

A SYSTEM AND METHOD FOR SUPPORTING DUAL SPEECH CODECS

Field of the Invention

[0001] The present invention generally relates to fixed codebook search of codecs. In particular, the invention relates to a method and system supporting dual speech codecs by modifying fixed codebook search of one of the codecs, thus allowing common hardware implementation on for example a co-processor.

Background of the Invention

[0002] Support for multiple speech codecs is a necessity in many communication systems, for e.g. in applications like DSVD and VoIP. Generally these codecs are implemented in software on a digital signal processor (DSP). Different codecs take different processing times depending on their complexities as well as processor speeds.

[0003] G.723.1 and G.729A are speech codecs that are widely used in various applications. These are complex codecs and usually take large amounts of processing time and memory of the processor. Both speech coders for G.723.1 and G.729A use Algebraic-Code-Excited Linear-Prediction (ACELP). The Algebraic-Code-Excited Linear-Prediction (ACELP) coder is based on the Code-Excited Linear-Prediction (CELP) coding model.

[0004] Due to growing VoIP market, VoIP and DSVD application products have to support multiple speech codecs for the applications. For gateway applications, one has to support multiple channels as well. A lot of processing power and memory is needed to support these higher end solutions.

[0005] A functional block diagram of a typical ACELP encoder is shown in FIG.1. The three main functional blocks in an ACELP encoder that consumes the highest proportion of processing power and memory are: Linear Predictive coding (LPC) analysis, Adaptive codebook search, and Fixed codebook search.

[0006] Implementing these three major blocks on a co-processor would advantageously free up on the processing capacity of the DSP for other computations and functions. However, the disparity between the different speech codecs disadvantageously requires that the varied functions to be performed on each codec be implemented on one separate co-processor. Having multiple codec compatibility would mean having multiple co-processors for handling the multiple codecs.

[0007] The fixed codebook search algorithms for G.723.1 (5.3kbps) and G.729A codecs are both based on algebraic codebook searches. By possibly implementing fixed codebook searches of both these codecs on a single co-processor can advantageously reduce the complexity of the system and allow unused processing power and memory of the DSP to be used for supporting multiple channels and others application specific modules.

[0008] Fixed codebook searches in G.729A adopt a “Depth-first tree search” algorithm, which is well discussed in US Patent No. 5,701,392 by Adoul et al. Fixed codebook searches in G.723.1 however adopt a “Nested-loop search” algorithm, which has since been improved upon using a “Focused Nested-loop search” algorithm. These search techniques are well documented in ITU-T Recommendation G.723.1: Dual Speech Coder for Multimedia Communications transmitting at 5.3 and 6.3 Kbits, 3/1996. The “Focused Nested-loop search” and the “Depth-first tree search” algorithms are distinctly different. Attempting to implement these two fixed codebook searches of different search algorithms of two

different codecs would not result in the desired effect of freeing up processing power or memory. Instead, additional processing burden would have been imposed on the co-processor, and implementing the fixed codebook searches on two co-processor would have been more effective but not necessarily more efficient.

[0009] Therefore, a need clearly exists for a method and system for implementing efficient support for dual or multiple codecs or at least alleviate the limitations of existing systems.

Summary of the Invention

[0010] The present invention seeks to provide a method and system supporting dual speech codecs by modifying fixed codebook search of one of the codecs.

[0011] Accordingly, in one aspect, the present invention provides, a method for performing a fixed codebook search of a codebook of G.723.1(5.3Kbps) codec, for forming an optimum codevector in accordance with a predetermined search criteria, the optimum codevector comprising a first pulse, a second pulse, a third pulse and a fourth pulse, each pulse assignable to a predetermined pulse position in the optimum codevector and each pulse having a shift bit for indicating an odd position; the method comprising the steps: providing the codebook of G.723.1(5.3Kbps) codec comprising a first track, a second track, a third track and a fourth track, each track comprising eight predetermined even pulse positions; partitioning the optimum codevector into a first subset comprising the first pulse and the second pulse, and a second subset comprising the third pulse and the fourth pulse; performing a first search for determining a first possible set of pulse positions of the optimum codevector; performing a second search for determining a second possible set of positions of the optimum codevector; and forming the optimum codevector.

[0012] In another aspect, the present invention provides, a method for performing a fixed codebook search of a codebook of a first codec, for forming an optimum codevector in accordance with a predetermined search criteria, the optimum codevector comprising a first pulse, a second pulse, a third pulse and a fourth pulse, each pulse assignable to a predetermined pulse position in the optimum codevector and each pulse having a shift bit for indicating an odd position; the method comprising the steps: providing the codebook of the first codec comprising a first track, a second track, a third track and a fourth track, each track comprising eight predetermined even pulse positions; partitioning the optimum codevector into a first subset comprising the first pulse and the second pulse, and a second subset comprising the third pulse and the fourth pulse; performing a first search for determining a first possible set of pulse positions of the optimum codevector; performing a second search for determining a second possible set of positions of the optimum codevector; and forming the optimum codevector.

[0013] In yet another aspect, the present invention provides, a system for supporting fixed codebook searches for G.723.1(5.3Kbps) codec and G.729A codec for forming an optimum codevector in accordance with a predetermined search criteria, the optimum codevector comprising a first pulse, a second pulse, a third pulse and a fourth pulse, each pulse assignable to a predetermined pulse position in the optimum, the system comprising: a DSP for performing and coordinating functions and calculations for encoding and decoding of received communication signals and a co-processor for performing the fixed codebook searches for G.723.1(5.3Kbps) codec and G.729A codec; wherein the G.723.1(5.3Kbps) codec is searched with the following steps: providing the codebook of G.723.1(5.3Kbps) codec comprising a first track, a second track, a third track and a fourth track, each track comprising eight predetermined even pulse positions; partitioning the optimum codevector into a first subset comprising the first pulse and the second pulse, and a second subset comprising the third pulse and the fourth pulse;

performing a first search for determining a first possible set of pulse positions of the optimum codevector; performing a second search for determining a second possible set of positions of the optimum codevector; and forming the optimum codevector.

Brief Description of the Drawings

[0014] A preferred embodiment of the present invention will now be more fully described, with reference to the drawings of which:

[0015] FIG.1 illustrates a functional block diagram of a typical ACELP encoder;

[0016] FIG.2 illustrates a flowchart of a method for performing a fixed codebook search in accordance with the preferred embodiment;

[0017] FIG.3 illustrates a flowchart of the step of applying Depth First Tree Search of FIG.2;

[0018] FIG.4 illustrates a flowchart of the step of performing a first search of FIG.3;

[0019] FIG.5 illustrates a flowchart of the step of performing a second search of FIG.3;

[0020] FIG.6A, FIG.6B and FIG.6C illustrates respectively simulation results for PESQ-MOS score, SNR and SEGSNR performances (dB);

[0021] FIG.7A illustrates an original speech sample of that is used for testing;

[0022] FIG.7B and FIG.7C illustrates respectively reconstructed signals of the speech sample in FIG.7A using respectively the original ITU-T algorithm and the algorithm of the preferred embodiment;

[0023] FIG.8 illustrates the processing flow for DSP and co-processor system, supporting the two speech codecs;

[0024] FIG.9 illustrates a functional block diagram of an encoder of ITU-T G.723.1;

[0025] FIG.10A illustrates a proposed DSP and Co-processor design for G.723.1; and

[0026] FIG.10B illustrates a proposed DSP and Co-processor design for G.729A.

Detailed description of the Drawings

[0027] A method and system for supporting dual speech codecs with a preferred embodiment is described. In the following description, details are provided to describe the preferred embodiment. It shall be apparent to one skilled in the art, however that the preferred embodiment may be practiced without such details. Some of the details may not be described at length so as not to obscure the preferred embodiment.

[0028] The preferred embodiment takes into consideration the fixed codebook search portion in supporting two codecs by a single co-processor. In particular, the two codecs are G.723.1 (5.3kbps) and G.729A. G.729A is a recommended improvement over G.729, one of the improvements being the adoption of an iterative "Depth-first tree search" algorithm being applied for the fixed codebook

search as compared to G.729 where “Focused Nested-loop search” was originally adopted. Details of G.729A implementations are well discussed in ITU-T Recommendation G.729 – Annex A: Reduced complexity 8 bit/s CS-ACCEPT Speech Coding Algorithm 11/1996.

[0029] By adopting a single fixed codebook search algorithm for both G.723.1 and G.729A, this advantageously simplifies the fixed codebook search process such that a single co-processor running one such fixed codebook search algorithm may be used for both codecs.

[0030] Modifying the fixed codebook search algorithm of G.723.1 to be similar to that of G.729A would advantageously result in a single fixed codebook search algorithm being used for both these codecs. Present G.723.1 fixed codebook search algorithms are also based on “Focused Nested-loop search”, proposing a new G.723.1 codebook search algorithm to be based on “Depth-first tree search” would then have the desired effect of having one fixed codebook search for both G.723.1 and G.729A in accordance with the preferred embodiment.

Conventional G.723.1 Fixed Codebook Search

[0031] A codebook, in the CELP context, is an indexed set of L-sample long sequences, which will be referred to as L-dimensional codevectors. The codebook comprises an index ξ ranging from 1 to M, where M represents the size of the codebook sometimes expressed as a number of bits b :

$$M=2^b \quad (1)$$

[0032] An algebraic codebook is a set of indexed codevectors of which the amplitudes and positions of the pulses of the ξ^{th} codevector can be derived from a

corresponding index ξ through a rule requiring minimal physical storage. Therefore, the size of algebraic codebooks are not limited by storage requirements and are also designed for efficient searches.

[0033] Algebraic codebooks comprises a set of codevectors $\mathbf{v}_\xi$, each defining a plurality of different positions p and N non-zero amplitudes pulses, each assignable to a predetermined valid position p of the codevector.

[0034] The conventional G.723.1 (5.3 kbps) code book search uses a 17bit algebraic codebook for a fixed code excitation $v[n]$. Each fixed codevector contains, at most, four non-zero pulses. The four pulses can assume the signs and positions as shown in Table. 1.

[0035]

Pulse Number	Track	Sign	Positions
0	T_0	$S_0: \pm 1$	$m_0: 0, 8, 16, 24, 32, 40, 48, 56$
1	T_1	$S_1: \pm 1$	$m_1: 2, 10, 18, 26, 34, 42, 50, 58$
2	T_2	$S_2: \pm 1$	$m_2: 4, 12, 20, 28, 36, 44, 52, (60)$
3	T_3	$S_3: \pm 1$	$m_3: 6, 14, 22, 30, 38, 46, 54, (62)$

Table. 1

[0036] The codebook vector $v(n)$ is constructed by taking a zero vector of dimension 60, and putting the four unit pulses at the found locations, multiplied with their corresponding sign:

$$v(n) = s_0\delta(n - m_0) + s_1\delta(n - m_1) + s_2\delta(n - m_2) + s_3\delta(n - m_3), n = 0, \dots, 59 \quad (2)$$

Where $\delta(0)$ is a unit pulse.

[0037] The positions of all pulses can be simultaneously shifted by one (to occupy odd positions), which needs one extra bit. Note that the last position of each of the last two pulses falls outside the subframe boundary, which signifies that the pulses are not present.

[0038] Each pulse position is encoded in 3 bits and each pulse sign is encoded in 1 bit. This gives a total of 16 bits for the 4 pulses. Further, an extra bit is used to encode the shift resulting in a 17-bit codebook.

[0039] The codebook is searched by minimizing the mean square error between the weighted speech signal, $r[n]$, and the weighted synthesis speech given by:

$$E_{\xi} = ||\mathbf{r} - \mathbf{G}\mathbf{H}\mathbf{v}_{\xi}||^2 \quad (3)$$

[0040] Where $\mathbf{r}$ is the target vector consisting of the weighted speech after subtracting the zero-input response of the weighted synthesis filter and the pitch contribution, G is the codebook gain; $\mathbf{v}_{\xi}$ is the algebraic codeword at index ξ ; and $\mathbf{H}$ is a lower triangular Toeplitz convolution matrix with diagonal $h(0)$ and lower diagonals $h(1), \dots, h(L-1)$, with $h(n)$ being the impulse response of the weighted synthesis filter $S_i(z)$.

[0041] It can be shown that the optimum codeword is one, which maximizes the term:

$$\tau_{\xi} = \frac{C_{\xi}^2}{\mathcal{E}_{\xi}} = \frac{(\mathbf{d}^T \mathbf{f}_i)^2}{\mathbf{f}_i^T \mathbf{\Phi} \mathbf{f}_i} \quad (4)$$

[0042] Where C_{ξ} is the correlation value at index ξ and $\mathcal{E}_{\xi}$ energy at index ξ .

$\mathbf{d} = \mathbf{H}^T \mathbf{r}$ is the correlation between the target vector signal, $r[n]$, and the impulse response, $h(n)$. $\mathbf{\Phi} = \mathbf{H}^T \mathbf{H}$ is the covariance matrix of the impulse response. The vector $\mathbf{d}$ and the

matrix Φ are computed prior to the codebook search. The elements of the vector d are computed by:

$$d(j) = \sum_{n=j}^{59} r[n] \cdot h[n-j], 0 \leq j \leq 59 \quad (5)$$

and the elements of the symmetric matrix $\Phi(i, j)$ are computed by:

$$\phi(i, j) = \sum_{n=j}^{59} h[n-i] h[n-j], j \geq i; \text{ and } 0 \leq i \leq 59 \quad (6)$$

[0043] The algebraic structure of the codebook allows for very fast search procedures since the excitation vector v_{ξ} contains only 4 non-zero pulses. The conventional G.723.1 (5.3 kbps) code book search is performed in 4 nested loops, corresponding to each pulse position, where in each loop the contribution of a new pulse is added. The correlation in equation (4) is given by:

$$C = \alpha_0 d[m_0] + \alpha_1 d[m_1] + \alpha_2 d[m_2] + \alpha_3 d[m_3] \quad (7)$$

where m_k is the position of the k th pulse and α_k is its sign (± 1). The energy for even pulse position codevectors in equation (4) is given by:

$$\varepsilon = \sum_{i=0}^3 \phi(m_i, m_i) + 2 \sum_{i=0}^2 \sum_{j=i+1}^3 \alpha_i \alpha_j \phi(m_i, m_j) \quad (8)$$

[0044] For odd pulse position codevectors, the energy in equation (4) is approximated by the energy of the equivalent even pulse position codevector obtained by shifting the odd position pulses to one sample earlier in time. To simplify the search procedure, the functions $d[j]$ and $\phi(m_i, m_j)$ are modified. The simplification is performed as follows (prior to the codebook search). First, the signal $s[j]$ is defined and then the signal $d'[j]$ is constructed.

$$\begin{aligned} s[2j] &= s[2j+1] = \text{sign}(d[2j]) & \text{if } |d[2j]| > |d[2j+1]| \\ s[2j] &= s[2j+1] = \text{sign}(d[2j+1]) & \text{Otherwise} \end{aligned} \quad (9)$$

[0045] The signal d' is further given by $d'[j] = d[j]s[j]$. The matrix Φ is further modified by including the signal information; that is, $\Phi'(i, j) = s[i]s[j]\Phi(i, j)$. The correlation in equation (7) is now given by:

$$C = d'[m_0] + d'[m_1] + d'[m_2] + d'[m_3] \quad (10)$$

and the energy in equation (8) is given by:

$$\varepsilon = \sum_{i=0}^3 \phi'(m_i, m_i) + 2 \sum_{i=0}^2 \sum_{j=i+1}^3 \phi'(m_i, m_j) \quad (11)$$

[0046] Which is further expanded to obtain:

$$\begin{aligned} \varepsilon = & \phi'(m_0, m_0) \\ & + \phi'(m_1, m_1) + 2\phi'(m_0, m_1) \\ & + \phi'(m_2, m_2) + 2[\phi'(m_0, m_2) + \phi'(m_1, m_2)] \\ & + \phi'(m_3, m_3) + 2[\phi'(m_0, m_3) + \phi'(m_1, m_3) + \phi'(m_2, m_3)] \end{aligned} \quad (12)$$

[0047] In conventional G.723.1 (5.3 kbps), where there are four pulses divided into four tracks, each pulse position corresponds to one track. Each track having eight possible pulse positions. In “exhaustive nested-loop” search approach, there are then four nested loops. “Focused nested loop search” is used to further simplify the search procedure. A predetermined threshold is tested before entering the last loop, and the loop is entered only if this threshold is exceeded. The maximum number of times the loop can be entered is fixed so that a lower percentage of the codebook is searched. This threshold is computed based on the correlation C as given in equation (10). The maximum absolute correlation and the average correlation due to the contribution of the first three pulses, max_3 and av_3 , are found prior to the codebook search. The threshold is given by:

$$thr_3 = av_3 + (max_3 - av_3) / 2 \quad (13)$$

[0048] The fourth loop is entered only if the absolute correlation (due to three pulses) exceeds thr_3 . Note that this results in a variable complexity search. To further control the search, the number of times the last loop is entered (for the 4 sub frames) is not allowed to exceed 600. (The average worst case per subframe is 150 times. This can be viewed as searching only $150 \times 8 = 2000$ entries of the codebook, ignoring the overhead of the first three loops.). But in the case of exhaustive nested –loop search $8^4 = 4096$ possible pulse positions are searched.

Conventional G.729 Fixed Codebook Search

[0049] In G.729, the fixed codebook is based on an algebraic codebook structure using an Interleaved Single-Pulse Permutation (ISPP) design. In this codebook, each codebook vector contains four non-zero pulses. Each pulse can have either the amplitudes $+1$ or -1 , and can assume the positions given in Table 2 where the structure of the fixed codebook is illustrated.

Pulse Number	Track	Sign	Positions
0	T_0	$S_0: \pm 1$	$m_0: 0, 5, 10, 15, 20, 25, 30, 35$
1	T_1	$S_1: \pm 1$	$m_1: 1, 6, 11, 16, 21, 26, 31, 36$
2	T_2	$S_2: \pm 1$	$m_2: 2, 7, 12, 17, 22, 27, 32, 37$
3	T_3	$S_3: \pm 1$	$m_3: 3, 8, 13, 18, 23, 28, 33, 38$ 4, 9, 14, 19, 24, 29, 34, 39

Table. 2

[0050] The codebook vector $v(n)$ is constructed by taking a zero vector of dimension 40, and putting the four unit pulses at the found locations, multiplied with their corresponding sign:

$$v(n) = s_0 \delta(n - m_0) + s_1 \delta(n - m_1) + s_2 \delta(n - m_2) + s_3 \delta(n - m_3), n = 0, \dots, 39 \quad (14)$$

Where $\delta(0)$ is a unit pulse.

[0051] The fixed codebook is searched by minimizing the mean-squared error between the weighted input speech $r(n)$ and the weighted reconstructed speech as given in equation (3). The matrix $\mathbf{H}$ is defined as the lower triangular Toeplitz convolution matrix with diagonal $h(0)$ and lower diagonal $h(1), \dots, h(39)$. The matrix $\Phi = \mathbf{H}^T \mathbf{H}$ contains the correlations of $h(n)$, and the elements of this symmetric matrix are given by:

$$\phi(i, j) = \sum_{n=j}^{39} h[n-i] \cdot h[n-j], j \geq i; \text{ and } 0 \leq i \leq 39 \quad (15)$$

[0052] The correlation signal $d(n)$ is obtained from the target signal $r(n)$ and the impulse response $h(n)$ by:

$$d(j) = \sum_{n=j}^{39} r[n] h[n-j], 0 \leq j \leq 39 \quad (16)$$

If $\mathbf{v}_\xi$ is the ξ th fixed-codebook vector, then the codebook is search by maximizing the term:

$$\tau_\xi = \frac{C_\xi^2}{\epsilon_\xi} = \frac{(\sum_{n=0}^{39} d(n) v_\xi(n))^2}{\mathbf{1}_\xi^T \Phi \mathbf{1}_\xi} \quad (17)$$

[0053] The signal $d(n)$ and the matrix Φ are computed before the codebook search. Note that only the elements actually needed are computed and an efficient storage procedure has been designed to speed up the search procedure.

[0054] The algebraic structure of the codebook allows for a fast search procedure since the codebook vector $\mathbf{v}_\xi$ contains only four non-zero pulses. The correlation in the numerator of Equation (17) for a given vector $\mathbf{v}_\xi$ is given by:

$$C = \alpha_0 d[m_0] + \alpha_1 d[m_1] + \alpha_2 d[m_2] + \alpha_3 d[m_3] \quad (18)$$

where m_i is the position of the i th pulse and α_i is its amplitude. The energy in the denominator of Equation (17) is given by:

$$\varepsilon = \sum_{i=0}^3 \phi(m_i, m_i) + 2 \sum_{i=0}^2 \sum_{j=i+1}^3 \alpha_i \alpha_j \phi(m_i, m_j) \quad (19)$$

[0055] To simplify the search procedure, the pulse amplitudes are predetermined by quantizing the signal $d(n)$. This is done by setting the amplitude of a pulse at a certain position equal to the sign of $d(n)$ at the position. Before the codebook search, the following steps are done. First, the signal $d(n)$ is decomposed into two parts: its absolute value $|d(n)|$ and its sign “ $sign[d(n)]$ ”. Second, the matrix Φ is modified by including the sign information; that is,

$$\phi'(i, j) = sign[d(i)]sign[d(j)]\phi(i, j), i = 0, \dots, 39 \quad j = i + 1, \dots, 39 \quad (20)$$

The main-diagonal elements of Φ are scaled to remove the factor 2 in Equation (19)

$$\phi'(i, i) = 0.5\phi'(i, i), i = 0, \dots, 39 \quad (21)$$

The correlation in Equation (18) is now given by:

$$C = |d(m_0)| + |d(m_1)| + |d(m_2)| + |d(m_3)| \quad (22)$$

and the energy in Equation (19) is given by:

$$\varepsilon = \sum_{i=0}^3 \phi'(m_i, m_i) + \sum_{i=0}^2 \sum_{j=i+1}^3 \phi'(m_i, m_j) \quad (23)$$

[0056] It is further expanded to obtain:

$$\begin{aligned} \varepsilon = & \phi'(m_0, m_0) \\ & + \phi'(m_1, m_1) + \phi'(m_0, m_1) \\ & + \phi'(m_2, m_2) + \phi'(m_0, m_2) + \phi'(m_1, m_2) \\ & + \phi'(m_3, m_3) + \phi'(m_0, m_3) + \phi'(m_1, m_3) + \phi'(m_2, m_3) \end{aligned} \quad (24)$$

[0057] A focused search approach is used to further simplify the search procedure. In this approach a precomputed threshold is tested before entering the last loop, and the loop is entered only if this threshold is exceeded. The maximum number of times the loop can be entered is fixed so that a low percentage of the codebook is searched. The threshold is computed based on the correlation C . The maximum absolute correlation and the average correlation due to the contribution of the first three pulses, max_3 and av_3 , are found before the codebook search. The threshold is given by:

$$thr_3 = av_3 + K_3(max_3 - av_3) \quad (25)$$

[0058] The fourth loop is entered only if the absolute correlation (due to three pulses) exceeds thr_3 , where $0 \leq K_3 < 1$. The value of K_3 controls the percentage of codebook search and it is set here to 0.4. Note that this results in a variable search time. To further control the search the number of times the last loop is entered (for the two subframes) cannot exceed a certain maximum, which is set here to 180 (the average worst case per subframe is 90 times), that total possible pulse search combination would be $180 \times 8 = 1440$, but in exhaustive “nested-loop search” approach takes $8^4 \times 2 = 2^{13} = 8192$ positions.

[0059] In fixed codebook search of G.729A, “depth-first tree search” algorithm is used in place of “focused search”. In G.729, a fast search procedure based on nested-loop search approach is used. In that approach only 1440 possible position combinations are tested in the worst case out of the 2^{13} position combinations (17.5 percent). In G.729A, search criteria C^2/ϵ is tested for a smaller percentage of possible position combinations using a depth-first tree search approach. In this approach, the P excitation pulses in a subframe are partitioned into M subsets of N_m pulses. The search begins with subset 1 and proceeds with subsequent subsets according to a tree structure whereby subset m is searched at the m^{th} level of the

tree. The search is repeated by changing the order in which pulses are assigned to the position tracks.

[0060] In this particular codebook structure the pulses are partitioned into two subsets ($M=2$) of two pulses ($N_m=2$). The codebook search is started with the following pulse assignment to tracks: pulse i_0 is assigned to track T_2 , pulse i_1 to track T_3 , pulse i_2 to track T_0 , pulse i_3 to track T_1 .

[0061] The search starts with determining the pulse positions (i_0, i_1) by testing a predetermined search criteria for $2 \times 8 = 16$ position combinations, i.e. the positions at two maxima of $|d(n)|$ in track T_2 are tested in combination with the eight positions in track T_3 . Once the positions (i_0, i_1) are found, the search proceeds to determine the positions (i_2, i_3) by testing the search criteria for the $8 \times 8 = 64$ position combination in tracks T_0 and T_1 . The procedure is repeated by cyclically shifting the pulse assignment to the tracks; that is, pulse i_0 is assigned to track T_3 , pulse i_1 to track T_0 , pulse i_2 to track T_1 , pulse i_3 to track T_2 . Then the whole procedure is repeated twice by replacing track T_3 by T_4 since the fourth can be placed in either T_3 or T_4 . Thus in total $(64+16=80) \times 4 = 320$ position combinations are tested, about 3.9 % of all possible position combinations. About 50% of the complexity reduction in the coder part is attributed to the new algebraic codebook search. This was at the expense of slight degradation in coder performance about 0.2 dB drops in signal-to-noise ratio (SNR).

[0062] The pulse positions of the pulses i_0, i_1 and i_2 , are encoded with 3 bits each, while the position of i_3 is encoded with 4 bits. Each pulse amplitude is encoded with 1 bit. This gives a total of 17 bits for the 4 pulses. By defining $s = 1$ if the sign is positive and $s = 0$ if the sign is negative, the sign codeword is obtained from:

$$S = s_0 + 2s_1 + 4s_2 + 8s_3 \quad (25)$$

and the fixed-codebook codeword is obtained from:

$$C = (m_0/5) + 8(m_1/5) + 64(m_2/5) + 512(2(m_3/5) + jx) \quad (26)$$

where $jx = 0$ if $m_3 = 3, 8, \dots, 38$, and $jx = 1$ if $m_3 = 4, 9, \dots, 39$.

[0063] Focus nested loop search” algorithm is currently used for conventional G.723.1 and G.729 codebook searches. A “depth-first tree search” algorithm has been currently used for G.729A.

[0064] By adopting a single fixed codebook search algorithm for both G.723.1 and G.729A, this advantageously simplifies the fixed codebook search process such that a single co-processor running one such fixed codebook search algorithm may be used for both codecs.

[0065] Modifying the fixed codebook search algorithm of G.723.1 to be similar to that of G.729A would advantageously result in a single fixed codebook search algorithm being used for both these codecs. The present preferred embodiment proposes a new G.723.1 codebook search algorithm based on “Depth-first tree search” thus having the desired effect of one fixed codebook search for both G.723.1 and G.729A.

New proposed G.723.1 Fixed Codebook Search

[0066] A “depth first search algorithm” has previously also been proposed for G.723.1 (5.3Kbps) Codebook search by Huijuan Cui, Kun Tang and Taiyi Cheng in, “Audio as a support to Low Bitrate Multimedia Communication”, International Conference on Communication Technology, ICCT 1998, Vol.1, Pages”544-547. This previously proposed codebook search involves the following steps:

- a. Search first two pulses in full range.
- b. Search last two pulses in full range after the first two pulses are fixed in step1.

- c. Re-search the first two pulses after the last two pulses are fixed in step2.
- d. Re-search the last two pulses after the first two pulses are fixed in step3.

[0067] In the above approach, in each step, two pulses are searched in whole range of codebook from (0-62) possible pulse position combinations. This differs from the proposed approach of the preferred embodiment, where in each step two pulses are searched in only two tracks and not in full range. As such, the approach of the present invention, involves less number of possible pulse positions being searched as compared to the disclosure by Huijian Cui et al. The details of the proposed codebook search of the preferred embodiment for G.723.1 (5.3kbps) is further discussed.

[0068] The similarities and differences between G.723.1 and G.729A speech codecs fixed codebook searches are shown below. There are a few fixed parameters for both speech codecs:

- Number of pulses (N): 4 (in both speech codecs)
- Number of samples per Subframe: 40/60 (G.729A/G.723.1)
- Number of Tracks : 4(in both speech codecs)
- Number of pulse position in each track: 8 (in both speech codec)
- Step for both speech codecs : 5/8(G.729A/G.723.1)

[0069] Furthermore, the initial pulse positions for both speech codecs are different. For G.723.1 it is ($i_0=0, i_1=2, i_2=4, i_3=6$) and for G.729A, it is ($i_0=0, i_1=1, i_2=2, i_3=3$). This can be seen by comparing Table 1 and Table 2.

[0070] Referring to FIG.2, the preferred embodiment adopts the “depth-first tree search” algorithm approach for G.723.1 Fixed Codebook search. The method **200** in accordance with the preferred embodiment has the following steps:

- Sign of correlation signal $d[n]$ is computed **210** in similar manner as in conventional ITU-T G.723.1;

- Depending on the sign, cross correlation values $d(n)$ between target signal $r[n]$ and impulse response $h[n]$ are modified **215**;
- Main diagonal elements of $\phi(n)$ are scaled **220** to remove the factor of 2 as given in equation (11);
- Apply **225** depth first tree search approach to find the best possible pulse positions, which maximizes the search criteria; and
- Compute **230** the 17-bit codebook vector.

[0071] Depth first tree search algorithm of the preferred embodiment for G.723.1 (5.3kbps) is further discussed in detail. Table 1 shows the ACELP codebook for G.723.1 (5.3kbps), in which 4 pulses have to be searched in four tracks. Referring to FIG.3, the method **225** for applying the depth first tree search in accordance with the preferred embodiment is shown. In the present codebook structure, the pulses of the optimum codevector are first partitioned **310** into a first subset and a second subset ($M = 2$), the first subset having a first pulse and a second pulse, while the second subset having the third and fourth pulse ($N_m = 2$).

[0072] The method **225** then proceeds with performing a first **315** search for determining a first possible set of pulse positions, followed by performing a second **320** search for determining a second possible set of pulse positions. The two searches, where each search comprises of two phases A and B. For each search, the algorithm flow should be as follows:

- Search 1 and Phase A
- Search 1 and Phase B
- Search 2 and Phase A
- Search 2 and Phase B

[0073] Start the codebook search with the following pulse assignment to tracks: pulse i_0 is assigned to third track T_2 , pulse i_1 to fourth track T_3 , pulse i_2 to first track T_0 , pulse i_3 to second track T_1 .

[0074] Referring to FIG.4, the step of performing the first search **315** for determining the first possible set of pulse positions is shown.

[0075] In search 1 and Phase A, determining the pulse positions (i_0, i_1) by testing the search criteria for $2 \times 8 = 16$ position combinations, i.e. the positions at two maxima of $|d(n)|$ in track T_2 including even and odd indexed pulse positions and tested in combination with the eight positions in track T_3 including odd and even indexed pulse positions. In this manner (i_0, i_1) is found.

[0076] The step **315** starts with the determining **410** of the two maximum pulse positions in the third track assignable to the first pulse i_0 . Next, the step of testing **415** all the pulses in the fourth track in combination with each of the two maximum pulse positions in the third track for one maximum pulse assignable to the second pulse i_1 . The pulse positions (i_0, i_1) for the first set of possible pulse positions are then determined **420** in accordance with the predetermined search criteria.

[0077] In search 1 and Phase B, the search proceeds to determine the positions (i_2, i_3) by testing the search criteria for the $8 \times 8 = 64$ position combination in tracks T_0 and T_1 including odd and even indexed pulse positions. The step of testing **425** all the pulse positions in the second track in combination with each of the pulse positions in the first track for assigning the pulse positions to the third pulse and the fourth pulse of the first set of possible pulse positions is thus performed. The determining **430** of the pulse positions of the third pulse and the fourth pulse of the first set of possible pulse positions in accordance with the predetermined search criteria is then performed.

[0078] So, in this manner (i_2, i_3) are found and this gives a total of $(16 + 64 = 80)$ possible pulse positions combinations are searched.

[0079] However, for better performance, the correlation signal values of each pulse positions of the first set of possible pulse positions are compared at both even and odd indexed pulse positions. Whichever value is higher is then selected and re-assigned as the pulse position. If the odd indexed correlation signal value is higher, the “shift bit” value is further set at 1 otherwise if the even correlation signal value is higher than it is set at 0.

[0080] The algorithm is shown below:

```

if (dn[i] > dn[i+1]) // where i is even index
{
    shift = 0;
} else
{
    shift = 1;
}

```

[0081] Referring to FIG.5, search 2, which is the step of performing **320** the second search for determining the second set possible set of pulse positions, starts with the step of performing **510** a cyclical shift of the pulse assignment to the tracks; that is, pulse i_0 is assigned to track T_3 , pulse i_1 to track T_0 , pulse i_2 to track T_1 , pulse i_3 to track T_2 .

[0082] In search 2, Phase A, a similar procedure is repeated to find the second possible set of pulse positions. The step **320** then proceeds with the step of determining **515** the two maximum pulse positions in the fourth track assignable to the first pulse i_0 . Next, the step of testing **520** all the pulses in the first track in combination with each of the two maximum pulse positions in the fourth track for one maximum pulse assignable to the second pulse i_1 . The pulse positions (i_0, i_1)

for the first set of possible pulse positions are then determined **525** in accordance with the predetermined search criteria.

[0083] In search 2 Phase B, the search proceeds to determine the positions (i_2, i_3) by testing the search criteria for the $8 \times 8 = 64$ position combination in tracks T_3 and T_0 including odd and even indexed pulse positions. The step of testing **530** all the pulse positions in the third track in combination with each of the pulse positions in the second track for assigning the pulse positions to the third pulse and the fourth pulse of the second set of possible pulse positions is thus performed. The determining **535** of the pulse positions of the third pulse and the fourth pulse of the first set of possible pulse positions in accordance with the predetermined search criteria is then performed.

[0084] For better performance, the correlation signal values of each pulse positions of the second set of possible pulse positions are again compared at both even and odd indexed pulse positions. Thus in total $(64+16=80) * 2 = 160$ position combinations are searched in the preferred embodiment as compared to, approximately 2000 positions searched in original ITU-T G.723.1 Fixed Codebook search. This is about 8% of the original ITU-T G.723.1 Fixed Codebook search.

[0085] The first and second sets of possible pulse positions are then further compared. The four pulse positions from the first and second set of possible pulse positions are then selected and together with their sign and shift values, the 17-bit codebook vector is computed in a similar manner as the original ITU-T G.723.1. This way the decoder compatibility will not be lost due to the change in algorithm.

[0086] Using the method of the preferred embodiment, there is up to 50% reduction in complexity of G.723.1 (5.3 Kbps) algebraic codebook search.

Validation Results

[0087] Results for the new fixed codebook search for G.723.1 (5.3kbps) of the preferred embodiment are shown in FIG.6A, FIG.6B and FIG.6C. Simulations were performed for both ITU-T version algorithm and algorithm of the preferred embodiment for 23 speech test vectors. About 20 speech test vectors are taken from ITU-T P.862 standards, where these test vectors are generated from different sources ranging from women, men, and children as well as different language speakers. Other three test vectors are sample test speech vectors of about one minute each. For these test vectors, three types of validation tests- (PESQ-MOS score, SNR and SEGSNR) are carried out and these results are shown in FIG.6.

[0088] Figure 6A shows the PESQ-MOS score comparison for the algorithm of the preferred embodiment and the ITU-T algorithm for 23 test vectors. It shows a 5-8% degradation of PESQ-MOS score on the algorithm of the preferred embodiment as compared to the original ITU-T algorithm. However, 5-8% degradation in performance is balanced by more than 50% savings on the complexity. PESQ-MOS score for modified algorithm varies from 3.4 to 3.55 for different test vectors as compared to the original ITU-T algorithm (3.5 to 3.8).

[0089] FIG.6B and FIG.6C, show respectively the SNR and SEGSNR performances (dB) respectively for both algorithms for the 23 speech test vectors. The results show around 2dB SNR degradation and 1.5dB SEGSNR degradation in the algorithm of the preferred embodiment as compared to the original ITU-T algorithm.

[0090] FIG.7A shows the original speech sample that is used for testing the original ITU-T algorithm and the algorithm of the preferred embodiment. FIG.7B and

FIG.7C shows reconstructed signals of the speech sample in FIG.7A using respectively the original ITU-T algorithm and the algorithm of the preferred embodiment

[0091] Listening tests were also carried out for different speech test vectors by different subjects. There was generally no significant degradation in perceived speech quality as compare to the standard ITU-T algorithm. So, the algorithm of the preferred embodiment while providing slight degradation in speech quality, results in saving of more then 50% of processing power over the standard ITU-T algorithm.

[0092] Based on these algorithmic changes in G.723.1 codebook search algorithm, it is possible to implement a single co-processor solution, which allows the supporting of codebook searches for multiple speech codecs, which in accordance to the preferred embodiment are: G.723.1 (5.3kbps) and G.729A.

Hardware Implementation and Design

[0093] When considering the G.729A speech codec, the fixed codebook search is performed twice in each frame, while in the algorithm of the preferred embodiment of G.723.1; it is performed four times in a frame. This does not present any concerns in co-processor design, as it is the number of times this is called by the DSP is different.

[0094] The re-configurable parameters of both speech codecs can be configured before the start of co-processor processing by the DSP and passed to the coprocessor. These re-configurable parameters of concern are:

- Number of pulses (N): 4
- Number of samples per Sub frame (SubFrLen): 40/60 (G.729A/G.723.1)

- Number of Tracks: 4
- Number of pulse position in each track: 8
- Step for both speech codec: 5/8 (G.729A/G.723.1)
- Initial pulse positions for both speech codecs are different.
For G.723.1 it is ($i_0=0, i_1=2, i_2=4, i_3=6$) and for G.729A, it is ($i_0=0, i_1=1, i_2=2, i_3=3$).

[0095] In addition to the above, there is an additional reconfigurable parameter called SubFrLen2 for G.723.1. SubFrLen is fixed at 40 for G.729A and 60 for G.723.1. However, when considering track T_2 and track T_3 of G.723.1, to accommodate the maximum pulse position index of 60 and 62 respectively as shown in Table 1, SubFrLen2 is set at 62. As such, during a codebook search of G.723.1, pulses searched in track T_2 and track T_3 , ends at SubFrLen2 i.e. 62 instead of SubFrLen i.e. 60. But, if the pulses are found at positions 60 and 62, it will not be considered.

[0096] From the codebook structure for both speech codecs in Table 1 and Table 2, it can be seen that G.729A codebook structure has continuous pulse positions from 0-39 pulses, while G.723.1 (5.3kbps) codebook structure has only even indexed pulse positions from 0-62. Odd indexed pulse positions conditions are taken care of by comparing the correlation signal $|d(n)|$ values at both indexes. Depending on this comparison, a “shift” value is computed, as explained previously. But in G.729A, there is no concept of even and odd indexed pulse positions and is therefore unaffected.

[0097] In the co-processor design for supporting both codecs in accordance with the present invention, a codec flag would be implemented for identifying to the co-processor which codec is to be handled. The codec flag would also indicate to the co-processor which codec is used and hence which parameters to adopt. As such,

the same codec flag may also be used to handle the added indexed pulses of G.723.1.

[0098] During the codebook search of G.729A, the fourth pulse i_3 is selected from track T_3 and track T_4 . The whole algorithm thus starts from track T_3 . Then, the process is repeated by replacing track T_3 by track T_4 . When considering this in the co-processor, the same codec flag may be used to indicate for G.729A the repetition of the whole algorithm by replacing track T_3 by track T_4 .

[0099] While maintaining the decoder compatibility with ITU-T G.723.1 and ITU-T G.729A decoders, other portions of the fixed codebook search remains the same. The other portions of the algorithm comprises: computing the sign of correlation signal $d(n)$, modification of cross correlation values and computation of the 17-bit codebook vector.

[00100] Codebook search for both speech codecs includes computation of the autocorrelation value $\phi(n)$ of impulse response $h(n)$, and also the cross correlation value $d(n)$ by using target signal $r(n)$ and impulse response $h(n)$. These values are computed before the start of codebook search. The way these values are computed is similar for both speech codecs, except for the difference in subframe size, which is a reconfigurable parameter.

[00101] Using the new proposed algorithm of the preferred embodiment of G.723.1 (5.3kbps) fixed codebook search, a single implementation of G.723.1 and G.729A codebook search on the co-processor is made. Referring to FIG.8, the processing flow for the system of the DSP 10 and co-processor 20 supporting these two speech codecs is shown. The codec selection being made by using the codec flag and re-configurable parameters, but controlled by the DSP 10. The co-

processor **20** mainly handling aspects of the fixed codebook search. The common functionality of the co-processor **20** are:

- i. Check Codec Flag for G.723.1 or G.729A Encoder;
- ii. Configure re-configurable parameters depending on Codec Flag;
- iii. Computing Co-variance $\phi(n)$ and cross-correlation value $d(n)$;
- iv. Computing sign and modify co-variance values depending on codec flag;
- v. Pulse assignment and “depth first tree” depending on codec flag (For G.729A, whole range search will be repeated for track T3, and for G.723.1, “shift” value is computed depending on even and odd index value;
- vi. Computing 17-bit codevector based on the pulse position indexes and flags.

[00102] Further referring to disclosure made by S.M. Mishra and A. Balaram in “Efficient Hardware-Software Co-design for the G.723.1 algorithm targeted at VoIP application”, IEEE International Conference in Multimedia and Expo, 2000 (ICME 2000), vol 3, pgs 1379-1382. Referring to FIG.9, a detailed functional block diagram of a G.723.1 encoder is shown with certain modules grouped into Block A 30 and Block B 32. Mishra et al considered implementing Block A 30 and Block B 32 independently. As such, one of the blocks may be performed on the DSP 10 and another on the Co-processor 20 simultaneously.

[00103] Mishra et al disclosed the processing of Block A 30 on hardware and Block B 32 on the DSP 10 via software. Block A 30 contains pitch estimator, Formant Perceptual Weighting filter and the Harmonic Noise Shaping module, and Block B 32 contains LSP routines. Both Block A 30 and B 32 is synchronized such that the weighted speech $W(z)$ and noise shaper response $P(z)$ are available for the

Impulse Response calculation. In this manner, about 17% of processing power in 5.3kbps and 11% in 6.3 kbps, are reduced.

[00104] Presently, the proposed efficient Hardware-Software co-design in accordance with the preferred embodiment for G.723.1 is shown in Figure 10a. Where the DSP 10 will first be used for High Pass Filter and LPC analysis before the co-processor 20 takes over for the processing of Block A 30, while Block B 32 continues to be processed by the DSP 10. The co-processor 20 can then perform the fixed codebook search upon completion of processing Block A 30. This allows for the simultaneous processing of both Block A 30 and Block B 32. It is estimated that by using this proposed design, one can save around 30-40% processing power. Similarly, Proposed Hardware-Software co-design for G.729A is shown in Figure 10b and it can save around 30% processing power. The DSP 10 will similarly be used for High Pass Filter LPC/LSP analysis as well as for Adaptive Codebook searches while the co-processor would be used for fixed codebook searches.

[00105] While the preferred embodiment refers to specifically the two codecs: G.723.1 and G.729A, it will be appreciated that various modifications and improvements can be made by a person skilled in the art without departing from the scope of the present invention. Particularly in considering other codecs having ACELP coding which have substantially similar structure to the above codecs described.

Claims

1. A method for performing a fixed codebook search of a codebook of G.723.1(5.3Kbps) codec, for forming an optimum codevector in accordance with a predetermined search criteria, the optimum codevector comprising a first pulse, a second pulse, a third pulse and a fourth pulse, each pulse assignable to a predetermined pulse position in the optimum codevector and each pulse having a shift bit for indicating an odd position; the method comprising the steps:
 - a. providing the codebook of G.723.1(5.3Kbps) codec comprising a first track, a second track, a third track and a fourth track, each track comprising eight predetermined even pulse positions;
 - b. partitioning the optimum codevector into a first subset comprising the first pulse and the second pulse, and a second subset comprising the third pulse and the fourth pulse;
 - c. performing a first search for determining a first possible set of pulse positions of the optimum codevector;
 - d. performing a second search for determining a second possible set of positions of the optimum codevector; and
 - e. forming the optimum codevector.
2. The method in accordance with claim 1, wherein the method may further be used to search for another optimum codevector of a codebook of G.729A codec with minor changes in parameters.
3. The method in accordance with claim 2, wherein the method may be implementable on a processor for supporting both G.723.1(5.3Kbps) codec and G.729A codec.

4. The method in accordance with claim 1, wherein step c. comprises the steps:
 - c1. assigning the first pulse, the second pulse, the third pulse and the fourth pulse of the first possible set of pulse positions respectively to the third track, the fourth track, the first track and the second track of the codebook of G.723.1(5.3Kbps) codec for searching;
 - c2. determining two maximum pulse positions in the third track assignable to the first pulse;
 - c3. testing all the pulse positions in the fourth track in combination with each of the two maximum pulse positions in the third track for one maximum pulse assignable to the second pulse;
 - c4. determining the pulse positions of the first pulse and the second pulse of the first set of possible pulse positions in accordance with the predetermined search criteria;
 - c5. testing all the pulse positions in the second track in combination with each of the pulse positions in the first track for assigning the pulse positions to the third pulse and the fourth pulse of the first set of possible pulse positions; and
 - c6. determining the pulse positions of the third pulse and the fourth pulse of the first set of possible pulse positions in accordance with the predetermined search criteria.
5. The method in accordance with claim 4, further comprising the steps:
 - c7. comparing correlation signal values of each pulse positions of the first set of possible pulse positions with the correlation signal values of each corresponding pulse positions incremented by one; and
 - c8. re-assigning the pulse position to the corresponding pulse position of the first set of possible pulse positions and setting the shift bit of the pulse position to one, if the correlation signal value of the corresponding pulse position is higher.

6. The method in accordance with claim 4, wherein the step d. comprises the steps:

d1. performing a single position cyclical shift of assignments of pulses of the second possible set of pulse positions to the tracks of the codebook of G.723.1(5.3Kbps) codec for searching;

d2. determining two maximum pulse positions in the fourth track assignable to the first pulse;

d3. testing all the pulse positions in the first track in combination with each of the two maximum pulse positions in the fourth track for one maximum pulse assignable to the second pulse;

d4. determining the pulse positions of the first pulse and the second pulse of the second set of possible pulse positions in accordance with the predetermined search criteria;

d5. testing all the pulse positions in the third track in combination with each of the pulse positions in the second track for assigning the pulse positions to the third pulse and the fourth pulse of the first set of possible pulse positions; and

d6. determining the pulse positions of the third pulse and the fourth pulse of the second set of possible pulse positions in accordance with the predetermined search criteria.

7. The method in accordance with claim 6, further comprising the steps:

d7. comparing correlation signal values of each pulse positions of the second set of possible pulse positions with the correlation signal values of each corresponding pulse positions incremented by one; and

d8. re-assigning the pulse position to the corresponding pulse position of the second set of possible pulse positions and setting the shift bit of the pulse position to one, if the correlation signal value of the corresponding pulse position is higher.

8. A method for performing a fixed codebook search of a codebook of a first codec, for forming an optimum codevector in accordance with a predetermined search criteria, the optimum codevector comprising a first pulse, a second pulse, a third pulse and a fourth pulse, each pulse assignable to a predetermined pulse position in the optimum codevector and each pulse having a shift bit for indicating an odd position; the method comprising the steps:

- a. providing the codebook of the first codec comprising a first track, a second track, a third track and a fourth track, each track comprising eight predetermined even pulse positions;
- b. partitioning the optimum codevector into a first subset comprising the first pulse and the second pulse, and a second subset comprising the third pulse and the fourth pulse;
- c. performing a first search for determining a first possible set of pulse positions of the optimum codevector;
- d. performing a second search for determining a second possible set of positions of the optimum codevector; and
- e. forming the optimum codevector.

9. The method in accordance with claim 8, wherein the method may further be used to search for another optimum codevector of a codebook of a second codec with minor changes in parameters.

10. The method in accordance with claim 9, wherein the method may be implementable on a processor for supporting both the first codec and the second codec.

11. The method in accordance with claim 1, wherein step c. comprises the steps:

- c1. assigning the first pulse, the second pulse, the third pulse and the fourth pulse of the first possible set of pulse positions respectively to the third track, the fourth

track, the first track and the second track of the codebook of the first codec for searching;

c2. determining two maximum pulse positions in the third track assignable to the first pulse;

c3. testing all the pulse positions in the fourth track in combination with each of the two maximum pulse positions in the third track for one maximum pulse assignable to the second pulse;

c4. determining the pulse positions of the first pulse and the second pulse of the first set of possible pulse positions in accordance with the predetermined search criteria;

c5 testing all the pulse positions in the second track in combination with each of the pulse positions in the first track for assigning the pulse positions to the third pulse and the fourth pulse of the first set of possible pulse positions; and

c6. determining the pulse positions of the third pulse and the fourth pulse of the first set of possible pulse positions in accordance with the predetermined search criteria.

12. The method in accordance with claim 11, further comprising the steps:

c7. comparing correlation signal values of each pulse positions of the first set of possible pulse positions with the correlation signal values of each corresponding pulse positions incremented by one; and

c8. re-assigning the pulse position to the corresponding pulse position of the first set of possible pulse positions and setting the shift bit of the pulse position to one, if the correlation signal value of the corresponding pulse position is higher.

13. The method in accordance with claim 11, wherein the step d. comprises the steps:

- d1. performing a single position cyclical shift of assignments of pulses of the second possible set of pulse positions to the tracks of the codebook of the first codec for searching;
 - d2. determining two maximum pulse positions in the fourth track assignable to the first pulse;
 - d3. testing all the pulse positions in the first track in combination with each of the two maximum pulse positions in the fourth track for one maximum pulse assignable to the second pulse;
 - d4. determining the pulse positions of the first pulse and the second pulse of the second set of possible pulse positions in accordance with the predetermined search criteria;
 - d5. testing all the pulse positions in the third track in combination with each of the pulse positions in the second track for assigning the pulse positions to the third pulse and the fourth pulse of the first set of possible pulse positions; and
 - d6. determining the pulse positions of the third pulse and the fourth pulse of the second set of possible pulse positions in accordance with the predetermined search criteria.
14. The method in accordance with claim 13, further comprising the steps:
- d7. comparing correlation signal values of each pulse positions of the second set of possible pulse positions with the correlation signal values of each corresponding pulse positions incremented by one; and
 - d8. re-assigning the pulse position to the corresponding pulse position of the second set of possible pulse positions and setting the shift bit of the pulse position to one, if the correlation signal value of the corresponding pulse position is higher.
15. A method for performing a fixed codebook search of a codebook of a first codec, for forming an optimum codevector in accordance with a predetermined search criteria, the optimum codevector comprising a plurality of pulses, each pulse assignable to a

predetermined pulse position in the optimum codevector and each pulse having a shift bit for indicating an odd position; the method comprising the steps:

- a. providing the codebook of the first codec comprising a plurality of tracks, each track comprising a plurality of even pulse positions;
- b. partitioning the optimum codevector into a first subset and a second subset, each subset;
- c. performing a first search for determining a first possible set of pulse positions of the optimum codevector;
- d. performing a second search for determining a second possible set of positions of the optimum codevector; and
- e. forming the optimum codevector.

16. The method in accordance with claim 15, wherein the method may further be used to search for another optimum codevector of a codebook of a second codec with minor changes in parameters.

17. The method in accordance with claim 16, wherein the method may be implementable on a processor for supporting both the first codec and the second codec.

18. The method in accordance with claim 15, wherein step c. comprises the steps:

- c1. assigning a plurality of pulses of the first possible set of pulse positions respectively to the plurality of tracks of the codebook of the first codec for searching;
- c2. determining two maximum pulse positions in one of the tracks assignable to the one of the pulses of the first subset;
- c3. testing all the pulse positions in a successive track in combination with each of the two maximum pulse positions in the one of the tracks for one maximum pulse assignable to another pulse of the first subset;
- c4. determining the pulse positions of the first subset of the first set of possible pulse positions in accordance with the predetermined search criteria;

c5 testing all the pulse positions in another successive track in combination with each of the pulse positions in yet another successive track for assigning the pulse positions to the second subset of the first set of possible pulse positions; and

c6. determining the pulse positions of the second subset of the first set of possible pulse positions in accordance with the predetermined search criteria.

19. The method in accordance with claim 18, further comprising the steps:

c7. comparing correlation signal values of each pulse positions of the first set of possible pulse positions with the correlation signal values of each corresponding pulse positions incremented by one; and

c8. re-assigning the pulse position to the corresponding pulse position of the first set of possible pulse positions and setting the shift bit of the pulse position to one, if the correlation signal value of the corresponding pulse position is higher.

20. The method in accordance with claim 15, wherein the step d. comprises the steps:

d1. performing a single position cyclical shift of assignments of pulses of the second possible set of pulse positions to the plurality of tracks of the codebook of the first codec for searching;

d2. determining two maximum pulse positions in one of the tracks assignable to the one of the pulses of the first subset;

d3. testing all the pulse positions in a successive track in combination with each of the two maximum pulse positions in the one of the tracks for one maximum pulse assignable to another pulse of the first subset;

d4. determining the pulse positions of the first subset of the second set of possible pulse positions in accordance with the predetermined search criteria;

c5 testing all the pulse positions in another successive track in combination with each of the pulse positions in yet another successive track for assigning the pulse positions to the second subset of the second set of possible pulse positions; and

c6. determining the pulse positions of the second subset of the second set of possible pulse positions in accordance with the predetermined search criteria.

21. The method in accordance with claim 20, further comprising the steps:

d7. comparing correlation signal values of each pulse positions of the second set of possible pulse positions with the correlation signal values of each corresponding pulse positions incremented by one; and

d8. re-assigning the pulse position to the corresponding pulse position of the second set of possible pulse positions and setting the shift bit of the pulse position to one, if the correlation signal value of the corresponding pulse position is higher.

22. A system for supporting fixed codebook searches for G.723.1(5.3Kbps) codec and G.729A codec for forming an optimum codevector in accordance with a predetermined search criteria, the optimum codevector comprising a first pulse, a second pulse, a third pulse and a fourth pulse, each pulse assignable to a predetermined pulse position in the optimum, the system comprising:

a DSP for performing and coordinating functions and calculations for encoding and decoding of received communication signals and

a co-processor for performing the fixed codebook searches for G.723.1(5.3Kbps) codec and G.729A codec;

wherein the G.723.1(5.3Kbps) codec is searched with the following steps:

a. providing the codebook of G.723.1(5.3Kbps) codec comprising a first track, a second track, a third track and a fourth track, each track comprising eight predetermined even pulse positions;

b. partitioning the optimum codevector into a first subset comprising the first pulse and the second pulse, and a second subset comprising the third pulse and the fourth pulse;

c. performing a first search for determining a first possible set of pulse positions of the optimum codevector;

- d. performing a second search for determining a second possible set of positions of the optimum codevector; and
- e. forming the optimum codevector.

23. The system in accordance with claim 22, wherein a codec flag is used to indicate to the co-processor which codec is used.

24. The system in accordance with claim 22, wherein re-configurable parameters are configured according to the codec used.

25. The system in accordance with claim 23, wherein sub frame length for a third and fourth track of a codebook of G.723.1(5.3Kbps) codec is set to sixty two.

26. The system in accordance with claim 22, wherein a pitch estimator, a Formant Perceptual Weighing filter and a Harmonic Noise Shaping module may be implemented on the co-processor for simultaneous processing with the DSP functions.