Indexing and Browsing of Color Images: Design Considerations

Christos Diou, Nikos Batalas, and Anastasios Delopoulos

Multimedia Understanding Group, Department of Electrical and Computer Engineering, Aristotle University of Thessaloniki, Greece diou@mug.ee.auth.gr, amv@ee.auth.gr, adelo@eng.auth.gr

Summary. This chapter deals with the various problems and decisions associated with the design of a content based image retrieval system. Image descriptors and descriptor similarity measures, indexing data structures and navigation approaches are examined through the evaluation of a set representative methods. Insight is provided regarding their efficiency and applicability. Furthermore the accuracy of using low dimensional FastMap point configurations for indexing is extensively evaluated through a set of experiments. While it is out of the scope of this chapter to offer a review of state of the art techniques in the problems above, the results presented aim at assisting in the design and development of practical, usable and possibly large scale image databases.

1 Introduction

The main goal of content based image retrieval research is to devise suitable representations of images in order to allow query and retrieval based on the visual properties of images instead of manually inserted user annotations. Often the queries themselves are images and the user expects similar images to be retrieved.

Significant research has been performed on image retrieval systems in the past few years and the promising results contributed to the development of the MPEG-7 standard [1, 2]. The ultimate goal of automatic semantic characterization of images based on their visual content remains largely unsolved (even though there are partially successful approaches under controlled environments e.g., [3–5]). Still, descriptions of images based on color, textures, shapes etc. provide adequate results for a user to begin a search.

Practical, real world image retrieval applications, however, have additional requirements. Queries must be answered fast, while at the same time the design and implementation must be scalable, allowing searches within large image datasets. This requirement implies that an efficient indexing mechanism must be employed for storing image descriptions in the database. Several data structures have been proposed to this end in the literature. Examples include a family of methods based on *kd*-trees [6, 7], hashing (e.g., [8]), P-Sphere trees [9] and others. While these approaches offer an improvement over the exhaustive search for results, they are less efficient than the corresponding indexing structures used for text. Furthermore, they cannot always be used in conjunction with high dimensional image descriptors. In fact, this is one of the major limiting factors preventing the creation of content based image databases at large scales (and ultimately, the Internet).

One may summarize the most important design issues that have to be resolved during the development of a content based image database system into the following:

1. The description of images.
2. The similarity measure for the selected description. Several image descriptors can be compared with more than one measure. One must choose the measure that provides the best retrieval results for the application at hand.
3. The indexing mechanism to be employed.
4. The proper number of dimensions for the target descriptors for practical retrieval times.
5. If needed, a method that will provide low dimensional feature vectors, given the original image descriptors.
6. The visualization and browsing interface.

This chapter discusses the above issues by providing evaluations of known methods, thus providing an overview of the design of an image retrieval system. For the description of images in Sect. 2, only global color descriptors are considered, the histogram and dominant color. For the dominant color a comparison between two distance metrics is performed that provides indications concerning the choice of a similarity measure.

The *kd*-tree is considered regarding the indexing of image descriptors in Sect. 3. It is illustrated that data structures of this kind are not efficient when the number of dimensions used by the image descriptors exceed a certain limit. Additionally, such indexing structures index points in a k-dimensional space but the dominant color descriptor does not define points in such a space.

Section 4 presents a solution to these problems, based on point configurations provided by methods such as Multidimensional Scaling and FastMap. Thus one can can derive low dimensional feature vectors from the original image descriptors, allowing the efficient use of indexing structures. In addition, point configurations allow the implementation of intuitive browsing interfaces by visualization of results on the 2- or 3-dimensional space. But the benefits of using compact, low dimensional image representations come with a certain cost in retrieval accuracy. Through a set of experiments, Sect. 5 attempts to quantify the deviation of retrieval results obtained through point configurations with respect to the ones obtained through the initial image descriptions. Finally, Sect. 6 summarizes the conclusions drawn from the presented evaluations.

2 Color Descriptors and Similarity Measures

The first step in the design of a content based image retrieval system involves the representation of images using a set of descriptors. These provide a compact description of visual cues (color, texture, shape) or interest points (e.g., SIFT [10]) that allows the definition of similarity measures between images. The following choices need to be made:

1. The visual cues or types of interest points that will be utilized (e.g., color and texture).
2. The descriptors for each cue.
3. The similarity measure to be used with each descriptor.
4. If more than one descriptors are used, a fusion strategy that will combine them into a single descriptor (e.g., [11]), or alternatively, combine the results of each similarity measure into a single value (e.g., [12]).

In this section it is assumed that a single global color descriptor will be used, thus an evaluation example is provided for issues 2 and 3 above.

2.1 Histogram and Dominant Color

Global color descriptors are used to describe color properties of an image independent of spatial color distribution. The most important descriptors of this form are the well-known histogram and dominant color descriptors.

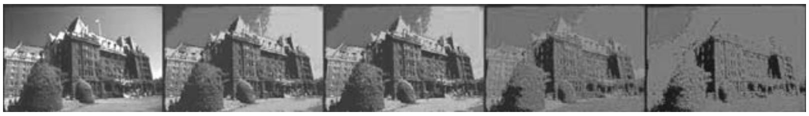
Both these descriptors can be given by

$$D = \{(c_i, p_i)\,, \quad i = 1, \ldots, N\}\,, \tag{1}$$

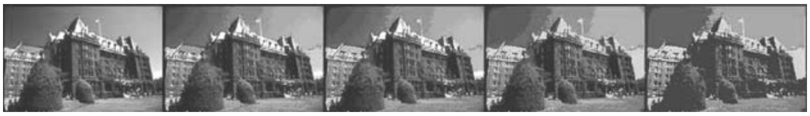
where c_i is a color from a predefined colorspace and p_i is the percentage of image pixels having that color. Of course, having a 3-channel 8-bits per channel image described by (1) is very impractical, since (1) $2N = 2^{25}$ values are used to describe a single image and (2) this level of granularity is not informative for the purpose of image retrieval. Therefore, images are quantized prior to extraction of D.

In histograms, the given colorspace (e.g., RGB) is usually quantized to a predefined number of “bins” independent of the images. While this approach can reduce the number of values required for D (the colors c_i are essentially predefined for a given colorspace), the description does not adapt to each image. Consider, for example, a 24-bit RGB image with only 64 different colors all at the same color region. Then, if the colorspace is uniformly quantized at $N = 27$ levels (three levels for each color channel), all these colors will be concentrated at a single bin; the rest of the 26 color-value pairs are left unused.

The dominant color descriptor, on the other hand, overcomes this issue by allowing the use of the more general form of (1) where the colors c_i and their number N can be different for each individual image. Naturally, a method for



(a)



(b)

Fig. 1. Color reduction using 64, 32, 16 and 8 colors: (**a**) using uniform color space quantization and (**b**) octrees

selecting the appropriate dominant colors c_i for each image must be defined. In [2] regarding the MPEG-7 standard color descriptor the use of the Generalized Lloyd Algorithm is proposed. In this work, a different approach utilizing octrees for color reduction [13] was used.

Figure 1 illustrates an example of colorspace quantization (histogram) and quantization adapted to the image (dominant color) using octrees. In the latter case a more accurate description of the image is derived for image retrieval purposes. It is therefore reasonable to select the dominant color over the histogram descriptor for global color representation.The next section deals with the definition of similarity measures for these descriptors.

2.2 Distance Metrics

Given a predefined set of colors c_i, $i = 1, \ldots, N$, a $N \times N$ matrix A with $a_{jk} = 1 - d_{jk}/\max(d_{jk})$ where d_{jk} is the distance between c_j and c_k in their colorspace and two vectors h_1 and h_2 with the percentages of each color c_i, the quadratic histogram distance is given by

$$d_h(h_1, h_2) = (h_1 - h_2)^T A (h_1 - h_2). \tag{2}$$

However, (2) cannot be used if the colors c_i and their number N are different for each image.

Deng et al. proposed a similar quadratic metric in [14], for the dominant color descriptor. If $D_1 = \{(c_i, p_i), i = 1, \ldots, N_1\}$ and $D_2 = \{(b_j, q_j), j = 1, \ldots, N_2\}$ are two dominant color descriptors, then the distance between D_1 and D_2 is defined to be

$$d_q(D_1, D_2) = \sum_{i=1}^{N_1} p_i^2 + \sum_{j=1}^{N_2} q_j^2 - \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} 2a_{ij} p_i q_j, \tag{3}$$

where the similarity coefficient a_{ij} is

$$a_{ij} = \begin{cases} 1 - d_{ij}/d_{\max}, & d_{ij} \le T_d, \\ 0, & d_{ij} > T_d, \end{cases} \tag{4}$$

$d_{ij} = \|c_i - b_j\|$ is the euclidean distance between c_i and b_j, $d_{\max} = \alpha T_d$, α is an arbitrary value and T_d is the maximum distance for two colors to be considered similar.

Another metric that has been proposed for comparing two dominant color descriptors D_1 and D_2 is the Earth Mover's Distance (EMD). In simple terms, the EMD is a dissimilarity measure between two images indicating the amount of "work" required to "move" from the descriptor D_1 of the first image to D_2 of the second. Imagine the colors c_i in the first descriptor as locations in a field with piles of p_i mass of earth each. The colors b_j are also locations, but they consist of holes with capacity q_j earth each. EMD denotes the minimum work required to distribute the piles of earth at c_i to the holes in b_j. Computation of the EMD is based on a solution of the transportation problem and is covered in [15].

A simple experiment was set up in order to evaluate the performance of each distance metric in a realistic image database scenario. A total of 5,022 images from the corel dataset were used, where each of the images belongs to a predefined category C. All images were indexed using the dominant color descriptor with 16 color – percentage pairs. Each image was successively used as a query and a ranked list of results was retrieved. The performance of a distance metric was evaluated based on the semantic correspondence of the results, using the following precision measure:

$$precision_C = \frac{\sum\limits_{n=1}^{N_C} A_n}{\sum\limits_{n=1}^{N_C} 1/n}, \tag{5}$$

where N_C is the number of images in the query image category C, $A_n = 1/n$ if the nth result belongs in C and zero otherwise. Maximum precision is achieved when the first N_C results for a query $I \in C$ belong to C as well. Note that it is too optimistic to expect retrieval of the best results at the semantic level (same category) using only a global color descriptor; however the two distance measures can be compared in this manner.

A graph of the results is given in Fig. 2, where the horizontal axis corresponds to categories C and the vertical axis is the precision (average for all images in a category).The results are clearly in favor of the EMD, that consistently achieved higher precision compared to the quadratic distance.

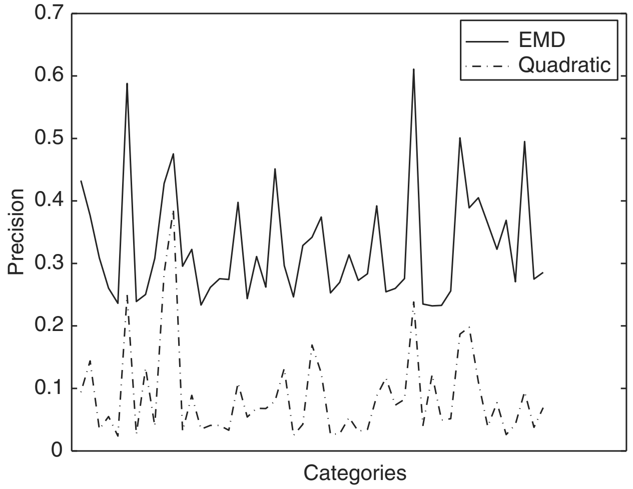


Fig. 2. Results for retrieval based on dominant color descriptors with quadratic distance and EMD. Precision (5) vs. category

3 Indexing and Dimensionality

Assume that the image descriptors and associated distance metrics have been determined and also that the descriptors can be expressed as feature vectors in a k-dimensional space S. In order to construct an image database, a method for solving the "Nearest Neighbor" problem must be selected: Given a set of points P (descriptors) in S (k-dimensional descriptor space) and a query point $q \in S$, find the closest point to q in P.

The simplest solution is to compare q against all images in the database, P. This approach, however, poses strict limitations to the size of the image database due to its high computational cost. To enable implementation of image databases at larger scales, indexing data structures have to be used that solve the Nearest Neighbor problem without visiting the entire database.

One of the most popular data structures proposed is the kd-tree [6, 7]. The idea is to construct a binary tree by successively using elements of the dataset as pivot points to partition the k-dimensional space into hyperrectangles, each containing at most one point. When searching, an initial estimate of the nearest neighbor is provided (by finding the hyperrectangle that contains the query) and then only hyperrectangles and pivot points that are possible to contain a point closer to the query than the initial estimate are visited. Thus, with kd-trees only a subset of the indexed points (i.e., database images) are visited, compared to the exhaustive search where the query is compared against all points in the dataset. Searching in n images with the kd-tree requires at least $\mathcal{O}(\log n)$ visits and $\mathcal{O}(n)$ at the worst case (same as the exhaustive search). The drawback is that the complexity for each point visit is increased (since branching conditions, etc. have to be evaluated).

Still, a very important problem remains, known as "the curse of dimensionality" that affects *kd*-tree efficiency. As the number of dimensions increases, an exponentially increasing number of hyperrectangles (thus points in the dataset) will have to be visited to find the nearest neighbor of a query point.

3.1 Limits of *kd*-Tree Effectiveness

In order to examine the behavior of *kd*-trees with respect to the dimensionality of the space considered, the *kd*-tree data structure and associated algorithms were implemented and a dataset of 10^5 uniformly distributed random points was created for various dimensions. The number of nodes visited per dimension was measured and the results are given in Fig. 3a.

The number of points visited for a number of dimensions close to 30 is practically the complete dataset and the *kd*-tree has no advantage over the exhaustive search. In fact, the upper limit of dimensions that the *kd*-tree is useful is lower, since each visit has additional costs in terms of CPU time. Figure 3b provides the time (in ms) required per dimension for indexing performed on the same dataset using *kd*-trees and exhaustive search on an average personal computer.

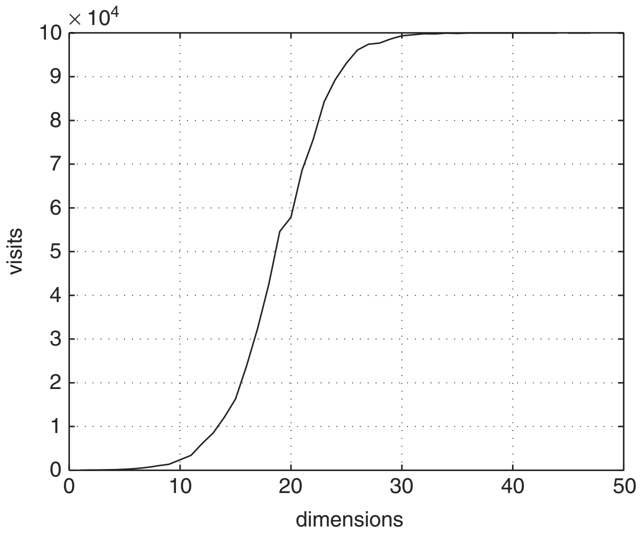
These results indicate that for the test computer and implementation the *kd*-tree keeps an advantage in terms of computational time for eight dimensions or less. Clearly, an optimized implementation of the algorithm would increase this limit, but practically it cannot exceed 15 dimensions. Furthermore, through the experiments conducted, it was observed that the efficiency of the *kd*-tree search is largely dependent on the size of the dataset to be searched. Larger datasets allow for the *kd*-tree to be more efficient in even higher dimensions, compared to exhaustive searching. Also, note that a uniform dataset (used in the empirical evaluation above) is the worst case scenario; *kd*-tree searches are significantly faster within distinctively clustered datasets.

4 Point Configurations

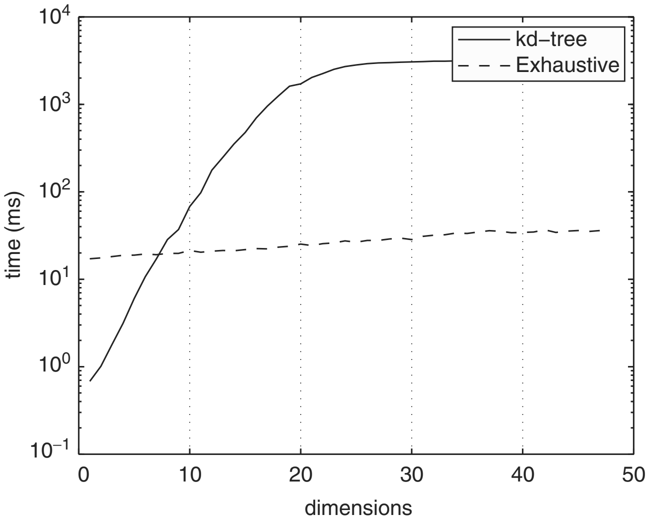
Two major problems can be identified with the use of *kd*-trees and similar indexing structures in image databases:

1. The number of dimensions used by image descriptors is prohibitive for efficient indexing.
2. Descriptors do not always define points in a k-dimensional space and thus *kd*-trees cannot be employed for indexing. The dominant color presented in Sect. 2.1 is an example of such a descriptor.

Low-dimensional embeddings of descriptors such as those produced by PCA is a possible solution to the high dimensionality problem. Again, however, this approach is only applicable to k-dimensional points. Both the above issues



(a)



(b)

Fig. 3. (**a**) Number of visits for nearest neighbor search in the *kd*-tree for a dataset of 10^5 uniformly distributed random points. (**b**) Time for nearest neighbor search in ms for the *kd*-tree and exhaustive search for various dimensions on an average personal computer can be tackled if a low-dimensional point configuration is produced from the original descriptors.

This problem can be formulated as follows: "Given the observed distances d_{ij} between any two objects (images in the database), produce a configuration of points in the k-dimensional space, such that the new (euclidean) distances d'_{ij} are as close as possible to the original d_{ij} for all the points". A measure of effectiveness for a solution is Kruskal's *stress* function (6) [16].

$$\text{stress} = \left[\frac{\sum_{i,j} (d'_{ij} - d_{ij})^2}{\sum_{i,j} d_{ij}^2} \right]^{1/2} . \tag{6}$$

In the case of the dominant color descriptor, the original distances are given by the EMD, while the derived points are in an k-dimensional space with euclidean distances d'_{ij}.

4.1 MDS and FastMap

Two techniques that produce point configurations were evaluated, namely metric Multidimensional Scaling (MDS) [17] and the FastMap algorithm [18]. Metric MDS is a technique that receives as input the observed dissimilarities d_{ij} between objects and produces a configuration P' of points in the k-dimensional space through an iterative optimization process. Roughly, each object is assigned an k-d point (e.g., randomly) and then every point is examined by computing its distance from all the $N - 1$ other points. The point is moved so that it optimizes the stress function. The complexity of performing MDS is $\mathcal{O}(N^2)$, where N is the number of objects.

FastMap, on the other hand, solves the same problem by starting from $1 - d$ and recursively determining the coordinates of the N objects on a new axis, up to k–d. Computationally, this process is much more efficient than MDS, with its complexity being $\mathcal{O}(kN)$, where k is the number of dimensions of the target configuration. MDS and FastMap were compared with respect to the stress function (6) using the corel dataset. It can be seen in Fig. 4 that MDS achieved better results. But the main strength of FastMap is the $\mathcal{O}(1)$ complexity required for the insertion of a new point, contrary to $\mathcal{O}(N)$ required by MDS. In practice, answering a query with MDS (thus embedding the query object in the k-d space and finding its nearest neighbors) requires as much time as the exhaustive search at best. MDS is therefore not suitable for retrieval applications.

4.2 Browsing

An important aspect of any image retrieval system is the visualization and browsing interface. In the simplest case, thumbnails of result images can be provided in a list for the user to browse. In the case of images, however, a

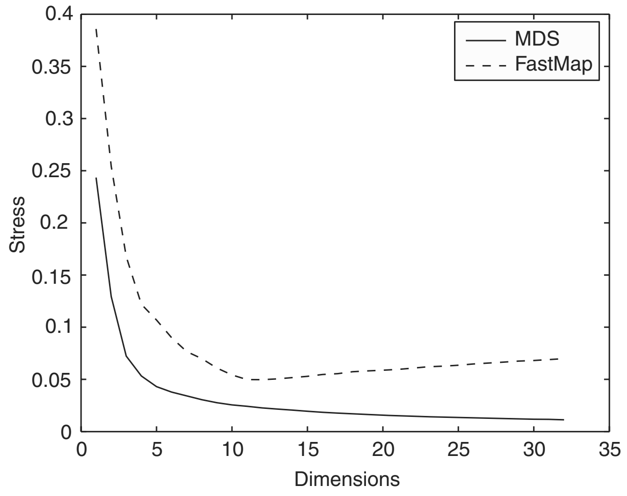


Fig. 4. MDS vs. FastMap stress performance for various dimensions. Experiment for 500 images

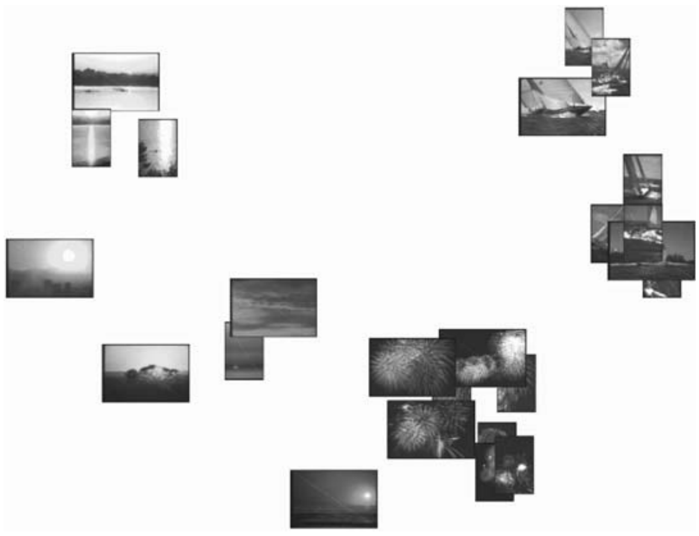
more efficient visualization and browsing interface would provide a grouping of similar results. Using MDS or FastMap to produce a configuration of points at two or three dimensions from the result dataset allows its visualization, as illustrated in Fig. 5. The user is able to view the entire result dataset, zoom to specific regions and select images. The examples in the figure are derived using FastMap. If the number of results is relatively small, MDS can be employed as well.

5 Efficiency of FastMap Configurations

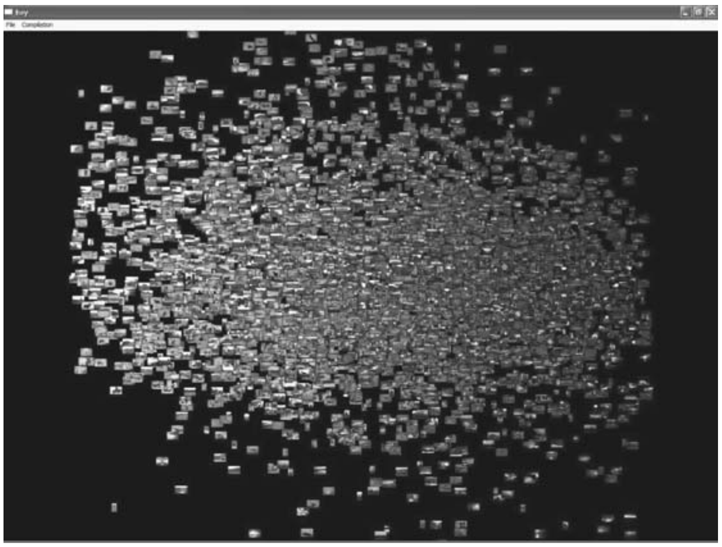
Given a dominant color descriptor D_i for each image I_i in the database, the EMD measures d_{ij} between I_i and I_j are computed for all i, j. Subsequently, FastMap is applied to create a configuration P_k of k-dimensional points, one for each image. This allows the efficient use of kd-trees for indexing. The questions that naturally arise have to do (1) with the quality of the retrieval results and (2) how these results are affected by the choice of k.

In order to evaluate the performance of FastMap configurations for image retrieval, the ranking results obtained from EMD-based queries (Sect. 2) were used as ground truth with the same collection of 5,022 images from the corel dataset. For the same query, the difference in rank between the EMD and mapped results was used, as follows.

Initially, point configurations of the entire dataset were constructed for dimensions $k = 1, \ldots, 32$. For each k, each image I_j, $j = 1, \ldots, 5022$ was submitted as a query, and returned a ranking $r_{jk} = \left[I_1^{jk} \ldots I_{5022}^{jk} \right]$ of all images



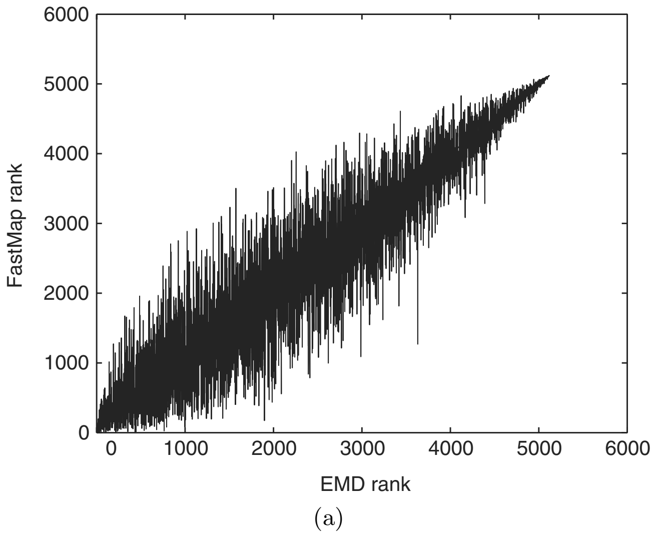
(a)



(b)

Fig. 5. (**a**) Result of applying FastMap for two dimensions on a set of images from three categories of the corel dataset. (**b**) Navigation on a larger dataset. Users can zoom in specific areas, modify all distances by a factor and select specific images

extracted from search at the k-d space. This ranking was compared with the corresponding EMD rank r_j^{EMD} for the same image, yielding a set of pairs $L_{jk} = \{(p_{j1}, p_{j1}^k), \ldots, (p_{j5022}, p_{j5022}^k)\}$. These pairs indicate the rank in EMD and FastMap for the same result image. For example, a query image from the



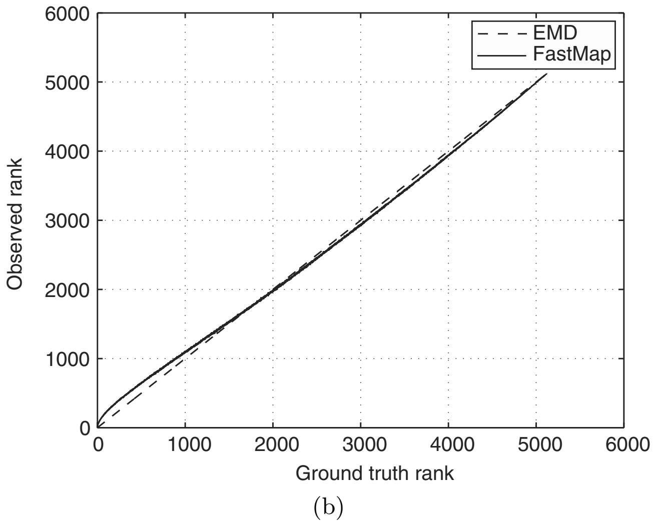


Fig. 6. Example rank results for $k = 6$ dimensions. The EMD rank (ground truth) is the $y = x$ line. (**a**) Results for a random image. (**b**) The average for all images

"action sailing" category gives $(2, 4)$ which reads "the 2nd result of EMD was ranked 4th using FastMap". Figure 6 shows an example of the rank results for a random image and the average for all images at $k = 6$ dimensions.

It can be seen that even though the results of EMD ranking and those of FastMap configurations are not identical, they are averagely very close to each other at six dimensions. The similarity is practically not improved if more than 10 dimensions are used. Hence this experiment is an indication that *by using FastMap configurations for indexing, the retrieval does not deviate significantly from the EMD results.* It is therefore highly unlikely to find images ranked in the first results by EMD at the lower ranks (i.e., least similar results) of FastMap based retrieval.

To quantify this observation, another experiment was conducted. Images were selected at random from all categories (resulting in 100 images in total) and the retrieval results for these images were observed for EMD and FastMap, as in the previous. Then, the distribution (pdf) of the random variable $\mathbf{d}_{I_q,k} = p_{jr} - p^k_{jr}$ was calculated for $k = 1, \ldots, 32$ dimensions, with p_{jr} and p^k_{jr} as defined above. This random variable quantifies the difference in rank of a result for a query image I_q at k dimensions, compared to EMD. The distribution (histogram) of $\mathbf{d}_{I_q,k}$ for a random query image I_q at $k = 6$ dimensions is shown in Fig. 7.

In order to remove the dependence of the estimated distribution of $\mathbf{d}_{I_q,k}$ on the query image I_q, a number of Monte Carlo experiments were performed, yielding the ensemble average over 100 randomly selected query images I_q. The resulting Monte Carlo mean $\mathbf{d}_k = E_I\{\mathbf{d}_{I_q,k}\}$ is depicted in in Figs. 8a–c for $k = 3$, 6 and 12 FastMap dimensions. The Monte Carlo variance for all ordinates of the estimate of $\mathbf{d}_{I_q,k}$ was pretty low for fixed k (e.g., Fig. 8d for $k = 6$), thus ensuring that the adopted distribution estimators are meaningful and representative.

These distributions allow the computation of the probability $P(-d < \mathbf{d}_k \leq d)$ that the difference of results between EMD and FastMap will be less than d. Figure 9 displays how these probabilities are affected by the number of dimensions k for various d. It can be deduced that the FastMap results are, in fact, expected to deviate from the original EMD ranking (for $d = 25$, P

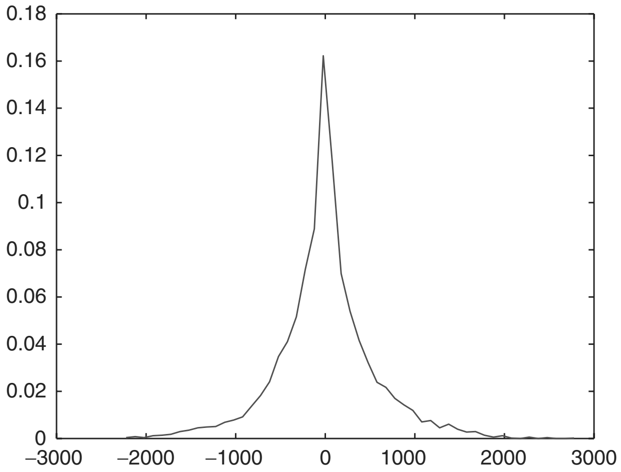


Fig. 7. The distribution of $\mathbf{d}_{I_q,k}$ for a random image at six dimensions

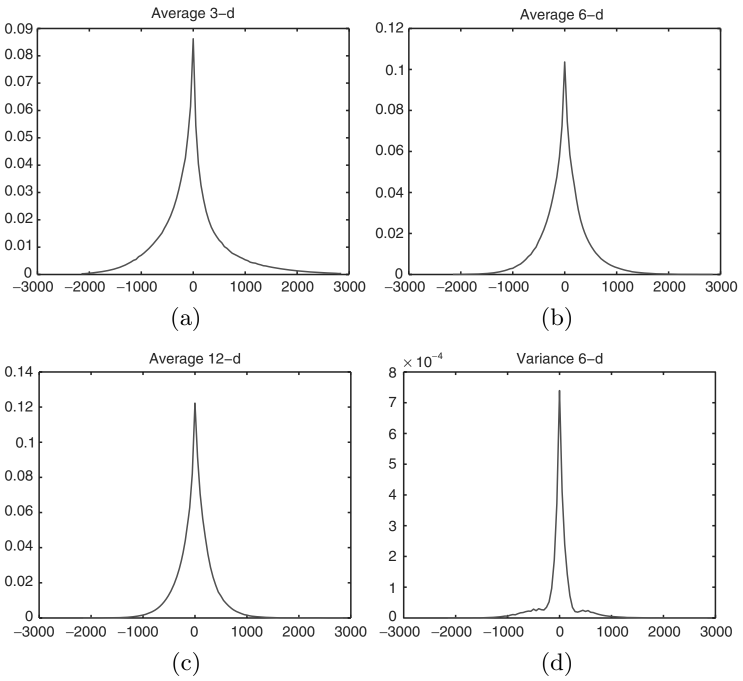


Fig. 8. (**a**), (**b**), (**c**) The distribution of $\mathbf{d}_k$ for $k = 3, 6, 12$ dimensions, respectively. (**d**) The variance of $\mathbf{d}_{I_q,k}$ for $k = 6$ across different query images. Notice that peak variance is 7.4×10^{-4}

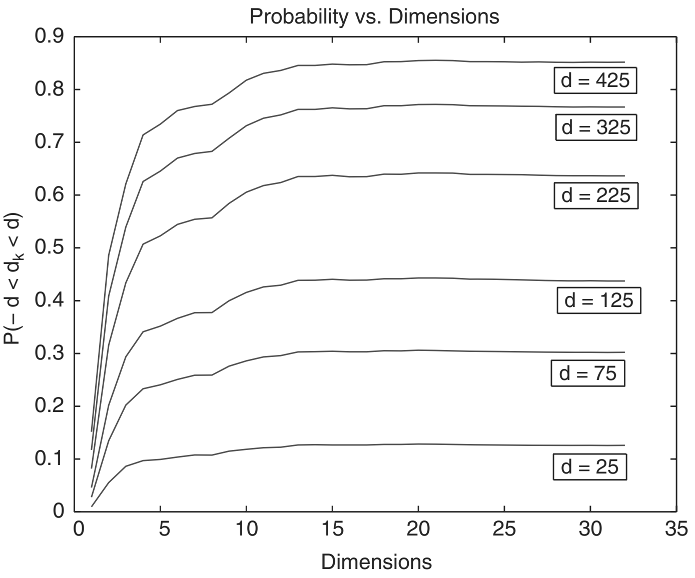


Fig. 9. The probability $P(-d < \mathbf{d}_k \leq d)$ vs. dimension k for various d

is low), however it is highly unlikely for the best EMD results to be ranked lower than 400 using FastMap.

Whether these results can support the practical use of FastMap in image database retrieval largely depends on the application. More specifically, it depends on the browsing and retrieval interface used and the number of results that are originally visualized. There is a tradeoff between the number of results returned (larger result datasets possibly require more browsing effort from the user) and query response times.

Assume the design requirement: "For a query image I_q, the first r results of the corresponding EMD results should be returned". One must find the number of results r_{fk} that must be returned to the user if the dataset is mapped to k dimensions using FastMap. These r_{fk} results should contain the first r results of EMD with a high probability. Again a set of Monte Carlo experiments are performed that provide the ensemble average over 100 randomly selected query images I_q. The resulting Monte Carlo mean $p(r, r_{fk})$ indicates the estimated percentage of the first r results of EMD present in the first r_{fk} of FastMap at k dimensions. Its value is depicted for various r_{fk} in Fig. 10. For example, if $k = 8$ and $r_{f8} = 100$ then it is estimated that 92% of the first $r = 10$ EMD results will appear. In other words, the probability for any of the first 10 EMD results will be found if 100 results are returned using FastMap at eight dimensions, is estimated to be 0.92.

6 Conclusions

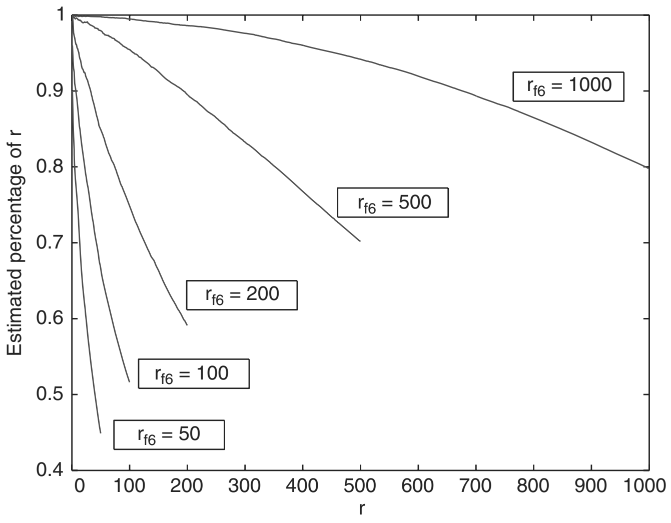
From the results of the previous sections several useful conclusions can be drawn.

For the global color representation and specifically the dominant color descriptor, the Earth Mover's Distance appeared to be the most accurate distance metric in the conducted experiments.

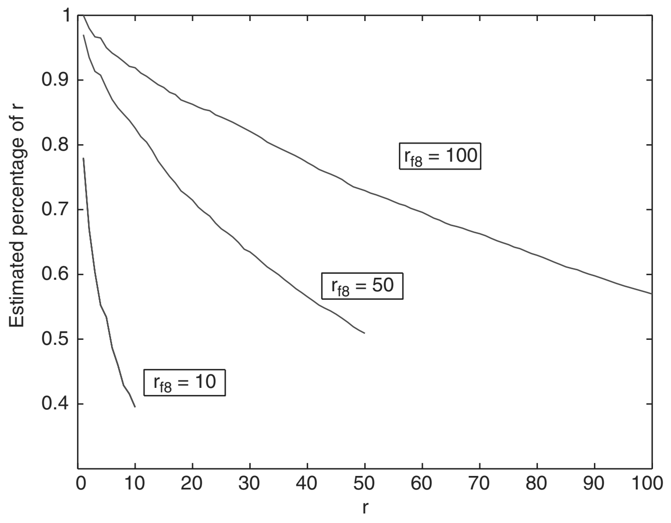
For the indexing problem, *kd*-trees were more effective than exhaustive search, but their practical application is limited to a relatively low number of dimensions. Moreover, *kd*-tree structures index points in a k-dimensional space that dominant color descriptors do not provide.

For visualization and navigation purposes, the MDS approach proved more precise than FastMap in terms of the stress measure. However its practical use is limited due its high computational cost. FastMap is better suited to large-scale image databases.

FastMap was also used to provide low dimensional point configurations for efficient indexing. A series of experiments illustrated how the EMD ranking results are affected by the number of dimensions used in the FastMap configurations. Using a configuration with relatively low dimensionality (e.g., 6–8 dimensions) in conjunction with a browsing interface that allows the concurrent visualization of a large result set seems to be a reasonable compromise between retrieval accuracy and fast response times.



(a)



(b)

Fig. 10. (**a**) $p(r, r_{f6})$ for various result numbers and $k = 6$. (**b**) More detailed view for small number of results and $k = 8$

These evaluations outline the problems that must be tackled and the decisions that need to be made for the various stages of a content based image retrieval system design. For the example evaluations above, an image database system implementation can be proposed. The operations related to the construction of the database as well as query answering are outlined in Figs. 11a,b, respectively.

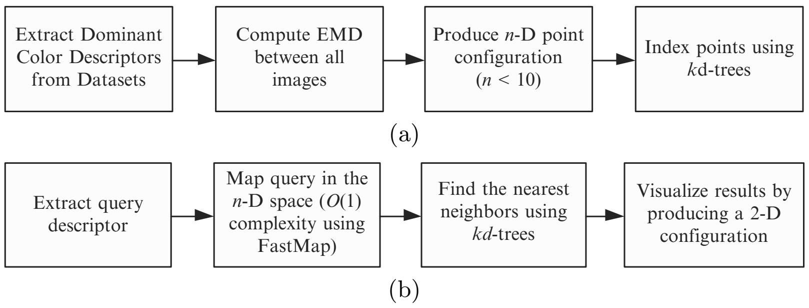

Extract Dominant Color Descriptors from Datasets
Compute EMD between all images
Produce n-D point configuration (n < 10)
Index points using kd-trees
(a)
Extract query descriptor
Map query in the n-D space (O(1) complexity using FastMap)
Find the nearest neighbors using kd-trees
Visualize results by producing a 2-D configuration
(b)

Fig. 11. (**a**) Steps for constructing an image database. (**b**) The process of query answering

Acknowledgment

Christos Diou is supported by the Greek State Scholarships Foundation.

References

1. T. Sikora. The mpeg-7 visual standard for content description – an overview. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(6):696–702, 2001.
2. B. S. Manjunath, P. Salembier, and T. Sikora. *Introduction to MPEG-7: Multimedia Content Description Interface*. Wiley, New York, 2002.
3. A. Dorado and E. Izquierdo. Semantic labeling of images combining color, texture and keywords. In *IEEE International Conference on Image Processing, ICIP*, volume 3, pages 9–12, Barcelona, Spain, September 2003.
4. A. Yavlinsky, E. Schofield, and S. Ruger. Automated image annotation using global features and robust nonparametric density estimation. In *Proceedings of the International Conference on Image and Video Retrieval (CIVR'05)*, 2005.
5. J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV 2005)*, 2005.
6. J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of ACM*, 18(9):509–517, 1975.
7. A. W. Moore. An introductory tutorial on kd-trees. Technical Report 209, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Computer Laboratory, University of Cambridge, 1991.
8. A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *The VLDB Journal*, 518–529, 1999.
9. Jonathan Goldstein and Raghu Ramakrishnan. Contrast plots and p-sphere trees: Space vs. time in nearest neighbour searches. In *VLDB 2000, Proceedings of 26th International Conference on Very Large Data Bases, September 10–14, 2000, Cairo, Egypt*, pages 429–440, 2000.
10. D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

11. J. Yang, J. Yang, D. Zhang, and J. Lu. Feature fusion: parallel strategy vs. serial strategy. *Pattern Recognition*, 36(6):1369–1381, 2003.
12. A. Kushki, P. Androutsos, K. N. Plataniotis, and A. N. Venetsanopoulos. Retrieval of images from artistic repositories using a decision fusion framework. *IEEE Transactions on Image Processing*, 13(3):277–292, 2004.
13. M. Gervautz and W. Purgathofer. A simple method for color quantization: Octree quantization. In *New Trends in Computer Graphics*. Springer, Berlin Heidelberg New York, 1988.
14. Y. Deng, B. S. Manjunath, C. Kenney, M. S. Moore, and H. Shin. An efficient color representation for image retrieval. *IEEE Transactions on Image Processing*, 10(1):140–147, January 2001.
15. Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, November 2000.
16. J. B. Kruskal. Multi-dimensional scaling by optimizing goodness-of-fit to a nonmetric hypothesis. *Psychometrica*, 29:1–27, 1964.
17. Y. Rubner, C. Tomasi, and L. J. Guibas. Adaptive color-image embeddings for database navigation. In *Proceedings of the 1998 IEEE Asian Conference on Computer Vision*, Hong Kong, 1998.
18. C. Faloutsos and K.-I. Lin. FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, pages 163–174, San Jose, CA, 1995.