Towards a Universal Content-Based Image and Video Retrieval System

Michael Nölle, Wolfgang Herzner
Austrian Research Centers Seibersdorf (ARCS)
A-2444 Seibersdorf, Austria
Phone: +43 (0) 50550-3123/3121, Fax: +43 (0) 2254-74060
e-mail: michael.noelle@arcs.ac.at, wolfgang.herzner@arcs.ac.at

Erwin Laure
University of Vienna, Institute for Software Science
Liechtensteinstrasse 22, A-1090 Vienna, Austria
Phone: +43 1 4277-38812, Fax: (-9388), e-mail: erwin@par.univie.ac.at

Sven Siggelkow
Albert-Ludwigs-Universität Freiburg, Computer Science Department
Georges-Köhler-Allee 52, D-79085 Freiburg, Germany
Phone: +49-(0)761-203-8215 Fax: (-8262), e-mail: siggelkow@informatik.uni-freiburg.de

Bernd Wender
SDS Software Daten Service Ges.m.b.H.
Handelskai 94-96 (Millenniumstower), A-1200 Vienna, Austria
Phone: +43 1 24056 3993, e-mail: Bernd.Wender@sds.at

Abstract: *Content-based retrieval allows to find information by searching about its content rather than about attributes like keywords. Based on the current state of content-based image and video retrieval, this paper addresses this problem in distributed environments with large, heterogeneous, and dynamically changing data repositories. Besides motivation, it describes an appropriate system architecture, and outlines an automatic and generic retrieval approach based on invariants.*

Keywords. Content-based retrieval, image retrieval, distributed environments, dynamic data repositories, invariant image transformations.

1. Introduction

1.1 Content-based Retrieval

"Content-based retrieval" denotes techniques for finding a certain information (in large data volumes like libraries) by searching about its content rather than about attributes like keywords. With respect to textual data, obviously this became quite common. Whether you are searching in a publication database, the web, or in the document you are currently working on with your word processor: it will always be possible to specify words, phrases, or combinations of several of them, which shall be found in – or possibly excluded from – the (available) content of the set of candidate items. More advanced systems will even allow to search for variations (e.g. due to declination or flexion) of the given words, or even for synonyms.

The reason for the availability of content-based retrieval of textual information is that the most developed query format in information technology is the textual format. This again is caused by the fact that searching shorter strings – i.e. query terms like "universe" – in longer strings – i.e. candidate documents – is trivial. In addition, more advanced text retrieval techniques have been at the focus of research for a long time (e.g. [PfPe97]).

However, when it comes to other content categories like visual ones, e.g. graphics[1], im-

The work described in this paper was partially supported by the *German Science Foundation (DFG)* and the Special Research Program SFB F011 "AURORA" of the *Austrian Science Fund*.

[1] In the context of this paper, the term 'graphics' shall denote 2D- or 3D-scenes composed of graphical primitives like lines or B-splines.

ages[2], or videos[3], this no longer holds: it is (still) not common practice to search for visual information by content. Instead, search is usually applied to associated textual attributes like title, originator, or keywords. One can say: search about objects of one kind of media is surrogated by a search about objects of another kind of media.

The reason for the limited availability of content-based retrieval of non-textual information is on the one hand its complexity and on the other hand the lack of advanced algorithms that compute meaningful (as compared to the human sense of similarity) matches between data items with very general contents: although for humans it appears simple to recognize even extreme complex objects like individual faces within a glimpse, we are far away from understanding how much simpler perception tasks – e.g. distinguishing cars from pedestrians in traffic environments – can be implemented in a highly robust (i.e. independent of weather and light conditions) and fast (or better: cheap) way. In fact, we would already be happy if we would have some techniques at hand which would allow to search images for "visual primitives" as efficiently and robust as words in textual documents.

Although searching for non-textual information via textual attributes is commonly used with considerable success, it has its limits: on the one hand, creation of substantial keywords is expensive, since in most cases performed manually, and may not be applicable at all if the data production rate is too high; and on the other hand, several aspects may not be well expressible at all in textual form (which may be even more valid for acoustic information than for visual one).

1.2 Current Situation

Consequently, the quest for methods allowing to retrieve audio-visual information by content is underway for at least one decade, and several results have been achieved in the meantime. A wide variety of models have been considered for image retrieval, from simple early vision techniques, like colour histograms, to rather sophisticated methods, applicable for specific and restricted environments. There are also commercial systems, e.g. [Exca,Flic+95,Vira], as well as research approaches available, [SQUID, Simba]. The type of retrieval relevant features, computed therewith, can be classified into the following categories: colour with/without spatial constraints [Flic+95, PaZa99, SmCh96, Smit97, StDi96, Webe], texture [Dima99,Flic+95,LiPi95,Webe], shape [Flic+95, KlRi98], combinations of colour/texture/shape [Cars+99, DeBe99, Krey+97 ,Pica+96, Vira], motion parameters, inline text (e.g. in TV news), and simple semantics [DeBe99, Flic+95, Vira].

For short, the current situation could be characterized as follows:

- There exist good results for 'static' data sets of 'limited size' and 'good quality'.
  - 'Static' means that the data set does not change or only in a quite limited amount, e.g. only occasional additions of items and no or difficult removal of items.
  - 'Limited size' means several hundredth up to some thousands of individual items (e.g. images) in the data set.
  - 'Good quality' means both relative high resolution – e.g. at least 512x512 – and low noise – e.g. uncompressed or only loss-free compressed encoding.
- Most computational effort is performed prior to query. That means that at some 'preparation time', so-called *features* are computed for the complete data set, and collected in some sort of *index*. Examples for (low level) features are color histograms or texture elements. Then, queries are actually performed against this index rather than against the data set itself.
- Sometimes, feature extraction is interactively supported, e.g. for segmentation.

Examples for such systems are SQUID [SQUID], QBIC [Flic+95], or SIMBA [Simba] for images, and possibly [LoMo95] for graphics. Figure 1 illustrates the potential of modern content-based image retrieval systems like SIMBA. (The concept of invariant features is discussed in more detail in chapter 3.)

1.3 Generalization Requirements

In application environments, where at least one of the first two conditions (i.e. 'static' and 'limited size') does not hold, the current approach will not work satisfyingly. Examples for such environments are

[2] In the context of this paper, the term 'image' shall denote regular 2D-arrays of pixels.

[3] In the context of this paper, the term 'video' shall denote sequences of images.

- ***The Internet***. As the largest, perfectly distributed, and highly dynamic "data pool", it requires adequate techniques if content-based retrieval shall be applicable to non-textual information at all.
- ***Press agencies.*** Press photographs and videos are repeatedly added to their digital data pools, while older items might be removed in similar frequency.

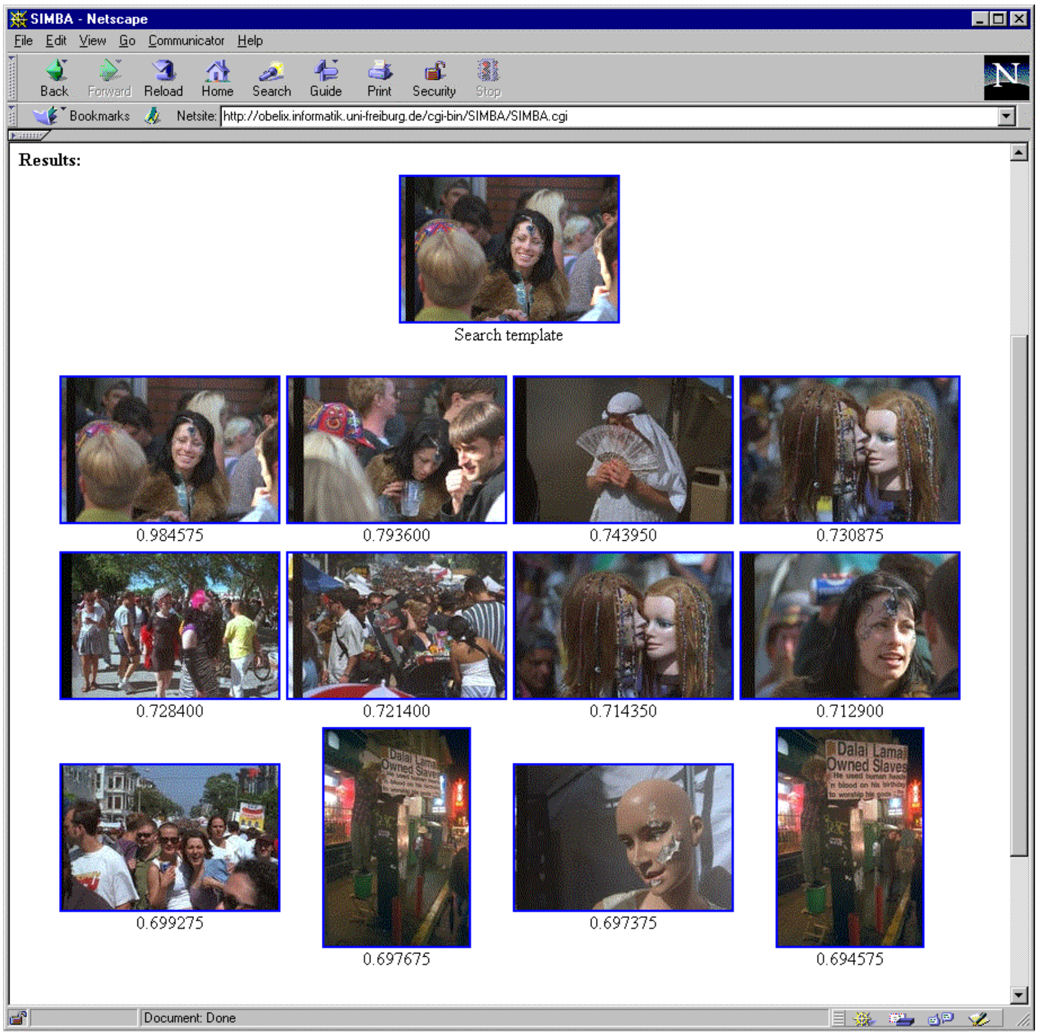

Figure 1 Example from the SIMBA image retrieval system of the University of Freiburg based on the MPEG-7 test set of 2343 images, and using invariant features for retrieval.[4]

- ***Surveillance and monitoring systems.*** Images or videos are continuously generated and stored, replacing older items. Retrieval occurs only sparsely in general, but then it should be as fast as possible.

Taking into account applications like these, as well as the fact that feature extraction is still extremely computational expensive (and will be for the next years), a list of basic requirements for approaches of general content-based retrieval of visual information will comprise at least the following:

---

[4] We acknowledge Tristan Savatier, Alejandro Jaimes and the Deptartment of Water Resources, California for providing the displayed images within the MPEG-7 test set.

- It must be possible to handle a conceptually infinite set of data items, distributed over several data pools.
- It must be possible to pre-compute as many features as possible,
- but also to keep any store(s) of pre-computed features aligned with any changes to the data pool(s).
- Feature computation (*indexing*) should be as automatic as possible,
- and it should be as robust and efficient as possible.
- If available, other sources of information, like annotations, should be included.
- It should be possible to add or remove data pools during runtime.
  (Consider the web: servers may enter or leave at any time.)
- It should be possible to add or replace feature extraction methods, without loosing the ability to exploit already computed features.

Further requirements, like safety and security aspects, and the ability to handle various formats, will not be explicitly addressed here, but, of course, have to be kept in mind.

In the next chapter, a system architecture will be presented, which addresses all of the requirements listed above. Chapter 3 will then discuss how fully automatic feature extraction could be realized with the concept of invariants. A conclusion in chapter 4 will summarize the paper and give a short outlook.

Although the approach described in this paper is intentional, the basic components of the design are not as they are based on results and experience in different fields of research of the authors.

2. System Architecture

Coming from various disciplines like pattern recognition, parallel processing, and video surveillance, the authors elaborated a general system architecture meeting all requirements listed above, and called it GISE (Global Image Search Environment), since it is primarily dedicated for image retrieval. Figure 2 depicts the global architecture of GISE, together with its main components:

- The GISE Kernel System, GKS, which includes
  - a Feature Extraction Engine, Feature Caches and a Matching Engine,
  - a Distributed Computing Framework that connects GISE Servers through a Communication Engine, pre-processes queries in a Query Translator and provides supporting tools for security, routing and accounting, and
- GISE External Interfaces consisting of
  - Graphical, web-based user interfaces (GUIs), and
  - Resource Management Engines which connect the Image Data Pools and the computer resources to the GKS.

The GISE Server, comprising the Feature Extraction Engine, the Feature Caches, the Matching Engine and the Resource Management Engine, combines the software that is operating at content provider sites.

In the following, these components are discussed in some detail.

2.1 User Interface

The user interacts with GISE via a graphical user interface, which usually will be a web-based browser. This interface allows the formulation of complex queries by means of a query language, which, however, is not a hard to learn formal grammar. Instead, the user may select several constraints such as locality or image content, the latter by inputting/selecting some sample image, or by drawing a sketch. Moreover, the user may select the methods which (s)he regards as most appropriate (from a set of available algorithms), and is also able to fine-tune these algorithms by specifying certain parameters. Based on the user's choices, the input data is processed locally on the user site such that the features necessary to handle the query are extracted.

Queries may combine a set of images, but in some cases only a particular region of a given image is of interest to the user, either because it corresponds to the sort of object the user is looking for, or because it contains a color or texture that is of interest to the user. Hence, the User Interface requires some interactive tools for region segmentation [Beuc94,Falc+98,Zano+99], so that the user can identify regions within sample images, e.g. (parts of) a certain building, which to search for.

The User Interface shall also allow annotation and/or (textual) attribute based queries. Hence, the system can take advantage of already annotated or classified data repositories by performing an annotation search in addition to a content-based search. Thereby a focused search in annotated or classified repositories such as catalogues is possible.

In general a query will result into a stream of items rather than a fixed set. As it might take a long time until a query is processed within the entire system, preliminary results shall be delivered as early as possible. These first results possibly do not contain the best overall matches but will be enhanced over time as new results are computed. Thereby, the user is able to quit a query if either satisfying results are among the answers gotten so far or the answers do not satisfy the expectation and the query has to be reformulated. Depending on the systems ability to visualize search results, they may be presented to the user as textual list (e.g. of textual attributes like source and creation date/time), as series of thumbnails, or

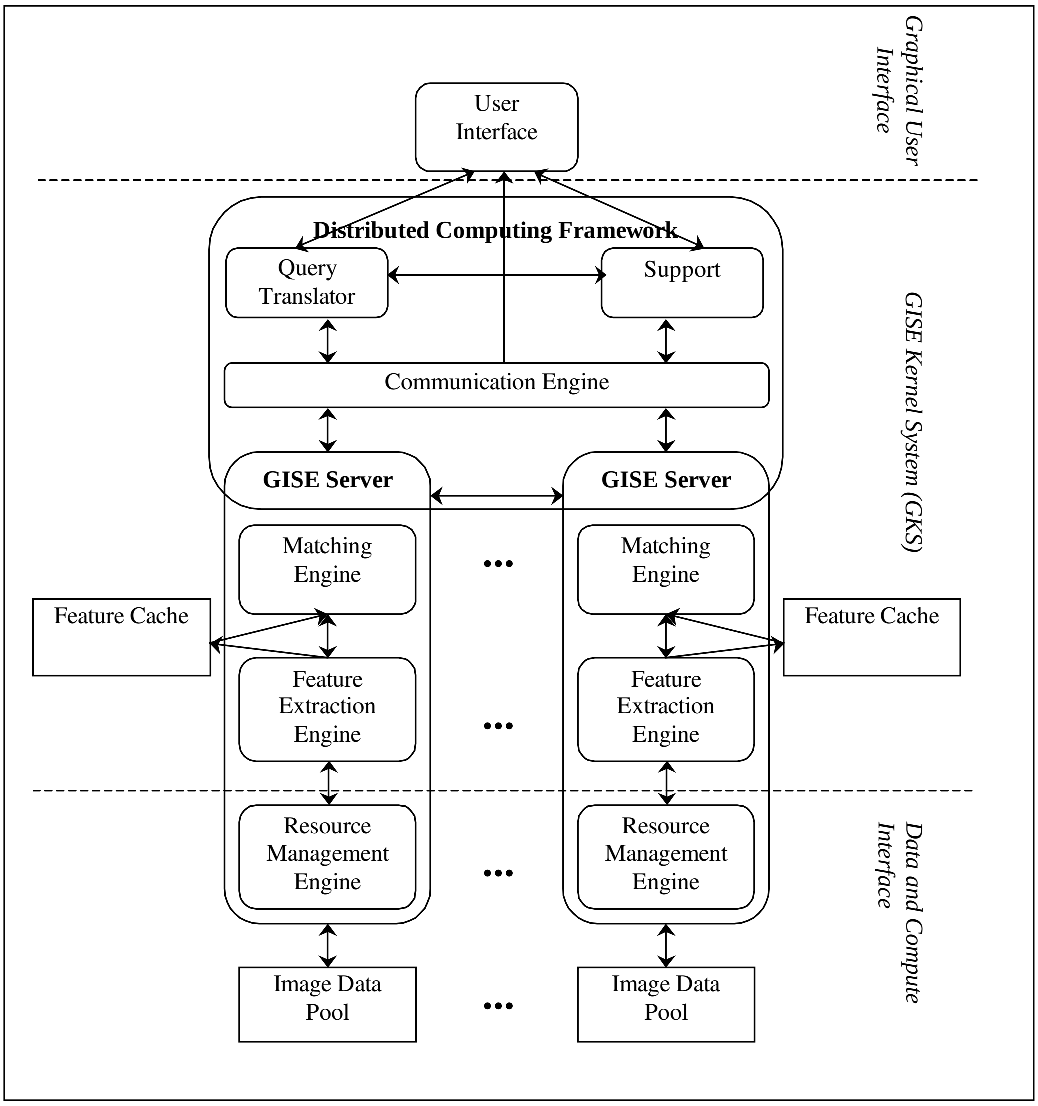

Figure 2 The GISE Architecture

even in a stream of full images. Of course, the latter only under consideration of potential copyright/security constraints, which may cause that images are provided in reduced resolution and/or quality with respect to the stored original.

It is possible to restrict queries to the set of already found images, thus reducing their number further until a satisfying small set of images is left, which then should match the expectations best. In fact, iterative query refinement can be considered as an important feature of GISE: at begin, a rough but computational inexpensive query like with a coarse color histogram selects a first set of potential results. Then, increasingly complex queries are applied to increasingly smaller result sets, thus keeping the overall computational effort in acceptable limits.

2.2 Distributed Computing Framework

Comprising Query Translators, the Communication Engine, and Support Systems, the Distributed Computing Environment enables GISE to cope with following constraints:

- *Decentralized Organization*: Conceptually, there is no central controller in the system. A node in the system is only aware of a number of neighbouring nodes similar to the current Internet organization structure.
- *Routing and Fault Tolerance*: Due to the decentralized organization of GISE, a node posting a query is not aware of all the nodes which will work on the problem. Every node propagates a query only to its known neighbours, which will distribute the query to their neighbouring nodes in turn, and so on. Thus, a query is spread over the whole system in a spanning graph like manner. A *query graph* consists of all the nodes participating in processing a query. Hence, an intelligent routing mechanism is needed which transfers results from any node back to the poster. This routing mechanism not only has to avoid bottlenecks but should also be fault tolerant to some extent such that a failure of a node does not prevent the results of its children from reaching the poster.
- *Security*: In many applications, not only on the Internet, security plays a central role, in particular if sensitive data are involved. More specifically, means for user authentication and data classification are needed. Moreover, data providers must be able to control and restrict accesses to their data in a dynamic way in order to enforce copyright protections etc.
- *Dynamic system configuration*: Resource Management Engines can join or leave the system at any time, even during the ongoing evaluation of a query.
- *Off-the-shelf* Systems: A wide spread use of GISE can only be obtained if easily available off-the-shelf software systems are involved. The use of customized systems which require specialized compilation and runtime-support that is not widely available is often a major obstacle for the acceptance of a system.

A realization of the Distributed Computing Framework will leverage existing distributed- and meta-computing technologies such as the Java programming language [Java] and the Globus toolkit [glob97]. The *Grid Computing Environments Working Group* [grid] provides extensive information on Grid technologies already existing or under development.

2.2.1 Query Translator

The Query Translator normalizes textually and graphically specified informal user queries and translates them into formalized queries which will be further processed by the matching engines. Moreover, queries are optimized (i.e. redundancies in queries are eliminated) and encoded such that Matching Engines are able to process queries efficiently. Apart from processing queries also the input image(s) are processed (i.e. their features are extracted). All this work is executed on the user's site.

The Query Translator triggers the GISE Servers via the Communication Engine.

2.2.2 Communication Engine

In applications with more than one GISE server, the Communication Engine interconnects GISE Servers with their neighbours. It provides high level communication mechanisms for both propagating queries down and sending results up the query graph. Efficient communication of possibly distributed data will be facilitated by similar means as described in [Laur+99]. The Communication Engine hides heterogeneity in physical interconnection networks and provides failure recovery mechanisms which allow a propagation of results even if some nodes do not work properly anymore.

2.2.3 Support Systems

Support Systems provide additional functionality needed by the other components like for security, authentication and accounting. In addition, routing services support the Distributed Computing Framework by its task to propagate queries from User Interfaces to a GISE Server (and results the opposite way), as well as GISE Servers to communicate with each other.

2.3 GISE Servers

GISE Servers, which are the central part of the architecture, are located on the data provider's site and are composed of the matching, feature extraction and resource management engines as well as of a feature cache. Their task is not only the evaluation of queries by triggering the local matching engine but also the propagation of queries to neighbouring nodes and the condensation[5] of results which are sent back towards the originator of the query. Every server has access to a "road map" in which neighbouring nodes are registered. Based on this road map, a GISE Server propagates queries to its neighbours ensuring that the query pervades the system. An incoming query is processed as follows:

- The server first validates the query with respect to security constraints that may be applicable and eliminates duplicate queries.
- The query is then propagated to the nodes registered in the road map. This propagation may be restricted with respect to criteria such as locality, data content constraints, etc. The server may control its load by deciding how to contribute to the condensation of results. In particular, a server may refuse to receive results from its neighbours; in that case results are bypassed to other nodes in the query graph.
- The local matching engine is triggered to process the query locally.
- The local results from the matching engine and potential remote results from neighbours are condensed depending on a customisable optimality criterion and the (locally) best results are sent back to a server further up in the query graph as soon as first results are available.

2.3.1 Matching Engine

Data features are compared (i.e. with those provided in queries) by the Matching Engine (ME) which does not work on the Image Data Pool directly but on the Feature Cache. It is a distinct feature of GISE that all evaluation takes place on the Feature Cache instead of the Image Data Pool itself. This enhances the flexibility of the system and provides means for respecting access restrictions, copyright procedures, etc. as imposed by data providers. If features are required which are not already computed for a data pool, the Feature Extraction Engine is triggered to provide these features on the fly. If necessary additional parameters or new feature extraction functions coming along with the current query will be passed to the Feature Extraction Engine.

2.3.2 Feature Cache

The features generated by the Feature Extraction Engine are stored in the Feature Cache. Since feature extraction is a time consuming task, commonly used features are pre-computed and stored in the Feature Cache. Thus, only those features which are not yet computed but needed have to be generated during the evaluation of a query on the fly. Depending on the available storage space, both pre- and on-the-fly computed features are kept in the Feature Cache until either some have to be deleted due to space constraints, or may be deleted due to erasure of the associated items from the Image Data Pool. The latter has to be triggered by the Resource Management Engine.

Feature deletion due to space constraints will be guided by an importance model, which considers age of the associated image, the usage statistics of the feature, the amount of memory occupied by it, as well as the price for its re-computation.

The presentation or encoding, respectively, of features will rely on or at least consider MPEG-7 [MPEG7, Sigg+99].

2.3.3 Feature Extraction Engine

The Feature Extraction Engine (FEE) provides both the pre-calculated features like color histograms and on-the-fly computed features for images provided to it by the Resource Management Engine, and adds them to the Feature Cache.

According to the computational cost of the feature extraction task, parallelism should be exploited wherever possible. In particular, fine grained data parallelism such as those approaches developed in [NSSM94,Nöll+96,Nöll96] can be intermixed with coarse grained data and task parallelism on the image level. Previous work on hybrid parallelism in heterogeneous environments as documented in [Chap+97,Laur+99,Laur01] should be considered.

The FEE is triggered by both, the Resource Management Engine (e.g. on arrival of new images) and the ME.

[5] 'Condensation' means merging of results by keeping the (locally) best ones. Distribution of condensation allows for early removal of relatively poor results, and thus helps to reduce the size of the resulting sets of items as well as the response traffic in general.

2.3.4 Resource Management Engine

The Resource Management Engine (RME) provides the FEE with a uniform interface to heterogeneous data pools enabling dedicated data base functionality and converts data from the original storage format to the data format required by the FEE, if needed. The interface to the FEE will, amongst others, cover the following aspects:

- Connect/disconnect: RMEs should be able to connect to and disconnect from GISE.
- Image/video identification; to safely match features stored in Feature Caches with the images/videos they belong to, and to send the right data to the user.
- Data pool updates; to inform the FEE about new or deleted objects in data pool.
- Image properties; to inform the FEE about (the availability of) special attributes like creation date, source, owner, encoding format etc. .
- Query session support; e.g. to prepare images/videos which match a query for efficient sending to user. This includes data compression in order to reduce network traffic, as well as consideration of copyrights, as already addressed shortly in 2.1.
- Access control; e.g. to exclude images/videos from a query based on user identification and access rights.

The RME will allow GISE only to retrieve images (videos) from external data repositories, but not to modify them. However, it is responsible to keep GISE, and particularly the feature cache, consistent with the data repository.

While the interface to the FEE is fixed, the interface to the Image Data Pool has to be customised for every pool to meet its particular organisation requirements.

2.4 Image Data Pool

The Image Data Pool contains all images for which features have been extracted and stored in the Feature Cache, in an organisation and format known to and possibly controlled by the RME, but outside the scope of GISE. The Image Data Pool may contain items not known to GISE.

3. Automatic Feature Extraction

The feature extraction methods used will be a crucial factor for the performance of any content-based retrieval system. In this section we want to review briefly the main trends found in the literature and then give an exemplary approach. Following the arguments outlined in Section 1 we would like to stress that it is most desirable to be able to add new algorithms or recombine the existing ones. Thereby, we can adapt the response of the system better to the expectation of the user.

3.1 Existing approaches

The approaches considered in the community for content-based image retrieval can be classified as one of these three types: manual extraction of attributes, automatic feature extraction and object recognition and a combination of these techniques.

Manual systems require too much manpower taking into account the amount of image data available nowadays. Additionally, new image data is faster produced than annotations can be added; finally annotations reflect a subjective rating.

Semi-automatic systems provide tools to accelerate the annotation process; however, they require manual interaction for database generation. The user might manually (probably computer assisted) segment objects in the image, followed by an automatic analysis and annotation by the computer [Flic+95]. Such systems considerably accelerate the image annotation.

Both *manual* and *semi-automatic* systems can only work offline, i.e., a database has to be generated before queries can be provided. Hence, they are not well suited for heavily fluctuating data, like, e.g., a collection of Internet sites or video surveillance systems.

Fully *automatic* systems overcome these problems since the analysis can be done at query time, if the data set has not been analysed before. Nonetheless, for unrestricted image retrieval problems, there is still a large gap between objective image features, which have been extracted, and the semantics of the image. As an example assume we have an algorithm that compares images and gives the same result independent of the rotational position of a given image. By this algorithm all three images in Figure 3 will be the same. The human observer will clearly distinguish between the princess in the first and the old lady in the last picture. But, (s)he might be puzzled by the middle one. Whether the middle picture looks more like the first or the last depends on the direction of the approach. Obviously our understanding of what we see influences the way how we see.

There are currently two ways to circumvent this problem: The first is to give a precise definition which similarity model(s) the system considers and provide this information to the users, so that they can choose which fits the individual needs best. The second possibility is to hide the similarity models from the user and to learn from the feedback (e.g., marking good and bad) [Pica+96,SMMR99]. However, there might not always exist a general solution. The most natural way to circumvent this problem might be to help the user to adjust the way the query is formulated to the expectation of the result (s)he might have.

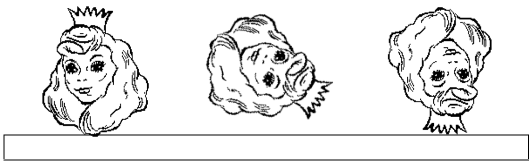

Figure 3 Three images or one image in three rotations. An algorithm we design would most likely assume all three are similar. Would you? Is the middle picture more princess or old lady? What happens if you turn the page upside down?

3.2 Feature Extraction by invariants

The feature extraction algorithms used within the framework of GISE have to be dynamically changeable. Thereby, we rely on the ability of the used software environment to either load code dynamically at runtime or to interpret incoming (new) functions. Nonetheless, there should be a basic set of feature extraction algorithms implemented in the FEE, which satisfy the basic demands.

From the above discussions it becomes evident that any approach towards image indexing working without restrictions on the type of repository items should use the original data itself. As soon as we know we want to analyse a specialised category of images like comics, paintings, pictures of faces, etc. we might develop adopted techniques. I.e. if the images we are looking for are composed out of straight lines, we will look for straight lines. But if we are interested in circles, texture, etc., a technique based on straight lines presumably will fail. Therefore, in this paper we will concentrate on those methods that do not restrict the general context and will work on any type of visual data.

As we deal with possibly very large data repositories, we have to apply computational 'cheap' algorithms in order to pre-select good candidates from the repository. On these we may continue the selecting process by using more sophisticated methods that produce accurate matches with the given search template(s).

Some types of so-called invariant transformations seem to be good candidates for our purpose. We will review the main ideas behind these algorithms and derivations thereof. Thereby, we will briefly recall basic properties of invariants as well as a technique called group averaging that allows a generic generation of invariant transformations. Inspired by some results from [Burk+01], we will introduce feature histograms as 'cheap' pre-selection mechanism.

3.3 Invariant Transformations

Frequently the generation of image features is based on some well known (but still under heavy investigation) techniques using so-called invariant transformations.

Given a n-dimensional pattern-space $\mathbf{M} : \mathbf{m}(x_1, \ldots, x_n) \in \mathbf{M}$, (short $\boldsymbol{m(x)}$, $x \in I$) and a set of transformations T: $\boldsymbol{M} \rightarrow \boldsymbol{M}$, a function F will be called *invariant* on the pattern-space $\boldsymbol{M}$ with respect to T if

$$\forall\, \mathbf{m} \in \mathbf{M}, t \in T : F(\mathbf{m}) = F(t \circ \mathbf{m}). \qquad (3\text{-}1)$$

Informally an invariant function will give the same result when applied to a pattern or any of its transformations. Usually the invariant function F will be complex valued. If F has the additional property that it will produce the same result if and only if it is applied to patterns that are connected through a transformation, i.e.

$$\begin{aligned} &\forall\, \mathbf{m}, \mathbf{m}' \in \mathbf{M} : F(\mathbf{m}) = F(\mathbf{m}') \\ &\quad \Leftrightarrow \exists\ t \in T : \mathbf{m}' = (t \circ \mathbf{m}), \end{aligned} \qquad (3\text{-}2)$$

then function F is called *complete*.

Although not strictly necessary, we will restrain the further discussion on finite dimensional pattern spaces with a finite range (R) and (mainly) finite transformation sets. Furthermore, the transformations will be taken from sets that form algebraic groups. This is not required by the definitions above, but appropriate for many practical cases. Typical examples are the Euclidean motion group, which covers translation and rotation, affine or perspective mappings. In this paper we will mainly restrict the transformation group to Euclidean motion.

Simple but important examples of invariant transformations are

1. the global sum :

$$S(\mathbf{m}) = \sum_{x \in I} \mathbf{m}(x) , \qquad (3\text{-}3)$$

2. the histogram:

$$H(\mathbf{m}) = (h_1, \ldots, h_l) \text{ where}$$
$$h_i = \left| \left\{ b_{i-1} < \mathbf{m}(x) \leq b_i \right\} \right|, \qquad (3\text{-}4)$$
$$\text{and } \min(R) < b_1 < \cdots < b_l < \max(R)$$

h_i usually is called a *bin*. If we consider the symmetric group of all permutations over the index-space I as transformation group then the histogram will be complete, whereas the global sum will not[6]. In the sequel we will denote this group of all permutations of N elements as Ψ_N .

Nonetheless, the symmetric group is of little interest when it comes to images, since arbitrarily permuted image points in general will not look very alike. The Euclidean motion or rigid body motion is more suited to describe some transformations that images can undergo without changing their content. Those transformation leave the local neighbourhood around every point (pixel) in the image intact, i.e. points that are close to each other before the transformation will also be close afterwards. In fact, the distance of any two points will be an invariant with respect to the Euclidean motion transformation.

For transformations taken from algebraic groups a systematic approach, the so-called group average, to generate invariant functions is given in [SchM94a,SchM94b,SchM95], which, if necessary, can be complete. We will briefly describe this approach and some derivations of it used in the SIMBA system of the University of Freiburg [Simba]. Let $\mathbf{m} = \{ \mathbf{m}(i, j),\ 0 \leq i < N,\ 0 \leq j < M \}$ be a two dimensional intensity (or grey scale) image and G a transformation group acting on images. For $g \in G$ the action of g on $\boldsymbol{m}$ will be denoted $g\boldsymbol{m}$. For the Euclidean motion group this yields

$$(g\mathbf{m})(i, j) = \mathbf{m}(g(i, j)) = \mathbf{m}(k, l), \text{ with}$$
$$\begin{pmatrix} k \\ l \end{pmatrix} = \begin{pmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{pmatrix} \begin{pmatrix} i \\ j \end{pmatrix} - \begin{pmatrix} t_0 \\ t_1 \end{pmatrix}. \qquad (3\text{-}5)$$

The indices have to be read modulo N respectively M. We can split the Euclidean motion into a rotational ($\varphi \in R$) and a translation part ($t \in T$).

[6] Simply consider that the sum of two numbers might be identical although the numbers are not, i.e. 3+5=2+6.

Given the pattern space and the transformation group, we can construct an invariant function F by choosing an arbitrary function f and averaging over the action of the group:

$$F(\mathbf{m}) = \frac{1}{|G|} \int_{g \in G} f(g\mathbf{m})\, dg \qquad (3\text{-}6)$$

Informally we apply every element g of the group G on *f(m)* and calculate the integral. From the group properties of G it is easy to see that *F(m)* = *F(gm)* for every $g \in G$. The set $G\boldsymbol{m} = \{ g\boldsymbol{m}, g \in G \}$ is sometimes called the *orbit* of G on m. For the Euclidean motion this yields

$$F(\mathbf{m}) = \frac{1}{|T|} \frac{1}{|R|} \iint_{t \in T\ \varphi \in R} f(g(t, \varphi)\mathbf{m})\, d\varphi\, dt\ . \qquad (3\text{-}7)$$

And in the discrete case we get[7]:

$$F(\mathbf{m}) = \frac{1}{|T|} \frac{1}{|R|} \sum_{t \in T} \sum_{\varphi \in R} f(g(t, \varphi)\mathbf{m})\ . \qquad (3\text{-}8)$$

3.3.1 The choice of kernel functions

As explained before, so-called kernel function f can be chosen arbitrarily. If we choose *f*(***m***) = ***m***(0,0) we get the *weighted global sum* as given in Equation (3-3). In order to get some insight into the meaning of Equation (3-7) and to show how it can be effectively computed, we will decompose it.

Assume the local function *f*(***m***) = ***m***(0, -*r*) ***m***(0, *r*). f multiplies two values that can be interpreted as being at opposite positions of a circle with radius r. We then have to integrate over all φ followed by the integration over all translations t = (x_t y_t) (see Figure 4). The translations basically move the circle over the entire image. Informally, f can be thought as a coding function that preserves some local information of the pattern before averaging. From this it becomes evident that f may generically be described by

1. the neighbourhood pattern involved to select the operands, i.e. (0,-r), (0,*r*), and
2. the operation required, i.e. the multiplication in the above example.

In general f will be non linear. Note that the integration over all angles φ results in an invariant

[7] Strictly speaking, Equation (3-8) is only valid for multiples of 90° rotations and $t \in \mathbf{Z}_{M \times N}$. Any other rotation respectively translation will not map pixels onto pixels. From a practical point of view we will have to interpolate between pixel values whenever we apply non grid conform transformations.

not for the entire image but for the co-ordinate position (x_t ,y_t).

The choice of the coding function *f* naturally is most decisive for the performance of the method. In the literature we will find mainly monomials of small degree. As shown in [SchM95], this approach leads to completeness if the neighbourhood involved and the degree of the monomial become large. Especially in the presence of errors the latter requirement will lead to extremely sensitive invariant results. Both requirements result in somehow expensive computations as the number of operations necessary for every evaluation of *f* has to be multiplied by the number of elements per image. Nonetheless, it was shown in [SchM95] that for a given not too big set of objects that shall be discriminated, the invariants computed by Equation (3-7) work well using monomials of small degree (< 4).

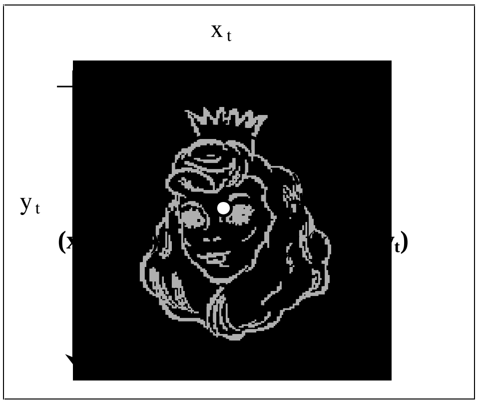

Figure 4 The function *f* can be described by the neighbourhood involved selecting the operands and the operations acting thereon.

Another choice of *f* is to use a relational operator as done in [Scha01]. As an example we get $f(\boldsymbol{m}) = \boldsymbol{m}(0, -r) \geq \boldsymbol{m}(0, r)$, which again has to be integrated over all rotational angles φ and all translations *t*. Thereby, the kernel function becomes invariant against monotonic intensity changes within the image.

Which kernel function meets the requirements within a given application 'best' is still an open issue which needs further investigations. With no doubt the type of application as well as what 'best' might mean will influence the choice of *f*. At the moment this choice has to be left as the 'art of algorithmic design'. But, the method described is generic in terms of the neighbourhood involved as well as by the function used. By using more neighbourhood information we preserve a greater surrounding and code more of the 'structure' of a given image within the result of the kernel function.

3.3.2 Histograms versus group average

A quite new approach based on some parts of the described group averaging technique, but with additional helpful properties, is given in [BuSi00,Burk+01]. Instead of calculating the integral over the group, basically the results of the kernel function are collected into a histogram, thus obtaining an estimate of the feature distribution of the image instead of just the mean of the features. The features computed in the histograms are able to deal with non-cyclic boundary conditions and the problem of detecting sub-images quite well. Furthermore, the traditional histogram was extended to a so-called *fuzzy histogram*, which does not assign a feature value to only one bin but to all nearby bins with different weights (according to the distance to these bins). In contrast to the traditional histogram the fuzzy histogram has the advantage that a small change in the feature value always results in a small change of the histogram (continuity). From Equation (3-4) one can directly see that the continuity is not given at the bin boundaries of the traditional histogram. A feature value of b_i -e is assigned to another bin than a feature value of b_i+e. Thus the fuzzy histogram offers some robustness against local deterioration of quality in the input data. The SIMBA system [Simba] is based on this technique (cf. Figure 1 as an example). What makes the methods additionally very appealing is the fact that there exists a fast Monte Carlo based algorithm to compute the fuzzy histograms. This algorithm is of *constant (!)* complexity [SiSc99], i.e. independent of the image size. The constant given is roughly around 40.000 computations of the kernel function per image. Thereby, this algorithms becomes a perfect candidate to calculate the basic features in order to match the query data against the bulk of information in the data repositories.

Nonetheless, as can be seen in Figure 1, some of the results obtained are similar without a good intuitive reason. This might be caused by the fact that the similarity measure, as given by the numbers below each picture is not very compliant to the intuitive 'feeling' of similarity. As it originates from a distance of quite abstract feature vectors there is no obvious translation. Using this feature similarity there is no direct way to relate the resulting images to the query images.

As mentioned before, we can use invariant result, i.e. the integration over all angles φ to describe the co-ordinate position ((x_t, y_t) in Figure 4).

$$R(\mathbf{m}(x_t, y_t)) = \frac{1}{|R|} \sum_{\varphi \in R} f(g(\varphi)\mathbf{m}(x_t, y_t))$$

If we take the text search as an analogy, the preselected image set may be seen as a number of books that are in some way similar to a given 'search' book. Then, in this analogy, $R(\boldsymbol{m}(x_t, y_t))$ relates to a word in our book or image, respectively. Thereby, we can use these local invariants to compare pixel locations in the image and related them to each other. Some experiments already done in this direction are quite promising and will be pursued in future work.

4. Conclusion

Based on the current state of the art in content-based image or video retrieval, a system architecture has been presented, which could pave the way to search for visual information entities like graphics, images, or videos in open, dynamic, and distributed environments in a form, which today is common for textual information: namely by exploiting content rather than (textual) attributes. In addition, with the concepts of invariants an approach has been presented, that could serve as a base of automatic feature extraction, which is a sine qua non condition for a successful realization of this system architecture called GISE. GISE and invariants are currently under further design, research and implementation, with – potentially amongst others – a surveillance system [Herz+97] as an intended testbed.

Finally, it should be emphasized that an important goal of these activities is to contribute to the development of a system of visual elements comparable to letters in textual data, which will allow to search over visual contents in way analogous to that already quite common in full text retrieval.

References

[Beuc94] S. Beucher. Watershed hierarchical segmentation and waterfall algorithm. *In International Symposium on Mathematical Morphology (ISMM'94)*, pages 69-76, Fontainebleau, France, 1994.

[BuSi00] H. Burkhardt and S. Siggelkow. Invariant features in pattern recognition - fundamentals and applications. In C. Kotropoulos and I. Pitas, editors, Nonlinear Model-Based Image/Video Processing and Analysis. John Wiley & Sons, 2000.

[Burk+01] S. Siggelkow and H. Burkhardt. Fast invariant feature extraction for image retrieval. To appear in H. Burkhardt, H.-P. Kriegel, R. Veltkamp, editors, State-of-the-Art in Content-Based Image and Video Retrieval, Kluwer Computational Imaging and Vision Series, 2001.

[Cars+99] Chad Carson et al. Blobworld: A system for region-based image indexing and retrieval. Submitted to *3rd International Conference on Visual Information Systems*, 1999.

[Chap+97] B. Chapman, M. Haines, P. Mehrotra, J. Van Rosendale, and H. Zima. OPUS: A Coordination Language for Multidisciplinary Applications. Scientific Programming, 6/9:345-362, Winter 1997.

[DeBe99] C.H. Demarty and S. Beucher. Morphological tools for indexing video documents. In Proceedings of the International Conference on Multimedia Computing and Systems. June 1999, Florence, Italy.

[Dima99] A. Dimai. Rotation invariant texture description using general moment invariants and gabor filters. In Proceedings of the 11th Scandinavian Conference on Image Analysis, Kangerlussuaq, Greenland, vol. 1, pages 391-398, June 7-11, 1999.

[DuOl99] A.G. Dugue and A. Oliva. Classification of natural images from distributions of local dominant orientations. In Proceedings of the 11th Scandinavian Conference on Image Analysis, Kangerlussuaq, Greenland, vol. 1, pages 203-211, June 7-11, 1999.

[Exca] Excalibur Visual Retrievalware Demo, http://vrw.excalib.com:8015/cst.

[Falc+98] A.X. Falcao, J.K. Udupa, S. Samarasekera and S. Sharma. User-steered image segmentation paradigms: live wire and live lane. *Graphical Models and Image Processing*, Volume 60, pages 233-260, 1998.

[Flic+95] M. Flickner et al. Query by image and video content: The QBIC system. *IEEE Computer*, 28(9):23-32, September 1995.

[glob97] : I. Foster and C. Kesselman. Globus: A Metacomputing Infrastructure Toolkit. Intl. J. Supercomputer Applications, 11(2):115-128, 1997.

[grid] Grid Computing Environments Working Group, http://www.computingportals.org/

[Herz+97] W. Herzner, M. Kummer, M. Thuswald. DVS - A System for Recording, Archiving and

Retrieval of Digital Video in Security Systems. In: Proceedings of HIM'97 (Hypertext - Information Retrieval - Multimedia '97), Dortmund, Germany, October 1997. Schriften zur Informationswissenschaft, vol. 30; pp.67-80, UVK (Universitätsverlag Konstanz), October 1997

[Java] Java. WWW: http://java.sun.com/.

[KlRi98] M. Kliot and E. Rivlin. Invariant-based shape retrieval in pictorial databases. In *Proceedings of the ECCV 1998*, volume I, pages 491-507, Freiburg, June 1998, Springer.

[Krey+97] J. Kreyß et al. Video retrieval by still image analysis with Imageminer. In *Proceedings of the SPIE*, volume 3022, pages 36-44, 1997.

[Laur+99] E. Laure, P. Mehrotra, and H. Zima. Opus: Heterogeneous Computing With Data Parallel Tasks. Parallel Processing Letters, 9(2):275-289, June 1999.

[Laure01]: E. Laure. OpusJava: A Java Framework for Distributed High Performance Computing. Future Generation Computer Systems, in print, 2001.

[LiPi95] F. Liu and R.W. Picard. Periodicity, directionality, and randomness: Wold features for image modelling and retrieval. *Technical Report TR-320*, MIT Media Lab, 1995.

[LoMo95] O. Lorenz and G. Monagan. Automatic Indexing for Storage and Retrieval of Line Drawings. Proceedings of IS&T/SPIE's Symposium on Electronic Imaging Science & Technology. San Jose Convention Center, San Jose, California, USA. Feb 1995.

[NSSM94] M. Nölle, G. Schreiber, and H. Schulz-Mirbach. PIPS-a general purpose Parallel Image Processing System. In W.G. Kropatsch and H. Bischof, editors, *16. DAGM - Symposium ''Mustererkennung''*, pages 609-623, Wien, September 1994. Series Informatik XPress, TU-Wien

[Nöll96] M. Nölle. Konzepte zur Entwicklung paralleler Algorithmen der digitalen Bildverarbeitung. *PhD thesis*, Technische Universität Hamburg-Harburg, December 1994. Appeared as Fortschritt-Bericht, VDI-Reihe 10, Nr. 410, VDI-Verlag, 1996.

[Nöll+96] M. Nölle and G. Schreiber. Data distribution concepts for parallel image processing. In *Proc. of the 13th International Conference on Pattern Recognition*, volume 4, Vienna, Austria, August 1996. 728-733.

[PaZa99] G. Pass, R. Zabih, Comparing images using joint histograms, Multimedia Systems (1999), in press.

[Pica+96] R.W. Picard et al. Modelling user subjectivity in image libraries. In *Proceedings of ICIP*, Lausanne, September 1996.

[PfPe97] U. Pfeifer and S. Pennekamp. Incremental Processing of Vague Queries in Interactive Retrieval Systems. In Proceedings of HIM'97 (Hypermedia - Information Retrieval - Multimedia 1997), Dortmund, Germany, October 1997. Schriften z. Informationswissenschaft, vol..30; pp.233-236, UVK (Universitätsverlag Konstanz), October 1997

[Scha01] Marc Schael. Invariant grey scale features for texture analysis based on relational kernels and group averaging. Internal report 01-01. University of Freiburg.

[SchM94a] H. Schulz-Mirbach. Algorithms for the Construction of Invariant Features. In W.G. Kropatsch and H. Bischof, editors, *16. DAGM – Symposium "Mustererkennung"*, pp. 324-332, Wien, September 1994, Reihe Informatik Xpress, TU-Wien.

[SchM94b] H. Schulz-Mirbach. Constructing Invariant Features by Averaging Techniques. *In Proceedings of the 12th International Conference on Pattern Recognition*, volume II, pp. 387-390, Jerusalem, October 1994.

[SchM95] H. Schulz-Mirbach. Anwendung von Invarianzprinzipien zur Merkmalgewinnung in der Mustererkennung, Phd Thesis, Reihe 10, Nr.\ 372, VDI-Verlag, February 1995.

[Sigg+99] S. Siggelkow, J. Heuer, H. Burkhardt, A. Kaup. Image retrieval based on local invariant features. *Proposal for MPEG-7 standard*, January 1999, accepted for the MPEG-7 core experiment.

[SiSc99] S. Siggelkow and M. Schael. Fast estimation of invariant features. In W. Förstner, J.M. Buhmann, A. Faber, and P. Faber, editors, Mustererkennung, DAGM 1999, Informatik aktuell, Bonn, Germany, September 1999. Springer.

[SMMR99] D.McG. Squire, W. Müller, H. Müller, and J. Raki. Content-based query of image data-bases, inspirations from text retrieval: inverted files, frequency-based weights and relevance feedback. In Proceedings of the 11th Scandinavian Conference on Image Analysis, Kangerlus-suaq, Greenland, vol. 1, pages 143-150, June 7-11, 1999.

[Simba] 'Search IMages By Appearance' (SIMBA) search engine at the University of Freiburg http://simba.informatik.uni-freiburg.de

[Smit97] J.R. Smith. Integrated Spatial and Feature Image Systems: Retrieval, Analysis and Compression. *PhD thesis*, Columbia University, 1997.

[SmCh96] J. R. Smith and S.-F. Chang. Visualseek: a fully automated content-based image query system. In Proc. ACM Multimedia '96 , 1996.

[Smit97] J.R. Smith. Integrated Spatial and Feature Image Systems: Retrieval, Analysis and Compression. *PhD thesis*, Columbia University, 1997.

[SQUID] University of Surrey, Centre for Vision, Speech, and Signal Processing. Shape Queries Using Image Databases http://www.ee.surrey.ac.uk/Research/VSSP/imagedb/demo.html

[Squi+99] D.McG. Squire, W. Müller, H. Müller, and J. Raki. Content-based query of image databases, inspirations from text retrieval: inverted files, frequency-based weights and relevance feedback. In Proceedings of the 11th Scandinavian Conference on Image Analysis, Kangerlussuaq, Greenland, vol. 1, pages 143-150, June 7-11, 1999.

[StDi96] M. Stricker and A. Dimai. Colour indexing with weak spatial constraints. In *Storage and Retrieval for Image and Video Databases IV*, volume 2670 of SPIE Proceedings Series, pages 29-40, February 1996.

[Virage] Virage Image Engine at Altavista, http://image.altavista.com.

[Weber] R. Weber. Final CHARIOT demonstration at http://mercator.ethz.ch/Chariot/Chariot.html

[Zano+99] M.F. Zanoguera, B. Marcotegui, F. Meyer. An interactive colour image segmentation system. *Workshop on Image Analysis for Multimedia Services*, pages 137-144, Berlin, Germany, 1999.

Biographies

Wolfgang Herzner received the PhD degree in Computer Science at the Vienna University of Technology in 1984. Until 1997, he headed ARCS' working group "Computer Graphics and Multimedia", addressing computer graphics standardisation, multimedia documents and authoring tools. 1994-95 participation in EU-project OMHEGA (frame IV). Since 1996, project management and development of large digital video surveillance and monitoring systems. Since 1984, he is in charge of the working group for Computer Graphics and Multimedia within the Austrian Standards Institute, ON. Herzner is member of the Austrian Computer Graphics Society (OCG), Eurographics, ACM, and reviewer of IEEE Software.

Erwin Laure received his Master's degree in Computer Science and Business Administration in 1996 and will receive his PhD in Computer Science and Business Administration in 2001, both from the University of Vienna. In 1996 he became a research assistant at the Institute for Software Science (former Institute for Software Technology and Parallel Systems) of the University of Vienna. His research interests include programming environments, languages, compilers and runtime systems for parallel and distributed computing.

Michael Nölle received his Diploma degree in Computer Science from the Technical University of Braunschweig, Germany in 1988 and his PhD degree in Electrical Engineering from the Technical University of Hamburg-Harburg, Germany in 1994. In 1996 he joined the Institute for Software Science of the University of Vienna, Austria as a EU research assistant. He became University assistant at the Institute for Pattern Recognition and Image Processing of the University of Freiburg in 1998. Currently he is affiliated with the Austrian Research Centre Seibersdorf. He is member of the Marie Curie Fellowship Association. His main research interest are in the area of image processing and parallel computations and algorithms.

Sven Siggelkow received the diploma degree in computer engineering from the Technical University of Hamburg-Harburg in 1996. He has been affiliated with Philips Semiconductors Product Concept and Application Labs in Hamburg and currently is working towards his PhD at the Institute for Pattern Recognition and Image Processing of Freiburg University. His research interests are image analysis, invariant features, and image retrieval. Additionally he is author of two patents for the segmentation of image sequences in the context of object oriented coding.

Bernd Wender studied computer science at the University of Technology in Vienna. He received his master degree in 1992. From 1993 to 1999 he worked as a research and teaching assistant at the Institute for Software Science (former Institute for Software Technology and Parallel Systems) at the University of Vienna. His research interests focused on compilation and runtime environments for parallel and distributed systems. Since 1999 he is affiliated with a banking software company. He currently works as a systems architect and Java developer in an internet banking project.