ISSN 1392–124X (print), ISSN 2335–884X (online) INFORMATION TECHNOLOGY AND CONTROL, 2015, T. 44, Nr. 4

Efficient Statistics-Based Algebraic Codebook Search Algorithms Derived from RCM for an ACELP Speech Coder

Tzu-Hung Lin[1], **Cheng-Yu Yeh**[2,*], **Shaw-Hwa Hwang**[1]

[1] Department of Electrical Engineering, National Taipei University of Technology
1, Sec. 3, Chung-hsiao E. Rd., Taipei 10608, Taiwan, R.O.C.
e-mail: t100319011@ntut.edu.tw, hsf@ntut.edu.tw

[2] Department of Electrical Engineering, National Chin-Yi University of Technology
57, Sec. 2, Zhongshan Rd., Taiping Dist., Taichung 41170, Taiwan, R.O.C.
e-mail: cy.yeh@ncut.edu.tw

crossref http://dx.doi.org/10.5755/j01.itc.44.4.9847

Abstract. Applied to an algebraic codebook search conducted on an algebraic code-excited linear-prediction (ACELP) speech coder, two improved versions of reduced candidate mechanism (RCM), designated as Fixed-G1-RCM and Fixed-2Track-RCM, are presented in this study for further search performance improvement. It is mainly derived from two major research findings in a piece of our prior work. The first finding is that a pulse with a high contribution in a track is more likely to serve as the optimal pulse in the optimal vector pertaining to the track, and the second is that the speech quality can be well maintained at a search accuracy above 50%. In this proposal, the trade off can be tuned between the search accuracy and the search complexity so as to reach a nearly consistent speech quality. With this presented search algorithm implemented on a G.729A speech codec, it is experimentally demonstrated that either Fixed-G1-RCM (N = 3), or Fixed-2Track-RCM (M = 2, N = 3, or M = 3, N = 4) can provide a highly superior search performance relative to a global pulse replacement method (iteration = 2) and an iteration-free pulse replacement method.

Keywords: speech codec; VoIP; algebraic code-excited linear-prediction (ACELP); algebraic codebook search; reduced candidate mechanism (RCM).

1. Introduction

An algebraic code-excited linear-prediction (ACELP) based speech coding technique [1-3] is the type of technique most widely applied to digital speech communication systems, and serves as a mainstream technique adopted in a great number of speech coding standards due to the double advantage of low bit rates and high speech quality. The main coding flow for an ACELP coding technique is to perform a linear predictive coding (LPC) on the input speech signal, and then perform an adaptive codebook (the long term prediction) as well as an algebraic codebook codings on the LPC residual signal.

However, the price paid is a high computational complexity requirement, particularly in an algebraic codebook search. The reason is quite simple that it necessitates a tremendous computational load when conducting a full search over the algebraic codebook to locate the optimal pulses. As suggested in [4], the computational load is dominated by two parts, namely, the load in a search process, and the load during the algorithm initialization phase. The former and the latter, respectively, account for 74.9% and the remaining 25.1% of the entire computational load. Provided that there is a way to reduce the computational load to a great extent, an ACELP based coding technique can be extensively applied to an embedded system on a handheld device. In this way, a high performance embedded system is not seen as required, making electronic devices cost competitive. Moreover, due to a computational load reduction, the aim of energy saving is reached for an extended operation time period.

For this sake, full search scheme is hardly adopted in most prominent speech coding standards. There have been a great number of studies proposed on search load reduction, say, the focus search in G.729 [3] adopted in a vast majority of VoIP systems [5-10],

---

[*] Corresponding author

and the depth-first tree search in G.729A [11], among other approaches. In recent times, a number of studies on this issue [12-20] cover the candidate scheme [12], the least important pulse replacement [13], both the global pulse replacement (GPR) [14] and the iteration-free pulse replacement (IFPR) [15, 16], and the reduced candidate mechanism (RCM) approach [17], i.e. a piece of our prior work. An attempt in [12] is made to decrease the number of the candidate positions, the least significant pulse is replaced in an iterative manner in [13], and GPR and IFPR [14-16] are developed on the basis of [13]. In RCM, individual pulse contribution is evaluated in the associated track and sorted in descending order. Subsequently, a full search is performed on the sorted top N pulses treated as candidates. In this way, the optimal pulse combination is acquired following N^4 searches, that is, a significant reduction in search complexity is achieved.

A further investigation into RCM is made in this work based on the two findings as presented in [17]. Accordingly, two improved versions of RCM are proposed to reach the aim of search performance improvement, but without speech quality degradation. The proposed approaches for search performance improvement are implemented on a G.729A speech codec for performance comparison of this proposal versus existing approaches in the literature in terms of search complexity, speech quality, etc.

This paper is outlined as follows. The coding criterion of an algebraic codebook and various search methods are briefly reviewed in Section 2. Presented in Section 3 are two proposed approaches for the efficient complexity reduction. Experimental results are demonstrated and discussed in Section 4. This work is summarized at the end of this paper.

2. Algebraic codebook search

With the determination of an optimal codevector as the goal of the algebraic codebook search, the codebook in G.729 is configured as tabulated in Table 1, on the basis of which each codevector contains 4 nonzero pulses extracted out of associated track. Each pulse's amplitude can be either +1 or -1.

Table 1. A structured algebraic codebook in G.729

Track	Pulse	Sign		Positions
T_0	i_0	s_0	m_0:	0, 5, 10, 15, 20, 25, 30, 35
T_1	i_1	s_1	m_1:	1, 6, 11, 16, 21, 26, 31, 36
T_2	i_2	s_2	m_2:	2, 7, 12, 17, 22, 27, 32, 37
T_3	i_3	s_3	m_3:	3, 8, 13, 18, 23, 28, 33, 38 4, 9, 14, 19, 24, 29, 34, 39

The optimal codevector $\mathbf{c}_k = \{c_k(n)\}$ is thus found by minimizing the mean squared weighted error between the original and the synthesized speeches [2, 3], defined as

$$\varepsilon_k = \left\| \mathbf{x} - g\mathbf{H}\mathbf{c}_k \right\|^2 \tag{1}$$

where $\mathbf{x}$ denotes the target vector, g a scaling gain factor, and $\mathbf{H}$ a lower triangular convolution matrix. It can be shown that the optimal codevector is the one maximizing the term Q_k:

$$Q_k = \frac{\left(\mathbf{x}^{\mathrm{T}}\mathbf{H}\mathbf{c}_k\right)^2}{\mathbf{c}_k^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{H}\mathbf{c}_k} = \frac{\left(\mathbf{d}\mathbf{c}_k\right)^2}{\mathbf{c}_k^{\mathrm{T}}\mathbf{\Phi}\mathbf{c}_k} \tag{2}$$

where $\mathbf{d} = \mathbf{x}^{\mathrm{T}}\mathbf{H}$, the correlation function, is expressed as

$$d(n) = \sum_{i=n}^{L-1} x(i)h(i-n), \quad 0 \le n \le L-1 \tag{3}$$

where L is the speech subframe size. The correlations of $h(n)$ are contained in the symmetric matrix $\mathbf{\Phi} = \mathbf{H}^{\mathrm{T}}\mathbf{H}$, where the entries are given by

$$\phi(i,j) = \sum_{n=j}^{L-1} h(n-i)h(n-j), \quad 0 \le i \le L-1;\ i \le j \le L-1. \tag{4}$$

It takes a total of 8192 (8*8*8*16) searches, a tremendous computational load, to conduct a full search, i.e. repeated computations and comparisons in (2), for the identification of the optimal codevector. Therefore, a focused search method is adopted in G.729 to reduce the search times to below 1440. However, the number of searches is further reduced to 320, adopting a depth-first tree search method in G.729A. Besides, three existing methods, the GPR, IFPR, and RCM methods, will be discussed in this section.

2.1. The GPR search method

The GPR method stems from the least important pulse replacement algorithm [13]. In order to prevent the termination of the pulse replacement procedure without finding the optimal codevector in the GPR algorithm, except for the only track that contains the least important pulse, all the tracks are searched for a new pulse. That is, the new pulse is sought by replacing each pulse in each track with a new one so that the Q_k associated with a new codevector is maximized. On the ground that the variation in Q_k is always maximized during the replacement procedure, the codevector approaches the optimal solution rapidly as this procedure is repeated. When the value of Q_k once reaches the upper bound, the search procedure is then terminated.

A system flowchart of the GPR method is sketched in Fig. 1. Following an application of the GPR method to G.729A at the first stage, the initial Q_k is evaluated and the initial codevector is yielded with a search. At the second stage, it requires 36 searches to seek the new pulse during the first pulse replacement procedure, and requires an average of 27 during the second. Therefore, the overall search complexity is evaluated as 37+27*(R-1) for $R \ge 1$, where R is the number of iterations of the pulse replacement procedure.

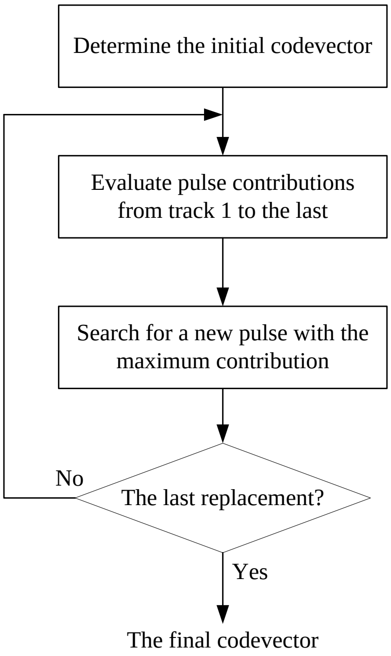

Figure 1. A system flowchart of the global pulse replacement search

2.2. The IFPR search method

In the previously mentioned pulse replacement methods, [13] and [14], the computational load increases with the number of iterations of the replacement procedure. In the IFPR method, new pulses are sought by a number of pulse replacements at a time following pulse contributions evaluated for every track so as to maximize over all combinations a search criterion, which replaces the pulses pertaining to the initial codevector with the most significant pulses for every track.

Presented in Fig. 2 is a system flowchart of the IFPR method. Applying IFPR method to G.729A at the first stage, the initial Q_k is evaluated and the initial codevector is then yielded with a search. At the second stage, a total of 36 searches are performed to measure the pulse contribution so as to sort out the most significant pulses in each track. In order to find the final codevector in the end, it requires a total of 11 searches for all combinations, i.e. the number required from 2 pulses replacement to 4 pulses replacement, to replace the pulses of the initial codevector with the most significant pulses for every track, that is, an overall search complexity of 48.

2.3. The RCM search method

Ahead of a search task, the number of candidate pulses in each track is reduced for the purpose of search complexity reduction. This is done in this work according to the contribution of individual pulses. It is that in each track a pulse sorting is made by the contribution thereof in descending order as the first step, and then the top N pulses are chosen as the candidate pulses for a full search. In this way, the search process needs to be performed for merely N^4 number of times for the optimal pulse combination, and the search complexity is reduced remarkably in particular for low values of N.

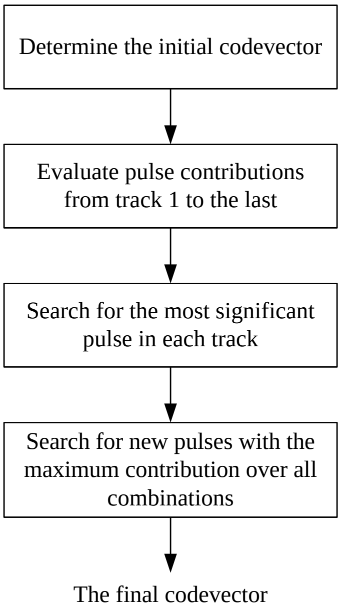

Figure 2. A system flowchart of the iteration-free pulse replacement search

The contribution made by individual pulses is given as (2), that is, a higher value of Q_k reflects a higher contribution. In consideration of merely a single pulse contribution, the number of nonzero pulses in the codevector $\mathbf{c}_k$ of length 40 is reduced to 1 from 4. Therefore, (2) can be simplified into (5), where the numerator of (5) is derived from (2) and (3), and the denominator of (5) is derived from (2) and (4), respectively. Just as in (2), the contribution of the *i*th pulse is reflected by the value of Q_k^i:

$$Q_k^i = \frac{d^2(i)}{\phi(i,i)}, \quad 0 \le i \le L-1. \tag{5}$$

In [17], it is verified that a single pulse with a higher contribution within each track is more likely to be the optimal pulse out of the optimal codevector within the associated track. Thus, the RCM approach is used to reduce the search complexity by the reduction in the number of candidate pulses in each track. This approach is decomposed into two stages as follows. The first is to evaluate individual pulse contribution with (5), which indicates that a higher value of Q_k^i denotes a higher pulse contribution. Subsequently, the top N pulses, $1 \le \mathrm{N} \le 8$, are extracted out of the Q_k^i sorting in each track as the prerequisite of the second stage. Then, in the second stage, it is proven that the pulses combination with the highest value of Q_k, as given in (2), is indeed the optimal solution through a nest-loop search. Finally, this RCM approach is presented in an algorithmic form as follows.

Algorithm 1: The RCM search procedure.

Step 1. Sort the pulses in each track in a descending order by individual pulse contribution evaluated as (5).

Step 2. Specify the value of the parameter *N*, and select the top *N* pulses in each track as the candidate pulses.

Step 3. Search for the optimal pulses over all the combinations of the candidate pulses through a full search by means of (2).

Step 4. Terminate a searching task at the moment the combination of the optimal pulses is acquired.

3. Proposed approaches

The first finding in [17] indicates that a pulse with a high level of contribution in the associated track is more likely to serve as the optimal pulse in the optimal codevector, and the second reveals that the speech quality can be well maintained on a condition that the search accuracy exceeds a threshold, say 50% in [17]. Underlain by such findings, two improved versions of RCM, designated as Fixed-G1-RCM and Fixed-2Track-RCM, are presented to achieve the aim of search performance elevation in the absence of speech quality degradation.

Moreover, a hit probability $p_h(\mathrm{T}_t, n)$ is defined as (6), where $NH(\mathrm{T}_t, n)$ denotes the number of times that the *n*th pulse, in terms of the contribution priority, within track T_t, hits the optimal codevector, *TSF* is the total number of testing subframes, and *NP* is the number of pulses within track T_t:

$$p_h(\mathrm{T}_t, n) = \frac{NH(\mathrm{T}_t, n)}{TSF}, \ 0 \le t \le 3, \quad 1 \le n \le NP. \qquad (6)$$

Subsequently, a cumulative probability $p_c(\mathrm{T}_t, \mathrm{N})$ is defined as (7) to accumulate the previous N number of $p_h(\mathrm{T}_t, n)$, and an accuracy is defined as (8), that is, the probability to successfully locate four intended pulses:

$$p_c(\mathrm{T}_t, \mathrm{N}) = \sum_{n=1}^{\mathrm{N}} p_h(\mathrm{T}_t, n), 0 \le t \le 3, \quad 1 \le \mathrm{N} \le NP \qquad (7)$$

$$p_a(\mathrm{N}) = \prod_{t=0}^{3} p_c(\mathrm{T}_t, \mathrm{N}), \ 1 \le \mathrm{N} \le NP. \qquad (8)$$

Tabulated in Table 2 are the statistics made through (7) and (8) based on a speech database in Chinese language, containing 9,650 syllables out of 100 sentences for a duration over 41 minutes and 495,608 subframes, that is, *TSF*=495,608. Taking the case of *N*=2, the first two pulses in the Q_k^i sorting, as an instance, the hit probabilities, in each search, of such first two pulses against the optimal codevector are 0.8571, 0.8417, 0.8519 and 0.8129, respectively. That is, the adoption of the top two pulses in the Q_k^i sorting in each track causes a hit probability higher than 0.8. Likewise, an adoption of top four pulses leads to a hit probability above 0.9. Besides, the accuracy given by (8) is tabulated in Table 2, where a higher value of accuracy represents a search result closer to that by a full search.

3.1. The Fixed-G1-RCM approach

Based on two findings mentioned above, the first improved version of RCM, referred to as Fixed-G1-RCM approach, is presented in this study. As its name indicates, the top 1 pulse contribution in a global sorting, termed as G1 pulse, is presumed to be one of the four optimal pulses, following which the rest of optimal pulses are located over the remaining 3 tracks through RCM.

Thus, it is an issue of our interest whether there is a linkage between the top 1 pulse contribution and the possibility that such pulse is indeed the one of optimal pulses. Hence, over entire tracks, a hit probability in a global sorting $P_{G,h}(n)$ is defined as (9), where $NH_G(n)$ denotes the number of times that the *n*th pulse, in terms of the contribution priority, hits the optimal codevector, and TSF_G is the total number of testing subframes:

$$P_{G,h}(n) = \frac{NH_G(n)}{TSF_G}, \ 1 \le n \le 40, \qquad (9)$$

Subsequently, a global sorting is conducted by pulse contribution over entire tracks. As tabulated in Table 3, it is seen that there is a 0.8321 hit probability that the No. 1 pulse is indeed the optimal pulse, while the hit probability drops dramatically from 0.8321 to 0.5857 in case the No. 2 pulse acts as the optimal one. A graphic illustration of Table 3 is shown in Fig. 3. The statistics in Table 3 is based on the same speech database mentioned above.

Table 2. Cumulative probability pertaining to each track

N	Cumulative probability of codebook track				Probability of locating 4 intended pulses
	T_0	T_1	T_2	T_3	
1	0.6698	0.6483	0.6630	0.6351	0.1828
2	0.8571	0.8417	0.8519	0.8129	0.4996
3	0.9278	0.9185	0.9249	0.8902	0.7016
4	0.9605	0.9539	0.9588	0.9282	0.8154
5	0.9778	0.9740	0.9769	0.9509	0.8847
6	0.9887	0.9863	0.9879	0.9648	0.9294
7	0.9955	0.9943	0.9952	0.9745	0.9600
8	1	1	1	0.9813	0.9813

The above analysis confirms that the top 1 pulse contribution in a global sorting has a highly hit probability that such pulse is indeed the one of optimal pulses. Thus, the Fixed-G1-RCM approach is presented in this study as an efficient way to further speed up the searching process. In this approach, the G1 pulse is presumed to be one of the four optimal pulses, following which the rest of optimal pulses are located over the remaining 3 tracks through RCM. In this context, the number of searches required is reduced to N^3 for $N \geq 2$. This proposal, as opposed to RCM, is developed in an attempt to considerably reduce the search load to N^3 from N^4 identical value of *N*. Furthermore, since the highest hit probability is as high as 0.8321, the search performance improvement can be made together with well maintained speech quality at a search accuracy of 50% approximately. Here, this search scheme is presented in an algorithmic form as follows, and is illustrated in Fig. 4.

Table 3. Hit probability of each pulse contribution in a global sorting

Global sorting for pulse contribution	Hit probability	Global sorting for pulse contribution	Hit probability
1	0.8321	21	0.0189
2	0.5857	22	0.0161
3	0.4589	23	0.0144
4	0.3621	24	0.0127
5	0.2925	25	0.0116
6	0.2383	26	0.0103
7	0.1954	27	0.0096
8	0.1595	28	0.0084
9	0.1314	29	0.0079
10	0.1085	30	0.0071
11	0.0908	31	0.0063
12	0.0760	32	0.0059
13	0.0635	33	0.0054
14	0.0527	34	0.0048
15	0.0450	35	0.0047
16	0.0385	36	0.0042
17	0.0329	37	0.0040
18	0.0281	38	0.0037
19	0.0245	39	0.0032
20	0.0213	40	0.0030

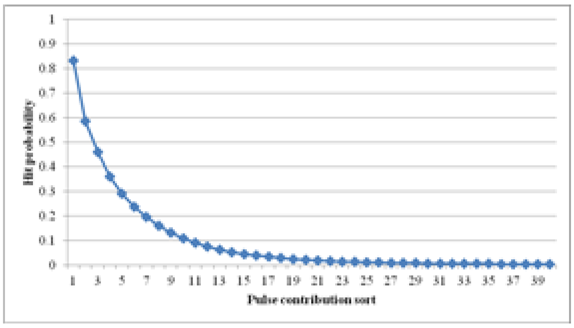

Figure 3. Plot of the hit probability of each pulse contribution in a global sorting

Algorithm 2: The Fixed-G1-RCM search procedure.

Step 1. Individual pulse contribution is evaluated by (5), and a sorting is made by pulse contribution within the associated track.

Step 2. The one with the global maximum pulse contribution, named as *G1*, is located out of all the top 1 pulses among all the tracks.

Step 3. *G1* is presumed to be one of four optimal pulses, and then the value of *N* is determined for the searching task conducted over the remaining three tracks through RCM.

Step 4. A searching task terminates at the moment the combination of the optimal pulses is acquired.

3.2. The Fixed-2Track-RCM approach

The second approach, referred to as the Fixed-2Track-RCM approach, is developed due to the fact that there is a 89% hit probability that a pair of No. 1 pulses in two tracks are indeed the optimal pulses

[17]. For this sake, such two pulses are presumed to be two of the optimal pulses, and then remaining two are located through RCM. In this manner, a high efficiency search algorithm is successfully developed to significantly reduce the search complexity as intended.

Subsequently, the first pulses in respective sortings are further sorted and designated as R_1, R_2, R_3, R_4, where R_1 is, in fact, the G1 pulse in the preceding section. As tabulated in Table 4, the probabilities that pairs serve as the optimal pulses are evaluated and then treated as the priority when conducting a sequence of search tasks. The pairs are sorted by priority in descending order as (R_1, R_2), (R_1, R_3), (R_2, R_3), (R_1, R_4), (R_2, R_4) and (R_3, R_4), and a point worthy of mention is that a higher priority is exhibited in (R_2, R_3) than in (R_1, R_4).

Table 4. Values of pair accuracy for Fixed-2Track-RCM

Pair	Accuracy	Priority
(R_1, R_2)	0.5674	1
(R_1, R_3)	0.4910	2
(R_1, R_4)	0.4080	4
(R_2, R_3)	0.4153	3
(R_2, R_4)	0.3453	5
(R_3, R_4)	0.3014	6

Due to the lack of high search accuracies among various pairings in Table 4, a multiple pairing strategy is adopted to elevate the search accuracy. For instance, the pair (R_1, R_2) is firstly taken for performing RCM to find the highest Q_k and locate the optimal pulse combination. Secondly, the same process is repeated on the pairs (R_1, R_3), (R_1, R_4), and so forth. In this context, there is a maximum of 6 search tasks performed on sorted pairs by the priority thereof.

A further investigation into the search complexity is made as follows. With a single pair, the searching process is done following N^2 searches. Yet, an advantage gained is that the search complexity is a function of M, the number of pairings. Hence, the search number is expressed as (10), where the terms subtracted represent the number of repeated searches:

$$\text{Search number} = \begin{cases} MN^2 - (M-1)N & 1 \le M \le 3 \\ MN^2 - (M-1)N - (M-3)(N-1) & 4 \le M \le 6 \end{cases} \quad (10)$$

Lastly, this proposed Fixed-2Track-RCM is presented in an algorithmic form as follows.

Algorithm 3: The Fixed-2Track-RCM search procedure.

Step 1. Specify the values of the parameters M and N.

Step 2. Pulses in each track are sorted by individual pulse contribution evaluated as (5).

Step 3. The No. 1 pulses in individual tracks are further sorted, and then designated as R_1, R_2, R_3, R_4.

Step 4. Reference Table 4 in the determination of M value, and then perform the search tasks over the remaining two tracks through RCM by means of (2).

Step 5. The search process is repeated until the optimal pulse combination is found.

4. Experimental results

There are three experiments conducted in this work. The first is a search accuracy comparison between the full search and other search approaches. Subsequently, the second is a computational complexity comparison between the preceding search approaches. Lastly, the third is that various approaches are compared with ITU-T P.862 perceptual evaluation of speech quality (PESQ) [21] and ITU-T P.862.1 mean opinion score, listening quality objective (MOS-LQO) [22] as an objective measure of speech quality. The test objects are those selected out of a speech database in Chinese language, containing 9,650 syllables out of 100 sentences for a duration over 41 minutes and 495,608 subframes.

For the brevity of the following discussion, the RCM approach with *N* candidate pulses is abbreviated as RCM-N, $1 \le N \le 8$. For instance, RCM-1 symbolizes the one with merely a candidate pulse extracted out of each track. Similarly, the GPR approach with the number R of repetitions is designated as GPR-R.

Tabulated in Table 5 is the search accuracy analysis between various approaches, that is, the hit probability of individual approach against the optimal pulse identified through a full search. During the search process, the best case is the one to successfully locate 4 intended pulses, the all right case, and the worst is to locate none, the all wrong case. As tabulated in Table 5, taking the all right case as an instance, the accuracies made by G.729A, GPR-2, IFPR and RCM-2 are 68.6438%, 76.1053%, 68.0824% and 50.3579%, respectively, while that by the proposed Fixed-G1-RCM method falls between 17.3353% (*N*=1) and 81.7921% (*N*=8). Likewise, the search accuracy for various combinations of (*M, N*) in Fixed-2Track-RCM is tabulated in Table 6. It is found that a search accuracy above 50% is exhibited in the cases of *M* >= 2 and *N* >= 3.

Listed in Table 7 is the comparison of the search complexity, that is, the number of searches performed and those required in the evaluation of Q_k defined in (2). It is found that G.729A requires 320 searches, GPR-2 64, IFPR 48, RCM-2 16, and the proposed Fixed-G1-RCM method a number somewhere between 1 (*N*=1) and 512 (*N*=8). Accordingly, the

Table 5. Comparison of the search accuracy among various methods

Methods		**Search accuracy for locating various number of intended pulses (%)**				
		1 pulse	**2 pulses**	**3 pulses**	**4 pulses (all right)**	**0 pulse (all wrong)**
G.729A		98.3475	92.1456	80.9918	68.6438	1.6525
GPR	R=1	98.7032	90.7750	76.1053	55.0718	1.2968
	R=2	98.4946	91.8335	80.4779	76.1053	1.5054
	R=3	98.5246	92.1081	81.2227	80.4779	1.4754
	R=4	98.5283	92.1547	81.3393	81.2227	1.4717
IFPR		98.6810	92.4963	80.0048	68.0824	1.3190
RCM	N=1	99.3295	89.0873	55.8720	17.3353	0.6705
	N=2	98.3009	90.8486	73.6532	50.3579	1.6991
	N=3	98.8545	94.0394	83.5763	70.7329	1.1455
	N=4	99.2014	95.9617	89.2187	81.8716	0.7986
	N=5	99.4609	97.3146	92.9551	88.6031	0.5391
	N=6	99.6489	98.2930	95.5634	92.9918	0.3511
	N=7	99.7978	99.0166	97.4591	96.0049	0.2022
	N=8	99.9011	99.5341	98.8319	98.1148	0.0989
Fixed-G1-RCM	N=2	98.2054	89.8500	70.0755	44.0814	1.7946
	N=3	98.2060	91.2376	76.7365	60.2498	1.7940
	N=4	98.2369	92.1541	80.4684	69.0717	1.7631
	N=5	98.2430	92.8193	82.9734	74.3685	1.7570
	N=6	98.2472	93.3058	84.7438	77.8040	1.7528
	N=7	98.2529	93.6545	86.0402	80.1414	1.7471
	N=8	98.2531	93.9055	86.9845	81.7921	1.7469

Table 6. Comparison of the search accuracy for Fixed-2Track-RCM method

Fixed-2Track-RCM		**Search accuracy for locating various number of intended pulses (%)**				
M	**N**	**1 pulse**	**2 pulses**	**3 pulses**	**4 pulses (all right)**	**0 pulse (all wrong)**
1	2	98.7081	89.2092	65.4533	34.6203	1.2919
	3	98.5775	89.3408	68.7967	44.1536	1.4225
	4	98.5309	89.5197	70.5108	49.1522	1.4691
	5	98.5051	89.6640	71.6318	52.0756	1.4949
	6	98.4899	89.7364	72.3509	53.9331	1.5101
2	2	98.5069	90.1602	69.3798	41.6428	1.4931
	3	98.4948	91.0486	74.5141	54.9382	1.5052
	4	98.5010	91.6039	77.1497	61.8798	1.4990
	5	98.4984	91.9600	78.8506	65.8855	1.5016
	6	98.4908	92.1646	79.9733	68.4156	1.5093
3	2	98.3566	90.5589	71.7928	45.6833	1.6434
	3	98.4746	92.2043	78.3912	60.8624	1.5254
	4	98.5095	93.0651	81.7269	68.6418	1.4905
	5	98.5305	93.6099	83.8074	73.1140	1.4695
	6	98.5305	93.9349	85.1879	75.9261	1.4695
4	2	98.3848	90.7637	72.5144	47.3693	1.6152
	3	98.6150	92.8442	80.0219	64.0484	1.3850
	4	98.7137	93.9162	83.8861	72.5884	1.2863
	5	98.7654	94.6093	86.2706	77.5107	1.2346
	6	98.7960	95.0467	87.8664	80.5994	1.2040
5	2	98.3659	90.8119	72.9086	48.2551	1.6342
	3	98.6861	93.1478	80.9460	65.5932	1.3139
	4	98.8251	94.3316	85.0779	74.4829	1.1749
	5	98.8999	95.0885	87.6108	79.5869	1.1001
	6	98.9429	95.5780	89.2932	82.7751	1.0571
6	2	98.3402	90.7907	73.1332	48.8313	1.6598
	3	98.7066	93.2578	81.5223	66.5643	1.2934
	4	98.8771	94.5358	85.8314	75.6368	1.1229
	5	98.9706	95.3389	88.4717	80.8452	1.0294
	6	99.0230	95.8602	90.2191	84.0971	0.9770

Table 7. Comparison of the search complexity among various methods

Methods		Search complexity
G.729A		320
GPR	R=1	37
	R=2	64
	R=3	91
	R=4	118
IFPR		48
RCM	N=1	1
	N=2	16
	N=3	81
	N=4	256
	N=5	625
	N=6	1296
	N=7	2401
	N=8	4096
Fixed-G1-RCM	N=2	8
	N=3	27
	N=4	64
	N=5	125
	N=6	216
	N=7	343
	N=8	512

search complexity is reduced as intended in comparison with the original RCM approach.

Similarly, tabulated in Table 8 is the search complexity given by (10) for various combinations of (*M*, *N*) in Fixed-2Track-RCM. It is noted that the search complexity increases more rapidly with *N* than with *M*. For instance, it requires 21 searches in the case of (M = 3, N = 3), while the number of searches is elevated to 40 in the case of (M = 3, N = 4), but reduced to merely 25 in the case of (M = 4, N = 3). In simple terms, the increase of *M*, as opposed to *N*, is found to be a superior way when dealing with the issue of speech quality improvement in the aspect of search complexity reduction.

Tabulated in Tables 9 and 10 are the comparisons of both PESQ and MOS-LQO, each comprising the mean and the standard deviation (STD). In comparison with MOS-LQO, G.729A, all the approaches provide a comparable speech quality within a 1% deviation, except that RCM-1 exhibits a 3% drop. Furthermore, as suggested in [17], there is a nearly consistent subjective speech quality with listening test in the presence of 1% MOS-LQO deviation.

The analysis on the experimental results is summed up as follows. Firstly, with a marginal variation in MOS-LQO, a low search complexity reflects a high search performance. This proposal is validated as an efficient and tunable means to achieve the aim of search performance elevation. Taking Fixed-G1-RCM as an instance, the choice of N = 3 is recommended with a 60.25% search accuracy, 3.9518 MOS-LQO and 27 searches, that is a figure 42.19% of GPR-2, 56.25% of IFPR and 33.33% of RCM-3.

Likewise, it is recommended in Fixed-2Track-RCM to choose (M = 2, N = 3) and (M = 3, N = 4) for a relatively low and a relatively high search accuracy, respectively. With a 54.94% search accuracy and a 3.9471 MOS-LQO, the case of (M = 2, N = 3) requires 15 searches, a number 23.44% of GPR-2 and 31.25% of IFPR, while the case of (M = 3, N = 4) demonstrates a 68.64% search accuracy, a 3.9690 MOS-LQO, and requires 40 searches, which is 62.5% of GPR-2 and 83.33% of IFPR.

Table 8. Comparison of the search complexity for Fixed-2Track-RCM method

Fixed-2Track-RCM		Search complexity
M	**N**	
1	2	4
	3	9
	4	16
	5	25
	6	36
2	2	6
	3	15
	4	28
	5	45
	6	66
3	2	8
	3	21
	4	40
	5	65
	6	96
4	2	9
	3	25
	4	49
	5	81
	6	121
5	2	10
	3	29
	4	58
	5	97
	6	146
6	2	11
	3	33
	4	67
	5	113
	6	171

5. Conclusion

Applied to an algebraic codebook search conducted on a G.729A speech codec, two improved versions of RCM, Fixed-G1-RCM and Fixed-2Track-RCM, are presented in this work as elegant ways for

Table 9. Comparison of the speech quality among various methods

Methods		PESQ		MOS-LQO	
		Mean	STD	Mean	STD
G.729A		3.8126	0.0838	3.9502	0.1026
GPR	R=1	3.8145	0.0757	3.9521	0.0883
	R=2	3.8324	0.0777	3.9725	0.0906
	R=3	3.8376	0.0739	3.9786	0.0859
	R=4	3.8382	0.0749	3.9793	0.0867
IFPR		3.8276	0.0761	3.9672	0.0884
RCM	N=1	3.7083	0.0807	3.8250	0.1026
	N=2	3.8064	0.0819	3.9423	0.0971
	N=3	3.8256	0.0740	3.9651	0.0868
	N=4	3.8331	0.0735	3.9735	0.0859
	N=5	3.8393	0.0677	3.9810	0.0780
	N=6	3.8457	0.0736	3.9877	0.0846
	N=7	3.8466	0.0718	3.9888	0.0819
	N=8	3.8446	0.0766	3.9863	0.0881
Fixed-G1-RCM	N=2	3.7937	0.0775	3.9279	0.0927
	N=3	3.8143	0.0766	3.9518	0.0906
	N=4	3.8312	0.0682	3.9717	0.0791
	N=5	3.8359	0.0733	3.9767	0.0840
	N=6	3.8324	0.0687	3.9730	0.0795
	N=7	3.8335	0.0724	3.9741	0.0840
	N=8	3.8373	0.0720	3.9783	0.0825

Table 10. Comparison of the speech quality for Fixed-2Track-RCM method

Fixed-2Track-RCM		PESQ		MOS-LQO	
M	N	Mean	STD	Mean	STD
1	2	3.7829	0.0697	3.9156	0.0837
	3	3.8047	0.0792	3.9406	0.0948
	4	3.8083	0.0670	3.9455	0.0791
	5	3.8135	0.0695	3.9513	0.0814
	6	3.8154	0.0697	3.9535	0.0821
2	2	3.7963	0.0666	3.9316	0.0791
	3	3.8101	0.0757	3.9471	0.0896
	4	3.8220	0.0767	3.9607	0.0896
	5	3.8285	0.0735	3.9682	0.0851
	6	3.8253	0.0737	3.9647	0.0868
3	2	3.8011	0.0781	3.9364	0.0934
	3	3.8173	0.0725	3.9555	0.0848
	4	3.8290	0.0724	3.9690	0.0845
	5	3.8284	0.0720	3.9682	0.0834
	6	3.8299	0.0770	3.9696	0.0893
4	2	3.8023	0.0783	3.9379	0.0936
	3	3.8231	0.0685	3.9624	0.0803
	4	3.8348	0.0746	3.9754	0.0860
	5	3.8300	0.0761	3.9699	0.0894
	6	3.8346	0.0755	3.9751	0.0876
5	2	3.8009	0.0801	3.9361	0.0959
	3	3.8253	0.0707	3.9648	0.0825
	4	3.8318	0.0679	3.9723	0.0796
	5	3.8337	0.0715	3.9744	0.0831
	6	3.8342	0.0800	3.9744	0.0931
6	2	3.8045	0.0829	3.9401	0.0998
	3	3.8262	0.0739	3.9657	0.0862
	4	3.8334	0.0731	3.9739	0.0854
	5	3.8335	0.0722	3.9741	0.0841
	6	3.8355	0.0778	3.9761	0.0899

further search performance elevation. The trade off can be tuned between the search accuracy and the search complexity so as to achieve non degraded speech quality. Either Fixed-G1-RCM ($N = 3$) or Fixed-2Track-RCM ($M = 2$, $N = 3$, or $M = 3$, $N = 4$) is demonstrated as a highly superior candidate relative to GPR-2 and IFPR in terms of search complexity reduction. In addition, the proposed approaches can be implemented to other ACELP-based speech coders.

Furthermore, this improved G.729A speech codec can be utilized to improve the VoIP performance on a smart phone. As a consequence, the energy saving target is met for an extended operation time period due to computational load reduction.

Acknowledgment

This research was financially supported by the National Science Council under Grant No. NSC 102-2221-E-167-027, Taiwan, Republic of China.

References

[1] **J. P. Adoul, P. Mabilleau, M. Delprat, S. Morissette.** Fast CELP coding based on algebraic codes. In: *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1987, pp. 1957-1960.

[2] **C. Laflamme, J. P. Adoul, R. Salami, S. Morissette, P. Mabilleau.** 16 kbps wideband speech coding technique based on algebraic CELP. In: *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1991, pp. 13-16.

[3] **R. Salami, C. Laflamme, J. P. Adoul, A. Kataoka, S. Hayashi, T. Moriya, C. Lamblin, D. Massaloux, S. Proust., P. Kroon, Y. Shoham.** Design and description of CS-ACELP: A toll quality 8 kb/s speech coder. *IEEE Trans. Speech and Audio Processing*, 1998, Vol. 6, No. 2, 116-130.

[4] **S. M. Tsai, J. F. Yang.** Efficient algebraic code-excited linear predictive codebook search. *IEE Proceedings Vision, Image and Signal Processing*, 2006, Vol. 153, No. 6, 761-768.

[5] **B. Goode.** Voice over internet protocol (VoIP). *Proceedings of the IEEE*, 2002, Vol. 90, No. 9, 1495–1517.

[6] **T. Zourzouvillys, E. Rescorla.** An Introduction to Standards-Based VoIP: SIP, RTP, and Friends. *IEEE Internet Computing*, 2010, Vol. 14, No. 2, 69-73.

[7] **A. E. Frey, G. J. Zenner.** The Role of SIP in the Migration of Service Provider Networks to VoIP. *Bell Labs Technical Journal*, 2004, Vol. 9, No. 3, 199-216.

[8] **IETF RFC 3261.** SIP: Session Initiation Protocol, 2002.

[9] **IETF RFC 3515.** The Session Initiation Protocol (SIP) refer method, 2003.

[10] **IETF RFC 3665.** Session Initiation Protocol (SIP) Basic Call Flow Examples, 2003.

[11] **R. Salami, C. Laflamme, B. Bessette, J. P. Adoul.** ITU-T G.729 Annex A: reduced complexity 8 kb/s CS-ACELP codec for digital simultaneous voice and data. IEEE Communications Magazine, 1997, Vol. 35, No. 9, 56-63.

[12] **F. K. Chen, J. F. Yang, Y. L. Yan.** Candidate scheme for fast ACELP search. In: *IEE Proc. Vision, Image and Signal Processing*, 2002, Vol. 149, No. 1, 10-16.

[13] **H. Park, Y. Choi, D. Lee.** Efficient codebook search method for ACELP speech codecs. In: *Proc. IEEE Workshop on Speech Coding*, 2002, pp. 17-19.

[14] **E. D. Lee, M. S. Lee, D. Y. Kim.** Global pulse replacement method for fixed codebook search of ACELP speech codec. In: P*roc. Second IASTED International Conference on Communications, Internet, and Information Technology*, 2003, pp. 372-375.

[15] **E. D. Lee, S. H. Yun, S. I. Lee, J. M. Ahn.** Iteration-free pulse replacement method for algebraic codebook search. *Electronics Letters*, 2007, Vol. 43, No. 1, 59–60.

[16] **E. D. Lee, J. M. Ahn.** Efficient fixed codebook search method for ACELP speech codecs. *Lecture Notes in Computer Science 4413*, 2007, 178-187.

[17] **C. Y. Yeh, Y. J. Su.** Reduced candidate mechanism for an algebraic code-excited linear-prediction codebook search. *IET Communications*, 2012, Vol. 6, No. 17, 2864-2869.

[18] **F. K. Chen, G. M. Chen, B. K. Su, Y. R. Tsai.** Unified pulse-replacement search algorithms for algebra codebooks of speech coders. *IET Signal Processing*, 2010, Vol. 4, No. 6, 658-665.

[19] **C. P. Chu, C. Y. Yeh, S. H. Hwang.** An efficient search strategy for ACELP algebraic codebook by means of reduced candidate mechanism and iteration-free pulse replacement. *Information Technology and Control*, 2014, Vol. 43, No. 2, 183-187.

[20] **L. Li, G. Chen.** Study on fast fixed-codebook search algorithm in G.729 speech coding. In: *Proc. IEEE International Workshop on Computer Science and Engineering*, 2009, pp. 279-282.

[21] **ITU-T Recommendation P.862.** Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs, 2001.

[22] **ITU-T Recommendation P.862.1.** Mapping function for transforming P.862 raw result scores to MOS-LQO, 2003.

Received February 2015.