World Applied Sciences Journal 14 (1): 63-70, 2011
ISSN 1818-4952
© IDOSI Publications, 2011

Computational Complexity Reduction of AMR-WB Speech Coding Algorithm Using GA-Optimized Fast Codebook Search Techniques

Mansour Sheikhan and Sahar Garoucy

Department of Electrical Engineering, Islamic Azad University,
South Tehran Branch, Tehran, Iran

Abstract: Most of the complexity in algebraic code excited linear prediction (ACELP), is due to codebook search. So, finding the efficient codebook search method has become necessary. In this paper, four new codebook search methods are proposed. The first method is based on removing ineffective components (RIC). In this method by investigation the behavior of components related to codebook search, effective components are determined and then ineffective components are removed from the search process. The second codebook search method is based on the magnitude behavior of components (MBC) which considers the correlation between target signal and impulse response to select the positions corresponding to larger amplitudes. The third method is the hybrid combination of MBC and RIC. The last proposed method is a hybrid combination of reordering search sequence (RSS) method and RIC. To improve the efficiency of third and fourth methods genetic algorithm (GA) is used. The methods are used in adaptive multi-rate wideband (AMR-WB) speech coder in 12.65 kbps mode. Experimental results show that the proposed methods, as compared to traditional one, improve the performance of codebook search by reducing computational load without significant degradation in mean opinion score (MOS) and signal to noise ratio (SNR) quality metrics.

Key words: Codebook search · Speech coding · AMR-WB · Genetic algorithm

INTRODUCTION

The code excited linear prediction (CELP) structure is a widely used model in most of speech coders [1]. In the CELP model, the codebook search is a selection of the optimal excitation vector using a closed-loop mechanism according to the analysis by synthesis (AbS) method [2] to find the vector with minimum perceptually-weighted distortion. In the family of CELP coders, the algebraic code excited linear prediction (ACELP) which is a CELP coder with the fixed algebraic codebook structure [3] is widely used in digital speech communication. ACELP is recommended as conjugate structure ACELP (CS-ACELP) in ITU-T G.729 [4] and in ITU-T G.723.1 [5], adaptive multi-rate (AMR) [6], adaptive multi-rate wideband (AMR-WB) [7, 8], extended adaptive multi-rate wideband (AMR-WB+) [9] and variable-rate multimode wideband (VMR-WB) [10] speech coding algorithms due to its high performance.

Most of the complexity in ACELP structure comes from the codebook search. Each codebook vector in the ACELP coder consists of some pluses that only a number of them are nonzero. It is necessary to send the position and sign of nonzero pulses to the decoder. Because of the huge number of candidate codevectors, full search is impossible. Many methods have been proposed to improve the codebook search so far [11-27]. These methods are used to find the best codevector and reduce the complexity in the field of codebook search. For example, to improve the speed of codebook search, the focused search that is concentrated on a portion of the algebraic codebook has been performed in [11]. Also in [15], four methods are investigated to improve the codebook search: a) depth first tree search (DFTS) which is a sub-optimal solution to search the codebook, b) a pruning tree method in which the sub-trees, that are less possible to be selected, are pruned, c) maximum-take-precedence (MTP) in which all the positions of each track is divided to several regions and the search process is performed only for some regions, d) reordering search sequence (RSS) method in which the sequence of

Corresponding Author: Dr. Mansour Sheikhan, Department of Electrical Engineering, Islamic Azad University,
South Tehran Branch, Tehran, Iran. P.O. Box 11365/4435, Post-Graduate Center,
South Tehran Branch, Islamic Azad University, Iran. Tel: +98 21 88605565.
Fax: +98 21 33717140.

Table 1: Structure of algebraic codebook in AMR-WB-12.65 kbps mode [8]

Track	Pulses	Positions in codebook
Track_0	i_0, i_4	4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44, 48, 52, 56, 60
Track_1	i_1, i_5	1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 53, 57, 61
Track_2	i_2, i_6	2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54, 58, 62
Track_3	i_3, i_7	3, 7, 11, 15, 19, 23, 27, 31, 35, 39, 43, 47, 51, 55, 59, 63

codebook search is reordered according to mean square weighted error. In [20], the maximum-take-precedence ACELP (MTP-ACELP) search method has been proposed in which by predicting the location of pulses, the computational complexity is reduced. In order to reduce the codebook search time, a multistage technique for CELP coder has been proposed in [23], which is called Trellis search.

In this paper, four new methods for searching the codebook based on removing ineffective components (RIC), magnitude behavior of components (MBC), the hybrid combination of RIC and MBC (RIC+MBC) modified by genetic algorithm (GA) are proposed. In addition, as the fourth proposed method, we modify the RSS technique to improve its performance. In this way, a hybrid combination of RSS and RIC (RSS+RIC) is proposed and to determine the optimum parameters of RSS method, genetic algorithm (GA) is used.

This paper is organized as follows. Section 2 gives a brief overview of codebook search structure in ACELP. In Section 3, the proposed codebook search methods (RIC, MBC, RIC+MBC and RSS+RIC modified by GA) are introduced. The experimental results are discussed in Section 4. Finally, conclusions are provided in Section 5.

ACELP Codebook Search: In the ACELP coder, algebraic codebook is divided into T tracks and each track is consisted of P nonzero pulses. As an application of ACELP coding, in this paper we focus on AMR-WB (in 12.65 kbps mode) [8]. Thus, the typical excitation vector is denoted by P nonzero pulses with amplitude +1 and -1. In 12.65 kbps mode of AMR-WB coder, size of the codebook is 64 samples and contains 8 nonzero pulses (it means that T=4 and P=8 ($P_{0...,}P_7$)). There are two nonzero pulses in each track. The structure of algebraic codebook in AMR-WB is summarized in Table 1 [8].

The aim of algebraic codebook search is to find the codevector C_k which minimizes the square error E_k between the weighted target speech x and the reconstructed speech $\hat{x}$ as follows [8]:

$$E_k = \|x-\hat{x}\|^2 = \|x - g_c HC_k\|^2 \tag{1}$$

where k is the algebraic codevector index, g_c denotes codebook gain and C_k is k^th codevector in subframe with size of 64 samples. Also, H is a lower triangular Toeplitz matrix (64×64). According to ITU-T G.722.2 recommendation, minimizing (1) is equivalent to maximizing the following equation [8]:

$$Q_k = \frac{(d^t c_k)^2}{c_k^t \varphi\, c_k} \tag{2}$$

where d is the inverse filtered target signal and ϕ is the correlation matrix of the impulse response given by the following relations:

$$d(n) = H^t x_2 = \sum_{i=n}^{63} x_2(i)h(i-n) \quad ;n=0,...,63 \tag{3}$$

$$\varphi(i,j) = H^t H = \sum_{n=j}^{63} h(n-i)h(n-i), \;;\begin{cases} i=0,...,63 \\ j=i,...,63 \end{cases} \tag{4}$$

where $x_2(n)$ in (3) is achieved by subtracting the contribution of adaptive codebook from $x(n)$.

$$x_2(n) = x(n) - g_p y(n)\,;\text{n} = 0,....63 \tag{5}$$

Indeed, $x_2(n)$ is the contribution of fixed codebook. $y(n)$ and g_p are the filtered adaptive codebook vector and unquantized adaptive codebook gain, respectively.

Before starting the search process, $d(n)$ and ϕ (i,j) should be calculated [8]. The detailed explanation of codebook search in AMR-WB at 12.65 kbps rate is as follows:

In each time, two pulses which are always corresponding to consecutive tracks are searched. These pulses are in T_0-T_1, T_1-T_2,T_2-T_3 or T_3-T_0 in which T stands for track. In AMR-WB, the codebook search is based on tree search algorithm with 4 levels. At the first level, pulse p_0 is assigned to T_0 and p_1 to T_1. At this level, no search is performed and two pulse positions are set to the position of maximum value of $b(n)$ in four tracks. $b(n)$ is given by:

$$b(n) = \sqrt{\frac{E_d}{E_r}}\, r_{LTP}(n) + \alpha d(n) \tag{6}$$

in which

$$E_\text{d} = d^t d \tag{7}$$

$$E_r = r_{LTP}{}^t r_{LTP} \tag{8}$$

and $r_{LTP}(n)$ is the residual signal after long-term prediction. The scaling factor α which controls the amount of dependence of the reference signal on $d(n)$, is set to 1 in 12.65 kbps mode.

Table 2: All combinations of searched nonzero pulses [8]

	Iteration			
Level	1	2	3	4
1	$P_0\_T_0$	$P_0\_T_1$	$P_0\_T_2$	$P_0\_T_3$
	$P_1\_T_1$	$P_1\_T_2$	$P_1\_T_3$	$P_1\_T_0$
2	$P_2\_T_2$	$P_2\_T_3$	$P_2\_T_0$	$P_2\_T_1$
	$P_3\_T_3$	$P_3\_T_0$	$P_3\_T_1$	$P_3\_T_2$
3	$P_4\_T_1$	$P_4\_T_2$	$P_4\_T_3$	$P_4\_T_0$
	$P_5\_T_2$	$P_5\_T_3$	$P_5\_T_0$	$P_5\_T_1$
4	$P_6\_T_3$	$P_6\_T_0$	$P_6\_T_1$	$P_6\_T_2$
	$P_7\_T_0$	$P_7\_T_1$	$P_7\_T_2$	$P_7\_T_3$

At the second level, p_2 is assigned to T_2 and p_3 to T_3. At this level, four positions for pulse p_2 are tested to maximize (2) for all 16 positions of pulse p_3. It is noted that the selection of these 4 tested positions among 16 positions of T_2 is performed based on the maxima of *b(n)* in its track. Indeed, only four positions of *b(n)* that have the maximum values are selected. At the third level, p_4 is assigned to T_1 and p_5 to T_2. Similar to the pervious level, 8 positions for p_4 and 16 positions for p_5 are searched. At the fourth level, p_6 is assigned to T_3 and p_7 to T_0. Similar to the pervious level, 8 positions for p_6 and 16 positions for p_7 are searched. These four mentioned levels should be repeated 4 times until nonzero pulses are assigned to different tracks. Table 2 shows the total search process in 4 levels and iterations.

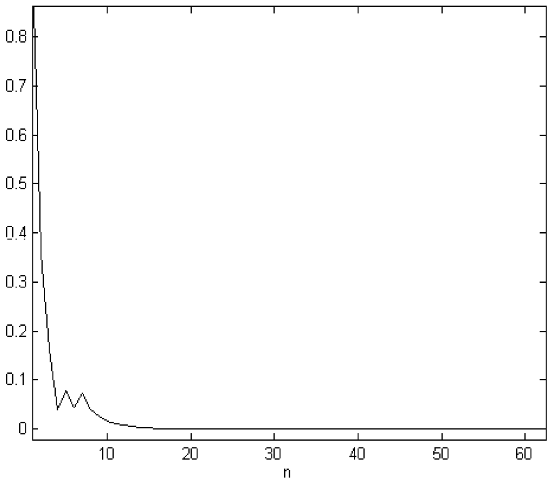

Fig. 1: Amplitude of h(n) averaged over 16000 subframes

Proposed Codebook Search Methods: In most of the speech coding algorithms, it is desirable to decrease the codebook search time and its computational load. In this paper, we focus on the coders with ACELP-based structure. Generally, in the ACELP family speech coding algorithms codebook search takes up about one-fourth of computational load of algorithm [24]. So, finding the methods which decrease this computation and improve the performance of the algorithm is desirable. As mentioned earlier, *d*,ϕ and *h* have the main role in the search process. It is noted that the influences of *d* and ϕ are indirect. Because they are derived from *h*. In all of the codebook search methods which are presented so far, *d*, ϕ and *h* are the main components and all suggestions are based on some modifications on them. For example in [24], to decrease the computational load approximated correlation matrix (ACM) method has been proposed. The experimental results have been shown that in autocorrelation matrix, ϕ, some components are very small. So, it is possible to ignore them. As mentioned earlier, all of the complexity in codebook search is due to search loop and computation of *d(n)* and ϕ. In this paper, we present suggestions in both of these two sources of complexity with the aim of occurring no significant degradation in quality, e.g. in terms of signal to noise ratio (SNR). In the following, removing ineffective components (RIC) technique is proposed as the first method. Then, a new method based on the magnitude behavior of components (MBC) is presented for codebook search. The hybrid method for codebook search based on RIC and MBC is introduced in the next step that is optimized by GA, too. Finally, the hybrid method of RSS and RIC which is also optimized by GA is discussed.

Removing Ineffective Components (RIC) Technique: During the codebook search, some of the computations are not necessary. So by removing them, computational load is reduced while no significant changes are occurred in operation of codebook search module and SNR of the coder. In a typical coder, based on ACELP structure, the codebook search takes up about one-fourth of computational load of the algorithm. This computational load is divided into two parts. One part corresponds to search loops (which takes three- fourths of computational load) and another one is related to computing of *d(n)* and ϕ [24]. According to G.722.2 recommendation [8], *d(n)* and ϕ should be computed before the search process. *d(n)* and ϕ play important roles in the codebook search process. As can be seen in the equations of *d(n)* and ϕ in (3) and (4), the influence of *h(n)* is considerable. To investigate this, it is necessary to first analyze the *h(n)* behavior. For this purpose, the amplitude of *h(n)*, averaged over 16000 subframes, is depicted in Fig. 1.

Table 3: Load of computing d(n) in RIC method for a subframe

	No. of operation	
Removed indices	Addition	Multiplicaton
$40\le n\le 63$	1716	1780
$30\le n\le 63$	1412	1485
$20\le n\le 63$	1026	1090
$15\le n\le 63$	791	855
$10\le n\le 63$	531	595

Table 4: Load of computing ϕ in RIC method for a subframe

	No. of operation	
Removed indices	Addition	Multiplicaton
$40\le n\le 63$	5769	5800
$30\le n\le 63$	4327	4350
$20\le n\le 63$	2884	2900
$15\le n\le 63$	2163	2175
$10\le n\le 63$	1442	1450

Table 5: Changes in SNR when removing some indices of d(n) and ϕ

Removed indices of d(n)	Removed indices of ϕ	ΔSNR (dB)
$40\le n\le 63$	No removing	0
$30\le n\le 63$	No removing	0
$20\le n\le 63$	No removing	0
$15\le n\le 63$	No removing	-0.0032
No removing	$40\le n\le 63$	-0.0028
No removing	$30\le n\le 63$	-0.0023
No removing	$20\le n\le 63$	-0.0116

As shown in Fig. 1, the average amplitude of *h(n)* becomes very small for indices greater than 10. So, in equations of *d(n)* and ϕ, *h(n)* with *n* greater than 10 can be removed. We call this technique removing ineffective components (RIC). In Table 3 and Table 4 the numbers of addition and multiplication operations that are needed to compute *d(n)* and ϕ after removing ineffective components are listed. It is noted that in the algorithm of G.722.2 recommendation, the numbers of addition and multiplication operations in computing *d(n)* are 2016 and 2080, respectively. Also, the numbers of addition and multiplication operations in computing ϕ are 9231 and 9280, respectively. It should be noted that these operations are for a subframe of input signal and each frame of input speech is divided into four subframes. It is noted that when the removed indices of *d(n)* are $10\le n\le$ 63, the SNR can be calculated using the following equation:

$$SNR = 10\log(\frac{\sum_{n} x^2(n)}{\sum_{n} (x(n)-y(n))^2}) \qquad (9)$$

where x(n) is the input signal to system and y(n) is the output signal.

It should be noted that the SNR does not change significantly in spite of removing the mentioned indices (Table 5). So, the quality does not degrade significantly as compared to traditional codebook search method recommended in ITU-T G.722.2 with SNR=27.78 dB. However, when the removed indices of ϕ are $20\le n\le$ 63, the SNR degrades. So, as a practical solution, the numbers of addition and multiplication operations in computing of both *d(n)* and ϕ are reported in Table 6, when the removed indices are $10\le n\le$ 63 and $25\le n\le$ 63, respectively.

Codebook Search Based on Magnitude Behavior of *d*(*n*): In this section, we introduce a new method to search the codebook based on magnitude behavior of components. The aim of this method is to find candidate pulses which have significant magnitudes. The main idea is to focus on *d(n)* behavior. Before the search procedure, the *m* larger pulses are picked up in *d(n)* sequence. Then, codebook search is only performed for the position of selected pulses, because the maximum of *d(n)* is related to maximum of excitation pluses. The advantage of this method is its prevention from unnecessary search processes. To find the mentioned *m* pulses, let's define the following equation:

Table 6: Load of computing *d*(*n*) and ϕ in RIC method as a practical solution for a subframe

Removed indices	No. of operations		Percentage of reduction in number of operations					
	Addition	Multiplication	Addition in computing of d(n)	Multiplication in computing of d(n)	Addition in computing of ϕ	Multiplication in computing of ϕ	Weighted average in computing d(n) and ϕ	SNR (dB)
d : $10\le n\le 63$ φ : $25\le n\le 63$	4136	4220	73.66	71.39	60.95	60.94	63.04	27.77

$$\beta = \frac{1}{\alpha}\left(\min[|d(n)|]\right) \;;n = 0,...,63 \qquad (10)$$

where αϵ[0.1]. The aim of defining this equation is to find the positions of *d(n)* which their amplitudes are larger than $\beta(\frac{1}{\alpha}$ of minimum of *d(n)* sequence in each frame). It is noted that *α* is determined by user and by choosing smaller values for *α*, the codebook search process will be accelerated. Our experimental results show that the appropriate range of *α* is [0.03 0.40]. However, in this paper we use genetic algorithm for finding the optimum values of *α* to reduce the computational complexity and achieving acceptable SNR. In this way, we define the following equation, as a cost function, to consider simultaneously both the speech quality and complexity reduction:

$$C = \varepsilon_1(\frac{SNR}{30}) + \varepsilon_2\frac{n_1}{L} \qquad (11)$$

where ε_1 is a constant in range [0 1] and ε_2 is equal to (1-ε_1). $n_1$is the number of *m* selected positions. *L* is equal to 64 (total positions). If α=1, then all of the 64 positions are selected and the proposed method performs similar to the standard. The determination of α is a trade-off between computational load reduction and speech quality. Selecting low values of α, results in computational load reduction and also quality degradation. In our simulations, we set α to 0.0909 to achieve higher quality and moderate reduction in complexity. The advantage of our method is to select *m* pulse positions adaptively. It means that for example when α=0.0909, 30 positions are remained in frame 1 and only 20 positions are remained in frame 2. In this way, the minimum of *d(n)* is determined for each frame and then based on the value of α, some of the positions are omitted.

The simulation and experimental results for 600 random frames of speech are reported in Table 7 with α=0.3. The SNR value is 26.25 dB with α=0.0909.

RIC+MBC Method: As mentioned earlier, most of the complexity in codebook search is due to computation of *d(n)* and *ϕ* and codebook search loops. The RIC method introduced in section 3.1, reduces the computational load of *d(n)* and *ϕ*. Also codebook search based on MBC, introduced in Section 3.2, reduces the complexity in codebook search loops. Thus, the hybrid structure of both methods can be effective for this purpose. The simulation results of this hybrid method for codebook search based on RIC+MBC method are reported in Table 8. The SNR value is 26.14 dB with α=0.0909.

In this method, for more complexity reduction in computation of *ϕ* matrix, only the components corresponding to the selected positions in *d(n)* are considered. It means that when the selected positions are determined in RIC method, only the components in *ϕ* matrix that include these positions, are computed. For example, if we consider position 20, only $\varphi(1, 20)$, $\varphi(2, 20)$,....$\varphi(63, 20)$ are computed. It should be noted that ignoring of the residual components has no influence on the result of search codebook procedure.

Table 7: Performance of MBC codebook search method

α	C	Percentage of search loop reduction	Mean of n_1 (per subframe)
0.2000	0.9096	7.43	59
0.1667	0.8989	8.98	58
0.1429	0.8889	10.50	57
0.1000	0.8562	15.19	54
0.0909	0.8451	16.77	53
0.0667	0.8035	22.67	49

Table 8: Performance of RIC+MBC codebook search method

α	C	Percentage of search loop reduction	Mean of n_1 (per subframe)
0.2000	0.9087	7.42	59
0.1667	0.8977	9.04	58
0.1429	0.8875	10.65	57
0.1000	0.8538	15.47	54
0.0909	0.8417	17.10	53
0.0667	0.7998	23.20	49

Table 9: Characteristics of GA used in RIC+MBC codebook search method

Parameter/Function	Value/Type
Population size	20
Selection function	Stochastic uniform
Crossover function	Scattered
Mutation function	Gaussian
Scaling function	Rank
Crossover fraction	0.8
Elite count	2

Table 10: Performance of hybrid RIC+MBC method modified by GA

				Percentage of search	Percentage of reduction in number	Mean of n_l
α	ε_1	SNR (dB)	C	loop reduction	of operations in computing of d and ϕ	(per subframe)
0.0428	0.3	27.29	0.7395	33.34	66.73	42

As mentioned earlier, before implementing codebook search based on MBC method, the value of α should be determined by user. Selection of this parameter based on try-and-error is too difficult and time consuming. So, it is suggested that the value of α is determined by genetic algorithm (GA). GA is a particular class of evolutionary algorithms that uses techniques inspired by evolutionary biology such as inheritance, mutation and recombination [28]. GA can be used as an optimization tool to determine the optimal or sub-optimal parameter of α. In GA, after randomizing population, the fitness function F for each individual in the population is evaluated. Then, the first two individuals with the highest fitness value are selected and used in the next generation. To reproduce the individuals in the next generation, crossover and mutation are performed which are GA operations on the remaining individuals (except for two pervious individuals with the highest value) in the current generation. The value and type of GA parameters and functions, used in our simulation, are listed in Table 9.

The simulation result of this hybrid method when using GA to determine optimum value of α is reported in Table 10.

As seen in Table 10, selection the proper value for α results to SNR improvement and also reduction of selected positions per subframe. It must to be noticed that in the procedure that we use in optimization, a threshold is defined for SNR. It means that if GA algorithm in each generation results to a SNR value which is lower than the pre-defined threshold, that generation is removed. This threshold is set to 17 dB in our simulation.

From the simulation results, the superiority of the modified method by GA is obvious in comparison to the non-modified method, by reducing 33.34% of search loops and also 66.73% reduction in the number of operations in computing $d(n)$ and ϕ.

RSS+RIC Method Modified by GA: One of the proposed methods for codebook search in ACELP algorithm is reordering search sequence (RSS) [18]. In this method, the sequence of codebook search is reordered based on minimizing the mean squared weighted error. For each position, the existence probability of pulse is computed. Then, the search process is only carried out for the position with existence probability higher than a threshold Th and the remaining positions are neglected. According to [18], the probability for each position is calculated using the following relation:

$$P = \frac{E_{ki}}{C_{ki}^2} = \frac{s_i d(k_i)}{\phi(k_i, k_i)} \tag{12}$$

where s_i and k_i (k_i [0,…,63]) are the sign and the position of pulses, respectively. In [18], no exact value has been proposed for threshold and how the parameters should be found. In this work, we add two improved steps to the original RSS. At the first step, RSS method is combined with RIC. At the second step, the threshold Th is found by GA. The GA characteristics are assumed as the listed values/types in Table 9. The simulation results of this modified hybrid method are listed in Table 11.

Performance Comparison: In this section, for codebook search the performances of proposed methods are compared. In this work, AMR-WB coder in 12.65 kbps

Table 11: Performance of hybrid RSS+RIC method modified by GA

Th	ε_1	SNR (dB)	C	Percentage of search loop reduction	Percentage of reduction in number of operations in computing of *d* and ϕ	Mean of n_1 (per subframe)
0.0419	0.3	27.71	0.7080	38.43	66.73	39

Table 12: Performance comparison between G.722.2 and proposed codebook search methods

Codebook search method	SNR (dB)	MOS	Approximate reduction of codebook search operations per subframe as compared to G.722.2 (%)
Recommended in G.722.2	27.78	4.33	NA[a]
RIC	27.77	4.33	15.8[b]
RIC+MBC	26.25	4.25	12.6
RIC+MBC modified by GA	27.29	4.30	41.7
RSS+RIC modified by GA	27.71	4.33	45.5

a: Not-Applicable

b: Weighted average reduction in number of operations is reported.

mode is implemented. The simulation of encoder and decoder are performed by using MATLAB software. The database which is used consists of 1,048,576 vectors from 51 different speakers (twenty five men and twenty six women). The speech database which have been used in this work is a part of FARSDAT corpus. FARSDAT is continuous speech Farsi corpus including 6000 utterances from 300 speakers with various accents [29]. The performance of codebook search of original ITU-T G.722.2 algorithm and its comparison to proposed methods is reported in Table 12. It should be noted that approximate reduction of codebook search operations is calculated based on the results reported in Tables 6, 7, 10 and 11 and also the fact that three-fourths of codebook search computational load corresponds to search loops and one-fourth is related to computing *d(n)* and ϕ.

CONCLUSIONS

Most of the computational complexity in ACELP coding, is due to codebook search. In this paper, four new methods have been proposed to reduce computational complexity. Removing ineffective components (RIC) in codebook search process is the main idea in the first proposed method. Based on the magnitude behavior of components (MBC), the second method has been proposed. The third method is the combination of two previous proposed methods (RIC+MBC). The performance of this method has been modified by using genetic algorithm (GA). The last proposed method is a hybrid combination of reordering search sequence (RSS) method and RIC. To improve the efficiency of this method GA has been used, as well. All these methods have been applied to adaptive multi-rate wideband (AMR-WB) speech coder, recommended by ITUT-T G.722.2, in 12.65 kbps mode. Experimental results have shown that the proposed methods, especially modified by GA "RIC+MBC" and "RSS+RIC" hybrid methods, improve the performance of codebook search by considerable reduction in computational complexity without significant degradation in MOS and SNR as compared to traditional codebook search implementation.

REFERENCES

1. Kondoz, A.M., 2007. Digital Speech Coding for Low Bit Rate Communication Systems, Wiley, 2nd Edition.
2. Atal, B.S. and R. Remde, 1982. A New Model of LPC Excited for Producing Natural-Sounding Speech at Low Bit Rates. Proceedings of the International Conference on Acoustics, Speech and Signal Processing, pp: 614-617.
3. Adoul, J., P. Mabilleau, M. Delprat and S. Morisette, 1987. Fast CELP Coding Based on Algebraic Codes. Proceedings of the International Conference on Acoustics, Speech and Signal Processing, pp: 1957-1960.
4. ITU-T, G., 729 Recommendation, 1996. Coding of Speech at 8kb/s Using Conjugate Structure Algebraic Code Excited Linear Prediction.
5. ITU-T, G., 723 Recommendation, 1996. Dual-Rate Speech Coder for Multimedia Communications Transmitting at 5.3 and 6.3 kb/s.
6. Etsi En, 301 704, 1998. Adaptive Multi-Rate (AMR) Speech Transcoding. Digital Cellular Telecommunications System (Phase 2+), version 7.2.1.

7. Bessette, B., R. Salami, R. Lefebvre, M. Jelinek, J. Rotola-Pukkila, J. Vainio, H. Mikkola and K. Jarvinen, 2002. The Adaptive Multirate Wideband Speech Codec (AMR-WB). IEEE Transactions on Speech and Audio Processing, 10: 620-636.
8. ITU-T, G., 722.2 Recommendation, 2002. Wideband Coding of Speech at Around 16kbit/s Using Adaptive Multi-Rate Wideband (AMR-WB).
9. 3GPP Organizational Partners, Extended Adaptive Multi-Rate wideband (AMR-WB+) Codec, 2005. ARIB STD-T63-26.290 V6.2.0.
10. Jelínek, M. and R. Salami, 2007. Wideband Speech Coding Advances in VMR-WB Standard. IEEE Transactions on Audio, Speech and Language Processing, 15: 1167-1179.
11. Laflamme, C., J.P. Adoul, R. Salami, S. Morissette and P. Mabilleau, 1991. 16 kbps Wideband Speech Coding Technique Based on Algebraic CELP. Proceedings of the International Conference on Acoustics, Speech and Signal Processing, pp: 13-16.
12. Pujalte, S. and A. Moreno, 2001. Wideband ACELP at 16 kbit/s with Multi-Band Excitation. Proceedings of the Eurospeech Conference on Speech Communication and Technology, pp: 2001-2004.
13. Ramirez, M.A. and M. Gerken, 1999. A Multistage Search of Algebraic CELP Codebooks. Proceedings of the International Conference on Acoustics, Speech and Signal Processing, 1: 17-20.
14. Ramirez, M.A. and M. Gerken, 2000. Joint Position and Amplitude Search of Algebraic Multipulses. IEEE Transactions on Speech and Audio Processing, 8: 633-637.
15. Pravin Kumar, R., 2007. High Computational Performance in Code Excited Linear Prediction Speech Model Using Faster Codebook Search Techniques. Proceedings of the IEEE International Conference on Computing: Theory and Applications, pp: 458-462.
16. Wang, M.L. and J.F. Yang, 2004. A Generalized Candidate Scheme of Stochastic Codebook Search for Scalable CELP Coders. IEE Proceedings-Vision, Image and Signal Processing, 151: 443-452.
17. Chen, F.K., J.F. Yang and Y.L. Yan, 2002. Candidate Scheme for Fast ACELP Search. IEE Proceedings-Vision, Image and Signal Processing, 149: 10-16.
18. Ha, N.K., 1999. A Fast Method of Algebraic Codebook Search by Reordering Search Sequence. Proceedings of the International Conference on Acoustics, Speech and Signal Processing, 1: 21-24.
19. Byun, K.J., H.B. Jung, M. Hahn and K.S. Kim, 2002. A Fast ACELP Codebook Search Method. Proceedings of the International Conference on Signal Processing, 1: 422-425.
20. Chen, F.K. and J.F. Yang, 2001. Maximum Take Precedence ACELP: A Low Complexity Search Method. Proceedings of the International Conference on Acoustics, Speech and Signal Processing, 2: 693-696.
21. Byun, K.J., H.B. Jung, M. Hahn and K.S. Kim, 2003. Computationally Efficient Implementation of AMR Speech Coder. Proceedings of the 3rd International Symposium on Image and Signal Processing and Analysis, 1: 528- 531.
22. Lin, J., J.A. Storer and M. Cohn, 1991. On the Complexity of Optimal Tree Pruning for Source Coding. Proceedings of the Data Compression Conference, pp: 63-72.
23. Halimi, M., A. Kaddai and M. Bengherabi, 2003. A New Multistage Search of Algebraic CELP Codebooks Based on Trellis Coding. IEICE Transaction on Information Systems, E86-D: 406-411.
24. Tsai, S.M. and J.F. Yang, 2006. Approximated Correlation Matrix and Pulse Prediction for Fast Algebraic Code Excited LPC Speech Coders. Proceedings of the IEEE International Conference on Multimedia, pp: 181-184.
25. Sheikhan, M., V. Tabataba Vakili and S. Garoucy, 2009. Complexity Reduction of LD-CELP Speech Coding in Prediction of Gain Using Neural Networks. World Applied Sciences J., 7(Special Issue of Computer & IT): 38-44.
26. Sheikhan, M., V. Tabataba Vakili and S. Garoucy, 2009. Codebook Search in LD-CELP Speech Coding Algorithmin Based on Multi-SOM Structure. World Applied Sciences J., 7(Special Issue of Computer & IT): 59-68.
27. Sheikhan, M. and S. Garoucy, 2010. Reducing the Codebook Search Time in G.728 Speech Coder Using Fuzzy ARTMAP Neural Networks. World Applied Sciences J., 8: 1260-1266.
28. Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization and Learning, Addison Wesley.
29. Bijankhan, M., J. Sheikhzadegan, M.R. Roohani, Y. Smareh, C. Lucas and M. Tebiani, 1994. The Speech Database of Farsi Spoken Language. Proceedings of the 5th Australian International Conference on Speech Science and Technol., pp: 826-831.