Signal Processing: *Image Communication* 24 (2009) 775–790

Contents lists available at ScienceDirect

Signal Processing: *Image Communication*

journal homepage: www.elsevier.com/locate/image

An efficient high-dimensional indexing method for content-based retrieval in large image databases

I. Daoudi [a,c,*], K. Idrissi [a], S.E. Ouatik [b], A. Baskurt [a], D. Aboutajdine [c]

[a] *INSA de Lyon, Laboratoire d'Informatique en Images et Systèmes d'information, LIRIS, UMR 5205 CNRS, France*
[b] *Laboratoire d'Informatique, Statistiques et Qualité, LISQ. Faculté Des Sciences Dhar Mahraz, Fès, Maroc*
[c] *Laboratoire de Recherche en Informatique et Télécommunication, Faculté Des Sciences, Rabat, Maroc*

ARTICLE INFO

Article history:
Received 29 September 2008
Received in revised form
28 August 2009
Accepted 4 September 2009

Keywords:
CBIR
High-dimensional vector space
Region approximation approach
Kernel
Image databases
Relevance feedback

ABSTRACT

High-dimensional indexing methods have been proved quite useful for response time improvement. Based on Euclidian distance, many of them have been proposed for applications where data vectors are high-dimensional. However, these methods do not generally support efficiently similarity search when dealing with heterogeneous data vectors. In this paper, we propose a high-dimensional indexing method (KRA$^+$-Blocks) as an extension of the region approximation approach to the kernel space. KRA$^+$-Blocks combines nonlinear dimensionality reduction technique (KPCA) with region approximation approach to map data vectors into a reduced feature space. The created feature space is then used, on one hand to approximate regions, and on the other hand to provide an effective kernel distances for both filtering process and similarity measurement. In this way, the proposed approach achieves high performances in response time and in precision when dealing with high-dimensional and heterogeneous vectors.

1. Introduction

The amount of multimedia data has strongly increased in recent years. Therefore, new efficient and powerful applications that handle this data are needed, such as image search engines, bio-medical imaging, education, commerce, etc.

In CBIR applications, the idea of image indexing consists in extracting descriptors from images, then mapping them into a high-dimensional space. The distance between two vectors is frequently used to estimate the similarity between the related images. Therefore, the problem of finding the most similar images to a given query can be seen as a problem of *k–NN* search in high-dimensional vector space. Many methods have been proposed and they are known as high-dimensional indexing methods. However, they encounter two kinds of problems. Firstly, they suffer from the notorious dimensionality curse because they deal with high-dimensional vectors. Secondly, when the considered data is heterogeneous and not distributed uniformly, these methods are confronted with the choice of the distances that have to be used both for data space partitioning and similarity estimation. In this work, we propose a novel high-dimensional indexing method based on vector approximation approach so as to reduce the distance computations (CPU time) and to improve the similarity search quality.

1.1. Related work

In the past, many high=dimensional indexing methods were proposed to solve the *k–NN* search problem efficiently. Most of them, however, are restricted to a

[*] Corresponding author at: INSA de Lyon, Laboratoire d'Informatique en Images et Systèmes d'information, LIRIS, UMR 5205 CNRS, France.
E-mail addresses: dimane@liris.cnrs.fr (I. Daoudi), kidrissi@liris.cnrs.fr (K. Idrissi), s_ouatik@yahoo.com (S.E. Ouatik), abaskurt@liris.cnrs.fr (A. Baskurt), Aboutaj@ieee.org (D. Aboutajdine).

0923-5965/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.image.2009.09.001

small number of dimensions or very specific data distributions. They typically degenerate to perform worse than a sequential scan if the number of dimensions is high. This phenomenon is often referred to as a curse of dimensionality problem.

Many approaches have been proposed to overcome the curse of dimensionality. They can be classified into three major categories: spatial access methods, dimensionality reduction approaches and filtering (approximation) approaches.

In spatial access methods, an image is represented by a descriptor and the Euclidean distance between the descriptors of two images is used both to compute the similarity between them and to divide the space/data vectors into spatial clusters.

KD-Tree [1] and R-Tree [2] are the first examples of spatial access methods, then, several enhanced methods have been proposed. R*-Tree[3] provides a consistently better performance than the R-Tree and R$^+$-Tree[4] by introducing a policy called ‘forced reinsert’. Lin et al. [5] proposed TV-Tree, which uses so-called telescope vectors. Berchtold et al. [6] introduced X-Tree, which is particularly designed for indexing higher-dimensional data. X-Tree avoids an overlapping of region bounding boxes in the directory structure using a new organization of the directory, and it outperforms both TV-Tree and R*-Tree significantly. However, bounding rectangles can still overlap in higher dimensions. White and Jain [7] proposed the SS-Tree as an alternative to R-Tree structure, which uses minimum bounding spheres instead of rectangles. Even though SS-Tree outperforms R*-Tree, overlapping in high dimensions still occurs. Thereafter, several other spatial access methods are proposed such as SR-Tree [8], S^2-Tree [9], Hybrid-Tree [10], etc.

All these conventional methods work efficiently if the number of dimensions is low; however, they still suffer from the curse of dimensionality (up to d=16) and their performance degenerates to being worse than brut-force sequential scan.

Dimensionality reduction approach is well known as an effective process for mapping the original high-dimensional features space into low-dimensional one by eliminating the redundant information from the original data. The most well-known method is the principal component analysis (PCA) [11]. The idea is to remove linear correlation between dimensions by rotating the data space. This can be done by solving the eigenvalue problem on a covariance matrix, and the resulting eigenvalues reflect the variance of the resulting dimensions. Unfortunately, only linear correlation can be removed.

Multidimensional scaling (MDS) is another technique for discovering the underlying spatial structure of a set of data items from the similarity information among them [12,13]. It assumes the existence of a monotonic relationship between the original and the projected pairwise distances.

LLE [14] and Isomap [15], are two representative methods of the nonlinear dimensionality reduction approach. The idea of LLE is to map high-dimensional vectors into a lower-dimensional space, preserving as well as possible the neighborhood of each vector and the global distance between rest of the vectors. The intrinsic structure is thus preserved. Isomap constructs neighborhood graph and computes shortest path distances (geodesic distances) for each pair of points in the graph. The classical MDS method is then used with geodesic distances.

Although many reduction techniques exist, they are restricted in different ways. The performances of linear methods decrease dramatically when the manifold is nonlinear, whilst the computational complexity of nonlinear approaches is generally higher than the linear methods.

The filtering approach has been proposed as a palliative solution for the dimensionality curse. It includes VA-File [16], RA-Blocks [17], LPC-File [18], GC-Tree [19], etc. The idea is to divide the data space into cells or regions, and to use bit strings to approximate the data vectors that fall into those cells or regions. During the *k–NN* search step, for a given query vector, the relatively small approximation file is sequentially scanned instead of reading the complete data file. As the original vectors are filtered, only a small fraction of them will be accessed, and thus, limited access to the complete database is performed.

In VA-File [16], the space is partitioned into a number of hyper-rectangular cells. Each non-empty cell location is encoded with a unique bit strings and stored in the approximation file. In nearest neighbors search step, the file is sequentially read, and then, upper and lower distance bounds between the query vector and each cell are estimated. The bounds are used to prune the irrelevant vectors' cells. Finally, the set of candidate vectors are read from the hard disk and the *k–NN* are determined.

Chen et al. [17] noticed that for very large databases, the approximation file cannot be completely stored in memory. Hence, they proposed the region approximation method, called RA-Blocks. The data space is partitioned into regions that contain cells as described in Section 2.1. Each region is approximated by two bit strings, corresponding to the bottom-left and top-right cells of this region. Hence, the computation time and the number of I/O access are reduced, as the lower and upper bounds are computed for regions and not for cells. However, the partitioning strategy of the RA-Blocks usually generates both empty regions and regions that contain only few vectors compared to their capacity. To overcome these limitations and to enhance the subdivision strategy used by RA-Blocks, RA$^+$-Blocks was developed in our previous paper [20].

As the filtering approach outperforms a sequential scan and spatial access methods for high dimensionalities, we focus on performing a *k–NN* search using this approach.

1.2. *Our contributions*

The objectives of this paper are to significantly reduce the *k–NN* search response time and to improve the similarity search quality. We propose a novel approximation method: KRA$^+$-Blocks, and a nonlinear similarity measurement including relevance feedback mechanism.

To tackle the dimensionality curse problem when using *k–NN* search, we combine high-dimensional indexing method based on the approximation approach with nonlinear dimensionality reduction technique. The proposed kernel region approximation blocks (KRA$^+$-Blocks) method reduces the vectors' dimension using kernel principal component analysis (KPCA) and then builds an efficient index structure in the kernel feature space. In contrast to the VA-file, region approximations are arranged in an index structure and high-dimensional vectors are assigned to the corresponding region approximations in the index. The approximation related to each region is then given by the corresponding beginning and ending vertices.

To improve the search quality, our idea is to provide a general similarity framework, both for indexing and retrieval. The proposed approach learns a good kernel function from data automatically, and then uses the kernel to map vectors into a new feature space allowing a more effective data representation. Relevance feedback mechanism is used in the induced feature space to improve the similarity measurement.

This paper is organized as follows. In Section 2, two high-dimensional indexing methods based on region approximation approach are briefly presented. Then the proposed KRA$^+$-Blocks is detailed in Section 3. Experiments and results are described and commented in Section 4. Section 5 is dedicated to our conclusion.

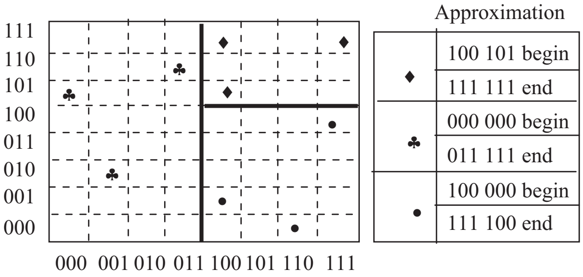
2. High-dimensional indexing methods based on region approximation

2.1. RA-Blocks

As already mentioned, in RA-Blocks data space is subdivided into hyper-rectangular cells. Each dimension d_i is split into 2^{b_i} intervals, which are encoded with b_i bits. Then, data space is subdivided into compact and disjoined regions having the same capacity (the maximum number of vectors that can be accommodated in a disk page). Each region will be approximated by the two bit strings corresponding to the beginning (bottom-left cell) and the ending (top-right cell) of this region (Fig. 1). The strategy of the data space partitioning is based on K-D-B-Tree algorithm [21].

k–NN search in the RA-Blocks is a two-phase process. The first one is known as *filtering* phase which consists in selecting the candidate regions based on their upper and lower distance bounds to the query vector. The upper (respectively lower) distance U (resp. L) is computed as the maximum (resp. minimum) between the distance from a query q to the beginning of the region and from this query to the ending of the region as shown in Fig. 2.

The second phase (access to original vectors) selects vectors contained in the candidate regions based on their Euclidean distances to the query. The *k–NN* search algorithm of the RA-Blocks is given in Fig. 3.



♦	100 101 begin
	111 111 end
♣	000 000 begin
	011 111 end
•	100 000 begin
	111 100 end

Fig. 1. Partitioning of the data space (d=2, capacity=3).

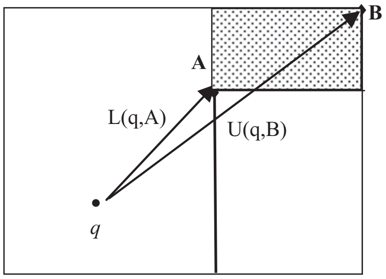
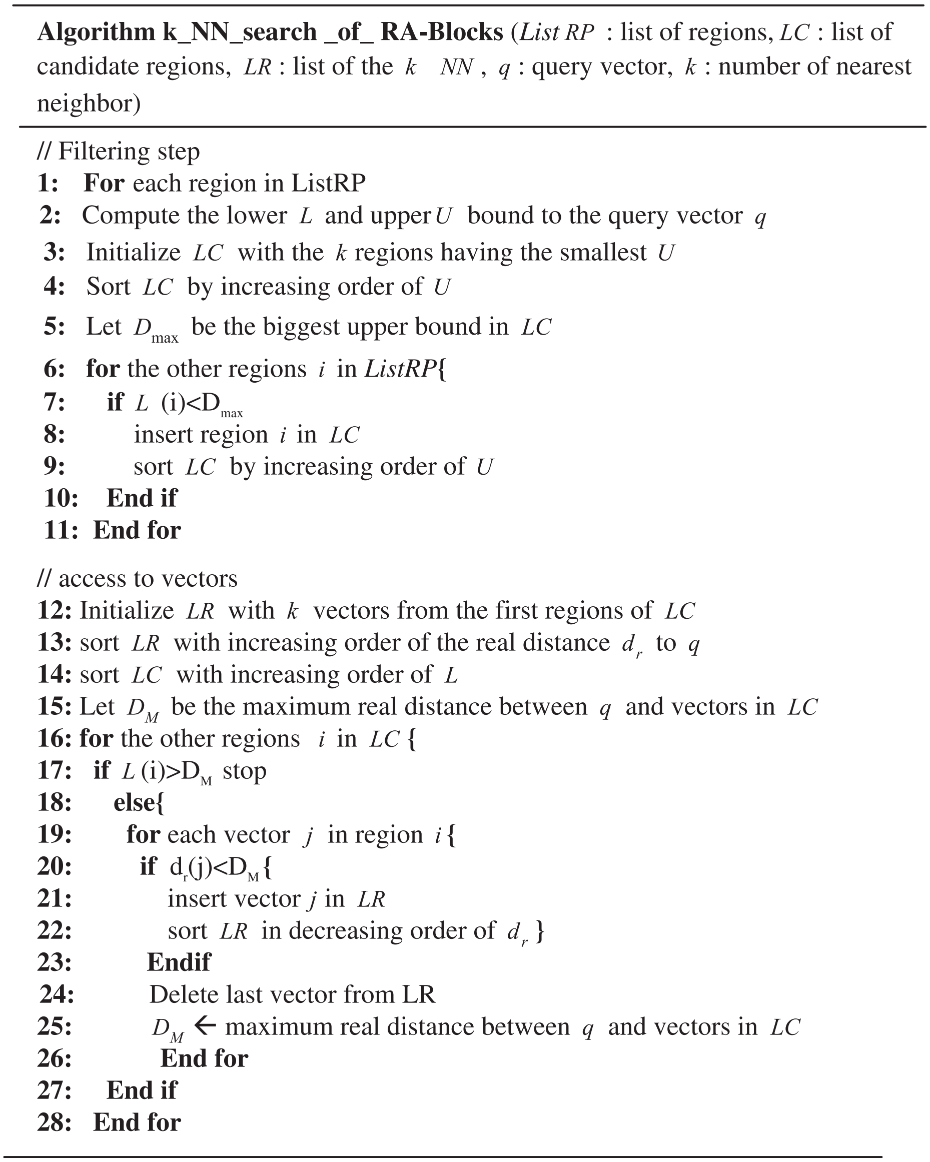


Fig. 2. Upper and lower distance bounds.

The RA-Blocks outperforms other methods based on the approximation vector approach in very high-dimensional spaces. However, it presents some limitations in the strategy of partitioning, which is used according to the K-D-B-Tree algorithm. It generates a significant number of empty regions and regions containing few vectors compared to their capacity. This increases significantly the number of regions to be treated during the filtering step involving a relatively large approximation file, and therefore, an increase in CPU time due to the lower and the upper distance bounds computing. To overcome these limitations, we have developed in a recent work [20] an improved region approximation block method, called RA$^+$-Blocks, which defines a better strategy of subdivision.

2.2. RA$^+$-Blocks

The RA$^+$-Blocks [20] is inspired from RA-Blocks and the main difference between the two methods is the partitioning strategy of the data space into regions. The splitting algorithm of the RA$^+$-Block is inspired by the K-D-B-Tree as it uses the same strategy for finding the splitting element. Splitting regions according to the splitting element, results two new regions containing the same number of vectors. The obtained regions will be approximated by two bit strings corresponding to the bottom-left and top-right cells, which will constitute the approximation file. Let us note that RA$^+$-Blocks splitting algorithm is based on the improvement of the internal nodes partitioning (downward subdivision) of the K-D-B-Tree structure. The main drawback of this structure is the fact that, when partitioning is performed, it may produce an important reorganization of the tree in order to preserve the properties of the K-D-B-Tree structure. Often



Algorithm k_NN_search _of_ RA-Blocks (*List RP* : list of regions, *LC* : list of candidate regions, *LR* : list of the *k NN* , *q* : query vector, *k* : number of nearest neighbor)

```
// Filtering step
For each region in ListRP
Compute the lower $L$ and upper $U$ bound to the query vector $q$
 Initialize $LC$ with the $k$ regions having the smallest $U$
 Sort $LC$ by increasing order of $U$
 Let $D_{max}$ be the biggest upper bound in $LC$
 for the other regions $i$ in $ListRP${
    if $L$ (i)<$D_{max}$
        insert region $i$ in $LC$
        sort $LC$ by increasing order of $U$
    End if
  End for

// access to vectors
Initialize $LR$ with $k$ vectors from the first regions of $LC$
sort $LR$ with increasing order of the real distance $d_r$ to $q$
sort $LC$ with increasing order of $L$
Let $D_M$ be the maximum real distance between $q$ and vectors in $LC$
for the other regions $i$ in $LC$ {
  if $L$ (i)>$D_M$ stop
    else{
      for each vector $j$ in region $i${
        if $d_r$(j)<$D_M${
            insert vector $j$ in $LR$
            sort $LR$ in decreasing order of $d_r$ }
         Endif
         Delete last vector from LR
         $D_M$ ← maximum real distance between $q$ and vectors in $LC$
          End for
    End if
  End for
```

Fig. 3. *k–NN* search algorithm of RA-Blocks method.

it creates a large number of empty leafs which cannot guarantee an optimal use of data space, and thereafter, increases the response time of *k–NN* search. Hence, the basic idea of the partitioning strategy in the RA^{+}-Blocks is to partition only the overflowed regions without using any tree structure, setting a difference with the K-D-B-Tree method, which actually divides all the regions that have an intersection with the splitting element. On one hand this guarantees, in most cases, that the *k–NN* are present in the same disk page and, on the other hand, it reduces the total number of the obtained regions and thereafter reduces the I/O time. An example of data space partitioning into two dimensions is presented in Fig. 4, where region capacity value is 3. As we can see, only the overflowed data pages and their corresponding region pages are split (in Fig. 4, region page 2 is split and region page 3 is not).

The RA^{+}-Blocks outperforms both RA-Blocks and VA-File in large image databases even with a very high dimensionality. Comparison results between RA^{+}-Blocks and various indexing techniques in large image database can be found in [20].

Although RA^{+}-Blocks overcomes some difficulties of high-dimensional vectors space, it suffers from certain problems in CBIR applications. Particularly, the large-scale image database, and the use of non-uniform and heterogeneous data, corresponding to the extracted images attribute as color, texture and shape descriptors.

3. Kernel region approximation blocks (KRA^{+}-Blocks)

This section presents the KRA^{+}-Blocks, which can be seen as an extension of the regions approximation

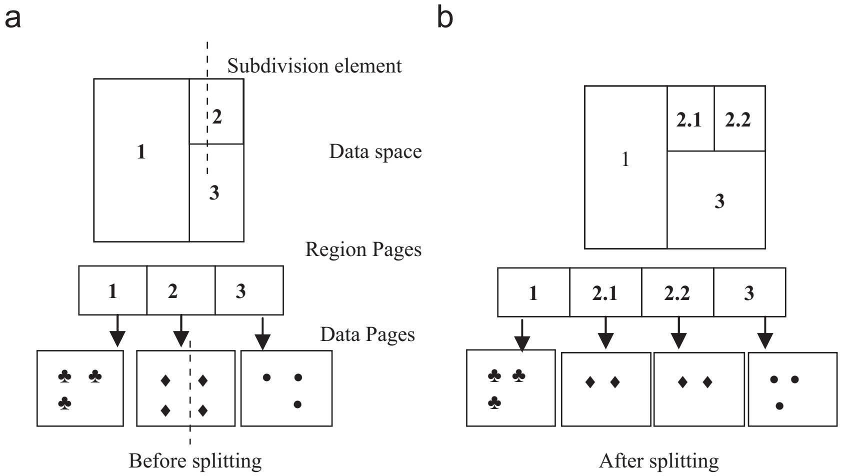


Fig. 4. Example of regions splitting according to the RA^{+}-Blocks (capacity=3).

approach for the kernel-based methods. The proposed approach combines a nonlinear dimensionality reduction technique with region approximation approach in feature space. It addresses the different issues that are related to CBIR, i.e., dimensionality reduction, indexing, similarity measure and relevance feedback.

3.1. Dimensionality reduction

Principal component analysis (PCA) and singular value decomposition (SVD) have been successfully used in image processing, dimensionality reduction, reconstruction and classification [22,23]. Generally, these methods work well when data exhibit linear correlation. In this case, a small number of eigenvalues may account for a large proportion of the data variance. However, when some nonlinear correlations exist in data, these correlations will not be detected by linear techniques using linear classifier. This happens for CBIR applications that deal with images, which are described with heterogeneous attribute as color, texture, shape, etc. Therefore, linear transformation techniques cannot offer effective or satisfactory representation of the heterogeneous data and better data representation can be expected by the use of nonlinear approaches. Kernel principal component analysis has been widely used in this case and has demonstrated excellent performances [24,25].

Given a data set S in an input space χ. KPCA maps S into a kernel space F, also called feature space, through a possibly nonlinear mapping ϕ, associated with a given kernel function, k, where $k(p,q)$=$\langle \Phi(p), \Phi(q)\rangle$, p and $q \in S$ and $\langle , \rangle$ denotes the dot product. KPCA finds the set of eigenvectors e_i, and their corresponding eigenvalues λ_i that satisfy

$$Me_i = \lambda_i e_i \tag{1}$$

where M is the covariance matrix given by

$$M = \frac{1}{N}\sum_{p \in S} \Phi(p)\Phi(p)^t \tag{2}$$

with N is the cardinal of S. As shown in [26], the problem in Eq. (1) can be reformulated as

$$K\omega_i = \lambda_i \omega_i \tag{3}$$

where ω_i=$(\omega_{i,1}, \ldots, \omega_{i,N})$ is a vector so as

$$e_i = \sum_{p \in S} \omega_{i,rank(p)} \Phi(p) \tag{4}$$

and K is a symmetric matrix defined by

$$\underset{p,q \in S}{K} = k(p,q)$$

Solving the problem in Eq. (3), ω_i is obtained, and the projection α_i^p of vector p along the ith kernel principal component is given by

$$\alpha_i^p = \langle e_i, \Phi(p)\rangle = \sum_{q \in S} \omega_{i,rank(q)} k(p,q) \tag{5}$$

The most popular kernel functions are Gaussian and polynomial at different orders. The appropriate kernel function selection has been discussed in [27]. Gaussian radial basis function (GRBF) kernel is commonly used as kernel function [28], and is adopted in this paper. The GRBF kernel is defined as $k(p,q) = e^{-(\|p-q\|^2/2\delta^2)}$, where p and q are two vectors in the input space. In CBIR context, p and q are image descriptors containing heterogeneous attributes such as shape, color, etc. δ is known as the *width* of the GRBF, often set by users beforehand. This kernel parameter determines the nature of the nonlinear mapping $\Phi(\cdot)$ to the kernel space. The feature space dimensionality, denoted d, which corresponds to the number of eigenvectors that we keep in the reduced space, has also an influence on kernel data mapping. δ and d are two parameters to set when we deal with GRBF kernel. Thus, different values of these parameters will induce a series of different kernel spaces [29], and therefore different representation of the vectors. In the next section, the influence of δ and d on data representation is studied.

3.2. Similarity measure

The similarity measure is a very important concept in information retrieval system. It is used to compare two items and to determine whether they can be considered as similar or not. In order to compare images, it is necessary to define a similarity model whose choice is sensitive due to its impact on retrieval results of any CBIR system. To evaluate the similarity between images described by heterogeneous attributes, a nonlinear similarity model based on kernel functions is proposed. This approach does not provide a particular metric for each descriptor type, but a formal framework which offers many advantages.

In fact, the visual section of human brain uses a nonlinear processing system for tasks such as pattern recognition and classification [30], so the linear model is not relevant for the nonlinear nature of human perception. Moreover, nonlinear model allows the use of relevance feedback mechanism to refine the similarity model under human perception. Therefore, we propose to use a nonlinear model to simulate human perception for similarity search.

The kernel trick lets us define a nonlinear similarity measurement into a projected high-dimensional space F. Specifically, given two vectors p and q, the similarity function $k(p,q)$ is defined as the inner product of $\Phi(p)$ and $\Phi(q)$ where Φ is the function that maps the vectors p and q from χ to F. The inner product between two vectors can be considered as a measure of their similarity. Therefore the distance between p and q is defined as

$$\begin{aligned} d(p,q)^2 &= \langle \Phi(p)-\Phi(q), \Phi(p)-\Phi(q) \rangle \\ &= k(p,p)+k(q,q)-2k(p,q) \end{aligned} \tag{6}$$

In this work, we adopt *a* Gaussian radial basis function (GRBF) as a similarity model, so the nonlinear similarity model is defined as

$$k(p,q) = e^{-(\|p-q\|^2/2\delta^2)} = e^{-dist(p-q)^2/2\delta^2}$$

where the scaling parameter δ controls how rapidly the similarity $k(p,q)$ falls off with the distance between p and q in F, and $dist(p,q)$ is the distance in χ.

3.3. Indexing scheme

In order to minimize the number of disk accesses and the number of distances computation in the search step, a feature space needs to be well partitioned into clusters containing similar data and constituting the index structure. To create the index structure of the KRA$^+$-Blocks, we use both KPCA and regions approximation approach. Similarly to SVD, the KPCA maps data vectors in reduced feature space via a nonlinear kernel and then, one can capture the nonlinearity in the heterogeneous features. This offers a compact and informative image representation, which facilitates indexing mechanism.

The index structure is created by mapping data vectors to a feature space, implicitly defined by the kernel function choice. We start with a Gaussian kernel according to our similarity model.

To map data vectors into the feature space, a reduced set of orthogonal basis vectors "e": $\{e_0, \ldots, e_{d-1}\}$ is computed by resolving Eq. (3). Thus, for any point p in χ, its corresponding point $\Phi(p)$ in F, can be expressed by

$$\Phi(p) = \sum_{i=0}^{d-1} \alpha_i^p e_i + \alpha_d^p e^{\perp} \tag{7}$$

where $\alpha_i{}^p$ is the ith component of the vector p in F (Eq. (5)) and $e^{\perp}$ is orthogonal to the projection basis "e".

Thus, we can find the magnitude of the projection error $(\alpha_d{}^p e^{\perp})^T(\alpha_d{}^p e^{\perp})$ given by

$$(\alpha_d^p e^{\perp})^T(\alpha_d^p e^{\perp}) = \frac{\sum_{i=d}^{din} eign(i)}{\sum_{i=0}^{din} eign(i)} \tag{8}$$

where din and d are, respectively, the dimensionality of χ and F, and $eign()$ the eigenvalues of KPCA.

Based on the data projection, the feature space is subdivided into a set of intervals; specifically each dimension d_i of feature space is split into 2^{b_i} intervals where each interval is encoded with b_i bits. The feature space is then subdivided into regions according to the strategy used by the RA$^+$-Blocks, as described in Section 2.2 and [20]. Each region is approximated by two bit strings which constitutes the index structure, and which is used in the searching step to reduce the CPU cost.

3.4. Search step

Given a query vector q, and using Eqs. (6) and (7), the square distance between $\Phi(p)$ and $\Phi(q)$ in the feature space is given by

$$\begin{aligned} &\|\Phi(p)-\Phi(q)\|^2 \\ &\quad = \|(\sum_{i=0}^{d-1} \alpha_i^p e_i + \alpha_d^p e^{\perp}) - (\sum_{i=0}^{d-1} \alpha_i^q e_i + \alpha_d^q e^{\perp})\|^2 \\ &\quad = \underbrace{\sum_{i=0}^{d-1}(\alpha_i^p - \alpha_i^q)^2} + \underbrace{\alpha_d^p e^{\perp}(\alpha_d^p e^{\perp})^T + \alpha_d^q e^{\perp}(\alpha_d^q e^{\perp})^T - 2\alpha_d^p e^{\perp}(\alpha_d^q e^{\perp})^T} \end{aligned} \tag{9}$$

Thus, in order to determine the distance between two vectors p and q in F, we used the vectors coordinates in F (first term in Eq. (9)), and the projection error (second term in Eq. (9)) obtained by KPCA algorithm. The only unknown term is $2\alpha_d{}^p e^{\perp}(\alpha_d{}^q e^{\perp})^T$, which is equal to $2\sqrt{\alpha_d^p e^{\perp}(\alpha_d^p e^{\perp})^T}\sqrt{\alpha_d^q e^{\perp}(\alpha_d^q e^{\perp})^T}\cos\theta$, where $\cos\theta$ is the angle between the two vectors. The extreme values of this expression are used to generate the distance bounds. Instead of using the exact distance between two vectors, we compute an approximation by considering $\cos\theta$ equal to, respectively, $+1$ for the upper distance and -1 for the lower distance. Therefore, the upper distance and the lower distance are approximated, respectively, with their maximum and minimum values.

The similarity search in KRA$^+$-Blocks is similar to the RA$^+$-Blocks method [20], which is performed in two steps. In the first one, called the filtering step, the candidate regions are found. As mentioned before, the approximation file is sequentially scanned, and the candidate regions are selected according to their upper and lower distance bounds to the query vector. The upper distance bound U and the lower distance bound L are then given by

$$
{}^{U2}(q,B)=\sum_{i=0}^{d-1}(\alpha_i^q-\alpha_i^B)^2+\alpha_d^q e^{\perp}(\alpha_d^q e^{\perp})^T+\alpha_d^B e^{\perp}(\alpha_d^B e^{\perp})^T
+2\sqrt{\alpha_d^q e^{\perp}(\alpha_d^q e^{\perp})^T}\sqrt{\alpha_d^B e^{\perp}(\alpha_d^B e^{\perp})^T} \qquad (10)
$$

$$
L^2(q,A)=\sum_{i=0}^{d-1}(\alpha_i^q-\alpha_i^A)^2+\alpha_d^q e^{\perp}(\alpha_d^q e^{\perp})^T+\alpha_d^A e^{\perp}(\alpha_d^A e^{\perp})^T
+2\sqrt{\alpha_d^q e^{\perp}(\alpha_d^q e^{\perp})^T}\sqrt{\alpha_d^A e^{\perp}(\alpha_d^A e^{\perp})^T} \qquad (11)
$$

where B and A, respectively, are the beginning and the ending vertices of each region (Fig. 2) with

$$
B=\sum_{i=0}^{d-1}\alpha_i^B e_i+\alpha_d^B e^{\perp}, A=\sum_{i=0}^{d-1}\alpha_i^A e_i+\alpha_d^A e^{\perp}, \Phi(q)
=\sum_{i=0}^{d-1}\alpha_i^q e_i+\alpha_d^q e^{\perp}
$$

During the second step, the original vectors are accessed. The k–NN are determined by computing the kernel distance (Eq. (9)) between query vector q, and vectors contained in the candidate regions.

3.5. *Relevance feedback scheme*

To bridge the semantic gap between the high-level user intention and low-level heterogeneous descriptors, the concept of relevance feedback (RF) has been proposed. As a response to a submitted query, the system provides a set of images, and then the user is asked to designate specific images as positive or negative. Positive means that the image contains the semantic concepts queried by the user and negative indicates that the image does not contain such concepts. Two of the most classical approaches in CBIR systems consist in modifying the image query or the similarity model, accordingly to positive, and/or negative images designated by the user [31,32].

In our approach, we propose the creation of a new feature space, where relevant vectors are brought closer to each other, and irrelevant ones are moved far from relevant vectors. This can be achieved by the use of adaptive quasi-conformal kernel (AQK) [33]. Assuming that p and p' are two descriptors related to two images I_p and $I_{p'}$, this method creates a new kernel function derived from the previous one given by

$$
\tilde{k}(p,p')=c(p)c(p')k(p,p') \qquad (12)
$$

where $c(p)$=$(P_r(I_p/I))/(P_r(I_p/R))$,
with

$$
P_r(I_p/I)=\frac{1}{NI}\sum_{I_q\in I}((NI+NR)-r)e^{-(\|p-q\|^2/2\delta^2)}
$$

and

$$
P_r(I_p/R)=\frac{1}{NR}\sum_{I_q\in R}((NI+NR)-r)e^{-(\|p-q\|^2/2\delta^2)}
$$

where R and I are, respectively, the sets of relevant and irrelevant images given by the user feedback, NR and NI are, respectively, the numbers of relevant and irrelevant images and r the rank of image I_q.

Based on the AQK Metric, a new distance is deduced by

$$
dist_{AQK}^2(p,q)=\sum_{i=0}^{d-1}(c(q)\alpha_i^q-c(p)\alpha_i^p)^2+c(q)^2(\alpha_d^q e^{\perp})^T\alpha_d^q e^{\perp}
+c(p)^2(\alpha_d^p e^{\perp})^T\alpha_d^p e^{\perp}-2c(p)c(q)(\alpha_d^p e^{\perp})^T\alpha_d^q e^{\perp} \qquad (13)
$$

The upper and lower distance bounds for AQK can be expressed by

$$
U^2(q,B)=\sum_{i=0}^{d-1}(c(q)\alpha_i^q-c(B)\alpha_i^B)^2+c(q)^2\alpha_d^q e^{\perp}(\alpha_d^q e^{\perp})^T+c(B)^2\alpha_d^B e^{\perp}
\times(\alpha_d^B e^{\perp})^T+2c(q)c(B)\sqrt{\alpha_d^q e^{\perp}(\alpha_d^q e^{\perp})^T}\sqrt{\alpha_d^B e^{\perp}(\alpha_d^B e^{\perp})^T} \qquad (14)
$$

$$
L^2(q,A)=\sum_{i=0}^{d-1}(c(q)\alpha_i^q-c(A)\alpha_i^A)^2+c(q)^2\alpha_d^q e^{\perp}(\alpha_d^q e^{\perp})^T+c(A)^2\alpha_d^A e^{\perp}
\times(\alpha_d^A e^{\perp})^T-2c(q)c(A)\sqrt{\alpha_d^q e^{\perp}(\alpha_d^q e^{\perp})^T}\sqrt{\alpha_d^A e^{\perp}(\alpha_d^A e^{\perp})^T} \qquad (15)
$$

The new kernel uses the information provided by the user feedback to create new distances which are used to select the relevant regions to the query (Eqs. (14) and (15)), then to find the k images closest to the query image (Eq. (13)). Thanks to the new kernel, the spatial resolution will be expanded around irrelevant vectors and contracted around relevant ones [33].

4. Experimental results

To evaluate the quality of the KRA$^+$-Blocks similarity search, three databases have been used. The first one is the well-known COIL-100 image database of Columbia University [34]. It contains 100 object classes, and the 72 images of each class are generated by rotating the object at an interval of 5° (Fig. 5).

The second one is a synthetic descriptor database that we create starting from COIL-100, in order to get a larger database (40,000) with the same number of classes.

The third one is also a synthetic descriptor database, whose size is 2,200,000, and where descriptors have been created randomly, and distributed uniformly in the range [0–1] with a dimension of 250.

The first two databases are suitable for accuracy evaluation while the last one has been used to study the response time of the proposed approach.

As we aimed to work with high-dimensional and heterogeneous descriptors, color and shape are used to describe the images, and heterogeneous vectors of 252 values are computed. For color descriptor, we use LAB histogram [35], which is quantized upon 192 bins, and RGB dominant colors, spatial consistency, and percentage of colors [36] upon a vector of 25 bins. As we can see in Fig. 5, color is discriminatory for our test databases. Angular radial transform (ART) [37] is used as shape descriptor, which is also well adapted to COIL-100 database, as each image contains one single object on a black background. The final image descriptor is a vector of 252 components (217 for color and 35 for shape).

An example of COIL-100 image databases is presented in Fig. 5. As it can be seen, these databases contain several and various objects with different views. This will induce some ambiguous situation when trying to retrieve all the images related to a query. On one hand, some particular views can be considered as similar, even though they are related to different classes and different object, and on the other hand, within the same class, dissimilar views related to the same object can be found.

The precision/recall curve used shows the results, and it is computed up to R_{max}=2 · *Card*, where *Card* represents the classes cardinal.

Experiments have been conducted to evaluate both the optimal kernel parameters and the performances of the KRA^{+}-Blocks method. Three of them are dedicated to study the kernel parameters and to evaluate their impact on classification and retrieval. The other experimentations compare retrieval performances with other methods.

4.1. An illustrative example

As described previously, KPCA deals with nonlinear transformation via nonlinear kernel functions. In the used kernel function (GRBF), there are two parameters d and δ that must be predetermined, knowing that they have significant impact on image representation in feature space. Ideally, compact and informative image representation will facilitate the indexing mechanism and the search process. An illustrative example of the influence of d and δ on the classification task is presented below.

Three clusters corresponding to three different classes are built (Fig. 6a), and samples for each cluster are generated randomly. Please note that the third cluster contains two sub-clusters. The number of vectors for each class is 40 and the dimensionality of each vector is 10. A common representation should capture the principal structure of the data set, i.e., the three clusters, while a better one should be able to capture also the two sub-clusters of the third class.

PCA and KPCA are performed on this data set and different values of δ are used for KPCA. For a comprehensive illustration, only the three first principal components (PC) are used to plot data. Fig. 6a shows the original data and Figs. 6b–f, show the projected data on the first, second and third principal component obtained by PCA and KPCA with δ=0.01, 0.1, 0.5 and 1. Plots have been rotated so as to obtain the best point of view and, therefore, to allow a better understanding.

We can easily see from these figures that the parameter δ has a significant effect on class separability. The KPCA using an appropriate δ can improve the classification performances compared to PCA (Fig. 6b). As we can see in this example, for δ=0.01, KPCA separates correctly the two sub-clusters related to the third class (represented by squares). However, when the value of δ increases (Fig. 6d–f), data representation becomes coarser, so these two sub-clusters are merged. Thus, better data representation can be achieved when KPCA is using the appropriate value of δ.

When data are projected, the user's expectations are often a good separation, a significant dimension reduction and no loss of information. Actually, the last two points are in conflict. Generally, the top principal eigenvectors capture the major variance while the remaining ones correspond to the less significant details. However, the amount of variance contained in the top PC depends both on d and δ parameters as illustrated in Table 1. Therefore, the values of the kernel parameters δ and d, influence widely the relevance of the projection that will be used to represent data in the feature space.

4.2. Kernel parameters estimation

To select the best values of d and δ, respectively, the dimensionality (i.e., the number of eigenvectors) and the

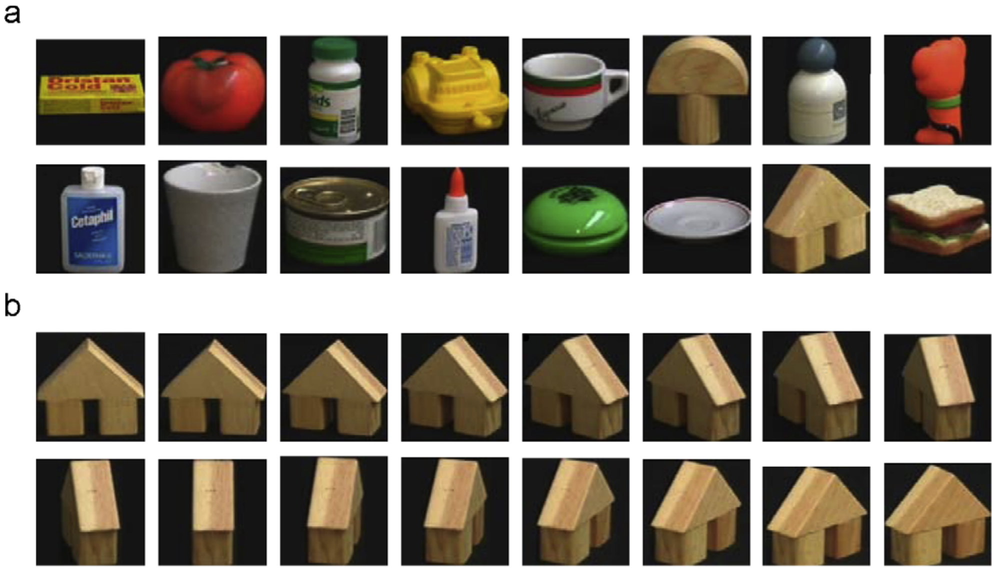


Fig. 5. An example of images (a) and classes (b) of COIL-100 database.

width of RGBF, two parameters γ (performance indicator) and σ (sum of the data variance in top) are used as a criteria.

The performance indicator parameter γ is defined as

$$\gamma = \frac{Intercluster}{Intracluster} \tag{16}$$

where *Intercluster* and *Intracluster* are, respectively, the inter cluster and intra cluster distances. *Intercluster* represents the separation ability, and can be calculated as the mean distances between the two closest vectors those are belonging to two different clusters. *Intracluster* represents the clustering ability, and can be computed as the mean of the average distances among all the clusters, between one cluster vector and its gravity center.

The parameter σ is given by$\sigma = \sum_{i=0}^{d-1} eign(i)$, where d is the dimensionality of the feature space and $eign()$ are the eigenvalues of KPCA.

Note that a large values of γ and σ are obtained for, respectively, better class separation and small loss of information.

To show the effect of the two kernel parameters d and δ on γ and σ, we conducted some tests on COIL-100 image

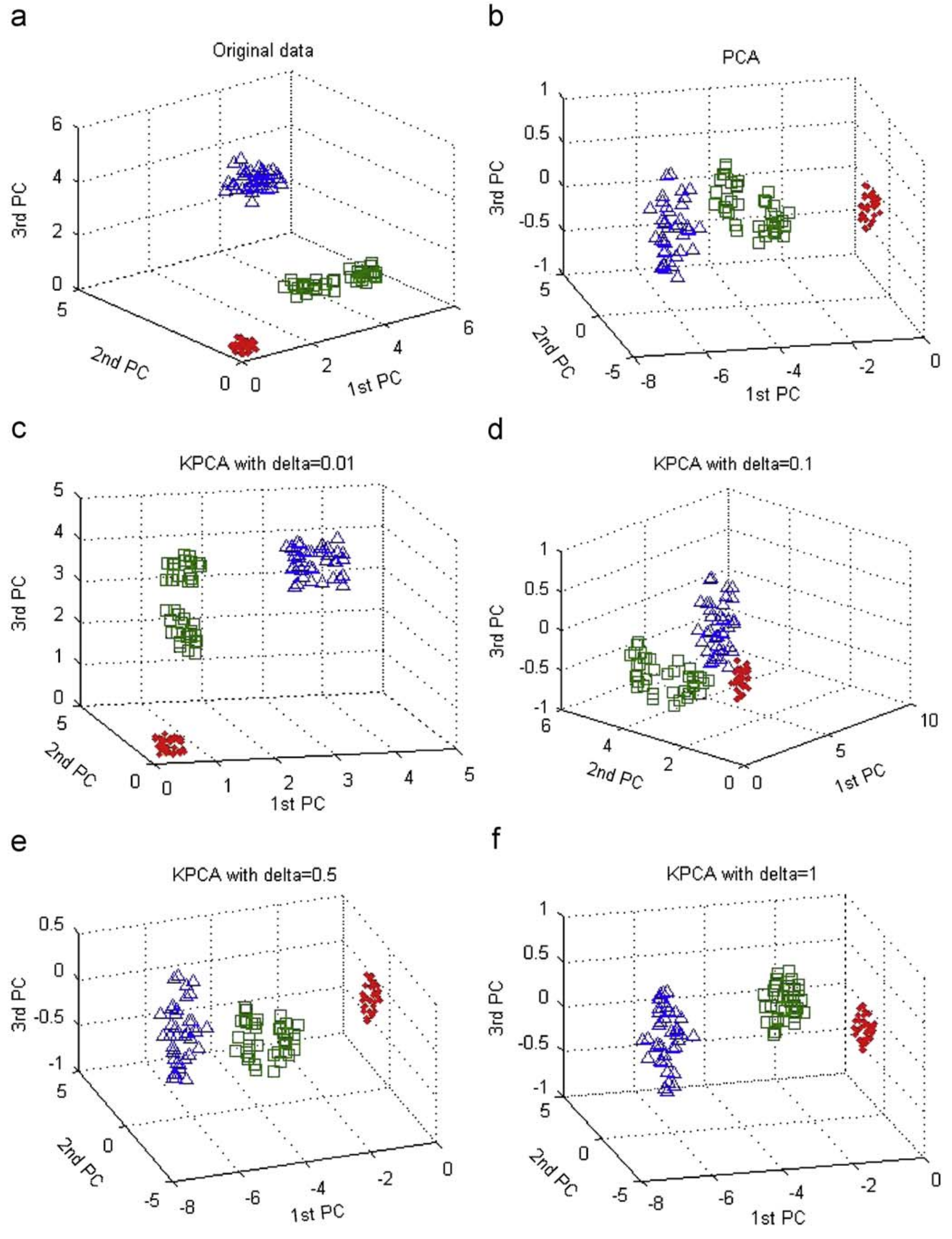


Fig. 6. PCs for different δ values. (a) Original data. (b) PCA. (c) KPCA, δ=0.01. (d) KPCA, δ=0.1. (e) KPCA, δ=0.5. (f) KPCA, δ=1.

database, where the values of γ and σ are computed for different values of d and δ. The parameter δ varies from 1 to 100 according to a logarithmic scale, and the dimensionality d varies from 2 to 252 according to a linear scale, where 252 is the size of image descriptors.

Figs. 7a and b show, respectively, γ and σ variations for different values of d (lines) and δ (columns), where pixels darkness corresponds to γ and σ low values. They also show the iso-parameter curve corresponding to a fixed value of σ and γ

As we expected, large values of σ and γ, corresponding to better performances, are obtained for large values of d and δ. However, from previous sections, it was shown that a large value of d induces the dimensionality curse problem and large value of δ gives poor data representation. Consequently, a compromise must be found when fixing those parameters.

In order to find the optimal values of (d, δ), we first select all the couples whose corresponding value of γ verifies

$$\gamma_{d,\delta} \geq 98\% \gamma_{max} \tag{17}$$

where γ_{max} is the maximum value of γ. In Fig. 7a, these points are located above the iso-parameter curve, which correspond to $\gamma_{d,\delta}$=98%γ_{max}

In the second step, we select from these couples, those whose variances verifies

$$\sigma_{d,\delta} \geq 98\% \sigma_{max} \tag{18}$$

where σ_{max} is the maximum value of σ.

Table 2 gives the corresponding verified equation for each region (from 1 to 4 in Fig. 7c).

Table 1
Sum of variance in the top PCs according to parameter δ.

δ	PC1	PC1 to PC2	PC1 to PC3	PC1 to PC4	PC1 to PC5	PC1 to PC6
0.01	0.11	0.22	0.33	0.44	0.56	0.67
0.10	0.11	0.22	0.33	0.44	0.56	0.67
0.50	0.23	0.40	0.53	0.63	0.72	0.82
1	0.57	0.73	0.82	0.87	0.91	0.95
2	0.85	0.92	0.95	0.97	**0.98**	**0.99**
3	0.93	0.96	**0.98**	**0.99**	**0.99**	**0.99**

Bold values corresponds to the variance sum greater or equal to 98%.

Thus, the optimal couples are located in region 4. As our objective is to select among the obtained couples those having the lowest values of d and δ, the selected ones are located around the intersection point (the circle in Fig. 7c).

4.3. Application to image retrieval

The following experiment illustrates the influence of the kernel parameters on the effectiveness of retrieval on COIL-100 image database. RP curve is plotted for five different values of (d, δ), one value per region in Fig. 7c and one value corresponding to the intersection point that we consider as the value which guarantees the best compromise. In Fig. 8, we can easily see the influence of (d, δ) on

Table 2
The corresponding verified equation for each region.

Region number	Eq. (17)	Eq. (18)
1	No	No
2	No	Yes
3	Yes	No
4	Yes	Yes

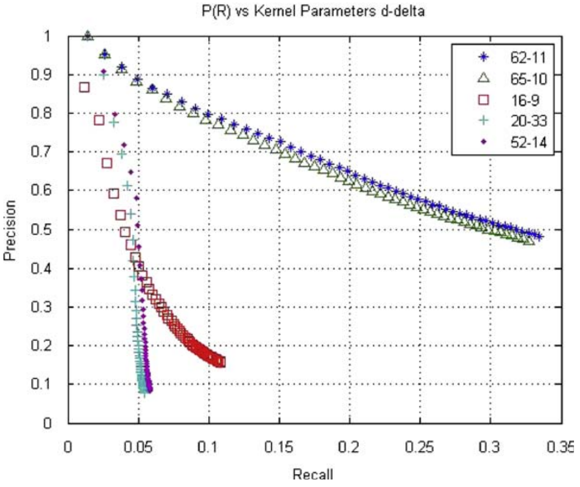


Fig. 8. Precision–recall curves for different values of kernel parameters.

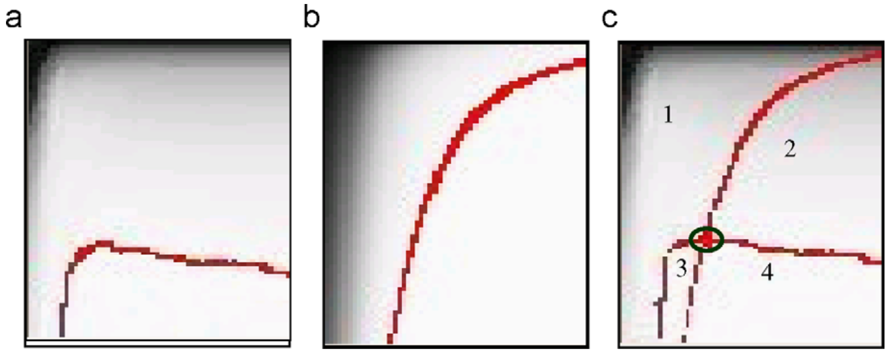


Fig. 7. (a) $\gamma(d, \delta)$, (b) $\sigma(d, \delta)$ and (c) optimal values of kernel parameters.

retrieval performances. The best results are obtained for (d=62, δ=11) (intersection point) and (d=65, δ=10) (region 4 in Fig. 7c). Examples are presented in Fig. 13.

Table 3
The four methods used for similarity search quality comparison.

		SCAN: linear SCAN	PCA: RA$^+$ Blocks using PCA	RA$^+$-Blocks	KRA$^+$-Blocks
Distance	Euclidian		X	X	
	Kernel	X			X

From this experiment, we note that the kernel parameters have a significant influence in the retrieval results. In fact, as shown before, a better data representation in the feature space is obtained when using appropriate kernel parameters. This allows the KRA$^+$-Blocks index to group similar data vectors in the same region and then to perform *k–NN* search efficiently.

4.4. Similarity search quality

This experiment aims to show the retrieval quality effectiveness of the KRA$^+$-Blocks. To study the performance of kernel-based similarity measurement, 600

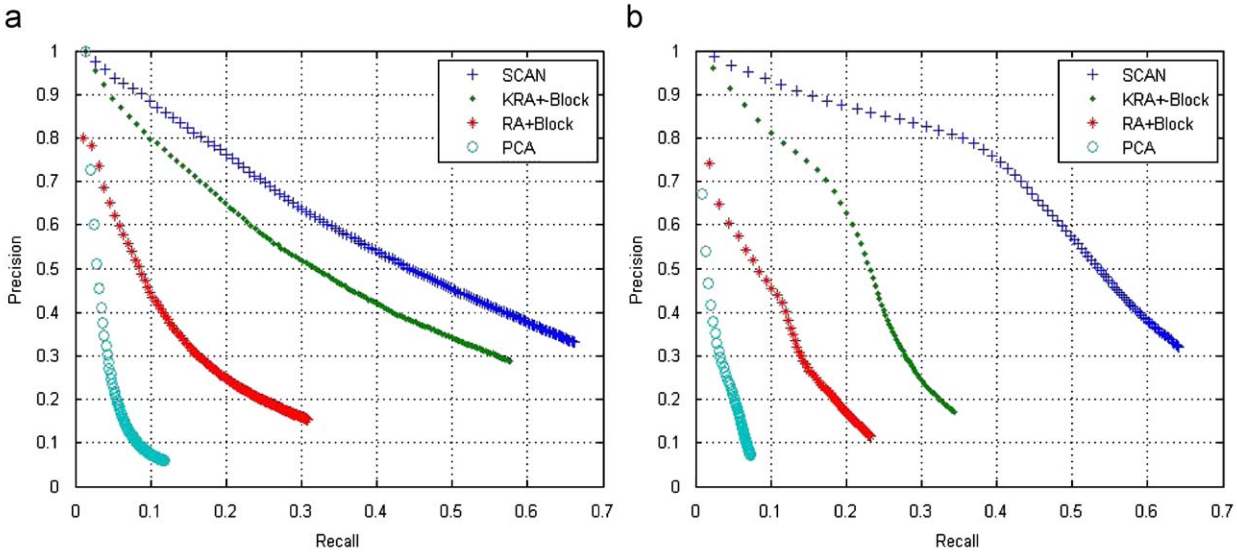


Fig. 9. Precision–recall curves using (a) COIL-100 database and (b) extended COIL-100 database (40,000).

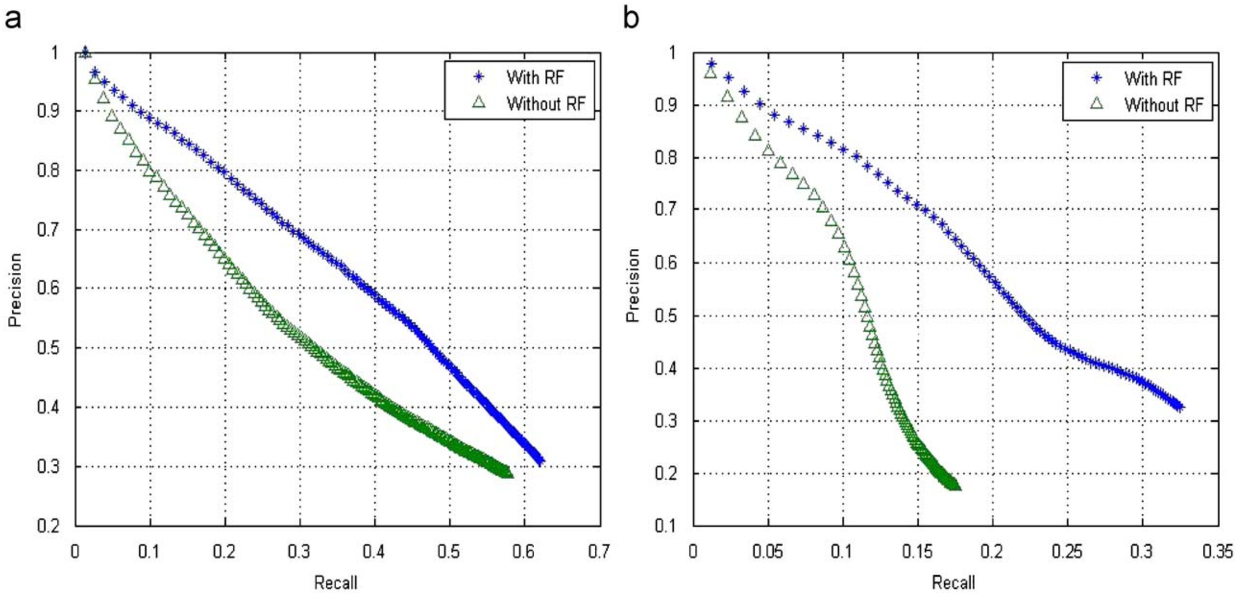


Fig. 10. Precision–recall using (a) COIL-100 database and (b) extended COIL-100 database (40,000).

queries are performed on the COIL-100 database, using the four different approaches SCAN, PCA, RA$^+$-Blocks and KRA$^+$-Blocks as described in Table 3.

Linear scan is considered as a reference to test the new high-dimensional indexing methods. To be under the same conditions as in KRA$^+$-Blocks, linear scan was performed after applying a nonlinear dimensionality reduction (KPCA) using kernel distances.

Optimal kernel parameters are computed as described in Section 4.2. The precision/recall curve is used to evaluate the effectiveness of similarity retrieval and some examples are presented in Fig. 14. This test has been conducted for two different sizes of database: 7200 and 40,000. As we can see from Fig. 9, the proposed kernel approach performs better than RA$^+$-Blocks and PCA/RA$^+$-Block as it improves both data indexing and similarity model. It is clear that the accuracy of the sequential search is better than our method on both COIL-100 database and extended COIL-100 database. This may be explained by the fact that our method is an approximation of a full search; it tolerates deterioration in the quality of the search against a better gain in response time. Note that the degradation of the quality of research increases with the size of the database (Fig. 9(b)).

On one hand, as the studied methods are approximation based, data must be separated as well as possible during the filtering step. This separation is better achieved by a kernel approach as kernel parameters are deduced from the data itself. On the other hand, the use of kernel-based similarity model allows combining distances

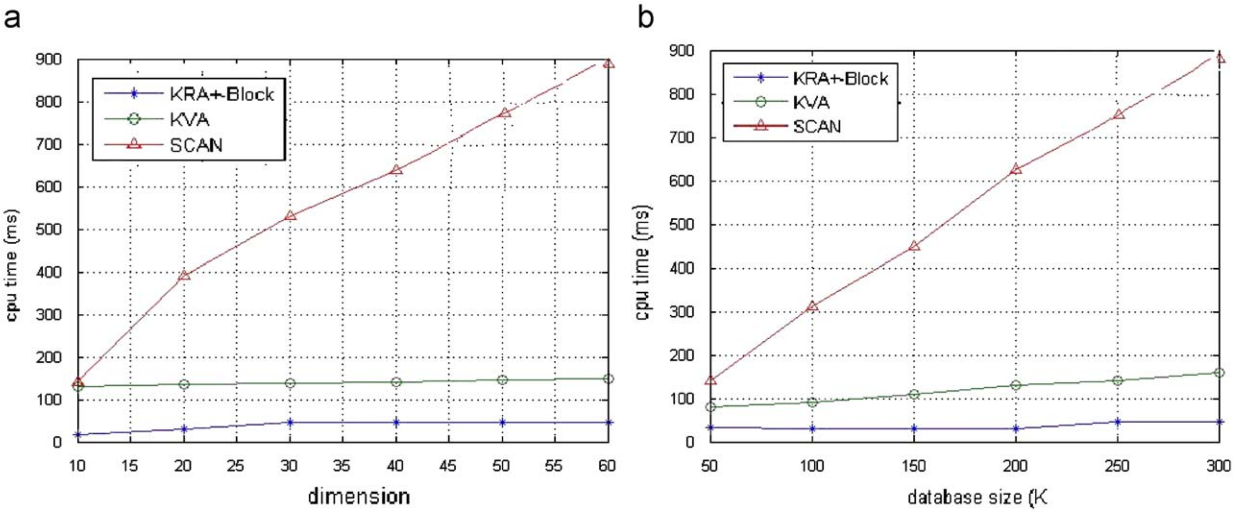


Fig. 11. Cost comparison for (a) 100,000 data vectors. (b) 60-dimensional data.

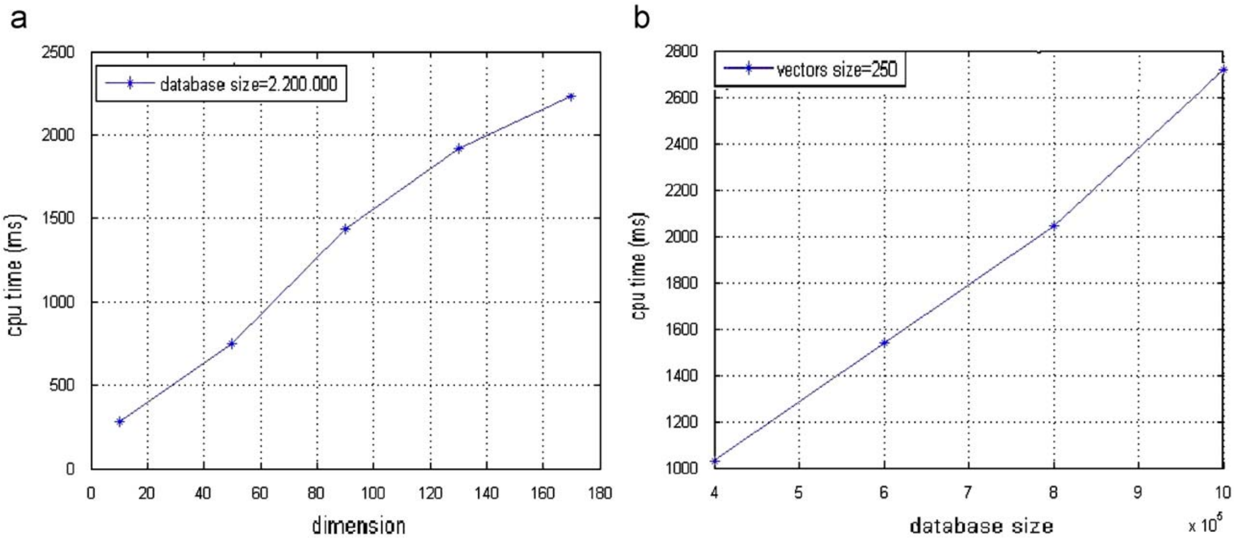


Fig. 12. Cost comparison for (a) 2,200,000 data vectors. (b) 250-dimensional data.

that are related to heterogeneous data in the same formalism.

As mentioned before, images that are belonging to the same class could have a very different aspect, which involves false positive images.

4.5. Relevance feedback

To improve the retrieval performances, KRA$^+$-Blocks is used with relevance feedback to find new kernel distances as described in 3.5. The experiments were performed on both COIL-100 database and extended COIL-100 database, and we fixed *k–NN*, respectively, to 144 and 400. 600 images from the database are used as queries in order to evaluate the average performances, and only one RF cycle is performed for each query. As our database image classes are well known, each retrieved image can automatically be marked as relevant or not, according to whether it belongs to the query class or not. PR curves in Fig. 10 reflect the achieved improvement through the proposed method and some examples are presented in Fig. 15. As we can see, from Fig. 15, the RF mechanism significantly improves the results, showing the relevance of the kernel distances used both in the filtering (Eqs. (14) and (15)) and the searching steps (Eq. (13)). Based on the relevant and irrelevant images, the relevance feedback estimates flexible kernel distances that are used to create new kernel space which offers a better data representation.

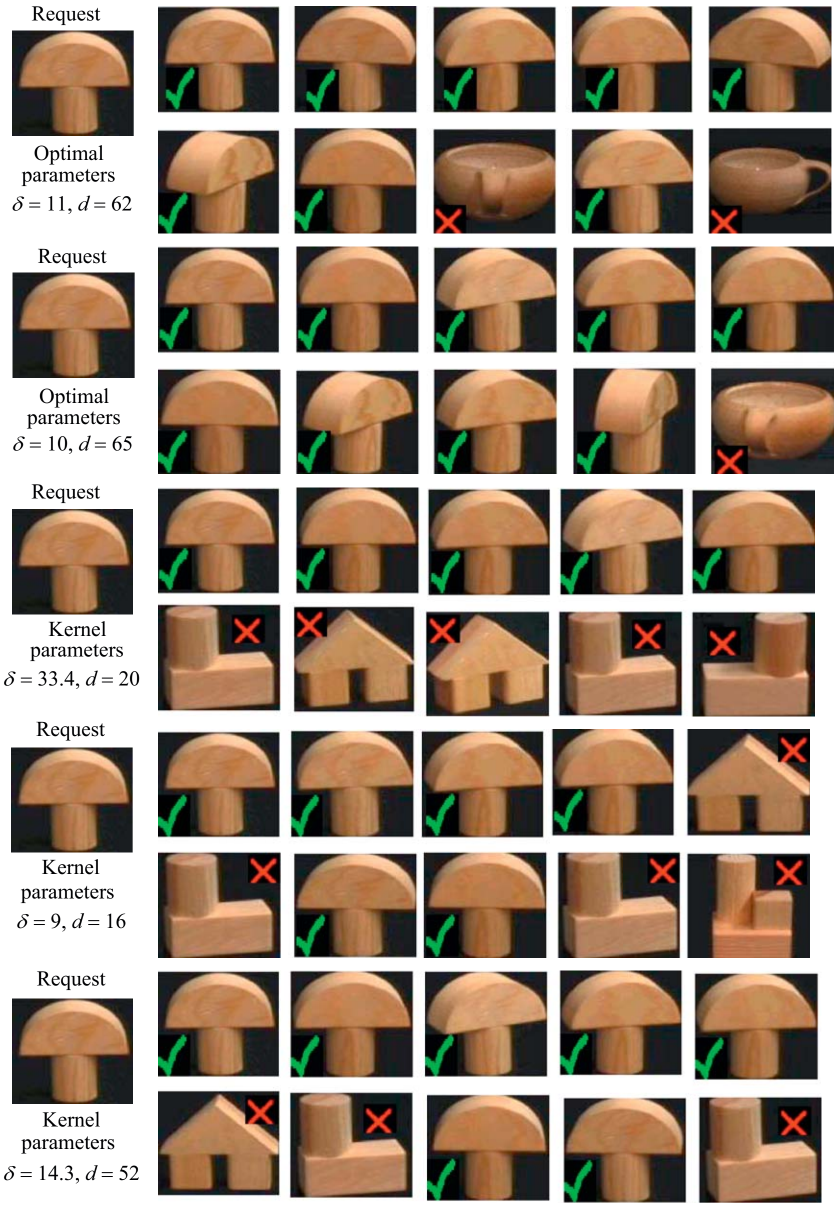


Fig. 13. The first 10 nearest neighbor images using KRA$^+$-Block method for different δ and d.

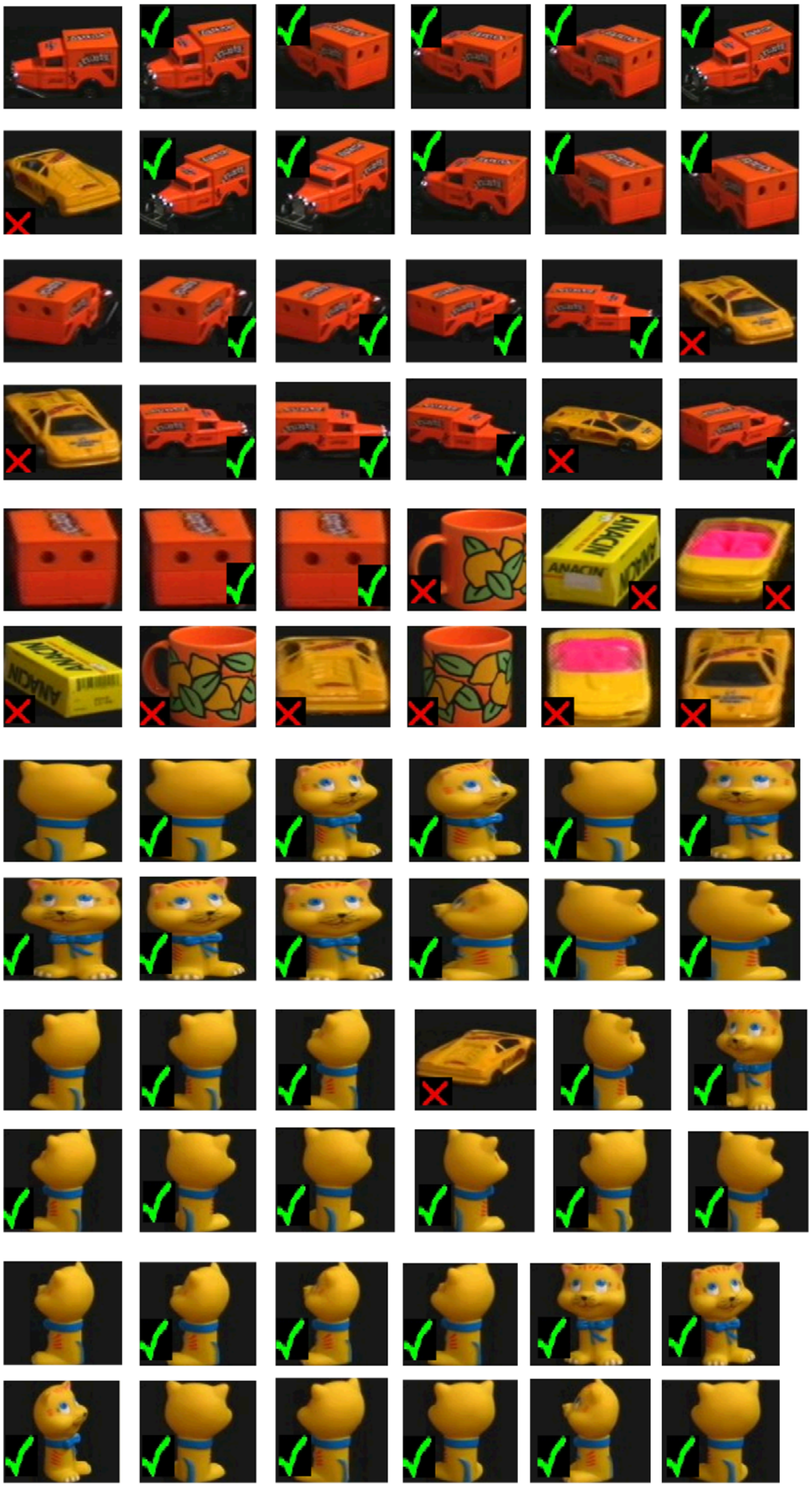


Fig. 14. Retrieval results using KRA^{+}-Block with selected kernel parameters for the COIL-100 images: first image of each row is the query image and the others are the top 11 resulting images.

Indeed, in this new space, the spatial resolution has been expanded around irrelevant vectors and contracted around relevant ones according to the user needs, involving the improvement of the retrieval process.

4.6. CPU time study

We conducted two experiments to show *k–NN* search performance of the KRA$^+$-Blocks in terms of response time. Both of them are conducted on a Microsoft Windows XP machine with 2.3 GHz CPU, 2Go RAM, and 250Go in local disk.

First, we compare KRA$^+$-Blocks, linear scan and KVA-File on a synthetic database with various dimensions and various numbers of vectors as shown in Fig. 11. We choose the VA-File approach since it is considered as a reference to test the new high-dimensional indexing methods based on approximation approach. The linear scan is also used for comparison because it outperforms all spatial access methods. This experiment has used the synthetic database, considering only 300,000 of 2,200,000 descriptors, and only 60 components instead of 250 for each descriptor. Here we fixed *K–NN*=10, capacity=10,000, disk page size=200 kB, and the bits number per dimension b_i=8.

As we can see in Figs. 11a and b, the KRA$^+$-Blocks performs better than the linear scan and KVA-File [38], when the dimensionality or the database size increases. In both cases, the ratio between KVA-File and KRA$^+$-Blocks response time seems to remain constant in this plot, and especially it remains lower than KVA-File.

Therefore, in the next experiment we evaluate the KRA$^+$-Blocks scalability versus vectors dimensionality and database size for large database (up to 2,200,000). We can see from Fig. 12 that the KRA$^+$-Blocks method maintains a lower CPU times while the database and the dimensionality increase. (Figs. 13–15)

Unlike most multidimensional indexing methods that exist in the literature, the response times of the proposed method has a linear cost according to the database size and the dimensionality. This is due to the partitioning strategy used to split the data space into compact regions. This reduces the dimensionality curse problem that affects most multidimensional indexing techniques.

This robustness to database size and dimensionality is closely linked to the use of regions' approximations and to the nonlinear dimensionality reduction strategy. Indeed, the KRA$^+$-Blocks method uses the regions filtering approach to compute the *k–NN* of the query. Much computation time can be saved in retrieval step compared to the KVA-file method as only regions' lower and upper bounds have to be computed. Moreover, the neighboring vectors in KRA$^+$-Blocks structure are more likely to be in the same physical page or consecutive pages due to the preprocessing step. The aim of this step consists in finding the appropriate nonlinear mapping, which allows a better data separability and dimensionality reduction of the feature space. Therefore, the performances of the KRA$^+$-Blocks are better than the KVA-File and linear scan.

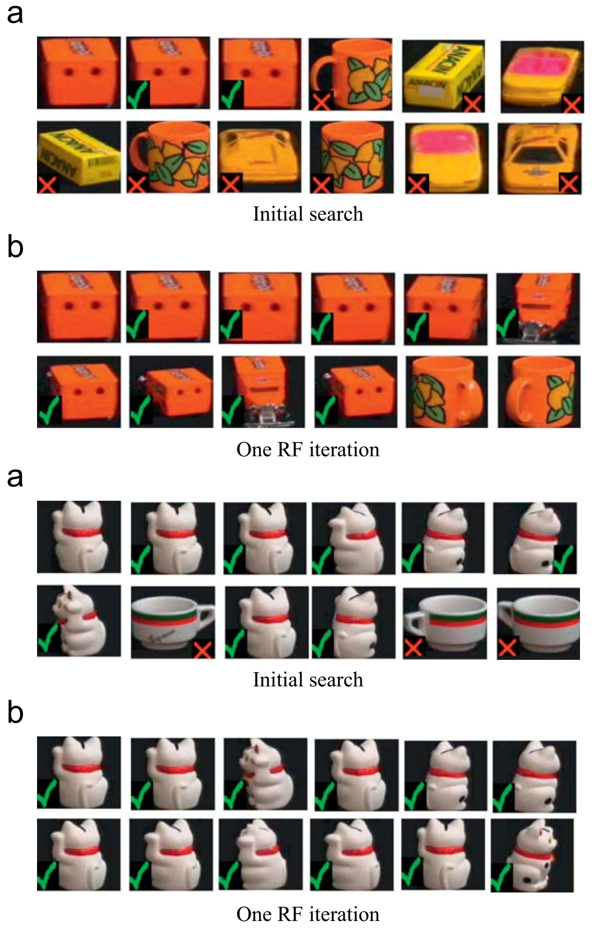


Fig. 15. Retrieval results using (a) KRA$^+$-Block with optimal kernel parameters and (b) using one RF iteration: first image of each row is the query image and the others are the top 11 resulting images.

5. Conclusion

This paper introduces KRA$^+$-Blocks as an extension of RA$^+$-Blocks for kernel-based methods, and combines nonlinear dimensionality reduction (KPCA) and region approximation. The proposed approach is well suited to CBIR applications as they deal with very large databases, high-dimensional vectors and heterogeneous data. We showed that the use of the region approximation approach can significantly decrease the number of distances computations and then the response times, while the use of the kernel approach helps to define new distances which are suitable to the heterogeneous nature of the descriptors. Since the kernel parameters have a great influence on data representation, we proposed a strategy to determine the optimal values which allows better data separability in feature space. This one is then used to build KRA$^+$-Blocks index structure and to perform the similarity measurement. Finally, an effective kernel distances are provided for kernel-based relevance feedbacks to adapt distances to the user needs.

Acknowledgement

This work was partly supported by the STIC France-Morocco program (Sciences et Technologies de l'Information et de la Communication).

References

[1] J.L. Bentley, Multidimensional binary search trees used for associative searching, Commun. ACM 18 (9) (1975) 509–517.

[2] A. Guttman, R-Trees: a dynamic index structure for spatial searching, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, Boston, MA, June 1984, pp. 47–57.

[3] N. Beckmann, H.-P. Kriegel, R. Schneider, B. Seeger, The R*-Tree: an efficient and robust access method for points and rectangles, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, Atlantic City, NJ, 1990, pp. 322–331.

[4] T.K. Sellis, N. Roussopoulos, C. Faloutsos, The R^+-Tree: a dynamic index for multi-dimensional objects, in: Proceedings of the 13th International Conference on Very Large Data Bases, September 1987, pp. 507–518.

[5] K. Lin, H.V. Jagadish, C. Faloutsos, The TV-Tree: an index for high dimensional data, VLDB J. 3 (4) (1994) 517–543.

[6] S. Berchtold, D. Keim, H.-P. Kriegel, The X-Tree: an index structure for high-dimensional data, in: Proceedings of the International Conference on Very Large Databases, 1996, pp. 28–39.

[7] D.A. White, R. Jain, Similarity indexing with the SS-Tree, in: Proceedings of the 12th International Conference on Data Engineering, New Orleans, Louisiana, USA, 1996, pp. 516–523.

[8] N. Katayama, S. Satoh, The SR-Tree: an index structure for high dimensional nearest neighbor queries, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, Tucson, Arizona, US, 1997, pp. 69–380.

[9] H. Wang, C.-S. Perng, The S^2-Tree: an index structure for subsequence matching of spatial objects, in: Fifth Pacific-Asic Conference on Knowledge Discovery and Data Mining (PAKDD), Hong Kong, 2001.

[10] K. Chakrabarti, S. Mehrotra, The hybrid Tree: an index structure for high dimensional feature spaces, In: Proceedings of the International Conference on Data Engineering, February 1999, pp. 440–447.

[11] K. Pearson, On lines and planes of closest fit to systems of points in space, Philos. Mag. 2 (6) (1901) 559–572.

[12] W.S. Torgerson, Multidimensional scaling: I. theory and method, Psychometrika 17 (1952) 401–419.

[13] T.F. Cox, M.A.A. Cox, 2nd ed., 'Multidimensional Scaling', vol. 88 of Monographs on Statistics and Applied Probability Chapman & Hall/CRC, London, 2000.

[14] S. Roweis, L. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (5500) (2000) 2323–2326.

[15] J. Tenenbaum, V. de Silva, J. Langford, A global geometric framework for nonlinear dimensionality reduction, Science 290 (5500) (2000) 2319–2323.

[16] R. Weber, H.-J. Schek, S. Blott, A quantitative analysis and performance study for similarity-search methods in high-dimensional space, in: Proceedings of the 24th VLDB Conference, USA, 1998.

[17] T. Chen, M. Nakazato, T.S. Huang, Speeding up the similarity search in multimedia database, Proc. IEEE ICME (2002).

[18] G.-H. Cha, X. Zhu, D. Petrovic, C.-W. Chang, An efficient indexing method for nearest neighbours searches in high dimensional image databases, IEEE Trans. Multimedia (2002) 76–87.

[19] G.H. Cha, C.W. Chung, The GC-Tree: a high-dimensional index structure for similarity search in image databases, IEEE Trans. Multimedia 4 (2002) 235–247.

[20] I. Daoudi, S.E. Ouatik, A. El Kharraz, K. Idrissi, D. Aboutajdine, Vector approximation based indexing for high-dimensional multimedia databases, Eng. Lett. 6 (2) (2008) 210–218.

[21] J.T. Robinson, The K-D-B-Tree: a search structure for large multi-dimensional dynamic indexes, in: Proceedings of the ACM SIGMOD, 1981.

[22] A.D. Bimbo, P. Pala, S. Santini, Image retrieval by elastic matching of shapes and image patterns, in: Proceedings of Multimedia, 1996, pp. 215–218.

[23] I.K. Fodor, A Survey Of Dimension Reduction Techniques, LLNL Technical Report, UCRL-ID-148494, June 2002.

[24] X. Dong, L. Wang, Q. Tian, A novel multi-resolution video representation schema based on kernel PCA, Visual Comput. 1 (2006) 357–370.

[25] S. Hichem, Kernel PCA for similarity invariant shape recognition, Neurocomput. Arch. 70 (16–18) (2007).

[26] B. Schôlkopf, A.J. Smola, K.-R. Mller, Kernel principal component analysis. Advances in Kernel methods, in: B. Schlkopf, C.J.C. Burges, A.J. Smola (Eds.), Support Vector Learning, MIT Press, Cambridge, 1999, pp. 327–352.

[27] V. Vapnik, Statistical Learning Theory, Wiley, New York, 1998.

[28] K.R. Muller, S. Mika, et al., An introduction to kernel-based learning algorithms, IEEE Trans. Neural Networks 12 (2) (2001) 181–202.

[29] Y. Chen, X. Zhou, T.S. Huang, One-class SVM for learning in image retrieval, in: Proceedings of IEEE ICIP, Thessalonica, Greece, 2001, 2001, pp. 815–818.

[30] S. Haykin, Neural Networks: A Comprehensive Foundation, Maxilla, New York, 1994.

[31] G. Ciocca, I. Gagliardi, R. Schettini, Quicklook2: an integrated multimedia system, Int. J. Visual Lang. Comput. 12 (2001) 81–103.

[32] J. Fournier, M. Cord, S. Philipp-Foliguet, Retin: a content based image indexing and retrieval system, Pattern Anal. Appl. 4 (2001) 153–173.

[33] D. Heisterkamp, J. Peng, H. Dai, An adaptive quasi-conformal kernel metric for image retrieval, in: Proceedings of IEEE CVPR, Kauai Marriott, Hawaii, 2001, pp. 236–243.

[34] http://www1.cs.columbia.edu/CAVE/software/softlib/coil-100.php.

[35] K. Idrissi, G. lavoué, J. Ricard, A. Baskurt, Object of interest based visual navigation, retrieval and semantic content identification system, Comput. Vision Image Understanding 94 (1) (2004) 271–294.

[36] B.S. Manjunath, P. Salembier, T. Sikora (Eds.), Introduction to MPEG-7, Wiley, 2002.

[37] W.Y. Kim, Y.S.Kim, A New Region-Based Shape Descriptor, TR# 15-01, December 1999.

[38] S. Yasushi, Y. Masatoshi, U. Shunsuke, K. Haruhiko, The A-Tree: an index structure for high-dimensional spaces using relative approximation, in:, Proceedings of the 26th International Conference on Very Large Data Bases (VLDB), 2000.