A Flexible Scheme for Representation, Matching and Retrieval of Images

C. V. Jawahar, P. J. Narayanan, Subrata Rakshit
Centre for Artificial Intelligence and Robotics
Raj Bhavan Circle, High Grounds
Bangalore - 560 001, INDIA
{jawahar,pjn,subrata}@cair.res.in

Abstract

Image databases index images in them using features extracted from the image. The indexing scheme is decided apriori and is optimized for a specific querying criterion. This is not suitable for a generic database of images which may be queried by multiple users based on different criteria. In this paper, we present a flexible scheme which adapts itself to the user's preferences. Though the method uses a conservative set of features during indexing that includes a large number and type of fundamental features, the query processing time does not increase due to this redundancy. A boot-strapping mechanism allows the user to build up a "desired class" from a few samples. The feature selection computation scales linearly with the size of the desired class, rather than that of the entire database. This feature makes our algorithm viable for very large databases. We present an implementation of our scheme and some results from it.

1 Introduction

Representing images for efficient matching and retrieval is a central issue in image databases and content-based image retrieval [1, 2, 3]. The representation should abstract the relevant information present in each image for an efficient, content-based retrieval at a later time. The abstractions used conventionally are based on image features, or parameters computed from the whole or the parts of the image. The content-based retrieval mechanisms also abstract the example image using the same features so that they can be matched for efficient retrieval.

Reduction in the size of the representation is only one reason why the images are abstracted using a few features. A properly designed representation scheme can support a wide range of queries, especially those based on the "content" of the image, very effectively and efficiently. Such a query will be difficult, if not impossible, to perform using the full images themselves, other than by computing a similar representation at retrieval time. Thus, the selection of the features used in the representation of the images is a critical design decision that determines what types of retrievals can be supported effectively by the collection of images. Selection of the most appropriate features to index a database of images is an area that has received a lot of attention [4, 5, 6]. Most work has focussed on finding an optimal set of features for specific types of queries for which the database was designed.

The problem of reusability of non-specialized collections of images for a variety of purposes is an important issue today. With the growth of the Internet, various individuals and organizations have made available vast collections of images, often without any specific applications in mind. Designing representations for such generic collections of images to support a wide range of queries based on content at a later date, is the aim of this work. While generic image collections are the primary target, even specialised collections can benefit from more flexible indexing schemes. Focussed representations are advantageous when used for the purposes for which they were designed. For example, the contours and intersections of road-like features may play a dominant role in the representation of a database of aerial images of cities meant for town planners and civic authorities. However, the same collection of images might be used for queries with a different focus. For example, the same database may be of interest to an architect studying the nuances of the shapes of buildings or a sociologist looking for growth patterns in cities. Thus it is desirable that even a specialized collection of images be able to support queries based on characteristics that were not catered for explicitly during the database construction.

Formulating a comprehensive set of features is the first design issue in designing a flexible image retrieval system. However, using a large number of features leads to a large database. This can increase the query processing time as the matching gets more complicated. It can also degrade the query results as all features, including those not relevant to a query, get to influence the matching process. This leads to the second design issue: the design of a feature selection scheme that selects features suitable to a given query. In addition, one needs to pick a metric for matching images based on the selected subset of features. The final design issue has to do with the user interface. For a generic database meant for non-specialist users, queries cannot be framed using any special set of keywords or attributes. Querying by example seems to be the most intuitive method. Since this means that a user must have a few examples in order to query the database for more examples, an efficient boot-strapping method has to be devised. This constitutes the third design issue.

In this paper, we present a scheme that addresses the second and third design issues, namely, a scheme for adaptive feature selection and an algorithm for boot strapping a desired class of images. We have addressed the first issue (definition of a large feature set) only to the extent of illustrating the principle of using large, redundant feature sets. Our scheme uses an adaptive procedure to tune the retrieval using a set of example images given by the user. The goal is to get a number of images in the collection of images that are most relevant to the example images. Query by example is an effective way of retrieving relevant images out of a diverse collection of images [7, 8]. Our method uses a set of positive exemplars that define images that are similar to what the user is looking for and a set of negative exemplars that define what he or she is *not* interested in. The system learns the features of importance from the exemplars and tunes the retrieval using this information to come up with a set of images from the collection similar to what the user seeks. The user has then the ability to accept or reject the images retrieved by the system, thereby augmenting the sets of positive or negative exemplars. This process can be repeated till the required results are achieved. Thus, the process can boot-strap from the intermediate results to tune itself to the user's requirements and provide the user with as many of the images as possible from the collection.

We describe the basic philosophy in Section 2. Implementation details and results are described in Sections 3 and 4. Relevance of the approach is discussed in Section 5. Concluding remarks and comments on future directions to follow can be found in Section 6.

2 Approach

Our scheme for efficient representation and retrieval of images is shown in Figure 1. A large number of features are used in the representation so as to cater to various types of queries that may be addressed in the future. We make no recommendations on the specific features to be used. The features should be sufficiently diverse and fundamental that they can be used to answer a wide range of content based queries. Thus colour, textures, principal components, etc., are all candidates for a comprehensive representation. The use of local features, in addition to global ones, is essential for answering queries that involve spatial relationships, *i.e.,* a search for all images with a blue sky above white mountains. In order to pose a query to the system, the user must provide a set of positive and negative exemplars. These sets can be small to begin with but are built up iteratively as explained below.

Our scheme uses a large number, M, of features for representing the images; the entire collection is indexed using these M features. A query by content scheme will compute these M features for the example images also. If the image collection is queried based on all M features, the query time would be proportional to M, which is not desirable as M can be large. Moreover, many of the features used in the comprehensive representation may not be relevant to a specific query and their inclusion in the matching process is likely to degrade the search results. It is thus necessary to select a subset of the features of cardinality m, where $m \ll M$, for the matching and selection process. These features, identified using both the positive and negative exemplar images, are most relevant to the user's query. This feature reduction step is crucial because (i) it improves the search results by eliminating extraneous features (ii) it makes the query time independent of M, and (iii) it provides a parameter for trading off performance versus time.

The images of the database that best match the positive exemplar images based only on the selected m features are retrieved from the database for the user's perusal. The user can classify them as positive, negative, or indifferent. Positive images are added to the positive exemplar set, negative images are added to the negative exemplar set and images marked indifferent are not added to either set. A fresh set of m features is now computed based on the enhanced exemplars and the entire exercise can be repeated.

Tuning using positive and negative examples is performed as follows. The values of each of the M features are computed in each exemplar image. The means and variances for each feature is then computed separate-

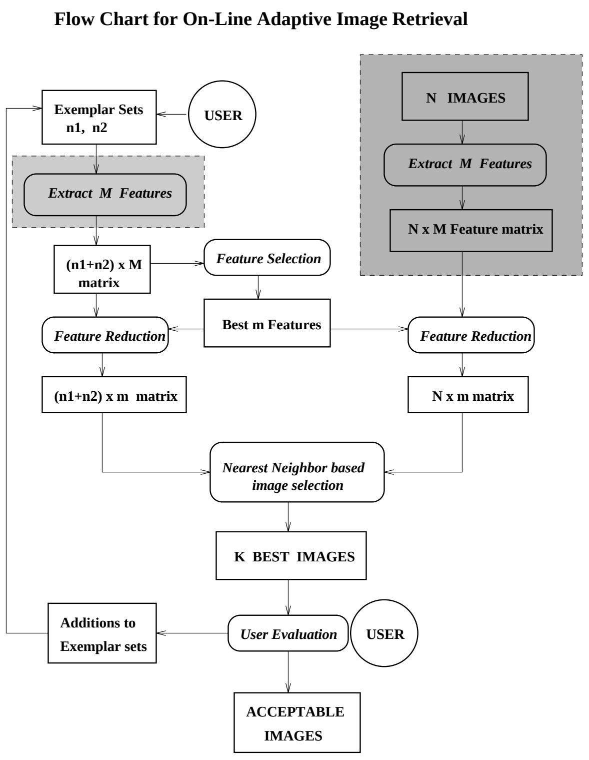

Figure 1: Overview of the proposed algorithm for flexible image indexing and retrieval. The shaded region indicates the one time off-line computation required to set up the representation of the database.

ly within the set of positive exemplars and the set of negative exemplars. Let μ_i^+ and μ_i^- be the means of the feature i in the set of positive and negative exemplars respectively and let σ_i^+ and σ_i^- be its variances in the respective sets. Consider the *feature relevance measure* for feature i defined as follows.

$$f_i = \left|\frac{\mu_i^+ - \mu_i^-}{\sigma_i^+ + \sigma_i^-}\right| \qquad (1)$$

A particular feature is useful for a particular demarcation problem, only if it can lead to two well separated and compact clusters in the feature space corresponding to the positive and negative exemplar sets. The measure f_i has been formulated so as to measure this separation. A high f_i will mean that the feature i effectively discriminates between the two classes and is a good feature to represent the image as far as this particular query is concerned. One can either keep m features with high f_i where m is decided a priori, or can keep all features with f_i greater than an acceptable threshold. The reduced set of features serve as a modified representation of the images in the database for the purposes of the specific query being processed.

We have implemented our algorithm to pick a user-specified number of features, m, that can be specified for each query.

After identifying the reduced set of features, the images from the database are matched against the images in the positive and negative exemplar collection using the m-dimensional features. For an image k from the database, the best match among the positive exemplars and the negative exemplars is found by maximizing the normalised dot product, $(\vec{p}.\vec{q})/(||\vec{p}||.||\vec{q}||)$, of the reduced, m-dimensional feature vectors. Let t_k^+ and t_k^- be the normalised dot products corresponding to the best match among the positive and negative exemplars respectively. Consider the *image match measure* s_k defined as follows.

$$s_k = \frac{t_k^+(1 - t_k^-) - t_k^-(1 - t_k^+)}{2 - t_k^+ - t_k^-} \qquad (2)$$

It is clear $-1 \leq s_k \leq 1$. Moreover, if the exemplars are chosen from the database itself, they will always be correctly classified as ± 1 as either t_k^+ or t_k^- will be 1. This ensures that during the bootstrapping process, each iteration produces a match set that is *consistent* with the exemplar sets. A high s_k implies that the nearest image from the positive class is much closer than the nearest from the negative class. Note that when $t_k^+ = t_k^-$ the match measure $s_k = 0$, indicating no preference. The matching succeeds on all images of the database with a match measure greater than an acceptability threshold. They are retrieved for presenting to the user.

Each iteration of the search process follows the procedure given below.

1. Calculate the M feature values for each image in the sets of positive and negative exemplars.

2. For each feature i, compute its means μ_i^+ and μ_i^- and the variances σ_i^+ and σ_i^- in the sets of positive and negative exemplars.

3. Compute the feature relevance measure f_i given by Equation 1 for each feature.

4. Rank the features in the descending order of its relevance measure. Use the top m features for the reduced representation of the images for the actual search through the database.

5. For each image k in the database, compute the closest distances to a member of the positive and negative classes t_k^+ and t_k^- and the match measure s_k given by Equation 2.

6. Best K images with highest s_k pass the test and are selected for further processing, where K is a number specified by the user.

The above procedure can be iterated till either no new image is added to the positive class or till a required number of images are obtained, according to the user's preference. The above search procedure learns the user's preferences and adapts the features for the correct retrieval.

3 Implementation

Our approach has been specifically designed to ensure that the addition of extra features to the database will only refine the selection process without significantly increasing the computation time. In the following paragraphs we give the details of the features used in our implementation. It may be emphasized again that *our choice of features is illustrative, not prescriptive.* We do not make any claims of optimality or universality regarding our choice of number or types of features.

Many types and kinds of features are used to index images of a collection [1]. We decided to employ texture, colour and brightness features to represent the image in our implementation. We also realise that the structural details of the images are better represented with local features than global features. Local image characteristics are computed in sub-images, in 32×32 non-overlapping windows, while global features are computed across the entire image. The images we use are of size 128×128 or are extrapolated up that size. Features are computed based on histograms, co-occurrence matrices and convolution with a set of kernels based on principal components (PC's).

The PC's were computed based on a large number of 32×32 extracts from various black/white images considered as typical samples from the image collection. These mutually orthogonal kernels essentially provide a texture based characterisation of an image that is optimised for an image set. This approach has been used successfully to characterise and register images [9]. Of the 1024 32×32 PC's, we use only the first 20 PC's. Each image is subdivided into 16 small sub-images of equal size and each of these sub-images is convolved with the 20 kernels, generating $16 \times 20 = 320$ features. In addition, for each PC, the average of its feature values across the 16 sub-images is also computed. This produces another 20 features. Thus a total of 340 features were computed from the principal components.

Mean and variance of the three colours i.e., red, green and blue, are computed in the 16 sub-images and also for the entire image. For a finer description of the colours, colour histograms are used. Colour histograms are very popular for representing the colour content of images. We employ a simple colour histogram with the number of colours quantized into 18 (3 shades of red, 3 shades of green, 2 shades of blue). This coarse colour histogram is computed for local sub-images as well as the whole images. These colour histograms are used as features. Thus a total of 24 parameters (3 means, 3 variances, 18 histogram values) were computed for 16 sub-images and the full image leading to 408 color based features.

We also use co-occurrence matrices to give another type of texture characterisation that is sensitive to color. An entry p_{ij} in the co-occurrence matrix denotes the probability of neighbouring pixel pairs in the image to have colour indices i and j respectively. Since a 32×32 sub-image would not be able to provide enough entries to fill the 18×18 co-occurrence matrix, these features are calculated only at the global level. The entries in the colour co-occurrence matrices with 18 quantized colour levels define $18 \times 18 = 324$ features. In addition, we use four additional measures, namely Angular Second Moment, Contrast, Inverse Difference Moment and Entropy, computed from the co-occurrence matrix using a 512-level color quantisation. Thus, the co-occurrence matrices in all provide an additional 328 features. The total number of features used in our implementation thus is 1076. Each image in the collection was represented using these 1076 values in a preprocessing step.

4 Results

We used a database of 10000 images[1] for verification of the methodology. This is a vast collection of images without any common theme and has not been put together for any specialised application.

4.1 Two sample queries

In the first experiment, RoseGarden, we tried to retrieve reddish flowers in a greenish garden. This query was clearly dependent on color features and this was reflected in the features selected in the reduced representation. An initial set of 11 acceptable and 11 rejectable images were given to the system and the best 50 images were displayed. In the first iteration our algorithm produced 31 acceptable images and converged after five iterations. That is, no new image of the top 50 was added to the positive or negative exemplars after five iterations. In each iteration, most of the images retrieved by the system were acceptable, a few were rejected and the rest were ignored. After

[1]Available at `http://WWW-DB.Stanford.EDU/IMAGE`

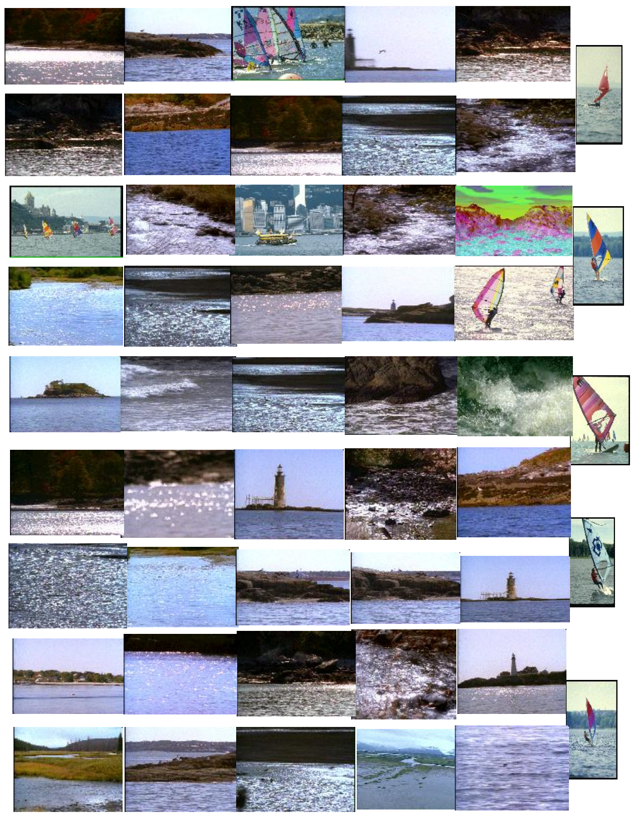

Figure 2: Best 50 results of a query for sea/river shores

convergence, the retrieval process yielded 50 acceptable images among the top 50 and more than 75 out of the top 100 images selected.

In the second experiment, SeaShore, we searched for sea/river shores. In this case, the texture of the rough water provided features in the form of some principal components in addition to color features for water/land. These were reflected prominently in the reduced representation. The query started with 7 acceptable exemplars and 9 rejectable ones. The best 50 images selected after 4 iterations are shown in Figure 2. The next best 50 images contained some, but not all, acceptable images.

4.2 Analysis of actual performance

We use the performance of our system on the RoseGarden and SeaShore problems to check on the validity of the various measures defined by us. To begin with, we address the question of feature selection. It was an implicit assumption that there would be some features that would separate the two exemplar sets into two clusters. Due to the high dimensionality of our feature space, we cannot directly visualise the clusters. However, we can see if there are features with higher difference of means between the exemplar sets than others. This is shown in Figure 3. When ranked by the ensuing feature relevance measure f_i (Figure 3), it becomes obvious that there are a well defined set of features relevant to the problem. Thus the feature selection/reduction step significantly reduces subsequent computations while improving the signal to noise ratio.

We selected the RoseGarden and SeaShore problems to illustrate the ability of our system to adapt on a per query basis. This adaptation is highlighted in Figure 4. The plot shows the value of f_i computed for each of the 1076 features for the two queries. As can be seen, two very distinct sets of features have been selected for the two queries.

One area of concern for any iterative system is the issue of convergence. For the present system, the rate of convergence depends on the consistency with which a user adds samples to the positive and negative exemplar sets. It is clearly impossible to guarantee convergence in any mathematical way for such a system. However, our experiments show that the achievable rates of convergence are quite good. This is illustrated in Figure 5 which shows how many acceptable images were displayed among the top 50 at each iteration. At each iteration, the user was shown the top 50 picks, which necessarily included the images already put into the positive exemplar set. Thus the number of new images viewed by the user for each iteration was 50 - (number of acceptable images in the previous iteration). From Figure 5 it can be seen that the user had to manually screen 65 new images for the RoseGarden query and 101 new images for the SeaShore query before getting the 40 new acceptable images for each class. The size of the full data set was 10,000 images.

5 Discussions

5.1 Computational complexity

For image retrieval algorithms, there are two computational complexity issues: (i) the off-line, initial computation in setting up the database and (ii) the online, recurring computation to be performed for every query. Our approach has been guided by the philosophy that the computation per query must be minimised since this is the deciding factor for the "response time." Retrieval systems that are adaptive rather than fixed are necessarily more complex. The advantage of our method is that this extra cost has been projected almost entirely into the initial, off-line process of creating the database. Let N denote the total number of images to be represented and let M denote the total number of features used in the rep-

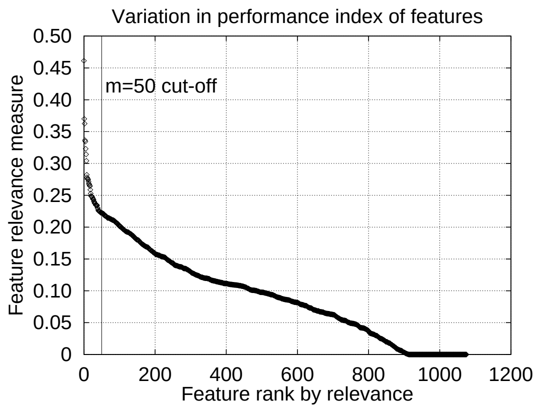

Figure 3: Relevance of various features in clustering the exemplars for the RoseGarden problem. Note the existence of a well defined set of 'good' features.

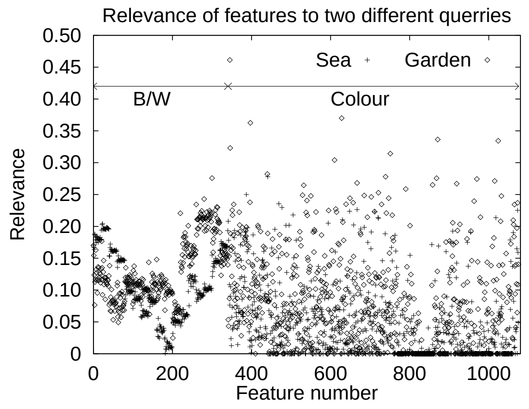

Figure 4: Adaptation to various queries by feature selection. For the RoseGarden the best features are colour-based and for SeaShore PCs.

resentation. Clearly, the cost of creating the database is $O(NM)$.

The cost per query is as follows. Let n be the number of exemplars provided and m be the number of features to be selected. Then the cost of doing the feature selection is $O(nM)$, as the feature relevance measure (f_i) depends only on the exemplars. Once m features are selected, the match/selection process requires comparisons between N feature vectors of m components each. This cost scales as $O(Nm)$. Given that $n \ll N$ and $m \ll M$, the overall cost per query is $O(nM + Nm)$, which is much less than $O(NM)$. This low cost per query is an important feature of our system. Since we use feature *selection*, there is no re-computation of features involved in the adaptation process. Adaptive techniques which use optimal feature formulation based on existing features (SVD, PCA etc) do need to compute fresh feature values in the adaptation process. As a result, the cost of adaptation is too high to be incurred for every query.

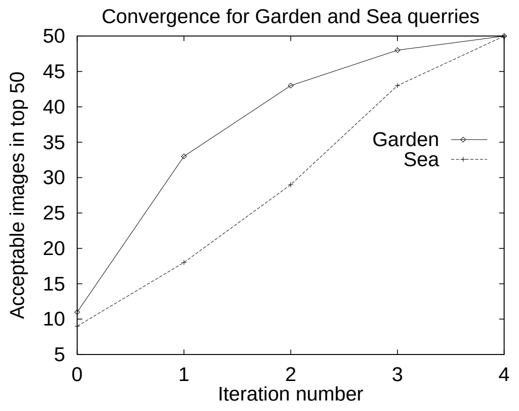

Figure 5: Rate of convergence for RoseGarden and SeaShore queries.

5.2 Flexibility

An important feature of any practical retrieval system meant for generic use is its flexibility *vis a vis* database updates, user interaction, computation resources and adaptability to different search criteria.

Database updates: In our approach, the features are extracted for each image in isolation. Moreover, each feature is defined independently. Thus an additional image or an additional feature can be added without any recomputation.

User interaction: The user interacts solely by way of selecting positive and negative exemplars. This makes the interface very intuitive and obviates the need for learning any special terminology or syntax [7, 8, 10].

Computational resources: An important feature of our scheme is that the user can decide how "costly" to make each query using m. This means that, depending on the computation resources available and the size of the database, the user can pick an accuracy versus time trade-off.

Adaptability: As mentioned before, the same set of features are not relevant for all types of queries and most implementations of image retrieval have one form of adaptation/optimisation or the other. However, one should distinguish between optimising features for a database as opposed to a query. The former assumes that there is an underlying similarity between the images and a common basis for all queries. In contrast, a query based optimisation like our scheme can deal with collections of disparate images and diverse types of queries. Here the optimisation is done independently for each query. This gives the system *run-time* adaptability as opposed to a *compile-time* adaptability offered by the schemes that select optimal features for representing a collection of images.

5.3 Selection criterion

In the present implementation, we made the selection process based on a nearest neighbor approach. An alternate approach would have been to use the mean distance to positive and negative exemplars (or, equivalently, distance to cluster means of positive and negative exemplars). The use of a mean to represent a distribution is justified only if there is a central tendency, i.e., if the data is clustered. Though the individual features are selected based on their relative ability to cluster the two exemplar sets, it is not guaranteed that *both* sets will be clustered. Given that the user is at liberty to define the two sets, this clustering assumption is not valid, especially for the set of negative exemplars. The nearest neighbor approach is ideally suited for situations where the class boundary may be complex and sparsely populated by data points. As the user iterates through the selection process, more points are added near the boundary leading to better results. The nearest neighbor approach also ensures that an image put in the positive exemplar set will always be selected and an image put in the negative exemplar set will never be selected again. This property cannot be ensured by a selection process based on cluster means.

6 Conclusions and Future Work

This paper demonstrates that it is possible to implement representation and retrieval schemes for generic image databases, such that they can be adapted to various types of queries later. The adaptation can be made per query, rather than for the database as a whole. The computational complexity of our algorithm is low because it decouples the dependence on the large number of features and the dependence on the collection size. This makes it very scalable for practical applications involving large collections and diverse image/query types. The exemplar-based user interface makes the technique accessible for lay users. Results indicate that the system converges at a reasonable rate for consistent user inputs and retrieves relevant matches.

A desirable addition to the present work is a consistency checker for the exemplar set. This can be enforced by way of minimum separation between positive and negative exemplars or a maximum within-class variance for each set. Another possible extension is to attach a weight to each of the m selected features, based on the feature relevance measure f_i computed for it. A fuzzy framework will be ideal for this. This will further emphasize the features that are relevant to a particular query, rather than treat each feature of the reduced representation equally.

References

[1] Y. Rui, T. S. Huang, and S.-F. Chang, "Image Retrieval: Past, Present and Future," *JVCIR*, 1998.

[2] B. Srinivasan, S. Nepal, and M. V. Ramakrishna, "Recent Trends in Content Based Image Retrieval Systems," in *Proceedings of ICAPRDT*, pp. 21–28, 1999.

[3] A. Gupta and R. Jain, "Visual Information Retrieval," *Communications of the ACM*, vol. 40, no. 5, 1997.

[4] D. L. Swets and J. Weng, "Using Discriminant Eigenfeatures for Image Retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1996.

[5] R. Ng and A. Sedighian, "Evaluating Multi-dimensional Indexing Structures for Images Transformed by Principal Component Analysis," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1996.

[6] J. R. Smith and S.-F. Chang, "Transform Features for Texture Classification and Discrimination in Large Image Databases," in *Proc. of IEEE International Conference on Image Processing*, 1994.

[7] N. S. Chang and K. S. Fu, "Query by Pictorial Example," *IEEE Transactions on Software Enginnering*, vol. 6, no. 6, 1980.

[8] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafine, D. Lee, D. Petkovic, D. Steele, and P. Yanker, "Query by Image and Video Conteng: The QBIC System," *IEEE Computer*, 1995.

[9] S. Rakshit, D. Deodhare, and R. Krishnan, "Fast Registration of Satellite Images in the Presence of Significant Scene Content Variation," in *Proceedings of SPIE Signal and Data Processing of Small Targets*, vol. 3809, pp. 42–50, 1999.

[10] A. Pentland and R. W. Picard and S. Sclaroff, "Photobook: Tools for Content Based Manipulation of Image Database," *International Journal of Computer Vision*, vol. 18, no. 3, pp. 233–254, 1995.