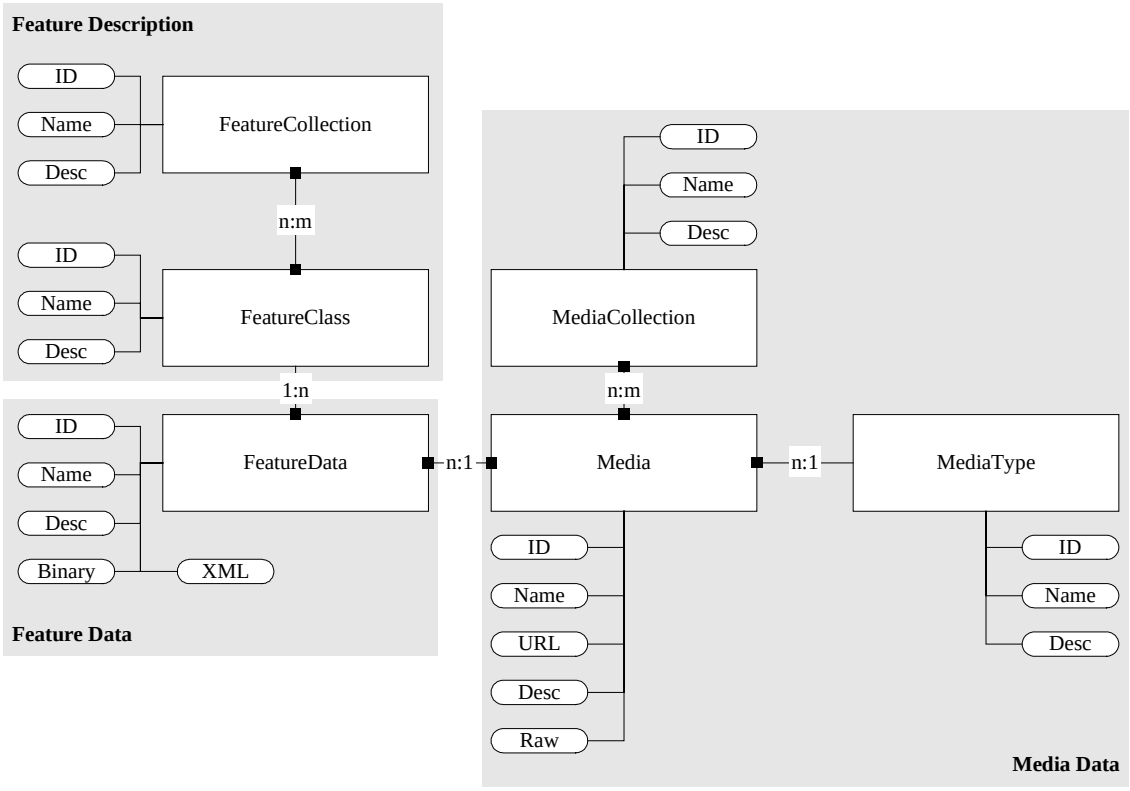

Figure 1: EER database diagram. Visual media is stored in table "Media" and associated with a single "MediaType". Each media may belong to n collections and each collection may contain m elements. Feature classes are described in table "FeatureClass" with the MPEG-7 descriptor definition language (DDL; based on XML schema). Features are organized in collections as well. Feature data is stored in binary and DDL format in table "FeatureData".

3.2.1 Querying framework design

The querying framework includes all classes, interfaces and other components that are related to querying tasks. This includes feature extraction and indexing methods, query engines, database access and media access.

As mentioned above, the VizIR prototype will be based on a standard relational database. 1 gives an overview of its tables and relations for media and feature storage. Figure 2 outlines the likely class structure of the VizIR prototype. The layout of all feature classes is predefined by the interface "Descriptor". Database access (SQL-based) is encapsulated in a database manager that offers static methods for feature and media access. Media access is hidden in media content classes. This allows changing the database access or the media access package

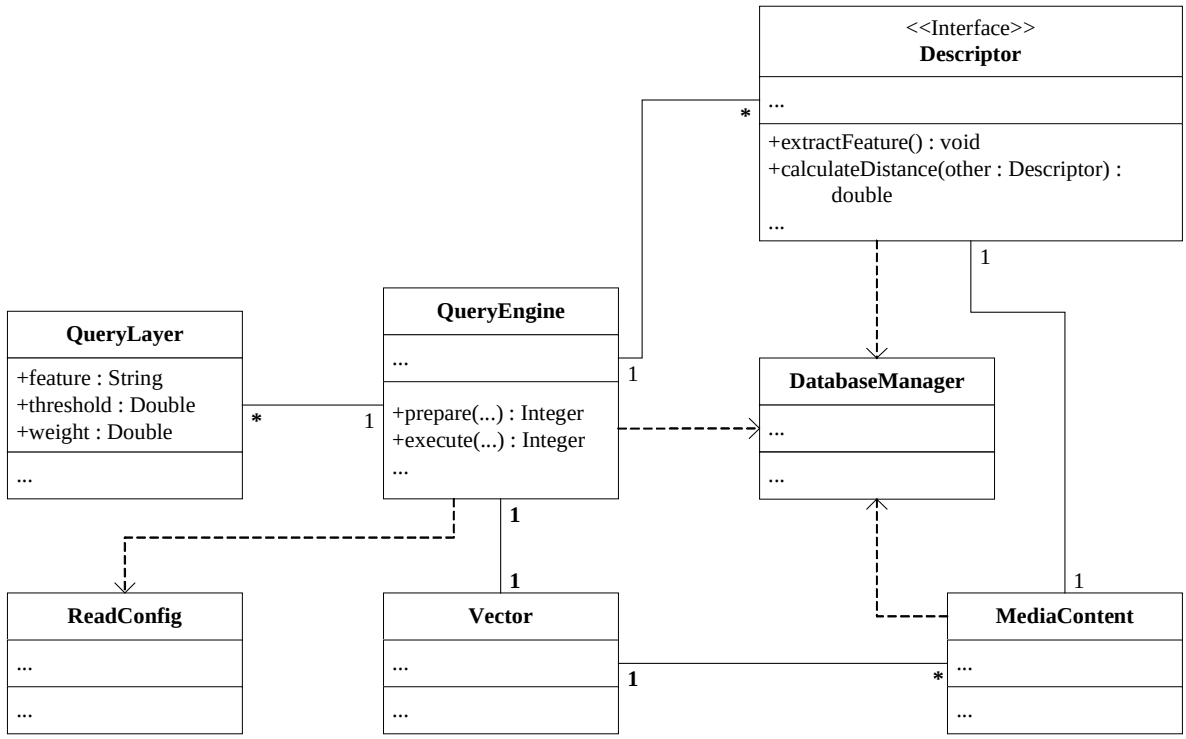

Figure 2: UML class diagram for an ideal implementation of the VizIR class framework. Key element is class "QueryEngine", which contains the methods for query generation and execution. Each query consists of a number of "QueryLayer" elements that implement exactly one feature each. All feature classes – MPEG-7 descriptors as well as all others - are derived from the interface "Descriptor" and contain methods for descriptor extraction ("extractFeature()") and distance measurement ("calculateDistance()"). Feature classes take their media content from instances of the class "MediaContent". The result of each query is a set of images (represented as MediaContent objects), which is stored in a result set "Vector" object. The methods of class "DatabaseManager" encapsulate the database access. "ReadConfig" is used to analyze input XML query descriptions.

without having to change the VizIR framework.

Based on this provisional class structure, Figure 3 shows the dynamic behavior of its components during the querying process. It has to be stressed that this is a conceptual diagram: calls to the database manager are omitted as well as details on container data structures. VizIR will support arbitrary querying paradigms. It is left to the user interface designer, which interaction panels he implements (see the following subsection for details on the user interface aspect). The implemented query engines are based on these querying paradigms. Custom query engines can be added by subclassing existing ones.

For the implementation of the basic MPEG-7 descriptors for still images and video we

intend to follow the reference implementation in part 6 of the standard. For the reasons given above and especially, because the algorithms of the reference implementation are not optimized, the redesign and implementation of the MPEG-7 descriptors is a very time- and human resources consuming task. Additionally, an API has to be defined for the creation of descriptor schemes.

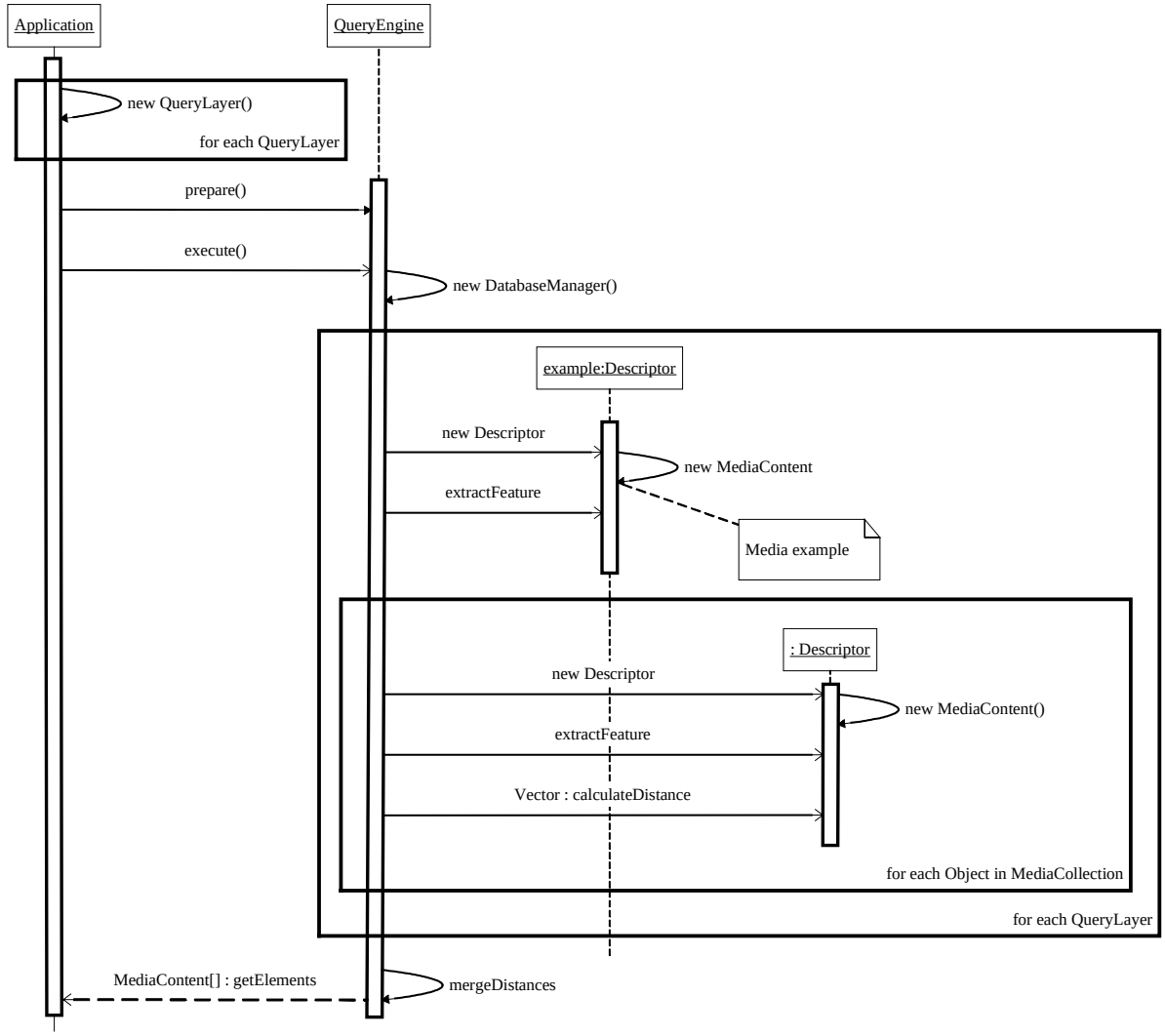

Figure 3: Schematic UML sequence diagram for the querying process. Each type of application (server, servelet, client, applet, etc.) can initiate a query by instancing a “QueryEngine” object and calling the “prepare()” method. The “execute()” method of a query creates a feature class for each “QueryLayer” of a query and extracts a descriptor by calling “extractFeature()”. These objects of a subclass of “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.

3.2.2 User interface framework design

The design of user interfaces for combined image and video querying means the implementation of methods for the following tasks: (1) similarity definition, (2) query refinement, and (3) video handling. These interfaces have to be designed as intuitive and self-explanatory as possible to guarantee high usability and in consequence increasing acceptance

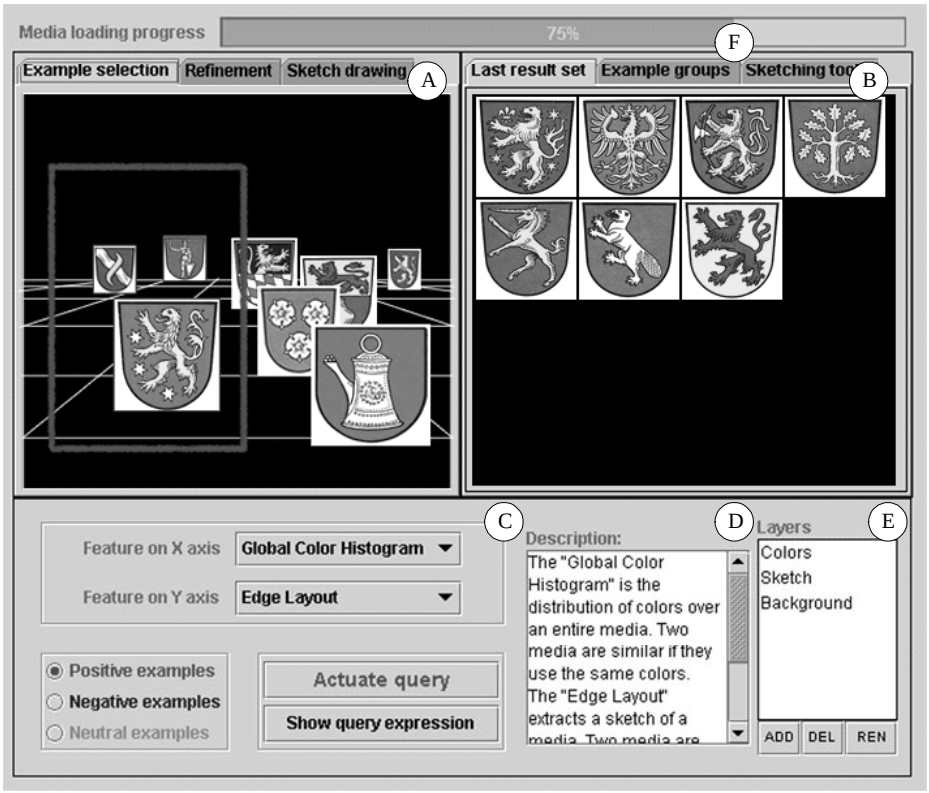

Figure 4. Screenshot of a user interface prototype. This screenshot shows a likely combination of VizIR user interface elements. Element A and B are 2.5D media panels. Elements C contains control components for feature selection, example categorization and query executions. Element D is a help panel for the currently active panel. Element E is a layer manager panel for sketch drawing and finally, Element F shows a progress bar for media loading.

for CBVR. In VizIR we follow the following approach: *We do not implement full (monolithic) user interfaces but a framework of interaction panels and other components (event classes, etc.).* This allows the user of the VizIR framework to (1) build his own personally preferred user interfaces from the given framework and (2) to extend the existing framework with additional methods (querying paradigms, interaction panels, etc.). This subsection gives just a sketch. A detailed description of the proposed VizIR user interface components framework can be found in [12].

In our previous work we have developed user interface methods that are integrated and refined in VizIR. As an example, one of the most important methods is the 2.5D media panel for query definition, result display and query refinement. For an example see element A of the screenshot in Figure 4. Basically, it is a general-purpose 3D interaction panel. The new idea is

that *we give up one dimension of movement resp. interaction of the 3D space to gain an advantage in ease of use*. Information items are represented as thumbnails and are always displayed in parallel to the screen (image plane) on the same level of the X-Z-plane. The angle of the image plane and the X-Y-plane can be varied between 0° and 90°. Movement and interaction are restricted to the horizontal (left-right) and depth (forward-backward) dimension. Because this method uses a perspective 3D view and 2D interaction, we call it 2.5 dimensional (2.5D). The limited movement and interaction has two advantages: (1) movement is less confusing than in 3D spaces and (2) can be easily mapped to 2D interaction metaphors (mouse, keypad, etc.).

Using 3D information visualization techniques instead of 2D methods has several advantages. Generally, each 3D view is just a 2D projection [47]. 3D views take advantage of human spatial memory and allow the display of 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 concerning reaction time, number of incorrect retrievals and failed trials [40]. 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 [47]. The major open problem of 3D systems in this context is the development of 3D user interaction techniques [40, 27].

In the context of CBVR, this panel is used for query formulation, result display and query refinement. Thumbnails are the rendered substitutes of media objects. Rendering (of videos, etc.) is done by user interface components as well. Groups of objects can be selected, moved and categorized (positive examples, negative examples, neutral examples). The amount of dissimilarity of thumbnails and/or groups can be defined by the distance between them. This has turned out to be a very intuitive and effective way for query formulation. Apart from that, it can be directly transferred into a query description based on the Multimedia Retrieval Markup Language and fed into various query engines. The panel shows the spatial relationships for two selected dimensions (features). Which features are chosen for the X- or Y-axis, can be controlled by a visual control component (element C of the screenshot). The upper part of this panel is initialized with all dimensions of the media space to be displayed (all implemented features). The view changes whenever new dimensions are chosen for the X-

or Y-axis.

The same panel is used for query refinement. Iterative query refinement by relevance feedback is a technique that has become state-of-the-art in information retrieval applications in the last years [35, 49]. As frequently stressed in publications on information retrieval this is a crucial task for the quality of a retrieval system [10]. The effect of such a component stands and falls with an intuitive user interface that allows the user to enter his feedback in an intuitive way. With the 2.5D media panel, the result of a query can displayed in the same panel as the query was formulated. The user can then refine his query by changing groups and distances between them. This is very intuitive. In addition, further media panels can be used to display the last result set or the elements of the various groups (e.g. element E of the screenshot: it is just another media panel with a display angle of 0°).

Video handling methods can be integrated in the user interface directly. We are developing a number of renderer classes that produce thumbnails from video objects. These classes implement state-of-the-art video handling paradigms (e.g. micons, panoramas, paper video, etc.; [23]) as well as new, better metaphors. One interesting alternative could be a spatio-temporal onion view on video objects (like in Macromedia Flash). Another, more sophisticated approach could be an object viewer for all objects and their temporal trajectories in a video shot.

Query by Example (QBE) is of course not the only querying paradigm that is implemented in VizIR. Other paradigms, like Query by Sketch (QBS) are implemented as well. Sketches for QBS can be drawn in a sketch drawing panel. This panel offers similar functionality as a painting application and interacts with toolboxes and a layer manager (like in Adobe Photoshop). The communication of panels is outlined in Figure 5.

3.2.3 Communication and configuration

The VizIR project has to cover several communication aspects: (1) Communication of user interfaces with query engines, (2) communication of query engines with the database, (3) communication of user interface components, (4) communication of distributed system components and (5) database distribution.

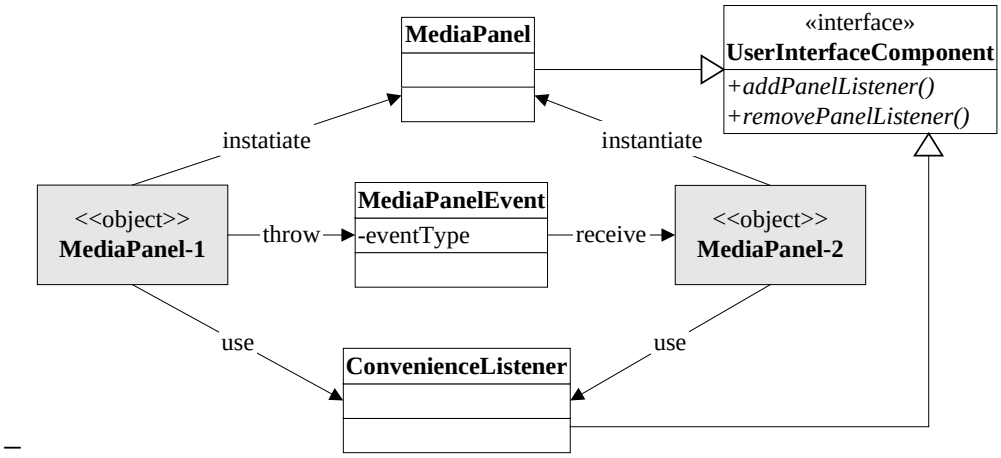

Figure 5. Event model for panel communication. Media panels communicate through "MediaPanelEvent" objects. "ConvenienceListener" objects implement standard event handling procedures for various user interface elements. Interface "UserInterfaceComponent" is implemented by each component with visual output.

The communication of user interfaces with query engines will be based on the Multimedia Retrieval Markup Language (MRML, [34]). MRML is an XML language that has been developed for exactly this purpose. Still, it suffers from several shortages. That is why we extended it with custom elements and attributes in [12]. We are not intending to create a new querying language but will use this extended MRML version in VizIR.

As mentioned above, the communication of query engines with the underlying database will be performed by a database manager class (based on SQL and JDBC) and user interface components interact through events and event listener methods.

For communication of distributed system components the Java environment permits the network-independent distribution of applications (as Applets, Servelets, etc.), objects (JavaBeans) and methods (through the Remote Method Invocation (RMI) library) to increase the performance of an application by load balancing and multi-threading. Query engines could be implemented as JavaBeans, feature extraction methods with RMI, database management through Servelets and user interfaces as Applets. Finally, database distribution is realized through standard replication mechanisms.

3.2.4 Quality assessment

Quality assessment is an important part of the VizIR project. In detail, the following tasks

have to be performed:

- Analysis of common evaluation models (recall, precision, etc.; [21, 38]) and application of other methods (systematic measures, etc.; [22]).
- Extended evaluations on the MPEG-7 descriptors and descriptor schemes as well as on the other implemented descriptors and aggregates with statistical methods.
- Evaluation of the performance optimization methods implemented in VizIR in comparison to other comparable retrieval systems (like in [14]).
- Finally, assessment of the user interfaces by volunteers who judge the video handling methods, similarity definition concepts and the overall usability of the system.

Benchathlon [4] is a recent initiative to develop benchmarks for CBVR. In Benchathlon, a benchmark algorithm replaces the user interface and steers the tested query engine through MRML. VizIR is compatible to Benchathlon because it uses MRML for the communication of user interfaces with query engines. We are planning to contribute benchmarks and test sets to the Benchathlon project.

Descriptor analysis 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 become able to evaluate the visual part of the MPEG-7 standard and extend it by new independent descriptors.

For user-based assessment we will build CBVR user interfaces from the framework and evaluate, how appropriate and self-explanatory our panels for query formulation and query refinement are.

3.2.5 Implementation aspects

The underlying relational database for the VizIR prototype could be Oracle, IBM’s DB2 or any other professional database, because all of these databases offer comprehensive features for managing large amounts of data (e.g. organization in tablespaces, etc.), data replication (online database links, per table update replication, etc.) and performance tuning (e.g. buffer sizes of the system global area, etc.). Design and implementation will follow a UML-based

incremental design process [26] and prototyping, because UML is state-of-the art in engineering and because of the valuable positive effect of rapid prototyping on the employee's motivation. TogetherSoft will be used for system design and implementation because of its outstanding features and quality considerations.

To satisfy the needs for portability and distributed querying, programming will be done in Java and within the Java environment, especially employing the Java Media Framework (JMF) for video handling, Java 2D and Java Advanced Imaging (JAI) for image analysis, Gl4Java (instead of Java3D) and Java AWT/Swing for interface design, JDBC for universal database access and Java Remote Method Invocation (RMI) and CORBA for distributed computing. Gl4Java is used, because it has a better performance than Java3D, less bugs and is easier to install. XML-handling will be done with the JavaSDK standard parsers (SAX and DOM). The preferred development environment for programming and API documentation will be Forte for Java from SUN Microsystems.

Standard statistical packages (like SPSS) and Perl scripts will be used for performance evaluation and Self-organizing Maps [28] and Advanced Resonance Theory (ART; [46]) neural networks as well as genetic algorithms for tasks like pattern matching and (heuristic) optimization (like in [8]).

3.3 Work and time plan

While drawing an organization chart seems inappropriate for a project with about five people, workflow and time planning is even more important, because there are no hidden reserves in human resources that could be activated in case of delays and unexpected difficulties.

We request funding of the VizIR project for three years. Within this time the following milestones should be achieved:

- Final design and implementation of the media database and the basic class frameworks.
- Implementation of the visual MPEG-7 features.
- Design and implementation of other features (including those features mentioned in subsection 3.1.1).

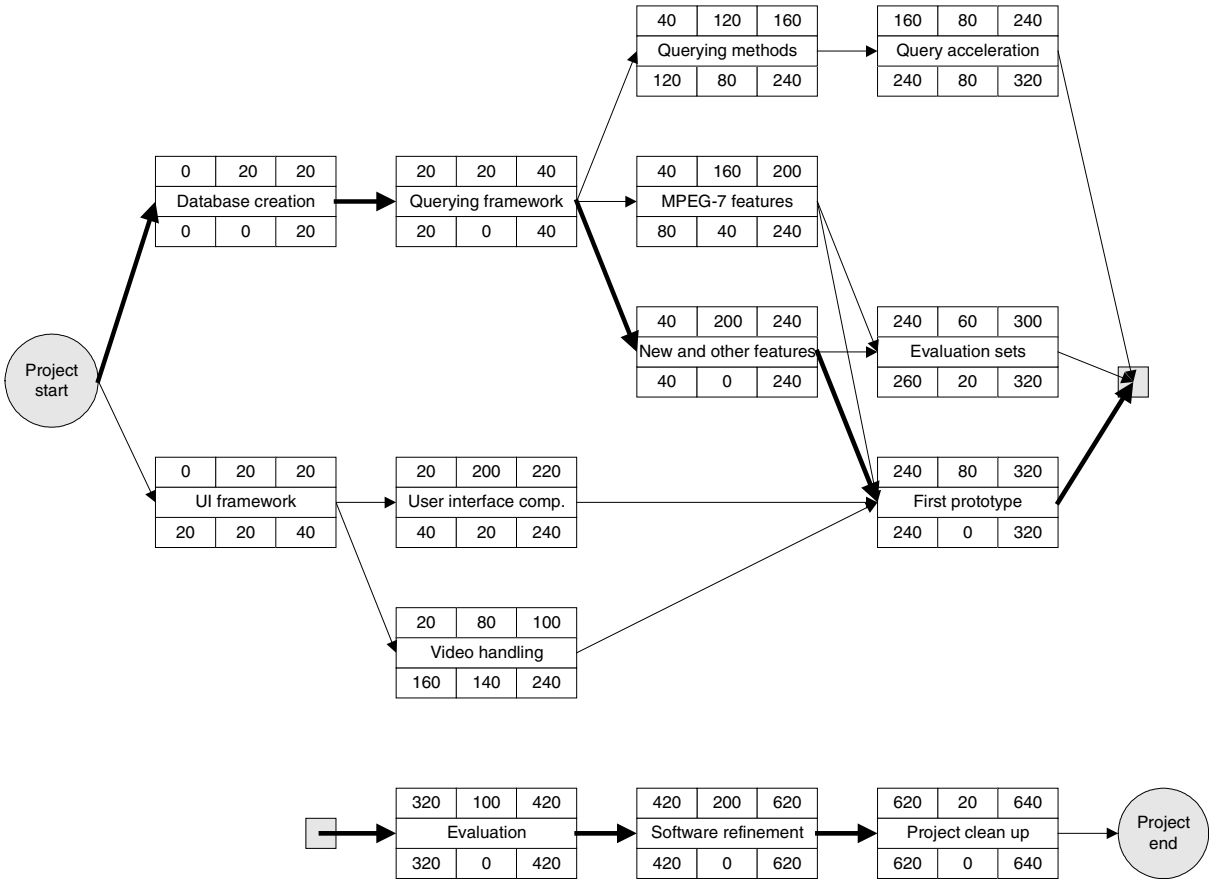

Figure 6: Project PERT diagram. Clearly, the critical path goes through the feature design and implementation task group where the design and implementation of new descriptors is the longest one. Time is given in units. We request funding for a project duration of three years, each year has about 210 workdays, therefore the overall project duration is 630 workdays or 640 units. Thus, a unit is about one workday.

- Development of suitable querying methods for flexible similarity definition and iterative refinement.
- Implementation of existing and new video handling methods like micons, hierarchical video browser, etc.
- Design and implementation of user interface components for visual querying (including feature selection, weighting, refinement, etc.)
- Design and implementation of new query acceleration methods.
- Construction and collection of evaluation sets for feature and distance function assessment.

- Assembly of visual information retrieval prototypes out of the elements of the framework for the evaluation and refinement phase.
- Refinement of the visual information retrieval framework based on a detailed evaluation of the prototype.
- Final presentation of the project outcome and project-related publications.

The PERT diagram in Figure 6 gives an overview of the project workflow and the estimated duration of all tasks. The critical path of the project goes through the feature design and implementation tasks. From this fact the authors draw the conclusion that the project management and project member deployment should be flexible enough to allow shifts from other tasks to delayed feature design and implementation tasks during the project to guarantee the prevention of overall project delays. This will be achieved by the application of a standard software development process (Rational Unified Process), appropriate tools (TogetherSoft, UML) and frequent technical reports.

4 References

1. Bach, J., Fuller, C., Gupta, A., Hampapur, A., Horowitz, B., Humphrey, R., Jain, R., Shu, C., The Virage image search engine: An open framework for image management, Proc. of SPIE Storage and Retrieval for Image and Video Databases, 1996
2. Barnsley, M.F., Hurd, L.P., Gustavus, M.A., Fractal video compression, Proc. of IEEE Computer Society International Conference, 1992
3. Barros, J., French, J., Martin, W., Using the triangle inequality to reduce the number of comparisons required for similarity based retrieval, SPIE Transactions, 1996
4. Benchathlon website, last checked 2002-06-11, http://benchath.hpl.external.hp.com
5. Breiteneder, C., Eidenberger, H., A Retrieval System for Coats of Arms, Proc. of International Symposium on Intelligent Multimedia and Distance Education, Baden-Baden, 1999
6. Breiteneder, C., Eidenberger, H., Automatic Query Generation for Content-based Image Retrieval, Proc. of IEEE Multimedia Conference, 2000

7. Breiteneder, C., Eidenberger, H., Fiedler, G., Raab, M., Lookmark: A 2.5D information visualization system, Technical report, submitted for publication at EURASIA-ICT Conference 2002, available at http://www.ims.tuwien.ac.at/~hme/tr/tr-1882-2002-04.pdf

8. Breiteneder, C., Eidenberger, H., Performance-optimized feature ordering for Content-based Image Retrieval, Proc. of X European Signal Processing Conference, Tampere, 2000

9. Chua, T., Ruan, L., A Video Retrieval and Sequencing System, ACM Transactions on Information Systems, Vol. 13, No. 4, 1995, pp. 373-407

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

11. Dimitrova, N., Golshani, F., Motion Recovery for Video Content Classification, ACM Transactions on Information Systems, Vol. 13, No. 4, 1995, pp. 408-439

12. Eidenberger, H., Breiteneder, C., A framework for user interface design in visual information retrieval, Technical report, submitted for publication at Symposium on Multimedia Software Engineering, available at http://www.ims.tuwien.ac.at/~hme/tr/tr-1882-2002-09.pdf

13. Eidenberger, H., Breiteneder, C., Hitz, M., A framework for visual information retrieval, Proc. of Visual Information Systems Conference 2002, Springer LNCS, HSinChu, Taiwan, 2002, pp. 105-116

14. Eidenberger, H., Breiteneder, C., An experimental study on the performance of visual information retrieval similarity models, Technical report, submitted for publication at IEEE Multimedia Signal Processing Workshop 2002, available at http://www.ims.tuwien.ac.at/~hme/tr/tr-1882-2002-05.pdf

15. Eidenberger, H., Breiteneder, C., A novel approach to include the Feature Contrast Model into visual information retrieval, Technical report, submitted for publication at SPIE Storage and Retrieval of Media Databases Conference, available at http://www.ims.tuwien.ac.at/~hme/tr/tr-1882-2002-12.pdf

16. Eidenberger, H., Breiteneder, C., Macro-level similarity measurement in VizIR, Proc. of IEEE Multimedia Conference 2002, Lausanne, Switzerland, 2002.

17. Eidenberger, H., Breiteneder, C., Semantic feature layers in Content-based Image Retrieval: Implementation of human world features, Technical report, submitted for publication at International Conference on Automation, Robotics and Computer Vision, available at http://www.ims.tuwien.ac.at/~hme/tr/tr-1882-2002-08.pdf

18. Eidenberger, H., Breiteneder, C., Visual similarity measurement, Technical report, submitted for publication at ACM Multimedia Conference 2002, available at http://www.ims.tuwien.ac.at/~hme/tr/tr-1882-2002-03.pdf

19. Faloutsos, C., “Indexing of Multimedia Data”, in: Apers, P. M. G., Blanken, H. M. and Houtsma, M. A. W. (Eds.), Multimedia Databases in Perspective, Springer, Berlin, 1997, pp. 219-246

20. Flickner, M., Sawhney, H., Niblack, W., Ashley, J., Huang, Q., Dom, B., Gorkani, M., Hafner, J., Lee, D., Petkovic, D., Steele, D., Yanker, P., Query by Image and Video Content: The QBIC System, IEEE Computer, 1995

21. Frei, H., Meienberg, S., Schäuble, P., "The Perils of Interpreting Recall and Precision Values", in: Fuhr, N. (Eds.), Information Retrieval, Springer, Berlin, 1991, pp. 1-10

22. Fuhr, N., Script: Information Retrieval, University of Dortmund, Dortmund, 1998

23. Furht, B., Smoliar, S. W., Zhang, H., Video and Image Processing in Multimedia Systems, second print, Kluwer, Boston, 1996

24. GIFT website, last checked 2002-06-11, http://www.gnu.org/software/gift/

25. Hampapur, A., Weymouth, T., Jain, R., Digital video segmentation, Proc. of the second ACM international conference on Multimedia, 1994, pp. 357-364

26. Hitz, M., Kappel, G., Uml @ Work, dpunkt Verlag, 1999

27. Keim, D.A., Visual exploration of large data sets, Communications of the ACM 44/8, 2001, pp. 38-44

28. Kohonen, T., Hynninen, J., Kangas, J., Laaksonen, J., SOM-PAK: The Self-organizing Map Program Package, Helsinki, 1995

29. Lasfar, A., Mouline, S., Aboutajdine, D., Cherifi, H., Content-Based Retrieval in Fractal Coded Image Databases, Proc. of Visual Information and Information Systems

Conference, Amsterdam, 1999

30. Lin, F., Picard, R. W., Periodicity, directionality, and randomness: Wold features for image modelling and retrieval, IEEE Transactions on Pattern Analysis and Machine Intelligence, 1996

31. Mehrotra, R., Gary, J., Feature-index-based similar shape retrieval, Proc. of Conference on Visual Database Systems, 1995

32. MPEG-7 standard: GMD page, last checked 2002-06-11, http://ipsi.fhg.de/delite/Projects/MPEG7

33. MPEG-7 standard: working papers, last checked 2002-06-11, http://mpeg.telecomitalialab.com/working_documents.htm#MPEG-7

34. MRML website, last checked 2002-06-11, http://www.mrml.net

35. Nastar, C., Mitschke, M., Meilhac, C., Efficient Query Refinement for Image Retrieval, Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1998

36. Oomoto, E., Tanaka, K., OVID: design and implementation of a video-object database system, IEEE Transactions on Knowledge and Data Engineering, 1993

37. Ortega, M., Yong, R., Chakrabarti, K., Porkaew, K., Mehrotra, S., Huang, T.S., Supporting ranked boolean similarity queries in MARS, IEEE Transactions on Knowledge and Data Engineering, 10/6, 1998, pp. 905-925

38. Payne, J. S., Hepplewhite, L., Stonham, T. J., Evaluating content-based image retrieval techniques using perceptually based metrics, SPIE Proceedings, Vol. 3647, 1999, pp. 122-133

39. Pentland, A., Picard, R. W., Sclaroff, S., Photobook: Content-Based Manipulation of Image Databases, SPIE Storage and Retrieval Image and Video Databases II, 1994

40. Robertson, G., Czerwinski, M., Larson, K., Data Mountain: Using Spatial Memory for Document Management, Proc. of ACM Symposium on User Interface Software and Technology, San Francisco, USA, 1997, pp. 153-162

41. Santini, S., Jain, R., Beyond Query By Example, ACM Multimedia, 1998

42. Santini, S., Jain, R., Similarity Matching, IEEE Transactions on Pattern Analysis and Machine Intelligence, 1999

43. Santini, S., Jain, R., Integrated browsing and querying for image databases, IEEE Multimedia, 2000, Nr. 7, Vol. 3, pp. 26-39

44. Sheikholeslami, G., Chang, W., Zhang, A., Semantic Clustering and Querying on Heterogeneous Features for Visual Data, ACM Multimedia, 1998

45. Smith, J. R., Chang, S., VisualSEEk: a fully automated content-based image query system, ACM Multimedia, 1996

46. Sonka, M., Hlavac, V., Boyle, R., Image Processing, Analysis and Machine Vision, 2nd edition, Brooks/Cole Publishing, Pacific Grove, 1999

47. Tavanti, M., Lind, M., 2D vs. 3D, implications on spatial memory, Proc. IEEE Symposium on Information Visualization, San Diego, USA, 2001, pp. 139-145

48. Wang, H., Guo, F., Feng, D. D., Jin, J. S., A Signature for Content-based Image Retrieval Using a Geometrical Transform, ACM Multimedia, 1998

49. Wood, M., Campbell, N., Thomas, B., Iterative Refinement by Relevance Feedback in Content-Based Digital Image Retrieval, ACM Multimedia, 1998

50. Wu, J. K., Lam, C. P., Mehtre, B. M., Gao, Y. J., Desai Narasimhalu, A., Content-Based Retrieval for Trademark Registration, Multimedia Tools and Applications, No. 3, Vol. 3, 1996, pp. 245-267