International Journal of Hybrid Information Technology
Vol.9, No.3 (2016), pp. 389-398
http://dx.doi.org/10.14257/ijhit.2016.9.3.36

Adaptive Difference Compensation Vector Quantization Using Dynamic Image Block Adjustment

Meisen Pan and Fen Zhang

College of Computer Science and Technology Hunan University of arts & science, Changde 415000 China
pmsjjj@126.com

Abstract

A method for image compression, adaptive difference compensation vector quantization using dynamic image block adjustment is proposed in this paper. Before image coding, this proposed method analyzes the similarity values between the encoding image sub-block and its 8-neighbor sub-blocks, then determines the encoding rule according to the similarity value and the preset threshold. If the similarity value is approximate to the threshold, we use the same codeword to encode the neighbor block and the encoding image block; otherwise, the neighbor block is separately encode. When encoding, the difference between each image block and its matching codeword is first computed to obtain the difference image, and then the sign bits of the pixel difference is imposed the running length coding on and attached after the codeword index. When decoding, this proposed method restores the compressed image according to the codeword index, performs the running length encoding to deal with decode the attached information, uses the window of 3×3 *size to tackle the adaptive difference compensation and derive the final decoding image. The experiment results reveal that this proposed method can improve the encoding speed and image restoration performance against the normal vector quantization.*

Keywords: *Vector Quantization, Image compression, Codebook, LBG Algorithm*

1. Introduction

Vector Quantization (VQ), as an efficient data compression technology, has been widely used in the fields of data clustering, speech coding, image compression and so on. In general, VQ consists of the following three processes: designing codebook, encoding and decoding. Designing codebook and the efficient methods for searching the codebook have serious influence on the performance of VQ. Therefore, the researchers around the world presented many practical improved algorithms to further boost the performance of VQ. Neto, *et al.*, [1] introduced vector quantization algorithms for building reduced-set support vector machines (SVD) and least squares support vector machines (LSSVM) classifiers, and comprehensive computer simulations using synthetic and real-world datasets reveal that the proposed approach is very efficient and independent of the type of VQ algorithm used. Rao, *et al.*, [2] proposed two-stage segmentation approach for splitting the TV broadcast news bulletins into sequence of news stories and codebooks derived from vector quantization are used for retrieving the segmented stories. Hossan, *et al.*, [3] presented a new and novel Automatic Speaker Recognition (ASR) system, which includes novel feature extraction and vector classification steps utilizing distributed discrete cosine transform based Mel frequency central coefficients and fuzzy vector quantization. Corte, *et al.*, [4] proposed a fuzzy classification VQ for image coding. Chen, *et al.*, [5] proposed image retrieval based on quadtree classified vector quantization for a color image retrieval scheme. Akbari, *et al.*, [6] presented a novel multiresolution, perceptual and vector quantization based video

ISSN: 1738-9968 IJHIT

coding scheme and the detail subbands are vector quantized using an adaptive vector quantization scheme. Experimental results indicate that the proposed codec outperforms the adaptive subband vector quantization coding scheme subjectively and objectively at all bit rates. Chaux, *et al.*, [7] investigated vector quantization combined with regularity constraints, and presented an optimization approach to the problem involving a novel two-step, iterative, flexible, joint quantizing-regularization method featuring both convex and combinatorial optimization techniques. Li, *et al.*, [8] proposed a novel watermarking approach which involves robust watermark and fragile watermark in a two-stage quantization technique, and selected the codeword in the original codebook to ensure optimality. Experimental results show that the proposed method can be used respectively for protecting the copyright and authenticating the integrity of the audio aggregation. Hu, *et al.*, [9] proposed an improved image coding scheme based on vector quantization, and the mean value of the image block is taken as an alternative block encoding rule to improve the image quality in the proposed scheme. The results show that the proposed scheme achieves better image qualities than vector quantization while keeping low bit rates.

2. Vector Quantization Technology and its Existing Drawback

VQ works in k -Dimensional Euclidean space R^k with $k>1$. Suppose that $\boldsymbol{T}$ expressing the input vector is viewed as a random vector with a given probability distribution function in space R^k , $\boldsymbol{t}$ is a value of $\boldsymbol{T}$, and $\boldsymbol{A}\subset R^k$ is a value space of random vectors. Now we define the I-level quantizer $\boldsymbol{Q}=\{\boldsymbol{Y},\pi\}$ of $\boldsymbol{A}$, which includes the following three elements:

(1) The codebook $\boldsymbol{Y}=\{\boldsymbol{y}_i;i=1,2,\cdots,I\}$ and $\boldsymbol{y}_i$ is the ith codeword.

(2) For a partition $\varphi=\{\boldsymbol{R}_i;i=1,2,\cdots,I\}$ of $\boldsymbol{A}$, $\bigcup_{i=1}^{I}\boldsymbol{R}_i=\boldsymbol{A}$ and $\bigcap_{i=1}^{I}\boldsymbol{R}_i=\Phi$.

(3) The mapping is defined by the following the expression

$$\boldsymbol{Q}:\ \boldsymbol{A}\rightarrow\boldsymbol{Y} \tag{1}$$

$$\boldsymbol{y}_i=\boldsymbol{Q}(\{\boldsymbol{t}\,|\,\boldsymbol{t}\in\boldsymbol{R}_i\}) \tag{2}$$

The statement above is so called the formalization description of VQ. When VQ is used to quantize the image, the sending side first divides the image into the sub-blocks with size being $n\times n$ to build the column vector with size being $n^2\times1$ in row- or column-major order, and then probe the corresponding codeword approximate to each sub-block in the codebook based on the rule of minimum distortion. When the image is transmitted, only the codeword index of each sub-block in the codebook is sent. When the receiving side encodes the image, the decoder only needs to restore the image in the order of the codeword index transmitted. For a compressing image, generally speaking, the Bit Rate (BR) is defined by

$$BR=\frac{\log_2^I}{k} \tag{3}$$

here I denotes the size of the codebook (namely the number of the codewords), k signifies the dimensional number of the codewords (namely the size $k=n\times n$ of each sub-block). For I , its size is usually 256, 512 and 1024. Therefore, if the codebook size I is determined, then the bit rate is a definite value. So overall the less is the bit rate and the more is the compression rate, then the less is the data of the compressed image. In this paper, we select $n=4$, which means that the dimensional number $k=16$.

How to extract the similarity and the difference between sub-blocks, improve the encoding speed and promote the image quality is our main problem which needs to be addressed after determining I and k . There exists a lot of redundant information in the encoded image after decoding the image by VQ, as shown in Figure 1, where the block labeled by a rectangle needs to first divide into 9 sub-blocks with size being 4×4 and encode them respectively. Viewed from the point of human visual system, however, these 9 sub-blocks are distinguished, which is so called visual redundancy existing in the image. Therefore, only needing to search the responding codeword of a sub-block in the codebook, the other sub-blocks have the same codeword as it, which can reduce the encoding time and improve the encoding speed. In addition, each sub-block is substituted the corresponding codeword for and as a results there are some differences between the restore image and the original image, which has a great influence on the quality of the restore image to a large extent. These differences have a relatively less influence on the quality of the restore image when VQ is used to quantize the image which produces the codebook, shown in Figure 2a. We note that, however, these differences are much higher, have a bit greater influence on the quality of the restore image, and create serious image distortion when VQ is employed to quantize the image which doesn't produce the codebook, shown in Figure 2b.

142	143	140	140	141	139	137	138	139	144	150	158
128	130	129	127	128	126	126	127	128	136	144	147
128	130	128	129	129	127	126	126	131	139	145	146
129	129	129	128	129	127	125	131	139	143	145	143
128	129	130	131	129	127	127	136	142	143	142	139
127	130	131	132	129	128	129	136	144	141	140	138
130	130	131	132	129	130	135	141	144	142	137	134
131	130	133	133	131	135	141	146	143	139	133	131
131	134	135	134	136	141	145	145	144	137	131	132
135	135	135	135	142	144	142	141	139	135	130	133
135	134	136	140	145	140	135	136	133	133	131	134
135	133	139	145	139	130	127	127	128	136	135	132

Figure 1. Block with Size being 12×12 in the Top-Left Corner Pixels in Image Lena

(a) (b) (c) (d)

Figure 2. The Encoding and Restore Images. (a) Encoding Mage Producing the Codebook; (b) Decoded Image by the Codebook which is Produced by Image (a); (c) Encoding Image Producing no Codebook; (d) Decoded Image from Image (c) by the Codebook which is Produced by Image (a)

3. Adaptive Difference Compensation Vector Quantization using Dynamic Image Block Adjustment

From the analysis above, VQ cannot deal with the existing problems mentioned previously. In order to address the drawback, we propose Adaptive Difference Compensation Vector Quantization Using Dynamic Image Block Adjustment (ADCVQ).

First, we divide the encoding image into the 4×4 sub-blocks. For a encoding sub-block $\boldsymbol{B}$, we mark its 8-neighborhood $\boldsymbol{NW}=\{\boldsymbol{NW}_i; i=0,1,\cdots,7\}$ illustrated in Figure 3, where each sub-block built in row order to form a column vector with dimension being 16×1. In order to tell each sub-block $\boldsymbol{NW}_i$, 3 bits are used to express its position, namely, the range from 000 to 111.

$\boldsymbol{NW}_3$	$\boldsymbol{NW}_2$	$\boldsymbol{NW}_1$	$\cdots$
$\boldsymbol{NW}_4$	$\boldsymbol{B}$	$\boldsymbol{NW}_0$	$\cdots$
$\boldsymbol{NW}_5$	$\boldsymbol{NW}_6$	$\boldsymbol{NW}_7$	$\cdots$
$\cdots$	$\cdots$	$\cdots$	$\cdots$

Figure 3. 8-Neighborhood of Sub-Block $\boldsymbol{B}$

Before encoding, the similarity between sub-block $\boldsymbol{B}$ and its each sub-block $\boldsymbol{NW}_i$ is computed by the following expression,

$$S(\boldsymbol{B},\boldsymbol{NW}_i)=\frac{\boldsymbol{B}'\boldsymbol{NW}_i}{\|\boldsymbol{B}\|\cdot\|\boldsymbol{NW}_i\|} \tag{4}$$

where $\|\bullet\|$ denotes the norm. $\|\boldsymbol{B}\|=\sqrt{\sum_{j=0}^{15}t_j^2}$, $\|\boldsymbol{NW}_i\|=\sqrt{\sum_{j=0}^{15}nw_{ij}^2}$, $\boldsymbol{B}'\boldsymbol{NW}=\sum_{j=0}^{15}b_j nw_{ij}$, b_j is the gray value of the jth element in the sub-block $\boldsymbol{B}$, nw_{ij} signifies the gray value of the jth element in the sub-block $\boldsymbol{NW}_i$ ($i=0,1,\cdots,7$), and $S(\boldsymbol{B},\boldsymbol{NW}_i)\in[0,1]$. Now we rewrite the equation (4)

$$S(\boldsymbol{B},\boldsymbol{NW}_i)=\frac{\sum_{j=0}^{15}b_j nw_{ij}}{\sqrt{\sum_{j=0}^{15}b_j^2}\cdot\sqrt{\sum_{j=0}^{15}nw_{ij}^2}} \tag{5}$$

If $S(\boldsymbol{B},\boldsymbol{NW}_i)$ satisfies

$$S(\boldsymbol{B},\boldsymbol{NW}_i)\ge T_d \tag{6}$$

then $\boldsymbol{NW}_i$ is viewed as similarity to $\boldsymbol{B}$, in this case, the same codeword is used to encode $\boldsymbol{NW}_i$ and $\boldsymbol{B}$. Otherwise, $\boldsymbol{NW}_i$ needs to separately search the matching codeword in the codebook. Also, T_d, as a preset threshold, denotes the minimum similarity between two Adjacent sub-blocks.

In the encoding process stated above, the corresponding codeword is substituted for each sub-block and there must be difference between the sub-block and codeword, which results in the discrepancy between the corresponding pixels. We note that, the discrepancy is very crucial to derive a high-quality decoded image, especially when decoding the image producing no codebook.

However, unfortunately, VQ takes no consideration of the discrepancy, so the quality of the decoded image with blocky effect is obviously unsatisfactory. In this paper we try to use the discrepancy as much as possible to improve the decoded quality in the circumstances of only the public codebook.

When the above-mentioned method quantizes the image, there is no consideration of the discrepancy between the pixels before and after quantized. The difference image with the same size as the encoding image is acquired by computation of the difference between the sub-block and its corresponding codeword, which is essentially equivalent to commutation of the differences between the pixels in the sub-block and its corresponding values in the codeword. If the difference image is directly sent to the receiving side without any compressing operation, then this proposed method is meaningless. Therefore, we further handle and compress the difference image to reduce the difference information as soon as possible and then send them along with the codeword indexes in this paper. Suppose that $\boldsymbol{X}_i$ is the ith sub-block of the encoding image and $\boldsymbol{y}_j$ is its corresponding codeword, then the difference $\boldsymbol{D}_i$ between $\boldsymbol{X}_i$ and $\boldsymbol{y}_j$ constructs the ith sub-block of the difference image, that is

$$\boldsymbol{D}_i = \boldsymbol{X}_i - \boldsymbol{y}_j \tag{7}$$

Besides, $\boldsymbol{D}_{i,l}$ denotes the lth element of $\boldsymbol{D}_i$ and also means the difference between the lth pixel in the sub-block $\boldsymbol{X}_i$ and the lth value in the corresponding codeword $\boldsymbol{y}_j$. In order to compress the difference information as soon as possible, we only extract the sign $Sign_{i,l}$ of $\boldsymbol{D}_{i,l}$, namely

$$Sign_{i,l} = \begin{cases} 1 & \boldsymbol{D}_{i,l} \geq 0 \\ 0 & \boldsymbol{D}_{i,l} < 0 \end{cases} \tag{8}$$

here $Sign_{i,l} \in \{0,1\}$ is called the difference sign bit (DSB) and only needs a binary bit to express the pixel difference, which creates the DSB image with the same size as the encoding image. There are many consecutive 0s or 1s in the DSB image, so we use the run-length coding (RLC) to further compress the DSB and the bit rate, which produces little information along with the codeword indexes.

In conclusion, we can design the code structures of the encoding sub-block $\boldsymbol{B}$ and the sub-block $\boldsymbol{NW}_i$ in its 8-neighorhood, respectively, shown in Figure 4.We need to note that for the sub-block $\boldsymbol{NW}_i$ dissimilar to $\boldsymbol{B}$, we use the matching codeword to encode it and each pixel in it creates 1 DSB, shown Figure 4a. Moreover, the sub-block $\boldsymbol{NW}_i$ dissimilar to $\boldsymbol{B}$, we use three bits to encode it and each pixel in it also creates 1 DSB shown Figure 4b. Finally, all the DSBs (namely the DSB image) are encoded by RLC and attached to the codeword indexes in order, shown in Figure 4c.

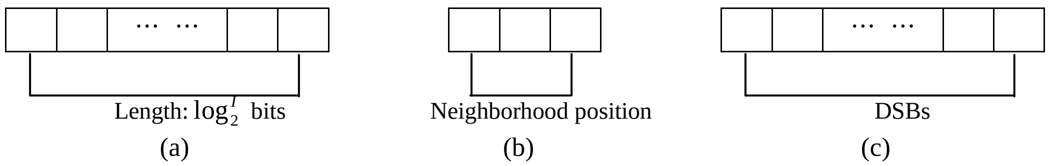

Figure 4. Code Structure (a) Structure of the Codeword of B; (b) Structure of NW_i (3 bits); (c) Structure of DSB (1 bit)

As stated above, the encoding process of this proposed method is introduced as follows.

Step1.For the encoding sub-block $\boldsymbol{B}$, the searching algorithm is first used to explore its matching codeword in the codebook and then it is encoded into $\log_2^I$ bits.

Step2.The similarity $S(\boldsymbol{B}, \boldsymbol{NW}_i)$ between $\boldsymbol{B}$ and the sub-block $\boldsymbol{NW}_i$ in its 8-neighborhood is computed. If $S(\boldsymbol{B}, \boldsymbol{NW}_i) \geq T_d$, then $\boldsymbol{NW}_i$ is similar to $\boldsymbol{B}$, which means that they can be encoded by using the same codeword and $\boldsymbol{NW}_i$ needs 3 bits. Instead, if $S(\boldsymbol{B}, \boldsymbol{NW}_i) < T_d$, then $\boldsymbol{NW}_i$ is dissimilar to $\boldsymbol{B}$, which means that $\boldsymbol{NW}_i$ needs to separately search the matching codeword in the codebook and needs $\log_2^I$ bits. The difference $\boldsymbol{D}_i$ between the sub-block and the matching codeword is calculated and $Sign_{i,l}$ is derived according to equation (8).

Step3. If there still exist the encoding sub-blocks, then go to Step1; otherwise go to Step4.

Step4. All the DSBs are encoded by RLC to reduce the transmitted information.

Step5. All the DSBs encoded by RLC are first attached to in order, and then sent to the receiving side along with, the codeword indexes.

When decoding, the receiving side restores the image in order of the codeword indexes received. In addition, adaptive difference compensation is carried out, namely, a 3×3 compensation window is used to deal with adaptive difference compensation the according to the DSBs received. Now we explain the decoding process of this proposed method as follows.

Step1. According to the codeword indexes received, the restore image $f(x, y)$ is produced.

Step2. The DSBs are decoded by RLC to obtain $Sign(x, y)$ in pixel (x, y).

Step3. For each pixel $f(x, y)$, its 3×3 compensation window $W_{x,y}$ with central pixel at (x, y) is extracted, namely,

$$W_{x,y} = \{ f(x+k, y+r) \mid k, r = -1,0,1 \} \tag{9}$$

Then the average value $Mean(W_{x,y})$ of all the pixels in $W_{x,y}$ is computed by the following expression

$$Mean(W_{x,y}) = \frac{1}{9} \sum_{k=-1}^{1} \sum_{r=-1}^{1} f(x+k, y+r) \tag{10}$$

Step4. According to the $Sign(x, y)$ of each pixel $f(x, y)$, the pixel gray value is adaptively compensated as the following rule

$$f'(x,y) = \begin{cases} f(x,y) + \left| f(x,y) - Mean(W_{x,y}) \right| & Sign(x,y) = 1 \\ f(x,y) - \left| f(x,y) - Mean(W_{x,y}) \right| & Sign(x,y) = 0 \end{cases} \tag{11}$$

where $f'(x, y)$ is the final decoded image.

In this proposed method, due to the consideration of the similarity between encoding sub-block and its 8-neighborhood, the computational load used for searching the codeword is significantly reduced. In addition, although the adaptive difference compensation improves the decoded image quality, it leads to an increase in the rate bit.

4. Results and Analysis

In the section, we choose Lena images with size being 256×256 and 512×512, and grayscale 256 respectively as the training images used for producing the codebooks with size being 256, 512 and 1024 respectively. This proposed method is performed in MATLAB 6.5 on PC with an Intel Dual-Core E5500 2.80GHz and 1GB RAM, running Windows XP. The codebook, being a 16×1 column vector, is generated by the K-means clustering algorithm, and the threshold $T_d = 0.98660$. In order to evaluate the quality of the restore image, peak signal to noise ratio (PSNR), as the performance index, is defined as follows.

$$PSNR = 10\log_{10}\left(\frac{M \times N \times 255^2}{\sum_{x=1}^{N} \sum_{y=1}^{M} (g(x,y) - f'(x,y))^2} \right) \tag{12}$$

where $g(x, y)$ and $f'(x, y)$ express the encoding and decoded images respectively, with size being $M \times N$. In the experiments, we compare this proposed method with VQ based on Full Search (VQFS).

The codebook with size being 256 is first created by Lena image in Figure 2a and then is used to compress Figure 2a (Lena) and 2c (MRI). Figure 5 reveals the restore results produced by this proposed method, and Figures 2b and 2d shows the restore results produced by VQFS. In addition, the experimental data are listed in Table 1.

Also, we first use Lena image with size being 512×512 to generate the codebook with size being 56, 512 and 1024 respectively, and then, the Lena and Pepper images with sizes all being 512×512 are encoded and decoded. The experimental data are demonstrated in Table 2.


(a)

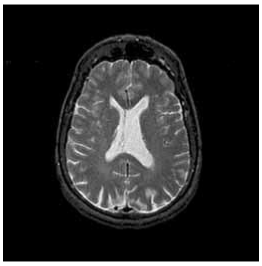
(b)

Figure 5. The Restore Images by this Proposed Method (a) Decoded Lena Image by the Codebook which is Produced by it (Lena Image in Figure 2a); (c) Decoded MRI Image by the Codebook which is Produced by Lena Image in Figure 2a

Table 1. Comparisons of Data Created by Tackling the 256×256 Experimental Images

Codebook size	Index	Lena image in Figure 2a		MRI image in Figure 2c	
		VQFS	ADCVQ	VQFS	ADCVQ
256	Time(s)	4.78	0.38	5.15	0.46
	BR(bpp)	0.50	0.557	0.50	0.563
	PSNR(dB)	29.641	30.803	29.614	32.510
512	Time(s)	8.08	1.46	8.29	1.50
	BR(bpp)	0.5625	0.613	0.5625	0.615
	PSNR(dB)	31.5303	32.079	30.293	33.655
1024	Time(s)	12.04	1.94	12.44	2.31
	BR(bpp)	0.625	0.687	0.625	0.710
	PSNR(dB)	34.4063	34.966	30.914	34.836

Table 2. Comparisons of Data Created by Tackling the 512×512 Experimental Images

Codebook size	Index	Lena image		Pepper image	
		VQFS	ADCVQ	VQFS	ADCVQ
256	Time(s)	13.05	2.11	13.38	2.32
	BR(bpp)	0.50	0.568	0.50	0.570
	PSNR(dB)	33.609	35.950	28.85	31.644
512	Time(s)	16.07	3.34	16.27	3.62
	BR(bpp)	0.5625	0.615	0.5625	0.625
	PSNR(dB)	34.973	36.127	29.2	31.937
1024	Time(s)	19.43	4.15	19.97	4.81
	BR(bpp)	0.625	0.693	0.625	0.717
	PSNR(dB)	36.521	37.885	29.609	32.585

From Figures 2 and 5, and, Tables 1 and 2, we can find that,

1) When the codebook size is the same, the computational costs of this proposed method are significantly reduced, as compared to VQFS, In other words, the simple computation results show that the processing time of the former is 3.96-11.34 times faster than that of the latter. This is because VQFS needs to explore all the codewords in the codebook, but this proposed method only needs to explore the matching codewords of the sub-blocks dissimilar to the encoding sub-block in an 8-neighborhood; in addition, since the DSBs are produced, it limits its ability to further reduce the processing time.

2) When the codebook size is the same, the restore quality of this proposed method is obviously improved, compared with VQFS. Considering the performance index PSNR, that of the former runs 0.4517-2.774dB ahead of that of the latter, which is in accordance with Tables 1 and 2. In addition, the difference also can be distinguished between Figure 2 and Figure 5. There exists relatively more distinct blocky effect in Figure 2d, while the visual effects presenting in Figure 5b are satisfactory.

3) When the codebook size is the same, the BR generated by this proposed method increases to some degree in comparison with VQFS. That is, the compression rate of the former decreases and the compressed image has a slightly larger size. This is because each NW_i allocates 3 bits to represent the position information and inserts the DSB. Although RLC is used to further reduce to the bits, the BRs still slightly increases. For VQFS, its sub-block only needs $\log_2^N$ to be decoded. Therefore, how to further improve the restore image quality and the same time drastically reduce the BRs, is the next focus in our research.

4) The increment of PSNR derived from the restore image whose original image (For example Figure 2a) is used to produce the encoding and decoding codebook, is less; Conversely, he increment of PSNR derived from the restore image whose original image (For example Figure 2c) doesn’t produce the encoding and decoding codebook, which shows that this proposed method can further tap the potential of VQ to reduce the data by the additional DSB, and provides a new approach to improving the decoded quality of VQ in the circumstances of only the public codebook.

5. Conclusions

In this paper, we propose adaptive difference compensation vector quantization using dynamic image block adjustment, and this proposed method makes full use of the similarity between sub-blocks and the additional DSB created by computing the difference image to encode the sub-blocks in the 8-neighoborhood. The experiments reveal that this proposed method is superior to VQFS in reducing the processing time and improving the restore image quality.

Acknowledgements

This work is supported by the key Scientific Research Fund of Hunan Provincial Education Department, P.R. China (No.13A064), supported by National Natural Science Foundation of China (No.61403136), supported by the Construct Program of the key Discipline in Hunan University of Arts and Science, and supported by the Doctor Scientific Research Startup Project Foundation of Hunan University of Arts and Science.

References

[1] A. R. R. Neto and G. A. Barreto, “Opposite Maps: vector quantization algorithms for building reduced-set SVM and LSSVM classifiers”, Neural Process Lett, vol. 37, no. 1, **(2013)**, pp. 3-19.

[2] K. S. Rao and K. Pachpande, “Segmentation, indexing and retrieval of TV broadcast news bulletins using Gaussian mixture models and vector quantization codebooks”, Int J Speech Technol, vol. 17, no. 3, **(2014)**, pp. 259-269.

[3] M. A. Hossan and M. A. Gregory, "Speaker recognition utilizing distributed DCT-II based Mel frequency cepstral coefficients and fuzzy vector quantization", Int J Speech Technol, vol. 16, no. 1, **(2013)**, pp. 103-113.
[4] R. L. Corte and A. P. A. Alves, "fuzzy classified vector quantizer for image coding", IEEE T Commun, vol. 43, no. 234, **(1995)**, pp. 207-215.
[5] H. H. Chen, J. J. Ding and H. T. Sheu, "Image retrieval based on quadtree classified vector quantization", Multimed Tools Appl, vol. 72, no. 2, **(2014)**, pp. 1961-1984.
[6] A. S. Akbari, P. B. Zadeh and T. Buggy, "Multiresolution, perceptual and vector quantization based video codec", Multimed Tools Appl, vol. 58, no. 3, **(2012)**, pp. 569-583.
[7] C. Chaux, A. Jezierska and J. C. Pesquet, "A spatial regularization approach for vector quantization", J Math Imaging Vis, vol. 41, no. 1, **(2011)**, pp. 23-38.
[8] J. Li, R. D. Wang and D. Q. Yan, "A multipurpose audio aggregation watermarking based on multistage vector quantization", Multimed Tools Appl, vol. 68, no. 3, **(2014)**, pp. 571-593.
[9] Y. C. Hu, W. L. Chen and C. C. Lo, "Improved vector quantization scheme for grayscale image compression", Opto-Electron Rev, vol. 20, no. 2, **(2012)**, pp. 187-193.

Authors

Meisen Pan, He was born in 1972 and graduated from Hunan Normal University, China, in 1995. He received the M.S. degree from Huazhong University of Science and Technology, China, in 2005. He obtained the Ph.D. degree from Central South University, China, in 2011. He has published more than 30 papers on journals and conferences. His research interests include biomedical image processing, information fusion and software engineering.

Fen Zhang, He was born in 1972 and graduated from Hunan University, China, in 1996. He received the M.S. degree from Hunan University, China, in 2005. He has published more than 10 papers on journals and conferences. His research interests include image processing and computer graphics.