Microprocessor Implementation of an LPC-Based Isolated Word Recognizer

John G. Ackenhusen

Bell Laboratories
Murray Hill, New Jersey 07974

L. R. Rabiner

Acoustics Research Department
Bell Laboratories
Murray Hill, New Jersey 07974

Abstract

A digital-based isolated word recognition system has been implemented in a module of dedicated hardware that uses a microprocessor and programmable digital signal processing circuitry. The recognizer is based upon the minimum prediction residual principle of Itakura. The recognition algorithm has been developed and tested on a general-purpose minicomputer and array processor, where it has been shown to be suitable for several recognition tasks. The recognition hardware consists of an Intel 8086 16-bit microprocessor operating in parallel with a digital speech processing peripheral (DSPP) tailored to the algorithm. The microprocessor performs the supervisory and decision operations; the DSPP performs the 200,000T + 4,500N multiply-add operations (and associated data transfers) associated with the recognition of a word of duration T sec from an N word vocabulary with 1 template per word. The recognizer is compact (board area of 250 sq. in.) and inexpensive (commercial component cost of about $1200 for 40 word templates).

1. Introduction

Most microprocessor implementations of speech recognition hardware preclude the precision afforded by digital processing of speech by selecting analog preprocessing of the speech waveform for feature extraction. This is because digital-based speech recognition presents a heavy computational load. Feature extraction by digital means requires on the order of 200,000 multiply-add operations per second of speech, and usually requires complicated hardware to perform in real time. Hardware for analog feature extraction, on the other hand, requires much less speed to operate in real time. For example, one common analog feature extractor uses 16 parallel channels of bandpass filters followed by rectifiers and low pass filters with the channel output digitized and recorded every 10 msec. This requires only 1600 read operations per second of speech. The recognizer described here uses a microprocessor and peripheral speech processing circuitry consisting of large scale integrated circuits to achieve near real time response in a compact, all digital module.

The recognition algorithm is based upon a feature set of linear predictive coding (LPC) parameters and pattern comparison using dynamic time warping (DTW), as proposed by Itakura in 1975.[1] Since then, the Acoustics Research Department at Bell Laboratories has carried out tests on a version of the recognizer which uses a Data General Eclipse minicomputer and CSP MAP-200 array processor. These tests used experienced and inexperienced talkers speaking over dialed-up telephone lines to examine the performance of most aspects of the recognition algorithm. The tests included speaker-trained isolated word recognition,[1,2] speaker-independent isolated word recognition,[3] connected digit recognition,[4] methods of endpoint detection,[5] techniques of dynamic time warping,[6] and procedures for training.[7] Other tests have imbedded the recognizer in systems which used vocabulary partitioning, directory searches, and syntactic analysis to perform such voice-activated tasks as repertory dialing of telephone numbers,[8] retrieving telephone directory information,[9] and making airline reservations.[10] In all simulations, the recognizer was shown to attain performance sufficient for practical use over phone lines for a wide variety of talkers. Therefore, the recognizer may be considered understood sufficiently to warrant the design of dedicated hardware.

The goal of this effort was to develop dedicated hardware and software that 1) met the computational requirements of the algorithm with sufficient speed to provide a reasonable (<1 sec) response time, 2) was compact, economical, and independent of a host minicomputer, and 3) was sufficiently versatile to follow the evolution of research activities in more advanced speech recognition based upon the same acoustic processor (speaker independence, connected words, syntactic analysis). Therefore, a widely-used, well-supported 16-bit microprocessor was combined with a special-purpose signal processor designed for the recognition algorithm. The microprocessor is thus fully supported with development systems and high-level languages. These can be used to simplify programming of global, non-computational functions such as process control, vocabulary partitioning, and syntax analysis. (However, programming for the acoustic processing is in assembly language due to speed requirements.) The signal processor is referred to here as DSPP (Digital Speech Processing Peripheral). The DSPP, while presently constructed of commercial components, was limited to a complexity that did not exceed single-chip integration capabilities of current VLSI technology.

2. System Architecture

The recognizer consists of two processors sharing common address, data, and control buses (Fig. 1). The host processor is a 16-bit microcomputer based upon the Intel 8086 microprocessor. The second processor is the DSPP, which performs the bulk of the 300,000 multiply-add operations required during the recognition of a word from a 40 word vocabulary. Each processor has its own data and program memory and the two can run simultaneously. The DSPP functions as either 1) a real-time autocorrelation analyzer for three simultaneous analysis frames (Mode 1), 2) a comparator of feature vectors, performing the Itakura log-likelihood distance

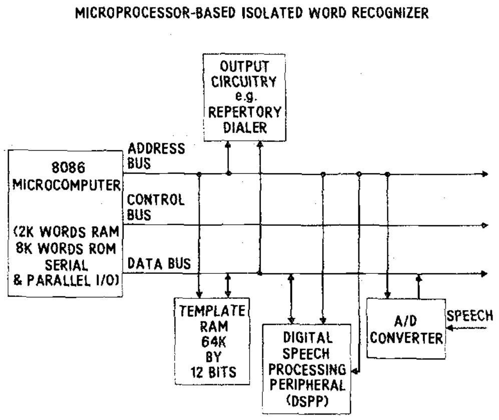
MICROPROCESSOR-BASED ISOLATED WORD RECOGNIZER
OUTPUT CIRCUITRY e.g. REPERTORY DIALER
8086 MICROCOMPUTER
(2K WORDS RAM 8K WORDS ROM SERIAL & PARALLEL I/O)
ADDRESS BUS
CONTROL BUS
DATA BUS
TEMPLATE RAM 64K BY 12 BITS
DIGITAL SPEECH PROCESSING PERIPHERAL (DSPP)
A/D CONVERTER
SPEECH

Fig. 1 Block Diagram of Hardware for Word Recognition

CH1610-5/81/0000-0746 $00.75 © 1981 IEEE

measure to compare one test vector to several sequential reference vectors (Mode 2), or 3) a general-purpose multiplier, to which the microprocessor writes operands and reads products by standard memory operations (Mode 3).

The recognizer is contained on five S-100 bus cards and is partitioned as follows (Fig. 2): 16-bit microcomputer with program and scratch-pad memory and input and output ports (1 card); DSPP (2 cards); 16K 12-bit words of dynamic random access memory for storage of up to 160 reference templates (1 card); speech prefiltering and digitization circuitry and user space for custom output circuitry (e.g. telephone dialer) (1 card).

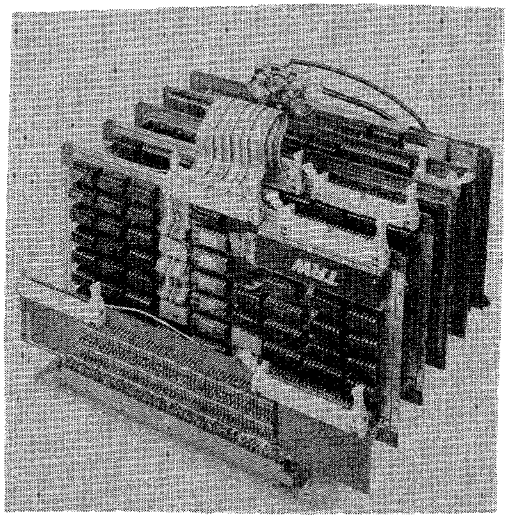

Fig. 2 Photograph of Word Recognition Hardware

3. Execution of Computations

The recognition algorithm is shown in block diagram form (Fig. 3) and has been described in detail elsewhere.[1,9] In the preparation of reference and test patterns, the autocorrelation method is used to perform an eighth-order LPC analysis on a speech signal digitized at a 6.67 kHz sampling rate. An analysis frame size of 300 samples (45 msec) is used with a new frame beginning every 100 samples (15 msec). Thus, each speech sample falls within three consecutive analysis frames.

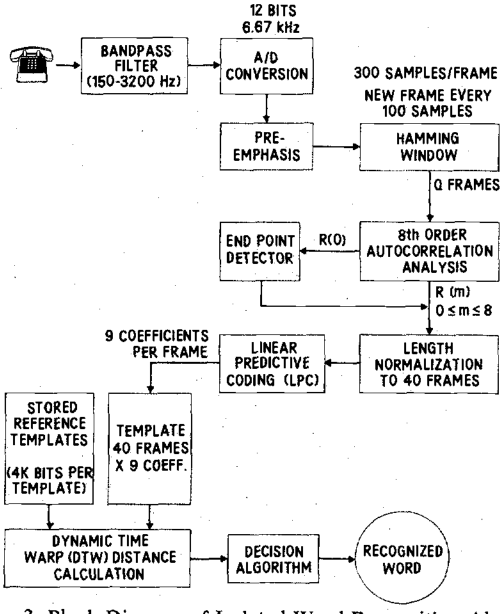
12 BITS
6.67 kHz
BANDPASS FILTER (150-3200 Hz)
A/D CONVERSION
300 SAMPLES/FRAME
NEW FRAME EVERY 100 SAMPLES
PRE-EMPHASIS
HAMMING WINDOW
Q FRAMES
END POINT DETECTOR
R(0)
8th ORDER AUTOCORRELATION ANALYSIS
R (m)
0≤m≤8
9 COEFFICIENTS PER FRAME
LINEAR PREDICTIVE CODING (LPC)
LENGTH NORMALIZATION TO 40 FRAMES
STORED REFERENCE TEMPLATES
(4K BITS PER TEMPLATE)
TEMPLATE 40 FRAMES X 9 COEFF.
DYNAMIC TIME WARP (DTW) DISTANCE CALCULATION
DECISION ALGORITHM
RECOGNIZED WORD

Fig. 3 Block Diagram of Isolated Word Recognition Algorithm

Selection of analysis parameters was verified in a parametric study[11] that examined the effect upon recognition error rate of variations of p (number of analysis poles), L (number of samples between frame), and N (number of samples in the analysis frame). The results indicated that changes that reduced computations (increasing L or N, or decreasing p) led to significant increases in error rate, while changes that increased computation did not provide enhanced performance. Therefore, no sacrifice in choice of analysis parameters was made to facilitate hardware design. The reference templates (autocorrelations of LPC coefficients) were quantized to 12-bit coefficients after scaling each coefficient to use the entire 12 bits. Scaling factors were based on histograms for the various coefficients obtained over several talkers.[11] A simulation indicated that representing templates in this manner did not incur any increase in error rate over the floating-point representation used in the minicomputer studies.

An important difference between the minicomputer simulations and the hardware implementation is the change from floating point arithmetic to fixed point arithmetic. All portions of the computation have at least 16 bits of dynamic range. Both the log likelihood distance calculation and the autocorrelation analysis are carried out with 32 bits of precision. Prior to LPC analysis, the autocorrelation vector for a completed analysis frame is scaled by shifting to attain 16 bits of precision in the zero-order coefficient. Thus, the LPC calculation is carried out with 16-bit precision regardless of frame energy. For the LPC calculation, 16 bits has been shown to be adequate for a pre-emphasized signal at this sampling rate.[12]

We next examine the partitioning of computation among the various sections of hardware shown in Fig. 1, beginning with feature extraction.

The recognizer accomplishes most of the feature extraction in synchrony with the incoming speech samples. This has several advantages. The entire waveform of the utterance is never stored, thus reducing storage requirements from the 100 words of memory per frame required for waveform storage to 9 words per frame to store feature vectors of autocorrelation coefficients. Furthermore, signal energy of all previous analysis frames is available for endpoint detection. This allows the recognizer to make a keep/discard decision as speech is in progress, further reducing storage to eliminate frames of silence before and after the word. Finally, most of the operations associated with feature extraction are completed by the time the word ends, reducing response time.

The following operations are performed on a sample-by-sample basis and are completed in the 150 μsec sampling period:

1) speech sample $s(n)$ is read from analog-to-digital converter,

2) signal is preemphasized to yield

$$\tilde{s}(n) = s(n) - as(n-1) \qquad (1)$$

where $a=15/16$.

The following operations are performed three times per sample (once for each analysis frame j, $j=0,1,2$):

3) allocation to overlapping frames j, $j=0,1,2$:

$$\tilde{x}_j(n) = \tilde{s}(n); \qquad (2)$$

where the sample is placed in the final third of one frame ($j=0$), the middle third of the next frame ($j=1$) and the first third of the next ($j=2$);

4) windowing, for $j=0,1,2$;

$$x_j(n) = \tilde{x}_j w(n-LJ), \qquad (3)$$

$$w(n) = .54 - .46\cos\left[\frac{2\pi n}{N-1}\right] \qquad (4)$$

(Hamming window), where L, the shift between frames, is 100 samples and N, the frame size, is 300 samples.

5) autocorrelation update - for j=0, 1, 2 and m=0, 1,...,p;

$$R_n^j(m) = R_{n-1}^j(m)+x_j(n)x_j(n-m), \tag{5}$$

where $R_{n-1}^j(m)$ is the autocorrelation coefficient for sample n-1 given by the recursion formula

$$R_{n-1}^j(m)=\sum_{k=m}^{n-1} x_j(k)x_j(k-m) \tag{6}$$

and

$$R_0^j(m)=0 \tag{7}$$

Steps 1, 2, and 3 are performed by the microprocessor, Step 4 is performed by the microprocessor with DSPP operating in Mode 3, and Step 5 is performed by the DSPP in Mode 1 for all m after receiving $x_j(n)$ from the microprocessor.

Every L samples, an analysis frame is completed and read from the DSPP and the signal energy is used for real-time endpoint detection.[5]

After the end of the utterance, the utterance length is normalized to a fixed length, typically 40 frames for isolated words of short length, by interpolating between frames. Then Durbin's recursion[13] is used to transform vectors of autocorrelation coefficients to vectors of LPC coefficients. The LPC coefficients are then autocorrelated, scaled, rounded to 12 bits, and stored as a reference template. In these computations, the DSPP operates in Mode 3.

The pattern similarity calculation uses the log likelihood distance metric[1] to obtain a distance $d(i,j)$ between frame i of the test pattern and frame j of reference pattern a, as follows:

$$d(i,j)=\log\left[\sum_{m=0}^{p} v^i(m)r_a^j(m)\right] \tag{8}$$

where

$$v^i(m)=\frac{R^i(m)}{E}, \tag{9}$$

$0 \leq m \leq p$ (test pattern), E is the LPC error, and

$$r_a^j(0)=\sum_{l=0}^{p}(a_l^j)^2 \tag{10}$$

and

$$r_a^j(m)=2\sum_{l=0}^{p-m} a_l^j a_{l+m}^j, \tag{11}$$

for $0 \leq m \leq p$ (reference pattern).

Dynamic time warping is used to determine the optimum mapping between test frames i and reference frames j by determining the path $j=w(i)$ to minimize the average distance given by

$$\tilde{D}=\frac{\min\limits_{w(i)}\left[\sum_{i=1}^{NT} d(i,w(i))\right]}{NT} \tag{12}$$

Here, NT is the number of test frames. The DTW algorithm proceeds by calculating an accumulated distance $D(i,j)$ for each reference-test frame index pair allowed by the global constraints,

$$D(i,j)=d(i,j)+\min[D(i-1,j)g(i-1,j), \tag{13}$$
$$D(i-1,j-1),D(i-1,j-2)]$$

where $g(i-1,j)$ is a weighting function to prevent the path from staying flat for two consecutive frames.

During the pattern similarity calculation, the computation is partitioned between the DSPP, which calculates $d(i,j)$, and the microprocessor, which determines the $D(i,j)$. The computation proceeds on a frame-by-frame basis through the test pattern. After computing a distance score for each reference template, the list of distances is ordered and used for the decision operation to complete the recognition.

4. Architecture of Digital Speech Processing Peripheral (DSPP)

The DSPP performs nearly all of the multiply-add operations required during recognition. Its central element is a 16 bit by 16 bit multiplier-accumulator integrated circuit (TRW 1010J). The additional circuitry around the multiplier accesses and stores operands and results for the multiply-add operations. The multiplier receives its 16-bit operands X and Y and outputs the upper word of the their 32-bit product P on the three data buses X, Y, and P (Fig. 4). The lower word of the product is accessed via the Y bus, and the Y and P buses may also be used as a single 32-bit bus for preloading the accumulator and retrieving results.

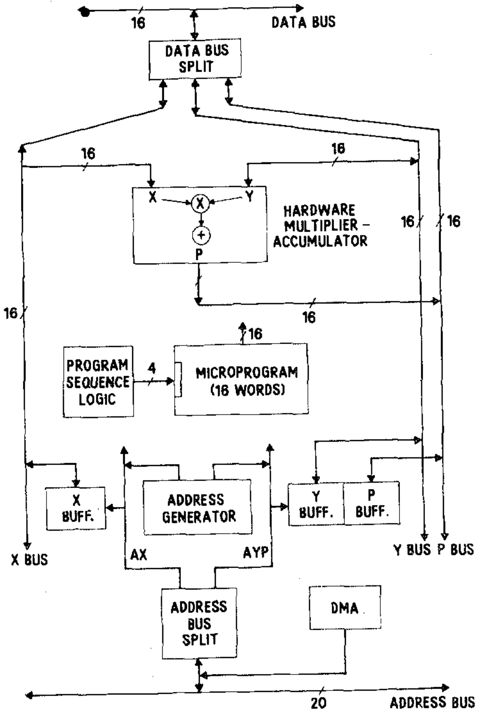

Fig. 4 Hardware Structure of DSPP

Each data bus has its own 16 bit, 64 word data buffer. The X buffer and YP buffer pair have separate address buses, AX and AYP. A programmable address generator circuit provides the desired addresses. To control the operation of the DSPP, a 16-bit-wide microprogram of 16 words is loaded into program memory on the DSPP. The output microprogram bits fall into the four categories of multiplier control, buffer address generation, buffer read/write control, and program sequence control. The final block of the DSPP is the direct memory access channel, which is used to bring reference feature vector coefficients from template memory into the multiplier via the Y bus.

It is instructive to examine the means by which the DSPP retrieves operands and stores results for the multiplier. The multiplier may be preloaded with a 32-bit value into its P port, may then be loaded with operands into ports X and Y, and will then calculate $[X]*[Y]+[P] \rightarrow P$. In this notation, the symbol $[\alpha]$ indicates the contents of address α; $[\alpha]=x$ indicates that the location with address α contains the value x. In the following discussion, symbols α_X, α_{YP}, and α_S refer to addresses on the AX, AYP, and main system address buses, respectively. The addresses X, Y, and P refer to the ports of the multiplier. The arrow, →, indicates direction of data transfer.

In Mode 1, the DSPP is performing the calculation described by (5). The DSPP iterates through all m values, m=0, 1,...,p (inner loop) for constant j, then iterates through j values j=0, 1, 2 (outer

loop). Mode 1 operation proceeds as follows:
for each n, iterate through (14)-(18) for $j=0,1,2$:

$$x_j(n)\rightarrow Y \tag{14}$$

iterate through (15)-(18) for $m=0,1,\ldots,p$:

$$[\alpha_{YP}^{n}(m,j)]=R_{n-1}^{j}(m)\rightarrow P \tag{15}$$

$$[\alpha_X^{n}(m,j)]=x_j(n-m)\rightarrow X \tag{16}$$

$$[X]*[Y]+[P]\rightarrow P \tag{17}$$

$$P\rightarrow[\alpha_{YP}^{n}(m,j)]=R_n^{j}(m) \tag{18}$$

The address sequences are given by

$$\alpha_X^{n}(m,j)=(m-n)_{Mod\ p+1}+Sj \tag{19}$$

$$\alpha_{YP}^{n}(m,j)=m+Sj \tag{20}$$

where S, the address displacement between pages of different j, is 16. In (19), the address sequence is circular with each new sample $x_j(n)$ overwriting the oldest sample $x_j(n-1-p)$.

In Mode 2, the DSPP is performing the calculation indicated by (8). The test feature vector of frame i is first written to the DSPP:

$$v^i(m)\rightarrow[\alpha_x(m)] \tag{21}$$

for $m=0,1,\ldots,p$. The global constraints on the warping path define a minimum and maximum reference frame index, j, as a function of test frame index i. These constraints determine the reference frames to which the test frame is compared. The DSPP iterates through the following steps for all allowed j, $j_{\min}(i)\leq j\leq j_{\max}(i)$:

$$0\rightarrow P \tag{22}$$

iterate through (23)-(25) for $m=0,1,\ldots,p$:

$$[\alpha_S(m,j)]=r_a^j(m)\rightarrow Y \tag{23}$$

$$[\alpha_X(m)]=v^i(m)\rightarrow X \tag{24}$$

$$[X]*[Y]+[P]\rightarrow P \tag{25}$$

After completing above loop, continue:

$$P\rightarrow[\alpha_{YP}(j)]=\sum_{m=0}^{p}v^i(m)r_a^j(m). \tag{26}$$

The address sequences are given by

$$\alpha_S(m,j)=\alpha_S(0,j_{\min})+m+(p+1)j \tag{27}$$

$$\alpha_X(m)=m \tag{28}$$

$$\alpha_{YP}(j)=j-j_{\min}. \tag{29}$$

The DSPP is an independent processor that performs the feature extraction and frame-to-frame similarity measurement about 80 times more rapidly than possible with the microprocessor alone. The DSPP is the key to using a standard microprocessor system for the intense computation required by digital-based speech recognition. During recognition, the DSPP proceeds at an average speed of about 1 μsec per multiply-add, with each multiply-add consisting of the steps of 1) calculate X and Y operand addresses, 2) send X and Y operands to multiplier, 3) calculates address of accumulating sum, 4) preload multiplier with accumulating sum, 5) multiply X and Y, 6) add to the accumulator, 7) write the result to the appropriate YP buffer, 8) increment iteration counter.

5. Conclusion

We have described the hardware implementation of a speaker-trained isolated word recognizer with the following features: 1) recognition achieved by digital processing of the speech waveform according to the principles of minimum prediction residual, 2) recognition algorithm and analysis parameters supported by minicomputer simulations, 3) greater processing speed, smaller size, and lower cost than minicomputer and array processor of simulations, 4) operation over telephone lines, 5) host processor which is an industry-supported microprocessor, 6) custom digital peripheral processor which is of complexity not exceeding single chip integration capability.

Although the hardware implementation of a word recognizer could be done more simply using analog feature extraction, by duplicating the algorithm and analysis parameters of a familiar digital system, we maintain the ability to 1) make experimentally valid choices concerning all features of the recognizer, 2) evolve naturally into more advanced recognition tasks based on simulation results, 3) simulate and analyze recognizer performance in unanticipated adverse conditions imposed by practical use, and 4) exploit the ever-increasing ability to achieve large-scale integration of digital circuits.

References

[1] F. Itakura, "Minimum Prediction Residual Principle Applied to Speech Recognition," *IEEE Trans. Acoust., Speech, Signal Processing,* Vol. ASSP-23, pp. 67-72, February 1975.

[2] A. E. Rosenberg and F. Itakura, "Evaluation of an Automatic Word Recognition System over Dialed-Up Telephone Lines," *J. Acoust. Soc. Amer.,* Suppl. 1, Vol. 60, p. 512 (Abstract), 1976.

[3] L. R. Rabiner, S. E. Levinson, A. E. Rosenberg, and J. G. Wilpon, "Speaker-Independent Recognition of Isolated Words Using Clustering Techniques," *IEEE Trans. Acoust., Speech, Signal Processing,* Vol. ASSP-27, No. 4, pp. 336-349, August, 1979.

[4] C. S. Myers and L. R. Rabiner, "Connected Digit Recognition Using a Level Building DTW Algorithm," *IEEE Trans. Acoust., Speech, Signal Processing,* (to appear).

[5] L. F. Lamel, "Methods of Endpoint Detection For Isolated Word Recognition," M.S. Thesis, Massachusetts Institute of Technology, June, 1980.

[6] C. S. Myers, L. R. Rabiner, and A. E. Rosenberg, "Performance Tradeoffs in Dynamic Time Warping Algorithms for Isolated Word Recognition," *IEEE Trans. Acoust., Speech, Signal Processing,* (to appear).

[7] L. R. Rabiner and J. G. Wilpon, "A Simplified, Robust Training Procedure for Speaker-Trained, Isolated Word Recognition Systems," *J. Acoust. Soc. Amer.,* Vol. 68, No. 5, pp. 1271-1276, November,1980.

[8] L. R. Rabiner, J. G. Wilpon, and A. E. Rosenberg, "A Voice-Controlled Repertory-Dialer System," *Bell System Tech. J.,* Vol. 59, No. 7, pp.1153-1163, Sept. 1980.

[9] B. Aldefeld, L. R. Rabiner, A. E. Rosenberg, and J. G. Wilpon, "Automated Directory Listing Retrieval System Based on Isolated Word Recognition," *Proc. IEEE,* Vol. 68, No. 11, pp. 1364-1379, November, 1980.

[10] S. E. Levinson and K. L. Shipley, "A Conversational Mode Airline Information and Reservation System Using Speech Input and Output," *ICASSP 80 Proceedings,* pp. 203-208, Denver, CO, April, 1980.

[11] L. R. Rabiner, J. G. Wilpon, and J. G. Ackenhusen, "On the Effects of Varying Analysis Parameters of an LPC Based, Isolated Word Recognizer," *Proc. of 100th Meeting of the Acoustical Society of America,* (Abstract), 1980.

[12] J. D. Markel and A. H. Gray, Jr., *Linear Prediction of Speech,* Berlin: Springer-Verlag, 1976.

[13] J. Makhoul, "Linear Prediction: A Tutorial Review," *Proc. IEEE,* Vol. 63, pp. 561-580, 1975.