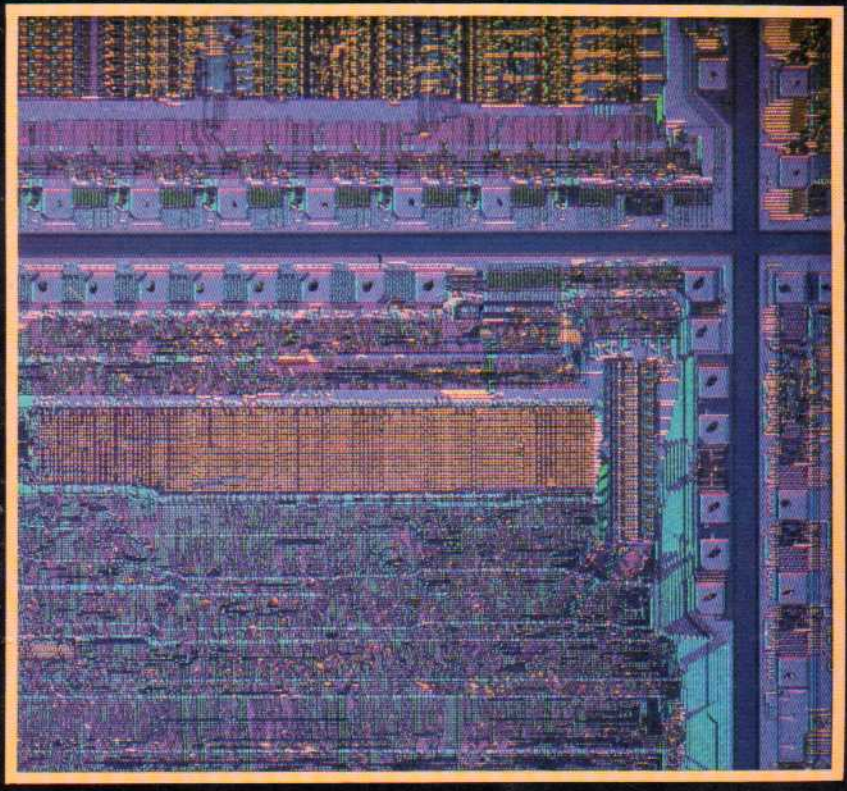
MICRO-PROCESSOR APPLICATIONS HANDBOOK



DAVID F. STOUT

Library of Congress Cataloging in Publication Data

Main entry under title:
Microprocessor applications handbook.

Includes index.
1. Microprocessors. I. Stout, D. F.
TK7895.M5M463 621.3819'58 81-11787
AACR2

1234567890 KPKP 8987654321

ISBN 0-07-061798-8

The editors for this book were Harold B. Crawford and Ruth L. Weine, the designer was Naomi Auerbach (with binding design by Mark E. Safran), and the production supervisor was Teresa F. Leaden. It was set in Melior by University Graphics. Printed and bound by The Kingsport Press

CHAPTER FOURTEEN

Voice Recognition

Frank Koperda

IBM Corp., Information Systems Division, Boca Raton, Florida

HISTORICAL OVERVIEW

Voice recognition, the subject of research for more than 20 years, is a potential application for microprocessors. Initial experimental systems utilized large processors because of the necessary extensive computations. Eventually "minicomputers" were developed; their computing power has increased and new mathematical and programming techniques have evolved. Because today's microprocessor is approaching the computing power of the minicomputer, it is now practical to consider a voice-recognition system based upon a microprocessor.

There is a twofold rationale for a voice-recognition system: (1) It is an easier means for noncomputer professionals to enter data into the computer. (2) In certain applications, such as in semiautomated quality-control inspection procedures, computer users need to use their hands. The cost and generally limited vocabulary of some commercial units have prevented their wide acceptance. One common limitation, for example, is that the system and speaker have to be "trained" to adapt the system to the particular characteristics of

the speaker. Ideally, a truly general voice-recognition system would be capable of recognizing a large vocabulary of words, independent of the speaker.

This chapter describes an experimental system being developed by the author as a hobby. Although not yet totally implemented, major portions of the system have been completed and are operating. Based upon the results to date, the remainder of the system has been planned and is currently being constructed. The approach to the system design utilizes information about the speech waveform that has been reported in the literature as well as characteristics which can be observed on an oscilloscope or plotter.

14.1 System Goals

The audio waveform created from speech is a complex signal of considerable variation, depending upon the speaker's physical characteristics, emotional state, and learned speaking habits.[1] Fundamentally, speech is created by the vibration of the vocal cords, which gives rise to the sounds, such as the vowels. In addition, different sounds are created by air passing through constrictions formed by the tongue, nose, and lips (for example, the /s/ and /f/ sounds). These basic sounds are modified by the size and shape of cavities in the mouth, nose, and other air passages in the head. Differences in all these physical factors cause variations in the speech waveform from speaker to speaker, and even from day to day in the same person.

To be effective, therefore, a recognition system must be able to deal with these variations; that is, the system should be speaker-independent.[2] In speech waveforms, the variations are manifested as changes in frequency, relative amplitude, and time duration. The system must be designed to normalize these factors, to create parameters independent of absolute time, amplitude, and frequency. This system's approach is to use ratios of quantities that can be measured by the system.

Two additional goals are to make the system insensitive to extraneous sounds and to give it the ability to recognize continuous speech. Insensitivity to extraneous sounds is desired so the system can be used in practical applications where background noise must be tolerated. For example, in a manufacturing location, nearby machinery or other people create sounds. Requiring a noise-free environment is not feasible in such a situation. The ability to recognize continuous speech is also a practical consideration. Certain systems quite accurately recognize discrete words, spoken with distinct interword gaps. Continuous speech, consisting of more than one word spoken in a normal sequence, is much more difficult to recognize because the system must determine when one word ends and the next begins. Since many applications require recognition of sequences of words, a system that restrains the speaker to an artificial way of saying phrases is not satisfactory.

The final goal is to design a system that can be implemented using microprocessors. With this restriction, it is not practical to use some of the approaches that have been investigated in the past because many require pro-

cessing power in excess of that available with a microprocessor. For example, many research approaches utilize the fast Fourier transform (FFT) to convert the time-domain signal into the frequency domain.[3] It is not practical to implement a real-time FFT on a microprocessor, so the system described here does not convert to the frequency domain. Similarly, because the microprocessor is not as powerful as larger computers, this design utilizes multiple processors to provide concurrency and hence greater apparent processing power. These are examples of the types of tradeoffs necessary when designing a microprocessor-based voice-recognition system.

14.2 System Overview

Six steps in the recognition of speech are common to most voice recognition systems: (1) converting the input analog signal into a digital form by sampling, (2) compressing or selecting the relevant data for subsequent processing, (3) determining the boundaries of the word, (4) detecting patterns within the word, (5) pattern classification, and (6) association of pattern sequences with words in the vocabulary.

With a powerful processor, it is possible to time-share the processor among these six tasks. With microprocessors, however, it is more practical to use several processors in a "pipelined" architecture. The first processor in the pipeline performs one or more of the six tasks and passes the resulting data on to a second processor. The first processor can then continue on the next speech segment while the second performs subsequent steps in the processing. The number of processors required depends upon the complexity of the steps, the speed of the processor, cost considerations, and the desired accuracy.

Figure 1 is a block diagram of the pipelined system described in this chapter. It incorporates three separate microprocessors in addition to the signal-

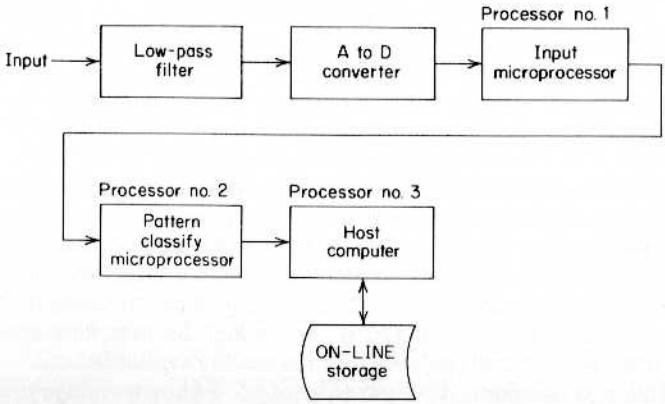


FIG. 1 *Block diagram of pipeline speech-recognition system.*

input hardware. The input stage consists of a microphone preamplifier whose characteristics attenuate frequencies above about 8 kHz. The analog-to-digital converter (ADC) changes the input to a digital representation of the analog voice signal. Processor 1, the input microprocessor, receives the data from the ADC and determines when a voice signal, as opposed to noise, is present. It also detects signal peaks (local maxima and minima) and records their amplitude and the time interval between peaks. This significantly reduces the amount of data that must be handled in subsequent processing. The selected data is passed on to Processor 2, which decides when the compressed data constitutes a pattern and then generates parameters based upon the type of pattern. Processor 3, the host computer, uses the sequences of parameters to decide which word has been spoken and to take the programmed action, such as print the word or control an external device. The pipeline configuration essentially follows the six processing steps in the recognition procedure. Sections 14.3 to 14.8 provide information about accomplishing each of the six steps and details about the specific configuration that has been implemented.

An alternate approach, one that reduces computational time at the expense of additional hardware, is to separate the speech into 8 to 15 frequency bands using filters and then sample the filter outputs. The frequency and amplitude information can then be used directly in Fourier series and other frequency-domain computations. In essence, the fine detail of the speech waveform has been transformed from the time domain to the frequency domain. However, the filters are relatively expensive and implementing them digitally requires significant computing power, which is not consistent with the desire to use current microprocessors.

14.3 Input-Signal Processing

The speech waveform for the word "ship" shown in Fig. 2 illustrates the type of signal that must be processed by the system. The first processing step is to sample the analog waveform and to convert the amplitude of the signal at the sampling instants into a digital word that can be manipulated by the computer. Important considerations in this conversion are the sampling rate and filtering requirements.

Analog-to-Digital Converters (ADCs) The basic equipment used to convert an analog signal into its digital equivalent is the ADC, which is essentially a digital voltmeter that can be controlled by the processor. Many different techniques are available to perform the conversion, including successive approximation, voltage-to-frequency conversion, and dual-slope integration.[4] The successive-approximation (S/A) method is fast enough for sampling speech, and low-cost integrated circuit (IC) versions are readily available.

The S/A technique compares the input voltage and a known voltage to generate the successive bits in the binary representation. First the unknown input voltage is compared to a reference voltage equal to half the full-scale range

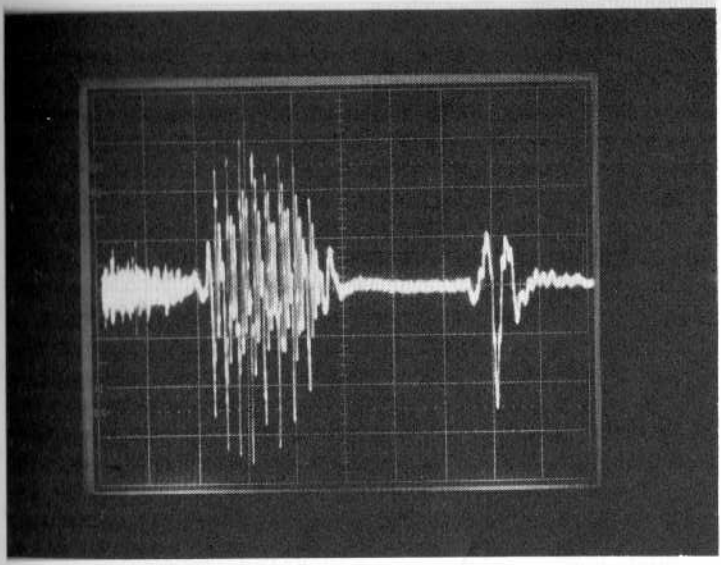


FIG. 2 *Pattern of the word "ship."*

of the ADC. A decision is made as to whether the input voltage is greater than the reference voltage. If it is, the most significant bit is set to binary **1**. The next comparison occurs between one-half the reference voltage added to one-fourth the reference voltage. The second bit is set to **1** if the signal voltage is greater than this comparison voltage. By proceeding in this fashion, the total binary representation of the input signal value is generated. This comparison process is inherent in the IC, so the processor merely needs to initiate the conversion and read the data at its completion.

IC ADCs are available with up to 12 bits resolution (1 part in 4096), with 8 bits being quite common and inexpensive. Based upon experimental results obtained with the system, it appears that an 8-bit ADC is adequate for the approach described in this chapter.

The speed requirement for the ADC is determined by the sampling rate and the desired time resolution. As discussed in the next section, a minimum sampling rate of 6 kHz is required. However, to provide time resolution, the conversion time should be only a small fraction of the 160-μs period associated with the 6-kHz sample rate. In the implemented system, an ADC having a conversion time of 20 μs is used; this is equivalent to a sampling rate of 50,000 samples per second.

Another way to encode data is pulse-code modulation (PCM). There are several forms of PCM, including delta modulation (DM) and differential pulse-code modulation. These alternative analog conversion techniques are discussed in later sections.

Sampling Rate Requirement The frequencies in a speech signal range from somewhat less than 100 Hz to in excess of 8 kHz. The Nyquist criterion for sampling states that a periodic signal can be described completely by $2f$ periodic samples, where f is the highest frequency component of the signal. Thus, accurate representation requires a sampling rate of at least 16,000 samples per second or a sampling interval of about 60 μs. This requires, however, that there be no frequencies greater than 8 kHz; in reality, there are higher frequencies present due to harmonics. To prevent errors caused by inadequate sampling speed ("aliasing errors"), the input signal should be band-limited by using a low-pass filter with a cutoff frequency of about 8 kHz. In the implemented system, the sampling rate of 50,000 samples per second allows a signal bandwidth of 25 kHz which, when coupled with the characteristics of the microphone equalization preamplifier, ensures that the sampled data accurately represents the speech signal.

Noise Suppression The physical environment in which a person speaks is seldom free of background noise. To help suppress these extraneous signals, a unidirectional microphone with a windscreen is recommended. This type of microphone helps ensure that the speech signal will have a significantly higher amplitude than that of the background noise.

Low-frequency (high-pass) filters also can help reduce noise generated by low-frequency mechanical equipment such as electric motors. These filters can be implemented by using analog techniques. However, the microprocessor can provide their digital equivalent. Random sounds with a duration of up to several milliseconds can be efficiently detected and eliminated with a microprocessor. The approach used here is further discussed in Sec. 14.5, *Discrete Word-Boundary Determination.*

14.4 Data Compression

The amount of data generated by sampling the voice signal at 16,000 samples per second would require a large amount of storage if each sample amplitude were stored. In addition, this quantity of data would overburden the microprocessor in the subsequent processing steps. As a result, it is desirable to find some technique either of reducing the amount of data stored or of storing it more efficiently, without losing the information needed for subsequent recognition.

One technique for reducing the amount of data is to use a different encoding scheme. For a continuous waveform, such as a speech signal, DM can be used. This approach encodes the differential change (the delta) in the signal magnitude between sampling instants. If the differential is less than a preset value, no data is recorded, thus significantly reducing the amount of data for slowly varying signals. In practice, a delta modulator can be implemented with an integrator having a constant input with reversible polarity.

Starting at time $t = 0$, the output of the integrator is a ramp (the integral of a constant), as shown in the first segment of Fig. 3. This output is compared to

the input signal at each clock time. When the integrator output exceeds the input signal, the polarity of the integrator input is reversed. The integrator output now decreases, as in the second segment of Fig. 3. When the integrator output becomes less than the input signal, the integrator input polarity is again reversed. In this way the integrator's output "tracks" the signal input using a triangular wave that lags slightly behind the differential changes in the input. By recording the times at which the integrator polarity is reversed, a digital representation of the signal is obtained. Either the number of clock pulses in each interval or a **1** or a **0** at each clock pulse can be recorded.

The precision with which the integrator output tracks the actual signal depends upon the clock rate and the magnitude of the integrated voltage. With the proper selection of these parameters, and depending upon the desired precision (quantization resolution), the amount of data can be reduced significantly, as low as 2 kbyte/s for speech signals.

A further improvement (see Fig. 4) can be obtained by using several fixed input levels to the integrator. When the input is changing rapidly (i.e., has a high slope or large differential), a large integrator input voltage is used, thus allowing the integrator output to "catch up" to the input signal more rapidly. When the differential is small, a smaller integrator input voltage is used, thus minimizing overshoot of the integrator. To encode the signal, both time of reversal and indication of which integrator input voltage is used must be recorded.

In both approaches, determining the actual value of the input requires a numerical integration of the recorded data. The recognition algorithm used in the system described here depends on ratios of actual amplitudes of the input

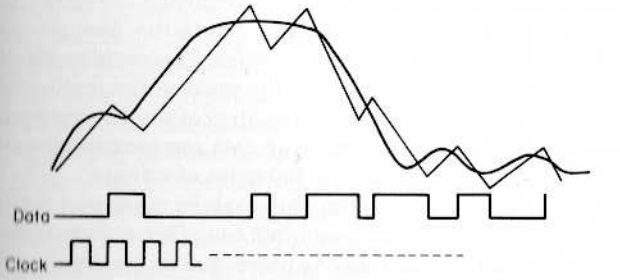


FIG. 3 *Delta modulation.*

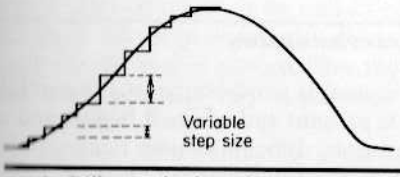


FIG. 4 *Differential pulse code modulation.*

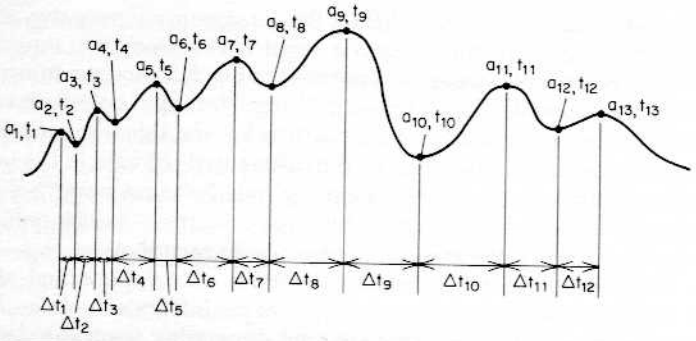


FIG. 5 *Frequency and amplitude separation.*

signal. For this reason the DM encoding technique was not used because the requirement for numerical integration would have increased computation, even though the amount of data stored might be less.

Using a similar idea, however, the system employs immediate processing of the data, to select only those data points needed in subsequent processing. As the data from the ADC is received, it is immediately compared to the previous sample to determine if a change in the sign of the slope has occurred. If it has, the amplitude value is stored. In addition, the value of the time interval since the last slope change is recorded. In a sense, the approach is a peak detector in which the input signal is approximated by a triangle wave of variable amplitude and frequency, as shown in Fig. 5.

This approach also separates the amplitude and frequency characteristics of the speech signal to the extent that the signal frequency is related to the time intervals between slope reversals. This is called the frequency and amplitude separation (FAS) method of data collection. The reciprocals of the time intervals do represent true frequencies in the sense of the mathematical description of sinusoids. However, they are indicative of the frequency components in the waveform. As an indication of the data compression provided, most small words (like "ship") require about 500 bytes of storage.

The FAS method does require that the time intervals be measured with reasonable precision. Experimentally, it appears that sampling at least every 80 μs (12,500 samples per second) is required. As noted earlier, the implemented system uses a 20-μs interval (50,000 samples per second), providing more than adequate time resolution.

14.5 Discrete Word-Boundary Determination

The next step in a voice-recognition system is word-boundary determination, the process of determining if a word is present and where it begins and ends. Although this seems to be a trivial process, difficulties arise because of such factors as background noise and electronic interference. There are at least

four signal characteristics that can be used to distinguish speech from noise: amplitude, frequency, duration, and sound patterns. Individually, none of these characteristics is sufficient for all cases, but, when used in conjunction with each other, an effective means of word detection is possible.

Because of the multiple characteristics being used, word-boundary determination must be done in several stages within a system. Easily identified word boundaries are determined at the time of input, to allow isolated (discrete) words to be identified. For continuous speech, where one word rapidly follows or is slurred into another, word-boundary determination must be done at a later step of the recognition process.

Variable-Amplitude Threshold A variable-amplitude threshold input section is useful for determining the start and end of valid voice input. This technique is used to lower the threshold amplitude until a signal that exceeds the threshold voltage is received. The pattern-recognition component in the system identifies the data as background noise and increases the threshold amplitude. After a delay that is short compared to the length of a valid speech sound, the threshold amplitude is reduced slightly and the sequence repeats. By using this approach, the system continues to search for a valid speech signal without burdening the system with erroneous noise data.

An interesting phenomenon that occurs in speech can cause end-of-word-detection problems. At the end of a word, the vocal chords have lost their excitation but are still oscillating. The speech waveform begins as erratic decay that sometimes can be interpreted as multiple speech dead times. This decay can cause an ambiguity in defining the exact end of a word. To help more clearly delineate the end of a word, the variable-amplitude threshold can again be used to advantage. When the interval between slope reversals exceeds 10 ms, the threshold amplitude is increased and the sampling rate is reduced to 1,000 samples per second (1-ms intervals). After a short delay, the original threshold level is restored, but sampling still continues at 1-ms intervals. When the input amplitude again exceeds the threshold amplitude, the sampling rate increases to its maximum rate.

There is another phenomenon that can cause false word-end detection. When a word consists of multiple phonemes (the basic acoustic unit of words), there is a finite transition time between phonemes that can be greater than 10 ms. It is very useful to be able to detect these transition times for future processing. After a 10-ms period has elapsed, the pause may be caused by a true end-of-word or by an end-of-phoneme. Further sampling of the input data at the 1-ms time interval and the original amplitude threshold can determine which case exists. If a signal amplitude greater than the threshold is encountered after 10 ms, then an end-of-phoneme signal is saved. If multiple consecutive 10-ms periods exist, during which no signal is detected, then the pause is the end of a word. The 10-ms criterion used was determined experimentally and may vary in the final implementation.

Frequency The frequency of the input signal, as reflected in the time interval between slope reversals, may also be used to distinguish between noise

and the presence of voiced input. Many types of background noise (such as that caused by motor-driven machinery) have a low frequency. Although an analog filter could be used to block this noise, a microprocessor can provide a similar function. The microprocessor can check the frequency of an input and determine if it is below 300 Hz. If the FAS method is used for data compression, the microprocessor can reject all input data until the interval between slope changes exceeds about 1.5 ms, or half the period of a 300-Hz sinusoid. Similarly, the endpoint of a word can be determined by the lack of a minimum 300-Hz input. Thus, the microprocessor acts as a fast-attack and fast-decay comparator and low-frequency filter. Even if the background noise is loud, false detection is minimized.

Pattern Another way to distinguish between noise and voice is to examine input patterns. Most noise above 300 Hz is random and of short duration. By examining the pattern and the length of the noise, it is possible to ignore large-amplitude noise. On the basis of experimental results, if an isolated sound lasts less than 100 ms, a word is not in progress and the preceding data can be ignored.

Continuous Speech These techniques have proved effective for rejecting noise and identifying word boundaries in discrete speech or in continuous speech in which words are distinctly separated. Sometimes, however, these techniques may not be effective when speech habits result in connected speech. For example, the southern "you all" is often spoken as the single word "y'all," and therefore must be considered as a distinct word for recognition purposes. Because the specific system being described has not been implemented completely, its ability to recognize continuous speech has not been evaluated. Most likely, additional techniques will be required for word-boundary detection, or certain combinations of words will have to be considered as single words for recognition purposes.

Detection Algorithms The preceding requirements for noise suppression, data compression, and word boundary detection using the FAS approach enable the construction of an algorithm for the initial processing of the speech signal. Figure 6 is a flowchart of an algorithm that has been experimentally successful for accurately tracking voice input. On a standard 8-bit microprocessor, the algorithm takes about 100 lines of assembler code.

The top third of Fig. 6 samples for a slope reversal and should be less than 80 μs (preferably faster), so that peak times can be accurately determined. The section on the right side of Fig. 6 shows that changing the sampling rate has no effect on the amount of data needed to represent the waveform. Only when the slope of the waveform changes sign are any data saved. Thus, increasing the sampling rates simply provides more precision in determining the time between peaks. When a slope reversal occurs, the maximum or minimum amplitude can be either saved or forwarded to the next microprocessor. The time between slope reversals also is saved, and the scanning continues looking for the next slope change.

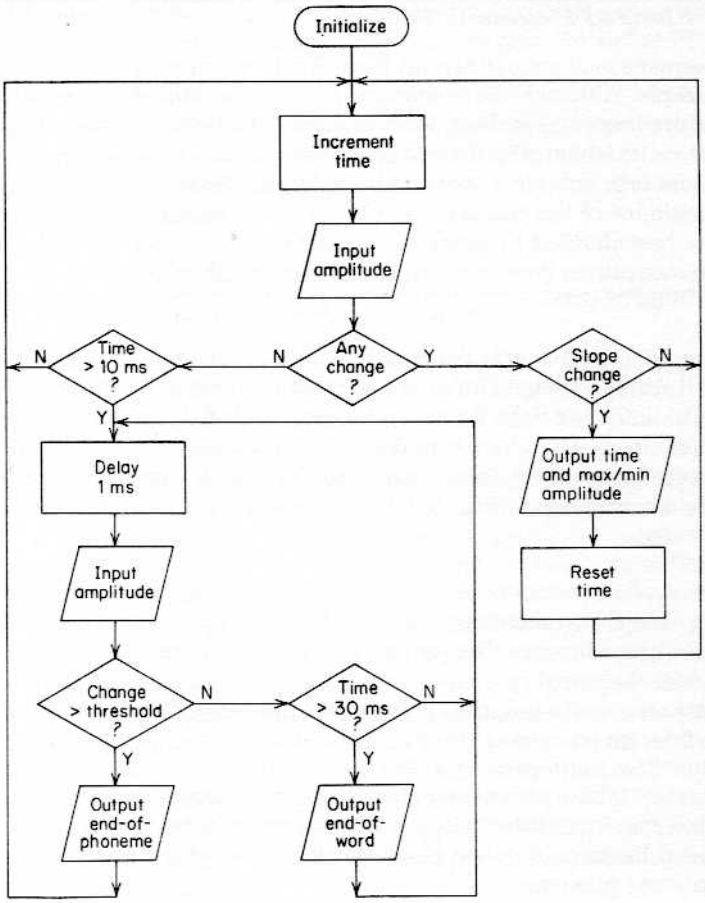


FIG. 6 *Flowchart of FAS algorithm.*

The lower-left section of Fig. 6 suppresses data output during speech dead times, such as between words or during some consonants. A slow sampling rate and variable-amplitude threshold help eliminate background noise and false end-of-word detection. Because data is not saved until a word actually exists, a considerable amount of data need not be analyzed subsequently. For simple words (e.g., the word "one"), the FAS method reduces the input waveform to only 256 to 512 bytes of data.

As a side benefit, data resulting from the FAS method allows very simple speech-output hardware. A digital-to-analog converter (DAC) may be used in conjunction with the transition time. This data is sent to the DAC to generate triangular wave shapes that approximate the speech waveform. For higher-quality output, the microprocessor can construct a sine-wave segment between two points.

14.6 Definitions of Linguistic Terms

The subsequent sections use certain linguistic terms that may be unfamiliar to most people. Although the definitions are not considered technically rigorous by pure linguists—indeed, there is much disagreement among linguists as to what is the technically correct definition of many of these terms—their purpose here is to provide a common basis for understanding the terms used in the remainder of the chapter. Definitions of the linguistic components of speech are best clarified by numerous examples. This section also illustrates that voice recognition must concern itself mainly with sounds, not the traditional spellings of words.[5]

Morpheme A morpheme is the smallest unit of meaning in language. In English, all words are made up of at least one morpheme. Many morphemes are words in their own right, for example, "pin" and "fish." Some morphemes must be combined with others to make what we refer to as words. For example, the morphemes "un," "pin," and "ing" have their own meanings, with "un" being a negative and "ing" indicating an action, but neither can be used independently.

Phoneme A phoneme is the basic unit of distinctive sound. For example, the word "cat" has three phonemes, with each letter representing a phoneme. This is not always the case because some phonemes require more than one letter, such as the sound represented by "ou" in "house." On the other hand, some letters represent a number of phonemes. For example: "a" can represent the three different phonemes heard in "cat," "father," and "make." Occasionally, a letter does not represent a phoneme at all, as, for example, the silent "e" in "tame." These inconsistencies between spellings and sounds (grapheme-phoneme inconsistencies) are very common in the English language and have led linguists to invent phonetic alphabets, where one symbol represents only one phoneme.

Consonants Consonants are all the letters of the alphabet except the vowels "a," "e," "i," "o," and "u." Taken singly, most consonants represent a single phoneme. There are exceptions, as, for example, the letter "c," which has no sound exclusive of its own; it sometimes borrows the sound of "k" (cat) and sometimes the sound of "s" (cent). Consonants taken in combination produce even more phonemes, as is discussed in the *Digraph* section.

Vowels Vowels are the five letters "a," "e," "i," "o," and "u" ("y" also may be used as a vowel). Every English morpheme contains at least one vowel. Vowels present an extreme number of grapheme-phoneme inconsistencies. Even taken singly, vowels represent more than one phoneme, depending on the spelling of the word (graphemic environment). The sounds of "a" in "pan" and "pane" are affected by the silent "e." Vowels are often "overruled" by consonants; for example, the /o/ sound, as in "boat," is significantly affected by the "overruling r" in "more."

Digraph A digraph is a combination of two vowels or two consonants that results in only one phoneme. Examples are the "ch" in "chair" and the "ai" in "sail."

Diphthong A diphthong is a combination of two adjacent vowels in a morpheme, where each vowel is sounded and each produces sounds it does not produce in other graphemic environments. The "oi" in "oil" and "ow" in "out" are diphthongs.

14.7 Pattern Detection Within a Word

The next processing step in the voice-recognition system begins the analysis phase. There is some uncertainty as to what constitutes the intelligence of speech and what characteristics of a speech waveform allow the brain to recognize words. However, it appears that the intelligence recognized by the brain is related to patterns of frequency and amplitude. Taking this view, speech consists of patterns of several types and transitions between patterns.

It is difficult to describe precisely what constitutes a pattern because of the many waveforms that can occur in speech. A rough definition of a pattern might be multiple occurrences of a particular distinctive waveform.

Observing the waveforms of phonetic sounds on a scope or plotter resulted in identifying three general pattern characteristics. Most unaspirated sounds, such as /s/, appear random in nature, but even randomness can be defined as a distinct characteristic. The vowel sounds are complex periodic oscillations caused by vocal-chord vibrations. Stop consonants are characterized by a short pause followed by a high-amplitude burst of sound. Variations of basic sounds are caused by the mouth, temporary positions of the tongue and lips, and the permanent characteristics of other facial features. Thus, each pattern variation is unique to an individual, and so the patterns must be defined broadly to be useful.

The patterns do not remain constant, even within a phoneme. As one phoneme ends and another begins, there is a transition between patterns that may range from silence to a complex blending of the initial and final pattern. The first processing task, therefore, is to recognize the existence of a relatively stable pattern that, in subsequent analysis, can be classified for use in identifying the spoken word.

The saved input data are the time intervals and amplitudes of maxima and minima. The time intervals are very useful for determining random variation; the amplitude values are used for identifying oscillations. However, neither can be used independently to determine the parameters of a pattern.

Pattern Characteristics The random waveform sounds change slope polarity frequently, and the time between slope reversals is usually less than 160 μs. Potential stop consonants (/b/, /d/, /g/, /p/, /t/, /k/; see Table 2, p. 14-23) are detected in the data-input section or during the pattern-detection phase of analysis and are identified by a pause followed by a high-amplitude signal.

If a 10-ms interval between large amplitude changes occurs, the following pattern is probably a stop consonant.

Amplitude variations associated with periodic oscillations are more difficult to detect. Most types of oscillations observed are damped; that is, the peak amplitudes within a pattern decrease progressively in size, as shown in Fig. 7*a*. The damped type of oscillation can be detected by subtracting successive peak amplitudes, determining the absolute values of the change, and noting that three or more points form a negative slope. The data resulting from performing this algorithm are shown in Fig. 7*b*.

Pattern-Detection Procedure Determining the precise beginning and ending of a pattern is difficult. Simple words such as "one" contain three to four distinct types of patterns, and each type has 10 to 12 repetitions. Detection of these groups of the same pattern type is a complicated problem because of variations in the same pattern and the leading and trailing transitions between patterns.

In the system being described, the detection of the beginning and ending of a pattern depends upon recognition of the random and decaying oscillation signals previously described. As soon as one or the other is detected, the data is tagged as the beginning of a pattern. When the data no longer satisfies the characteristics of the identified pattern, the data is tagged as the end of the pattern and a search for the next pattern begins.

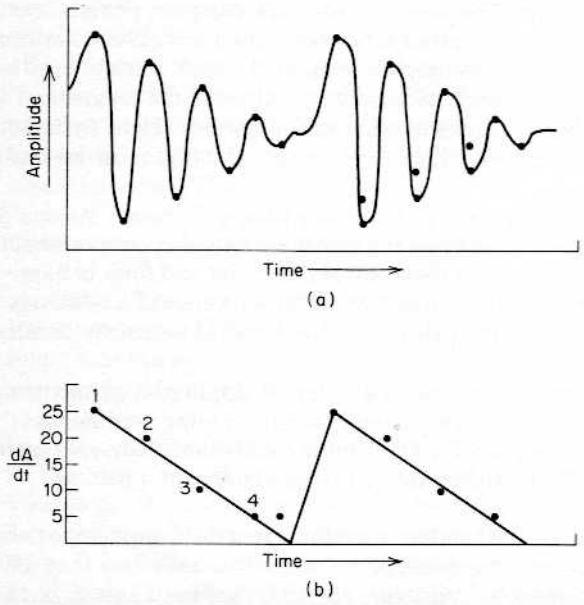


FIG. 7 (a) *Damped oscillation;* (b) *absolute value of change in damped oscillation.*

In more detail, the procedure is as follows. At the identified beginning of a word, the program concurrently examines the data for both randomness, as indicated by slope reversals in less than 160 μs, and decaying oscillations, as indicated by periodic monotonic decreases in the absolute values of the peak amplitudes. If the randomness criterion is satisfied for about 30 slope reversals, the search for damped oscillations is terminated. The random-pattern analysis continues until seven time intervals fail to satisfy the criterion, at which time the end-of-pattern is declared. Similarly, when damped oscillations are found, the end-of-pattern is declared after the appearance of seven slope reversals which do not satisfy the criterion.

After an end-of-pattern is declared, the preceding data are examined to determine the number of slope reversals in the pattern. If there are fewer than about 40, it is assumed that the data represent a transition period and they are ignored in subsequent analysis.

This procedure was derived from experimental results and observations of speech waveforms. Tests to date indicate that it is effective in segmenting the word for subsequent analysis. Due to variations in speakers and utterances of a single speaker, the same word may result in differing members of identified patterns. These multiple patterns must be resolved in the subsequent pattern-classification and word-recognition portions of the system.

14.8 Pattern Parameters and Analysis

After segmenting the word into patterns using the procedures described in Sec. 4.7, the next step is to derive a set of measurement parameters that can be used for the further classification and trends of the patterns. If patterns are described too precisely, the chances of finding them again in the same word are small. If patterns are examined too coarsely, the distinction between vowels, for example, is very difficult. To produce speaker-independent recognition, some degree of variation must be accommodated.

A number of parameters can be used to describe a speech waveform. Some parameters require large amounts of mathematical manipulation, implying significant processing time or additional hardware. Cost, performance, and accuracy must be balanced to create a reasonable system configuration.

Types of Patterns In the system being described, two independent searches of a pattern are made to extract the necessary parameters. One search does a detailed analysis of types of oscillations; another search develops the trends of the pattern. In each search, both time and amplitude data are examined. The analysis is based upon changes in time and amplitude between peaks, and the parameters are expressed as ratios, so as to make the resultant parameters independent of the absolute time and amplitude.

Eighteen parameters currently are calculated in this system; sixteen are grouped into four categories. The remaining two parameters are single values: one is a number and the other is an average taken over the total pattern. The eighteen parameters can be summarized thus:

1. Damped oscillations—amplitude analysis (four ratios)
2. Damped oscillations—time-interval analysis (four ratios)
3. Peaks—amplitude analysis (four ratios)
4. Peaks—time-interval analysis (four ratios)
5. Number of repetitions
6. Average time interval

Damped Oscillations—Amplitude Analysis As described earlier, a damped-oscillation pattern is identified by comparing the absolute value of the amplitudes of successive peaks and noting that three or more of them are monotonically decreasing. Within a pattern of this type, there typically are a number of repetitions of the damped behavior. Within each repetition, the number of oscillations (peaks) appears related to the particular vowel. The existence of the repetitions and of the varying number of peaks in a repetition is illustrated in Fig. 8 for two vowel sounds.

The pattern data based upon these observations is examined to identify the repetitions. The total number of repetitions is determined and saved. Similarly, the number of positive peaks in each repetition is saved. The time intervals between the first five positive peaks also are saved for use in the later time-interval analysis. The saved data are indicated graphically for two repetitions in Fig. 9. For convenience, only the positive peaks are used in this analysis. The concept could be used with both positive peaks (maxima) and negative peaks (minima), but this has been found unnecessary in this experimental system.

If we let R be the total number of repetitions (reps) in the pattern, four ratios arbitrarily named r_1 through r_4 are calculated:

$$r_1 = \frac{R}{\text{no. of } P_2\text{'s in } R \text{ reps}}$$

$$r_2 = \frac{R}{\text{no. of } P_3\text{'s in } R \text{ reps}}$$

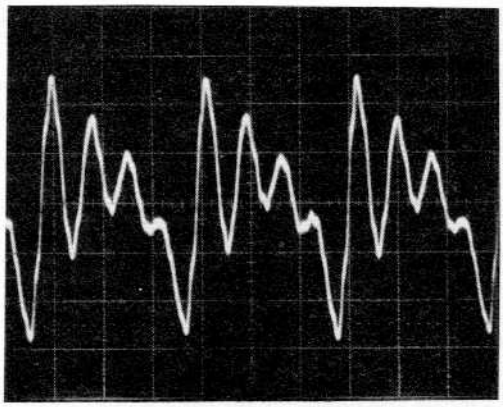
$$r_3 = \frac{R}{\text{no. of } P_4\text{'s in } R \text{ reps}}$$

$$r_4 = \frac{\text{total no. of peaks } (P) \text{ in } R \text{ reps}}{R}$$

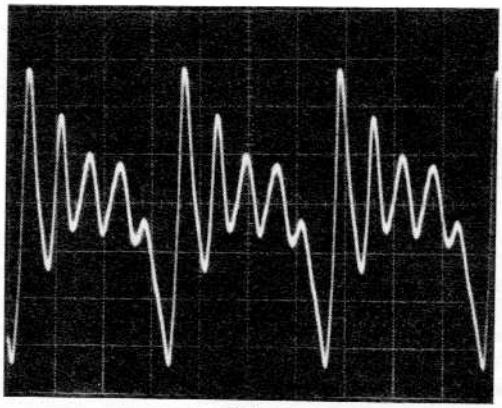
Because it is conceivable that the denominator of some of these ratios might be zero, the program must test its value to avoid an attempt to divide by zero.

All calculations are done by using integer arithmetic, which is usually faster than floating-point arithmetic. This is why the ratios are designed to provide numbers greater than 1. The ratio r_4 provides the average number of peaks in a damped oscillation; this number is a clue as to the type of vowel or pattern. If there are more than 16 damped-oscillation segments ($R > 16$), the pattern is a candidate for classification as a vowel.

Damped Oscillations—Time-Interval Analysis Referring again to Fig. 9, a similar set of ratios, arbitrarily designated r_5 through r_8, are calculated. They



(a)



(b)

FIG. 8 *Patterns of* (a) *the vowel /a/, and* (b) *the vowel /o/.*

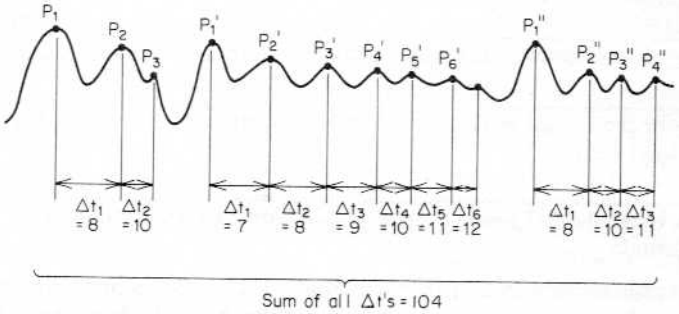


FIG. 9 *Example of damped oscillation analysis.*

represent the average time intervals associated with each of the first four peaks and with the entire pattern.

The ratios are calculated as:

$$r_5 = \frac{\text{sum of } \Delta t_1\text{'s in } R \text{ reps}}{\text{no. of } P_2\text{'s in } R \text{ reps}}$$

$$r_6 = \frac{\text{sum of } \Delta t_2\text{'s in } R \text{ reps}}{\text{no. of } P_3\text{'s in } R \text{ reps}}$$

$$r_7 = \frac{\text{sum of } \Delta t_3\text{'s in } R \text{ reps}}{\text{no. of } P_4\text{'s in } R \text{ reps}}$$

$$r_8 = \frac{\text{sum of all } \Delta t\text{'s in } R \text{ reps}}{R}$$

Again, the program should test the denominator for zero before attempting division. Also note that r_8 is the average time duration of the damped oscillations.

As a specific example of the calculation of r_1 through r_8, consider the waveform in Fig. 9, which shows three repetitions of a damped-oscillation pattern so that $R = 3$. The number of P_1's, P_2's, P_3's, and P_4's, are 3, 3, 3, and 2, respectively. The time intervals are as shown in the figure. Using this data, the ratios are:

3/3, 3/3, 3/1, 13/3, 23/3, 28/2, 20/2, 104/3

Peaks–Amplitude Analysis This analysis provides four additional ratios that describe characteristics relating to the peaks in the waveform. Both positive and negative peaks (maxima and minima) are used, and the ratios represent relative occurrences of certain characteristics compared to the total number of peaks P in all repetitions R. The ratios arbitrarily are designated as r_9 through r_{12}. In the calculation of r_9, "equality" means that the peak amplitudes are equal to within 6 of the 8 data bits; that is, the subsequent peaks vary less than 4 parts out of 2^8, or 256.

The ratios are defined as:

$$r_9 = \frac{P}{\text{no. of sequential peaks almost zero in total pattern}}$$

$$r_{10} = \frac{P}{\text{no. of sequential peaks equal in total pattern and} > \text{zero}}$$

$$r_{11} = \frac{P}{\text{no. of sequential peak amplitudes monotonic (positive or negative) in total pattern}}$$

$$r_{12} = \frac{P}{\text{no. of sequential peaks in total pattern for which differentials alternate}}$$

The last parameter requires further explanation. The denominator is determined by considering three peaks at a time. If $P_1 > P_2 < P_3$, then one differential $(P_1 - P_2)$ is positive, and differential $(P_2 - P_3)$ is negative. This is what is meant by alternating differentials. It is extended to P_4, P_5, . . . , until the condition is no longer true.

Peaks–Time-Interval Analysis This analysis is analogous to that used for the amplitude data. Because the numbers of peaks and time intervals in the pattern differ by only 1, P is used in calculating the ratios. With the same interpretations of equality and alternating differentials, the ratios r_{13} through r_{16} are defined as:

$$r_{13} = \frac{P}{\text{no. of sequential } \Delta t\text{'s} < 280\ \mu s \text{ in total pattern}}$$

$$r_{14} = \frac{P}{\text{no. of sequential } \Delta t\text{'s} = \text{or} > 280\ \mu s \text{ in total pattern}}$$

$$r_{15} = \frac{P}{\text{no. } \Delta t\text{'s monotonic (positive or negative) in total pattern}}$$

$$r_{16} = \frac{P}{\text{no. of sequential } \Delta t\text{'s in total pattern for which differentials alternate}}$$

Parameter-Calculation Procedure The remaining two parameters are the number of repetitions in the pattern (earlier defined as R) and the average time interval in the entire pattern. This is defined as:

$$r_{17} = \frac{\text{sum of all } \Delta t\text{'s in total pattern}}{P}$$

The total procedure is summarized in Fig. 10. The word-boundary and data-compression (selection) procedures were described earlier and are shown in more detail in Fig. 6. This procedure executes very quickly on a microprocessor. It is implemented in about 300 lines of assembler code and requires about 600 bytes of storage.

Pattern Analysis Specific computer routines that examine the parameters described have not yet been completed. As a result, experimental evidence that the 18 parameters are sufficient for voice recognition has not yet been obtained. Development of the pattern-analysis and word-recognition algorithms may result in modifications, additions, or deletions to the set of parameters. However, the work to date suggests that this set may be sufficient. On the basis of manual computation and visual observation of the ratio parameters for many different words, several observations can be made that will be the basis for developing the analysis routines.

There are multiple parameters generated for a pattern, and a decision as to the nature of a particular pattern cannot be made until all patterns are considered. The pattern must be examined as a total entity because of the number of identifying characteristics within that pattern.

The hard consonant (such as /t/ in "two") has as its main characteristic r_{13}, r_{14} (small ratio). Words like "one" (Fig. 11, p. 14–25) and "four" have similar parameters (r_9 through r_{16}) across a pattern and similar oscillation types in the amplitude domain (r_1 through r_4) but vary in the time domain of the oscillations (r_5 through r_8). Unaspirated sounds have a large number of random changes and small average time within a pattern (r_{13}, r_{17}).

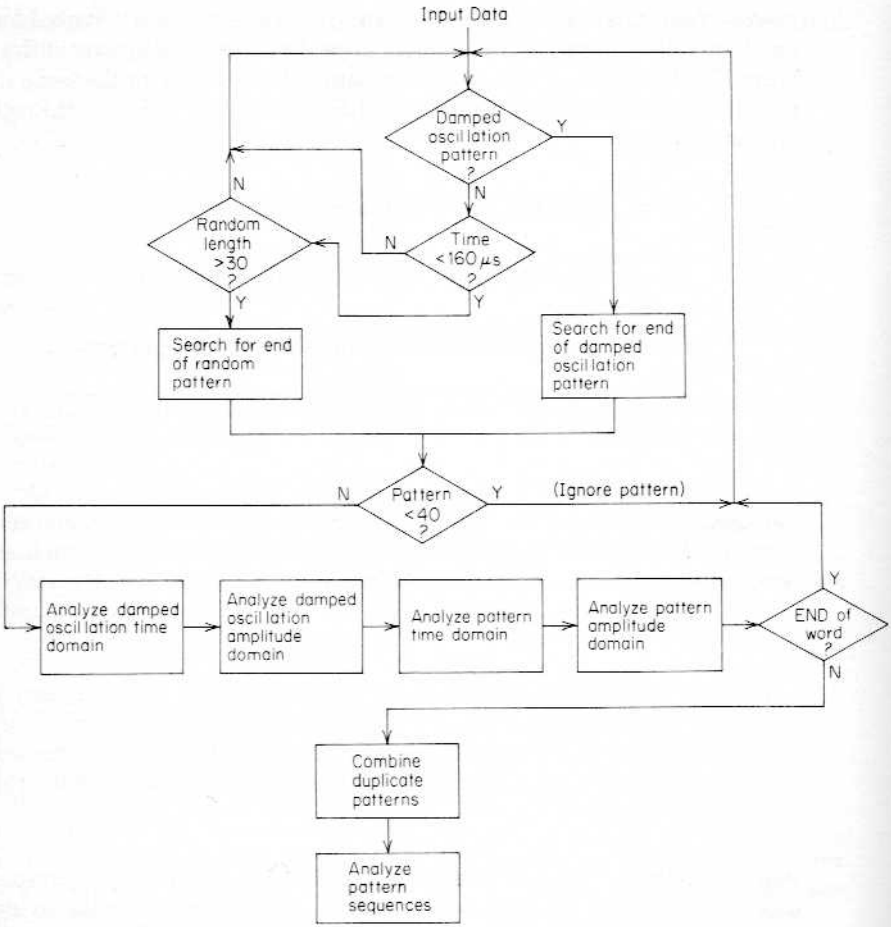


FIG. 10 *Block diagram of pattern classification.*

These observations can be illustrated by using the actual ratios generated by the system for the word "ship." A facsimile of the output is shown in Table 1. Pattern 1 shows a large number of peaks which change slope (r_{12}). The time associated with this change is short (r_{17}), and the short time interval is the predominant occurrence (r_{13}). The large number of damped oscillations (R and r_{15}) is consistent with the idea of randomness because all pattern types can exist in a random pattern. Pattern 1, then, is at the least an unaspirated sound (that is, /s/, /t/, /f/); to help distinguish these ratios (r_5 through r_8) indicate a probable /s/ sound. Patterns 2 to 3 represent the middle of the word. Pattern 3 is terminated by a stop-consonant flag derived by the input processor. Pattern 4 then identifies the /p/ as the stop consonant.

TABLE 1 Parameters for "ship"

	Damped oscillations									Peaks								
	Amplitude					Time interval				Amplitude				Time interval				
	r_1	r_2	r_3	r_4	R	r_5	r_6	r_7	r_8	r_9	r_{10}	r_{11}	r_{12}	r_{13}	r_{14}	r_{15}	r_{16}	r_{17}
Pattern 1	1	1	1	3	109	4	5	6	38	15	2	8	2	2	8	3	16	23
2	1	1	2	3	5	21	32	44	80	42	42	2	2	13	40	2	2	37
3	1	1	1	2	12	8	27	34	128	0	0	1	2	50	50	2	2	39
4	1*	1	1	4	3	7	32	16	64	0	12	2	3	...	10	2	2	48

*Represents an inter-phoneme-gap flag.

It appears that the unique identification of vowels will be very important in the success of this, or any, speech-recognition system. There are many similar words that differ in only the vowel sound, as, for example, "hit," "hat," and "hot." As shown in Table 2, half the phonemes are vowel-like sounds (i.e., vowels, semivowels, and diphthongs). Although characterized by the multiple repetitions of decaying oscillations, unique identification appears to depend upon the ability to identify the frequency content of the sound.

Some techniques for voice recognition depend heavily upon a frequency-domain analysis of the sound. Directly (by use of a filter bank) or indirectly (by the FFT, for example), the relative energy in certain frequency bands has been measured. Vowel-like sounds typically show three or more distinct frequency ranges, called "formants," in which significant energy is concentrated. Table 3 shows the frequencies on which these bands of energy are centered for the vowel sounds.[6] The values are averages taken over different speakers uttering a variety of words.

In this system, the burden of converting to the frequency domain was not considered reasonable, so all analyses are being done in the time domain. However, some frequency information is inherent in the time-interval data that has been saved by the system. The exact relationships have not been derived, but it is expected that the average values of the various time intervals can be related consistently to the formant frequencies. For example, the average length of the repetitions in the first cycle of a damped-oscillation waveform appears to be related to the third formant frequency, and r_8 is related to the first formant frequency.

Phoneme Sequences The concepts discussed serve to associate the individual patterns with phonemes. The procedures do not result in a unique relationship between any given pattern and a single phoneme. The patterns resulting from transitions between phonemes and limitations in the analysis techniques limit the ability to totally and accurately identify phonemes.

However, it is expected that the procedures will narrow the possible phonemes for a given pattern to a manageable number. To a greater or lesser degree, the procedures will permit the construction of possible phoneme sequences that approximate the actual phoneme sequence in the word. Using the phoneme sequences in conjunction with the calculated ratios, a search technique of the dictionary of words, their phoneme sequences, and probable variations of these sequences can be constructed to lead to the actual word recognition.

As a more specific example, the word "six" (phonetically, /siks/) is rather easily separated into a sequence of four types of phonemes: a random sound, a vowel sound, a stop consonant indicated by a pause, and a final random sound. The identification procedures discussed for the vowel sounds can provide more detail about the vowel sound. Similarly, random sounds will be analyzed to separate some of the fricatives, such as distinguishing /s/ from /z/.

An intelligent pattern-analysis algorithm is very necessary for identifying phoneme sequences. Some patterns are meaningless because they merely

TABLE 2 Orthographic representations of phonemes in American English and representative words

Vowels		Semivowels		Diphthongs	
IY	beet	W	wag	eI	bay
I	bit	L	lamb	oU	boat
E	bet	R	rod	aI	buy
AD	bat	Y	yes	aU	how
UH	but			oI	boy
A	hot			jU	few
OW	bough	(These sounds are particularly			
U	foot	dependent upon their location			
OO	boot	in the word and			
ER	bird	surrounding vowels.)			

Consonants					
Nasals		Stops		Whisper	
M	miss	B	big	H	hand
N	knot	D	dig		
NG	sing	G	give		
		P	pin		
		T	tin		
		K	cat		
Fricatives		Affricates			
V	van	DZH	gem		
TH	then	TSH	chin		
Z	zip				
ZH	rouge				
F	fan				
THE	thin				
S	sod				
SH	shove				

TABLE 3 Table of formant frequencies for vowels[a]

Typewritten symbol for vowel	Typical word	F1	F2	F3
IY	beet	270	2290	3010
I	bit	390	1990	2550
E	bet	530	1840	2480
AE	bat	660	1720	2410
UH	but	520	1190	2390
A	hot	730	1090	2440
OW	bought	570	840	2410
U	foot	440	1020	2240
OO	boot	300	860	2240
ER	bird	490	1350	1690

indicate a transition, and some patterns are only repeats of previous patterns. An exact match of the parameters for each phoneme is not possible because of the variations of each voice. What *is* possible is a trend analysis of the patterns. The key parameters of each pattern parameter are used, and a best-fit approach is used for comparison.

14.9 Word-Identification Techniques

The final step in the word-recognition sequence is the association of phoneme sequences to words. This will be done by establishing a dictionary that relates actual words to phoneme sequences and other identifying parameters. Finding the exact word involves searching the dictionary for a reasonable match.

The more powerful 16- and 32-bit microprocessors perform this task more efficiently than do the 8-bit microprocessors. Their powerful instruction sets, larger addressing space, additional addressing modes, and word size can be used to reduce computation time and main storage requirements.

The search technique, the amount of storage, and the desired sophistication of "understanding" will influence the amount of storage required for the dictionary. Simple digit-recognition systems might require as little as several hundred bytes of storage. Very complex phrase- or sentence-understanding systems require multimegabyte dictionaries with syntactical parsing that breaks down a sentence into nouns, verbs, and other components of a sentence.

A number of well-known searching techniques are available. In some cases, combinations of the techniques are used in a sequential procedure. The following sections introduce three of the search techniques likely to be useful in this system: binary, indexed, and hashing.

Binary Search The binary search technique can be very efficient for a reasonable number of entries, such as might be encountered in digit recognition. The lookup table can reside in memory; a search divides the table in half and decides which half the observed phoneme sequence is in. That half is cut in half, and a similar decision is again made. This process continues until the word is found. A table of 1024 entries can be searched with only 10 decisions by using this method.

Indexed Search An indexed search is used for larger vocabularies. A table of parameters is maintained that gives an address of a group of words with similar properties. For example, an index table composed of phonemes can be used for start-of-word sounds. If the first pattern of a word contains an /s/ sound, the /s/ entry has associated with it a starting address for a table in memory. The table contains all words with a starting /s/ sound, their likely phoneme sequences, and other key parameters. This table of phoneme-constructed words can have the English spelling as part of its entry. This spelling entry could be used to directly print the word or to initiate some other computer response.

Hashing Parameters Hashing algorithms take all the key parameters from the phoneme sequence and generate an address based upon some algorithm. Some key parameters that can be used are the number of unique patterns, the placement of vowels within a word, and beginning sounds. Hashing algorithms are very efficient for converting the large number of parameters (40 to 50 bytes) in a word to a 16- to 24-bit address.

14.10 Hardware Implementation

There are many possible hardware configurations for performing voice recognition. The configuration that was implemented is shown in Fig. 1. Now that some of the problems and possible techniques associated with recognition have been mentioned, a more descriptive functional hardware implementation is possible. Using this system, the parameters associated with a word can be determined in about 1.5 s, and several words may be in the process of being analyzed because of the pipeline architecture.

The first processor is the input and includes an ADC that has 128 bytes of random access memory (RAM) and 512 bytes of read-only memory (ROM). The sampling rate of the ADC is 50,000 samples per second. Implemented in the code are the variable-amplitude threshold, word-boundary determination, and the data-compression algorithms. About 500 to 1000 parameters per second are transferred to the next processor for subsequent processing.

Processor 2 has 8 kbyte of RAM and 8 kbyte of ROM. It performs the pattern-detection algorithm and calculates the 18 ratios. The data from Processor

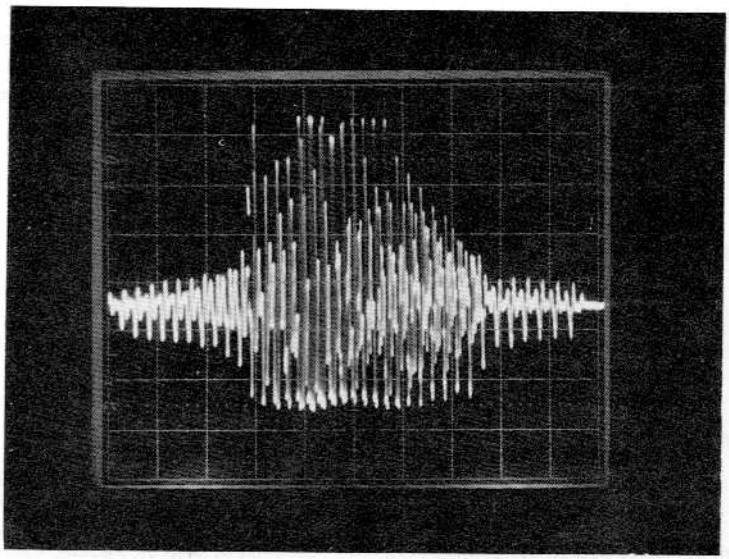


FIG. 11 *Recognition pattern for the word "one."*

1 is received and buffered while the previous word is being analyzed. This processor sends 100 to 200 parameters per second to the subsequent processor.

Processor 3, the system processor, has a standard complement of input-output (I/O), devices, such as keyboard/CRT and disk (diskette). This processor has 64 kbyte of RAM and will perform the dictionary lookup using a combination of indexed and hashing search techniques. It will also perform the response to the verbal request.

14.11 Status

As indicated earlier, not all of this system has been constructed. Processors 1 and 2 have been implemented, and the software up through the calculation of the ratio parameters is operational. The basic operating system kernel is currently being written.

Because the system implementation is incomplete, it is not now possible to verify the accuracy of its voice-recognition capabilities. It is certainly likely that, when completed, the system will not be completely successful initially. Further experimentation, which could involve modifications to the procedures described in this chapter, may be necessary. It is even conceivable that the system cannot reach its initial goals because of limitations in currently available microcomputer processing power or fundamental errors in the experiments, research, or heuristic reasoning upon which the system is based.

Nevertheless, this experimental system represents an investigation that is possible only because of the emergence of the microprocessor. The problem of voice recognition has been a topic of research for more than 20 years and a low-cost, speaker-independent, large-vocabulary system has yet to become commercially practical. Someday professionals will find a microprocessor-based system which meets the elusive goals and which could be of significant value in hundreds of practical applications. In the meantime, voice recognition continues as a fascinating and challenging problem for researchers in this field.

ACKNOWLEDGMENTS

I would like to express appreciation to my wife Bonita for her linguistics support and to Dr. Thomas J. Harrison for his efforts in the preparation of this chapter.

REFERENCES

1. Dinneen, Francis P.: *An Introduction to General Linguistics*, Holt, New York, 1967.
2. Dixon, N. Rex, and Thomas B. Martin: *Automatic Speech and Speaker Recognition*, IEEE Press, New York, 1979.
3. Rabiner, L. R., and R. W. Schafer: *Digital Processing of Speech Signals*, Prentice-Hall, Englewood Cliffs, N.J., 1978.

4. Harrison, Thomas J.: *Handbook of Industrial Control Computers*, Wiley-Interscience, New York, 1972.
5. Heilman, Arthur W.: *Principles and Practices of Teaching Reading*, Merrill, Columbus, Ohio, 1972.
6. Peterson, G. E., and H. L. Barney: "Control Methods Used in a Study of the Vowels," *J. Acoust. Soc. Am.*, Vol. 24, No. 2, March 1952, pp. 175–184.