
International Journal of Computer Vision 35(3), 245–268 (1999)

Spatial Color Indexing and Applications

JING HUANG, S. RAVI KUMAR, MANDAR MITRA, WEI-JING ZHU AND RAMIN ZABIH
Cornell University, Ithaca, NY 14850
huang@cs.cornell.edu
ravi@cs.cornell.edu
mitra@cs.cornell.edu
wjzhu@msc.cornell.edu
rdz@cs.cornell.edu

Abstract. We define a new image feature called the *color correlogram* and use it for image indexing and comparison. This feature distills the spatial correlation of colors and when computed efficiently, turns out to be both effective and inexpensive for content-based image retrieval. The correlogram is robust in tolerating large changes in appearance and shape caused by changes in viewing position, camera zoom, etc. Experimental evidence shows that this new feature outperforms not only the traditional color histogram method but also the recently proposed histogram refinement methods for image indexing/retrieval. We also provide a technique to cut down the storage requirement of the correlogram so that it is the same as that of histograms, with only negligible performance penalty compared to the original correlogram.

We also suggest the use of color correlogram as a generic indexing tool to tackle various problems arising from image retrieval and video browsing. We adapt the correlogram to handle the problems of image subregion querying, object localization, object tracking, and cut detection. Experimental results again suggest that the color correlogram is more effective than the histogram for these applications, with insignificant additional storage or processing cost.

Keywords: image indexing, image features, content-based image retrieval, model-based object recognition, spatial correlation

1. Introduction

In recent times, the availability of image and video resources on the World-Wide Web has increased tremendously. This has created a demand for effective and flexible techniques for automatic image retrieval and video browsing (Cox et al., 1996; Flickner et al., 1995; Forsyth et al., 1996; Hampapur et al., 1994; Ogle and Stonebraker, 1995; Pass and Zabih, 1996; Pentland et al., 1996; Smith and Chang, 1996). Users need high-quality image retrieval (IR) systems in order to find useful images from the masses of digital image data available electronically. 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.

Besides the basic image retrieval and video processing tasks, several related problems also need to be addressed. While most IR systems retrieve images based on overall image comparison, users are typically interested in finding objects (Fleck et al., 1996; Forsyth, 1996; Enser, 1993). In this case, the user specifies an "interesting" subregion (usually an interesting object) of an image as a query. The system should then retrieve images containing this subregion (according to human perception) or object from a database. This

task, called *image subregion querying*, is made challenging by the wide variety of effects (such as different viewing positions, camera noise and variation, object occlusion, etc.) that cause the same object to have drastically different appearances in different images.

The system should also be able to solve the *localization* problem (also called the *recognition* problem), i.e., it should find the location of the object in an image. The lack of an effective and efficient image segmentation process for large, heterogeneous image databases implies that objects have to be located in unsegmented images, making the localization problem more difficult.

Similar demands arise in the context of content-based video browsing. A primary task in video processing is *cut detection*, which segments a video into different camera shots and helps to extract key frames for video parsing and querying. A flexible tool for browsing video databases should also provide users with the capability to pose object-level queries that have semantic content, such as "track this person in a sequence of video". To handle such queries, the system has to find which frames contain the specified object or person, and has to locate the object in those frames.

The various tasks described above—image retrieval, image subregion querying, object localization and cut detection—become especially challenging when the image database is gigantic. For example, the collection of images available on the Internet is huge and unorganized. The image data is arbitrary, unstructured, and unconstrained; and the processing has to be done in real-time for retrieval purposes. For these reasons, traditional (and often slow) computer vision techniques like object recognition and image segmentation may not be directly applicable to these tasks and new approaches to these problems are required.

Consider first the basic problem of content-based image retrieval. This problem has been widely studied and several IR systems have been built (Flickner, 1995; Ogle and Stonebraker, 1995; Pentland et al., 1996; Cox et al., 1996; Pass and Zabih, 1996). Most of these IR systems adopt the following two-step approach to search image databases (Stricker and Swain, 1994): (i) (*indexing*) for each image in a database, a feature vector capturing certain essential properties of the image is computed and stored in a featurebase, and (ii) (*searching*) given a query image, its feature vector is computed, compared to the feature vectors in the featurebase, and images most similar to the query image are returned to the user. An overview of such systems can be found in (1995).

For a retrieval system to be successful, the feature vector $f(\mathcal{I})$ for an image $\mathcal{I}$ should have the following qualities: (i) $|f(\mathcal{I}) - f(\mathcal{I}')|$ should be large if and only if $\mathcal{I}$ and $\mathcal{I}'$ are not "similar", (ii) $f(\cdot)$ should be fast to compute, and (iii) $f(\mathcal{I})$ should be small in size.

Color histograms are commonly used as feature vectors for images (Swain and Ballard, 1991; Flickner et al., 1995; Ogle and Stonebraker, 1995; Pentland et al., 1996). It has been shown that the color histogram is a general and flexible tool that can be used for the various tasks outlined above.

1.1. Our Results

In this paper, we propose a new color feature for image indexing/retrieval called the *color correlogram* and show that it can be effectively used in the various image and video processing tasks described above. The highlights of this feature are: (i) it includes the spatial correlation of pairs of colors, (ii) it describes the global distribution of local spatial correlations of colors, (iii) it is easy to compute, and (iv) the size of the feature is fairly small. Experimental evidence shows that this new feature (i) outperforms both the traditional histogram method and the recently proposed histogram refinement method for image indexing/retrieval, and (ii) outperforms the histogram-based approaches for the other video browsing tasks listed above.

Informally, a correlogram is a table indexed by color pairs, where the kth entry for $\langle i, j \rangle$ specifies the probability of finding a pixel of color j at a distance k from a pixel of color i. Such an image feature turns out to be robust in tolerating large changes in appearance of the same scene caused by changes in viewing positions, changes in the background scene, partial occlusions, camera zoom that causes radical changes in shape, etc. (see Fig. 1). We provide efficient algorithms to compute the correlogram.

We also investigate a different distance metric to compare feature vectors. The L_1 distance metric, used commonly to compare vectors, considers the absolute component-wise differences between vectors. The *relative distance metric* we use calculates relative differences instead and in most cases performs better than

the absolute metric (the improvement is significant especially for histogram-based methods).

We investigate the applicability of correlograms to image retrieval as well as other tasks like image subregion querying, object localization, and cut detection. We propose the *correlogram intersection* method for the image subregion querying problem and show that this approach yields significantly better results than the histogram intersection method traditionally used in content-based image retrieval. The histogram-backprojection approach used for the localization problem in (Swain and Ballard, 1991) has serious drawbacks. We discuss these disadvantages and introduce the idea of *correlogram correction*. We show that it is possible to locate objects in images more accurately by using local color spatial information in addition to histogram backprojection. We then use correlograms to compare video frames and detect cuts by looking for adjacent frames that are very different. Once again, we show that using the correlogram as the feature vector yields superior results compared to using histograms.

Our preliminary results thus indicate that the correlogram method is a more accurate and effective approach to these tasks compared to the color histogram method.

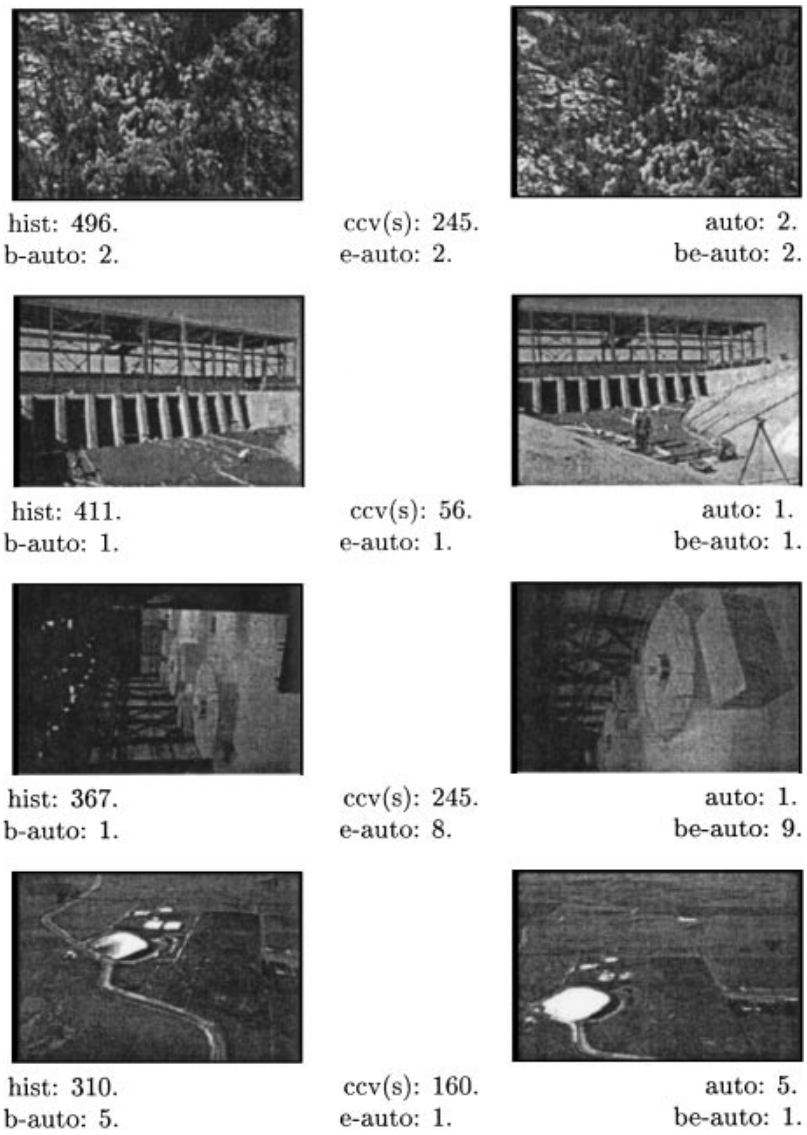

Figure 1. Sample queries and answers with ranks for various methods. (Lower ranks are better.)

(*continued on next page*)

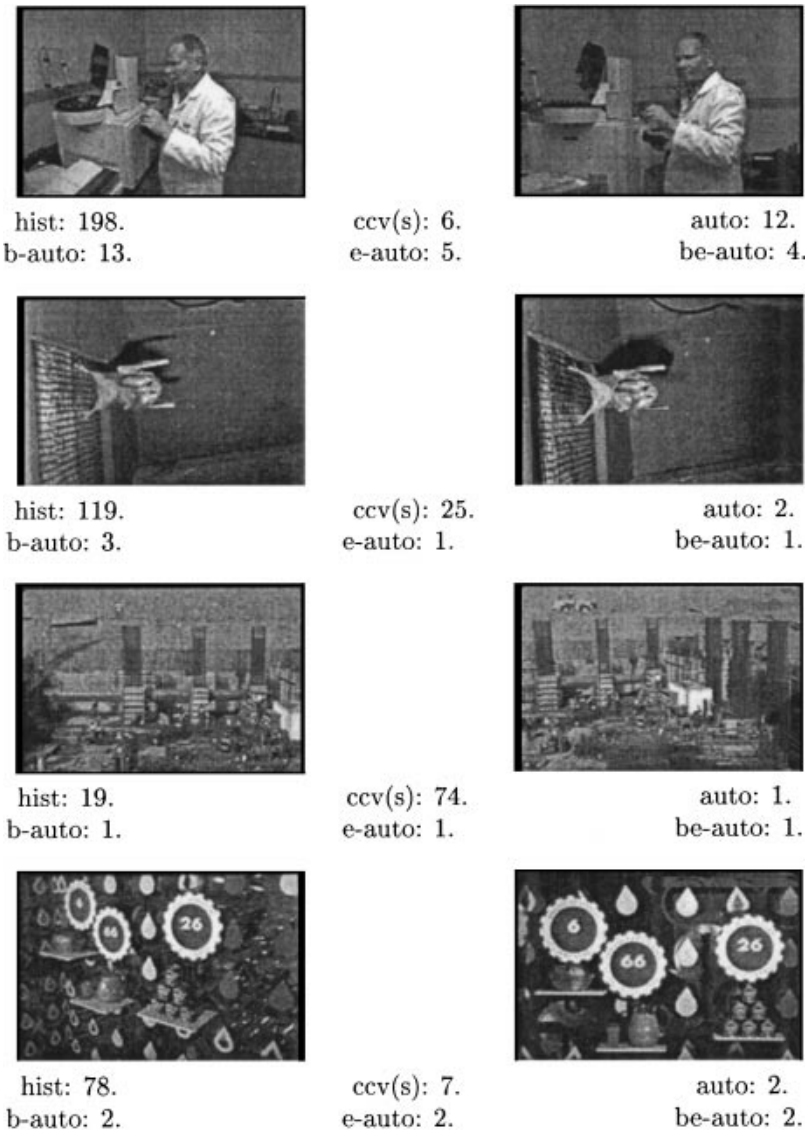

Figure 1. (*continued.*)

What is more, the computational cost of the correlogram method is about the same as that of other simpler approaches, such as the histogram method.

1.2. Organization

Section 2 gives a brief summary of related work. In Section 3, we define the color correlogram and show how to compute it efficiently. Section 4 deals with the content-based image retrieval problem and the use of the correlogram for this problem. Section 5 discusses the use of the correlogram for image subregion querying. Applications of the correlogram to video browsing problems are described in Section 6. Finally, Section 7 concludes with some remarks and scope for further work.

2. Related Work

Color histograms are commonly used as image feature vectors (Swain and Ballard, 1991; Flickner et al., 1995; Ogle and Stonebraker, 1995; Pentland et al., 1996) and have proved to be a useful and efficient general tool for various applications, such as content-based image

Figure 2. Two "similar" images with different histograms.

retrieval (Flickner et al., 1995; Ogle and Stonebraker, 1995; Pentland et al., 1996), object indexing and localization (Swain and Ballard, 1991; Matas et al., 1995), and cut detection for video segmentation (Boresczky and Rowe, 1996). A color histogram describes the global color distribution in an image. It is easy to compute and is insensitive to small changes in viewing positions and partial occlusion. As a feature vector for image retrieval, it is susceptible to false positives, however, since it does not include any spatial information. This problem is especially acute for large databases, where false positives are more likely to occur. Moreover, the histogram is not robust to large appearance changes. For instance, the pair of images shown in Fig. 2 (photographs of the same scene taken from different viewpoints) are likely to have quite different histograms.[1]

Color histograms are also used for image subregion querying and object localization (Swain and Ballard, 1991). These two problems are closely related to object recognition, which has been studied for a long time in computer vision (Roberts, 1965). Since conventional object recognition techniques cannot recognize general objects in general contexts (as in the natural imagery and real videos), some work has been done for finding objects from image databases (Fleck et al., 1996; Forsyth, 1996). These techniques, however, are trained for some specific tasks, such as finding naked people, grouping trees, etc. Color histograms are also widely used in video processing. Though there are several sophisticated techniques for video cut detection, Boreczky and Rowe (1996) report that the simple color histogram yields consistently good results compared to five different techniques.

We now briefly discuss some other related work in the areas of content-based image retrieval, image subregion querying, object localization, and cut detection.

2.1. Content-Based Image Retrieval

Several recently proposed schemes incorporate spatial information about colors to improve upon the histogram method (Hsu et al., 1995; Smith and Chang, 1996; Stricker and Dimai, 1996; Rickman and Stonham, 1996; Yihong et al., 1994; Pass et al., 1996; Pass and Zabih, 1996). One common approach is to divide images into subregions and impose positional constraints on the image comparison. Another approach is to augment the histogram with some spatial information.

Hsu et al. (1995) select two representative colors signifying the "background" and the principal "object" in an image. The maximum entropy algorithm is then used to partition an image into rectangular regions. Only one selected color dominates a region. The similarity between two images is the degree of overlap between regions of the same color. The method is tested on a small image database. Unfortunately, this method uses coarse color segmentation and is susceptible to false positives.

Smith and Chang (1996) also partition an image, but 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 (Swain and Ballard, 1991). The binary color sets and their location information constitute the feature. The absolute spatial position allows the system to deal with "region" queries.

Stricker and Dimai (1996) 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 low. The use of overlapping regions makes the feature vectors relatively insensitive to small rotations or translations.

Pass et al. (1996a, 1996b) 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 perform much better than histograms. A detailed comparisons of CCV's with the other methods mentioned above is given in (Pass and Zabih, 1996).

The notion of CCV was further extended in (Pass and Zabih, 1999) where additional feature(s) are used

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(s)) and performs better than CCV's.

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 and this makes the correlogram robust to large appearance changes (see Fig. 1). Moreover, this information is not a local pixel property that histogram refinement approaches can capture.

2.2. Other Image/Video Problems

The image subregion querying problem is closely related to the object recognition problem, which has been studied for a long time by the computer vision community (Roberts, 1965). Some of the early work in object recognition and detection was pioneered by Marr (1978), who suggest that geometric cues such as edge, surface and depth information be identified before object recognition is attempted. Most of such object recognition systems compare the geometric features of the model with those of an image using various forms of search, some of which are computationally quite intensive (Huttenlocher and Ullman, 1986; Grimson and Lozano-Pérez, 1987).

Such geometric information is hard to extract from an image, however, because geometric and photometric properties are relatively uncorrelated (Rao and Ballard, 1995), and the central tasks involved in this approach—edge detection and region segmentation—are difficult for unconstrained data in the context of image retrieval and video browsing.

An alternative approach to model-based recognition is *appearance matching*. First, a database of object images under different view positions and lighting conditions is constructed. Then, *principal component analysis* is used to analyze only the photometric properties and ignore geometric properties (Murase and Nayar, 1995; Huttenlocher et al., 1996; Rao and Ballard, 1995). This model-based method is effective only when the principal components capture the characteristics of the whole database. For instance, it yields good results on the Columbia object database in which all images have a uniform known background. If there is a large variation in the images in a database, however, a small set of principal components is unlikely to do well on the image subregion querying task. In addition, the learning process requires homogeneous data and deals poorly with outliers. Therefore, this approach seems suitable only for domain-specific applications, but not for image subregion querying from a large heterogeneous image database such as the one used in (Pass and Zabih, 1996; Huang et al., 1997).

Since the color information (e.g. histogram) is very easy to extract from an image, it has been successfully used for object indexing, detection, and localization (Swain and Ballard, 1991; Flickner et al., 1995; Ogle and Stonebraker, 1995; Matas et al., 1995; Slater and Healey, 1995; Brock-Gunn and Ellis, 1992; Syeda-Mahmood, 1997; Forsyth, 1996). We briefly review some of these approaches below.

Swain and Ballard (1991) propose *histogram intersection* for object identification and *histogram backprojection* for object localization. The technique is computationally easy, does not require image segmentation or even foreground/background separation, and is insensitive to small changes in viewing positions, partial occlusion, and object deformation. Histogram backprojection is a very efficient process for locating an object in an image. It has been shown that this algorithm is not only able to locate an object but also to track a moving object. The advantages and disadvantages inherent to histograms in general are discussed in detail in Section 5.

One disadvantage of color histograms is that they are sensitive to illumination changes. Slater and Healey (1995) propose an algorithm that computes invariants of local color distribution and uses these invariants for 3-D object recognition. Illumination correction and spatial structure comparison are then used to verify the potential matches.

Matas et al. (1995) propose the color adjacency graph (CAG) as a representation for multiple-colored objects. Each node of a CAG represents a single color component of the image. Edges of the CAG include information about adjacency of color components. CAGs improve over histograms by incorporating coarse color segmentation into histograms. The set of visible colors and their adjacency relationship remain stable under changes of viewpoint and non-rigid transformations. The recognition and localization problems are solved

by subgraph matching. Their approach yields excellent results, but the computational cost of subgraph matching is fairly high.

Forsyth et al. (1996) offer different object models in order to achieve object recognition under general contexts. Their focus is on classification rather than identification. The central process is based on grouping (i.e., segmentation) and learning. They fuse different visual cues such as color and texture for segmentation; texture and geometric properties for trees; color, texture and specialized geometric properties for human bodies.

Cut detection, as a first step to video segmentation and video querying, has been given much attention (Boresczky and Rowe, 1996). The simple histogram approach gives reasonably good results on this problem. Histograms, however, are not robust to local changes in images. Dividing an image into several subregions may not overcome the problem (Hampapur et al., 1994) either.

3. The Correlogram

A color correlogram (henceforth correlogram) expresses how the spatial correlation of color changes with distance. A color histogram (henceforth histogram) captures only the color distribution in an image and does not include any spatial correlation information. Thus, the correlogram is one kind of spatial extension of the histogram.[2]

3.1. Notation

Let $\mathcal{I}$ be an $n \times n$ image. (For simplicity, we assume that the image is square.) The colors in $\mathcal{I}$ are quantized into m colors $c_1, \ldots, c_m$. (In practice, m is deemed to be a constant and hence we drop it from our running time analysis.)

For a pixel $p = (x, y) \in \mathcal{I}$, let $\mathcal{I}(p)$ denote its color. 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, i.e., for pixels $p_1 = (x_1, y_1)$, $p_2 = (x_2, y_2)$, we define $|p_1 - p_2| \triangleq \max\{|x_1 - x_2|, |y_1 - y_2|\}$. We denote the set $\{1, 2, \ldots, n\}$ by $[n]$.

3.2. Definitions

The *histogram* h of $\mathcal{I}$ is defined for $i \in [m]$ by

$$h_{c_i}(\mathcal{I}) \triangleq n^2 \cdot \Pr_{p \in \mathcal{I}}\left[p \in \mathcal{I}_{c_i}\right]. \quad (1)$$

For any pixel in the image, $h_{c_i}(\mathcal{I})/n^2$ gives the probability that the color of the pixel is c_i. The histogram can be computed in $O(n^2)$ time, which is linear in the image size.

Let a distance $d \in [n]$ be fixed a priori. Then, the *correlogram* of $\mathcal{I}$ is defined for $i, j \in [m], k \in [d]$ as

$$\gamma^{(k)}_{c_i,c_j}(\mathcal{I}) \triangleq \Pr_{\substack{p_1 \in \mathcal{I}_{c_i} \\ p_2 \in \mathcal{I}}}\left[p_2 \in \mathcal{I}_{c_j} \,\middle|\, |p_1 - p_2| = k\right]. \quad (2)$$

Given any pixel of color c_i in the image, $\gamma^{(k)}_{c_i,c_j}$ gives the probability that a pixel at distance k away from the given pixel is of color c_j. Note that the size of the correlogram is $O(m^2 d)$. The *autocorrelogram* of $\mathcal{I}$ captures spatial correlation between identical colors only and is defined by

$$\alpha^{(k)}_c(\mathcal{I}) \triangleq \gamma^{(k)}_{c,c}(\mathcal{I}). \quad (3)$$

This information is a subset of the correlogram and requires only $O(md)$ space.

While choosing d to define the correlogram, we need to address the following. A large d would result in expensive computation and large storage requirements. A small d might compromise the quality of the feature. We consider this issue in Section 4.1.

Example 1. Consider the simple case when $m = 2$ and $n = 8$. Two sample images are shown in Fig. 3. The autocorrelograms corresponding to these two images are shown in Fig. 4. The change of autocorrelation

Figure 3. Sample images: image 1, image 2.

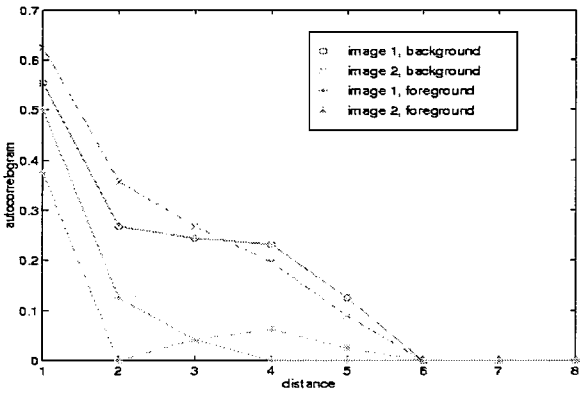

Figure 4. Autocorrelograms for images in Fig. 3.

of the foreground color with distance is perceptibly different for these images.[3]

3.3. Distance Metrics

The L_1 and L_2 norms are commonly used distance metrics when comparing two feature vectors. In practice, the L_1 norm performs better than the L_2 norm because the former is more robust to outliers (Rousseeuw and Leroy, 1987). Hafner et al. (1995) suggest using a more sophisticated quadratic form of distance metric, which tries to capture the perceptual similarity between any two colors. To avoid intensive computation of quadratic functions, they propose to use low-dimensional color features as filters before using the quadratic form for the distance metric.

We will use the L_1 norm for comparing histograms and correlograms because it is simple and robust. The following formulae are used to compute the distance between images $\mathcal{I}$ and $\mathcal{I}'$:

$$|\mathcal{I} - \mathcal{I}'|_{h,L_1} \triangleq \sum_{i\in[m]} \left|h_{c_i}(\mathcal{I}) - h_{c_i}(\mathcal{I}')\right| \tag{4}$$

$$|\mathcal{I} - \mathcal{I}'|_{\gamma,L_1} \triangleq \sum_{\substack{i,j\in[m]\\ k\in[d]}} \left|\gamma^{(k)}_{c_i,c_j}(\mathcal{I}) - \gamma^{(k)}_{c_i,c_j}(\mathcal{I}')\right| \tag{5}$$

From these equations, it is clear that the contributions of different colors to the dissimilarity are equally weighted. Intuitively, however, this contribution should be weighted to take into account some additional factors.

Example 2. Consider two pairs of images $\langle \mathcal{I}_1, \mathcal{I}_2\rangle$ and $\langle \mathcal{I}'_1, \mathcal{I}'_2\rangle$. Let $h_{c_i}(\mathcal{I}_1) = 1000$, $h_{c_i}(\mathcal{I}_2) = 1050$, $h_{c_i}(\mathcal{I}'_1) = 100$, and $h_{c_i}(\mathcal{I}'_2) = 150$. Even though the absolute difference in the pixel count for color bucket i is 50 in both cases, clearly the difference is more significant for the second pair of images.

Thus, the difference $|h_{c_i}(\mathcal{I}) - h_{c_i}(\mathcal{I}')|$ in Eq. (4) should be given more importance if $|h_{c_i}(\mathcal{I}) + h_{c_i}(\mathcal{I}')|$ is small and vice versa. We could therefore consider replacing the expression $|h_{c_i}(\mathcal{I}) - h_{c_i}(\mathcal{I}')|$ in Eq. (4) by

$$\frac{\left|h_{c_i}(\mathcal{I}) - h_{c_i}(\mathcal{I}')\right|}{1 + h_{c_i}(\mathcal{I}) + h_{c_i}(\mathcal{I}')} \tag{6}$$

(the 1 in the denominator is added to prevent division by zero).

This intuition has theoretical justification in (Haussler, 1992) which suggests that it is sometimes better to use a "relative" measure of distance d_μ. For $\mu > 0, r, s \geq 0$, d_μ is defined by

$$d_\mu(r,s) = \frac{|r-s|}{\mu + r + s}. \tag{7}$$

It is straightforward to verify that (i) d_μ is a metric, (ii) for $r, s \geq 0$, $d_\mu(r,s) \in [0,1)$, and (iii) for $0 \leq r \leq s \leq t$, $d_\mu(r,s) \leq d_\mu(r,t)$, $d_\mu(s,t) \leq d_\mu(r,t)$.

d_μ can be applied to feature vectors also. We have set $\mu = 1$. So the d_1 distance metric for histograms and correlograms is:

$$|\mathcal{I} - \mathcal{I}'|_{h,d_1} \triangleq \sum_{i\in[m]} \frac{\left|h_{c_i}(\mathcal{I}) - h_{c_i}(\mathcal{I}')\right|}{1 + h_{c_i}(\mathcal{I}) + h_{c_i}(\mathcal{I}')} \tag{8}$$

$$|\mathcal{I} - \mathcal{I}'|_{\gamma,d_1} \triangleq \sum_{\substack{i,j\in[m]\\ k\in[d]}} \frac{\left|\gamma^{(k)}_{c_i,c_j}(\mathcal{I}) - \gamma^{(k)}_{c_i,c_j}(\mathcal{I}')\right|}{1 + \gamma^{(k)}_{c_i,c_j}(\mathcal{I}) + \gamma^{(k)}_{c_i,c_j}(\mathcal{I}')} \tag{9}$$

3.4. An Algorithm

In this section, we look at an efficient algorithm to compute the correlogram. Our algorithm is amenable to easy parallelization. Thus, the computation of the correlogram could be enormously speeded up.

First, to compute the correlogram, it suffices to compute the following count (similar to the *cooccurrence matrix* defined in (Haralick, 1979) for texture analysis of gray images)

$$\Gamma^{(k)}_{c_i,c_j}(\mathcal{I}) \triangleq \left|\left\{p_1 \in \mathcal{I}_{c_i},\, p_2 \in \mathcal{I}_{c_j} \;\middle|\; |p_1 - p_2| = k\right\}\right| \tag{10}$$

because,

$$\gamma_{c_i,c_j}^{(k)}(\mathcal{I}) = \frac{\Gamma_{c_i,c_j}^{(k)}(\mathcal{I})}{8k h_{c_i}(\mathcal{I})}. \quad (11)$$

The denominator is the total number of pixels at distance k from any pixel of color c_i. (The factor $8k$ is due to the properties of L_∞-norm.) The naive algorithm would be to consider each $p_1 \in \mathcal{I}$ of the color c_i and for each $k \in [d]$, count all $p_2 \in \mathcal{I}$ of color c_j with $|p_1 - p_2| = k$. Unfortunately, this takes $O(n^2d^2)$ time. To obviate this expensive computation, we define the quantities

$$\lambda_{(x,y)}^{c,h}(k) \triangleq |\{(x+i, y) \in \mathcal{I}_c \mid 0 \le i \le k\}| \quad (12)$$

$$\lambda_{(x,y)}^{c,v}(k) \triangleq |\{(x, y+j) \in \mathcal{I}_c \mid 0 \le j \le k\}| \quad (13)$$

which count the number of pixels of a given color within a given distance from a fixed pixel in the positive horizontal/vertical directions.

Our algorithm works by first computing $\lambda_p^{c_j,v}$ and $\lambda_p^{c_j,h}$. We now give an algorithm with a running time of $O(n^2d)$ based on dynamic programming.

The following equation is easy to check

$$\lambda_{(x,y)}^{c,h}(k) = \lambda_{(x,y)}^{c,h}(k-1) + \lambda_{(x+k,y)}^{c,h}(0) \quad (14)$$

with the initial condition

$$\lambda_p^{c,h}(0) = 1 \quad \text{if } p \in \mathcal{I}_c \text{ and 0 otherwise.} \quad (15)$$

Now, $\lambda_p^{c,h}(k)$ is computed for all $p \in \mathcal{I}$ and for each $k = 1, \ldots, d$ using Eq. (14). The correctness of this algorithm is obvious. Since we do $O(n^2)$ work for each k, the total time taken is $O(n^2d)$.

In a similar manner, $\lambda_p^{c,v}$ can also be computed efficiently. Now, ignoring boundaries, we have

$$\begin{aligned}\Gamma_{c_i,c_j}^{(k)}(\mathcal{I}) &= \sum_{(x,y)\in\mathcal{I}_{c_i}} \Big(\lambda_{(x-k,y-k+1)}^{c_j,v}(2k-2) + \lambda_{(x-k,y-k)}^{c_j,h}(2k) \\ &\quad + \lambda_{(x-k,y+k)}^{c_j,h}(2k) + \lambda_{(x+k,y-k+1)}^{c_j,v}(2k-2)\Big)\end{aligned}$$

This computation takes just $O(n^2)$ time.

The hidden constants in the overall running time of $O(n^2d)$ are very small and hence this algorithm is extremely efficient in practice for small d.

3.5. *Some Extensions*

In this section, we will look at some extensions to color correlograms. The general theme behind the extensions are: (1) improve the storage efficiency of the correlogram while not compromising its image discrimination capability, and (2) use additional information (such as intensity edges) to further refine the correlogram, boosting its image retrieval performance. These extensions can not only be used for the image retrieval problem, but also in other applications like cut-detection (see Section 6.2).

3.5.1. Banded Correlogram. In Section 3.4, we saw that the correlogram (resp. autocorrelogram) takes m^2d (resp. md) space. Though we will see that small values of d actually suffices, it will be more advantageous if the storage requirements were trimmed further. This leads to the definition of *banded correlogram* for a given b. (For simplicity, assume b divides d.) For $1 \le k \le b$,

$$\bar{\gamma}_{c_i,c_j}^{(k)}(\mathcal{I}) \triangleq \sum_{k'=kb}^{(k+1)b-1} \gamma_{c_i,c_j}^{(k')}(\mathcal{I}). \quad (16)$$

In a similar manner, the banded autocorrelogram $\bar{\alpha}_{c_i}^{(k)}(\mathcal{I})$ can also be defined. The space requirements for the banded correlogram (resp. banded autocorrelogram) is m^2d/b (resp. md/b). (Note that when $b = d$, $\bar{\gamma}$ measures the *density* of a color c_j near the color c_i, thus suggesting the local structure of colors.) The distance metric defined in Eq. (5) is easily extended to this case.

Note that banded correlograms are seemingly more susceptible to false matches since

$$|\mathcal{I} - \mathcal{I}'|_{\bar{\gamma},L_1} \le |\mathcal{I} - \mathcal{I}'|_{\gamma,L_1}, \quad (17)$$

which follows by the triangle inequality. Although the banded correlograms have less detailed information as correlograms, our results show that the approximation of γ by $\bar{\gamma}$ has only negligible effect on the quality of the image retrieval problem and other applications.

3.5.2. Edge Correlogram. The idea of exploiting spatial correlation between pairs of colors can also be extended to other image features such as edges. In the following, we augment the color correlogram with edge information. This new feature, called the *edge correlogram*, is likely to have increased discriminative power.

Suppose $\mathcal{E}:\mathcal{I} \to \{0,1\}$ is the edge information of image $\mathcal{I}$, i.e., $\mathcal{E}(p) = 1$ if p is on an edge and 0 otherwise. (Such information can be obtained using various edge-detection algorithms.) Now, the question is if this useful information can be combined with (auto)correlograms so as to improve the retrieval quality even further. We outline one scheme to do this. In this scheme, each of the m color bins is refined to get $\mathcal{I}'$ with $2m$ bins.

$$\mathcal{I}'(p) = \begin{cases} c_+ & \text{if } \mathcal{I}(p) = c \text{ and } \mathcal{E}(p) = 1 \\ c_- & \text{if } \mathcal{I}(p) = c \text{ and } \mathcal{E}(p) = 0 \end{cases} \tag{18}$$

It is easy to see that the definition of both correlograms and autocorrelograms directly extend to this case. The storage requirements become $4m^2d$ (resp. $2md$) for correlograms (resp. autocorrelograms). Note however that the number of p such that $\mathcal{E}(p) = 1$ is usually very small. Since we mostly deal with autocorrelograms, the statistical importance of $\alpha_{c_+}^{(k)}$ becomes insignificant, thus rendering the whole operation meaningless. A solution to this problem is to define an *edge autocorrelogram* in which cross correlations between c_+ and c_- are also included. The size an edge autocorrelogram is thus only $4md$. We can further trim the storage by the banding technique in Section 3.5.1.

4. Image Retrieval Using Correlograms

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 $\text{rank}(\mathcal{I}) \in [|\mathcal{S}|]$ 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 metric defined on feature vectors.

Performance Measure. Let $\{\mathcal{Q}_1, \ldots, \mathcal{Q}_q\}$ be the set of query images. For a query $\mathcal{Q}_i$, let $\mathcal{I}_i$ be the unique correct answer. The following are two obvious performance measures:

1. *r-measure* of a method which sums up over all queries, the rank of the correct answer, i.e., $\sum_{i=1}^{q} \text{rank}(\mathcal{I}_i)$. 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 given by $\sum_{i=1}^{q} 1/\text{rank}(\mathcal{I}_i)$, 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.

3. *Recall* vs. *Scope*: Let $\mathcal{Q}$ be a query and let $\mathcal{Q}'_1$, $\ldots$, $\mathcal{Q}'_a$ be multiple "answers" to the query ($\mathcal{Q}$ is called a *category query*). Now, the *recall* r is defined for a *scope* $s > 0$ as $|\{\mathcal{Q}'_i \mid \text{rank}(\mathcal{Q}'_i) \le s\}|/a$. Since it is very hard to identify all relevant images in a huge database like ours, using this measure is much simpler than using the traditional *recall vs. precision*. Note however that this measure still evaluates the effectiveness of the retrieval (Hsu et al., 1995; Smith and Chang, 1996).

Organization. Section 4.1 lists some efficiency considerations we take into account while using correlograms for image retrieval. Section 4.2 describes our experimental setup and Section 4.3 provides the results of the experiments.

4.1. *Efficiency Considerations*

As image databases grow in size, retrieval systems need to address efficiency issues in addition to the issue of retrieval effectiveness. We investigate several general methods to improve the efficiency of indexing and searching, without compromising effectiveness.

Parallelization. The construction of a featurebase for an image database is readily parallelizable. We can divide the database into several parts, construct featurebases for these parts simultaneously on different processors, and finally combine them into a single featurebase for the entire database.

Partial Correlograms. In order to reduce space and time requirements, we choose a small value of d. This does not impair the quality of correlograms or autocorrelograms very much because in an image, local correlations between colors are more significant than global correlations. Sometimes, it is also preferable to work with *distance sets*, where a distance set D is a subset of $[d]$. We can thus cut down storage requirements, while still using a large d. Note that our algorithm can be modified to handle the case when $D \subset [d]$.

Though in theory the size of a correlogram is $O(m^2d)$ (and the size of an autocorrelogram is

$O(md)$), we observe that the feature vector is not always dense. This sparsity could be exploited to cut down storage and speed up computations.

Filtering. There is typically a tradeoff between the efficiency and effectiveness of search algorithms: more sophisticated methods which are computationally more expensive tend to yield better retrieval results. Good results can be obtained without sacrificing too much in terms of efficiency by adopting a two-pass approach (Hafner et al., 1995). In the first pass, we retrieve a set of N images in response to a query image by using an inexpensive (and possibly crude) search algorithm. Even though the ranking of these images could be unsatisfactory, we just need to guarantee that useful images are contained in this set. We can then use a more sophisticated matching technique to compare the query image to these N images only (instead of the entire database), and the best images are likely to be highly ranked in the resulting ranked list. It is important to choose an appropriate N in this approach[4] —the initially retrieved set should be good enough to contain the useful images and should be small enough so that the total retrieval time is reduced.

4.2. Experimental Setup

The image database consists of 14,554 color JPEG images of size 232×168. This includes 11,667 images used in Chabot (Ogle and Stonebraker, 1995), 1,440 images used in QBIC (Flickner et al., 1995), and 1,005 images from Corel. It also includes a few groups of images in PhotoCD format and a number of MPEG video frames from the web (Pass and Zabih, 1996). Our heterogeneous image database is thus very realistic and helps us evaluate various methods. It consists of images of animals, humans, landscapes, various objects like tanks, flags, etc.

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. We use $b = 4$ for the banded autocorrelogram. This results in a feature vector that is as small as a histogram.

Our query set consists of 77 queries. Each of these queries was manually picked and checked to have a *unique answer*. Therefore they serve as ground truth for us to compare different methods in a fair manner. In addition, the queries are chosen to represent various situations like different views of the same scene, large changes in appearance, small lighting changes, spatial translations, etc. We also run 4 category queries, each with $a > 1$ answers – Query 1 ($a = 22$ owl images), Query 2 ($a = 17$ fox images), Query 3 ($a = 6$ movie scenes), and Query 4 ($a = 6$ moving car images). The correct answers to the unique answer queries are obtained by an exhaustive manual search of the whole image database.

We use the L_1 norm for comparing feature vectors. The feature vectors we use are histograms (hist), color coherent vectors with successive refinement (ccv(s)) (Pass and Zabih, 1996), autocorrelograms (auto), banded autocorrelograms (b-auto), edge autocorrelograms (e-auto), and banded edge autocorrelograms (be-auto). Examples of some queries and answers (and the rankings according to various methods) are shown in Fig. 1. The query response time for autocorrelograms is under 2 sec on a Sparc-20 workstation (just by exhaustive linear search).

4.3. Results

4.3.1. Unique Answer Queries. Observe that all the correlogram-related methods are on par in terms of performance and significantly better than histogram and CCV(s). On average, in the autocorrelogram-based method, the correct answer shows up second while for histograms and CCV-based methods, the correct answer shows up at about 80th and 40th places. The banded autocorrelograms perform only slightly worse than the original ones. With the same data size as histograms, the banded autocorrelograms retrieve the correct answers more than 79 ranks lower than histograms. Since the autocorrelograms achieve strong retrieval results, the edge correlograms do not generate too much improvement.

Also note that the banded edge autocorrelograms have higher p_1-measure than the edge autocorrelograms. This is because most of the ranks go higher while only a few go lower. Though the r-measure becomes worse, the p_1-measure becomes better. It is remarkable that banded autocorrelogram has the same amount of information as the histogram, but seems lot more effective than the latter.

For 73 out of 77 queries, autocorrelograms perform as well as or better than histograms. In the cases where autocorrelograms perform better than color histograms, the average improvement in rank is 104 positions. In the four cases where color histograms perform

Table 1. Comparison of various image retrieval methods.

Method	Hist	CCV(s)	Auto	b-auto	e-auto	be-auto
r-Measure	6301	3272	172	196	144	157
Avg. r-measure	81.8	42.5	2.2	2.5	1.9	2.0
p_1-Measure	21.25	31.60	58.06	55.77	60.26	60.88
Avg. p_1-measure	0.28	0.41	0.75	0.72	0.78	0.79

better, the average improvement is just two positions. Autocorrelograms, however, still rank the correct answers within top six in these cases.

Statistical Significance Analysis. We adopt the approach used in (Pass and Zabih, 1996) to analyze the statistical significance of the improvements. We formulate the null hypothesis H_0 which states that the autocorrelogram method is as likely to cause a negative change in rank as a non-negative one. Under H_0, the expected number of negative changes is $M = 38.5$, with a standard deviation $\sigma = \sqrt{77}/2 \approx 4.39$. The actual number of negative changes is 4, which is less than $M - 7\sigma$. We can reject H_0 at more than 99.9% standard significance level.

For 67 out of 77 queries, autocorrelograms perform as well as or better than CCV(s). In the cases where autocorrelograms perform better than CCV(s), the average improvement in rank is 66 positions. In the ten cases where CCV(s) perform better, the average improvement is two positions. Autocorrelograms, however, still rank the correct answers within the top 12 in these cases. Again, statistical analysis suggets that autocorrelograms are better than CCV(s).

From a usability point of view, we make the following observation. Given a query, the user is guaranteed to locate the correct answer by just checking the top two search results (on average) in the case of autocorrelogram. On the other hand, the user needs to check at least the top 80 search results (on average) to locate the correct answer in the case of histogram (or top 40 search results for the CCV(s)). In practice, this implies that the former is a more "usable" image retrieval scheme than the latter two.

4.3.2. Recall Comparison. Table 2 shows the performance of three features on our four category queries. The L_1 distance metric is used. Once again, autocorrelograms perform the best.

4.3.3. Relative Distance Metric. Table 3 compares the results obtained using d_1 and L_1 distance measures on different features (64 colors). Using d_1 distance measure is clearly superior. The improvement is specially noticeable for histograms and CCV(s) (for instance, for the owl images in Fig. 5).

Table 2. Scope *vs.* recall results for category queries. (Larger numbers indicate better performance.)

	Recall		
Scope	Hist	CCV(s)	Auto
Query 1			
10	.14	.19	.24
30	.19	.19	.38
50	.19	.24	.57
Query 2			
10	.13	.19	.38
30	.31	.38	.63
50	.31	.38	.75
Query 3			
10	.20	.20	1.0
30	.40	.20	1.0
50	.40	.60	1.0
Query 4			
10	.20	.20	.60
30	.20	.20	.80
50	.20	.20	.80

A closer examination of the results shows, however, that there are instances where the d_1 distance measure performs poorly compared to the L_1 distance measure on histograms and CCV(s). An example is shown in Fig. 6.

It seems that the failure of the d_1 measure is related to the large change of overall image brightness (otherwise, the two images are almost identical). We need to examine such scenarios in greater detail. Autocorrelograms, however, are not affected by d_1 in this case. Nor does d_1 improve the performance of autocorrelogram much. In other words, autocorrelograms seem

Table 3. Comparison of L_1 and d_1.

	L_1 distance measure			d_1 distance measure		
Method	Hist	CCV(s)	Auto	Hist	CCV(s)	Auto
r-Measure	6301	3272	172	926	326	164
Avg. r-measure	82	42	2	12	4	2
p_1-Measure	21.25	31.60	58.03	47.94	52.09	59.92
Avg. p_1-measure	0.28	0.41	0.75	0.62	0.68	0.78

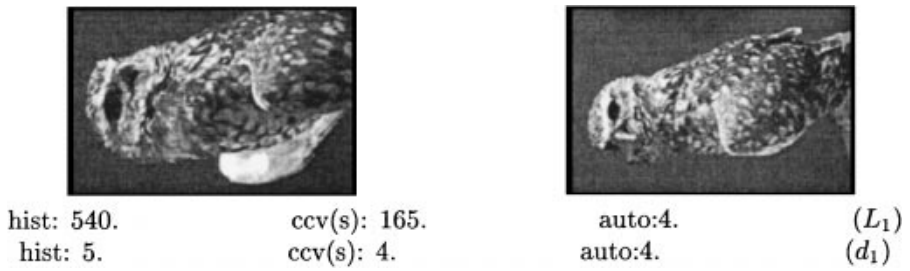

Figure 5. A case where d_1 is much better than L_1.

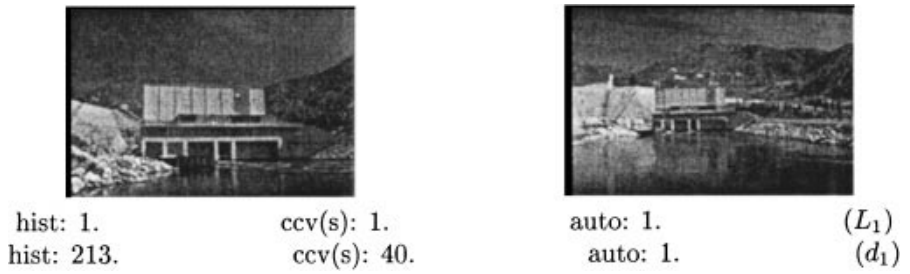

Figure 6. A case where d_1 is worse than L_1.

indifferent to the d_1 distance measure. This needs to be formally investigated.

4.3.4. Filtering. Table 4 shows the results of applying a histogram filter before using the autocorrelogram (we use 64-color histograms and autocorrelograms).

As we see, the quality of retrieval even improves somewhat (because false positives are eliminated). As anticipated, the query response time is less since we consider the correlograms of only a small filtered subset of the featurebase.

Table 4. Performance of auto(L_1) with hist(d_1) filter.

Method	Unfiltered	Filtered
r-Measure	172	166
p_1-Measure	58.02	58.60

4.3.5. Discussion. The results show that the autocorrelogram tolerates large changes in appearance of the same scene caused by changes in viewing positions, changes in the background scene, partial occlusions, camera zoom that causes radical changes in shape, etc. Since we chose small values {1, 3, 5, 7} for the distance set D, the autocorrelogram distills the global distribution of local color spatial correlations. In the case of camera zoom (for example, the third pair of images on the left column of Fig. 1), though there are big changes in object shapes, the local color spatial correlations as well as the global distribution of these correlation do not change that much. We illustrate this by looking at how the autocorrelation of yellow color changes with

Figure 7. The query image, the image ranked one, and the image ranked two.

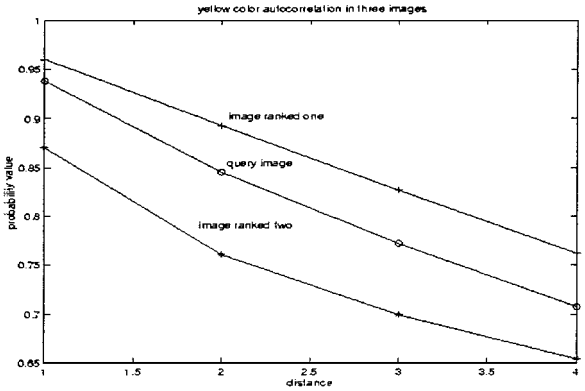

Figure 8. The change of autocorrelation of yellow color with distance.

distance in the following three images (Fig. 7). Notice that the size of yellow circular and rectangular objects in the query image and the image ranked one are different. Despite this, the correlation of yellow with yellow for the local distance of the image ranked one is closer to that of the query image than the image ranked, say, two (Fig. 8).

5. Image Subregion Querying Using Correlograms

The image subregion querying problem is the following: given as input a subregion query $\mathcal{Q}$ 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 different (such as changing viewpoint, camera noise and occlusion). The image subregion querying problem arises in image retrieval and in video browsing. For example, a user might wish to find other pictures in which a given object appears, or other scenes in a video with a given appearance of a person.

Performance Measure. We use the following measures to evaluate the performance of various competing image subregion querying algorithms. If $\mathcal{Q}_1, \ldots, \mathcal{Q}_q$ are the query images, and for the i-th query $\mathcal{Q}_i$, $\mathcal{I}_1^{(i)}, \ldots, \mathcal{I}_{a_i}^{(i)}$ are the only images that "contain" $\mathcal{Q}_i$, (i.e., $\mathcal{Q}_i$ "appears in" $\mathcal{I}_j^{(i)}$, $j = 1, \ldots, a_i$,) yet due to the presence of false matches the image subregion querying algorithm may return this set of "answers" with various ranks.

1. *Average r-measure* gives the mean rank of the answer-images averaged over all queries. It is given by either of the following expressions:

$$\frac{1}{q}\sum_{i=1}^{q}\frac{1}{a_i}\sum_{j=1}^{a_i}\operatorname{rank}\left(\mathcal{I}_j^{(i)}\right) \tag{19}$$

$$\left(\sum_{i=1}^{q}\sum_{j=1}^{a_i}\operatorname{rank}\left(\mathcal{I}_j^{(i)}\right)\right) \Bigg/ \sum_{i=1}^{q} a_i \tag{20}$$

   The *macroaveraged r*-measure given by Eq. (19) treats all queries with equal importance, whereas the *microaveraged r*-measure defined by Eq. (20) gives greater weightage to queries that have a larger number of answers. In both cases a lower value of the r-measure indicates better performance.
2. *Average precision* for a query $\mathcal{Q}_i$ is given by $(1/a_i)\sum_{j=1}^{a_i} j/\operatorname{rank}(\mathcal{I}_j^{(i)})$, where $\mathcal{I}_1^{(i)}, \ldots, \mathcal{I}_{a_i}^{(i)}$ are the answers for query $\mathcal{Q}_i$ in the order that they were retrieved. This quantity gives the average of the precision values over all recall points (with 1.0 being perfect performance).
3. *Recall/Precision vs. Scope*: For a query $\mathcal{Q}_i$ and a *scope* $s > 0$, the *recall* r is defined as $|\{\mathcal{I}_j^{(i)} \mid \operatorname{rank}(\mathcal{I}_j^{(i)}) \le s\}|/a_i$, and the *precision* p is defined as $|\{\mathcal{I}_j^{(i)} \mid \operatorname{rank}(\mathcal{I}_j^{(i)}) \le s\}|/s$. These measures are simpler than the traditional average precision measure

but still evaluate the effectiveness of the subregion query retrieval. For both measures, higher values indicate better performance.

Organization. Section 5.1 explains our approach to the problem. Section 5.2 describes the experimental setup and Section 5.3 presents the results.

5.1. Correlogram Intersection

The image subregion querying problem is a harder problem than image retrieval based on whole image matching. To avoid exhaustive searching subregions in an image, one scheme is to define *intersection* of color histograms (Swain and Ballard, 1991). The scheme can be interpreted in the following manner. (This interpretation helps us to generalize the method to correlograms easily.)

Given the histograms for a query $\mathcal{Q}$ and an image $\mathcal{I}$, the intersection of these two histograms can be considered as the histogram of an abstract entity notated as the intersection $\mathcal{Q} \cap \mathcal{I}$, (which will not be defined but serves as a conceptual and notational convenience only). With the color count of the intersection defined as

$$H_{c_i}(\mathcal{Q} \cap \mathcal{I}) \triangleq \min\{H_{c_i}(\mathcal{Q}), H_{c_i}(\mathcal{I})\}, \tag{21}$$

we can define the intersection of the histograms of $\mathcal{Q}$ and $\mathcal{I}$ as

$$h_{c_i}(\mathcal{Q} \cap \mathcal{I}) \triangleq \frac{H_{c_i}(\mathcal{Q} \cap \mathcal{I})}{|\mathcal{Q}|}. \tag{22}$$

Note that this definition is *not* symmetric in $\mathcal{Q}$ and $\mathcal{I}$. The distance $|\mathcal{Q} - \mathcal{Q} \cap \mathcal{I}|_{h,L_1}$ is a measure of the presence of $\mathcal{Q}$ in $\mathcal{I}$. When $\mathcal{Q}$ is a subset of $\mathcal{I}$, $|\mathcal{Q} - \mathcal{Q} \cap \mathcal{I}|_{h,L_1} = 0$ because all the color counts in $\mathcal{Q}$ are less than those in $\mathcal{I}$, and the histogram intersection simply gives back the histogram for $\mathcal{Q}$.

In an analogous manner, we define the intersection correlogram as the correlogram of the intersection $\mathcal{Q} \cap \mathcal{I}$ (again, merely an abstract entity.) With the count

$$\Gamma^{(k)}_{c_i,c_j}(\mathcal{Q} \cap \mathcal{I}) \triangleq \min\{\Gamma^{(k)}_{c_i,c_j}(\mathcal{Q}), \Gamma^{(k)}_{c_i,c_j}(\mathcal{I})\}, \tag{23}$$

we can define the intersection correlogram as follows:

$$\gamma^{(k)}_{c_i,c_j}(\mathcal{Q} \cap \mathcal{I}) \triangleq \frac{\Gamma^{(k)}_{c_i,c_j}(\mathcal{Q} \cap \mathcal{I})}{H_{c_i}(\mathcal{Q} \cap \mathcal{I}) \cdot 8k}. \tag{24}$$

Again we measure the presence of $\mathcal{Q}$ in $\mathcal{I}$ by the distance $|\mathcal{Q} - \mathcal{Q} \cap \mathcal{I}|_{\gamma,L_1}$, say if L_1 distance measure were chosen. If $\mathcal{Q} \subseteq \mathcal{I}$, then the latter "should have at least as many counts of correlating color pairs" as the former. Thus the counts Γ and H for $\mathcal{Q} \cap \mathcal{I}$ are again those of $\mathcal{Q}$, and the correlogram of $\mathcal{Q} \cap \mathcal{I}$ becomes exactly the correlogram of $\mathcal{Q}$, giving $|\mathcal{Q} - \mathcal{Q} \cap \mathcal{I}|_{\gamma,L_1} = 0$.

We see that the distance between $\mathcal{Q}$ and $\mathcal{Q} \cap \mathcal{I}$ vanishes when $\mathcal{Q}$ is actually a subset of $\mathcal{I}$; this affirms the fact that both correlograms and histograms are global features. Such a *stable* property is not satisfied by all features—for instance, spatial coherence (Pass and Zabih, 1996) is not preserved under subset operations. Therefore, methods for subregion querying based on such unstable features are not likely to perform as good as the histogram or correlogram based methods.

5.2. Experiments

The image database is the same as in Section 4.2. We use 64 color bins for histograms and autocorrelograms. The distance set for autocorrelograms is $D = \{1, 3, 5, 7\}$. Our query set for this task consists of 30 queries. Queries have 2 to 16 answer images with the average number of answers per query being nearly 5. The query set is constructed by selecting "interesting" portions of images from the image database. The answer images contain the object depicted in the query, but often with its appearance significantly changed due to changes in viewpoint, or different lighting, etc. Examples of some queries and answers (and the rankings according to the histogram and autocorrelogram intersection methods) are shown in Fig. 9.

5.3. Results

The histogram and autocorrelogram intersection methods for subregion querying are compared in Tables 5 and 6. For each of the evaluation measures proposed above, the autocorrelogram performs better. The average rank of the answer images improves by over 30 positions when the autocorrelogram method is used, and the average precision figure improves by an impressive 56% (see Table 5). Table 6 shows the precision and recall values for the two methods at various scopes. Once again, autocorrelograms perform consistently better than histograms at all scopes. Doing a query-by-query analysis, we find that autocorrelograms do better in terms of the average r-measure on 23 out of the 30 queries. Similarly, autocorrelograms

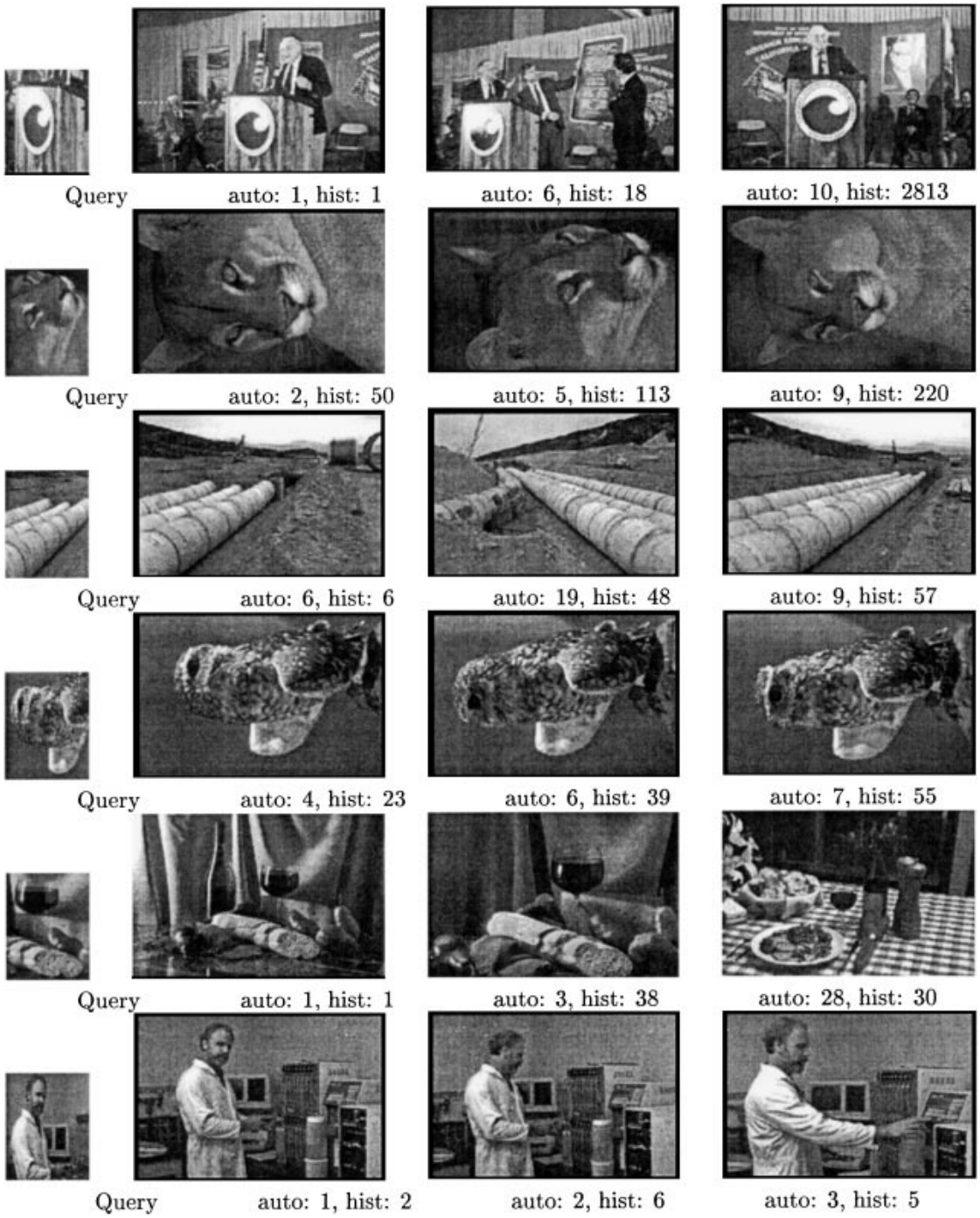

Figure 9. Sample queries and answer sets with ranks for various methods. (Lower ranks are better.)

Table 5. Performance of Histogram and Autocorrelogram Intersection methods—I. (Smaller r-measure and larger precision are better.)

Method	Avg. r-measure (macro)	Avg. r-measure (micro)	Avg. precision
Hist	56.3	61.3	0.386
Auto	22.5	29.1	0.602

Table 6. Performance of Histogram and Autocorrelogram Intersection methods—II. (Larger values are better.)

	Method			
	Precision (%)		Recall	
Scope	Hist	Auto	Hist	Auto
5	0.273	0.493	0.311	0.541
10	0.223	0.347	0.460	0.718
25	0.133	0.165	0.681	0.850

yield better average precision on 26 out of 30 queries. Thus, for a variety of performance metrics, autocorrelograms yield better results. This suggests that the autocorrelogram is a superior method for subregion querying problem.

6. Other Applications of Correlograms

6.1. Localization Using Correlograms

The location problem is the following: given a query image (also referred to as the target or model) $\mathcal{Q}$ and an image $\mathcal{I}$ such that $\mathcal{Q} \subseteq \mathcal{I}$, find the "location" in $\mathcal{I}$ where $\mathcal{Q}$ is "present". It is hard to define the notion of location mathematically because the model is of some size. We use the location of the center of the model for convenience.

This problem arises in tasks such as real-time *object tracking* or *video searching*, where it is necessary to localize the position of an object in an image. Given an algorithm that solves the location problem, tracking an object $\mathcal{Q}$ in an image frame sequence $\vec{\mathcal{I}} = \mathcal{I}_1, \ldots, \mathcal{I}_t$ is equivalent to finding the location of $\mathcal{Q}$ in each of the $\mathcal{I}_i$'s. Efficiency is also required in this task because huge amounts of data need to be processed.

To avoid exhaustive searching in the whole image (template matching is of such kind), histogram backprojection was proposed to handle the location problem efficiently. In the following, we study the histogram backprojection algorithm first. Then we show how the correlogram can be used to improve the performance.

The location problem can be viewed as a special case of the image retrieval problem in the following manner. Let $\mathcal{I}|_p$ denote the subimage in $\mathcal{I}$ of size $\mathcal{Q}$ located at position p. (The assumption about the size of the subimage is without loss of generality.) The set of all subimages $\mathcal{I}|_1, \ldots, \mathcal{I}|_{|\mathcal{I}|}$ present in $\mathcal{I}$ constitutes the image database and $\mathcal{Q}$ is the query image. The solution $\mathcal{I}|_p$ to this retrieval problem gives p, the location of $\mathcal{Q}$ in $\mathcal{I}$. The above interpretation is a *template matching* process. One straightforward approach to the location problem is *template matching*. Template matching takes the query $\mathcal{Q}$ as a template and moves this template over all possible locations in the image $\mathcal{I}$ to find the best match. This method is likely to yield good results, but is computationally expensive. Attempts have been made to make template matching more efficient (Margalit and Rosenfeld, 1990; Vinod et al., 1996; Brock-Gunn and Ellis, 1992). The *histogram backprojection* method is one such approach to this problem. This method has some serious drawbacks, however. In the following, we explain the problem with the histogram backprojection scheme.

The basic idea behind histogram backprojection is (1) to compute a "goodness value" for each pixel in $\mathcal{I}$ (the goodness of each pixel is the likelihood that this pixel is in the target); and (2) obtain the subimage (and hence the location) whose pixels have the highest goodness values.

Formally, the method can be described as follows. The *ratio histogram* is defined for a color c as

$$\pi_{c,h}(\mathcal{I} \mid \mathcal{Q}) \triangleq \min\left\{\frac{H_c(\mathcal{Q})}{H_c(\mathcal{I})}, 1\right\}. \tag{25}$$

The goodness of a pixel $p \in \mathcal{I}_c$ is defined to be $\pi_{c,h}(\mathcal{I} \mid \mathcal{Q})$. The contribution of a subimage $\mathcal{I}|_p$ is given by

$$\Pi_p(\mathcal{I} \mid \mathcal{Q}) \triangleq \sum_{q \in \mathcal{I}|_p} \pi_{\mathcal{I}(q),h}(\mathcal{I} \mid \mathcal{Q}). \tag{26}$$

Then, the location of the model is given by

$$\arg\max_{p \in \mathcal{I}} \Pi_p(\mathcal{I} \mid \mathcal{Q}). \tag{27}$$

The above method generally works well in practice, and is insensitive to changes of image resolution or histogram resolution (Swain and Ballard, 1991). Note

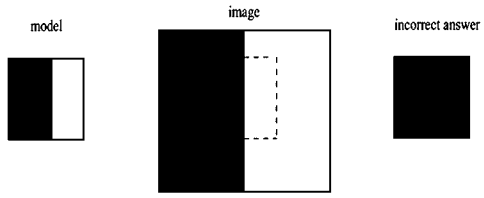

Figure 10. A false match for histogram-backprojection.

that backprojecting the ratio-histogram gives the *same* goodness value to all pixels of the same color. It emphasizes colors that appear frequently in the query but not too frequently in the image. This could result in overemphasizing certain colors in $\mathcal{Q}$. A color c is said to be *dominant* in $\mathcal{Q}$, if $\pi_{c,h}(\mathcal{I} \mid \mathcal{Q})$ is maximum over all colors. If $\mathcal{I}$ has a subimage $\mathcal{I}|_p$ (which may be totally unrelated to $\mathcal{Q}$) that has many pixels of color c, then this method tends to identify $\mathcal{Q}$ with $\mathcal{I}|_p$, thus causing an error in some cases.

Figure 10 shows a simple example illustrating this problem. Suppose $\mathcal{Q}$ has 6 black pixels and 4 white pixels, and image $\mathcal{I}$ has 100 black pixels and 100 white pixels. Then $\pi_{b,h} = 0.06$, $\pi_{w,h} = 0.04$. The location of the model according to the backprojection method is in an entirely black patch, which is clearly wrong.

Another problem with histogram backprojection is inherited from histograms which have no spatial information. Pixels of the same color have the same goodness value irrespective of their position. Thus, false matches occur easily when there are multiple similarly colored objects, as shown in the examples of roses and zebras in Fig. 11.

Performance Measure. Let an indicator variable $\mathrm{loc}(\mathcal{Q}, \mathcal{I})$ be 1 if the location returned by a method is within reasonable tolerance of the actual location of $\mathcal{Q}$ in $\mathcal{I}$. Then, given a series of queries $\mathcal{Q}_1, \ldots, \mathcal{Q}_q$ and corresponding images $\mathcal{I}_1, \ldots, \mathcal{I}_q$, the *success ratio* of the method is given by

$$\frac{\sum_{i=1}^{q} \mathrm{loc}(\mathcal{Q}_i, \mathcal{I}_i)}{q} \tag{28}$$

For tracking an object $\mathcal{Q}$ in a sequence of frames $\vec{\mathcal{I}} = \mathcal{I}_1, \ldots, \mathcal{I}_t$, the *success ratio* is therefore $\sum_{i=1}^{t} \mathrm{loc}(\mathcal{Q}, \mathcal{I}_i)/t$.

Organization. Section 6.1.1 introduces the correlogram correction for the location problem. Section 6.1.2 contains the experiments and results.

6.1.1. Correlogram Correction. To alleviate the problems with histogram backprojection, we incorporate local spatial correlation information by using a *correlogram correction* factor in Eq. (26). The idea is to integrate discriminating local characteristics while avoiding local color template matching (Ennesser and Medioni, 1995). We define a *local correlogram contribution* based on the autocorrelogram of the subimage $\mathcal{I}|_p$ so that the goodness of a pixel depends on its position in addition to its color.

$\alpha_c^{(k)}(\mathcal{Q})$ is considered to be the average contribution of pixel of color c in $\mathcal{Q}$ (for each distance k).

For each pixel $p \in \mathcal{I}$, the local autocorrelogram $\alpha_p^{(k)}$ is computed for each distance $k \in D$ (D should contain only small values so that $\alpha_p^{(k)}$ captures local information for each p)

$$\alpha_p^{(k)} = \frac{\Gamma^{(k)}_{\mathcal{I}(p),\mathcal{I}(p)}(\{p\})}{8k} \tag{29}$$

where $\{p\}$ represents the pixel p along with its neighbors considered as an image. Now, the correlogram contribution of p is defined as

$$\pi_{p,\gamma}(\mathcal{I} \mid \mathcal{Q}) \triangleq \left|\mathcal{Q}_{\mathcal{I}(p)} - \{p\}\right|_{\gamma,L_1} \tag{30}$$

In words, the contribution of p is the L_1-distance between the local autocorrelogram at p and the part of the autocorrelogram for $\mathcal{Q}$ that corresponds to the color of p.

Combining this contribution with Eq. (26), the final goodness value of a subimage $\mathcal{I}|_p$ is given by

$$\Pi_p(\mathcal{I} \mid \mathcal{Q}) \triangleq \sum_{q \in \mathcal{I}|_p} \big(\beta \pi_{\mathcal{I}(q),h}(\mathcal{I} \mid \mathcal{Q}) + (1-\beta)\pi_{q,\gamma}(\mathcal{I} \mid \mathcal{Q})\big). \tag{31}$$

where $0 \le \beta \le 1$.

It turns out that the correlogram contribution by itself is also sensitive and occasionally overemphasizes less dominant colors. Suppose c is a less dominant color (say, the background color) that has a high autocorrelation. If $\mathcal{I}$ has a subimage $\mathcal{I}|_p$ (which may be totally irrelevant to $\mathcal{Q}$) that has many pixels of color c with high autocorrelations, then correlogram backprojection has a tendency to identify $\mathcal{Q}$ with $\mathcal{I}|_p$, thus potentially causing an error. Since the problems with histograms and correlograms are in some sense complementary to each other, the best results are obtained when the

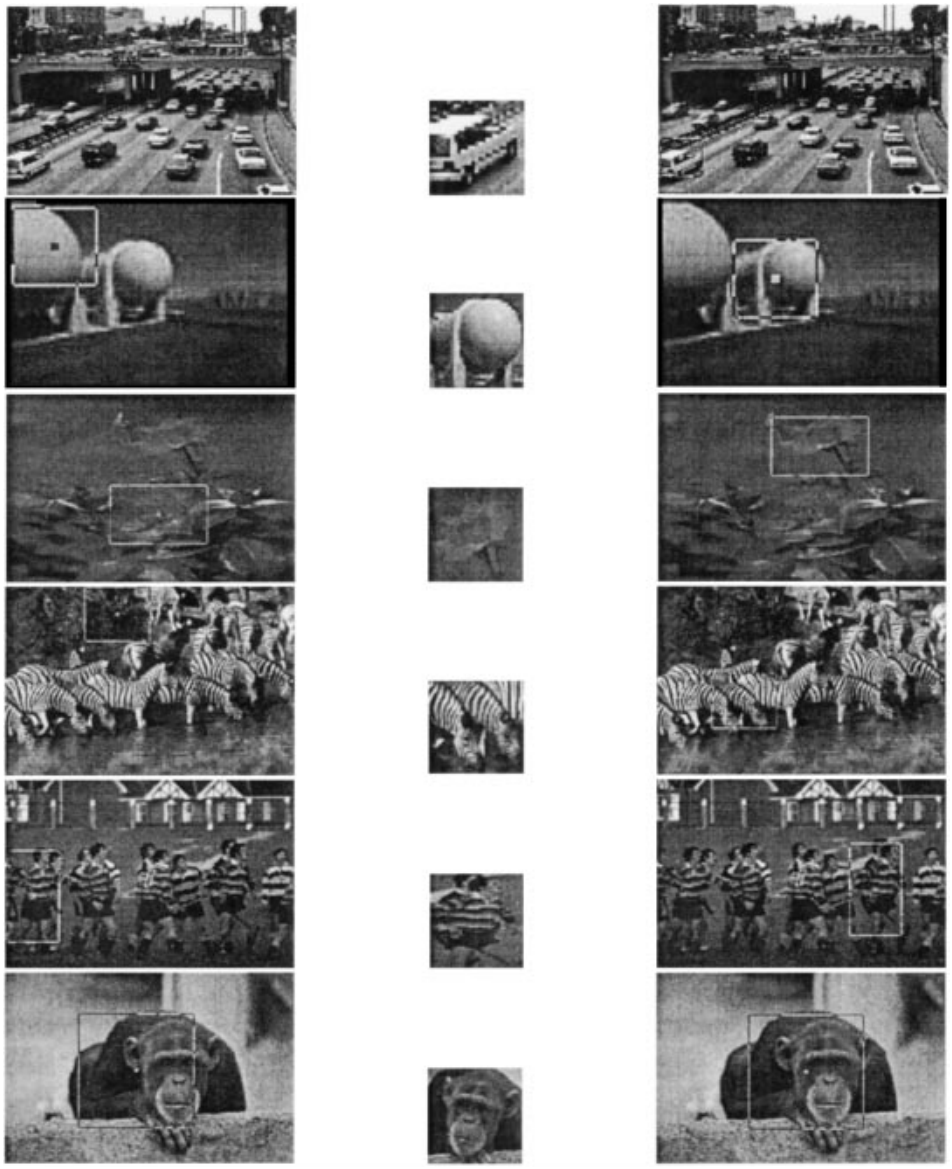

Figure 11. Location problem: histogram output, query image, and correlogram output.

goodness of a pixel is given by a weighted linear combination of the histogram and correlogram backprojection contributions—adding the local correlogram contribution to histogram backprojection remedies the problem that histograms do not take into account any local information; the histogram contribution ensures that background colors are not overemphasized. We call this *correlogram correction*.

This can also be understood by drawing an analogy between this approach and the Taylor expansion. The goodness value obtained from histogram backprojection is like the average constant value in the Taylor expansion; the local correlogram contribution is like the first order term in the approximation. Therefore, the best results are obtained when the goodness value of a pixel is a weighted linear combination of the histogram backprojection value and the correlogram contribution.

6.1.2. Experiments and Results. We use the same database to perform the location experiments. A model image and an image that contains the model are chosen.

Table 7. Results for location problem (66 queries).

Method	Hist	Auto
Success ratio	0.78	0.96

Table 8. Results (success ratios) for the tracking problem.

Method	Hist	Auto
Bus	0.93	0.99
Clapton	0.44	0.78
Skydive	0.96	0.96

For the location problem, 66 query images and 52 images that contain these models are chosen and tested. Both the histogram backprojection and autocorrelogram correction are tried. $D = \{1, 3, 5\}$ and $\beta = 0.5$ were the parameters.

For the tracking problem, we choose three videos bus (133 frames), clapton (44 frames), skydive (85 frames). We use $D = \{1, 3, 5\}$ and $\beta = 0.8$ for this problem.

For the location problem, Table 7 shows the results for 66 queries, and Fig. 11 shows some examples.

For the tracking problem, Table 8 shows the result of histogram backprojection and correlogram correction for the three test videos. These results clearly show that correlogram correction alleviates many of the problems associated with simple histogram backprojection.

Figure 12 shows sample outputs.

6.2. Cut Detection Using Correlograms

The increasing amount of video data requires automated video analysis. The first step to the automated video content analysis is to segment a video into camera shots (also known as *key frame extraction*). A camera shot $\vec{\mathcal{I}} = \mathcal{I}_1, \ldots, \mathcal{I}_t$ is an unbroken sequence of frames from one camera. If $\vec{\mathcal{J}}$ denotes the sequence of cuts, then a cut $\mathcal{J}_j$ occurs when two consecutive frames $\langle \mathcal{I}_i, \mathcal{I}_{i+1} \rangle$ are from different shots.

Cut detection algorithms assume that consecutive frames in a same shot are somewhat more *similar* than frames in a different shot (other gradual transition, such as fade and dissolve, are not studied here because certain mathematical models can be used to treat these chromatic editing effects). Different cut detectors use different features to compare the similarity between two consecutive frames, such as pixel difference, statistical differences, histogram comparisons, edge differences, etc. (Boresczky and Rowe, 1996). One way to detect cuts using a feature f is by ranking $\langle \mathcal{I}_i, \mathcal{I}_{i+1} \rangle$ according to $|\mathcal{I}_i - \mathcal{I}_{i+1}|_f$. Let cuts($\vec{\mathcal{I}}$) be the number of actual cuts in $\vec{I}$ and rank($\mathcal{J}_i$) be the rank of the cut $\mathcal{J}_i$ according to this ranking.

Histograms are the most common used image features to detect cuts because they are efficient to compute and insensitive to camera motions. Histograms, however, are not robust to local changes in images that false positives easily occurs in this case (see Fig. 13). Since correlograms have been shown to be robust to large appearance changes for image retrieval, we use correlograms for cut detection.

Performance Measure. *Recall* and *Precision* are usually used to compare the performance of cut detection. However, it is difficult to measure the performance of different algorithms based on recall vs. precision curves (Boresczky and Rowe, 1996). Therefore we look at recall and precision values separately. In order to avoid using "optimal" threshold values, we use *precision vs. scope* to measure false positives and *recall vs.*

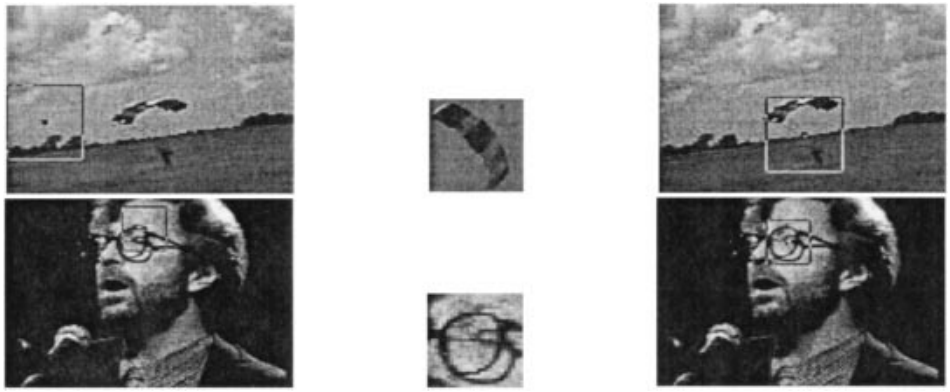

Figure 12. Tracking problem: histogram output, query image, and correlogram output.

Figure 13. Cut detection: False cuts detected by histogram but not by correlogram.

scope to measure false negatives (misses). We choose scope values to be the exact cut number cuts($\vec{I}$) and 2 cuts($\vec{I}$). We also use the *excessive rank value*, which is defined by

$$\sum_{i=1}^{\text{cuts}(\vec{I})} (\text{rank}(\mathcal{J}_i) - i)\,, \tag{32}$$

and the *average precision value* which is defined by

$$\frac{1}{\text{cuts}(\vec{I})} \sum_{i=1}^{\text{cuts}(\vec{I})} \frac{i}{\text{rank}(\mathcal{J}_i)}. \tag{33}$$

Note that a smaller excessive rank value and a larger average precision value indicate better result (perfect performance would have values 0 and 1 respectively).

6.2.1. Experiments and Results. We use 64 colors for histograms, and banded autocorrelograms which have the same size as histograms. We use 5 video clips from television, movies, and commercials. The clips are diverse enough to capture different kinds of common scenarios that occur in practice. The results are shown in Table 9 and Table 10.

The results of our experiments show that banded autocorrelograms are more effective than histograms while the two have the same amount of information. It is certainly more efficient than dividing an image into 16 subimages (Hampapur et al., 1994). Thus the autocorrelogram is a promising tool for cut detection.

7. Conclusions

In this paper, we introduced the color correlogram—a new image feature—for solving several problems that arise in content-based image retrieval and video browsing. The novelty in this feature is the characterization of images in terms of the spatial correlation of colors instead of merely the colors *per se*. Experimental evidence suggests that this information discriminates between "different" images and identifies "similar" images very well. We show that correlograms can be computed, processed, and stored at almost no extra

Table 9. Recall vs. Scope for cut detection. (Smaller values are better.)

	Method					
	Hist			Banded-auto		
Video	Cuts($\vec{\mathcal{I}}$)	2 Cuts($\vec{\mathcal{I}}$)	Ex. rank value	Cuts($\vec{\mathcal{I}}$)	2 Cuts($\vec{\mathcal{I}}$)	Ex. rank value
1 (9 Cuts)	0.78	1.0	14	1.0	1.0	0
2 (15 Cuts)	0.87	1.0	7	1.0	1.0	0
3 (16 Cuts)	0.75	1.0	25	0.81	1.0	9
4 (7 Cuts)	1.0	1.0	0	1.0	1.0	0
5 (10 Cuts)	0.9	0.9	>10	0.9	1.0	3

Table 10. Precision vs. Scope for cut detection. (Larger values are better.)

	Method					
	Hist			Banded-auto		
Video	Cuts($\vec{\mathcal{I}}$)	2 Cuts($\vec{\mathcal{I}}$)	Agv. prec. value	Cuts($\vec{\mathcal{I}}$)	2 Cuts($\vec{\mathcal{I}}$)	Agv. prec. value
1 (9 Cuts)	0.78	0.5	0.90	1.0	0.5	1.0
2 (15 Cuts)	0.87	0.5	0.98	1.0	0.5	1.0
3 (16 Cuts)	0.75	0.5	0.93	0.81	0.5	0.97
4 (7 Cuts)	1.0	0.5	1.0	1.0	0.5	1.0
5 (10 Cuts)	0.8	0.45	$<$0.95	0.9	0.5	0.98

cost compared to competing methods, thereby justifying using this instead of many other features to get better image retrieval quality.

The most important application of correlograms is to content-based image retrieval (CBIR) systems. Viewed in this context, a correlogram is neither a region-based nor a histogram-based method. Unlike purely local properties, such as pixel position, and gradient direction, or purely global properties, such as color distribution, a correlogram takes 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 is likely to be sensitive to large appearance changes, correlograms are more stable to tolerate these changes and while any scheme that is based on purely global properties is susceptible to false positive matches, correlograms seem to be scalable for CBIR. This is corroborated by our extensive experiments on large image collections, where we demonstrate that correlograms are very promising for CBIR.

One issue that still needs to be resolved satisfactorily is the following: in general, illumination changes are very hard to handle in color-based CBIR systems (Gong, 1998; Syeda-Mahmood and Cheng, 1996; Funt and Finlayson, 1995; Slater and Healey, 1995). During our experiments, we encountered this problem occasionally. Though the correlogram method performs better on a relative scale, its absolute performance is not fully satisfactory. The question is, can correlograms, with some additional embellishments, be made to address this specific problem?

On a related note, it also remains to be seen if correlograms, in conjunction with other features, can enhance retrieval performance. For instance, how will the correlogram perform if shape information is used additionally? This brings up the question of object-level retrieval using correlograms. More work needs to be done in this regard as to finding a better representation for objects.

Further applications of color correlograms are image subregion querying and localization, which are indispensable features of any image management system. Our notions of correlogram intersection and correlogram correction seem to perform well in practice. There is room for improvement of course, and these need to be investigated in greater detail. We also apply correlograms to the problem of detecting cuts in video sequences. An interesting question that arises here is, can this operation be done in the compressed domain (Yeo and Liu, 1995)? This would cut down the computation time drastically and make real-time processing feasible.

Another major challenge in this context is: what distance metric for comparing images is close to the human perception of similarity? Does a measure need to be a metric (Jacobs et al., 1998)? We also plan to use supervised learning to improve the results of image retrieval and the subregion querying task (we have some initial results in (Huang et al., 1997)).

In general, the algorithms we propose for various problems are not only very simple and inexpensive but are especially easy to incorporate into a CBIR system if the underlying indexing scheme is correlogram-based. It pays off more in general if there is a uniform feature vector that is universally applicable to providing various functionalities expected of a CBIR system (like histograms advocated in (Swain and Ballard, 1991)). It is unreasonable to expect any CBIR system to be absolutely fool-proof; furthermore, it is needless to state that the correlogram is not a panacea. The goal, however, is to build better CBIR systems. Based on various

experiments, we feel that there is a compelling reason to use correlograms as one of the basic building blocks in such systems.

Acknowledgments

Jing Huang and Ramin Zabih were supported by DARPA grant DAAL 01-97-K-0104, by NSF grant EIA 97-03470, and by a grant from Microsoft. S Ravi Kumar was supported by the ONR Young investigator award N00014-93-1-0590, the NSF grant DMI-91157199, and the career grant CCR-9624552. Mandar Mitra was supported by the NSF grant IRI 96-24639. Wei-Jing Zhu was supported by the DOE grant DEFG02-89ER45405.

Notes

1. In our database of 14,554 images, the right image is considered the 353-rd most similar with respect to the left image by color histogram.
2. The term "correlogram" is adapted from spatial data analysis: "correlograms are graphs (or tables) that show how spatial autocorrelation changes with distance." (Upton and Fingleton, 1985)
3. Interestingly, histogram or CCV's may not be able to distinguish between these two images.
4. Equivalently, we could select some threshold image score.

References

Boreszcky, J.S. and Rowe, L.A. 1996. A comparison of video shot boundary detection techniques. In *Storage & Retrieval for Image and Video Databases IV*, *Proc. SPIE 2670*, pp. 170–179.

Brock-Gunn, S.A. and Ellis, T.J. 1992. Using color templates for target identification and tracking. In *Proc. British Machine Vision Conference*, pp. 207–216.

1995. Content-based image retrieval systems, *IEEE Computer*.

Cox, I.J., Matt L. Miller, Stephen M. Omohundro, and Peter N. Yianilas, 1996. PicHunter: Bayesian relevance feedback for image retrieval. In *Intl. Conf. on Pattern Recognition*, Vienna, Austria.

Ennesser, F. and Medioni, G. 1995. Finding waldo, or focus of attention using local color information. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(8).

Enser, P.G.B. 1993. Query analysis in a visual information retrieval context. *J. Document and Text Management*, 1:25–52.

Fleck, M.M., Forsyth, D.A., and Bregler, C. 1996. Finding naked people. In *European Conf. on Computer Vision*, Vol. 2, pp. 590–602.

Flickner, M. *et al.* 1995. Query by image and video content: The QBIC system. *IEEE Computer*, 28(9):23–32.

Forsyth, D.A. *et al.* 1996. Finding pictures of objects in large collections of images. In *Proc. Intl. Workshop on Object Recognition*, Cambridge.

Funt, B. and Finlayson, G. 1995. Color constant color indexing. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17:522–529.

Gong, Y. *et al.* 1994. An image database system with content capturing and fast image indexing abilities. In *Intl. Conf. on Multimedia Comp & Systems*, pp. 121–130.

Gong, Y. 1998. *Intelligent Image Databases: Towards Advanced Image Retrieval*. Kluwer Academic Publishers.

Grimson, W.L. and Lozano-Pérez, T. 1987. Localising overlapping parts by searching the interpretation tree. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 9:469–482.

Hafner, J., Sawhney, H., Equitz, W., Flickner, M., and Niblack, W. 1995. Efficient color histogram indexing for quadratic form distance functions. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(7):729–736.

Hampapur, A., Jain, R., and Weymouth, T. 1994. Digital video indexing in multimedia systems. In *Proc. AAAI-94 Workshop on Indexing and Reuse in Multimedia Systems*.

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

Haussler, D. 1992. Decision theoretic generalization of the PAC model for neural net and other learning applications. *Information and Computation*, 100:78–150.

Hsu, W., Chua, T.S., and Pung, H.K. 1995. An integrated color-spatial approach to content-based image retrieval. In *Proc. 3rd ACM Multimedia Conf.*, pp. 305–313.

Huang, J., Kumar, S.R., and Mitra, M. 1997. Combining supervised learning with color correlograms for content-based image retrieval. In *Proc. 5th ACM Multimedia Conf.*, pp. 325–334.

Huang, J., Kumar, S.R., Mitra, M., Zhu, W.J., and Zabih, R. 1997. Image indexing using color correlograms. In *Proc. 16th IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 762–768.

Huang, J., Kumar, S.R., Mitra, M., and Zhu, W.J. 1998. Spatial color indexing and applications. In *Proc. 8th Intl. Conf. on Computer Vision*.

Huttenlocher, D.P. and Ullman, S. 1986. Object recognition using alignment. In *Proc. Intl. Conf. on Computer Vision*, pp. 102–111.

Huttenlocher, D.P., Klanderman, G.A., and Rucklidge, W.J. 1993. Comparing images using the Hausdorff distance. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15:850–863.

Huttenlocher, D.P., Lilien, R.H., and Olson, C.F. 1996. Object recognition using subspace methods. In *Proc. European Conf. on Computer Vision*, pp. 536–545.

Jacobs, D.W., Weinshall, D., and Gdalyahu, Y. 1998. Condensing image databases when retrieval is based on non-metric distances. In *Proc. Intl. Conf. on Computer Vision*.

Marr, D. and Nishihara, H.K. 1978. Representation and recognition of the spatial organization of three-dimensional shapes. *Proc. Royal Soc. Lond. B.*, 200:269–294.

Margalit, A. and Rosenfeld, A. 1990. Using probabilistic domain knowledge to reduce the expected computational cost of template matching. *Computer Vision*, *Graphics*, *and Image Processing*, 51:219–234.

Matas, J., Marik, R., and Kittler, J. 1995. On representation and matching of multi-colored objects. In *Proc. IEEE 5th Intl. Conf. on Computer Vision*, pp. 726–732.

Murase, H. and Nayar, S.K. 1995. Visual learning and recognition of 3-D objects from appearance. *Intl. Journal of Computer Vision*, 14:5–24.

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

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

Pass, G. and Zabih, R. 1999. Comparing images using joint histograms. In *Journal of Multimedia Systems*, 7(3):234–240.

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

Rao, R.P. and Ballard, D. 1995. Object indexing using an iconic sparse distributed memory. In *Proc. IEEE 5th Intl. Conf. on Computer Vision*, pp. 24–31.

Rickman, R. and Stonham, J. 1996. Content-based image retrieval using color tuple histograms. *SPIE Proc.*, 2670:2–7.

Roberts, L.G. 1965. Machine perception of three-dimensional solids. In *Optical and Electro-Optical Information Processing*. MIT Press.

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

Slater, D. and Healey, G. 1995. Combining color and geometric information for the illumination invariant recognition of 3-D objects. In *Proc. IEEE 5th Intl. Conf. on Computer Vision*, pp. 563–568.

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

Stricker, M. and Swain, M. 1994. The capacity of color histogram indexing. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 704–708.

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

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

Syeda-Mahmood, T.F. 1997. Data and model-driven selection using color regions. In *Intl. Journal of Computer Vision*, 21(1/2): 9–36.

Syeda-Mahmood, T.F. and Cheng, Y.-Q. 1996. Indexing colored surfaces in images. In *Intl. Conf. on Pattern Recognition*.

Upton, G.J. and Fingleton, B. 1985. *Spatial Data Analysis by Example*. Vol. 1. John Wiley & Sons.

Vinod, V.V., Murase, H., and Hashizume, C. 1996. Focused color intersection with efficient searching for object detection and image retrieval. *IEEE Proc. Multimedia*, pp. 229–233.

Wan, X. and Jay Kuo, C.-C. 1996. Color distribution analysis and quantization for image retrieval. *SPIE Proc.*, 2670:8–16.

Yeo, B.-L. and Liu, B. 1995. Rapid scene analysis on compressed videos. In *IEEE Trans. Circuits Syst. Video Technology*, 5, 6: 533–544.