# Performance Comparison of Automatic Speaker Recognition using Vector Quantization by LBG KFCG and KMCG

**Dr. H. B. Kekre** hbkekre@yahoo.com
*Senior Professor, MPSTME, SVKM's NMIMS*
*University Mumbai-400056.*

**Ms. Vaishali Kulkarni** vaishalikulkarni6@yahoo.com
*Ph.D Research Scholar, MPSTME,*
*SVKM's NMIMS, Mumbai-400056.*
*Associate Professor, MPSTME,*
*NMIMS University Mumbai-400056.*

**Abstract**

In this paper, three approaches for automatic Speaker Recognition based on Vector quantization are proposed and their performances are compared. Vector Quantization (VQ) is used for feature extraction in both the training and testing phases. Three methods for codebook generation have been used. In the 1st method, codebooks are generated from the speech samples by using the Linde-Buzo-Gray (LBG) algorithm. In the 2nd method, the codebooks are generated using the Kekre's Fast Codebook Generation (KFCG) algorithm and in the 3rd method, the codebooks are generated using the Kekre's Median Codebook Generation (KMCG) algorithm. For speaker identification, the codebook of the test sample is similarly generated and compared with the codebooks of the reference samples stored in the database. The results obtained for the three methods have been compared. The results show that KFCG gives better results than LBG, while KMCG gives the best results.

**Keywords:** Speaker Identification, Vector Quantization (VQ), Code Vectors, Code Book, Euclidean Distance, LBG, KFCG, KMCG

## 1. INTRODUCTION

The goal of speaker recognition is to extract the identity of the person speaking. Speaker recognition technology [1] – [3] makes it possible to use the speaker's voice to control access to restricted services, for example, for giving commands to computer, phone access to banking, database services, shopping or voice mail, and access to secure equipment. Speaker Recognition is the process of automatically recognizing who is speaking on the basis of individual information included in speech signals. It can be divided into Speaker Identification and Speaker Verification [3] – [5]. Speaker identification determines which registered speaker provides a given utterance from amongst a set of known speakers (also known as closed set identification). Speaker verification accepts or rejects the identity claim of a speaker (also known as open set identification).

Speaker Identification task can be further classified into text-dependent or text-independent task [4] – [5]. In the former case, the utterance presented to the system is known beforehand. In the latter case, no assumption about the text being spoken is made, but the system must model the general underlying properties of the speaker's vocal spectrum. In general, text-dependent systems are more reliable and accurate, since both the content and voice can be compared [3], [4].

Speaker Recognition systems have been developed for a wide range of applications [6] – [9]. Still, there are a number of practical limitations because of which widespread deployment of applications and services is not possible.

Vector Quantization (VQ) maps a 'k' dimensional vector space to a finite set C = {C1, C2, C3… CN}.The set C is called codebook consisting of 'N' number of codevectors and each code vector Ci= {ci1, ci2, ci3… cik} is of dimension k. The key to VQ is the good codebook. The method most commonly used to generate codebook is the Linde-Buzo-Gray (LBG) algorithm [10], [11] which is also called as Generalized Lloyd Algorithm (GLA).VQ [10] – [12], [20] is an efficient data compression technique and has been used in various applications involving VQ-based encoding and VQ based recognition. VQ has been very popular in the field of speech recognition. [13] – [19]. We have proposed speaker identification using VQ by LBG algorithm [24] and KFCG algorithm [25]. In this paper we propose speaker identification using VQ by KMCG algorithm. Also comparison of the results obtained by LBG, KFCG and KMCG is shown.

Recognition systems have been developed for a wide range of applications. [15] Although many new techniques were invented and developed, there are still a number of practical limitations because of which widespread deployment of applications and services is not possible. Vector Quantization [1] - [4] is an efficient data compression technique and has been used in various applications involving VQ-based encoding and VQ based recognition. Vector Quantization has been very popular in the field of speech recognition. [5] – [7], [13, 14].

**The recognition Process**

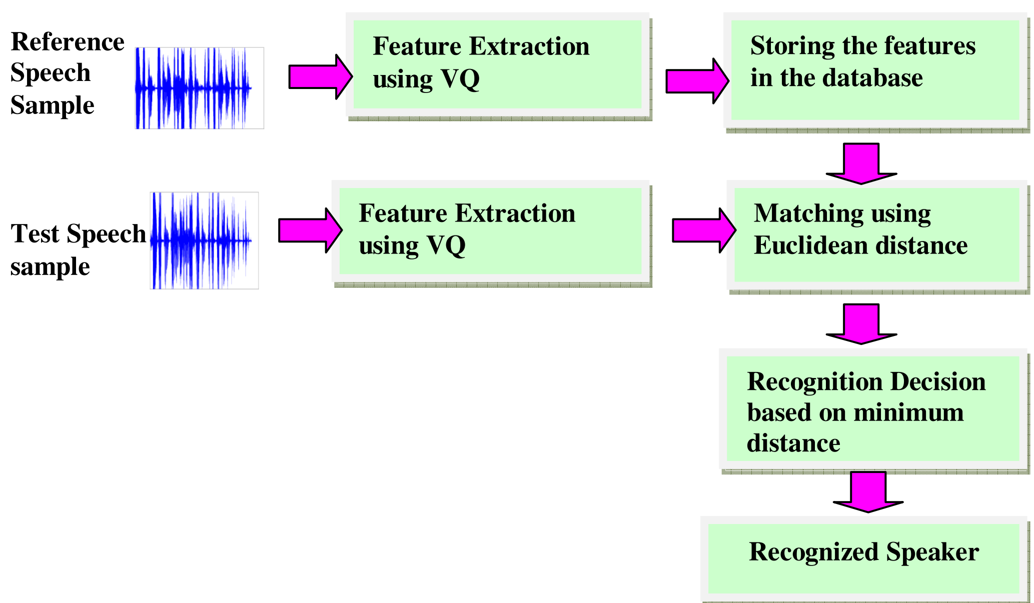



**FIGURE 1:** Speaker identification system

The General scheme for Speaker Identification using vector quantization is shown in Fig. 1. At the training stage, features are extracted from the reference speech samples by using VQ and these are stored as feature vectors in the database. In the testing phase, again features are extracted from the test pattern and compared against the reference templates at the pattern matching stage. Matching is done by comparing the Euclidean Distance. After comparison, the test pattern is labeled to a speaker model at the decision stage. The labeling decision is generally based on the minimum distance criterion.

In the next section we present the two codebook generation approaches. Section 3 describes the three codebook generation algorithms. Section 4 consists of results and conclusions in section 5.

## 2. CODE BOOK GENERATION APPROACH

### a. Without Overlap

The speech signal has amplitude range from -1 to +1. It was first converted into positive values by adding +1 to all the sample values. Then the sample values were converted into a 16 dimensional vector space. (Training vector 'v'). The code books for different size of code vectors were found using the algorithms (LBG, KFCG and KMCG) discussed in the next section.

### b. With Overlap

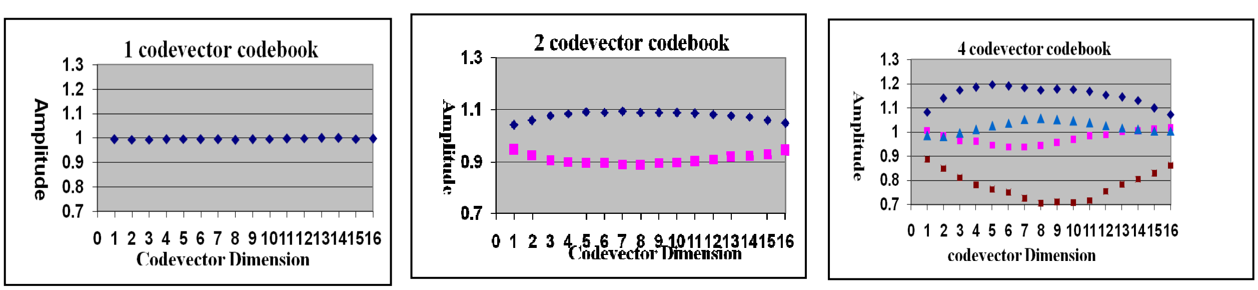
The speech signal was converted into positive range in the same manner as in approach A. The samples were converted into a 16 dimensional vector space by considering an overlap of 4 between the samples of consecutive blocks. E.g. the first vector was from sample 1 to 16, whereas second vector was from 13 to 28 and the third from 25 to 40 and so on. The code books were then generated similarly as in approach A.

## 3. CODEBOOK GENERATION ALGORITHMS

### c. LBG (Linde-Buzo-Gray) Algorithm

For generating the codebooks, the LBG algorithm [11, 12] is used. The LBG algorithm steps are as follows [1, 11]:
Design a 1-vector codebook; this is the centroid of the entire set of training vectors.
Double the size of the codebook by splitting each current codebook yn according to the rule
yn+ = yn(1+ε)
yn+ = yn(1-ε)
where n varies from 1 to the current size of the codebook, and ε is a splitting parameter.
Find the centroids for the split codebook. (i.e., the codebook of twice the size)
Iterate steps 2 and 3 until a codebook of size M is designed.

Figure 2 shows the process of obtaining four codevectors using the LBG algorithm. Figure 2(A) shows the one vector codebook which is the centroid of the entire training set. Figure 2(B) shows the two vector codebook obtained by splitting the training set. Figure 2(C) shows the four vector codebook.




(A) (B) (C)

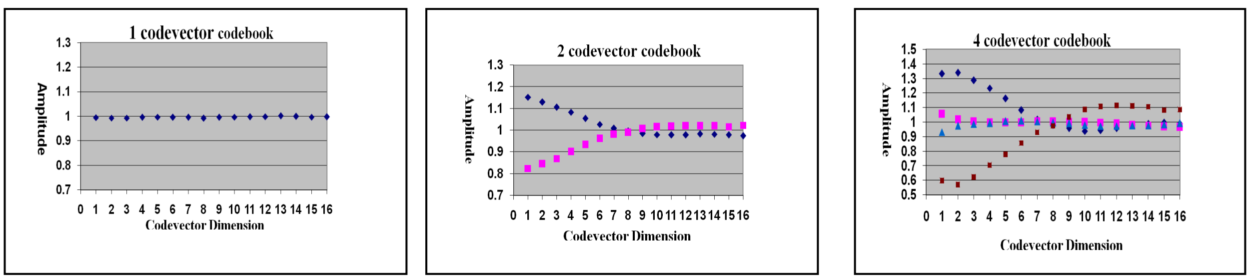
**FIGURE 2:** Generation of four codevectors using LBG algorithm

### d. Kekre's Fast Codebook Generation Algorithm (KFCG)

In this algorithm for generating the codebook the following procedure is used [20] – [23]:

1. Initially we have only one cluster which is the entire set of training vectors. Design a 1-vector codebook; which is the centroid of the cluster.
2. Split the cluster into two by comparing the first element of all the training vectors in the cluster with the first element of the centroid as follows:
3. If vi, 1>c1, 1 then vi,1 is grouped into C1 (cluster 1).
4. Else vi,1 is grouped into C2 (cluster 2).
5. Where v is the training vector and c is the centroid.
6. Find the centroids of C1 and C2 (this is 2-vector codebook). Now split C1 into two clusters by comparing the second element of all the training vectors in C1 with the second element of its centroids explained in step 2 above. Similarly split C2 into two clusters by comparing the second element of all the training vectors in C2 with the second element of its centroid.
7. Now four clusters are formed. Centroids of these four clusters are computed (this is 4-vector codebook). These four clusters are split further by comparing the third element of the training vectors in that cluster with the third element of its centroid as explained in step 2 above.
8. The process is repeated until a codebook of size M is designed.

Figure 3 shows the process of obtaining four codevectors using the KFCG algorithm. Figure 3(A) shows the one vector codebook which is the centroid of the entire training set. Figure 3(B) shows the two vector codebook obtained by splitting the training set by comparing with the first element of all the training vectors in the cluster with the first element of the centroid. Figure 3(C) shows the four vector codebook obtained similarly by splitting the two clusters.




(A) (B) (C)

**FIGURE 3:** Generation of four codevectors using KFCG algorithm

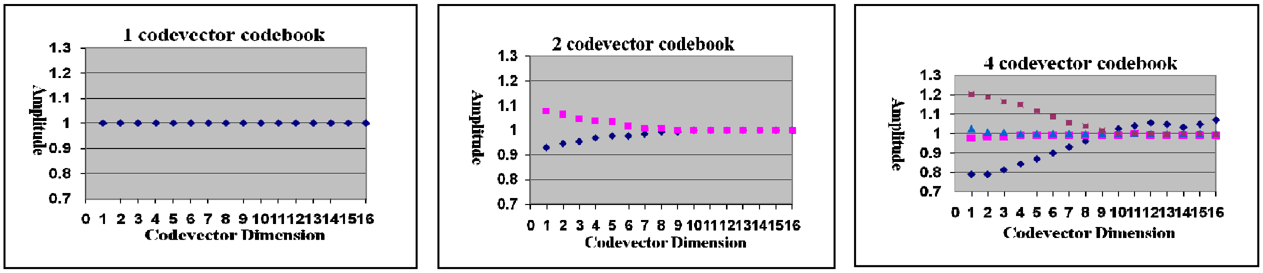
**a.**

## A. Kekre's Median Codebook Generation Algorithm (KMCG)

In this algorithm for generating the codebook the following procedure is used [26]:

1. Initially we have only one cluster which is the entire set of training vectors. Sort the training vectors with respect to the first element of the vector, i.e. with respect to the first column of vector v. Design a 1-vector codebook; which is the median of the cluster.
2. The training vector is then split into two by considering the median. Each of these parts is then again sorted with respect to the second element of the training vectors i.e. with respect to the second column of the vectors. We will obtain two clusters with equal number of training vectors. The median of these clusters is then found and thus we get a two vector codebook.
3. Each of the two vectors is again split into half i.e. four parts. These four parts are further sorted with respect to the third column of the vectors and four clusters are obtained and accordingly four codevectors are obtained.
4. The above process is repeated till we get the codebook of the desired size.

Figure 4 shows the process of obtaining four codevectors using the KMCG algorithm. Figure 4(A) shows the one vector codebook which is the median of the entire training set. Figure 4(B) shows the two vector codebook obtained by dividing the training set into half by sorting with respect to the first element of all the training vectors i.e. with respect to the first column of all the training vectors in the cluster, and then taking the median of these two clusters. Figure 4(C) shows the four vector codebook obtained similarly by diving the two clusters into half by sorting with respect to the second column of the training vectors in the cluster and then taking median of these four clusters.




(A) (B) (C)

**FIGURE 4:** Generation of four codevectors using KMCG algorithm

## 4. RESULTS

**Basics of Speech Signal**

The speech samples used in this work are recorded using Sound Forge 4.5. The sampling frequency is 8000 Hz (8 bit, mono PCM samples). Table 1 shows the database description. The samples are collected from different speakers. Samples are taken from each speaker in two sessions so that training model and testing data can be created. Twelve samples per speaker are taken. The samples recorded in one session are kept in database and the samples recorded in second session are used for testing.

**TABLE 1:** Database Description

| Parameter | Sample characteristics |
|---|---|
| Language | English |
| No. of Speakers | 50 |
| Speech type | Read speech |
| Recording conditions | Normal. (A silent room) |
| Sampling frequency | 8000 Hz |
| Resolution | 8 bps |

The feature vectors of all the reference speech samples are stored in the database in the training phase. In the matching phase, the test sample that is to be identified is taken and similarly processed as in the training phase to form the feature vector. The stored feature vector which gives the minimum Euclidean distance with the input sample feature vector is declared as the speaker identified.

Figure 5 shows the results obtained for text-dependent system by varying the number of feature vectors (code vectors) without overlap for a sample set of 50 speakers. As seen from the figure, for text-dependent samples, maximum accuracy (76%) is achieved with 8 feature vectors for LBG (distortion of 0.005). For LBG (distortion of 0.01) the maximum accuracy obtained is 74%. In both the cases (distortion of 0.005 and 0.01), the accuracy decreases with the increase in the number of feature vectors. For KFCG the results are better and consistent. Accuracy does not drop as the number of feature vectors is increased. The maximum accuracy is 90% for feature vectors size of 64 or more. For KMCG accuracy increases as the number of feature vectors (code vector size) is increased. The maximum accuracy is 96% using 128 feature vectors (codebook size).

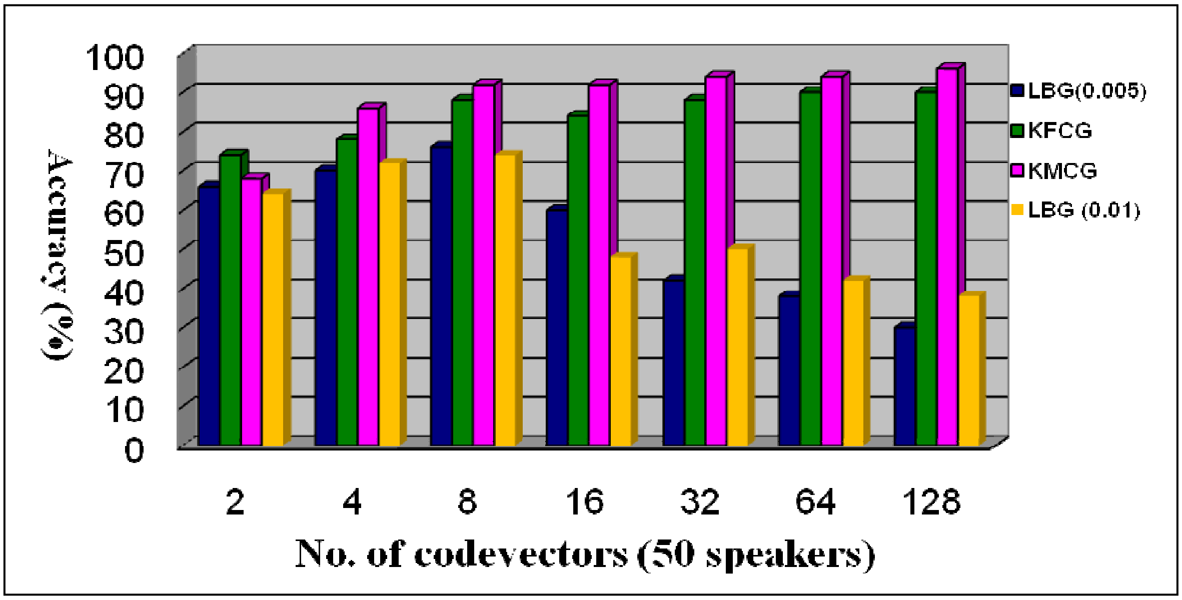



**FIGURE 5:** Performance comparison of LBG (0.005), LBG (0.01), KFCG and KMCG (without overlap)

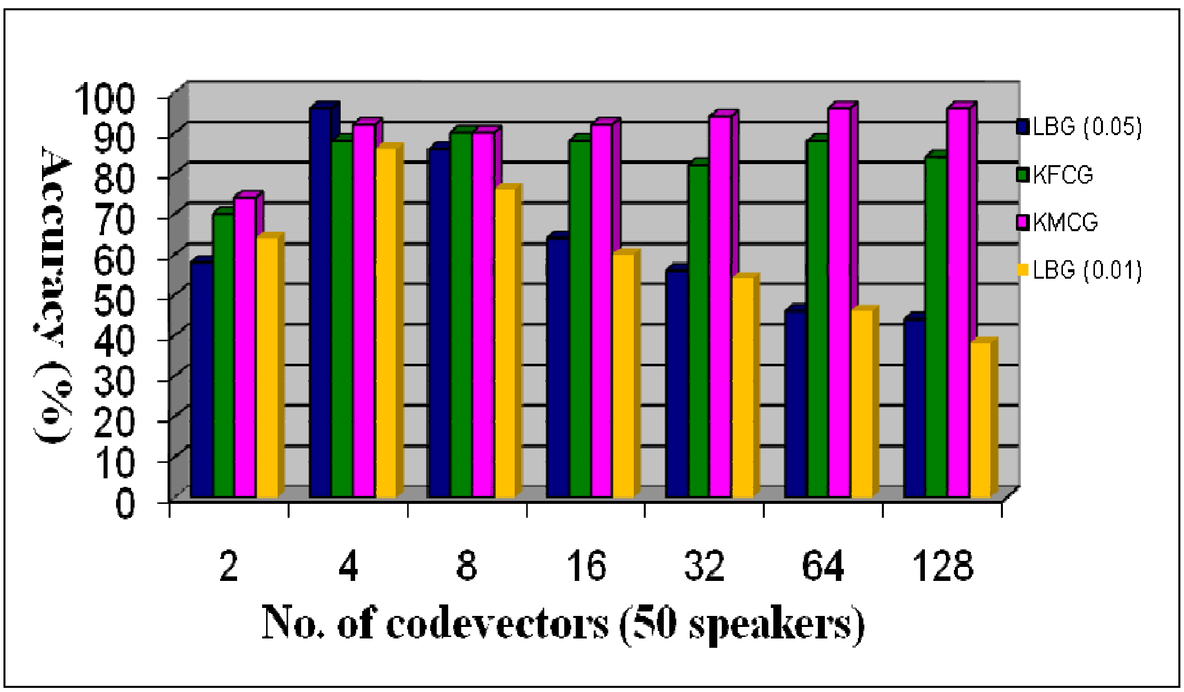



**FIGURE 6:** Performance Comparison of LBG (0.005), LBG (0.01), KFCG and KMCG (with overlap)

Figure 6 shows the results obtained for text-dependent identification by varying the number of features for a sample set of 50 speakers with overlap. Here LBG (distortion of 0.005) gives maximum accuracy of 96% using only four feature vectors (code vector size). LBG (distortion of 0.01) gives maximum accuracy of 86% using four feature vectors. Again the accuracy decreases as the number of feature vectors are increased for both the cases. Here the performance of KFCG is consistent and it gives maximum accuracy of 90% using 8 feature vectors (code vector size), whereas the performance of KMCG increases as the number of feature vectors are increased. The maximum accuracy is 96% using 128 feature vectors (code vector size). As KFCG and KMCG algorithms for codebook generation are based on comparison they are less complex and very fast compared to LBG which needs Euclidean distance calculations. For LBG the number of calculations required for generating the vectors by Euclidean distance comparison for a 16-dimensional vector (16 additions + 16 Multiplications + 16 comparisons) are much more than KFCG and KMCG (16 comparisons). This reduces computational time by a factor ten.

## 5. CONCLUSION

Very simple techniques based on the lossy compression using vector quantization have been introduced. The results show that accuracy decreases as the number of feature vectors are increased with or without overlap for LBG. For KFCG, the results are consistent and also accuracy increases with the increase in the number of feature vectors. KMCG gives the best results for with and without overlap and the accuracy increases as the number of feature vectors are increased. Also KFCG and KMCG algorithms for code book generations are simple and faster as only simple comparisons are required as against Euclidean distance calculations for LBG.

## 6. REFERENCES


1. Lawrence Rabiner, Biing-Hwang Juang and B.Yegnanarayana, “Fundamental of Speech Recognition”, Prentice-Hall, Englewood Cliffs, 2009.

2. S Furui, “50 years of progress in speech and speaker recognition research”, ECTI Transactions on Computer and Information Technology, Vol. 1, No.2, November 2005.

3. D. A. Reynolds, “An overview of automatic speaker recognition technology”, Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP’02), 2002, pp. IV-4072–IV-4075.

4. Joseph P. Campbell, Jr., Senior Member, IEEE, “Speaker Recognition: A Tutorial”, Proceedings of the IEEE, vol. 85, no. 9, pp. 1437-1462, September 1997.

5. F. Bimbot, J.-F. Bonastre, C. Fredouille, G. Gravier, I. Magrin-Chagnolleau, S. Meignier, T. Merlin, J. Ortega-García, D.Petrovska-Delacrétaz, and D. A. Reynolds, “A tutorial on text-independent speaker verification,” EURASIP J. Appl. Signal Process., vol. 2004, no. 1, pp. 430–451, 2004.

6. D. A. Reynolds, “Experimental evaluation of features for robust speaker identification,” IEEE Trans. Speech Audio Process., vol. 2, no. 4, pp. 639–643, Oct. 1994.

7. Tomi Kinnunen, Evgeny Karpov, and Pasi Fr¨anti, “Realtime Speaker Identification”, ICSLP2004.

8. Marco Grimaldi and Fred Cummins, “Speaker Identification using Instantaneous Frequencies”, IEEE Transactions on Audio, Speech, and Language Processing, vol., 16, no. 6, August 2008.

9. Zhong-Xuan, Yuan & Bo-Ling, Xu & Chong-Zhi, Yu. (1999). “Binary Quantization of Feature Vectors for Robust Text-Independent Speaker Identification” in IEEE Transactions on Speech and Audio Processing, Vol. 7, No. 1, January 1999. IEEE, New York, NY, U.S.A.

10. R. M. Gray.: ‘Vector quantization’, IEEE ASSP Marg., pp. 4-29, Apr. 1984.

11. Y. Linde, A. Buzo, and R. M. Gray.: ‘An algorithm for vector quantizer design,” IEEE Trans. Commun.’, vol. COM-28, no. 1, pp. 84-95, 1980.

12. A. Gersho, R.M. Gray.: ‘Vector Quantization and Signal Compression’, Kluwer Academic Publishers, Boston, MA, 1991.

13. F. K. Soong, et. al., “A vector quantization approach to speaker recognition”, At & T Technical Journal, 66, pp. 14-26, 1987.

14. A. E. Rosenberg and F. K. Soong, “Evaluation of a vector quantization talker recognition system in text independent and text dependent models”, Computer Speech and Language 22, pp. 143-157, 1987.

15. Jeng-Shyang Pan, Zhe-Ming Lu, and Sheng-He Sun.: ‘An Efficient Encoding Algorithm for Vector Quantization Based on Subvector Technique’, IEEE Transactions on image processing, vol 12 No. 3 March 2003.

16. F. Soong, E. Rosenberg, B. Juang, and L. Rabiner, "A Vector Quantization Approach to Speaker Recognition", AT&T Technical Journal, vol. 66, March/April 1987, pp. 1426.

17. Md. Rashidul Hasan, Mustafa Jamil, Md. Golam Rabbani Md. Saifur Rahman , “Speaker Identification using Mel Frequency Cepstral Coefficients”, 3rd International Conference on Electrical & Computer Engineering ICECE held at Dhaka, Bangladesh , 28-30 December 2004.

18. Poonam Bansal, Amrita Dev, Shail Bala Jain, “Automatic Speaker Identification using Vector Quantization”, Asian Journal of Information Technology 6 (9): 938-942, 2007.

19. Jyoti Singhai, “Automatic Speaker Recognition :An Approach using DWT based Feature Extraction and Vector Quantization”, IETE Technical Review, vol. 24, No 5, pp 395-402, September-October 2007

20. H. B. Kekre, Tanuja K. Sarode, “Speech Data Compression using Vector Quantization”, WASET International Journal of Computer and Information Science and Engineering (IJCISE), Fall 2008, Volume 2, Number 4, pp.: 251-254, 2008. http://www.waset.org/ijcise.

21. H. B. Kekre, Tanuja K. Sarode, “New Fast Improved Codebook Generation Algorithm for Color Images using Vector Quantization,” International Journal of Engineering and Technology, vol.1, No.1, pp. 67-77, September 2008.

22. H. B. Kekre, Tanuja K. Sarode, “Fast Codebook Generation Algorithm for Color Images using Vector Quantization,” International Journal of Computer Science and Information Technology, Vol. 1, No. 1, pp: 7-12, Jan 2009.

23. H. B. Kekre, Tanuja K. Sarode, “An Efficient Fast Algorithm to Generate Codebook for Vector Quantization,” First International Conference on Emerging Trends in Engineering and Technology, ICETET-2008, held at Raisoni College of Engineering, Nagpur, India, 16-18 July 2008, Avaliable at online IEEE Xplore.

24. H B Kekre, Vaishali Kulkarni, “Speaker Identification by using Vector Quantization”, International Journal of Engineering Science and Technology, May 2010 edition.

25. H B Kekre, Vaishali Kulkarni, “Performance Comparison of Speaker Recognition using Vector Quantization by LBG and KFCG”, International Journal of Computer Applications, vol. 3, July 2010.

26. H B Kekre, Tanuja Sarode, “2-level Vector Quantization Method for Codebook Design using Kekre's Median Codebook Generation Algorithm”,International Journal of Advances in Computational

Sciences and Technology Year:2009,Volume:2,Issue:2.

27. H.B. Kekre, Archana Athawale, Tanuja K. Sarode, Kalpana Sagvekar, “Comparative Performance of Information Hiding in Vector Quantized Codebooks using LBG, KPE, KMCG and KFCG”, International Journal of Computer Science and Information Security, 2010 Vol: 8 Issue: 2,pp 89-95.

28. H B Kekre, Archana Athawale, Tanuja Sarode and Kalpana Sagvekar, “Increased Capacity of Information Hiding using Mixed Codebooks of Vector Quantization Algorithms: LBG, KPE and KMCG, International Journal of Advances in Computational Sciences and Technology, Volume 3 Number 2 (2010) pp. 245–256.