Proc. Biennial Symp. Commun. (Kingston, ON), pp. 320-323, June 1990

Low-Delay Speech Coders at 16 kb/s: A CELP and A Tree Coder

Majid Foodeei [1] and Peter Kabal [1,2]

[1] Electrical Engineering
McGill University
Montréal, Québec,
Canada, H3A 2A7

[2] INRS-Télécommunications
Université du Québec
Verdun, Québec,
Canada, H3E 1H6

ABSTRACT

For speech coders to be used in network applications, a transparent or near transparent quality (Mean Opinion Score rating of 4.0) is required. Though this is a necessary criterion, other desired properties include: low-delay, robustness to channel errors, moderate complexity, capability to handle non-speech signals in the telephone band, and good tandeming performance. The CCITT's current consideration for standardization of the 16 kb/s network-quality speech coders (to be finalized in 1991), requires a maximum delay of 5 ms and has set 2 ms as the objective. A better understanding of the trade-offs resulting from the use of different schemes is required. In this work, results and schemes of a delay-decision tree coder based on the (M,L) algorithm (ML-TREE) [1,2] and a Low-Delay Code Excited Linear Predictive (LD-CELP) coder, proposed by the AT&T laboratories for the 16 kb/s CCITT's standardization [3], are studied and compared. For the comparison, the LD-CELP and the ML-TREE coders are simulated. Results obtained to date show that the segSNR of the coded speech using ML-TREE and LD-CELP coders under clear channel are comparable. The design of LD-CELP coder has emphasis on the channel error robustness while the ML-TREE coder design has not considered this issue closely yet. The performance of the two coders under noisy channel conditions reflects this.

1. INTRODUCTION

The delay-decision tree coder based on the (M,L) algorithm of [1,2] (ML-TREE) and the Low-Delay Code Excited Linear Predictive coder of [3] (LD-CELP) may both be considered as potential candidate coders for the low-delay network-quality applications. Performance quality equivalent to 7 bits/sample log-PCM with delays less than 2 ms under clear channel conditions is accomplished by the two coders. Satisfactory performance quality, under noisy channel conditions, is also reported for the LD-CELP. The CCITT requirement under these conditions is that the performance of the 16 kb/s coder should not be worst than the 32 kb/s ADPCM standards (G.721) at Bit Error Rates (BER) of 10^{-3} and 10^{-2}. In this contribution, first the structure and methods used in the ML-TREE and LD-CELP are discussed. Next, results of a comparison between the two coders along with a discussion on their similarities and differences are presented.

2. ML-TREE CODER

The ML-TREE Coder discussed in detail in [1,2] is a delayed-decision tree coder based on the generalized ADPCM. The Block diagram of the ML-TREE is given in Figure 1. The ML-TREE of [1,2] suggested a pitch filter and postfiltering for better results. As will be discussed later, since the proposed LD-CELP of [3] does not include pitch filter and postfiltering, for comparison purposes similar conditions are used for the ML-TREE coder. Other than this difference, the ML-TREE used in this study is kept identical to the original one in [1] and the same simulation programs are used.

Stochastic Tree: The stochastic tree speech coders are known to perform better than the deterministic ones. In a stochastic tree coder (ML-TREE), distinction is made between the innovation code tree which represents the quantized residual signal and the reconstruction code tree which represents the reconstructed output signal. The nodes of the innovation tree are populated from a Laplacian random number dictionary of size 2^k (in this study 2^k=4096). The reconstructed code tree is obtained by multiplying each of the innovation tree node values by a gain factor and then passing it through a synthesis filter (innovation and reconstruction node value pairs). The concept of the generalized predictor coder which is used in the ML-TREE coder includes a formant predictor $P(z)$ ($p = 8$) which acts on the short-term redundancies in the input speech

$$P(z) = \sum_{i=1}^{p} a_i z^i.$$

The adaptation of the above all-pole predictor is done in a backward fashion using the adaptive Lattice algorithm. The reflection coefficients are converted to the direct form before using in the prediction filter. The perceptual weighting filter used has the form

$$W(z) = \frac{1 - P(z)}{1 - P(z/\lambda)}, \quad \text{where} \quad P(z/\lambda) = \sum_{i=1}^{p} a_i \lambda^i z^i$$

is the bandwidth expanded version of $P(z)$ (λ=0.85 is used). The residual signal is passed through the perceptual weighting filter before the (M,L) tree search algorithm is applied. MSE minimization is used in an analysis by synthesis configuration (Figure 1).

Gain Adaptation: To increase the dynamic range of the input signals, each innovation value is multiplied by the node gain to yield the innovation sample e_q. The node gain is adaptive and the backward adaptation is done according to

$$\widehat{G}^2 = \delta^2 G^2 + (1 - \delta) e_q^2 \quad \text{where} \quad 0 < \delta < 1 \quad (\delta = 0.86).$$

The above adaptation is an exponentially averaged variance estimate and δ controls the effective length of the exponential window. This adaptation strategy is better suited for the stochastic tree coding.

Multipath Search: In the multipath (M,L) tree search algorithm, M denotes the maximum number of kept paths in contention and L is the number of samples in each of these paths or the decision delay length. This study uses M=16 and L=8 which results in encoding delay of maximum 8 samples or end-to-end delay comparable to the one obtained in the LD-CELP. The branching factor of the tree is 4, which means the tree coder produces 2 bits/sample. At time instant n, each of the M (maximum) paths in contention are extended. The error

accumulated for each of the 4M extended paths are calculated. The path with the lowest accumulated error is selected. The two bit branch code of the root of this path L samples back is the only information transmitted to the receiver at time n (indicated by c(n-L) in Figure 1). Only valid paths that stem from this root are kept (maximum M).

Delayed Prediction Update: The formant prediction filter coefficients are updated in a delayed update configuration. In Figure 1 this is shown as $\{a_i(n-2L)\}$ or $\{a_i(n-L)\}$ which means that the update algorithm at time instant n uses samples as recent as 2L or L samples back. Reference [1] shows that L sample delay update strategy actually results in better prediction gains than zero delay update strategy. As seen in Figure 1, this also results in complexity reduction of the coder since only one update of the LPC coefficients is done for all branches in contention.

3. LD-CELP CODER

The LD-CELP, like the conventional CELP [4], searches the codebook for the best matching codevector (each vector is 5 samples long) using analysis by synthesis and by minimizing the perceptually weighted error. Figure 2 shows the block diagram of the LD-CELP encoder and decoder. Although the analysis by synthesis structure resembles that of ML-TREE in Figure 1, as indicated by the thick lines in this figure, the signal processing is on a block-by-block basis. This "block" characteristic of the LD-CELP verses the "averaged" characteristic of the ML-TREE is the main conceptual difference between the two coders. This fundamental difference is also the main source of performance differences between the two coders and the type of trade-offs which exist within each coder. In computation-load consideration for the real time implementation, this translates to "bursts" of calculation loads in the case of LD-CELP and more distributed computation load in the case of ML-TREE.

LD-CELP, unlike the conventional CELP but like the ML-TREE, performs the estimation of the time-varying model for the spectral envelope in a backward fashion to fulfill the low-delay requirement. This means that the LPC analysis (using the autocorrelation method) for obtaining the all-pole predictor coefficients is done based on the history of the *quantized* speech. The selected code vector index is the only information sent to the receiver and no side information is transmitted.

Elimination of the Pitch Synthesis Filter: Unlike the conventional CELP which includes a pitch (long term) synthesis filter before the LPC (short term) synthesis filter, the LD-CELP has eliminated the pitch synthesis filter. Reference [3] explains this by saying that the backward pitch predictor adaptation is too sensitive to the channel errors. Instead a higher order LPC predictor filter is used which has a high price in complexity. Since the prediction gain of the adaptor and the coder SNR saturates at order 20 for male speakers and around order 50 for female speakers, LD-CELP chooses order 50 for the predictor filter (p=50). Bandwidth expansion is used to make the noise less perceivable. In effect the formant peaks are widened in the frequency response by moving the poles away from the unit circle or concentrating the noise in the formant regions. Higher robustness to channel errors is accomplished. Therefore a modified LPC predictor $P(z/\lambda)$ with $\lambda = e^{-2\pi B/8000} = 0.988$ (B = 15 Hz) is defined and used. As in the case of the ML-TREE, white noise correlation technique is used to "clamp" the spectral dynamic range to 40 dB and to reduce the ill-conditioning problem.

The perceptual weighting filter of the conventional CELP or the ML-TREE coder is replaced with a different one

$$W(z) = \frac{1 - Q(z/\lambda_1)}{1 - Q(z/\lambda_2)}, \quad 0 < \lambda_2 < \lambda_1 \leq 1 \quad (\lambda_1 = 0.9, \lambda_2 = 0.4)$$

where

$$Q(z/\lambda_1) = \sum_{i=0}^{M} q_i \lambda_1^i z^i \quad \text{and} \quad Q(z/\lambda_2) = \sum_{i=0}^{M} q_i \lambda_2^i z^i.$$

Instead of using the 50th-order analysis filter which results in artifact synthesis speech, a separate LPC analysis of order 10 (M=10) is needed. The analysis for computing q_i's uses the *unquantized* speech.

Robust Gain Adaptation: To accommodate a wide dynamic range of input signals, a vector is normalized by the estimated gain before encoding, then the selected codevector is multiplied by the same estimated gain before passing it through the synthesis filter. Two methods of vector generalized Jayant gain adaptor [5] and adaptive logarithmic gain predictor are suggested. The former has a fixed coefficient and typically shorter response and thus is more robust to channel errors, while a higher clean channel performance may be obtained with the latter. In both methods a leakage factor close to unity is used in order to improve the robustness.

Gray Coding of the Codebook Indices: Shape and gain codevector indices are Pseudo Gray coded [6,7] so in the case of a single error occurrence the received codevector tend to be close to the transmitted one. This results in a significant improvement in a noisy channel environment.

Tandeming and Non-voice Considerations: Postfiltering which improves the performance of the coder in the conventional CELP or in the original ML-TREE proposed in [1], is eliminated in the LD-CELP for severe accumulated distortion during tandem coding. The simulations in this study did not include tandeming performance. Reference [8] has reported that the performance of the LD-CELP under asynchronous tandeming condition is worst than the G.721 requirements and needs further research. When decoding non-speech (such as modem) signals, postfiltering results in phase distortion. Elimination of postfiltering also removes the concern for this undesirable effect.

Use of Gain/Shape Vector Quantization and Zero State Response (ZSR:) Product vector quantization (VQ) [9] is used to bring down the computation load by using a 7-bit shape and a 3-bit gain book. The codevector index in LD-CELP is the concatenation of 3 indices i, j, and k. Therefore the codevector is the product of m_k, the sign portion of the gain vector (+1 or -1), g_i, the magnitude portion of the gain vector, and y_j, the selected shape codevector. ZSR method of [10] is used for further reducing the computation load during the search for the best matching codevector. If $F(z)$ is the synthesis filter transfer function and $W(z)$ the perceptual weighting transfer function, one can form the cascaded filter: $H(z) = F(z)W(z)$. The MSE minimization of the distortion between the difference vector $\mathbf{x}(n)$ and the synthesis vector $\mathbf{x}_{i,j,k}$ maybe written as

$$D = |\mathbf{x}(n) - \mathbf{x}_{i,j,k}|^2 = |\mathbf{x}(n) - \delta(n) m_k g_i \mathbf{H} y_j|^2$$

where $\delta(n)$ is backward adaptive gain known prior to the search. $\mathbf{H}$ is the lower triangular matrix

$$\mathbf{H} = \begin{bmatrix} h(0) & 0 & 0 & 0 & 0 \\ h(1) & h(0) & 0 & 0 & 0 \\ h(2) & h(1) & h(0) & 0 & 0 \\ h(3) & h(2) & h(1) & h(0) & 0 \\ h(4) & h(3) & h(2) & h(1) & h(0) \end{bmatrix}$$

and $\{h(0), h(1), .., h(4)\}$ are samples of the impulse response of the cascaded filter.

Adaptation Coefficient Updates: Since the spectral changes in the speech signals are relatively slow varying, significant computation load reduction is obtained with a minimum loss of performance by updating the coefficients every 8th vector instead of every vector.

Training of the Codebooks: Using the gain adaptive VQ of [5] and the product VQ training methods of [9], the shape and gain codebooks are trained. In the closed loop gain-adaptive training algorithm used, the distortion-versus-iteration does not monotonically decrease or converge. The codebook with the lowest distortion after a preset number of iterations is stored. When the individually optimized gain/shape VQ algorithm suggested in [9] was used, the initial shape book codevectors were chosen from numbers with gaussian probability distribution, and the initial 3 bit gain book scalar values were selected from uniformly distributed gain values.

4. RESULTS AND DISCUSSION

The ML-TREE coder showed great promise by producing subjective quality equivalent to 7-bit/sample log-PCM with encoding delay not more than 1.125 ms [2]. Results of the simulation shown in Table 1, imply that the segSNR's of the coded speech using the ML-TREE and LD-CELP coders for the two speech sentences "CAT" and "OAK" are very close. The ML-TREE does somewhat better for the male utterances and the LD-CELP is slightly better for female utterances. The informal subjective tests also agree with the above conclusion. The above comparison is for the clean channel condition. The design of the present version ML-TREE coder (unlike the LD-CELP) does not have an emphasis on the noisy channel performance. The coded speech using the ML-TREE degrades rapidly under the noisy channel conditions while the LD-CELP withstands BERs of 10^{-3} and 10^{-2} with acceptable levels of quality loss. When ML-TREE is operating under noisy channel condition, catastrophic effects resulting from switching between paths are the main longstanding problem with the tree structure. Once an error occurs, the effects are long lasting and should at least be shortened in an effort to address this problem.

The use of the 50th order prediction filter in the LD-CELP as opposed to the 8th order filter in the ML-TREE constitutes a main difference between the two coder. Reference [8] reports that the prediction gains and SNR gains obtained as a result of using high-order predictor justify the complexity added. The perceptual weighting filters used in the ML-TREE and LD-CELP coders not only differ in form (as seen earlier) but also are different in the use of *quantized* or *unquantized* speech signal to update their coefficients. The use of high-order synthesis filter in the LD-CELP has forced the coder to use a separate predictor filter for the perceptual weighting filter. Although the LD-CELP perceptual weighting filter has some advantages, the price of a separate predictor filter is not negligible.

The LD-CELP uses a 20 ms Hamming window or alternatively a recursive modified Barnwell window (to distribute the computation load for implementation considerations) for the backward adaptation of the prediction filter. The method of choice for the analysis is autocorrelation and the update of the coefficients is done every 8th vector (5 ms). The ML-TREE on the other hand uses the Lattice adaptation algorithm to obtain reflection coefficients which are then converted to the direct form for use in the 8th order predictor filters. A one-pole or exponential window is used on the analysis data. The shape and the effective length of this window maybe controlled by a parameter. Computational savings are obtained by using this window. The coefficient updates are done on a sample-by-sample basis but in a delayed update configuration. The Lattice adaptation algorithm used in the ML-TREE significantly improves the performance over the autocorrelation method. In the LD-CELP, the complexity increase as a result of high-order predictor makes the use of Lattice update unsuitable. The exponential window used in the ML-TREE has a computational advantage over the windowing methods used in the LD-CELP, yet with the one-pole exponential window the control over the shape and the effective length is restricted and may not be suitable for the noisy channel conditions. The bandwidth expansion applied to the high-order LPC predictor of LD-CELP can also be applied to the LPC predictor in the ML-TREE coder. This improves the robustness to channel errors by making the noise less perceivable.

Coder	CAT		OAK	
	Male	Female	Male	Female
LD-CELP	17.9	19.1	18.4	20.1
ML-TREE	19.1	19.1	19.6	19.3

Table 1. Comparison of Coders' segSNR.

Training the dictionary in the ML-TREE coder (as it is done for the LD-CELP codebooks) boosts the performance of the coder. In an experiment segSNR improvements of about 1 dB were obtained when the stochastic innovation dictionary was trained.

To improve robustness of the coder to channel errors, Pseudo Gray Coding used on the transmitted indices in the LD-CELP can also be considered in the ML-TREE coder. Although the exponentially averaged gain adaptation method of ML-TREE has good results for clean channels, a better gain adaptor suited for the stochastic tree coders is required to overcome the malfunctioning of the ML-TREE coder under noisy channel conditions. The reason for this problem is the long lasting memory in the gain adaptation strategy. Although the Jayant update is not suited for the stochastic tree coders, a modified Jayant update does seem suitable.

Coarse simulation comparisons show that the two coders have comparable complexities. As explained earlier, one advantage of the ML-TREE coder over CELP coders is that the computation load is better distributed and does not have the "burst" characteristics of the CELP coders. A more precise complexity comparison between the two coders should take into account real-time implementation considerations. There is evidence that using similar methods employed in the LD-CELP coder and other methods (more suitable to the tree coding structure), the ML-TREE coder performance under noisy channel conditions may be improved. It may be concluded that the ML-TREE coder is also a good candidate for the 16 kb/s network applications.

References

1. V. Iyengar, "A Low delay 16 kb/s coder for speech signals," *Master of Eng. Thesis , Dept. Of Elec. Eng. McGill University*, (Aug. 1987).
2. V. Iyengar and P. Kabal, "A low delay 16 kb/s speech coder," *Proc. ICASSP Conf.*, pp. 243-246 (1988).
3. AT&T contributions to CCITT Study Group XV and T1Y1.2 (October 1988 - July 1989).
4. M. R. Schroeder and B.S. Atal, "Code-Excited Linear Prediction (CELP): High quality speech at very low bit rates," *Proc. ICASSP Conf.*, pp. 937-940 (1985).
5. J. H. Chen and A. Gersho, "Gain-adaptive vector quantization with application to speech coding," *IEEE Trans. Comm.*, pp. 918-930 (Sep. 1987).
6. J. R. B. De Marca and N. S. Jayant, "An algorithm for assigning binary indices to the codevectors of multidimentional quantizer," *ICC Conf. Rec.*, pp. 1128-1132 (1987).
7. K. A. Zeger and A. Gersho, "Zero redundancy channel coding in vector quantization," *Electronics Letters*, pp. 654-656 (June 1987).
8. J. H. Chen, "A robust low-delay CELP speech coder at 16 kb/s," *Proc. GLOBCOM Conf.*, pp. 1237-1240 (1989).

9. M. J. Sabin and R. M. Gray, "Product code vector quantizers for waveform and voice coding," *IEEE Trans. ASSP*, vol. ASSP-32, no. 3, pp. 474–488, June 1984.

10. J. H. Chen and A. Gersho, "Real-time vector APC speech coding at 4800 bps with adaptive postfiltering," *Proce. ICASSP Conf.*, pp. 2185–2188 (1987).

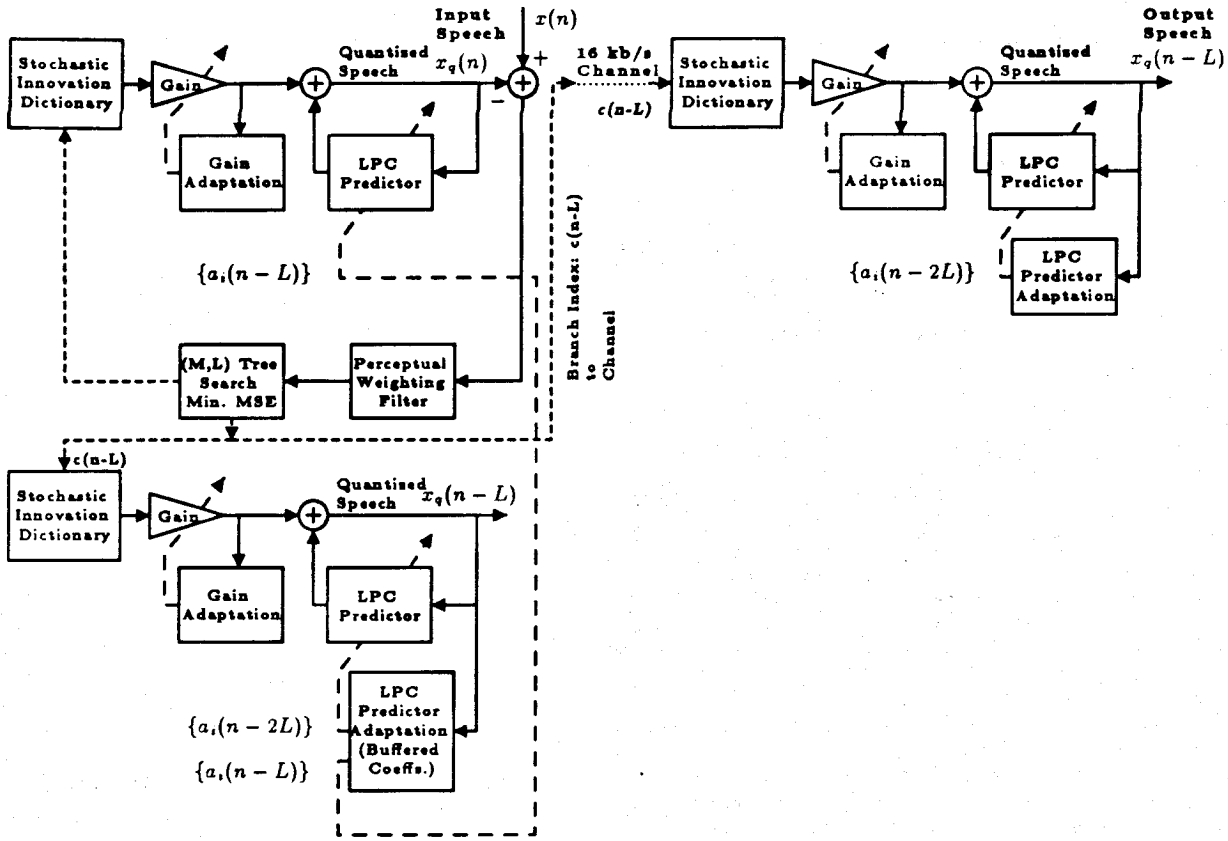

Fig. 1 ML-TREE Encoder and Decoder Block Diagram

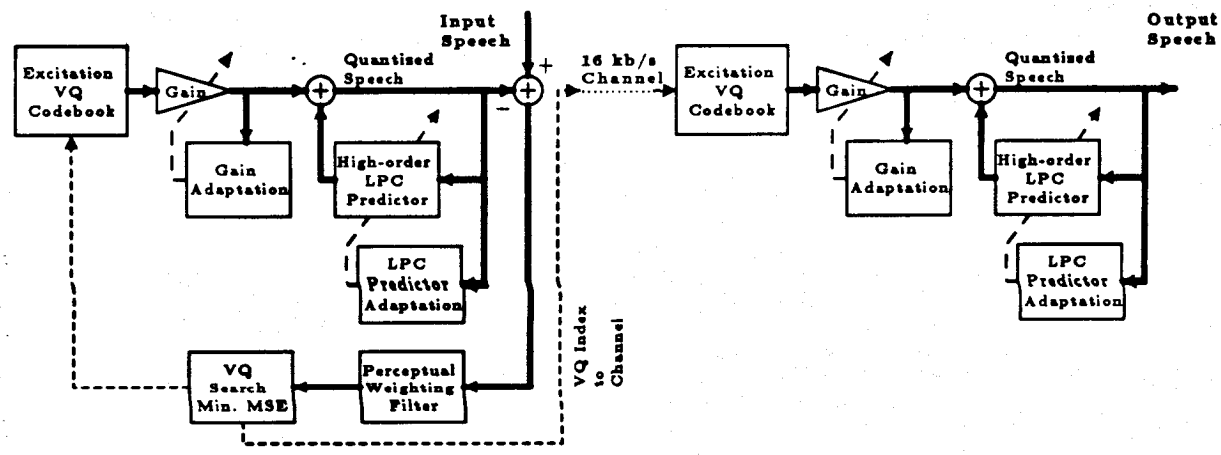

Fig. 2 LD-CELP Encoder and Decoder Block Diagram