Hierarchical clustering algorithm for fast image retrieval

Santhana Krishnamachari
Mohamed Abdel-Mottaleb
Philips Research
345 Scarborough Road
Briarcliff Manor, NY 10510
{sgk,msa}@philabs.research.philips.com

ABSTRACT

Image retrieval systems that compare the query image exhaustively with each individual image in the database are not scalable to large databases. A scalable search system should ensure that the search time does not increase linearly with the number of images in the database. We present a clustering based indexing technique, where the images in the database are grouped into clusters of images with similar color content using a hierarchical clustering algorithm. At search time the query image is not compared with all the images in the database, but only with a small subset. Experiments show that this clustering based approach offers a superior response time with a high retrieval accuracy. Experiments with different database sizes indicate that for a given retrieval accuracy the search time does not increase linearly with the database size.

Keywords: image, retrieval, color, histogram, scalable, clustering, hierarchical.

1. Introduction

Content-based image and video retrieval has become an important research topic in recent years. Research interest in this field has escalated because of the proliferation of video and image data in digital form. The growing popularity of the internet, the introduction of new consumer products for digital image and video creation, and the emergence of digital standards for television broadcasting have resulted in a greater demand for efficient storage and retrieval of multimedia data. Content-based retrieval systems for images[1-5,10] based on various image features have been presented. Many of the image retrieval systems extract specific features from a query image and compare these features with the corresponding pre-computed features of all the images in the database. The search time, therefore, increases linearly with the size of the database. Efficient feature representations and similarity measures have been used to speed-up the search process. Still, the growing size of the database results in long search delays that may be unacceptable in many practical situations. Even if the time required to compare two images is very short, the cumulative time needed to compare the query image with all database images is rather long and is probably longer than the time an average user wants to wait.

To achieve the scalability of search to large databases, it must be ensured that the search time does not increase linearly with the database size. Our approach to solving this problem is to create an indexing scheme by grouping the images beforehand based on the content, so that at the time of the query, only the relevant set of clusters need to be examined. The clustering must be performed in such a way that the retrieval accuracy is not sacrificed in this process. Retrieval accuracy here is measured in terms of the retrieval obtained with exhaustive search. A fast search technique presented by Barros, *et.al.*,[8] utilizes the triangle inequality to reduce the search. A technique based on a tree structured vector quantizer and the triangle inequality for fast search has been presented by Chen, *et.al.*[7]. Both these techniques require that the similarity measures used to compare the images be a metric, *i.e.*, the similarity measure satisfies the triangle inequality. However, many useful similarity measures do not satisfy the triangle inequality measure. Indexing techniques using R* and K-D-B trees[11,12] have been proposed for fast range search. However the large dimensionality of the feature vectors may be a problem for these indexing techniques. Dimensionality reduction techniques using principal component analysis or orthogonal transforms can be used to reduce the feature dimension to a manageable number[13]. Performance comparison[14] of different indexing structures has been presented by Gong. The technique presented here does not explicitly place any restriction on the feature dimension nor does it use the triangle inequality property.

Part of the IS&T/SPIE Conference on Storage and Retrieval for Image and Video Databases VII
San Jose, California, January 1999. pp427-435

Here we present a technique for image retrieval based on color from large databases. The goal is to group similar images into clusters and to compute the cluster centers, so that during retrieval, the query image need not be compared exhaustively with all the images in the database. To retrieve similar images for a given query, the query image is initially compared with all the cluster centers. Then a subset of clusters that have the largest similarity to the query image is chosen and all the images in these clusters are compared with the query image. The clustering technique is not directly dependent on the nature of similarity measure used to compare the images, so this technique can be used for any general similarity measure. The experimental evaluation shows that this clustering based indexing technique offers a high retrieval accuracy with a considerable reduction in the number of similarity comparisons required. Experimental evaluation with databases of different sizes shows the scalability of the technique.

The rest of the paper is organized as follows. Section 2 presents an introduction to image clustering. Section 3 presents the details of image representation and similarity measure. Section 4 presents the details of the clustering algorithm. Section 5 presents the results of experimental evaluation by comparing the results of retrieval based on our clustering scheme with the exhaustive search. Section 6 presents the conclusions.

2. Image Clustering

Searching large databases of images is a challenging task especially for retrieval by content. Most search engines calculate the similarity between the query image and all the images in the database and rank the images by sorting their similarities. One problem with this exhaustive search approach is that it does not scale up for large databases. The retrieval time for exhaustive search is the sum of two times: T_{sim} and T_{sort}. T_{sim} is the time to calculate the similarity between the query and every image in the database, and T_{sort} is the time to rank all the images in the database according to their similarity to the query.

$$T_{exhaustive} = nT_{1sim} + O(n\log n)$$

where n is the number of images in the database, T_{1sim} is the time to calculate the similarity between two images, and $O(n\log n)$ is the time to sort n elements.

When the images in the database are clustered, the retrieval time is the sum of three times, the time to calculate the similarity between the query and the cluster centers, the time to calculate the similarity between the query and the images in the nearest clusters and the time to rank these images. Therefore the total search time is:

$$T_{cluster} = kT_{1sim} + lT_{1sim} + O(l\log l)$$

Here k is the number of clusters, l is the number of images in the clusters nearest to the query. Since $k<<n$ and $l<<n$, $T_{cluster} << T_{exhaustive}$.

3. Image representation and Similarity Measure

3.1: Image representation

Several histogram-based approaches have been proposed for image retrieval by color. These approaches are based on calculating a similarity measure between the color histogram of the query image and the images in the database. The difference between these approaches is mainly in their choice of the color space and the similarity measure. Since these approaches use a single histogram to calculate similarities, the results are expected to reflect only global similarity. For example, if a user submits a query image with a sky at the top and sand at the bottom, the retrieved results would have a mix of blue and beige, but not necessarily with blue at the top and beige at the bottom. This can be achieved only if the image representation reflects the local color information.

In this paper, we use a scheme[2] that allows retrieval based on local color features. Images in the database are divided into rectangular regions. Then every image is represented by the set of normalized histograms corresponding to these rectangular regions. It should be noted here that the choice of the rectangular region size is important. In one extreme, the whole image is considered as a single region which reflects the global color information. As the size of the region becomes smaller, the local variations of color information is captured by the histograms. The size of the region should be small enough to emphasize the local color and large enough to offer a statistically valid histogram. In the experiments, images were partitioned into 16 rectangular regions. The number of partitions is represented by P in the following sections.

3.2: Similarity measures

The similarity between two images is measured by calculating the similarity between the histograms of the corresponding rectangular regions. Then a single measure of similarity between the two images is calculated by adding the individual similarities between the corresponding regions. We have used the histogram intersection measure to compare the individual histograms (probabilities). Given two *normalized* histograms, $p = \{p_1, p_2, ..., p_m\}$, $q = \{q_1, q_2, ..., q_m\}$, the similarity measure is defined by

$$s_{p,q} = \sum_i min(p_i, q_i)$$

4. Hierarchical Clustering

Let n be the number of images in the database, the similarity between all pairs of images is precomputed. The hierarchical clustering[6] is performed as follows:

1. The n images in the database are placed in n distinct clusters. These clusters are indexed by $\{C_1, C_2, ..., C_n\}$. For the kth cluster, the set E_k contains all the images contained in that cluster and N_k denote the number of images in the cluster. $E_k = \{k\}$ and $N_k = 1$ for $k = 1,2,...,n$.
2. Two clusters C_k and C_l are picked such that the similarity measure $S_{k,l}$ is the largest. (The similarity measure between two clusters is defined in the following subsection). These two clusters are *merged* into a new cluster C_{n+1}. This reduces the total number of unmerged clusters by one. E_{n+1} is updated to $E_{n+1} = \{E_k \cup E_l\}$ and N_{n+1} is updated to $N_{n+1} = N_k + N_l$. Out of the two children of C_{n+1}, one is referred to as the right child $RC_{n+1} = k$ and the other as the left child $LC_{n+1} = l$. The similarity measures between the new cluster C_{n+1} and all the other unmerged clusters are computed as discussed below.
3. Steps 2 and 3 are repeated until the number of clusters has reduced a required number or the largest similarity measure between clusters has dropped to some lower threshold.

Figure 1 shows a simple example of hierarchical clustering with 8 images. The clustering is stopped after reaching two clusters. For this example, $N_{14} = 5$, $N_{12} = 3$, $E_{14} = \{1,2,3,4,5\}$ and $E_{12} = \{6,7,8\}$. Also, $RC_{14} = 5$ and $LC_{14} = 13$. Each cluster has an associated tree. The clustering algorithm presented here does not directly depend on the nature of similarity measure. We have presented a performance evaluation[9] of the clustering algorithm for different similarity measures elsewhere.

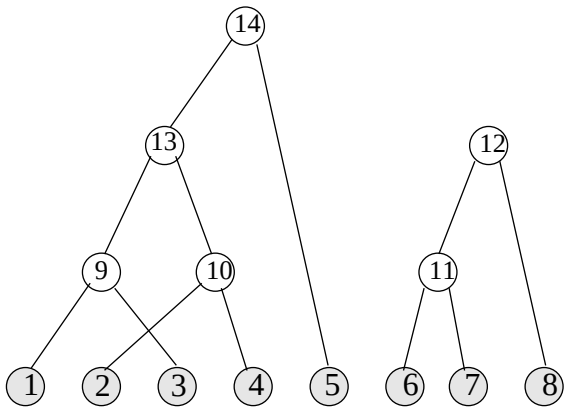

Figure 1: A sample of cluster merging process with hierarchical clustering.

The similarity measure between two images is defined in the previous section. The measure of similarity, $S_{k,l}$, between two clusters, C_k and C_l, is defined in terms of the similarity measures of the images that are contained in those clusters as follows:

$$S_{k,l} = \frac{\sum_{i,j \in \{E_l \cup E_k\}, i \neq j} s_{i,j}}{P_{(N_k + N_l)}} \quad (1)$$

where $s_{i,j}$ is the similarity measure between images i and j, E_k is the set of images present in the cluster C_k and N_k is the number of images in cluster C_k. P_n is the number of pairs of images in a cluster with n images:

$$P_n = (n-1)\frac{n}{2}$$

In other words, $S_{k,l}$ is defined to be the average similarity between all pairs of images that will be present in the cluster obtained by merging C_k and C_l. Since the similarity between clusters is defined in terms of the similarity measures between the images in the clusters, there is no need to compute the cluster centers every time two clusters are merged.

When two clusters, C_k and C_l, are merged to form a new cluster C_m, then it is necessary to compute the similarity of this cluster with all other clusters as given in Eq. (1). This computation is cumbersome as shown below. For any cluster C_t, the similarity measure between C_m and C_t is:

$$S_{m,t} = \frac{\sum_{i,j \in \{E_l \cup E_k\}, i \neq j} s_{i,j} + \sum_{i,j \in E_t, i \neq j} s_{i,j} + \sum_{i \in E_l \cup E_k, j \in E_t} s_{i,j}}{P_{(N_l + N_k + N_t)}} \quad (2)$$

and $S_{m,m}$ is set equal to $S_{k,l}$.

A simple recursive method to achieve the same can be obtained using the fact that the first term in Eq. (2) is equal to $P_{(N_l + N_k)} S_{l,k}$, the second term is equal to $P_{N_t} S_{t,t}$, and the third term is equal to $P_{(N_l + N_t)} S_{l,t} + P_{(N_k + N_t)} S_{k,t} - P_{N_l} S_{l,l} - P_{N_k} S_{k,k} - 2P_{N_t} S_{t,t}$. Thus the similarity $S_{m,t}$ can be obtained from, $S_{l,k}$, $S_{l,t}$, $S_{k,t}$, $S_{t,t}$, $S_{l,l}$, and $S_{k,k}$. The following equation requires far less computation compared to the one above.

$$S_{m,t} = \frac{P_{(N_l + N_k)} S_{l,k} + P_{(N_l + N_t)} S_{l,t} + P_{(N_k + N_t)} S_{k,t} - P_{N_l} S_{l,l} - P_{N_k} S_{k,k} - P_{N_t} S_{t,}}{P_{(N_l + N_k + N_t)}} \quad (3)$$

In Eq. (2), $S_{m,t}$ is computed by summing up the similarity measures of all pairs of images in C_m and C_t, and hence the computation grows as the square of number of images present in the two clusters. The computation in Eq. (3) is independent of the number of images in the clusters. At the beginning of clustering, for all the clusters $S_{i,j}$ is set equal to $s_{i,j}$ and $S_{i,i}$ is set equal to zero.

4.1. Cluster Center Computation

After clustering, an appropriate center has to be obtained for each cluster. Since, every image is represented by P histograms corresponding to P partitions, it is apt to use a similar representation for cluster centers. A simple representation would be the average of histograms of all the images in the cluster. Since the number of images in each cluster can be large, averaging over all the images may be computationally expensive. One solution is to find a smaller number of representative images and use the averages of their corresponding histograms to represent the cluster center. In the following discussion, the maximum number of representative images for a cluster is limited to R. These representative images have to be chosen carefully so that the cluster center computed from them is close to all the images in the cluster.

The tree structure that is obtained as a by-product of the clustering algorithm can be effectively used to select the representative set of images. In Figure 1, let us consider selecting the representative images for cluster C_{14}. From the tree structure it can be inferred that the images 1 and 3 belong to cluster 9 and images 2 and 4 belong to cluster 10. Hence a good selection of representative images, for R=3, is to select one from {1,3}, another from {2,4} and 5. If R=2, then it is apt to select one from {1,2,3,4} and 5 as representatives. Similarly for C_{12}, it is better to select 6 and 8 or 7 and 8 instead of 6 and 7. Such a selection

would result in a representative set that captures the diversity of images present in the cluster.

Let C_i be a cluster for which a set of representative images is to be selected. A set of R nodes is chosen from the tree associated with C_i and from each of these nodes a representative image is selected, resulting in R representative images. The following steps explain this procedure:

1. Set n=0 and form a set $\Re_n = \{i\}$. If R=1, then go to Step 5.
2. Each element in $\Re_n$ is an index of one of the nodes in the tree associated with C_i. Find an element k such that N_k is the largest.
3. Form a new set $\Re_{n+1}$ by copying all the elements of $\Re_n$ except k and adding RC_k and LC_k.
4. Repeat steps 2 and 3 until the number of elements contained in $\Re_n$ is equal R.
5. Now $\Re_n$ contains R nodes from the tree associated with C_i. From each of these nodes a representative image is chosen. If $k \in \Re_n$, and N_k=1 and the selection is straightforward, *i.e.*, the associated image is selected. If N_k>1, then it is necessary to select a single image representative from E_k. This is done by selecting the one that has the maximum average similarity measure with the other N_k-1 images of E_k.

For the example shown in Figure 1, finding a representative set of images for C_{14} with R=2, begins with $\Re_0 = \{14\}$ and $\Re_1 = \{13, 5\}$ and the iteration stops here as $\Re_1$ contains two elements already. Now, since C_5 has a single element, image 5, is chosen as one representative. Another representative is selected from C_{13} that contains four images {1,2,3,4} by calculating the average similarity of each image with the other three images and then choosing the one with the maximum average similarity. Assuming that image 2 has the largest average similarity, the representative set of images for cluster C_{14} is {2,5}.

After selecting a set of R representative images, the averages of their P corresponding histograms are used to represent the cluster center. In the example presented above, the cluster center for C_{14} is represented by P histograms which are obtained by averaging the corresponding histograms of partitions of images 2 and 5.

4.2. Cluster Center Optimization

The technique presented in the previous subsection enables us to compute a cluster center for each unmerged cluster. After computing the cluster center, it is necessary to evaluate the optimality of the clustering and the computation of cluster centers. Cluster centers are optimal, if for each image contained in a cluster, the similarity measure between the image and that cluster center is larger than the similarity measure between the image and all the other cluster centers. However this may not be true, especially given the fact that we have only used a representative set and not all the images in the cluster to compute the cluster centers. As a result, an image may have a larger similarity measure with other cluster centers than with its own cluster center. To optimize the cluster centers, it is necessary to move all such images to their closest clusters. Note that the similarity measure between an image and a cluster center is computed in exactly the same manner as presented in Section 3 for computing the similarity measure between two images.

The cluster center optimization is performed as follows:

1. For each of the n images in the database, find the similarity measures between the image and all the cluster centers. If the cluster with the maximum similarity measure is the same cluster in which the image is present then nothing is done. If not, move the image from the cluster in which it resides to the cluster that it is most similar to. Rearrange the trees of both clusters to reflect this removal and addition as presented in Sections 4.2.1 and 4.2.2.
2. Recompute the cluster centers for all the clusters that have been rearranged as described in Section 4.1.
3. Repeat steps 1 and 2 until the number of images to be moved is below a threshold. Then, move these images as in Step 1 and do not recompute the centers as in Step 2.

Thus at the end of Step 3, all images exhibit the largest similarity measure with the center of the cluster in which they are present and hence the clustering is optimal.

4.2.1. Removal of Nodes:

When removing an image from a cluster, there are two possible scenarios depending on whether the removed node is a direct child of the root of the cluster or not. These two scenarios are discussed with the example presented in Figure 1, by removing nodes 5 and 2 from C_{14}. Node 5 is a direct child of $C_{14,}$ whereas node 2 is not. Removing these two nodes is done rather differently. When node 5 is removed, node 13 becomes redundant and hence it is removed and replaced with node 14. Now $N_{14}=4$, $E_{14}=\{1,2,3,4\}$, $RC_{14}=10$, and $LC_{10}=9$.

When node 2 is removed from the original tree, node 10 will have only one child and hence it is removed. Node 4 becomes a child of node 13 as shown in Figure 2. Now, $N_{13}=3$, $E_{13}=\{1,3,4\}$, and $RC_{13}=4$. Appropriate changes are also made for the parameters associated with C_{14}.

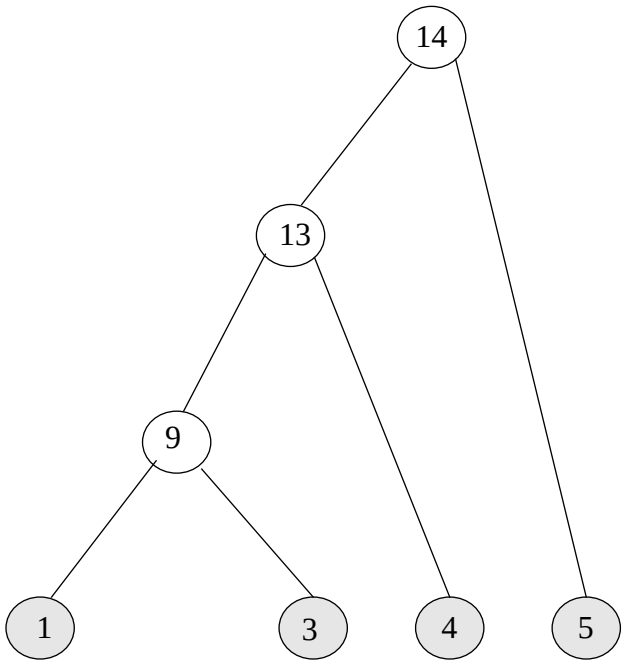

Figure 2: Removing node 2 from cluster C_{14}.

4.2.2. Addition of Nodes:

When adding an image to a cluster, it is necessary to decide where to insert the new node. This decision is made top-to-bottom. Suppose a node is to be added to C_k with right and left children C_i and C_j respectively. First E_k and N_k are updated to reflect the addition of the node to C_k. The decision of whether to add the node to C_i or C_j is again based on the similarity measure. The node is added to C_i, if the average similarity of the node (image) with all the images in C_i is larger than the average similarity of the node with all the images in C_j and vice versa. If the node is added to C_i, the parameters associated with C_i are updated. The node is then added to either the right or left child of C_i. This process is repeated recursively until the leaves of the tree are reached. This is shown in the Figure 5, where image 2 is added to cluster C_{12} of Figure 2. First the parameters associated with C_{12} are updated, $N_{12}=4$ and $E_{12}=\{6,7,8,2\}$. Then to decide whether it has to be added to C_{11} or C_8, the similarity measure $s_{2,8}$ is compared with $(s_{2,6}+s_{2,7})/2$. If the latter is larger, then E_{11} is updated to $\{6,7,2\}$ and N_{11} is updated to 3. Then $s_{2,6}$ and $s_{2,7}$

are compared and if $s_{2,7}$ is larger, a new node C_{15} is created with RC_{15}=2 and LC_{15}=7.

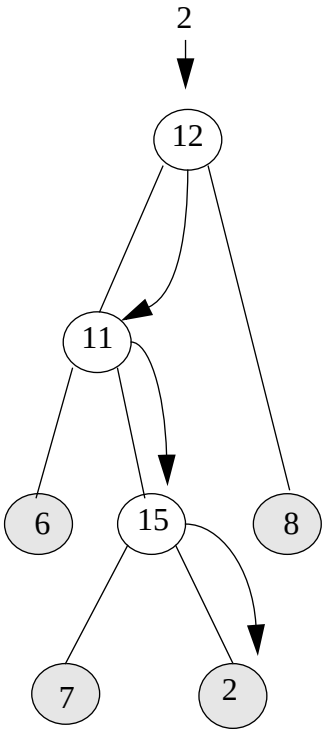

Figure 3: Adding node 2 to cluster C_{12}.

5. Performance Evaluation

Performance evaluation has long been a difficult problem in image processing and computer vision, and content-based retrieval is no exception. This is primarily because of the difficulty associated with relevant quantitative measures for evaluation. In content-based retrieval, precision and recall measures have been frequently used[5] to evaluate the performance of retrieval algorithms. In our case, there is no need to evaluate the subjective quality of the retrieval, but it is only necessary to compare the retrieval with clustering against the exhaustive search.

We have used a quantitative measure to compare the retrieval results based on clustering against the retrieval results based on exhaustive search. A user searching through a large database, is interested in only the top few best matches (say 10 or 20). Hence, if the retrieval with clustering returns the same few best matches as the ones returned by retrieval based on exhaustive search, then the retrieval with clustering is accurate. Let us assuming that the user is interested in only the top K best matches and that M is the number of images that are present in both the top K results returned by retrieval with and without clustering. Then, the retrieval accuracy with clustering ψ_i, when i th image is used as a query is defined as:

$$\psi_i = \frac{M}{K} 100$$

The average retrieval accuracy with clustering, A_K, is obtained by taking the average ψ_i over all query images.

$$A_K = \frac{1}{n} \sum_{i=1}^{n} \psi_i$$

The results of hierarchical clustering presented here are obtained with a database of 3856 images, 200 of these images are taken from two collections of COREL Professional Photo CD-ROMs, the Sampler II - Series 400000 and the Sampler - Series 200000. The rest of the images are obtained from the Department of Water Resources, California. The images are of widely

varying colors and scene content. The hierarchical clustering algorithm was applied to the 3856 images in the database resulting in 174 clusters with the largest cluster having 80 images and the smallest cluster having 5 images. The number of clusters, n_c, is chosen such that the average number of images per cluster is 22, *i.e.*, n_c=3856/22=174.

We conducted two sets of experiments. In the first set, we used each of the 3856 images in the database as query images and in the second set we used a set of 300 images that are not a part of the database as query images. The query images are compared with all the cluster centers and the top 3,4,7,10,13,19,25, and 31 clusters are chosen. All the images in the selected clusters are compared exhaustively with the query and the most similar images are displayed to the user as the result of the search. The average retrieval accuracy and the average number of required similarity comparisons are obtained by taking the average over all the query images. We found the performance results are almost similar for the two sets of experiments, so we present the results for the latter case. Figure 4 shows the plot of the average retrieval accuracy against the average number of the similarity comparisons required. The plot consists of 8 points, obtained by examining the top 3,4,7,10,13,19,25, and 31 clusters. The leftmost point corresponds to the result obtained by examining 3 clusters and the rightmost point corresponds to the result obtained by examining 31 clusters. From the figure it can be seen that for K=10, *i.e.*, the user is interested in the top 10 retrieved images, 90% retrieval accuracy can be obtained with 540 similarity comparisons as opposed to 3856 comparisons that would be required for exhaustive search. We expect that the computational gain will be larger as the size of database increases. Table 1 shows the number of similarity comparisons required to achieve a 90% retrieval accuracy for three different database sizes. From the table it can be seen that the number of similarity comparisons required does not increase linearly with the number of images in the database and the speed-up factor increases with the size of the database.

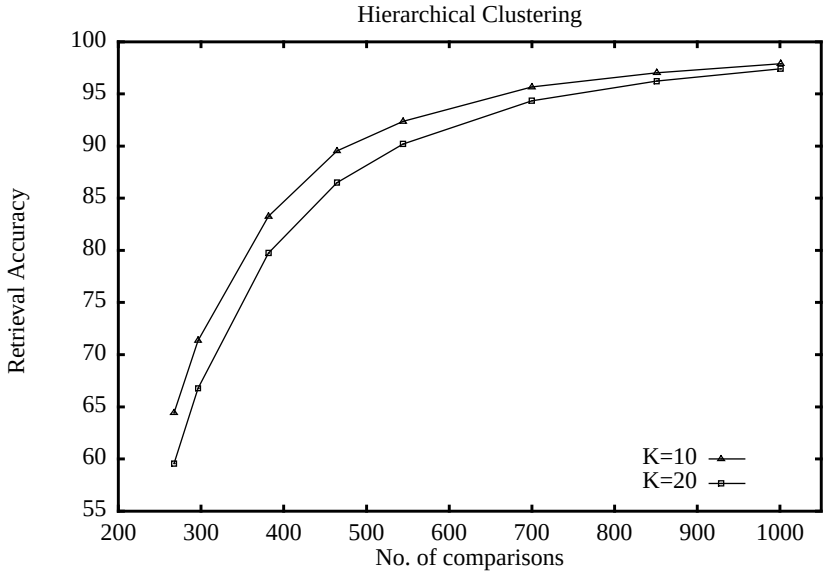

Figure 4: Retrieval accuracy with hierarchical clustering

Table 1: Retrieval Performance for different database sizes

No. of database images	No. of similarity comparisons	Speedup
1000	277	3.61
2000	358	5.59
3856	540	7.14

6. Conclusions

In this paper we have presented an algorithm for scalable image retrieval by color, where images are represented by local color histograms. The similarity between images is calculated based on local color properties. Images in the database are clustered into groups having similar color contents. This grouping enables searching only images that are relevant to the query image. For a database containing 3856 images, a retrieval accuracy of 90% can be achieved with only 540 similarity comparisons. Experiments with different database sizes show that the number of similarity comparisons required to achieve a given retrieval accuracy does not increase linearly with the size of the database thus making the algorithm scalable to large databases.

Acknowledgments

We wish to thank Dave Kearney of Dept. of Water Resources, California and Ginger Ogle of University of California, Berkeley for providing us with the images used in this paper.

References

1. M. J. Swain and D. H. Ballard, "Color Indexing", *Intl. J. of Computer Vision*, 7(1), pp. 11-32, 1991.
2. M. Abdel-Mottaleb, N. Dimitrova, R. Desai, and J. Martino, "CONIVAS: CONtent-based image and video access system", *Proc. of ACM Intl. Multimedia Conference, Nov.* 1996.
3. W. Niblack, *et.al*., "The QBIC Project: querying images by content using color, texture and shape." *In Storage and Retrieval for Image and Video Databases I, Vol.* 1908, *SPIE Proceedings, Feb.* 1993.
4. J. Smith and S.-F. Chang, "A Fully Automated Content-based Image Query System", *Proc. of ACM Intl. Multimedia Conference, Nov.* 1996.
5. H. Zhang, Y. Gong, C. Y. Low and S. W. Smoliar, "Image retrieval based on color features: an evaluation study", *Proc of SPIE, Vol.* 2606, pp. 212-220, 1995.
6. A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, New Jersey, 1988.
7. J-Y. Chen, C.A. Bouman, and J.P. Allebach, "Fast Image Database Search using Tree-Structure VQ", *Proc. of ICIP, Vol* 1, 1997.
8. J. Barros, *et.al.*, "Using the triangle Inequality to reduce the number of computations required for similarity-based retrieval", *Proc. of SPIE/IS&T Conf. on Storage and Retrieval of Image and Video Databases IV, Vol*. 2670, 1996.
9. M. Abdel-Mottaleb, S. Krishnamachari, and N.J. Mankovich, "Performance Evaluation of Clustering Algorithms for Scalable Image Retrieval", *In IEEE Computer Society Workshop on Empirical Evaluation of Computer Vision Algorithms", Santa Barbara*, 1998.
10. W.Y. Ma and B.S. Manjunath, "NeTra: A Toolbox for Navigating Large Image Databases", pp 568-572, *Proc. of ICIP*, Vol 1, 1997.
11. J. T. Robinson, "The K-D-B-tree: a search structure for large multidimensional dynamic indexes", *In Proc. of ACM SIGMOD, Ann Arbor, April* 1981.
12. N. Beckman, *et. al*., "The R*-tree: An efficient and robust access method for points and rectangles", I*n Proc. of ACM SIGMOD, Atlantic City, May* 1990.
13. C. Faloutsos, *et.al.,* "Efficient and Effective Querying by Image Content", *Journal of Intelligent Information Systems*, July 1994.
14. Y. Gong, *Intelligent Image Databases: Towards Advanced Image Retrieval*, Kluwer Academic Publishers, 1998.