11

High-Dimensional Indexing for Video Retrieval

Catalin Calistru[2,3], Cristina Ribeiro[1,3] and Gabriel David[1,3]
[1]*FEUP, Faculdade de Engenharia da Universidade do Porto*
[2]*ISPGAYA, Instituto Superior Politécnico Gaya*
[3]*INESC, Porto*
Portugal

1. Introduction

The video retrieval task raises many fundamental questions in computer vision and information retrieval, such as how to represent video items, what information can be directly extracted from them, and how to explore such information in order to satisfy the user's information need. Video items are intrinsically complex, and the analysis of their content requires heavy computing processes. Excepting the case of text, multimedia content analysis does not result in the high-level concepts required by the generality of the search tasks. This brings about the so-called "semantic gap", clearly identified as the main issue in multimedia retrieval Gudivada & Raghavan (1995); Smith (2007). Text communication is based on concepts, expressed in the user's language and close to the way humans think. Searching text items may require a number of processing techniques like pattern matching, stemming, finding synonyms, translating, natural language analysis. Supposing that the user expresses his information need through words, the set of retrieved documents includes those containing precise or imprecise word matches and can be continuously enlarged to more and more semantically related documents. In the case of a video repository, however, there is not such a clear semantic channel between the object's content and the user information need. The automatic video analysis may produce many descriptors related to the contents, the so-called low-level descriptors Bober (Jun 2001); Manjunath et al. (2001); Mufit Ferman et al. (2000), but hardly produces accurate descriptions close enough to human concepts Snoek & Smeulders (2010); Tesic & Smith (2006). This is a problem for a wide range of video-based applications, other than search and retrieval, such as human-computer interface, security and surveillance, copyright protection, and personal entertainment.

While significant progress has been achieved towards narrowing the 'semantic gap' in image and audio retrieval, Celma & Serra (2008); Smeulders et al. (2000); Yan & Hauptmann (2007), there are no satisfactory systems for video retrieval Smith (2007); Snoek & Worring (2005); Snoek, Worring, van Gemert, Geusebroek & Smeulders (2006). In fact, the video retrieval techniques overlap significantly with the image-based ones. In practice, video repositories Westerveld (2005) are treated as statical sets of representative images, the keyframes. True video search systems, capable of analyzing the whole spatio-temporal information available in video are nor yet available Ekin et al. (2004); Snoek et al. (2003).

The 'semantic gap' is not the single problem to be solved in video retrieval. For example, a retrieval system should rank all the video items in a dataset by their degree of similarity to a given query, where the similarity is typically assessed by means of some similarity distance between the video items and the query. However, it is not yet fully understood how to convey the human similarity judgments in terms of distance computations. It is not even clear whether the similarity computation should be of metric nature or not Eidenberger (9-11 Dec. 2002); Santini & Jain (1997). Although metric-based similarity computations are frequently preferred, the choice is not based on a clear criteria for the selection of similarity measures. Recent studies on distance metrics Yu et al. (2006) provide some insight on this issue. However, except in the case of specific application domains, where the ranking criteria are established a priori, the video retrieval systems must account for flexibility in the choice of similarity measures.

While the issues mentioned earlier, namely the 'semantic gap' and the variability of metrics, are concerned with retrieval effectivity, there is more to retrieval that must be considered. For example, the emerging retrieval systems are expected to be seamlessly integrated into existing databases, which raises questions related to database organization, such as the data/metadata dichotomy Bulterman (2004); Kosch et al. (2005), or how to integrate the video search engines with the existing search facilities Chaudhuri et al. (2005). But the most important issue, from the point of view of integration with existing database systems, is the search efficiency in the context of continuously growing datasets. It is widely accepted that the growth rate of multimedia content production increases the length of time it takes to process a dataset, from its production until it is available for search; we will refer to this as the time to search issue.

Analyzing current multimedia search approaches IBM (2008); Wactlar et al. (1999) from the time to search point of view, we observe that, if machine learning techniques are used to extract high-level concepts, large training times are required. The manual annotation alternative Shneiderman et al. (2006), although essential in many cases, cannot keep the pace with the growth rate of video productions. On the other hand, if search relies on low-level similarity, in a query-by-example fashion, a short time to search can be achieved Calistru et al. (2006); Tesic (2004). Two main directions can be followed: effectivity-oriented, where finding relevant objects is the priority, and efficiency-oriented, where the quality of results is traded for speed. The goal here is to tackle the video search and retrieval tasks from both the efficiency and the effectiveness points of view. The idea is to build an indexing system that can search—with the shortest possible time to search—in large datasets of low-level descriptors, and incrementally integrate, as they become available, high-level information from the manual and automatic annotators.

The chapter is organized as follows. Sections, 2, 3, and 4 offer insights into three essential retrieval aspects, namely features and descriptors, similarity models, and multidimensional indexing. As retrieval systems do not operate with the video items themselves, but with representations thereof, Section 2 reviews the feature types that can be extracted, how they are being represented, and what is their impact in retrieval. In Section 3, we discuss a second important issue in retrieval, namely how to discriminate between items. In order to be comparable, all the video items must share a common representation type, and, depending on the representation, a range of similarity measures can be applied. Considering the large volume of high-dimensional feature data and the variety of similarity measures, efficient

and versatile indexing methods are needed. This brings about the efficiency aspect, which we strengthen in Section 4. This section comprises a review of the current state of the art in multidimensional indexing, and continues to identify a set of requirements that indexes for video data should comply. In sections 5 and 6 we illustrate a video retrieval case study where, using a custom-designed high-dimensional index, the BitMatrix, we efficiently index a large set of high-dimensional descriptors. Section 5 presents the BitMatrix, showing that it relies on bitwise operations, can be conveniently arranged for efficient sequential access, and can be easily broken into segments for distributed or parallel processing. Moreover, it adapts gracefully to continuously growing datasets, such as the video contents produced by cameras. In the sequel, a set of BitMatrix-based experiments, carried on in both synthetic and real datasets, such as TRECVID 2007 are presented in Section 6. We show that our search strategy can cope with the trade-off between flexible video ranking, required for capturing the multitude of similarity facets, and query execution speed. The last section includes a chapter summary and some current research trends.

2. Features and descriptors

Feature extraction is the process of obtaining descriptors from multimedia items. When the items are digitized versions of some real world objects, it is important to note that the items properties are not necessarily the properties of the objects in the world. The digitizations are obtained by sensory means and sensor limitations introduce a so-called "sensory gap". For example, due to lighting conditions, a red car can appear as white in a video shot. It is easy to imagine that a color-based search for shots that contain red cars would not find that video shot. Although this is a trivial example, in many critical domains such as medicine, the 'sensory gap' must be seriously taken into consideration. However, throughout this section we are not concerned with this issue; the relationship between real world objects and multimedia items is not explored.

In a first, and most common categorization, the features can be either low or high-level. The low-level features are the ones directly obtained from the multimedia items, such as color, texture, shape, motion and audio features Deselaers et al. (2004). They are used in several application domains, namely object recognition, surveillance, diagnostics and content-based retrieval.

The high-level features consist of keywords or more complex natural language formulations that capture human concepts. For the text-based multimedia items, high-level descriptors can be directly extracted from the content itself, but for the other modalities a direct semantic connection to human concepts does not exist. High-level descriptors can be obtained by means of either automatic Jiang et al. (2005); Zhang & Chen (2003) or manual annotations. Whatever the annotation type, there are arguments for and against their use. For instance, the manual annotations are considered subjective and expensive to obtain, but if domain experts validate the annotations, they can be accurate Shneiderman et al. (2006); Yee et al. (2003). An argument in favor of automatic annotations is that they are cheaper to obtain when trained concept detectors already exist. However, such detectors are available only for a small number of concepts and often provide low accuracy rates. If concepts such as 'water', 'sky', 'cars', 'faces' or 'outdoors' are relatively well-detected, concepts such as 'entertainment' or aspects such as preferences Bartolini et al. (2005) or moods Hanjalic (2006) are far from

being correctly identified. To cope with the automatic concept detection challenges, the multimedia communities are currently establishing concept lexicons Naphade et al. (2006); Snoek, Worring, van Gemert, Geusebroek & Smeulders (2006), focusing on the concepts that are feasible for automatic detection Hauptmann et al. (2007); Snoek, Worring, Geusebroek, Koelma, Seinstra & Smeulders (2006); Yang & Hauptmann (2006).

Beside the low/high-level categorization, the features can also be local/global and variant/invariant. If the feature extraction targets specific item regions, the features are called local and if the whole item is analyzed, the features are called global. A feature is considered either variant or invariant depending on how the feature values are sensitive to item transformations. For example, if the shape feature values of an image are insensitive to rotation, such a feature is called rotation-invariant. The variance/invariance can be judged with respect to other transformations, such as scaling, location change, illumination variation, viewpoint transformations, or occlusions.

Feature representations are typically incorporated as descriptors. According to the feature types presented above, the descriptors can be considered low/high-level, local/global and variant/invariant. Between features and descriptors there is a one to many relation, as a feature can be represented in multiple ways. For example, DominantColor, ColorStructure and ScalableColor descriptors Manjunath et al. (2001) are all color descriptors. Low-level descriptors consist predominantly of vectorial data. The DominantColor descriptor, for example, consists of an RGB-tuple (red, green, blue). The color histogram descriptors are even larger vectors of 128 or 256 values. The entire set of descriptors can be viewed as a vectorial space, also called the feature space, which generally has hundreds or thousands of dimensions.

A possible question to address here is: what descriptors should be extracted that would help finding relevant multimedia items Deselaers et al. (2004); Jiang et al. (2007)? As an answer to such a question, one may expect a set of descriptors that guarantees up to some degree of confidence that an effective retrieval system can be built on top of it. Such an answer is difficult to obtain, because the retrieval quality is a problem that surpasses the choice of descriptors. However, we can say that retrieval based on low-level features is already mature Flickner et al. (1995); Rehatschek et al. (2004); Smith & Chang (1996); Wactlar et al. (1996), but inferring high-level features is still a challenging task Gevers & Smeulders (2004); Hanjalic (2006); Huijbregts et al. (2007); Smeulders et al. (2000); Xiong et al. (2006). As the high-level feature extraction depends, up to some extent, on the low-level features, we expect that the use of low-level features will still grow Burghouts & Geusebroek (2009).

Recent years have been marked by a shift from global and variant descriptors, such as color histograms and global shape descriptors, to local invariant ones, such as keypoints Burghouts & Geusebroek (2009), region-based, and local shape characterizations. This shift is explained by the complexity of the multimedia items. The semantics covered by a whole item is too deep for global descriptors to cope with. It has been observed that local descriptors correspond better to item parts such as objects or persons. Invariance also became increasingly important because it helps identifying objects under various circumstances.

3. Similarity models

The similarity models are the means by which the multimedia items can be compared. Although the choice of a specific similarity model is generally dependent on the features that are involved, in this section we ignore this type of dependency. The main question has a general nature: how can one discriminate among video items? We start with a preliminary discussion on what similarity is, and then continue with a review of several similarity models.

3.1 About similarity

The similarity between two multimedia items can be seen generically as a relationship between them. There are domains, such as geometry, in which the similarity is precisely defined: two geometrical objects are called similar if one is congruent to the result of a uniform scaling (enlarging or shrinking) of the other. With this definition we can easily assess the similarity between geometric objects. For example, two circles are always similar to each other, two squares are always similar to each other, and two triangles are similar if and only if they have the same values for the three angles. But unlike geometric similarity, which is precise, similarity in retrieval is imprecise and depends on numerous facets of the items, on the user and on the interaction context.

There is a difference between similarity and matching Kherfi et al. (2004), which may lead to different approaches. Matching requires a comparison between two items—a binary operator— to check whether the items are identical or not; it is used for copy identification, for example. But similarity, although it sometimes uses matching techniques, is a more complex process, requiring knowledge from very diverse domains. In modeling similarity, the similarity judgments are often based on subjective features grouping which yield user-dependent degrees of similarity Gentner (1988); Santini & Jain (1997).

In retrieval, similarity is evaluated by some comparison between a given query and the items' feature values (descriptors). The features that can be checked for relevance range from low-level ones such as color or shape, to high-level and subjective ones such as a feelings or moods Dimitrova (2004); Hanjalic (2006). In the sequel, the most common similarity models are presented.

3.2 The Vector Space Model

In the Vector Space Model, (VSM) both the queries and the objects are represented as points in high-dimensional vector spaces. Although created as a text retrieval model, Faloutsos & Oard (1995); Salton (1971); Van Rijsbergen (1979) VSM is not restricted to the text modality. Generic multimedia items can be captured, given that their feature types, such as color, texture, shape, or motion have vectorial representations.

Assuming a vector space with N dimensions, an item D_k is represented by a vector $D_k = [d_{k0}, d_{k1}, .., d_{kN}]$ and a query Q by a vector $Q = [q_0, q_1, .., q_N]$, where N depends on the feature type, d_{ki} and q_i are the D_k and Q feature values for dimension i. The similarity between Q and D_k is then computed as a distance $d(Q, D_k)$, where d can take a large variety of forms, Datta et al. (2008) such as cosine similarity, Faloutsos & Oard (1995); Salton (1971); Van Rijsbergen (1979) quadratic distance, Böhm, Kriegel & Seidl (2001); Ishikawa et al. (1998) L_p distance, Aggarwal et al. (2001); Howarth & Rüger (2005b); Yan & Hauptmann (2007) Chebyshev

distance, Li et al. (2006) Earth Mover's Distance, Carson et al. (1999); Rubner et al. (2000); Wu & Bretschneider (2004) and localized metrics Aggarwal & Yu (2000); Cha (2003); Howarth & Rüger (2005a).

3.3 Feature Contrast Model

The Feature Contrast Model (FCM), also known as Tversky's model Tversky (1977), is a similarity model that does not assume a metric nature for the human perception, which is the default assumption in the VSM model. The triangle inequality and the symmetry axioms were considered too restrictive.

Unlike the vector space model, which represents the items as points in a vector space, Tversky treats them as sets of binary predicates. Let X and Y be two feature sets corresponding to items x and y. The contrast model obtains the similarity of the two items by combining the common features ($X \cap Y$) and the distinctive features ($X \setminus Y$ and $Y \setminus X$):

$$Sim(x,y) = f(X \cap Y) - \alpha f(X \setminus Y) - \beta f(Y \setminus X). \quad (1)$$

The formula in Equation 1 is not a metric, because the symmetry axiom does not hold. Tversky's view on similarity assessment was extended to geometric Santini & Jain (1997) and fuzzy feature contrast Santini & Jain (1999) models. Comparisons with the Euclidean distance Eidenberger & Breiteneder (2003), and with various other metrics for MPEG-7 descriptors Eidenberger (2003) have shown that in many cases the FCM performs better. However, the FCM has been criticized for not capturing relationships between features Rada et al. (1989) .

3.4 Probabilistic Model

The idea behind the Probabilistic Model (PM) is to predict the probability that a given multimedia item will be relevant to a given query. It relies on the assumption that the distribution of some concepts throughout the collection, or within some subset of it, may be informative of the likely relevance of the items. With this assumption, accurate estimates of the probabilities can be obtained and the documents can be ranked according to this probability of relevance Benitez & Chang (2002); Bohm et al. (2007); H.R.Turtle (1991); Larson et al. (1996); Macdonald & Ounis (2006); Robertson (1997).

3.4.0.1 Language-based models

A special category of probabilistic models, the Language-based Models (LM) Kraaij (2005); Ponte & Croft (1998); Zhai & Lafferty (2001) propose to statistically model the use of language in a multimedia collection in order to estimate the probability that a query is generated from a particular document. The main idea is that, if the query could have come from the document, then that document is likely to be relevant.

3.5 Generative Model

In this model, every object is considered as the outcome of a process that generated it. The idea is to include in the similarity assessment knowledge about the likelihood of existence of the objects to be compared. It has been argued that the generative processes are important since they help the selection of critical features for similarity comparison Kemp et al. (2005).

As an example, Kemp et al. ask which is more similar to a given nutritious mushroom: a mushroom identical except for its size, or a mushroom identical except for its color? They suggest that knowing how mushrooms are formed, i.e. their generative process, we can be sure that mushrooms grow from small to large and their final size depends on the amount of sunlight and soil fertility. Therefore, it is more likely that the differently-sized mushroom is more similar than the differently-colored.

When applied to multimedia retrieval, the generative models follow probabilistic approaches. Under the assumption that each item was obtained from some specific generative process, the similarity is assessed through the probability that the query is an outcome of the item's process Westerveld (2004).

3.6 Rank Aggregation Model

In this model the assumption is that several independent rankings with respect to a query object already exist and these have to be merged into a single results list. Thus, the model is not directly applicable on the original feature values, but on intermediate rank lists. As an example, we can have a situation, where several descriptors are involved in the similarity computation and metrics adapted to each descriptor are required. In such a case, descriptor-wise rank lists are obtained, and a further aggregation strategy is required.

The common approach for the aggregation of several rank lists is to use scoring functions, such as *min* or *average* in order to compute overall scores. Depending on the aggregation function and the termination condition, several aggregation algorithms such as *Fagin's Algorithm*, *Threshold Algorithm*, *medrank* Fagin et al. (2003) and *Quick-Combine* Güntzer et al. (2000), have been proposed.

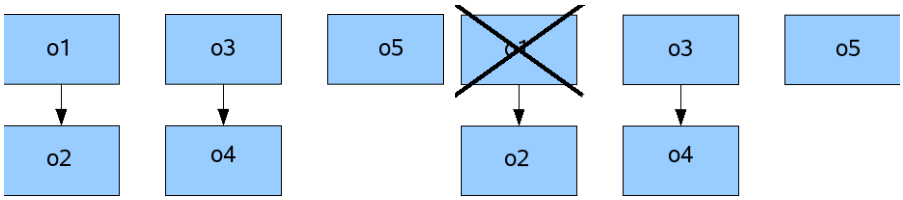
3.7 Preference Relations Model

The scoring functions used in aggregation models are quantitative and assign scores to multimedia items based on their feature values. When multiple rankings exist, user preferences for one result set or another are modeled with weights. However, the scores and weights have a limited expressive power, since not all the user preferences can be translated into quantitative expressions. The preference-based similarity model appeared as an alternative.

Figure 1(a) illustrates a first example. A small database composed of five objects is assumed, where object o_1 is preferred to object o_2 and o_3 is preferred to o_4. There is no preference between o_1, o_3 and o_5. If we try to capture these preferences with a scoring function, the object scores can be assigned in the following manner:

$$S(o_1) = S(o_3) = S(o_5) > S(o_2) = S(o_4). \tag{2}$$

Let's assume now that object o_1 is deleted from our database, as in Figure 1(b). Looking at the preferences we see that in the absence of o_1, o_2 should not be second to other objects, but looking at the scores we have $S(o_3) = S(o_5) > S(o_2) = S(o_4)$, which do not place o_2 among the top objects. That happens because the scoring function evaluates the similarity only by quantitative means, not accounting for the relations with the other objects in the database.



(a) Before o_1 removal (b) After removing o_1

Fig. 1. Preference Diagram

As an alternative to scoring functions, Chomicki Chomicki (2002; 2003) proposes a similarity approach based on qualitative preference relations. This technique requires that, given two objects o_i and o_j, there must exist a binary relation —the preference relation— that states whether o_i is preferred to o_j ($o_i \succ o_j$) or not ($o_i \not\succ o_j$). If $o_i \succ o_j$ we also say that o_i *dominates* o_j. If neither $o_i \not\succ o_j$ nor $o_j \not\succ o_i$, then we have that $o_i \sim o_j$, which represents the *indifference relation*. Assuming that queries are expressed by means of preference relations, it is possible to define an operator, called *skyline-operator* Börzsönyi et al. (2001) or *winnow* Chomicki (2002), that computes the set of preferred objects, i.e. all the objects that are not dominated by the others. The skyline operator is defined as:

$$skyline(DB) = \{o \in DB | \nexists p \in DB, p \succ o\}. \quad (3)$$

In the previous example, after the deletion of o_1, o_2 becomes a skyline object, i.e. is not dominated by any other object. In preference-based retrieval applications such as Bartolini et al. (2005); Leubner & Kießling (2002), sequent skyline iterations are reported and the partial object sets are ranked: the set of skyline objects, followed by the skyline of the remaining objects, and so on. Although it is a ranking without scores, also called a qualitative ranking, the interest in having retrieval systems that take into account the user preferences led to an increasing adoption of skyline-based similarity models Balke & Güntzer (2004); Bartolini & Ciaccia (2005); Godfrey et al. (2005); Papadias et al. (2005); Pei et al. (2005); Yuan et al. (2005)

3.8 Network Model

The idea behind this model is to represent the items as nodes in a network, with *part-of* or *is-a* relations between items. The most representative class for network models are the semantic networks, where the nodes of the network are concepts, eventually coming from predefined ontologies. Such a representation mode is often referred to as a conceptual graph Nguyen & Corbett (2006). The measure of similarity between two nodes(concepts), is the length of the shortest path between them. Usually the semantic neighborhood of radius r of a concept C is defined as the set of all the concepts that have the distance to C smaller than r.

Ekin et al. Ekin et al. (2004) propose a semantic model dedicated to video retrieval with entities and relations between them. Instantiations of this model are graphs, allowing for graph-based queries, while the similarity is obtained by graph-based matching.

3.9 Hybrid approaches

In current multimedia retrieval prototypes, similarity is assessed with hybrid approaches, i.e. by combining various similarity models. Schwering, for example Schwering (2005), introduces a hybrid model for semantic similarity that combines the network model, in the form of semantic network, with the geometric model. The resulting model becomes a network of vector spaces, where each node of the net represents a concept, and each concept is further mapped to a vector space. The similarity is obtained in two phases: first the query concepts are aligned with the concepts available in the model using semantic networks techniques and then a metric is applied on the corresponding vector spaces. Raubal proposes a similar approach Raubal (2004).

4. Multidimensional indexing methods

We have seen in the previous section that the similarity between multimedia items can be evaluated with a wide range of models. Among them, the VSM-based retrieval models are by far the most used. Therefore, there is an increasing interest in speeding similarity-based retrieval on top of such models. We remind that in VSM the multimedia items are modeled as points in a high-dimensional vector space and the similarity between them is assessed with metric-based distance computations. Given a query object q from a universe of objects $\mathcal{O}$ and a metric function d, finding similar objects means identifying particular sets of objects:

- the *Nearest Neighbor Set* $NN(q)$, defined as $\{o \in \mathcal{O} | \forall v \in \mathcal{O}, d(q,o) \leq d(q,v)\}$;
- the *k- Nearest Neighbors Set* $NN_k(q)$, defined as the set of k elements closest to q in $\mathcal{O}$, i.e. a set objects $A \subseteq \mathcal{O}$, such that $\mid A \mid = k$ and $\forall o \in A, v \in \mathcal{O} - A, d(q,o) < d(q,v)$;
- the *approximate Nearest Neighbor Set* ($NN_A(q)$), which is the set of objects $\{o \in \mathcal{O} | d(q,o) \leq (1+\epsilon) * d(q, NN(q)), \epsilon > 0\}$.

The computation of these neighbor sets becomes challenging when dimensionality increases, a problem often referred to as the "curse of dimensionality" Beyer et al. (1999); Indyk & Motwani (1998). In practice, the naive and inefficient approach of simply comparing the query vector to all feature vectors has proved comparable, sometimes even more efficient than especially designed indexing methods such as the "R-tree" Beckmann et al. (1990). Due to the increasing importance of search in high-dimensional spaces, a great diversity of indexing methods have been proposed in recent years. In spite of their diversity, they all avoid looking at every object by creating groups of objects with common properties. Under the assumption that the grouped objects are requested or pruned together, checking the groups instead of individual objects saves time. For example, the objects in a distance range can be pruned or not just by checking the range's minimum and maximum values. In the following we review the most used multidimensional indexing methods.

4.1 Spatial Access Methods

Spatial Access Methods (SAM), also called feature-based methods, partition the space based on the values of the vectors along each independent dimension.

SAM are based on tree data structures with two types of nodes: data nodes (the leaves) and directory nodes. The information stored in directory nodes describe space regions obtained

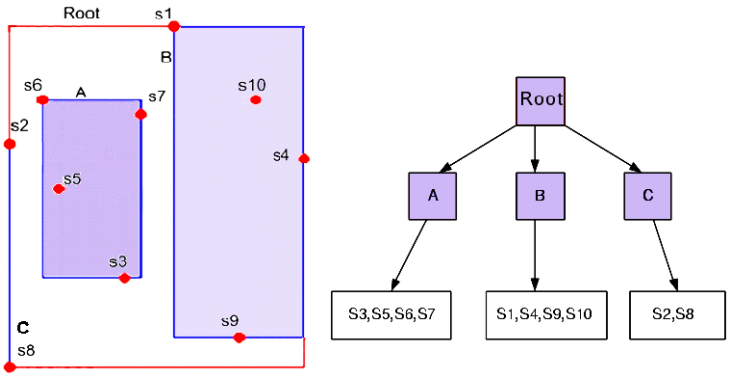


Fig. 2. An R-Tree example

with various partitioning strategies. There can be strategies such as data partitioning (DP) which uses minimum bounding regions (MBR) such as *R-tree*, R^**-tree* Beckmann et al. (1990), *X-tree* Berchtold et al. (1996), bounding spheres such as $SS-tree$ White & Jain (1996), MBR and bounding spheres such as $SR-Tree$, generic minimum bounding regions (hyper rectangle, cube, sphere) such as the $TV-tree$, the $BBD-tree$ Arya et al. (1998) and space partitioning methods (SP) such as the $kDB-tree$, $Hybrid-tree$, $SH-tree$ Böhm, Berchtold & Keim (2001).

The most representative SAM approach, the *R-tree*, is illustrated in Figure 2. The left-hand side of the figure shows the MBR around 10 data points s_1, to s_{10}. The tree structure on the right-hand side of Figure 2 follows the containment hierarchy obtained from the data partitioning.

The nodes of the SAM trees contain information about the MBR that they cover, such as their coordinates. This kind of information grows exponentially with the number of dimensions, leading to the growth of each tree node and of the index itself. The larger the nodes are, the fewer can fit in a disk page; accessing such an index becomes more difficult. Another important issue is the high-overlapping between the MBR stored at the same level in the tree. Although they cannot be observed in our example because we have few data points and few dimensions, the overlappings lead to an increased number of branches to be searched. The costs of maintaining SAM structures cover aspects such as space, index re-creation, updates, insertions and MBR split managements.

4.2 Metric Access Methods

Like SAM, the Metric Access Methods (MAM) Chávez et al. (2001); Hjaltason & Samet (2003) are also based on tree-structures, but they work with relative distances between the objects rather than their absolute positions in space. MAM have gained an important role due to the fact that conventional SAM approaches stop being efficient in high-dimensional data. They are also required for search in distance-only data sets, i.e. those that cannot be mapped to vector spaces. An example thereof is a set of text documents that use the edit metric Yujian & Bo (2007) to measure the distance between documents.

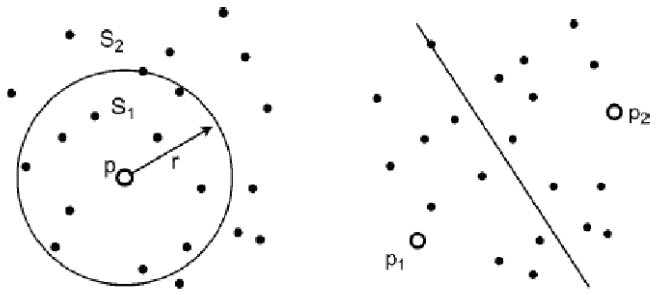


Fig. 3. Ball and Hyperplane partitioning; figure taken from Hjaltason & Samet (2003)

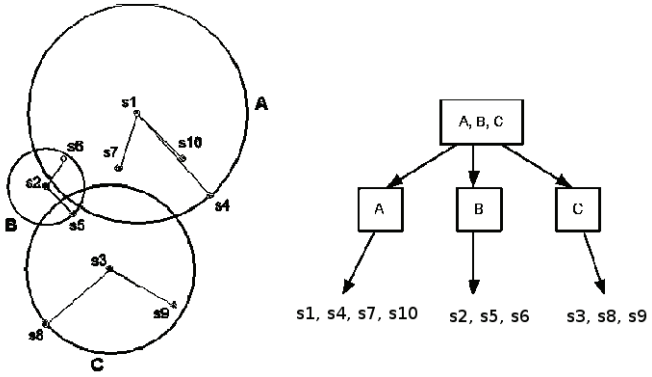


Fig. 4. An M-Tree example

MAM build their tree-structures by recursively partitioning the data set into subsets at each node level Chávez et al. (2001). Two main partitioning schemes have been identified by Hjaltson and Samet Hjaltason & Samet (2003): *ball partitioning* and *generalized hyperplane partitioning*, illustrated in Figure 3. With the *ball partitioning* approach, the data set is partitioned based on the distance from one specified object, called *vantage point* or *pivot*; two subsets are generated: the first subset inside the circle around the pivot, and the second subset outside the ball. Among ball partitioning trees the most referenced are *Vantage-Point Tree (VP-tree)*, *Multi-Vantage Point Tree (MVPT)* Bozkaya & Ozsoyoglu (1999), *Vantage-Point Forest (VPF)*, *Burkhard-Keller Tree (BKT)* and *Fixed Queries Tree (FQT)* Hjaltason & Samet (2003). With the *hyperplane partitioning* approach, at each step two points p_1 and p_2 are selected. Elements closer to p_1 than to p_2 go into the left sub tree and those closer to p_2 go into the right sub tree. Among hyperplane partitioning structures we enumerate *Bisector Tree (BST)*, *Generalize Hyperplane Tree (GHT)*, *Geometric Near-neighbor Access Tree (GNAT)*. and the *M-tree*Ciaccia et al. (1997).

An M-tree example for a small set of 10 objects is shown in Figure 4. The objects are stored in the leaf nodes, while the internal nodes, also called *routing objects*, store pointers to child nodes and the covering radius for the children they enclose.

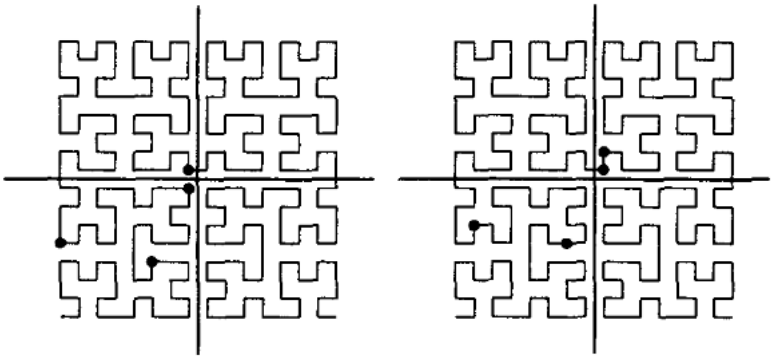


Fig. 5. Space filling curve; figure taken from Liao et al. (2001)

The applicability of the MAM methods ranges from "native" distance-only datasets to high-dimensional datasets for which conventional SAM are no longer efficient Digout & Nascimento (2005). Their advantage is that the relative distances between objects can be pre-computed at index creation time, avoiding heavy distance computations at search time. However, this advantage is also a constraint because the distance measure used at creation time must be used at search time. Given that the distance measure must be established in advance, searching with user-defined metrics that capture personalized criteria becomes difficult. It has been shown, however, that metrics from certain classes of parameterizable distance functions could be used Ciaccia & Patella (2002).

4.3 Single-dimension mapping

Single-dimension mapping approaches map the points in the high-dimensional space to single-dimensional values for which efficient techniques such as the B-tree Bayer & McCreight (1972) exist.

Querying high-dimensional data in single-dimensional space considers the smallest and largest values among all the dimensions of each data point Yu et al. (2004) and Ooi et al. (2000). Another single-dimensional mapping method sorts the data points according to their positions on a space-filling curve Liao et al. (2001). The list obtained in this way is stored in a B-tree structure. In practice, several shifted copies of the data points (maximum $N + 1$, where N is the space dimensionality) are used. Each copy of the data points produces a separate, differently ordered list, which is stored in a separate B-tree. The left part of Figure 5 illustrates a space filling curve that touches the original data points. On the right part, the same curve touches the shifted data with one unit up and one to the right.

With the "iDistance" approach, Jagadish et al. (2005) a data partitioning (see Section 4.1) technique is initially applied, followed by a single-dimensional mapping within each region. The mapping process consists of sorting the objects in each region on the distance to a specific reference point, such as the region's center. The reference points are then indexed in a $B^+ - tree$ structure. The "iDistance" has been applied on local subspaces Shen et al. (2007), previously obtained with principal components analysis Jolliffe (1986).

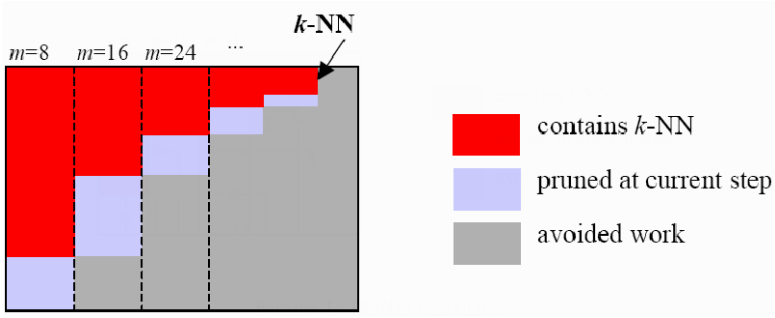


Fig. 6. Example of BOND technique; figure taken from Arjen et al. (2002)

4.4 Data decomposition

Data decomposition methods treat each dimension as a separate list, and their goal is to obtain the answer of the query by accessing a minimum number of lists and as few objects as possible in each of the visited lists.

Branch and Bound on Vertically Decomposed Data (BOND) Arjen et al. (2002), adopts vertical decomposition as storage organization. That means it decomposes the data into multiple tables, one for each dimension. Therefore, the information for a specific object is distributed into multiple tables. The algorithm accumulates the distances between the query object and all data vectors, by scanning the dimensional projections one-by-one. After processing a few dimensions, *partial* distances of k-nearest neighbors are exploited to discard safely from further consideration those vectors that cannot possibly participate in the result. This process is graphically illustrated in Figure 6. It can be observed that groups of 8 dimensions are successively scanned; m represents the total number of visited dimensions. After reading each group, based on partial distances, a set of objects is pruned. The iterative application of this process quickly reduces the candidate set (the upper part of the figure) to just a small database sample.

4.5 Data approximation structures

The methods described here are based on a result obtained by Weber et al. Weber et al. (1998), which indicate that MAM and SAM are outperformed by a simple sequential scan whenever the dimensionality is above 10. Considering this result, a sequential scan is inevitable when searching high-dimensional datasets. Under this assumption, the VA-file method Tesic & Manjunath (2003); Weber et al. (1998) constructs a vector of approximations, significantly smaller than the original data and sequentially scans it. The vector of approximations is the result of a space division in a number of cells; all the objects contained in a cell are represented by a common approximation. When searching for nearest neighbor for example, the entire vector of approximations is scanned. Based on their minimum/maximum distance to the query the majority of approximations are filtered. For the remaining ones the corresponding exact data points have to be accessed. The critical factor of VA-file' performance is the filtering step. If too many approximations remain, a lot of objects have to be accessed and

the advantage of an approximation file is lost. Discussions and improvements of the VA-file pruning strategy have been reported Wang & You (2006).

Also using a grid of cells, the IGrid Aggarwal & Yu (2000) maintains separate lists for the objects in each cell. The similarity between any two objects uses only the set of dimensions for which the two objects lie in the same range (the *proximity set*). The number of objects that are accessed is kept small as the dimensionality increases, but the storage overhead is 100% because all the objects are copied into the indexing structure. Bitmap IGrid variants have also been proposed Cha (2003); Goldstein et al. (2004).

The process of partitioning a high-dimensional space is itself a specific problem requiring dimension-wise discretization strategies Fayyad & Irani (1992). One possibility is to create dimension-wise histograms of the distance values from the points to the data center Jagadish, H.V. and Beng Chin Ooi and Heng Tao Shen and Kian-Lee Tan (2006) . The number of bins in each dimension's histogram gives the number of intervals. Another data approximation technique uses a partitioning strategy based on a graph model Aggarwal et al. (1999).

4.6 Which indexing method to use?

The SAM and MAM high-dimensional indexing methods divide the search space in a set of minimum bounding regions hierarchically organized in tree structures. At search time, the MBR that don't overlap the query region are filtered. As dimensionality increases, the distances from a query object to the nearest and the farthest objects are almost in the same range. Practically, the nearest neighbor becomes indistinctive from other neighbors Katayama & Satoh (2001), loosing its meaningfulness Aggarwal (2002); Beyer et al. (1999). In such conditions, the smallest query region that contains the nearest neighbor will overlap most of the MBR, making them non-prunable. The consequence is that the search methods end up accessing all the nodes of their structures in a pseudo-random fashion, which is more time consuming than a simple sequential scan —a problem often referred to as the "curse of dimensionality" Beyer et al. (1999); Indyk & Motwani (1998).

To overcome this problem, Katayama et al. have proposed a distinctive-sensitive approach for tree-based structures Katayama & Satoh (2001), testing the distinctiveness of the nearest neighbor in the course of search operation. When the nearest neighbor is considered indistinctive, search returns a partial result. Other indexing methods, such as the ones presented in Section 4.3, resort to single-dimension mapping. Another approach, presented in Section 4.4, uses specific aggregations with the goal of visiting only a fraction of the dataset. Finally, the data approximation methods sequentially access a compressed version of the data.

Choosing the proper index for a given situation should be the result of especially designed tests that consider various approaches. A given indexing approach could behave well in some situations and worse in others. For example, in Wang & You (2006) an important improvement over the original method in Weber et al. (1998) has been reported just by using a different metric. To our knowledge, test frameworks for the whole range of high-dimensional indexing methods are not yet publicly available. The currently available frameworks, such as GIST Hellerstein et al. (1995), SP-GIST Aref & Ilyas (2001) and XXL den Bercken et al. (2001) provide only a subset of the existing indexing methods. GIST provides a tree-based template indexing structure. XXL, while also offering tree-based indexing templates, focuses on implementations

of advanced database queries (cursors, iterators) independent from the underlying data types and data structures. MMDI Gonçalves et al. (2007) is built with the goal of supporting virtually any high-dimensional index. However, at the moment, only approaches that cannot be tested in GIST or XXL have been implemented.

4.7 Video retrieval requirements

Considering the variety of indexing approaches, it is important to have a set of requirements specific to the video indexing domain, against which any given index can be checked for compliance. The following set of requirements strengthen the need for flexibility of the high-dimensional indexes.

- Data preprocessing;
- Frequency of updates;
- Varying dimensionality;
- Any dimensional subspace;
- Support for different metrics;
- Support for weighted queries;
- Multiple query objects.

Data preprocessing can be a dominant effort when dealing with diverse descriptors and high-dimensional spaces. Tasks such as the normalization of the feature vectors and dimensional reduction may be required for index construction.

Many multimedia databases are statical in nature; this is the case with archives, where append is the common operation. But there are cases where updates are quite often. When new objects are added to the database, they have to be processed and included in the indexes. On the other and, if a new feature is extracted, or the quality of some feature extraction tool improves, existing objects will be processed and the new descriptors incorporated as an update.

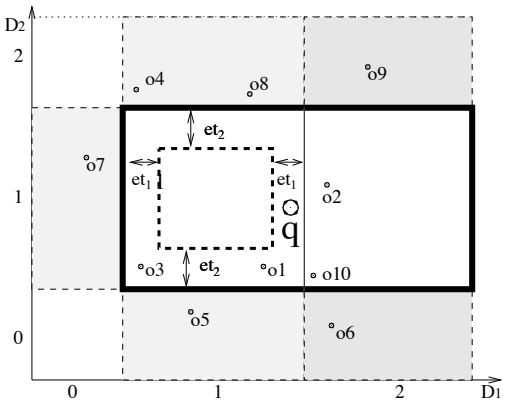
The dimensionality of the search space is directly related to the number of descriptors, as each descriptor contributes new dimensions. Adding a histogram descriptor, for instance, increases the dimensionality by the number of bins the histogram has been quantized (8, 16, 32, 256).

If the user is familiar with the descriptors that are being used, the interface may offer the choice of a subset of features at query time. The user may then use only color descriptors in one search and texture descriptors in another one, for instance.

Searching by different components of the feature space often implies using different metrics for each component. Aggregation is required in this case and it may be done in different ways, such as using arithmetic aggregation functions.

The user may want to stress the relevance of some features, and in this case weights can be used.

The query may be specified as a composition of various objects. This happens, for example, when relevance feedback is used and the user selects multiple objects as positive/negative examples after some exploratory search.



	D1			D2			Cardinalities	
	0	1	2	0	1	2	Naive	Exp
o1		1			1		2	2
o2			1		1		1	2
o3		1			1		2	2
o4		1				1	1	1
o5		1		1			1	1
o6			1	1			0	1
o7	1				1		1	1
o8		1				1	1	1
o9			1			1	0	1
o10			1		1		1	2
q_{Naive}		1			1			
q_{Exp}		1	1		1			

Fig. 7. BitMatrix

5. BitMatrix-based multidimensional indexing

In this section the focus is also maintained on high-dimensional indexing by presenting an indexing approach, BitMatrix, Calistru et al. (2006) that satisfies most of the requirements previously identified, and has been successfully applied in video retrieval.

The main idea behind the BitMatrix is to construct a collection of bitmap signatures that can be sequentially analyzed and processed with bitwise operations in order to prune the search space. Following a data approximation approach, in the spirit of VA-File Weber et al. (1998) and IGrid Aggarwal & Yu (2000), each of the N dimensions are partitioned in k ranges. A partition of a dimension D is a set of ranges

$$\pi_D = \{r_i = [l_i, u_i]\,,\, i = 1 \ldots k_D\}\,,$$

where l_i, u_i are the lower and upper bounds of range i. The partitioning scheme $(k_1, k_2, \ldots, k_N)$ is used to obtain bitmap signatures for all the objects in a dataset $\mathcal{O}$ arranged as lines in a matrix. For each dimension the signatures contain 1 for the range where the object belongs and 0 for the other ranges. The search algorithm selects objects based on the cardinality (number of bits set to 1) of the bitwise AND between object and query signatures. Only the objects that obtain scores above an established cardinality threshold are then exhaustively analyzed to compute their exact distance. Figure 7 illustrates a two-dimensional space having each dimension partitioned in three ranges. The 10 objects in Figure 7 have their signatures arranged in a BitMatrix; for simplicity, we ignore the zeros. For a given query object q, with signature q_{Naive}, the cardinality column $Naive$ contains the results of the bitwise AND between each of the 10 object signatures and q_{Naive}. In this example, if the cardinality threshold is set to 2, the search algorithm prunes object o_2, which happens to be the nearest neighbor. This happens because, for all the dimensions, only the objects in the same range as q are considered. However, this effect can be controlled by using modified query regions (expansions) and various cardinality thresholds. Expanding the query region along dimension D_1 corresponds to a change in the query signature from q_{Naive} to q_{Exp}. The cardinality column Exp contains the results of the bitwise AND between the 10 object signatures and q_{Exp}. With the same cardinality threshold, i.e. 2, o_2 becomes part of the selected

objects. In general, the variations in query signature and cardinality threshold control the trade-off between precision and speed.

5.1 The BitMatrix and the retrieval requirements

The BitMatrix index and the associated similarity search methods are proposed to solve the common requirements in multimedia retrieval.

5.1.0.2 Multiple query objects

The proposed method easily supports multiple query objects. This is a common requirement in multimedia retrieval in general, and in particular for supporting relevance feedback search iterations. The independent object signatures are merged into a common query signature which is used in the sequel to search the BitMatrix.

5.1.0.3 Subspace selection

The importance of each individual dimension of the search space has already been examined in works like BOND Arjen et al. (2002) and LDC Koudas et al. (2004). Similarly, the BitMatrix search algorithm can be applied on selected subspaces. The subspace selection feature is useful for speeding up the search in at least two situations. First, when the dimensions of the original search space are not of equal importance. They may be ordered by decreasing importance and only a first subset of them used in the search. The second situation arises when enough objects have been pruned after analyzing only some of the subspaces. The analysis of the remaining subspaces can thus be avoided and the search can stop earlier.

5.1.0.4 Index management: insert, update, delete

The insertion of a new object in the BitMatrix, accounts for computing its signature and adding it as a new line in the matrix. The size of the BitMatrix grows linearly with the number of objects and with the dimensionality. The precise size of the BitMatrix is $(\Sigma_{D=1}^{N} k_D) * |\mathcal{O}|$ bits. To update an existing object its signature has to be modified through bitwise operations. To delete an object, the corresponding line is removed from the matrix.

The BitMatrix index retains most of sequential scan's flexibility with good quality of the approximations and a much better time performance. It can be conveniently arranged for efficient sequential access and optimized bitwise operations. It can also be broken into segments for distributed or parallel processing.

5.2 Evaluation of the BitMatrix method

The BitMatrix evaluation has been performed on two datasets: a set of 9908 real image histograms having 256 dimensions, named the i9000 dataset, and a synthetic dataset, i10000, of 10000 objects independent and identically-distributed in all of its 80 dimensions. The dimensions partitioning strategy for both datasets was 7 ranges per dimension, where the range computation is based on k-means clustering ($k = 7$).

5.2.1 Recall performance

The experiments were geared towards observing the BitMatrix performance for nearest neighbor search ($NN(q)$) and k-nearest neighbors search ($NN_k(q)$). To illustrate the $NN_k(q)$

	i9000 256 dimensions, 7 ranges/dimension							
	$NN(q)$				$NN_{10}(q)$			
ct	Naive(et=0)		**et=0,01**		Naive(et=0)		**et=0,01**	
	Recall	accessed	Recall	accessed	Recall	accessed	Recall	accessed
0.73	0.6	0.2%	0.7	0.4%	0.39	0.2%	0.48	0.4%
0.67	0.78	0.8%	0.87	1.0%	0.61	0.8%	0.68	1.0%
0.55	0.93	2.9%	0.95	3.8%	0.86	2.9%	0.9	3.8%
0.47	**0.97**	**6.0%**	**0.99**	**7.4%**	**0.91**	**6.0%**	**0.94**	**7.4%**
0.40	**1.0**	**10.9%**	**1.0**	**13.5%**	**0.93**	**10.9%**	**0.96**	**13.5%**
	i10000 80 dimensions, 7 ranges/dimension							
	$NN(q)$				$NN_{10}(q)$			
	Naive (et=0)		**et=0,01**		Naive (et=0)		**et=0,01**	
0.60	0.24	0.19%	0.25	0.21%	0.08	0.19%	0.09	0.21%
0.50	0.55	1.99%	0.57	2.31%	0.33	1.99%	0.35	2.31%
0.40	0.78	8.16%	0.84	9.32%	0.57	8.16%	0.62	9.32%
0.35	0.90	17.0%	0.93	19.2%	0.77	17.0%	0.80	19.2%
0.30	0.90	32.2%	0.94	34.21%	0.85	32.2%	0.87	34.21%

Table 1. Testing the BitMatrix on two datasets

search performance, we compare the approximate BitMatrix results with the exact *k-nearest neighbors*. We use the recall rate for this purpose. Assuming R as the $NN_k(q)$ set and A as the set of approximate neighbors obtained with the BitMatrix, the recall rate is

$$\frac{|A \cap R|}{|R|}.$$

Beside the recall rate scores, we also record the percentage of objects that are effectively accessed, i.e that remain after the pruning phase. This is expressed as the ratio of $|A|$ to the size of the dataset: $\frac{|A|}{sizeofdataset}$.

Table 1 shows average values for the two measures (recall rate, and % of objects accessed) with respect to $NN(q)$ and $NN_{10}(q)$ across random sets of 100 queries. The cardinality thresholds are in the first column. With a cardinality threshold of 0.55 for instance, less than 3% of i9000 is accessed, the average recall rate is 0.93 (relative to NN) and 0.86 (relative to NN_{10}). The experiments have shown that the trade-off between quality of retrieval and speed can be tuned with the available expansion mechanism. For example, with the cardinality threshold 0.47, about 6% of i9000 is accessed, and the recall rate relative to NN_{10} is 0.91. If range expansion is performed the NN_{10} recall becomes 0.94 at 7.4% accessed objects, while for a smaller cardinality threshold (ct = 0.4) the NN_{10} recall is 0.93 at 10.9% accessed objects. With an increase in recall (0.94 vs 0.93) and less 3.5% (7.4% vs 10.9%) objects accessed, expansion should be preferred in this case.

The results for the second dataset, i10000, are presented in the second half of Table 1. The numbers indicate lower performance than on i9000. Much larger amounts of the i10000

have to be analyzed in order to obtain acceptable recall rates. Note however that i10000 is a synthetic dataset independent and identically-distributed (uniform distribution) across all the dimensions, thus not a realistic one. The expansion mechanism clearly improves the recall rate in this case as well.

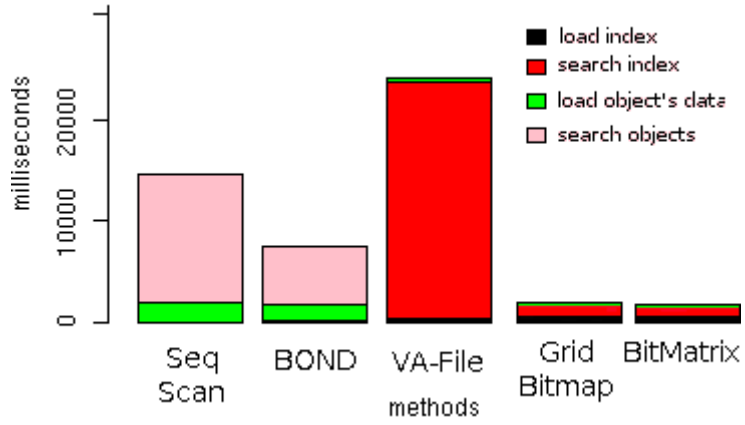


Fig. 8. Comparing methods that use the euclidean metric

5.2.2 Time performance

Another set of experiments have been designed to observe the time performance of the BitMatrix as a memory-based indexing method. A specific framework for high-dimensional indexing evaluation has been used Gonçalves et al. (2007), and the experiments were run on i9000 dataset. The MMDI allowed us to test a set of 5 methods: Sequential Scan, Bond, VA-File, GridBitmap, and BitMatrix. The time columns in Figure 8 have four components: the time to load the index in memory (if such an index exists), the time to search it, the time to load objects data, and the time to search them. The total time for the BitMatrix is clearly smaller than the values for Sequential Scan and Bond and is in the same range as the GridBitmap. The VA-file's time is not favorable to the method as it is tested using the same partitioning scheme as GridBitmap and BitMatrix; with 7 ranges in each of the 256 dimensions, there are 7^{256} approximation cells, with the non-empty ones having at most one object. Thus, VA-file has to access much more cells than real objects.

BitMatrix has been also evaluated in an independent multimedia retrieval test Acar et al. (2008), where it has been shown to outperform a slim-tree index.

6. BitMatrix-based video retrieval—A case study

In this section we report on the implementation of a BitMatrix-based video retrieval system. The experiments have been performed on the TRECVID 2007 dataset, which consists of 100 hours of videos from the BBC and the Danish national television.

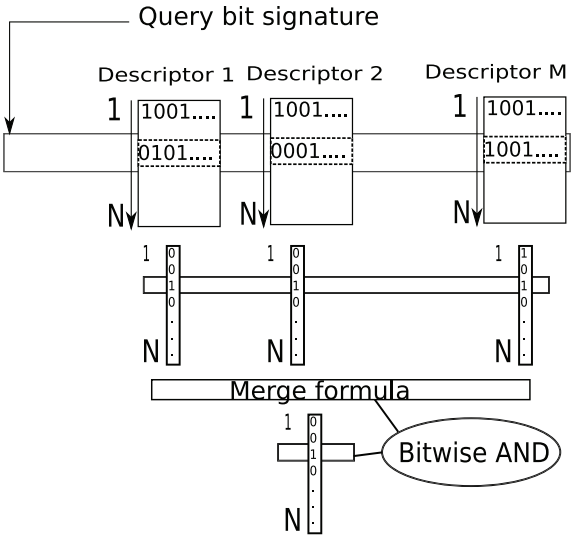


Fig. 9. Answer computation

6.1 Features and descriptors

This case study integrates the whole range of available descriptors, from low- to high-level ones. The list of low-level descriptors includes ColorLayout, ColorStructure, ScalableColor and ColorMoments for color, EdgeHistogram, Homogeneous Texture, Wavelet texture and Haralick for texture, RegionShape for shape, Scale Invariant Feature Transform Lowe (2003) (for interest points, Spectral Centroid, Spectral Rollof Point, Spectral Flux, Compactness, Spectral Variability, Root Mean Square, Fraction of Low Energy Windows, Zero Crossings, Strongest Beat, Beat Sum, Strength of Strongest Beat, Mel Frequency Cepstral Coefficient, Linear Predictive Coding and Area Method of Moments for audio.

The high-level features, such as the concepts automatically obtained with special detectors Jiang et al. (2007) have been used to create a high-level descriptor, named the "Concepts" descriptor. The "Concepts" descriptor construction is based on a 39-dimensional concept space —a vector space where each concept is a dimension. The concept space $C^n = \{c_1, c_2, .., c_n\}$, where c_i is a dimension corresponding to the i^{th} concept and n is the total number of concepts. Each of the annotated objects can be represented as a vector: $(v_1, v_2, .., v_n)$, where each v_i represents the value for the i^{th} conceptual dimension.

With this representation, the "Concepts" descriptor can be indexed with multidimensional techniques. This allows the use the BitMatrix for the entire range of low and high-level descriptors.

6.2 Indexing and search

The indexing strategy has been to build descriptor-wise BitMatrix indexes off-line. This allows an independent analysis of several similarity facets and an experimental evaluation of the contribution of each descriptor to the retrieval tasks. The answer of a query-by-example

Fig. 10. Color-based results

with respect to the color feature, for example, can be obtained with several color descriptors, each one showing a different facet of color similarity. Their usefulness for the search topics is evaluated empirically.

The overall search process fits into the query-by-example paradigm, where the answer computation is performed with respect to a combination of descriptors. Note however, that when the "Concepts" descriptor is included in the combination , the search paradigm becomes a combination of a typical query-by-example and a query-by-semantic-example Rasiwasia et al. (2006). For example, the search can start with an image or video shot of a natural scene and expect similar video items with respect to the ColorLayout, ColorStructure (color), EdgeHistogram (texture), and "Concepts" descriptors.

Starting with a set of example objects, the first step is to translate their feature vectors into a bit signature and then pass it to the BitMatrix-based answer computation step, as illustrated in Figure 9. In the sequel, several binary lists, one for each selected descriptor, are obtained by searching the corresponding BitMatrix indexes. We refer to these lists as binary lists because, as shown in Section 5, the objects that obtain scores above an established cardinality threshold are marked with 1, while the others are marked with 0.

Finally, the binary lists are aggregated with merge formulas. For example, a merge formula may count the number of binary lists in which each object appears; weights can also be assigned to each list. Figure 9 illustrates a merge formula that performs a bitwise AND between corresponding positions of each binary list. As all have their third positions set to 1,

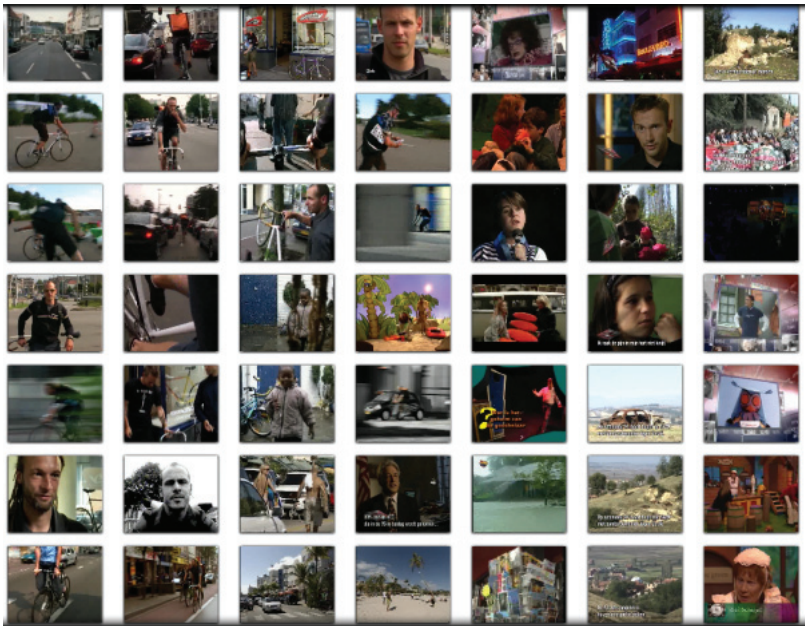


Fig. 11. “Concepts” and audio-based results

the bitwise AND yields a 1 in the final result set, meaning that the third object will be among the retrieved objects.

Note that instead of constructing binary lists, the object cardinalities, or even their exact distances to the query objects can be recorded. In such a case, the lists aggregation can be achieved by implementing rank-aggregation approaches, such as Fagin et al. (2003) and Güntzer et al. (2000). The preference for binary lists is motivated by the lack of distinctiveness between the scores of the nearest neighbors in high-dimensional spaces Katayama & Satoh (2001). Moreover such binary lists can be aggregated with bitwise operations, which improves the overall efficiency.

6.3 Parametrization

The cardinality thresholds used to obtain the binary lists were empirically determined by performing several query-by-example iterations per descriptor. If high thresholds are used, few objects are returned, while the use of small thresholds allows too many objects to appear in the answer set. Cardinality thresholds were considered acceptable when allowing a number of objects in the 500-700 range—an appropriate number of objects to be visually inspected in a user interface.

The merge formula for the binary list aggregation has been the bitwise AND, which can be efficiently computed with bitwise operations and sequential access.

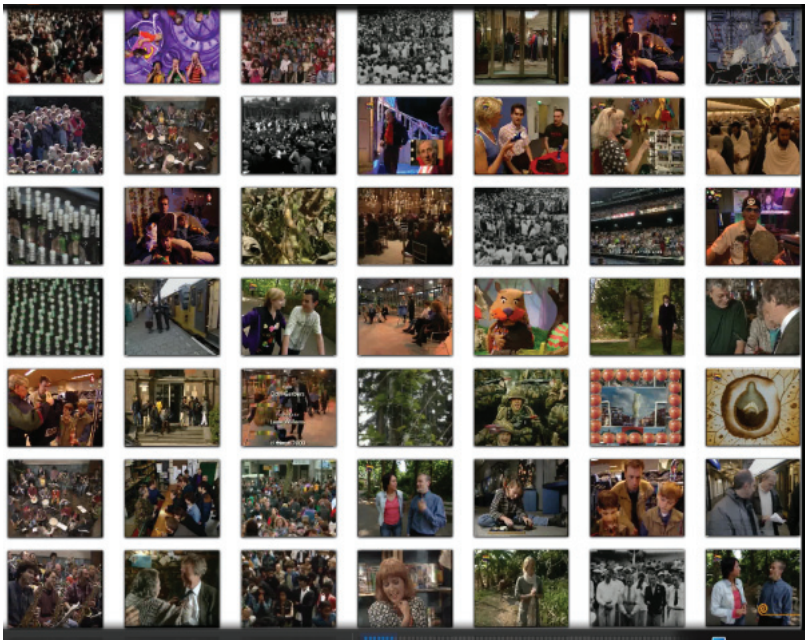


Fig. 12. Texture-based results

6.4 Search results

Search experiments were performed on top of the TRECVID 2007 dataset, containing educational, cultural and youth-oriented programs, news magazines and historical footage videos. Beside the great variety of subject matter, the video material has been primarily in Dutch, without repetitive parts such as commercials, or repeated news footage.

While the TRECVID community is mainly focused on benchmarking the effectiveness of video retrieval, the goal here was to tackle the retrieval problem from both effectiveness and efficiency aspects.

Figures 10, 11 and 12 show several result sets consisting of video shots from the TRECVID 2007 dataset. Figure 10 shows the results of a color-based query-by-example; all the color descriptors enumerated in Section 6.1 are involved. Figure 11 shows the results for the "Find shots of a people walking or riding a bicycle" query; the search started with video samples of people walking or riding bicycles and involved the "Concepts" and all the audio descriptors. Figure 12 shows the results of a texture-based query-by-example search; the query image is the one in the upper left corner of Figure 12 and the answer was obtained using all the texture descriptors mentioned earlier: Edge Histogram, Haralick texture, Homogeneous Texture, Region Shape, and Wavelet Texture.

The quality of the retrieval is evaluated by calculating average precision results. Four components are typically required: a fixed number of multimedia objects to be searched on, a fixed number of topics from which the queries will be chosen, an evaluation criteria such as

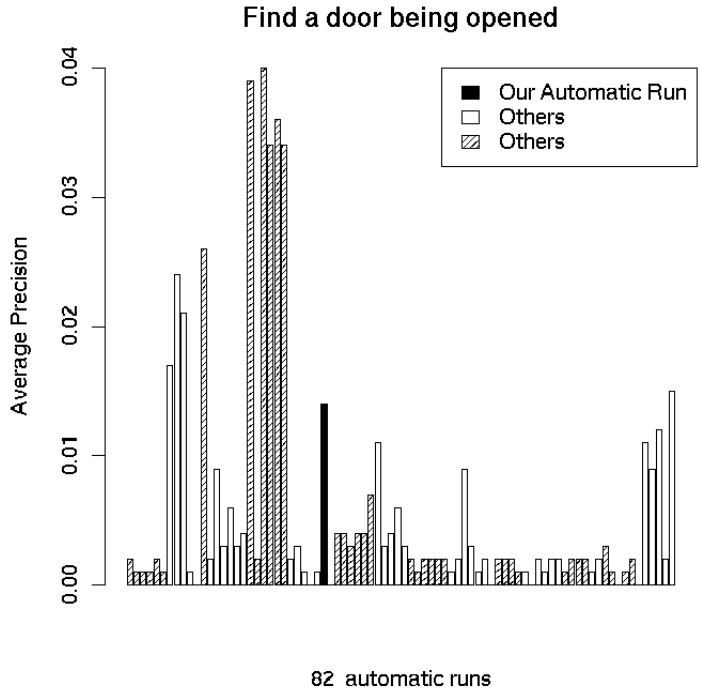


Fig. 13. Average precision results

the precision/recall measure, and the relevance judgments or ground truth; TRECVID offered them all. The average precision was measured per-topic, based on the positions of the relevant documents. The precision is measured at every rank at which a relevant document is obtained and then averaged over all relevant documents to obtain the average precision:

$$AP = \frac{\sum_{r=1}^{N} P(r) \times relevant(r)}{size\ of\ ground\ truth}.$$

r is the rank, N is the number of retrieved documents, $relevant()$ is a binary function on the relevance at a given rank, and $P()$ is the precision at that rank. Figures 13 and 14 illustrate the average precision results obtained for two topics, namely "Find shots of a door being opened" and "Find shots of a waterfront with water and buildings".

Overall, the precision results indicate that our retrieval system's effectivity was comparable to the others. However, by indexing the whole spectrum of descriptors with the same index type—the BitMatrix—we have been able to speed up the query computation, making a step further towards a database-ready retrieval system.

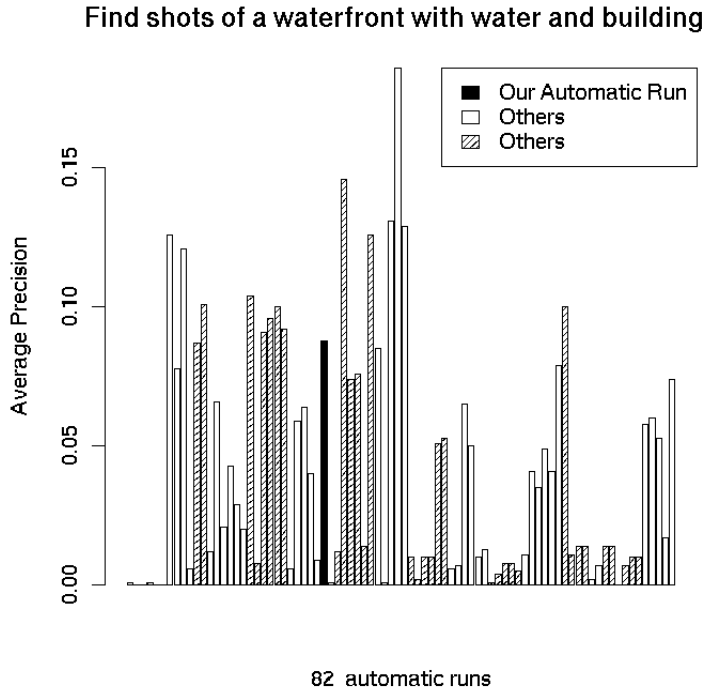


Fig. 14. Average precision results

7. Summary

Representing, managing and retrieving video data is required in a wide range of application domains. Current tools for video analysis are time consuming and can be very time consuming on large datasets. With the growth of multimedia mass production we must shift from simple collection paradigms to well-organized multimedia repositories.

Existing multidimensional indexing methods are not yet prepared for the variety of features and similarity models in multimedia objects. In order to work with as many feature representations and similarity models as possible, the high-dimensional indexing techniques should at least be able to access individual dimensions, should easily incorporate new objects as well as new sets of dimensions. Flexible ranking mechanisms that capture user preferences are also an important requirement.

The indexing structure that we have presented, the BitMatrix, is a highly parameterizable index structure offering a large space for experimentation: similarity threshold, number of dimensions processed in each step, and dimension processing order for the case of weighted dimensions. It also supports multiple query objects at a time, which is especially useful for relevance feedback. The BitMatrix index retains most of sequential scan's flexibility with good quality of the approximations and a much better time performance. It can be conveniently

arranged for efficient sequential access and optimized bitwise operations. It can also be broken into segments for distributed or parallel processing.

However, when application requirements are not that tough, other type of indexing methods may be appropriate. For example, if a specific metric is established, or when the significance of individual dimensions can be lost, metric-based access methods, such as LSH Datar et al. (2004) or iDistance Jagadish et al. (2005) can be used. As we have stated in Section 4.6, a given indexing approach could behave better in some situations than others.

The proposed BitMatrix index is expected to adapt well to the distributed and parallel paradigms, allowing its use in software frameworks for parallel computations over large data sets, such as "MapReduce" Dean & Ghemawat (2008). MapReduce allows to split an application among a set of machines by dividing the job into Map and Reduce parts. The BitMatrix search algorithm can have a Map phase, taking the initial bit matrix, splitting it into smaller bit matrices, and sending the parts to different machines—so all would be searched at the same time. The Reduce phase will then combine the partial results to get a single answer set. This approach will allow larger amounts of high-dimensional data to be efficiently searched, in a scalable manner.

8. References

Acar, E., Arslan, S., Yazici, A. & Koyuncu, M. (2008). Slim-tree and BitMatrix index structures in image retrieval system using MPEG-7 Descriptors, *Content-Based Multimedia Indexing, 2008. CBMI 2008. International Workshop on* pp. 402–409.

Aggarwal, C. C. (2002). Towards meaningful high-dimensional nearest neighbor search by human-computer interaction, *Data Engineering, 2002. Proceedings. 18th International Conference on*, pp. 593–604.

Aggarwal, C. C., Hinneburg, A. & Keim, D. A. (2001). On the surprising behavior of distance metrics in high dimensional space, *Lecture Notes in Computer Science* 1973: 420–434. URL: *citeseer.ist.psu.edu/aggarwal01surprising.html*

Aggarwal, C. C., Wolf, J. L. & Yu, P. S. (1999). A new method for similarity indexing of market basket data, *SIGMOD '99: Proceedings of the 1999 ACM SIGMOD international conference on Management of data*, ACM Press, New York, NY, USA, pp. 407–418.

Aggarwal, C. C. & Yu, P. S. (2000). The IGrid index: reversing the dimensionality curse for similarity indexing in high dimensional space, *KDD '00: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM Press, New York, NY, USA, pp. 119–129.

Aref, W. G. & Ilyas, I. F. (2001). SP-GiST: An Extensible Database Index for Supporting Space Partitioning Trees, *J. Intell. Inf. Syst.* 17(2-3): 215–240.

Arjen, P. d. V., Mamoulis, N., Nes, N. & Kersten, M. (2002). Efficient k-NN search on vertically decomposed data, *SIGMOD '02: Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, ACM Press, pp. 322–333.

Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R. & Wu, A. Y. (1998). An optimal algorithm for approximate nearest neighbor searching fixed dimensions, *J. ACM* 45(6): 891–923.

Balke, W.-T. & Güntzer, U. (2004). Multi-objective query processing for database systems., *VLDB*, pp. 936–947.

Bartolini, I. & Ciaccia, P. (2005). Optimal incremental evaluation of preference queries based on ranked sub-queries., *SEBD*, pp. 308–315.

Bartolini, I., Ciaccia, P., Oria, V. & Özsu, M. T. (2005). Integrating the results of multimedia sub-queries using qualitative preferences, *SEBD*, pp. 308–315.

Bayer, R. & McCreight, E. M. (1972). Organization and maintenance of large ordered indices, *Acta Informatica.* 1: 173–189.

Beckmann, N., Kriegel, H.-P., Schneider, R. & Seeger, B. (1990). The r*-tree: an efficient and robust access method for points and rectangles, *SIGMOD Rec.* 19(2): 322–331.

Benitez, A. B. & Chang, S.-F. (2002). Multimedia knowledge integration, summarization and evaluation., *MDM/KDD*, pp. 39–50.

Berchtold, S., Keim, D. A. & Kriegel, H.-P. (1996). The X-tree: An index structure for high-dimensional data, *in* T. M. Vijayaraman, A. P. Buchmann, C. Mohan & N. L. Sarda (eds), *Proceedings of the 22nd International Conference on Very Large Databases*, Morgan Kaufmann Publishers, San Francisco, U.S.A., pp. 28–39.
URL: *citeseer.ist.psu.edu/berchtold96xtree.html*

Beyer, K., Goldstein, J., Ramakrishnan, R. & Shaft, U. (1999). When Is "Nearest Neighbor" Meaningful?, *Lecture Notes in Computer Science* 1540: 217–235.
URL: *citeseer.ist.psu.edu/article/beyer99when.html*

Bober, M. (Jun 2001). Mpeg-7 visual shape descriptors, *Circuits and Systems for Video Technology, IEEE Transactions on* 11(6): 716–719.

Böhm, C., Berchtold, S. & Keim, D. A. (2001). Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases, *ACM Comput. Surv.* 33(3): 322–373.

Bohm, C., Gruber, M., Kunath, P., Pryakhin, A. & Schubert, M. (2007). Prover: Probabilistic video retrieval using the gauss-tree, *Data Engineering, International Conference on* 0: 1521–1522.

Böhm, C., Kriegel, H.-P. & Seidl, T. (2001). Adaptable similarity search using vector quantization, *DaWaK '01: Proceedings of the Third International Conference on Data Warehousing and Knowledge Discovery*, Springer-Verlag, London, UK, pp. 317–327.

Börzsönyi, S., Kossmann, D. & Stocker, K. (2001). The skyline operator, *Proceedings of the 17th International Conference on Data Engineering*, IEEE Computer Society, Washington, DC, USA, pp. 421–430.

Bozkaya, T. & Ozsoyoglu, M. (1999). Indexing large metric spaces for similarity search queries, *ACM Trans. Database Syst.* 24(3): 361–404.

Bulterman, D. C. A. (2004). Is It Time for a Moratorium on Metadata?, *IEEE Multimedia* 11(4): 10–17.

Burghouts, G. J. & Geusebroek, J.-M. (2009). Performance evaluation of local colour invariants, *Comput. Vis. Image Underst.* 113(1): 48–62.

Calistru, C., Ribeiro, C. & David, G. (2006). Multidimensional Descriptor Indexing: Exploring the BitMatrix., *in* Sundaram et al. (2006), pp. 401–410.

Carson, C., Thomas, M., Belongie, S., Hellerstein, J. M. & Malik, J. (1999). Blobworld: A system for region-based image indexing and retrieval, *Third International Conference on Visual Information Systems*, Springer.
URL: *citeseer.ist.psu.edu/carson99blobworld.html*

Celma, Ò. & Serra, X. (2008). Foafing the music: Bridging the semantic gap in music recommendation, *J. Web Sem.* 6(4): 250–256.

Cha, G.-H. (2003). Bitmap indexing method for complex similarity queries with relevance feedback, *MMDB '03: Proceedings of the 1st ACM international workshop on Multimedia databases*, ACM Press, New York, NY, USA, pp. 55–62.

Chaudhuri, S., Ramakrishnan, R. & Weikum, G. (2005). Integrating db and ir technologies: What is the sound of one hand clapping?, *CIDR*, pp. 1–12.

Chávez, E., Navarro, G., Baeza-Yates, R. & Marroquín, J. L. (2001). Searching in metric spaces, *ACM Comput. Surv.* 33(3): 273–321.

Chomicki, J. (2002). Querying with Intrinsic Preferences, *EDBT '02: Proceedings of the 8th International Conference on Extending Database Technology*, Springer-Verlag, London, UK, pp. 34–51.

Chomicki, J. (2003). Preference formulas in relational queries, *ACM Trans. Database Syst.* 28(4): 427–466.

Ciaccia, P. & Patella, M. (2002). Searching in metric spaces with user-defined and approximate distances, *ACM Trans. Database Syst.* 27(4): 398–437.

Ciaccia, P., Patella, M. & Zezula, P. (1997). M-tree: An efficient access method for similarity search in metric spaces, *VLDB '97: Proceedings of the 23rd International Conference on Very Large Data Bases*, Morgan Kaufmann Publishers Inc., pp. 426–435.

Datar, M., Immorlica, N., Indyk, P. & Mirrokni, V. S. (2004). Locality-sensitive hashing scheme based on p-stable distributions, *SCG '04: Proceedings of the twentieth annual symposium on Computational geometry*, ACM Press, New York, NY, USA, pp. 253–262.

Datta, R., Joshi, D., Li, J. & Wang, J. Z. (2008). Image retrieval: Ideas, influences, and trends of the new age, *ACM Computing Surveys* to appear: to appear.
URL: *http://infolab.stanford.edu/ wangz/project/imsearch/review/JOUR/datta.pdf*

Dean, J. & Ghemawat, S. (2008). Mapreduce: simplified data processing on large clusters, *Commun. ACM* 51(1): 107–113.

den Bercken, J. V., Blohsfeld, B., Dittrich, J.-P., Krämer, J., Schäfer, T., Schneider, M. & Seeger, B. (2001). XXL - A Library Approach to Supporting Efficient Implementations of Advanced Database Queries, *VLDB '01: Proceedings of the 27th International Conference on Very Large Data Bases*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 39–48.

Deselaers, T., Keysers, D. & Ney, H. (2004). Features for image retrieval – a quantitative comparison, *DAGM 2004, Pattern Recognition, 26th DAGM Symposium*, LNCS, Tbingen, Germany.
URL: *citeseer.ist.psu.edu/deselaers04features.html*

Digout, C. & Nascimento, M. A. (2005). High-dimensional similarity searches using a metric pseudo-grid., *ICDE Workshops 1174*.

Dimitrova, N. (2004). Context and memory in multimedia content analysis, *IEEE MultiMedia* 11(3): 7–11.

Eidenberger, H. (2003). Distance measures for MPEG-7-based retrieval, *MIR '03: Proceedings of the 5th ACM SIGMM international workshop on Multimedia information retrieval*, ACM Press, New York, NY, USA, pp. 130–137.

Eidenberger, H. & Breiteneder, C. (2003). Visual similarity measurement with the feature contrast model, *in* M. M. Yeung, R. W. Lienhart & C.-S. Li (eds), *Proceedings SPIE Storage and Retrieval for Media Databases Conference*, Vol. 5021, SPIE, pp. 64–76.

Eidenberger, H.; Breiteneder, C. (9-11 Dec. 2002). An experimental study on the performance of visual information retrieval similarity models, *Multimedia Signal Processing, 2002 IEEE Workshop on* pp. 233–236.

Ekin, A., Tekalp, A. M. & Mehrotra, R. (2004). Integrated Semantic-Syntactic Video Modeling for Search and Browsing, *IEEE Transactions on Multimedia* 6(6): 839–851.

Fagin, R., Kumar, R. & Sivakumar, D. (2003). Efficient similarity search and classification via rank aggregation, *SIGMOD '03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, ACM Press, New York, NY, USA, pp. 301–312.

Faloutsos, C. & Oard, D. W. (1995). A survey of information retrieval and filtering methods, *Technical report*, Univ. of Maryland Institute for Advanced Computer Studies Report, College Park, MD, USA.

Fayyad, U. M. & Irani, K. B. (1992). On the handling of continuous-valued attributes in decision tree generation, *Mach. Learn.* 8(1): 87–102.

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

Gentner, D. (1988). Structure-mapping: A theoretical framework for analogy, *in* A. Collins & E. E. Smith (eds), *Readings in Cognitive Science: A Perspective from Psychology and Artificial Intelligence*, Kaufmann, San Mateo, CA, pp. 303–310.

Gevers, T. & Smeulders, A. W. M. (2004). *Content-Based Image Retrieval: An Overview*, IMSC Press Multimedia Series, 1st edn, Prentice Hall.

Godfrey, P., Shipley, R. & Gryz, J. (2005). Maximal vector computation in large data sets, *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, VLDB Endowment, pp. 229–240.

Goldstein, J., Platt, J. C. & Burges, C. J. C. (2004). Redundant Bit Vectors for Quickly Searching High-Dimensional Regions., *Deterministic and Statistical Methods in Machine Learning*, pp. 137–158.

Gonçalves, B., Calistru, C., Ribeiro, C. & David, G. (2007). An Evaluation Framework for Multidimensional Multimedia Descriptor Indexing, *Workshop on Multimedia Databases and Data Management*.

Gudivada, V. N. & Raghavan, V. V. (1995). Content-based image retrieval systems, *Computer* 28(9): 18–22.

Güntzer, U., Balke, W.-T. & Kießling, W. (2000). Optimizing multi-feature queries for image databases, *in* A. E. Abbadi, M. L. Brodie, S. Chakravarthy, U. Dayal, N. Kamel, G. Schlageter & K.-Y. Whang (eds), *VLDB 2000, Proceedings of 26th International Conference on Very Large Data Bases, September 10-14, 2000, Cairo, Egypt*, Morgan Kaufmann, pp. 419–428.

Hanjalic, A. (2006). Extracting moods from pictures and sounds: towards truly personalized tv, *Signal Processing Magazine, IEEE* 23(2): 90–100.
URL: *http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1621452*

Hauptmann, A., Yan, R. & Lin, W.-H. (2007). How many high-level concepts will fill the semantic gap in news video retrieval?, *CIVR '07: Proceedings of the 6th ACM international conference on Image and video retrieval*, ACM, New York, NY, USA, pp. 627–634.

Hellerstein, J. M., Naughton, J. F. & Pfeffer, A. (1995). Generalized search trees for database systems., *VLDB*, pp. 562–573.

Hjaltason, G. R. & Samet, H. (2003). Index-driven similarity search in metric spaces, *ACM Trans. Database Syst.* 28(4): 517–580.

Howarth, P. & Rüger, S. (2005a). *Image and Video Retrieval*, Vol. http://www.springerlink.com/content/eq86j0456l73ax78, Springer Berlin / Heidelberg, chapter Trading Precision for Speed: Localised Similarity Functions, pp. 415–424.

Howarth, P. & Rüger, S. M. (2005b). Fractional distance measures for content-based image retrieval, *ECIR*, pp. 447–456.

H.R.Turtle (1991). *Inference Networks for Document Retrieval*, PhD thesis, University of Massachusetts.
URL: *citeseer.ist.psu.edu/turtle91inference.html*

Huijbregts, M., Ordelman, R. & de Jong, F. (2007). Annotation of heterogeneous multimedia content using automatic speech recognition, *Proceedings of the second international conference on Semantics And digital Media Technologies (SAMT)*, Lecture Notes in Computer Science, Springer Verlag, Berlin.

IBM (2008). Marvel: Multimedia Analysis and Retrieval, http://mp7.watson.ibm.com/marvel/.

Indyk, P. & Motwani, R. (1998). Approximate nearest neighbors: towards removing the curse of dimensionality, *Proc. of 30th STOC*, pp. 604–613.
URL: *citeseer.ist.psu.edu/article/indyk98approximate.html*

Ishikawa, Y., Subramanya, R. & Faloutsos, C. (1998). Mindreader: Querying databases through multiple examples, *VLDB '98: Proceedings of the 24rd International Conference on Very Large Data Bases*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 218–227.

Jagadish, H. V., Ooi, B. C., Tan, K.-L., Yu, C. & Zhang, R. (2005). iDistance: An adaptive B+-tree based indexing method for nearest neighbor search, *ACM Trans. Database Syst.* 30(2): 364–397.

Jagadish, H.V. and Beng Chin Ooi and Heng Tao Shen and Kian-Lee Tan (2006). Toward Efficient Multifeature Query Processing, *Knowledge and Data Engineering, IEEE Transactions on* 18(03): 350–362.

Jiang, W., Chan, K. L., Li, M. & Zhang, H. (2005). Mapping low-level features to high-level semantic concepts in region-based image retrieval, *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2*, IEEE Computer Society, Washington, DC, USA, pp. 244–249.

Jiang, Y.-G., Ngo, C.-W. & Yang, J. (2007). Towards optimal bag-of-features for object categorization and semantic video retrieval., *in* N. Sebe & M. Worring (eds), *CIVR*, ACM, pp. 494–501.
URL: *http://dblp.uni-trier.de/db/conf/civr/civr2007.html#JiangNY07*

Jolliffe, I. T. (1986). *Principal component analysis*, Springer-Verlag.

Katayama, N. & Satoh, S. (2001). Distinctiveness-Sensitive Nearest Neighbor Search for Efficient Similarity Retrieval of Multimedia Information, *Proceedings of the 17th International Conference on Data Engineering*, IEEE Computer Society, Washington, DC, USA, pp. 493–502.

Kemp, C., Bernstein, A. & Tenenbaum, J. B. (2005). A generative theory of similarity, *The Twenty-Seventh Annual Conference of the Cognitive Science Society*.

Kherfi, M. L., Ziou, D. & Bernardi, A. (2004). Image retrieval from the world wide web: Issues, techniques, and systems, *ACM Comput. Surv.* 36(1): 35–67.

Kosch, H., Boszormenyi, L., Doller, M., Libsie, M., Schojer, P. & Kofler, A. (2005). The Life Cycle of Multimedia Metadata, *IEEE Multimedia* 12(1): 80–86.

Koudas, N., Ooi, B. C., Shen, H. T. & Tung, A. K. H. (2004). Ldc: Enabling search by partial distance in a hyper-dimensional space, *ICDE '04: Proceedings of the 20th International Conference on Data Engineering*, IEEE Computer Society, Washington, DC, USA, p. 6.

Kraaij, W. (2005). Variations on language modeling for information retrieval, *SIGIR Forum* 39(1): 61.

Larson, R. R., McDonough, J., O'Leary, P., Kuntz, L. & Moon, R. (1996). Cheshire II: designing a next-generation online catalog, *J. Am. Soc. Inf. Sci.* 47(7): 555–567.

Leubner, A. & Kießling, W. (2002). Personalized keyword search with partial-order preferences., *SBBD*, pp. 181–193.

Li, H., Shi, R., Chen, W. & Shen, I. (2006). Image tangent space for image retrieval, *ICPR06*, pp. II: 1126–1130.

Liao, S., Lopez, M. A. & Leutenegger, S. T. (2001). High dimensional similarity search with space filling curves, *Proceedings of the 17th International Conference on Data Engineering*, IEEE Computer Society, Washington, DC, USA, pp. 615–622.

Lowe, D. (2003). Distinctive image features from scale-invariant keypoints, *International Journal of Computer Vision*, Vol. 20, pp. 91–110.
URL: *citeseer.ist.psu.edu/lowe04distinctive.html*

Macdonald, C. & Ounis, I. (2006). Searching for expertise using the terrier platform, *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM, New York, NY, USA, pp. 732–732.

Manjunath, B., Ohm, J.-R., Vasudevan, V. & Yamada, A. (2001). Color and texture descriptors, *Circuits and Systems for Video Technology, IEEE Transactions on* 11(6): 703–715.

Mufit Ferman, A., Krishnamachari, S., Murat Tekalp, A., Abdel-Mottaleb, M. & Mehrotra, R. (2000). Group-of-frames/pictures color histogram descriptors for multimedia applications, *Image Processing, 2000. Proceedings. 2000 International Conference on* 1: 65–68 vol.1.

Naphade, M., Smith, J. R., Tesic, J., Chang, S.-F., Hsu, W., Kennedy, L., Hauptmann, A. & Curtis, J. (2006). Large-Scale Concept Ontology for Multimedia, *IEEE MultiMedia* 13(3): 86–91.

Nguyen, P. H. P. & Corbett, D. (2006). A basic mathematical framework for conceptual graphs, *IEEE Transactions on Knowledge and Data Engineering* 18(2): 261–271.

Ooi, B. C., Tan, K.-L., Yu, C. & Bressan, S. (2000). Indexing the edges - a simple and yet efficient approach to high-dimensional indexing, *Proceedings of the Nineteenth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, May 15-17, 2000, Dallas, Texas, USA*, ACM, pp. 166–174.

Papadias, D., Tao, Y., Fu, G. & Seeger, B. (2005). Progressive skyline computation in database systems., *ACM Trans. Database Syst.* 30(1): 41–82.

Pei, J., Jin, W., Ester, M. & Tao, Y. (2005). Catching the best views of skyline: a semantic approach based on decisive subspaces, *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, VLDB Endowment, pp. 253–264.

Ponte, J. M. & Croft, W. B. (1998). A language modeling approach to information retrieval, *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, ACM, New York, NY, USA, pp. 275–281.

Rada, R., Mili, H., Bicknell, E. & Blettner, M. (1989). Development and application of a metric on semantic nets, *Systems, Man and Cybernetics, IEEE Transactions on* 19(1): 17–30.

Rasiwasia, N., Vasconcelos, N. & Moreno, P. J. (2006). Query by semantic example, *in* Sundaram et al. (2006), pp. 51–60.

Raubal, M. (2004). Formalizing conceptual spaces, *FOIS 2004: Proceedings of the Third International Conference*, IOS Press, A. Varzi and L. Vieu. Amsterdam, NL, pp. 153–164.

Rehatschek, H., Schallauer, P., Bailer, W., Haas, W. & Wertner, A. (2004). An innovative system for formulating complex, combined content-based and keyword based queries, *in* S. Santini & R. Schettini (eds), *Proceedings of the SPIE conference on Internet Imaging*, pp. 160 – 169.

Robertson, S. E. (1997). *The probability ranking principle in IR*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Rubner, Y., Tomasi, C. & Guibas, L. J. (2000). The earth mover's distance as a metric for image retrieval, *Int. J. Comput. Vision* 40(2): 99–121.

Salton, G. (1971). *The SMART Retrieval System—Experiments in Automatic Document Processing*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Santini, S. & Jain, R. (1997). Similarity is a Geometer, *Multimedia Tools Appl.* 5(3): 277–306.

Santini, S. & Jain, R. (1999). Similarity measures, *IEEE Trans. Pattern Anal. Mach. Intell.* 21(9): 871–883.

Schwering, A. (2005). Hybrid model for semantic similarity measurement, *in* R. Meersman, Z. Tari, M.-S. Hacid, J. Mylopoulos, B. Pernici, Ö. Babaoglu, H.-A. Jacobsen, J. P. Loyall, M. Kifer & S. Spaccapietra (eds), *OTM Conferences (2)*, Vol. 3761 of *Lecture Notes in Computer Science*, Springer, pp. 1449–1465.

Shen, H. T., Zhou, X. & Zhou, A. (2007). An adaptive and dynamic dimensionality reduction method for high-dimensional indexing, *The VLDB Journal* 16(2): 219–234.

Shneiderman, B., Bederson, B. B. & Drucker, S. M. (2006). Find that photo!: interface strategies to annotate, browse, and share, *Commun. ACM* 49(4): 69–71.

Smeulders, A. W. M., Worring, M., Santini, S., Gupta, A. & Jain, R. (2000). Content-Based Image Retrieval at the End of the Early Years, *IEEE Trans. Pattern Anal. Mach. Intell.* 22(12): 1349–1380.

Smith, J. (2007). The real problem of bridging the 'semantic gap', *MCAM07*, pp. 16–17.

Smith, J. R. & Chang, S.-F. (1996). VisualSEEk: A Fully Automated Content-Based Image Query System, *ACM Multimedia*, pp. 87–98.
URL: *citeseer.ist.psu.edu/smith96visualseek.html*

Snoek, C. G. M., Member, S. & Worring, M. (2003). Multimedia event based video indexing using time intervals, *IEEE Trans. on Multimedia* 7: 2004.

Snoek, C. G. M. & Smeulders, A. W. M. (2010). Visual-concept search solved?, *IEEE Computer* 43(6): 76–78.

Snoek, C. G. M. & Worring, M. (2005). Multimodal video indexing: A review of the state-of-the-art, *Multimedia Tools Appl.* 25(1): 5–35.

Snoek, C. G. M., Worring, M., Geusebroek, J., Koelma, D., Seinstra, F. & Smeulders, A. (2006). The Semantic Pathfinder: Using an Authoring Metaphor for Generic Multimedia Indexing, *IEEE Trans. Pattern Anal. Machine Intell.* 28(10): 1678–1689.

Snoek, C. G. M., Worring, M., van Gemert, J. C., Geusebroek, J.-M. & Smeulders, A. W. M. (2006). The challenge problem for automated detection of 101 semantic concepts in multimedia, *MULTIMEDIA '06: Proceedings of the 14th annual ACM international conference on Multimedia*, ACM, New York, NY, USA, pp. 421–430.

Sundaram, H., Naphade, M. R., Smith, J. R. & Rui, Y. (eds) (2006). *Image and Video Retrieval, 5th International Conference, CIVR 2006, Tempe, AZ, USA, July 13-15, 2006, Proceedings*, Vol. 4071 of *Lecture Notes in Computer Science*, Springer.

Tesic, J. (2004). *Managing Large-scale Multimedia Repositories*, PhD thesis, University of California, Santa Barbara.
URL: *http://vision.ece.ucsb.edu/publications/04ThesisTesic.pdf*

Tesic, J. & Manjunath, B. (2003). Nearest neighbor search for relevance feedback, *cvpr* 02: 643.

Tesic, J. & Smith, J. R. (2006). Semantic labeling of multimedia content clusters., *ICME*, IEEE, pp. 1493–1496.
URL: *http://dblp.uni-trier.de/db/conf/icmcs/icme2006.html#TesicS06*

Tversky, A. (1977). Features of similarity, *Psychological Review* 84: 327–352.

Van Rijsbergen, C. J. (1979). *Information Retrieval, 2nd edition*, Dept. of Computer Science, University of Glasgow.
URL: *citeseer.ist.psu.edu/vanrijsbergen79information.html*

Wactlar, H. D., Christel, M. G., Gong, Y. & Hauptmann, A. G. (1999). Lessons learned from building a terabyte digital video library, *Computer* 32(2): 66–73.

Wactlar, H. D., Kanade, T., Smith, M. A. & Stevens, S. M. (1996). Intelligent access to digital video: Informedia project, *Computer* 29(5): 46–52.

Wang, Q. & You, S. (2006). Fast similarity search for high-dimensional dataset, *ISM '06: Proceedings of the Eighth IEEE International Symposium on Multimedia*, IEEE Computer Society, Washington, DC, USA, pp. 799–804.

Weber, R., Schek, H.-J. & Blott, S. (1998). A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces, *Proc. 24th Int. Conf. Very Large Data Bases, VLDB*, pp. 194–205.
URL: *citeseer.ist.psu.edu/weber98quantitative.html*

Westerveld, T. (2004). *Using generative probabilistic models for multimedia retrieval*, Ph.d. thesis, University of Twente, Enschede, The Netherlands.

Westerveld, T. (2005). TRECVID as a Re-Usable Test-Collection for Video Retrieval, *Proceedings of the Multimedia Information Retrieval Workshop 2005*.

White, D. A. & Jain, R. (1996). Similarity indexing with the ss-tree, *ICDE '96: Proceedings of the Twelfth International Conference on Data Engineering*, IEEE Computer Society, Washington, DC, USA, pp. 516–523.

Wu, L. & Bretschneider, T. (2004). Vp-emd tree: An efficient indexing strategy for image retrieval, *in* H. R. Arabnia (ed.), *CISST*, CSREA Press, pp. 421–426.

Xiong, Z., Zhou, X. S., Tian, Q. & TS, Y. R. H. (2006). Semantic retrieval of video - review of research on video retrieval in meetings, movies and broadcast news, and sports, *Signal Processing Magazine, IEEE* 23(2): 18–27.

Yan, R. & Hauptmann, A. G. (2007). A review of text and image retrieval approaches for broadcast news video, *Inf. Retr.* 10(4-5): 445–484.

Yang, J. & Hauptmann, A. G. (2006). Annotating News Video with Locations., *CIVR*, pp. 153–162.

Yee, K.-P., Swearingen, K., Li, K. & Hearst, M. (2003). Faceted metadata for image search and browsing, *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM, New York, NY, USA, pp. 401–408.

Yu, C., Bressan, S., Ooi, B. C. & Tan, K.-L. (2004). Querying high-dimensional data in single-dimensional space, *The VLDB Journal* 13(2): 105–119.

Yu, J., Amores, J., Sebe, N. & Tian, Q. (2006). A new study on distance metrics as similarity measurement, *Multimedia and Expo, IEEE International Conference on* 0: 533–536.

Yuan, Y., Lin, X., Liu, Q., Wang, W., Yu, J. X. & Zhang, Q. (2005). Efficient computation of the skyline cube, *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, VLDB Endowment, pp. 241–252.

Yujian, L. & Bo, L. (2007). A normalized levenshtein distance metric, *IEEE Trans. Pattern Anal. Mach. Intell.* 29(6): 1091–1095.

Zhai, C. & Lafferty, J. (2001). A study of smoothing methods for language models applied to ad hoc information retrieval, *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM, New York, NY, USA, pp. 334–342.

Zhang, C. & Chen, T. (2003). *From Low Level Features to High Level Semantics*, CRC Press, chapter 27.
URL: *citeseer.ist.psu.edu/619954.html*