# A Graph–Theoretic Approach to Image Database Retrieval

Selim Aksoy and Robert M. Haralick
Intelligent Systems Laboratory
Department of Electrical Engineering
University of Washington
Seattle, WA 98195-2500 U.S.A.
{aksoy,haralick}@isl.ee.washington.edu
http://isl.ee.washington.edu

**Abstract.** Feature vectors that are used to represent images exist in a very high dimensional space. Usually, a parametric characterization of the distribution of this space is impossible. It is generally assumed that the features are able to locate visually similar images close in the feature space so that non-parametric approaches, like the k-nearest neighbor search, can be used for retrieval.
This paper introduces a graph–theoretic approach to image retrieval by formulating the database search as a graph clustering problem to increase the chances of retrieving similar images by not only ensuring that the retrieved images are close to the query image, but also adding another constraint that they should be close to each other in the feature space. Retrieval precision with and without clustering are compared for performance characterization. The average precision after clustering was 0.78, an improvement of 6.85% over the average precision before clustering.

## 1 Motivation

Like in many computer vision and pattern recognition applications, algorithms for image database retrieval have an intermediate step of computing feature vectors from the images in the database. Usually these feature vectors exist in a very high dimensional space where a parametric characterization of the distribution is impossible. In an image database retrieval application we expect to have visually similar images close to each other in the feature space. Due to the high dimensionality, this problem is usually not studied and the features are assumed to be able to locate visually similar images close enough so that non-parametric approaches, like the k-nearest neighbor search, can be used for retrieval.

Unfortunately, none of the existing feature extraction algorithms can always map visually similar images to nearby locations in the feature space and it is not uncommon to retrieve images that are quite irrelevant simply because they are close to the query image. We believe that a retrieval algorithm should be able to retrieve images that are not only close (similar) to the query image but also close (similar) to each other.

In this work, we introduce a graph–theoretic approach for image retrieval by formulating the database search as a graph clustering problem. Graph–theoretic approaches have been a popular tool in the computer vision literature, especially in object matching. Recently, graphs were used in image segmentation [8, 7, 4] by treating the image as a graph and defining some criteria to partition the graph. Graphs did not receive significant attention in image retrieval algorithms mainly due to the computational complexity of graph-related operations. Huet and Hancock [5] used attributed graphs to represent line patterns in images and used these graphs for image matching and retrieval.

Clustering the feature space and visually examining the results to check whether visually similar images are actually close to each other is an important step in understanding the behavior of the features. This can help us determine the effectiveness of both the features and the distance measures in establishing similarity between images. In their Blobworld system, Carson *et al.* [3] used an expectation-maximization based clustering algorithm to find canonical blobs to mimic human queries. In our work we also use the idea that clusters contain visually similar images but we use them in a post-processing step instead of forming the initial queries.

The paper is organized as follows. First, the features used are discussed in Section 2. Then, a new algorithm for image retrieval is introduced in Section 3, which is followed by the summary of a graph–theoretic clustering algorithm in Section 4. Experiments and results are presented in Section 5. Finally, conclusions are given in Section 6.

## 2 Feature Extraction

The textural features that are used were described in [1, 2]. The feature vector consists of two sets of features which intend to perform a multi-scale texture analysis which is crucial for a compact representation in large databases containing diverse sets of images. The first set of features are computed from the line-angle-ratio statistics which is a texture histogram of the angles between intersecting line pairs and the ratio of the mean gray levels inside and outside the regions spanned by those angles. The second set of features are the variances of gray level spatial dependencies and are computed from the co-occurrence matrices for different spatial relationships. Each component in the 28-dimensional feature vector is normalized to the $[0, 1]$ interval by an equal probability quantization.

## 3 Image Retrieval

After computing the feature vectors for all images in the database, given a query image, we have to decide which images in the database are relevant to it. In most of the retrieval algorithms, a distance measure is used to rank the database images in ascending order of their distances to the query image, which is assumed to correspond to a descending order of similarity. In our previous work [1, 2] we defined a likelihood ratio to measure the relevancy of two images, one being the query image and one being a database image, so that image pairs which had a high likelihood value were classified as "relevant" and the ones which had a lower likelihood value were classified as "irrelevant". The distributions for the relevance and irrelevance classes were estimated from training sets and the likelihood values were used to rank the database images.

We believe that a retrieval algorithm should be able to retrieve images that are not only similar to the query image but also similar to each other, and formulate a new retrieval algorithm as follows. Assume we query the database and get back the best $N$ matches. Then, for each of these $N$ matches we can do a query and get back the best $N$ matches again. Define $S$ as the set containing the query image and at most $N^2+N$ images that are retrieved as the results of the original query and $N$ additional queries. Then, we can construct a graph with the images in $S$ as the nodes and can draw edges between each query image and each image in the retrieval set of that query image. We call these edges the set $R$ where $R = \{(i, j) \in S \times S \mid \text{image } j \text{ is in the retrieval set when image } i \text{ is the query}\}$. The distances between images which correspond to two nodes that an edge connects can also be assigned as a weight to that edge. We want to find the connected clusters of this graph $(S, R)$ because they correspond to similar images. The clusters of interest are the ones that include the original query image. The ideal problem now becomes finding the maximal $P$, where $P \subseteq S$ such that $P \times P \subseteq R$. This is called a clique of the graph. The images that correspond to the nodes in $P$ can then be retrieved as the results of the query.

An additional thing to consider is that the graph $(S, R)$ can have multiple clusters. In order to select the cluster that will be returned as the result of the query, additional measures are required. In the next section we define the term "compactness" for a set of nodes. The cluster with the maximum compactness can then be retrieved as the final result. If more than one such cluster exist, we can select the one with the largest number of nodes or can compute the sum of the weights of the edges in each of the clusters and select the one that has the minimum total weight.

This method increases the chance of retrieving similar images by not only ensuring that the retrieved images are close to the query image, but also adding another constraint that they should be close to each other in the feature space. In the next section we describe a graph–theoretic clustering algorithm which is used to find the clusters. Section 5 presents experimental results.

## 4 Graph–Theoretic Clustering

In the previous section, we proposed that cliques of the graph correspond to similar images. Since finding the cliques is computationally too expensive, we use the algorithm by Shapiro and Haralick [6] that finds "near-cliques" as dense regions instead of the maximally connected ones. Another consideration for speed is to compute the $N$-nearest neighbor searches offline for all the images in the database so that only one $N$-nearest neighbor search is required for a new query, which is the same amount of computation for the classical search methods.

In the following sections, first we give some definitions, then we describe the algorithm for finding dense regions, and finally we present the algorithm for graph–theoretic clustering. The goal of this algorithm is to find regions in a graph, i.e. sets of nodes, which are not as dense as major cliques but are compact enough within some user specified thresholds.

### 4.1 Definitions

- $(S, R)$ represents a *graph* where $S$ is the set of nodes and $R \subseteq S \times S$ is the set of edges.
- $(X, Y) \in R$ means $Y$ is a *neighbor* of $X$. The set of all nodes $Y$ such that $Y$ is a neighbor of $X$ is called the *neighborhood* of $X$ and is denoted by $\text{Neighborhood}(X)$.
- *Conditional density* $D(Y|X)$ is the number of nodes in the neighborhood of $X$ which have $Y$ as a neighbor; $D(Y|X) = \#\{N \in S \mid (N, Y) \in R \text{ and } (X, N) \in R\}$.
- Given an integer $K$, a *dense region* $Z$ around a node $X \in S$ is defined as $Z(X, K) = \{Y \in S \mid D(Y|X) \geq K\}$. $Z(X) = Z(X, J)$ is a *dense region candidate* around $X$ where $J = \max\{K \mid \#Z(X, K) \geq K\}$.
- *Association* of a node $X$ to a subset $B$ of $S$ is defined as

$$A(X|B) = \frac{\#\{\text{Neighborhood}(X) \cap B\}}{\#B} , \qquad 0 \leq A(X|B) \leq 1. \qquad (1)$$

- *Compactness* of a subset $B$ of $S$ is defined as

$$C(B) = \frac{1}{\#B} \sum_{X \in B} A(X|B) , \qquad 0 \leq C(B) \leq 1. \qquad (2)$$

### 4.2 Algorithm for Finding Dense Regions

To determine the dense region around a node $X$,

1. Compute $D(Y|X)$ for every other node $Y$ in $S$.
2. Use the densities to determine a dense–region candidate set for node $X$ by finding the largest positive integer $K$ such that $\#\{Y \mid D(Y|X) \geq K\} \geq K$.
3. Remove the nodes with a low association (determined by the threshold MINASSOCIATION) from the candidate set. Iterate until all of the nodes have high enough association.
4. Check whether the remaining nodes have high enough average association (determined by the threshold MINCOMPACTNESS).
5. Check the size of the candidate set (determined by the threshold MINSIZE).

When MINASSOCIATION and MINCOMPACTNESS are both 1, the resulting regions correspond to the cliques of the graph.

### 4.3 Algorithm for Graph Theoretic Clustering

Given dense regions, to find the clusters of the graph,

1. Merge the regions that have enough overlap, determined by the threshold MINOVERLAP, if all of the nodes in the set resulting after merging have high enough associations.
2. Iterate until no regions can be merged.

The result is a collection of clusters in the graph. Note that a node can be a member of multiple clusters because of the overlap allowed between them.

## 5 Experiments and Results

The test database consists of 340 images which were randomly selected from a database of approximately 10,000 aerial and remote sensing images. The images were grouped into 7 categories; parking lots, roads, residential areas, landscapes, LANDSAT USA, DMSP North Pole and LANDSAT Chernobyl, to form the groundtruth.

### 5.1 Clustering Experiments

The first step of testing the proposed retrieval algorithm is to check whether the clusters formed by the graph–theoretic clustering algorithm are visually consistent or not. First, each image was used as a query to search the database, and for each search, $N$ top-ranked images were retrieved. Then, a graph was formed with all images as nodes and for each node $N$ edges correspond to its $N$ top-ranked images. Finally, the graph was clustered by varying the parameters like $N$, MINASSOCIATION and MINCOMPACTNESS. In order to reduce the possible number of parameters, MINSIZE and MINOVERLAP were fixed as 12 and 0.75 respectively. The resulting clusters can overlap. This is a desired property because image content is too complex to be grouped into distinct categories. Hence, an image can be consistent with multiple groups of images.

To evaluate the consistency of a cluster, we define the following measures. Given a cluster of $K$ images,

$$CorrectAssociation_k = \frac{\#\{i \mid GT(i) = GT(k), i = 1, \ldots, K\}}{K} \quad (3)$$

gives the percentage of the cluster that image $k$ is correctly associated with, where $GT(i)$ is the groundtruth group that image $i$ belongs to. Then, consistency is defined as

$$Consistency = \frac{1}{K} \sum_{k=1}^{K} CorrectAssociation_k. \quad (4)$$

To select the best set of parameters, we define a cost function

$$Cost = 0.7(1 - Consistency) + 0.3(Percentage\ of\ unclustered\ images) \quad (5)$$

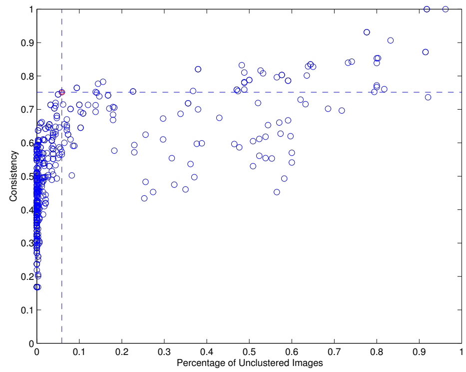



**Fig. 1.** *Consistency* vs. *Percentage of unclustered images* for $N \in \{10, \ldots, 70\}$, MINCOMPACTNESS $\in \{0.3, \ldots, 1.0\}$, MINASSOCIATION $\in \{0, \ldots, \text{MINCOMPACTNESS}\}$, MINSIZE $= 12$, MINOVERLAP $= 0.75$. Dashed lines correspond to the minimum cost.

and select the parameter set that minimizes it. Here *Consistency* is averaged over all resulting clusters.

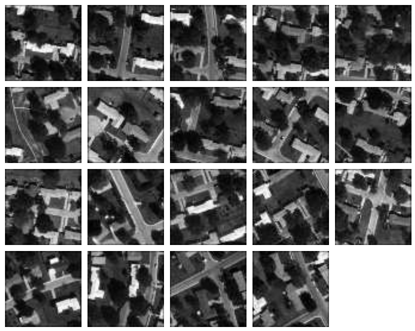
Among all possible combinations of the parameters given in Figure 1, the best parameter set was found as $\{N, \text{MINCOMPACTNESS}, \text{MINASSOCIATION}\} = \{15, 0.6, 0.4\}$, corresponding to an average *Consistency* of 0.75 with 6% of the images unclustered. Example clusters using these parameters are given in Figure 2. We observed that decreasing $N$ or increasing MINCOMPACTNESS or MINASSOCIATION increases both *Consistency* and *Percentage of unclustered images*.

### 5.2 Retrieval Experiments

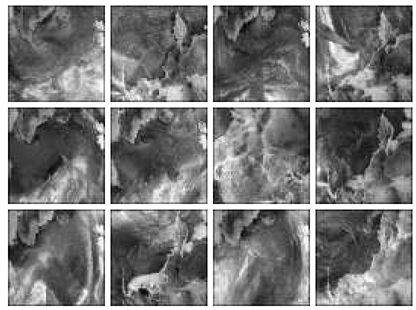
We also performed experiments using all of the 340 groundtruthed images in the database as queries and, using the parameter set selected above, retrieved images in the clusters with the maximum compactness for each query. For comparison, we also retrieved only 12 top–ranked images (no clustering) for each query.

Example queries without and with clustering are shown in Figures 3 and 4. We can observe that some images that are visually irrelevant to the query image can be eliminated after the graph–theoretic clustering. An average precision of 0.78 (compared to 0.73 when only 12 top-ranked images are retrieved) for the whole database showed that approximately 9 of the 12 retrieved images belong to the same groundtruth group, i.e. are visually similar to the query image.

We also observed that, in order to get an improvement by clustering, the initial precision before clustering should be large enough so that the graph is not dominated by images that are visually irrelevant to the query image. In our experiments, when the initial precision was less than 0.5, the average precision after clustering was 0.19. For images with an initial precision greater than 0.5, the average precision after clustering was 0.93. The better the features are, the larger the improvement after clustering becomes.

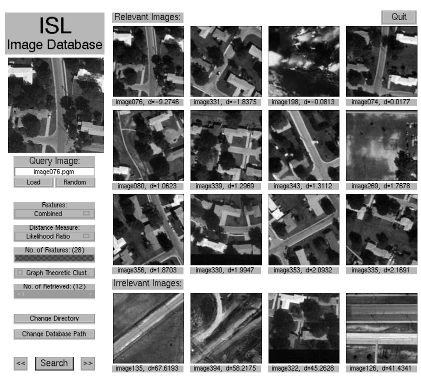


(a) *Consistency* = 1

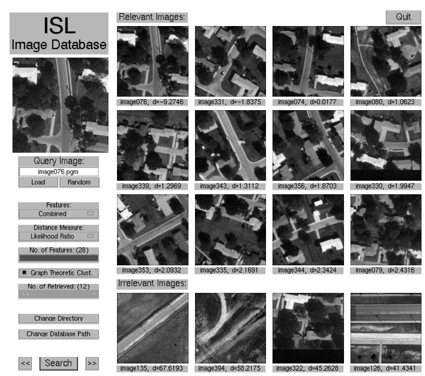


(b) *Consistency* = 1

**Fig. 2.** Example clusters for $N$=15, MINCOMPACTNESS=0.6, MINASSOCIATION=0.4, MINSIZE=12, MINOVERLAP=0.75.

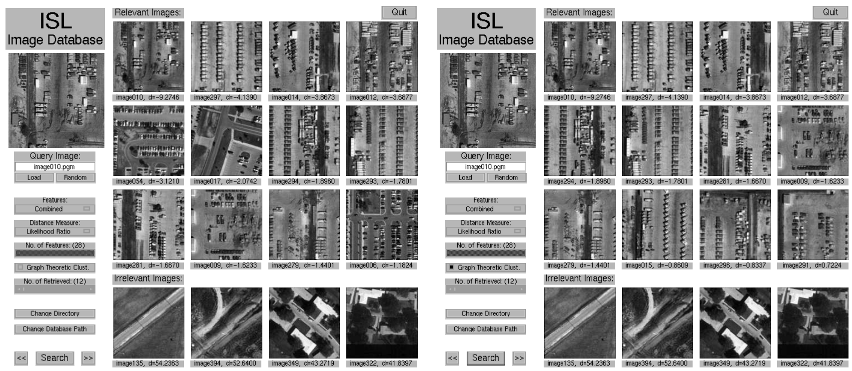



(a) Using only 12 top–ranked images.




(b) Using graph–theoretic clustering.

**Fig. 3.** Example query 1. Upper left image is the query. Among the retrieved images, first three rows show the 12 most relevant images in descending order of similarity and the last row shows the 4 most irrelevant images in descending order of dissimilarity. When clustering is used, only 12 images that have the smallest distance to the original query image are displayed if the cluster size is greater than 12.




(a) Using only 12 top–ranked images. (b) Using graph–theoretic clustering.

**Fig. 4.** Example query 2.

## 6 Conclusions

This paper addressed the problem of retrieving images that are quite irrelevant to the query image, which is caused by the assumption that the features are always able to locate visually similar images close enough in the feature space.

We introduced a graph–theoretic approach for image retrieval by formulating the database search as a problem of finding the cliques of a graph. Experiments showed that some images that are visually irrelevant to the query image can be eliminated after the graph–theoretic clustering. Average precision for the whole database showed that approximately 9 of the 12 retrieved images belong to the same groundtruth group, i.e. are visually similar to the query image.

## References


1. S. Aksoy and R. M. Haralick. Textural features for image database retrieval. In *Proc. of IEEE Workshop on CBAIVL, in CVPR'98*, pages 45–49, June 1998.
2. S. Aksoy, "Textural features for content-based image database retrieval," Master's thesis, University of Washington, Seattle, WA, June 1998.
3. C. Carson *et al.*. Color- and texture-based image segmentation using EM and its application to image querying and classification. *submitted to PAMI.*
4. P. Felzenszwalb and D. Huttenlocher. Image segmentation using local variation. In *Proc. of CVPR*, pages 98–104, June 1998.
5. B. Huet and E. Hancock. Fuzzy relational distance for large-scale object recognition. In *Proc. of CVPR*, pages 138–143, June 1998.
6. L. G. Shapiro and R. M. Haralick. Decomposition of two-dimensional shapes by graph-theoretic clustering. *IEEE PAMI*, 1(1):10–20, January 1979.
7. J. Shi and J. Malik. Normalized cuts and image segmentation. In *Proc. of CVPR*, pages 731–737, June 1997.
8. Zhenyu Wu and Richard Leahy. An optimal graph theoretic approach to clustering: Theory and its application to image segmentation. *IEEE PAMI*, 15(11):1101–1113, November 1993.