THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 53, No. 6, July–August 1974
Printed in U.S.A.

An Algorithm for Locating the Beginning and End of an Utterance Using ADPCM Coded Speech

By L. H. ROSENTHAL, R. W. SCHAFER, and L. R. RABINER

(Manuscript received December 10, 1973)

We describe a simple algorithm for locating the beginning and end of a speech utterance. The algorithm is based on the fact that the code words for an adaptive differential (ADPCM) representation of speech exhibit considerable variation among all quantization levels during both voiced and unvoiced speech intervals while, because of a constraint on the minimum step size, during silent intervals the code words vary only slightly within the smallest quantization steps.

The use of the algorithm is illustrated for automatically locating the beginning and end of vocabulary entries for a computer voice response system.

I. INTRODUCTION

The need to automatically locate the beginning and end of a speech utterance frequently arises in speech processing for automatic speech recognition and speaker verification. We have also encountered this problem in implementing an automatic vocabulary preparation scheme for a multiline computer voice response system.[1] Since our solution to the problem of automatically locating the endpoints of an utterance is based on some unique properties of the adaptive differential PCM (ADPCM) representation of speech waveforms,[2] we must first discuss the fundamentals of ADPCM waveform coding.

II. ADPCM SPEECH CODING

Two characteristics of speech signals that are of concern in digital coding are the wide range of amplitudes of speech sounds and the redundancy of the speech signal. The ADPCM coder depicted at the top of Fig. 1 is based on a conventional differential PCM structure

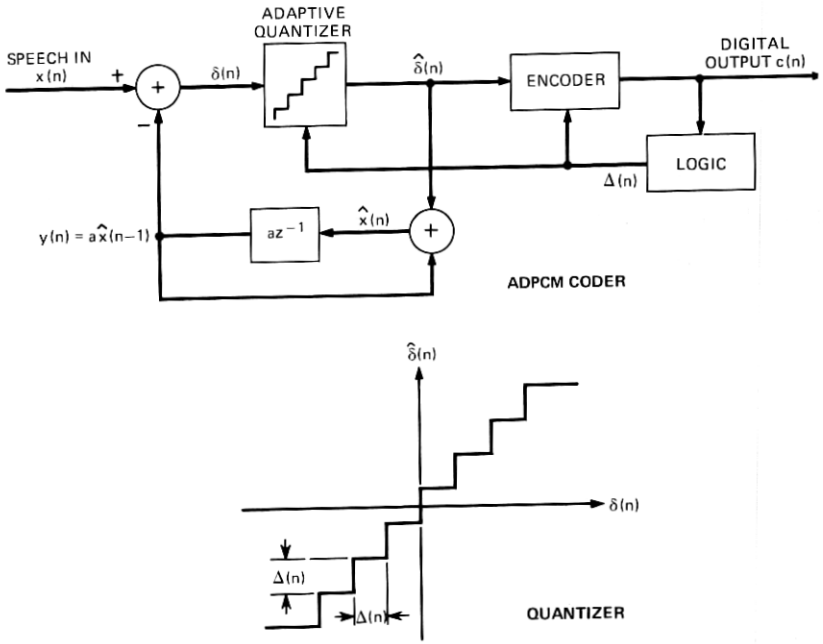

Fig. 1—Adaptive differential PCM (ADPCM) coder showing adaptive quantizer.

with a first-order fixed predictor in the feedback loop. To remove some of the redundancy, we form the difference between the input speech sample, $x(n)$, and an estimate of the input, $y(n)$. If the estimate of the input is good, then the difference should be small and, thus, more accurately represented by a fixed number of bits than the input samples. The difference signal is quantized and encoded for transmission or storage in a computer memory. An approximation to the input speech, $\hat{x}(n)$, is reconstructed by adding the quantized difference to the estimate $y(n)$. The next estimate of the input speech is obtained by linear prediction based on the previous value of the reconstructed signal, $\hat{x}(n-1)$.

The difference signal, although somewhat less redundant, still has a wide range of amplitudes. To make most efficient use of the quantization levels, the peak excursion of the signal should be matched to the range of the quantizer. Thus, for low-level signals such as fricatives, the absolute amplitude value of the step size should be small compared to that required for high-level voiced sounds. In our hardware implementation, we use a four-bit or 16-level quantizer; however, for simplicity we show a three-bit quantizer at the bottom of Fig. 1.

The block labelled LOGIC monitors the coded output and provides for adaptation of the step size on the basis of the most recent quantizer output. For example, if the previous code word corresponds to one of the extreme levels, the quantizer is overloaded and the step size should be increased. On the other hand, if the previous code word corresponds to one of the lowest levels, the step size should be decreased. The step size $\Delta(n)$ satisfies the following equation:

$$\Delta(n) = M \cdot \Delta(n-1),$$

where $M > 1$ if it is determined that the step size should increase and $M < 1$ if the step size should decrease. The details (see Ref. 2) of implementing such an adaptation strategy need not concern us here, except to note the rather important fact that there are strong practical reasons for imposing limits on how large or how small the step size may be; i.e., the step size satisfies the equation

$$\Delta_{\min} \leqq \Delta(n) \leqq \Delta_{\max}.$$

The step-size adaptation acts effectively to compress the amplitude variations so that the quantizer treats low-level unvoiced speech signals much the same as high-level voiced speech signals. The objective is that each quantizer level be used a significant portion of time regardless of the absolute amplitude level of the speech. However, when the input amplitude is on the order of the minimum step size, the adaptation logic insures that the step size will seek its minimum value and the difference signal will then fall within the very lowest quantization levels. Thus, when no speech is present at the input, it is expected that the code words will vary only slightly. It is this feature of ADPCM speech coding that is the basis of our endpoint location scheme.

III. THE ENDPOINT LOCATION ALGORITHM

Figure 2 shows a typical code-word sequence at the beginning of a word. Since the sampling rate is 6 kHz and there are 256 samples per line, each line corresponds to roughly 40 milliseconds of the signal. We note that, for the top line and most of the second line, the code words show little activity, remaining for the most part within the middle four quantization levels. This first part of the sequence corresponds to silence. However, at the end of the second line and then for the remaining two lines, the code-word sequence fluctuates much more rapidly and with greater amplitude. This segment corresponds to

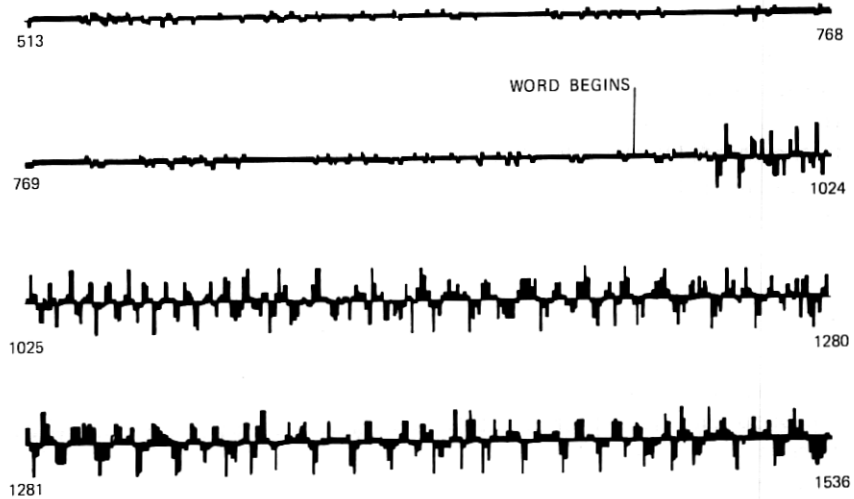

Fig. 2—Code-word sequence for the utterance /o/ showing coded silence followed by voicing.

voiced speech, as is evident from Fig. 3, which shows the decoded speech waveform corresponding to the previous code-word sequence.

These properties of the ADPCM representation of speech are reflected in what we call the *code-word energy*, defined as the sum of squares of the code words over a 101 sample, or 16-millisecond window

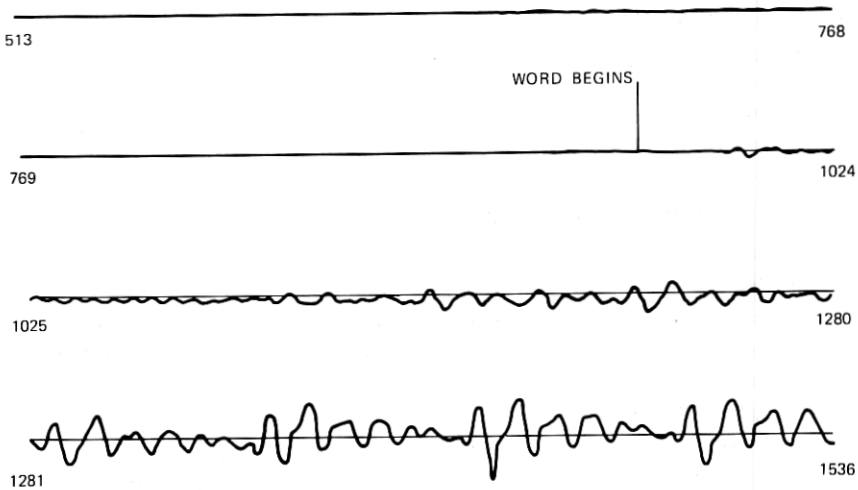

Fig. 3—Decoded speech waveform corresponding to code-word sequence of Fig. 2.

centered around the current sample. That is, the code-word energy $E(n)$ is

$$E(n) = \sum_{i=n-50}^{n+50} [c(i) - 7.5]^2.$$

In our hardware implementation (see Ref. 2), the largest negative quantization level is represented by the binary code word 0000, while the largest positive quantization level is represented by the binary code word 1111, or the decimal number 15. Thus, 7.5 is subtracted from the code words to make the dc level of the code words equal to zero. We have found that performance is only slightly degraded if $|c(i) - 7.5|$ is used instead of $[c(i) - 7.5]^2$. The use of only the magnitude leads to simplifications in hardware implementations of the algorithm.

Using either of these definitions of code-word energy, the endpoint location algorithm is as follows. The code-word energy is computed at each sample and compared with a threshold which is set midway between the measured energy of silence and the average for speech. When the code-word energy exceeds this threshold for 300 consecutive samples or 50 milliseconds, the point at which the energy first exceeds the threshold is recorded as the beginning of an utterance. The energy comparison is continued, and when the code-word energy falls below the threshold for 1000 consecutive samples, or 160 milliseconds, the point at which the energy first falls below threshold is recorded as the end of the utterance. The 160-millisecond criterion ensures that a stop consonant within a word or phrase will not be mistaken for the end of the utterance.

An example of the operation of the above algorithm is illustrated by the sequence of waveforms in Figs. 4 to 9. Figure 4 shows the sequence of code words for the beginning of the word /three/. The left half of the first line shows very little code-word variation and corresponds to low-level tape noise. The right half of the first line and the next two lines, corresponding to the initial fricative /th/, show markedly greater variation, as does the last line which corresponds to the beginning of voicing. The marker in the middle of the first line denotes the beginning point as located by the algorithm just described. Figure 5 shows the code-word energy plotted as a function of time. The marker again denotes the point at which the energy exceeded threshold and remained above for at least 50 milliseconds. Note that the code-word energy is roughly the same for both the voiced and unvoiced segments, while it is significantly lower when no speech is present. These assertions are confirmed by Fig. 6, which shows the actual

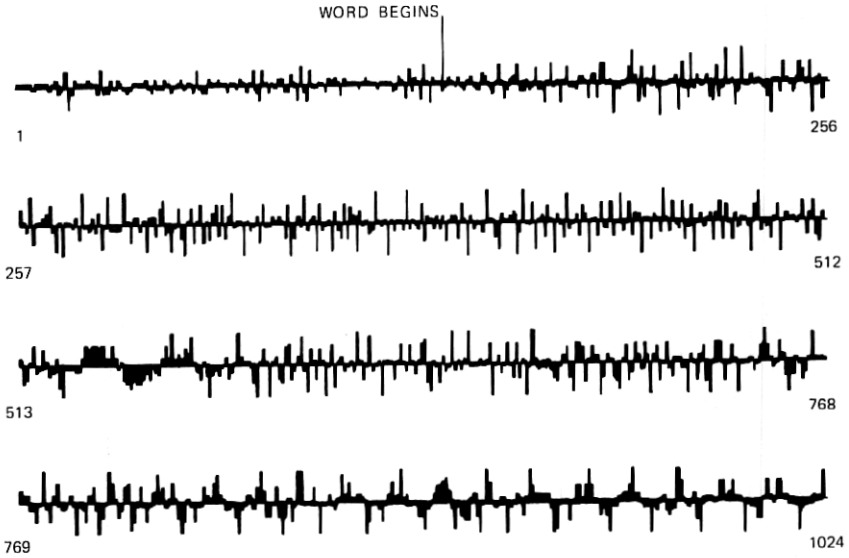

Fig. 4—Code-word sequence for the beginning of the utterance /three/.

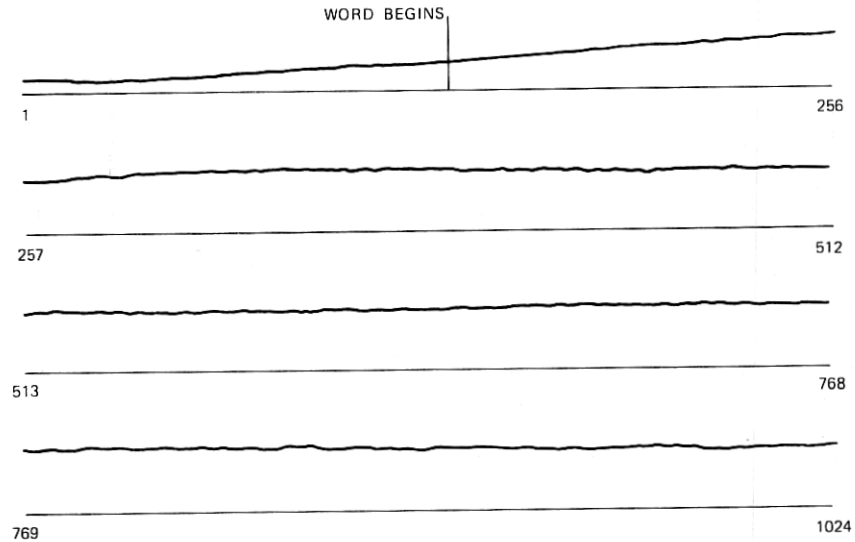

Fig. 5—Code-word energy for the code-word sequence of Fig. 4.

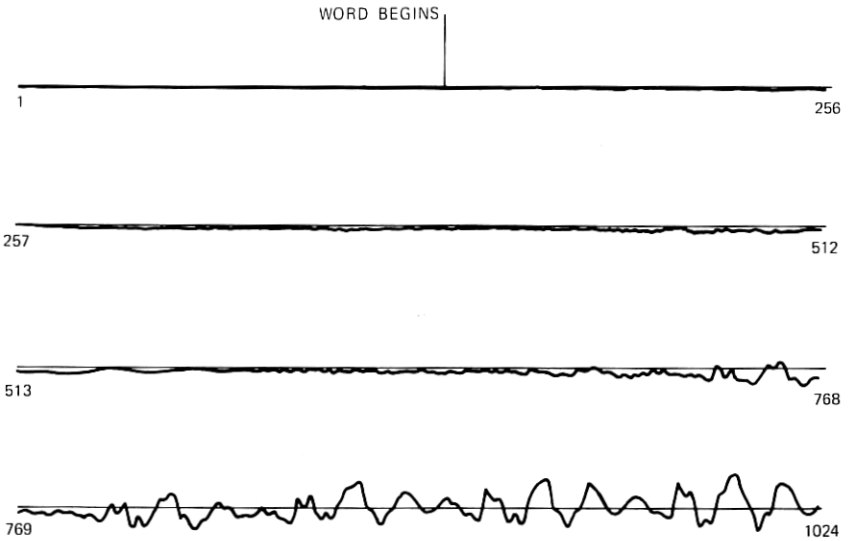

Fig. 6—Decoded speech waveform corresponding to the code-word sequence of Fig. 4.

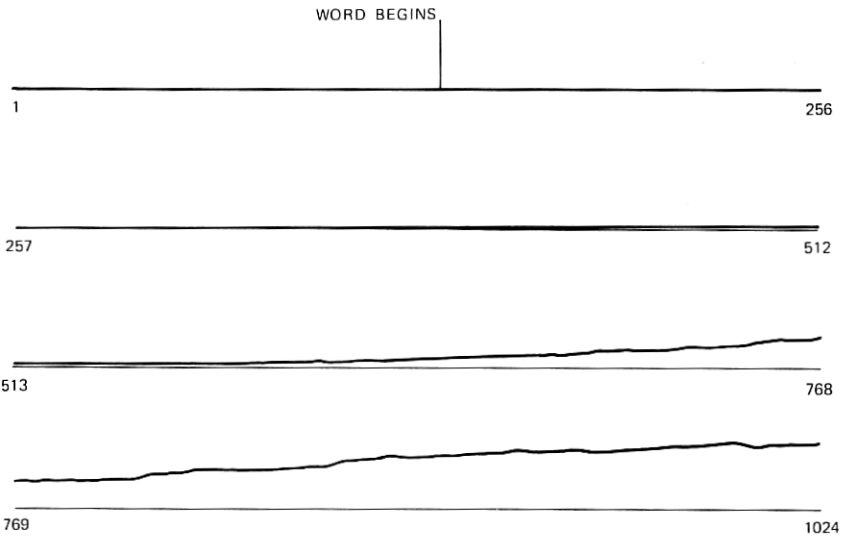

Fig. 7—Energy of speech waveform of Fig. 6.

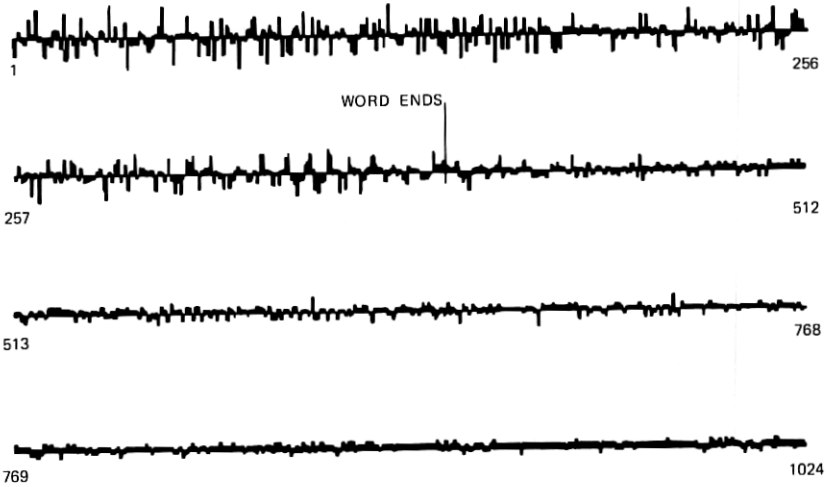

Fig. 8—Code-word sequence at end of the utterance /three/.

speech waveform represented by the previous code-word sequence. We see the beginning unvoiced segment and the following voiced segment. The actual beginning of the word is not nearly as evident as in the code-word sequence. Figure 7, which shows the energy of the speech waveform, emphasizes the fact that the simple algorithm that we have proposed would not be effective when operating upon uncoded samples

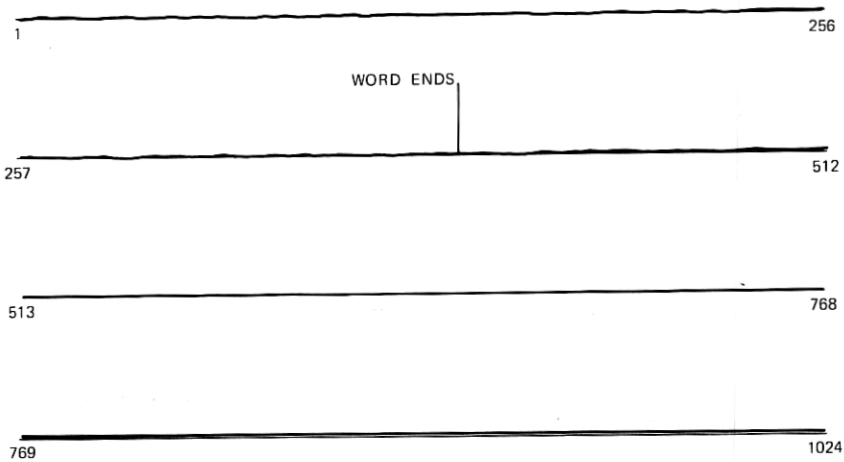

Fig. 9—Decoded speech waveform corresponding to the code-word sequence of Fig. 8.

of the speech waveform. Figure 8 shows the code-word sequence at the end of the word /three/. Also shown is the automatically determined endpoint; i.e., the point at which the code-word energy first fell below threshold and remained below for a period of 160 milliseconds. Figure 9 shows the corresponding decoded speech waveform. The endpoint, which was clearly in evidence in the code-word sequence, is much less prominent in the speech waveform itself.

IV. CONCLUSION

This scheme was tested on a large number of typical entries for a voice response vocabulary with no errors. Auditory and visual inspection indicated no evidence of shortening or inclusion of extra silence for any of the words. The performance of this simple scheme has allowed us to implement a completely automatic system for cataloging words for a computer voice response system.[1]

REFERENCES

1. L. H. Rosenthal, "An Automatic Voice Response System Utilizing Adaptive, Differential Pulse-Code Modulation," M. S. Thesis, Department of E.E., Massachusetts Institute of Technology, June 1974.
2. P. Cummiskey, N. S. Jayant, and J. L. Flanagan, "Adaptive Quantization in Differential PCM Coding of Speech," B.S.T.J., *52*, No. 7 (September 1973), pp. 1105–1118.