INTERNATIONAL TELECOMMUNICATION UNION

ITU-T **G.723.1**

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

(03/96)

GENERAL ASPECTS OF DIGITAL TRANSMISSION SYSTEMS

DUAL RATE SPEECH CODER FOR MULTIMEDIA COMMUNICATIONS TRANSMITTING AT 5.3 AND 6.3 kbit/s

ITU-T Recommendation G.723.1

FOREWORD

The ITU-T (Telecommunication Standardization Sector) is a permanent organ of the International Telecommunication Union (ITU). The ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Conference (WTSC), which meets every four years, establishes the topics for study by the ITU-T Study Groups which, in their turn, produce Recommendations on these topics.

The approval of Recommendations by the Members of the ITU-T is covered by the procedure laid down in WTSC Resolution No. 1 (Helsinki, March 1-12, 1993).

ITU-T Recommendation G.723.1 was prepared by ITU-T Study Group 15 (1993-1996) and was approved under the WTSC Resolution No. 1 procedure on the 19th of March 1996.

____

NOTE

In this Recommendation, the expression “Administration” is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

CONTENTS

Summary

This Recommendation specifies a coded representation that can be used for compressing the speech or other audio signal component of multimedia services at a very low bit rate as part of the overall H.324 family of standards. This coder has two-bit rates associated with it, 5.3 and 6.3 kbit/s. The higher bit rate has greater quality. The lower bit rate gives good quality and provides system designers with additional flexibility. Both rates are a mandatory part of the encoder and decoder. It is possible to switch between the two rates at any frame boundary. An option for variable rate operation using discontinuous transmission and noise fill during non-speech intervals is also possible.

This coder was optimized to represent speech with a high quality at the above rates using a limited amount of complexity. It encodes speech or other audio signals in frames using linear predictive analysis-by-synthesis coding. The excitation signal for the high rate coder is Multipulse Maximum Likelihood Quantization (MP-MLQ) and for the low rate coder is Algebraic-Code-Excited Linear-Prediction (ACELP). The frame size is 30 ms and there is an additional look ahead of 7.5 msec, resulting in a total algorithmic delay of 37.5 msec. All additional delays in this coder are due to processing delays of the implementation, transmission delays in the communication link and buffering delays of the multiplexing protocol.

The description of this Recommendation is made in terms of bit-exact, fixed-point mathematical operations. The ANSI C code indicated in clause 5 constitutes an integral part of this Recommendation and shall take precedence over the mathematical descriptions in this text if discrepancies are found. A non-exhaustive set of test sequences which can be used in conjunction with the C code are available from the ITU.

DUAL RATE SPEECH CODER FOR MULTIMEDIA COMMUNICATIONS TRANSMITTING AT 5.3 AND 6.3 kbit/s

(Geneva, 1996)

1 Introduction

1.1 Scope

This Recommendation specifies a coded representation that can be used for compressing the speech or other audio signal component of multimedia services at a very low bit rate. In the design of this coder, the principal application considered was very low bit rate visual telephony as part of the overall H.324 family of standards.

1.2 Bit rates

This coder has two bit rates associated with it. These are 5.3 and 6.3 kbit/s. The higher bit rate has greater quality. The lower bit rate gives good quality and provides system designers with additional flexibility. Both rates are a mandatory part of the encoder and decoder. It is possible to switch between the two rates at any 30 ms frame boundary. An option for variable rate operation using discontinuous transmission and noise fill during non-speech intervals is also possible.

1.3 Possible input signals

This coder was optimized to represent speech with a high quality at the above rates using a limited amount of complexity. Music and other audio signals are not represented as faithfully as speech, but can be compressed and decompressed using this coder.

1.4 Delay

This coder encodes speech or other audio signals in 30 msec frames. In addition, there is a look ahead of 7.5 msec, resulting in a total algorithmic delay of 37.5 msec. All additional delays in the implementation and operation of this coder are due to:

i) actual time spent processing the data in the encoder and decoder;

ii) transmission time on the communication link;

iii) additional buffering delay for the multiplexing protocol.

1.5 Speech coder description

The description of the speech coding algorithm of this Recommendation is made in terms of bit-exact, fixed-point mathematical operations. The ANSI C code indicated in clause 5, which constitutes an integral part of this Recommendation, reflects this bit-exact, fixed-point description approach. The mathematical descriptions of the encoder and decoder, given respectively in clauses 2 and 3, can be implemented in several other fashions, possibly leading to a codec implementation not complying with this Recommendation. Therefore, the algorithm description of the C code of clause 5 shall take precedence over the mathematical descriptions of clauses 2 and 3 whenever discrepancies are found. A non-exhaustive set of test sequences which can be used in conjunction with the C code are available from the ITU.

2 Encoder principles

2.1 General description

This coder is designed to operate with a digital signal obtained by first performing telephone bandwidth filtering (Recommendation G.712) of the analogue input, then sampling at 8000 Hz and then converting to 16-bit linear PCM for the input to the encoder. The output of the decoder should be converted back to analogue by similar means. Other input/output characteristics, such as those specified by Recommendation G.711 for 64 kbit/s PCM data, should be converted to 16-bit linear PCM before encoding or from 16-bit linear PCM to the appropriate format after decoding. The bitstream from the encoder to the decoder is defined within this Recommendation.

The coder is based on the principles of linear prediction analysis-by-synthesis coding and attempts to minimize a perceptually weighted error signal. The encoder operates on blocks (frames) of 240 samples each. That is equal to 30 msec at an 8 kHz sampling rate. Each block is first high pass filtered to remove the DC component and then divided into four subframes of 60 samples each. For every subframe, a 10th order Linear Prediction Coder (LPC) filter is computed using the unprocessed input signal. The LPC filter for the last subframe is quantized using a Predictive Split Vector Quantizer (PSVQ). The unquantized LPC coefficients are used to construct the short-term perceptual weighting filter, which is used to filter the entire frame and to obtain the perceptually weighted speech signal.

For every two subframes (120 samples), the open loop pitch period, L_{OL}, is computed using the weighted speech signal. This pitch estimation is performed on blocks of 120 samples. The pitch period is searched in the range from 18 to 142 samples.

From this point the speech is processed on a 60 samples per subframe basis.

Using the estimated pitch period computed previously, a harmonic noise shaping filter is constructed. The combination of the LPC synthesis filter, the formant perceptual weighting filter, and the harmonic noise shaping filter is used to create an impulse response. The impulse response is then used for further computations.

Using the pitch period estimation, L_{OL}, and the impulse response, a closed loop pitch predictor is computed. A fifth order pitch predictor is used. The pitch period is computed as a small differential value around the open loop pitch estimate. The contribution of the pitch predictor is then subtracted from the initial target vector. Both the pitch period and the differential value are transmitted to the decoder.

Finally the non-periodic component of the excitation is approximated. For the high bit rate, Multi-pulse Maximum Likelihood Quantization (MP-MLQ) excitation is used, and for the low bit rate, an algebraic-code-excitation (ACELP) is used.

The block diagram of the encoder is shown in Figure 1.

2.2 Framer

File : LBCCODEC.C	Procedure : main()	Reads 240 samples input frames
File : CODER.C	Procedure : Coder()	Performs subframe division

The coder processes the speech by buffering consecutive speech samples, $y[n]$, into frames of 240 samples, $s[n]$. Each frame is divided into two parts of 120 samples for pitch estimation computation. Each part is divided by two again, so that each frame is finally divided into four subframes of 60 samples each.

2.3 High pass filter

File : UTIL_LBC.C	Procedure : Rem_Dc()	Performs high pass filter

This block removes the DC element from the input speech, $s[n]$. The filter transfer function is:

$$H(z) = \frac{1 - z^{-1}}{1 - \frac{127}{128} z^{-1}} \qquad (1)$$

The output of this filter is: $x[n]_{n\,=\,0..239}$.

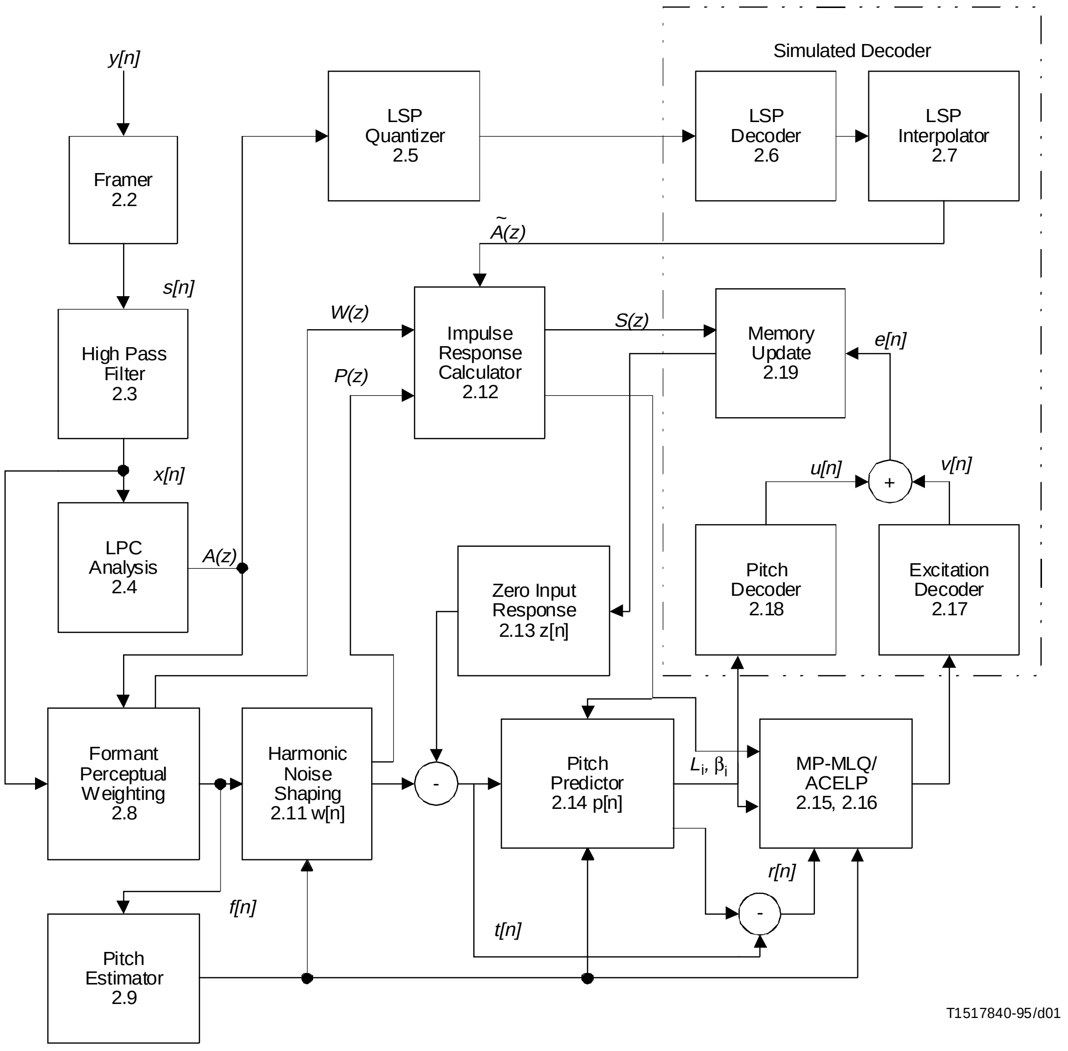

FIGURE 1/G.723.1

Block diagram of the speech coder – For each block the corresponding reference number is indicated

2.4 LPC analysis

File : LPC.C	Procedure : Comp_Lpc()	Performs LPC coefficients calculation
File : LPC.C	Procedure : Durbin()	Levinson-Durbin recursion

The LPC analysis is performed on signal $x[n]$ in the following way. Tenth order Linear Predictive (LP) analysis is used. For each subframe, a window of 180 samples is centered on the subframe. A Hamming window is applied to these samples. Eleven autocorrelation coefficients are computed from the windowed signal. A white noise correction factor of (1025/1024) is applied by using the formula $R[0] = R[0](1 + 1/1024)$. The other 10 autocorrelation coefficients are multiplied by the binomial window coefficients table. (The values for this table and all others are given in the C code.) The Linear Predictive Coefficients (LPC) are computed using the conventional Levinson-Durbin recursion. For every input frame, four LPC sets are computed, one for every subframe. These LPC sets are used to construct the short-term perceptual weighting filter. The LPC synthesis filter is defined as:

$$A_i(z) = \frac{1}{1 - \sum_{j=1}^{10} \boldsymbol{a}_{ij} z^{-j}}, \quad 0 \le i \le 3 \tag{2}$$

where i is the subframe index and is defined to be between 0 and 3.

2.5 LSP quantizer

File : LSP.C	Procedure : AtoLsp()	Converts LPC to LSP coefficients
File : LSP.C	Procedure : LspQnt()	LSP vector quantization
File : LSP.C	Procedure : Lsp_Svq()	LSP sub-vectors quantization

First, a small additional bandwidth expansion (7.5 Hz) is performed. Then the resulting $A_3(z)$ LP filter is quantized using a predictive split vector quantizer. The quantization is performed in the following way:

1) The LP coefficients, $\{a_j\}_{j=1..10}$, are converted to LSP coefficients, $\{\boldsymbol{p}'_j\}_{j=1..10}$, by searching along the unit circle and interpolating for zero crossings.

2) The long term DC component, $\boldsymbol{p}_{DC}$, is removed from the LSP coefficients, $\boldsymbol{p}'$, and a new DC removed LSP vector, $\boldsymbol{p}$, is obtained.

3) A first order fixed predictor, $b = (12/32)$, is applied to the previously decoded LSP vector $\tilde{\boldsymbol{p}}_{n-1}$, to obtain the DC removed predicted LSP vector, $\bar{\boldsymbol{p}}_n$, and the residual LSP error vector, $\boldsymbol{e}_n$ at time (frame) n.

$$\boldsymbol{p}_n^T = [p_{1,n} p_{2,n} \cdots p_{10,n}] \tag{3.1}$$

$$\bar{\boldsymbol{p}}_n^T = [\bar{p}_{1,n} \bar{p}_{2,n} \cdots \bar{p}_{10,n}] \tag{3.2}$$

$$\bar{\boldsymbol{p}}_n = b[\tilde{\boldsymbol{p}}_{n-1} - \boldsymbol{p}_{DC}] \tag{3.3}$$

$$\boldsymbol{e}_n = \boldsymbol{p}_n - \bar{\boldsymbol{p}}_n \tag{3.4}$$

4) The unquantized LSP vector, $\boldsymbol{p}'_n$, the quantized LSP vector, $\tilde{\boldsymbol{p}}_n$, the residual LSP error vector, $\boldsymbol{e}_n$, are divided into 3 sub-vectors with dimension 3, 3 and 4 respectively. Each *m*th sub-vector is vector quantized using an 8 bit codebook. The index, *l*, of the appropriate sub-vector codebook entry that minimizes the error criterion $E_{l,m}$ is selected.

$$\boldsymbol{p}'^T_m = [p'_{1+3m}\ p'_{2+3m} \cdots p'_{K_m+3m}],\ K_m = \begin{cases} 3, & m=0 \\ 3, & m=1 \\ 4, & m=2 \end{cases} \tag{4.1}$$

$$\tilde{\boldsymbol{p}}^T_{l,m} = [\tilde{p}_{1,l,m}\ \tilde{p}_{2,l,m} \cdots \tilde{p}_{K_m,l,m}],\ \begin{matrix} 0 \le m \le 2 \\ 1 \le l \le 256 \end{matrix} \tag{4.2}$$

$$\boldsymbol{p}' = \boldsymbol{p} + \boldsymbol{p}_{DC} \tag{4.3}$$

$$\tilde{\boldsymbol{p}}_{l,m} = \bar{\boldsymbol{p}}_m + \boldsymbol{p}_{DC_m} + \boldsymbol{e}_{l,m},\ \begin{matrix} 0 \le m \le 2 \\ 1 \le l \le 256 \end{matrix} \tag{4.4}$$

$$E_{l,m} = (\boldsymbol{p}'_m - \tilde{\boldsymbol{p}}_{l,m})^T\ \boldsymbol{W}_m\,(\boldsymbol{p}'_m - \tilde{\boldsymbol{p}}_{l,m}),\ \begin{matrix} 0 \le m \le 2 \\ 1 \le l \le 256 \end{matrix} \tag{4.5}$$

where $\boldsymbol{e}_{l,m}$ is the *l*th entry of the *m*th split residual LSP codebook and $\boldsymbol{W}_n$ is a diagonal weighting matrix, determined from the unquantized LSP coefficients vector $\boldsymbol{p}'$, with weights defined by:

$$\begin{aligned} w_{j,j} &= \frac{1}{\min\{p'_j - p'_{j-1},\ p'_{j+1} - p'_j\}},\ 2 \le j \le 9 \\ w_{1,1} &= \frac{1}{p'_2 - p'_1} \\ w_{10,10} &= \frac{1}{p'_{10} - p'_9} \end{aligned} \tag{5}$$

5) The selected indices are transmitted to the channel.

2.6 LSP decoder

File : LSP.C	Procedure: Lsp_Inq()	Inverse quantization of LSP

The decoding of the LSP coefficients is performed in the following way:

1) First, the three sub-vectors, $\{e_{m,n}\}_{m=0..2}$, are decoded to form a tenth order vector, $\tilde{\boldsymbol{e}}_{\boldsymbol{n}}$.

2) The predicted vector, $\bar{\boldsymbol{p}}_n$, is added to the decoded vector, $\tilde{\boldsymbol{e}}_n$, and DC vector, $\boldsymbol{p}_{DC}$, to form the decoded LSP vector, $\tilde{\boldsymbol{p}}_n$.

3) A stability check is performed on the decoded LSP vector, $\tilde{\boldsymbol{p}}_n$, to ensure that the decoded LSP vector is ordered according to the following condition:

$$\tilde{p}_{j+1,n} - \tilde{p}_{j,n} \geq \Delta_{\min}, 1 \leq j \leq 9 \tag{6}$$

$\Delta_{\min}$ is equal to 31.25 Hz. If this stability check (6) fails for $\tilde{p}_i$ and $\tilde{p}_{i+1}$, then $\tilde{p}_j$ and $\tilde{p}_{j+1}$ are modified in the following way:

$$\tilde{p}_{avg} = (\tilde{p}_j + \tilde{p}_{j+1}) / 2 \tag{7.1}$$

$$\tilde{p}_j = \tilde{p}_{avg} - \Delta_{\min} / 2 \tag{7.2}$$

$$\tilde{p}_{j+1} = \tilde{p}_{avg} + \Delta_{\min} / 2 \tag{7.3}$$

The modification is performed until condition (6) is met. If after 10 iterations the condition of stability is not met, the previous LSP vector is used.

2.7 LSP interpolation

File : LSP.C	Procedure : Lsp_Int()	LSP interpolator
File : LSP.C	Procedure : LsptoA()	Converts LSP to LPC coefficients

Linear interpolation is performed between the decoded LSP vector, $\tilde{\boldsymbol{p}}_n$, and the previous LSP vector, $\tilde{\boldsymbol{p}}_{n-1}$, for each subframe. Four interpolated LSP vectors, $\{\tilde{\boldsymbol{p}}_i\}_{i=0..3}$, are converted to LPC vectors, $\{\tilde{\boldsymbol{a}}_i\}_{i=0..3}$.

$$\tilde{\boldsymbol{p}}_{ni} = \begin{cases} 0.75\tilde{\boldsymbol{p}}_{n-1} + 0.25\tilde{\boldsymbol{p}}_n, i = 0 \\ 0.5\tilde{\boldsymbol{p}}_{n-1} + 0.5\tilde{\boldsymbol{p}}_n, i = 1 \\ 0.25\tilde{\boldsymbol{p}}_{n-1} + 0.75\tilde{\boldsymbol{p}}_n, i = 2 \\ \tilde{\boldsymbol{p}}_n, i = 3 \end{cases} \tag{8}$$

$$\tilde{\boldsymbol{a}}_i^T = [\tilde{\boldsymbol{a}}_{i1}\tilde{\boldsymbol{a}}_{i2} \ldots \tilde{\boldsymbol{a}}_{i10}]^T, 0 \leq i \leq 3 \tag{9}$$

The quantized LPC synthesis filter, $\tilde{A}_i(z)$, is used for generating the decoded speech signal and is defined as:

$$\tilde{A}_i(z) = \frac{1}{1 - \sum_{j=1}^{10} \tilde{\boldsymbol{a}}_{ij} z^{-j}}, \ 0 \leq i \leq 3 \tag{10}$$

2.8 Formant perceptual weighting filter

File : LPC.C	Procedure : Wght_Lpc()	Computes perceptual filter coefficients
File : LPC.C	Procedure : Error_Wght()	Applies perceptual weighting filter

For each subframe a formant perceptual weighting filter is constructed, using the unquantized LPC coefficients $\{a_{ij}\}_{j=1,..,10}$. The filter has a transfer function:

$$W_i(z) = \frac{1 - \sum_{j=1}^{10} a_{ij} z^{-j} \gamma_1^j}{1 - \sum_{j=1}^{10} a_{ij} z^{-j} \gamma_2^j}, 0 \le i \le 3 \tag{11}$$

where $\gamma_1 = 0.9$ and $\gamma_2 = 0.5$. The input speech frame, $\{x[n]\}_{n=0..239}$, is then divided to four subframes and each subframe is filtered using the $W_i(z)$ filter, and the weighted output speech signal, $\{f[n]\}_{n=0..239}$ is obtained.

2.9 Pitch estimation

File : EXC_LBC.C	Procedure : Estim_Pitch()	Open loop pitch estimation

Two pitch estimates are computed for every frame, one for the first two subframes and one for the last two. The open loop pitch period estimate, L_{OL}, is computed using the perceptually weighted speech $f[n]$. A crosscorrelation criterion, $C_{OL}(j)$, maximization method is used to determine the pitch period, using the following expression:

$$C_{OL}(j) = \frac{\left(\sum_{n=0}^{119} f[n] \cdot f[n-j]\right)^2}{\sum_{n=0}^{119} f[n-j] \cdot f[n-j]}, \quad 18 \le j \le 142 \tag{12}$$

The index j which maximizes the crosscorrelation, $C_{OL}(j)$, is selected as the open loop pitch estimation for the appropriate two subframes. While searching for the best index, some preference is given to smaller pitch periods to avoid choosing pitch multiples. Maximums of $C_{OL}(j)$ are searched for beginning with $j = 18$. For every maximum $C_{OL}(j)$ found, its value is compared to the best previous maximum found, $C_{OL}(j')$. If the difference between indices j and j' is less than 18 and $C_{OL}(j) > C_{OL}(j')$, the new maximum is selected. If the difference between the indices is greater than or equal to 18, the new maximum is selected only if $C_{OL}(j')$ is greater than $C_{OL}(j)$ by 1.25 dB.

2.10 Subframe processing

From this point on, all the computational blocks are performed on a once per subframe basis.

2.11 Harmonic noise shaping

File : EXC_LBC.C	Procedure : Comp_Pw()	Computes harmonic noise filter coefficients
File : EXC_LBC.C	Procedure : Filt_Pw()	Applies harmonic noise filter

In order to improve the quality of the encoded speech, a harmonic noise shaping filter is constructed. The filter is:

$$P_i(z) = 1 - \beta z^{-L} \tag{13}$$

The optimal lag, L, for this filter is the lag which maximizes the criterion, $C_{PW}(j)$, while considering only positive correlation values for the numerator, $N(j)$, before squaring:

$$N(j) = \sum_{n=0}^{59} f[n] \cdot f[n-j] \tag{14.1}$$

$$C_{PW}(j) = \frac{(N(j))^2}{\sum_{n=0}^{59} f[n-j] \cdot f[n-j]}, L_1 \le j \le L_2 \tag{14.2}$$

where $L_1 = L_{OL} - 3$ and $L_2 = L_{OL} + 3$. The maximum value will be defined as C_L. The optimal filter gain, G_{opt}, is:

$$G_{opt} = \frac{\sum_{n=0}^{59} f[n]\, f[n-L]}{\sum_{n=0}^{59} f[n-L]\, f[n-L]} \tag{15}$$

G_{opt} is limited to the range [0,1]. The energy, E, of the weighted speech signal, $\{f[n]\}_{n=0..59}$ is given by:

$$E = \sum_{n=0}^{59} f^2[n] \tag{16}$$

Then the coefficient β of the harmonic noise shaping filter, $P(z)$, is given by:

$$\beta = \begin{cases} 0.3125 G_{opt}, & \text{if } -10 \log_{10}\left(1 - \frac{C_L}{E}\right) \ge 2.0 \\ 0.0, & \text{otherwise} \end{cases} \tag{17}$$

After computing the harmonic noise filter coefficients, the formant perceptually weighted speech, $f[n]$, is filtered using $P(z)$ to obtain the target vector, $w[n]$.

$$w[n] = f[n] - \beta f[n-L], \quad 0 \le n \le 59 \tag{18}$$

2.12 Impulse response calculator

File : LPC.C	Procedure : Comp_Ir()	Impulse response computation

For closed loop analysis the following combined filter, $S_i(z)$, is used:

$$S_i(z) = \tilde{A}_i(z) \cdot W_i(z) \cdot P_i(z), \quad 0 \le i \le 3 \tag{19}$$

where the components of $S_i(z)$ are defined in the formulas 10, 11 and 13. The impulse response of this filter is computed and will be referred as $\{h_i[n]\}_{n = 0..59,\, i = 0..3}$.

2.13 Zero input response and ringing subtraction

File : LPC.C	Procedure : Sub_Ring ()	Performs ringing subtraction

The zero input response of the combined filter, $S_i(z)$, is obtained by computing the output of that filter when the input signal is all zero-valued samples. The zero input response is denoted $\{z[n]\}_{n = 0..59}$. The ringing subtraction is performed by subtracting the zero input response from the harmonic weighted speech vector, $\{w[n]\}_{n = 0..59}$. The resulting vector is defined as $\boldsymbol{t}[n] = \boldsymbol{w}[n] - \boldsymbol{z}[n]$.

2.14 Pitch predictor

File : EXC_LBC.C	Procedure : Find_Acbk()	Adaptive codebook contribution. Calls Get_Rez() and Decod_Acbk()
File : EXC_LBC.C	Procedure : Get_Rez()	Gets residual from the excitation buffer
File : EXC_LBC.C	Procedure : Decod_Acbk()	Decodes the adaptive codebook contribution

The pitch prediction contribution is treated as a conventional adaptive codebook contribution. The pitch predictor is a fifth order pitch predictor (see equation 41.2). For subframes 0 and 2 the closed loop pitch lag is selected from around the appropriate open loop pitch lag in the range ±1 and coded using 7 bits. (Note that the open loop pitch lag is never transmitted.) For subframes 1 and 3 the closed loop pitch lag is coded differentially using 2 bits and may differ from the previous subframe lag only by −1, 0, +1 or +2. The quantized and decoded pitch lag values will be referred to as L_i from this point on. The pitch predictor gains are vector quantized using two codebooks with 85 or 170 entries for the high bit rate and 170 entries for the low bit rate. The 170 entry codebook is the same for both rates. For the high rate if L_0 is less than 58 for subframes 0 and 1 or if L_2 is less than 58 for subframes 2 and 3, then the 85 entry codebook is used for the pitch gain quantization. Otherwise the pitch gain is quantized using the 170 entry codebook. The contribution of the pitch predictor, $\{p[n]\}_{n = 0..59}$, is subtracted from the target vector $\{t[n]\}_{n = 0..59}$, to obtain the residual signal $\{r[n]\}_{n = 0..59}$.

$$\boldsymbol{r}[n] = \boldsymbol{t}[n] - \boldsymbol{p}[n] \tag{20}$$

2.15 High rate excitation (MP-MLQ)

File : EXC_LBC.C	Procedure : Find_Fcbk()	Fixed codebook contribution
File : EXC_LBC.C	Procedure : Find_Best()	Residual signal quantization
File : EXC_LBC.C	Procedure : Gen_Trn()	Generates a train of Dirac functions
File : EXC_LBC.C	Procedure : Fcbk_Pack()	Combinatorial coding of pulse positions

The residual signal $\{r[n]\}_{n\,=\,0..59}$, is transferred as a new target vector to the MP-MLQ block. This block performs the quantization of this vector. The quantization process is approximating the target vector $r[n]$ by $r'[n]$:

$$r'[n] = \sum_{j=0}^{n} h[j] \cdot v[n-j], \quad 0 \le n \le 59 \tag{21}$$

where $v[n]$ is the excitation to the combined filter $S(z)$ with impulse response $h[n]$ and defined as:

$$v[n] = G \sum_{k=0}^{M-1} \alpha_k \, d[n - m_k], \quad 0 \le n \le 59 \tag{22}$$

where G is the gain factor, $\delta[n]$ is a Dirac function, $\{\alpha_k\}_{k=0..M-1}$ and $\{m_k\}_{k=0..M-1}$ are the signs (± 1) and the positions of Dirac functions respectively and M is the number of pulses, which is 6 for even subframes and is 5 for odd subframes. There is a restriction on pulses positions. The positions can be either all odd or all even. This will be indicated by a grid bit. So, the problem is to estimate the unknown parameters, G, $\{\alpha_k\}_{k\,=\,0..M\,-\,1}$, and $\{m_k\}_{k\,=\,0..M\,-\,1}$, that minimize the mean square of the error signal $err[n]$:

$$err[n] = r[n] - r'[n] = r[n] - G \sum_{k=0}^{M-1} \alpha_k \, h[n - m_k] \tag{23}$$

The parameters estimation and quantization processes are based on an analysis-by-synthesis method. The G_{max} parameter is estimated and quantized as follows. First the crosscorrelation function $d[j]$ between the impulse response, $h[n]$, and the new target vector, $r[n]$, is computed:

$$d[j] = \sum_{n=j}^{59} r[n] \cdot h[n-j], \; 0 \le j \le 59 \tag{24}$$

The estimated gain is given by:

$$G_{\max} = \frac{\max\{\,|\,d[j]\,|\,\}_{j=0..59}}{\sum_{n=0}^{59} h[n] \cdot h[n]} \tag{25}$$

Then the estimated gain G_{max} is quantized by a logarithmic quantizer. This scalar gain quantizer is common to both rates and consists of 24 steps, of 3.2 dB each. Around this quantized value, $\tilde{G}_{max}$, additional gain values are selected within the range $[\tilde{G}_{max} - 3.2, \tilde{G}_{max} + 6.4]$. For each of these gain values the signs and locations of the pulses are sequentially optimized. This procedure is repeated for both the odd and even grids. Finally the combination of the quantized parameters that yields the minimum mean square of $err[n]$ is selected. The optimal combination of pulse locations and gain is transmitted. $\binom{30}{M}$ combinatorial coding is used to transmit the pulse locations. Furthermore, using the fact that the number of codewords in the fixed codebooks is not a power of 2, 3 additional bits are saved by combining the 4 most significant bits of the combinatorial codes for the 4 subframes to form a 13-bit index. The C code provides the details on how this information is packed.

To improve the quality of speech with a short pitch period, the following additional procedure is used. If L_0 is less than 58 for subframes 0 and 1 or if L_2 is less than 58 for subframes 2 and 3, a train of Dirac functions with the period of the pitch index, L_0 or L_2, is used for each location m_k instead of a single Dirac function in the above quantization procedure. The choice between a train of Dirac functions or a single Dirac function to represent the residual signal is made based on the mean square error computation. The configuration which yields the lowest mean square error is selected and its parameter indices are transmitted.

2.16 Low rate excitation (ACELP)

File : EXC_LBC.C	Procedure : search_T0()	Pitch synchronous excitation
File : EXC_LBC.C	Procedure : ACELP_LBC_code()	Computes innovative vector
File : EXC_LBC.C	Procedure : Cor_h()	Correlations of impulse response
File : EXC_LBC.C	Procedure : Cor_h_X()	Correlation of target vector with impulse response
File : EXC_LBC.C	Procedure : D4i64_LBC()	Algebraic codebook search
File : EXC_LBC.C	Procedure : G_code()	Computes innovation vector gain

A 17-bit algebraic codebook is used for the fixed codebook excitation $v[n]$. Each fixed codevector contains, at most, four non-zero pulses. The 4 pulses can assume the signs and positions given in Table 1:

TABLE 1/G.723.1

ACELP excitation codebook

Sign	Positions
±1	0, 8, 16, 24, 32, 40, 48, 56
±1	2, 10, 18, 26, 34, 42, 50, 58
±1	4, 12, 20, 28, 36, 44, 52, (60)
±1	6, 14, 22, 30, 38, 46, 54, (62)

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 pulse is not present.

Each pulse position is encoded with 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.

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

$$E_\xi = ||\boldsymbol{r} - \mathrm{G}\boldsymbol{H}\boldsymbol{v}_\xi||^2 \tag{26}$$

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; $\boldsymbol{v}_\xi$ is the algebraic codeword at index ξ; and $\mathbf{H}$ is a lower triangular Toeplitz convolution matrix with diagonal $h(0)$ and lower diagonals $h(1),..., h(L-1)$, with $h(n)$ being the impulse response of the weighted synthesis filter $S_i(z)$.

It can be shown that the optimum codeword is the one which maximizes the term:

$$\tau_\xi = \frac{C_\xi^2}{\varepsilon_\xi} = \frac{(\boldsymbol{d}^T \boldsymbol{v}_\xi)^2}{\boldsymbol{v}_\xi^T \Phi \boldsymbol{v}_\xi} \tag{27}$$

where $\boldsymbol{d} = \boldsymbol{H}^T \boldsymbol{r}$ is the correlation between the target vector signal, $r[n]$, and the impulse response, $h(n)$, and $\boldsymbol{\Phi} = \boldsymbol{H}^T \boldsymbol{H}$ is the covariance matrix of the impulse response. The vector $\boldsymbol{d}$ and the matrix $\boldsymbol{\Phi}$ are computed prior to the codebook search. The elements of the vector $\boldsymbol{d}$ are computed by:

$$d(j) = \sum_{n=j}^{59} r[n] \cdot h[n-j], \quad 0 \le j \le 59 \tag{28}$$

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

$$\Phi(i,j) = \sum_{n=j}^{59} h[n-i] \cdot h[n-j], \quad \begin{matrix} j \ge i \\ 0 \le i \le 59 \end{matrix} \tag{29}$$

NOTE – Only the elements actually needed are computed and an efficient storage has been designed that speeds the search procedure up.

The algebraic structure of the codebook allows for very fast search procedures since the excitation vector $\boldsymbol{v}_\xi$ contains only 4 non-zero pulses. The search is performed in 4 nested loops, corresponding to each pulse positions, where in each loop the contribution of a new pulse is added. The correlation in equation (27) is given by:

$$C = \alpha_0 d[m_0] + \alpha_1 d[m_1] + \alpha_2 d[m_2] + \alpha_3 d[m_3] \tag{30}$$

where m_k is the position of the kth pulse and α_k is its sign (±1). The energy for even pulse position codevectors in equation (27) is given by:

$$\begin{aligned} \varepsilon = \; & \Phi(m_0, m_0) \\ + \; & \Phi(m_1, m_1) + 2\alpha_0\alpha_1\Phi(m_0, m_1) \\ + \; & \Phi(m_2, m_2) + 2[\alpha_0\alpha_2\Phi(m_0, m_2) + \alpha_1\alpha_2\Phi(m_1, m_2)] \\ + \; & \Phi(m_3, m_3) + 2[\alpha_0\alpha_3\Phi(m_0, m_3) + \alpha_1\alpha_3\Phi(m_1, m_3) + \alpha_2\alpha_3\Phi(m_2, m_3)] \end{aligned} \tag{31}$$

For odd pulse position codevectors, the energy in equation (27) 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 $\boldsymbol{\Phi}(m_1, m_2)$ 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.

$$\begin{aligned} & s[2j] = s[2j+1] = sign(d[2j]) && \text{if } |d[2j]| > |d[2j+1]| \\ & s[2j] = s[2j+1] = sign(d[2j+1]) && \text{otherwise} \end{aligned} \tag{32}$$

and the signal d' is given by $d'[j] = d[j]s[j]$. Second, the matrix $\boldsymbol{\Phi}$ is modified by including the signal information; that is, $\boldsymbol{\Phi}'(i, j) = s[i]s[j]\boldsymbol{\Phi}(i, j)$.The correlation in equation (30) is now given by:

$$C = d'[m_0] + d'[m_1] + d'[m_2] + d'[m_3] \tag{33}$$

and the energy in equation (31) is given by:

$$\begin{aligned}\varepsilon \; = \; & \Phi'(m_0, m_0) \\ + \; & \Phi'(m_1, m_1) + 2\Phi'(m_0, m_1) \\ + \; & \Phi'(m_2, m_2) + 2[\Phi'(m_0, m_2) + \Phi'(m_1, m_2)] \\ + \; & \Phi'(m_3, m_3) + 2[\Phi'(m_0, m_3) + \Phi'(m_1, m_3) + \Phi'(m_2, m_3)]\end{aligned} \qquad (34)$$

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 prior to the codebook search. The threshold is given by:

$$thr_3 = av_3 + (max_3 - av_3) / 2 \qquad (35)$$

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 subframes) is not allowed to exceed 600. (The average worst case per subframe is 150 times. This can be viewed as searching only 150×8 entries of the codebook, ignoring the overhead of the first three loops.)

A special feature of the codebook is that, for pitch delays less than 60, a pitch contribution depending on the index $PGInd_i$ of the LTP pitch predictor gain vector is added to the code. That is, after the optimum algebraic code $v[n]$ is determined, it is modified by $v[n] \leftarrow v[n] + b(PGInd_i)v[n - L_i - e(PGInd_i)]$, the values $\beta(PGInd_i)$ and $\varepsilon(PGInd_i)$ are tabulated and L_i being the integer pitch period. Note that prior to the codebook search, the impulse response should be modified in a similar fashion if $L_i < 60$.

The last step, after getting the $v[n]$ sequence, is quantizing the gain G. The gain is quantized in the same way as in the high rate excitation. This is done by stepping through the gain quantization table and selecting the index $MGInd_i$ which minimizes the following expression: $|G - \tilde{G}_j|$, $0 \le j \le 23$.

2.17 Excitation decoder

File : EXC_LBC.C	Procedure : Fcbk_Unpk()	Decode fixed codebook excitation

The decoding of pulses is performed as follows:

1) First the maximum gain G_{max} index is derived using:

$$MGInd_i = GInd_i - PGInd_i \cdot GSize \qquad (36)$$

where $GSize = 24$ is the size of the $\tilde{\boldsymbol{G}}$ table and $PGInd_i$ is obtained in 2.18.

2) The positions of the pulses are decoded using $\binom{30}{M}$ combinatorial decoding where M is either 6 or 5, for the high rate. For the low rate, direct decoding of the position indices is performed.

3) The grid position (even/odd) is derived from the grid bit.

4) The pulse signs are derived from the sign bits.

5) For the high rate coder, the decoding of the pulse train bit is performed only if $L_i < 58$.

6) Then the $v[n]$ vector is reconstructed using the decoded parameters.

7) Finally, the pitch contribution, $u[n]$, and the pulse contributions, $v[n]$, are summed together to form the excitation vector $e[n]$.

2.18 Decoding of the pitch information

File : EXC_LBC.C	Procedure : Get_Rez()	Gets residual from the excitation buffer
File : EXC_LBC.C	Procedure : Decod_Acbk()	Decodes the adaptive codebook contribution

The decoding of pitch information is performed as described below:

1) First, the lag of pitch predictor for even subframes is decoded:

$$L_i = PInd_i + 18, \ i = 0,2 \tag{37}$$

2) The lag of pitch predictor for odd subframes is decoded as follows:

$$L_i = L_{i-1} + \Delta_i, \ i = 1,3 \tag{38}$$

where $\Delta_i \in \{-1,0,+1,+2\}$.

3) The gain vector of the pitch predictor in *ith* subframe is derived from the gain index $GInd_i$. For the low rate this index contains the information about the pitch predictor gain vector and the index of the gain of the pulse sequence. In this case pitch gain index $PGInd_i$ is derived as follows:

$$PGInd_i = \lfloor GInd_i / GSize \rfloor, \ i = 0..3 \tag{39}$$

where $\lfloor x \rfloor$ indicates the greatest integer $\le x$. For the high rate, in the case that the condition $L_i \ge 58$ is met, this index is derived in the same manner as described in (39). In the above cases $PGInd_i$ is a pointer to 170 entries gain vector codebook. Otherwise, this index is a pointer to 85 entries gain vector codebook and contains an additional information about the impulse train bit. In this case pitch gain index is derived as follows:

$$PGInd_i = \lfloor GInd_i \,\&0x7FF / GSize \rfloor, \ i = 0..3 \tag{40}$$

The pitch predictor lag and gain vector are decoded from these indices and utilized for the pitch contribution $u[n]$ extraction as described below. First, a signal $e'[n]$ is defined by:

$$\begin{aligned} e'[0] &= e[-L_i - 2] \\ e'[1] &= e[-L_i - 1] \\ e'[n] &= e[(n \bmod L_i) - L_i], \ 2 \le n \le 63 \end{aligned} \tag{41.1}$$

where mod stands for the modulus operation. Then,

$$u[n] = \sum_{j=0}^{j=4} \beta_{ij}\, e'[n + j], \ 0 \le n \le 59 \tag{41.2}$$

2.19 Memory update

File : LPC.C	Procedure : Upd_Ring ()	Memory update

The last task of the *ith* subframe before proceeding to encode the next subframe is to update the memories of the synthesis filter $\tilde{A}_i(z)$, the formant perceptual weighting filter $W_i(z)$, and the harmonic noise shaping filter $P_i(z)$. To accomplish this, the complete response of combined filter $S_i(z)$, is computed by passing the reconstructed excitation sequence through this filter. At the end of the excitation filtering, the memory of the combined filter is saved and will be used to compute the zero input response during the encoding of the next speech vector.

2.20 Bit allocation

File : UTIL_LBC.C	Procedure : Line_Pack()	Bitstream packing

This subclause presents the bit allocation tables for both high and low bit rates. The major differences between two rates are in the pulse positions and amplitudes coding. Also, at the lower rate 170 codebook entries are always used for the gain vector of the long term predictor. See Tables 2, 3 and 4.

TABLE 2/G.723.1

Bit allocation of the 6.3 kbit/s coding algorithm

Parameters coded	Subframe 0	Subframe 1	Subframe 2	Subframe 3	Total
LPC indices					24
Adaptive codebook lags	7	2	7	2	18
All the gains combined	12	12	12	12	48
Pulse positions	20	18	20	18	73 (Note)
Pulse signs	6	5	6	5	22
Grid index	1	1	1	1	4
Total:					189
NOTE – By using the fact that the number of codewords in the fixed codebook is not a power of 2, 3 additional bits are saved by combining the 4 MSB of each pulse position index into a single 13-bit word.					

TABLE 3/G.723.1

Bit allocation of the 5.3 kbit/s coding algorithm

Parameters coded	Subframe 0	Subframe 1	Subframe 2	Subframe 3	Total
LPC indices					24
Adaptive codebook lags	7	2	7	2	18
All the gains combined	12	12	12	12	48
Pulse positions	12	12	12	12	48
Pulse signs	4	4	4	4	16
Grid index	1	1	1	1	4
Total:					158

TABLE 4/G.723.1

List of transmitted parameters

Name	Transmitted parameters	High rate # bits	Low rate # bits
LPC	LSP VQ index	24	24
ACL0	Adaptive CodeBook Lag	7	7
ACL1	Differential Adaptive CodeBook Lag	2	2
ACL2	Adaptive CodeBook Lag	7	7
ACL3	Differential Adaptive CodeBook Lag	2	2
GAIN0	Combination of adaptive and fixed gains	12	12
GAIN1	Combination of adaptive and fixed gains	12	12
GAIN2	Combination of adaptive and fixed gains	12	12
GAIN3	Combination of adaptive and fixed gains	12	12
POS0	Pulse positions index	20 (Note)	12
POS1	Pulse positions index	18 (Note)	12
POS2	Pulse positions index	20 (Note)	12
POS3	Pulse positions index	18 (Note)	12
PSIG0	Pulse sign index	6	4
PSIG1	Pulse sign index	5	4
PSIG2	Pulse sign index	6	4
PSIG3	Pulse sign index	5	4
GRID0	Grid index	1	1
GRID1	Grid index	1	1
GRID2	Grid index	1	1
GRID3	Grid index	1	1
NOTE – The 4 MSB of these codewords are combined to form a 13-bit index, MSBPOS.			

2.21 Coder initialization

File : CODER.C	Procedure : Init_Coder()	Coder initialization

All the static coder variables should be initialized to 0, with the exception of the previous LSP vector, which should be initialized to LSP DC vector, $\boldsymbol{p}_{DC}$.

3 Decoder principles

3.1 General description

The decoder operation is also performed on a frame-by-frame basis. First the quantized LPC indices are decoded, then the decoder constructs the LPC synthesis filter. For every subframe, both the adaptive codebook excitation and fixed codebook excitation are decoded and input to the synthesis filter. The adaptive postfilter consists of a formant and a forward-backward pitch postfilter. The excitation signal is input to the pitch postfilter, which in turn is input to the synthesis filter whose output is input to the formant postfilter. A gain scaling unit maintains the energy at the input level of the formant postfilter.

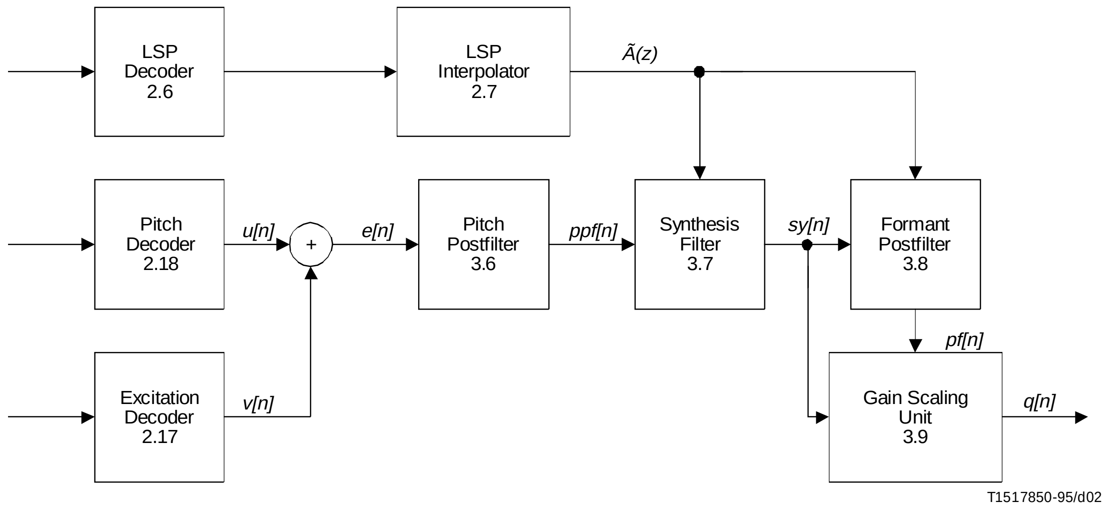

FIGURE 2/G.723.1

Block diagram of the speech decoder – For each block the corresponding reference number is indicated

3.2 LSP decoder

File : LSP.C	Procedure : Lsp_Inq()	Inverse quantization of LSP

This block is the same as in 2.6.

3.3 LSP interpolator

File : LSP.C	Procedure : Lsp_Int()	LSP interpolator
File : LSP.C	Procedure : LsptoA()	Converts LSP to LPC coefficients

This block is the same as in 2.7.

3.4 Decoding of the pitch information

File : EXC_LBC.C	Procedure : Get_Rez()	Gets residual from the excitation buffer
File : EXC_LBC.C	Procedure : Decod_Acbk()	Decodes the adaptive codebook contribution

This block is the same as in 2.18.

3.5 Excitation decoder

File : EXC_LBC.C	Procedure : Fcbk_Unpk()	Decode fixed codebook excitation

This block is the same as in 2.17.

3.6 Pitch postfilter

File : EXC_LBC.C	Procedure : Comp_Lpf()	Computes pitch postfilter parameters
File : EXC_LBC.C	Procedure : Find_F()	Forward crosscorrelation maximization
File : EXC_LBC.C	Procedure : Find_B()	Backward crosscorrelation maximization
File : EXC_LBC.C	Procedure : Get_Ind()	Gain computation
File : EXC_LBC.C	Procedure : Filt_Lpf()	Pitch postfiltering

A pitch postfilter is used to improve the quality of the synthesized signal. It is important to note that the pitch postfilter is performed for every subframe, and to implement it, it is required that the whole frame excitation signal $\{e[n]\}_{n\,=\,0..239}$ is generated and saved. The quality improvement is obtained by increasing the SNR at multiples of the pitch period. This is done in the following way. The postfiltered signal $\{ppf[n]\}_{n\,=\,0..59}$ is obtained from the decoded excitation signal $\{e[n]\}_{n\,=\,0..59}$ as given by the following expression:

$$\begin{aligned} ppf[n] &= g_p \cdot \{e[n] + g_{ltp}\,(w_f \cdot g_f \cdot e[n + M_f] + w_b \cdot g_b \cdot e[n - M_b])\} \\ &= g_p \cdot ppf'[n] \end{aligned} \tag{42}$$

where $e[n]$ is the decoded excitation signal. The computation of the gains, g_p, g_f, g_b, and the delays, M_f, M_b, is based on a forward and backward crosscorrelation analysis. The weights w_f, w_b may have the following values: (0,0), (0,1), and (1,0). The delays are selected by maximizing the crosscorrelations. The crosscorrelation for the forward pitch lag is given by:

$$C_f = \sum_{n=0}^{59} e[n]e[n + M_f],\ M_1 \le M_f \le M_2 \tag{43.1}$$

and the crosscorrelation for the backward pitch lag is given by:

$$C_b = \sum_{n=0}^{59} e[n]e[n - M_b],\ M_1 \le M_b \le M_2 \tag{43.2}$$

where $M_1 = L_i - 3$ and $M_2 = L_i + 3$ and $\{L_i\}_{i\ =\ 0.2}$ are the received pitch lags for the first and third subframes. L_0 is utilized for the first two subframes and L_2 for the last two. Note that if one of the maximum correlations is negative or if for some $n \in [0...59]$ there is no sample value $e[n + M_f]$ available, then the corresponding weight and delay are set to 0. This makes four possible cases: (0) both maxima are negative and no pitch postfilter weights need to be computed, (1) only the forward maximum is positive, so it is selected, (2) only the backward maximum is positive, so it is selected, or (3) both maxima are positive and the one making the larger contribution is selected. This procedure is described below. For cases (1), (2) and (3), the relevant signal energies (T_{en}, D_f and/or D_b) for the optimum pitch lag (M_f *or* M_b) are computed according to equations (44.1), (44.2) and (44.3):

$$D_f = \sum_{n=0}^{59} e[n + M_f]\, e[n + M_f] \tag{44.1}$$

$$D_b = \sum_{n=0}^{59} e[n - M_b]\, e[n - M_b] \tag{44.2}$$

$$T_{en} = \sum_{n=0}^{59} e[n]\, e[n] \tag{44.3}$$

The forward energy is given by:

$$E_f = \sum_{n=0}^{59} (e[n] - g_f\, e[n + M_f])^2 \tag{45.1}$$

and the backward energy is given by:

$$E_b = \sum_{n=0}^{59} (e[n] - g_b\, e[n - M_b])^2 \tag{45.2}$$

E is minimized by maximizing $\frac{C^2}{D}$. In case (3), the selection between forward and backward is made by selecting the larger of $\frac{C^2{}_f}{D_f}$ and $\frac{C^2{}_b}{D_b}$. The prediction gain is equal to $-10 \log_{10}\left(1 - \frac{C^2}{DT_{en}}\right)$. If this gain is less than 1.25 dB, then the contribution is judged to be negligible and no pitch postfilter is used. If a pitch postfilter is used, then the optimum gain is given by:

$$g = \frac{C}{D} \tag{46}$$

According to the speech coder bit rate, the optimum gain is multiplied by a weighting factor, γ_{ltp}, which equals 0.1875 for the high rate and 0.25 for the low rate. Finally, the scaling gain, g_p, is computed as:

$$g_p = \sqrt{\frac{\sum_{n=0}^{59} e^2[n]}{\sum_{n=0}^{59} (ppf'[n])^2}} \tag{47}$$

If the denominator in equation (47) is less than the numerator, the gain is set to 1.

3.7 LPC synthesis filter

File : LPC.C	Procedure : Synt()	Synthesizes reconstructed speech

The 10th order LPC synthesis filter $\tilde{A}_i(z)$ is used to synthesize the speech signal $sy[n]$ from the decoded pitch postfiltered residual $ppf[n]$.

$$sy[n] = ppf[n] + \sum_{j=1}^{10} \tilde{\boldsymbol{a}}_{ij}\, sy[n-j] \tag{48}$$

3.8 Formant postfilter

File : LPC.C	Procedure : Spf()	Formant postfiltering
File : UTIL_LBC.C	Procedure : Comp_En()	Computes synthesized signal energy

A conventional ARMA postfilter is used. The transfer function of the short term postfilter is given by the following equations:

$$k = \frac{\sum_{n=1}^{59} sy[n]sy[n-1]}{\sum_{n=0}^{59} sy[n]sy[n]} \tag{49.1}$$

$$k_1 = \frac{3}{4} k_{1old} + \frac{1}{4} k \tag{49.2}$$

$$F(z) = \frac{1 - \sum_{i=1}^{10} \tilde{\boldsymbol{a}}_i \lambda^i{}_1 z^{-i}}{1 - \sum_{i=1}^{10} \tilde{\boldsymbol{a}}_i \lambda^i{}_2 z^{-i}} (1 - 0.25 k_1 z^{-1}) \tag{49.3}$$

where $\lambda_1 = 0.65$ and $\lambda_2 = 0.75$, k is the first autocorrelation coefficient that is estimated from the synthesized speech $sy[n]$, and k_{1old} is the value of k_1 from the previous subframe. The postfiltered signal $pf[n]$ is obtained as the output of the formant postfilter with input signal $sy[n]$.

3.9 Gain scaling unit

File : UTIL_LBC.C	Procedure : Scale()	Gain adjustment of postfiltered signal

This unit receives two input vectors, the synthesized speech vector $\{sy[n]\}_{n\ =\ 0..59}$ and the postfiltered output vector $\{pf[n]\}_{n\ =\ 0..59}$. First the amplitude ratio g_s is computed, using:

$$g_s = \sqrt{\frac{\sum_{n=0}^{59} sy^2[n]}{\sum_{n=0}^{59} pf^2[n]}} \tag{50}$$

If the denominator is equal to 0, g_s is set to 1.

Then the output vector $q[n]$ is obtained by scaling the postfiltered signal $pf[n]$ and the gain $g[n]$ is updated using the following expressions respectively:

$$g[n] = (1 - \alpha)g[n - 1] + \alpha g_s \tag{51}$$

$$q[n] = pf[n] \cdot g[n] \cdot (1 + \alpha) \tag{52}$$

where α is equal to 1/16.

3.10 Frame interpolation handling

File : EXC_LBC.C	Procedure : Comp_Info()	Computes interpolation index
File : EXC_LBC.C	Procedure : Regen()	Current frame regeneration

This coder has been designed to be robust for indicated frame erasures. An error concealment strategy for frame erasures has been included in the decoder. However, this strategy must be triggered by an external indication that the bitstream for the current frame has been erased. Because the coder was designed for burst errors, there is no error correction mechanism provided for random bit errors. If a frame erasure has occurred, the decoder switches from regular decoding to frame erasure concealment mode. The frame interpolation procedure is performed independently for the LSP coefficients and the residual signal.

3.10.1 LSP interpolation

The decoding of the LSP coefficients in frame interpolation mode is performed in the following way:

1) The vector $\tilde{\boldsymbol{e}}_{\boldsymbol{n}}$ is set to zero.

2) The predicted vector, $\bar{\boldsymbol{p}}_{\boldsymbol{n}}$, is added to the vector, $\tilde{\boldsymbol{e}}_{\boldsymbol{n}}$, and DC vector, $\boldsymbol{p}_{\mathrm{DC}}$, to form the decoded LSP vector, $\tilde{\boldsymbol{p}}_{\boldsymbol{n}}$. For $\bar{\boldsymbol{p}}_{\boldsymbol{n}}$ generation a different fixed predictor is used: $b_e = 23/32$.

From this point the decoding of the LSP continues as in 2.6, except that $\Delta_{\min} = 62.5$ Hz, rather than 31.25 Hz.

3.10.2 Residual interpolation

The residual interpolation is performed in two different ways, depending on the last previous good frame prior to the erased frame. The frame is checked with a voiced/unvoiced classifier.

The classifier is based on a cross-correlation maximization function. The last 120 samples of the frame are cross-correlated with $L_2 \pm 3$. The index which reaches the maximum correlation value, is chosen as the interpolation index candidate. Then the prediction gain of the best vector is tested. If this gain is more than 0.58 dB the frame is declared as voiced, otherwise the frame is declared as unvoiced.

The classifier returns 0 for the unvoiced case and the estimated pitch value for the voiced case. If the frame was declared unvoiced, the average of the gain indices for subframes 2 and 3 is saved.

If the current frame was marked as erased, and the previous frame was classified as unvoiced, the current frame excitation is generated using a uniform random number generator. The random number generator output is scaled using the previously computed gain value.

In the voiced case, the current frame is regenerated with periodic excitation having a period equal to the value provided by the classifier.

If the frame erasure state continues for the next two frames, the regenerated vector is attenuated by an additional 2.5 dB for each frame. After 3 interpolated frames, the output is muted completely.

3.11 Decoder initialization

File : DECOD.C	Procedure : Init_Decod()	Decoder initialization

All the static decoder variables should be initialized to 0, with the exception of:

1) Previous LSP vector, which should be initialized to LSP DC vector, $\boldsymbol{p}_{DC}$.

2) Postfilter gain $g[-1]$, which should be initialized to 1.

4 Bitstream packing

NOTE – Each bit of transmitted parameters is named PAR(x)_By: where PAR is the name of the parameter and x indicates the subframe index if relevant and y stands for the bit position starting from 0 (LSB) to the MSB. The expression PARx_By..PARx_Bz stands for the range of transmitted bits from bit y to bit z. The unused bit is named UB (value = 0). RATEFLAG_B0 tells whether the high rate (0) or the low rate (1) is used for the current frame. VADFLAG_B0 tells whether the current frame is active speech (0) or non-speech (1). The combination of RATEFLAG and VADFLAG both being set to 1 is reserved for future use. Octets are transmitted in the order in which they are listed in Tables 5 and 6. Within each octet, the bits are with the MSB on the left and the LSB on the right.

5 ANSI C code

ANSI C code simulating the dual rate encoder/decoder in 16-bit fixed point arithmetic is available from ITU-T. Table 7 lists all the files included in this code.

6 Glossary

This is a glossary containing the mathematical symbols used in the text and a brief description of what they represent. See Table 8.

TABLE 5/G.723.1

Octet bit packing for the high bit rate codec

High rate

Transmitted octets	PAR*x* B*y*, ...
1	LPC_B5...LPC_B0, VADFLAG_B0, RATEFLAG_B0
2	LPC_B13...LPC_B6
3	LPC_B21...LPC_B14
4	ACL0_B5...ACL0_B0, LPC_B23, LPC_B22
5	ACL2_B4...ACL2_B0, ACL1_B1, ACL1_B0, ACL0_B6
6	GAIN0_B3...GAIN0_B0, ACL3_B1, ACL3_B0, ACL2_B6, ACL2_B5
7	GAIN0_B11...GAIN0_B4
8	GAIN1_B7...GAIN1_B0
9	GAIN2_B3...GAIN2_B0, GAIN1_B11...GAIN1_B8
10	GAIN2_B11...GAIN2_B4
11	GAIN3_B7...GAIN3_B0
12	GRID3_B0, GRID2_B0, GRID1_B0, GRID0_B0, GAIN3_B11...GAIN3_B8
13	MSBPOS_B6...MSBPOS_B0, UB
14	POS0_B1. POS0_B0, MSBPOS_B12...MSBPOS_B7
15	POS0_B9...POS0_B2
16	POS1_B2, POS1_B0, POS0_B15...POS0_B10
17	POS1_B10...POS1_B3
18	POS2_B3...POS2_B0, POS1_B13...POS1_B11
19	POS2_B11...POS2_B4
20	POS3_B3...POS3_B0, POS2_B15...POS2_B12
21	POS3_B11...POS3_B4
22	PSIG0_B5...PSIG0_B0, POS3_B13, POS3_B12
23	PSIG2_B2...PSIG2_B0, PSIG1_B4...PSIG1_B0
24	PSIG3_B4...PSIG3_B0, PSIG2_B5...PSIG2_B3

TABLE 6/G.723.1

Octet bit packing for the low bit rate codec

Low rate

Transmitted octets	PAR*x* B*y*, ...
1	LPC_B5...LPC_B0, VADFLAG_B0, RATEFLAG_B0
2	LPC_B13...LPC_B6
3	LPC_B21...LPC_B14
4	ACL0_B5...ACL0_B0, LPC_B23, LPC_B22
5	ACL2_B4...ACL2_B0, ACL1_B1, ACL1_B0, ACL0_B6
6	GAIN0_B3...GAIN0_B0, ACL3_B1, ACL3_B0, ACL2_B6, ACL2_B5
7	GAIN0_B11...GAIN0_B4
8	GAIN1_B7...GAIN1_B0
9	GAIN2_B3...GAIN2_B0, GAIN1_B11...GAIN1_B8
10	GAIN2_B11...GAIN2_B4
11	GAIN3_B7...GAIN3_B0
12	GRID3_B0, GRID2_B0, GRID1_B0, GRID0_B0, GAIN3_B11...GAIN3_B8
13	POS0_B7...POS0_B0
14	POS1_B3...POS1_B0, POS0_B11...POS0_B8
15	POS1_B11...POS1_B4
16	POS2_B7...POS2_B0
17	POS3_B3...POS3_B0, POS2_B11...POS2_B8
18	POS3_B11...POS3_B4
19	PSIG1_B3...PSIG1_B0, PSIG0_B3...PSIG0_B0
20	PSIG3_B3...PSIG3_B0, PSIG2_B3...PSIG2_B0

TABLE 7/G.723.1

List of software filenames

File name	Description
TYPEDEF.H	Data type definition is machine dependent
CST_LBC.H	Definition of constants for G.723
LBCCODEC.C	Main program for G.723 speech codecs
LBCCODEC.H	Functions prototypes
CODER.C	G.723 speech encoder for the two-bit rates
CODER.H	Functions prototypes
DECOD.C	G.723 speech decoder for the two-bit rates
DECOD.H	Functions prototypes
LPC.C	Linear predictive analysis
LPC.H	Functions prototypes
LSP.C	Line spectral pair related functions, quantizer
LSP.H	Functions prototypes
EXC_LBC.C	Adaptive and fixed (MP-MLQ, ACELP) excitation
EXC_LBC.H	Functions prototypes
UTIL_LBC.C	Miscellaneous functions (HPF, pack, unpack, I/O…)
UTIL_LBC.H	Functions prototypes
TAB_LBC.C	Tables of constants
TAB_LBC.H	External declaration for constant tables
BASOP.C	Fixed point arithmetic and logical operation
BASOP.H	Functions prototypes

TABLE 8/G.723.1

Glossary of symbols in the text

Symbol	Description
$y[j]$	Input speech samples
$s[j]$	Input speech frame of 240 samples
$x[j]$	High pass filtered speech frame
$R[j]$	Autocorrelation function, n = 0,1,...,10
a_i	LPC coefficient vector of subframe i
$\tilde{a}_i$	Quantized LPC coefficient vector of subframe i
p'	Unquantized LSP vector
p	DC-removed LSP vector
P_{DC}	Long-term DC vector of LSP values
$\tilde{p}_n$	Decoded LSP vector for frame n
$\bar{p}_n$	DC-removed predicted LSP vector
e_n	Residual LSP error vector for frame n
$\tilde{e}_n$	Quantized value of e_n
W_n	Diagonal weighting matrix for LSP quantization
γ_1, γ_2	Weights for perceptual weighting filter, 0.9, 0.5
W_i	Formant perceptual weighting filter for subframe i
$f[n]$	Formant perceptually weighted speech
L_{OL}	Open loop pitch period estimate
$C_{OL}(j)$	Open loop pitch estimate crosscorrelation criterion function
$C_{PW}(j)$	Harmonic noise shaping pitch estimate crosscorrelation criterion function
β	Harmonic noise shaping filter gain
L	Optimal lag for harmonic noise shaping filter
G_{opt}	Optimal gain for harmonic noise shaping filter
E	Energy of weighted speech signal
P_i	Harmonic noise shaping filter for subframe i
S_i	Combined harmonic and formant weighting and synthesis filters for subframe i
$h[n]$	Impulse response of combined filter
$z[n]$	Zero input response of combined filter
$w[n]$	Harmonic noise weighted speech
$t[n]$	Target vector
$p[n]$	Pitch predictor contribution vector
$r[n]$	Residual signal vector
$r'[n]$	Filtered excitation vector
$v[n]$	Fixed codebook excitation vector
M	Number of pulses

TABLE 8/G.723.1 (*end*)

Glossary of symbols in the text

α_k	Sign of pulse k
m_k	Position of pulse k
$d[n]$	Crosscorrelation function of $h[n]$ and $r[n]$
G_{max}	Estimated gain of pulses for high rate
L_i	Pitch lag for subframe i
H	Lower triangular Toeplitz convolution matrix with diagonals $h[n]$
Φ	Covariance matrix formed by $\boldsymbol{H}^T \boldsymbol{H}$
max_3	Maximum correlation of 1st 3 pulses for low rate
av_3	Average correlation of 1st 3 pulses for low rate
thr_3	Threshold for correlation of 1st 3 pulses for low rate
$MGInd_i$	Maximum gain index of subframe i
$GInd_i$	Gain index of subframe i
$PInd_i$	Pitch index of subframe i
$PGInd_i$	Pitch lag index of subframe i
$GSize$	Size of excitation gain codebook, 24
$\tilde{G}$	Quantized gain
β_{ij}	Pitch predictor gain vector
$u[n]$	Adaptive codebook excitation vector
$e[n]$	Decoded combined excitation vector
$ppf[n]$	Pitch postfiltered excitation signal
M_f, M_b	Optimal forward and backward pitch postfilter lags
$ppf'[n]$	Unnormalized pitch postfiltered excitation signal
γ_{ltp}	Gain weighting factor, 0.1875 or 0.25
g_p	Pitch postfilter scaling gain
g_p, g_b	Optimal forward and backward gains for pitch postfilter
C_f, C_b	Forward and backward excitation crosscorrelations
D_f, D_b	Forward and backward excitation energies
E_f, E_b	Forward and backward energies for pitch postfilter
T_{en}	Energy of excitation signal
$sy[n]$	LPC synthesized speech signal
$pf[n]$	Formant postfiltered signal
$q[n]$	Output speech signal
$g[n]$	Formant postfilter gain signal
k_1	Interpolated reflection coefficient for tilt compensation filter
λ_1, λ_2	Weights for formant postfilter, 0.65, 0.75
g_s	Amplitude ratio of sy and pf vectors
α	0.0625 in gain scaling unit
b_e	Fixed predictor for LSP interpolation during frame erasure concealment, 23/32