A Multistep Approach for Shape Similarity Search in Image Databases

Mihael Ankerst,
Hans-Peter Kriegel, *Member*, *IEEE Computer Society*,
and Thomas Seidl

Abstract—Shape similarity search is a crucial task in image databases, particularly in the presence of errors induced by segmentation or scanning images. The resulting slight displacements or rotations have not been considered so far in the literature. We present a new simlarity model that flexibly addresses this problem. By specifying neighborhood influence weights, the user may adapt the similarity distance functions to her or his requirements or preferences. Technically, the new similarity model is based on quadratic forms for which we present a multistep query processing architecture particularly for high dimensions as they occur in image databases. Our algorithm to reduce the dimensionality of quadratic form-based similarity queries results in a lower-bounding distance function that is proven to provide an optimal filter selectivity. Experiments on our test database of 10,000 images demonstrate the applicability and the performance of our approach even in high dimensions such as 1,024.

Index Terms—Content-based image retrieval, adaptable similarity search, multistep query processing, searching and browsing large image databases, managing high-dimensional image data.

1 Introduction

In the last decade, a wide range of image databases has established its relevance in areas like medicine, journalism, fashion, art, and industry. However, current practice in querying and retrieving images is often based on partial and exact match queries. Image databases typically support only queries by filename, captions, and keywords, for instance. Obviously, this does not satisfy the requirements of modern image retrieval since the user often wants to query the image database by characteristics such as color, texture, and shape. Therefore, the importance of similarity search in large image and multimedia databases has increased rapidly and also has attracted a lot of interests in the research area.

Image databases are playing an important role for marketing agencies, medical diagnostics, and patent agencies among others. Graphical archives may contain clip arts or pictograms that contain a single object or a small group of objects. In case of medical imaging, the objects in the images are typically extracted from the original images by segmentation. For segmentation, a variety of methods is already available from image processing research. To find similar objects within a large database of shapes is a more and more challenging task due to the increasing size of image archives. Since segmentation may induce slight errors like minor shifts, similarity search in these databases cope with a new kind of challenge which has not been considered yet. In order to provide an adequate similarity search method, its underlying similarity model has to take slight displacements of shapes or shape components into account.

Previous work does not seem to face this problem. In the field of shape similarity search in databases, Jagadish proposes a technique for the retrieval of similar shapes in two dimensions [15]. He derives an appropriate object description from a rectilinear cover of an object, i.e., a cover consisting of axis-parallel rectangles. The rectangles belonging to a single object are sorted by size, and the largest ones serve as retrieval key for the shape of the object. Korn et al. propose a method for searching similar tumor shapes in a medical image database [16]. The similarity measure that they investigate is based on an extended notion of the volume difference of two objects including mathematical morphology. Another approach using a structural feature index is suggested in [11].

A different problem is solved by techniques for object registration [5], [27]. They address the task to detect shapes anywhere in a given image that are similar to the objects in a database. For this purpose, the similarity model has to be invariant with respect to rotation, translation, and even scaling. For example, the Querying By Image Content (QBIC) system [8] contains a component for 2D shape retrieval which is based on algebraic moment invariants. In our applications such as graphical archives of clip arts and histograms or medical image databases. However, the shapes are stored in a normalized representation. Major rotations and translations of shapes are not to be taken into account for similarity search. The problem we are concerned with requires that the similarity model considers minor rotations and slight displacements of objects resulting from errors of segmentation and sampling. Our similarity models flexibly meets this requirement by the concept of neighborhood influence weights and may be adapted by the user to her or his specific preferences.

Due to the large and still increasing size of image databases, efficient query processing is a crucial task. For high-dimensional data such as digital images, several techniques for reducing the dimensionality have been proposed. Up to now, they have been applied to the Euclidean distance only. Our new adaptable similarity model, however, is based on quadratic forms. We propose a new method that extends the existing reduction techniques thus fitting to our more general and flexible similarity model. The resulting filter distance function in the reduced space has an optimal filter selectivity and fulfills the lower-bounding property as required in our multistep similarity query processing architecture.

The paper is organized as follows: In Section 2, we introduce a flexible model for pixel-based similarity search that may be adapted to user requirements. In Section 3, we present a general architecture and efficient algorithms for similarity query processing, and particular attention is given to the problem of high dimensions. Our algorithm guarantees no false drops as well as a minimum number of exact evaluations. The experiments in Section 4 demonstrate the applicability and good performance, and Section 5 concludes the paper.

2 Adaptable Similarity Model for Images

The key technique to solve the problem of slightly displaced shapes is to take the neighborhood of each pixel into account. By this approach, a similarity to the original image can be recognized if the overall image or even components of the image are displaced by some pixels within the considered neighborhood. By specifying the extension of the neighborhood that should be considered, the similarity distance function can be adapted to specific requirements of an application or the preferences of individual users.

2.1 Neighborhood Influence Weights

Technically, we take the neighborhood of each pixel into account by introducing the *neighborhood influence weights*. The images of interest are overlaid, and the similarity distance function is defined in terms of a pixel-by-pixel comparison of the images. Instead of just comparing each pixel to the pixel at exactly the same position in the partner image, the neighboring pixels become involved in the computation of the similarity distance. For this purpose, a

• *M. Ankerst, H.-P. Kriegel, and T. Seidl are with the Institute for Computer Science, University of Munich, Oettingenstr. 67, D-80538 München, Germany. E-mail: kriegel@dbs.informatik.uni-muenchen.de.*

Manuscript received 15 November 1997; revised 6 July 1998.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number 107201.

weighting factor is assigned by the user to each neighboring position around every pixel. The weighting factors determine the influence of the pixel difference at the corresponding position on the overall similarity distance.

In the following, we present a method to produce adequate neighborhood influence weights. Let us prepare our formula by the definition of a neighborhood distance function: Given two weighting factors, a_x and a_y, the neighborhood function $d_{a_x,a_y}(x, y)$ is defined by the following equation:

$$d_{a_x,a_y}(x, y) = \sqrt{\left(\frac{x}{a_x}\right)^2 + \left(\frac{y}{a_y}\right)^2}$$

A closer look to the formula, reveals that by d_{a_x,a_y}, represents an ellipse whose extensions along the x and the y axis correspond to the factors a_x and a_y. Thus, we call the weights a_x and a_y to be the neighborhood extension. Based on the neighborhood function d_{a_x,a_y}, we define our particular formula to produce neighborhood influence weights:

DEFINITION 1 (Neighborhood Influence Weights). *Given a neighborhood extension* a_x, a_y), *the* neighborhood influence weights *are determined by the following formula:*

$$w_{a_x,a_y}(x, y) = \begin{cases} (1 - d_{a_x,a_y}(x, y))^2 & \text{if } d_{a_x,a_y}(x, y) \le 1 \\ 0 & \text{if } d_{a_x,a_y}(x, y) > 1 \end{cases}$$

By this definition, we suggest a useful example for neighborhood influence weights. Beside this formula, numerous other methods to provide appropriate neighborhood influence weights may be used. An advantage of our approach is that due to the clearness of the concept the weighting factors can be facilely provided by the user. Since the neighborhood influence area represents an ellipse, a graphical ellipse may be used to edit the neighborhood extension (a_x, a_y). Thus, a graphical user interface for a shape-oriented image similarity search system may include such an ellipse editor in order to specify or modify the desired neighborhood influence area.

In Fig. 1, we illustrate two example neighborhood areas and, in particular, the corresponding neighborhood influence weights which we produced according to the above formula. The examples are the (3, 3) extension which represents a spherical neighborhood up to the second neighbor in each direction, and the (4, 2) extension which is a more general ellipse.

Note that our formula is designed in such a way that the neighborhood (1, 1) represents the degenerated neighborhood that contains only the pixel itself and actually no neighboring pixels from the surrounding area.

2.2 Adaptable Image Distance Function

Being provided with the concept of neighborhood influence weights, we are prepared to introduce our new adaptable similarity distance function for digital images. This novel distance function aims at the similarity of shapes that are contained in the images. By exchanging or modifying the neighborhood weights, the user may adapt the similarity distance function to the specific requirements of the application, or to his or her personal preferences. In the following, we assume appropriate neighborhood influence weights already to be given.

We start our definitions by introducing the local similarity distance of images which is a distance value for images that corresponds to a certain pixel location and its neighborhood within the images. The pixels of both images are overlaid, and for each location (x, y) from $\Im_c \times \Im_r$, the local similarity distance is determined by summing up the pixel differences around (x, y) while being weighted by the corresponding neighborhood influence weights.

DEFINITION 2 (Local Similarity Distance of Images). *Given two* $c \times r$ *images* $F, G, \Im_c \times \Im_r \to \Re$ *and neighborhood influence weights* w: $\mathrm{dom}(w) \to [0, 1]$, *the* local similarity distance *of* F *and* G *at the pixel location* $(x, y) \in \Im_c \times \Im_r$ *is given by a weighted sum over the pixel differences around* (x, y) *according to the following equation:*

$$d_w(f, G)\Big|_{x,y} = \sum_{(\hat{x},\hat{y})\in \mathrm{dom}(w)} w(\hat{x}, \hat{y}) \cdot (F_{x+\hat{x},y+\hat{y}} - G_{x+\hat{x},y+\hat{y}})$$

Note that for this definition, we have to extend our notion of images. In general, the domain of the weights w includes negative numbers, and the value $F_{-1,-1}$ is not defined yet, for instance. However, the above definition only requires the differences of pixel values from outside the image to be available but not the values itself. Thus, it suffices to provide the value of pixel differences which leads us to the following definition:

$$(F_{x,y} - G_{x,y}) = \begin{cases} F_{x,y} - G_{x,y} & \text{if } (x, y) \in \Im_c \times \Im_r \\ 0 & \text{if } (x, y) \notin \Im_c \times \Im_r \end{cases}$$

Now, the local similarity distance of images is well defined, and we are going to present our adaptable image similarity distance which is based on the concept of local similarity distance. In particular, the local similarity distances of all pixels in the images are summed up, being weighted by the corresponding pixel difference at the respective location.

DEFINITION 3 (Adaptable Image Similarity Distance). *Given two* $c \times r$ *images* F, G: $\Im_c \times \Im_r - \Re$ *and neighborhood influence weights* w: $\mathrm{dom}(w) - [0, 1]$, *the* adaptable image similarity distance *of* F *and* G *is given by a weighted sum over all local similarity distances of* F *and* G *as follows:*

$$d_w(F, G) = \sum_{(x,y)\in \Im_c \times \Im_r} (F_{x,y} - G_{x,y}) \cdot d_w(F, G)\Big|_{x,y}$$

Fig. 1. Adaptable similarity model for digital images. The contribution of each pixel to the overall similarity distance depends on the user-specified neighborhood of the pixel which may be graphically modified by an ellipse editor. (a) neighborhood extension (3, 3); (b) neighborhood extension (4, 2).

This novel image similarity distance function supports the adaptability to specific applications of shape-oriented similarity search. The parameters which may be tuned to fit the characteristics of particular applications or specific user preferences are the neighborhood influence weights w which have to be provided to the system.

As every reasonable similarity distance function, the adaptable image similarity distance function is desired to be positive definite, i.e., $d_w(F, G) > 0$ for all images $F \neq G$. However, it depends on the weights w whether $d_w(F, G)$ is positive, definite, or not. This property can efficiently be tested by the Cholesky decomposition.

3 Efficient Shape Similarity Query Processing

In the preceding section, we provide a highly flexible and adaptable similarity model for digital images. In order to meet the performance requirements of interactive similarity search, the important question emerges how to efficiently support similarity query processing. Several problems occur when applying the new model to practical applications: First, modern image databases contain tens and hundreds of thousands images and are even increasing. Second, major problems arise from the high dimensionality of the data spaces which are 256-D, 1,024-D, and 4,096-D in our sample databases. At last, the complexity of our similarity model effects that the evaluation of a single image comparison takes longer than a disk page access, and the CPU time dominates the I/O as we will see in the following.

3.1 Index-Based Ellipsoid Query Processing

By the definitions in Section 2, we provide an illustrative and intuitive introduction of our new and flexible image similarity distance function that may be adapted to application requirements and user preferences. For the purpose of efficient query processing on large image databases, we switch to a vector-based representation of the adaptable similarity model. We particularly demonstrate that $d_w(F, G)$ is equivalent to a quadratic form in the high-dimensional space of pixel vectors:

THEOREM. *For every two images F, G:* $\Im_c \times \Im_r \to \Re$ *and any neighborhood influence area w:* $\mathrm{dom}(w) \to [0, 1]$, *the shape-oriented image similarity distance* $d_w(F, G)$ *is a* $c \cdot r$*-dimensional quadratic form where components of the* $c \cdot r \times c \cdot r$ *matrix* A_w *are defined as* $A_w[p, p'] = w(p' - p) = w(p'_x - p_x, p'_y - p_y)$ *for* $p, p' \in \Im_c \times \Im_r$ *or* $p, p' \in \Re^{c \cdot r}$, *resp. In all, the equivalence is as follows:*

$$d_w(F, G) = (F - G) \cdot A_w \cdot (F - G)^T$$

PROOF. Let F and G be two $c \cdot r$ -dimensional vectors, i.e., $F, G \in \Re^{c \cdot r}$ that represent the $c \times r$ grayscale images $F, G: \Im_c \times \Im_r \to \Re$, and derive:

$$\begin{aligned} d_w(F, G) &= \sum_{(x,y) \in \Im_c \times \Im_r} (F_{x,y} - G_{x,y}) \cdot d_w(F, G)\Big|_{x,y} \\ &= \sum_{p \in \Re^{c \cdot r}} (F_p - G_p) \cdot d_w(F, G)\Big|_p \\ &= \sum_{p \in \Re^{c \cdot r}} (F_p - G_p) \sum_{\Delta p \in \mathrm{dom}(w)} w(\Delta p) \cdot (F_{p+\Delta p} - G_{p+\Delta p}) \\ &= \sum_{p \in \Re^{c \cdot r}} \sum_{p' \in \Re^{c \cdot r}} (F_p - G_p) \cdot w(p' - p) \cdot (F_{p'} - G_{p'}) \\ &= (F - G) \cdot A_w \cdot (F - G)^T \qquad \square \end{aligned}$$

Quadratic forms already appeared as appropriate distance functions for similarity search. Examples include the color-oriented similarity of images based on color histograms [21], [12], [25] and the approximation-based shape similarity of 3D surface segments [18], [17] where quadratic forms occur in the filter step. Due to its geometric shape, we call similarity queries based on quadratic forms *ellipsoid queries* [23]. For the purpose of similarity range query processing as well as k-nearest neighbor search and similarity ranking, an efficient algorithm for ellipsoid query processing has been developed that works on rectilinearly organized multidimensional index structures [18], [25]. The method is based on a fundamental distance algorithm that measures the distance of the query ellipsoid to a box as it occurs in the index directory. Recently, an improvement by using conservative approximations has been suggested [1]. Anyone of the existing frameworks for similarity query processing on multidimensional index structures may be employed [13], [14], [25].

The outstanding characteristic of the shape-oriented image similarity distance is the ultra-high dimensionality. For a image, the corresponding vector has a dimension of $c \cdot r$, and the similarity matrix A_w has the size $c \cdot r \times c \cdot r$. Due to memory restrictions, we decided to use the resolutions 16×16, 32×32, and 64×64, which lead to the dimensions 256, 1,024, and 4,096, respectively, for the image vectors.

3.2 Problems with High Dimensions

In principle, these methods also apply to high-dimensional feature spaces. In practice, however, the performance becomes significantly worse with increasing dimensions. A common approach to overcome these problems is to employ techniques for the reduction of dimensionality. The question of interest is whether and how the existing reduction techniques may be adapted to ellipsoid queries. The answer is yes, and we present an algorithm for the reduction of high-dimensional similarity matrices that respects any given linear reduction technique. In any case, the reduced query is again an ellipsoid query, and it optimally fits to our multistep similarity query processing architecture: First, the algorithm guarantees no false drops and, therefore, fulfills the essential correctness criterion that no actual result is missed in the answer set. Second, the reduced similarity matrix represents the greatest of all possible lower-bounding ellipsoid distance functions. Thus, our method ensures the optimal selectivity for range queries as well as for k-nearest neighbor queries in the reduced data space.

For several applications of similarity search in multimedia and spatial database systems, high- and ultrahigh-dimensional feature spaces have to be managed, i.e., 1,024-D, and 4,096-D pixel vectors in our case. As we will see, the performance of similarity query processing becomes significantly worse with increasing dimensions. Therefore, it is not recommended to employ purely index-based algorithms for high- and ultra-high-dimensional spaces. The severe efficiency problems in high dimensions occur due to the following obstacles: The quadratic nature of ellipsoid distance functions, and the curse of dimensionality [8].

3.2.1 *Quadratic Nature of Ellipsoid Distance Functions*

Whereas the evaluation of (weighted) Euclidean distance functions is linear in the dimension, general ellipsoid distance functions require an evaluation time that is quadratic in the number of dimensions. Fig. 2 demonstrates this effect for various dimensions that actually occur in our example databases.

3.2.2 *Curse of Dimensionality*

With an increasing number of dimensions, the efficiency of multidimensional access methods becomes worse and worse. The fraction of pages from the overall index that have to be accessed for range query and nearest-neighbor query processing significantly increases with the number of dimensions. This general phenomenon is known as the curse of dimensionality [8] and has been ana-

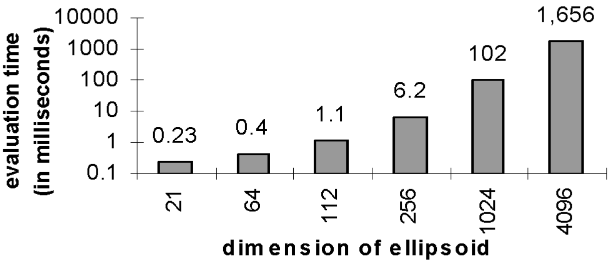

Fig. 2. Average evaluation time for a single ellipsoid distance.

lytically and experimentally investigated in [9], [20], [4], [2], for instance.

Recently, access methods have been developed that efficiently support query processing in medium dimensions such as the TV-tree [20] or the X-tree [4], [3]. These structures have been successfully applied to feature spaces of dimensions up to 20 and more. Nevertheless, the lower dimensionality effects the better performance.

3.3 Reduction of Dimensionality

A common approach to overcome the efficiency problems of query processing in high-dimensional spaces is to reduce the dimensionality. The basic idea is to project high-dimensional vectors down to a low-dimensional space which may be efficiently managed by multidimensional access methods. From the literature, several techniques to reduce the dimensionality of feature spaces are available such as the Karhunen-Loève Transform (KLT), the Discrete Fourier Transform (DFT), Discrete Cosine Transform (DCT), Wavelet Transforms, Multidimensional Scaling [19], and FastMap [9]. For the Euclidean distance function, the reduction techniques lead to filter distance functions which lower bound the exact distance function in order to prevent the query processor from producing false drops. For high-dimensional ellipsoid queries, the fundamental question is whether there exists a corresponding query in the reduced data space which fulfills the essential lower-bounding property.

In the QBIC project, quadratic form distance functions were used for a similarity model based on color histograms of dimensions 64 and 256 [8]. The proposed solution uses a reduction from N to k dimensions that is determined by a symmetric decomposition of the similarity matrix A [24], [12]. By design, the Euclidean distance of the reduced histogram vectors is a lower-bounding distance function of the original quadratic form as desired. Note that the resulting reduction directly depends on the matrix A. Therefore, A is required to be available already at index creation time which is the most important shortcoming of this method. Any modification or change of the similarity matrix effects that an existing index cannot be used any longer and has to be recomputed.

Other solutions deal with complex distance functions in general instead of reducing the dimensionality. Examples include the M-tree [7] and the SS-tree [28], the vp-tree or the mvp-tree [6]. They share the same problem to be committed to a distance function given in advance before index creation and do not support the flexibility that the similarity distance function may be modified and adapted to user requirements or preferences which may vary from query to query.

In our proposed shape similarity system, the neighborhood influence weights should be user parameters, and the similarity matrix which only depends on these user-specified weights is not known in advance, i.e., at index creation time. This means that the query processor should support a variety of similarity matrices for a single precomputed index. Again, the question arises whether for high-dimensional ellipsoid queries, a lower-bounding query exists in the reduced data space. Fortunately, the answer is yes: An optimal reduced filter distance function exists and, on top of it, it is an ellipsoid query. In the following, we present an algorithm to compute the reduced ellipsoid query by reducing the similarity matrix.

3.4 Karhunen-Loève Transform and Ellipsoid Queries

The Karhunen-Loève Transform (KLT) is a common technique for reducing the dimensionality of feature vectors [8], [20], [9]. It is known under several other names like Principal Components Analysis (PCA) or Singular Value Decomposition (SVD) that reflect the techniques to compute the KLT. From a linear algebra point of view, the KLT is a base transform and, therefore, can be represented by a orthogonal $N \times N$ matrix R, i.e., $RR^T = R^T R = Id$ the orthogonality of the KLT matrix R justifies the transformation of the similarity distance function $d_{A_N}(s, q)$ in the following way:

LEMMA 1 (Distance-Preserving Transformation). *Let A_N be an $N \times N$ similarity matrix and R an $N \times N$ orthogonal matrix (e.g., from KLT). Then the distance function $d_{A_N^R}$ for $A_N^R = R^T A_N R$ is equivalent to d_{A_N} when applied to transformed points pR and qR instead of the original points p and q, i.e., for all $p, q \in \Re^N$ holds: $d_{A_N}(p, q) = d_{R^T A_N R}(pR, qR)$.*

PROOF.

$$
\begin{aligned}
d_{A_N^R}(pR, qR) &= (pR - qR) \cdot R^T A_N R \cdot (pR - qR)^T \\
&= (p - q) \cdot RR^T A_N RR^T \cdot (p - q)^T \\
&= d_{A_N}(p, q). \qquad \square
\end{aligned}
$$

In the subsequent, we use the symbol Δ_k as an abbreviation of the difference vector $p - q$ in k dimensions. Note that the index only manages reduced entries which are obtained from pR and qR by truncating the last $N - r$ components. Therefore, we also reduce the matrix from $N \times N$ down to $r \times r$ by the algorithm described in the following.

3.5 Optimal Lower Bounding of Similarity Distances

The transformation of A_N to A_N^R is the first step on our way to a filter distance function $d_{A_r^R}(sR, qR)$ that lower-bounds the object distance function $d_{A_N}(s, q)$, as it is required to guarantee no false drops. In the following, we present the reduction of the query matrix from dimension $N \times N$ to $r \times r$ which is performed in a recursive way from A_k to A_{k-1}, $k = N, \ldots, r + 1$, such that the lower-bounding property holds in each step:

$$\forall \Delta_k \in \Re^k : \Delta_{k-1} \cdot A_{k-1} \cdot \Delta_{k-1}^T \le \Delta_k \cdot A_k \cdot \Delta_k^T$$

However, beyond this weak lower-bounding property, we use a matrix A_{k-1} that represents the greatest lower bound, i.e., the optimum of all reduced distance functions. For this purpose, we partition the matrix A_k by splitting off the last column and row, resulting in $A_k = \begin{bmatrix} \tilde{A}_{k-1} & \mathbf{col}_k^{\mathrm{T}} \\ \mathbf{row}_k & \tilde{a}_{kk} \end{bmatrix}$ where $\tilde{A}_{k-1}$ is a $(k - 1) \times (k - 1)$-matrix, $\tilde{a}_{kk} \in \Re$ is a scalar, and $\mathbf{row}_k, \mathbf{col}_k^{\mathrm{T}} \in \Re^{k-1}$ are two row vectors which we combine to the row vector $last_k = \frac{1}{2}(\mathbf{row}_k + \mathbf{col}_k)$.

Remember that reducing the dimensionality of a vector, $(\Delta_{k-1}, \delta_k) \in \Re^k$, $\delta_k \in \Re$, includes truncation of the kth component δ_k. Therefore, we assume that δ_k is not available.

LEMMA 2 (Greatest Lower Bound). *For any positive definite $k \times k$ similarity matrix, A_k, the $(k-1) \times (k-1)$ matrix $A_{k-1} = \tilde{A}_{k-1} - \frac{1}{\tilde{a}_{kk}}(\boldsymbol{last}_k^{\mathrm{T}} \cdot \boldsymbol{last}_k)$, which consists of $a_{ij} := a_{ij} - \frac{(a_{ik}+a_{ki})(a_{kj}+a_{jk})}{4\tilde{a}_{kk}}$ for $1 \le i, j \le k-1$, defines an distance function $d_{A_{k-1}}$ which is the minimum and, therefore, the greatest lower bound of d_{A_k} for the case that the value of $\delta_k \in \Re$ is unknown*:

$$\forall \Delta_k \in \Re^k: \Delta_{k-1} \cdot A_{k-1} \cdot \Delta_{k-1}^{\mathrm{T}} = \min\left\{\Delta_k \cdot A_k \cdot \Delta_k^{\mathrm{T}} \,\middle|\, \delta_k \in \Re\right\}$$

PROOF. Note that the matrix A_{k-1} is well defined since for every positive definite matrix A_k, the component $\tilde{a}_{kk}$ is positive: $\tilde{a}_{kk} = (0, \ldots, 0, 1) \cdot A_k \cdot (0, \ldots, 0, 1)^{\mathrm{T}} > 0$. By expanding A_k, followed by a quadratic complementation, we obtain:

$$\begin{aligned}\Delta_k \cdot A_k \cdot \Delta_k^{\mathrm{T}} &= \Delta_{k-1} \cdot \tilde{A}_{k-1} \cdot \Delta_{k-1}^{\mathrm{T}} + 2\delta_k \boldsymbol{last}_k \Delta_{k-1}^{\mathrm{T}} + \tilde{a}_{kk}\delta_k^2 \\ &= \Delta_{k-1} \cdot \tilde{A}_{k-1} \cdot \Delta_{k-1}^{\mathrm{T}} - \frac{1}{\tilde{a}_{kk}}(\boldsymbol{last}_k \cdot \Delta_{k-1}^{\mathrm{T}})^2 \\ &\quad + \frac{1}{\tilde{a}_{kk}}(\boldsymbol{last}_k \cdot \Delta_{k-1}^{\mathrm{T}} + \tilde{a}_{kk}\delta_k)^2 \\ &= \Delta_{k-1} \cdot \left(\tilde{A}_{k-1} - \frac{1}{\tilde{a}_{kk}}(\boldsymbol{last}_k^{\mathrm{T}} \cdot \boldsymbol{last}_k)\right) \cdot \Delta_{k-1}^{\mathrm{T}} \\ &\quad + \frac{1}{\tilde{a}_{kk}}(\boldsymbol{last}_k \cdot \Delta_{k-1}^{\mathrm{T}} + \tilde{a}_{kk}\delta_k)^2 \\ &= \Delta_{k-1} \cdot A_{k-1} \cdot \Delta_{k-1}^{\mathrm{T}} + \frac{1}{\tilde{a}_{kk}}(\boldsymbol{last}_k \cdot \Delta_{k-1}^{\mathrm{T}} + \tilde{a}_{kk}\delta_k)^2;\end{aligned}$$

The second term of the sum is a square, and therefore, its minimum is zero which will be reached for a certain $\delta_k \in \Re$. However, since δ_k is assumed to be not available, we may only rely on the minimum value, i.e., zero. Therefore, the first term of the sum, $\Delta_{k-1} \cdot A_{k-1} \cdot \Delta_{k-1}^{\mathrm{T}}$, represents the minimum of the left-hand side, $\Delta_k \cdot A_k \cdot \Delta_k^{\mathrm{T}}$, for all $\Delta_k \in \Re^k$. □

3.6 Overall Similarity Matrix Reduction

Provided with the preceding lemmata, we present the fundamental proposition of this section which is the formal background of our matrix reduction algorithm. By R_r, let us denote the $N \times r$ reduction matrix that is obtained from the $N \times N$ reduction matrix R by truncating the last $N - r$ columns:

THEOREM. *For any arbitrary $N \times r$ reduction matrix R_r and every $N \times N$ similarity matrix A_N, the reduced $r \times r$ matrix A_r^R represents the greatest lower bounding filter distance function $d_{A_r^R}$ with respect to the original distance function d_{A_N}, i.e., for all objects $s, q \in \Re^N$:*

$$d_{A_r^R}(sR_r, qR_r) \le d_{A_N}(s, q)$$

PROOF. According to Lemma 1, for any R and A_N, a matrix A_N^R exists which is computed by a distance preserving transformation. Lemma 2 allows a step by step reduction from the dimension N down to the reduced dimension r, producing the greatest lower-bounding distance function in each respective dimension. □

Our algorithm to reduce the matrix assumes the full orthogonal $N \times N$ reduction matrix R to be given. The lower-bounding property which we have shown in the theorem ensures the completeness of the method, i.e., guarantees no false drops. From a geometric point of view, step 1 performs a rotation, and step 2 a projection (not an intersection) of the N-dimensional query ellipsoid to r dimensions with respect to the KLT reduction matrix R. The overall runtime complexity is $O(N^3)$:

Step 1: Transform A_N to $A_N^R = R^T A_N R$ (cf. Lemma 1)
Step 2: For k from N down to $r + 1$, reduce A_k^R to A_{k-1}^R (cf. Lemma 2)

3.7 Multistep Query Processing Architecture

For the overall architecture, we follow the paradigm of multistep query processing where an index-based filter step produces a set of candidates and a subsequent refinement step performs the expensive exact evaluation of the candidates. Whereas the filter step has to guarantee completeness, i.e., no actual result is missing from the set of candidates, the refinement step ensures the correctness, i.e., no false hit belongs to the answer. For similarity range queries ("retrieve all images with a similarity distance of at most ε to the query image"), we employ the algorithm of [10]. For k-nearest neighbor queries ("retrieve the k most similar images from the database"), we may use the algorithm from [16] or, better, the version of [26] which is proven to be optimal, i.e., to produce the minimum number of candidates. Fig. 3 illustrates the architecture for similarity range queries where the filter step is based on a multidimensional index structure, and only in the refinement step, the exact object representation is retrieved from the object server.

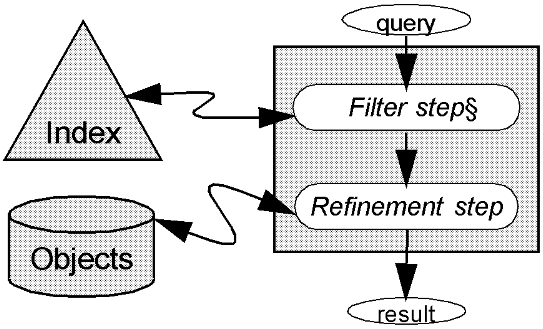

Fig. 3. Multistep similarity query processing.

The important issue to guarantee the completeness, i.e., to prevent the algorithm from producing false drops, is to provide the filter step with a distance function that lower bounds the exact object similarity distance function $d_{A_N}(s, q)$, in our case. Clearly, the reduced ellipsoid distance function $d_{A_r^R}(sR_r, qR_r)$ that applies to reduced vectors sR_r as they are managed by the underlying r-dimensional index meets this requirement. Beside the completeness, the efficient evaluation of this filter distance function on large databases is supported by available index structures. Moreover, since $d_{A_r^R}(sR_r, qR_r)$ is the greatest of all lower bounding distance functions in the reduced data space, it ensures the optimal selectivity in the filter step.

4 Experimental Evaluation

We implemented our algorithms in C++ and evaluated them on our test database of 10,000 grayscale images from a commercially available CD-ROM. We normalized the images in order to obtain representations by 256, 1,024, and 4,096 dimensions, respectively. We present results that illustrate the desired behaviour of our similarity model with respect to the returned answers, and that demonstrate the good filter selectivity and high performance in the high-dimensional image space.

4.1 Similarity Ranking

As a sample query object, we have chosen the image *violine1*. Fig. 4 presents the most similar images from the database for the neighborhood area (1, 1) which corresponds to the Euclidean distance. As expected, the reference image *violine1* is ranked to the top position among the 10,000 images in the database since the similarity distance exactly is zero.

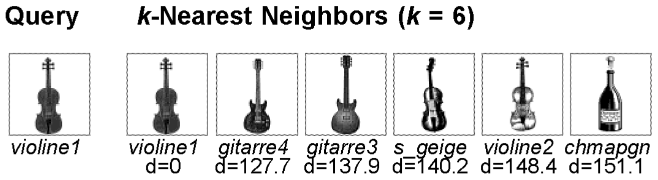

Fig. 4. *k*-nearest neighbors of *violine1* among 10,000 images.

Let us consider a slight displacement of *violine1* by 3 pixels to the right. Now, for the neighborhood area of (1, 1), the original (unshifted) *violine1* is ranked to position 328 which is far away from the top. By adapting the neighborhood area, better ranks are achieved. Fig. 5 demonstrates the improvement that is obtained in case of the neighborhood area (12, 1). The diagram at the bottom depicts the achieved ranks of *violine1* when varying the neighborhood influence area from (1, 1) to (12, 1). Whereas the reference image *violine1* ranks at position 328 for the neighborhood influence area (1, 1), the rank improves with the increasing a_x extension. Already for (5, 1), *violine1* ranks among the top hundred of all 10,000 images from the test database, and for (12, 1), a rank of 10 is achieved for the reference image *violine1*. Thus, the adaptable shape-oriented similarity model actually provides support for a flexible and adaptable similarity search in image databases.

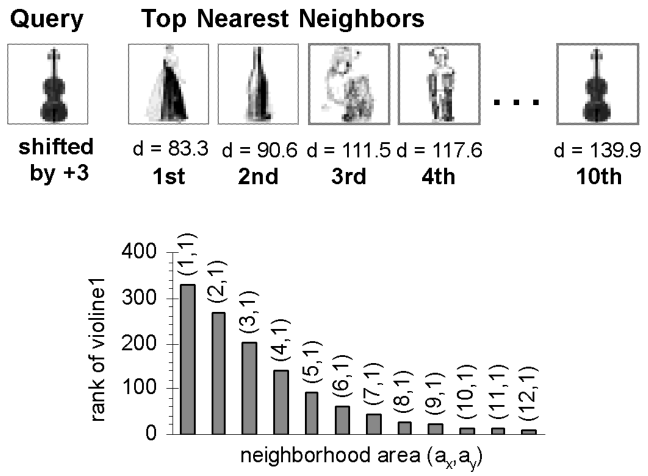

Fig. 5. Ranking of shifted query shapes for various neighborhood influence areas. The reference image *violine1* is used as a query image after being shifted to the right by +3 pixels. The diagram depicts the rank of the original image *violine1* within the overall database depending on the neighborhood influence area.

4.2 Performance Evaluation

For the performance experiments, we ran a sample of *k*-nearest neighbor queries and used different similarity matrices *sim-1-1*, *sim-3-1*, *sim-6-1*, and *sim-9-1*. We set *k* to values from 2 to 12 which are typical retrieval parameters in interactive environments. The matrices *sim-x-y* correspond to the neighborhood areas (*x*, *y*). The high-dimensional spaces are reduced down to *r* dimensions,, and the reduced vectors are managed by X-trees [4]. The experiments ran on HP 9000/780 machines under HP-UX 10.20.

In the first experiment, we set $k = 10$ thus retrieving 0.1 percent of the objects from the database. The top diagram in Fig. 6 depicts the number of candidates which corresponds to the filter selectivity, and the number of index pages which were accessed in the filter step. The bottom diagram demonstrates the overall runtime of query processing in the average. The results substantiate that the foregoing parameters represent the main cost factors in high-dimensional similarity query processing for complex similarity distance functions. From the diagrams, we observe that with increasing dimension of the index, the number of candidates significantly decreases since the index provides an increasing amount of information to the filter step. On the other hand, the number of accessed index pages increases due to the higher space requirement and the curse of dimensionality. In presence of this trade-off, a minimum for the overall runtime may be determined, and in our experiments, we found an index dimension of approximately 48 to be optimal.

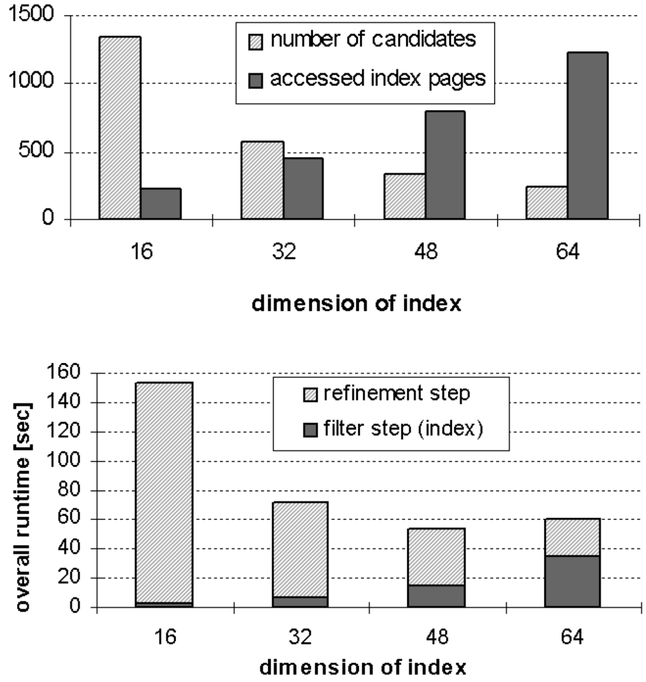

Fig. 6. Performance depending on the index dimension. For *k*-nn queries ($k = 10$), the top diagram depicts the average number of candidates and the average number of accessed index pages. Bottom diagram: The average overall runtime is composed from the runtimes of the filter and of the refinement step.

At next, we show the performance of the multistep query processing technique depending on the neighborhood influence area. The top diagram in Fig. 6 demonstrate that the number of accessed index pages does not significantly vary when modifying the similarity measure whereas the number of candidates highly depends on the neighborhood area. The bottom diagram depicts the overall runtime and its components. The diagram in Fig. 8 shows that the optimum index dimension highly depends on the neighborhood influence area, too. In particular, for (9, 1) and (6, 1), the optimum is approximately 48 whereas for (3, 1) and (1, 1) = id, the optimum is greater than 64. The diagrams in Fig. 9 depict the average values of filter selectivity and overall runtime in seconds depending on the query parameter *k*. In this case, the index has a dimensionality of 64. Obviously, the overall runtime highly depends on the filter selectivity, i.e., the number of candidates.

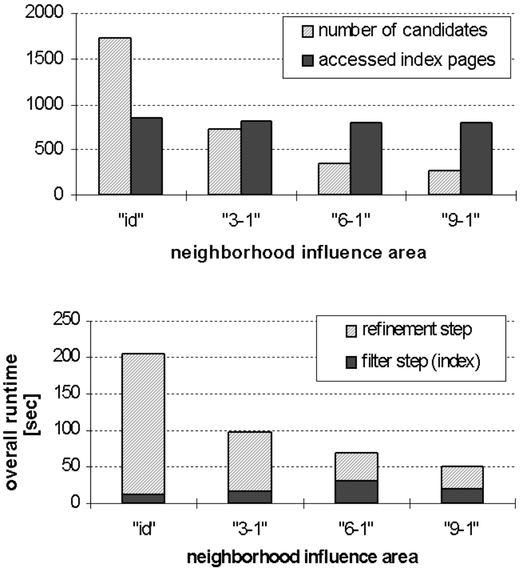

Fig. 7**.** Performance depending on neighborhood areas. For *k*-nn queries (*k* = 10), the average number of candidates and the number of accessed index pages are depicted (top diagram). The average overall runtime is composed from the runtimes of the filter and the refinement step (bottom diagram).

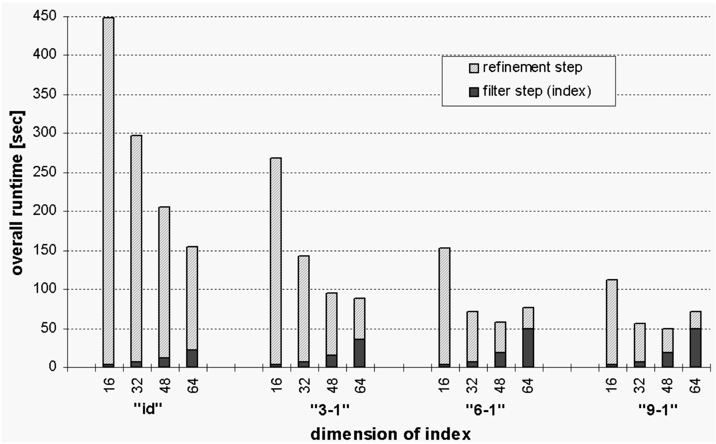

Fig. 8**.** Overall runtime depending on the index dimension (16, 32, 48, 64) and on the neighborhood influence area for a sample of *k*-nn queries (*k* = 10).

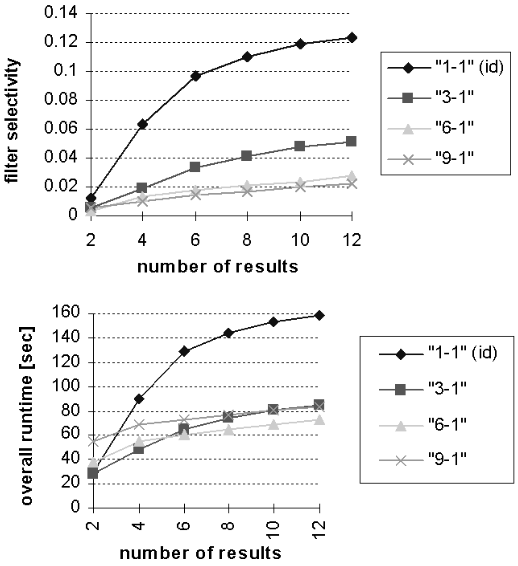

Fig. 9**.** Performance depending on the query parameter *k*. For various similarity matrices *sim*-*x*-*y*, the diagrams depict the filter selectivity (top) and the overall runtime (bottom).

In our experiments, we investigated images of resolutions 16 × 16, 32 × 32, and 64 × 64 which lead to the dimensions 256, 1,024, and 4,096, respectively, for the image vectors. The corresponding similarity matrices have 65,536, 1,048,576 and 16,777,216 components. If applying the method to images with a higher resolution such as 1K × 1K, several options seem to be reasonable. Medical images, for example, often contain the shapes of several objects. In this case, a segmentation of the images that extracts the shapes of interest helps and yields a set of smaller images. Another approach is to coarse the resolution for the purpose of similarity search which reduces the dimensionality of the corresponding pixel vectors. By following this approach, similarity queries on these images may be processed by the presented algorithms. Finally, extended reduction techniques have to be developed by which higher resolutions may be managed even under presence of current main memory restrictions.

5 CONCLUSIONS

In this paper, we introduce a novel similarity distance function for digital images that represent a pixel-based shape similarity model. This similarity model is particularly flexible and provides adaptability to specific application characteristics and individual user preferences. The adaptability is based on the concept of neighborhood influence weights which represent intuitive parameters that may be specified by a graphical editor. The models are evaluated on a database of 10,000 grayscale images.

We present a multistep architecture for similarity query processing according to our adaptable model. In order to meet the efficiency requirements of high-dimensional spaces, we employ a common technique to reduce the dimensionality of the vectors. We particularly extend the technique to the quite more general case of adaptable similarity distance functions given as quadratic forms. From our algorithm to reduce high-dimensional similarity matrices, we obtain a lower-bounding filter distance function as it is required for multistep query processing in order to guarantee no false drops. Moreover, the greatest of all lower-bounding filter distance functions is obtained which guarantees the optimal selectivity in the reduced data space. Extensive experimental evaluations of the method demonstrate its applicability and efficiency for dimensions up to 1,024.

In our future work, we plan to extend our new methods to even higher dimensions beyond 1,024-D or 4,096-D. Examples occur in applications such as similarity search in 3D databases where spatial objects are represented by fine-grain voxels as well as similarity query processing in 2D image databases that support high-resolution representations of images.

REFERENCES

[1] M. Ankerst, B. Braunmüller, H.-P. Kriegel, and T. Seidl: “Improving Adaptable Similarity Query Processing by Using Approximations,” *Proc. 24th Int'l Conf. Very Large Databases (VLDB'98)*, New York, 1998.

[2] S. Berchtold, C. Böhm, D. Keim, and H.-P. Kriegel, “A Cost Model for Nearest Neighbor Search in High-Dimensional Data Spaces,” *Proc. 16th ACM SIGACT-SIGMOD-SIGART Symp. Principles of Database Systems (PODS)*, pp. 78–86, Tucson, Ariz., 1997.

[3] S. Berchtold, C. Böhm, B. Braunmüller, D. Keim, and H.-P. Kriegel, “ Fast Parallel Similarity Search in Multimedia Databases,” *Proc. ACM SIGMOD Int'l Conf. Management of Dat*a, pp. 1–12, Tucson, Az., best paper award, 1997.

[4] S. Berchtold, D. Keim, and H.-P. Kriegel, “The X-tree: An Index Structure for High-Dimensional Data,” *Proc. 22nd Conf. Very Large Data Bases*, pp. 28–39, Mumbai, India, 1996.

[5] P.J. Besl and N.D. McKay, “A Method for Registration of 3-D Shapes,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[6] T. Bozkaya and M. Ozsoyoglu, “Distance-Based Indexing for High-Dimensional Metric Spaces,” *Proc. ACM SIGMOD Int'l Conf. Management of Data*, pp. 357–368, Tucson, Ariz., 1997.

[7] P. Ciacca, M. Patella, and P. Zezula, “M-Tree: An Efficient Access Method for Similarity Search in Metric Spaces,” *Proc. 23rd Int'l Conf. Very Large Databases (VLDB'97)*, pp. 426–435, Athens, 1997.

[8] C. Faloutsos, R. Barber, M. Flickner, J. Hafner et al., “Efficient and Effective Querying by Image Content,” *J. Intelligent Information Systems*, vol. 3, pp. 231–262, 1994.

[9] C. Faloutsos and K.I. Lin, “FastMap: A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Data,” *Proc. ACM SIGMOD Int'l Conf. Management of Data*, San Jose, Calif., pp. 163–174, 1995.

[10] C. Faloutsos M. Ranganathan, and Y. Manolopoulos, “Fast Subsequence Matching in Time-Series Database,” *Proc. ACM SIGMOD Int'l Conf. Management of Data, Minneapolis*, Minn., pp. 419–429, 1994.

[11] J.E. Gary and R. Mehrotra, “Similar Shape Retrieval Using a Structural Feature Index,” *Information Systems*, vol. 18, no. 7, pp. 525-537, 1993.

[12] J. Hafner, H.S. Sawhney, W. Equitz, M. Flickner, and W. Niblack, “Efficient Color Histogram Indexing for Quadratic Form Distance Functions,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 17, no. 7, pp. 729–736, 1995.

[13] A. Henrich, “A Distance-Scan Algorithm for Spatial Access Structures,” *Proc. Second ACM Workshop Advances in Geographic Information Systems*, Gaithersburg, Md., pp. 136–143, 1994.

[14] G.R. Hjaltason and H. Samet, “Ranking in Spatial Databases,” *Proc. Fourth Int'l Symp. Large Spatial Databases*, Portland, Maine, pp. 83–95, 1995.

[15] H.V. Jagadish, “A Retrieval Technique for Similar Shapes,” *Proc. ACM SIGMOD Int'l Conf. Management of Data*, pp. 208-217, 1991.

[16] F. Korn, N. Sidiropoulos, C. Faloutsos, E. Siegel, and Z. Protopapas, “Fast Nearest Neighbor Search in Medical Image Databases,” *Proc. 22nd VLDB Conf.*, Mumbai (Bombay), India, pp. 215–226, 1996.

[17] H.-P. Kriegel and T. Seidl, “Approximation-Based Similarity Search for 3-D Surface Segments,” *GeoInformatica J.*, Kluwer Academic, 1998. to appear

[18] H.P. Kriegel, T. Schmidt, and T. Seidl, “3D Similarity Search by Shape Approximation,” *Proc. Fifth Int'l Symp. Large Spatial Databases (SSD'97)*, Berlin, Germany, Lecture Notes in Computer Science, vol. 1,262, pp.11–28, 1997.

[19] J.B. Kruskal and M. Wish, “Multidimensional Scaling,” SAGE, Beverly Hills, Calif., 1978.

[20] K. Lin, H.V. Jagadish, and C. Faloutsos, “The TV-Tree: An Index Structure for High-Dimensional Data,” *VLDB J.*, vol. 3, pp. 517-542, 1995.

[21] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasmann, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin, “The QBIC Project: Querying Images by Content Using Color, Texture, and Shape,” *PIE 1993 Int'l Symp. Electronic Imaging: Science and Technology Conf. 1908, Storage and Retrieval for Image and Video Databases*, San Jose, Calif., 1993.

[22] N. Roussopoulos, S. Kelley, and F. Vincent, “Nearest Neighbor Queries,” *Proc. ACM SIGMOD Int'l Conf. Management of Data*, pp. 71-79, 1995.

[23] T. Seidl, “Adaptable Similarity Search in 3-D Spatial Database Systems,” PhD thesis, Inst. for Computer Science, Univ. of Munich, 1997; Herbert Utz Publishers, Munich, Germany, http://utzverlag.com

[24] H. Sawhney and J. Hafner, “Efficient Color Histogram Indexing,” *Proc. Int'l Conf. Image Processing*, pp. 66–70, 1994.

[25] T. Seidl and H.-P. Kriegel, “Efficient User-Adaptable Similarity Search in Large Multimedia Databases,” *Proc. 23rd Int'l Conf. Very Large Databases (VLDB'97)*, pp. 506–515, Athens, 1997.

[26] T. Seidl and H.-P. Kriegel, “Optimal Multistep k-Nearest Neighbor Search, *Proc. ACM SIGMOD Int'l Conf. Management of Data*, Seattle, 1998.

[27] G. Taubin and D.B. Cooper, “Recognition and Positioning of Rigid Objects Using Algebraic Moment Invariants,” *Geometric Methods in Computer Vision*, vol. 1,570, pp. 175–186, SPIE, 1991.

[28] D.A. White and R. Jain, “Similarity Indexing with the SS-Tree,” *Proc. 12th Int'l Conf. Data Eng. (ICDE)*, pp. 516–523, 1996.