Color Image Indexing Using BTC

Guoping Qiu

***Abstract*—This paper presents a new application of a well-studied image coding technique, namely block truncation coding (BTC). It is shown that BTC can not only be used for compressing color images, it can also be conveniently used for content-based image retrieval from image databases. From the BTC compressed stream (without performing decoding), we derive two image content description features, one termed the block color co-occurrence matrix (BCCM) and the other block pattern histogram (BPH). We use BCCM and BPH to compute the similarity measures of images for content-based image retrieval applications. Experimental results are presented which demonstrate that BCCM and BPH are comparable to similar state of the art techniques.**

***Index Terms*—Block truncation coding (BTC), color quantization, content-based image retrieval, image coding, image database.**

I. Introduction

THE rapid expansion of the Internet and fast advancement in color imaging technologies have made digital color images more and more readily available to professional and amateur users. The large amount of image collections available from a variety of sources (digital camera, digital video, scanner, the Internet, etc.) have posed increasing technical challenges to computer systems to store/transmit and index/manage the image data effectively and efficiently to make such collections easily accessible.

The storage and transmission challenge is tackled by image coding/compression, which has been studied for more than 30 years and significant advancements have been made. Many successful, efficient and effective image-coding techniques have been developed and the body of literature on image coding is huge. Well-developed and popular international standards, e.g., [6], on image coding have also long been available and widely used in many applications.

The challenge to image indexing/management is studied in the context of image database, which has also been actively researched by researchers from a wide range of disciplines including those from computer vision, image processing, and traditional database areas for over a decade [14]. One particularly promising approach to image database indexing and retrieval is the query by image content (QBIC) method [1], whereby the visual contents of the images, such as color distribution (color histogram), texture attributes and other image features are extracted from the image using computer vision/image processing techniques and used as indexing keys. In an image database, these visual keys are stored along with the actual imagery data and image retrieval from the database is based on the matching of the models visual keys with those of the query images. Because extra information has to be stored with the images, traditional approach to QBIC is not efficient in terms of data storage. Not only is it inefficient, it is also inflexible in the sense that image matching/retrieval can only based on the pre-computed set of image features.

Many image-coding methods developed over the years are essentially based on the extraction and retention of the most important (visual) information of the image. The retained important information, such as the DCT coefficients of JPEG can be used for image indexing and object recognition [2]. However, since JPEG and other similar methods are not explicitly designed for image indexing purpose, models and features have to be derived from the transform coefficients, which generally involves complicated and complex computation and also leads to an expansion of data. For example, it has been demonstrated that color is an excellent cue for image indexing [3], however, it is difficult to explicitly exploit color information from the transform coefficients without decoding. On the other hand, nontransform based image coding can have image features such as color more easily available. For example, recent work on color image coding using vector quantization has demonstrated that color as well as pattern information can be readily available in the compressed image stream (without performing decoding) to be used as image indices for effective and efficient image retrieval [4].

Block truncation coding (BTC) is a relatively simple image coding technique developed in the early years of digital imaging more than 20 years ago [5]. Although it is a simple technique, BTC has played an important role in the history of digital image coding in the sense that many advanced coding techniques have been developed based on BTC or inspired by the success of BTC. Even though the compression ratios achievable by BTC have long been surpassed by many newer image-coding techniques such as DCT (JPEG) [6] and wavelet [7], the computational simplicity of BTC has made it and BTC-like image coding techniques attractive in applications whereby real time fast implementation is desirable. Further more, with the rapid advancement in processor speed, storage device technology and faster network connection, low bit rate coding is no longer a critical factor in many practical applications. Based on a certain tradeoff, higher bit rate and complexity can be acceptable. On the other hand, with very large image collections becoming more and more common, effectively managing large image database, making images easily accessible have becoming a challenge. Modern imaging systems not only require efficient coding, but also easy manipulation, indexing and retrieval, the so-called "fourth criterion" in image coding [18].

In this paper we shall show another attractive feature of BTC-types image coding methods. We shall demonstrate that BTC

Manuscript received March 9, 2001; revised October 2, 2002. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Christine Guillemot.

The author is with the School of Computer Science, The University of Nottingham, Nottingham NG8 1BB, U.K. (e-mail: qiu@cs.nott.ac.uk).

Digital Object Identifier 10.1109/TIP.2002.807356

can not only be used for image coding, it can also be conveniently and effectively used in content based image indexing and retrieval for image database applications. In Section II, we first briefly review the original BTC idea and its application in color image coding. In Section III, we develop a method to derive image content description features directly from the BTC stream without performing decoding. Two types of features are derived, one characterizes the statistics of the color correlation within the blocks and the other characterizes the statistics of the spatial patterns of the blocks. In Section IV, we present experimental results on content-based image retrieval to demonstrate that the new method is comparable to state of the art techniques. Concluding remarks are given in Section V.

II. BTC for Color Image Coding

Block truncation coding (BTC) was first developed in 1979 for greyscale image coding [5]. This method first divides the image to be coded into small nonoverlapping image blocks (typically of size 4×4 pixels to achieve reasonable quality). The small blocks are coded one at a time. For each block, the original pixels within the block are coded using a binary bit-map the same size as the original block and two mean pixel values.[1] The method first computes the mean pixel value of the whole block and then each pixel in that block is compared to the block mean. If a pixel is greater than or equal to the block mean, the corresponding pixel position of the bitmap will have a value of 1, otherwise it will have a value of 0. Two mean pixel values, one for the pixels greater than or equal to the block mean and the other for the pixels smaller than the block mean are also calculated. At decoding stage, the small blocks are decoded one at a time. For each block, the pixel positions where the corresponding bitmap has a value of 1 is replaced by one mean pixel value and those pixel positions where the corresponding bitmap has a value of 0 is replaced by another mean pixel value.

It was quite nature to extend BTC to multispectrum images such as color images and a number of authors have suggested various methods [8]–[10]. The simplest extension was to view a color image as consisting of three independent greyscale images and apply BTC to each color plane independently. The disadvantage of this method is that three bit planes are needed hence the compression ratios achievable are low. A more aggressive approach is to exploit the redundancy exits between spectrum bands by using only a single bit map for all the spectral bands. In this work, we used single bit plane BTC.

Most color images are recorded in RGB space, which is perhaps the most well-known color space. Various other color spaces that are suited for different image processing tasks also exist. Apart from RGB space, the YCbCr space defined by the International Radio Consultative Committee (CCIR) for HDTV systems [11] and the CIE $L^*a^*b^*$ uniform color space [12] are often used in the literature. We have experimented with all these color spaces and in terms of coding and image retrieval performances in the context of this paper, the difference between various spaces is very small. We therefore only report results based on the RBG space.

As described previously, BTC divides the image to be coded into small blocks and code them one at a time. For single bit map BTC of color image, a single binary bitmap the same size as the block is created and two colors are computed to approximate the pixels within the block [10]. To create a binary bitmap in the RGB space, an inter-band average image (IBAI) is first created and a single scalar value is found as the threshold value. The bitmap is then created by comparing the pixels in the IBAI with the threshold value. Formally, Let $X = \{r(i,j), g(i,j), b(i,j), i = 1,2,\ldots,m, j = 1,2,\ldots,n\}$ be an $m \times n$ color image block in RGB space. Let $I = \{ib(i,j), i = 1,2,\ldots,m, j = 1,2,\ldots,n\}$ be the inter-band average image, then we have

$$ib(i,j) = \frac{1}{3}(r(i,j) + g(i,j) + b(i,j)) \quad \forall i,j \tag{1}$$

and the threshold is computed as the mean of I, i.e.,

$$T = \frac{1}{m \times n} \sum_{i=1}^{m} \sum_{j=1}^{n} ib(i,j). \tag{2}$$

The binary bitmap $\{bm(i,j,), i = 1,2,\ldots,m, j = 1,2,\ldots,n,$ is computed as

$$bm(i,j) = \begin{cases} 1 & \text{if } ib(i,j) \geq T \\ 0 & \text{if } ib(i,j) < T. \end{cases} \tag{3}$$

After the creation of the bitmap, two representative (mean) colors are then computed. The two mean colors, $\text{MC}_1 = (r_{m1}, g_{m1}, b_{m1})$ and $\text{MC}_2 = (r_{m2}, g_{m2}, b_{m2})$, are computed as follows:

$$\begin{aligned}
r_{m1} &= \frac{1}{\sum_{i=1}^{n}\sum_{j=1}^{m} bm(i,j)} \sum_{i=1}^{m}\sum_{j=1}^{n} bm(i,j) r(i,j) \\
g_{m1} &= \frac{1}{\sum_{i=1}^{n}\sum_{j=1}^{m} bm(i,j)} \sum_{i=1}^{m}\sum_{j=1}^{n} bm(i,j) g(i,j) \\
b_{m1} &= \frac{1}{\sum_{i=1}^{n}\sum_{j=1}^{m} bm(i,j)} \sum_{i=1}^{m}\sum_{j=1}^{n} bm(i,j) b(i,j) \qquad (4) \\
r_{m2} &= \frac{1}{m \times n - \sum_{i=1}^{n}\sum_{j=1}^{m} bm(i,j)} \\
&\quad \cdot \sum_{i=1}^{m}\sum_{j=1}^{n} (1 - bm(i,j)) r(i,j) \\
g_{m2} &= \frac{1}{m \times n - \sum_{i=1}^{n}\sum_{j=1}^{m} bm(i,j)} \\
&\quad \cdot \sum_{i=1}^{m}\sum_{j=1}^{n} (1 - bm(i,j)) g(i,j) \\
b_{m2} &= \frac{1}{m \times n - \sum_{i=1}^{n}\sum_{j=1}^{m} bm(i,j)} \\
&\quad \cdot \sum_{i=1}^{m}\sum_{j=1}^{n} (1 - bm(i,j)) b(i,j). \qquad (5)
\end{aligned}$$

[1] In the original implementation, the block mean and the variance of the pixels are used to preserve the first and second moments of the block. The descriptions here follow a later version of BTC, which was shown to give better performance [8] and suitable for our application.

For achievable compression bit rates, readers are referred to, e.g., [9], [10] for various techniques applicable to post BTC coding for reducing the bit rates. The images in our application are visual images, i.e., the final judgement of the image quality lies on the human observers. Through psychovisual experiment, we found that the visual quality of the BTC coded images were not greatly affected by color quantizing the mean colors [19]. Therefore, for each BTC coded image, a color table (palette) [20] consists of 256 colors are constructed and stored. The mean colors are coded (8 bits per mean color) as the indices of the colors in the color table that are the closest to the mean colors. Therefore the bit rate in our scheme is 2 bits/pixel or 12 : 1 compression, plus 768 bytes (256×3 bytes) color table overhead in each image. Our content descriptors are derived from such a BTC coded scheme.

III. Deriving Image Content Description Features From BTC Code

It is clear from above descriptions of single bit map BTC for color image coding that the BTC code for each small block consists of two color indices and a binary bitmap. In this section, we describe methods to derive image content description features from the mean color indices and the bitmap pattern for image retrieval in database applications.

Content based image indexing and retrieval for image database is an important topic of research, which has attracted extensive interests in recent years [1], [14]. Early approaches used color alone for indexing [3]. This approach has been very successful and is extensively used in many of today's research and commercial systems. However, it is well known that the biggest drawback of color histogram based method is that histogram is a global measure, it does not contain any spatial information. Recently, researchers have developed a better and more discriminative technique known as the color correlogram (CC) technique [15], which has been shown to provide significant improvement over the original color histogram approach. Color correlogram is very similar to co-occurrence matrix [16] developed some 20 years ago for greyscale texture classification. Formally, the CC of image $\{Z(x,y), x = 1, 2 \ldots M, y = 1, 2, \ldots, N\}$ is defined as

$$CC(i,j,k) = \Pr\big(Z(x_1,y_1) \in C_i | Z(x_2,y_2) \in C_j; \; k = \max\{|x_1 - x_2|, |y_1 - y_2|\}\big) \quad (6)$$

where the original image $Z(x,y)$ is quantized to J colors $C_1, C_2, \ldots, C_J$ and the distance between the two pixel $k \in [\min\{M,N\}]$ is fixed *a priori*. In words, the CC of an image is the probability of joint occurrence of two pixels some k distance apart that one pixel belongs to color C_i and the other belongs to color C_j. The size of CC is $O(L^2K)$. To reduce storage requirement, [15] concentrated on auto-correlogram whereby $i = j$ and its size is of $O(LK)$.

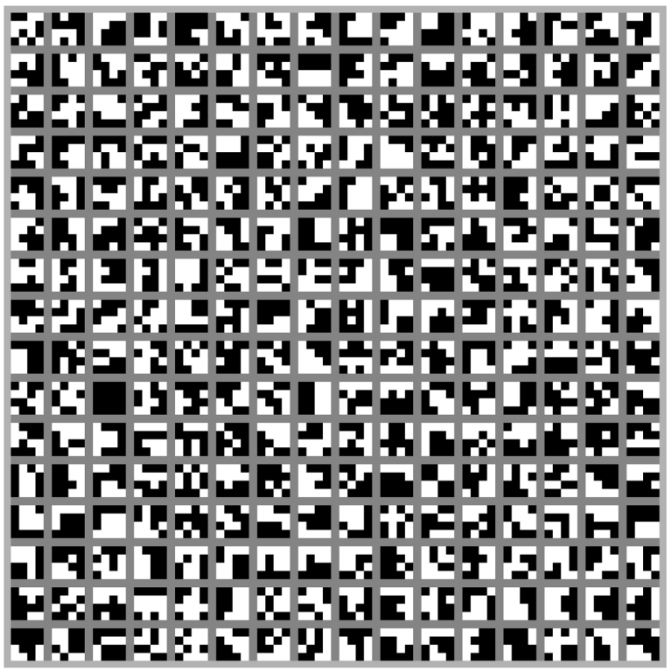


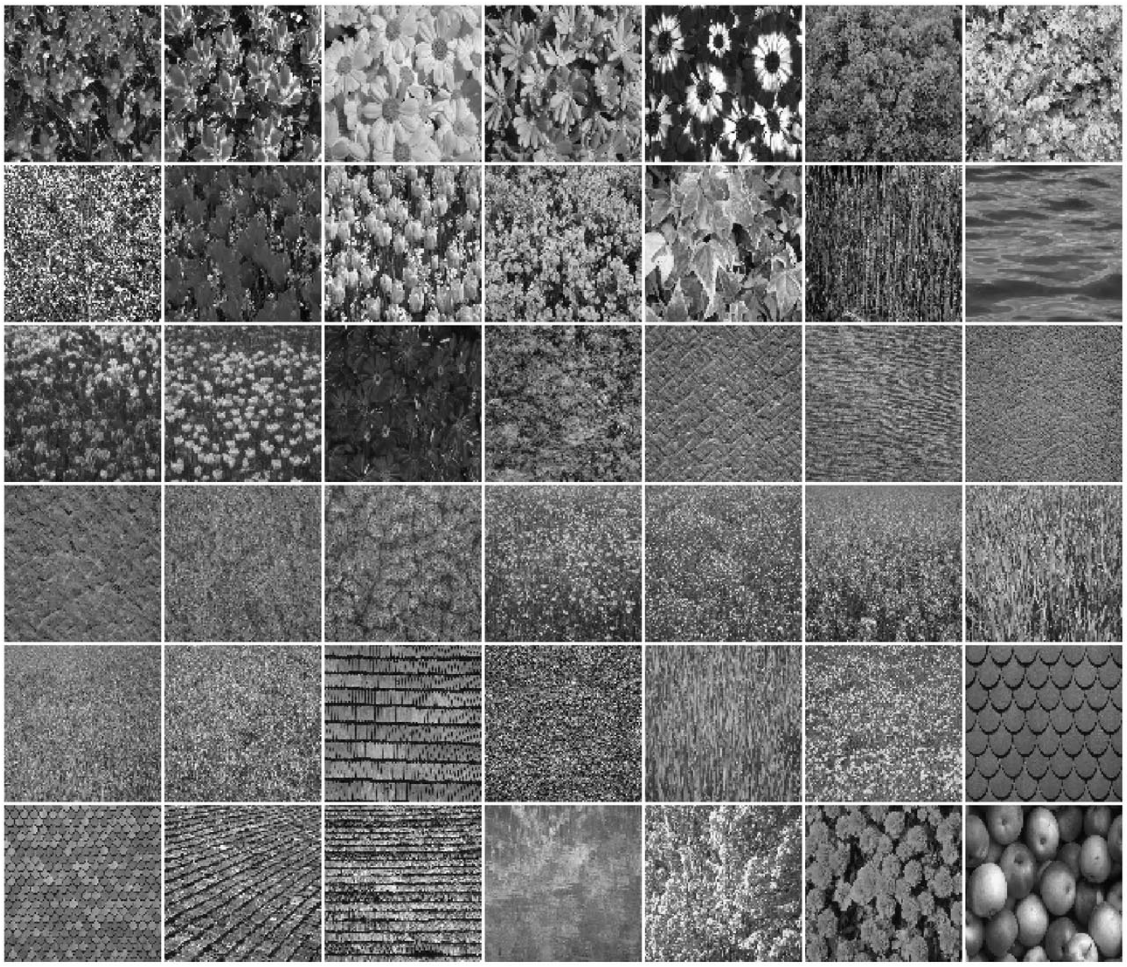
Fig. 1. Block bitmap codebook patterns. These 256 4×4 bitmap patterns were derived using binary vector quantization trained with over 3.5 million samples.

A. BTC Color Co-Occurrence Matrix (BCCM)

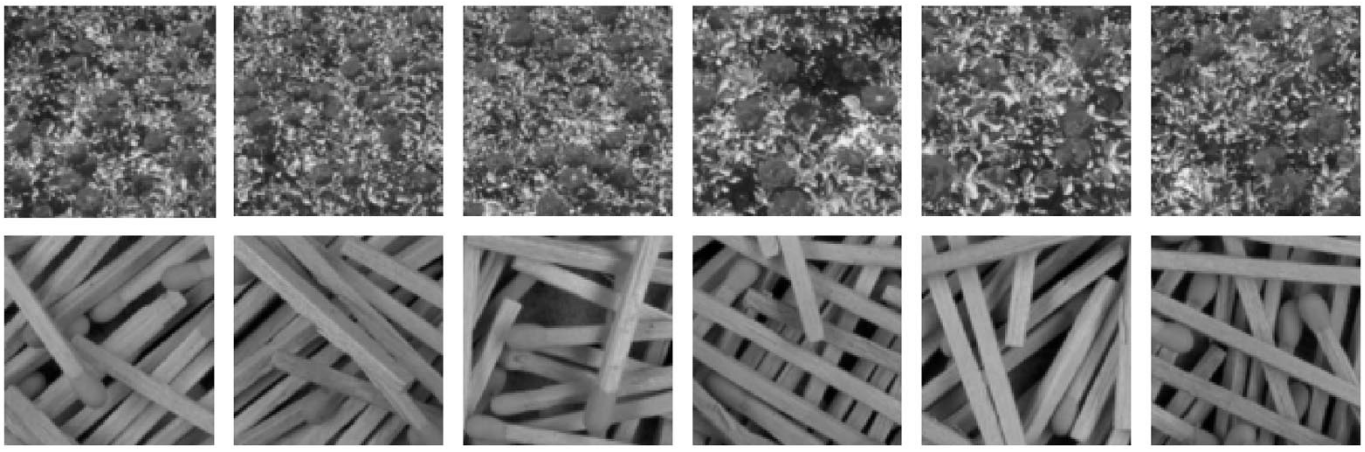
Similar to CC described above, we exploit the code of BTC described in the last section by introducing a similar statistical measure, which we term BTC or block color co-occurrence matrix (BCCM). As in the case of CC, a global color table consists of J colors is first designed. It is important to understand that there is only ONE global color table (its design will be described in the next section) which is used by all images, the query as well as all images in the database. This global color table is different from the color tables used by each individual image in the post BTC coding described in the last section. To make the distinction, we shall call the individual image's color table local color table. The BCCM of an image is constructed as follows.

Let $C_g = \{C_g(0), C_g(1), \ldots, C_g(J-1)\}$ be the J-color global color table, $C_f = \{C_f(0), C_f(1), \ldots, C_f(255)\}$ be the 256-color local color table of a BTC coded image. For a BTC coded image block, B, its bit stream consists of a bitmap and two 8-bit integer numbers, i.e., $B = \{bm(x,y), I_1, I_2\}$. These two integers, $I_1, I_2 \in [0, 255]$, are the color indices of the two mean colors pointing to two colors in the local color table. We construct a *hash* table indexing the global color indices with the local color indices. Let *IHash* be a color indices hash table which has the local color indices, I_f, as its key and the global color indices, I_g, as its value. The hash table can be constructed as (see (7) at the bottom of the page), i.e, each local color index is

$$\begin{cases} IHash = (I_f \Rightarrow I_g) \text{ and } I_g = \arg\left(\min_{\forall I_g \in [0,J-1]} \left(\|C_f(I_f) - C_g(I_g)\|\right)\right) \\ IHash\{I_f\} = I_g \end{cases} \quad \text{for all } I_f = 0, 1, 2, \ldots, 255 \quad (7)$$



(a)



(b)

Fig. 2. (a) Subset of the 120 texture classes used in the first experiment. Notice that these textures differ significantly from those used to train the global color table and the codebook of block bitmap patterns. (b) Examples of subimages belong to the same class.

mapped to a global color index based on a minimum distance criterion (it is possible that two or more local color indices may be mapped to the same global index). This hash table can either be constructed on-line or stored with the encoded image data, which will incur a slight overhead (we opt for storing this hash table in our experiments).

The BCCM of an image is defined as the probability of the co-occurrence of two mean colors within the BTC blocks. Formally

$$BCCM(i,j) = \Pr\left(IHash\left\{I_1\right\} = i | IHash\left\{I_2\right\} = j\right), \quad \forall i,j \in [0, J-1] \quad (8)$$

where I_1 and I_2 are the pair of local color indices of the BTC mean colors within the same block. Equivalently, $BCCM(i,j)$ gives the probability that a BTC coded block having one mean color MC_1 (local color quantized color) being mapped to the i-th color in the global color table and other mean color MC_2 (local color quantized color) being mapped to the j-th color in the global color table.

Obviously, since the mean colors of the block are coded into local color indices, the operation only involves hash table retrieval and is therefore computationally very efficient. If an image is of the size $M \times N$ and the block size used by the BTC is $m \times n$, then the number of blocks (pairs of colors) used to compute BCCM is $(M \times N)/(m \times n)$ and can be constructed "online" as and when it is required. The size of BCCM is $O(J^2)$. As have been noted in [15], BCCM is also very sparse and can be stored very efficiently. However, since the computation is so trivial, BCCM is computed "on the fly" and there is no need to store it permanently in an image database. It is also worth noting that there is no need to decode the BTC code in the computation of BCCM.

B. Block Pattern Histogram

We further introduce another measure, termed block pattern histogram (BPH) to characterize the image content. First, we create a codebook of binary patterns for the bitmap of BTC code using binary vector quantization. Let $Q = \{Q_1, Q_2, \ldots Q_N\}$ be the bitmap codebook consists of N codewords and Q_i is an $m \times n$ binary pattern.[2] Let $B = \{b(i,j), i = 1,2,\ldots,m, j = 1,2,\ldots,n\}$ be the bitmap of the BTC code, the pattern index of the bitmap, P_I, is the index of the bitmap codeword that is most similar to B, formally defined as

$$P_I = \arg\left(\min_{\forall j}\{DH(B, Q_j)\}\right) \tag{9}$$

where $DH(B, Q_j)$ denotes the Hamming distance between the two binary patterns (vectors). The BPH is defined as

$$BPH(i) = \Pr(P_I = i),\ i = 1,2,\ldots,N. \tag{10}$$

That is, $BPH(i)$ is the probability that a BTC bitmap is mapped to codeword Q_i in the bitmap codebook.

For the binary vector quantizer, we have developed a robust binary vector quantization scheme based on a neural network learning technique, the frequency sensitive competitive leaning (FSCL) [21], to design the bitmap codebook. In the author's experience, FSCL is a robust VQ codebook design technique. For a detailed description of the FSCL algorithm, readers are referred to [21].

IV. Experimental Results

We present experimental results of using the image content descriptors derived from the BTC coding streams for content-based image retrieval. The images in the database are coded by BTC and from the coded stream, we derive the images' BCCM and BPH. In order to construct these two image features, we first need to design a global color quantization table and a global bitmap codebook. To do so, we have used two hundred and thirty five 512×512 pixel, 24 bits/pixel true color texture images from MIT Media Lab's VisTex collection.[3] We designed a 64-color global color table using all the pixels of these images based on the FSCL [21] algorithm. To design the bitmap codebook, we created the training bitmap patterns using all these images based on (3) and a 256-pattern codebook was created. Fig. 1 shows the 4×4 bitmap patterns of the 256 codebook. In constructing the image database, we have also created a color index hash table (8) for each image.

TABLE I
Number of Queries and Their Retrieval Rates for the BTC and Color Correlogram Methods Performed on a Texture Image Database Consist of 720 Images and 120 Different Texture Classes. Each Image Was Used in Turn as the Query Image and Therefore 720 Queries Were Performed. This Is Interpreted as Follows: For CC, 482 Queries Achieved a Retrieval Rate of 100% and 130 Queries Achieved a Retrieval Rate of 80%, Etc.

	Retrieval Rates				
Methods	**5/5**	**4/5**	**3/5**	**2/5**	**1/5**
CC	**482**	**130**	**68**	**33**	**7**
BCCM only	**512**	**115**	**60**	**25**	**8**
BCCM + BPH	**566**	**86**	**46**	**17**	**4**

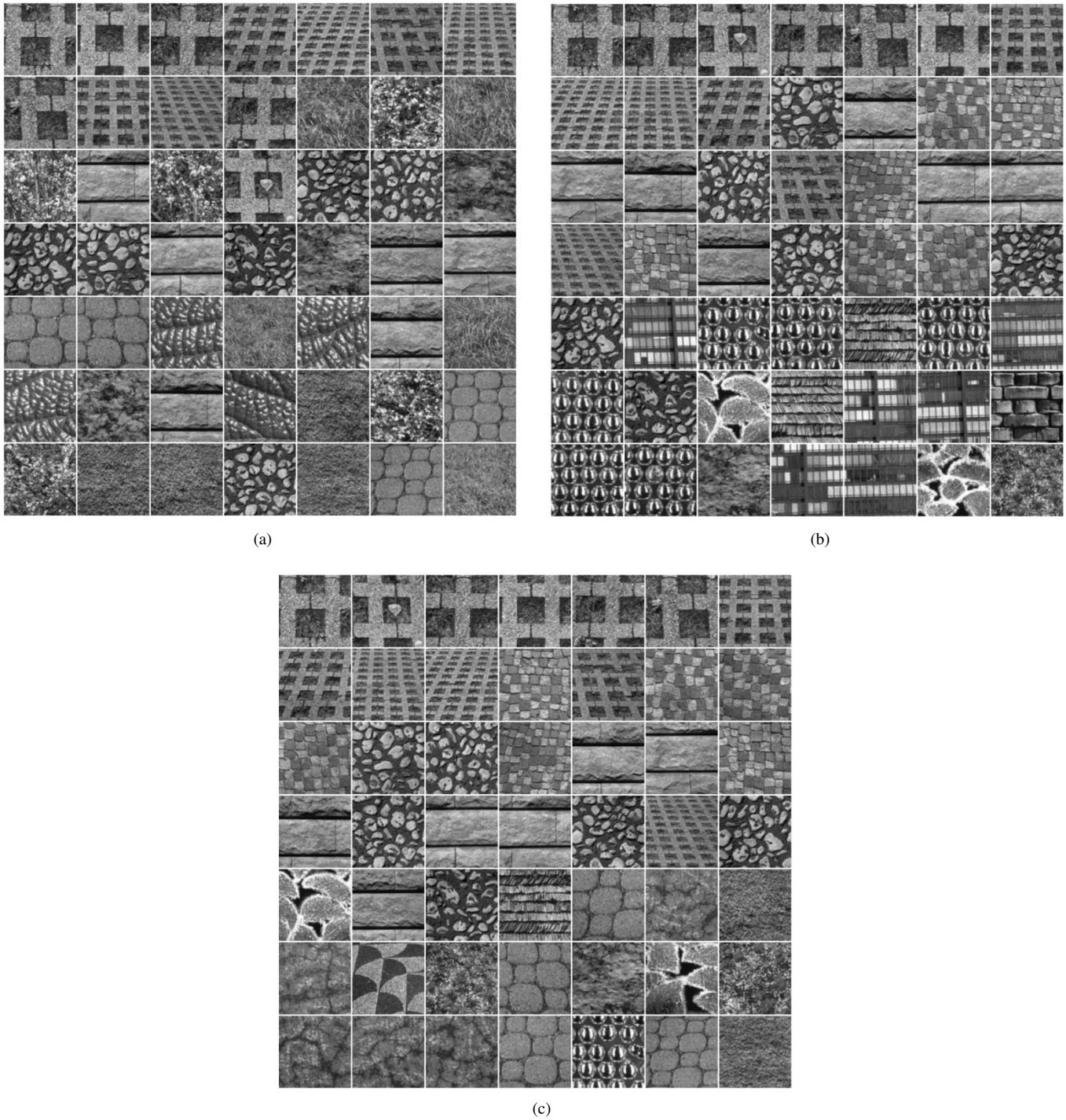
To retrieve images from the database coded by BTC based on querying by image example, the query image is first coded with BTC and its BCCM and BPH are derived and then compared with those of the images in the database. Those images that are most similar to the query image are returned to the user. To compare the similarity of two images based on BCCM and BPH, we can use a relative distance measure [15]. Let $BCCM_p$, BPH_p and $BCCM_q$, BPH_q be the BCCM and BPH of images p and q respectively. The similarity of the images is measured as the distances between the BCCM's and BPH's $d(p,q)$ and is calculated as follows:

$$d(p,q) = \lambda_1 \sum_{\forall C_i, C_j} \frac{|BCCM_p(i,j) - BCCM_q(i,j)|}{1 + BCCM_p(i,j) + BCCM_q(i,j)} + \lambda_2 \sum_{\forall k} \frac{|BPH_p(k) - BPH_q(k)|}{1 + BPH_p(k) + BPH_q(k)} \tag{11}$$

where λ_1 and λ_2 are weighting constants. Note that the two descriptors, BCCM and BPH belong to different modalities. Although it is popular to combine features of different modalities in image retrieval [22] and in computer vision in general [23], there is no systematic method for combining features of different modalities and this remains to be a challenging open problem in computer vision. The general practice (and probably the only way) to determine the relating weightings of different modalities is through experiment. However, weightings determined in this way are data dependent. It is therefore not possible to give a systematic guideline for choosing the values for λ_1 and λ_2. What we found that is conclusive, however, is that in any case, including both descriptors gave better performance than using only one of either descriptor. As a general guide, both descriptors should be used together and depending on the application, the optimal weighting values have to be determined

[2]Notice that there is only ONE (global) binary bitmap quantizer which is not used in coding.

[3]http://vismod.www.media.mit.edu/vismod/imagery/VisionTexture/vistex.htm



(a) (b) (c)

Fig. 3. Examples of top 48 retrieved images by different methods from a 720-texture image database. The first image on the top left-hand corner is the query image and subsequent images from left to right and top to bottom are retrieved image ordered according to their similarity to the query image. (a) Color correlogram, (b) BTC with BCCM only, and (c) BTC with BCCM and BPH.

experimentally. We found (empirically) that by setting $\lambda_1 = \lambda_2$ worked quite well.

A. Texture Image Database

We first used a texture database consists of 120 difference texture classes, a subset of which is shown in Fig. 2. Notice that this set of texture images is not those used to train the color table. The database was formed by dividing 120 single textured images (each image was judged by human observers as consisting of roughly a single type of color texture) of size 300×200 pixels into six 100×100 nonoverlapping subimages, thus creating a database of 720 texture images. A similar testing strategy was used in [17]. All 720 images were coded using BTC. In the experiment, each of the 720 images in the database was used in turn as the query image. For each query image q, the distances between q and images in the database, p_i, $d(q, p_i)$ where $i = 1, 2, \ldots, 720$, $p_i \neq q$ were calculated and sorted in increasing order and the closest set of images were then retrieved.

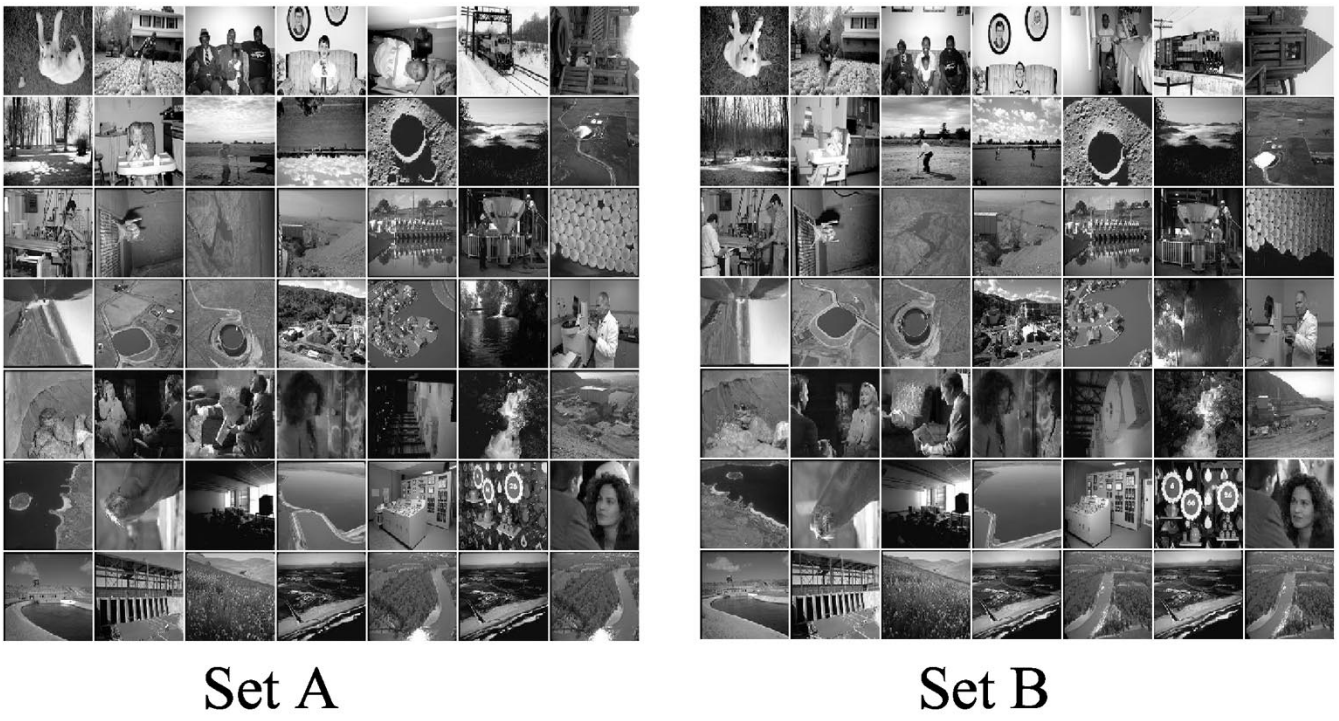


Fig. 4. Subset of the 96 querying image pairs. These pairs are divided into set A and B. For each image in set A there is a corresponding similar target image in set B.

In the ideal case the top 5 retrievals should come from the same large image as the query image. The performance was measured in terms of the retrieval rate (RR) defined as follow (this measure is also know as precision): Let q be the query image from texture class T_k, i.e., $q \in T_k$. and p_j, $j = 1, 2, 3, 4, 5$, be the first five retrievals, the retrieval rate for query image q is

$$RR(q) = \frac{1}{5}\sum_{j=1}^{5} S(p_j) \text{ where } S(p_j) = 1, \text{ if } p_j \in T_k \text{ and } S(p_j) = 0, \text{ if } p_j \notin T_k. \quad (12)$$

As a comparison, the color auto-correlogram of [15] was also implemented using $D = \{1, 3, 5, 7\}$ exactly as those described in the original paper. Both methods used the same color table consisting of 64 colors.

Table I shows the number of queries and their corresponding retrieval rates for different methods. For the color correlogram method of [15], 482 queries (out of 720) achieved an RR of 100%, 130 queries achieved an RR of 80%, 68 queries achieved an RR of 60%, 33 queries achieved an RR of 40% and seven queries only achieved an RR of 20%. For BCCM only ($\lambda_1 = 1$ and $\lambda_2 = 0$), 512 queries (out of 720) achieved an RR of 100%, 115 queries achieved an RR of 80%, 60 queries achieved an RR of 60%, 25 queries achieved an RR of 60% and eight queries only achieved an RR of 20%. Combining BCCM and BPH together ($\lambda_1 = 1$ and $\lambda_2 = 1$), 566 queries (out of 720) achieved an RR of 100%, 86 queries achieved an RR of 80%, 46 queries achieved an RR of 60%, 17 queries achieved an RR of 60%, and four queries only achieved an RR of 20%. It is seen that the new method provides a better performance. Fig. 3 shows an example of the retrieved images by various methods.

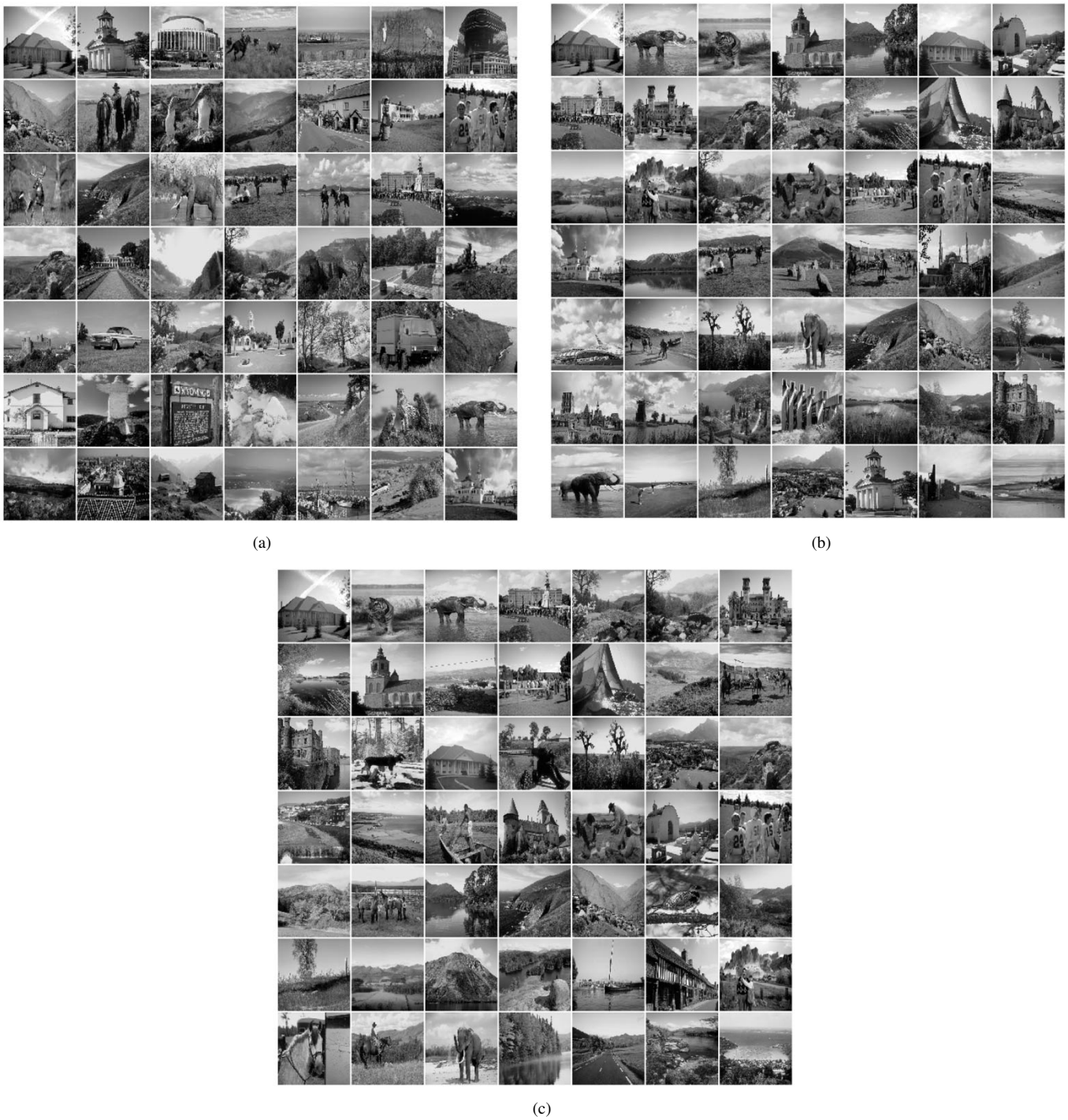
TABLE II
IMAGE RETRIEVAL PERFORMANCES OF COLOR CORRELOGRAM AND BTC ON A PHOTOGRAPHIC IMAGE DATABASE CONSIST OF 20 000 IMAGES. 96 QUERIES WERE PERFORMED. THE NUMBER OF QUERIES AND THEIR TARGET IMAGES' RANK RANGE IN THE RETURNED LIST. THEY ARE INTERPRETED AS FOLLOWS: FOR THE CC METHOD, 72 QUERIES FOUND THEIR CORRESPONDING TARGET IMAGE IN THE FIRST RANK, 81 QUERIES FOUND THEIR CORRESPONDING TARGET IMAGES WITHIN THE FIRST 10 RETURNED IMAGE, ETC.

	Ranks						
Methods	**1**	**<=10**	**<=30**	**<=100**	**>100**	**Lowest**	**Mean**
CC	**72**	**81**	**85**	**87**	**9**	**7299**	**300**
BCCM Only	**66**	**80**	**89**	**91**	**5**	**6864**	**96**
BCCM + BPH	**67**	**81**	**89**	**91**	**5**	**5126**	**77**

From these results, it is seen that for the color correlogram method, 67% (482/720) of the queries achieved 100% precision, for the BCCM only method, 71% (512/720) of the queries achieved 100% precision and for the combined BCCM and BPH method, 77% (566/720) of the queries achieved 100% precision. The combined method therefore achieved quite significant improvement over the color correlogram method in this particular test.

B. Color Photo Image Database

In this second experiment, we tested the method on a database consists of 20 000 color images from the commercially available Corel Photo Collections. We hand picked 96 pairs of similar images which were embedded into the database. Fig. 4 shows a subset of these images, which was divided into set A and set B. When using an image from set A as query, the goal was to



(a) (b)

(c)

Fig. 5. Examples of top 48 retrieved images by different methods from a photographic image database consists of 20 000 images. The first image on the top left-hand corner is the query image and subsequent images from left to right and top to bottom are retrieved image ordered according to their similarity to the query image. (a) Color correlogram, (b) BTC with BCCM only, and (c) BTC with BCCM and BPH.

retrieve the target image in set B, or vice versa. As a comparison, we have also implemented the color correlogram method. For each querying operation, all images in the database are ranked according to their similarity to the querying image. In the ideal case, the target image should be in the first rank. Table II shows the result of using set A as query and set B as target.

In practice, if viewed from a normal distance (about an arm's length), an ordinary monitor can display about 30 images on the screen at any one time to enable a viewer clearly make out the contents of each image. It is therefore reasonable to consider that a method which does not find the target image within the first 30 returned images as failure. With such a definition, from Table II, we can see that the success rate of color correlogram was 85/96, i.e., 88.5%, whilst the success rate of the new method was 89/96, i.e., 92.7%.

Another objective measure of retrieval performance is the average percentile [3]. Let $R_t(i)$ be the position of the target image for query image $Q(i)$, $i = 1, 2, \ldots, N_2$ ($N_2 = 96$ in our case), in the ordered list of N_1 returned images ($N_1 = 20\,000$ in our case). The values of $R_t(i)$ range from 1 for a perfect match to

N_1 for the worst possible match. The average ranking percentile is defined as

$$R_{pc} = \frac{1}{N_2}\sum_{i=1}^{N_2}\left(\frac{N_1 - R_t(i)}{N_1 - 1}\right)100\%. \quad (13)$$

For the color correlogram method, $R_{pc} = 98.5\%$, for the BCCM only method, $R_{pc} = 99.5\%$ and for the combined BCCM and BPH method $R_{pc} = 99.6\%$.

Fig. 5 shows an example of retrieved images. It is seen that the new method performed at least as well as the color correlogram method.

It is also interesting to note that the effect of combining BPH with BCCM was more significant for the color texture database than for the ordinary color photo database. From Fig. 1, it is not difficult to see that these binary patterns actually reflect the texture characteristics of the image in some way. Therefore, it is easy to understand the results.

V. Concluding Remarks

In this paper, we have presented a method for using a well known image coding technique to achieve both image coding and content based image retrieval in the compressed domain. Two image content description features derived directly from the compressed stream have been developed. To demonstrate the effectiveness of using the BTC code for image retrieval, we presented experimental results using a texture database and a large photographic image database. Results showed that the new method performed at least as well as, sometime even better than a state of the art technique. One significant advantage of the current method is that it achieves coding and retrieval simultaneously. As with other techniques that use multimodality features, an open question is how to set the relative weightings of the BCCM and BPH descriptors. In this paper, we followed the common practice and determined the weightings empirically.

As a final remark, image coding has been studied for a long time and has also been very successful. All image coding methods essentially try to extract the most important visual information and represent them in a compact manner. Indexing and retrieval for image database applications have becoming increasingly important. We believe results and experiences of more than 30 years image coding/compression research can play a fruitful role in the development of effective and efficient methods for image indexing and retrieval in large image database. We hope the present paper has shown a glimpse of such potential.

Acknowledgment

The author wishes to thank the associate editor and the referees sincerely, whose comments have helped improving the presentation of the paper.

References

[1] W. Niblack *et al.*, "Querying images by content using color, texture and shape," *Proc. SPIE*, vol. 1908, pp. 173–187, 1993.

[2] W. B. Seales *et al.*, "Object recognition in compressed imagery," *Image Vis. Comput.*, vol. 16, pp. 337–353, 1998.

[3] M. Swain and D. Ballard, "Color indexing," *Int. J. Comput. Vis.*, vol. 7, pp. 11–32, 1991.

[4] G. Qiu, "Image indexing using a colored pattern appearance model," in *Proc. SPIE Conf. Storage and Retrieval for Media Databases 2001*, San Jose, CA, Jan. 2001.

[5] E. J. Delp and O. R. Mitchell, "Image coding using block truncation coding," *IEEE Trans. Commun.*, vol. 27, pp. 1335–1342, Sept. 1979.

[6] W. B. Pennebaker and J. L. Mitchell, *JPEG Still Image Compression Standard* Van Nostrand Reinhold, New York, 1993.

[7] J. M. Shappiro, "Embedded image coding using zero trees of wavelet coefficients," *IEEE Trans. Signal Processing*, vol. 41, pp. 3445–3462, Dec. 1993.

[8] M. D. Lema and O. R. Mitchell, "Absolute moment block truncation coding and its application to color images," *IEEE Trans. Commun.*, vol. COM-32, pp. 1148–1157, Oct. 1984.

[9] T. Kurita and N. Otsu, "A method of block truncation coding for color image compression," *IEEE Trans. Commun.*, vol. 41, pp. 1270–1274, Sept. 1993.

[10] Y. Wu and D. C. Coll, "Single bit-map block truncation coding of color images," *IEEE J. Select. Areas Commun.*, vol. 10, pp. 952–959, June 1992.

[11] "Encoding Parameters of Digital Television for Studios," Int. adio Consult. Committee, Geneva, CCIR Recommend. 601-2, 1990.

[12] G. Sharma and H. J. Trussell, "Digital color imaging—Tutorial," *IEEE Trans. Image Processing*, vol. 6, pp. 901–932, July 1997.

[13] C. Bishop, *Neural Networks for Pattern Recognition*. Oxford, U.K.: Oxford Univ. Press, 1995.

[14] Y. Rui, T. S. Huang, and S. F. Chang, "Image retrieval: Current techniques, promising directions and open issues," *J. Vis. Commun. Image Represent.*, vol. 10, pp. 39–62, 1999.

[15] J. Huang *et al.*, "Image indexing using color correlogram," in *Proc. CVPR97*, 1997, pp. 762–768.

[16] R. M. Haralick, "Statistical and structural approaches to texture," *Proc. IEEE*, vol. 67, pp. 786–804, 1979.

[17] B. S. Manjunath and W. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 837–842, Aug. 1996.

[18] R. W. Picard, "Content Access for Image/Video Coding: 'The Fourth Criterion'," MIT Media Lab, Cambridge, MA, TR 295, 1994.

[19] G. Qiu, "Coding color quantized image by local color quantization," in *Proc. 6th Color Imaging Conf.*, Scottsdale, AZ, Nov. 1998.

[20] M. Gervautz and W. Purgathofer, "A simple method for color quantization: Octree quantization," in *Graphics Gems*. New York: Academic, 1990.

[21] S. C. Ahalt *et al.*, "Competitive learning algorithms for vector quantization," *Neural Networks*, vol. 3, pp. 277–290, 1990.

[22] N. Howe and D. Huttenlocher, "Integrating color, texture and geometry for image retrieval," in *Proc. CVPR 2000*, pp. 239–246.

[23] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, pp. 1254–1259, Nov. 1998.

Guoping Qiu received the B.Sc. degree in electronic measurement and instrumentation from the University of Electronic Science and Technology of China, Chendu, in July 1984 and the Ph.D. degree in electrical and electronic engineering from the University of Central Lancashire, Preston, U.K., in 1993.

He is currently a Lecturer of computer science at the School of Computer Science and IT, University of Nottingham, Nottingham, U.K. Before joining Nottingham in October 2000, he was a Lecturer at the School of Computing, the University of Leeds, U.K and the School of Computing and Mathematics, the University of Derby, U.K. His general research interests are image and signal analysis and processing algorithms for visual computing, including image database, content-based image and video retrieval; filtering theory and practice for image enhancement, super-resolution and high dynamic imaging, and visual object recognition.