

# Parallel Visual Information Retrieval in VizIR

Horst Eidenberger\*

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

## ABSTRACT

This paper describes how parallel retrieval is implemented in the content-based visual information retrieval framework VizIR. Generally, two major use cases for parallelisation exist in visual retrieval systems: distributed querying and simultaneous multi-user querying. Distributed querying includes parallel query execution and querying multiple databases. Content-based querying is a two-step process: transformation of feature space to distance space using distance measures and selection of result set elements from distance space. Parallel distance measurement is implemented by sharing example media and query parameters between querying threads. In VizIR, parallelisation is heavily based on caching strategies. Querying multiple distributed databases is already supported by standard relational database management systems. The most relevant issues here are error handling and minimisation of network bandwidth consumption. Moreover, we describe strategies for distributed similarity measurement and content-based indexing. Simultaneous multi-user querying raises problems such as caching of querying results and usage of relevance feedback and user preferences for query refinement. We propose a 'real' multi-user querying environment that allows users to interact in defining queries and browse through result sets simultaneously. The proposed approach opens an entirely new field of applications for visual information retrieval systems.

**Keywords:** Content-based Visual Information Retrieval, Parallel Query Execution, Peer to Peer Networks, Database Design, Distributed Databases, Reliability

## 1. INTRODUCTION

Content-based retrieval of visual information<sup>1, 17, 19</sup> (VIR) deals with vast amounts of data (media data, feature metadata, etc.). Hence, almost naturally VIR applications are distributed applications: Servers hold media data and metadata in storage system-based databases and provide considerable processing power for query execution. Clients communicate user interaction to the servers and visualise retrieval results in user interfaces. Retrieval is performed simultaneously by multiple clients. Consequently, parallel processing issues have to be considered in VIR system design. Surprisingly, in recent years only few contributions to this field of research can be identified (see Subsection 2.2 for a short overview).

This paper describes, how the various aspects of query parallelisation are tackled in the VIR framework VizIR<sup>8</sup>. VizIR is an open project for the development of a software workbench of VIR tools (see Subsection 2.1 for an introduction and Subsection 3.1 for a brief description of the VizIR querying process). VizIR is based on distributed environments that are, for example, connected by web services or classic remote procedure calls. Therefore, two means of parallelisation have to be dealt with: simultaneous querying of multiple users and querying in distributed databases. Subsection 3.2 lists, which parallelisation issues occur in VizIR regarding to the VizIR querying process. In addition to 'basic' parallelisation problems this paper describes an approach for simultaneous multi-user querying in *one* environment. This approach is called team retrieval. Team retrieval aims at accessing the expert knowledge of groups for the exploitation of media data spaces. Of course, in such a situation visualisation techniques are of highest importance to make parallel interaction explicit.

Similarity measurement in VizIR is a highly interactive process that is heavily based on visualisation techniques. Generally, the most important parallelisation issue in the querying process is accelerating query execution by clever caching of media metadata. By usage of caching algorithms and of sophisticated query distribution techniques, the drawback of parallelisation (complexity) can be turned into an advantage (performance gain by query threading).

The paper is organised as follows. Section 2 gives background information on the VizIR project and on relevant parallelisation issues in VIR systems. Section 3 sketches querying in VizIR and major parallelisation issues. Section 4

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

deals with parallel multi-user querying and Section 5 with distributed querying. Moreover, Section 6 introduces the novel real-time multi-user team querying approach.

## 2. BACKGROUND

### 2.1 The VizIR project

The VizIR project aims at developing an open and extendible framework (mainly, a software workbench) for content-based video and image retrieval. It implements state of the art technologies such as the visual MPEG-7 descriptors, the Multimedia Retrieval Markup Language<sup>24</sup> for loose coupling of query engines and user interfaces, etc. and novel paradigms for feature extraction, querying and evaluation. VizIR is free software: it is released as source code under GNU General Public License<sup>10</sup>. The VizIR project intends to provide a common software platform that can be used by researchers as a workbench for further VIR research, by software engineers for the development of VIR components for media and database applications and by lecturers for teaching VIR concepts. The VizIR framework was carefully designed to guarantee that these requirements can be met. Technically, the VizIR framework components are based on the Java programming language, the Java SDK, the Java Media Framework<sup>23</sup> for media processing and other freely available software libraries. The current status of the project, software releases, documentation and development resources can be found on the project website<sup>25</sup>. VizIR is an open project: new users and contributors are always welcome.

The most important part of VizIR is a class framework for feature extraction, querying, refinement, VIR user interfaces, communication, evaluation and benchmarking. Central element is a service kernel. This class is responsible for media administration and query execution. It communicates with query engines, user interfaces and media databases through XML messages (based on the Multimedia Retrieval Markup Language, MRML<sup>24</sup>) and web services. Media access is hidden in a class that enables accessing the view of visual media (at arbitrary resolution and based on an arbitrary colour model) at any point in time. By that, images and videos can be accessed with a uniform API. Descriptors and media renderer classes make use of the media access class to derive features<sup>5</sup> and to visualise media objects in user interfaces. All elements of user interfaces (including query definition and refinement panels, metadata panels, sketch drawing panels, etc.) are modelled as independent classes that interact with each other through well-defined events and listener methods (locally) or through XML-based web services (remotely). They can be combined arbitrarily to create VIR user interfaces and easily be supplemented by additional querying methods. VizIR query engines are derived from a common model. This model defines how queries are executed technically (not how the querying logic works). This includes the interface to MRML communication classes, the service kernel and paradigms for database access. All VizIR components are based on an object-oriented database model: If desired, the resources of instances of any class in the framework can be serialised and kept persistent in a database<sup>9</sup>. This feature is guaranteed by an underlying persistence system. The object-oriented database system may be chosen arbitrarily. In our test environment we are currently using a relational database (MySQL) and an object mapping tool (Hibernate). More information on the VizIR architecture can be found in the referenced papers<sup>8,4</sup>.

### 2.2 Parallelisation in visual information retrieval

Surprisingly, only few papers have been published in recent years that deal with parallelisation issues in content-based visual information retrieval systems. Below, we give a short overview. Most information retrieval papers discussing parallelisation issues focus on algorithms required for peer to peer retrieval applications. Such work is partially motivated by the recent tremendous success of Internet-based file sharing tools (e.g. Gnutella, Kazaa). The major problem of peer retrieval applications is identifying (or predicting) peers in a network that contain query-relevant information. Classic solutions to this problem originating from text information retrieval research include Breadth-First Search (BFS), Random Breadth-First Search (RBFS), Random Walker Searches (RWS), the Intelligent Search Mechanism (ISM), and Content-Addressable Networks (CAN)<sup>26,15</sup>.

In BFS (also known as flooding), every peer forwards query information to all peer he knows. Retrieval results are returned following the same path. The obvious drawbacks of this solution are bad performance and high network bandwidth consumption. The simplest way to prevent excessive network utilisation is using a Time-To-Live parameter like in IP broadcasting. RBFS differs from BFS in the point that it forwards a query only to a randomly chosen fraction of peers. This strategy reduces resource demand of BFS for the introduction of uncertainty in the peer retrieval process.

RWS differs from RBFS in a single aspect: RWS algorithms forward a query to one randomly chosen peer. ISM tries to estimate the likelihood of peers holding relevant data. This is achieved by building profiles of neighbouring peers and ranking the retrieval results of neighbours. ISM performs well in small environments but suffers from the phenomenon that mostly the same peers are selected for query propagation<sup>26</sup>. Finally, CAN methods compute features (hash values) for data elements that are distributed over the entire peer network. By comparing query text to hash values, peers can identify neighbours that are likely to hold relevant information.

Recently, first peer retrieval approaches on content-based image and video retrieval have been proposed<sup>21, 20, 14</sup>. Generally, as the CAN approach is very similar to the vector space model used in content-based media retrieval (feature transformations, etc.), peer image retrieval approaches follow the CAN direction. The major problem here is that feature vectors used for media description are usually of significantly higher dimensionality than hash values required for inverted hash tables in CANS. To overcome this problem, researchers propose usage of clustering techniques for dimensionality reduction (e.g. Self-Organising Maps<sup>16</sup>, Adaptive Resonance Theory<sup>2</sup>). Then, instead of feature vectors for all data elements, only centroid vectors of clusters are distributed over the entire peer network.

Apart from peer retrieval networks, we could only identify one recent contribution on parallelisation of query execution<sup>3</sup>. In this paper, the authors describe a multi-level approach for  $k$ -nearest-neighbour querying. The server forwards the query to all connected slaves and merges their query results to a global result set. Even though designed for a highly-integrated parallel computer, this approach would be suitable for networks of workstations as well.

### 3. VIZIR QUERYING PROCESS

This section describes the VizIR querying process (Subsection 3.1) and parallelisation issues that follow from the particular properties of the retrieval process (Subsection 3.2). Sections 4, 5 explain, how the listed parallelisation problems are solved in the VizIR project.

#### 3.1 Visual data mining

The VizIR querying process is named visual data mining<sup>7, 6</sup>. Visual data mining is an interactive process that tries to combine the advantages of retrieval and browsing approaches. A three-dimensional media panel is employed to display image and video objects (the latter as Micons<sup>11</sup>) in one view (see Figure 1 for an example). The image plane is used to display media objects while the two floor planes are used to display two-dimensional subspaces of distance space (the result of distance measurement on feature vectors). Hence, every distance relationship of two features can be visualised in the media panel. In addition to media data and distance information, visual data mining visualises all other relevant information in the media panel (e.g. feature data in diagrammatic views), since provision of rich visual information in the user interface is an important concept in visual data mining.

Retrieval in visual data mining is conducted by simply browsing through the distance space (camera movement, zoom in/out) and by defining clusters of positive/negative query examples (e.g.  $c_1, c_2$  in Figure 1). The features displayed on the floor dimensions of distance space can be exchanged arbitrarily. Repeated grouping of media objects defines hyper-clusters of at most as many dimensions as features used in the retrieval system. Cluster borders are computed from the selected media objects. Usage of examples is based on a visibility principle: only media objects explicitly selected in the media panel (of course, not all elements of distance space can be visualised in one view) are considered for query execution. Therefore, a second panel is required for visual data mining that lists the query results. See the referenced publications for more detailed information on hyper-cluster definition<sup>7, 6</sup>.

From what has been said so far, the question arises to which reference point distance is measured in distance space. In the beginning of a retrieval session, all distances are relative to the origin of distance space  $O$ . During the retrieval process the user may define so-called distance neighbourhoods (e.g. elements  $n_1, n_2$  in Figure 1). A distance neighbourhood is a spherical area defined by a median object (the query example) and a radius (defined by the employed distance measure). Within a distance neighbourhood all distances of objects are relative to the median object. The distance of the median object is still relative to  $O$ . Visual data mining allows for re-selection of distance measures in distance neighbourhoods. The user is allowed to drag media objects from their position to new locations in distance space. Then, the retrieval system tries to identify a distance measure (from a pre-defined catalogue) that fits with the user's interaction. If a suitable measure was successfully identified, all media objects in the same neighbourhood are re-

positioned according to this distance measure. Moreover, distance space itself is regarded as the default neighbourhood. Therefore, all actions defined in distance neighbourhoods are also applicable to distance space itself.

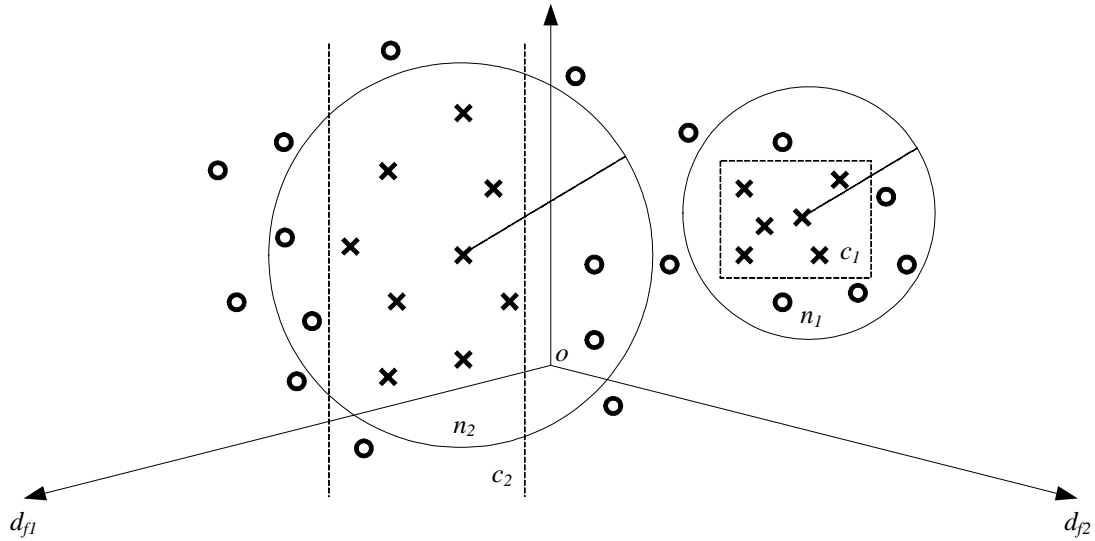


Figure 1: Media panel for distance space visualisation (two distance neighbourhoods, two hyper-clusters).

Visual data mining is a highly sophisticated but still easy to handle similarity-based retrieval process. It is completely different from traditional VIR similarity measurement approaches based on  $k$ -nearest-neighbour queries and iterative refinement using kernel-based learning<sup>15</sup>. One major design issue was that content-based media querying should be fun. Visual data mining allows for parallel definition of multiple queries during one retrieval session. The next subsection discusses parallelisation issues arising from the presented approach.

### 3.2 Parallelisation issues

Figure 2 illustrates the big picture of parallelisation:  $n$  users access a VIR server simultaneously; the server queries  $m$  databases in parallel. While multi-user access is a natural consequence of server-client architectures, the existence of multiple databases (on distributed hosts) follows from growth over time. Considering the visual data mining similarity measurement process, a number of specific parallelisation issues can be derived.

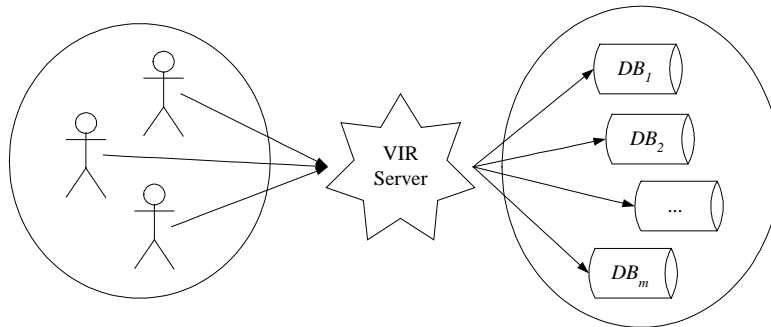


Figure 2: Parallel query execution:  $n$  users and  $m$  databases.

Firstly, multiple distance spaces have to be provided in parallel. Since at least in the beginning, all distance spaces for one media collection are relative to the same origin  $O$ , distance space data can partially be reused. Hence, it makes sense to cache distance space data. Moreover, this argument also applies to often used distance neighbourhoods (at least the elements nearest to the median element). Furthermore, for result set construction from hyper-clusters it is mandatory to keep centralised information on the location of media objects that are distributed over multiple databases. These and other topics will be discussed in sections 4, 5.

It is important to notice, that in the context of VizIR we are hardly dealing with peer to peer retrieval problems. Even though peer configurations can be constructed from the VizIR framework as easy as client-server scenarios, our main target are optimisation of parallel interactive querying (including simultaneous retrieval in one media collection) and parallel database access. Still, query propagation to peers is discussed in Section 5.

#### 4. PARALLEL QUERYING

This section discusses issues arising from serving multiple users in parallel. The solutions sketched below are the basis for the team retrieval approach outlined in Section 6. Subsection 4.2 deals with optimisation of definition and refinement of visual data mining queries. Subsection 4.3 introduces solutions for query execution.

##### 4.1 Introduction

Figure 3 illustrates the basic situation of multi-user querying.  $n$  parallel iterative retrieval sessions require the VIR server to keep session caches for each retrieval session. These session caches hold state information as well as frequently used metadata from the database back-end. This scenario is reality in almost any state of the art VIR system. As pointed out above, the difficulties connected to this situation can be turned into an advantage (in terms of query performance), if proper solutions can be identified for the reuse of session content between the parallel retrieval sessions.

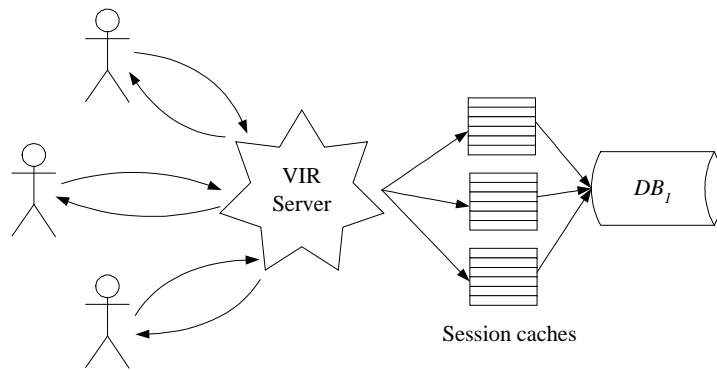


Figure 3: Parallel iterative querying of multiple users on one server.

In detail, parallel execution of VIR queries means parallel definition and refinement of queries as well as parallel query execution. Obvious challenges are reuse of query definition data between sessions (e.g. hyper-cluster definition as well as sophisticatedly computed cluster borders), reuse of similarity measurement data (e.g. distance values of media collection elements to the origin of distance space, distance values for frequently employed distance neighbourhood mean vectors) and reuse of information on frequently observed user behaviour (e.g. often visited areas of distance space, commonly grouped media objects). These topics will be discussed in the next two subsections.

The performance optimisation goal includes, of course, general acceleration of all components of the querying process (e.g. computation of distance neighbourhood layouts, distance measure selection), but also significant reduction of resource consumption (e.g. network bandwidth for media metadata transportation, processing power), minimisation of delays caused by server calls and provision of machine-learned expert feedback (knowledge gained from frequently observed user input). Figure 4 sketches the possible exchange of expert knowledge between parallel retrieval sessions.

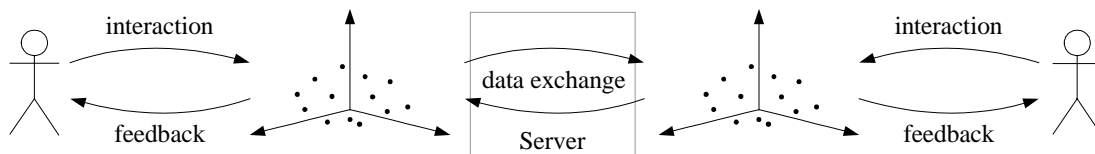


Figure 4: Retrieval data exchange in visual data mining.

## 4.2 Query definition and refinement

The main instruments used for query definition and query refinement in visual data mining are distance neighbourhood definition, implicit distance measure selection and hyper-cluster definition. All three elements are based on rich information visualisation and drag and drop interaction. The first major goal for retrieval metadata caching and reuse is distance space visualisation (depending on distance neighbourhoods and distance measures). Distance values of media objects to the origin of distance space (for the most frequently used distance measures) are required in almost every retrieval session. Hence, they should already be computed on media object cataloguing and stored in the database together with the feature data. Less frequently required distance data (e.g. of distance neighbourhoods) should be stored in primary storages and cache management (insert, update, delete) should be based on usage histograms of the associated distance neighbourhood mean vectors. Distance values can simply be queried from the cache by the identifiers of involved media objects. Next to data reuse, frequently employed distance neighbourhood data could be used for system suggestions (for example, if the user starts defining a distance neighbourhood from a well-known media object). Furthermore, cached distance values can be employed to accelerate the re-positioning process after selection of a new distance measure by the user.

In hyper-cluster definition, caches can be defined for frequently defined borders. Such a cache could be based on a hash table where the hash values are derived from the identifiers of the support vectors (media objects/feature vectors near the cluster border). Cache management could again be based on a simple histogram of frequently occurring hash values. Moreover, hyper-cluster data could be used for system suggestions (e.g. if the system recognises that the user tries to open a new hyper-cluster in a distance space region that has been exploited in other retrieval sessions before).

Finally, browsing information can be cached as well. For example, the users' browsing paths' can be stored in a trajectory-like data structure (e.g. by using the MPEG-7 descriptor of the same name<sup>18, 12</sup>). This data can be analysed to identify frequently visited areas of distance space and give hints to first-time users of obviously 'promising' retrieval areas. Since it is not likely that this type of information is required within short time from observation, it does not make sense to store it in a cache. Conversely, it should be stored in a database of its own. Furthermore, other visualisation (e.g. media object visualisation) should, for the sake of performance, as well be stored (referenced by object identifiers) after computation.

## 4.3 Query execution: caching of query results

In visual data mining, query execution means computation of distance values (for distance neighbourhoods) and evaluation of hyper-clusters (cluster border computation and media object selection). Distance measurement has already been discussed above. Storage of pre-computed distance values is an obvious performance improvement approach. The core problem is selecting those media objects/distance values that promise the highest performance gain. Two factors give good indication: the frequency of usage of certain media objects as neighbourhood means and the frequency of membership of media objects in certain neighbourhoods. Both factors can be estimated over time by simple histogram construction. The histograms can be used for cache management.

Caching of hyper-cluster information is a more sophisticated problem: Various hyper-cluster shapes may occur and small differences may already lead to completely different result sets. For example, the number of features employed in hyper-clusters may vary, the existence/non-existence of a single support vector may lead to different cluster border positions, etc. Hence, in the author's opinion it does not make sense to cache result sets depending on certain cluster configurations (e.g. described by hash values). Instead, only storage of cluster borders (depending on support vector configurations) seems to be promising. An obvious approach could be, as suggested for other issues before, computation of histograms of support vector configurations (described by appropriate hash values) and cache management based on histogram shapes. Here, hash values could, for example, be derived from the object identifiers of support vectors (unique over the entire media collection!). Furthermore, frequently appearing result set elements could also be indexed and, depending on their occurrence probability, be investigated first during the retrieval process.

In conclusion of this section, generally, caching is the appropriate tool to turn the difficulties of parallel VIR server access into an advantage. We gave indication for promising caching areas in the visual data mining approach. Caching should always follow the same paradigm: First, description of objects that should be cached by appropriate hash values. Second, computation of usage frequency histograms for hash values over time. Third and final, cache management

(insert, update, delete) depending on usage results. On cache usage, the question becomes relevant, if it would make sense to provide 'peer to peer shortcuts' (caches on clients)? In our opinion this would not make sense, because reasons cannot be given why clients should have faster access among each other, and because peer to peer communication is generally less error-robust than server to client communication. Therefore, caches should be located and maintained on the server side.

## 5. DISTRIBUTED QUERYING

This section discusses issues relevant for visual data mining raised by distributed VIR databases. Generally, two strategies fit to the problem: replication and indexing the media metadata (Subsection 5.2) and distributing the querying process as well (discussed in Subsection 5.3). VizIR supports both approaches.

### 5.1 Introduction

The basic scenario of database distribution is not as simple as serving multiple users in parallel. Various scenarios have to be distinguished (see Figure 5). For examples, databases containing both media data and media metadata (feature vectors, etc.) may be distributed (Figure 5a) or multiple databases may exist that contain either media data or media metadata (Figure 5b). Of course, mixed scenarios are equally thinkable. A manifold of reasons may be responsible for distributed databases: Media data is often separated from media metadata, because media access requires different data transport mechanisms that metadata (streaming on demand, etc.). Moreover, media objects and metadata are based on absolutely different block-sizes. Storing them in the same database may turn out to be very inefficient. Furthermore, step-wise growth of a system is often responsible for metadata databases that are distributed over multiple hosts.

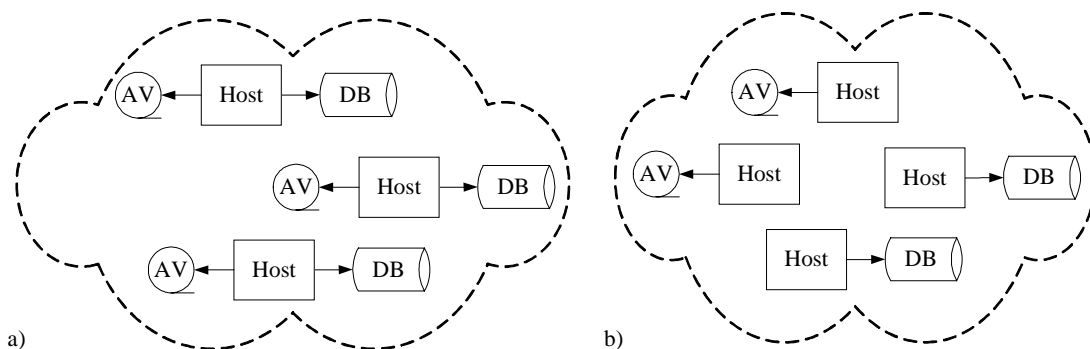


Figure 5: Media and media metadata distribution scenarios.

The major problem caused by multiple parallel databases is identification of the location of relevant media objects and feature vectors. This is the same problem as in peer retrieval networks. Several approaches can be followed to reduce its impact. As in case of multi-user retrieval, the problem can almost be turned into an advantage if we succeed in identifying suitable solutions.

### 5.2 Distributed media and media metadata access

Data replication is state of the art in relational database management systems<sup>22</sup>. So, what else is required in a VIR system? First of all, it is almost impossible (or, at least, would cause unbearable overhead) to replicate all feature data required for a certain query to all hosts involved in a retrieval system. Furthermore, often application-level indexing procedures are required in replication processes that cannot be implemented based on automatic table to table replication. The central question we have to deal with, is: Where do we have to look for the media objects and metadata required for a particular query? Generally, the two solution paths are data identification (e.g. by CAN methods, see Subsection 2.2) and query distribution to all involved databases (e.g. similarity measurement by stored procedures). This subsection surveys the first group of approaches.

The above mentioned simple replication solution is only applicable for a small fraction of required data. Hence, it should only be employed for the most frequently used bits of information. Less frequently required information should not be mirrored by itself, but just in form of indexes. One practically explored solution is similarity-based clustering of media

data and replication of cluster identifiers. Identifiers can be replicated by themselves or in form of hash values. Replication methods include database replication (reliable, easy to setup, not flexible), operating system-based replication using queues and replication daemons (reliable, if networking issues are sufficiently considered, flexible, since additional indexing procedures can easily be attached, resource-demanding), and application level replication by the user (e.g. already on database insert of media objects).

Similarly to database replication, retrieval data can temporarily be mirrored to a server-side cache. Figure 6 describes the workflow in database caching. A usage histogram (e.g. frequency of media usage) is employed as data source for cache management and an update procedure is responsible for media transport from databases to the cache. Of course, this approach requires system-wide knowledge of the location of media metadata. To construct a global data index, again clustering information should be used to derive the most representative data items from media collections. The cluster identifiers should be replicated to the main query server that holds the global database index.

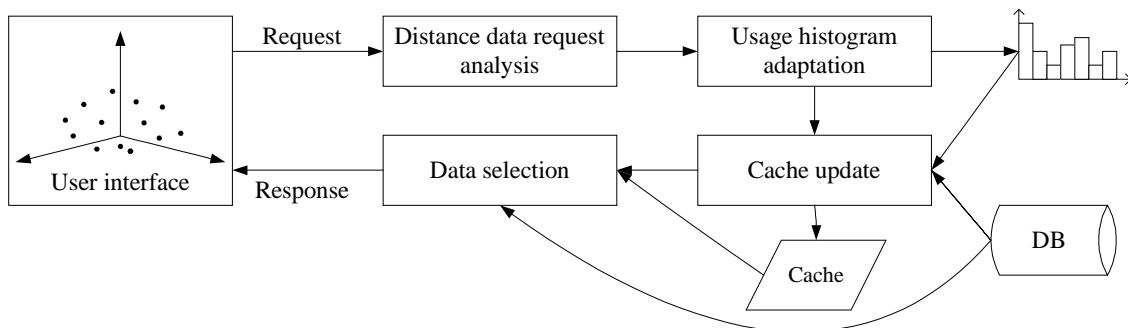


Figure 6: Workflow of retrieval data caching.

One special VIR problem originating from distributed databases is transport of media data and visualisations from database to client. Technically based e.g. on web service for transfer setup and control and streaming for media data transport, this domain is a structural exception from the general server-based concept: Basing media transport on peer to peer communication makes sense, since the opposite (communication over a server) would cause an undesired considerable overhead. Additionally, from the quality of service point of view, direct communication offers less points of failure. Still, in the author's opinion, the best solution is to communicate media transport control over the query server (that holds the global data index) and to establish a direct communication channel for the media data.

### 5.3 Distributed similarity measurement

Propagating distance measurement to the hosts that hold the data is the second solution to the distribution problem. It has the obvious advantage that distance measurement is executed in parallel on multiple hosts and consequently, query execution time is reduced. On the other hand, distance measurement distribution suffers from two drawbacks: Firstly, it still required a global index of media data. Otherwise it would be impossible for the server to identify the hosts holding the relevant data. The second drawback is more severe: It is not that easy to divide the distance measurement process and execute it in parallel on multiple hosts. The technical side of the problem is quite straightforward, as stored procedures are by now state of the art in database management systems and distribution of procedures can, for example, be achieved by replication mechanisms. Replication of query data (hyper-cluster information, distance neighbourhood information) can be performed using the same mechanisms as discussed above.

The algorithm used for distributed querying in VizIR is heavily based on the visual data mining approach: Firstly, distance neighbourhood and hyper-cluster data are transferred to all database nodes that may contain relevant media objects. Then, on these nodes, distance neighbourhoods are computed (or taken from caches) as interactively defined by the user. On these distance neighbourhoods, the hyper-clusters (their borders) are applied to distinguish relevant elements from non-relevant ones. The resulting set of elements is returned to the query server and merged with the result sets of other nodes. This straightforward algorithm uses the advantages of the hyper-clustering approach. Since the result set size is not limited to some number  $k$ , no intersection problems can occur in visual data mining. Still, for query acceleration it would be desirable to have an indexing structure available that allows to predict quicker, which elements may belong to a particular distance neighbourhood or hyper-cluster. Such an index could be implemented in form of a

hash tree similar to the caches described in Section 4.

## 6. TEAM RETRIEVAL

So far, we have dealt with 'basic' parallelisation problems that occur in any VIR system (of course, with tailor-made solutions for visual data mining). In this section we go one step further and describe an approach for simultaneous multi-user querying. Subsection 6.1 gives idea and motivation, Subsection 6.2 lists application scenarios and Subsection 6.3 sketches implementation issues.

### 6.1 Idea and motivation

The major idea behind team retrieval is the online interaction of two or more users. Visual data mining is a visualisation event. We think that visual information retrieval can be and should be fun. Retrieval results would be better, if VIR system design would allow the user for playing with the media space. The visual data mining user interfaces support following this direction. The proposed team retrieval approach is designed like a computer game ('multi-player retrieval'). Of course, this is a highly experimental idea, but it has some serious advantages. Firstly, by introducing social dynamics in the retrieval process, it allows users to gain a common retrieval knowledge. Secondly, it is likely that (over time) specialisation effects happen. Finally, it should not be underestimated that fun means motivation. Information retrieval can be hard, frustrating work. Not just then, motivation can become the decisive factor for success.

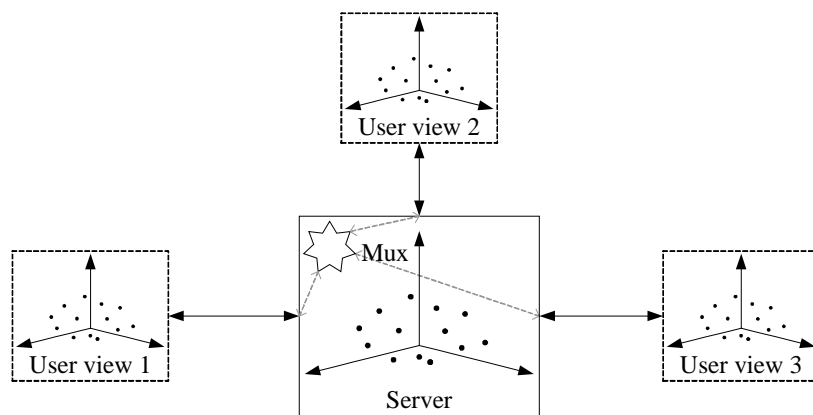


Figure 7: Retrieval cooperation in one distance space.

Figure 7 illustrates the basic concept of team retrieval. Multiple users work within the same media space that is located on the server side. For visualisation and interaction, the visual data mining user interface is used. A multiplexer on the server side distributes all interaction events to all cooperating users. The obvious advantages of this concept are: Similarity perception is shared by all involved users; misinterpretations of one users can easily be corrected by others. Expert knowledge of players can be contributed to the advantage of the entire retrieval process. Moreover, retrieval work can be split among users (e.g. by distance dimensions/features). The major problems are visualisation of parallel interaction and avoidance of inconsistencies in interaction and data access.

### 6.2 Application scenarios

Figure 8 illustrates two user interface configurations for team retrieval. In Figure 8a, all users share one media panel (one view per client), in Figure 8b every user has a view of every other user and himself. Therefore, for  $n$  users  $n$  media panels are required. Additionally, every user has a panel showing the global result set. The first proposed option works like a chatroom: Users communicate through one media that can, for example, be enriched by voice and text communication. The main advantage of this approach is that by focussing the users' attention in one media panel cooperation becomes highly interactive. The main disadvantage is that, especially, if large numbers of users are involved, the retrieval process can soon become confusing, users may lose orientation and understanding of the querying process. However, from the point of view of multi-user playing, this approach comes nearest to state of the art real-time strategy games. To further stress the 'motivation by gaming' aspects, a voting system could be added to the team retrieval approach that allows users

giving points to successful retrieval operations of colleagues. Scores would also allow for easier judgement of the relevance of retrieval operations (as a further factor for cache management in parallelisation and data distribution).

The approach presented in Figure 8b allows the best overview over the operations conducted by the entire team. One interaction panel per user would enable following the retrieval process and switching attention at any time to the session that looks most promising. Furthermore, it would for example allow to split the retrieval process feature-wise between users: Retrieval team members could act as domain experts and work only on specific features (e.g. colour, shape). The main advantage of this approach is that retrieval work is easy to overlook. The main disadvantages are high display overhead and, in case of many simultaneous users, a confusing lot of parallel movements in the user interfaces.

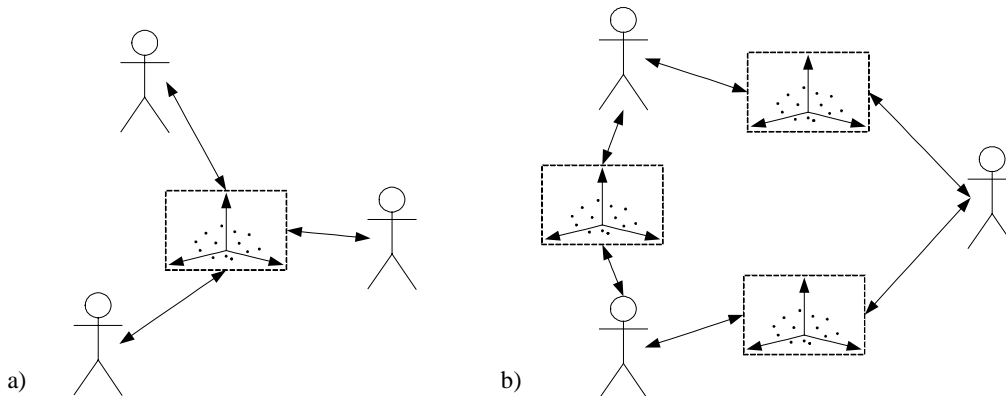


Figure 8: Examples of team retrieval user interface configurations.

From this sketch we can already see that team retrieval opens an entirely new field for visual information retrieval research. The two selected scenarios are just examples. Other options, tailor-made for particular application domains, can easily be constructed. In the next subsection, we discuss technical issues relevant to all team retrieval approaches.

### 6.3 Visual data mining implementation

For the VizIR project, we decided on a solution, where every user interacts with his colleagues through one media panel. All users are working in the same distance space and may choose any subspace of distance space. To distinguish operations of different users, concerned media objects, distance neighbourhoods and hyper-clusters are labelled by the user name. To guarantee coherency, it is necessary that all events (including feature selection, distance measure selection, neighbourhood definition and hyper-cluster definition) are reliably communicated to all peers. As described above, we use an event dispatcher located on the server side to fulfil this task. Histories of interaction of peers (e.g. movement paths through distance space) can be visualised in addition to the user's own history. In addition to the shared media panel, every user holds a second local media panel for experiments and a flat panel that shows the global result set. This configuration allows highly interesting interaction and team dynamics. Moreover, the retrieval system can compare the users' behaviour and draw conclusion on their similarity perception and retrieval goals.

Technically, in addition to event propagation (based on event queues and message-based replication) mechanisms are required for session initialisation and locking of shared data. In VizIR team retrieval, session initialisation is implemented straightforward: A new client simply connects the VIR server at a port reserved for team communication. Then, it is initialised with distance space and the current retrieval situation as it would be in a single-user retrieval session. Locking is only required for manipulated data: distance neighbourhood information and hyper-cluster information. Communicating over a server allows a simple solution: If two events arrive at the event multiplexer that would both manipulate the same item (e.g. a distance neighbourhood or a hyper-cluster), then the server forwards the conflict to all involved users. Based on their decision the system drops one event and propagates the other to all participants of the retrieval session.

One final problem is the implementation of a scoring system. In VizIR, we implement an algorithm that gives a user two points for every result set element that originates from one of his hyper-clusters and one point for every other result set element. Hence, the score of a user changes permanently during a retrieval session: If a user shrinks or deletes a hyper-

cluster, he lowers the creator's score but also his own. This scoring model should motivate users as well as encourage cooperation. The combination of user scores and interaction information can be exploited to derive successful retrieval patterns. By that, the visual data mining retrieval system learns from user behaviour.

## 7. CONCLUSIONS AND FUTURE WORK

This paper outlines the implementation of parallel visual information retrieval in the VizIR software workbench. Parallelisation comprises offering service to multiple users in parallel and accessing distributed database systems. All presented solutions are based on the visual data mining concept used for similarity measurement in VizIR. Mostly, caching techniques are employed to accelerate the parallel querying processes. In addition to basic parallelisation issues, the paper discusses approaches for team retrieval (simultaneous multi-user retrieval). The presented ideas are highly experimental and analogies to real-time multi-user games are obvious. Still, serious advantages (improvement of motivation, learning from user behaviour) can be gained from team retrieval.

The solutions sketched in this paper will be implemented in VizIR framework. VizIR is an open project. The outcome can be downloaded from the project website<sup>25</sup>. Our future work will include further research on plausible team retrieval scenarios (e.g. coupling of user interfaces), more sophisticated caching strategies for query data and retrieval results, storage and clustering of distance information instead of feature data in distributed databases and construction of global data indexes from distance information.

## ACKNOWLEDGEMENTS

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

## REFERENCES

1. A. Del Bimbo, *Visual Information Retrieval*, Morgan Kaufmann Publishers, San Francisco, 1999.
2. A. Baraldi, P. Blonda, "A survey of fuzzy clustering algorithms for pattern recognition," *IEEE Transactions on Systems, Man and Cybernetics*, 29/6, 786-801, 1999.
3. J.L. Bosque, O.D. Robles, A. Rodriguez, L. Pastor, "Study of a Parallel CBIR Implementation using MPI," *Proceedings IEEE Workshop on Computer Architectures for Machine Perception*, 195-204; 2000.
4. H. Eidenberger, "Media Handling for Visual Information Retrieval in VizIR," *Proceedings SPIE Visual Communications and Image Processing Conference*, vol. 5150, 1078-1088, SPIE, Lugano, 2003.
5. H. Eidenberger, "Modelling of Visual Feature Derivation in the VizIR Framework," *Proceedings European Signal Processing Conference*, Vienna, 2004 (accepted for publication).
6. H. Eidenberger, "Visual Data Mining," *Proceedings SPIE Optics East Conference*, Philadelphia, 2004 (accepted for publication).
7. H. Eidenberger, C. Breiteneder, "Visual Similarity Measurement with the Feature Contrast Model," *Proceedings SPIE Storage and Retrieval for Media Databases Conference*, vol. 5021, 64-76, SPIE, Santa Clara, 2003.
8. H. Eidenberger, C. Breiteneder, "VizIR – A Framework for Visual Information Retrieval," *Journal of Visual Languages and Computing*, 14/5, 443-469, 2003.
9. H. Eidenberger, R. Divotkey, "A Data Management Layer for Visual Information Retrieval," *Proceedings ACM Multimedia Data Mining Workshop*, Seattle, 2004 (to appear).
10. Free Software Foundation, General Public License website, <http://www.gnu.org/copyleft/gpl.html> (last visited 2004-07-31).

11. B. Furht, S.W. Smoliar, H. Zhang, *Video and Image Processing in Multimedia Systems*, Kluwer, Boston, 1996.
12. S. Jeannin, A. Divakaran, "MPEG-7 Motion Descriptors," *IEEE Transactions on Circuits and Systems for Video Technology*, 11/6, 720-724, 2001.
13. J.M. Jolion, "Feature Similarity," in M.S. Lew (ed.), *Principles of Visual Information Retrieval*, Springer, Heidelberg, 121-144, 2001.
14. V. Kalogeraki, A. Delis, D. Gunopulos, "Handling Multimedia Objects in Peer-to-Peer Networks," *Proceedings ACM Symposium on Cluster Computing and the Grid*, 408-409, 2004.
15. I.A. Klampanos, J.M. Jose, "An Architecture for Information Retrieval over Semi-Collaborating Peer-to-Peer Networks," *Proceedings ACM Symposium on Applied Computing*, 1078-1083, ACM, Nicosia, 2004.
16. T. Kohonen, "The Self-Organizing Map," *Proceedings of the IEEE*, 78/9, 1464-1480, 1990.
17. M.S. Lew (ed.), *Principles of Visual Information Retrieval*, Springer, Heidelberg, 2001.
18. B.S. Manjunath, P. Salembier, T. Sikora, *Introduction to MPEG-7*, Wiley, San Francisco, 2002.
19. O. Marques, B. Furht, *Content-Based Image and Video Retrieval*, Kluwer, Boston, 2002.
20. R. Maxim, S.C. Hui, "Intelligent Content-Based Retrieval for P2P Networks," *Proceedings Conference on Cyberworlds*, 318-325, 2003.
21. W. Müller, A. Henrich, "Fast Retrieval of High-Dimensional Feature Vectors in P2P Networks using Compact Peer Data Summaries," *Proceedings ACM Workshop on Multimedia Information Retrieval*, 79-86, ACM, Berkeley, 2003.
22. M.T. Ozsu, P. Valduriez, "Distributed database systems: where are we now?" *IEEE Computer*, 24/8, 68-78, 1991.
23. SUN Microsystems, Java Media Framework website, <http://java.sun.com/products/java-media/jmf/> (last visited 2004-07-31).
24. University of Geneva, Multimedia Retrieval Markup Language website, <http://www.mrml.net/> (last visited: 2004-07-31).
25. Vienna University of Technology, VizIR project website, <http://vizir.ims.tuwien.ac.at/> (last visited 2004-07-31).
26. D. Zeinalipour-Yazti, V. Kalogeraki, D. Gunopulos, "Information Retrieval Techniques for Peer-to-Peer Networks," *Computing in Science & Engineering*, **6/4**, 20-26, 2004.