Speech recognition techniques applied to speech therapy

by

Richard Mercer Johnson

A Thesis Submitted to the

Graduate Faculty in Partial Fulfillment of the

Requirements for the Degree of

MASTER OF SCIENCE

Department:	Electrical and Computer Engineering
Interdepartmental Program:	Biomedical Engineering
Co-majors:	Electrical Engineering
	Biomedical Engineering

For the Graduate College

Iowa State University
Ames, Iowa
1995

DEDICATION

To my loving wife Nicole, for her patience and support throughout this entire project. Also to my father for his guidance and support and, finally, in memory of my mother.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1. INTRODUCTION

Speech Recognition

Brief History

Research in automatic speech recognition has been going on since the 1950s. In the 1950s and 1960s much of the work on speech recognition focused on recognizing distinct syllables or monosyllabic words. Techniques for performing the recognition were mainly based on analog methods of spectral measurements. Elaborate hardware systems were developed to accomplish these tasks.

In the 1970s research into the area of isolated word recognition improved. Pattern recognition techniques also advanced. Isolated word recognition systems became usable. Linear predictive coding (LPC) techniques began to be used in recognition systems as an accurate spectral distance parameter. Also, dynamic programming techniques began to be developed as a tool for solving speech recognition problems.

In the 1980s the direction of research shifted to connected word recognition. Recognition techniques advanced from template based approaches to statistical modeling methods, especially the hidden Markov model approach. Neural networks also began to be applied in a widespread manner to speech recognition problems. The advanced computation power available through improvements in computer technology also helped to advance speech recognition research.

In the 1990s research in speech recognition is building on many of the same areas made popular in the 1980s. New techniques in hidden Markov models and neural networks are continuously being researched. Applications of speech recognition systems are spreading to many different fields. The widespread use of the personal computer is helping to make speech recognition available to more people.

One of the dreams of many researchers has been to develop a machine that can recognize and understand human speech. Ideally this machine would be able to

communicate with different people in their own language in any environment. However, there are many obstacles that stand in the way of accomplishing this goal.

Difficulties

Much of the difficulty encountered in developing an effective speech recognition system stems from the fact that it is an interdisciplinary problem. Among the disciplines involved in speech recognition are signal processing, pattern recognition, linguistics, anatomy, and physiology. Each discipline presents its own group of problems and its own approach to speech recognition.

The core of any speech recognition system is signal processing which is the process of extracting relevant information from the speech signal. There are many problems that can arise in the signal processing stage of a recognition system. First of all, no two people pronounce the same word in exactly the same way and the duration of the word is variable. Signal processing techniques need to account for these differences. It is also essential for the computer to know when the actual speech signal begins. This is accomplished through a process known as endpoint detection. Endpoint detection becomes difficult when the speech input is a continuous stream of words as opposed to isolated words. In addition, a noisy interface makes it difficult to get an accurate signal.

Pattern recognition can be described as a set of algorithms designed to classify a certain feature or set of features and assign it to a particular class. Pattern recognition techniques common in speech recognition use today are hidden Markov models and neural networks. These methods are computationally intensive and can require a large amount of time to implement. However, when these techniques work properly they can be valuable tools for building effective speech recognition systems.

Linguistics, the study of language, encompasses the relationship between sounds and words as well as the meaning of different words. One obstacle in speech recognition associated with linguistics is homonyms, when there are two words that sound the same but have different meanings. For example, the words *two*, *too*, and *to*. Linguistic techniques can be applied to choose the correct word based on context.

Recognition of phonemes, the basic unit of speech in a language, can create problems as well. Similar sounding phonemes, such as /s/, /sh/, /z/, and /zh/, have similar properties and can be misinterpreted. These sounds may be mispronounced by an individual because they were never learned correctly, are unfamiliar to the speaker, or perhaps because of an anatomical or physiological reason. Another major obstacle involved with speech recognition is *coarticulation*. Coarticulation can be described as overlapping vocal tract shapes during the pronunciation of a word. This can cause changes in the acoustic properties of a phoneme due to its phonetic context. Examples of these are the words *sue* and *say*. When the phoneme /s/ is pronounced in *sue* the lips are extended in anticipation of the ensuing vowel sound. However, when pronouncing *say*, the lips are retracted slightly to prepare for a different vowel sound. The resulting change in vocal tract shape effects the acoustic properties and makes the pattern recognition phase of speech recognition more difficult.

Knowledge of the anatomy and physiology that contribute to speech production is helpful in developing approaches to speech recognition. By examining the anatomy involved with speaking, models can be developed which describe speech production. Through these models more effective speech recognizers can be developed. Modern techniques try to pattern their functionality after the body's own physiological methods for speech production and perception. One common technique patterned after physiology is the neural network.

As seen from above speech recognition draws from a variety of fields and it also can be applied to a broad range of topics. This research project concentrates on speech recognition applications in the field of speech therapy.

Applications to Speech Therapy

Speech Therapy Using the Computer

Part of the job for a speech therapist is to work with a client to help him/her form his/her vocal tract into the shape necessary to produce correct sounds. This can be a very tedious experience for both parties involved. With recent advances in speech recognition techniques and personal computer power it is hoped that the computer can

be used as a tool for the speech therapist. Ideally, the computer could allow the client to practice pronunciation of sounds or words by himself/herself and then provide useful feedback. Appropriate feedback would allow the user to improve his/her pronunciation without the full-time assistance of a speech therapist.

Many people would potentially benefit from computer aided speech therapy. There are an estimated 22 million people in the United States with speech and/or hearing disorders [Boone and Elena, 1993]. Additionally, there are nearly two million people who immigrate into this country every year [Statistical Abstracts of the United States, 1988]. Of these immigrants, only a small fraction speak English well enough to function effectively in society. Many of these people need remedial spoken English programs to help them integrate into the American society. From these figures it is evident that there is a need for methods to help people improve their speaking skills.

Present Products Available

There are presently a number of products on the market designed to improve a person's speech and pronunciation; however, only a couple of these products use modern signal processing techniques. What follows is a brief description of some of these products.

Perfect English Pronunciation [Skills International, 1995] is a set of lessons on two videotapes which reviews forty-six of the most common sounds of the English language. It demonstrates how to form the sounds through the use of animations and actual video clips. Each demonstration is also accompanied by captions so the viewer can read as they practice. One obvious drawback of this product is the lack of feedback provided to the user, thus he/she will be less able to gauge his/her performance.

Video Voice is a product that provides graphic displays of pitch, amplitude, duration, and formant location. It cannot store and play back a user's speech pattern. It is more effective when working with vowels as opposed to consonants. Even with vowels it is highly speaker independent which makes this program unreliable [Ramabadran and Venkatagiri, 1993].

Another product, *Speech Viewer*, provides more information than *Video Voice*. For example, it allows playback of the user's speech patterns. However, it does not allow play back of the target speech sound that the user is trying to emulate. Also, its performance is unsatisfactory as it produces too many false alarms and misses. Finally, there is no feedback provided to the user as to where the problem in his/her pronunciation is [Ramabadran and Venkatagiri, 1993].

The final product to be discussed is *Say & See: Articulation Therapy Software,* a program that runs on Macintosh computers [Hutchins, 1992]. In response to speech input through a microphone, it displays an animated mid-sagittal view of the vocal tract. The purpose of the animation is to give an example of the correct positioning of the vocal tract to produce certain sounds. One drawback of the program is that it uses a sampling rate of 11 kHz which may be too low to capture some higher frequency producing sounds in the English language.

Goal of this Project

As quoted in a recent publication, Bill Meisel, the editor of *Speech Recognition Update* newsletter, said "we have not yet approached the science-fiction ideal of unconstrained continuous-speech dictation or of wide ranging voice conversations with a computer. Nonetheless, applications with more modest objectives are beginning to have a major impact on mainstream markets" [Sweeney, 1995]. One of these mainstream markets is rehabilitation engineering. Voice recognition is used in many devices to aid the physically disabled: keyboard and mouse replacements for data entry, wheelchair and even appliance control. Another branch of rehabilitation engineering deals with speech therapy.

The goal of this project was a "more modest objective" in the field of speech therapy. A common problem among all of the programs described previously is a lack of adequate feedback to the user. The goal of this project was to develop an interactive computer program that would provide adequate feedback to the user. The program was to be developed to run on an IBM compatible personal computer, equipped with a generic sound card and microphone, in the Windows™ environment. By adhering to

these specifications it was the goal of the researcher to produce a product that could be easily used by people in the privacy of their own homes. The scope of this project was limited to individual sounds, or phonemes. A future goal would be to have the option of allowing the user to practice pronouncing these sounds in a syllable, word, or even sentence.

CHAPTER 2. SPEECH PRODUCTION, PERCEPTION, AND PROCESSING

Introduction

When two people communicate through speech, one person produces a speech signal and, ideally, the other listens. The speech signal is in the form of pressure waves. Variations in the sound waves are produced by positioning the different parts of the vocal system in certain ways. The sound waves are picked up by the listener's ears where they are converted into neural firings in the inner ear. Finally the auditory nerve transmits these messages to the brain where they are converted into some meaningful information.

Speech Production

Anatomy and Physiology of Speech Organs

The organs that produce speech are also used for other bodily functions such as breathing, eating, and smelling. Speech is generally produced by exhaling air through the vocal system. The vocal system can be divided into three main groups: lungs, larynx, and vocal tract. Sometimes the term *vocal tract* can refer to the complete *vocal system;* however, in this paper *vocal tract* will only refer to the one group of the *vocal system*.

Lungs and the Thorax. The lungs are the source of airflow for the speech process. Their primary purpose is for breathing, inspiring and expiring air. Expiring constitutes about 60% of the breathing cycle for normal breathing. The breathing cycle is accomplished through the use of the diaphragm, the intercostal and the abdominal muscles. The diaphragm contracts as the external intercostals pull the rib cage up and outward. This expands the volume of the intrathoracic cavity creating a pressure gradient which allows air to be inspired. As the diaphragm relaxes and the internal intercostal muscles pull the ribs inward the intrathoracic volume decreases and air is

exhaled. It is during the exhalation process that sounds in virtually all languages are produced.

The lungs influence the amplitude (loudness) of speech because amplitude is related to airflow rate and volume. The *volume velocity* of exhaled air during speech is controlled by the chest and abdominal muscles at about 0.2 liter/second during sustained sounds. Total lung capacity for a normal adult male is about 6.0 liters. For a female this value is about 4.2 liters. The *residual volume* is the air left in the lungs after a maximal expiratory effort. Normal values for this volume are 1.2 liters for men and 1.1 liters for women. Finally, the *vital capacity* is the largest amount of air that can be expired after a maximal inspiratory effort. Normal values for this volume are 4.8 liters for men and 3.1 liters for women. Ordinary speech uses up to half of the vital capacity while very loud speech uses as much as 80% [Ganong, 1991]. While the lungs can help to produce different volumes and velocities of air passing through the vocal system, the larynx and vocal tract help to vary and modulate the airflow rate.

Larynx and Vocal Folds. After air leaves the lungs it passes through the bronchi and then through the trachea to the larynx. The larynx contains nine cartilages stabilized by ligaments and/or skeletal muscles [Martini, 1992]. Four notable cartilages in the larynx are the thyroid, cricoid, arytenoid, and epiglottis. The epiglottis seals off the larynx when eating. Within the larynx are vocal folds, a pair of elastic structures of tendon, muscles, and mucous membrane that lie in an anterior-posterior direction behind the thyroid cartilage (Adam's apple). Another structure located just above the vocal folds, called the ventricular folds, are not so elastic, these help to protect the vocal folds. Because the vocal folds are involved with speech production they are known as the true vocal cords, while the ventricular folds are often called the false vocal cords [Martini, 1992]. The vocal cords are typically 15 mm long in men and about 13 mm long in women. The glottis is the variable opening between the vocal cords. This opening is about 8 mm wide at rest [O'Shaughnessy, 1987].

During normal breathing the vocal cords are in an open position allowing air to freely move in and out. Generally there is no audible sound while breathing. Sound

occurs when there is an obstruction somewhere in the vocal system. If the vocal cords are adducted sufficiently, the air passing through the glottis vibrates the vocal cords and produces sound waves. When the vocal cords vibrate during speech this is called *voiced* speech. The vocal cords can be compared to the strings of a musical instrument. For example: short, thin, and tense strings produce higher frequency sounds, whereas long, thick, and loose strings produces lower frequency sounds. The true vocal cords of an adult male are thicker and longer and they produce lower tones than those of an adult female. The tension in the vocal cords is controlled by skeletal muscles that are under voluntary control. Although sound can be produced with the vocal cords in the larynx, clear speech requires further articulation by the vocal tract.

Vocal Tract. The vocal tract begins at the opening of the vocal cords and ends at the lips (see Figure 2.1). After air passes through the glottis, it next passes through the pharynx, the oral cavity, and finally through the lips. When the velum (a moveable tissue structure at the back of the mouth cavity) is lowered, the nasal cavity is also coupled to the vocal tract to produce the nasal sounds of speech. In the average male, the total length of the vocal tract is about 17 cm. For females, this value is about 15 cm. [Parsons, 1986].

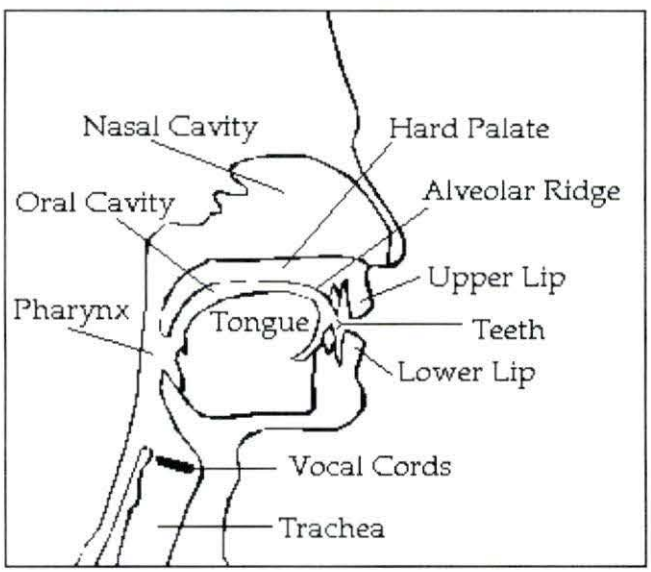


Figure 2.1. Simplified anatomy of the vocal tract.

Classes of Sounds

Every language has a certain set of linguistic units called phonemes to describe its sounds. A phoneme is the smallest meaningful unit of speech in a language. The number of phonemes varies from language to language, but is typically between 20-40. Phonemes can be classified into three main groups: fricatives, plosives, and sonorants.

Fricatives. Fricatives can be either voiced or unvoiced sounds. They are characterized by a narrow fixed obstruction of the vocal tract beyond the vocal cords. The obstruction usually involves either the tongue and the roof of the mouth or the lips and teeth. Examples of fricatives are the /sh/ in *shoe*, the /v/ in *valve*, and the /th/ in *thin*.

Stops or Plosives. Stops or plosives are produced when the vocal tract or glottis is closed completely, allowing pressure to build up, and then opened to allow the air to be released. Examples of stops are /p/ in *pop*, /t/ as in *tot*, and /k/ as in *kick*.

Sonorants. Sonorants are perhaps the most important class of sounds. Sonorants result from an excitation of the vocal folds. Air leaving the lungs is interrupted by periodic opening and closing of the vocal cords. The rate of vibration of the vocal cords is called the fundamental frequency (f0). Examples of sonorants include most vowels, such as /e/ as in *bet* and the /u/ sound in *boot*, and many consonants also.

Although most sounds can fit into one of the above classes there are still many different approaches to analyzing speech sounds. These include: articulatory, acoustic, phonetic, and perceptual. The remainder of this section will focus on articulatory phonetics. Following sections in this chapter will discuss the other approaches.

Articulatory Phonetics

The articulatory phonetics approach to analyzing speech sounds relates certain phonemes to positions and movements of the speech organs. There is little data in this area because it is difficult to obtain an accurate picture of the exact motion of the speech organs. Visual observation does not allow a full view of all the speech organs, and X-ray observation does not provide a complete three-dimensional model. When

describing the position and movements of the speech organs there are three basic categories of description.

Manner of Articulation. Manner of articulation describes the airflow through the vocal system and is concerned with the path air takes and the degree to which airflow is impeded. Vowels, diphthongs, and sonorants are characterized by air flow through the vocal system which meets no obstruction narrow enough to cause turbulent flow (frication) [O'Shaughnessy, 1987].

Glides and liquids are similar to vowels. The difference in glides is that they are caused by a narrow constriction in the vocal tract that sometimes can cause frication. Liquids use the tongue as an obstruction for the air to pass around. Nasals are produced when the velum is lowered and the oral cavity is completely closed off. In the English language all nasals are consonants. Other languages use nasalization to differentiate between different vowels. However, in English, nasality is not a distinctive feature of vowels.

Fricatives, affricates, and stops can also be characterized by their manner of articulation. As previously discussed, stops are produced by a complete closure of the vocal tract which allows air to build up and then released. Fricatives occur when the vocal tract is narrowly constricted at a certain location, causing turbulent flow. Affricates can be described as a stop followed by a fricative. Examples are shown in Table 2.1.

Voicing. Voicing is the second basic category of description in articulatory phonetics. Speech production can result from a periodic source, resulting in voiced speech, or from a noisy and aperiodic source, resulting in unvoiced speech. Voiced speech occurs when the vocal cords are drawn together close enough to cause them to vibrate as a result of the air passing through them. The fundamental frequency that the vocal cords vibrate at corresponds to the perceived pitch. If the speech is unvoiced, then there must be an obstruction of the vocal system in another location causing the speech sound. The location of this obstruction, or *place of articulation,* described next.

Place of Articulation. While the manner of articulation and voicing help to separate phonemes into broad classes, the place of articulation provides finer differentiation between phonemes. For example, the place of articulation for the /s/ in *sea* is at the tongue and hard palate just behind the front teeth, termed alveolar strident, while the place of articulation for the /sh/ in *shell* is at the tongue and middle of the alveolar ridge, termed palatal strident. For the vowel sounds the place of articulation is not necessarily at a certain location but can be described as a change in the vocal tract shape. For example the /u/ sound in *boot* is described as having it's place of articulation as "high back tense rounded" [O'Shaughnessy, 1987]. See Table 2.1 for more examples of the terminology used in describing the location of articulation.

Speech Perception

Articulatory phonetics provides a precise manner in terms of speech organ positioning to describe each phoneme. However, there are times when the speech organs are not in, what would be considered, the correct position and yet speech can still be understood. Speech perception is the method of understanding the speech message once it has left the source.

Many times in communication there are other cues that communicate meaning besides just the speech sound itself. The context of the speech may help one to understand the message. For example, homonyms such as *to*, *two*, and *too* all sound alike. It is only by the context of the sentence in which they are spoken that the meaning of the word is realized. In a like manner, the speaker can play a large role in communication. The speaker's personality, actions, geographic background and other factors can all influence the meaning of the spoken message.

By incorporating certain principles from the speech perception area of study, one can gain a fuller understanding of speech. A speech recognition device that does not account for these factors will not be as effective.

The purpose of this project, however, was not to construct a pure speech recognition device. Therefore the role of speech perception was limited. The focus of this project was to help a person practice producing isolated sounds.

Table 2.1. English phonemes and corresponding classifications [O'Shaughnessy, 1987].

Phoneme	Manner of Articulation	Place of Articulation	Voiced	Example
i	vowel	high front tense	yes	beat
I	vowel	high front lax	yes	bit
e	vowel	mid front tense	yes	bait
∈	vowel	mid front lax	yes	bet
æ	vowel	low front tense	yes	bat
α	vowel	low back tense	yes	cot
⊃	vowel	mid back lax rounded	yes	caught
o	vowel	mid back tense rounded	yes	coat
U	vowel	high back lax rounded	yes	book
u	vowel	high back tense rounded	yes	boot
∧	vowel	mid back lax	yes	but
ɜ	vowel	mid tense (retroflex)	yes	curt
ə	vowel	mid lax (schwa)	yes	*a*bout
αj (αI)	diphthong	low back →high front	yes	bit
⊃j (⊃I)	diphthong	mid back →high front	yes	boy
αw (αU)	diphthong	low back→high back	yes	bout
j	glide	front unrounded	yes	you
w	glide	back rounded	yes	wow
l	liquid	alveolar	yes	lull
r	liquid	retroflex	yes	roar
m	nasal	labial	yes	maim
n	nasal	alveolar	yes	none
η	nasal	velar	yes	bang
f	fricative	labiodental	no	fluff
v	fricative	labiodental	yes	valve
θ	fricative	dental	no	thin
δ	fricative	dental	yes	then
s	fricative	alveolar strident	no	sass
z	fricative	alveolar strident	yes	zoos
∫	fricative	palatal strident	no	shoe
zh	fricative	palatal strident	yes	measure
h	fricative	glottal	no	how
p	stop	labial	no	pop
b	stop	labial	yes	bib
t	stop	alveolar	no	tot
d	stop	alveolar	yes	did
k	stop	velar	no	kick
g	stop	velar	yes	gig
tsh	affricate	alveopalatal	no	church
dzh	affricate	alveopalatal	yes	judge

Speech Processing and Acoustic Phonetics

Acoustic refers to hearing, thus acoustic phonetics deals with describing speech by what it sounds like. Each sound has certain audible properties which can help identify it. Most speech recognition systems use the acoustic phonetic method as the basis for their operation.

Perhaps the most common way that people are used to seeing a plot of a speech signal is the *time domain* representation. In the time domain a speech signal is represented as a waveform on a plot of amplitude versus time (see Figure 2.2). Time domain plots can provide information about the loudness of the signal. They can also give the observer of the plot an rough idea of the frequency content of the signal.

In speech processing applications another common representation of speech signals is in the *frequency domain*. The frequency domain provides additional information about the speech signal. Often by examining the frequency domain of a speech signal certain characteristics are noticeable that may not have been as easily discernible in the time domain (see Figure 2.3).

The different frequencies that are present in a speech signal result from the shape of the vocal tract. The vocal tract can be modeled as an acoustic tube. When air is passed through a tube it resonates at certain frequencies. If the tube changes length

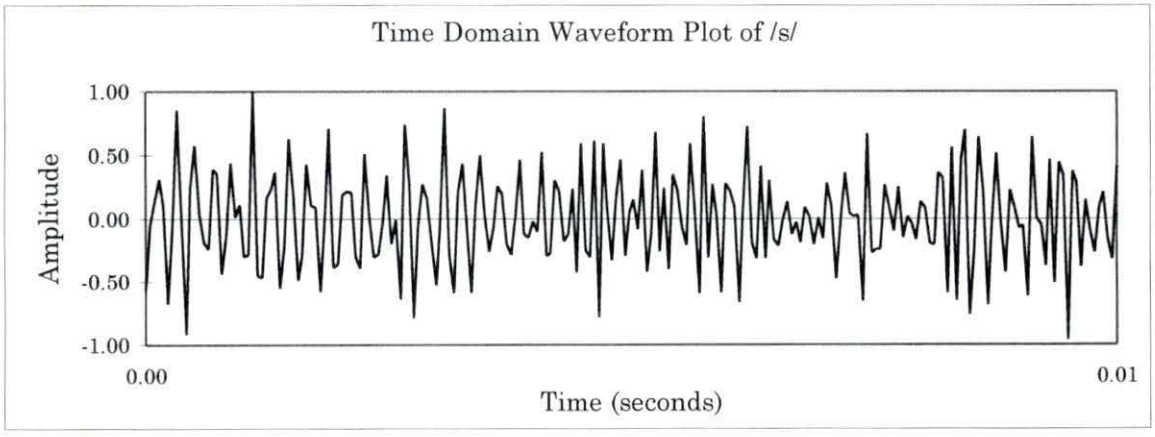


Figure 2.2. Time domain plot of the phoneme /s/.

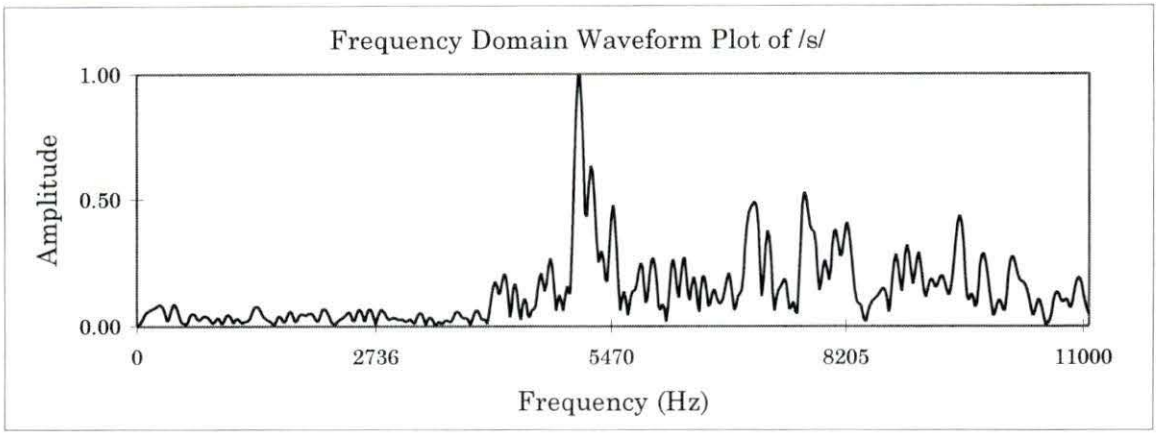


Figure 2.3. Frequency domain of /s/ using FFT.

the resonating frequencies change also [Halliday and Resnick, 1986]. In the vocal tract these resonating frequencies are called *formants* (abbreviated F*i*, where F1 is the formant with the lowest frequency). As the organs of the vocal tract change position to form new sounds the formants also change.

Spectral Characteristics of Phonemes

Vowels are often characterized by their first two formant frequencies, F1 and F2. Diagrams such as the one shown in Figure 2.4 show the range of values for F1 and F2 in different vowel sounds. There is some overlap between certain vowel sounds, however, by using F3 in a three-dimensional plot the separation between sounds is fairly distinct. Diphthongs consist of a changing vowel sound as the vocal tract changes position to produce the sound. Likewise, the representation of a diphthong in the frequency domain can be described in terms of formant frequencies that are rising or falling.

Fricatives and stops need to be separated in terms of whether they are voiced or unvoiced when describing their spectral characteristics. Fricatives that are unvoiced are not described in terms of their formants because low frequencies are not excited. Instead they are characterized by a high frequency spectrum proportional to the length of the vocal tract cavity. For example, palatal fricatives can have frequency spectrums

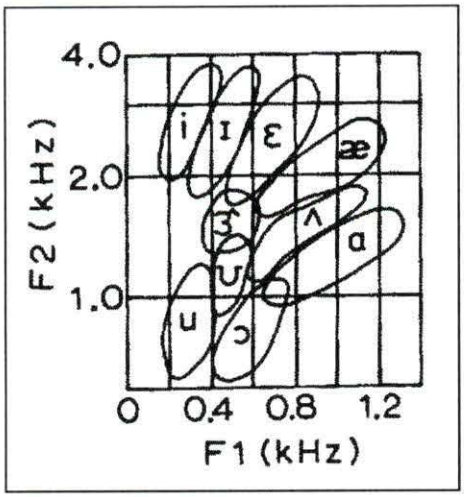


Figure 2.4. F1-F2 chart for several vowels [Kent, and Read, 1992].

beginning at around 2.5 kHz, while labial and dental fricatives can have much of their energy present around 8 kHz. Voiced fricatives are produced using two acoustic sources, the periodic glottal source (voicing) and the frication noise produced by the vocal tract constriction. Thus the frequency spectrum of a voiced fricative will have both low frequency energy present due to the voicing as well as high frequency energy due to unvoiced fricatives. Stops can be characterized as an absence of energy (or low frequency energy if voiced) followed by an sudden excitation of the frequencies that would be present in a fricative having the same place of articulation.

In the production of all phonemes it is important to realize that the vocal tract can change shape very quickly, sometimes in less than 20 milliseconds. As the vocal tract changes shape, the frequency characteristics also change. Therefore it is important to accurately capture the true spectral characteristics of the speech sample.

Obtaining Frequency Information

Analog to Digital Conversion. When using a digital computer to analyze the speech information, the speech signal needs to be converted from its usual analog form to digital form that can be stored in a computer. Three important factors in converting

this information accurately are the microphone, the sampling rate, and the sampling resolution.

Microphone. The microphone used in the analog to digital (A/D) conversion process usually introduces undesired side effects, such as 60 Hz line noise. It also can cause nonlinear distortion and loss of low and high frequency information [Picone, 1993]. Higher quality microphones are designed to minimize these undesired side effects.

Sampling Rate. Also of importance in obtaining accurate frequency information is the *sampling rate* at which the speech is obtained. The sampling rate is the number of samples of the original analog signal taken each second. If the sampling rate is too low it will be impossible to accurately represent the true speech signal. However, if the speech signal is sampled at or above the *Nyquist*[*] sampling rate then no frequency information will be lost.

Sampling Resolution. Sampling resolution refers to the number of bits used to describe each sample. A bit is a binary value which can be either a *one* or a *zero*. The more bits that are used to describe each sample, the better the resolution of the speech sample will be. For example, many sound boards use either 8-bit or 16-bit sampling resolution. 8 bits can be used to describe values from 0 to 255 units, while 16 bits can be used to describe values from 0 to 65536.

A/D Conversion Methods used in this Project. A high quality microphone was used in this project to acquire all of the test sounds. Windows™ programming provides support for sampling rates of 11, 22, and 44 kHz, therefore, in this project a sampling rate of 22 kHz was used. Finally, a sampling resolution of 16 bits was used in order to capture as much information as possible. Many sound cards that presently exist in many computers today are only equipped to handle 8-bit

---

[*] The Nyquist sampling rate is twice the highest frequency that is present in the original signal. Some fricatives such as /s/ can have frequencies of up to 8 kHz present.

resolution and lower sampling rates. Therefore, the program was also designed to operate at an 8-bit sampling resolution and 11 kHz sampling rate.

Digital Filtering. Once the signal has been converted to digital form the next step that is often executed is filtering the signal using a Finite Impulse Response (FIR) filter. Normally this is a one coefficient digital filter, known as a *preemphasis filter*, of the form in Figure 2.5, where a typical range for a_{pre} is between and including 0.4 to 1.0 [Picone, 1993].

$$H(z) = 1 - a_{pre} z^{-1}$$

Figure 2.5. Preemphasis filter transfer function.

There are two common explanations for using this filter [Picone, 1993]. The first is that voiced sections of speech naturally have a negative spectral slope of approximately 20 dB per decade due to physiological characteristics of the speech production system. The preemphasis filter serves spectrally flatten the signal and thus improve the signal analysis.

The second explanation is that hearing is more sensitive above the 1 kHz region of the spectrum. The preemphasis filter amplifies this section of the spectrum, thus giving greater emphasis to the perceptually important parts of the spectrum.

Digital Filter used in this Project. The digital filter used in this project was a preemphasis filter as shown in Figure 2.5. The user is given the option to choose a value for a_{pre} in order to suit the circumstances. The user is also given the option to use the preemphasis filter or not. For certain applications the preemphasis filter may not be necessary.

Segmentation. The next step in obtaining any frequency information about a speech signal is to segment the time domain signal into small enough sections where the vocal tract can be considered to be in a constant position. Then the signal can be considered *stationary* and can be analyzed appropriately. A signal can be considered

stationary if its statistical characteristics do not change with time. However speech signal properties vary considerably during the pronunciation of a word, and thus are considered to be nonstationary. The practical solution to this problem is to divide the nonstationary signal into blocks of short segments, in which each segment can be assumed to be stationary. However, the problem with this method is the length of the desired segment. Choosing a short segment may cause poor frequency resolution. However, if the segment chosen is too long, it can no longer be considered stationary.

Segmentation used in this Project. The proper segmentation period is ultimately dependent on the rate of change of the vocal tract. As the vocal tract changes shape, the spectral characteristics change also. Some speech sounds, such as stop consonants or diphthongs, exhibit sharp spectral transitions which can result in spectral shifts of as much as 80 Hz/ms. However, segmentation periods less than 8 milliseconds are not normally used [Picone, 1993]. Speech signals are often segmented into 10 millisecond sections and that is the segmentation period used in this project.

Fourier Transform. The principle behind the Fourier transform is that any series can be represented by a sum of sine and cosine functions. When the series is a sequence of discrete values the process is called the discrete-in-time, continuous-in-frequency Fourier transform (DTFT). The equations governing the DTFT are presented in Figure 2.6.

However, in many practical situations the discrete Fourier transform (DFT) is employed because it uses both discrete time and frequency components which allows easier implementation. The equations governing the DFT are given in Figure 2.7.

$x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X\left(e^{j\omega}\right) e^{j\omega n} \, d\omega$	Inverse Fourier Transform
$X\left(e^{j\omega}\right) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\omega n}$	Fourier Transform

Figure 2.6. Equations governing the DTFT.

$$x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]W^{nk} \qquad \text{Inverse DFT}$$

$$X[k] = \sum_{n=0}^{N-1} x[n]W^{-nk} \qquad \text{DFT}$$

$$\text{where, } W_N = e^{\frac{j2\pi}{N}} = \cos\frac{2\pi}{N} + j\sin\frac{2\pi}{N}$$

Figure 2.7. Equations governing the DFT.

Linear Predictive Coding (LPC). Another method of representing the signal is through the use of linear prediction. Linear prediction tries to predict the current output based on a knowledge of a certain number of previous outputs. The number of previous outputs used in the prediction is called the prediction order. The linear prediction equations are given in Figure 2.8. In these equations $\hat{y}$ is the predicted output, *y[n]* are the previously known outputs, *-a[i]* are the predictor coefficients, p is the prediction order, and *e[n]* is the prediction error.

When the Fourier transform is performed on the LPC coefficients, the result can be considered as the spectral envelope of the speech signal. As can be seen in Figure 2.9 the LPC spectrum is much smoother than the FFT spectrum which often makes it easier to work with. Figure 2.10 shows the LPC spectrum of the same speech sample for different prediction orders (p). As can be seen from the figure, the higher the prediction order the more accuracy is obtained. However, after a certain order the prediction error increases significantly. The optimal prediction order for speech samples is in the range of ten to fifteen, depending upon the sampling rate and the characteristics of the signal. While the LPC coefficients provide a smoother spectrum than the FFT, their main advantage is the compression of information into a smaller set of data.

Features

After processing the speech sample it is then necessary to develop a set of features which will represent the speech sample in a more compact form. For example,

$$\hat{y}[n] = -\sum_{i=1}^{p} a[i]y[n-i]$$

$$e[n] = y[n] - \hat{y}[n] = \sum_{i=0}^{p} a[i]y[n-i]$$

Figure 2.8. Equations describing linear prediction.

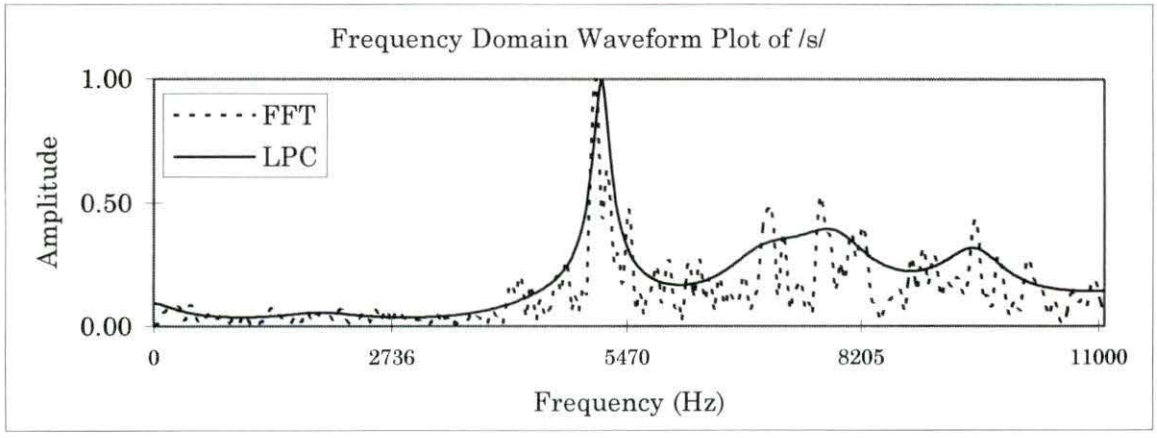


Figure 2.9. LPC frequency waveform versus FFT frequency waveform.

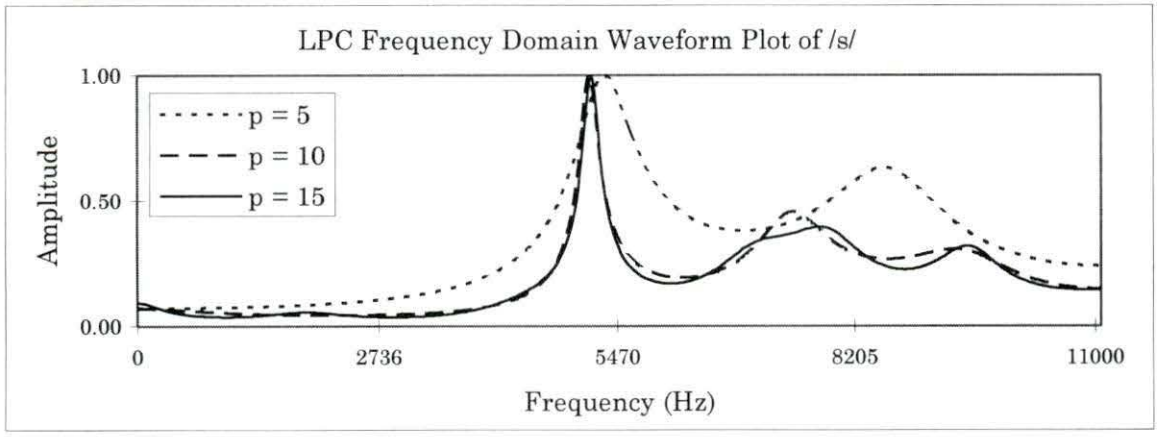


Figure 2.10. LPC frequency spectrum plotted with different prediction orders.

if one second of speech is digitized at 22.05 kHz there will be 22050 samples. Furthermore, if the sampling resolution is 16 bits/sample then there will be 44100 bytes of information for the computer to process. Using LPC techniques it is possible to represent the same information using the LPC coefficients. Assuming that LPC coefficients are derived from a 10 millisecond block of data, or 220 samples, then a prediction order of 11 would provide a compression ratio of 220:11. Therefore, the LPC coefficients of a speech sample can be considered a feature for that sample.

Features used in this Project. The features developed for this project were based solely on the LPC coefficients. A prediction order of fifteen was used for all the tests in this project, however, the user has the option to change the prediction order. As previously mentioned, only single sounds, assumed to be constant, were used in this project. This made it possible to select any 10 millisecond block of data in the recording. The LPC coefficients were then found for this block. The user of the program is also given the option to choose a few successive blocks of data and average the resulting LPC coefficients. This helps in obtaining more robust estimates. The resulting feature vector for each phoneme produced was a sequence of fifteen numbers. These feature vectors were used in the program to determine the phoneme that was produced. The method of classifying each phoneme is described in the next chapter.

CHAPTER 3. CLASSIFICATION TECHNIQUES

Over the past 25 years, four main classification techniques have been used for speech recognition: template matchers, rule-based systems, neural networks, and hidden Markov model (HMM) systems [Roe and Wilpon, 1993]. Common among all techniques is that the classifier must align the unknown feature vector with the optimum feature vector representing a speech unit (a phoneme for this project). In this project three of these classification techniques were implemented to determine which would be best suited for operation in the computer program for this project.

Template Matchers

The main idea behind template matchers is that each speech unit can be represented by a feature vector, or a series of feature vectors, called templates. A feature vector is then obtained for the unknown speech unit and compared to the templates for the known speech units. The template with the closest match is considered the correct result. A number of distance measures have evolved that are employed to determine which template is closest to the unknown feature vector. Among these distance measures are least mean square (LMS) distance, Mahalanobis distance, and even distance measures designed with LPC coefficients in mind: Itakura-Saito measure, and Itakura's minimum prediction residual [Parsons, p. 174].

In most speech recognition problems individual words are often being classified. The rapidity with which the words are spoken may vary from speaker to speaker and from instance to instance. This may cause problems with aligning the features correctly. When this is a concern, a Dynamic Time Warping (DTW) technique may be used to stretch or shrink the time axis to assist in alignment with the reference signal [Roe and Wilpon, 1993]. This project has avoided this problem because the sounds being analyzed are individual phonemes.

In this project an LMS distance measure was used as one of the classifying techniques. This was implemented in two different ways. In each method templates were first developed for the correct phoneme. Then, in the first method of

implementation, templates were also developed for the incorrect phonemes. Incorrect phonemes are those which the user substitutes in place of the correct ones. A feature vector from the test phoneme was then created and compared to all of the templates. The test phoneme was classified as the phoneme corresponding to the template it was closest to. In the second method of implementation, the feature vector from the test phoneme was compared only to the template for the correct phoneme. The LMS distance was then supplied as feedback to the user. This guides the user to pronounce his/her sounds closer to the correct phoneme.

Rule-Based Systems

Rule-based systems set up a series of criteria in a decision tree to determine which of the units of speech is present in the speech signal. One problem with this method is that if an incorrect decision is made early in the decision tree it is hard to recover from that error. In addition, it has been difficult to develop a comprehensive set of criteria for large and complex speech recognition tasks. The differences between the template and the rule-based approaches resulted in a philosophical split in the research community until the early 1980s when both approaches were surpassed by a more powerful theory, the hidden Markov model [Roe and Wilpon, 1993].

Hidden Markov Models

Hidden Markov model (HMM) systems are currently the most successful speech recognition algorithms [Rabiner, 1989]. They are so successful because they use a statistical approach to model the speech unit. Additionally, HMMs automatically incorporate time normalization into their methodology.

The HMM assumes that the speech signal can be modeled as a parametric random process and that the parameters of the stochastic process can be determined in a precise, well-defined way [Rabiner and Juang, 1993]. An ensemble of speech data is used to train the HMM, thus developing a probabilistic model which characterizes the entire ensemble. This resulting model is generally more effective for recognition purposes than a template based method.

Example of an HMM

As a simple example of a Markov model consider the following example. Suppose a model has been developed to describe the weather in Arizona, where it is usually sunny and dry. Suppose another model has been developed to describe the weather in Seattle, where it rains a lot. Hypothetical models for these two conditions are shown in Figures 3.1 and 3.2.

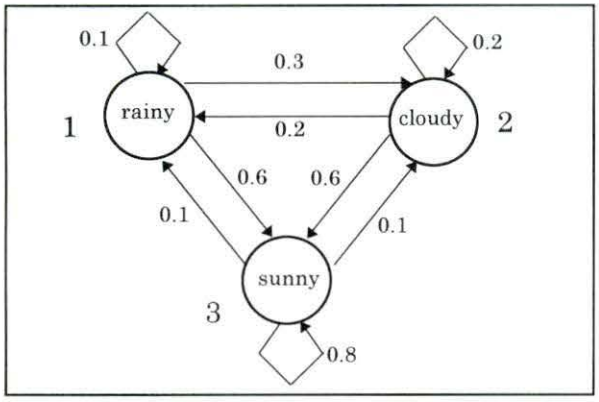


Figure 3.1. Markov model of the weather in a dry environment.

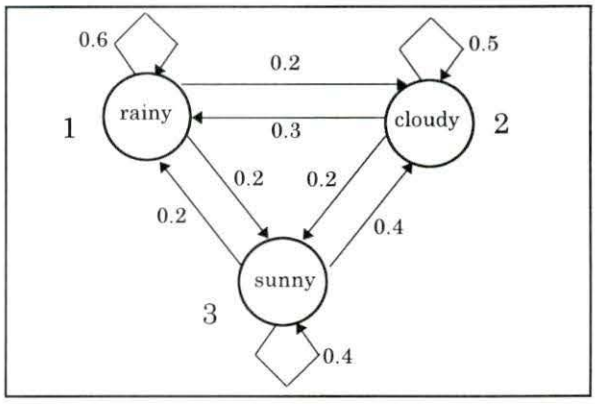


Figure 3.2. Markov model of the weather in a wet environment.

In this example there are three states, each describing a possible weather condition at a certain time once a day: state 1 is rainy, state 2 is cloudy, and state 3 is sunny. Each state has three arrows leaving it; one gives the probability of the next day remaining in that same state, the other two give the probabilities of the next day being in one of the other states. For the model in Figure 3.1, it can be seen that much of the time will be spent in state 3 as it has a relatively high probability of remaining in its present state. For the model in Figure 3.2, much of the time will be spent in either states 1 or 2.

Now, if a record is kept of the observations of the weather for a one week period it may look something like [sunny, sunny, rainy, rainy, cloudy, rainy, cloudy]. Then, by comparing the observation sequence to both models developed, a probability of the

sequence resulting from each model can be developed. The model with a higher probability would give the location that the observation sequence would match.

This same principle is used for speech recognition purposes. A distinct model is developed for each word in the recognizer's vocabulary. Then, a feature vector from a test word is compared to each model. Whichever model generates the highest probability is considered the matching word.

Notation for an HMM

In order to characterize an HMM it is necessary to first describe the terminology. The following terms, originally used by Rabiner and Juang [1986], help to describe an HMM:

1. N is the number of states in the model. Often the states have some physical meaning attached to them. Also, they are interconnected so that any state can be reached from any other state; however, different configurations are possible which may suit different applications, such as speech processing. The individual states are labeled as $\{q_1, q_2, \ldots, q_N\}$.

2. M is the number of distinct observation symbols per state. For example, if each state represented a coin that was being tossed for heads or tails, M would be two. The individual symbols are labeled as $\{v_1, v_2, \ldots, v_M\}$.

3. A is the probability of moving to any state from the present state, or the state transition probability. $A = \{a_{ij}\}$ where $a_{ij} = P[q_{t+1} = j | q_t = i], \quad 1 \le i, j \le N$.

4. The probability distribution for the symbols in state j, where $j = 1, 2, \ldots, N$, is given by: $B = \{b_j(k)\}$, where $b_j(k) = P[o_t = v_k | q_t = j]$, $1 \le k \le M$, where o_t is the observation at time t.

5. The initial state distribution is the probability of starting in a certain state. It is represented by $\pi = \{\pi_i\}$, in which $\pi_i = P[q_1 = i], \quad 1 \le i \le N$.

Using these five values it is possible to represent all the parameters necessary to describe an HMM. Often the compact notation $\lambda = (A, B, \pi)$ is used to represent an HMM [Rabiner and Juang, 1986].

Practical Applications for the HMM

In order to apply the HMM to practical applications it is necessary to solve two problems. The first problem is training. When given several observation sequences (feature vectors from the same sound) the HMM that maximizes the probability of generating those observations must be created. The model parameters are found by an iterative procedure known as the Baum-Welsh reestimation formula [Rabiner and Juang, 1986]. An outline for the Baum-Welsh reestimation procedure is shown in Figure 3.3. This process of re-estimation is equivalent to a steepest-descent gradient search procedure [Roe and Wilpon, 1993].

Baum-Welsh reestimation formulas

Step 1: Recalculate the initial state distribution using the formula:

$\bar{\pi}_i = \gamma_1(i), \qquad 1 \le i \le N$, where $\gamma_t(i) = \dfrac{\alpha_t(i)\beta_t(i)}{\Pr(O|\lambda)}$. The equations for $\alpha_t(i), \beta_t(i)$, and $\Pr(O|\lambda)$ are given in the forward-backward procedure (Figure 3.4).

Step 2: Recalculate the state transition probabilities using the formula:

$\bar{a}_{ij} = \sum_{t=1}^{T-1} \xi_t(i,j) \Big/ \sum_{t=1}^{T-1} \gamma_t(i)$, where $\xi_t(i,j) = \dfrac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\Pr(O|\lambda)}$. Equations for the other values are given in the forward-backward procedure.

Step 3: Recalculate the symbol probability distribution using the formula:

$\bar{b}_j(k) = \sum_{\substack{t=1 \\ O_t = k}}^{T} \gamma_t(j) \Big/ \sum_{t=1}^{T} \gamma_t(j)$.

Figure 3.3. Baum-Welsh reestimation formulas.

The second problem is classification. When given a set of several HMMs and an observation sequence it must be determined which model generated that sequence. The forward-backward algorithm [Rabiner and Juang, 1986] may be used to do this. An outline for the forward-backward algorithm is shown in Figure 3.4.

The forward-backward procedure

Given an observation sequence O, define the forward variable as $\alpha_t(i) = \Pr(O_1, O_2, \ldots, O_t, i_t = q_i | \lambda)$. Or in other words, it is the probability of the partial observation sequence (until time t) and state q_i at time t, given the model λ. It is then possible to solve for $\alpha_t(i)$ inductively, as follows:

Step 1: $\alpha_1(i) = \pi_i b_i(O_1), \quad 1 \le i \le N;$

Step 2: $\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right] b_j(O_{t+1}), \quad 1 \le j \le N$, and t = 1,2, . . . , T - 1.

Step 3: then, $\Pr(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$.

The backward variable can also be defined as:
$\beta_t(i) = \Pr(O_{t+1}, O_{t+2}, \ldots, O_T | i_t = q_i, \lambda)$. Or in other words, the probability of the partial observation sequence from t+1 to the end, given state q_i at time t and the model λ. Again, it is possible to solve for $\beta_t(i)$ inductively, as follows:

Step 1: $\beta_T(i) = 1, \quad 1 \le i \le N;$

Step 2: $\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j), \quad 1 \le i \le N$, and t = T - 1, T - 2, . . . , 1.

Figure 3.4. The forward-backward procedure.

For most speech recognition applications a left-to-right HMM is used, see Figure 3.5. This type of HMM has the property that as time increases, the state index increases or stays the same. This can be easily related to signals that change over time in a successive manner, such as speech. Constraints on this type of model are often modified to allow transitions to skip a state or two.

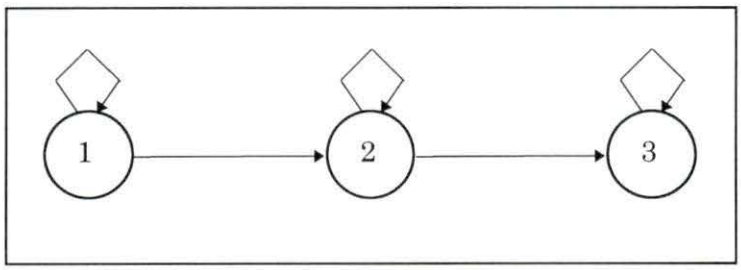


Figure 3.5. 3-state left-to-right HMM.

HMM Used In This Project

The type of HMM used in this project was a left-to-right HMM as shown in Figure 3.5. The number of states in the model was left as an option for the user to specify. The flexibility in specifying the number of states can be used as a tool for improving the overall performance. The feature vector for a particular sound was normalized to have values between -1 and 1. These values are rounded off to the nearest tenth, providing twenty-one distinct observations [-1.0, -0.9, -0.8, . . . , 0.0, . . . , 0.8, 0.9. 1.0]. The model was then trained using the Baum-Welsh re-estimation formula. Training was considered complete when a user specified error value was achieved. The Baum-Welsh forward-backward procedure was then used for classification purposes.

The left-to-right HMM was then implemented in two different ways, similar to the LMS distance measure. In each method an HMM was first developed for the correct reference phoneme. Then, in the first method of implementation, HMMs were also developed for the incorrect reference phonemes. A feature vector from the test phoneme was then created and compared to all of the HMMs using the Baum-Welsh

procedure. The test phoneme was classified as the phoneme corresponding to the HMM that gave the highest probability for producing that test vector. In the second method of implementation the feature vector from the test phoneme was compared only to the HMM for the correct phoneme. The resulting probability for the correct HMM producing the test vector was then supplied as feedback to the user. This helps the user to pronounce his/her sounds closer to the correct phoneme.

Neural Networks

The motivation behind neural networks comes largely from an attempt to model the networks of real neurons in the brain. The brain has many features that are desirable to incorporate into a computation system. To begin with, it is powerful, tolerant, and flexible. Also, it adjusts easily to new conditions by learning. It can handle information that is inconsistent, probabilistic, or noisy. In addition the brain is also highly parallel, small, and compact [Hertz et al., 1991]. Neural networks began to be applied to speech recognition applications in the mid-1980s. However it has proven difficult for neural networks to achieve the same time normalization that HMMs have. For this reason neural networks are often used as static pattern classifiers, often in conjunction with HMMs.

The brain is composed of about 10^{11} neurons. A simplified drawing of a single neuron is shown in Figure 3.6. Neurons transmit information to other neurons through

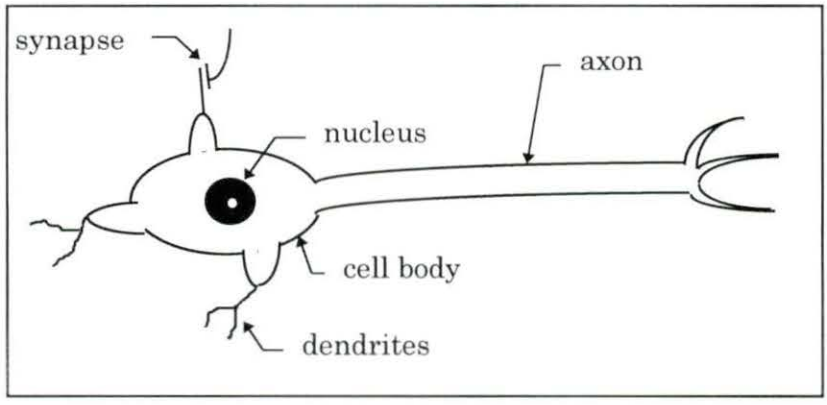


Figure 3.6. Simplified schematic drawing of a typical neuron

their synapses. The initiating neuron releases a specific neurotransmitter which has the effect of raising or lowering the electrical potential of the recipient cell. If the potential reaches a certain threshold, then an action potential, or an electric pulse, of fixed strength and duration is fired down it's axon. This action potential then branches out to synaptic junctions with other cells.

The basic neuron, or node, in a neural network works in a similar fashion. Figure 3.7 shows a basic node in a neural network. The node computes a weighted sum of its inputs from other units and then outputs a *one* or a *zero* depending if the sum is above a certain threshold, or bias. In Figure 3.7 the weights are labeled as *w1*, *w2*, and *w3*. The bias is labeled as *b*. When appropriate weights and biases are found, the neural network can be applied to many applications.

Multi-layer Perceptron

There are many types of neural networks. A picture of a two-layer perceptron is shown in Figure 3.8. Multi-layer perceptrons (MLP) have proven to be effective pattern classifiers. They are able to form complex decision regions in order to classify different sets of features. However, before an MLP is able to perform, it needs to be trained.

Training is accomplished in an MLP by using a procedure known as the back-propagation training algorithm. The training process begins by presenting feature

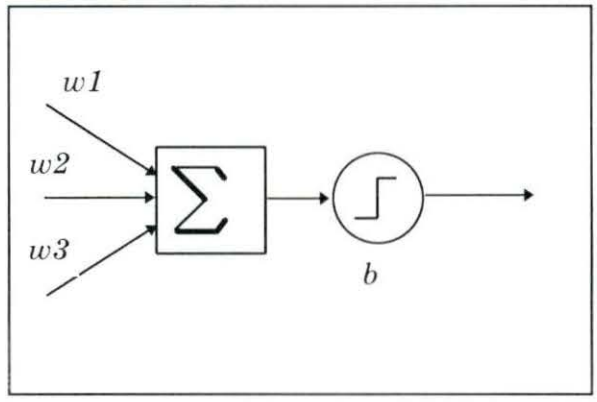


Figure 3.7. Schematic diagram of a single node in a neural network.

vectors as inputs to the network. The network then proceeds to calculate the output based on the initial values for weights and biases at each individual node. The final output is then compared to the desired output for each particular feature vector. Based upon the resulting error, the weights and biases are changed in an effort to help the network produce the correct output. This process is repeated until the error is below an acceptable value. A detailed description of this process can be found in Lippmann [1987].

Two-Layer Perceptron Used in this Project

A two-layer perceptron was used in this project to classify the speech sounds. The network has sixteen nodes in its input layer. This is where the LPC feature vector is presented to the network. The user then has the option to specify the number of hidden nodes. More hidden nodes will result in improved performance but will also cause calculations to be more time consuming. The number of output nodes is also a variable number depending upon how the neural network is being implemented.

There are two different ways that the neural network was implemented. In the first method, the neural network was used to classify the test phoneme as either the correct phoneme or one of the incorrect phonemes. In this case the number of output nodes was equal to the number of incorrect sounds plus one node for the correct sound. Each output node represented one of the sounds. In the second method of

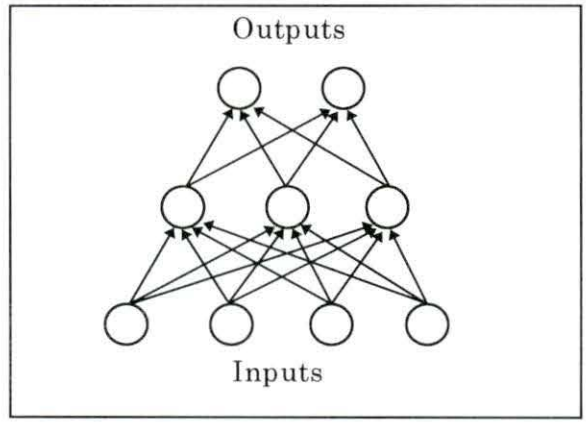


Figure 3.8. A two-layer perceptron.

implementation the neural network was used to let the user know how close the test sound was to the correct phoneme. In this case the number of output nodes was chosen to be ten. With this method the network was trained to produce *ones* at all output nodes when the feature vector of the correct sound was presented to the network. All other sounds would then produce an output a certain distance away from the correct output. This distance was used to provide feedback to the user.

As has been mentioned in this chapter, each of the classification techniques was implemented in two different ways. The reason for both methods of implementation will be discussed in the next chapter. Also, the method for developing and implementing the computer program which uses these classification techniques will be discussed.

CHAPTER 4. DEVELOPMENT AND IMPLEMENTATION OF THE COMPUTER PROGRAM

The purpose of this chapter is to explain how the computer program was developed. Details concerning the program and the strategy employed in developing the code are presented. Finally, this chapter will discuss the manner in which the program can be used for practical applications.

Overview of the Program

The computer program is the interface between the user and the speech recognition techniques embedded in the program; therefore, the program was designed to be easy to learn and simple to use. The visual feedback was designed to be helpful and visually appealing. Many parameters were left available for the user to specify, thus making the program flexible for different situations.

Technical Overview

The program was written using Borland® C++. One advantage of Borland® C++ is it offers many functions which are helpful for programming in the Windows™ environment. An additional software package called *TegoMM.VBX** is used to play, record, and process the audio files. The *TegoMM.VBX* functions are compatible with Borland® C++, enabling all of the programming to be done with Borland® C++ . All of the audio files were saved in *.wav* format which is the standard format used by Windows™. This makes the program portable among computers running Windows™.

System Requirements

In order for the program to operate on a personal computer a few requirements must be met. First of all, the computer must have Windows™ loaded and running. Also, the computer must be equipped with a sound card if the user intends to record new sounds (if the user is planning on using only prerecorded sounds then a sound card is not necessary). It is recommended that the sound card be capable of 16 bit sampling

* TegoMM.VBX is a product of TegoSoft Inc., Box 389, Bellmore, NY, 11710.

resolution and a sampling frequency of 22050 Hz. Most sound resident in modern computers meet these specifications. Another peripheral necessary to record sounds is a microphone. A high quality microphone is a prerequisite for good performance.

Organization of the Program

Previous chapters have detailed the operation of individual parts of the program. The final computer program is a combination of all of the individual parts: data acquisition, signal processing, and classification techniques. Figure 4.1 shows a flowchart of the organization of the program.

Using the Program

This program was designed to help people improve their pronunciation. It is capable of this because errors are often similar among mispronounced sounds. In fact, mispronunciation of a phoneme can be considered to fit into one of three classifications: *substitution*, *distortion*, and *omission*. Substitution occurs when a person substitutes a different phoneme in the place of the correct phoneme. An example of this is when a person pronounces the /sh/ phoneme in place of the /s/ phoneme. A listener might hear the word *she* instead of *see*. Distortion occurs when a person substitutes a different sound, which is not another phoneme, in the place of the correct phoneme. Omission, as the name suggests, is when a phoneme is simply left out of a word.

Modes of Operation

The specific purpose of this program was to deal with the first two cases, *substitution* and *distortion*. Isolated sounds are the only sounds being considered, thus the *omission* case does not apply. The program can be operated in one of two different modes, the *substitution mode* or the *distortion mode*. These two modes provide the basic outline for the program. When operating in the *substitution mode* the user records the correct sound and also a user defined number (up to four) of incorrect sounds. The test sound is then recorded and compared to both the correct and

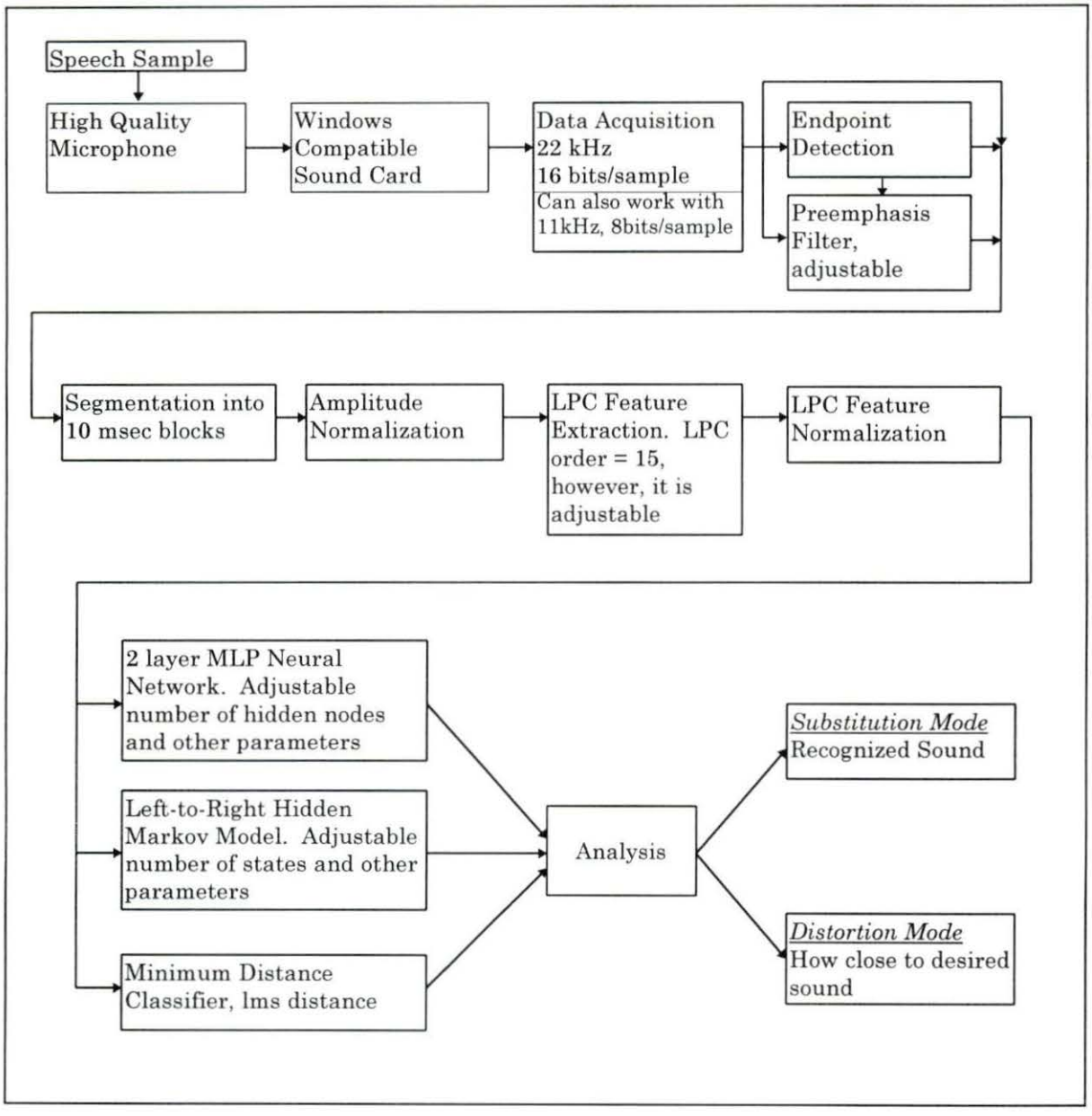


Figure 4.1. Technical flowchart of the program

incorrect sounds using the signal processing and classification techniques described in previous chapters. The resulting match is returned to the user as feedback. When operating in the *distortion mode* only two sounds are recorded and compared; the correct sound and the test sound. Using the signal processing and classification techniques previously described the test sound is compared to the correct sound and a resulting difference score is returned. As the user pronounces the test word closer to the correct sound the difference score becomes smaller. The overall method for operating the computer program in each of the modes is diagrammed in Figure 4.2. In order to better understand how the program is designed to operate in each of the modes it is helpful to know the layout of the program.

Layout of the Computer Program

When the computer program is started it checks to see if there is a sound card in the computer and, if present, it checks the sound card's sampling rate options and its sampling resolution. Once this is completed, the program is ready to begin operation. The initial layout of the program is shown in Figure 4.3.

When using the program for the first time, one can follow the pull-down menus (along the top of the display) from left to right. The menus of the program provide all of the necessary functions to execute the program. For example, the *Graph* menu options (shown in Figure 4.4) provide options to graph any previously recorded sound as a time plot, LPC spectrum plot, or an FFT spectrum plot. A complete description of each menu option and its function is given in the on-line help file included with the program. A text version of this help file is also included in the Appendix.

The *Quick Menu* is a group of buttons which will perform some helpful functions quickly. It is expected that the user will spend much of his/her time working with the test sound and the correct sound. Thus, buttons are provided for the user to quickly record or play back either of these sounds. The pull-down menus also provide options to perform these functions; however, two important functions that are not available through the pull-down menus are the *Use Endpoint Detection* button and the *Use Preemphasis Filter* button. These two choices are optional, and can be used if the user

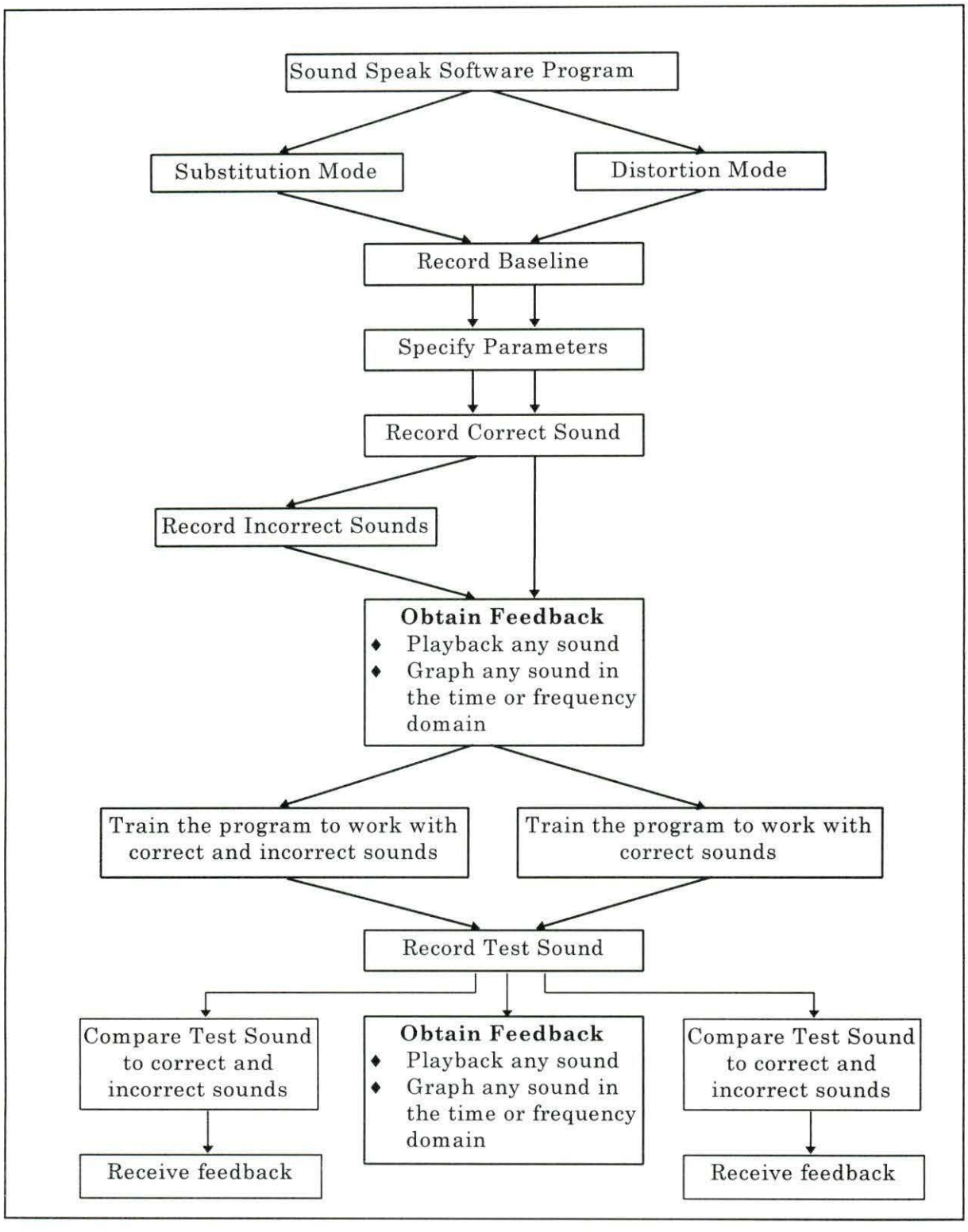


Figure 4.2. Flowchart for using the computer program.

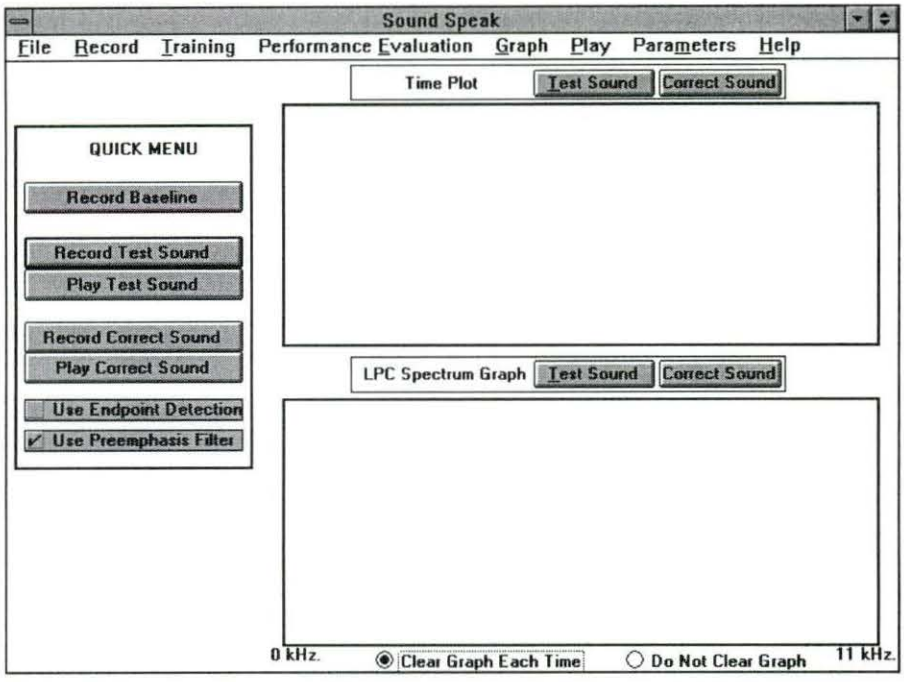


Figure 4.3. Initial layout of computer program.

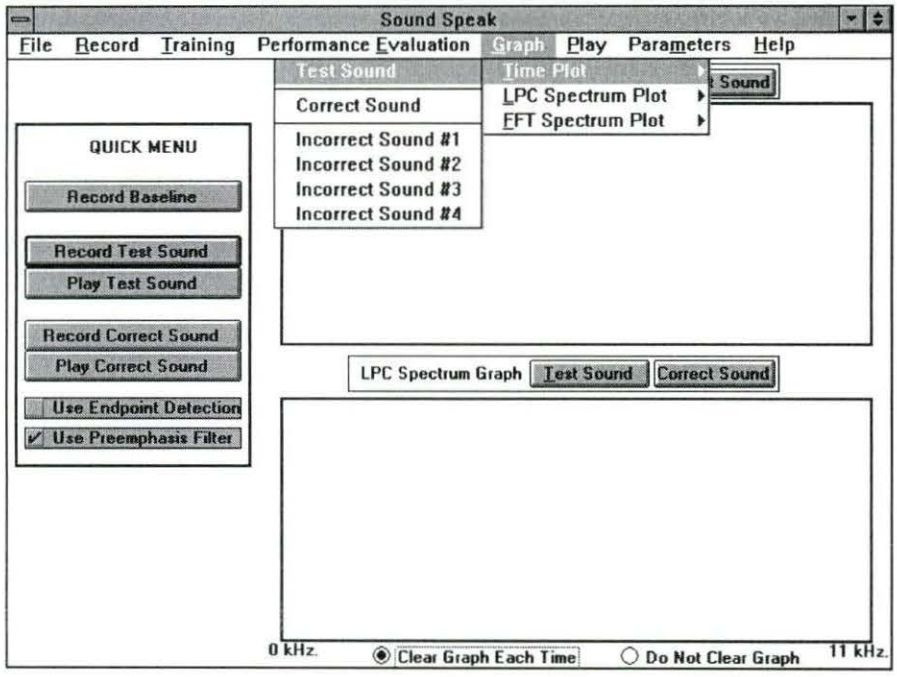


Figure 4.4. Graph menu options.

so desires. Both of these options take additional time to execute, although they may be required in some cases. For example, if the user can not sustain the test sound for the duration of the recording period then endpoint detection is necessary.

Visual and Audio Feedback

Regardless of the mode of operation, visual or audio feedback is always an option. Visual feedback consists of a plot of the signal in the time domain and/or the frequency domain (see Figure 4.5). The frequency domain is particularly helpful in providing feedback. For this reason there is an option in the program to allow the frequency spectrum plot of the correct sound to remain on the screen as a goal for the user. Test recordings can be plotted on the screen and compared to the goal. It is hoped that the user will be able to use this feedback to modify his/her pronunciation in

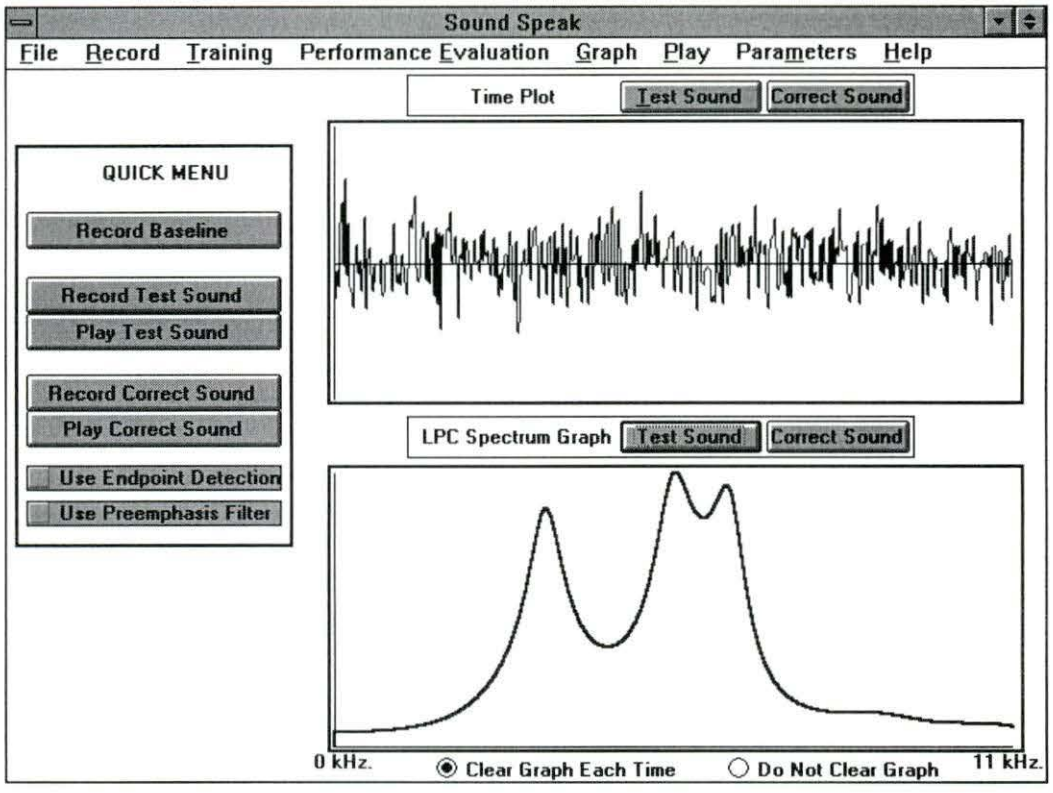


Figure 4.5. Visual feedback of a speech sound.

order to approach the target frequency spectrum. The test frequency spectrum will never exactly match the target frequency spectrum even if the pronounced sounds are correct because of individual variations. It is only used as a tool for qualitative evaluation.

Audio feedback, available as an option from the *Play* menu, allows the user to playback any speech sample being used in the program (see Figure 4.6). This allows the user to listen to the correct sound in an effort to match the sound. Audio feedback, combined with visual feedback will hopefully be beneficial to the user.

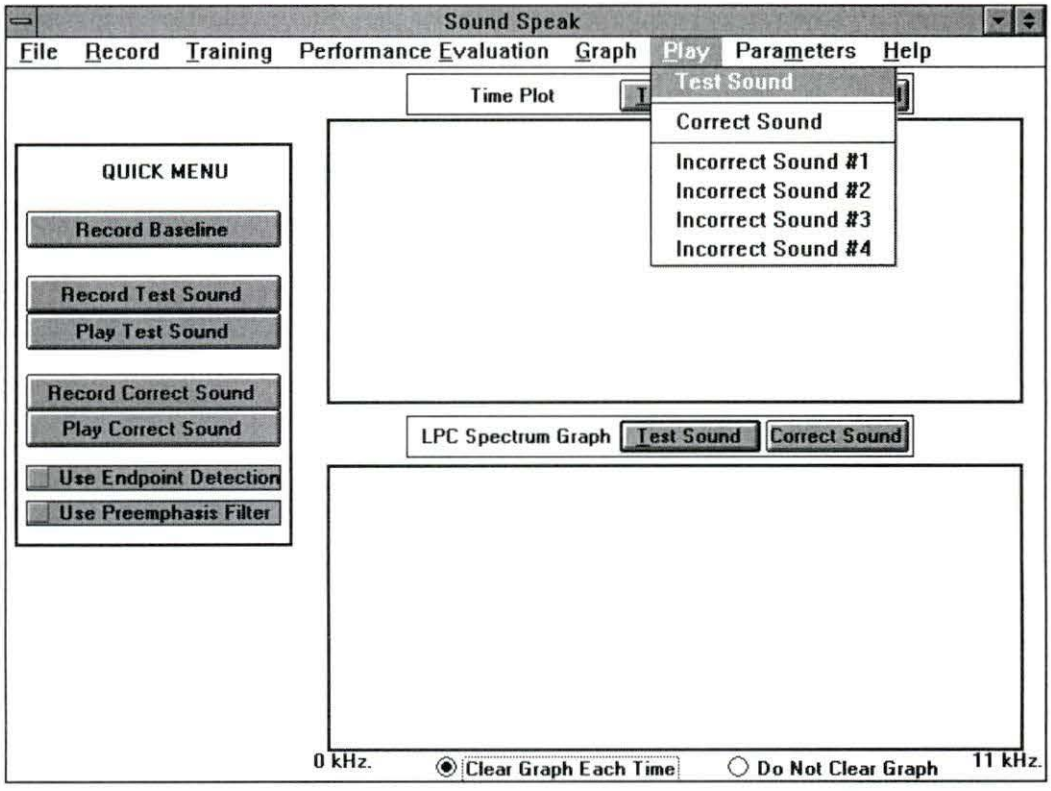


Figure 4.6. Audio feedback option from the menu

User's Manual

An on-line user's manual was developed as part of the software package. It is organized according to the menu options and it gives a description of each menu choice. The text version of this manual is included in the Appendix.

If the computer program is to be an effective teaching tool for speech therapy purposes, then not only must it be aesthetically pleasing, it must also be technically accurate. For example, if the user is trying to correctly pronounce the /s/ sound, but is instead pronouncing the /sh/ sound, it would not be helpful if the feedback from the computer said that the user was pronouncing the /z/ sound. Likewise, if the user were actually pronouncing the /s/ sound correctly and the computer returned with feedback saying that the user was not close to the correct sound then the user would get discouraged. Accurate feedback can be helpful and motivational to the user. The next chapter describes the results of some tests run with this program.

CHAPTER 5. RESULTS AND CONCLUSIONS

In order to assess the technical accuracy of the computer program many tests were run. Initial tests were performed with the program operating in the *substitution mode*. The phonemes /s/, /sh/, /z/, and /zh/ were chosen as the test data. These phonemes were chosen because they are articulated in a similar manner and location within the vocal tract. Additionally, in practical speech situations these phonemes are often substituted for each other erroneously. Following these tests, further testing was performed with the program operating in the *distortion mode*. More information about how these tests were performed is described in this chapter.

Results

Tests Results for the *Substitution Mode*

A total of five different tests were run on the four phonemes mentioned above. In each of these tests one of the phonemes was chosen as the correct sound and other phonemes were compared to it. For four of the tests, only one other phoneme was chosen as the incorrect sound. For the fifth test, all four phonemes were included; one as the correct sound and the other three as incorrect sounds.

For each of these five situations the program was trained using three different classification schemes: minimum distance, neural network, and the hidden Markov model. The purpose of the training was to teach the program to classify the test sound as either the correct sound or an incorrect sound. The effectiveness of each of the classification schemes for correctly classifying test sounds was evaluated on a group of test sounds.

The test sounds used were recordings from six different people, three males and three females. Each person recorded five samples of each of the four phonemes, for a total of twenty recordings. This resulted in a total of thirty recordings of each phoneme. After listening to all of the recordings, it was determined that, for various reasons, some of the phonemes recorded were incorrect and were therefore excluded. Thus, the overall total of recordings of each phoneme was reduced to twenty-five.

Using this test data, training tests were run using each type of classification scheme. With each test, parameters were adjusted in an effort to achieve better classification results. Although it was impossible to try every single combination of parameters, the training tests were considered complete when the performance of the classification scheme did not appear to improve with further adjustments of the parameters. Tables 5.1 - 5.5 show the results of the five main tests that were performed. An explanation of the parameters listed in these Tables is contained in the on-line help available in the Appendix.

Discussion of Test Results for the *Substitution Mode*

As can be seen from the results, all three classification schemes work fairly well when the test phoneme was compared to only two phonemes. However, when the test phoneme was compared to four phonemes the overall recognition performance decreased slightly. In the test using all four phonemes, the hidden Markov model (HMM) approach proved to be most reliable although there was room for improvement. Recall that the HMM used for this computer program was a left-to-right one shown in Figure 3.3. By modifying the HMM to allow *jumps* of states [Rabiner and Juang, 1986], as shown in Figure 5.1, more flexibility would be incorporated into the HMM and, most likely, better recognition accuracy would result.

Further understanding of the differences in the results of the three classification schemes can be gained by examining how each is designed to perform. The minimum distance method is similar to the HMM method because for each new sound to be

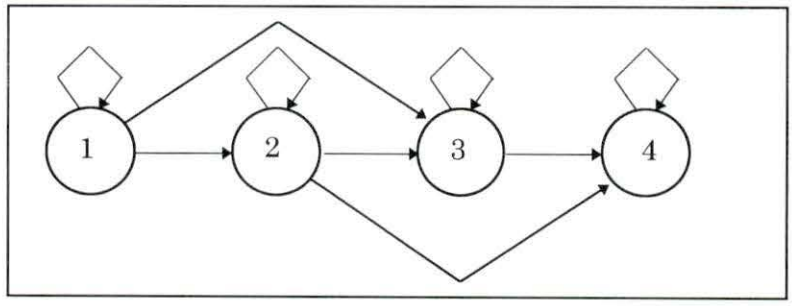


Figure 5.1. Modified left-to-right hidden Markov model.

recognized it develops a new template, or in the case of the HMM a new model. However, the neural network method uses only one network, no matter how many different sounds are being compared. In order to increase the capacity of the network to differentiate between more sounds it adds additional nodes to the output layer of the network. Thus, the training phase of the neural network becomes more complicated.

One problem encountered in this project with the neural network was that the error level could not be reduced to an acceptable level. The first attempt to solve this problem was to increase the number of nodes in the hidden layer of the neural network. By so doing, it was hoped to produce a decision boundary which would allow the neural network to accurately classify all four phonemes correctly. Despite attempts with many different hidden node numbers, the neural network was still not able to train to an acceptable error level, and consequently it could not consistently classify the phonemes correctly. To rectify this situation only two sets of training data were included, as can be seen in Table 5.5. This enabled the neural network to train to an acceptable error goal. However, by using only two training sets, an accurate representation of each phoneme was not developed, resulting in lower recognition rates than those obtained with the other two classification schemes.

Tests Results for the *Distortion Mode*

The purpose of operating the computer program in the *distortion mode* is to provide accurate feedback telling the user how close he/she is to the correct sound. The following test process was followed to see if the program was achieving this goal. The phoneme /s/ was chosen as the correct sound. An acoustically and articulately correct/s/ sound from a speech pathologist was first recorded. Next, five distortions of the /s/ sound were recorded. These five different distortions were ranked by the speech pathologist according to their perceptual level of closeness to the correct /s/ sound. Then, these same five distortions were introduced to the program as the test sound. Each of the three classification techniques was used on each of the test sounds in order to determine how close to the correct sound it was. The results of these tests are shown in Table 5.6.

Table 5.1. Classification results for the phonemes /s/ and /sh/.

Classification Results for Two Phonemes: /s/ and /sh/					
Classification Scheme	Model Parameters		Recognition Accuracy		
			/s/	/sh/	Overall
Minimum Distance	training sets:	30	96.0%	100.0%	98.0%
Hidden Markov Model	training sets: states: iterations: error goal: preemphasis filter:	30 15 50 .001 off	100.0%	100.0%	100.0%
Neural Network	training sets: hidden nodes: iterations: error goal: preemphasis filter:	10 15 500 .001 off	100.0%	100.0%	100.0%

Table 5.2. Classification results for the phonemes /s/ and /z/.

Classification Results for Two Phonemes: /s/ and /z/					
Classification Scheme	Model Parameters		Recognition Accuracy		
			/s/	/z/	Overall
Minimum Distance	training sets:	30	96.0%	92.0%	94.0%
Hidden Markov Model	training sets: states: iterations: error goal: preemphasis filter:	50 15 300 .001 off	88.0%	100.0%	94.0%
Neural Network	training sets: hidden nodes: iterations: error goal: preemphasis filter:	10 15 500 .001 off	100.0%	96.0%	98.0%

Table 5.3. Classification results for the phonemes /z/ and /zh/.

Classification Results for Two Phonemes: /z/ and /zh/					
Classification Scheme	Model Parameters		Recognition Accuracy		
			/z/	/zh/	Overall
Minimum Distance	training sets:	50	92.0%	72.0%	82.0%
Hidden Markov Model	training sets: states: iterations: error goal: preemphasis filter:	50 15 300 .001 off	84.0%	100.0%	92.0%
Neural Network	training sets: hidden nodes: iterations: error goal: preemphasis filter:	10 15 500 .001 off	96.0%	80.0%	88.0%

Table 5.4. Classification results for the phonemes /sh/ and /zh/.

Classification Results for Two Phonemes: /sh/ and /zh/					
Classification Scheme	Model Parameters		Recognition Accuracy		
			/sh/	/zh/	Overall
Minimum Distance	training sets:	50	64.0%	92.0%	76.0%
Hidden Markov Model	training sets: states: iterations: error goal: preemphasis filter:	50 15 300 .001 off	84.0%	96.0%	90.0%
Neural Network	training sets: hidden nodes: iterations: error goal: preemphasis filter:	10 15 500 .001 off	100.0%	100.0%	100.0%

Table 5.5. Classification results for the phonemes /s/, /sh/, /z/, and /zh/.

Classification Results for Four Phonemes: /s/, /sh/, /z/, and /zh/							
Classification Scheme	Model Parameters		Recognition Accuracy in Percentages (%)				
			/s/	/sh/	/z/	/zh/	Overall
Minimum Distance	training sets:	50	84.0	92.0	88.0	52.0	79.0
Hidden Markov Model	training sets: states: iterations: error goal: pre-filter:	50 15 300 .001 off	96.0	80.0	88.0	100	91.0
Neural Network	training sets: hidden nodes: iterations: error goal: pre-filter:	2 20 5000 .001 off	88.0	16.0	88.0	40.0	59.0

Table 5.6. Test results for operation in the *distortion mode.*

Level of Closeness to the /s/ Phoneme. 1=closest, 5=farthest				
Type of Distortion	Method of Classification			
	Speech Pathologist	Minimum Distance	Hidden Markov Model	Neural Network
high frequency	1	2	2	1
general distortion	2	3	3	4
cleft distortion	3	5	5	3
lateral lisp	4	4	4	5
interdental lisp	5	1	1	2

Discussion of Test Results for the *Distortion Mode*

The results from the classification techniques were not totally consistent with those of the speech pathologist. However, there was some consistency between the different techniques, especially between the minimum distance method and the HMM. In each of the three methods of classification the same two distortions were ranked as the two closest sounds to the correct /s/ phoneme. It was interesting to note that the *interdental lisp* mispronunciation of the /s/ phoneme was ranked as the farthest away, perceptually, by the speech pathologist but one of the two closest by the other techniques. By examining the LPC frequency plot of these two sounds (Figure 5.2) one can see why this would occur.

LPC frequency plots of the other distortions compared to the correct /s/ sound are shown in subsequent plots (Figures 5.3 - 5.6). Visual examination of these plots lends support to the level of closeness ranking by the classification techniques as shown in Table 5.6. This would indicate that the classification techniques may have correctly ranked the distortions on the basis of the LPC coefficients. However, the perceptual ranking of the distortions is still different. This indicates that LPC coefficients, by themselves, are not sufficient to correctly rank the distortions of the /s/ phoneme. Perhaps by using another feature, or a combination of features, the computer would be able to rank the distortions in accordance with their perceptual rankings.

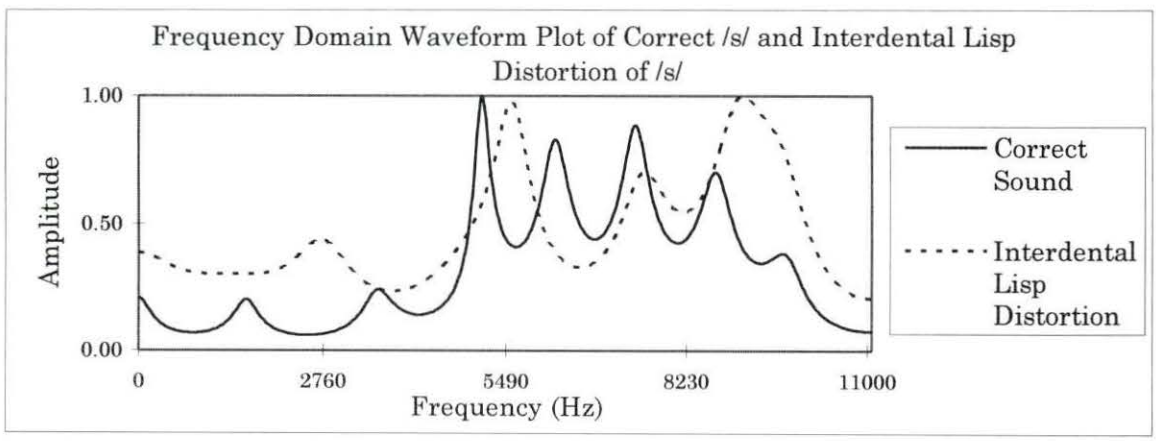


Figure 5.2. LPC frequency plot of correct /s/ and interdental distortion of /s/.

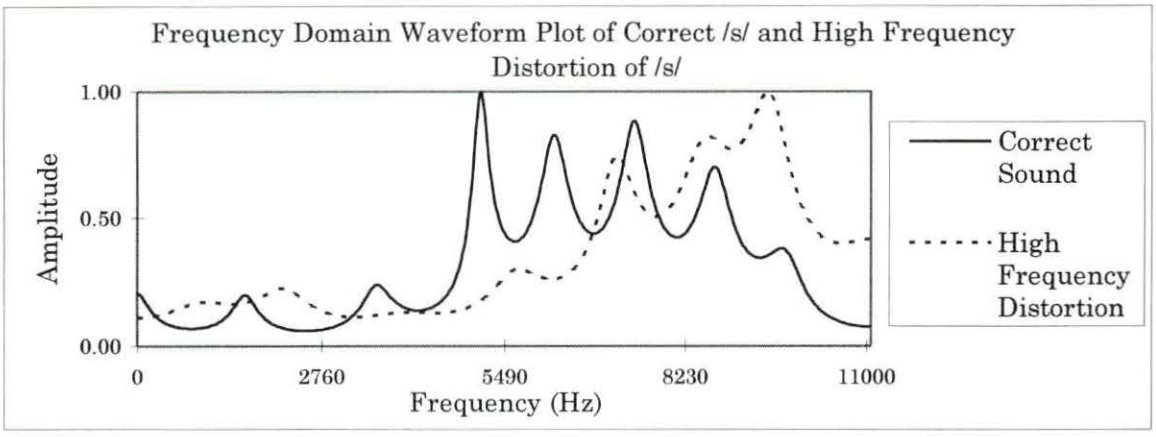


Figure 5.3. LPC frequency plot of correct /s/ and high frequency pronunciation of /s/

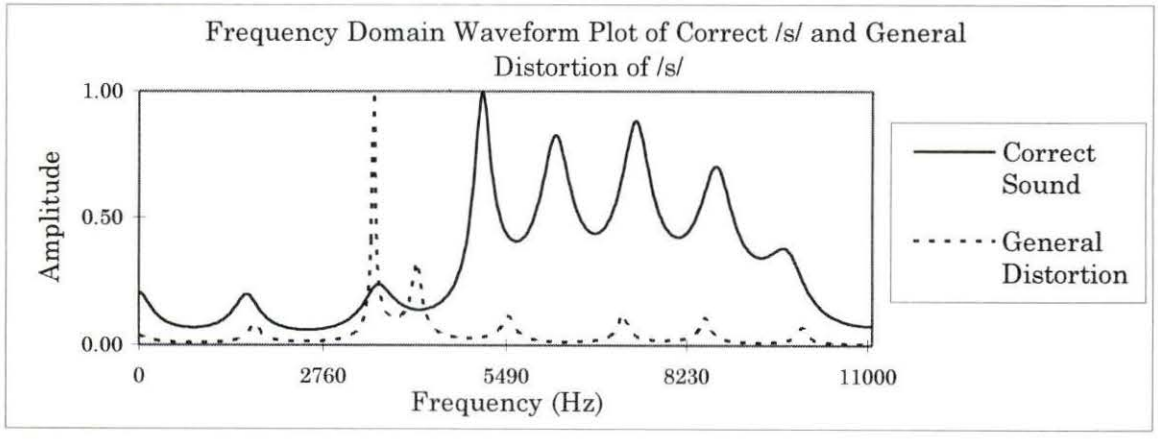


Figure 5.4. LPC frequency plot of correct /s/ and larger distortion of /s/

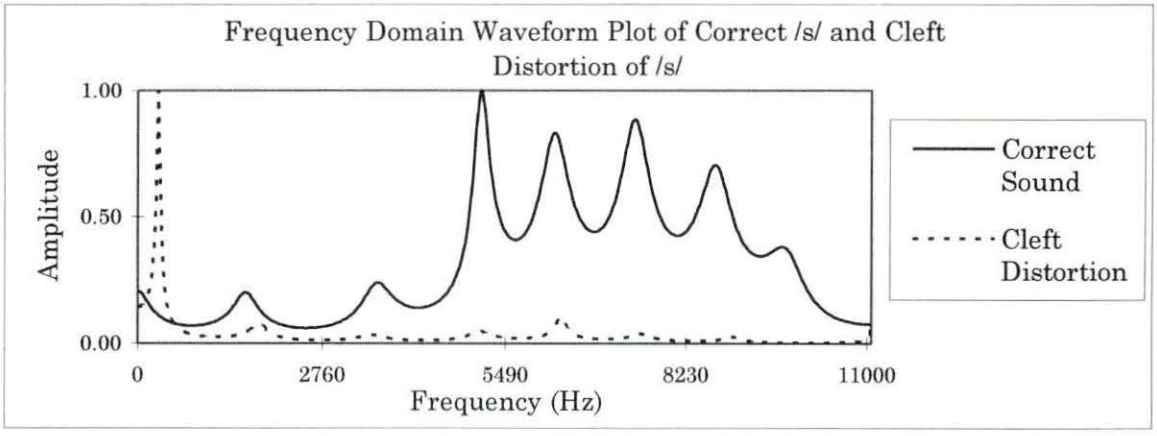


Figure 5.5. LPC frequency plot of correct /s/ and cleft distortion of /s/

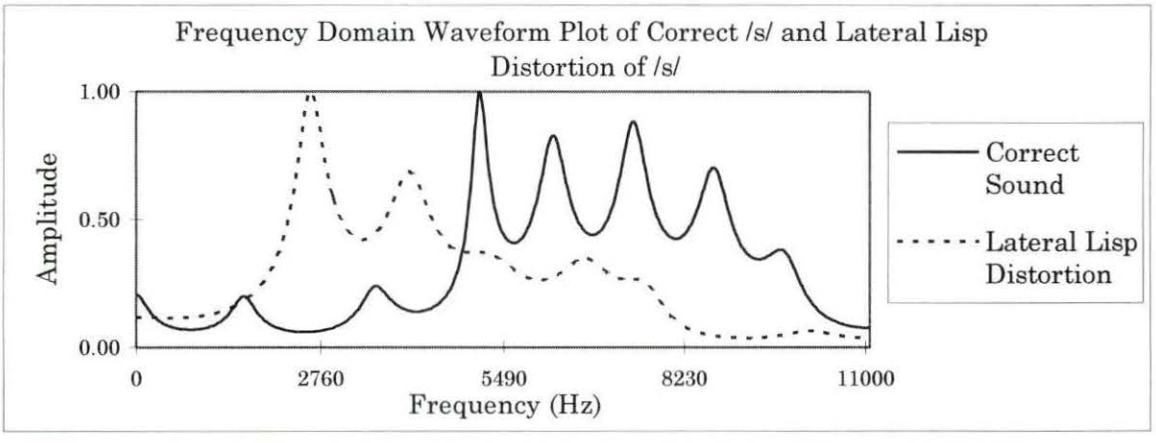


Figure 5.6. LPC frequency plot of correct /s/ and lateral lisp distortion of /s/

Conclusions and Future Work

The objective of this project was to develop a computer program which can be of assistance in certain aspects of speech therapy. The computer program proved to be successful in providing visual and audio feedback to the user. It also demonstrated that it could provide appropriate feedback when operated in the *substitution mode*. The program was most accurate when operating in this mode with two different phonemes utilizing the neural network as the classification scheme. The computer program also showed consistency. However, it was not always accurate while functioning in the *distortion mode*.

In order to improve the performance of the program, work could proceed in a few directions. By incorporating additional features more information about each speech signal could be utilized in the classification scheme. This can potentially lead to improved performance. Also, more elaborate classification schemes involving the use of such tools as the time-delay neural network or more sophisticated hidden Markov models could be used. Finally, in order to be a more effective aid in speech therapy, work needs to be done with regard to providing feedback on the pronunciation of phonemes in syllables, words, and even sentences.

LIST OF REFERENCES

Barnard, E., Cole, R.A., Vea, M.P, and Alleva, F.A., "Pitch Detection with a Neural-Net Classifier," *IEEE Transactions on Signal Processing*, Vol. 39, No. 2, pp. 298-306, February, 1991.

Elenius, K.O., and Traven, H.G.C., "Multi-Layer Perceptrons and Probabilistic Neural Networks for Phoneme Recognition," *Speech Transmission Laboratory Quarterly Progress and Status Report*, Royal Institute of Technology, Stockholm, Sweden, pp. 1-6, October 15, 1993.

Ganong, W.F., "Review of Medical Physiology," Appleton & Lange, East Norwalk, Connecticut, 1991.

Halliday, D., and Resnick, R., "Fundamentals of Physics, Second Edition," John Wiley and Sons, New York, 1986.

Hertz, J., Krogh, A., and Palmer, R.G., "Introduction to the Theory of Neural Computation," Addison-Wesley Publishing Company, New York, 1991.

Hutchins, S.E., "SAY & SEE: Articulation Therapy Software," *IEEE*, pp. 37-40, January, 1992.

Kent, R.D., and Read, C., "The Acoustic Analysis of Speech," Singular Publishing Group, Inc., San Diego, CA, 1992.

Kohonen, T., "The Neural Phonetic Typewriter," *Computer*, Vol. 21, pp. 11-22, March, 1988.

Lippmann, R.P., "An Introduction to Computing with Neural Nets," *IEEE ASSP Magazine*, pp. 4-22, April, 1987.

Levinson, S.E., and Roe, D.B., "A Perspective on Speech Recognition," *IEEE Communications Magazine*, Vol. 28, p. 28-34, January, 1990.

Martini, F., "Fundamentals of Anatomy and Physiology, Second Edition," Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1992.

O'Shaughnessy, D., "Speech Communication, Human and Machine," Addison-Wesley Publishing Company, New York, 1987.

Picone, J.W., "Signal Modeling Techniques in Speech Recognition," *Proceedings of the IEEE*, Vol. 81, No. 9, pp. 1214-1247, September, 1993.

Press, W.H., "Numerical recipes in C : the art of scientific computing," Cambridge University Press, New York, 1988.

Rabiner, L.R., "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE*, Vol. 77, No. 2, pp. 257-286, February, 1989.

Rabiner, L.R., and Juang, B.H., "An Introduction to Hidden Markov Models," *IEEE ASSP Magazine*, pp. 4-16, January, 1986.

Rabiner, L.R., and Juang, B.H., "Fundamentals of Speech Recognition," Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1993.

Ramabadran, T.V., and Venkatagiri, H.S., "Development of a Speech Analysis Software Package for Clinical Application," Preproposal for an Exploratory Research Project, Iowa State University, Ames, IA, 1993.

Roe, D.B., and Wilpon, J.G., "Whither Speech Recognition: The Next 25 Years," *IEEE Communications Magazine*, Vol. 31, No. 11, p. 54-62, November, 1993.

Schmidt, B.L., "Comparison of feature vectors for speech recognition using the time delay neural network," Thesis for Masters of Science, Iowa State University, Ames, 1993.

Skills International, "Perfect English Pronunciation," brochure by Skills International Co., 2849 West Dundee Road, Northbrook, Illinois, 60062, 1995.

Sweeney, L., "Talking to Machines," EDN Products Edition, pp. 15-17, February 13, 1995.

Waibel, A., and Hampshire, J., "Building Blocks for Speech," *Byte*, pp. 235-242, August, 1989.

Wang, K.S., and Shamma, S.A., "Auditory Analysis of Spectro-Temporal Information in Acoustic Signals," *IEEE Engineering in Medicine and Biology Magazine*, Vol. 14, No. 2, p. 186-194, March-April, 1995.

APPENDIX

The appendix contains the text version of the on-line help available for the computer program. The code of the computer program and the executable program is available on disk in the Electrical and Computer Engineering Department.

The text version of the on-line help was written in *rich text format (rtf)* and compiled into a Windows™ help file using the Borland® help file compiler. Each new page in the text file represents a new help topic.

Contents

$ K + Brief Overview

This program was written by Richard M. Johnson as part of his thesis project for EcEn/BME at Iowa State University. It was completed under the direction of Drs. Swift, Ramabadran, and Venkatagiri. May, 1995.

This program is designed to be used as a tool for speech therapy. It is designed to work with isolated sounds, such as /s/, /sh/, /z/, /zh/, etc. It's goal it to provide visual and audio feedback to the user in order to help him/her improve his/her pronunciation of these isolated sounds.

---

brief_overview
$ Brief Overview
K overview of program, summary of program
+ 00

$ K + How to Use this Program

This program was designed to help people improve their pronunciation. It is capable of this because errors are often similar among mispronounced sounds. In fact, mispronunciation of a phoneme can be considered to fit into one of three classifications: *substitution*, *distortion*, and *omission*. Substitution occurs when a person substitutes a different phoneme in the place of the correct phoneme. An example of this is when a person pronounces the /sh/ phoneme in place of the /s/ phoneme. A listener might hear the word *she* instead of *see*. Distortion occurs when a person substitutes a different sound, which is not another phoneme, in the place of the correct phoneme. Omission, as the name suggests, is when a phoneme is simply left out of a word.

Modes of Operation

The specific purpose of this program was to deal with the first two cases, *substitution* and *distortion*. Isolated sounds are the only sounds being considered, thus the *omission* case does not apply. The program can be operated in one of two different modes, the *substitution mode* or the *distortion mode*. These two modes provide the basic outline for the program. When operating in the *substitution mode* the user records the correct sound and also a user defined number (up to four) of incorrect sounds. The test sound is then recorded and compared to both the correct and incorrect sounds using the signal processing and classification techniques described in previous chapters. The resulting match is returned to the user as feedback. When operating in the *distortion mode* only two sounds are recorded and compared; the correct sound and the test sound. Using the signal processing and classification techniques previously described the test sound is compared to the correct sound and a resulting difference score is returned. As the user pronounces the test word closer to the correct sound the difference score becomes smaller. The technical flowchart of operation for the program is shown in Figure 1. The overall method for operating the computer program in each of the modes is diagrammed in Figure 2. In order to better understand how the program is designed to operate in each of the modes it is helpful to know the layout of the program.

---

using_program

$ How to Use this Program

K using program, getting started, beginning program

+ 01

$ K Figure 1. Technical Flowchart of the Program

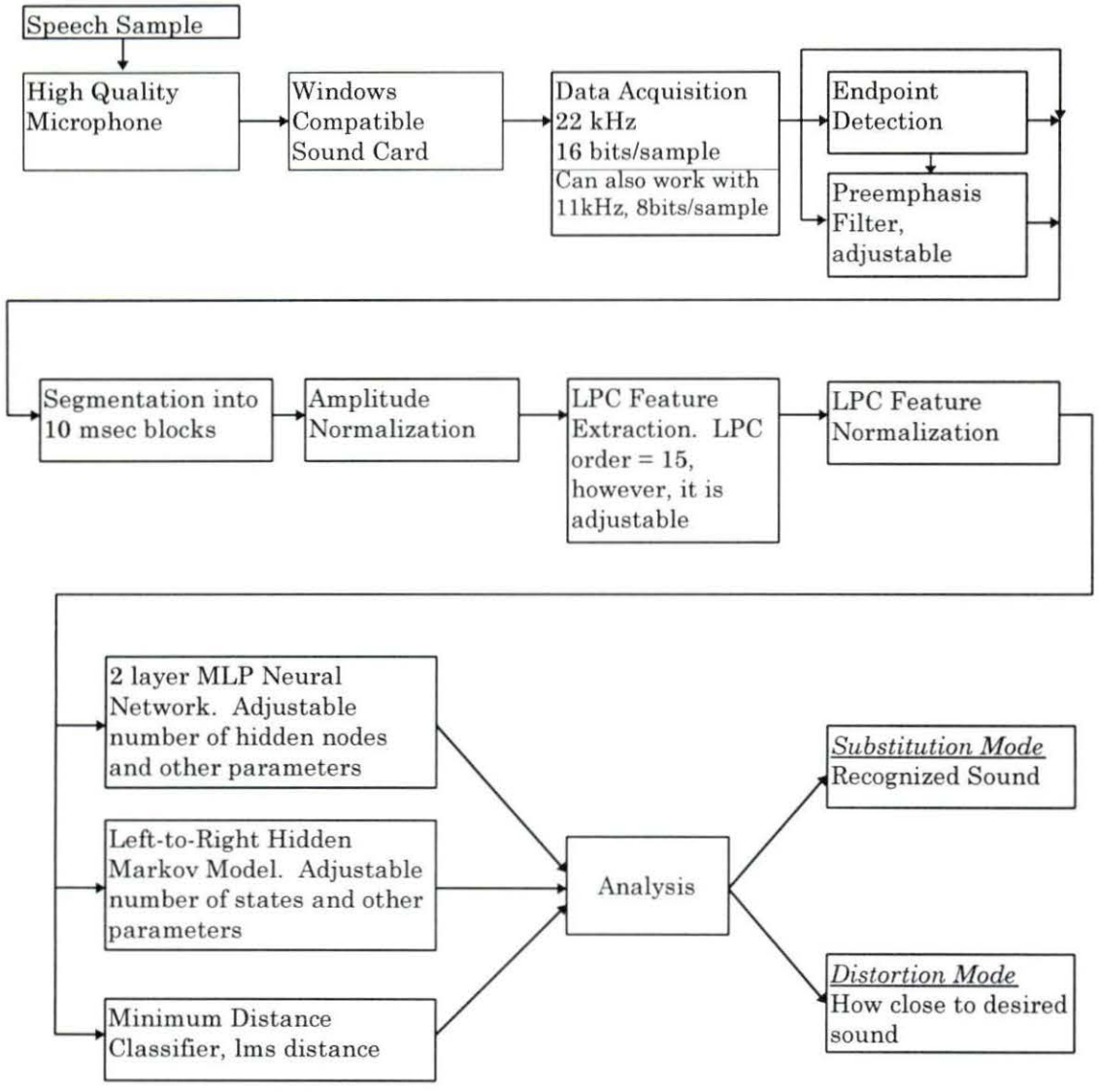
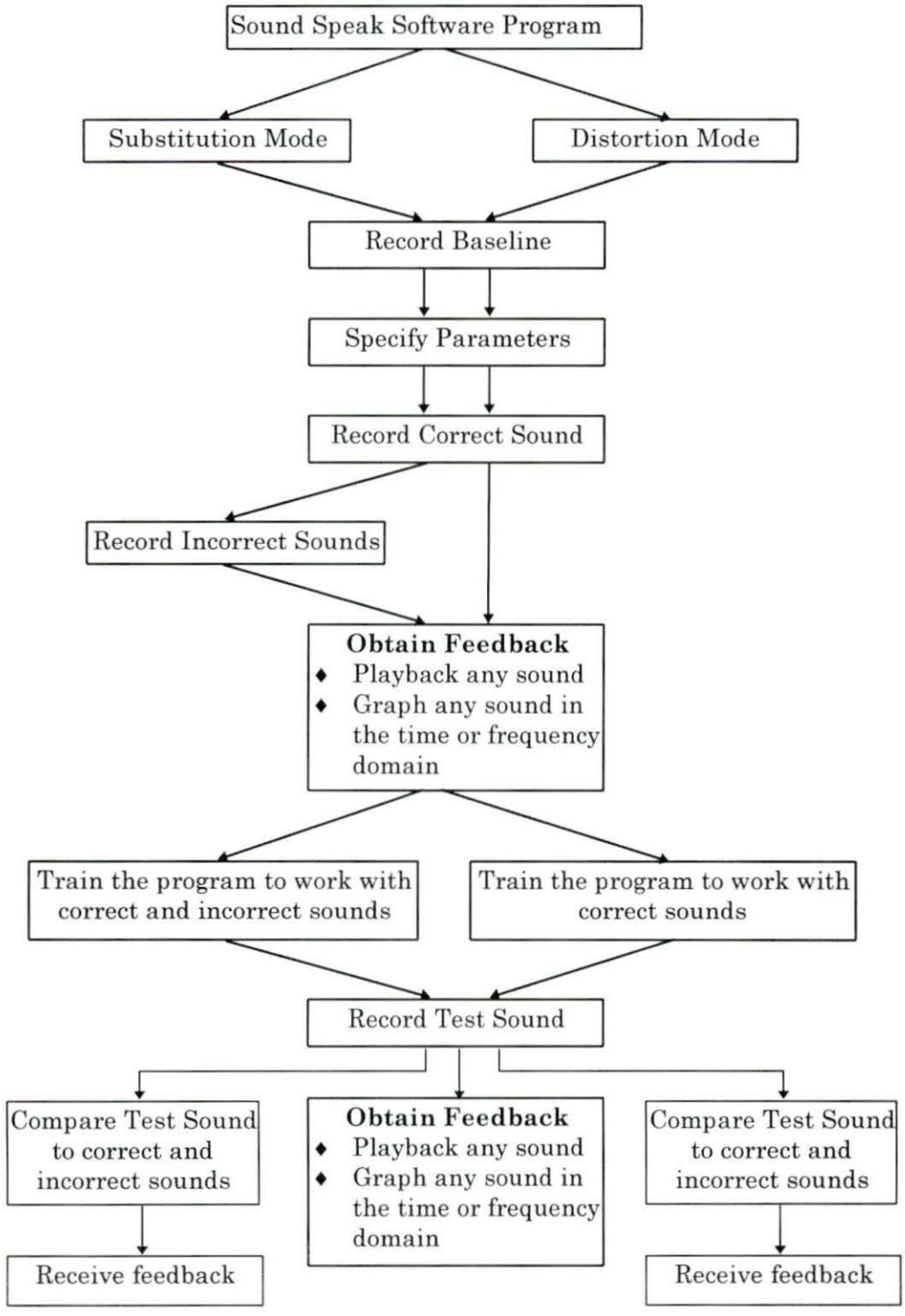


fig_1
$ Figure 1. Technical flowchart of the program
K flowchart, technical information

 # $ K Figure 2. Flowchart for using the computer program.

fig_2
$ Figure 2. Flowchart for using the computer program
K flowchart, using the program


Sound Speak Software Program
Substitution Mode
Distortion Mode
Record Baseline
Specify Parameters
Record Correct Sound
Record Incorrect Sounds
Obtain Feedback
Playback any sound
Graph any sound in the time or frequency domain
Train the program to work with correct and incorrect sounds
Train the program to work with correct sounds
Record Test Sound
Compare Test Sound to correct and incorrect sounds
Receive feedback
Obtain Feedback
Playback any sound
Graph any sound in the time or frequency domain
Compare Test Sound to correct and incorrect sounds
Receive feedback

$ K + File Menu

Presently the File menu only performs one useful function. It will let you exit the program by clicking on file/exit.

---

file_menu
$ File Menu
K opening files; closing files; files; exit
+ 02

$ K + Record Menu

Baseline

The first step in using *Sound Speak* is to record the baseline. The baseline is a recording of the ambient conditions in the general vicinity of the microphone. When recording the baseline try to be silent.

Record Test Sound

Use this menu selection to record a test sound. This test sound can then be compared to a *correct sound* or to a number (1-4) of *incorrect sounds*.

Load Test Sound

Instead of recording a test sound, this option is available to load a previously recorded test sound. It must be stored either on a floppy disk or on the hard disk in *WAV* format. When the dialog box prompts you for a filename please type out the complete path and filename.

Record Correct Sound

Use this menu selection to record a correct sound. The therapist will usually be the one to record this sound. Before recording the correct sound a dialog box is presented to allow text to be associated with the correct sound for identification purposes only. The user will then record *test sounds* in an effort to match the correct sound.

Load Correct Sound

Similar to *Load Test Sound*, this option is available to load any WAV file from disk as the correct sound.

Record Incorrect Sounds

Similar to *Record Correct Sounds*, this option will consecutively record all of the incorrect sounds.

Load Incorrect Sounds

Similar to the other *Load* options, selecting this menu choice will allow the user to load any *WAV* file from disk as an incorrect sound.

---

record_menu
$ Record Menu
K recording sounds; loading sounds
+ 03

$ K + Training Menu

This program is set up to work in two different modes. *Distortion Mode* is used when the subject distorts the correct sound. The resulting distortion is not necessarily a identifiable phoneme, but a distortion of the correct phoneme. If the distortion mode is desired you need to train the program *to work with correct sound only*. *Substitution Mode* is used when the subject substitutes incorrect sounds in place of the correct sound. If the substitution mode is desired you need to train the program *to work with correct and incorrect sounds*.

To Work with Correct Sound Only

Use this option if you are working on a distortion of a specific sound. There are three different methods to choose from:

Minimum Distance Method

This is the simplest and fastest of the three methods.

Hidden Markov Model

This method is more complex than the Minimum Distance Method. Be sure to specify the parameters that are desired in the Parameters Menu.

Neural Network

This method is also complex. Specify desired parameters in the Parameters Menu.

To Work with Correct and Incorrect Sounds

Use this option if you are working on substitutions of a certain sound. The three methods described above also apply to this option.

---

training_menu
$ Training Menu
K training; hidden Markov model (HMM); Neural Networks
+ 04

$ K + Performance Evaluation Menu

Before you use this menu you first need to train (see *Training Menu*) the program. If using the program in *distortion mode* then choose the menu selection *Compare test sound to correct sound only*. If using the program in *substitution mode* then choose the menu selection *Compare test sound to correct and incorrect sounds*.

Compare Test Sound to Correct Sound Only

Use this option if you are working in the distortion mode. Be sure that you have trained the program to work with the method that you plan on using to do the comparison. The program will return a *difference from the correct sound score*. This *score* will be different value for each method doing the comparison. However, each method should give the user a quantitative evaluation of how close the test sound was to the correct sound.

Compare Test Sound to Correct and Incorrect Sounds

Use this option if you are working in the substitution mode. Be sure that you have trained the program to work with the method that you plan on using to do the comparison. The program will tell the user which sound the test sound was closest to. If the user has input text associated with the correct and incorrect sounds, the program will display the text associated with that sound. If there was no text entered to be associated with the sounds, then the default text is as follows: *cs* corresponds to the correct sound, *inc1, inc2, inc3, inc4,* correspond to incorrect sounds 1-4, respectively.

Batch Comparison of Test Sound to All Sounds

This is another option to use if you are working in the substitution mode. This option is used if you have a number of previously recorded sounds saved on disk. In order to tell the program which sounds to compare they all need to be in the same directory. Also in that directory you need to create a text file listing the name of each sound file on a separate line. In addition you need to include a hard return at the end of the last filename in the text file. The program will prompt you for the name of this text file and then return the results in a new file.

---

performance_evaluation_menu
$ Performance Evaluation Menu
K evaluation; performance; recognize; compare;
+ 05

$ K + Graph Menu

This menu selection provides options to graph any previously recorded or loaded sound (see Record Menu). The user has the option to graph the sound in the time domain or in the frequency domain. Frequency domain plots are available by plotting the LPC (linear predictive coding) spectrum or the FFT (fast Fourier transform) spectrum.

Time Plot

This option provides a plot of time versus amplitude (of a speech sample). It can be helpful to make sure that a sample does exist, or that the endpoint detection algorithm performed correctly. It can also provide some simple visual feedback.

LPC Spectrum Plot

This option provides a plot of frequency versus normalized amplitude. Frequency content of a speech sample is often a distinguishing feature of the sound produced. Thus, this plot provides visual feedback to the user. Notice the option below the graph "Do Not Clear Graph" or "Clear Graph Each Time". In order to provide useful feedback it is possible to plot the correct sound and then click on "Do Not Clear Graph" and then plot the test sound. The user can then directly compare the test speech sample to the correct speech sample in the frequency domain.

The LPC Spectrum Plot results from the LPC coefficients derived from the particular speech sample. These LPC coefficients are also used in the training and performance evaluation phase of the program.

FFT Spectrum Plot

This option also provides a plot of frequency versus normalized amplitude. For more information about spectrum plots see the listing under LPC Spectrum Plot.

---

graph_menu
$ Graph Menu
K plot; graph; frequency; spectrum; linear predictive coding (LPC); LPC spectrum; Fast Fourier Transform (FFT);
+ 06

$ K + Play Menu

This menu selection allows the user to play back any of the sounds that have been previously recorded or loaded (see Record Menu). Simply single click on the sound that you want to hear and the computer will play the sound through the sound card and speakers.

Test Sound

Use this option to play back the test sound.

Correct Sound

Use this option to play back the correct sound.

Incorrect Sound 1

Use this option to play back the first incorrect sound, if it exists.

Incorrect Sound 2

Use this option to play back the second incorrect sound, if it exists.

Incorrect Sound 3

Use this option to play back the third incorrect sound, if it exists.

Incorrect Sound 4

Use this option to play back the fourth incorrect sound, if it exists.

---

play_menu
$ Play Menu
K play; audio; listen
+ 07

$ K + Parameters Menu

This menu selection allows the user to fine-tune the program to a desired specification by adjusting the parameters of the program.

Hidden Markov Model

This option gives the user the options to specify certain parameters relating to the hidden Markov model (HMM) method of classifying the speech samples.

Number of States

Use this option to specify how many states in the HMM. The default value is 10. A value close to the LPC order generally provided good results.

Maximum Iterations

Use this option to specify the maximum number of training iterations. If the minimum error goal has not been reached by this time, then the HMM will discontinue training. Otherwise, the HMM may continue training for a very long time.

Minimum Error Goal

Use this option to specify the minimum error goal desired. After training, iteration the HMM calculates the error. When the error is below the specified goal the training is considered complete. The HMM trains itself by changing its various probabilities after each iteration. The error is a function of how much each probability changes. When the probabilities do not change much after an iteration, then the probability will also be small.

Neural Network

Number of Hidden Nodes

Use this option to specify how many hidden nodes in the Neural Network. The default value is 10. Generally, the more number of hidden nodes, the greater the accuracy of the network.

Maximum Iterations

Use this option to specify the maximum number of training iterations. If the minimum error goal has not been reached by this time, then the Neural Network will discontinue training. Otherwise, the Neural Network may continue training for a very long time.

Minimum Error Goal

Use this option to specify the minimum error goal desired. After training iteration the Neural Network calculates the error. When the error is below the specified goal the training is considered complete. The error in a the neural network is a function of the desired output compared to the actual output. When the neural network is trained correctly, its output should be very close to the desired output, resulting in a small error value.

Overall Performance

Several parameters, which affect all classification methods, are adjustable with this menu option.

---

parameters_menu

$ Parameters Menu

K training sets; options; LPC order; parameters; number of hidden nodes; number of states; minimum error goal; number of training epochs; precision

+ 08

LPC Order

Use this option to specify the desired LPC order to use on the speech samples. The default value is 15. Larger values often give a more accurate representation of the speech sample, but also take more time to process.

Sets of Training Data

This option is used to specify the number of training sets that are extracted from each speech sample. Generally, the more sets available to be trained, the higher the accuracy. However, more sets also take more time.

Each set of training data is extracted from the same speech sample. For example, assume that there is one second of a sound recorded. Each set of training data is a rectangularly windowed segment 10 milliseconds in length. Therefore, for a one second recording there are 100 possible sets of training data. However, the program is designed to take the first set of training data after 250 milliseconds of the speech sample have already elapsed. This is to ensure the sample is a true representation of the sound, and not an initial fluctuation of the sound which may be present at the beginning of a sample. Therefore, a safe number of training sets to use is often around 50.

Filter Value

This value is only used if the "Use Preemphasis Filter" checkbox is checked in the "Quick Menu" of the program. It is advisable to use a preemphasis filter at all times, especially when the quality of the microphone is low. The effect of the filter is to attenuate the lower frequency components in a signal. A common value for the filter value is between .9-1.0

Enter Number of Incorrect Sounds

Upon selection of this menu option, a dialog box will prompt the user to enter the number of incorrect sounds.

$ K + Help Menu

Contents

This file you are reading now is the help contents.

About

A dialog box pops up explaining about the program.

---

help_menu
$ Help Menu
K help; about
+ 09

$ K + Quick Menu

The quick menu is a group of buttons and checkboxes visible when the program begins. The play and record buttons function exactly like their counterparts in the Record Menu. Endpoint detection should be checked whenever the recorded sound does not last for the duration of the recording time. For example, if the recording time is set to 10 seconds, and the speech sound is only recorded for 1 second then endpoint detection should be checked. Endpoint detection tends to be time consuming, so it is often better to do without it. The preemphasis filter is used to attenuate the lower frequency components. This is especially useful when the microphone being used is not of the highest quality. The preemphasis filter value can be adjusted in the Parameters Menu.

---

quick_menu
$ Quick Menu
K quick menu, endpoint detection, preemphasis filter, baseline, test sound, correct sound
+ 10