Combining Supervised Learning with Color Correlograms for Content-Based Image Retrieval

Jing Huang * S Ravi Kumar † Mandar Mitra‡

Department of Computer Science
Cornell University
Ithaca, NY 14853.

ABSTRACT

The paper addresses how relevance feedback can be used to improve the performance of content-based image retrieval. We present two supervised learning methods: *learning the query* and *learning the metric*. We combine the learning methods with the recently proposed *color correlograms* for image indexing/retrieval. Our results on a large image database of over $20,000$ images suggest that these learning methods are quite effective for content-based image retrieval.

INTRODUCTION

The recent explosion in Internet usage and the rapidly growing availability of multimedia resources on the World-Wide Web has created a demand for effective and flexible techniques for automatic image retrieval and video browsing [3, 8, 11, 1, 10]. Users need high-quality image retrieval (IR) systems in order to find useful images from the masses of electronically available digital image data. In a typical IR system, a user poses a query by providing an existing image (or creating one by drawing), and the system retrieves other "similar" images from the image database. Content-based video browsing tools also provide users with similar capabilities — a user provides an interesting frame as a query, and the system retrieves other similar frames from a video sequence.

*Supported by DARPA under a contract monitored by ARL. Email: huang@cs.cornell.edu.

†Supported by the ONR Young Investigator Award N00014-93-1-0590, the NSF grant DMI-91157199, and the career grant CCR-9624552. Email: ravi@cs.cornell.edu.

‡Supported by the NSF grant IRI-9624639. Email: mitra@cs.cornell.edu.

The problem of content-based image retrieval has been widely studied [3, 8, 11, 1, 10]. The color correlogram was recently proposed as an image feature vector for image indexing and retrieval in [5]. It was shown to be robust in tolerating large changes in the appearance of a scene caused by changes in viewing positions, changes in the background scene, partial occlusions, camera zoom that causes radical changes in shape, etc. The correlogram method proved to be very effective and efficient for content-based image retrieval. The color correlogram was also shown to be a generic tool for spatial color indexing [6] that could be applied to problems like image subregion querying, object searching, image localization, tracking, video cut detection, etc. The results in [6] indicate that the correlogram might be a better indexing tool than the traditional histogram.

It is a fact of life that no tool for image retrieval can be entirely fool-proof. Therefore, for any query image, not all the images labeled "relevant" by any retrieval method are actually relevant. It is reasonable in such situations to enlist the services of the user by requesting for some relevance feedback. For instance, in practical situations like searching for a particular image on the world-wide web, the system can retrieve a list of images that it considers relevant to the query image. Then, the user can be prompted to mark each retrieved image as "relevant" or "irrelevant". This information can be used to refine the search, thus potentially improving the quality of retrieval.

In this study, we investigate the use of information provided interactively by a user to improve the performance of correlograms. We outline two schemes that use feedback information (in the form of labeled examples). The first scheme is based on the spatial locality of feature vectors corresponding to similar images. Learning is effected by modifying the query vector to incorporate the positive examples. The second scheme is based on distortion of the feature space. By using a weighted metric, it is possible to selectively enhance (resp. suppress) the role of appropriate dimensions to retrieve images of the kind termed positive (resp. negative).

First, some related work is discussed. After defining the image retrieval problem, we briefly review the correlogram. We then present two learning methods for using feedback information. The experimental framework, results, and conclusions are presented next.

RELATED WORK

In this section we discuss some related work in the areas of content-based image retrieval and the use of learning to improve the performance of image retrieval.

The color histogram has been shown to be efficient and effective in content-based image retrieval [3, 8, 11]. As a feature vector for image indexing [16, 3, 8, 11], the color histogram is extremely easy to compute and insensitive to small changes in viewing positions and partial occlusion. Due to the lack of any spatial information, however, the histogram method is liable to false positives. This problem is especially acute for very large image databases. Moreover, the histogram is not robust to large appearance changes.

Several recently proposed schemes incorporate spatial information about colors to improve upon the histogram method [14, 13, 15, 9, 10, 5]. One approach [14, 15] divides images into subregions and imposes positional constraints on the image comparison (*image partitioning*). Another approach [9, 10] augments histograms with local spatial properties (*histogram refinement*). The *color correlogram* [5], makes use of spatial correlation between pairs of colors. The following briefly describes some of these methods.

Smith and Chang [14] partition an image into binary color sets. They first select all colors that are "sufficiently" present in a region. The colors for a region are represented by a binary color set that is computed using histogram back-projection [16]. The binary color sets and their location information constitute the feature for an image. This feature can be used to process *region queries* that our method cannot handle, for example, a query where the sun is setting in the upper left part of the image. Results from a 3100 image database suggest very good retrieval performance.

Stricker and Dimai [15] divide an image into five fixed overlapping regions and extract the first three color moments of each region to form a feature vector for the image. The storage requirements for this method are very low. The use of overlapping regions makes the feature vectors relatively insensitive to small rotations or translations. Their database contains over 11,000 images, but the performance of their method is only illustrated on 3 example queries.

Pass and Zabih [9, 10] use another approach. They partition histogram bins by the spatial coherence of pixels. A pixel is coherent if it is a part of some "sizable" similar-colored region, and incoherent otherwise. A color coherence vector (CCV) represents this classification for each color in the image. CCVs are fast to compute and appear to perform better than histograms. The notion of CCV is also extended in [10], by using additional feature(s) to further refine the CCV-refined histogram. One such extension uses the center of the image (the centermost 75% of the pixels are defined as the "center") as the additional feature. The enhanced CCV is called CCV with *successive refinement* (CCV/C) and performs much better than the color histogram on a 14553 image database.

Since the image partitioning approach depends on pixel position, it is unlikely to tolerate large image appearance changes. The same problem occurs in the histogram refinement method, which depends on local properties to further refine color buckets in histograms. The correlogram method, however, takes into account the local spatial correlation between colors as well as the global distribution of this spatial correlation. Correlograms have been shown to be robust to large image appearance changes and to be more effective in image retrieval than histograms. Furthermore, correlograms can be compressed to the same size as histograms [6], without loss in effectiveness.

Some work has been done on how to improve the performance of image retrieval by learning from user feedback. In PicHunter [1], relevance feedback from the user is used to search the target. This approach is based on a Bayesian framework that incorporates an explicit model of the user selection process. They show that the feedback system works much better than random chance in a queryless setting. They also conduct a user-study of the performance of their system.

Minka and Picard [7] introduce a learning component in their system by using positive and negative examples which lets the system choose image groupings within and across images based on color and texture cues. The user-chosen region defines the positive or negative examples which the system tries to generalize and label various parts of the scene. In [2], positive and negative examples are used to first construct histograms. Then, a scoring function is constructed from a comparison of the extent of overlap between the histograms. This scoring function is aimed to best partition the examples and counter-examples.

THE IMAGE RETRIEVAL PROBLEM

The image retrieval problem is the following: let $\mathcal{S}$ be an image database and $\mathcal{Q}$ be the query image. Obtain a permutation of the images in $\mathcal{S}$ based on $\mathcal{Q}$, i.e., assign $\mathrm{rank}(\mathcal{I}) \in \{1, \ldots, |\mathcal{S}|\}$ for each $\mathcal{I} \in \mathcal{S}$, using some notion of similarity to $\mathcal{Q}$. This problem is usually solved by sorting the images $\mathcal{I} \in \mathcal{S}$ according to $|f(\mathcal{I}) - f(\mathcal{Q})|$, where $f(\cdot)$ is a function computing feature vectors of images and

$|\cdot|_f$ is some distance measure defined on feature vectors.

Performance Measure. Let $\{\mathcal{Q}_1, \ldots, \mathcal{Q}_q\}$ be the set of query images. For a query $\mathcal{Q}$, let $\mathcal{A}$ be the set of images that are similar to $\mathcal{Q}$. $\mathcal{A}$ is ordered by the metric on the feature vector. We use the following performance measures:

1. ***Scope vs. Recall:*** For a scope s, we count $|\{\mathcal{I} \in \mathcal{A} \mid \operatorname{rank}(\mathcal{I}) \leq s\}|$, i.e., the number of relevant images ranked below s.
2. ***Average Precision:*** Let w be fixed *a priori*. Define $\mathcal{A}^{(w)} = \{\mathcal{I} \in \mathcal{A} \mid \operatorname{rank}(\mathcal{I}) \leq w\}$. Note that $\mathcal{A}^{(w)}$ is still ordered. Then, the average precision is given by $(1/|\mathcal{A}^{(w)}|) \sum_{\mathcal{I}_i \in \mathcal{A}^{(w)}} i / \operatorname{rank}(\mathcal{I}_i)$.

For a given scope, the higher the recall, the better. A higher value of average precision also indicates better performance (the maximum value of average precision is one). Note that these definitions of the terms recall and average precision are slightly different from the standard ones. The above measures can be averaged over all the queries $\mathcal{Q}_1, \ldots, \mathcal{Q}_q$ to get a fair estimation of the performance of a method.

In the case when the answer set $\mathcal{A}$ is a singleton (i.e., there is a unique correct answer), we use the following ranking measures:

1. ***r-measure*** of a method which sums up over all queries, the rank of the correct answer, i.e., $\sum_{i=1}^{q} \operatorname{rank}(\mathcal{A}_i)$. We also use the average r-measure which is the r-measure divided by the number of queries q.
2. ***p_1-measure*** of a method which is $\sum_{i=1}^{q} 1/\operatorname{rank}(\mathcal{A}_i)$, i.e., the sum (over all queries) of the precision at recall equal to 1. The average p_1-measure is the p_1-measure divided by q.

Images ranked at the top contribute more to the p_1-measure. Note that a method is good if it has a low r-measure and a high p_1-measure.

Object Searching. The object searching problem is the following: given as input a subregion $\mathcal{Q}$ of an image $\mathcal{I}$ and an image set $\mathcal{S}$, retrieve from $\mathcal{S}$ those images $\mathcal{Q}'$ in which the query $\mathcal{Q}$ appears according to human perception (denoted $\mathcal{Q}' \subseteq \mathcal{Q}$). The set of images might consist of a database of still images, or videos, or some combination of both. The problem is made even more difficult than image retrieval by a wide variety of effects that cause the same object to appear differently in different images (changing viewpoint, camera noise and occlusion are examples of such causes). The object searching problem arises in image retrieval and in video browsing. For example, a user might wish to find pictures in which a given object appears, or scenes in a video with a given appearance of a person.

BRIEF REVIEW OF THE CORRELOGRAM

A color correlogram[1] (henceforth correlogram) expresses how the spatial correlation of pairs of colors changes with distance. Informally, a correlogram for an image is a table indexed by color pairs, where the k-th entry for row $\langle i, j\rangle$ specifies the probability of finding a pixel of color j at a distance k from a pixel of color i in this image. Here k is chosen from a set of distance values D.

Notation. Let $\mathcal{I}$ be an $n_1 \times n_2$ image. The colors in $\mathcal{I}$ are quantized into m colors $c_1, \ldots, c_m$. For a pixel $p = (x, y) \in \mathcal{I}$, let $\mathcal{I}(p)$ denote its color. Let $\mathcal{I}_c \triangleq \{p \mid \mathcal{I}(p) = c\}$. Thus, the notation $p \in \mathcal{I}_c$ is synonymous with $p \in \mathcal{I}, \mathcal{I}(p) = c$.

For convenience, we use the L_∞-norm to measure the distance between pixels. We denote the set $\{1, 2, \ldots, n\}$ by $[n]$. The size of $\mathcal{I}$ is denoted $|\mathcal{I}| = n_1 n_2$.

Definitions. The *color histogram* (henceforth histogram) h of $\mathcal{I}$ is defined for $i \in [m]$ by

$$h_{c_i}(\mathcal{I}) \triangleq \Pr_{p \in \mathcal{I}}[p \in \mathcal{I}_{c_i}]. \tag{1}$$

$h_{c_i}(\mathcal{I})$ thus gives for any pixel in $\mathcal{I}$, the probability that the color of the pixel is c_i. Given the count $H_{c_i}(\mathcal{I}) \triangleq |\{p \in \mathcal{I}_{c_i}\}|$, it follows that $h_{c_i}(\mathcal{I}) = H_{c_i}(\mathcal{I})/(n_1 n_2)$.

Let a *distance set* D be fixed *a priori* (e.g., $D \subseteq [\min\{n_1, n_2\}]$). Let $d = |D|$. Then, the *correlogram* of $\mathcal{I}$ is defined for $i, j \in [m], k \in D$ as

$$\gamma_{c_i, c_j}^{(k)}(\mathcal{I}) \triangleq \Pr_{p_1 \in \mathcal{I}_{c_i}, p_2 \in \mathcal{I}}[p_2 \in \mathcal{I}_{c_j} \mid |p_1 - p_2| = k]. \tag{2}$$

Given any pixel of color c_i in the image, $\gamma_{c_i, c_j}^{(k)}$ gives the probability that a pixel at a distance k from the given pixel is of color c_j. Note that the size of the correlogram is $m^2 d$. If we define the following count (similar to the *cooccurrence matrix* defined in [4] for texture analysis of gray images)

$$\Gamma_{c_i, c_j}^{(k)}(\mathcal{I}) \triangleq \left|\{p_1 \in \mathcal{I}_{c_i}, p_2 \in \mathcal{I}_{c_j} \mid |p_1 - p_2| = k\}\right|, \tag{3}$$

then

$$\gamma_{c_i, c_j}^{(k)}(\mathcal{I}) = \frac{\Gamma_{c_i, c_j}^{(k)}(\mathcal{I})}{H_{c_i}(\mathcal{I}) \cdot 8k}. \tag{4}$$

The factor $8k$ is due to the properties of L_∞-norm used to compute distance between pixels. The *autocorrelogram* of $\mathcal{I}$

[1] The term "correlogram" is adapted from spatial data analysis: "correlograms are graphs (or tables) that show how spatial autocorrelation changes with distance." [17]

captures the spatial correlation between identical colors only and is defined by

$$\alpha_c^{(k)}(\mathcal{I}) \triangleq \gamma_{c,c}^{(k)}(\mathcal{I}). \tag{5}$$

This information is a subset of the correlogram and requires only md space. Since local correlations between colors are more significant than global correlations in an image, a small value of d is sufficient to capture the spatial correlation. An efficient algorithm for computing correlograms can be found in [5].

Correlograms, like histograms, can be viewed as feature vectors of images. We use $\gamma(\mathcal{I})$ (resp. $\alpha(\mathcal{I})$) to denote the correlogram (resp. autocorrelogram) of $\mathcal{I}$, treated as vectors in m^2d (resp, md) dimensional space. Thus, the notation $\alpha(\mathcal{I}) + \alpha(\mathcal{J}), \alpha(\mathcal{I}) - \alpha(\mathcal{J})$ for two images $\mathcal{I}, \mathcal{J}$ has the usual interpretation.

The dissimilarity between two images is measured by the distance between the corresponding feature vectors. The L_1 and L_2 distance measures are commonly used to compare two feature vectors. In practice, the L_1 distance measure performs better than the L_2 distance measure because the former is statistically more robust to outliers [12]. We will use the L_1 distance measure for comparing histograms and correlograms because it is simple and robust:

$$|\mathcal{I} - \mathcal{I}'|_{h,L_1} \triangleq \sum_{i \in [m]} |h_{c_i}(\mathcal{I}) - h_{c_i}(\mathcal{I}')| \tag{6}$$

$$|\mathcal{I} - \mathcal{I}'|_{\gamma,L_1} \triangleq \sum_{i,j \in [m], k \in D} |\gamma_{c_i,c_j}^{(k)}(\mathcal{I}) - \gamma_{c_i,c_j}^{(k)}(\mathcal{I}')| \tag{7}$$

Correlograms successfully achieve a balance between robustness to differences between similar images and the ability to eliminate false positives. This can be explained as follows: when D is chosen to be a set of small distances, correlograms take into account the local color spatial correlation as well as the global distribution of this spatial correlation.

While any scheme that is based on purely local properties, such as pixel position, gradient direction, is likely to be sensitive to large appearance changes, correlograms are more stable to these changes; while any scheme that is based on purely global properties, such as histograms, is susceptible to false positive matches, correlograms are largely successful in minimizing false matches by effectively using spatial information [5]. Moreover, without significant loss of performance, the space requirements for correlograms can also be made exactly the same as for histograms [6] (banded correlograms). Correlograms thus prove to be both efficient and effective for content-based image retrieval from a large image database.

Correlograms were also used to address the object searching problem [6]. The notion of intersection of a model with an image was extended to correlograms and the resulting method was shown to perform better than methods that use the traditional histogram intersection scheme [16].

SOME LEARNING METHODS

In this section, we look at the issue of using relevance feedback from the user to improve the performance of the correlogram. In a real-life scenario, the user can provide this information with very little effort. Our goal is to make use of this information to improve search results, so that the user has an incentive to make this small additional effort.

Let $\mathcal{F} = \mathcal{F}^+ \cup \mathcal{F}^-$ be the set of images for which the user provides relevance judgments, where $\mathcal{F}^+ = \{\mathcal{F}_1^+, \ldots, \mathcal{F}_{|\mathcal{F}^+|}^+\}$ (resp. $\mathcal{F}^- = \{\mathcal{F}_1^-, \ldots, \mathcal{F}_{|\mathcal{F}^-|}^-\}$) is the set of positive (resp. negative) examples.

Learning the Query Vector. The motivation for this method (called QLEARN) comes from the fact that feature vectors of similar images are usually geometrically close to each other in the feature space. Hence, a natural approach is to modify the feature vector of the query image $\mathcal{Q}$ to get a new feature vector. One such approach involves taking a weighted average of the positive examples together with the query image. The modified feature vector is then given by

$$(1 - \beta) \cdot \alpha(\mathcal{Q}) + \beta \cdot \frac{1}{|\mathcal{F}^+|} \sum_{i=1}^{|\mathcal{F}^+|} \alpha(\mathcal{F}_i^+)$$

Here, β controls the contribution of the feedback images. Note that β cannot be too large because there is the danger of "losing" the importance of the original query.

It is not obvious how to augment this method to handle images in $\mathcal{F}^-$. In the feature space, feature vectors of images in $\mathcal{F}^+$ are usually roughly clustered. The modification described above can be interpreted as moving the query vector closer to the cluster of positive examples. The corresponding way to use negative examples would be to move the query away from them. Top-ranked negative examples do not exhibit a similar clustering, however, and are usually scattered in a close neighborhood of the query. Thus, the notion of "moving away" from the negative examples is not well-defined.

Learning the Metric. The main idea for this method (called WLEARN) is to use a weighted L_1^ω metric instead of the usual L_1 metric to compute the distance between feature vectors. In this manner, we would be able to enhance the importance of those dimensions that "assist" in getting the similar images and diminish the importance of those dimensions that "hinder" this process.

More formally, the weighted L_1^ω metric for a given real weight vector $\omega \in R^{m \times d}$ is given by

$$|\mathcal{I} - \mathcal{I}'|_{\alpha, L_1^\omega} \triangleq \sum_{i \in [m], k \in D} \omega_{i,k} \cdot |\alpha_{c_i}^{(k)}(\mathcal{I}) - \alpha_{c_i}^{(k)}(\mathcal{I}')|$$

If $D^+ \subseteq [m] \times [d]$ is the set of dimensions that are deemed important, then using some weight update scheme, we can increase those weights $\omega_{i,j}$ for which $(i,j) \in D^+$. In an analogous manner, if $D^- = ([m] \times [d]) \backslash D^+$, then we can decrease the weights $\omega_{i,j}$ for which $(i,j) \in D^-$. See Figure 1 for an example. The dimensions indicated by the solid (resp. dotted) line can be considered to be in D^+ (resp. D^-). One

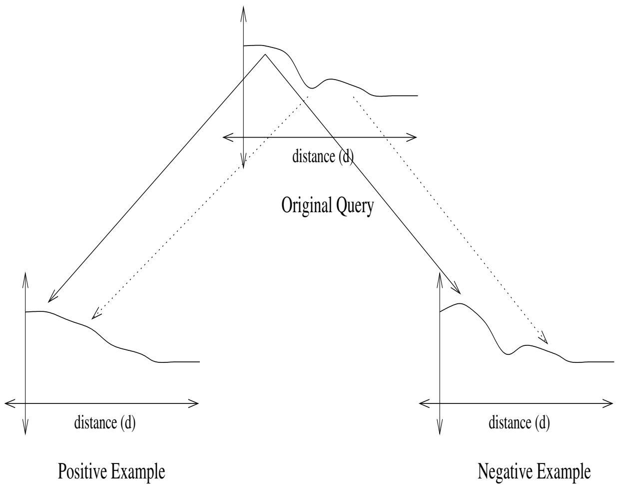


Figure 1: Learning the Metric.

scheme to implement this idea is given below.

Given $\mathcal{F}_i^+$, we analyze the difference vector $\delta = |\alpha(\mathcal{Q}) - \alpha(\mathcal{F}_i^+)|$. Since $\mathcal{F}_i^+$ is labeled positive, the dimensions that contribute larger quantities to δ can be considered to be less important. Similarly, the dimensions with lower contributions to δ can be considered to be more important. This intuition is formalized as a simple heuristic update rule, which is given by

$$\omega_{i,j} = \omega_{i,j} \cdot (1 + \overline{\delta} - \delta_{i,j})$$

where $\overline{\delta}$ is the mean of the components of δ.

In a similar manner, given $\mathcal{F}_i^-$, let $\delta = |\alpha(\mathcal{Q}) - \alpha(\mathcal{F}_i^-)|$. Since $\mathcal{F}_i^-$ is labeled negative, the dimensions that contribute larger quantities to δ can be considered to be more important. Similarly, the dimensions with lower contributions to δ can be considered to be less important. A simple update rule that implements this is given by

$$\omega_{i,j} = \omega_{i,j} \cdot (1 - \overline{\delta} + \delta_{i,j})$$

The weight update rule is applied for each example in $\mathcal{F}$. The final weight ω is used to compute the L_1^ω metric for subsequent queries.

EXPERIMENTS

We conduct several experiments to investigate how learning can be used to improve search results. We focus mainly on the image retrieval task, but we also report some preliminary results for the object searching problem.

We use a database of 20,026 color JPEG images for all our experiments. The images are 232×168 in size and are drawn from a variety of sources — the database includes 11,667 images used in Chabot [8], 1,440 images used in QBIC [3], and 6,477 images from Corel. It also includes a few groups of images in PhotoCD format and a number of MPEG video frames from the web [10]. The images deal with a wide variety of subjects — animals, humans, landscapes, photographs of paintings, various objects like tanks, flags, statues, etc. Our image database is thus large and heterogeneous and presents a challenging testbed for various methods.

We consider the RGB colorspace with quantization into 64 colors. To improve performance, we first smooth the images by a small amount. We use the distance set $D = \{1, 3, 5, 7\}$ (so, $d = 4$) for computing the autocorrelograms. The small value of d ensures that the feature vector has reasonable storage requirements.

To evaluate performance, we use a maximum scope of $s = 50$ for the scope vs. recall measure and $w = 50$ for average precision. We use $\beta = 0.4$ for QLEARN. For the rest of the paper, "auto" refers to the autocorrelogram method with no feedback.

Image Retrieval. We first consider a scenario in which the user examines a small but fixed number of top-ranked images retrieved in response to the original query, and provides relevance judgments about them. Our experiments investigate whether this small additional effort on the part of the user can result in improvements in retrieval effectiveness.

We use a set of 29 queries for this experiment. For each query, a set of images is retrieved. The user examines the top 10 images and marks the ones that are relevant. This relevance information is used to modify the feature vector for the initial query or to modify the distance measure used to compare feature vectors. The database is searched again using the modified query vector or distance measure.

This scheme for providing user feedback turns out to be inadequate in some situations, however. If the query is a "difficult" one, then it is possible that few (or even none) of the top 10 images are useful. Thus, the system gets very little new positive information in the above approach. Further, several queries have answer images in which the appearance of the object of interest is significantly different from its appearance in the query. Such answers may not be ranked within the top 10. In order to get more of these answer images within the top-ranks, the user should be able to give positive feedback to the system about such images.

Accordingly, we conduct a second set of experiments in which the user marks the first 5 relevant and first 5 non-relevant images (irrespective of their ranks). Since some of the 29 queries used above had fewer than five answer images, we use a subset of 19 queries for this experiment.

In addition to using the query- and metric-learning techniques individually in this experiment, we study the effect of combining both techniques during the final search. The results for all these experiments are presented in the section on results.

Object Searching. We also conduct some preliminary experiments exploring the usefulness of learning techniques for the object-searching problem. An "interesting" object in an image is selected as the query and images are retrieved using the Correlogram Intersection method [6]. The user then examines the top-ranked images (typically ten or less), and marks a few (typically three to five) of the useful images. The user also specifies the location of the object in these images (for example, by clicking on a point that approximately represents the center of the object in the image). The system uses this information to modify the query or the distance measure and a new search is conducted. No negative feedback is used in these experiments.

RESULTS

Image Retrieval. For completeness, we illustrate results that show that the correlogram seems to be a significantly superior image retrieval tool compared to other methods like the color histogram, the color coherence vector (CCV) [9] and its extensions like CCV/C [10]. The query set consists of 100 queries (on the same $20,026$ image database), each with a unique correct answer. The query-answer pairs represent various situations like different views of the same scene, large changes in appearance, small lighting changes, spatial translations, etc. Examples of some queries and answers (and the rankings according to the histogram, CCV, CCV/C, and autocorrelogram methods) are shown in Figure 2. The overall performance of the autocorrelogram, histogram, CCV, CCV/C using 64 color buckets is compared in Table 1. We can see that autocorrelograms performs much better than the other methods.

From a user's point of view, these results can be interpreted as follows: given a query, the user is guaranteed to locate the correct answer by just checking the top three search results (on average) using autocorrelograms. On the other hand, the user needs to check at least the top 161 search results (on average) to locate the correct answer in the case of histogram (or top 132 search results for the CCV/C). In practice, this suggests that the former is a more "usable" image retrieval scheme than the latter two.

Method	hist	ccv	ccv/c	auto
r-measure	16099	13741	13197	324
avg. r-measure	161	137	132	3
p_1-measure	22.75	29.49	34.85	71.36
avg. p_1-measure	0.23	0.29	0.35	0.71

Table 1: Performance of various methods (64 colors).

Table 2 shows the effectiveness of the initial search (no feedback) and the results obtained using feedback for the top 10 images. For comparison, we also provide the average precision value for the initial search if the histogram is used as the image feature vector instead of the autocorrelogram. Figure 3 shows some examples of some query images and ranks of similar images according to different methods. For the initial retrieval, autocorrelograms significantly outperform histograms. This agrees with the observations above. When feedback is used, results improve by 15 to 20% over the initial retrieval[2]. The QLEARN and WLEARN approaches both appear to be effective methods for using feedback information. The improvements obtained using feedback are shown graphically in Figure 4. The number of relevant images retrieved by the different methods is plotted against the scope. From the graph, it is clear that the use of feedback improves performance — the curves for QLEARN and WLEARN lie consistently above the curve for the initial retrieval.

Negative Examples. We also investigate the usefulness of negative examples. First, only the images marked positive are used to modify the distance measure in the WLEARN method. Next, the user's feedback about all the ten images is used (i.e., both positive and negative). The results for these two approaches are shown in the last two columns in Table 2. From the table, it appears that the use of negative information is marginally more useful on the whole. A more detailed analysis shows, however, that there are queries for which WLEARN(+only) does better than WLEARN. Typically, if only one or two out of the top ten images are judged useful,

[2] This improvement in average precision is partly due to the improvement in ranks of the images that are marked relevant by the user. The improvement for previously unseen images is really what is more relevant. Figure 4 shows that this improvement is also substantial — see the increase in recall for scope > 10, for example.

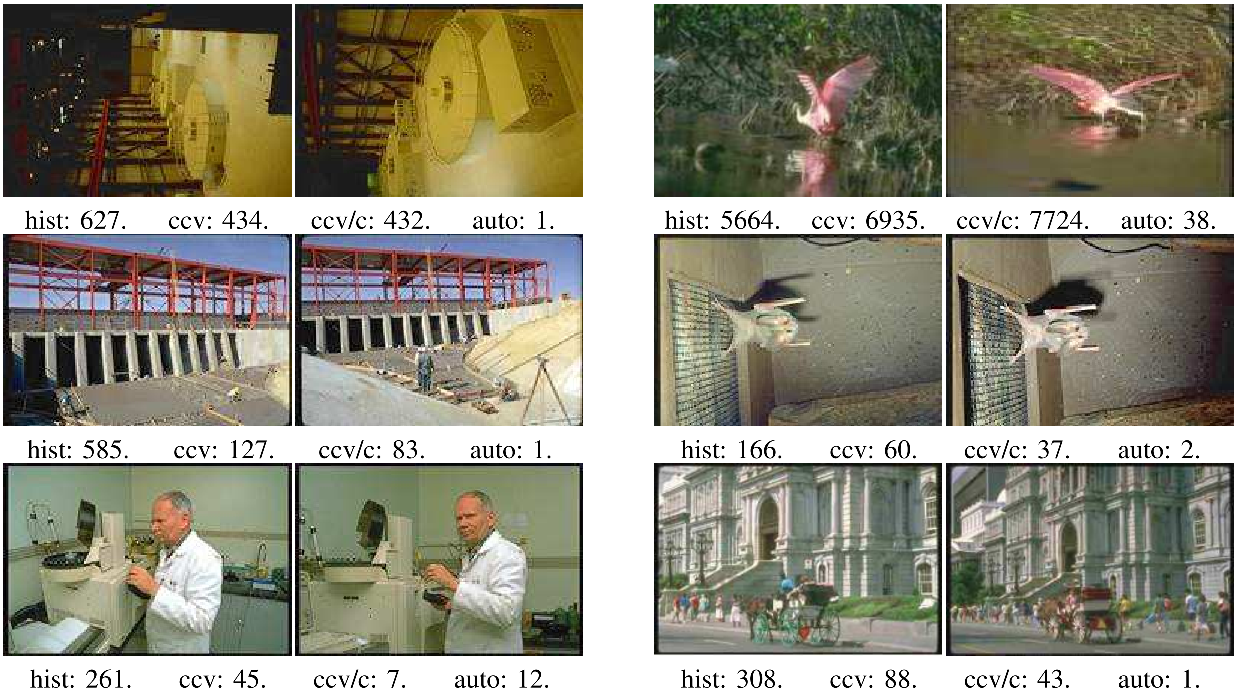


Figure 2: Sample queries and answers with ranks for various methods. (Lower ranks indicate better performance.)

	Histogram	Correlogram			
	hist	auto	QLEARN	WLEARN(+ only)	WLEARN
Average Precision	0.425	0.547	0.661	0.627	0.631
%-age Change	–	–	+ 20.8	+ 14.6	+ 15.4

Table 2: Retrieval effectiveness with and without feedback (top 10 images).

the large number of negative examples could change the weights in the WLEARN method in undesirable ways, and the final result would deteriorate. The next set of experiments investigates a setting in which an equal number of positive and negative examples are used.

Figure 5 and Table 3 show the results of the second set of experiments. For this experiment, the user provides feedback about the first 5 relevant and first 5 non-relevant images. Once again, the use of feedback results in significant improvements. Since the QLEARN and WLEARN methods are in some sense independent and both seem to be effective, we investigate a combination of these methods. We call this method QWLEARN. Table 3 and Figure 5 suggest that the two methods are indeed complementary, since the results for QWLEARN are significantly better than either method — the average precision for QWLEARN is about 8% better than that for WLEARN, and the curve for QWLEARN is consistently higher than the curves corresponding to QLEARN and WLEARN.

Object Searching. Figure 6 shows some sample queries and answers and their ranks before and after learning[3]. The examples suggest that a small amount of feedback from the user can improve results substantially. For instance, the image of the owl moves up from the 75th rank to the sixth position. The improvement in the rank of useful images usually implies that the number of useful images at a given scope also increases. The object-searching problem is a difficult one, however, and more experimentation is needed before reliable conclusions can be drawn about using learning methods for this problem.

[3] Note that these answers were not seen by the user at the end of the first pass. Of course, the ranks of the images that are judged relevant by the user are expected to improve.

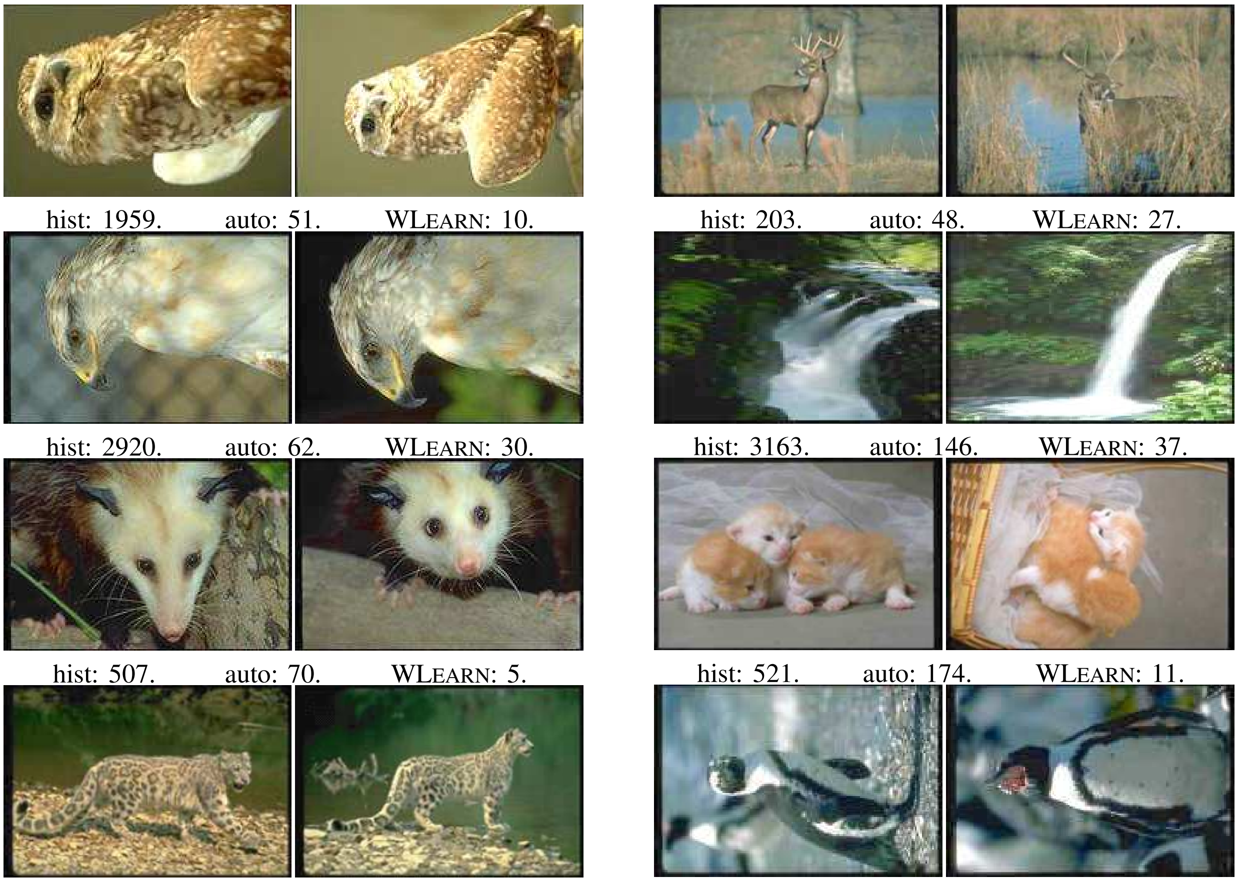


Figure 3: Comparison of various methods for image retrieval. (Lower numbers indicate better performance.)

	Histogram	Correlogram			
	hist	auto	QLEARN	WLEARN	QWLEARN
Average Precision	0.379	0.613	0.690	0.710	0.766
%-age Change	–	–	+ 12.6	+ 15.8	+ 25.0

Table 3: Retrieval effectiveness with and without feedback (first 5 positive and negative examples).


auto: 63.

QLEARN: 35.

auto: 17.

QLEARN: 4.

auto: 75.

QLEARN: 6.

auto: 29.

QLEARN: 15.

auto: 26.

QLEARN: 7.

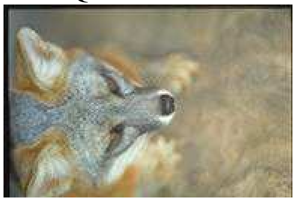
auto: 63.

QLEARN: 44.

Figure 6: Comparison of various methods for object searching. (Lower numbers indicate better performance.)

CONCLUSIONS

We show methods to incorporate feedback information to enhance the quality of image retrieval using correlograms. Our learning methods are in fact independent of feature vectors and could conceivably be used for other feature vectors as well. Initial experiments indicate that the proposed learning methods significantly improve retrieval quality using little effort on the part of the user. The application of these techniques to object searching needs to be studied in greater detail. In particular, it is not obvious if one of the methods actually outperforms the other, though they seem to have a synergic effect when combined. A user-study of the performance of these methods will also be interesting.

ACKNOWLEDGEMENTS

We thank Ramin Zabih for useful suggestions. We also thank the anonymous reviewers for their comments.

REFERENCES

1. I. J. Cox, M. L. Miller, S. M. Omohundro, and P. N. Yianilos. PicHunter: Bayesian relevance feedback for image retrieval. *Intl. Conf. on Pattern Recognition*, 1996.

2. R. L.Delanoy. Supervised learning of tools for content-based search of image databases. *SPIE proceedings*, 2670:194–205, 1996.

3. M. Flickner, H. S. Sawhney, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J .Hafner, D .Lee, D .Petkovic, D. Steele, and P .Yanker. Query by image and video content: The QBIC system. *IEEE Computer*, 28(9):23–32, September 1995.

4. R. M. Haralick. Statistical and structural approaches to texture. *Proceedings of IEEE*, 67(5):786–804, 1979.

5. J. Huang, S. R. Kumar, M. Mitra, W. J. Zhu, and R. Zabih. Image indexing using color correlograms. *Proc. Computer Vision and Pattern Recognition*, pages 762–768, 1997.

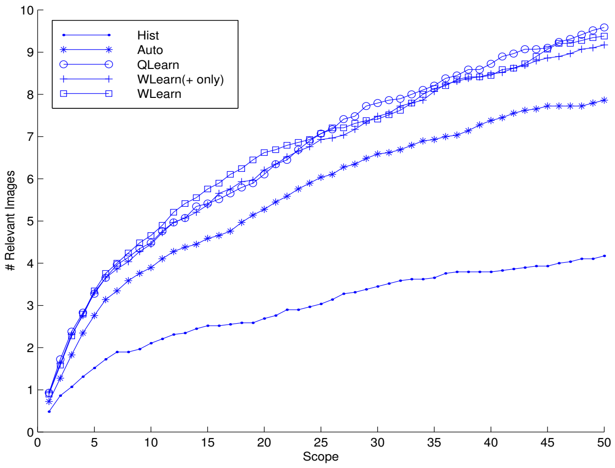


Figure 4: Graph showing number of relevant images retrieved at various scopes.

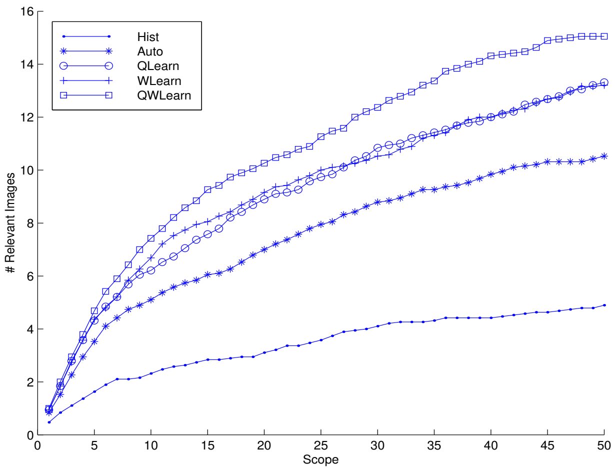


Figure 5: Graph showing number of relevant images retrieved at various scopes.

6. J. Huang, S. R. Kumar, M. Mitra, W. J. Zhu. Spatial color indexing and applications. *Intl. Conf. on Computer Vision*, 1998. To appear.

7. T. Minka and R. Picard. Interactive learning using a “society of models”. *Proc. Computer Vision and Pattern Recognition*, 1996.

8. V. Ogle and M. Stonebraker. Chabot: Retrieval from a relational database of images. *IEEE Computer*, 28(9):40–48, September 1995.

9. G. Pass, R. Zabih and J. Miller. Comparing Images Using Color Coherence Vectors. *Proceedings of the Fourth ACM Multimedia Conference*, pages 65–73, 1996.

10. G. Pass and R. Zabih. Histogram refinement for content-based image retrieval. *IEEE Workshop on Applications of Computer Vision*, pages 96–102, 1996.

11. A. Pentland, R. Picard, and S. Sclaroff. Photobook: Content-based manipulation of image databases. *Intl. Journal of Computer Vision*, 18(3):233–254, 1996.

12. P. J. Rousseeuw and A. M. Leroy. *Robust Regression and Outlier Detection.* John Wiley & Sons, 1987.

13. J. Smith and S-F. Chang. VisualSEEK: a fully automated content-based image query system. *Proceedings of the Fourth ACM Multimedia Conference*, pages 87–98, 1996.

14. J. Smith and S-F. Chang. Tools and techniques for color image retrieval. *SPIE proceedings*, 2670:1630–1639, 1996.

15. M. Stricker and A. Dimai. Color indexing with weak spatial constraints. *SPIE proceedings*, 2670:29–40, 1996.

16. M. Swain and D. Ballard. Color indexing. *Intl. Journal of Computer Vision*, 7(1):11–32, 1991.

17. G. J. G. Upton and B. Fingleton. *Spatial Data Analysis by Example. Vol I.* John Wiley & Sons, 1985.