

DCT-Domain Image Retrieval Via Block-Edge-Patterns

K.J. Qiu¹, J. Jiang^{1,2}, G. Xiao¹, and S.Y. Irianto²

¹ Faculty of Informatics & Computing, Southwest China University, Chongqin, China
j.jiang1@bradford.ac.uk, g.xiao@swcu.edu.cn

² Department of EIMC, University of Bradford, UK

Abstract. A new algorithm for compressed image retrieval is proposed in this paper based on DCT block edge patterns. This algorithm directly extract three edge patterns from compressed image data to construct an edge pattern histogram as an indexing key to retrieve images based on their content features. Three feature-based indexing keys are described, which include: (i) the first two features are represented by 3-D and 4-D histograms respectively; and (ii) the third feature is constructed by following the spirit of run-length coding, which is performed on consecutive horizontal and vertical edges. To test and evaluate the proposed algorithms, we carried out two-stage experiments. The results show that our proposed methods are robust to color changes and varied noise. In comparison with existing representative techniques, the proposed algorithms achieves superior performances in terms of retrieval precision and processing speed.

1 Introduction

At present, many images and videos distributed on the Internet or stored inside a database are represented in compressed formats, due to the limitation of space and bandwidth. Joint Picture Expert Group (JPEG) is one of the most typical compressed standards widely used among industrial communities. As a result, methods used for indexing or retrieving images directly in compressed domain have become a recent focus of research and developments. To detect or extract the features from those compressed images, traditional approaches need to decode the images first to provide pixel domain for content feature extraction [1]. To avoid such an overhead operation and computing cost, many research efforts are directed to feature extraction in compressed domain [2-9], including texture, color, and shape etc. Representative work in such areas can be summarized as follows.

Shneier and Abdel.Mottaleb [2] calculated average value of DCT coefficients computed over a window to generate keys of JPEG images for retrieval. S. F. Chang [3] computed the statistical measures of DCT coefficients to form texture feature. Chong-Wah Ngo, Ting-chuen Pong [4] described an image-indexing algorithm via reorganization of DCT coefficients in Mandala domain [5], and representation of color, shape and texture features in compressed domain. G.C. Feng and J. Jiang [6] extracted mean and standard deviation of statistical features directly from DCT coefficients to retrieve

JPEG compressed images. Sim, Kim [7] reported fast texture description and retrieval of DCT-based compressed images. Lay, Ling [8] proposed a method using energy histograms of the low frequency DCT coefficients for retrieving images. Jianghong Liu, Haiming Gu [9] developed a method, which can retrieve images from different kinds of domains by using the wavelet coefficients. Daidi Zhong, Irek Defee [10] described a method based on histograms of quantized DCT blocks. These histograms can be optimized in order to achieve best retrieval performance by optimizing the selection of quantization factor and the number of DCT blocks under normalization of luminance.

Edge is a strong feature for characterizing an image, which can be used to construct an important clue to understand the content of images. While many methods as highlighted above extract features in DCT domain, most techniques reported in the literature extract edge features in pixel domain, including those widely reported for image segmentations, and thus existing efforts on edge-based image retrieval are limited in pixel domain. For example, Jun Weihan, Lei Guo [11] proposed an image retrieval method based on salient edges, in which Canny operator is applied to detect edge points followed by a Water-Filling technique to extract edge curves and shape features for salient edge selection. Minakshi Banerjee and Malay K. Kundu [12] presented a technique for extracting edge map of an image and formed a global feature (like fuzzy compactness) to process image content.

Recent trend on image processing is characterized by detecting edges and classifying their patterns at a block level either in pixel domain or compressed domain [18]. Kim and Lee [13] found out that the edge location is well captured by the polarities of the projection of DCT coefficients. Soltane et al. [14] suggested an adaptive edge operator selection scheme for image segmentation based on the mean, variance, and entropy of DCT coefficients. Shen and Sethi [15] further proposed a method to determine edge strength and orientation through pattern analysis of DCT coefficients. They examined the DCT coefficient patterns induced from an ideal edge model and show how the relative values or signs of different coefficients can be determined at block level. Lee et al. [16] has extracted and defined their scheme to detect scene change using direct feature extraction from MPEG compressed videos. Li et al [17] proposed an effective approach to edge classification from DCT domain. Chang et al [18] proposed a fast and systematic scheme to classify the edge orientation of each block in DCT domain. Five directional edge patterns (no edge, 0-directional edge, $\pi/4$ -directional edge, and $3\pi/4$ -directional edge) were proposed in their paper. Their algorithm is similar to the one proposed in [19].

Based on the analysis of the existing work as described above, we propose a new compressed image retrieval algorithm, in which three block-edge-patterns are extracted directly from DCT domain, which include no edge, horizontal edge, and vertical edge. We then combine the three block edge patterns to form three block edge pattern histograms (3_D histograms, 4_D histograms, Run-length histograms) to characterize the compressed image content and use them as indexing keys for image retrieval. The rest of the paper is organized into four further sections, where Section II describes how the three block edge patterns can be extracted from DCT coefficients based on a brief review of an existing work, Section III describes our proposed

algorithm to form indexing keys, including 3_D histogram, 4_D histogram, and a so-called run-length histogram. Finally, experimental results and their analysis are given in Section IV and conclusions are given in Section V.

2 Block Edge Pattern Extraction in DCT Domain

In reference [18], Chang et al proposed a technique to extract five edge patterns directly from DCT coefficients to characterize the distribution of edges inside an image. In practice, however, our observation and analysis reveal that image edge distribution can actually be characterized with three edge patterns, i.e. no edge, vertical edge and horizontal edge as shown in Figure-1. This is because that an arbitrary edge inside an image can always be approximated by many small vertical or horizontal edge elements. This is illustrated in Figure-2. As a matter of fact, if we zoom into any part of image to a certain extent, the arbitrary shaped image edge can be seen by artifacts consist of vertical and horizontal edges. As a result, such five edge patterns can be further reduced to three edge patterns, and thus further improvement can be achieved in terms of both processing speed and computing cost when the characterization of edge distribution inside an image is implemented by only three edge patterns.

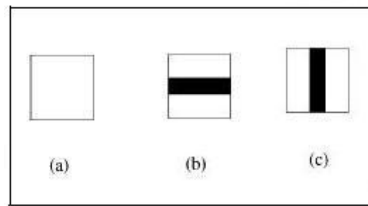


Fig. 1. Illustration of three edge patterns, where (a)=NE, (b)=HE, and (c)=VE

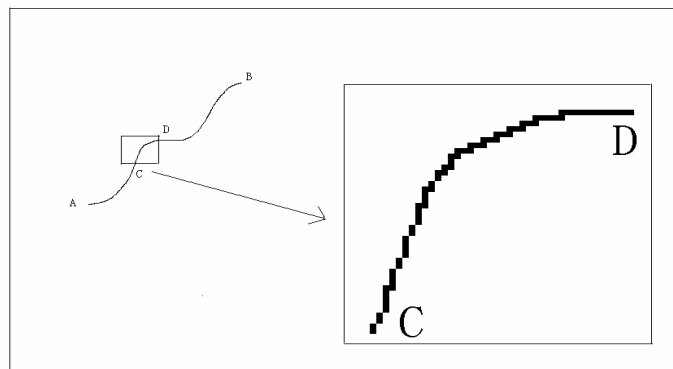


Fig. 2. Zoom-in of an arbitrary shaped edge

To support our analysis, we further compare the five edge pattern representation of two standard images, Lena and pepper, with their three edge pattern representations as shown in Figure-3 and Figure-4. From the comparisons, it can be seen that the two representations present little difference in terms of their edge distribution and content characterization. Therefore, the three edge patterns given in Figure-1 can be used to construct image content features, leading to effective image retrieval based on its content.

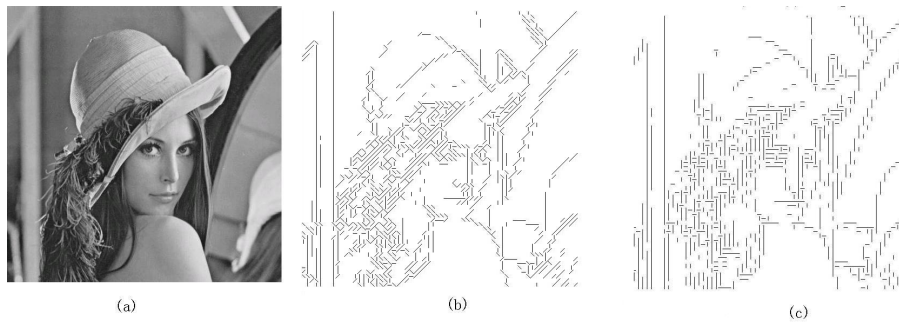


Fig. 3. Edge-pattern representation of image Lena, (b)=five pattern representation, and (c) = three pattern representation

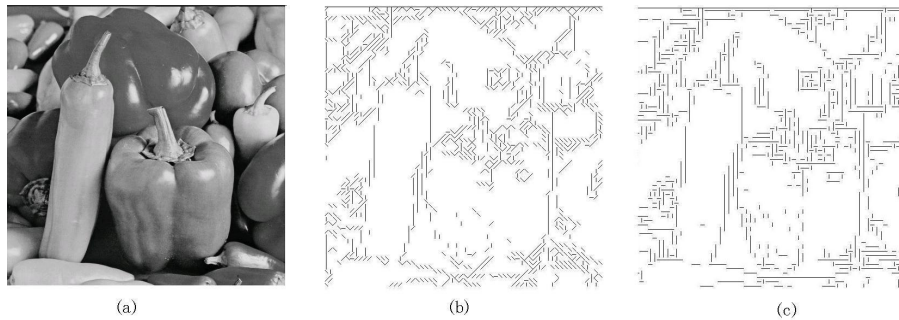


Fig. 4. Edge-pattern representation of image Pepper, (b)=five pattern representation, and (c) = three pattern representation

In 2005, Chang et al proposed a technique of extracting edge patterns from DCT coefficients in terms of 8×8 block of pixels [18]. This technique operates in compressed domain by just analyzing the value of the first four DCT coefficients and thus only limited decoding operations are required without involving the full decompression.

Given a block of 8×8 pixels, the technique divides the block into four regions, and the average pixel value for each region is represented as S_{ij} , $i, j \in [0, 3]$. In pixel domain, the edge pattern classification for the block can be conducted in terms of the four average pixel values by choosing the maximum measure out of all the measure values given in Table-1.

Table 1. Measure for Five Edge Patterns

EdgeDirection (in radian)	Measure values
No edge (NE)	δ_{NE} (set by user)
0	$\delta_0 = \left \frac{S_{00} + S_{01}}{2} - \frac{S_{10} + S_{11}}{2} \right $
$\pi/4$	$\delta_{\pi/4} = \max \left\{ \left S_{00} - \frac{S_{01} + S_{10} + S_{11}}{3} \right , \left S_{11} - \frac{S_{00} + S_{01} + S_{10}}{3} \right \right\}$
$\pi/2$	$\delta_{\pi/2} = \left \frac{S_{00} + S_{10}}{2} - \frac{S_{01} + S_{11}}{2} \right $
$3\pi/4$	$\delta_{3\pi/4} = \max \left\{ \left S_{01} - \frac{S_{00} + S_{10} + S_{11}}{3} \right , \left S_{10} - \frac{S_{00} + S_{01} + S_{11}}{3} \right \right\}$

The contribution of [18] is to calculate all the above measure values in DCT domain, leading to edge block pattern classification via DCT coefficients. Since we only need the horizontal edge and vertical edge pattern as analyzed earlier for compressed image content characterization, the measure values for these two patterns can be summarized as follows according to the work reported in [18].

$$\delta_0 = 2|V| = |X(1,0)| \quad (1)$$

$$\delta_{\pi/2} = 2|H| = |X(0,1)| \quad (2)$$

Where $X(1,0)$ and $X(0,1)$ are the DCT coefficients decoded from JPEG compressed image data.

From the (1) and (2), it can be seen that the two edge pattern can be implemented very fast since only two DCT coefficients are required. As a result, the three edge pattern classification can be implemented as summarized in Figure-5, where λ is a threshold selected via empirical approaches. In our implementation, λ is selected to be 50.

3 Construction of Indexing Keys

Following extraction of block edge patterns, the remaining issue is how to construct an effective indexing key to enable content-based search via these edge patterns. Standard techniques are represented by histogram approaches, due to the fact that histograms are effective in representing random variable distributions such as colour, intensity and luminance etc.

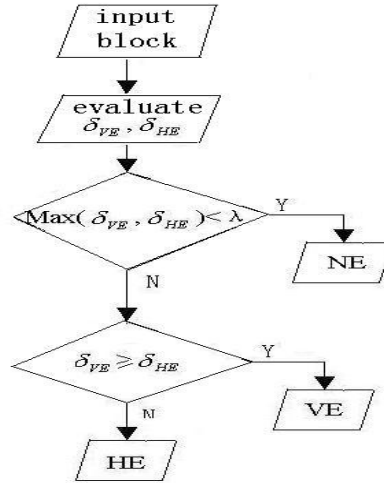


Fig. 5. Flow-chart for block edge pattern classifications (NE=no-edge, VE=vertical edge, HE=horizontal edge)

Given a block of 8x8 DCT coefficients, its edge pattern is firstly transformed into a set of three elements, $\psi = \{\#, -, | \}$, where # = no-edge, - = horizontal edge and | = vertical edge. When the histogram is constructed by calculating the number of occurrence for each edge pattern, the histogram will only have three elements, which is often referred to 1D edge-block-histogram (EBH). As such 1D edge-block-histogram has little information to reveal the image content, the occurrence of states can be implemented by considering combination of edge patterns, leading to construction of high dimensional histograms. As an example, if we examine the status of two edge patterns together, the total number of elements inside the histogram can be increased to 9 (a 2D EBH). A summary of the histogram dimension with respect to the number of it states is given in Table-II.

Table 2. Summary of histogram dimension with respect to its number of elements

1D EBH	2D EBH	3D EBH	4D EBH	5D EBH	6D EBH
3	9	27	81	243	729

From Table-2, it can be seen that 1D EBH and 2D EBH not only have small number of elements, but also contain little information for the location of those classified edge patterns inside the image, which are often useful to reflect the visual content. On the other hand, 5D EBH and 6D EBH incurs large number of states, which illustrate potential for high computing cost when millions of compressed images need to be indexed and searched via such keys. To this end, we focus on 3D EBH and 4D EBH in order to derive effective and efficient indexing keys.

To construct an edge block histogram based on combination of three or four edge block patterns, we define each state as shown in Figure-6.



Fig. 6. Definition of states for EBH, where (1) for 3D EBH and (2) for 4D EBH

Consequently, the edge block histogram (EBH) can be constructed as follows.

$$H_{rD} = \frac{\{C_1, C_2, \dots, C_N\}}{N} \tag{3}$$

Where $r \in [3,4]$, C_i stands for the total number of occurrence of state- i , and N for the number of elements inside the histogram, which is 27 for 3D EBH and 81 for 4D EBH.

To fully reflect the positional information of the edge patterns inside images, we need higher dimensional histograms, yet their negative side is that the size of the histograms would be increased to a non-manageable level. To achieve an appropriate balance, we propose a variable dimensional histogram following the spirit of run-length coding in the area of lossless data compression. Specifically, we raft scan all block edge patterns and count the number of same patterns inside one single run. Every time a different edge block pattern is encountered, it is regarded a broken run and thus a new run will be started. The output code for each run can be represented as follows:

$$R_i = \{\alpha_i, \eta_i\} \tag{4}$$

Where $\alpha_i \in [\#, -, |]$, and η_i stands for the number of α_i inside this single run.

As a result, the histogram can be represented as follows:

$$H_{run} = \frac{\{C_1, C_2, \dots, C_M\}}{M} \tag{5}$$

Where C_i counts the number of the state- i occurred inside the image after the run-length coding is finished, and R_i is regarded the same state as R_j only when $\alpha_i = \alpha_j$ and $\eta_i = \eta_j$.

4 Experimental Results and Analysis

Extensive experiments are carried out to evaluate the proposed algorithm, where two important issues regarding the evaluation design are addressed. These include: (i) test data set is prepared by establishing a database of 10287 JPEG compressed images, which are classified into categories of car, people, flowers, outdoor-scenes, and animals; (ii) a benchmark is selected out of the existing technique reported in the published literature [6], which presents a fair comparability since both algorithms search and retrieve images via DCT coefficients. To calculate the similarity between images, the same distance measurement as described in [6] is adopted, which is given below.

$$d(x, q) = \sum_{j=1}^n |k_x(j) - k_q(j)| \quad (6)$$

Where n is the dimension of the key, $k_x(j)$ and $k_q(j)$ are the keys of image x and q respectively after normalization.

Two phases of experiments were carried out for benchmarking purposes. In the first phase, we choose 100 images from a database, 20 out of each category, as the query images to search the database. The retrieval performance was measured in terms of precision, which is defined below.

Let n_c and n_f be the number of correctly retrieved images and wrongly retrieved ones respectively among the first M retrievals. The precision for the query image q is defined as:

$$Precision_q = \frac{n_c}{n_c + n_f} = \frac{n_c}{M} \quad (7)$$

In our experiments, we designed M to be 12 in line with the benchmark [6]. All results of the first phase experiments are summarized in Table-3, from which it can be seen that the proposed RLEBH (run-length edge block histogram) achieves the best performance in terms of retrieval precision rate, and all the proposed block-edge-pattern histograms outperform the benchmark. Figure 7 illustrates a sample comparison for visual inspections.

Table 3. Experimental results summary for the first phase

Average precision	Benchmark	3DEBH	4DEBH	RLEBH
Cars	0.17	0.67	0.64	0.63
Flowers	0.17	0.25	0.27	0.28
People	0.74	0.74	0.76	0.83
Outdoor-scene	0.38	0.33	0.39	0.43
Animals	0.23	0.18	0.11	0.23
Total-average	0.34	0.40	0.44	0.49

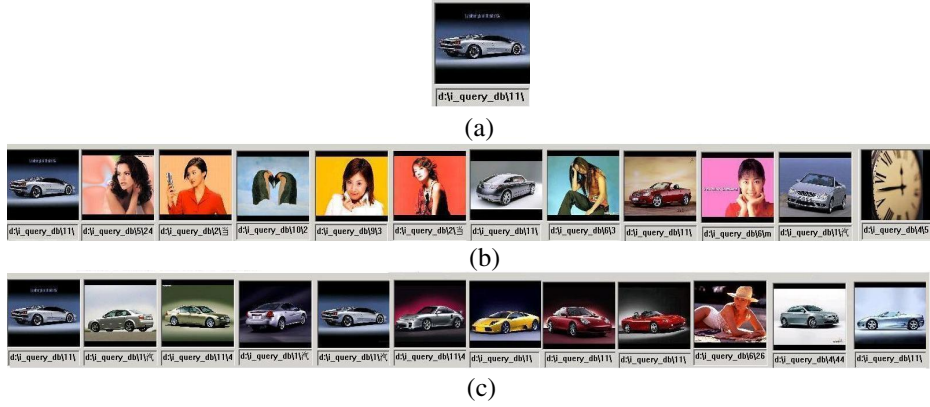


Fig. 7. Retrieval sample illustration, where (a) query image; (b) retrieved images by benchmark; (c) retrieved images by RLEBH

The second phase of experiments aims at evaluating the robustness of the proposed algorithm in retrieving images with orientation changes, such as rotation, zoom in or out, and noise affected etc. In this experiment, fifty images were chosen at random from the database. Specifically, each of the chosen images was processed as follows: clockwise rotation by 5 degrees, anti-clockwise rotation by 5 degrees, zoom out by 20%, zoom in by 20%, inverse color, horizontal reverse, vertical reverse, horizontal displacement by 10%, vertical displacement by 10%, adding Gaussian white noise (means equal to 0, variance equal to 0.04), adding salt & pepper noise (variance equal to 0.04) and adding speckle noise (variance equal to 0.04). Figure-8 illustrates an example of all the processes. Following that, all such processed images are added into the database. When one of them is taken as a query image, we would like to see how many of those processed images can be retrieved to test the robustness of indexing keys. To this end, we measure the retrieved results in terms of the recall and the average of the normalized modified retrieval rank over all queries (ANMRR) as described in [21].

Let n_c and n_m be the number of correct, and missed candidates, respectively, among the first M retrievals (M is 18 in our experiments). The recall for query image q is defined as:

$$Recall_q = \frac{n_c}{n_c + n_m} \tag{8}$$

To calculate ANMRR [22], we firstly produce an average rank for query image q as follows:

$$AVR(q) = \sum_{k=1}^{NG(q)} \frac{Rank(k)}{NG(q)}. \tag{9}$$

Where $NG(q)$ is the number of relevant images for query q .

Secondly, the average rank is modified into:

$$MRR(q) = AVR(q) - 0.5 - \frac{NG(q)}{2} \quad (10)$$

and thirdly, the normalized modified retrieval rank is derived by:

$$NMRR(q) = \frac{MRR(q)}{\max\left\{4 \times NG(q), \max_{q \in Q}(NG(q))\right\} + 0.5 - 0.5 * NG(q)} \quad (11).$$

Where Q is the number of all queries.

Finally, the ANMRR is calculated for all queries as given below:

$$ANMRR = \frac{1}{Q} \sum_{q=1}^Q NMRR(q). \quad (12)$$

All the experimental results produced in the second phase are summarized in Table-4, which clearly indicates that the proposed algorithm is more robust than the benchmark in terms recall rates, and achieves superior performances measured by ANMRR values.

Table 4. Summary of second phase experimental results

Methods	Benchmark	3_D EBH	4_D EBH	RL EBH
Average recall after clockwise rotate by 10°	0.78	0.52	0.50	0.50
Average recall after anti-clockwise rotate by 10°	0.74	0.40	0.48	0.52
Average recall after zoom out 20%	0.96	0.18	0.12	0.72
Average recall after zoom in 20%	0.94	0.44	0.62	0.90
Average recall after inverse color	0.02	0.98	1.00	1.00
Average recall after horizontal upset	0.92	0.92	0.92	0.98
Average recall after vertical upset	0.92	0.90	0.88	0.94
Average recall after horizontal displacement 10%	0.82	0.44	0.44	0.50
Average recall after vertical displacement 10%	0.86	0.44	0.40	0.52
Average recall after adding gaussian white noise	0.02	0.64	0.66	0.90
Average recall after adding salt& pepper noise	0.02	0.18	0.22	0.42
Average recall after adding speckle noise	0.08	0.44	0.44	0.60
Average recall	0.59	0.54	0.56	0.71
ANMRR	0.1	0.12	0.14	0.08

5 Conclusions

In this paper, we proposed an effective algorithm for content-based image retrieval. In comparison with existing techniques, the proposed algorithm illustrates the following advantages and features: (i) robust to a range of orientation changes which are frequently encountered during transmission, processing and storages; (ii) directly operates in compressed domain, which are suitable for large compressed image databases; (iii) extremely low computing cost suitable for real-time implementation and fast image content management applications. As a matter of fact, the cost for classifying edge-blocks only requires partial decoding of two DCT coefficients $X(0,1)$ and $X(1,0)$ from (1) and (2). After that, all needed is to construct a histogram by counting the number of occurrences for each edge block pattern state. Finally, extensive experiments also support that the proposed algorithm outperforms the similar counterpart in terms of retrieval precision rates. Therefore, the proposed algorithm would have significant potential for practical applications either as a stand-alone tool or as a component working with other tools towards efficient and effective visual content management.

Finally, the authors wish to acknowledge the financial support under EU IST FP-6 Research Programme as funded for the integrated project: LIVE (Contract No. IST-4-027312).

References

- [1] M.K. Mandal, F. Idris and S. Panchanatha, A critical evaluation of image and video indexing techniques in the compressed domain, *Image and Vision Computing*, Vol.17, pp.513-529, 1999
- [2] M. Shneier, M. Abdel-Mottaleb, Exploiting the JPEG compression scheme for image retrieval, *IEEE Trans. Pattern Anal. Mach. Intell.* 18(8) ,pp.849-853, 1996.
- [3] Shih-Fu Chang, Compressed domain techniques for image/video indexing and manipulation, *IEEE International Conference on Image Processing*, pp.314-317, 1995.
- [4] Chong-Wah Ngo, Ting-chuen Pong, Exploiting image indexing techniques in DCT domain, *Pattern Recognition* 34 pp.1841-1851, 2001.
- [5] Y.S. Hsu, S. Prum, J.H. Kagel, H.C. Andrews, Pattern recognition experiments in the Mandala/Cosine domain, *IEEE Trans. Pattern Anal. Mach. Intell.* 5(5), pp.512-520, 1983.
- [6] Guocan Feng, Jianmin Jiang, JPEG compressed image retrieval via statistical features, *Pattern Recognition* 36, pp. 977-985, 2003.
- [7] Lay Jose A, Ling Guang, Image retrieval based on energy histograms of the low frequency DCT coefficients [A]. In: *Proc of International Conference on Acoustics, Speech and Signal Processing[C]*, Phoenix, Arizona, USA, 6: 3009-3012, 1999.
- [8] D.G. Sim, H.K. Kim, R.H. Park, Fast texture description and retrieval of DCT-based compressed image [J], *Electronic Letters*, 37(1): 18-19, 2001.
- [9] Jianghong Liu, Haiming Gu, Image retrieval in various domains, *Computers & Graphics* 27, pp.807-812, 2003.
- [10] Daidi Zhong, Irek Defee, DCT histogram optimization for image database retrieval, *Pattern Recognition Letters*, 2005.
- [11] Jun Wei Han, Lei Guo, A shape-based image retrieval method using salient edges, *Signal processing: Image communication* 18, pp.141-156, 2003.

- [12] Minakshi Banerjee, Malay K.Kundu, Edge based features for content based image retrieval, *Pattern Recognition* 36, pp.2649-2661, 2003.
- [13] D.S. Kim and S.U. Lee, Image vector quantizer based on a classification in the DCT domain, *IEEE Trans. Commun.*, vol. 39, no.4, pp.549-556, Apr.1991.
- [14] S. Soltane, N. Kerkeni, J.C. Angue, The use of two dimensional discrete cosine transform for an adaptive approach to image segmentation, *Proceedings of the SPIE Image and Video Processing IV*, pp.242-251, 1996.
- [15] B. Shen and I.K. Sethi, Direct feature extraction from compressed images, in *Proc. SPIE: Storage and Retrieval for Still Image and Video Databases IV*, vol.2670, pp.404-414, Mar.1996.
- [16] S.-W. Lee, Y.-M. Kim, and S.W. Choi, Fast scene change detection using direct feature extraction from MPEG compressed videos, *IEEE Trans. Multimedia*, vol.2, no.4, pp.240-254, Dec.2000.
- [17] H. Li, G. Liu, and Y. Li, An effective approach to edge classification from DCT domain, in *Proc. IEEE Int. Conf. Image Processing*, pp.940-943, Sep.2002.
- [18] Hyun Sung Chang, Kyeongok Kang, A compressed domain scheme for classifying block edge patterns, *IEEE Transactions on image processing*, vol.14, no.2, Feb. 2005.
- [19] C.S. Won, D.K. Park, and S.-J. Park, Efficient use of MPEG-7 edge histogram descriptor, *ETRI J.*, vol.24,no.1, pp.23-30, Feb.2002.
- [20] H.J. Bartsch *Handbook of mathematical formulas*, Academic press, 1974;
- [21] "MPEG Vancouver Meeting", ISO/IEC JTC1/SC29/WG11, Experimentation Model Ver.2.0, Doc. N2822, July 1999.
- [22] Ho Young Lee, Ho Keun Lee, and Yeong Ho Ha, Spatial color descriptor for image retrieval and video segmentation, *IEEE Transaction on Multimedia*, vol.5, No.3, Sep. 2003.