Theory of Keyblock-based Image Retrieval

Lei Zhu, Aibing Rao and Aidong Zhang
Department of Computer Science and Engineering
State University of New York at Buffalo
Buffalo, NY 14260

The success of text-based retrieval motivates us to investigate analogous techniques which can support the querying and browsing of image data. However, images differ significantly from text both syntactically and semantically in their mode of representing and expressing information. Thus, the generalization of information retrieval from the text domain to the image domain is non-trivial. This paper presents a framework for information retrieval in the image domain which supports content-based querying and browsing of images. A critical first step to establishing such a framework is to construct a codebook of "keywords" for images which is analogous to the dictionary for text documents. We refer to such "keywords" in the image domain as "keyblocks." In this paper, we first present various approaches to generating a codebook containing keyblocks at different resolutions. Then we present a keyblock-based approach to content-based image retrieval. In this approach, each image is encoded as a set of one-dimensional index codes linked to the keyblocks in the codebook, analogous to considering a text document as a linear list of keywords. Generalizing upon text-based information retrieval methods, we then offer various techniques for image-based information retrieval. By comparing the performance of this approach with conventional techniques using color and texture features, we demonstrate the effectiveness of the keyblock-based approach to content-based image retrieval.

Categories and Subject Descriptors: H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; I.4.7 [**Image Processing and Computer Vision**]: Feature Measurement; I.4.10 [**Image Processing and Computer Vision**]: Image Representation

General Terms: Algorithms, Experimentation

Additional Key Words and Phrases: keyblock, content-based image retrieval, clustering, codebook

1. INTRODUCTION

With the advance of multimedia technology, image data in various formats are becoming available at an explosive rate. The effective use of such enormous data resources is predicated on the ability to search and retrieve information from image databases. Thus, content-based image retrieval (CBIR) has become an active research area. However, current capabilities for querying and browsing image data based on content have not been highly satisfactory. Content-based image retrieval using low-level features such as color [Swain and Ballard 1991; Smith and Chang 1996b; Pass et al. 1996], texture [Manjunath and Ma 1996; Smith and Chang 1994; Sheikholeslami and Zhang 1997; Stromberg and

This research is supported by NSF and National Imaging and Mapping Agency (NIMA).
URL of demo: http://vangogh.cse.buffalo.edu:8080/.

Farr 1986; Mandelbrot 1977], shape [Syeda-Mahmood 1996; Mehrotra and Gary 1995; Hirata and Kato 1993; Mokhtarian et al. 1996a; 1996b; Horn 1988; Faloutsos et al. 1994; Korn et al. 1996; Wang and Acharya 1998a; 1998b; Shahabi and Safar 1999; Tao and Grosky 1999; Safar et al. 2000], and others [Picard 1996; Ahuja and Rosenfeld 1981; Modestino et al. 1981; Dougherty and Pelz 1989] extracted from images has been well studied. Various image querying systems, including QBIC [Flickner M., Sawhney H., Niblack W., Ashley J., Huang Q. and Dom B. et al. 1995], VisualSeek [Smith and Chang 1996b], PhotoBook [Pentland et al. 1994], and Virage [Bach et al. 1996], have been built based on the low-level features for general or specific image retrieval tasks. However, effective and precise image retrieval still remains an open problem because of the extreme difficulty of fully characterizing images. Successful techniques have been developed for some specific applications, such as face and finger-print recognition. However, an effective approach for querying and browsing images in general still remains elusive.

In contrast, many techniques have been developed for text-based information retrieval (IR), and some keyword-based text information retrieval systems, such as Yahoo, Lycos, and Google, have proven highly successful for indexing and querying web sites. Their success may also shed light on the area of content-based image retrieval, because the relatively mature theories and techniques of text-based information retrieval may be applicable to the image domain. One existing approach is to annotate images with text keywords and captions, and then retrieve the image database using text-based techniques. Since both text and image retrieval are involved in information retrieval, the success of the text-based information retrieval motivates us to develop a general theory of content-based image retrieval which is analogous to the techniques developed in text-based information retrieval.

The generalization of information retrieval from the text domain to the image domain is, however, non-trivial. The greatest obstacle arises from the intrinsic difference between text and image in representing and expressing information. In the syntactic, representational realm, a text document is one-dimensional, while an image is two-dimensional. In their expressive or semantic aspect, the units (words) of a text document, especially keywords, carry direct semantic content which is related to the semantics of text documents. In contrast, the units of an image offer little semantic content. Pixels generally provide no clue about the semantics of an image, and, image segments usually offer only an unreliable object description.

However, content-based image retrieval does not require perfect object recognition. To provide a cogent foundation for effective image retrieval, we must first construct a general theory of content-based querying and browsing of images. Building upon the theories and techniques of text retrieval, a codebook of commonly-accepted "keywords" for images can be established analogous to the the dictionary of keywords for text documents. An image can then be represented by a sequence of these "keywords," just as a text document is summarized as a sequence of keywords. The models of text IR can be generalized to image feature representation and retrieval. Thus, the central questions are:

—*What are the "keywords" of an image?*

—*How does one generate "keywords" from images?*

As mentioned above, pixels are not good "keywords" because they do not effectively convey image semantics. Objects are good keyword candidates since they represent a kind of "pictorial language" and have been part of human communication since the dawn of language. However, full object recognition still presents many challenges.

While recognizing the existing problems in the field of CBIR, this paper presents a general theory and a number of techniques for image information retrieval which are analogous to keyword-based text retrieval. A critical issue to be addressed is the construction of keyword-like image feature segments, or *"keyblocks."* The codebook for images consists of the set of keyblocks selected. Using this codebook of keyblocks, an image can be encoded as the indices of the keyblocks in the codebook. Based on this image representation, information retrieval techniques developed in the text domain can thus be generalized for image retrieval.

1.1 Related work

Some initial work related to the proposed approach was introduced in [Idris and Panchanathan 1996; Lu and Teng 1999]. In that method, an image is first compressed based on Vector Quantization (VQ) [Gersho and Gray 1992], and then a code vector usage map or a code vector histogram is used as the image features. The work is a natural analogue of text-based techniques, although the authors might not realize it. A VQ-encoded image is simply a one-dimensional vector of codes in which each code corresponds to an entry in a pre-selected codebook. The codebook is generated through the clustering of all image blocks of some training images.

Another related approach uses color-based image retrieval techniques. The concept of color histograms [Swain and Ballard 1991; Pass et al. 1996; Niblack W., Barker R., Equitz W., Flickner M., Glasman E. and Petkovic D. et al. 1993; Hunt 1989; Netravali and Haskell 1988; Russ 1995] is an analogue of text-based techniques, where the "keywords" are the colors after color quantization, the "text" is the one-dimensional vector of the pixel colors of an image, and the histogram is the frequency of these "keywords" in the "text". A color histogram captures the global color distribution of an image, but it lacks information regarding the spatial relationship among pixels. Enhanced approaches have been proposed which refine the color histogram based on the spatial information of colors [Hsu et al. 1995; Rickman and Stonham 1996; Smith and Chang 1996a; Rao et al. 1999; Stricker and Dimai 1996; Pass and Zabih 1996; Huang et al. 1997; Huang 1998]. Representative methods include the *color coherent vector* presented by Pass and Zabih [Pass and Zabih 1996; Pass et al. 1996] and the *color correlogram* proposed by Huang [Huang et al. 1997; Huang 1998].

Many text-based retrieval techniques can also provide useful models for retrieval in the image domain. Baeza-Yates and Ribiero-Neto [Baeza-Yates and Ribiero-Neto 1999] presented a taxonomy of fifteen text-based information retrieval models. Among them, two classic models, the Boolean Model and the Vector Model, are widely used and have proven suitable for text retrieval. These two models use keywords to index and retrieve documents based on the assumption that both documents in the database and queries can be described by a set of keywords, which are referred as index terms. For most text search engines, a keyword is any word which appears in the documents. For a given database of M text documents, after N keywords have been determined, a $M \times N$ matrix W, called a *weight matrix*, is defined. Each entry in W, which is associated with a particular document and a keyword, represents the importance of the keyword in describing the content of the document. A similarity metric is defined to calculate the degree of similarity between the query and database documents. For different models, the weight matrices and similarity metrics are defined differently and thus give rise to different retrieval performance. More detail regarding these models can be found in [Baeza-Yates and Ribiero-Neto 1999].

1.2 Our contribution

This paper introduces a systematic framework, called the *keyblock-based* approach, which provides a practical solution for content-based image retrieval analogous to text-based IR techniques. Using this framework, we will develop strategies to generate keyblock codebooks at different resolutions. These codebooks are constructed for a variety of databases, ranging from general-purpose image databases to domain-specific databases. In particular, we present approaches for keyblock selection by expanding the approaches developed in Vector Quantization (VQ) so that the keyblocks can be generated from large volumes of image segments, including blocks, regions, and objects. We will then develop novel methods for extracting comprehensive image features on the basis of the frequency and appearance of keyblocks within images. We will also provide retrieval techniques which support content-based image retrieval. The keyblock-based approach represents the first attempt to establish a general framework for information retrieval in the image domain, bridging the gap between information retrieval and content-based image retrieval. Experimental results will be presented to demonstrate the effectiveness of this keyblock-based approach.

The remainder of this paper is organized as follows. Section 2 introduces the main components of the keyblock-based approach. In Section 3, techniques for generating keyblocks from images and for encoding images are described. Section 4 presents models for keyblock-based image feature extraction and retrieval, while Section 5 offers a performance evaluation of the proposed approach. A summary and concluding remarks appear in Section 6.

2. A FRAMEWORK FOR KEYBLOCK-BASED IMAGE RETRIEVAL

The keyblock-based approach includes the following main components:

—*Codebook generation*: For each image database, a codebook is generated containing keyblocks at different resolutions. Keyblocks can be constructed by applying clustering algorithms.

—*Image encoding*: Each image in the database, as well as the query, is decomposed into blocks or segments. Then, for each block/segment, the closest entry in the codebook is located and the corresponding index is stored. Each image is thus represented as one-dimensional codes linked to the keyblocks in the codebook, analogous to the representation of a text document as a linear list of keywords in text-based information retrieval.

—*Image feature representation and retrieval*: This process extracts comprehensive image features on the basis of the frequency and appearance of keyblocks within images and provides retrieval techniques to support content-based image retrieval.

Figure 1 presents a flowchart illustrating our approach. Originally introduced in [Zhu et al. 2000; Zhu et al. 2000a; 2000b], this concept is expanded here into a fundamental theory of keyblock-based image retrieval.

3. KEYBLOCK GENERATION AND IMAGE ENCODING

This section describes approaches to the generation of keyblocks and encoding of images. Keyblocks can be generated from the following spaces of image data sets:

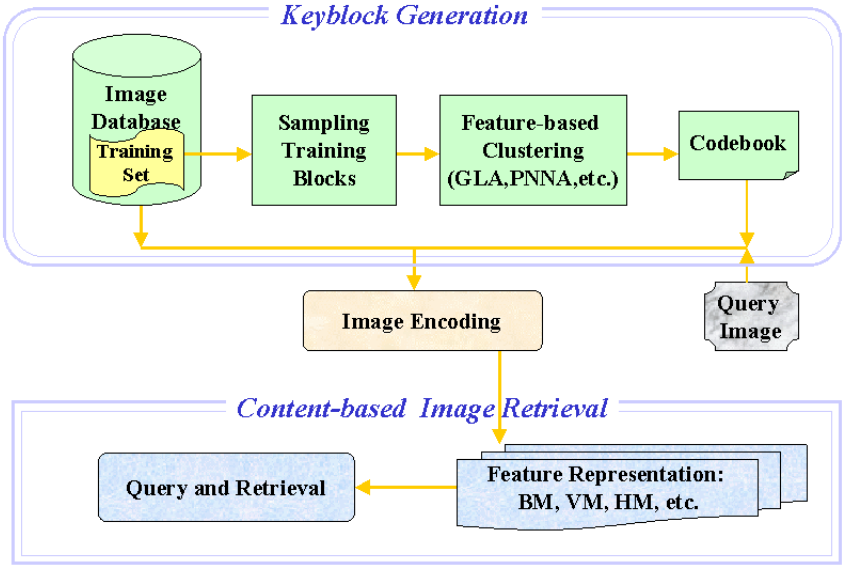

Fig. 1. Flowchart illustrating keyblock-based image retrieval

—*The original space*: Images are partitioned or segmented into smaller blocks or segments. A subset of representative blocks/segments is then selected to serve as as keyblocks to represent the images.

—*The feature space*: Low-level feature vectors, such as color, texture, and shape, are extracted from image segments/blocks. A subset of representative feature vectors can then be used as keyblocks to represent the images.

3.1 Unsupervised selection of keyblocks

A variety of approaches can be used to select keyblocks from the centroids of clusters within either the original space or feature space. As a first step, let $C = \{c_1, ..., c_i, ..., c_N\}$ be the "codebook" of keyblocks representing the images, where N is the codebook size and c_i, $1 \leq i \leq N$, are the keyblocks. Let F be a mapping:

$$F : R^k \longrightarrow C = \{c_1, ..., c_i, ..., c_N \mid c_i \in R^k\},$$

where R^k is the Euclidean space of dimension k. Given a sequence $T = \{t_1, ..., t_j, ..., t_l \mid t_j \in R^k\}$, the mapping F gives rise to a partition of T which consists of N cells $P = \{p_1, ..., p_i, ..., p_N\}$, where $p_i = \{t \mid t \in T, F(t) = c_i\}$. For a given distortion function $d(t_j, c_i)$, which is the distance between the input t_j and output code c_i (for example, the Euclidean distance), an optimal mapping should satisfy the following conditions:

—*Nearest Neighbor Condition:* For each p_i, if $t \in p_i$, then $d(t, c_i) \leq d(t, c_j)$, for all $j \neq i$.

—*Centroid Condition:* For a given partition P, the optimal code vectors satisfy

$$c_i = \frac{\sum_{t \in p_i} t}{k_i}, 1 \leq i \leq N, k_i \; is \; the \; cardinality \; of \; p_i.$$

We will now discuss the clustering algorithms used to generate keyblocks. There are a variety of clustering algorithms available which can be applied to data sets of different types [Kaufman and Rousseeuw 1990; Ng and Han 1994; Wang et al. 1997; Zhang et al. 1996; Pauwels et al. 1997; Ester et al. 1996; Pauwels et al. 1997; Sheikholeslami et al. 1998; 2000]. We present a practical and efficient algorithm which expands upon the algorithms used in Vector Quantization [Gersho and Gray 1992]. In this algorithm, clustering is applied to the set of data obtained from a training set of the images, using either the original space or the feature space. The centroid of each cluster is then used as a codebook entry. Two commonly-used algorithms which can serve as the basis for this approach are the *Generalized Lloyd Algorithm* (GLA) and the *Pairwise Nearest Neighbor Algorithm* (PNNA) [Gersho and Gray 1992]. After explaining these two algorithms, we will present an approach which integrates their good features for the efficient generation of keyblock codebooks.

GLA. GLA is an iterative clustering algorithm which is designed to approximate optimality conditions during each iteration in the design of a vector quantizer. During each iteration, first, the set of training vectors is re-partitioned into cells according to the *nearest-neighbor condition* based on the codebook generated in the previous iteration; thus, each training vector is grouped into the cluster represented by the codebook vector which is its nearest neighbor. Second, based on the *centroid condition*, the centroid of each cell is computed. These centroids then replace the codebook vector, so that all the centroids now draw upon the codebook generated at the current stage. Finally, the overall distortion is computed and the change in distortion is examined. If the change in distortion exceeds a specified threshold, the process described above is repeated; if not, no further iterations are performed. This iterative process is guaranteed to be convergent, although it may not arrive at the optimal codebook [Gersho and Gray 1992].

Note that the iterations may converge to a local minimum if the initial codebook is poorly selected. Gersho and Gray [Gersho and Gray 1992] presented various methods to set the initialization of the process. Furthermore, GLA is computationally expensive, while PNNA is more efficient.

PNNA. PNNA is a simple clustering algorithm which starts with the training set. At each iteration, two nearest code vectors are merged and replaced by their centroid, thus decreasing the size of the codebook by one. This process is repeated until the desired final codebook size is reached. While each merge is optimal, but the overall result may not be optimal; thus PNNA is generally faster but slightly inferior to GLA.

Since an image database is usually very large, a proper balance must be found between the efficiency and effectiveness of codebook generation. This can be achieved by combining GLA and PNNA. The complete procedure for codebook generation through this enhanced algorithm will now be described.

Let D be an image database. First, a sampling procedure is conducted to select a training set of images $\hat{D} = \{i_1, ..., i_k, ..., i_t\}$. In general, the larger the training set, the less the average distortion. In the ideal case, the whole database D can be used as the training set, but this would be too costly. Sampling is intended to select a set of images which can effectively represent D. Second, the training images are divided into segments or blocks

which are represented as vectors, with smaller block sizes yielding higher resolutions. Again, quality of image representation can be improved at a higher cost. A block size below 16×16 is usually needed to satisfy quality requirements.

Since the block size is small, the set of training blocks B decomposed from the training image set $\hat{D}$ can still be very large. Therefore, directly applying GLA to B would still be too costly. For example, an image at 512×512 resolution would contain $16,384$ (4×4) blocks. If the number of training images is t, then there are $16,384 \times t$ vectors to which a clustering algorithm such as GLA would be applied to obtain the codebook of 4×4 blocks. To alleviate this problem, the number of training vectors need to be reduced. One approach would be to again apply a sampling strategy to each training image to reduce the number of blocks to a manageable size. However, in order to ensure that representative blocks remain in the set of training vectors, it is preferable to first apply GLA to the blocks of each training image and then use the resulting cluster centroids to form the set of training vectors. We then apply PNNA to generate a codebook C', which serves as an initial codebook from which GLA generates the final codebook C. In addition to increasing efficiency, this combination of GLA and PNNA reduces the possibility that GLA will converge to a local minimum.

Generation of keyblocks for image database D proceeds according to the following steps:

Step 1: D is sampled to produce training set $\hat{D}$. This sampling can be accomplished either manually or automatically. With an automatic procedure, random sampling may be used, but the optimally representative set will not be guaranteed. An expensive but effective approach is to apply existing feature extraction methods, such as color histogram, texture, or shape, to conduct a clustering process on the entire database. The centroid of each cluster is then used as a training image.

Step 2: Each training image i_k is decomposed into blocks or segments to generate a sequence of vectors $T_k = \{t_1, ..., t_j, ..., t_{k_l}\}$. GLA is then applied to T_k to extract a codebook C_k of size N, which is randomly initialized.

Step 3: All vectors in C_k are placed in the training set T.

Step 4: PNNA is applied to T to generate a codebook C'.

Step 5: GLA is again applied to T, with C' as an initial codebook to yield the final codebook C.

After the clusters are created, their centroids are formulated and selected as keyblocks. The entire procedure of keyblock selection can be an unsupervised learning process, minimizing overall distortion between inputs and the best-matching centroids (keyblocks). In general, the keyblocks in the resulting codebook represent the most general structures extracted from all the inputs. This approach can be applied to any general image domain, including web images.

3.2 Knowledge-based keyblock generation

For some specific domains, such as geographic images, domain knowledge can be incorporated into the keyblock selection process. For example, some important geographic and land use features include water, agricultural areas, forest, grasslands, and residential areas. To generate keyblocks which reflect the semantic content of these geographic features, we use a three-stage keyblock generation strategy, which is illustrated in Figure 2.

Stage I: Keyblock generation for each semantic class. For each semantic class, a corresponding codebook is generated. For example, if there are five geographic classes, such as water, agricultural areas, forest, grasslands, and residential areas, this stage will

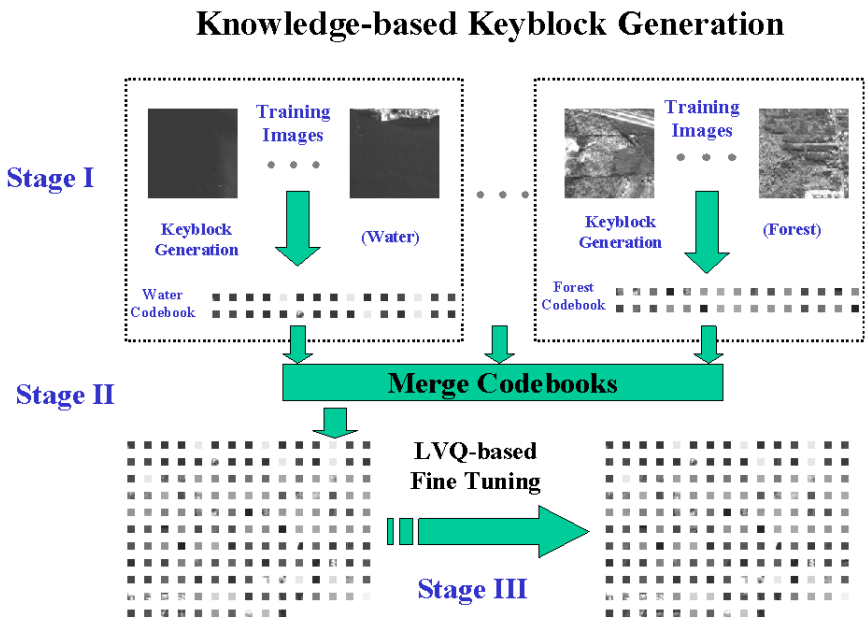

Fig. 2. Three-stage process for knowledge-based keyblock generation

generate five codebooks. Each keyblock in these codebooks will retain its original pixel intensity values and will also carry a class label corresponding to the type of geographic feature it represents. Two methods can be used in keyblock selection:

—Manual selection: Codebooks can be generated by manually selecting image blocks from images. This method needs a lot of human labors, which is time consuming and lack of accuracy.

—Automatic selection: For each semantic class, domain experts need only provide some training images to initiate the standard keyblock generation procedure presented in Section 3.1. This method is more practical and is used in the process described below.

The purpose of Stage I is to assign semantic meaning to each keyblock, since domain knowledge can be imported.

Stage II: Codebook merge. The codebooks generated in Stage I are now merged into a bigger codebook. This codebook comprises keyblocks with a range of meanings and can be directly used in image encoding and decoding. However, both because the component keyblocks come from different training sets, and because unsupervised clustering algorithms may have been employed in Stage I, there may be overlap between the boundaries of clusters centered on the keyblocks. The quality of this codebook will therefore be improved through the fine-tuning process in Stage III.

Stage III: Fine-tuning the codebook using Learning Vector Quantization. Fine-tuning is performed in this stage by using Learning Vector Quantization (LVQ) algorithms [Kohonen et al. 1995]. LVQ is a commonly-used method for classification which modifies

cluster boundaries in each iteration generated by clustering algorithms such as GLA.

The codebook generated in Stage II will be used as the initial codebook for this process. Each keyblock and each training block has a class label. In each iteration of the clustering algorithm, updates are performed on those two data items (keyblocks) c_i and c_j which are nearest to a training input (training block) t. This update is performed when one of these data items belongs to the correct class while the other belongs to an incorrect class, and t falls within an update zone defined around the mid-plane of cluster boundaries formed by c_i and c_j. Assuming that d_i and d_j are the distances of t from c_i and c_j, respectively, this update zone is defined as the region where $min(d_i/d_j, d_j/d_i) > T$, with T being a threshold with a typical value between 0.5 and 0.7. This update zone restricts updates to only those highly ambiguous pairs of data items that are around the cluster boundaries. In iteration m, the two data items are updated as follows (assuming that c_i belongs to the same class as t, and c_j belongs to an incorrect class):

$$c_i^{m+1} = c_i^m + \alpha(t - c_i^m), \tag{1}$$

$$c_j^{m+1} = c_j^m - \alpha(t - c_j^m). \tag{2}$$

The learning rate α usually has an initial value around 0.1, which may be decreased gradually in the course of training. The training process will end when no more updates occur.

3.3 Image encoding

To encode an image using the codebook, the image is first partitioned or segmented into blocks or segments, and each block/segment (or its feature vector) is then replaced by the index of the nearest codebook entry. At this point, each image is a two-dimensional array of codebook keyblock indices. It can also be considered as a list of keyblocks similar in format to a text document defined by a list of keywords. We can reconstruct the image using the codebook to test whether the codebook was properly selected. Figure 3 illustrates the general procedure for image encoding and decoding.

4. KEYBLOCK-BASED IMAGE FEATURE REPRESENTATION AND RETRIEVAL

The general framework for keyblock-based image retrieval consists of five main components as follows:

—Database D: a list of encoded images

—Codebook C: a list of keyblocks

—Feature representation model: a model which extracts those features of each image which indicate the importance of each keyblock in the image

—Q: a set of visual queries, where each query has a feature vector which is similar to the feature vector of an image

—Similarity measure between a query and an image: a metric which computes the ranking of images

Given an image encoded as a two-dimensional array of keyblock indices in the codebook, image features can be extracted on the basis of either single context-free codes, which consider only the existence of individual codes, or of multiple context-sensitive codes, which consider the relationships between codes. In the following section, we will

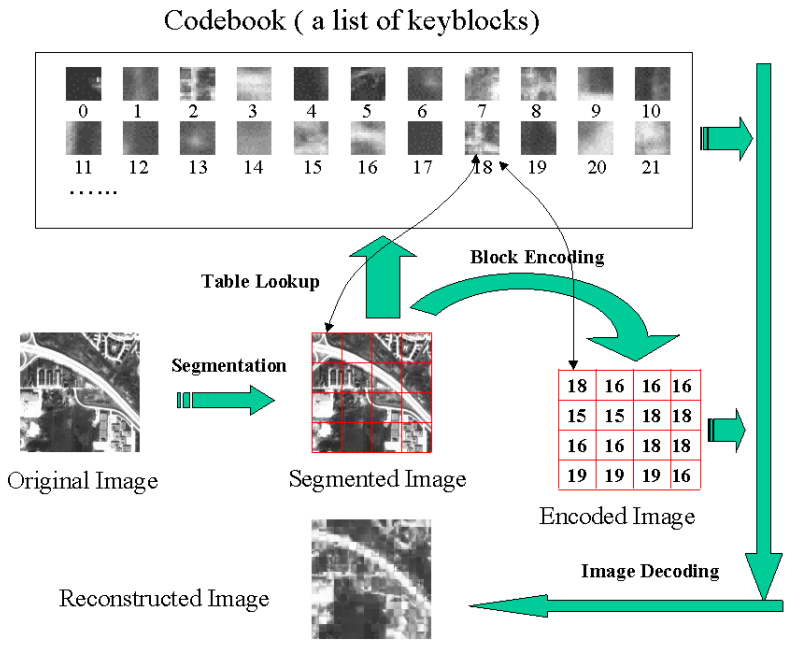

Fig. 3. General procedure for image encoding and decoding

discuss various specific models which are based on different approaches to feature representation and similarity measurement. For these models, we assume that there are M images in database $D = \{I_1, ..., I_j, ..., I_M\}$ and N keyblocks in codebook $C = \{c_1, ..., c_i, ..., c_N\}$.

4.1 Context-free feature models

As with text-based retrieval models, image feature representation models can be context-free. We will discuss two of the most popular models, the Boolean Model (BM) and the Vector Model (VM), to illustrate the construction of image feature vectors. A special case of VM, called the Histogram Model (HM), will also be presented. Let $\vec{I_j} = \{w_{1j}, ..., w_{ij}, ..., w_{Nj}\}$ be the feature vector for image I_j and $\vec{q} = \{w_1(q), ..., w_i(q), ..., w_N(q)\}$ be the feature vector for query q.

4.1.1 *Boolean model.* BM generates feature vectors for database images and queries by considering whether or not a keyblock appears. We define $w_{i,j} \in \{0,1\}$ as:

$$w_{i,j} = \begin{cases} 1 & if \quad f_{i,j} \geq \theta, \\ 0 & \text{otherwise}, \end{cases}$$

where $f_{i,j}$ is the frequency of keyblock c_i appearing in image I_j and θ is a threshold. $w_i(q)$ is defined similarly as $w_{i,j}$. Thus, both $\vec{q}$ and $\vec{I_j}$ can be considered as strings of length N where the i-th bit indicates whether or not c_i appears. Comparison between $\vec{q}$ and $\vec{I_j}$ is bit by bit, so the similarity between query q and image I_j is defined as:

$$S_{BM}(\vec{q}, \vec{I_j}) = W_1 * n_{11} + W_0 * n_{00},$$

where n_{11} is the number of bits at which both $\vec{I_j}$ and $\vec{q}$ are 1, n_{00} is the number of bits at which both $\vec{I_j}$ and $\vec{q}$ are 0, and W_1 and W_0 are the weights assigned to n_{11} and n_{00}, respectively.

The simultaneous appearance or disappearance of certain keyblocks in both I_j and q provides a strong indication of the similarity between I_j and q. Moreover, "appearance" is more important than "disappearance." So, normally, $W_1 \geq W_0$. For example, in our implementation, we notice that $W_1 = 1$, $W_0 = 0.5$ gives the best result. Suppose $\vec{q} = (1,0,0)$, $\vec{I_1} = (0,1,0)$, $\vec{I_2} = (1,0,1)$; then $S_{BM}(\vec{q},\vec{I_1}) = 0*1+1*0.5 = 0.5$, $S_{BM}(\vec{q},\vec{I_2}) = 1*1+1*0.5 = 1.5$, which indicates that I_2 is more similar to q than I_1.

4.1.2 *Vector model.* VM generates feature vectors for database images and queries by assigning non-binary weights to keyblocks based on their frequency of appearance in database images and queries. Let $f_{i,j}$ be the frequency of keyblock c_i appearing in image I_j and M_i be the number of images in which c_i is found. The normalized frequency $\hat{f}_{i,j}$ is given by:

$$\hat{f}_{i,j} = \frac{f_{i,j}}{\max_{1 \leq l \leq N} f_{l,j}}.$$

Another important concept, analogous to *inverse document frequency* in the text domain, is *inverse image frequency*, which is defined as

$$id\,f_i = log(\frac{M}{M_i}), \quad for\ c_i \in C.$$

Keyblock frequency describes how often a keyblock appears in the encoded image. It also provides one measure of how well a keyblock describes the image content. However, as with stopwords in the text domain, the keyblocks which appear in many images are not very useful for distinguishing relevant images from non-relevant ones. VM balances these two effects by defining the keyblock weights as $w_{i,j} = \hat{f}_{i,j} * id\,f_i$. Weights for query q, denoted as $w_i(q)$, are computed similarly.

VM defines the similarity measure as the inner product of two unit feature vectors

$$S_{VM}(\vec{q},\vec{I_j}) = \frac{\vec{I_j} \bullet \vec{q}}{|\,\vec{I_j}\,| \times |\,\vec{q}\,|} = \frac{\sum_{i=1}^{N} w_{i,j} * w_i(q)}{\sqrt{\sum_{i=1}^{N} w_{i,j}^2} * \sqrt{\sum_{i=1}^{N} w_i(q)^2}}.$$

4.1.3 *Histogram model.* HM can be regarded as a significant and special case of VM. Here, $w_{i,j} = f_{i,j}$, or the frequency of c_i appearing in I_j. Similarly, $w_i(q) = f_i(q)$, or the frequency of c_i appearing in q. The feature vectors $\vec{I_j}$ and $\vec{q}$ are the *keyblock histograms*. The similarity measure is defined as

$$S_{HM}(\vec{q},\vec{I_j}) = \frac{1}{1+dis(\vec{q},\vec{I_j})},$$

where the distance function is

$$dis(\vec{q},\vec{I_j}) = \sum_{i=1}^{N} \frac{|w_{i,j} - w_i(q)|}{1 + w_{i,j} + w_i(q)}.$$

The sum is weighted to remove the excessive influence of frequently-appearing keyblocks on the similarity measure.

Note that the above models focus only on the appearance of individual keyblocks in images. The correlations between keyblocks are not addressed.

4.2 Correlation-enhanced models

We will now discuss the integration of keyblock correlations in the feature models. We define a *keyblock-keyblock correlation matrix* $(k_{i,l})_{N\times N}$ in which the rows and columns are associated with the keyblocks in a codebook C, where $|C| = N$. In this matrix, a normalized correlation factor $k_{i,l}$ between two keyblocks c_i and c_l can be defined as

$$k_{i,l} = \frac{n_{i,l}}{n_i + n_l - n_{i,l}},$$

where n_i is the number of images which contain keyblock c_i, n_l is the number of images which contain keyblock c_l, and $n_{i,l}$ is the number of images which contain both keyblocks, according to certain spatial configurations. In the current work, we have considered only the four kinds of neighborhood configuration illustrated in Figure 4, since the neighborhood correlation between keyblocks is usually more important than other relationships. $k_{i,l}$ measures the ratio between the number of images where both c_i and c_l appear and the total number of images where either c_i or c_l appear. If c_i and c_l have many co-occurrences in images, then the value of $k_{i,l}$ increases, and the images are considered to be more correlated.

We can use the keyblock-keyblock correlation matrix to redefine the feature vectors presented in Section 4.1. A new histogram model, termed the *correlation-enhanced histogram model*, can be defined. In this new model, let $\vec{I_j} = \{w_{1,j}, ..., w_{i,j}, ..., w_{N,j}\}$ be the feature vector for image I_j; then

$$w_{i,j} = f_{i,j} + f^*_{i,j} = f_{i,j} + \sum_{c_l \in I_j \bigwedge k_{i,l} \geq \alpha} f_{l,j} * k_{i,l},$$

where $f_{i,j}$ is the frequency of keyblock c_i appearing in image I_j, and $f^*_{i,j}$ is the correlation weight calculated by combining the frequencies of c_i's correlated keyblocks together with their correlation factor. The feature vector for query q is defined similarly. These new feature vectors incorporate the effect of keyblock itself as well as the keyblock correlationships. Note that α is a threshold (usually $0.3 \leq \alpha \leq 0.5$) used to cut off the effects of less correlated keyblocks. The similarity measure and distance function are same as those defined for the histogram model.

4.3 Context-sensitive feature models

We will now present more sophisticated models which consider the correlation between keyblocks in images. Our purpose is to extract feature vectors which not only include information regarding the occurrence of blocks but also the context of neighboring blocks. Similar considerations have been incorporated in the statistical modeling of language, where n-gram models use the history of the preceding $n-1$ words to predict the next word in a text document or in a speech stream [Jurafsky and Martin 2000]. It was observed that, in many circumstances, the occurrence of a word in a document is strongly related to the document history. In the keyblock framework, since images have linear representation properties similar to text documents, context information can also be obtained for keyblocks.

For a given number n, the probability of a complete string of codes (indices of keyblocks) $c_1c_2...c_n$ in an encoded image distributed according to some geometric configuration (or

correlation) [Rao et al. 2000] characterizes the distribution correlation of these n codes in the image. Let substrings $c_1c_2...c_k$ be c_1^k, for $k = 1, 2, ..., n$. If we consider the occurrence of each code in its correct location as an independent event, then, according to the chain rule,[1] this probability is represented as

$$P(c_1^n) = \prod_{k=1}^{n} P(c_k|c_1^{k-1}),$$

where $P(c_1|c_1^0) = P(c_1)$ and $P(c_k|c_1^{k-1})$ is the conditional probability of code c_k, given a long sequence of preceding codes c_1^{k-1}.

It is quite difficult to accurately compute these conditional probabilities, but approximations may be made on the basis of some assumptions. The following are commonly-used assumptions which underlie various models:

—*Uni-block*: The distributions of the codes (the keyblocks in the image) are independent; i.e.

$$P(c_k|c_1^{k-1}) = P(c_k), \ \ so \ \ P(c_1^n) = \prod_{k=1}^{n} P(c_k).$$

This assumption leads to the *uni-block* (one-block) model, which is the histogram model discussed in the preceding section.

—*Bi-block*: The distribution of a code (keyblock) depends only on the previous code (keyblock); i.e.

$$P(c_k|c_1^{k-1}) = P(c_k|c_{k-1}), \ \ so \ \ P(c_1^n) = \prod_{k=1}^{n} P(c_k|c_{k-1}).$$

This assumption leads to the *bi-block* (two-block) model or *Markov* assumption.

—*Tri-block*: The distribution of a code depends only on the preceding two codes; i.e.

$$P(c_k|c_1^{k-1}) = P(c_k|c_{k-2}c_{k-1}), \ \ so \ \ P(c_1^n) = \prod_{k=1}^{n} P(c_k|c_{k-2}c_{k-1}),$$

where we have conventions

$$P(c_1|c_{-1}c_0) = P(c_1), \ \ P(c_2|c_0c_1) = P(c_2|c_1).$$

This assumption underlies the *tri-block* (three-block) model.

These assumptions can be generalized to an arbitrary length n. However, the larger the n, the more difficult and costly the approximation of the conditional probabilities becomes, and the larger the training data set is required.

To estimate the probabilities, various code correlation-ships are used. For the bi-block model, a code correlation with cardinality 2 is considered, and for the tri-block model, a code correlation with cardinality 3 is considered. These two cases are discussed in more detail below.

[1]The chain rule states that, to calculate the joint probability of a number of events happening, the calculation can be decomposed into a series of simpler calculations. For example, if A, B, C, D, E are five independent events, then $P(ABCDE) = P(E|ABCD) * P(D|ABC) * P(C|AB) * P(B|A) * P(A)$.

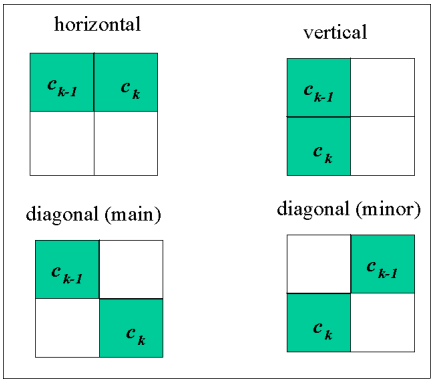

Fig. 4. Bi-block code configurations

4.3.1 *Bi-block model.* Given an encoded image with a codebook C and a code string $s \in C^*$, where C^* is the set of all possible code strings generated on C, we count the occurrences of s in the image and denote the number as $Num(s)$. Then, for $\forall c_k, c_{k-1} \in C$, the probability of string $c_k c_{k-1}$ is

$$P(c_k c_{k-1}) = \frac{Num(c_{k-1}c_k)}{\sum_{c_1,c_2 \in C} Num(c_1 c_2)}, \tag{3}$$

and the conditional probability is

$$\begin{aligned} P(c_k|c_{k-1}) &= \frac{Num(c_{k-1}c_k)}{\sum_{c \in C} Num(c_{k-1}c)} \\ &= \frac{Num(c_{k-1}c_k)}{Num(c_{k-1})}. \end{aligned} \tag{4}$$

Since the encoded image is a matrix of code indices, there are four choices for configuring a pair of neighboring codes c_{k-1} and c_k:

—*Horizontal configuration*: c_{k-1} precedes c_k horizontally. The probability and the conditional probability are denoted as $P_{-}(c_k c_{k-1})$ and $P_{-}(c_k|c_{k-1})$, respectively.

—*Vertical configuration*: c_{k-1} precedes c_k vertically. The probability and the conditional probability are denoted as $P_{|}(c_k c_{k-1})$ and $P_{|}(c_k|c_{k-1})$, respectively.

—*Diagonal (main) configuration*: c_{k-1} precedes c_k in the main diagonal direction. The probability and the conditional probability are denoted as $P_{\diagdown}(c_k c_{k-1})$ and $P_{\diagdown}(c_k|c_{k-1})$, respectively.

—*Diagonal (minor) configuration*: c_{k-1} precedes c_k in the minor diagonal direction. The probability and the conditional probability are denoted as $P_{/}(c_k c_{k-1})$ and $P_{/}(c_k|c_{k-1})$, respectively.

These configurations are listed in Figure 4. According to the configurations, we can extract four types of feature vectors for an image by using the probability of string $c_k c_{k-1}$: for $1 \le i, j \le |C|$,

(1) $P_{-} = P_{-}(c_i c_j)$, (2) $P_{|} = P_{|}(c_i c_j)$, (3) $P_{\diagdown} = P_{\diagdown}(c_i c_j)$, (4) $P_{/} = P_{/}(c_i c_j)$.

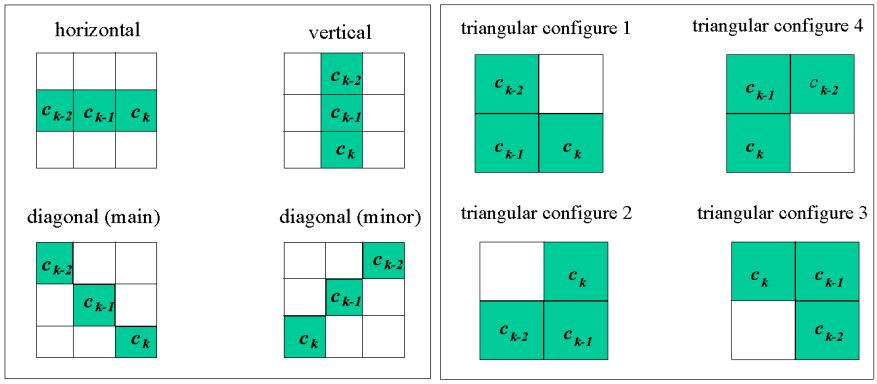

Fig. 5. Line and triangular configurations of tri-blocks

If the conditional probability of string $c_k c_{k-1}$ is used, then four additional types of feature vectors can be extracted: for $1 \le i, j \le |C|$,

(1) $P_{\_} = P_{\_}(c_i|c_j)$, (2) $P_{|} = P_{|}(c_i|c_j)$, (3) $P_{\backslash} = P_{\backslash}(c_i|c_j)$, (4) $P_{/} = P_{/}(c_i|c_j)$.

Each of the above $|C|^2$-dimensional feature vectors can be regarded as the *co-occurrence matrix* or *correlation matrix* $(w_{ij})_{|C|\times|C|}$ of an image, where each entry w_{ij} in this matrix represents the co-occurrence or correlation of two keyblocks in the image.

For the similarity measure, given one type of feature vector $\vec{I} = (w_{ij}), \vec{q} = (w'_{ij})$ for an image I and a query q, where $1 \le i, j \le |C|$, the similarity is defined by

$$sim(\vec{q}, \vec{I}) = \frac{1.0}{1.0 + dis(\vec{q}, \vec{I})}, \tag{5}$$

where the distance could be defined by any distance measure, such as L^1 and the *weighted sum*:

$$dis(\vec{q}, \vec{I}) = \sum_{1\le i,j\le|C|} \frac{|w_{ij} - w'_{ij}|}{1.0 + w_{ij} + w'_{ij}}.$$

4.3.2 *Tri-block model.* With the tri-block model, for $\forall c_k, c_{k-1}, c_{k-2} \in C$, we use $Num(c_{k-2}c_{k-1}c_k)$ to denote the number of occurrences of code string $c_{k-2}c_{k-1}c_k$ in an image. The probability is then calculated as

$$P(c_k c_{k-2} c_{k-1}) = \frac{Num(c_{k-2}c_{k-1}c_k)}{\sum_{c_1,c_2,c_3\in C} Num(c_1c_2c_3)}, \tag{6}$$

and the conditional probability is

$$\begin{aligned} P(c_k|c_{k-2}c_{k-1}) &= \frac{Num(c_{k-2}c_{k-1}c_k)}{\sum_{w\in C} Num(c_{k-2}c_{k-1}w)} \\ &= \frac{Num(c_{k-2}c_{k-1}c_k)}{Num(c_{k-2}c_{k-1})}. \end{aligned} \tag{7}$$

With the tri-block model, in addition to the four possible line configurations similar to the bi-block model, there are four possibilities arising from the triangular configurations,

as shown in Figure 5. For a given configuration in one of the eight possibilities, we can extract $|C|^3$-dimensional feature vectors by using either the probability or the conditional probability of string $c_k c_{k-2} c_{k-1}$. The feature vectors of an image I and a query q have the form $\vec{I} = (w_{ijk}), \vec{q} = (w'_{ijk})$, where $1 \leq i, j, k \leq |C|$. The similarity between I and q can be defined using Equation (5), where $dis(\vec{q}, \vec{I})$ is defined as

$$dis(\vec{q}, \vec{I}) = \sum_{1 \leq i,j,k \leq |C|} \frac{|w_{ijk} - w'_{ijk}|}{1.0 + w_{ijk} + w'_{ijk}}.$$

4.3.3 *Identical bi-block and identical tri-block models.* For a codebook C of size $|C|$, the bi-block model generates feature vectors of dimension $|C|^2$, and the tri-block model generates feature vectors of dimension $|C|^3$. If all bi-blocks and tri-blocks are used, the dimensions of the extracted feature vectors are very high. This is the main disadvantage of using the bi-block and tri-block models in image retrieval. High-dimensional feature vectors quickly consume storage capacity, reduce efficiency by complicating the application of indexing techniques, and diminish retrieval performance because they provide only sparse representation of image content.

To alleviate the above problems caused by the high dimensional feature vectors generated by the bi-block and tri-block models, we can select the most useful bi-blocks or tri-blocks to formulate feature vectors. The selection of useful *n*-blocks can be achieved by using clustering algorithms. The set of all bi-blocks or tri-blocks of an image is quantized into several clusters, with each element in a cluster represented by the centroid of the cluster. Also, since a large percentage of feature vector elements are empty, hashing can be used both to store feature vectors and to calculate similarity between two vectors.

Feature vectors can also be simplified by using the identical bi-block and identical tri-block models. The identical bi-block model is similar to the bi-block model but captures only the spatial correlations between identical keyblocks. That is, when calculating $P(c_k c_{k-1})$ or $P(c_k|c_{k-1})$ as presented in Section 4.3.1, $c_k = c_{k-1}$ is required. In other words, it takes only the elements on the diagonal line of the *co-occurrence matrix* generated by the bi-block model as the feature vector, thus reducing the dimension of the feature vectors to $|C|$.

Similarly, the identical tri-block model captures the co-occurrence of three neighboring identical keyblocks, thus reducing the dimension of the feature vectors to $|C|$. When calculating $P(c_k c_{k-2} c_{k-1})$ or $P(c_k|c_{k-2} c_{k-1})$ as presented in Section 4.3.2, $c_k = c_{k-1} = c_{k-2}$ is required.

Although the identical models may lose some information captured by the original bi-block and tri-block models, they offer the advantage of compact feature vectors which facilitate indexing and retrieval, especially for large image databases.

4.4 Feature combination model

The models described above capture different image content under various contexts. For example, the uni-block model only considers the occurrence of a single keyblock, while the bi-block and tri-block models consider the co-occurrence of multiple keyblocks. Even the same model applied to keyblocks of different size will focus on image content of different granularities. Since any single model cannot satisfy the entire range of requirements for image content extraction and retrieval, it is necessary to combine models to improve retrieval performance.

Table I lists all the models presented in the previous sections and the length of their corresponding feature vectors, given a codebook with $|C|$ keyblocks. Combinations based on the bi-block and tri-block models are omitted due to the high dimensionality of their feature vectors.

Model	Feature Vector Length (FVL)
Boolean Model	$\|C\|$
Vector Model	$\|C\|$
Histogram/Uni-block Model	$\|C\|$
Bi-block Model	$\|C\|^2$
Tri-block Model	$\|C\|^3$
Identical Bi-block Model	$\|C\|$
Identical Tri-block Model	$\|C\|$
Correlation-enhanced Histogram Model	$\|C\|$

Table I. Keyblock-based image retrieval models and their corresponding feature vector lengths (FVL), given a codebook with $|C|$ keyblocks

Many possible combinations of these models can be made. Without loss of generality, consider the combination of two models α and β, both with the same feature vector length. Let $\{\alpha_1,...,\alpha_i,...,\alpha_N\}$ be the feature vector of an image I in model α, $\{\beta_1,...,\beta_i,...,\beta_N\}$ be the feature vector of I in model β, and $\{\gamma_1,...,\gamma_i,...,\gamma_N\}$ be the feature vector in the combination model γ. Two strategies can be employed to define the feature combinations.

—*Feature fusion models:*

In this model,

$$\gamma_i = \alpha_i * w_\alpha + \beta_i * w_\beta, 1 \leq i \leq N,$$

where w_α and w_β are weights satisfying $0 \leq w_\alpha, w_\beta \leq 1$ and $w_\alpha + w_\beta = 1$. Each element of the new feature vector is the merged result of the corresponding elements of the feature vectors in the original models. The optimal values of w_α and w_β must be experimentally obtained. The similarity measure and the distance function are defined as described for the histogram model in Section 4.1.

—*Feature composite models:*

In this model,

$$\gamma_i = (\alpha_i, \beta_i), 1 \leq i \leq N.$$

The elements of the new feature vector are the composite of the corresponding elements of the feature vectors in the combined models. Thus, a feature vector of form

$$\{(\alpha_1,\beta_1),...,(\alpha_i,\beta_i),...,(\alpha_N,\beta_N)\}$$

is a vector of bi-pairs, one for each keyblock. Let the i-th bi-pair element in the feature vector for image I be (α_i,β_i), and for query q be (α'_i,β'_i); then, the distance between q and I is defined as:

$$dis(\vec{q},\vec{I}) = \sum_{i=1}^{N}[\frac{|\alpha_i - \alpha'_i|}{1+\alpha_i+\alpha'_i} + \frac{|\beta_i - \beta'_i|}{1+\beta_i+\beta'_i}].$$

The similarity measure is defined as

$$S_{FCM}(\vec{q},\vec{I}) = \frac{1}{1+dis(\vec{q},\vec{I})}.$$

Our experiments show that the feature composite model always achieves higher performance than the feature fusion model. We therefore usually use the feature composite model as the feature combination model.

Some model combinations deserve additional investigation. For example, the combination of the uni-block model with the identical bi-block model may improve the capability of system in capturing image content because these models capture different image content under various contexts. The combination of models under different keyblock size is also important because that might improve the capability of system in capturing image content at different granularities.

It should also be noted that, in the retrieval phase, for each query image, we can also combine the result list of top matches returned by different models to get a revised result list. This technique, *Result Fusion*, is described in [Zhu et al. 2000b; Zhu and Zhang 2000].

5. EXPERIMENTS

Comprehensive experiments are conducted to investigate the effectiveness of the keyblock approach. First, results will be presented for two small databases with well-defined ground truth. The overall performance of the proposed approach is then evaluated with a large-scale COREL image database containing 31646 photographs.

5.1 Experiments on small databases

The two small databases tested consists of web color images (CDB) and grey-scale Brodatz [Brodatz 1966] texture images (TDB), respectively. For the CDB, the proposed approach is compared with the traditional color histogram [Swain and Ballard 1991] and color coherent vector [Pass et al. 1996] techniques. For the TDB, it is compared with the wavelet texture techniques [Smith and Chang 1994; Strang and Nguyen 1996].

5.1.1 *CDB and TDB setup.* The CDB consists of 500 images manually divided into 41 well-defined groups, with each group containing either 10 or 20 similar images. The criterion for manual grouping is based on overall appearance. Image sizes vary from 364×369 to 807×1061. The images are downloaded from a variety of websites so that no pre-restrictions, such as different camera models and lighting conditions are assumed for testing. CDB serves as a small snapshot of representative Internet images. The database includes 120 art images classified into 11 groups such as paintings or calligraphy, and 380 real-world images which are categorized into 30 classes such as scenery, human activities, models, animals, and arsenals. To select the training images for codebook generation, 40 images (8% of the total number) are randomly chosen.

The TDB database consists of 2240 grey-scale Brodatz texture images of size 96×96, which are divided into 112 categories. Each category consists of 20 images with similar texture. We randomly select 5% of the total number of images (112) as the training set.

5.1.2 *Keyblock generation.* Three block sizes, 2×2, 4×4, and 8×8, are used to generate keyblocks for both CDB and TDB. In addition, CDB is also tested with a 16×16 block size, since its images are in larger size than those of TDB. Intuitively, it is expected

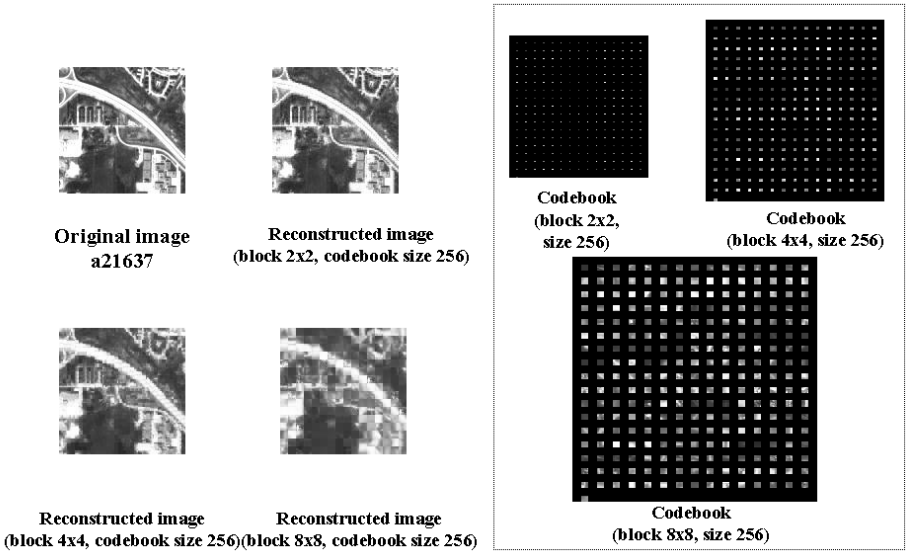

Fig. 6. A raw image and the encoded images after reconstruction with different codebooks. Each codebook is obtained with the same training set.

that blocks of different sizes may capture information at different granularities. Smaller blocks usually exploit local aspects of image content, such as edges and regions with high spatial frequency. Larger blocks may capture correlations among neighboring sub-blocks as well as an overview of the global variation.

For each block size, experiments are performed to generate codebooks of four different sizes – 256, 512, 1024, and 2048. In this implementation, the distortion, which is the objective function for optimization when generating a codebook, is the Euclidean distance (d) between the vectors of the intensity values of an original block and that of the corresponding keyblock. In short, the testing is conducted with 16 (4 block sizes $\times$ 4 codebook sizes) codebooks for CDB, and with 12 (3×4) codebooks for TDB. After the codebooks are generated, all images in the databases are then encoded correspondingly.

As an example, Figure 6 shows an image and its reconstructed images using different codebooks. Each codebook is obtained by applying the algorithm described in Section 3 to the selected training set. Table II presents the average distortion of CDB and TDB with each of the codebooks. The average distortion for an image database D is calculated by

$$d_{avg} = \frac{\sum_{i \in D} d(i, \hat{i})}{|D|}, \tag{8}$$

where i is an image and $\hat{i}$ is the reconstructed image after decoding. The table shows the relationship between the average distortion of a given database and various codebooks with different keyblock sizes. For a given codebook size, a smaller keyblock size yields lower distortion. On the other hand, for a given block size, a larger the codebook yields smaller distortion. Therefore, block size and codebook size will affect retrieval performance because a large distortion leads to a large degradation of image quality after encoding, thus

	blocks	256	512	1024	2048
CDB	2×2	615.57	464.10	374.95	311.37
	4×4	1000.70	888.58	799.83	707.34
	8×8	1602.50	1444.33	1346.86	1258.06
	16×16	2373.23	2182.95	2084.16	1976.10
TDB	2×2	129.47	111.15	75.45	59.82
	4×4	485.96	423.99	354.98	314.33
	8×8	895.21	855.22	800.99	764.54

Table II. Average distortion of the databases with different codebooks

possibly misleading the image retrieval process.

5.1.3 *Image feature representation, retrieval and comparison of results.* Experiments using the BM, HM and VM models are performed on CDB and TDB with different codebooks. For each codebook, database images are encoded correspondingly and their feature vectors are also generated following the keyblock-based image feature representation procedure presented in Section 4.1. The dimensions of each feature vector are the same as the respective codebook sizes – e.g., 256, 512, 1024, or 2048. For evaluation and comparison, the database is also taken as the visual query set for testing. So, there are 500 query examples for CDB and 2240 query examples for TDB. For each query, the precision and recall pairs corresponding to the top 1, 2, ..., through 40 retrieved images are calculated. The average precision and recall are then calculated over all queries for a given database.

Two conventional color feature techniques, the traditional color histogram proposed by Swain and Ballard [Swain and Ballard 1991] and the color coherent vector (CCV) proposed by Pass and Zabih [Pass et al. 1996] are also applied to CDB as a comparison. The color space, the HVC (*hue, brightness and saturation*), is quantized into blocks by performing uniform partitions in each color dimension. For the traditional histogram, the HVC space is quantized into $32 \times 8 \times 8$ blocks so that the histogram vector is of dimension 2048; for CCV, the HVC space is quantized into $16 \times 8 \times 8$ blocks so that the coherent vector and the non-coherent vector together contribute a vector of dimensions 2048. The similarity measure is simply the Euclidean distance measured by regarding each feature vector as a one-dimensional vector.

Haar and Daubechies wavelet texture techniques [Smith and Chang 1994; Strang and Nguyen 1996] are applied to TDB for comparison. Three levels of sub-band decomposition are performed on each image. Each of the sub-bands obtained after filtering has uniform texture information. Each three-level transformation produces 10 sub-bands. Features were extracted by energy estimation in sub-bands. Two energy features, mean and variance, are computed for each sub-band. Thus, each feature vector has 20 elements. More details on this approach can be found in [Sheikholeslami 1999].

Figure 7 presents the results of this comparative testing. Figure 7 (a) and (b) indicate that BM, HM, and VM all outperform traditional techniques. For example, in the case of CDB, almost 70% of the relevant images are among top 40 images retrieved by HM and BM, while only 40% are returned by CCV. In the case of TDB, at a recall level of 0.2, the precisions for each model are 0.96 (HM), 0.91 (BM), 0.90 (VM), 0.88 (Haar wavelets), and 0.81 (Daubechies); hence, HM performs best. Figure 7 (c) and (d) show comparisons of the proposed models with fixed parameters. Figure 7 (c) shows that, for codebooks with 4×4 blocks and of size 1024 or 512, BM performs best on CDB, while HM performs best

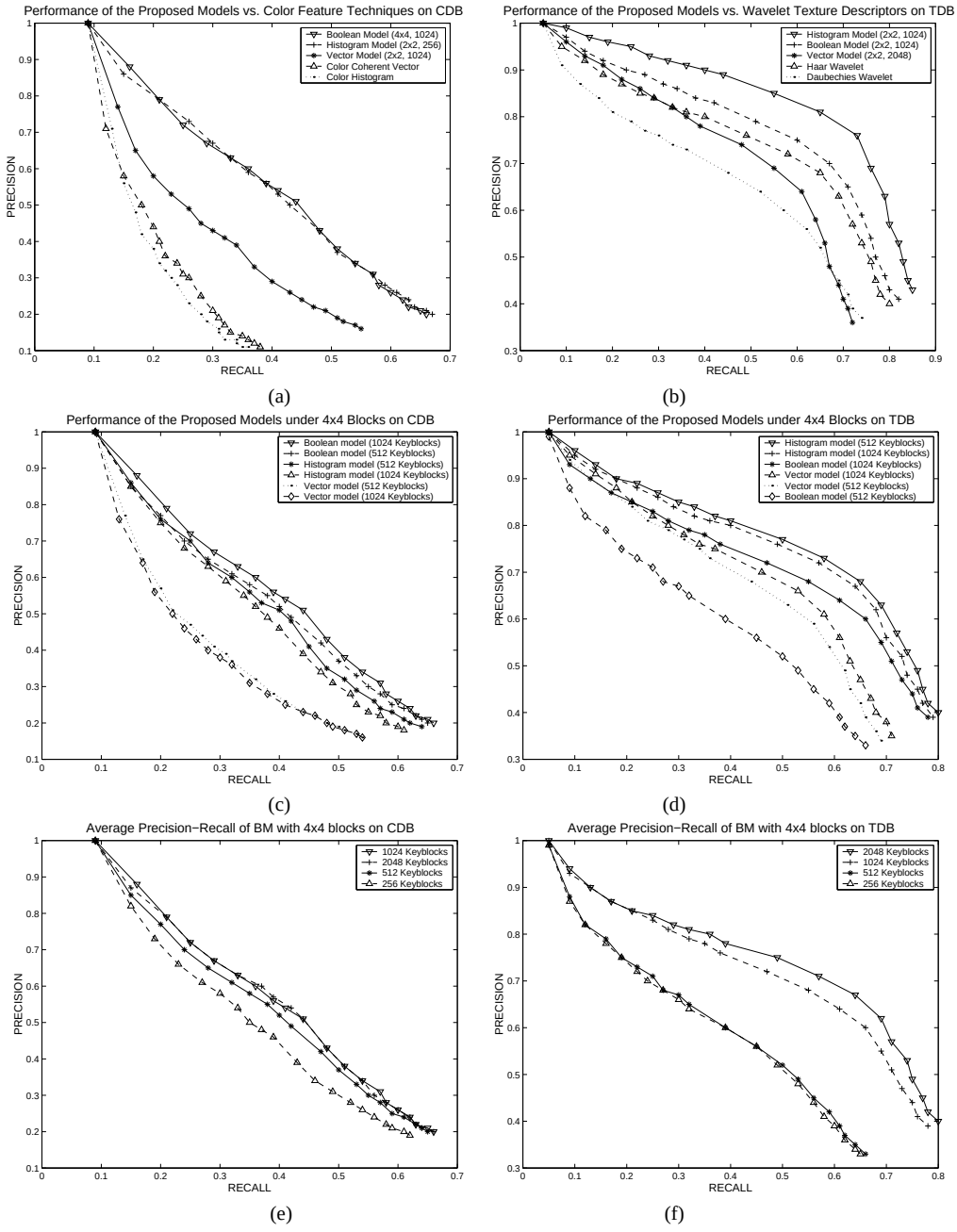

Fig. 7. Average precision-recall on CDB and TDB: (a), (b) present a comparison of BM, HM, and VM with CCV and the traditional color histogram on CDB, and with Haar and Daubechies wavelets on TDB, respectively; (c), (d) present additional comparisons among BM, HM and VM for fixed parameters; (e), (f) show the effects of codebook size on performance.

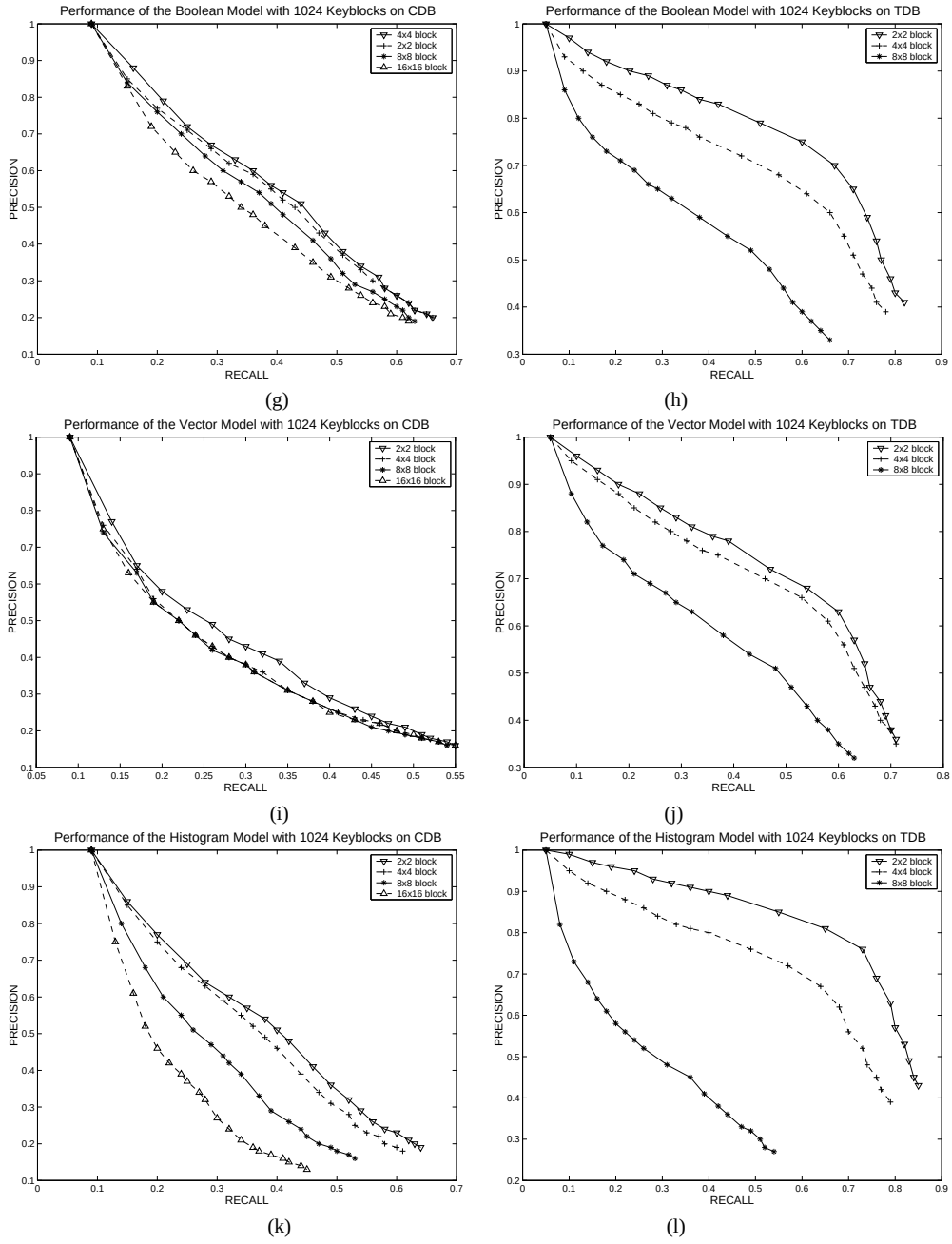

Fig. 8. The effects of block size on the average precision-recall of each model on CDB and TDB: (g), (h) are for BM; (i), (j) are for VM; and (k), (h) are for HM. HM is more sensitive to block size than BM and VM. Moreover, it seems that the sensitivity of every model is increased with TDB.

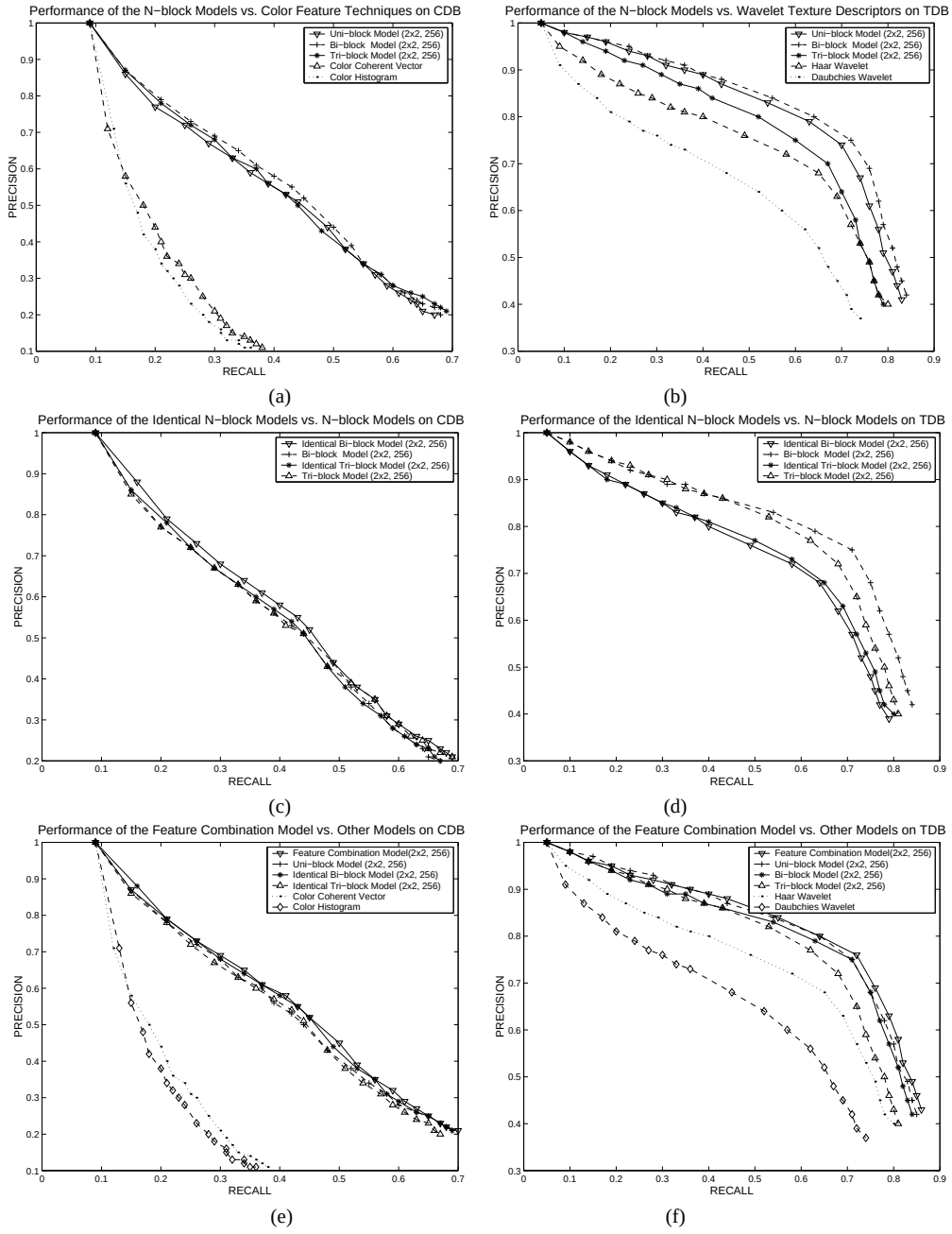

Fig. 9. Average precision-recall: (a) (b) comparison of N-block models with CCV and the traditional color histogram on CDB, and with Haar and Daubechies wavelets on TDB, respectively; (c) (d) comparison between identical N-block models and N-block models; (e) (f) the effectiveness of the feature combination model.

on TDB. Figure 7 (e) and (f) demonstrate the effect of codebook size on performance, using BM with 4×4 keyblocks as an example. On CDB, the performance is not as sensitive as on TDB with respect to the codebook size. In addition, for a fixed keyblock size, increasing codebook size yields better performance. Together with Table II, these results verify the assumption that a smaller distortion leads to better performance.

Figure 8 presents the effect of block size on the three models. In general, for each model and for a fixed codebook size, a smaller block size leads to better performance. Again, this verifies that performance of a keyblock-based image retrieval model improves with reduced distortion of the encoded images. This is especially true for TDB because it is a slightly simpler database. However, for both databases, in the extreme case of the least distortion when 1×1 blocks (pixels) is used (which reduces to the color quantization [Orchard and Bouman 1991]), the performance is inferior to the cases with 2×2 and 4×4 blocks. For example, the best average precision with 1×1 blocks achieved at the top five retrieved CDB images is only 0.65, which is less than 0.68 for the case of 2×2 blocks. Precision drops because 1×1 blocks do not reflect local spatial information which is useful for retrieval. Moreover, it is too costly to encode an image with 1×1 blocks. Note that GLA has a complexity of $O(l * C * I)$, where l is the size of the training sequence, C is the codebook size, and I is the iteration time. Thus, the smaller the block size, the larger the training sequence, and the longer the image encoding time. Smaller blocks are therefore more computationally expensive. Experimental results indicate that 2×2 and 4×4 block sizes strike a good balance. Figure 8 also shows that HM is more sensitive to block size than the other two models and that the sensitivity of every model is higher with TDB than with CDB. It also shows that HM performs well with 2×2 blocks.

It may also be observed that the optimal parameters and the model are database-dependent. For example, HM with a $(2 \times 2, 256)$ codebook performs very well in both databases. With its shorter feature vectors, it requires less storage and computational power and simplifies indexing. Considering all factors, using HM with a $(2 \times 2, 256)$ codebook would appear to be an optimal approach.

The bi-block and tri-block models are also tested on the CDB and TDB databases with different codebooks. It is observed that the four different configurations of the bi-block model achieve very similar performance using the conditional probability of keyblocks as presented in Sections 4.3.1 and 4.3.2; this is also true for the tri-block model. Thus, we will discuss only the horizontal configuration. Figure 9 (a) and (b) indicate that all three *n*-block models achieve higher performance than the traditional techniques, while the bi-block and uni-block models perform better on these two datasets. The performance of the tri-block model itself is not as good as the uni-block and bi-block models, especially on TDB. This is due to the fact that, as discussed in Section 4.3, the feature vectors in the tri-block model have high dimensions and may be sparse in representing image content. Figure 9 (c) and (d) show that identical *n*-block models achieve equivalent or even higher performance than the *n*-block models on CDB, but this does not hold true for TDB. Also, from Figure 9 (e) and (f), it is clear that the feature combination model, which combines the feature vectors generated by the uni-block, bi-block and tri-block models, performs best in all cases. This verifies that different models capture different aspects of image content, and it is fruitful to combine their representational capabilities to further improve retrieval performance.

5.2 Experiments on a large database of images

Experiments are also conducted on a general-purpose image database (denoted COREL) which consists of 31,646 color photos from CD7 and CD8 of *COREL Gallery 1,300,000*. These photos are stored in JPEG format with at sizes of either 120×80 or 80×120. Since the content of these photos is highly diverse and complex, accurate retrieval and subsequent evaluation are non-trivial. For this database, the performance of the keyblock approach is compared with CCV and the color histogram technique.

To generate the codebooks for COREL, we randomly select 320 images (around 1% of the total image set) as the training set. Three block sizes, 2×2, 4×4, and 8×8, are used. For each block size, experiments have been performed to generate codebooks of sizes 256, 512, and 1024. Testing is therefore conducted with nine (3 block sizes $\times$ 3 codebook sizes) codebooks. After the codebooks are generated, each image in the database is encoded correspondingly.

We have conducted comprehensive experiments for feature extraction and retrieval on COREL. In general, the keyblock approach performs better than CCV and the color histogram technique. Figure 10 shows the retrieval results of different approaches on a fish image. For each approach, the query image is at the upper-left corner. The two numbers below the images are the image ID and the similarity between the query and the matched image. The quality of the top 12 images returned by the different keyblock approaches is apparently better because the semantics of these images are more closely related to the query image.

To evaluate retrieval performance, we have categorized 6895 images into 82 categories. These categories include human activities such as skiing and swimming, objects such as tables and tools, animals such as goats and sharks, landscapes such as beach and forest, etc. These 6895 images are also served as the query set. For each query, the precision and recall corresponding to the top 1, 2, ..., through 100 retrieved images are calculated. Finally, the average precision is calculated over all 6895 queries.

Figure 11 (a) shows that the keyblock approach outperforms traditional techniques. For example, in the case of the feature combination model, 12% of relevant images are among top 100 retrieved images, while only 9% are returned by the color histogram and 6.5% by CCV. At each recall level, the keyblock approach offers higher precision. Figure 11 (b) and (c) present the effect of block size when the codebook size and model are fixed. In each case, the 2×2 block size achieves the best performance. Figure 11 (d) shows the effect of codebook size with fixed block size and model. For the uni-block model, the best performance is achieved with a codebook size of 256. The identical bi-block and identical tri-block models achieve the similar results. Figure 11 (e) compares the performance of different models when both the codebook size and block size are fixed. For example, with 256 (2×2) keyblocks, the uni-block model outperforms the identical bi-block and the identical tri-block models, and the feature combination model performs even better. The results for COREL, as with the small CDB database, show that the keyblock approach performs better than CCV and the color histogram technique.

Based on the best-case block size 2×2 and codebook size 256, we have constructed a keyblock-keyblock correlation matrix and implemented the correlation-enhanced histogram model. Figure 11 (f) shows that, by introducing a keyblock-keyblock correlation matrix describing keyblock correlations, the correlation-enhanced histogram model performs better than the histogram model. In this test, keyblock correlations are calculated

Fig. 10. Retrieval results of different approaches on a fish image (864072.jpg) in COREL

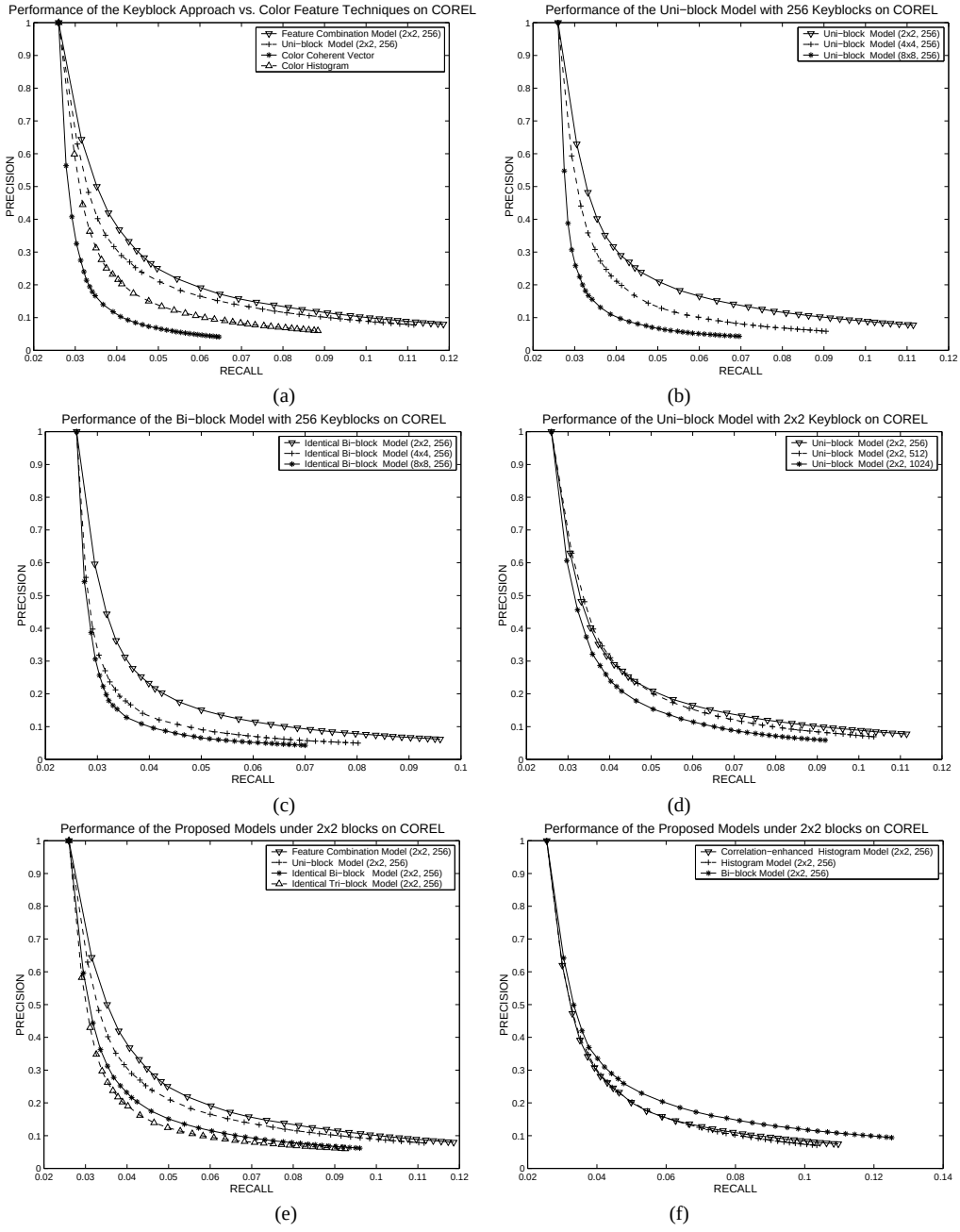

Fig. 11. Average Precision-Recall on COREL: (a) comparison of the keyblock approach with CCV and the color histogram; (b) (c) the effect of keyblock size on different models; (d) the effect of codebook size on the uni-block model; (e) (f) comparison of different models with 256 (2×2) keyblocks.

based on the statistics of the entire image database. If keyblock correlations are considered within individual images (e.g., as in the bi-block model), retrieval performance will be further improved. This can be verified by Figure 11 (f), since the bi-block model achieves the highest performance among the three models tested.

5.3 Discussion

As with the text-based retrieval, where different dictionaries may be used for different document sets, the keyblock framework may use different codebooks for different image datasets to achieve best retrieval performance. The selection of the training image set which well represents the image database is critical to keyblock generation. For domain-specific image databases, supervised keyblock generation is recommended, as in the case of keyblock generation for geographic images [Zhang and Zhu 2001]. The selection of the training set can thus be more delicate because the domain knowledge is available. For general-purpose image databases where no domain information exists, unsupervised keyblock generation randomly selects the training images.

In general, we can measure and compare the accuracy of the training set in representing the image database by calculating distortion and retrieval performance. Table III presents the average distortion (calculated by the Equation 8) of COREL using different codebooks and training sets with a codebook size of 256. This table shows that consistent average distortions are achieved by the training sets of COREL, while large distortions are obtained when the codebooks generated from CDB are used. It also indicates that our sampling approach remains quite stable with training sets representing different percentages of the entire database.

Keyblock Size	**CDB Codebook**	**COREL Training Set Size**			
		160 *images*(0.5%)	320 (1%)	1600 (5%)	3200 (10%)
2×2	741.28	586.97	579.18	579.12	581.18
4×4	1385.42	1210.63	1175.49	1182.5352	1168.09

Table III. Average distortion of COREL with different codebooks and training sets

It is also observed that there is a relationship between retrieval performance and the average distortion achieved by different training sets. In general, a large distortion may lead to severe degradation of the quality of images after encoding, thus misleading the image retrieval process. To test this assumption, we have conducted experiments on COREL using the codebooks of CDB. Figure 12 (a) and (b) compare the respective retrieval performances of the histogram model when the COREL and CDB codebooks are used. Retrieval performance falls when the CDB codebooks are used, because their keyblocks do not represent the content of COREL images as well as the keyblocks generated from the COREL training images. Thus, for each image database, it is most effective to generate keyblocks based on training images which are from that image database.

The experiments also reveal that for small image databases, 5% to 10% of the total image database should be selected as the training set, while, for large image datasets, it is sufficient to use 1% to 5% as the training set. For example, for COREL, we conducted experiments based on different training sets. Figure 12 (c) and (d) present the retrieval performance of the histogram model with 256 (2×2) and (4×4) keyblocks using four training sets formed by randomly selecting 0.5%, 1%, 5%, or 10% of the images from

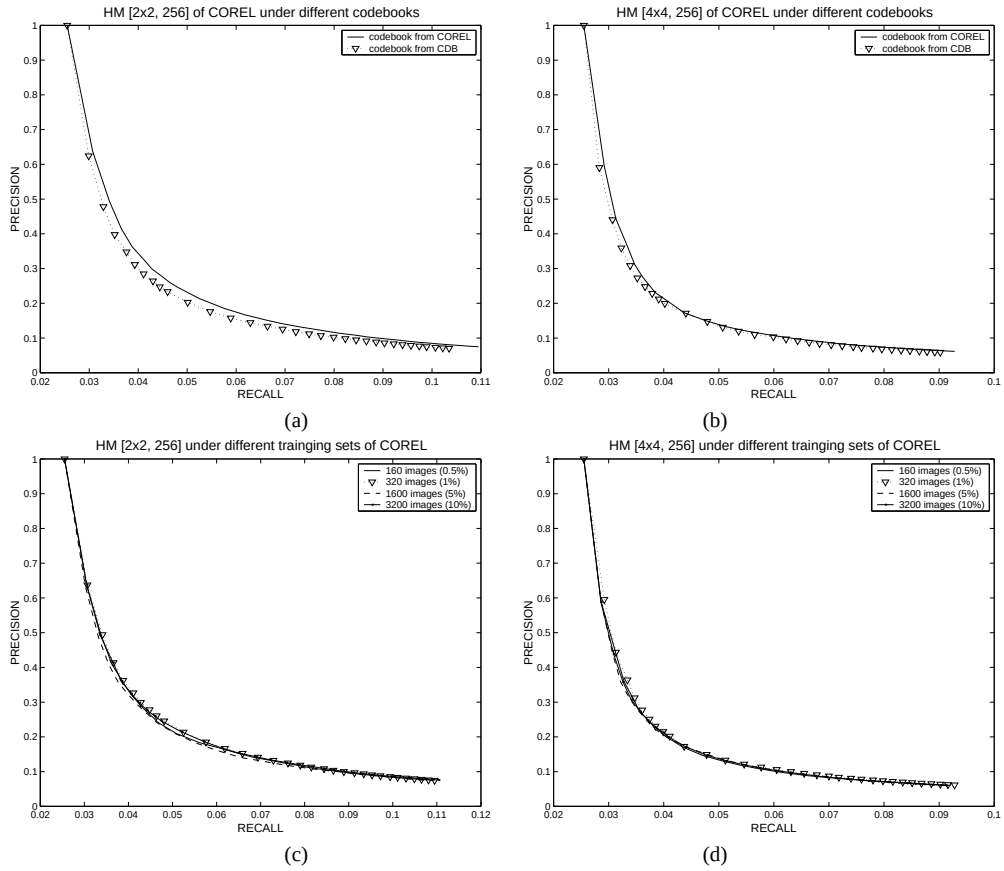

Fig. 12. Average precision-recall of the histogram model with 256 (2×2) or (4×4) keyblocks on COREL: (a) and (b) present retrieval performance with codebooks from COREL and CDB; (c) and (d) present the retrieval performance with different training sets of COREL.

the entire image set, respectively. Only small variations in performance occur between the four training sets. These results demonstrate that the proposed approach to training set selection is highly robust. From these two figures, we can also see that the retrieval performance achieved by using the codebook from CDB is lower than the performance achieved by each of the above four COREL-specific training sets.

6. CONCLUSION

This paper presents a new framework for content-based image retrieval, called the keyblock approach, which provides a practical solution for content-based image retrieval analogous to text-based IR techniques. Within this framework, we have developed strategies to generate keyblock codebooks for a variety of databases ranging from general-purpose image databases to domain-specific databases. Based on the keyblock representation formats, we have developed methods for extracting image features which are comprehensive descriptions of the content of the image. These features are more semantics-oriented than the existing lower-level features such as color and texture.

Experimental results presented here have demonstrated that the proposed models are superior not only to the color histogram and color coherent vector approaches which exploit color features, but also to wavelet approaches which exploit texture features. We have also demonstrated that the performance of the keyblock approach can be further improved if the codebook generated is domain-specific.

The proposed keyblock approach can be integrated with the relevance feedback methods to support a powerful, user-friendly browsing environment.

REFERENCES

AHUJA, N. AND ROSENFELD, A. 1981. Mosaic models for texture. *IEEE Transactions on Pattern Analysis and Machine Intelligence 3,* 1, 1–11.

BACH, J., FULLER, C., GUPTA, A., HAMPAPUR, A., HOROWITZ, B., JAIN, R., AND SHU, C. 1996. The virage image search engine: An open framework for image management. In *Proceedings of SPIE, Storage and Retrieval for Still Image and Video Databases IV*. San Jose, CA, USA, 76–87.

BAEZA-YATES, R. AND RIBIERO-NETO, B. 1999. *Modern Information Retrieval*. Addison Wesley.

BRODATZ, P. 1966. *Textures: A Photographic Album for Artists and Designers*. Dover, New York.

DOUGHERTY, E. AND PELZ, J. 1989. Texture-based segmentation by morphological granulometrics. In *Advanced Printing of Paper Summaries, Electronic Imaging '89*. Vol. 1. Boston, Massachusetts, 408–414.

ESTER, M., KRIEGEL, H., SANDER, J., AND XU, X. 1996. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of 2nd International Conference on KDD*. Portland, OR, 226–231.

FALOUTSOS, C., BARBER, R., FLICKNER, M., HAFNER, J., NIBLACK, W., PETKOVIC, D., AND EQUITZ, W. 1994. Efficient and effective querying by image content. *Journal of Intelligent Information Systems 3,* 3/4 (July), 231–262.

FLICKNER M., SAWHNEY H., NIBLACK W., ASHLEY J., HUANG Q. AND DOM B. ET AL. 1995. Query by Image and Video Content: The QBIC System. *IEEE Computer 28,* 9 (Sept.), 23–32.

GERSHO, A. AND GRAY, R. M. 1992. *Vector Quantization and Signal Compression*. Kluwer Academic Publishers.

HIRATA, K. AND KATO, T. 1993. Rough sketch-based image information retrieval. *NEC Research & Development 34,* 2, 263–273.

HORN, B. K. P. 1988. *Robot Vision,* Forth ed. The MIT Press.

HSU, W., CHUA, T., AND PUNG, H. K. 1995. An integrated color-spatial approach to content-based image retrieval. *Proceeding of the ACM Multimedia Conference,* 305 – 313.

HUANG, J. 1998. Color-spatial image indexing and applications. Ph.D. thesis, Cornell University.

HUANG, J., KUMAR, S., MITRA, M., ZHU, W., AND ZABIH, R. 1997. Image indexing using color correlograms. In *IEEE Conference on Computer Vision and Pattern Recognition.* 762–768.

HUNT, R. W. G. 1989. *Measuring Color*. Ellis Horwood series in applied science and industrial technology. Halsted Press, New York, NY.

IDRIS, F. AND PANCHANATHAN, S. 1996. Algorithms for indexing of compressed images. In *Proceedings of International Conference on Visual Information Systems.* Melbourne, 303–308.

JURAFSKY, D. AND MARTIN, J. H. 2000. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.* Prentice Hall.

KAUFMAN, L. AND ROUSSEEUW, P. J. 1990. *Finding Groups in Data: an Introduction to Cluster Analysis*. John Wiley & Sons.

KOHONEN, T., HYNNINEN, J., KANGAS, J., LAAKSONEN, J., AND TORKKOLA, K. 1995. Lvq pak: The learning vector quantization program package.

KORN, F., SIDIROPOULOS, N., FALOUTSOS, C., SIEGEL, E., AND PROTOPAPAS, Z. 1996. Fast nearest-neighbor search in medical image databases. In *Conference on Very Large Data Bases (VLDB96)*.

LU, G. AND TENG, S. 1999. A novel image retrieval technique based on vector quantization. In *Proceedings of International Conference on Computational Intelligence for Modelling, Control and Automation.* Vienna, Austria, 36–41.

MANDELBROT, B. 1977. *Fractals - Form, Chance, Dimension.* W.H. Freeman, San Francisco, California.

MANJUNATH, B. AND MA, W. 1996. Texture Features for Browsing and Retrieval of Image Data. *IEEE Transactions on Pattern Analysis and Machine Intelligence 18,* 8 (August), 837–842.

MEHROTRA, R. AND GARY, J. E. 1995. Similar-shape retrieval in shape data management. *IEEE Computer 28,* 9 (September), 57–62.

MODESTINO, J., FRIES, R., AND VICKERS, A. 1981. Texture discrimination based upon an assumed stochastic texture model. *IEEE Transactions on Pattern Analysis and Machine Intelligence 3,* 5, 557–580.

MOKHTARIAN, F., ABBASI, S., AND KITTLER, J. 1996a. Efficient and Robust Retrieval by Shape Content through Curvature Scale Space. In *Proc. International Workshop on Image Databases and MultiMedia Search*. Amsterdam, The Netherlands, 35–42.

MOKHTARIAN, F., ABBASI, S., AND KITTLER, J. 1996b. Robust and efficient shape indexing through curvature scale space. In *Proceedings of British Machine Vision Conference.* Edinburgh, UK, 53–62.

NETRAVALI, A. N. AND HASKELL, B. G. 1988. *Digital Pictures: representation and compression. Applications of Communications Theory.* Plenum Press, New York, NY.

NG, R. T. AND HAN, J. 1994. Efficient and Effective Clustering Methods for Spatial Data Mining. In *Proceedings of the 20th VLDB Conference.* Santiago, Chile, 144–155.

NIBLACK W., BARKER R., EQUITZ W., FLICKNER M., GLASMAN E. AND PETKOVIC D. ET AL. 1993. The qbic project: Querying images by content using color, texture, and shape. Tech. rep., IBM Technical Report.

ORCHARD, M. T. AND BOUMAN, C. A. 1991. Color Quantization of Images. *IEEE Transactions on Signal Processing 39,* 12 (December), 2677–2690.

PASS, G. AND ZABIH, R. 1996. Histogram refinement for content-based image retrieval. *IEEE Workshop on Applications of Computer Vision,* 96–102.

PASS, G., ZABIH, R., AND MILLER, J. 1996. Comparing images using color coherence vectors. In *Proceedings of ACM Multimedia 96*. Boston MA USA, 65–73.

PAUWELS, E., FIDDELAERS, P., AND GOOL, L. V. 1997. DOG-based unsupervized clustering for CBIR. In *Proceedings of the 2nd International Conference on Visual Information Systems.* San Diego, California, 13–20.

PENTLAND, A., PICARD, R., AND SCLAROFF, S. 1994. Photobook: Tools for Content-based Manipulation of Image Databases. In *Proceedings of the SPIE Conference on Storage and Retrieval of Image and Video Databases II*. 34–47.

PICARD, R. 1996. A society of models for video and image libraries. Tech. Rep. 360, MIT Media Laboratory Perceptual Computing.

RAO, A., SRIHARI, R. K., AND ZHANG, Z. 1999. Spatial color histograms for content-based image retrieval. *Proceedings of the Eleventh IEEE International Conference on Tools with Artificial Intelligence.*

RAO, A., SRIHARI, R. K., AND ZHANG, Z. 2000. Geometric histogram: A distribution of geometric configurations of color subsets. In *Proceedings of SPIE - Internet Imaging.* Vol. 3964. San Joes, California, 91–101.

RICKMAN, R. AND STONHAM, J. 1996. Content-based image retrieval using color tuple histograms. *SPIE proceedings: Symposium on Electronic Imaging: Science and Technology — Storage and Retrieval for Image and Video Databases IV 2670,* 2 – 7.

RUSS, J. C. 1995. *The Image Processing Handbook.* CRC Press, Boca Raton.

SAFAR, M., SHAHABI, C., AND SUN, X. 2000. Image retrieval by shape: A comparative study. In *Proceedings of IEEE International Conference on Multimedia and Exposition ICME.* USA.

SHAHABI, C. AND SAFAR, M. 1999. Efficient retrieval and spatial querying of 2d objects. In *IEEE International Conference on Multimedia Computing and Systems (ICMCS99).* Florence, Italy, 611–617.

SHEIKHOLESLAMI, G. 1999. Multi-resolution content-based image retrieval and clustering in large visual databases. Ph.D. thesis, Department of Computer Science and Engineering, State University of New York at Buffalo.

SHEIKHOLESLAMI, G., CHATTERJEE, S., AND ZHANG, A. 1998. *WaveCluster*: A Multi-Resolution Clustering Approach for Very Large Spatial Databases. In *Proceedings of the 24th VLDB conference.* 428–439.

SHEIKHOLESLAMI, G., CHATTERJEE, S., AND ZHANG, A. 2000. *WaveCluster*: A Wavelet-Based Clustering Approach for Multidimensional Data in Very Large Databases. *The VLDB Journal 8,* 4 (February), 289–304.

SHEIKHOLESLAMI, G. AND ZHANG, A. 1997. An Approach to Clustering Large Visual Databases Using Wavelet Transform. In *Proceedings of the SPIE Conference on Visual Data Exploration and Analysis IV*. San Jose, 322–333.

SMITH, J. R. AND CHANG, S.-F. 1994. Transform Features For Texture Classification and Discrimination in Large Image Databases. In *Proceedings of the IEEE International Conference on Image Processing.* 407–411.

SMITH, J. R. AND CHANG, S.-F. 1996a. Tools and techniques for color image retrieval. *SPIE proceedings 2670,* 1630 – 1639.

SMITH, J. R. AND CHANG, S.-F. 1996b. VisualSeek: a fully automated content-based image query system. In *Proceedings of ACM Multimedia 96*. Boston MA USA, 87–98.

STRANG, G. AND NGUYEN, T. 1996. *Wavelets and Filter Banks.* Wellesley-Cambridge Press, Wellesley, MA.

STRICKER, M. AND DIMAI, A. 1996. Color indexing with weak spatial constraints. *SPIE Proceedings 2670,* 29 – 40.

STROMBERG, W. AND FARR, T. 1986. A Fourier-based textural feature extraction procedure. *IEEE Transactions on Geoscience and Remote Sensing 24,* 5, 722–732.

SWAIN, M. AND BALLARD, D. 1991. Color Indexing. *Int Journal of Computer Vision 7,* 1, 11–32.

SYEDA-MAHMOOD, T. 1996. Finding shape similarity using a constrained non-rigid transform. In *International Conference on Pattern Recognition.*

TAO, Y. AND GROSKY, W. 1999. Delaunay triangulation for image object indexing: A novel method for shape representation. In *Proceedings of the Seventh SPIE Symposium on Storage and Retrieval for Image and Video Databases.* San Jose, California, 631–942.

WANG, J. AND ACHARYA, R. 1998a. Efficient access to and retrieval from a shape image database. In *IEEE Workshop on Content Based Access of Image and Video Libraries, (CBAIL 98)*. Santa Barbara.

WANG, J. AND ACHARYA, R. 1998b. A vertex based shape coding approach for similar shape retrieval. In *ACM symposium on Applied Computing.* Atlanta, GA, 520–524.

WANG, W., YANG, J., AND MUNTZ, R. 1997. STING: A Statistical Information Grid Approach to Spatial Data Mining. In *Proceedings of the 23rd VLDB Conference.* Athens, Greece, 186–195.

ZHANG, A. AND ZHU, L. 2001. Metadata generation and retrieval of geographical imagery. In *Proceedings of National Conference for Digital Government Research (dg.o2001)*. Los Angeles, California, USA, 76–83.

ZHANG, T., RAMAKRISHNAN, R., AND LIVNY, M. 1996. BIRCH: An Efficient Data Clustering Method for Very Large Databases. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*. Montreal, Canada, 103–114.

ZHU, L., RAO, A., AND ZHANG, A. 2000a. Advanced feature extraction for keyblock-based image retrieval. In *Proceedings of International Workshop on Multimedia Information Retrieval (MIR2000)*. Los Angeles, California, USA, 179–183.

ZHU, L., RAO, A., AND ZHANG, A. 2000b. Keyblock: An approach for content-based geographic image retrieval. In *Proceedings of First International Conference on Geographic Information Science (GIScience 2000)*. Savannah, Georgia, USA, 286–287.

ZHU, L. AND ZHANG, A. 2000. Supporting multi-example image queries in image databases. In *Proceedings of IEEE International Conference on Multimedia and Expo (ICME2000)*. New York City, NY, USA, 697–700.

ZHU, L., ZHANG, A., RAO, A., AND SRIHARI, R. 2000. Keyblock: An approach for content-based image retrieval. In *Proceedings of ACM Multimedia 2000.* Los Angeles, California, USA, 157–166.

...