Give an Ear to Your Computer, Byte Magazine Vol 3, #6 June 1978

By Bill Georgiou

A Speech Recognition Primer for Computer Experimenters

    - You pick up the phone and slowly and clearly you pronounce into the microphone:"Number one two one three five five five one two one two verify." You listen as out of the earpiece an awkward but quite intelligible voice repeats what you just have said. Then you say "Dial number." The sounds of dialing follow, then a ringing in the distance and you find yourself talking to the directory assistance operator in Los Angeles. A sequel to 2001 with personal HALs? Not quite. Just plain old 1977 reality. A couple of cards in your computer and some programming can do it. Speech recognition and voice control have come a long way from the "Open Sesame!" of the Arabian tales. Its technical foundations were laid in the 1950s and 1960s. The microprocessor revolution has brought it within the reach of any computer experimenter, opening up a fascinating frontier of voice control and spoken communication between human and machine.
    - In the following pages I will try to give you an introduction to speech processing and pattern recognition. To demonstrate the principles involved, we will go into some details of the workings of a speech recognizer suitable for a small personal computer. I hope the material to be presented will be enough to give you an idea of what speech recognition is, how it is done and what are its present limitations. It is left to the reader to get excited, read the literature to find more about speech recognition and then buy, borrow or build a recognizer and start using it imaginatively.

- The Speech Signal
    - If we connect a microphone to an oscilloscope and then speak into it, we will get a jittery trace similar to the one shown in figure 1. The vertical axis represents voltage, the output of the microphone. The horizontal axis is time and for this reason such a representation of speech is called a "time domain" representation. You may ask can we use a time domain representation for speech recognition? It would be nice if we could because it is so easy to get; all we need is a microphone. The voltage in the output leads of the mike is, by definition, the time domain signal. Of course we would like to input this signal into our computer. Since it is an analog signal and our computer is a digital machine we will need an analog to digital converter (ADC). The analog to digital converter gives a binary number that corresponds to the amplitude of the signal at the particular time when the measurement is made. This process is called sampling. If we take and store equally spaced samples often enough so that the signal does not change very much between samples we will have a fairly accurate representation of the time domain signal in our computer's memory. Figure 2 shows such a situation, in which an arbitrary input waveform is sampled over some time interval, and the results of conversion are stored as values in successive memory locations.

- It has been mathematically proven (the sampling theorem) that if we are to have an accurate representation of the signal, the sampling frequency should be at least twice the highest frequency in the signal. It is then called the Nyquist frequency and it is the lowest usable sampling frequency. One could sample at higher than Nyquist frequency but this would not give a more accurate representation of the signal. Instead there would be a lot more data words to deal with, an unwelcome situation. If we try to apply the Nyquist theorem to speech we are faced with the question: what is the highest frequency in speech? Well, for high fidelity sound a bandwidth of 20 to 20,000 Hz is necessary. This means that speech has frequencies up to this limit, perhaps even higher. On the other hand, telephone speech is band limited to 3200 Hz and it is still quite intelligible. Since in speech recognition we are interested in what has been said rather than the quality of the sound, if we limit the signal using a filter with a cutoff at 3200 Hz we will retain the information needed for recognition. Then we can use Nyquist's theorem to get a practical sampling rate since we will know that the highest frequency in the signal is 3200 Hz) due to the filter. Sampling at twice that frequency we will get 6400 samples per second of speech. Assuming our word length is eight bits which is about the minimum usable for speech and that the average word duration is about half a second, we will need 3.2 K bytes of memory space per word. If we had a 20 word vocabulary and we wished to store each word in main memory once to be used as reference, we would run out of memory space in any micro and most minis.
- Using the time domain signal is out of the question because of the huge amounts of memory required to store it. This is not the only disadvantage of the time domain signal. Assuming we had enough memory to store the data, processing it would take too much processor time because every operation we would perform on the data would have to be performed to such a great number of data points. Real time operation (processing the signal as it occurs such that there is no appreciable time lag between the moment the signal ends and the time the recognizer decides which word has been said) would be out of the question. Yet real time operation is highly desirable in most situations in speech recognition. Another basic disadvantage of the time domain signal is its variability between different pronunciations of the same word. People do not repeat words exactly the same, down to the minute details of amplitude. Quite the contrary; there is a tremendous amount of variation in the time domain signal even for the same person within several consecutive pronunciations of the same word.
- In speech recognition we would like to transform the time domain signal into some other signal or representation with low data rate and which remains more or less invariant as long as the same word is pronounced. This is easier said than done. If we try to reduce the data rate we run the risk of throwing away important information. For example suppose that before going into the analog to digital converter we rectify the signal, then process it with a low pass filter. A block diagram of this operation is shown in figure 3 together with the resulting waveforms. Such an operation will extract the envelope (c) of the signal. The highest frequency of interest in the envelope is only about 50 Hz. This is due to constraints imposed by the mechanism that produces speech. We can thus set the low pass filter to 50 Hz and sample at 100 Hz, the Nyquist frequency. That gives us a very reasonable data rate, 100 words per second and well within the capabilities of all machines. Yet, if we try to build a recognizer using only this information we would get very poor performance. The reason is that we have thrown away a lot if not most of the information in the speech wave in the process of rectification. The information that remains is quite variable between different pronunciations of the same word, further degrading recognition. But it is not totally useless. If we select a vocabulary of a few words carefully so their envelopes have distinct characteristics we may get usable performance out of a very simple recognizer. We could select for example the words "one," "three," "zero." Typical envelopes for those words are shown in figure 4. We note that the envelope for "one" consists of one hump. "Zero" has two humps and "three" has one main hump preceded by a small peak that corresponds to the sound "th." Based on these observations we can write a simple program that would examine the input data and decide which one of the three words has been said.

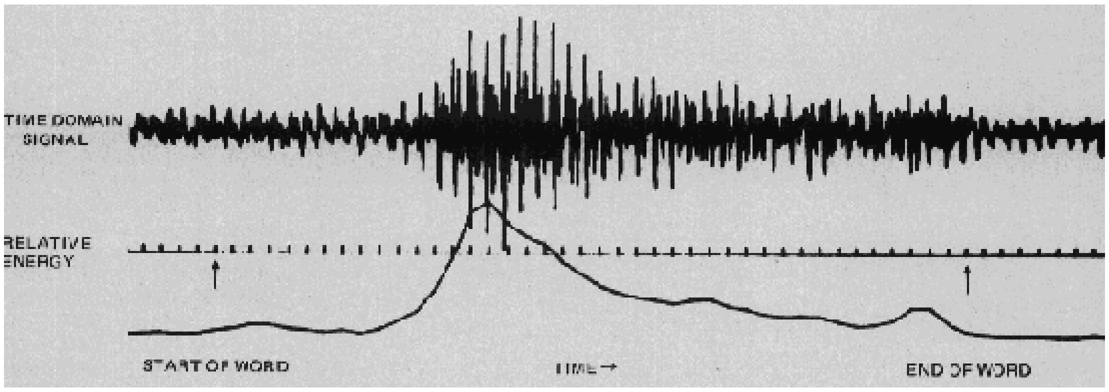


Figure 1: A time domain voice waveform and its energy. The top trace is the time domain signal for the word "three." The bottom trace is the energy in the above signal computed every 70 ms. Note that the signal before and after the word (arrows mark the beginning and end of the word) Is not zero. This is due to background noise pkked up by the microphone, In this case computer coollng fans and air conditioning noise.

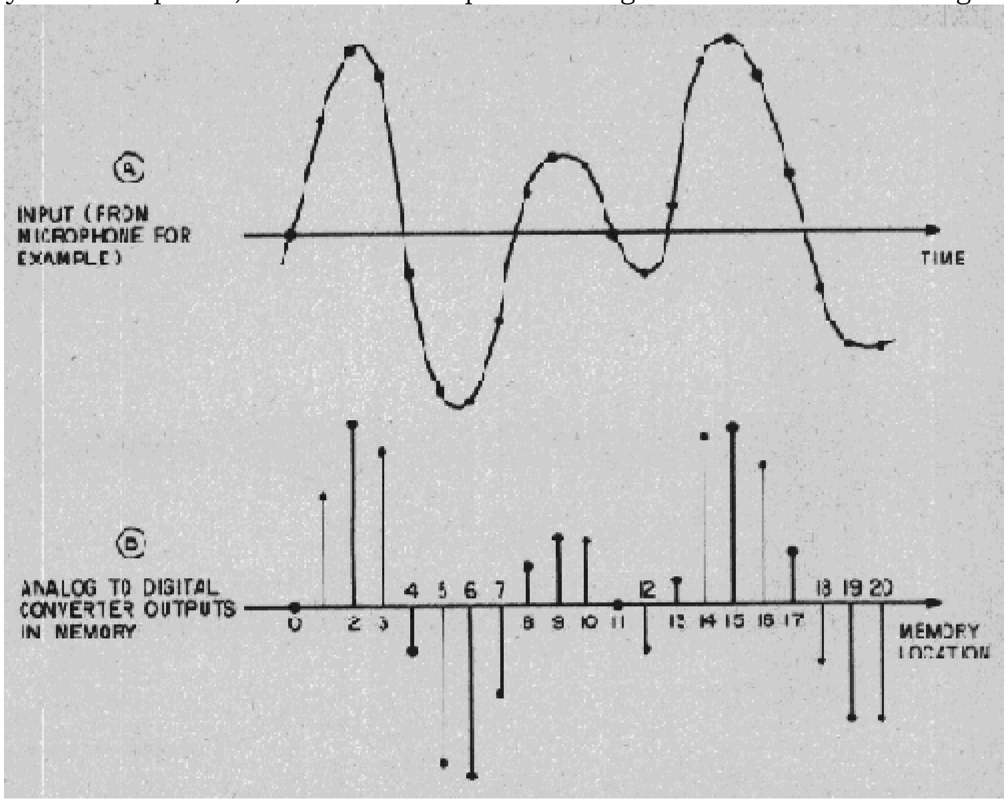

A
INPUT (FROM MICROPHONE FOR EXAMPLE)
TIME
B
ANALOG TO DIGITAL CONVERTER OUTPUTS IN MEMORY
0 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
MEMORY LOCATION

Figure 2: Sampllng in the time domain. Waveform A is a time domain signal. If we sample it at equally spaced intervals we will retain the ampiltude of the waveform at the dotted points and the signal will be zero between these points, as shown in waveform B. Although A and B look very different if the sampling is done with a frequency higher than the Nyquist frequency (see text), both signals will contain exactly the same information.

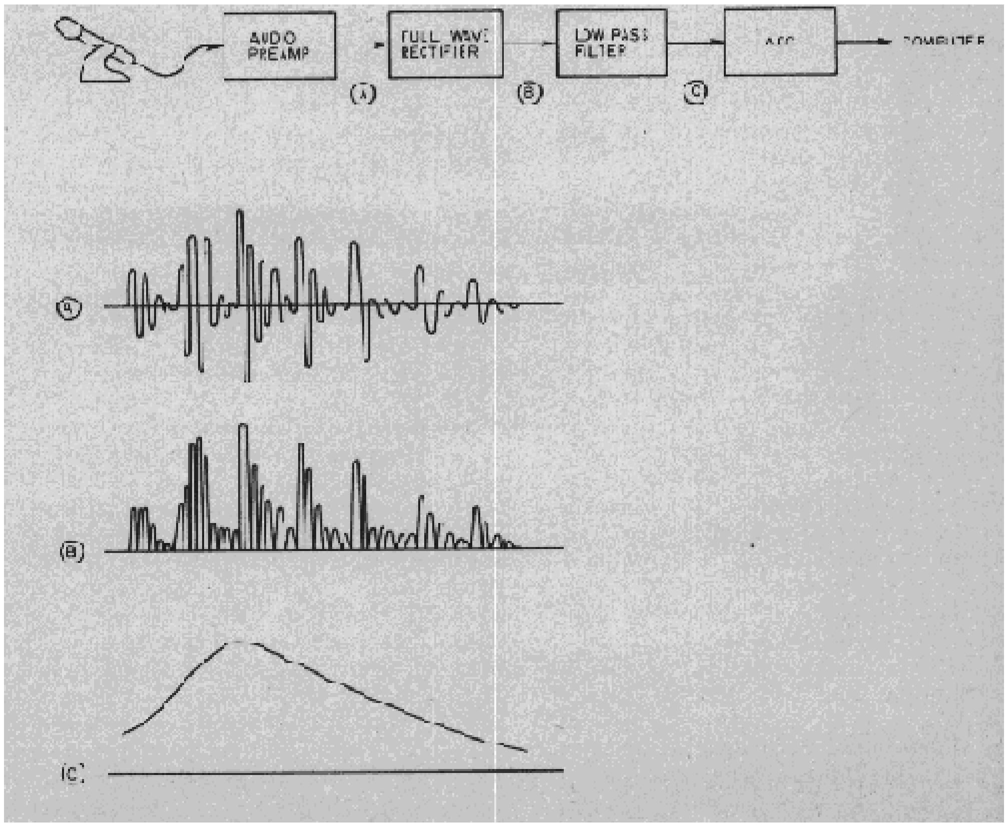


Figure 3: Rectifying the microphone signal at A we obtain signal B which contains a slowly varying DC component proportional to the volume of the signal and various high frequency components due to the formants. The low pass filter separates these and the analog to digital converter (ADC) sees only the volume signal.

- Feature Extraction
  - The process of extracting a set of slowly varying parameters that represent a word is called feature extraction. It has two objectives: First, as we have seen, it tries to reduce the amount of data necessary to process for recognition. This is a very important consideration for any practical implementation. Second (and extremely important), the features extracted must contain the relevant information in the signal. The speech signal conveys a lot of information about the sex, age, regional origin and emotional state of the speaker. Also it contains a lot of technical information incidental to speech production such as phasing and power spectra of the glottal pulses. All this information is related but not relevant to the meaning of the word pronounced. Ideally the features will contain as little as possible of this extraneous information.
  - A good feature extraction scheme will reduce the data rate in the recognizer by throwing away all the unnecessary information and retaining the information useful to the recognizer. The information pertaining to the meaning conveyed by speech is estimated to be from 15 to 30 bits per second, or about three orders of magnitude less than the data rate of the unprocessed speech signal. Practical feature extractors are not even close to the theoretical data rate and, most important, they tend to throw away significant amounts of the useful information. An example of a very inept feature extractor is, of course, the envelope detector mentioned previously. But it does have the advantage of extreme simplicity.
  - In order to design more sophisticated feature extractors some knowledge about speech production is necessary. Knowledge about speech perception would also be very useful but at this point we have no concrete information as to how people are perceiving speech. In contrast, the mechanism that produces speech is well understood and rather simple. It has been previously explained in the August1976 BYTE (page 16) in connection with speech synthesis. To summarize, speech is produced by exciting the vocal tract with either glottal pulses or noise while its shape is varied by controlling the position of the tongue and the jaws. Noise excitation produces unvoiced sounds. Glottal excitation produces voiced sounds, for example vowels. The movement of the articulators (tongue and jaws) changes the resonant frequencies of the vocaL tract, called formants. It is well known that the first three formants, designated Fl, F2, F3, carry most if not all meaning in speech together with timing considerations and type of voicing.

The glottal frequency, although very important in speech synthesis because it carries information about the particular speaker, carries no information about meaning. This rather surprising fact is easy to ascertain in two ways: First, synthetic speech is equally well understood whether the pitch (another name for glottal frequency) varies according to rules or remains constant. Second, whispered speech, which does not contain any glottal excitation, is well understood.

- It seems then that a good set of features for speech recognition would be the formant values. Information as to whether speech is voiced or unvoiced would be useful but not necessary. Unfortunately at this point it is not possible to experimentally either assert or refute this statement, simply because it is not possible to reliably and accurately extract the formants all the time. In addition, the best formant extraction methods known are at this point out of the reach of any imaginable personal computing machine. They involve digital signal processing methods which for real time processing strain the computational capabilities of even the big number crunching computers. There is a lesson to be learned from the above analysis: Speech is intimately related to the resonances of a time varying resonator (the voca tract) and perhaps it is more reasonable to expect to find better features in a frequency domain representation of speech rather than in a time domain one.

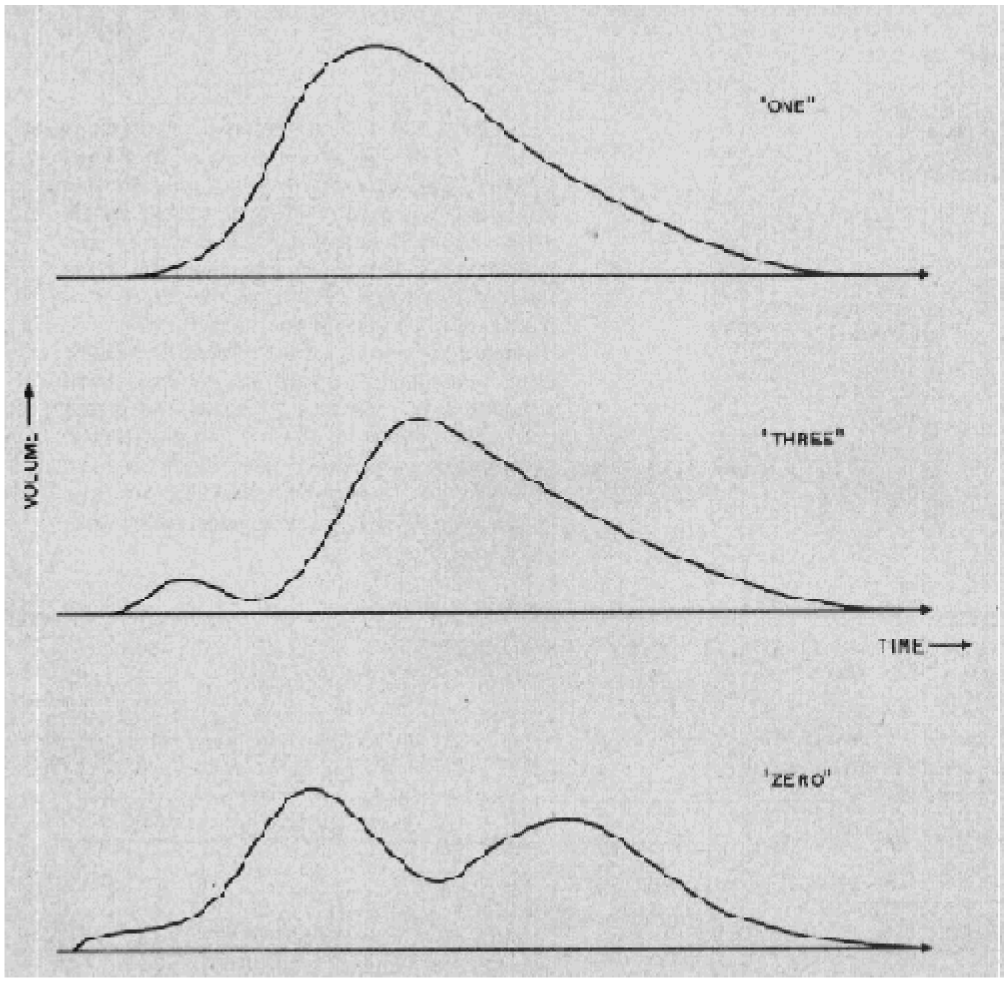


Figure 4: Amplitude envelopes of the words "one)" "three," "zero." Note that " has one hump while "three" has two. The smaller one corresponds to "th" and the larger one to "ee". In "zero", "z" is the low amplitude area in the beginning. The dip corresponds to "r"and the humps to "e" and "o." Vowels always have high amplitude because they are produced with the mouth open and with strong excitation.

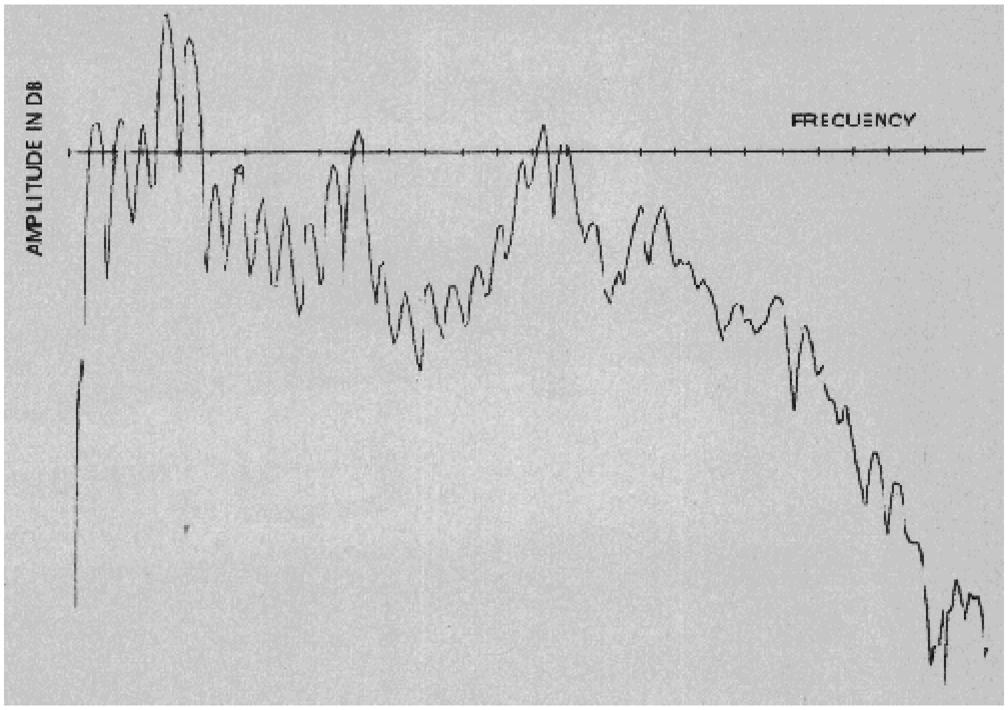


Figure 5: The frequency (logarithmic magnitude) spectrum of a 25 ms segment of speech. it covers the frequencies fmm 0 to 3.2 kHz. The malor peaks correspond to the formant frequencies while the smaller, regulady spaced peaks are harmonics of the glottal frequency. In this case It Is obvious that the sound is voiced both from the harmonics of glottal frequency which are highly visible and from the fact that the lower frequencies have more energy than the higher ones.

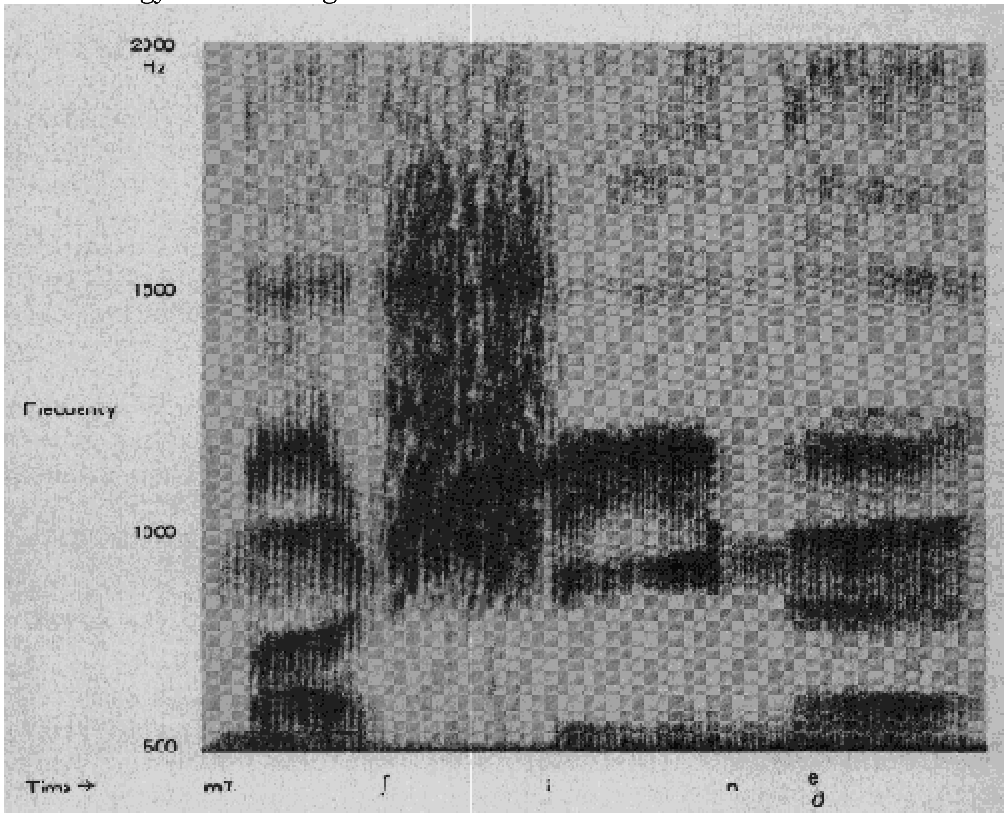


Figure 6: A typical Sonagram of an utterance. In a Sonagram the dark areas represent high intensity. This example represents the word "machine." The vertical (y) axis is frequency and the horizontal axis is time. The formants are seen as the dark bands that change with time. The large dark area about 1/5 of the way Into the pattern corresponds to the sound of the "ch" in machine.

- Frequency Domain Analysis
    - A usual frequency domain representation of a signal is its spectrum (amplitude versus frequency). It can be obtained from the time domain representation by applying what is called

the Fourier Transform to it. Figure 5 shows the Fourier Transform of a speech signal 25 ms long. Note that in the time domain signal the horizontal axis of a plot is time whereas in the frequency domain representation the horizontal axis is frequency. A drawback of the Fourier Transform is that it is a digital signal processing method and as such it operates on the sampled waveform directly, requiring very fast computers for real time operation. If we are willing to settle for less accurate results, there exist analog methods which can give us spectral representations in real time using relatively simple hardware. An interesting spectral representation, based on analog methods, is the sound spectrogram or "Sonagram" obtained using a device called the "Sonagraph." This device has been in existence since the late 1940's and has been widely used in speech research. It gives a three-dimensional display of frequency in the y axis (vertical), time in the x axis (horizontal), and intensity on the z axis. The z axis is actually represented by shades of black since we are using two-dimensional paper. Figure 6 shows a Sonagram. The dark areas correspond to areas of high intensity and the evolution of formants can be observed as dark bands changing with time. The Sonagraph is a useful tool in speech research and it is mentioned here because it is widely referred to in the literature. It is of no value as a feature extractor to a practical speech recognizer because it is not a real time machine and because it costs thousands of dollars.

- Using Filter Banks for Feature Extraction
  - A less accurate yet simple and inexpensive method for obtaining a spectral representation is a filter bank. It is widely used in existing speech recognition machines, including the most successful commercial speech recognition system. As shown in figure 7, a filter bank consists of a number of bandpass filters, covering adjacent frequency bands. The output of each filter is full wave rectified, then smoothed and sampled by an analog to digital converter for computer input. The output of the converter for any given filter is related to the energy present in the frequencies of the filter's passband.
  - A filter bank is a useful tool for feature extraction. It should be obvious that the speech spectrum can change only as fast as we can move our articulators. The constraints here are about the same as in the case of volume changes mentioned before. That is, a bandwidth of about 50 Hz is sufficient to accurately represent spectral changes. This gives us a sampling frequency of 100 Hz. Assuming eight filters in our filter bank, we get 800 digital words per second of speech. If the converter is accurate to eight bits, the data rate is 6400 bits per second, quite reasonable for the degree of accuracy retained. What makes a filter bank even more exciting for feature extraction is that a number of parameters are available that can be manipulated to get various tradeoffs of data rate versus accuracy. Thus both simple and sophisticated systems can be built using the same type of hardware. The parameters available for experirnentation are: –Number of bandpass filters. –Bandpass filter bandwidth and center frequency. –Bandpass filter skirt (selectivity) characteristics. –Amplitude compression. –Smoothing cutoff frequency (sampling rate).
  - It should be obvious that the more filters we have, the more accurate the spectral representation. A filter acts as an averaging device over the frequency in its bandpass and thus it destroys local information. By increasing the number of filters arid restricting their passbands this averaging is done over a smaller frequency range.
  - The bandwidth of the bandpass filters can be the same for all filters but usually it varies depending on the filter's center frequency. It has been shown that the ability of the ear to discriminate between closely spaced frequencies decreases when these closely spaced frequencies are high. Based on this observation, the bandwidth of the filters in a filter bank is rarely, if ever, the same. Instead it is set so that the higher the center frequency of the bandpass filter, the wider its bandwidth. The setting of a bandwidth for a given center frequency is done usually on the basis of experience in an effort to have enough filters covering the areas of greatest importance. Once a setting is decided upon, based on some criteria, experimentation may be required to optimize that particular filter bank. Another approach is to decide on some rule for setting the bandwidths. For example, a successful recognition system has been built using filter banks of third octave bandpass filters covering the frequencies of 150 Hz to 10 kHz. Such filters are commercially available for audio work.
  - The skirt or cutoff characteristics of the filters in a filter bank is determined by the following factors: –A steep skirt will give better frequency separation than a gently sloping one. It will also make for abrupt transitions from one filter to the other when a formant crosses the boundary. This is not necessarily bad but it might affect some systems. Both skirt types are used in practice. –It is desirable that the filter exhibit linear phase shift in its passband. For a given filter order, that means that the skirts will be less steep than the case without phase

restrictions. In general, this is not a serious consideration for a practical system. –The cutoff characteristics depend on the order of the filter (the number of mathematical poles it has in its design equations). Higher order filters give steeper characteristics but require more poles and thus cost more. –It is desirable that filters overlap at the three db points of their slopes.

- The amplitude (volume) of the speech signal is dependent on two factors that cannot be controlled easily. These are the distance and orientation of the speaker with respect to the microphone and the loudness of the voice at any given time. Volume also varies constantly within a given word as we have already seen. This can result in a 50 db range for the amplitude of the speech signal. The dynamic range of an 8 bit converter which is most likely to be used in the environment of computer experimenters is 20 log (256) = 48.16 db. This seems sufficient but it will give severe quantization errors for low amplitude signals. The lower the amplitude of the signal, the fewer bits will be used to encode it, thus throwing a lot of the information out. A 12 bit analog to digital converter would help but it costs too much and it wastes memory space in an 8 bit machine because it takes a word and a half to store its output.
- A different approach would be to use some form of amplitude compression before the analog to digital conversion. An audio automatic gain control circuit can be inserted between the audio source and the filter bank. It will limit the dynamic range of the signal by the amount of compression it offers, usually in the range of 20 to 40 db. Such an automatic gain control circuit (AGC) should have very fast attack. Its release time should be about two to three times the lowest glottal period expected. It should not be less than that because then it will tend to distort the signal by compressing between glottal pulses. Amplitude compression can also be applied after the rectifier smoothing filters by taking the logarithm of the signal at that point. The logarithm operation is suggested by the fact that our ears (and all of our body sensors) respond to stimuli intensity in a logarithmic fashion. This fact is utilized in every audio amplifier in the design of the volume control potentiometer which is made logarithmic so sound will seem to increase linearly with shaft rotation. The disadvantage of taking the logarithm of the smoothed signal is of course the number of required log circuits which is equal to the number of the filters in the filter bank. The logarithm can also be taken by having an analog to digital converter that converts logarithmically. There is such a product available from Precision Monolithics Inc. It is the DAC-76 companding digital to analog converter that can be used to build an 8 bit logarithmic analog to digital converter.
- As a final note to amplitude processing we should mention a method used in a number of small systems. In its simplest form it consists of using in the place of the multiplexer and the analog to digital converter a number of comparators which change state when the input voltage exceeds a preset threshold as shown in figure 8. The output of the comparators forms an N bit binary word where N is the number of the filters in the filter bank. This gives a significant reduction in data rate and works for small vocabularies of the order of 8 to 12 words. It requires a good automatic gain control before the filter bank, and its major weakness is that the comparator thresholds have to be manually adjusted for each individual speaker. A variation of this method is used in a commercial recognizer made by Scope Electronics (US Patent #3,812,291). In that system the comparators are connected between two consecutive filter outputs to detect the slope or frequency profile. Figure 9 shows this differential comparator connection. Compared to the fixed threshold method it has the advantage that no adjustment for the individual talker is needed and the automatic gain control before the filter bank is not necessary. Its disadvantage is that it throws away more information than the fixed threshold method.
- The final parameter that can be adjusted in the filter bank feature extractor is the cutoff frequency of the smoothing filters. Reducing it reduces the data rate. Depending on the amount of reduction significant information may or may not be lost in this step. Sampling the output of the filters every 20 to 30 ms is tolerable in small vocabulary systems. The main consideration here is to find a combination of number of filters, analog to digital conversion quantization and sampling rate that will maximize the information retained and minimize the data rate of the pattern.
- In summary, a filter bank offers a good yet inexpensive real time feature extractor which can be utilized in a speech recognition system. Properly used, it has performance similar to systems based on linear prediction residuals, a state of the art digital signal processing technique.

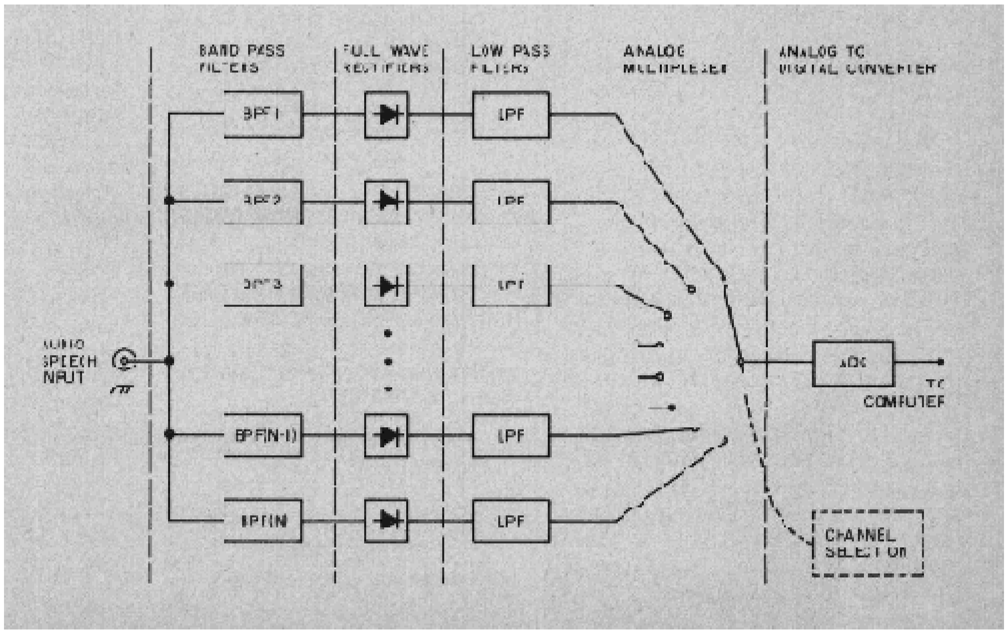


Figure 7.. A filter bank feature extractor consists of a number of bandpass filters (BPF1 to BPF (N)) covering the range from about 100 to 10 kHz. The output of the filters is rectified and low pass filtered (LPF). Then it is multiplexed and digitized by the analog to digital converter for input into the computer.

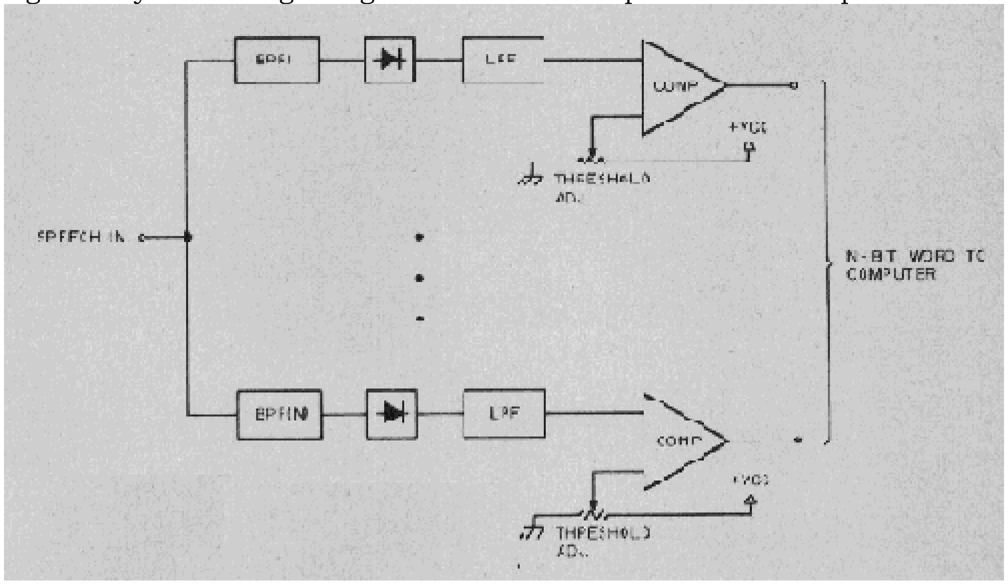


Figure 8: Using a comparator after the low pass filter in the filter bank simpilfies hardware and reduces data rate at the cost of discarding useful information. The thresholds should be adlusted for the voice of each individual speaker. In addition, a good automatic gain control circuit should precede the filter bank to normalize the time domain signal.

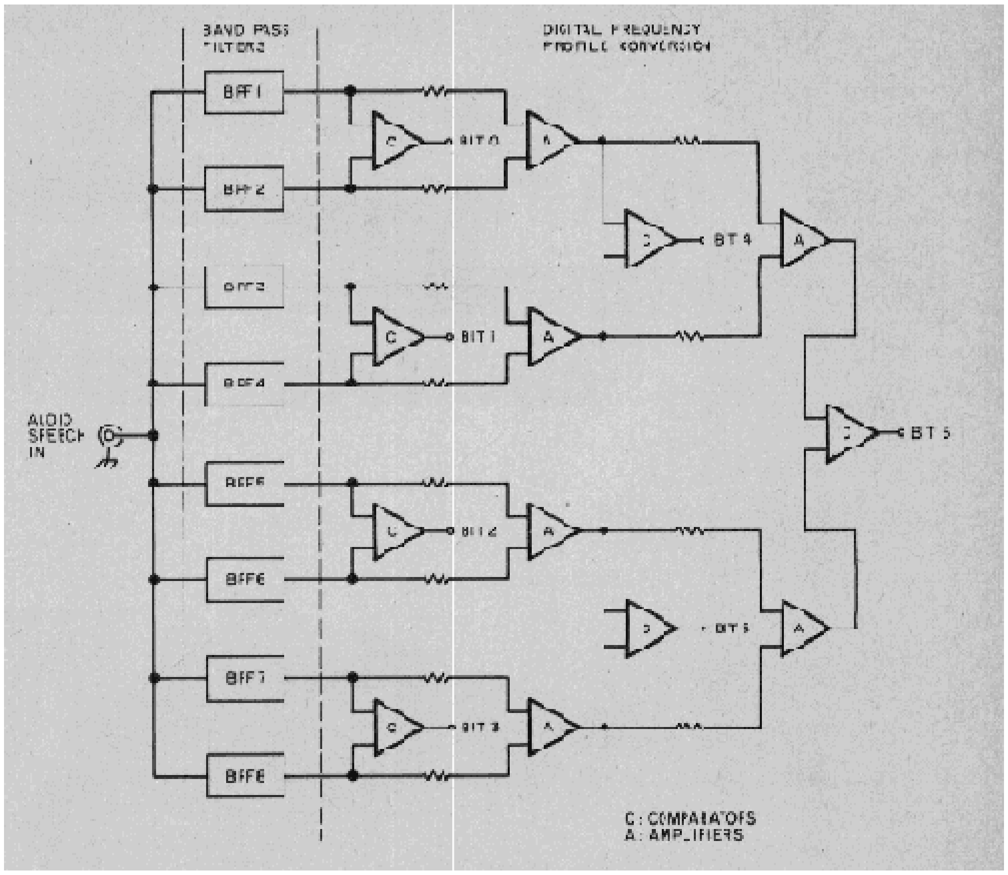


Figure 9: The differential comparator feature extractor detects which one of two adjacent bandpass filters has the highest energy. The output of each pair of filters is summed together and compared with the sum of the next pair of filters, yielding another, coarser comparison from the frequency viewpoint. In this example, the output of eight filters (we assume that the output of the blocks labeled BPF is the rectified and smoothed output of the bandpass filters) is encoded into a 7 bit digital word. The computer performs pattern recognition on time varying sequences of these 7 bit words.

- Zero Crossing Detectors
  - Another feature extraction technique that was popular for its simplicity in the early days of speech recognition is formant tracking using zero crossings. It is interesting to note that probably the first widely known speech recognizer capable of recognizing the digits was built in 1952 at Bell Labs using formant tracking based on zero crossing measurements. The contraption was grotesque by today's standards, using vacuum tubes as active elements and capacitors for memory. It was claimed that it achieved 97 percent recognition accuracy. Its formant tracker was based on two well known principles: First F1 and F2 formants span two different frequency ranges, that is F1 moves roughly between 200 and 1000 Hz and F2 moves between 800 and 3500 Hz. There are cases for which this is not true but for about 98 percent of the time this holds for any speaker, man or woman. Second, F1 and F2 of a given speaker for a given vowel remain about the same day after day of testing. (This is not true for the same vowel pronounced by different speakers and thus the machine has to be trained to a given speaker before it can recognize his or her speech.) This means that we could isolate the first two formants and they would be useful as features. Two bandpass filters centered around the respective formant regions would separate the formants except when they are in the overlapping area of 800 to 1000Hz.
  - It turns out that when we highly amplify, then clip a signal to obtain the instances when the signal crosses zero, the problem is taken care of automatically. If the bandpass filters slope in the overlap region as shown in figure 10, F1 will always be stronger than F2 at the output of the F1 filter and vice versa. The clipper has a property called "capture" which means that any signal that is a few db or more stronger than another will swamp out the weaker signal and the output of the clipper will contain only the strong signal. This phenomenon occurs also in FM radio reception with a limiting (clipping) intermediate frequency (IF) stage. It is one of the reasons FM was chosen for quality broadcasting, ie: it is not susceptible to interference from other weaker stations or interference from noise. Filtering and capture ensure that the output of the clipper contains only one frequency, the formant we are tracking. Determining what that

frequency is requires nothing more than counting how many times the square wave output of the clipper crosses zero within a given interval, perhaps 20 ms. The data rate of the zero crossing formant extractor (see figure 11) is quite low, about 50 samples per second consisting of two 8 bit words per sample, a total of 800 bits per second.

- In practice it works well with vowels, especially if there is no significant F1 and F2 overlap. Formant overlap and noise do affect its performance and thus limit its applicability. It is perhaps the simplest viable feature extractor that can be used and it does not contain any critical analog circuitry. One word of caution about the implementation, learned the hard way: an 8080 processor running with interrupts generated by the rising and falling edges of the clipped F1 and F2 signals is not fast enough to keep up with the pace in some cases and data is lost. Rather than trying an all software approach, I recommend that two hardware counters be used, read and reset by the computer every 20 ms. These counters are shown in figure 11.

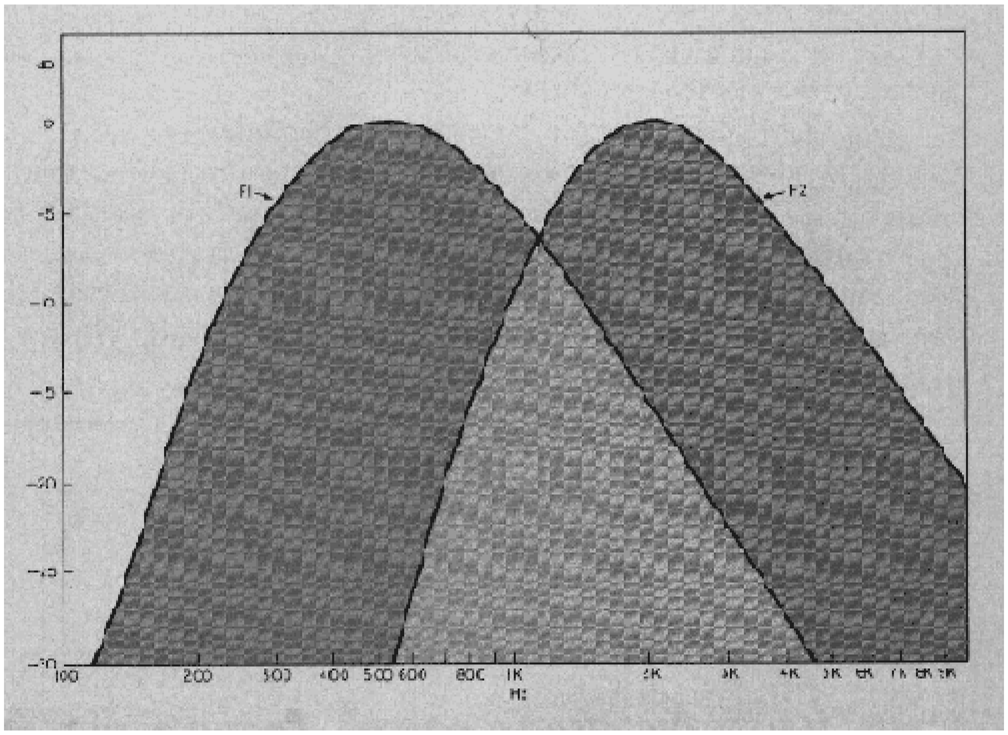


Figure 10: The frequency response of the two filters is designed to separate F1 and F2, when used in a zero crossing based formant detector. The filter charactedstics overlap In the region of formant overlap but their slopes are designed to separate the two formants.

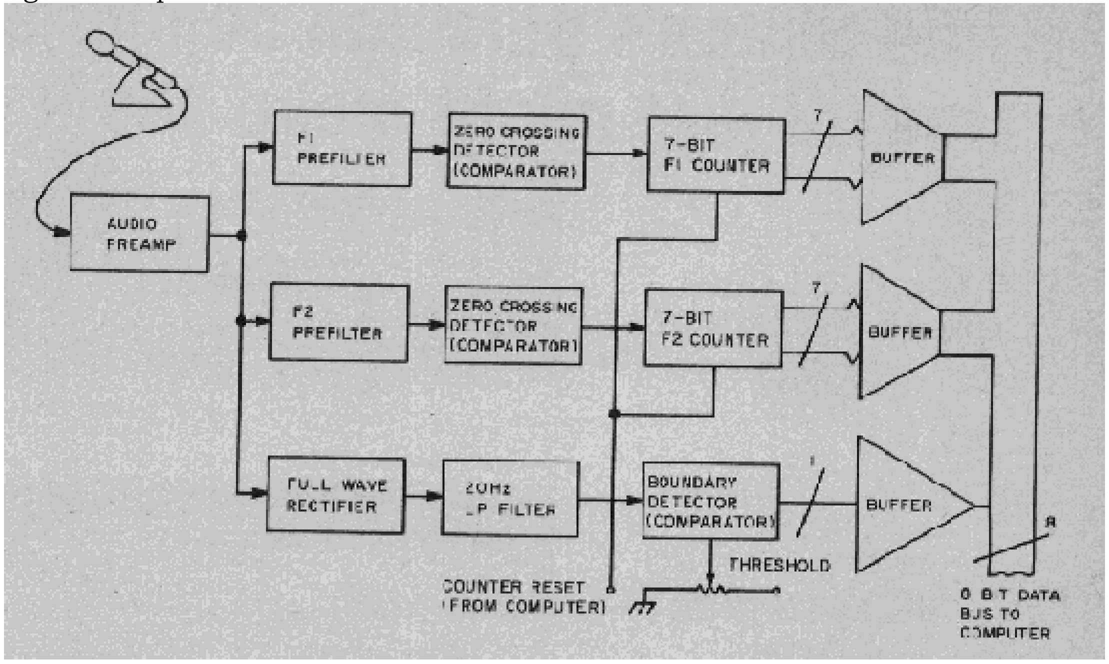

AUDIO PREAMP
F1 PREFILTER
F2 PREFILTER
FULL WAVE RECTIFIER
ZERO CROSSING DETECTOR (COMPARATOR)
ZERO CROSSING DETECTOR (COMPARATOR)
20Hz LP FILTER
7-BIT F1 COUNTER
7-BIT F2 COUNTER
BOUNDARY DETECTOR (COMPARATOR)
7
7
1
BUFFER
BUFFER
BUFFER
8
THRESHOLD
COUNTER RESET (FROM COMPUTER)
8 BIT DATA BUS TO COMPUTER

Figure 11: A zero crossing formant extractor. The zero crossing method of formant extraction uses two special bandpass filters to separate the formants. The output of the filters is passed through a zero crossing detector (a comparator whose threshold is set to zero) that puts out a logical "1", or a "0" depending on whether its input is positive or negative. The output is fed into a counter for each formant and the number of "0" to "1" transitions is counted for 20 ms. Then the counters are read by the computer and reset to start the next 20 ms counting perlod. An envelope detector (rectifier followed by a low pass filter) feeds a comparator whose threshold Is set to detect word beginnings and endings.

- Pattern Recognition
    - A set of features representing a word is a pattern corresponding to this word. Pattern recognition is the process by which, given an unknown pattern, we decide which word this pattern represents. An alternate term is pattern classification. A basic process of pattern recognition is template matching. The unknown pattern is compared to a number of reference patterns stored in mem-ory and the differences between the unknown and the references are noted. Obviously the smallest difference indicates a best match and the unknown word is the word that corresponds to the reference pattern that gave the smallest difference.
    - There exist many ways to compare two patterns. Since features are represented by numbers, the most straightforward way is to compute the numerical difference between corresponding samples and then sum all those differences. That sum would be a measure of similarity between the patterns. There is a small precaution to be taken in practice when computing the sum of the differences. All differences should be positive. Otherwise when we sum them, negative differences will cancel positive ones and the effect would be a smaller total difference than is actually the case. To make sure that all differences are positive we can take the absolute value of the differences before we sum them. This is done simply by making any negative distances positive and leaving the positive ones alone. A fancy name for the absolute value of feature differences is "Chebyshev distance." "Euclidean distance" is another fancy name used to denote the square root of the sum of the squares of individual differences. The square of a number is always a positive number and this overcomes the problem of adding differences with different signs. Squaring takes much more time than finding the absolute value and it turns out it is no better than the absolute value as an indicator of degree of match.
    - Another pattern recognition technique not based on template matching is "linear discrimination." It is not very useful in practice but it excites people's mathematical instincts enough to write papers about it so we will mention it for the sake of completeness. If we view the features as dimensions in a multidimensional space (also called hyperspace), a pattern becomes a point in that space. If the points for all pronunciations of a word cluster together in a region of this hyperspace and regions corresponding to different words do not overlap, we can define hyperplanes that separate these regions. To classify a pattern then we can check to see if it is in a region enclosed by a given set of hyperplanes and keep checking until we find which region it corresponds to. If the regions cannot be separated by hyperplanes the feature space is not linearly separable which means that we can try to enclose the regions in sections of hyperplanes. It often turns out to be a matter of luck whether the feature space turns out to be even nonlinearly separable. The net result is that linear discrimination is often difficult if not impossible to use in practice.
    - Pattern recognition can also be achieved using a set of tests on a pattern to decide which word it represents. A simple example of that method is the rules used in the previously mentioned case of the volume signal to determine which one of the three words acceptable to the system has been pronounced. In a more complex system it is not very easy or accurate to do all the recognition by rule. Too many rules might be required and mistakes in the application of key rules might cause misrecognition or rejection. This last problem does not occur in template matching because each feature has equal say in the decision process; therefore a few bad features do not affect the result.
    - Rules can be used successfully together with template matching to speed up the matching procedure or to act as an additional accuracy check. For example, in a recognition system that accepts the digits 0 to 9, we might recognize a word as "six" by template matching but we may not be very sure that it is actually a "six": while the distance from the reference corresponding to "six" was the smallest it might also be too close to the threshold for a conclusive determination. Applying the rule that both the beginning and end of "six" have to be unvoiced we can either confirm our recognition or increase the suspicion that something is wrong, depending on the outcome of the test. In another situation we might check before template matching to see if the volume of the word has two approximately equal humps. If this test is passed we can then do template matching on the words "seven" and "zero" which are most

    likely to exhibit a 2 humped volume pattern. If template matching on one of these words gives a very good match well below threshold, we could accept it without further testing of the other patterns, thus saving testing time.
  - An interesting hybrid (template matching combined with discrimination by rule) recognition scheme is phonemic pattern recognition. It is based on the observation of phoneticians that human words are made up of a limited number of building blocks called phonemes. In English there are about 50 distinguishable phonemes. These phonemes were designated mainly by listening tests. Trying to automatically extract the phonemes is very difficult because our machines do not even approach the generalizing capabilities of the ear when it comes to speech. Proposals have been made to use from 200 to 700 basic units of speech for machine recognition in the hope that it will be easier to discriminate between those less general "phonemes." Reducing a word to a sequence of phonemes gives us a very low data rate pattern and simplifies pattern recognition, assuming of course that phoneme extraction is accurate. Phoneme extraction can be done using template matching, and recognition of a phonemic sequence can be done by rule.
  - Unfortunately in real life things are not that easy. The number of individual phonemes a machine can recognize depends on the type and number of features its preprocessor extracts. There exists no way to find out for sure what are the essential features that determine the phonemes for a given system except by tedious trial and error. In addition, coarticulation effects tend to change the pattern of phonemes depending on the phonemes that come before and after it. These effects can generate quite drastic changes in the pattern of a phoneme to the point that template matching is useless for recognition. In those cases some form of recognition by rule should be applied to separate the phonemes.
  - These problems are compounded by the fact that when we decide which phoneme represents a segment of speech we throw away all the features that make up the pattern for that speech segment As a result, if our decision was mistaken for some reason, we have lost that portion of the speech pattern and even worse, we have substituted a potentially confusing phoneme. This is always a problem when we compress data too much. A piece of bad information can have very unpleasant consequences because due to the high compression factor there is not very much information available on which to base alternate decisions. A way around this problem would be to store the raw data until the final decision is made. If we are not confident in our decision for some reason we could go back and check the point of dispute. This is an example of the hypothesize and test method where on the basis of incomplete information a hypothesis is made as to what the word is and then the data is checked closely to either verify or reject the hypothesis. A disadvantage of this method is that it is time consuming computationally. The alternative to storing thousands of reference patterns in memory for template matching on a large vocabulary is cumbersome, to say the least. Finding a best match with templates when the vocabulary is big will take too much time even on very fast computers. Training the machine by storing reference patterns is still another disadvantage of template matching when the vocabulary is large because it takes a lot of user time to pronounce the training words.
  - Recognition of connected speech cannot be done using a simple template matching recognizer for two reasons that also complicate recognition of phonemes: First, it is very difficult to find word beginnings and endings in connected speech. Second, adjacent words affect pronunciation because of coarticulation. Practical recognizers at this time are limited to isolated word recognition and they require that each word be spoken individually. This is not as restrictive as it might seem because, with practice, words can be pronounced in sequence quite rapidly, up to 70 words per minute. Other usual restrictions on current pattern recognizers are the need for low ambient noise and a cooperative speaker. Cooperative here means a speaker who is willing to pronounce the words clearly, evenly, and with a conscious attempt at uniformity.

- An Example of a Speech Recognizer
  - The basic components of any speech recognizer are the feature extractor and the pattern recognizer. The feature extractor is also called the "front end" or "preprocessor" and it is usually made up of analog circuits. The pattern recognizer nowadays is always a digital computer. We have seen an overview of what these two basic functions are, how they relate to the speech signal and how they are usually implemented. It is time now to give a block diagram of a speech recognition system using a particular method and go into greater detail about the functions and implementation of the various elements. I have chosen for this example a filter bank analyzer because I believe this particular approach will give the best results with the 8 bit microprocessors popular with computer amateurs. It is also a well known and proven

approach, thus more likely to give consistent results than any other method. It is not the simplest possible approach and I expect to see a number of simple recognizers for personal computing machines based on zero crossing detection methods or combinations of zero crossing analysis and some sort of filtering. While it is possible that unusual feature extractors could give surprising results, the chances of something like that happening are very small. Speech recognition has a history about 30 years long and during that time many schemes have been tried, none of which gave better results in practical real time situations than the filterbank feature extractor.

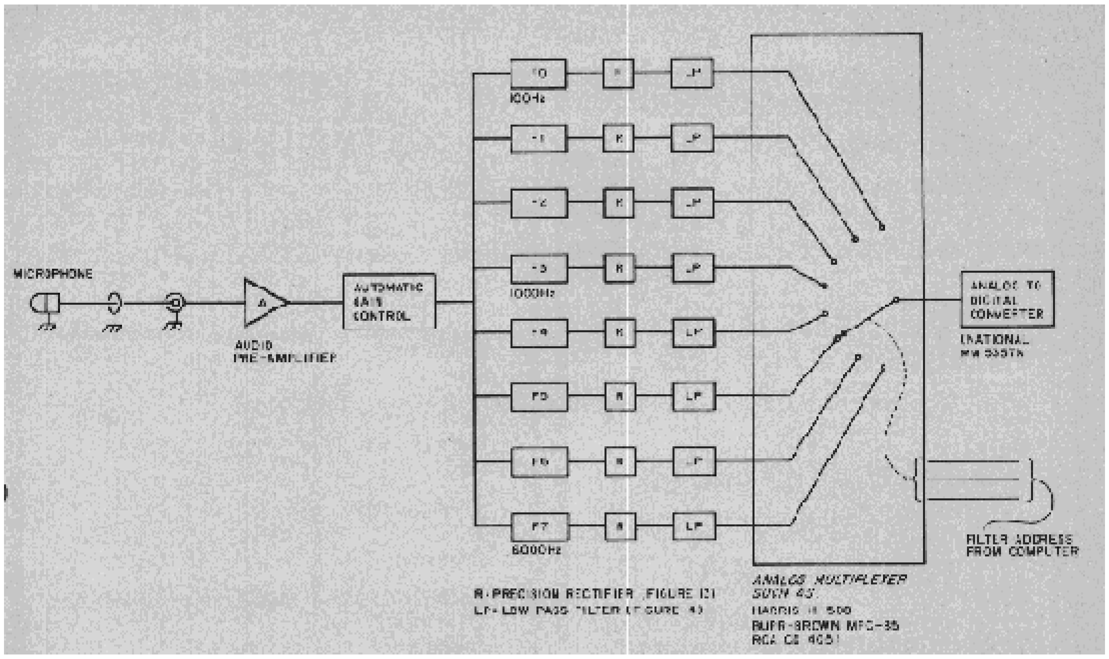


Figure 12 is a block diagram of a complete filter bank feature extractor. Let us pay close attention to the specifics of its elements.

- Microphone
    - If the system is to operate in a quiet room, the only consideration for the microphone would be a reasonably flat frequency response extending to about 8 kHz. In practice a quiet room is rarely the case. Radios, passing cars, the TV, the Teletype, the phone ringing, people talking are all usual sources of background noise which might trigger the recognition system or interfere with the words to be recognized. Once this background noise mixes with the desirable signal there is not very much we can do to separate them. The most benefit in a noisy situation can be derived by achieving high signal to noise ratio at the microphone. This can be accomplished either by using a close talking microphone like the ones used by aircraft pilots or telephone operators or by using a directional microphone. A directional microphone such as a cardiold will work best when the speaker is facing the noise source and there is no sound reflecting surface behind him. This last point is easy to overlook when using a directional mike so it is very important to remember that noise can get to the mike in its sensitive direction by reflection. A close talking mike depends basically on its distance from the speaker's mouth to separate between noise and speech and it is most helpful when the speaker moves unpredictably with respect to the noise source and when the environment is very noisy and reflective. The microphone should be positioned close to the mouth using a headphone strap or an eyeglass clip. This may or may not be a disadvantage depending on the application.

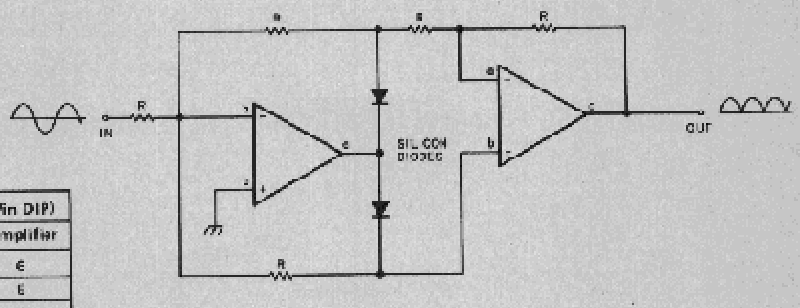
- Audio Preamplifier
    - The audio preamplifier is quite straightforward in design being just a low level audio amplifier with flat frequency response unless it is used to equalize the response of the mike. There should be provisions to ajust its gain so that it can be matched to different mikes or background noise conditions. This is an infrequent adjustment and it should be hidden from the casual user.
    - Changing the gain of the preamp affects the point at which the automatic gain control starts compressing and it should be done carefully.

- Automatic Gain Control
    - As mentioned previously, the AGC should have very fast attack because in many instances the volume of speech increases very rapidly. Its release time should be 30 to 60 ms, fast enough to follow instances of quick decaying speech volume but not so fast that it will compress between glottal periods. If the AGC has a compression range of about 40 db it functions as amplitude normalizer so we do not have to do any normalization of the filter outputs in the computer.

- Filter Bank
    - Eight filters were chosen for the filter bank simply because analog multiplexers are available with either 8 or 16 inputs and 16 filters are just too many. Another reason is that a workable system can be made with six filters as shown by George White (see references) and eight will certainly give adequate performance. There is a compromise made here between cost required storage and processing time on one hand and frequency resolution on the other. An interesting study will be to vary the number of filters in a given recognition system from let's say 1 to 16 and plot the recognition accuracy versus the number of filters. The center frequencies and the bandwidths given for the filters are merely an educated guess, based on knowledge of where the "action" is in the speech spectrum and on experimental results published so far. In a practical system some "tweaking" is highly recommended to achieve best results and to the true experimenter it should be irresistible anyway. The skirt characteristics of the filters should be from 24 to 48 db per octave. Steeper slopes require more poles in the filter which in turn require precision components so the desire for steeper slopes should be traded off with the realities of implementation. There are basically two approaches to implementing the filters: First hybrid active filters using the biquad configuration such as the National AF-100 or the Kinetics Technology FS-50 or the Burr-Brown UAF-31 can be used. They are quite easy to use but they have the disadvantage that they cost a lot. Second, one can build filters out of operational amplifiers. Since we are dealing with low audio frequencies, under 5 kHz, inexpensive 741 type op amps can be used as active elements. This will be less expensive than the previous approach but you will have to build the filters yourselt not necessarily the kind of tradeoff everyone likes.

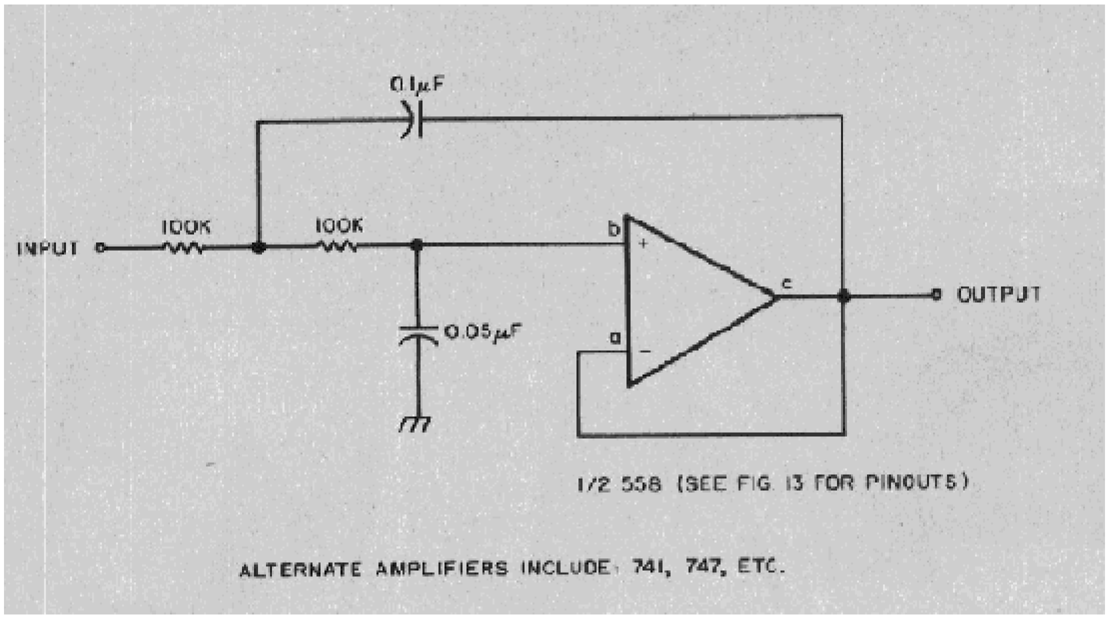
- Rectifiers and Low Pass Filters
    - The use of relatively low level signals in the system makes the use of a precision rectifier after the bandpass filters mandatory.
    - In precision rectifier circuits op amps are used to eliminate the forward voltage drop of the silicon diodes, that is 0.7 V, thus making the circuit behave like an ideal diode. Full wave rectification is desirable to give low ripple. A suitable precision rectifier circuit is shown in figure 13. A low pass filter is suggested to smooth the output of the rectifier. A 2 pole Butterworth filter with a cutoff frequency at about 20 Hz will work well and is shown in figure 14. Instead of the low pass filter we could use an integrator that is reset every time it is read by the analog to digital converter. The integrator output would be proportional to the average of the energy in the frequency band of the given filter during the period between reads. This of course is more complicated than using just a low pass filter and it is questionable whether it will give better results. It has been used in the past though and it is covered by a patent for use in vocoders.

Figure 13: A precision full wave rectifier (R) used to rectify the output of each bandpass filter in figure 12.



553 Dual Op Amp Wiring Table (8 Pin DIP)		
Pin	Amplifier A	Amplifier
a	2	6
b	3	5
c	1	7
Power: +12 V Pin 8 −12 V Pin 4		

- Multiplexer and Analog to Digital Converter
    - There are several monolithic 8 channel analog multiplexers such as the Harris HI 508 or the Burr-Brown MPC-85 but the RCA CD4051 is the best choice, costing only $1 in single quantities compared to around $15 for the others. There is no need for a sample and hold because the preceding low pass filter assures that the signal does not change quickly. An excellent choice for the analog to digital converter is the National MM5357N which at $11.95 is three times cheaper than the competition. Making your own analog to digital converter from scratch using an 8 bit digital to analog converter could cost halt as much assuming you do not count your time.
    - You will notice that neither volume nor information as to whether the signal is voiced or unvoiced is explicitly extracted by the feature extractor of figure 12. These features are not lost. They are buried in the output of the filters. The sum of the filter outputs, for instance, at any sample point in time is the volume for that sample. A voiced sound has most of its energy under 1000 Hz and the opposite is true for unvoiced sounds. If we divide the sum of the outputs of the four higher frequency filters by the sum of the outputs of the four lower frequency filters we will get the ratio of the energy in the 1000 to 5500 Hz band to the energy in the 100 to 1000 Hz band. If this ratio is less than 1, the sound is voiced; otherwise it is unvoiced.
    - There are many ways of utilizing the features extracted by the filter bank feature extractor in a pattern recognizer. Actually this versatility is one of the desirable characteristics of the filter bank. Here I will describe only one possible approach to illustrate the various principles involved. Since all pattern recognition is done in software, experimentation in this area is easy and fun since you don't have to buy or build additional hardware to improve your recognizer. A pattern recognizer has two basic modes of operation: training and recognition. During training it performs the following functions: –Word boundary detection (beginning and end). –Pattern input and normalization. –Reference pattern storage.
    - In the recognition mode, reference pattern storage is replaced by pattern matching (classification). Training is required to generate reference patterns for the words that form the vocabulary of the recognizer. When a different speaker wants to use the system, retraining is necessary to generate reference patterns for the new speaker's pronounciation of the words. Implicit in our discussion is the assumption that we limit our recognizer to isolated word recognition. Moreover, these words are to be drawn from a limited vocabulary of about 10 to 30 words in size.



0.1μF
INPUT
100K
100K
0.05μF
b
a
c
+
−
OUTPUT
1/2 558 (SEE FIG 13 FOR PINOUTS)
ALTERNATE AMPLIFIERS INCLUDE 741, 747, ETC.

Figure 14: A 2 pole 20 Hz lowpass filter (LPF) suitable for smoothing rectified speech audio in figure 12

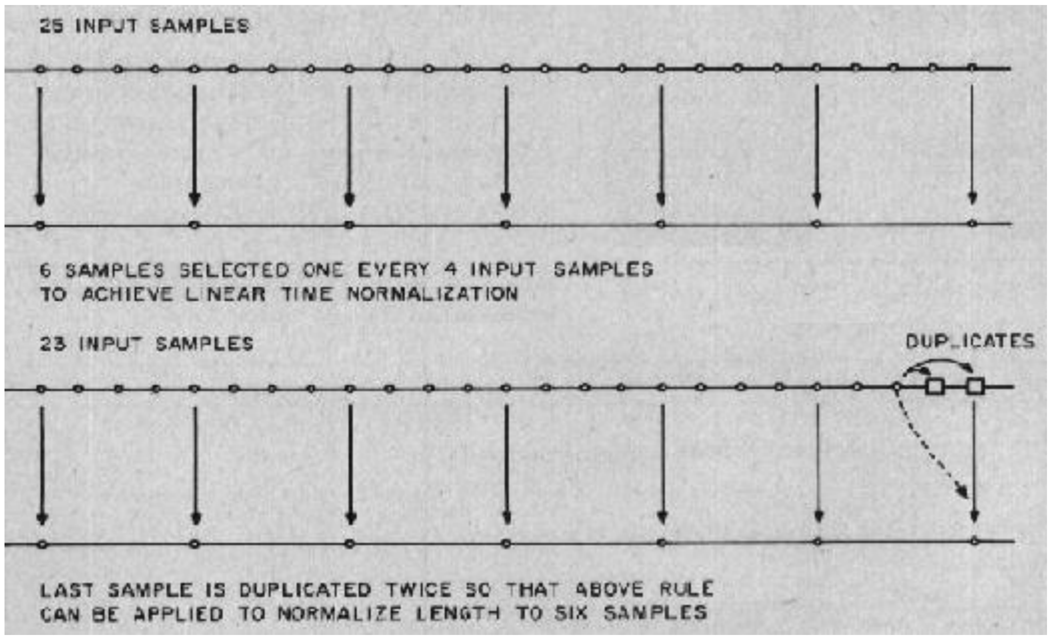
- Word Boundaries and Normalization
    - In operation, the word boundary detection module reads the output of the filter bank every 10 ms and stores it in a temporary storage area. To do this an input routine sequences the multiplexer, starts the analog to digital converter and reads the data into memory. A simple and usable means of detecting word beginnings and endings is to sum the output of the filters and compare it to a given threshold. If the energy exceeds the threshold, a word has begun.

There is a precaution to be taken however when using volume for word boundary detection. Certain consonants called plosives or stops (p,t,k,b,d,g) are generated by building up pressure and then suddenly releasing it. While the pressure is being built up no sound is generated and this silence period or "stop gap" can last up to 150 ms. If this occurs in the middle of a word as in "ago" for example, the simple boundary detector will be fooled to think that two separate words occurred. This can be corrected easily by considering any "words" less than 200 ms apart to be parts of one word. Another problem arises when a noise pulse occurs without any word being said as in the case where something is dropped on the table. To prevent the word boundary detector from accepting such noise as a word, we define any sound with duration less than 150 ms as noise because words are not that short. These modifications will make the boundary detector quite successful for operation in a quiet room with a careful speaker. For any other situation considerable additional sophistication might be necessary. In a noisy environment the threshold for word detection has to be increased so that the detector will not trigger on background noise. But increasing the threshold will tend to chop off the low amplitude beginnings and endings of some words and this can affect recognition significantly. If the speaker is careless or he is physically strained when he speaks because, for example, he is lifting something heavy, breathing noises will tend to be considered as part of the word being said, further degrading recognition. In these situations a sophisticated boundary detector is very helpful in maintaining system performance. Once the beginning of a word is detected, the pattern input starts. Every 10 ms the output of the filters is sampled and stored inconsecutive memory locations in the buffer area. The duration of the longest word can be expected to be less than 1.2 seconds, giving us a maximum of 120 samples to store in the buffer. Thus a 1 K buffer will be sufficient.

- Upon detection of the end of the word, some form of time normalization is required before the next step. This need arises from the fact that people do not time their pronunciations the same every time. They may say "zero" one time and "zeroo" the next. Before we can do template matching, if we are to be successful, these timing variations should be removed. Linear time normalization is a technique of doing just that. To demonstrate the principle assume that the normalized word length is 16 8 byte samples and that our buffer contains 65 8 byte samples. To do time normalization, all that is necessary is to retain every fourth sample and throw away the rest, as shown in figure 15. If the number of samples in the buffer is not a multiple of 16 plus 1 (16*N)+1, a number of rules can be implemented to resolve the problem. For example, if there are up to four less samples than (16*N)+1, then samples in areas of high amplitude can be duplicated to reach a number of samples that is exactly (16*N)+1. The assumption here is that in areas of high amplitude interesting things exist. It is not necessarily a correct assumption but it can give an idea of what can be done to adjust the length of the word without resorting to linear interpolation. Larger discrepancies can be treated similarly but duplication should occur throughout the word to avoid stretching only a part of it. Omission of samples can be used as well, to adjust the length. Insertions and deletions can be made in a number of locations in the word based on criteria other than local amplitude. For example, deletions in the beginning or ending of words, or slow spectral changes are other possibilities. The best guide as to what works for any given system is some experimentation with various techniques.
- In usual practice, all words in a vocabulary for a given system are normalized to the same length, regardless of their actual length. It might be useful, though, if a large proportion of the words of the vocabulary are distinctly different in length from the rest, for example "yes," "on," "off" compared to "television," "instrument," "telephone," to have two normalization lengths. An added benefit to the obvious savings in pattern storage memory would be a reduction in classification time because there is no point in trying to find a match between a long and a short word.
- A more advanced form of time normalization is based on dynamic programming techniques. It is basically a systematic approach to stretching and compressing the unknown word in time until it best fits the reference pattern, it gives excellent results but it has two drawbacks: It is time consuming and it requires storing all the input samples for a given word as reference patterns. Despite its shortcomings, its superior performance makes it a very interesting technique. Its detailed analysis is not within the scope of this article.
- The final phase of the processing of the unknown word depends on whether we are in the training mode or the recognition mode. In the training mode all that is necessary after time normalization is to store the resulting pattern in memory in an area designated for the word pronounced. To improve performance two or more reference patterns may be used for a given word to account for variations in pronunciation. For example the speaker may be required to pronounce each word in the vocabulary twice (although not one immediately after the other to

avoid similarity as much as possible) so that two references will be generated and stored for each word. Variations of this scheme include making the speaker repeat a word until it is sufficiently different from the first reference or changing the second reference adaptively as the system is used. In the recognition mode the normalized unknown word is compared to each reference pattern by computing the sum of the absolute values of the differences of the features. The smallest sum indicates the best match between the unknown and that reference. This alone is not sufficient for recognition because a word that does not belong to the vocabulary of the system will always give a best match because one of the reference patterns is bound to be more like it than the others. To avoid "recognizing" words that are not in the vocabulary, we impose the restriction that successful recognition requires that the smallest difference measure will be smaller than a given threshold. This threshold has to be found experimentally for any given vocabulary. If it is too high, it will allow for acceptance of words that are not part of the vocabulary. If it is too small, words that are in the vocabulary but not pronounced very close to the stored references will be rejected.

- Word beginnings and endings usually have low amplitude and it is possible that the word boundary detector could chop them off. If that happens, it will most likely happen only occasionally and as a result there will be instances where the reference and unknown patterns are shifted with respect to each other in time. For this reason, if computer time is available, the unknown may be shifted to the right one position and the difference calculated and then to the left and the difference computed again as shown symbolically in figure 16. The smallest of the three differences thus computed is to be used as the indicator of match quality.
- When a word is recognized it can be used directly to perform a function which could be, for example, the equivalent of pushing a key on a calculator. In some instances it is necessary or desirable that more than one word be used to perform a single function. For example the sequence of the words "light on" can perform the function of turning on the lights in a room. In these cases there should be a subroutine to check if the word sequence is correct. For example, the word "light" should be followed by the words "on" or "off." If the word "radio" follows, we know that either "radio" or "light" has been misrecognized. These syntactical constraints decrease the probability that an incorrectly recognized word will cause something undesirable to happen. This is due to the fact that for a sentence to be correct, all the words that form it have to be correct, at least syntactically. Syntactic constraints of course do not help when the words have the proper sequence but are still not correct. For example, if we are controlling a vehicle of sorts and we say "forward faster," if "faster" is recognized as "slower" the sentence will be syntactically correct but it will not do what we want.


25 INPUT SAMPLES
6 SAMPLES SELECTED ONE EVERY 4 INPUT SAMPLES
TO ACHIEVE LINEAR TIME NORMALIZATION
23 INPUT SAMPLES
DUPLICATES
LAST SAMPLE IS DUPLICATED TWICE SO THAT ABOVE RULE
CAN BE APPLIED TO NORMALIZE LENGTH TO SIX SAMPLES
Figure 15: Linear time normalization compresses the stream input samples to a fixed number of samples (in this case six) by selecting a sample at regular Intervals. The resulting fixed length for all words facilitates feature matching for pattern recognition. Various techniques (such as duplicating samples) are used to make the number of inputs fit the selection rule.

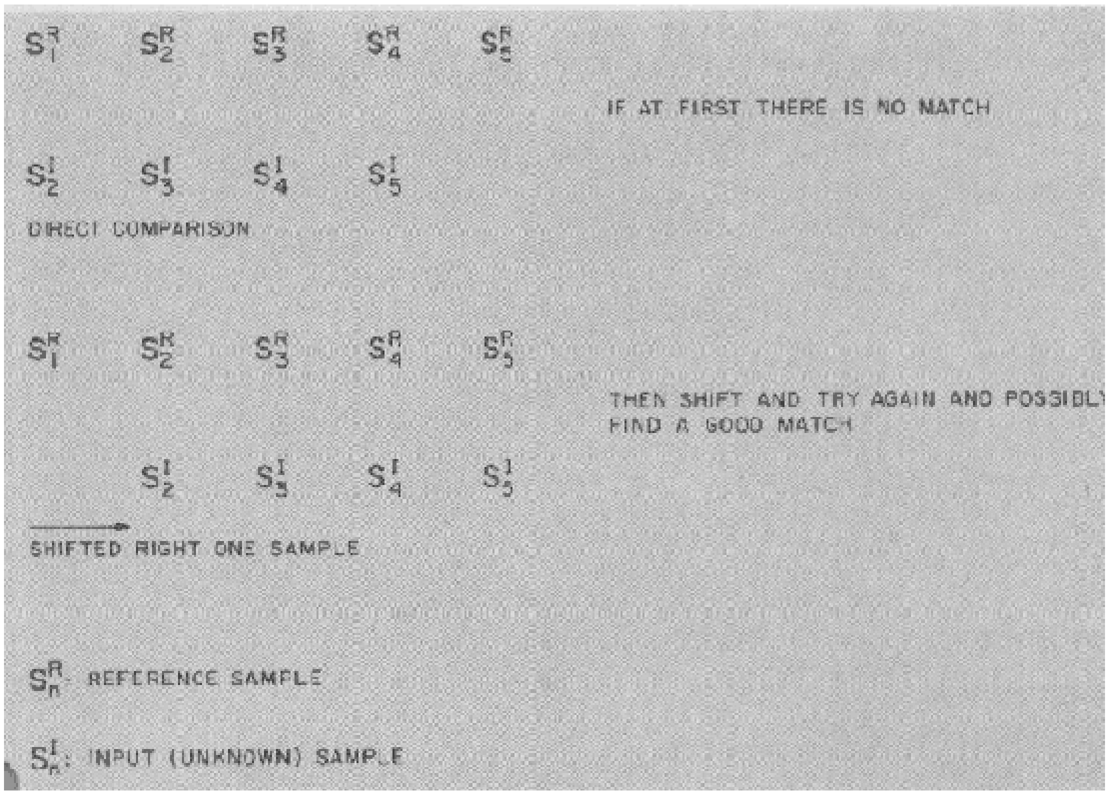

S_1^R S_2^R S_3^R S_4^R S_5^R
IF AT FIRST THERE IS NO MATCH
S_2^I S_3^I S_4^I S_5^I
DIRECT COMPARISON
S_1^R S_2^R S_3^R S_4^R S_5^R
THEN SHIFT AND TRY AGAIN AND POSSIBLY FIND A GOOD MATCH
S_2^I S_3^I S_4^I S_5^I
SHIFTED RIGHT ONE SAMPLE
S_n^R: REFERENCE SAMPLE
S_n^I: INPUT (UNKNOWN) SAMPLE

Figure 16: Shifting the unknown left and then right one position helps prevent mismatches due to missing ends or beginnings. in this case the right shift gave a good match. This test has been simplified to one parameter (S) which might be the output of one filter in a filter bank.

- Performance
  - The performance of a recognizer is evaluated on the basis of the percentage of the words it recognizes correctly out of the total pronounced. For a given recognizer the recognition score will vary from speaker to speaker, with the speaker's emotional condition and with the level of background noise, to mention some of the variables. Recognition scores quoted in articles and advertisements are likely to be the ones obtained with best conditions and they are not necessarily indicative of the performance of the recognizer under actual conditions. There exist three types of recognition errors: failure to recognize a given word, substitution (recognizing one word for another) and misrecognition (recognizing a word not in the vocabulary). The first type of error is in most cases merely an annoyance requiring that the user repeat the word. The last two types of error can cause problems because the recognizer will give a response which is wrong. Failure to recognize will occur when the recognition threshold (the maximum that the smallest difference sum can be to recognize a word) is set too low. Setting the threshold too high will result in misrecognition, as mentioned previously. Substitution becomes a problem when the vocabulary size is increased. To avoid substitution the vocabulary can be divided in two segments and a shift word used to switch between the two segments of the vocabulary. The shift word can be "shift" or "change" for example. This would correspond to the shift key on a typewriter or the function ("f") key on a calculator. The user will have to remember in what segrnent he currently is and when a word from the other vocabulary has to be pronounced he should preface it with the shift word. The system can provide the user with an indication such as an LED to remind him of the current segment. Using a number of different shift words it is possible to extend the technique to several segrnents of a large vocabulary.

- Advanced Systems
  - The hottest areas in speech recognition today are feature extraction using digital speech processing and speech understanding systems. Unfortunately both are out of the reach of the capabilities of the usual amateur computing experimenter's machines. In addition they tend to be involved theoretically so we will only briefly mention them here. Digital speech processing involves operating directly on the sampled time domain waveform and as it was mentioned earlier it requires very fast computers. One of the most usual operations is the Fourier Transform. It is computed using a fast algorithm called the FFT for Fast Fourier Transform and the result is the spectrum of the signal as was shown in the example of figure 5 (actually figure

    5 is the log of the magnitude of the spectrum). Its advantage over output of a filter bank is resolution and accuracy. A typical FFT derived speech spectrum is made up of 128 frequencies spanning the range from 0 to 5 kHz. The amplitude at each frequency is specified with 12 to 16 bit accuracy. Such performance is very difficult to achieve using analog components and certainly not economical. The output of the FFT can be used directly by making different measurements on it to extract features, such as the formants. IBM is using this approach in their research effort in speech recognition which, by the way, is very well funded, in the range of millions of dollars per year.
    - Another digital method for speech processing is "linear prediction." It is a method by which the impulse response of the vocal tract can be extracted from the speech signal. It is basically a time domain operation and it involves predicting the next sample of the waveform on the basis of a linear combination of the N previous samples. It can be used for formant extraction but it is at times quite sensitive to minor aspects of the input signal so it is not as reliable as might be desired. When it works it gives excellent results and therefore it has a lot of promise as a formant tracking tool. It has also been used in isolated word speech recognition by storing the vocal tract impulse response it generates as reference pattern and then comparing it with the impulse response of the unknown. It gave very good results in a vocabulary of 200 words. As it would be expected, this method yields results very similar to the filter bank approach because they are very similar in substance although totally different in implementation.
    - Formant extraction can also be achieved with "cepstral" analysis which is based on the Fast Fourier Transform. The cepstrum is the spectrum of the logarithm of a spectrum. These operations separate the impulse response of the vocal tract from the driving function. Its name is derived from the word spectrum spelled more or less backwards because the operations involved in obtaining it are very similar to spectral analysis of a time domain signal. It gives very good results when used for pitch extraction but when used for formant extraction it is not very reliable. It tends to fail in different ways than the linear prediction methods so perhaps the two methods combined will yield good results.
    - All these methods will become more commonly used as computer technology advances decrease the cost of very fast computers. Because of their accuracy (at least when the data is to their liking) these algorithms are bound to play a significant role in feature extraction of future recognizers.

- Speech Understanding
    - Speech understanding systems differ from speech recognition systems in that they do not just recognize sequences of words but instead they use knowledge about the subject of discussion to check if the word sequences make sense. They use syntactical analysis as previously mentioned as well as context analysis to improve recognition. For example, a speech understanding system used to input the moves of a chess player to the computer (an experimental system doing just that has been built at Carnegie-Mellon University) checks the recognized sentences for syntax errors and then checks whether the sentence gives a legal chess move for the particular situation. Speech understanding systems are usually designed with a specific problem domain in mind for which the semantics are known. The US Defense Department's Advanced Research Project Agency (ARPA) has spent about $30 million on the development of a speech understanding system in the last five years. The results have not been too exciting, an indication that high performance speech recognizers are hard to build.

- Commercial Systems
    - Threshold Technology Inc, Dialog Inc, Perception Technology and Scope Electronics all offer commercial speech recognition systems capable of recognizing isolated words from 16 to 32 word vocabularies. Of these, Threshold Technology is by far the most successful commercially and their systems are used in a variety of industrial applications, such as baggage handling in airports (the destination is spoken to the system which in turn routes the baggage) and quality control inspection (the inspector uses his or her hands to hold a micrometer and take measurements which then are spoken to the recognizer and entered in a computer for report generation). The Threshold Technology machine uses an LSI-11 and their feature extractor consists of a bank of 16 filters. It costs about $10,000. Threshold Technology claims 99.6 percent recognition rate and operation in noisy environments.
    - At the time of this writing (April 1977) 'Heuristics Inc has announced an Altair (S-100) compatible recognizer. It sells for $249 in kit form and a detailed description of its hardware appears in the May 1977 issue of Popular Electronics. It is a simple device consisting of three bandpass filters and a zero crossing detector. The filters cover the range of 150 Hz to 5 kHz.

Their output is peak detected and averaged and then quantized with six bits resolution. They claim 90 percent correct recognition in a quiet environment with a cooperative spleaker.

- As a final remark, a note on recognizer performance is in order. It was mentioned before that quoted performance figures are usually the best obtained with a given system under ideal conditions. This optimistic approach reflects more than anything the difficulty of making a very good recognizer. People choose to present the best results because the bad results may look terrible and the average unimpressive. While recognition scores can be taken with a grain of salt, it would be a mistake to view speech recognizers in the same light. The correct approach is to see them as a challenge to find applications in which they are useful despite their present shortcomings. You certainly would not want to control your car with a recognizer that has a 90 percent recognition score. Yet I have seen a recognizer with an average 80 percent score do a very good job in controlling the electric wheelchair of a paralyzed patient. Think how important this is if you can only move your head and speak and the rest of the body is paralyzed totally.
- I am sure that a great number of applications can be found for existing systems that I just haven't thought of yet, or that I would never imagine. Given the thousands of computer enthusiasts and their different backgrounds and the computer power they cummand, I wonder if there will be any stones left unturned in the applications area now that we can talk to our own speech recognizers.

- Addresses of Commercial Speech Recognition System Manufacturers
    - 1. Threshold Technology Inc 1829 Undervvood Blvd Del ran NJ 08075
    - 2. Heuristics Inc 900 N San Antonio Rd (Suite C-1) Los Altos CA 94022
    - 3. Dialog Systems Inc 639 Massachusetts Av Cambridge MA 02139
    - 4. Scope Electronics Inc Reston VA 22070
    - 5 Perception Technology Inc Winchester MA 01890

- REFERENCES

    - 1 Lindgren, Nilo, "Machine Recognition of Human Language," IEEE Specrrum, March and April 1965. A somewhat antique review of the field. This article covers history and some principles in rather nontechnical language.
    - 2. Otten, Klaus, "Approaches to the Machine Recognition of Conversational Speech," Advances In Compurers, vol 11, Academic Press, NY, 1971. Easily readable, a less antique review than Lindgren's.
    - 3 White, George, "Speech Recognition: A Tutorial Overview," IEEE Computer, May 1976. A very up to date review, rather technical at points but quite readable.
    - 4. Martin, Thomas: "Practical Applications of Voice Input to Machines," Proceedings of the IEEE, April 1976. The author s president of Threshold Technology and from that vantage point reviews the practical side of speech recognition.
    - The reference lists at she end of each of the above papers are good sources of additional information.