International Journal of Signal Processing, Image Processing and Pattern Recognition
Vol.9, No.8 (2016), pp.1-10
http://dx.doi.org/10.14257/ijsip.2016.9.8.01

An Adaptive Orthogonal M-Split Initialization Method for VQ Codebook Generation

Weijun He, Qianhua He and Jichen Yang

School of Electronic and Information Engineering, South China University of Technology, Guangzhou, China
h.weijun@mail.scut.edu.cn, parrot2003@sohu.com

Abstract

Linde–Buzo–Gray (LBG) algorithm is a universal method to design codebook in vector quantization(VQ). This paper proposed an adaptive orthogonal M-split initialization method to improve the computational efficiency of LBG algorithm. The method splits one code word into 2, 4 or 5 new code words with adaptive split coefficient vectors and set the increment to be orthogonal in 4-split and 5-split situations, aiming at decreasing the iterations of the following clustering. Experiment is conducted on both TIMIT and RASC863 speech database, which shows that the proposed algorithm provides a reduction of 18%~45% in designing codebook in size of 64~2048 with almost equal VQ performance, compared with the universal codebook generation algorithm.

Keywords*: LBG algorithm, vector quantization, codebook design, pattern recognition, low bit rate speech coding*

1. Introduction

Vector quantization(VQ) is a key technology in multimedia data compression and pattern recognition [1]. A vector quantizer can be regarded as a mapping Q from a l-dimensional Euclidean space $\boldsymbol{R}^l$ to a finite subset $\boldsymbol{C}$ as follows:

$$Q : \boldsymbol{R}^l \to \boldsymbol{C} \ . \tag{1}$$

Where $\boldsymbol{C} = \left\{ \boldsymbol{y}_i \middle| i = 1, \cdots, N ; \boldsymbol{y}_i \in \boldsymbol{R}^l \right\}$ is called the codebook, $\boldsymbol{y}_i$ represents i th code word in the codebook $\boldsymbol{C}$ and N is the codebook size.

Although codebook design for VQ is an off-line computational task, its high complexity and need for repeated access to a large file of training data generally place severe limitations on the size of the training set [2]. The generalized Lloyd clustering algorithm [3] (referred to as the LBG algorithm) proposed by Linde, Buzo and Gray is the most popular method for unstructured codebook design. LBG algorithm used in codebook generation is conceptually simple and easy to implement, but it is not computationally efficient due to its iterative procedure of exhaustive checking of $\boldsymbol{y}_i$ is the closest code word to training vector among all code words in codebook $\boldsymbol{C}$.

As an iterative procedure, LBG starts with an initial codebook $\boldsymbol{C}_0$, then performs the clustering process until the improvement of the overall average distortion between the last two successive iterations is not significant enough. To reduce the computational complexity of LBG algorithm, many fast algorithms which mainly consider the clustering process have been developed, such as partial distortion search (PDS) [4], triangle inequality eliminating rule (TIE) [5] and mean-ordered partial codebook search (MPS) [6]. Another approach toward fast codebook generation is to provide the LBG algorithm with a better initial codebook $\boldsymbol{C}_0$, including random method [7], pruning [8], pairwise nearest-neighbor(PNN)[9], product codes [10] and binary split [11] (b-split). Although

ISSN: 2005-4254 IJSIP

random method and product codes are easy to implement, they provide poor results, *i.e.* a larger number of iterations and poor ultimate codebooks. The pruning, PNN and b-split give better results but have lower computational efficiency. Some novel initial codebook methods are proposed in recent years [12-15]. In Ref. [13], the training vectors are sorted according to the norm and partitioned into groups. The initial codebook is composed of the centroid of each group. Chen *et al* [14-15] improve the method in Ref. [13] by transforming training vectors into the Hadamard domain. Among the above codebook initialization methods, the b-split initialization method has been universal in codebook generation application, as is introduced in many textbooks [11-16].

In this paper, an adaptive orthogonal M-split method is proposed to reduce the computational cost of LBG algorithm. The method splits one code word into 2, 4 or 5 new code words with adaptive split coefficient vectors, the increment of new code words is set to be orthogonal in 4-split and 5-split situations. The paper analyzes the complexity of split LBG algorithm in Section 2 and describes the adaptive orthogonal M-split method in Section 3.

2. Complexity Analysis

The general b-split LBG algorithm [11] includes two parts: codebook initialization and clustering iteration process. In codebook initialization, each codeword is split into two new code words, which are used to generate 2^N new codewords by iteration. In clustering iteration process, an exhaustive search is conducted to classify the training vectors. Each training vector is compared with all code words in current codebook with certain distortion measure and assigned to the nearest code word. Finally, each code word is updated with the centroid of training vectors that are mapped to this code word. Hence, the computational time of the b-split LBG algorithm $t_{b-split\ LBG}$ is as Equation (2):

$$t_{b-split\ LBG} = \sum_{r=1}^{\log_2 N_d} t_{distort}(r) + t_{com}(r) + t_{up}(r). \tag{2}$$

Where

$$t_{distort}(r) = m_b(r)\,2^r n t_d. \tag{3}$$

$$t_{com}(r) = m_b(r)\left(2^r - 1\right) n t_c. \tag{4}$$

$$t_{up}(r) = m_b(r)\,2^r t_{up}. \tag{5}$$

The desired codebook size N_d is power of 2. $t_{distort}(r)$ is time of computing distortion and $t_{com}(r)$ is time of comparing distortion, $t_{up}(r)$ is time of updating. n is number of training vectors, $m_b(r)$ is iterations after r^{th} split, t_d is time of computing the distortion between two vectors, t_c is time of comparing two distortion values, t_{up} is time of updating code words.

According to Equation (2), we consider to employ more splitting code words in codebook initialization in order to reduce the split times and its following iterations. Given $M \geq 2$, then

$$2 \leq M \;\Rightarrow\; \log_2 N_d \geq \log_M N_d \;\Rightarrow\; \log_2 N_d \geq \log_M (N_d - \delta). \tag{6}$$

Where $N_d = M^I + \delta$ and I is M-split times, δ is the complement. The computational time of M-split codebook generation algorithm is as Equation (7):

$$t_{M-split\ LBG} = \sum_{r=1}^{\log_M (N_d - \delta)} t_{M-distort}(r') + t_{M-com}(r') + t_{M-up}(r'). \quad (7)$$

Where

$$t_{M-distort}(r') = \begin{cases} m_M(r') M^{r'} n t_d & r' < I, \\ m_M(r')(M^{r'} + \delta) n t_d & r' = I. \end{cases} \quad (8)$$

$$t_{M-com}(r') = \begin{cases} m_M(r')(M^{r'} - 1) n t_c & r' < I, \\ m_M(r')(M^{r'} + \delta - 1) n t_c & r' = I. \end{cases} \quad (9)$$

$$t_{M-up}(r') = \begin{cases} m_M(r') M^{r'} t_{up} & r' < I, \\ m_M(r')(M^{r'} + \delta) t_{up} & r' = I. \end{cases} \quad (10)$$

Where $m_M(r')$ is number of iterations after r'^{th} M-split initialization.

From the above analysis, it is sufficient to show that computational time of codebook generation can be reduced by decreasing split times as soon as M is suitably selected and iterations $m(r')$ can be minimized in reason. Thus, split method is the key for the assumption and an adaptive orthogonal M-split method is presented in Section 3.

3. Adaptive Orthogonal M-Split Method

In VQ, codebook generation procedure is essentially to search the optimum positions for finite code words in l - dimension space adhering to certain distortion measure. The M-split LBG algorithm, which searches the optimum positions by clustering method after split initialization, is an iteration process of searching the optimum codebook from initial codebook. When split occurs, if M-split new code words can be appropriately located, then not only times of split are reduced, but also the number of iterations will be greatly minimized. Differ from the b-split method, adaptive orthogonal M-split method splits one code word into 2, 4 or 5 new code words with adaptive split coefficient vectors in codebook initialization, the increment of new code words is set to be orthogonal in 4-split and 5-split situations. There are three parts for adaptive orthogonal M-split initialization: codebook generation scheme, orthogonal M-split method and calculation of adaptive split coefficient vector.

3.1. Codebook Generation Scheme

The relationship of b-split and M-split in generating codebook in size of N_d is as Equation (11):

$$N_d = 2^R = \begin{cases} 4^{R/2} & R = 2t,\ 1 \le t \le 6, \\ 4^{\lfloor R/2 \rfloor} \cdot 2 & R = 2t-1,\ t = 1, 2, 3, 4, 6, \\ \left(5^{\lfloor R/2 \rfloor - 1} + \delta'\right) \cdot 4 & R = 2t-1,\ t = 5. \end{cases} \quad (11)$$

Where R is the b-split times. According to Equation (11), a scheme of codebook generation is given in Table 1 for codebook in size of N_d, including times of split initialization I, number of split code words M and number of code words compensation δ'.

Table 1. Scheme of Codebook Generation

Codebook size N_c	64	128	256	512	1024	2048	4096
Number of split M	4	4	4	5	4	4	4
Times of split I	3	4(note2)	4	4(note1)	5	6(note2)	6
Number of compensation δ'	0	0	0	3	0	0	0

Note 1: $M = 4$ for the last split. Note 2: $M = 2$ for the last split.

3.2. Adaptive Orthogonal M-Split Method

In order to split each code word $\boldsymbol{Y}$ into M new code words and minimize the iterations $m_M(r')$ mentioned in Equation (8)~ Equation (10), an adaptive orthogonal M-split method is proposed. The orthogonal split method for the first two dimensional components of code word $\boldsymbol{Y}$ is described as Figure 1.

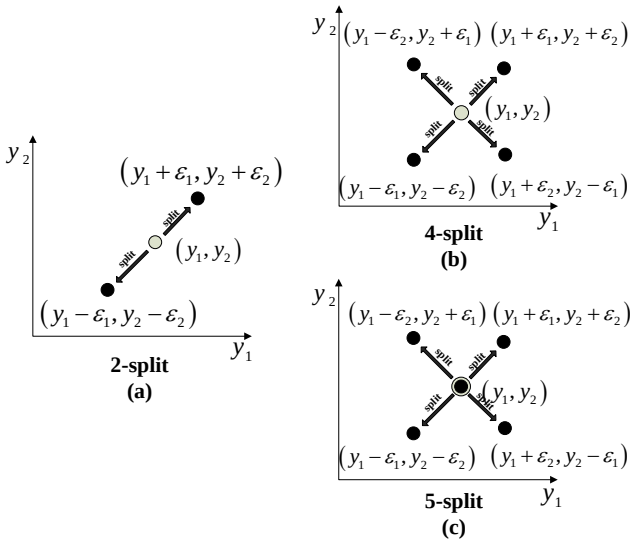

Figure 1. Orthogonal M-split Method for the First Two-Dimensional Components of Code Word Y ($M = 2$, $M = 4$ and $M = 5$)

In Figure 1, each code word $\boldsymbol{Y} = (y_1, y_2, y_3, y_4, \cdots, y_{2n-1}, y_{2n})$ is split into M new code words $\boldsymbol{Y}_m$ as follows:

$M = 2$, $\boldsymbol{Y}_1 = \boldsymbol{Y} + \Delta\boldsymbol{Y}$, $\boldsymbol{Y}_2 = \boldsymbol{Y} - \Delta\boldsymbol{Y}$ (Figure.1(a));

$M = 4$, $\boldsymbol{Y}_{1,2} = \boldsymbol{Y} \pm \Delta\boldsymbol{Y}$, $\boldsymbol{Y}_{3,4} = \boldsymbol{Y} \pm \Delta'\boldsymbol{Y}$, $\Delta\boldsymbol{Y} \cdot \Delta'\boldsymbol{Y} = 0$, *i.e.* $\Delta\boldsymbol{Y}$ and $\Delta'\boldsymbol{Y}$ are orthogonal (Figure.1(b));

$M = 5$, $\boldsymbol{Y}$ is split as $M = 4$ and retained as $\boldsymbol{Y}_5$ (Figure.1(c)) reasoning that $\boldsymbol{Y}$ is the centroid and is considered to be voted by all training vectors.

Where $\Delta\boldsymbol{Y} = (\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4, \cdots, \varepsilon_{2n-1}, \varepsilon_{2n})$ and $\Delta'\boldsymbol{Y} = (-\varepsilon_2, \varepsilon_1, -\varepsilon_4, \varepsilon_3, \cdots, -\varepsilon_{2n}, \varepsilon_{2n-1})$ $(n = 1, 2, \cdots, l/2)$ are the adaptive split coefficient vectors.

If k is even, components are paired with the continuous odd-even rule or interval odd-even rule. If k is odd, the minimal component $|y_{\min}|$ of vector $\boldsymbol{Y}$ is neglected and others are paired as k is even, then the minimal $\Delta\boldsymbol{Y} \cdot \Delta'\boldsymbol{Y} = |\Delta y_{\min}|^2$ is obtained. In this way, $\Delta\boldsymbol{Y}$ and $\Delta'\boldsymbol{Y}$ are close to be orthogonal.

Enough code words are compensated before last split initialization in order to reach the desired size according to Table 1. Suppose that $S_j = \left\{ \boldsymbol{x} \middle| d(\boldsymbol{x}, \boldsymbol{y}_j) \le d(\boldsymbol{x}, \boldsymbol{y}_i), i = 1, 2, \cdots, N, \boldsymbol{x} \in \text{Training Set} \right\}$ is an optimal partition voronoi of codebook size N and $Q(S_j)$ is the number of training vectors in S_j. $d(\square)$ is distortion measure and $\boldsymbol{x}$ is training vector. If δ' is not zero, S_j is sorted in descending order according to $Q(S_j)$, The first δ' code words $\boldsymbol{y}_c$ are selected according to Equation (12) and split through M-split method($M = 2$).

$$c = \arg\max_j Q(S_j). \tag{12}$$

3.3. Adaptive Split Coefficient Vector

When split occurs, the splitting amplitude can be different for each component of codeword with adaptive coefficient vector in order to make new code words in appropriate location and reduce the following iterations. We assume that $\Delta \boldsymbol{Y}_j = (\varepsilon_{j,1}, \varepsilon_{j,2}, \cdots, \varepsilon_{j,2n-1}, \varepsilon_{j,2n})$ is the adaptive split coefficient vector of the reproduction code word $\boldsymbol{y}_j$. $\varepsilon_{j,k}$ indicates k^{th} dimension component of the vector, which is determined as Equation (13):

$$\varepsilon_{j,k} = \frac{x_{j,k}^{\max} - x_{j,k}^{\min}}{\sigma}. \tag{13}$$

Where $x_{j,k}^{\max}$ and $x_{j,k}^{\min}$ indicate the maximum and minimum of k^{th} dimension component in optimal partition voronoi S_j respectively, σ is the scale parameter.

4. Adaptive Orthogonal M-Split LBG Algorithm

The adaptive orthogonal M-split LBG algorithm works by the following steps to generate a VQ codebook in size of N_d:

Step 1 Given a training sequence $TS = \{ \boldsymbol{x}_s | s = 1, 2, \cdots, L \}$, distortion threshold Tr and desired codebook size N_d, Set $f = 0$, $r' = 1$ and $D_{-1} = \infty$.

Step 2 The first code word $\boldsymbol{y}_0$ is computed from the centroid of training vectors.

Step 3 According to N_d, an M-split strategy is selected for codebook generation by checking Table 1.

Step 4 Adaptive M-split initialization. If $\delta' = 0$, each code word is split into M new code words with adaptive split coefficient vectors as described in Section 3.2. If $\delta' \neq 0$ and r' reach the split times I, δ' code words are compensated by the method mentioned in Section 3.2 and split as $\delta' = 0$ situation.

Step 5 Nearest-neighbor search. Each training vector is assigned to the cluster whose centroid is nearest to it, according to defined distortion measure.

Step 6 Centroids updating. Each code word is updated with the centroid of training vectors that are mapped to it. Then, the average distortion can be recaculated as Equation (14):

$$D_f = E\left[d(\boldsymbol{x}, \boldsymbol{y}_i) \right] = \sum_{i=1}^{N} p_i E\left[d(\boldsymbol{x}, \boldsymbol{y}_i) \middle| \boldsymbol{x} \in \boldsymbol{S}_i^{(f)} \right]. \tag{14}$$

Where $d(\boldsymbol{x}, \boldsymbol{y}_i)$ is the squared error distortion. If $(D_{f-1} - D_f) / D_f \le Tr$, the adaptive split coefficient vectors are calculated as described in Section 3.3 and go to Step 7; Otherwise, go to Step 5.

Step 7 If $N = N_d$, then stop; otherwise, $r' = r' + 1$, go to Step 4.

5. Results and Analysis

The proposed algorithm is implemented on Intel processor 2.80 GHz, 2GB RAM machine. Linear spectrum frequency (LSF) [17] which is commonly used in low bit rate speech coding, is transformed from 10-order linear predictive coefficient(LPC) parameters. The LPC parameters are extracted with 22.5ms window. The experiment uses approximate 100,000 vectors for training and 130,000 vectors for testing, which are extracted from the TIMIT [18] database and the RASC863 [19] database. We get various LSF codebooks with different sizes ranging from 64 to 2048. The orthogonal M-split LBG algorithm is compared with b-split LBG algorithm and LBG algorithm using method in Ref. [13]. The algorithms are evaluated by the average spectral distortion (ASD) [20] and mean square error(MSE), which are often used as an objective evaluation of the LSF VQ performance.

Figure 2, shows codebook generation time of all methods for codebook size 512. The time of orthogonal M-split LBG algorithm(200.01s) has a reduction of 18.01% and 78.12% respectively, compared with b-split LBG algorithm(243.97s) and method in Ref. [13] (914.18s). The computational time of method in Ref. [13] is much higher than the other two methods due to large amount of empty voronoi regions.

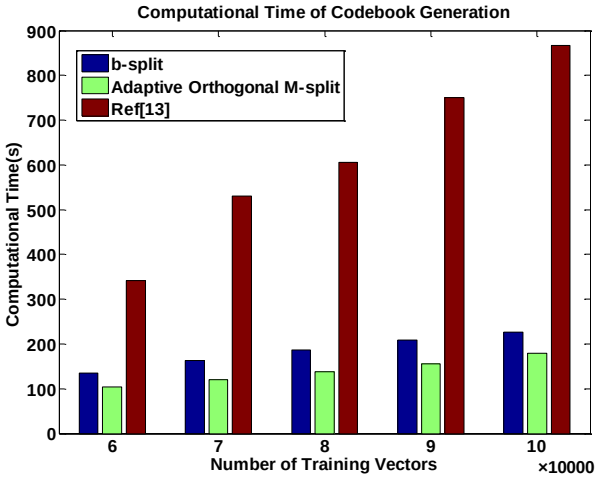

Figure 2. Comparison of Computational Time for Codebook (Size 512) Generation

Figure 3, describes the ASD with split times, where 100,000 vectors were used for training a codebook in size of 512. Although the split times of adaptive orthogonal M-split LBG algorithm (4 times) is less than b-split LBG algorithm (9 times), the ASD of the final codebook is nearly the same as b-split LBG algorithm.

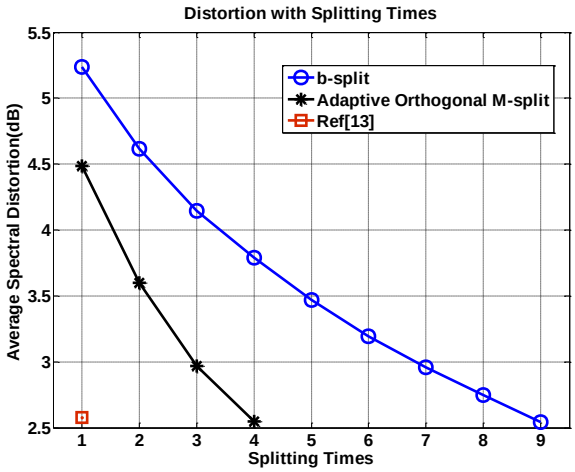

Figure 3. ASD with Split Times of 100,000 Training Vectors for a Codebook Size 512

Figure 4 depicts the ASD of the orthogonal M-split LBG algorithm with training vectors from 600 to 100,000 for codebook in size of 512. For training vectors of 100,000 frames, the ASD of adaptive orthogonal M-split LBG algorithm (2.55) is nearly the same as b-split LBG algorithm (2.54) and a little better than the method in Ref. [13] (2.57).

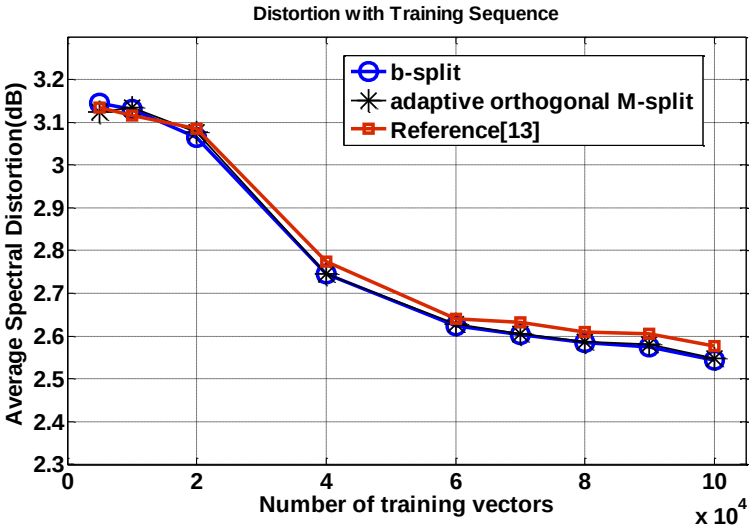

Figure 4. ASD with Training Vectors for Codebook (Size 512)

Table 2, shows the comparison of time, ASD and MSE for codebook of different size ranging from 64 to 2048. The computational time is reduced by 18% ~ 45% with nearly equal ASD and MSE, as compared with the b-split LBG algorithm. The orthogonal M-split LBG algorithms can reduce computational time with nearly equal VQ performance, even if the size of codebook increases. Time reduction for generating the codebooks in size of 64, 128, 256, 512, 1024, 2048 are 24.3%, 45%, 24.7%, 18%, 20.5% and 26.2% respectively compare with b-split LBG algorithm. The main reason is that adaptive orthogonal M-split method decrease split initialization times by splitting more code words with adaptive split coefficient vectors.

Table 2. Comparison of the Algorithms with Respect to ASD, MSE and TIME

Codebook Size	64			128			256		
Algorithm	b-split	M-split	Ref.[13]	b-split	M-split	Ref.[13]	b-split	M-split	Ref.[13]
Time/second	29.35	22.19	664.99	55.02	30.26	670.13	110.39	83.04	718.04
Avg SD/dB	3.19	3.20	3.20	2.95	3.01	2.98	2.75	2.74	2.76
MSE	0.045	0.045	0.045	0.038	0.039	0.039	0.032	0.032	0.033
Codebook Size	512			1024			2048		
Algorithm	b-split	M-split	Ref.[13]	b-split	M-split	Ref.[13]	b-split	M-split	Ref.[13]
Time/second	228.51	185.94	1073.2	453.34	360.36	2735.5	1098.8	810.16	14875
Avg SD/dB	2.54	2.55	2.58	2.36	2.36	2.43	2.18	2.19	2.27
MSE	0.027	0.027	0.028	0.023	0.023	0.024	0.019	0.019	0.021

6. Conclusion

An adaptive orthogonal M-split method is proposed to improve the computational efficiency of split LBG algorithm, which splits one code word into 2, 4 or 5 new code words with adaptive split coefficient vectors in codebook initialization, aiming at reducing the split times and minimizing the number of iterations. The computational time is reduced by 18% ~ 45% while maintaining almost equal VQ performance. The proposed algorithm is quite general and can be applied to the multimedia compression and pattern recognition, such as image compression, low bit rate speech coding, speech recognition, *etc.* Future research will focus on extending the number of split code words M (*eg.* $M = 6, 7, 8, \cdots$) in order to get better performance.

Acknowledgments

This work was supported by the National Nature Science Foundation of China (61301300) and China Postdoctoral Science Foundation (2013M531850).

References

[1] H. B. Kekre and T. K. Sarode, "Bi-Level Vector Quantization Method for Codebook Generation", in Emerging Trends in Engineering and Technology (ICETET), 2009 2nd International Conference on, **(2009)** December, pp. 16-18.
[2] J. Hamkins and K. Zeger, "Gaussian source coding with spherical codes. Information Theory", IEEE Transactions on, vol. 48, no. 11, **(2002)**, pp. 2980-2989.
[3] Y. Linde, A. Buzo and R. M. Gray, "An Algorithm for Vector Quantizer Design. Communications", IEEE Transactions on, vol. 28, no. 1, **(1980)**, pp. 84-95.
[4] D. B. Chang and R. M. Gray, "An Improvement of the Minimum Distortion Encoding Algorithm for Vector Quantization", Communications, IEEE Transactions on, vol. 33, no. 10, **(1985)**, pp. 1132-1133.
[5] S. H. Huang and S. H. Chen, "Fast encoding algorithm for VQ-based image coding", Electronics Letters, vol. 26, no. 19, **(1990)**, pp. 1618-1619.
[6] S. W. Ra and J. K. Kim, "A fast mean-distance-ordered partial codebook search algorithm for image vector quantization", Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on, vol. 40, no. 9, **(1993)**, pp. 576-579.
[7] Y. Liang, J. Yang and Y. Li, "One Effective Method to Design LBG Initial Codebook", in Intelligent Computation Technology and Automation (ICICTA), 2011 International Conference on, **(2011)**, March 28-29.
[8] R. S. Choraś, "Image Processing & Communications Challenges 2", Springer, **(2010)**.
[9] W. H. Equitz, "A new vector quantization clustering algorithm, Acoustics, Speech and Signal Processing", IEEE Transactions on, vol. 37, no. 10, **(1989)**, pp. 1568-1575.
[10] S. So and K. K. Paliwal, "Efficient product code vector quantisation using the switched split vector quantiser", Digital Signal Processing, vol. 17, no. 1, **(2007)**, pp. 138-171.
[11] L. Rabiner and B. H. Juang, "Fundamentals of Speech Recognition", Prentice Hall, **(1993)**.

[12] L. Yanxia, Y. Jiawei, L. Ye and L. Wei, "Most Dispersed and Greedy Tree Growing Algorithm for Designing LBG Initial Codebook", in Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd, **(2011)** May 15-18.
[13] S. X. Chen, F. W. Li, W. L. Zhu and T. Q. Zhang, "Initial Codebook Algorithm of Vector Quantizaton", IEICE TRANSACTIONS on Information and Systems, vol. E91-D, **(2008)**, pp. 2189-2191.
[14] S. X. Chen and F. W. Li, "Initial codebook method of vector quantisation in Hadamard domain", Electronics Letters, vol. 46, no. 9, **(2010)**, pp. 630-631.
[15] S. X. Chen and F. W. Li, "Fast codebook design of vector quantisation", Electronics Letters, vol. 48, no. 15, **(2012)**, pp. 921-922.
[16] Y. Li and H. Cui, "Digital Speech Coding Technologies", Publishing House of Electronics Industry, **(2013)**.
[17] M. Zhanyu, A. Leijon and W. B. Kleijn, "Vector quantization of LSF parameters with a mixture of dirichlet distributions, Audio, Speech, and Language Processing", IEEE Transactions on, vol. 21, no. 9, **(2013)**, pp. 1777-1790.
[18] J. S. Garofolo and L. F. Lamel, Texas Instruments/Massachusetts Institute of Technology(TIMIT), **(1993)**.
[19] A. Li, Z. Yin, T. Wang and Q. Fang, 863 annotated 4 regional accent speech corpus(RASC863), **(2003)**.
[20] S. E. Cheraitia and M. Bouzid, "Reduced complexity system for robust encoding of wideband speech LSF parameters in Electronics", Communications and Photonics Conference (SIECPC), 2013 Saudi International, **(2013)**, April 27-30.

Authors

Weijun He, he received the B.S. degree in electronic and information engineering from Guangdong University of Petrochemical Technology, China, in 2004, the M.S. degree in communication engineering from Chongqing University of post and telecommunications in 2010. He is currently a Ph.D candidate in communication engineering from South China University of Technology. His research interests include speech signal processing, low bit rate speech coding, pattern recognition.

Qianhua He, he received the B.S. degree in physics from Hunan Normal University in 1987, the M.S. degree in medical instrument engineering from Xi'an Jiaotong University in 1990, Ph.D degree in communication engineering from South China University of Technology in 1993. Since 1993, he has been at the School of Electronics & Information Engineering, South China University of Technology. His research interests include speech processing, digital audio forensic, speech coding, multimedia retrieval, audio event analysis and its applications.

Jichen Yang, he received the B.S. degree in electronic and information engineering from Guangdong University of Petrochemical Technology in 2004, received the M.S. degree in system engineering from Guangdong University of Technology in 2007, received the Ph.D degree in information and telecommuncation system from South China University of Technology in 2010. Now he is a post doctoral in South China University of Technology. His currents are speech and audio signal processing.

