

Query Decomposition: A Multiple Neighborhood Approach to Relevance Feedback Processing in Content-based Image Retrieval

Kien A. Hua Ning Yu Danzhou Liu
School of Electrical Engineering and Computer Science
University of Central Florida
Orlando, FL 32816
{kienhua, nyu, dzliu}@cs.ucf.edu

Abstract

Today's Content-Based Image Retrieval (CBIR) techniques are based on the "k-nearest neighbors" (k -NN) model. They retrieve images from a single neighborhood using low-level visual features. In this model, semantically similar images are assumed to be clustered in the high-dimensional feature space. Unfortunately, no visual-based feature vector is sufficient to facilitate perfect semantic clustering; and semantically similar images with different appearances are always clustered into distinct neighborhoods in the feature space. Confinement of the search results to a single neighborhood is an inherent limitation of the k -NN techniques. In this paper we consider a new image retrieval paradigm – the **Query Decomposition** model – that facilitates retrieval of semantically similar images from multiple neighborhoods in the feature space. The retrieval results are the k most similar images from different relevant clusters. We introduce a prototype, and present experimental results to illustrate the effectiveness and efficiency of this new approach to content-based image retrieval.

1. Introduction

Nearest neighbor search [14] is the standard technique for today's content-based image retrieval (CBIR) systems. In this framework, images are represented as feature vectors, or points in a high-dimensional feature space. Images are considered similar if they are located "close" to each other in this high-dimensional space, according to some distance measure. Query processing is performed by finding k nearest images to a given example image in the feature space. A limitation of this approach is due to the weak correlation between retrieval objects and their appearance in images. A given object type can have various color patterns and diverse shapes from different view points. Consequently, the best-matching images are not necessarily located near each other in any single neighborhood in the feature space. A better technique

will combine the top ranked images from all the semantically relevant clusters in order to optimize the overall precision and recall. This is the idea behind our *Query Decomposition* (QD) approach introduced in this paper. In other words, a QD search is not based on the traditional k nearest neighbors in a single neighborhood, but rather finding the k best-matching images wherever they might be in the feature space.

1.1 Limitation of Traditional k -NN Model

Since it is inconvenient and inherently difficult to express queries in terms of low-level visual features such as color, texture, and shape, these feature values are generally "described" indirectly through the use of example images. This query model, called *Query by Example* (QBE), is used in essentially all CBIR systems today. To support QBE, images are characterized using their low-level visual features. Each image can be viewed as a *data point* in a multidimensional feature space, where each dimension corresponds to a visual characteristic. Similarly, a query image (that is, the example image) is mapped to a *query point* in that same multidimensional space. Query processing is then accomplished by finding the images corresponding to the k data points nearest to the query point. This popular query processing strategy is known as the *k-Nearest Neighbors* (k -NN) model.

The k -NN model confines the search result to a single neighborhood in the feature space. In practice, however, the k nearest neighbors may not be the real semantic neighbors. Cars, for example, come in many different shapes and colors; even the same car may look very different from different standpoints. These images would belong to different clusters in the feature space; and a simple retrieval of the k neighbors nearest the query point would not capture all the "car" images. The inherently weak association between the high-level semantic concepts human perceive in images and the low-level visual features used to characterize images causes *poor recall* in today's CBIR systems. Although this situation can be improved upon by conservatively retrieving more

images (i.e., employing a larger k value), this strategy would likely result in *poor precision* with many irrelevant images in the query result.

To further illustrate the limitation of the k -NN model, let us consider the images of “a white sedan” in a real database. This database is organized according to a 37-dimensional feature space (see Section 4 for additional details). We applied *Principle Component Analysis* (PCA) to project the data set onto a 3-dimensional orthogonal subspace, as shown in Figure 1. PCA is a powerful method for dimensionality reduction. The similarity of data vectors is preserved well in Figure 1. We observe in this 3-dimensional space four distinct clusters of the “white sedan” images, namely “side-view,” “front-view,” “back-view,” and “angle-view.” In this circumstance, retrieval techniques based on the k -NN model would not be able to find all these semantically identical images because they are not in any single k -NN neighborhood. This problem causes poor recall (i.e., failing to include many relevant images) in today’s CBIR systems. Although this situation can be improved by using a very large k value to include more “Sedan” images in the database, the larger k value would inevitably also cover many semantically irrelevant images (represented by small triangles in Figure 1, scattered in-between the four “white sedan” clusters) resulting in poor precision.

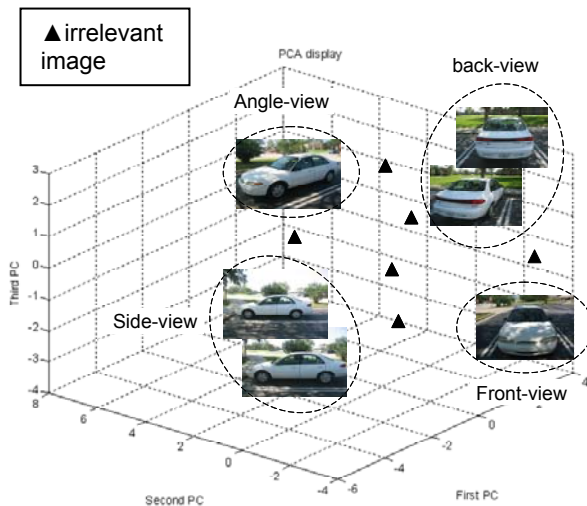


Figure 1. Four distinct “white sedan” clusters in a 3-dimensional feature space of an image database.

1.2 Multiple Neighborhoods Approach

To address the limitation of the k -NN model due to the confinement of the result images to a single neighborhood, we consider a *Query Decomposition* (QD) technique. The idea is to decompose an initial query into localized subqueries based on user relevance feedback. These subqueries are processed independently, and their local

results are merged to form the final result. This procedure is inspired by query processing techniques for distributed and parallel database management systems. It ensures that all semantically relevant data partitions are considered, even if they are positioned far apart in the feature space.

As an efficient way to facilitate query decomposition, we propose a *Relevance Feedback Support* (RFS) structure. This tree structure is constructed by hierarchically clustering the images in the database. With this facility, query decomposition becomes a multi-path tree descending procedure, in which the specific decomposition paths are influenced by user relevance feedback. The advantages of this computation technique are twofold. First, processing relevance feedback does not incur k -NN computation, and second, execution of the final subqueries requires only localized k -NN computation. We present experimental results later to show that the RFS approach is substantially less expensive than traditional relevance feedback processing based on a series of global k -NN computation.

1.3 Contributions

The primary contributions of this paper are as follows:

1. We introduce the Query Decomposition technique as a solution to address the limitation of the traditional single neighborhood k -NN model. The new method is capable of retrieving images with similar semantics but very different visual characteristics, i.e., relevant images that are scattered in the feature space.
2. We introduce the RFS structure to significantly reduce the cost of relevance feedback processing.

The remainder of this paper is organized as follows. Section 2 provides a survey of related work. In Section 3, we present the details of the Query Decomposition approach. The system prototype is introduced in Section 4 and we discuss our experimental study in Section 5. Finally, we conclude this study and discuss our future work in Section 6.

2. RELATED WORK

In trying to reduce the *semantic gap* [18] between low-level visual features and the high-level concepts conveyed by the query images, recent research has focused on retrieval techniques based upon *relevance feedback*. These schemes interact with the user. In each round, the user helps by identifying the relevant images within the set of images retrieved in the last round. The system then utilizes this feedback to modify the current query and thus to improve its retrieval results in the next round. This process is repeated until the user is satisfied with the

results. A query can be modified by a process called query point movement [7], the use of a multipoint query [13], or utilizing a more flexible query contour [9]. We briefly discuss these recent techniques in this section.

Query Point Movement [7]: A query is treated as a query point in the feature space. During every round of feedback, the centroid of the relevant images is used as the new query point in the next iteration. Furthermore, the distance function is weighted such that the query contour, enclosing the result images, is shaped in such a way as to optimize the trade-off between precision and recall.

Multipoint Query [13]: This technique offers another way to control the query contour. The relevant points according to user feedback are grouped into clusters, each represented by the data point nearest its centroid. These *representatives* are treated collectively as a multipoint query, and the distance of a data point to this multipoint query is defined as the weighed combination of the individual distances from the representatives, where the weight associated with each representative is proportional to the number of relevant images in its cluster. The effect is the expansion of the query contour, in accordance with the distribution of the centroids in the feature space, to capture more relevant images near by.

Qcluster [9]: The Multipoint Query likely includes irrelevant images when the desired images are in clusters significantly far apart in the feature space. In this circumstance, using separate contours can retrieve the desired images with greater precision. This is achieved in Qcluster by using a quadratic distance function to approximate any contour and to support disjunctive queries such that only images near the cluster representatives are considered relevant.

The design of the above and some other existing CBIR techniques, e.g., [2, 8, 10, 15, 20, 21], are based on the k -NN model: *objects of similar semantics look similar in many aspects*. This strategy fails when the image characterization and similarity measure do not follow perceptual characteristics. In this circumstance, images with similar semantics could be scattered in distinct neighborhoods in the feature space. Relevance feedback techniques based on k -NN would navigate along a single path to select, among these relevant neighborhoods, the one with the best tradeoff between precision and recall, instead of merging better matches from all these relevant neighborhoods in order to achieve a more optimal result.

To overcome the confinement of the traditional k -NN image retrieval, some top k retrieval approaches have been proposed recently. The result of such retrieval approaches are typically a ranked list of the *top k* objects that match the given attributes. The top k images can be acquired by merging information from multiple systems [3, 4], or using multiple viewpoints [5].

Merge information from Multiple Systems [3, 4]: This approach evaluates atomic queries (e.g., “find red

objects”) in separate subsystems consecutively. By using the first atomic query in the first subsystem, a result set is generated. Then the result set is evaluated in the second subsystem by applying the second atomic query (e.g., “find rectangular objects”). Consequently, the result set is ordered based on the query criteria in different subsystems. Finally, the top k images are selected from the overall ranked list as the result.

Multiple Viewpoints [5]: This recent technique improves on Qcluster by searching for relevant images using multiple queries; each considers only a subset of the visual features. As a result, images differing slightly in some visual aspect (e.g., a blue bus vs. a green bus) can still be found. In this model, multiple neighboring clusters can be returned as query results to achieve better retrieval effectiveness.

The above two techniques try to capture the user’s perception by selecting retrieval results from a ranked list. Though these top- k techniques can archive better performance than the standard k -NN techniques, they degrade significantly when the relevant images can not be covered by a single neighborhood. A neighborhood in this context may include more than one neighboring cluster, such as those identified by the Multiple Viewpoint approach.

3. Query Decomposition Approach

From the examples in Section 1, we can see that semantically identical images (e.g., “sedan”) may exhibit very different visual characteristics, and thus may not be projected to data points lying close together in the feature space. From this perspective, the problem of bridging the semantic gap becomes that of finding the semantically-related clusters, and more such clusters found results in improved precision and recall.

The goal of our Query Decomposition approach is to find all of these clusters. Existing relevance feedback methods are designed to avoid less relevant neighborhoods (local optimums) in order to find the most relevant one (the global optimum) in the feature space, where relevancy is defined in terms of precision and recall. In contrast, the Query Decomposition approach is designed to find the best matching images whether they are in a single or multiple relevant neighborhoods. We describe the details of the Query Decomposition technique in this section.

3.1 Relevance Feedback Support structure

The purpose of the *relevance feedback support* (RFS) structure is to make query decomposition more efficient. This structure is our implementation of the proposed QD model, with other implementation techniques also possible. A RFS structure is constructed in two stages. We discuss this procedure in the following.

Data Clustering: A hierarchical clustering technique, similar to the R*-tree [1], is used to organize the entire image database into a hierarchical tree structure. As each node in this hierarchy represents a cluster, we extend the original node structure of the R*-tree [1] to include also information to identify the corresponding representative images. Without loss of generality, we selected the R*-tree for our study because it is well known and has been widely used in practice. We could have also chosen other clustering techniques such as the *Hierarchical Generative Topographic Mapping* [12].

Representative Images Selection: A bottom-up representative-selection procedure is performed as follows:

- At the bottom level of the RFS structure, the images in each leaf node are clustered into subclusters by an unsupervised k -mean clustering algorithm. Here we use S_i to represent an image set in the i^{th} leaf node. By applying unsupervised k -mean clustering, S_i is clustered into subclusters: $\{C_{i1}, C_{i2}, C_{i3}, \dots, C_{ik}\}$. For each of the subclusters, one or more images nearest its center are selected as the *representative images* of S_i . We use R_i to denote the set of representative images of S_i .
- For each subsequent cluster in the upper level of the hierarchy, the representative images of all its child clusters are again aggregated and clustered by an unsupervised k -mean clustering algorithm. The images nearest these k -mean-cluster centers are selected as the representative images for the current node. For example, if a node has children S_1 through S_n , then R_1, R_2, \dots, R_n are aggregated and clustered, and representative images are selected from the new k -mean-clusters.

The number of representative images for each cluster is proportional to the number of images in that cluster. In other words, clusters in the upper levels of the RFS structure have more representative images than those in the lower levels of the hierarchy. The list of identifications of the representative images is stored in the modified R*-tree structure as part of the corresponding tree node. Thus, all of the information needed to support localized relevance feedback is self-contained in the RFS structure.

As our clustering method is based on visual features, a cluster may contain images with different semantics and therefore can have semantically different representative images. Each distinct category of objects, however, is likely to have one or more meaningful representative images somewhere in the RFS hierarchy. This form of clustering, although imperfect in terms of semantic clustering, is sufficient for the query decomposition process when combined with user relevance feedback. We will discuss this in more detail shortly. For convenience, in this paper we will refer to a cluster by the semantics of one of its representative images (e.g., a “desktop” cluster)

given that the context of the current discussion is clear (e.g., the query is “finding computers”).

3.2 Decomposition Technique

In the QD environment, two data clusters are *semantically related* if they have semantically similar subclusters. For instance, given the query “finding passenger cars,” a data cluster with a “sedan” subcluster is considered semantically related to another data cluster with a “coupe” subcluster. The idea of query decomposition is to split the initial query into more localized subqueries to explore distant relevant subspaces. Each subquery may be split again in the next iteration in order to cover more specific subclusters or to discard irrelevant ones. In this strategy, each subquery is processed independently to explore its own subspace. Initially, the semantically relevant clusters may also contain many irrelevant images. However, as the hierarchical decomposition makes progress, the localized subqueries become more refined, and ultimately the final subqueries identify only clusters with mostly relevant images.

To help the user formulate the initial query, the system displays some randomly selected representative images from the root node of the RFS structure. From these images, the user identifies the most relevant ones. If necessary, this process can be repeated with additional rounds of random displays in order to select additional relevant images. To process the initial query, the system determines the relevant subclusters to which these selected representative images correspond. If this process results in more than one relevant subcluster, the initial query is “split” into separate localized subqueries, one for each relevant subcluster. In other words, these subclusters are treated conceptually as separate databases, and localized relevance feedback is now performed separately on each of these individual subclusters.

We illustrate the query decomposition process with the example in Figure 2. We note that the images presented in this figure are not all of the representative images of the respective clusters. Similarly, only tree nodes relevant to the current discussion are shown in Figure 2. A more realistic diagram would include many more tree nodes and representative images. In this example, the RFS structure has three levels, with *Node 1* representing the root cluster of the entire image database. Suppose the user wants to find images of cars. She found, in the first feedback iteration, two relevant representative images in *Node 1* - a steamed car and a modern car. In response, the system determines that these two images came from *Node 2* and *Node 3* in *Level 2*, respectively; and the initial query is split into two subqueries as follows. Random representative images from *Node 2* and *Node 3* are presented to the user, at the beginning of the second iteration, for relevance feedback. The user can now find

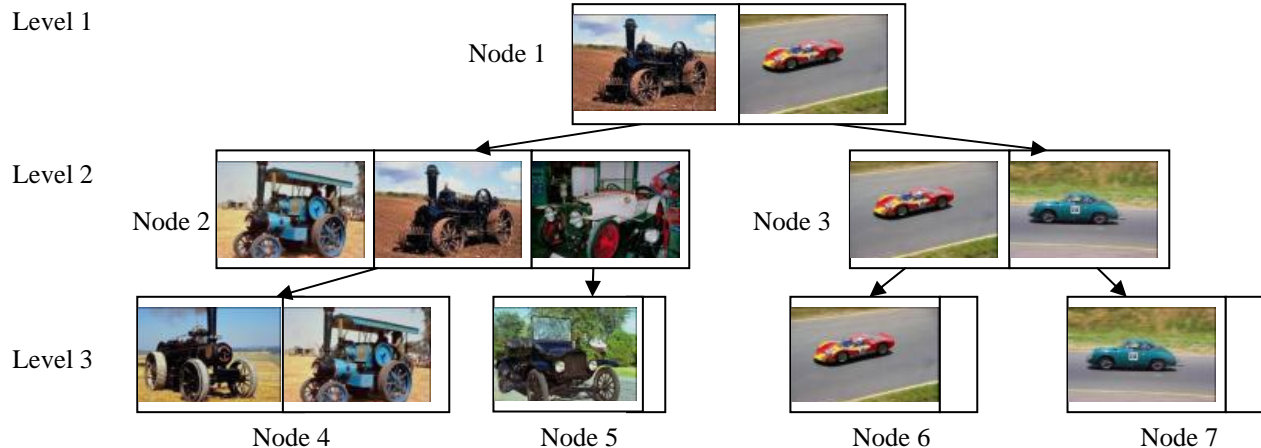


Figure 2. A 3-level RFS structure

more relevant images according to her interest, say, a steamed car and an antique car in *Node 2*, plus two modern cars of different colors in *Node 3*. With these new relevant images, the system recognizes in *Level 3* the corresponding relevant subclusters to be *Nodes 4, 5, 6, and 7*. Finally, the user identifies the relevant representative images in these subclusters as illustrated in Figure 2. These images are then used as the final four localized k -NN subqueries to search the corresponding subclusters independently; and their results are merged to form the overall result of car. We discuss the merging procedure in Section 3.4.

We observe from the above example that the user starts with an initial set of randomly selected representative images. As the user is subsequently presented with more relevant representative images from various subclusters, the user can indirectly refine the initial query by selecting more specific relevant images. In other words, the query is decomposed into multiple subqueries where, after several rounds, the subqueries better capture the user’s intent. We claim that because real-world objects can have very different appearances when represented in 2-dimensional images, multiple independent subqueries are better suited for their retrieval.

3.3 Localized k -NN Computation

To facilitate the query decomposition procedure, in each round of user relevance feedback, the system records each relevant image and its associated subcluster. In the final round, a localized multipoint query is computed for each subset of relevant images belonging to a given subcluster represented by a leaf node in the RFS structure. In the situation when some of these local query images are located near the boundary of the given leaf node, since some of their nearest images might actually reside in the sibling leaf nodes, the system instead computes the

multipoint query over the parent node, in order to also consider possibly relevant images from neighboring clusters. The detailed procedure is as follows.

To determine if a query image is near the boundary of its leaf node, we compute the ratio between the distance of this image from the center of the leaf node and the diagonal of the leaf node. If this ratio exceeds a predetermined threshold, the search area is expanded to the parent node. The same computation is repeated in the parent cluster and the expansion is continued, as necessary, to the higher levels in the hierarchy. The threshold used in this test can be determined empirically based on the database size and content as follows. If the images in the database are well clustered, we can set this threshold higher, which reduces the probability of expanding the search to the parent cluster. For our database with 15,000 images from about 150 categories, we set our threshold to 0.4.

3.4 Similarity Ranking

The Query Decomposition technique typically has results obtained from multiple localized multipoint k -NN queries. Any of the multipoint query processing techniques, discussed in Section 2, can be used to process these local queries. In our implementation, we compute the similarity score for each image, in a relevant subcluster, as the Euclidian distance between the image and the centroid of the local query points. To combine the results computed by the different localized k -NN subqueries, we include several top-ranked images from each set of local results. The number of result images selected from each local set is proportional to the number of query images the user has identified in the corresponding subcluster as relevant to the query. The rationale is that such a subcluster is more relevant to the intent of the query.

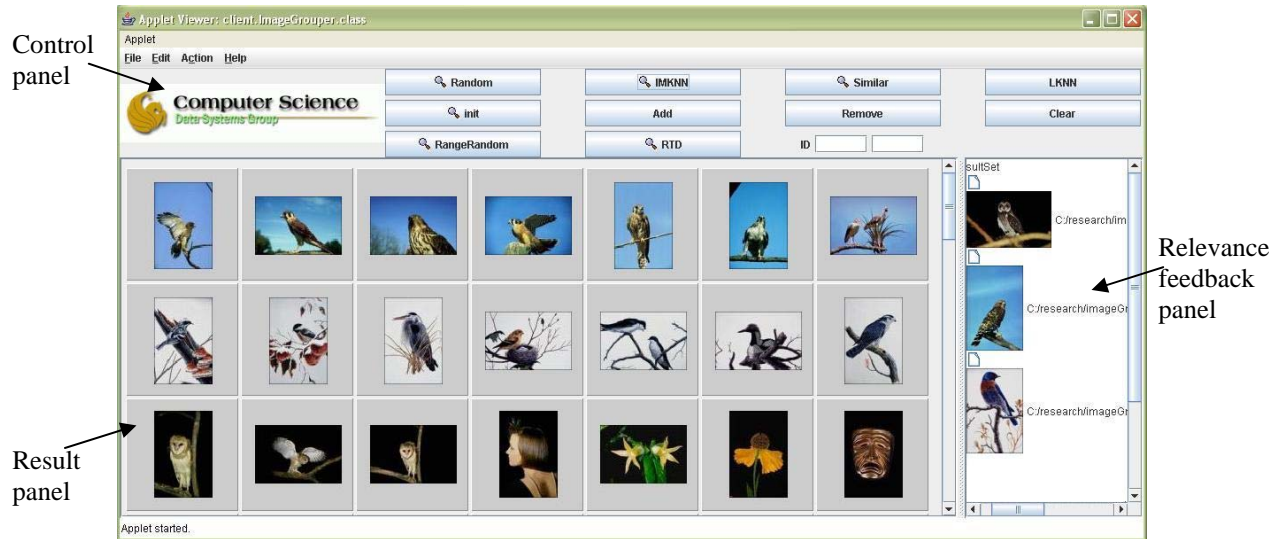


Figure 3. A screenshot of the prototype with query results displayed in the left pane and query images in the right. The localized subqueries are “eagle”, “sparrow”, and “owl”.

Various presentation techniques can be considered for displaying the final results on screen. In our current system, we present the result images in groups as follows. First, we compute the *ranking score* for each set of local results as the sum of the similarity scores of the individual result images contained therein. We present these result groups in the order of their ranking scores. Within each group, the images selected for the final result are displayed in the order of their individual similarity scores. An example screenshot is given in Figure 3, where the result images are presented in three groups. We will discuss the system prototype in details shortly.

This presentation scheme is used in this paper to illustrate the scattering nature of semantically similar images in the feature space and the effectiveness of the QD technique in capturing such distant clusters. In practice, the query decomposition process could be made more transparent to the user by displaying all the local result images in a single ranked list according to their individual similarity scores.

4. PROTOTYPE

We have constructed a CBIR system using the Query Decomposition model. This prototype system consists of the following four major components:

- **Feature Extraction Module:** This module extracts the low-level visual features for each image in the database. The system uses a 37-dimensional feature vector of numerical values, selected from three groups - *Color*, *Texture* and *Edge Structure*. More specifically, they consist of 9 color moment features

[17], 10 Wavelet-based texture features [16], and 18 edge-based structural features [22].

- **RFS Structure:** This access structure provides an efficient facility to support the new relevance feedback paradigm based on the query decomposition model. This subsystem includes a database builder for building the RFS structure and populating the image database. Our test database was populated with 15,000 images taken from the Corel database with a few hundreds new images we created to test the capability of proposed techniques. We specified the RFS-structure nodes to contain a maximum of 100 and minimum of 70 images each, resulting in a RFS structure that is 3 levels deep. By adjusting the aforementioned parameters we can change the breadth and depth of the RFS structure, however, the 3-level tree we used in our experiments is sufficient for research and illustrative purposes.
- **Query Processor:** This software module implements the Query Decomposition procedure presented in Section 2.
- **Presentation Manager:** This software subsystem, based on the ImageGrouper [11], provides the *graphical user interface* (GUI) for posting queries, providing relevance feedback, and receiving query results. A screenshot is shown in Figure 3, with current results displayed in the left panel and user relevance feedback accepted in the right panel. In each iteration, the user picks relevant images from the left panel and moves them to the right panel for feedback processing. Figure 3 shows the final query results for “bird”, which includes “eagle”, “sparrow”, and “owl”. The results are sorted within each of the subcategories and

then among the subcategories. We notice that the “owl” category is listed last because it contains more less relevant images affecting its overall ranking score.

In Figure 3, our system presents 21 images in the result panel at a time for user relevance feedback. During the first few rounds of feedback, since the nodes in the upper levels of the RFS tree represent large subsets of the database, the number of available representative images is much larger than 21. To make it convenient for the user to browse these candidates for interesting images to include in the query set, we provide a “Random” function in the GUI to allow the user to view 21 images at a time, randomly selected from the set of all candidate images. We note that the Query Decomposition model enables the user to provide localized and therefore more specific relevance feedback. To the best of our knowledge, this capability is a new contribution to CBIR research.

In our prototype system, 5% of the images are designated as representative images and is the only information we need to process relevance feedback. Since these images are substantially smaller than the total database size, in practice our software can be configured such that the RFS structure and relevance feedback mechanisms may run in the user computer. In this client-server configuration, the user would first identify the final query images on the client machine and only then submit them to the server to initiate the localized k -NN computations and final image retrieval. This capability is unique to the Query Decomposition approach and is not possible with any existing relevance feedback techniques; our system can be made to be highly scalable to support a very large user community.

5. Experimental Studies

We used the prototype system, presented in Section 4, for our performance study. Our objectives are twofold. First, we want to evaluate the capability of the Query Decomposition technique in handling scattered clusters of semantically similar images; and secondly, we would like to assess the efficiency of the proposed RFS structure as an implementation of the QD model. All experiments were performed on a 2.5-GHz Pentium IV-based computer with 1GBytes of RAM.

5.1 Datasets and Test Queries

Our test database includes 15,000 images taken from the Corel image database with a few hundreds new images we created to test the capability of the proposed techniques in handling the semantic gap in CBIR. We specified the RFS-structure nodes to contain a maximum of 100 and minimum of 70 images each, resulting in a RFS structure that is 3 levels deep. By adjusting the

aforementioned parameters we can change the breadth and depth of the RFS structure.

The Corel images have been classified into distinct categories by domain professionals. We also classified the additional images we created for this experimental study into the appropriate Corel categories. Since users search for images based on high level semantic concepts (as opposed to low level image features), we used the Corel category information as the ground truth in our experiments. To test our system’s capability of bridging the semantic gap we compare it with the *Multiple Viewpoints* (MV) approach using the queries listed in Table 1. This query set is carefully designed to investigate the impact on performance under both general (e.g., “finding computers”) and more specific (e.g., “finding laptop computers”) queries. Some of the test queries also allow us to compare the two techniques when the subject looks the same from most angles (e.g., mountain views).

5.2 Experimental Results

To facilitate a fair comparison, both the QD and MV techniques were evaluated using the same 11 queries listed in Table 1, and the same image dataset with the same feature vector. Both techniques retrieved the same number of images for each test query. For the MV approach, we combined the images returned by the four color channels to form the final result set. As relevance feedback is user subjective, we asked 20 students to test the systems by searching for the relevant images in the database. In addition, we evaluate the efficiency of our QD approach under various database sizes by using simulated queries to evaluate the computation times. We discuss our experimental results in the following subsections.

5.2.1 Retrieval Effectiveness

To evaluate the accuracy of the QD technique, we compared it with the MV approach in terms of precision and “ground truth inclusion ratio (GTIR)”. In our experiments, since the number of retrieved images equals the size of ground truth, the precision and recall have the same value and we report only precision in this paper. The concept of “ground truth inclusion ratio” is defined as follow:

$$GTIR = \frac{\text{Num of retrieved subconcept(s)}}{\text{Num of total subconcepts in ground truth}}$$

To illustrate the GTIR, we take the query “a person” from Table 1 as an example. In this case, the MV can only capture 1 out of 3 ground truth subcategories, thus

Table 1. Various Query Evaluation in QD & MV approaches

Query	MV		QD	
	Precision	GTIR	Precision	GTIR
A person (Hair-model, fitness, Kongfu)	0.25	0.33	0.81	1
Airplane (single, multiple)	0.21	1	0.85	1
Bird (eagle, owl, sparrow)	0.23	0.33	0.61	1
Car (modern sedan, antique car, steamed car)	0.35	0.33	0.85	1
Horse (polo, wild horse, race)	0.37	0.67	0.72	1
Mountain view (snow, with water)	0.38	1	0.46	1
Rose (yellow, red)	0.22	0.5	0.71	1
Water Sports (surfing, sailing)	0.11	0.5	0.44	1
Computer (server, desktop, laptop)	0.42	0.5	0.86	1
Personal computer (desktop, laptop)	0.44	0.5	0.69	1
Laptop (with clear background, with complicated background)	0.50	0.5	0.71	1
Average	0.32	0.56	0.70	1

resulting in GTIR=1/3; while QD can capture all three subcategories achieving a value of 1 for GTIR.

We compared the precision and GTIR of MV and QD in 3-round relevance feedback processing in Table 2. The results are based on the averages over the 11 test queries. The precision and GTIR at the end of each round of the 3-round relevance feedback are given in Table 2. We note that the QD technique does not actually commit any k -NN computation until Round 3, and consequently does not have any precision measurements for Rounds 1 and 2. In Round 3, the QD technique determines the result images by selecting top-ranked images from each of the localized k -NN queries, as described in Section 3.4. We observe in Table 2 that the QD technique performs significantly better than MV. Furthermore, the precision of MV cannot improve by much after the second round of relevance feedback. Due to the confinement of the traditional k -NN model, the GTIR of MV can not be improved after Round 2 with further relevance feedback.

Table 2. Quality Comparison

Feed-back Round	Techniques			
	MV		QD	
	Precision	GTIR	Precision	GTIR
1	0.1	0.51	n/a	0.695
2	0.30	0.56	n/a	0.907
3	0.32	0.56	0.70	1

We need to point out here that one should not interpret the experimental results as evidence for choosing QD over the MV technique. They are designed to address two orthogonal issues. While QD is a framework for decomposing queries into localized k -NN subqueries, MV is a technique to compute a given k -NN query. As we have mentioned previously, we could have used MV for

the computation of our localized k -NN subqueries. The lesser performance of MV, observed in this paper, is due to the limitation of the k -NN model (confining the search area to single, as opposed to multiple distant clusters), and not a drawback of the MV technique itself. In fact, we selected MV for our study because it performs very well in identifying the better cluster among multiple semantically relevant clusters.

The top eight images retrieved for “portable computer” are presented in Figures 4 and 5 for the MV and QD techniques, respectively. Figure 4 shows that the result images of MV only include one type of the “portable computer”: “laptop with bright background”. In contrast, we observe in Figure 5 that the QD approach is able to capture all semantically relevant neighborhoods of “portable computer” scattered in the feature space.

We also present the top 16 images for “personal computer” from MV in Figure 6 and QD in Figure 7. The “personal computer” includes “desktop” and “portable computer”. Under the “personal computer” concept, there are two subcategories: “computer on a table” and “computer on the floor.” Again, the MV approach can only find one neighborhood, and the feature space that MV covered by using multiple channels can only include one type of the computer in the ground truth. The set of “desktop on the floor” images and the set of “desktop on a table” images are too far apart in the feature space for the MV technique to capture both.

In Figure 8, we show the top 24 images of “computer” returned by MV. The top 24 images retrieved by QD, in this test, are shown in Figure 9. In the QD approach, both the “Sun workstation” and “HP workstation” subcategories are covered, in contrast, MV only finds one subcategory. We notice that with the QD approach, all the relevant ground truth subcategories are covered, while the MV approach can only catch one of the subcategories. These experimental results confirm our assertion that the

traditional k -NN model confines the query results of MV to a single neighborhood in the feature space.

The average precision and GTIR for the 11 test queries are presented in Table 1 for both MV and QD. For some cases such as “airplane,” the images with single airplane and images with multiple airplanes have similar feature characteristics, i.e., the background is clear. Under this circumstance, MV can also catch all the ground truth subcategories. However, QD can perform better in terms of precision because the MV approach brings some unrelated images in the color-negative, black-white, and black-white negative channels. For the case of “mountain view,” the QD approach is not significantly better than the MV approach for the reason that the “mountain” images are faraway views and due to the feature we selected, the complexity of “mountain view” images bring in a lot of unrelated images. Nevertheless, the QD approach could pick up both the “snow mountain” images and the “mountain with water” images by exploring the two corresponding subclusters independently giving it a small performance edge over MV in this case.

Finally, the average results over all 11 test queries are presented at the bottom of Table 1. We observe that for all the cases, the QD approach can catch all the semantic subcategories allowing it to outperform the MV approach in terms of both precision and GTIR.

5.2.2 Computation Efficiency

Another advantage of the QD approach is the efficiency attained by using the RFS structure. With this structure, the expensive k -NN computation in each round of relevance feedback is avoided. We used simulated queries to evaluate the computation time of our system. We randomly generated 100 initial queries and evaluated their average query processing time (Figure 10), as well as the average relevance feedback processing time (Figure 11) for a single round. For each query, we performed two rounds of relevance feedback in addition to the initial query processing and the final localized k -NN computation. From Figures 10 and 11, we can see that the overall query processing time and the average iteration processing time increase linearly with larger database sizes. The results indicate that the QD approach is very time efficient, suitable for very large databases with many concurrent users.

The efficiency of the Query Decomposition approach can also be attributed to low disk utilization. Since the information required for relevance feedback processing is stored in the RFS structure, this technique needs to access only one tree node for a given representative image when marked relevant by the user. This I/O cost is significantly less when multiple relevant representative images are chosen from the same cluster, and therefore share the same node in the RFS structure. The number of disk access for each localized k -NN computation is usually

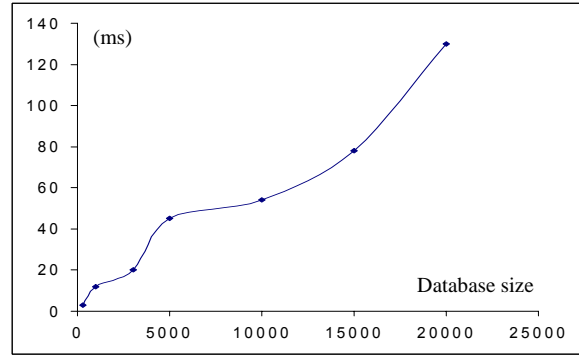


Figure 10. Overall query processing time for different database sizes

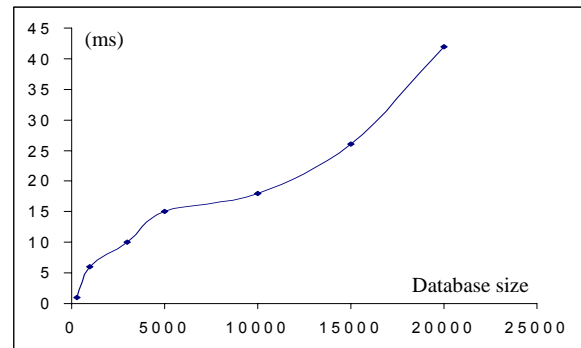


Figure 11. Average iteration processing time for different database sizes

one, unless the query image is located far away from the centroid of the cluster; in which case, the searching of parent nodes incurs additional disk accesses. Nevertheless, processing of all the localized k -NN subqueries need to access only a few neighborhoods in the feature space.

6. Conclusions and Future work

The three primary contributions of this paper are as follows:

Better Query Result: The traditional k -NN image retrieval framework confines query results to a single neighborhood in the feature space. The QD approach considers multiple neighborhoods, and therefore can better address the inherent challenge in image retrieval, namely, that objects of similar semantics can look quite different and may be scattered in the feature space.

More Efficient Query Processing: Unlike traditional relevance feedback processing which performs k -NN computation on the entire database in each round of relevance feedback, the QD approach only performs localized k -NN computation on very small subclusters in the final round of the relevance feedback process.

More Scalable: Since the relevance feedback mechanism of the QD approach relies on only a tiny

fraction of the database, relevance feedback processing can run on the client machines leaving the server mainly to retrieve the final query results for the small localized queries. This highly scalable solution is unique to the QD technique.

Future research may investigate more effective and intuitive user interfaces for the Query Decomposition environment. We can also investigate ways to leverage existing advanced techniques such as allowing the user to define the importance of specific image features, e.g., the user may define color as the most important feature in the retrieval procedure [6]. Another possible extension is to ask the user to draw a contour around the object of interest in the example images [19], thus decreasing unintended noise in the query formulation. Our system may also be extended to support video retrieval. Also conceivable is the development of an image search engine for the Web based upon the QD idea.

REFERENCES

- [1] N. Beckmann, H-P. Kriegel, R. Schneider, and B. Seeger, The R*-tree: an efficient and robust access method for points and rectangles. *In Proc. Of ACM SIGMOD*, pp. 322-331, 1990.
- [2] A. D. Bimbo, P. Pala, J. Assfalg, Three-dimensional interfaces for querying by example in content-based image retrieval. *IEEE Trans. on visualization and computer graphics*, 8(4): 305-318, 2002
- [3] R. Fagin, Combining fuzzy information from multiple systems. *In Proc. of PODS*, pp. 216-226, 1996.
- [4] R. Fagin, Fuzzy queries in multimedia database systems. *In Proc. of PODS*, pp. 1-10, 1998.
- [5] J. French and X-Y. Jin, An empirical investigation of the scalability of a multiple viewpoint CBIR system, *In Proc. of CIVR*, pp. 252-260, 2004.
- [6] T. Gevers, Color in image search engine. *Survey on color for image retrieval from Multimedia Search*, ed. M. Lew, Springer Verlag, 2001.
- [7] Y. Ishikawa, R. Subramanya, and C. Faloutsos, MindReader: Querying databases through multiple examples. *In Proc. of 24th VLDB*, pp. 218-227, 1998.
- [8] H. Koo and N. I. Cho, A relevance feedback algorithm based on the clustering and parzen window. *In Proc. of IEEE ICIP*, 3(II): 551-554, 2003.
- [9] D. H. Kim and C. W. Chung, Qcluster: relevance feedback using adaptive clustering for content-based image retrieval. *In Proc. of ACM SIGMOD*, pp. 599-610, 2003.
- [10] D. Liu, K. A. Hua, K. Vu, and N. Yu, Efficient target search with relevance feedback for large CBIR Systems. *In Proc. of the 21st ACM Symposium on Applied Computing*, 2006
- [11] M. Nakazato, L. Manola and T. S. Huang, ImageGrouper: a group-oriented user interface for content-based image retrieval and digital image arrangement. *Journal of Visual Languages and Computing*, 14 (4): 363-386, 2003.
- [12] I. T. Nabney, Y. Sun, P. Tino and A. Kaban, Semisupervised learning of hierarchical latent trait models for data visualization. *IEEE Trans. on Knowledge and data engineering*, 17(3): 384-400, 2005.
- [13] K. Porkaew, K. Chakrabarti, and S. Mehrotra, Query refinement for multimedia similarity retrieval in MARS. *In Proc. of ACM MM 1999*, pp. 235-238, 1999.
- [14] C. J. Van Rijsbergen, *Information Retrieval* (2nd edition). Butterworth-Heinemann, Newton, MA, USA, 1979.
- [15] Y. Rui, T. Huang, and S. Mehrotra, Content-based image retrieval with relevance feedback in MARS. *In Proc. of the IEEE international conference on image processing*, pp. 815-818, Oct. 1997.
- [16] J. R. Smith and S-F. Chang, Transform features for texture classification and discrimination in large image databases. *In Proc. of IEEE ICIP*, Vol. 3, pp. 407-411, 1994.
- [17] M. Sticker and M. Orenco, Similarity of Color Images. *In Proc. of SPIE*, Vol. 2420, pp. 381-392, 1995.
- [18] A. Smeulders, M. Worring, S. Santini, A. Gupta and R. Jain, Content-Based Image Retrieval at the end of the early years. *IEEE Trans. On PAMI*, 22(12): 1349-1380, 2000
- [19] K. Vu, K. A. Hua, and N. Jiang, Improving Image Retrieval Effectiveness in Query-By-Example Environment. *In Proc. of the 2003 ACM symposium on Applied computing*, pp. 774-781, 2003.
- [20] J. Yoon and N. Jayant, Relevance feedback for semantics based image retrieval. *In Proc. of IEEE ICIP*, pp. 42-45. 2001.
- [21] R. Zhao and W. I. Grosky, Narrowing the semantic gap – improved text-based web document retrieval using visual features. *IEEE Trans. on Multimedia*, 4(2): 189-200, 2002.
- [22] X. S. Zhou and T. S. Huang, Edge-based structural feature for content-based image retrieval. *Pattern Recognition Letters, Special issue on Image and Video Indexing*, 457-468, 2000



Figure 4. “Portable computer” from MV



Figure 5. “Portable computer” from QD



Figure 6. "Personal computer" from MV



Figure 7. "Personal computer" from QD

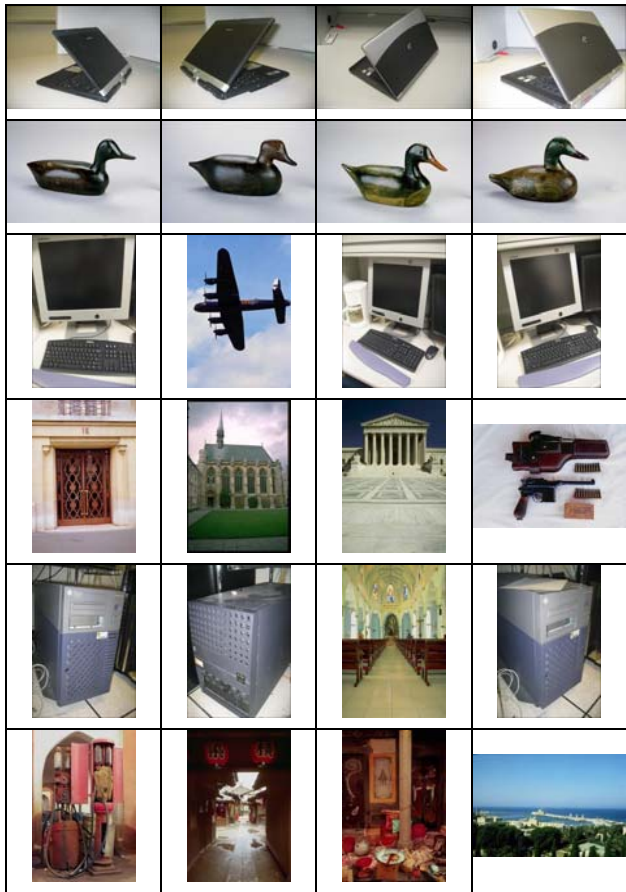


Figure 8. "Computer" from MV

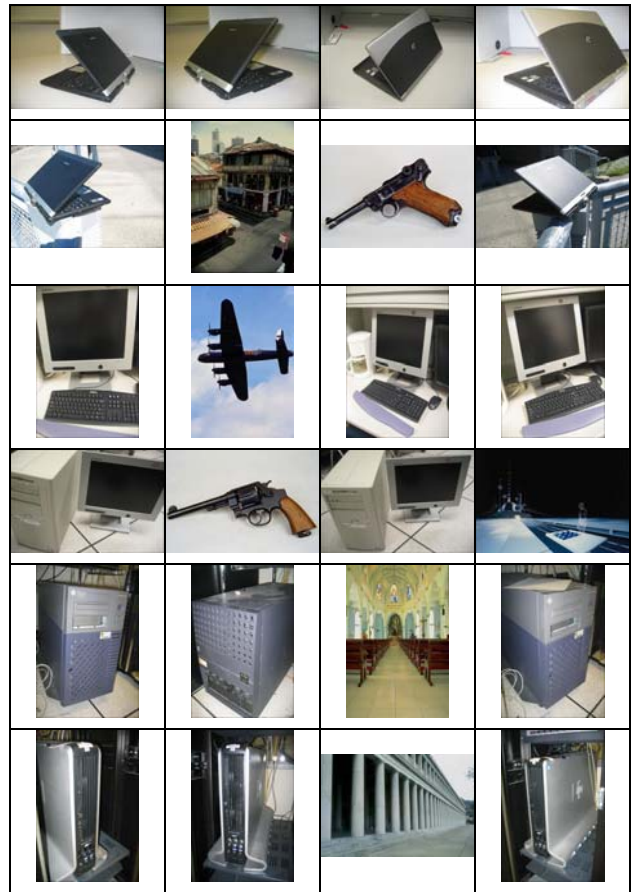


Figure 9. "Computer" from QD