ON-LINE SIMULATION MODULES FOR TEACHING SPEECH AND AUDIO COMPRESSION TECHNIQUES

Venkatraman Atti[1] and Andreas Spanias[1]

Abstract *¾ In this paper, we present a collection of software educational tools for introducing speech and audio compression (or coding) techniques to undergraduate and graduate students. These speech processing tools enable on-line simulations of speech compression algorithms that are being used in digital cellular phones, internet streaming applications, teleconferencing, and voice over internet protocol (VoIP) applications. This simulation software is accompanied by a series of computer experiments and exercises that can be used to provide hands-on training to class participants. With this on-line simulation tool and a set of well-complemented laboratory exercises, students can easily comprehend the basic techniques involved in speech and audio coding algorithms. Specific functions that have been developed include the typical modules that are usually embedded in CD-players, MP3-players, and mobile phones. Details on the software, exercises, and assessment data will be provided at the conference.*

Index Terms ¾ Web course, Java speech tools, J-DSP, LPC, ADPCM.

INTRODUCTION

Teaching DSP concepts at the undergraduate level requires extensive reference to motivating applications as well as hands-on computer experiences. Select algorithms in speech and audio compression represent such motivating examples because of their association with appealing applications such as cellular telephones, MP3 players, and surround sound DVD and cinema systems. In addition, using speech signals for hands-on DSP experiments is convenient on a PC and provides several possibilities for visualization of important signal properties. Speech coding is primarily concemed with obtaining compact representations of speech for efficient transmission and/or storage [1], while, audio coding deals with the efficient digital representation of high-fidelity wideband audio signals [2]. Exposure to both of these application areas can help students gain intuition in practical implementation of several DSP concepts. For example, coverage of some aspects of speech coding can help students learn how digital filters are used in speech modeling. Furthermore, students can get exposed to examples of spectral representations of speech by visualizing spectra of speech and frequency responses of filters representing the vocal tract. Interactive lessons in speech coding not only contain valuable examples of DSP algorithms but can also broaden student knowledge through exposition to a variety of international compression standards. Despite the fact that certain topics in speech coding could be covered only by static web content, some important aspects of speech processing are best communicated through the use of computer simulation tools. In this paper, we describe a series of speech coding simulation tools that we have developed using the J-DSP [4] simulation environment. J-DSP is a GUI-driven on-line *simulation environment* that enables a student to study and verify through graphics various aspects of speech processing and coding. The J-DSP speech coding functions developed exploit the graphical interface and allow the user to simulate complex speech coding algorithms by simply forming graphically simple block diagrams. The software tools are accompanied by exercises that are accessible by undergraduate students participating in digital signal processing (DSP) courses and computer science multimedia courses [3]. The tools described in this paper represent a significant extension of the J-DSP software tools, presented earlier by Spanias, et al, [4]-[6].

This paper is organized as follows. First, the simulation environment and the various speech coding functionalities implemented are presented. Then, we provide a high-level description of speech compression techniques with some example simulations for an LPC vocoder. Several concepts related to PCM, DPCM, ADPCM quantization techniques receive in-depth treatment. We review the methodologies that involve the LPC filter-parameter-transformations with laboratory exercises. With the help of a simulation model and an exercise, we describe the VQ, and RELP coding. This paper concludes with a discussion of on-line simulation exercises and applications.

SIMULATION ENVIRONMENT

A brief introduction to the simulation environment and the associated GUI follows. Figure 1 shows the J-DSP graphical user interface and the simulation tool's editor window. From this figure, it can be seen that all functions appear as graphical blocks. Each block is associated with a specific signal processing function. A variety of DSP systems can be simulated by connecting the blocks, and editing their parameters through dialog windows. System execution is dynamic, which means that any change at any point of a simulation will automatically take effect in all related blocks. For more detailed description on the block manipulation and the program infrastructure refer to [6].

[1] The authors are with the Electrical Engineering Department, MIDL lab, Arizona state university, Tempe, AZ, Email: atti@asu.edu, spanias@asu.edu

0-7803-7444-4/03/$17.00 © 2003 IEEE **November 5-8, 2003, Boulder, CO**
33rd ASEE/IEEE Frontiers in Education Conference

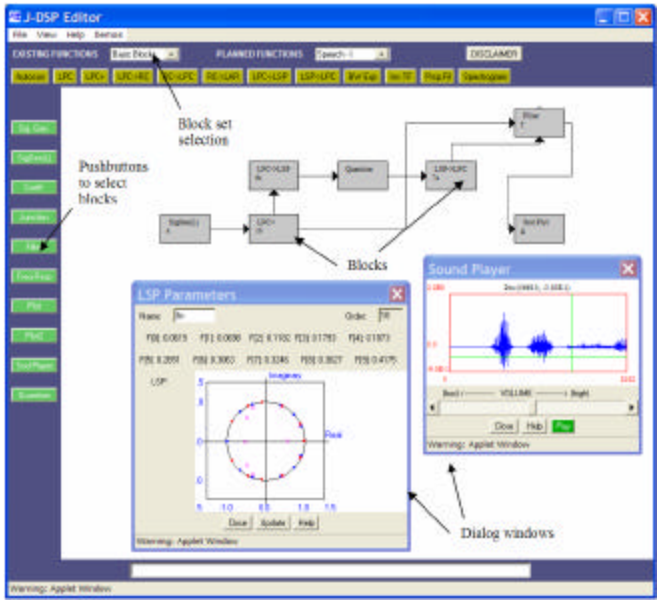

FIGURE 1
AN EXAMPLE SIMULATION DEPICTING THE GUI O F AN LPC VOCODER

SPEECH COMPRESSION MODULES

In this section, we give a brief summary of various speech processing modules developed in J-DSP. These modules include: PCM, DPCM, ADPCM, scalar quantization, VQ-design, LPC vocoder, and RELP coding. These blocks are divided into two sub-sections, i.e., SPEECH-I and SPEECH-II, based on their functionalities, as shown in Table 1. Note that the block names are represented as bold italics (Example: ***LPC***)

TABLE I
A LIST OF SPEECH COMPRESSION BLOCKS IMPLEMENTED

	Block Name	Block Description
SPEECH-I	***Autocorr***	Autocorrelation
	LPC	Linear predictive coding
	LPC->RC	LPC to reflection coefficients
	LPC->LAR	LPC to log-area-ratios
	LPC->LSP	LPC to line-spectral pairs
	Percp. Wght	Perceptual weighting
SPEECH-II	***PCM***	Pulse-code modulation
	DPCM	Differential-PCM
	ADPCM	Adaptive-differential-PCM
	Quantizer	Uniform/Non-uniform quantization
	VQ	Vector quantizer
	Pitch Est.	Pitch estimation

The process of extracting model parameters from the speech signal is called *speech analysis*. Depending on the vocoder type, these parameters usually include: filter coefficients, excitation, pitch, gain, etc, as shown in Figure 2. At the decoder, these parameters are used to reconstruct the speech by filtering the excitation signal with the LP synthesis filter. This process is called *speech synthesis*. Most of the well-known speech coding standards are based on *linear prediction* – a process that characterizes the shape of the speech spectrum with a small number of parameters that can be coded efficiently.

LPC predicts a time-domain speech sample by a linear combination of previous samples. This is given by,

$$s_{pred}(n) = \sum_{k=1}^{L} a_k \, . s(n-k) \qquad (1)$$

where $s_{pred}(n)$ is the predicted value of speech sample based on the previous L samples of the input speech $s(n)$, and a_k are the prediction coefficients or weights. The prediction coefficients are obtained by minimizing the prediction error or residual. This is given by,

$$e(n) = s(n) - \sum_{k=1}^{L} a_k \, . s(n-k) \qquad (2)$$

The LP analysis filter generates parameters for an all-pole filter that models the vocal tract. The filter transfer function is given by,

$$A(z) = 1 - \sum_{k=1}^{L} a_k \, . z^{-k} \qquad (3)$$

and the LP synthesis filter is given by, $1/A(z)$.

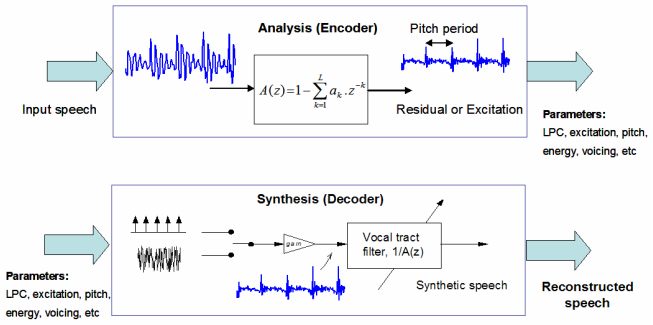

FIGURE 2
A TYPICAL ANALYSIS-SYNTHESIS MODEL

Exercises and Learning Modules

Several laboratory exercises have been developed to expose the students to the concepts of LPC parameter transformations, perceptual weighting, open-loop and closed-loop pitch estimation, etc. In particular, more emphasis has been given on the following: linear predictive coding of speech and its applications in current cellular standards, PCM, DPCM, and ADPCM quantization, VQ and its usage in the design of codebooks, and filtering of audio signals with a Quadrature Mirror Filter (QMF) bank.

A typical scenario is for a student to read a high-level tutorial on each of these methods and then form and execute simulations with a real-life signal and visualize and evaluate the compression process. The exercises that have been outlined in this paper as well as additional ones not described here are given to students in detailed write-ups. These include a theory section and step-by-step instructions

for operating the software and analyzing the intended results. The students perform the experiments and respond to questions in a quiz section. They then have to submit a typed report that describes the results obtained along with several relevant figures and graphs that are ported from the simulation environment. In the following sections, we present some of the speech compression topics that we cover using J-DSP simulations and on-line exercises that the students perform using J-DSP.

PCM, DPCM, AND ADPCM

Typical applications of PCM, DPCM, and ADPCM are in telephony, VoIP, data streaming applications, etc. Basic concepts such as filtering, quantization, and digitizing are usually introduced in a senior-level undergraduate class. An introduction to these concepts with the help of a GUI-simulation and an exercise, from the perspective of speech compression, will help the students understand the concepts better and quicker.

The steps associated with PCM are similar to any digitization scheme: First, the speech is digitized and the samples are converted to binary-digits (bits). These bits are represented by *pulses* that encode the sample amplitudes, and therefore, the name *pulse-code modulation*. PCM is one of the simplest forms of encoding and is usually considered as a reference for evaluating the performance of other speech coders.

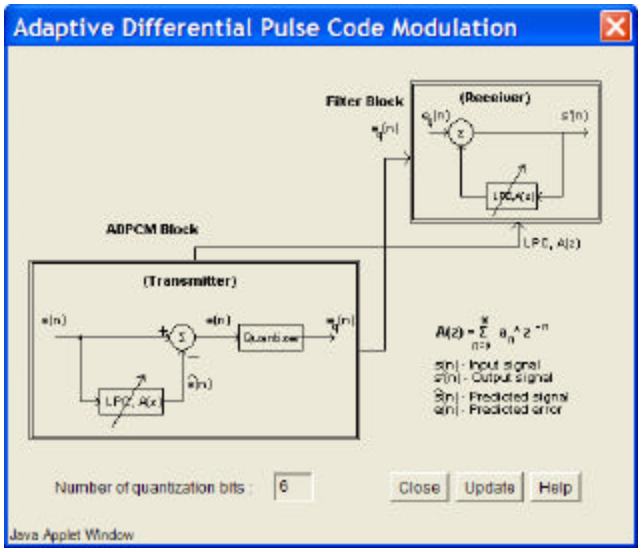

FIGURE 3
ADPCM TRANSMITTER AND RECEIVER SIMULATION

DPCM and ADPCM are differential encoding techniques, in which, rather than encoding the input waveform directly, we code the difference between the original waveform and one reconstructed from using a predictions process. If the prediction filter is fixed, we have *DPCM coding*. An adaptive prediction filter results in *ADPCM coding*. Figure 3 shows an underlying model of the ADPCM simulated with J-DSP. In this figure, *A(z)* is the prediction filter, the ***ADPCM*** block acts as an ADPCM transmitter and the ***filter*** block can be viewed as the receiver.

Example Simulation – 1

A simulation that compares the performance of PCM, DPCM, and ADPCM techniques is shown in Figure 4. The parameters used for the simulation are as follows. The number of quantization bits, n = 3, a male speaker input, the number of frames is 32, and the frame size is 256.

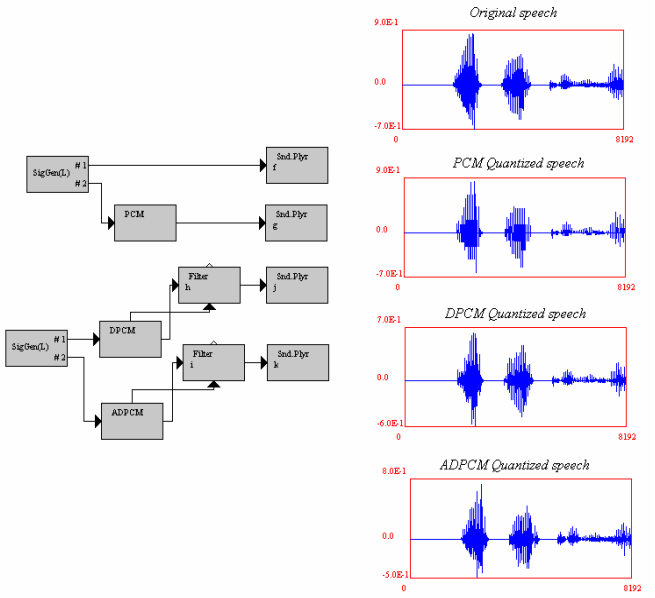

FIGURE 4
COMPARISON OF THE PCM, THE DPCM, AND THE ADPCM

Exercises were designed to highlight some of the salient features of PCM, DPCM, and ADPCM techniques, i.e.,

- A 6dB improvement in the SNR for every 1bit increase in the number of quantization bits, when the input signal is a random stationary signal with uniform distribution. The same example can be run for input signals with Gaussian random distribution.
- Effects of uniform and non-uniform quantization on the perceptual quality of speech.
- The underlying concepts of ADPCM coding from the dialog window shown in Figure 3.
- Computation of segmental and over-all SNR values between the unquantized signal *s(n)*, and the quantized signal $s_q(n)$.
- Improvement in SNR values for the DPCM and the ADPCM coding over PCM, for the same number of bits. For example, in the simulation shown in Figure 4, SNR values obtained for the PCM, the DPCM, and the ADPCM coding are 2.34dB, 4.76dB, and 5.52dB respectively.

- ADPCM coding based on the following three cases: fixed prediction filter and adaptive quantization, adaptive prediction filter and uniform quantization, and adaptive prediction filter and adaptive quantization.

Another experiment that can help students gain insight to the coding concepts is to quantize the prediction filter coefficients instead of the residual, and evaluate the coder performance.

LPC VOCODER

A brief description of speech spectra, speech synthesis, and vocal tract parameterization using all-pole filters are provided in the earlier sections. Being one of the first parametric coders, LPC is present in most of the speech coding standards. In this section, with the help of a simulation, we explain how these concepts are used in cell phones. The idea is to parameterize a speech signal as shown in Figure 2. An example of this parameterization that demonstrates clearly how data compression is accomplished using LPC is as follows: 256 samples (32ms) of speech can be represented with a 10^{th} order all-pole filter and 3-4 additional parameters that specify the excitation (pitch, voicing, gain). Hence the information needed to represent the speech in one 32ms frame is reduced considerably, and thereby reduces the bit rate.

Example Simulation – 2 (Linear Prediction)

Figure 5 shows an example simulation of the LPC vocoder. In the figure, the ***Sig.Gen(L)*** block provides some elementary framing capabilities, the ***LPC*** block computes the residual *e(n)* and the LP coefficients a_i. The ***filter*** block reconstructs the speech by filtering the residual *e(n)* with the LP synthesis filter *1/A(z)*. The synthesis is done on a frame-by-frame basis, and in the end the frames are concatenated to produce a new speech record that we can then listen and evaluate, using a ***Snd.Player*** block. Both analytical and experimental questions have been drafted to introduce the students to the following concepts:

- The relationship between the LPC and FFT spectra,
- Measuring the pitch period, the voicing, and the first three formant frequencies of successive speech frames,
- The time- and frequency-domain plots of the voiced and unvoiced speech frames,
- Objective measures such as the segmental and over-all SNR values, and cepstral distances.
- Effects of the bandwidth expansion on the LP coefficients,
- The plots of the residual signal and their corresponding dB-spectra for a voiced and an unvoiced frame, and,
- The pitch contour, the voicing, and the frame energy plots as a function of frame number, and analyze their relationship.

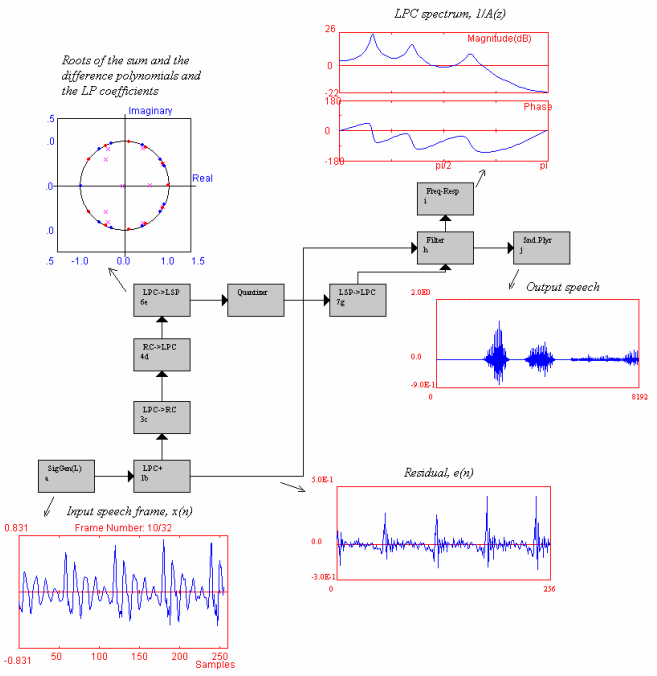

FIGURE 5
AN LPC MODULE

A sample plot that describes the relationship between the LPC and the FFT spectra is shown in Figure 6. From the figure, it can be noted that the LPC spectrum is a smoothed version of the FFT spectrum. That is, LP analysis represents the frequency shaping attributes of the vocal tract in the source-system filter model (Figure 2) but fails to model the fine structure.

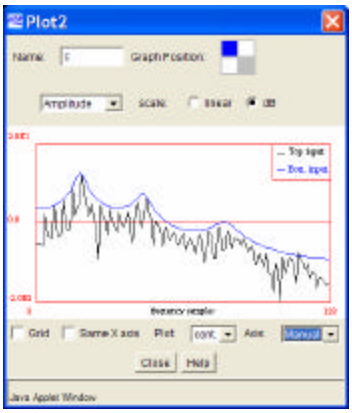

FIGURE 6
THE LPC AND THE FFT SPECTRA

Example Simulation – 3 (LPC Transformations)

Direct-quantization of the LP coefficients results in a poor perceptual quality of the reconstructed speech, and at the same time the LP coefficients are highly sensitive to quantization errors. Therefore, the LP coefficients are converted to reflection coefficients (RCs), log-area-ratios

(LARs), or line spectral pairs (LSPs) and then quantized. These parametric representations are more suitable for encoding/transmission due to their improved quantization and interpolation properties.

Figure 7 shows a sample simulation that explains the quantization properties of the LP coefficients and RCs. From this figure, students can observe that the LPC spectrum could change considerably after quantization of the LP parameters. Moreover, the LPC spectrum with quantized RCs results in a better performance relative to quantization of direct form coefficients. This is evident from the plot at the bottom of Figure 7. As part of an exercise problem, the students are asked to prepare a table with perceptual quality of the reconstructed speech on a scale of 1 to 5 (1=very poor, 2=poor, 3=good, 4=very good, 5=excellent) for the following cases: direct-quantization of LP coefficients, quantization of RCs and LSPs. This process also exposes the students to the process of estimating the subjective quality of speech using the Mean Opinion Score (MOS). In addition, more advanced computer exercises have been developed that can be used in a speech and audio processing graduate course [7]. Some of them are mentioned below.

- The concepts of pre-processing, perceptually weighted filtering, short-term and long-term predictors.
- Significance of the symmetric and asymmetric windows in the LP analysis, and how the windowing of autocorrelation coefficients improves the estimates of LP coefficients,
- The z-plane plots of LPCs and LSFs, and to experiment why the roots of the sum and the difference polynomials of LSFs alternate on the unit circle.
- Importance of the bandwidth expansion of the autocorrelation coefficients, overlapping of frames, etc.

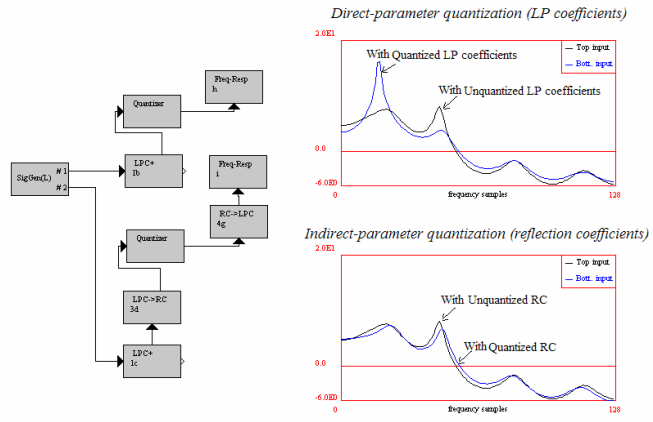

FIGURE 7
COMPARISON OF THE LPC SPECTRA FOR THE DIRECT- AND THE INDIRECT-PARAMETER QUANTIZATION

Example Simulation – 4 (RELP)

A speech coder that uses explicitly the residual and the configuration in Figure 2 is called residual-excited linear predictive coder (RELP). A variant of RELP that uses regularly spaced pulses to represent the excitation is used in the first generation GSM digital cellular system in Europe. An example simulation designed to introduce the concepts of RELP coding is shown in Figure 8. In this problem, we down-sample and up-sample the prediction residual by different factors and see the effects on speech synthesis. The reconstructed signal is assessed in terms of the SNR values and the subjective evaluation. The subjective scores are on a scale of 1 to 5 – one being very poor and five being excellent. The guidelines for the RELP simulation are as follows.

- For every frame down-sample the prediction residual by a factor of 4. Then up-sample it by a factor of 4. The processed signal in every frame after the rate changes will now contain several zero samples (3 out of 4 samples will be zero).
- As an extension to the above problem, students can design a Butterworth filter to realize the anti-aliasing and the reconstruction filters.

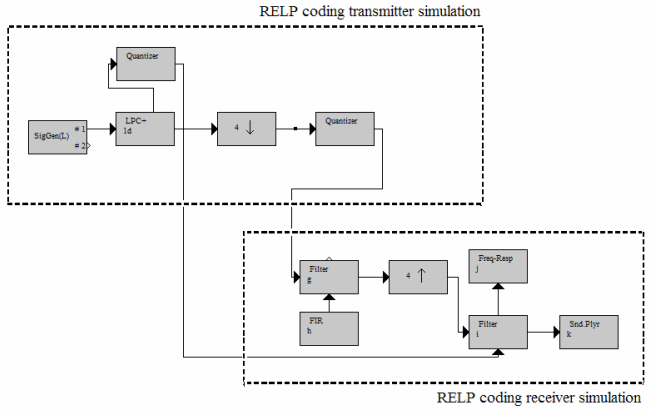

FIGURE 8
RELP CODING SIMULATION IN J-DSP

VECTOR QUANTIZER

Speech compression via vector quantization (VQ) is achieved by encoding a set of analysis parameters in vector form. In addition to speech coding applications, VQ has found several applications in pattern recognition, speech recognition, image processing, etc. In speech coding, typical parameters usually encoded using VQ are: LP coefficients, pitch, gain, voicing decision, excitation, etc. For example, the tenth-order LP filter coefficients obtained from the LP analysis can be vector quantized using a 10-dimensional VQ. The vector quantization block in J-DSP uses the Linde, Buzo and Gray (LBG) algorithm to design codebooks. Figure 9 shows the dialog window of the vector quantizer block, and the various options provided. These include viewing the designed codebook vector, mean-squared error (MSE) curve, etc. The codebooks can be designed based on training vectors generated with in the VQ block, or obtained from a different block. The Codebook lengths supported are, c = 2, 4, 8, 16, 32, 64, and 128. The training vector lengths

provided are: c, $2c$, $5c$, $10c$, and $20c$. The vector sizes provided are, k = 1, 2 … 5.

An overview of the exercises that have been designed to highlight the underlying features of the VQ is given below.

- Design codebooks based on training vectors generated with in the VQ block or obtained from a different block, and test the VQ for various test signals.
- Improvement in the SNR values by increasing the number of training vectors per entry
- Evaluate the designed codebooks based on the segmental and over-all SNR values for the test vectors with in the training sequence or outside the training sequence.
- Effects of choosing different vector sizes on the codebook design, for a fixed codebook and training sequence lengths.

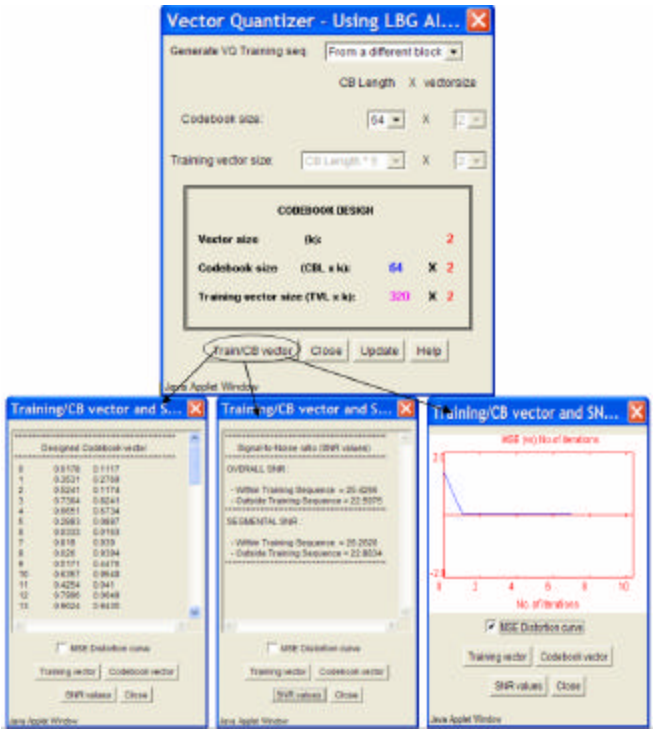

FIGURE 9
VECTOR QUANTIZATION BASED ON THE LBG ALGORITHM

CONCLUSION

This paper presented J-DSP simulation modules and exercises to teach speech and audio compression techniques to undergraduate and graduate students. The J-DSP simulation program is not only useful to undergraduate and graduate students but also to engineers in the industry concerned with the understanding and implementation of speech coding algorithms.

The exercises designed based on the J-DSP simulation tool, may be used by instructors in a class setting to demonstrate key signal processing concepts associated with the processing of speech in digital cellular phones and other applications.

ACKNOWLEDGMENT

The J-DSP project was funded in part by the NSF CCLI program grant DUE 0089075.

REFERENCES

[1] Spanias, A., "Speech Coding: A Tutorial Review", in the *Proc. of IEEE*, Vol. 82, No.10, pp. 1541-1582, Oct. 1994

[2] Painter, E. and Spanias, A., "Perceptual Coding of Digital Audio", in the *Proc. of IEEE*, Vol. 88, No. 4, pp. 451-515, Apr. 2000

[3] Atti, V., *Development of MATLAB and JAVA software models for Algebraic CELP Algorithms*. Master's thesis, Arizona State University, Tempe, Arizona, U.S.A., Dec, 2000.

[4] Spanias, A., et al, "Development and Evaluation of a Web-Based Signal and Speech Processing Laboratory for Distance Learning", *Proc. of IEEE ICASSP-2000*, Istanbul, Vol. 6, pp. 3534-3537, June 2000.

[5] Spanias, A., et al, 'On-line laboratories for speech and image processing and for communication systems using J-DSP", in the *Proc. of IEEE, 2nd DSP-Education workshop*, Pine Mountain, Oct 13-16, 2002.

[6] Argyris, C., *Development of an Internet based digital signal processing tool for on-line simulations.* Master's thesis, Arizona State University, Tempe, Arizona, U.S.A., 1999.

[7] Atti, V., and Spanias, A., "A Simulation Tool for Introducing Algebraic CELP (ACELP) Coding Concepts in a DSP Course", in the *Proc. of IEEE, 2nd DSP-Education workshop*, Pine Mountain, Oct 13-16, 2002.