Spatial Color Indexing and Applications

Jing Huang [*] S Ravi Kumar [†] Mandar Mitra[‡] Wei-Jing Zhu[§]
Cornell University
Ithaca, NY 14853.

Abstract

We suggest the use of the color correlogram as a generic indexing tool to tackle various computer vision problems. Correlograms were shown to be very effective for content-based image retrieval [4]. We adapt the correlogram to handle the problems of image subregion querying, object localization, object tracking, and cut detection. Experimental results suggest that the color correlogram is much more effective than the histogram for these applications, with insignificant additional computational, storage, or processing cost. We also provide a technique to cut down the storage requirement of correlograms so that it is the same as that of histograms, with only negligible performance penalty compared to the original correlogram.

1 Introduction

The recent explosion in Internet usage and the 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, 7, 9, 8]. 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.

The general problem of content-based image retrieval has been widely studied [3, 7, 9, 8]. The color correlogram was recently proposed as an image feature vector in [4] and was shown to be very effective and efficient for content-based image retrieval.

While most image retrieval systems retrieve images based on overall image comparison, users are typically interested in *object searching*, where 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 a wide variety of effects (e.g., different viewing positions, camera noise, 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, i.e. it should find the location of the object in an image. The lack of a good image segmentation process suitable for large, heterogeneous image databases implies that objects have to be located in unsegmented images, making the localization problem 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 color histogram is a general and flexible tool that can be used for the various tasks described above [14, 1]. In this study, we investigate the applicability of *color correlograms* to these tasks, and show that correlograms also constitute a versatile tool that can be used for a variety of problems. Our preliminary results indicate that the correlogram method is a more accurate and effective approach to these tasks compared to the color histogram method. 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. We believe, therefore, that correlograms form a significantly superior but equally economical alternative to the histogram.

Organization. We propose extensions to the correlogram that improve its already good performance and efficiency (Section 2). In Section 3, we propose the *correlogram intersection* method for image subregion querying. The histogram-backprojection approach used for the localization problem in [14] has serious drawbacks. In Section 4 we discuss these disadvantages and show how *local correlogram contribution* can be used to locate objects in images more accurately. We also examine the problem of cut detection in Section 5 and show that correlograms as feature vectors yield superior results compared to using histograms.

2 Extensions to the Correlogram

In this section, we look at some extensions to color correlograms. The general theme behind the extensions are:

[*] 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 NSF Career grant CCR-9624552. Email: ravi@cs.cornell.edu.

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

[§] Supported by DOE Grant DEFG02-89ER45405. Email: wjzhu@msc.cornell.edu.

(1) improve the storage efficiency of the correlogram while not compromising its image discrimination capability and (2) use additional information (such as edge) 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 (Section 5).

Notation. Let $\mathcal{I}$ be an $n_1 \times n_2$ image, 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 $|\mathcal{I}| = n_1 n_2$.

Histogram. The *color histogram* (henceforth histogram) h of $\mathcal{I}$ is defined for $i \in [m]$ such that $h_{c_i}(\mathcal{I})$ 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 $h_{c_i}(\mathcal{I}) = H_{c_i}(\mathcal{I})/|\mathcal{I}|$.

Correlogram. Let a *distance set* $D \subseteq [\min\{n_1, n_2\}]$ be fixed *a priori*. Let $d = |D|$. Then, the *correlogram* of $\mathcal{I}$ is defined for $i, j \in [m], k \in D$ such that $\gamma^{(k)}_{c_i,c_j}$ gives the probability that a pixel at a distance k from the given pixel of color c_i is of color c_j. The size of the correlogram is $m^2 d$. If we define the following count $\Gamma^{(k)}_{c_i,c_j}(\mathcal{I}) \triangleq \left|\{p_1 \in \mathcal{I}_{c_i}, p_2 \in \mathcal{I}_{c_j} \mid |p_1 - p_2| = k\}\right|$, then $\gamma^{(k)}_{c_i,c_j}(\mathcal{I}) = \Gamma^{(k)}_{c_i,c_j}(\mathcal{I})/(8kH_{c_i}(\mathcal{I}))$. The *autocorrelogram* of $\mathcal{I}$ captures the spatial correlation between identical colors only and is defined by $\alpha^{(k)}_c(\mathcal{I}) \triangleq \gamma^{(k)}_{c,c}(\mathcal{I})$. This 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. Correlograms were shown to be both efficient and effective feature vectors for content-based image retrieval [4].

For two images $\mathcal{I}, \mathcal{J}$, $|\mathcal{I} - \mathcal{J}|_h$ (resp. $|\mathcal{I} - \mathcal{J}|_\gamma$) denotes the distance between the images when histograms (resp. correlograms) are used as feature vectors.

Performance Measure. Let $\mathcal{I}_i$ be the unique correct answer for $\mathcal{Q}_i$. Let $\mathrm{rank}(\mathcal{I}_i)$ be the rank assigned by the retrieval method. We use the following measures: (1) r-*measure* of a method $\triangleq \sum_{i=1}^{q} \mathrm{rank}(\mathcal{I}_i)$, the sum over all queries of the rank of the correct answer, (2) p_1-*measure* of a method $\triangleq \sum_{i=1}^{q} 1/\mathrm{rank}(\mathcal{I}_i)$, the sum over all queries of the precision at recall equal to 1. Note that a method is good if it has a low r-measure and a high p_1-measure.

Banded Correlogram. Though it was shown in [4] that small values of d actually suffice, it would be more advantageous if the storage requirements of correlogram were trimmed further. This leads to the definition of *banded correlogram* for a given b. For $1 \leq k \leq b$, $\overline{\gamma}^{(k)}_{c_i,c_j}(\mathcal{I}) \triangleq \sum_{k'=kb}^{(k+1)b-1} \gamma^{(k')}_{c_i,c_j}(\mathcal{I})$. In a similar manner, the banded autocorrelogram $\overline{\alpha}^{(k)}_{c_i}(\mathcal{I})$ can also be defined. The space requirements for the banded correlogram (resp. banded autocorrelogram) is $m^2 d/b$ (resp. md/b). (Note that when $b = d$, $\overline{\gamma}$ measures the density of a color c_j near the color c_i, thus suggesting the local structure of colors.) Although banded correlograms have less information than correlograms (and hence seem more susceptible to false matches), our results show that this approximation has only negligible effect on the quality of image retrieval and other applications.

Edge Correlogram. 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. We outline one scheme to combine this information with (auto)correlograms so as to improve the retrieval quality even further. Each of the m color bins is refined to get $\mathcal{I}'$ with $2m$ bins. If $\mathcal{I}(p) = c$, then $\mathcal{I}'(p) = c_+$ when $\mathcal{E}(p) = 1$ and $\mathcal{I}'(p) = c_-$ otherwise. The storage requirement for this quantization is $4m^2 d$ (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 $\alpha^{(k)}_{c_+}$ becomes insignificant, thus rendering the whole operation meaningless. A solution to this problem is to define *edge autocorrelogram* in which cross correlations between c_+ and c_- are also included. The storage is only $4md$. We can trim this by banding.

Experiments. Our heterogeneous image database consists of 14,554 color JPEG images of size 232×168. We consider the RGB colorspace with quantization into 64 colors. 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 the histogram.* Our query set consists of 77 queries, each with a unique correct answer. The queries represent various situations like different views of the same scene, large changes in appearance, small lighting changes, spatial translations, etc. We use the L_1 measure for comparing feature vectors. The feature vectors we use are histograms (hist), color coherent vectors with successive refinement (ccv(s))[8], 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 Figure 1.

Results. Observe that all the correlogram-related methods are on a par in terms of performance and significantly better than histogram and CCV(s). On average, in the correlogram-based method, the correct answer shows up second while for histograms and CCV(s)-based methods, the correct answer shows up at about 40th and 80th places. 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

Measure	hist	ccv(s)	auto	b-auto	e-auto	be-auto
avg. r	81.8	42.5	2.2	2.5	1.9	2.0
avg. p_1	.28	.41	.75	.72	.78	.79

Table 1: Comparison of various image retrieval methods.

hist: 411. ccv(s): 56. auto: 1. b-auto: 1. e-auto: 1. be-auto: 1.

hist: 310. ccv(s): 160. auto: 5. b-auto: 5. e-auto: 1. be-auto: 1.

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

autocorrelograms have the same amount of information as the histogram, but seems a lot more effective than the latter.

3 Image Subregion Querying

The image subregion querying problem is the following: given as input query 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 problem is made even more difficult than image retrieval by a wide variety of effects (such as changing viewpoint, camera noise and occlusion) that cause the same object to appear different in different images. The image subregion querying problem arises in image retrieval and in video browsing.

Performance Measure. Let $\mathcal{Q}_1, \ldots, \mathcal{Q}_q$ be the query images, and for the i-th query $\mathcal{Q}_i$, $\mathcal{I}_1^{(i)}, \ldots, \mathcal{I}_{a_i}^{(i)}$ be the only images that "contain" $\mathcal{Q}_i$, (i.e., $\mathcal{Q}_i$ "appears in" $\mathcal{I}_j^{(i)}, j = 1, \ldots, a_i$,) Due to the presence of false matches the image subregion querying algorithm may return this set of "answers" with various ranks. We use the following measures to evaluate the performance of various competing image subregion querying algorithms: (1) *Average r-measure* given by $\frac{1}{q}\sum_{i=1}^{q}\frac{1}{a_i}\sum_{j=1}^{a_i}\mathrm{rank}(\mathcal{I}_j^{(i)})$, (2) *Average precision* for a query $\mathcal{Q}_i$ given by $(1/a_i)\sum_{j=1}^{a_i} j/\,\mathrm{rank}(\mathcal{I}_j^{(i)})$, 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, and (3) *Recall/Precision vs. Scope*: For a query $\mathcal{Q}_i$ and a *scope* $s > 0$, the *recall* r is defined as $|\{\mathcal{I}_j^{(i)} \mid \mathrm{rank}(\mathcal{I}_j^{(i)}) \leq s\}|/a_i$, and the *precision* p is defined as $|\{\mathcal{I}_j^{(i)} \mid \mathrm{rank}(\mathcal{I}_j^{(i)}) \leq s\}|/s$.

Correlogram Intersection. The image subregion querying problem is a harder problem than image retrieval based on whole image matching. One scheme to avoid exhaustively searching subregions in an image is to define *intersection* of color histograms [14]. The scheme can be interpreted in the following manner.

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 $\mathcal{Q} \cap \mathcal{I}$ and defined as

$$h_{c_i}(\mathcal{Q} \cap \mathcal{I}) \triangleq \frac{\min\{H_{c_i}(\mathcal{Q}), H_{c_i}(\mathcal{I})\}}{|\mathcal{Q}|}. \tag{1}$$

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

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

$$\gamma_{c_i,c_j}^{(k)}(\mathcal{Q} \cap \mathcal{I}) \triangleq \frac{\min\{\Gamma_{c_i,c_j}^{(k)}(\mathcal{Q}), \Gamma_{c_i,c_j}^{(k)}(\mathcal{I})\}}{\min\{H_{c_i}(\mathcal{Q}), H_{c_i}(\mathcal{I})\} \cdot 8k}. \tag{2}$$

Again we measure the presence of $\mathcal{Q}$ in $\mathcal{I}$ by the distance $|\mathcal{Q} - \mathcal{Q} \cap \mathcal{I}|_\gamma$. 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 $|\mathcal{Q} - \mathcal{Q} \cap \mathcal{I}|_\gamma = 0$.

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 [8] is not preserved under subset operations. Therefore, methods for subregion querying based on such unstable features are not likely to perform as well as the histogram or correlogram based methods.

Experiments. The experimental setup is the same as in Section 2. Our query set for this task consists of 30 queries. Queries have 2 to 16 answer images with an average of 5 answers per query. 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, often with its appearance significantly changed. Examples of some queries and answers (and the rankings according to the histogram and autocorrelogram intersection methods) are shown in Figure 2.

Results. The histogram and autocorrelogram intersection methods for subregion querying are compared in Tables 2 and 3. 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 2). Table 3 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 yield better average precision on 26 out of 30 queries. These suggest that the autocorrelogram is a significantly superior method for subregion querying.

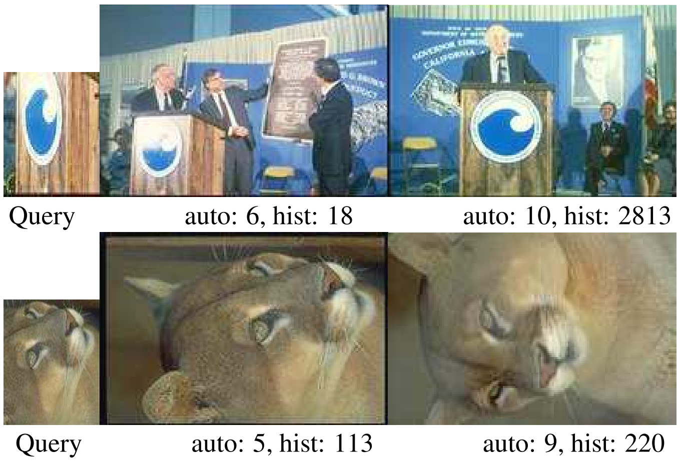

Figure 2: Sample queries and answer sets with ranks for various methods. (Lower ranks are better.)

Method	Avg. r-measure	Avg. precision
Hist	56.3	.386
Auto	22.5	.602

Table 2: Performance of Intersection methods – I. (Smaller r-measure and larger precision are better.)

4 Location Problem

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 a sequence of frames. Efficiency is also required in this task because huge amounts of data need to be processed.

Template matching is one straightforward approach to the location problem. This method is likely to yield good results, but is computationally expensive. Attempts have been made to make template matching more efficient [5]. The *histogram backprojection* method is one such approach to this problem [14]. Though this method generally works well in practice, it has some serious drawbacks. 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 $\pi_{c,h}(\mathcal{I}|\mathcal{Q}) \triangleq \min\{H_c(\mathcal{Q})/H_c(\mathcal{I}), 1\}$. The goodness of a pixel $p \in \mathcal{I}_c$ is defined to be $\pi_{c,h}(\mathcal{I}|\mathcal{Q})$. The contribution of $\mathcal{I}|_p$, the

Method ↓	Precision(%)			Recall		
Scope →	5	10	25	5	10	25
Hist	.27	.22	.13	.31	.46	.68
Auto	.49	.35	.17	.54	.72	.85

Table 3: Performance of Intersection methods – II. (Larger values are better.)

subimage of $\mathcal{I}$ centered at p is given by

$$\Pi_p(\mathcal{I}|\mathcal{Q}) \triangleq \sum_{q \in \mathcal{I}|_p} \pi_{\mathcal{I}(q),h}(\mathcal{I}|\mathcal{Q}). \tag{3}$$

Then, the location of the model is given by $\arg\max_{p \in \mathcal{I}} \Pi_p(\mathcal{I}|\mathcal{Q})$.

Note 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 $\pi_{c,h}(\mathcal{I}|\mathcal{Q})$ is maximum over all colors. If $\mathcal{I}$ has a subimage (which may be totally unrelated to $\mathcal{Q}$) that has many pixels of color c, then this method tends to identify $\mathcal{Q}$ with this subimage, thus causing an error in some cases.

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 zebras in Figure 3.

Correlogram Correction. To alleviate the problems with histogram backprojection, we incorporate local spatial correlation information by using a *correlogram correction* factor in Equation 3. The idea is to integrate discriminating local characteristics while avoiding local color template matching [2]. 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.

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). $\alpha_p^{(k)} = \Gamma^{(k)}_{\mathcal{I}(p),\mathcal{I}(p)}(\{p\})/8k$ where $\{p\}$ represents the pixel p along with its neighbors considered as an image. Now, the correlogram contribution of p is defined as $\pi_{p,\gamma}(\mathcal{I}|\mathcal{Q}) \triangleq 1 - |\mathcal{Q}_{\mathcal{I}(p)} - \{p\}|_\gamma$. In words, the contribution of p is the L_1-similarity 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 Equation 3, the final goodness value of a subimage $\mathcal{I}|_p$ is given by $\Pi_p(\mathcal{I}|\mathcal{Q}) \triangleq \sum_{q \in \mathcal{I}|_p} \left(\beta \pi_{\mathcal{I}(q),h}(\mathcal{I}|\mathcal{Q}) + (1-\beta)\pi_{q,\gamma}(\mathcal{I}|\mathcal{Q})\right)$, where $0 \leq \beta \leq 1$.

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, 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

Figure 3: Hist output, query image, and auto output.

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 goodness of a pixel is given by a weighted linear combination of the histogram and correlogram backprojection contributions. We call this *correlogram correction*.

Experiments. We use the same database to perform the location experiments. For the simple location task, 66 query images are run on 52 images that contain these models. A distance set $D = \{1, 3, 5\}$ and $\beta = 0.5$ are used. For the tracking task, we choose three videos bus (133 frames), clapton (44 frames), skydive (85 frames). Both histogram backprojection and autocorrelogram correction methods are tried for these tasks.

Results. The results are evaluated using the *success ratio*, defined as the proportion of images for which the location returned by a method is "reasonably close" to the true location of the target in the image.

Table 4 shows the results of our experiments. For the 66 queries, the autocorrelogram correction method correctly finds the location of a model 96% of the time. Similarly, this method correctly locates the tracked object in 78% of the frames in the clapton video. Figure 3 shows some example objects and the location of the objects (indicated by a box) in the images as found by different methods. These results clearly show that correlogram correction alleviates many of the problems associated with simple histogram backprojection.

Method	hist	auto
66 queries	0.78	0.96
bus	0.93	0.99
clapton	0.44	0.78
skydive	0.96	0.96

Table 4: Success ratios for location/tracking experiments.

5 Cut Detection

The increasing availability of video data makes automated video analysis a necessity. The first step to automated video content analysis is to segment a video into camera shots (also known as *key frame extraction*). A camera shot is an unbroken sequence of frames from one camera, and a *cut* is said to occur when two consecutive frames are from different shots.

Cut detection algorithms usually work as follows: adjacent frames are compared using some image feature and frames that are "sufficiently" similar are assumed to belong to the same shot, and dissimilar frames are taken to signify a cut. Different cut detectors use different features to compute the similarity between consecutive frames, e.g. pixel difference, statistical differences, histogram comparisons, edge differences, etc [1]. Histograms are most commonly used to detect cuts because they are efficient to compute and insensitive to camera motions. They are not robust to local changes in images, however, so false positives easily occur in this method (see Figure 4). Since correlograms have been shown to be robust to large appearance changes for image retrieval, we use correlograms for cut detection.

Notation. We use $\vec{\mathcal{I}} = \mathcal{I}_1, \ldots, \mathcal{I}_t$ to denote a sequence of video frames. Let $\rho(\vec{\mathcal{I}})$ be the number of cuts in $\vec{I}$. Let $\mathcal{J}_k$ be the k-th cut and suppose it occurs between images $\mathcal{I}_i$ and $\mathcal{I}_{i+1}$. Now suppose pairs of adjacent images $\langle \mathcal{I}_i, \mathcal{I}_{i+1} \rangle$ are ranked according to $|\mathcal{I}_i - \mathcal{I}_{i+1}|_f$, where f is the feature used to compare images. We use $\mathrm{rank}(\mathcal{J}_k)$ to denote the rank of the cut $\mathcal{J}_k \equiv \langle \mathcal{I}_i, \mathcal{I}_{i+1} \rangle$ according to this ranking.

Performance Measure. *Recall* and *precision* are usually used to measure the performance of cut detection algorithms, but it is difficult to compare the performance of different algorithms based on recall vs. precision curves [1]. We therefore 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. scope* to measure false negatives (misses). We choose two scope values: $\rho(\vec{\mathcal{I}})$ and $2\rho(\vec{\mathcal{I}})$. We also use the *excessive rank value* (ERV), which is defined by $\sum_{i=1}^{\rho(\vec{\mathcal{I}})} (\mathrm{rank}(\mathcal{J}_i) - i)$ and the *average precision value* (AvgP), which is defined by $\sum_{i=1}^{\rho(\vec{\mathcal{I}})} \frac{i}{\mathrm{rank}(\mathcal{J}_i)} / \rho(\vec{\mathcal{I}})$. Note that a smaller excessive rank value and a larger average precision value indicate better results.

Experimental Results. We use 64 colors histograms, and banded autocorrelograms (which have the same size as histograms) for cut detection in 5 video clips. The results are shown in Table 5 and Table 6. The results of our experiments show that banded autocorrelograms are more effective than histograms while having the same storage requirement.

Method →	hist			banded-auto		
Video ↓	ρ	2ρ	ERV	ρ	2ρ	ERV
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 5: Recall vs. Scope for cut detection.

Figure 4: False cuts detected by hist but not by auto.

Method →	hist			banded-auto		
Video ↓	ρ	2ρ	AvgP	ρ	2ρ	AvgP
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

Table 6: Precision vs. Scope for cut detection.

6 Some Related Work

In this section we discuss related ideas in the areas of image retrieval, image subregion querying, and object localization.

The color histogram has shown its efficiency and advantages as a general tool for various applications, such as content-based image retrieval [3, 7, 9], object indexing and localization [14, 6, 2], and cut detection for video segmentation [1]. As a feature vector for image retrieval, the color histogram is very easy to compute and insensitive to small changes in viewing positions. However, due to the lack of any spatial information, the histogram is liable to false positives. This problem is especially acute for large 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 [12, 13, 8, 4]. One approach [12, 13] divides images into subregions and imposes positional constraints on the image comparison (*image partitioning*). Another approach [8] augments histograms with local spatial properties (*histogram refinement*). Our approach, the *color correlogram* [4], makes use of spatial correlation between pairs of colors.

The image subregion query problem is closely related to the object recognition problem, which has been studied for a long time. Since a color histogram is very easy to extract from an image, it has been successfully used for object indexing, detection and localization [14, 3, 7, 6, 11, 15]. Swain and Ballard [14] propose *histogram intersection* for object identification and *histogram backprojection* for object localization and report impressive results. The advantages and disadvantages inherent to histograms in general have been discussed in detail in Section 3.

One disadvantage of color histograms is that they are sensitive to illumination changes. Slater and Healey [11] proposed an algorithm that computes invariants of local color distribution and uses these invariant for 3-D object recognition. Matas et al. [6] proposed the color adjacency graph (CAG) as an alternative representation for multiple-colored objects. CAGs improve over histograms by incorporating coarse color segmentation into histograms.

7 Conclusions

We have shown the correlogram to be a generic tool for solving various problems in computer vision and multimedia applications. Correlograms turn out to be an easy to compute, compact to represent, and effective to use feature for many such problems. We demonstrate this by applying correlogram to image retrieval, image subregion querying, object localization, object tracking, and cut detection problems. We also suggest schemes to cut down the storage requirements and to improve retrieval performance.

References

[1] J. S. Boresczky and L. A. Rowe. A Comparison of Video Shot Boundary Detection Techniques. Storage & Retrieval for Image and Video Databases IV, SPIE 2670, pp. 170-179, 1996.

[2] F. Ennesser and G. Medioni. Finding Waldo, or Focus of Attention Using Local Color Information. *IEEE PAMI*, Vol. 17, No. 8, 1995.

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

[4] J. Huang, S. R. Kumar, M. Mitra, W. J. Zhu, and R. Zabih. Image indexing using color correlograms. *CVPR*, 1997.

[5] A. Margalit and A. Rosenfeld. Using Probabilistic Domain Knowledge to Reduce the Expected Computational Cost of Template Matching. *Comp. Vision, Graphics, and Image Proc.*, 51, 219–234, 1990.

[6] J. Matas, R. Marik, and J. Kittler. On Representation and Matching of Multi-Colored Objects. *5th ICCV*, page 726–732, 1995.

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

[8] G. Pass and R. Zabih. Histogram refinement for content-based image retrieval. *IEEE WACV*, 96–102, 1996.

[9] A. Pentland, R. Picard, and S. Sclaroff. Photobook: Content-based manipulation of image databases. *IJCV*, 18(3):233–254, 1996.

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

[11] D. Slater and G. Healey. Combining Color and Geometric Information for the Illumination Invariant Recognition of 3-D Objects. *5th ICCV*, page 563–568, 1995.

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

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

[14] M. Swain and D. Ballard. Color indexing. *IJCV*, 7(1):11–32, 1991.

[15] T. F. Syeda-Mahmood. Data and Model-Driven Selection Using Color Regions. *IJCV*, 21(1/2):9–36, 1997.