A SPEECH TRAINING AID FOR THE DEAF

A dissertation submitted in partial fulfilment of
the requirements for the degree of
Master of Technology

By
KISHORI N. TAKALIKAR
Roll No. 89307301

Guides:
Dr P.C. Pandey
&
Dr T. Anjaneyulu

Department of Electrical Engineering
Indian Institute of Technology, Bombay
January, 1991

DISSERTATION APPROVAL SHEET

Dissertation entitled ''A SPEECH TRAINING AID FOR THE DEAF'' is approved for the award of the degree of Master of Technology in Electrical Engineering.

GUIDE	CO-GUIDE	CHAIRMAN	EXAMINERS
28/1/91			

Kishori N. Takalikar: A Speech Training Aid for the Deaf. MTech dissertation, Electrical Engineering, I.I.T. Bombay, January 1991.

ABSTRACT

Profoundly deaf persons face difficulty in acquiring and producing proper articulation and prosodic features of speech due to lack of auditory feedback. This project is aimed at developing a speech training aid for the deaf which will display a realistic vocal tract shape, pitch, and energy corresponding to the input speech waveform, with the objective of providing feedback for learning place of articulation, intonation, stress, and rhythm.

In the first stage of aid development, processing of speech waveform and display for visual feedback were carried out in off-line mode on a PC. A realistic vocal tract shape is estimated from the reflection coefficients obtained from linear predictive coding, and energy and pitch values are estimated from the autocorrelation analysis of short segments of the discretized input. Vocal tract shape, energy, and pitch for the selected segment are simultaneously displayed on the PC screen.

Finally, a system for real-time analysis of speech signal and display was implemented by using a PC, a DSP TMS-32010 Evaluation Module (EVM) from Texas Instruments and extension and interface hardware developed in an earlier project. Using this system, estimates of realistic vocal tract shape and energy values for the selected short segments from a speech signal of up to one second duration can be displayed.

ACKNOWLEDGEMENT

I am deeply indebted to my guides Dr P.C. Pandey and Dr T. Anjaneyulu for their constant encouragement and guidance throughtout this work.

I am thankful to Dr U.B. Desai for allowing me to use the DSP TMS-32010 Evaluation Module. I wish to express my sincere thanks to Mr Milind Gupte and Mr Anil Vartak for their timely help.

I would also like to thank Dr M.N. Nagaraja, Deputy Director of AYJ National Institute for the Hearing Handicapped, Bombay, and other staff members of the same institute for the kindly help extended by them in taking the speech recordings from deaf students.

KISHORI TAKALIKAR

C O N T E N T S

CHAPTERS

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW OF THE PROBLEM

The deaf people have considerable difficulty in understanding speech of others, and they cannot monitor their own speech very well resulting in abnormal speech patterns. Thus the deaf people become handicapped in two ways. Children who are either born deaf or who become deaf very early in life are likely to become dumb because their environment does not contain the auditory cues used by normal-hearing children in acquiring speech.

For the most part, the speech training of deaf children is based on a method in which the child is taught the speech gestures through visual observation of the teacher´s lips and face, through use of residual hearing, and through tactile sensing of the teacher´s face, neck, and breath stream (Levitt, Pickett et al, 1980). Recently, computer based speech training aids, which are flexible, easy to use, and versatile are being developed (Nickerson & Stevens, 1973 and Kushler et al, 1985). It is relatively easy with a computer driven display to adjust scale factors and other display features through software, store speech parameters for future displays, and display reference patterns.

1.2 PROJECT OBJECTIVES

This project is aimed at developing a computer based speech training aid for displaying speech information in the desired format such as temporal patterns of vocal tract shape, pitch, and energy. The hearing impaired person can see his vocal tract shape and can compare it with the target one in order to understand how he articulated and how he should articulate. Information like pitch, and energy variation can be used to improve the prosodic characteristics of speech.

A speech processor and display for the deaf has been earlier developed by Gupte (1990), for analysing the speech signal in real-time using a PC, a DSP TMS-32010 Evaluation Module (EVM) from Texas Instruments, and the extension and interface hardware developed by him. Although this system handles speech analysis for displaying vocal tract shape in real-time, the overall real-time performance is not achieved. Also, vocal tract shape display is in a staircase form not very suitable for the speech training. The present project is essentially an extension of the same, using the same hardware set-up. Its objectives are to improve display updating functions, achieving natural vocal tract shape, pitch, and energy for off-line speech data as well as for real-time speech. The display for off-line speech data can be used for experimentation and processing of speech signal where non-real-time analysis and display is permitted. The real-time speech analysis and display can be used for developing useful speech training aid.

1.3 OUTLINE OF THE REPORT

Chapter 2 provides an overview of speech signal and its features, speech production mechanism, and speech training aids. This is meant to serve as a background for the following chapters.

Chapter 3 presents the information about the estimation of vocal tract shape from acoustic waveform, along with a brief description of the earlier work by Gupte (1990), the development of a speech processor and display system. Chapter 4 presents the developement of a speech processing and display technique experimented in off-line mode, i.e., for synthesised speech data. The display procedure for a realistic form of vocal tract shape for real-time speech data is discussed in Chapter 5. Test results for the off-line and the real-time speech data for the simultaneous display of realistic form of vocal tract shape, pitch and energy are also presented in Chapter 4 and Chapter 5 respectively. Chapter 6 summarises the dissertation.

The appendices provide the supplementary information and data not included in the main body. Classification of the phonemes are briefly overviewed in Appendix A. Appendix B provides description about linear predictive coding. Program listings are included in Appendix C.

CHAPTER 2

SPEECH TRAINING AIDS

2.1 INTRODUCTION

Traditionally, speech training for the deaf children has been based on a method in which the teacher teaches speech by providing the child with a visual observation of his articulatory gestures during speech production and those of the student himself with the help of a mirror, teaching him to make use of his residual hearing, and through tactile sensation of his face, neck and breath stream. Now, several electronic devices are being developed which display information like voicing, pitch and energy variation, vocal tract shape, temporal speech patterns, etc (Gulian et al, 1984 and Kushler et al, 1985).

This chapter begins with a brief introduction to acoustic phonetics and the fundamentals of speech production mechanism, and provides an overview of some of the speech training aids reported in the literature.

2.2 SPEECH SIGNAL: AN OVERVIEW

The acoustics of speech production can be understood with the help of a schematic diagram of the vocal apparatus as shown in Fig.2.1. The air from the lungs is forced through the larynx into the mouth, and nose and finally passes to the surrounding acoustic medium. The vocal cords (or glottis) are situated roughly in the middle of the larynx. The region above the larynx consisting of the pharynx, nose, mouth, and lips is known as the vocal tract

(Rabiner & Schafer, 1978).

Speech signals are composed of a sequence of distinctive sounds, known as phonemes. These sounds, and the transitions between them, symbolically represent the information. Speech sounds can be classified into three groups according to the mode of excitation of the vocal tract filter. Voicing refers to the excitation consisting of a sequence of quasiperiodic pulses of air produced by forcing air through the glottis with the tension of the vocal cords adjusted so that they vibrate in a relaxtion oscillation. Frication refers to the excitation of a broad spectrum noise source which is generated by forcing air at a high enough velocity, through a constriction at some point in the vocal tract after the larynx. As a result of which, a turbulent flow is created though the constriction. Plosive sounds are produced by making a complete closure (usually toward the front of the vocal tract), building up pressure behind the closure and abruptly releasing it.

Vocal tract can be represented by a tube of non-uniform cross-sectional area. The frequency selectivity of the tube shapes the spectrum of the radiated sound. The resonant frequencies of the tract are called formant frequencies which vary with its shape, and dimension and can be changed with the movement of the articulators (velum, tongue, teeth, lips, and jaw). Diffefent speech sounds are produced by varying the vocal tract configuration, and its mode of excitation.

Phonemes of American English sounds are given in Table 2.1 and phonemes for Hindi are given in Table 2.2 and Table 2.3. Each of these phoneme can be classified as continuant or non-continuant, according to the movement of articulators. Continuant sounds are produced by a fixed (non-time varying) vocal tract configuration excited by the appropriate source and are characterized by steady-state spectral formants. This class includes the vowels, the nasals, and the fricatives. The change in vocal tract configuration produces non-continuant sounds which are marked by changing patterns in the spectrogram (a two dimensional pattern in which vertical dimension corresponds to frequency and the horizontal dimension to time). The diphthongs, the semivowels, the stops, and the affricates are included in this category. Further details about the classification of sounds are given in Appendix A.

The suprasegmental or prosodic characteristics of speech either directly affect the meaning or convey information about voice quality (Levitt et al, 1980). Factors that affect voice quality are loudness, average pitch, nasality etc. The key suprasegmentals that affect the meaning directly are intonation, stress, rhythm and phrasing. Intonation is the modulation of the voice pitch, i.e., the frequency of vibration of the vocal cords. Rhythm is the pattern in which the syllables are stressed. Phrasing refers to the way in which the words are grouped together according to the linguistic structure of the utterance.

2.3 LIPREADING AND LIPREADING AIDS

The visual signals from the speaker's face provide certain cues about the manner and place of articulation during speech production. A deaf child is taught to speak using these visual cues along with any available auditory information. This method of teaching deaf to speak is known as lipreading or speechreading. For profoundly deaf persons, lipreading becomes the most important means for speech perception. All kinds of aids, whether they are auditory, tactile or visual are used in combination with lipreading for this group of hearing impaired people.

Only 40% of the speech sounds are visible, and out of these, several sounds are visually indistinguishable. Further, the information about voicing, and nasality and the prosodic information (intonation contour, syllable stress, word segmentation cues) are not available in the visual signal. Here we will briefly discuss examples of lipreading aids employing visual, residual auditory, and tactile modes for information presentation.

Upton Eyeglass speechreader (Upton, 1968) extracts certain acoustic features that are associated with speech distinctions that are difficult to lipread, such as high frequency friction, low frequency friction, plosives and the low frequency murmur of nasal consonants. Whenever a specific acoustic feature occurs in the signal, it is displayed as a bar of light. The separate bars of light are arranged using segments of very small alpha-numeric module of light emmiting diodes. The analyser circuit for extracting speech features consists of four bandpass filters, level detectors, and a logic network. There are limitations of the speech analyser like errors in feature extraction or speech segment categorization.

The transposer hearing aid (Martony & Spens, 1972) is a two channel amplifying system in which, channel one acts as a normal hearing aid while channel two transforms the high frequency energy in phonemes such as [s] and [ʃ] into low frequency noise. The output of both the channels are then combined, so that listener gets normally amplified speech in addition to high frequency information now transposed into low frequencies. Therefore hearing impared listeners with residual hearing may benefit by using transposer in addition to lipreading.

Tactile speech training aids provide the information of speech sounds without any visual display. The spatial patterns of either vibration (vibrotactile aids) or electrical stimulation (electrotactile aids) along the skin can be represented as the frequency spectrum or features of speech. Tactile aids are also used in combination with lipreading. It provides indications of speech features that are difficult to identify by watching the talker. The major problem of tactile aids is the limitation of the skin to deal with a stimulus as complex as the acoustic speech signal. In the tactile vocoder developed by Pickett (1963), the speech signal is subjected to pre-emphasis of the high frequencies and then divided by filters into ten channels having different centre frequencies in the range 200 to 7700 Hz. The output of each filter is rectified and smoothed to obtain a control voltage. A 300 Hz sinusoidal signal is amplitude-modulated by each control voltage. The modulated voltage is amplified to drive a bone conduction transducer which serves as a vibrator. Each channel has a separate vibrator. Deaf child has to place his fingertips, palms down, on the vibrators so as to sense the vibratory representation of the speech patterns. Thus the tactile vocoder analyses the frequencies of the speech signal and presents them as an array of vibratory stimuli to the fingertips of the deaf child. Pickett has reported that using the above tactile vocoder Swedish vowels /i/ and /e/ can be discriminated better tactually than by lipreading but the tactual discrimination between the Swedish vowels /y/ and /u/ is as good as discrimination by lipreading the two degrees of rounding. Also, the consonant and vowel durational patterns, and the number of syllables in a word can be perceived better tactually using the above vocoder than by lipreading.

2.4 VISUAL AIDS

Visual aids provide a visual feedback of speech characteristics by displaying the information regarding speech features like fundamental frequency of voice, energy level of the speech signal, sound speech spectra. These systems are used in addition to the conventional hearing aids. Here we will briefly discuss a few of these.

2.4.1 Visual speech training system

'Vocal-2', a visual speech training system, is presently under use in AYJ National Institute for the Hearing Handicapped, Bombay (Dr Maniram, personal communication, from the same Institute). It can be used for speech training in a clinical environment. It is a two channel acoustic processing instrument that displays the sound speech spectrum on the screen of a video monitor. One channel is for the teacher and the another is for the student. Input signal is adjustable over a 40 db range. Processing and display can be selected from the two available modes: amplitude vs time and frequency vs time. In frequency vs time mode, the vertical amplitude of the display represents the frequency characteristics produced during vocalisation. There are three frequency bands: F1 (70-140 Hz) for males, F2 (140-280 Hz) for females and F3 (280-560 Hz) for children. One frequency band has been designed for specific practice by the student to allow visualisation of fricative information. This is called band S1 (4-8 kHz). It is useful in diagnosis and rehabilitation of cases with misarticulation, voice disorders, and errors in supra-segments.

2.4.2 Voice pitch display

One of the major difficulties that the hearing impaired person has to face while learning to speak is learning to control the pitch of their voice in order to produce intelligible, natural sounding speech (Kushler & Misu, 1985). Voice pitch display indicator displays the variation of pitch (fundamental frequency of vibration of the vocal cords) of the input voice signal with respect to time. The display is useful for the training of three different aspects of speech production: intonation, voice register control, and phonation (Risberg, 1968).

One such system which has been earlier developed by Nickerson & Stevens (1972) gives a simple graph of fundamental frequency plotted on a logarithmic scale as a function of time. This is shown in Fig. 2.2. The left half of the trace shown in this figure represents the utterance 'Is he coming? ' with the emphasis on the third syllable. The right half represents the same question with the emphasis on 'he'. The horizontal line indicates either silence or voiceless sounds.

2.4.3 Rhythm indicator

Commonly occuring error in the speech of the hard of hearing and the deaf is uncorrect rhythm. Rhythm indicator can be used for training of intensity, rhythm, and phonation. In one such system which is developed by Risberg (1968), the signal is passed through a low-pass filter with 100 Hz cut-off frequency and further through a high-pass filter with a cut-off frequency of 4 kHz. Both the filters are followed by rectification and smoothing. The output from the low-pass channel deflects the beam of the oscilloscope upwards and the high-pass channel deflects the beam downwards. The rhythm pattern of the word [sku: la] is shown in Fig. 2.3. The unvoiced sounds deflect the beam downwards, voiced sounds deflect the beam upwards and the amount of deflection is proportional to the intensity. The authors have not given test results for other sounds, but the scheme does not seem to be a robust one. The basic assumption here is that unvoiced sounds have higher freqency components and voiced sounds have frequencies below 100 Hz.

2.4.4 Nasalisation indicator

In the denasalization and nasalization training, the speaker is let to feel the vibration on the nose in nasalized sounds. In the nasalization indicator developed by Risberg (1968), a vibration signal is picked up on the nose by means of a contact microphone and the intensity gives a reading on a meter. The signal picked up by the microphone can also be heard in headphones. The author has not reported any of the test results.

Table 2.1. Phonemes in American English. A keyword is given in the parentheses along with the phonetic symbol for each vowel. Adapted from Pandey (1987).

Class	Subclass		Phonemes
Vowel	Front		i (beet), I (bit), e (bet), ae (bat)
	Mid		a (balm), A (but), ow (bought)
	Back		u (boot), U (book), O (bore)
Diph-thong			aI (buy), oI (boy), aU (how), eI (bay), oU (boat), ju (hue)
Semi-vowel	Liquid		r (red), l (led)
	Glide		w (wet), y (yes)
Conso-nant	Nasal		m (ram), n (ran), ng (rang)
	Stop	Voiced	b (bye), d (dye), g (guy)
		Unvoiced	p (pea), t (tea), k (key)
	Fricative	Voiced	v (vision), dh (this), z (zip), zh (vision)
		Unvoiced	f (fin), th (thin), s (sin), sh (shin)
	Affricate	Voiced	dzh (jail)
		Unvoiced	tsh (chain)
	Whisper		h (hate)

Table 2.2. Stop, affricate, fricative, and nasal consonants of Hindi and English, along with the place of articulation. Hindi consonants are shown in Devnagari letters and English consonants are shown in IPA (International Phonetic Alphabet) symbols. Adapted from Chafekar (1990).

UV = unvoiced, VO = voiced, UA = unaspirated , and AS = aspirated.

Place of articulation	Stop				Fricative		Affricate				Nasal
	UV		VO		UV	VO	UV		VO		VO
	UA	AS	UA	AS	UA	UA	UA	AS	UA	AS	UA
Velar	k		g								ŋ
	क्	ख्	ग्	घ्							ङ्
Palatal					ʃ	ʒ	tʃ		dʒ		
					श		च्	छ्	ज्	झ्	ञ्
Alveolar/ Retroflex	ट्	ठ्	ड्	ढ	ष्						ण्
Alveolar	t		d		s	z					n
					स्						
Dental					θ	ð					
	त्	थ्	द्	ध्							न
Labio-dental					f	v					
Bi-labial	p		b								m
	प्	फ्	ब्	भ्							म्

Table 2.3. Pronounciation keys for vowels in Hindi. Vowel in the middle position in each word is the pronounciation key.

vowel		word containing the vowel	
अ	/ʌ/	कल	/kʌl/
आ	/a/	काल	/kal/
इ	/I/	दिन	/d̪In/
ई	/i/	दीन	/d̪in/
उ	/ʊ/	सुर	/sʊɤ/
ऊ	/u/	सूर	/suɤ/
ए	/ɛ/	केश	/kɛʃ/
ओ	/o/	कोश	/koʃ/

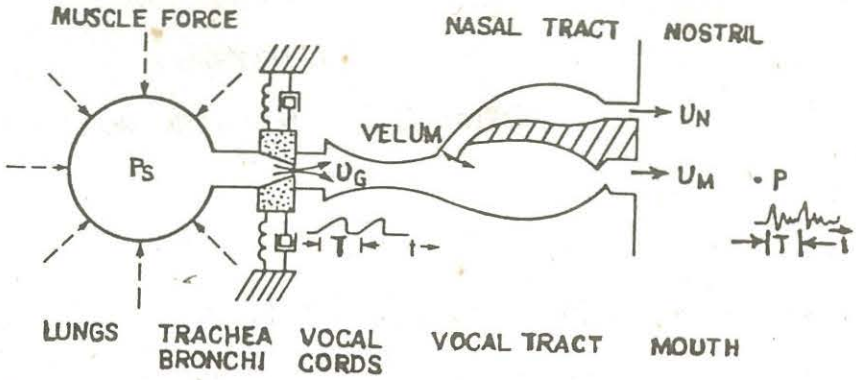


Fig. 2.1. Schematic diagram of the vocal apparatus. Adapted from Rabiner & Schafer (1978).

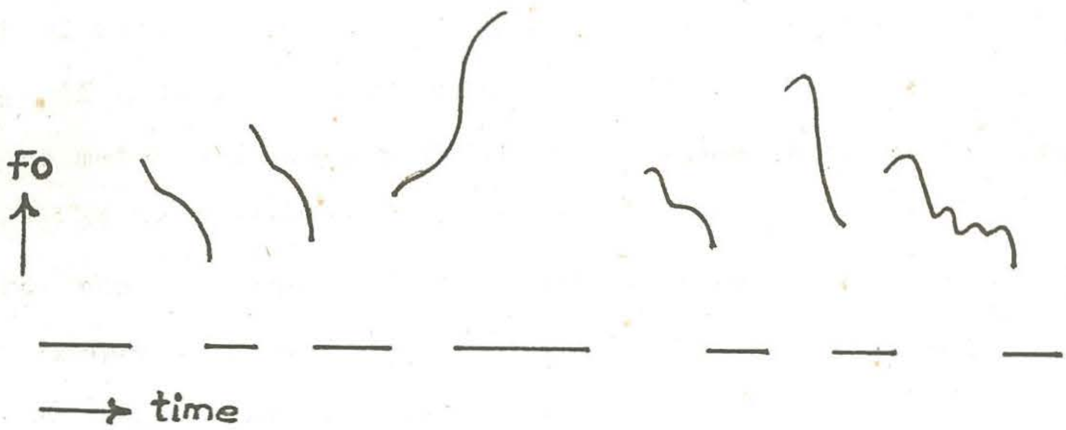


Fig. 2.2. Fundamental frequency versus time. Adapted from Nickerson & Stevens (1973).

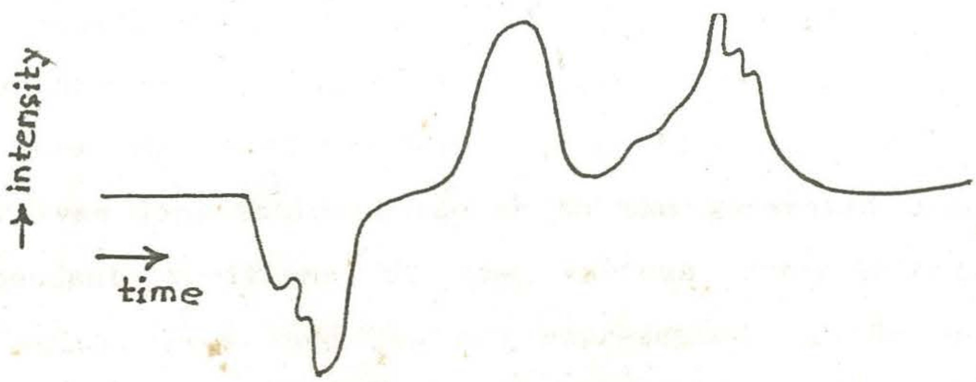


Fig. 2.3. Rhythm pattern of the word [sku: la]. Adapted from Risberg (1968).

CHAPTER 3

ESTIMATION OF VOCAL TRACT SHAPE

3.1 INTRODUCTION

This chapter is concerned with estimating the vocal tract shape from the acoustic waveform. The vocal tract shape can be determined from low formant frequencies or by using linear predictive coding (LPC) (Mermelstein, 1967 and Crichton et al, 1974). The display of vocal tract shape helps in understanding the movement of articulators, i.e., tongue, lip, jaw, and velum while speaking. If a target trace is also displayed, the deaf child can attempt to match his trace with the target one in order to correct his manner of articulation.

Here, the techniques of estimating the vocal tract shape from formant frequencies and LPC coefficients will be reviewed. A system for displaying the vocal tract shape estimated from LPC coefficients developed by Gupte (1990) will also be discussed.

3.2 VOCAL TRACT SHAPE FROM FORMANT FREQUENCIES

The speech wave results from the excitation of the vocal tract either by a quasiperiodic source at the glottis or by a noise source. The vocal tract modulates the excitation signal and thereby gives linguistic character to the generated signal. The time-dependent positions of the various articulators, e.g., tongue, velum, lips, and jaw are represented in the short-time power spectrum of the speech signal from which the formant frequencies, i.e., the resonant frequencies of the vocal tract may be determined (Mermelstein, 1967).

This section gives information about the extent to which the cross-sectional area functions are obtained from formant frequencies. The direct information about the shape of the entire vocal tract and the position of the articulators is available only through X-ray studies (Fant, 1959). The indirect way of obtaining articulatory data from acoustic data has got importance because of the difficulties in obtaining direct articulatory data like limited time resolution by exposure limitations etc.

The vocal tract may be modelled (Mermelstein, 1967) as a lossless acoustic tube with a sufficiently small rate of change of cross-sectional area with distance 'x' along the tract. The sound pressure p(x) is represented by Webster's horn equation,

$$\frac{d}{dx} \left[A(x) \frac{dp}{dx} \right] + \lambda A(x) p = 0 \qquad (3.1)$$

where A(x) = cross-sectional area

λ = eigen value

The formant frequencies are the frequencies of the normal modes of vibration of the vocal tract. From the first-order perturbation theory, the area function can be represented as

$$\log A(x) = \log A(0) + \sum_{j=1}^{\infty} a_j \cos (j\pi x/L) \qquad (3.2)$$

where L = length of vocal tract

The output admittance of the vocal tract measured at the lips as a function of frequency 'w' is represented as follows,

$$Y_t(w) = -U(L,w)/p(L,w) \propto (\delta\ [p(L,w)]/\delta x)\ /p(L,w) \qquad (3.3)$$

where U = volume velocity

The closed-lip boundary condition eigen-frequencies are the frequencies for which $Y_t(w) \rightarrow 0$, open-lip eigen-frequencies are obtained from the condition $Y_t(w) \rightarrow \infty$. The infinite sets of poles and zeros of the admittance function thus correspond to the eigen frequencies under the two sets of boundary conditions, respectively. At least up to first-order perturbations from the uniform-tract shape, they uniquely determime the perturbing area Fourier components.

The vocal tract area functions band limited to six components determined from the first six admittance poles/zeros along with the X-ray derived area function are shown in Fig. 3.1.

3.3 VOCAL TRACT SHAPE FROM LPC

Here, we will review estimation of vocal tract shape from linear predictive coding, as reported by Crichton & Fallside (1974). An all-pole digital filter model is used in this technique. The speech signal is analysed using LPC to obtain autocorrelation and reflection coefficients (details of LPC are given in Appendix B). The transfer function of an idealised acoustic tube is equivalent to that of the linear prediction model. The acoustic tube is terminated at the lips by an infinite-length, infinite-area tube, and the transfer function $A_n(z)$ of the inverse tube is the ratio of $U_n(z)$, the forward volume velocity at the point of excitation to $U_R(z)$, the radiant volume velocity at stage 'n'. $B_n(z)$ is the ratio of the backward volume velocity $V_n(z)$ at the point of excitation to the radiated volume velocity $U_R(z)$.

$$A_n(z) = U_n(z)/U_R(z) \tag{3.4}$$

$$B_n(z) = V_n(z)/U_R(z) \tag{3.5}$$

The area A(n) at stage 'n' is given by

$$A(n) = A(n-1)\ [1+F_K(n)]/[1-F_K(n)] \tag{3.6}$$

where $F_K(n)$ is the reflection coefficient at stage 'n'. If the area A(0) at the lips is given, the area of each stage can be calculated backward from the lips to the glottis.

Crichton & Fallside (1974) have used the foregoing technique as an aid to deaf speech training. In this application the computer is used to display the smoothed area functions (which are generated by a parabolic interpolation between discrete areas) plotted logarithmically against linear distance along the vocal tract. A fixed target trace produced by a child with normal hearing can also be displayed and the deaf child tries to match visually his trace to the target trace. Fig. 3.2 shows typical attempts by deaf child to match target.

3.4 A SPEECH PROCESSOR AND DISPLAY SYSTEM

A PC based real-time system for analyzing and displaying the vocal tract shape by LPC coefficients and digital signal processing techniques was earlier developed by Gupte (1990). A block diagram of the system is shown in Fig. 3.3. The digital signal processor TMS-32010 Evaluation Module (EVM) from Texas Instruments is used in this system. Apart from it, the analog pre-processor consisting of pre-amplifier, low pass filter (anti-aliasing filter), an extension card to the EVM and an interface to the IBM-PC are also incorporated in the system. After acquiring and analysing the speech signal in real-time, by the TMS-32010, the relevant information is transferred to the PC. Then the information about the vocal tract shape and energy contour is displayed by the PC. Using this system, one can speak into microphone for about one second and observe the variation of the vocal tract area function and the energy.

Although the speech analysis done here is in real-time, overall real-time performance is not achieved. The vocal tract shape display is in staircase form. By writing programs for the communication between the PC and the TMS-320 in the assembly language, the parameters could be transferred on a frame by frame basis. As mentioned in Section 1.2, the current work is an extension of this work by Gupte (1990), in which he has developed a speech processor and display for speech training for the deaf.

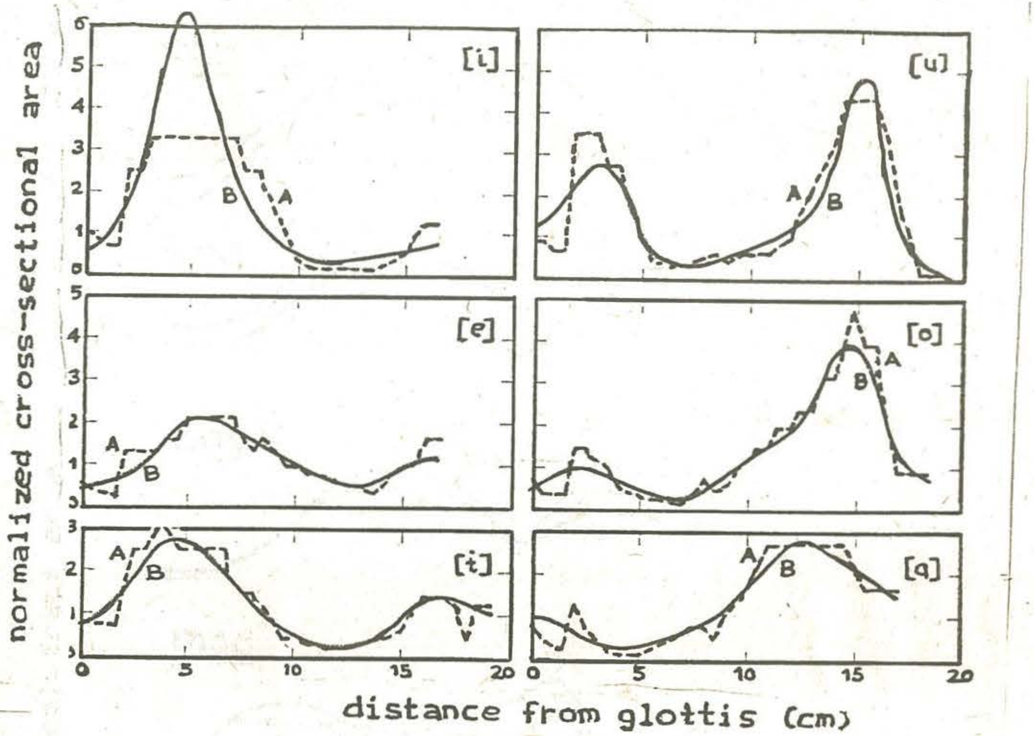


Fig. 3.1. Estimated area functions, (A) X-ray derived (B) Computed band-limited approximation. Adapted from Mermelstein (1967).

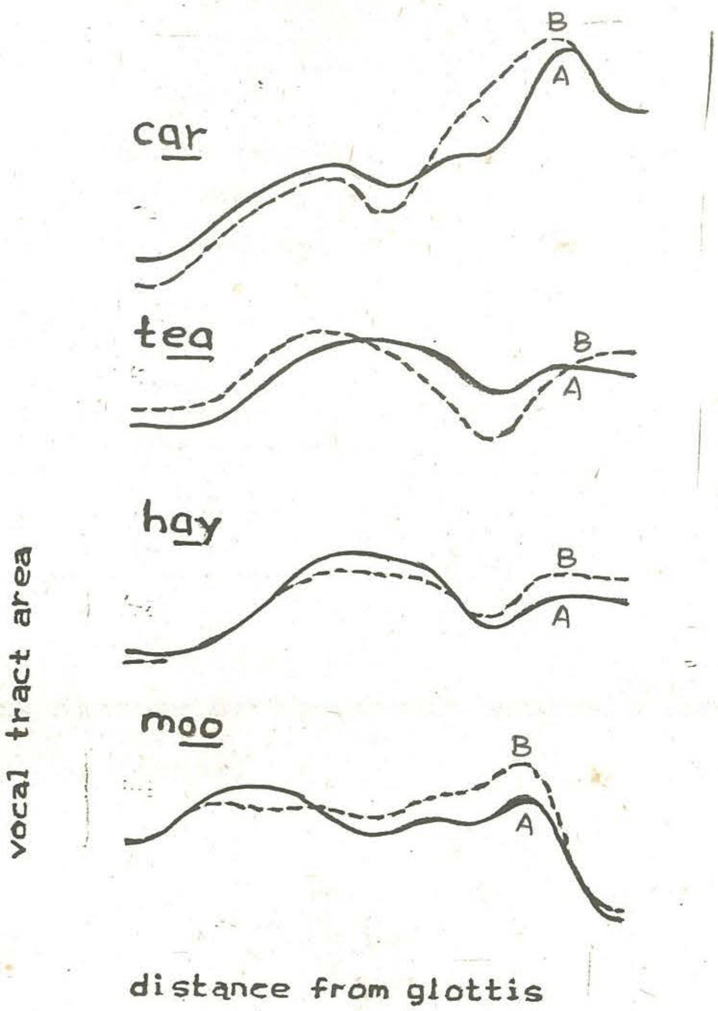


Fig. 3.2. Typical attempts by deaf child to match target trace, (A) attempt (B) target. Adapted from Crichton & Fallside (1974).

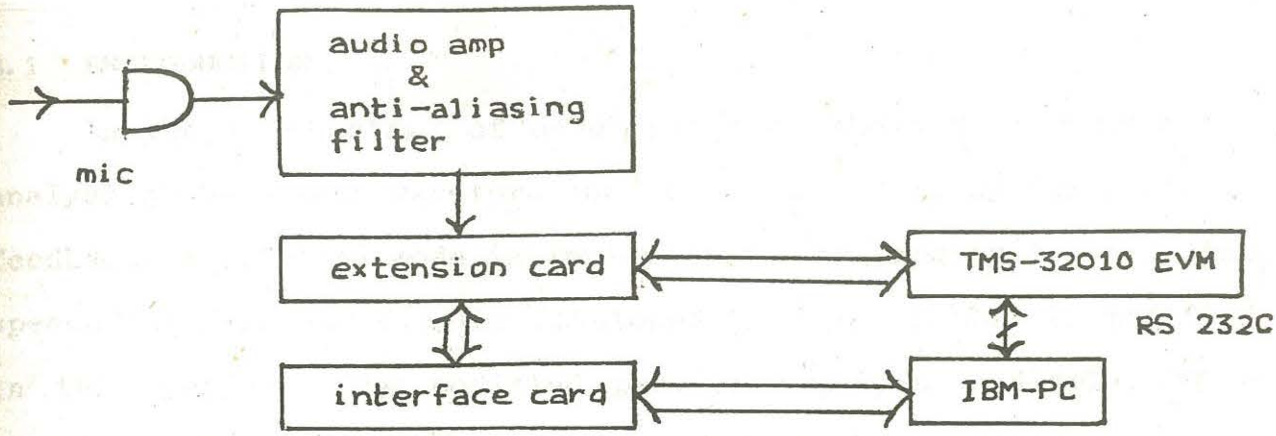


Fig. 3.3. Block diagram of the speech analysis and display system developed by Gupte (1990).

CHAPTER 4

OFF-LINE SPEECH ANALYSIS AND DISPLAY

4.1 INTRODUCTION

In the first stage of development of the aid, a system for analysing the speech waveform and providing a display for a visual feedback in off-line mode is implemented. The program for off-line speech analysis and display developed by Gupte (1990) is modified in this project. The modified program provides a display of a realistic form of vocal tract shape, pitch and energy values for the selected frames of speech data input. A realistic vocal tract shape is estimated from the reflection coefficients which are obtained from linear prediction coding, and energy and pitch values are estimated from autocorrelation analysis of short segment of the discretized input. The algorithm for generating synthetic data (Klatt, 1980), the autocorrelation algorithm (Morris, 1983), the Leroux & Gueguen algorithm (Leroux & Gueguen, 1977), are earlier implemented by Gupte (1990) and have been used in this project. The modified program SPEECH_1.PAS and the algorithms used in it are described in this chapter.

4.2 DESCRIPTION OF THE PROGRAM SPEECH_1.PAS

In the program SPEECH_1.pas, the speech data file is divided into a number of frames (speech segments). The frame length, i.e., the number of samples per frame can be changed. The data for successive frames are read and analyzed, one frame at a time. After reading the data for a frame, the autocorrelation coefficients are computed using autocorrelation algorithm (Morris, 1983 & Gupte, 1990) as follows,

1. The speech data samples D(0) to D(N-1) for one frame are read from a speech data file which is a file of integers.
2. Pre-emphasis and windowing of data is done using either rectangular or Hamming window to get the data samples D´[0] to D´[N-1].
3. For I = 0 to N-1 do
   (i) R[I] = 0, where R[I] is the I^{th} autocorrelation coefficient.
   (ii) For J = I to N-1 do
   R[I] = R[I] + D´[J] * D´[J-I]

From the autocorrelation coefficients, pitch period is estimated. The largest peak of the autocorrelation function is located and the peak value is compared with a fixed threshold (e.g. 30% of R[0]). If the peak value is less than the threshold then the speech segment is classified as unvoiced and if it is larger, then the pitch period is defined as the location of the largest peak. The total energy in a speech signal is represented by zeroth autocorrelation coefficient for that particular frame (Rabiner & Schafer, 1978).

The reflection coefficients are computed from autocorrelation coefficients using L-G algorithm which is described in Appendix-B. The vocal tract area functions are computed from the reflection coefficients using the formula,

$A(n) = A(n-1)\ [1+F_K(n)]/[1-F_K(n)]$

where $A(n)$ = area coefficient at stage 'n'

$F_K(n)$ = reflection coefficient at stage 'n'

and vocal tract is considered to be a lossless acoustic tube terminated by infinite length, infinite area-tube (Crichton & Fallside, 1974).

Only the area coefficients, pitch, and energy values are stored for each frame. After the analysis has been completed, any particular frame can be selected for display. For obtaining the realistic vocal tract shape display, the discrete area coefficients are scaled and then interpolated. The speech analysis and display program, finally provides a display of a typical human vocal tract shape which is based on the model given in Fig. 4.1. A realistic form of vocal tract is displayed in the form of a variation of a lower portion of the vocal tract according to the interpolated area coefficients. The energy and pitch values are displayed in the form of bars. Thus the changing positions of lips, jaw, and tongue for different speech segments can be seen on PC screen along with the variation of energy and pitch values. The speech analysis and display program can be summarised as follows,

1. The parameters like speech data file, sampling frequency, starting frame number (SFNO), total number of frames (NF) can be selected from menu.
2. Pre-emphasis and windowing is done after reading the speech data for SFNO.
3. Autocorrelation coefficients are computed using autocorrelation

algorithm.

4. Pitch is determined using autocorrelation coefficients. Energy value is computed from zeroth autocorrelation coefficient.
5. Reflection coefficients are computed from autocorrelation coefficients using L-G algorithm.
6. Area coefficients are computed from reflection coefficients.
7. A typical human vocal tract shape is displayed on PC screen along with the variation of its lower portion according to the interpolated area functions. The area fnctions in the staircse form is also displayed along with the variation of energy and pitch values.
8. Key ´C´ can be pressed in order to refresh area functions completely or key ´S´, to refresh the area functions segmentwise.
9. Key ´+´ or ´-´ can be pressed for the display of next or previous frame respectively. Any particular frame can be selected for display if the key ´=´ is pressed. Beep is produced if the selected frame is not in the given range. Key ´E´ can be pressed in order to exit from the display at any time.

4.3 TEST RESULTS

The Russian vowels (/a, e, i, u/) and a sample sound /ex1/ were synthesized by Gupte (1990) using Klatt-synthesizer (Klatt, 1980). The control parameters used for this synthesis are given in Table 4.1 and Table 4.2. From Table 4.1 and Table 4.2, it is clear that there is a variation in formant frequencies and also the bandwidths for different vowels. The intensity variation for synthetic data is given in Fig. 4.5 which is a plot of voicing amplitude AV in db against time. The speech data originally available as ASCII files was stored in integer data files using program INP.PAS.

The program SPEECH_1.PAS is run on the IBM-PC. The test results with simultaneous display of realistic vocal tract shape, pitch, and energy variation for different frames are shown in Fig. 4.3. Display of vocal tract area is in the form of variation of lower portion of the vocal tract as well as the area functions in staircase form. The general shapes of the area functions computed by SPEECH_1.PAS clearly match to those given by Fant (Rabiner & Schafer, 1978) as shown in Fig. 4.2. It is clear from Fig. 4.3 that for vowel /a/, there is a constriction in back side, while for vowel /e/, there is a constriction in the front side and degree of constriction is medium. Also, for vowel /i/, the constriction is in the front side but degree of constriction is high, and for vowel /u/, constriction is in the back side and because of lip rounding the area between two lips is coming out to be very small.

The energy and pitch values are shown in the form of bars.The energy values obtained for the first and last few frames are very small compared to the maximum one. Thus the energy variation talies with the one given in Fig. 4.5. In order to study the variation of pitch, pitch values for a synthetically generated sample sound /ex1/, which is an /a/ like sound are obtained for a duration of 500 ms. As shown in Fig. 4.4, except for two or three points, pitch variation obtained matches with the reference one.

Table 4.1. Fundamental frequency, formant frequencies, & bandwidths for some vowels, & sample sound /ex1/ which are generated by Klatt-Synthesizer, and pitch variation for /ex1/. Adapted from Rabiner & Schafer (1978), and Gupte (1990).

sound	F0	F1	F2	F3	BW1	BW2	BW3
/a/	125	650	1076	2463	94	91	107
/e/	125	415	1979	2810	54	101	318
/i/	125	223	2317	2974	53	59	388
/u/	125	232	507	2395	61	57	66
/ex1/	**	700	1220	2600	130	70	160

Duration 500 ms, voicing amplitude AV (db) versus time

Time	0	100	395	495
AV	0	60	60	0

** for /ex1/

Time	0	50	80	140	160	220	260	395	415	500
F0	125	125	175	175	125	125	225	225	175	175

Table 4.2. Control parameters for the all-pole cascade/parallel synthesizer. The list also shows the permitted ranges of values for each parameter, and a typical value. V/C indicates whether the parameter is normally variable (V), or constant (C).
Adapted from Klatt (1980), Table I.

N	V/C	Sym	Name	Min	Max	Typ
1	C	SW	Cascade/parallel switch	0(Cas)	1(Par)	0
2	C	NF	Number of formants	4	6	4
3	V	F0	Fundamental freq. of voicing (Hz)	0	500	0
4	V	AV	Ampl. of voicing (dB)	0	80	0
5	V	AF	Ampl. of frication (dB)	0	80	0
6	V	AS	Ampl. of sinusoidal voicing (dB)	0	80	0
7	V	AH	Ampl. of aspiration (dB)	0	80	0
8	V	F1	First formant freq. (Hz)	150	900	450
9	V	F2	Second formant freq. (Hz)	500	2500	1450
10	V	F3	Third formant freq. (Hz)	1300	3500	2400
11	V	BW1	First formant bandwidth (Hz)	40	500	50
12	V	BW2	Second formant bandwidth (Hz)	40	500	70
13	V	BW3	Third formant bandwidth (Hz)	40	500	110
14	C	A1	First formant amplitude (dB)	0	80	0
15	V	A2	Second formant amplitude (dB)	0	80	0
16	V	A3	Third formant amplitude (dB)	0	80	0
17	V	A4	Fourth formant amplitude (dB)	0	80	0
18	V	A5	Fifth formant amplitude (dB)	0	80	0
19	V	A6	Sixth formant amplitude (dB)	0	80	0
20	V	AB	Bypass path amplitude (dB)	0	80	0
21	C	AN	Nasal formant amplitude (dB)	0	80	0
22	V	FNZ	Nasal zero freq. (Hz)	200	700	250
23	V	FNP	Nasal pole freq. (Hz)	200	500	250
24	C	UPDT	Parameter update int. (ms)	2	20	5
25	C	SR	Sampling rate (Hz)	5000	20000	10000
26	C	G0	Overall gain control (dB)	0	80	0
27	C	F4	Fourth formant frequency (Hz)	2500	4500	3300
28	C	F5	Fifth formant frequency (Hz)	3500	4900	3750
29	C	F6	Sixth formant frequency (Hz)	4000	4999	4900
30	C	BW4	Fourth formant bandwidth (Hz)	100	500	250
31	C	BW5	Fifth formant bandwidth (Hz)	150	700	200
32	C	BW6	Sixth formant frequency (Hz)	200	2000	1000
33	C	BWNZ	Nasal zero bandwidth (Hz)	50	500	100
34	C	BWNP	Nasal pole bandwidth (Hz)	50	500	100
35	C	FGP	Glottal res. 1 freq. (Hz)	0	600	0
36	C	BWGP	Glottal res. 1 bandwidth (Hz)	100	2000	100
37	C	FGZ	Glottal zero freq. (Hz)	0	5000	1500
38	C	BWGZ	Glottal zero bandwidth (Hz)	100	9000	6000
39	C	BWGS	Glottal res. 2 bandwidth (Hz)	100	1000	200

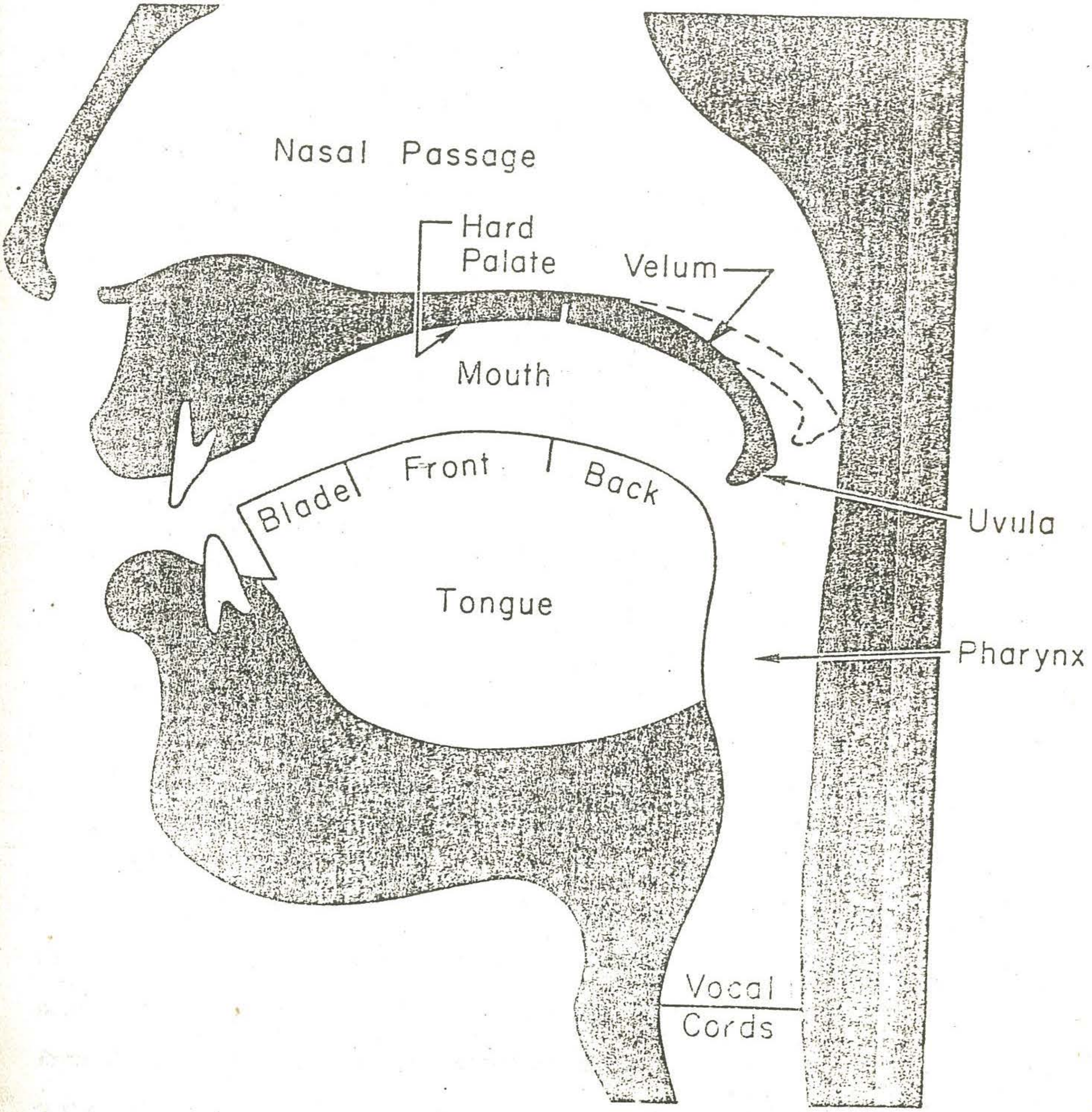


Fig. 4.1. A typical human vocal tract shape. Adapted from Levitt, Pickett, & Houde (1980).

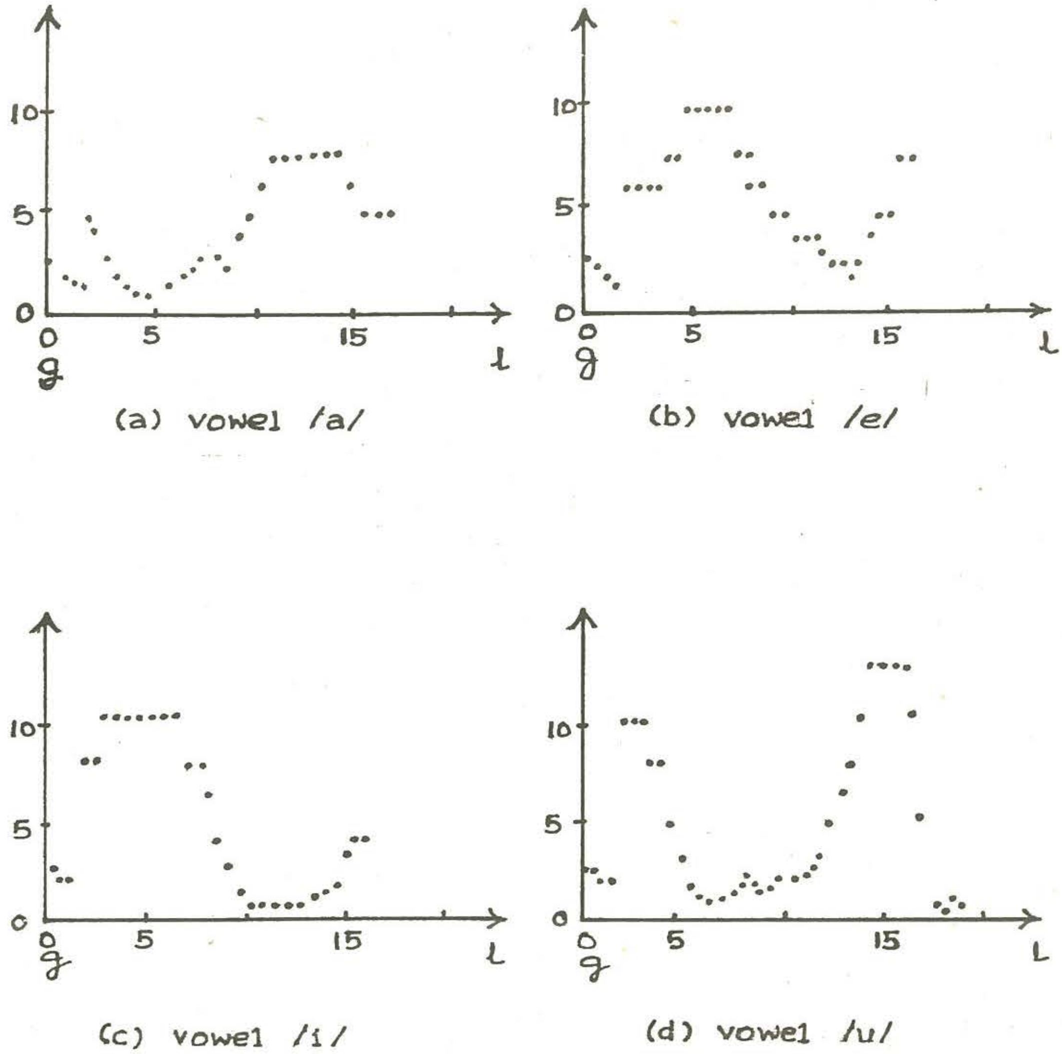
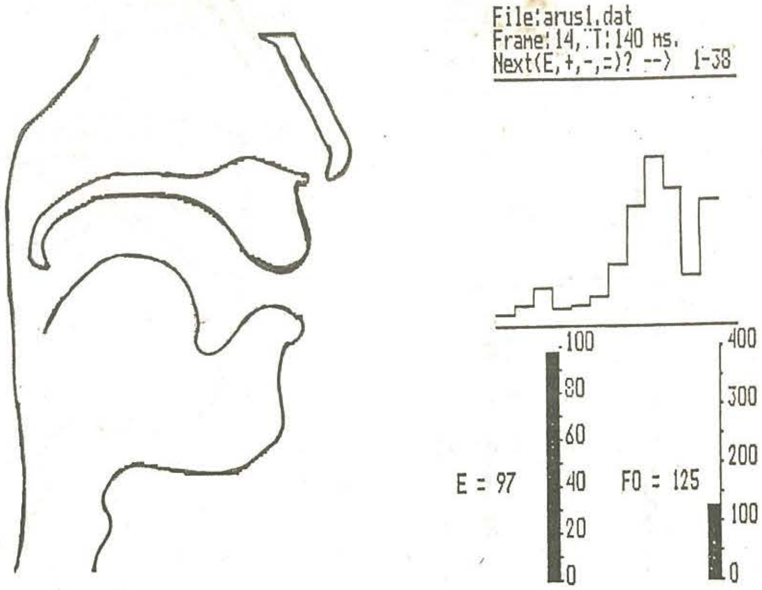
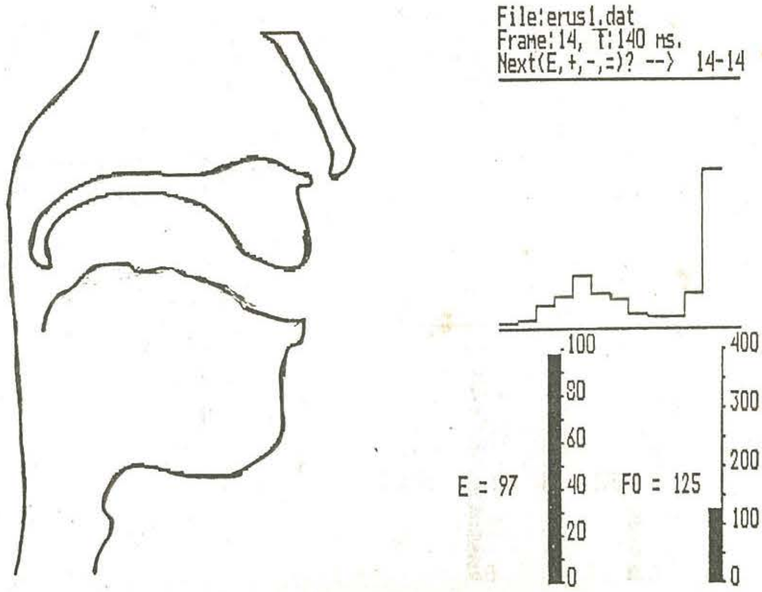


Fig. 4.2. Normalized vocal tract area functions as given by Fant. Adapted from Rabiner & Schafer (1978). X-axis indicates distance from glottis & Y-axis indicates normalized area functions.

g = glottis & l = lips.

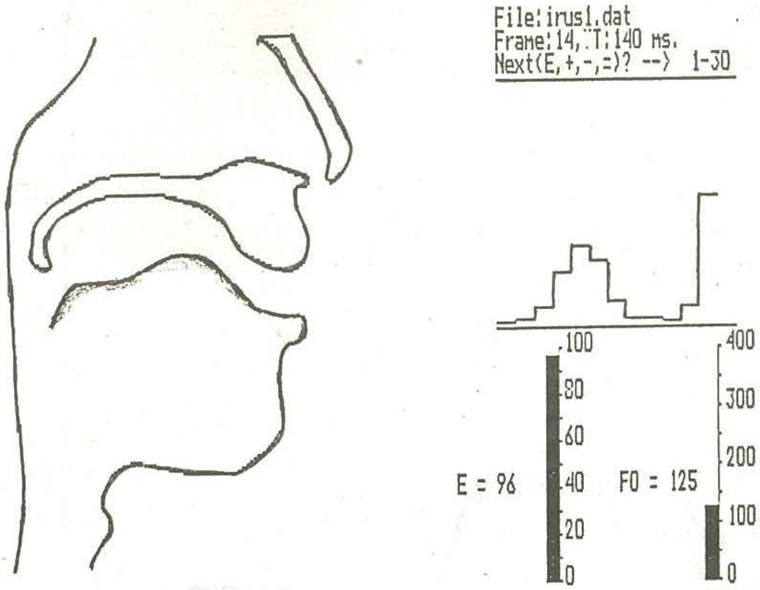


(a) vowel /a/

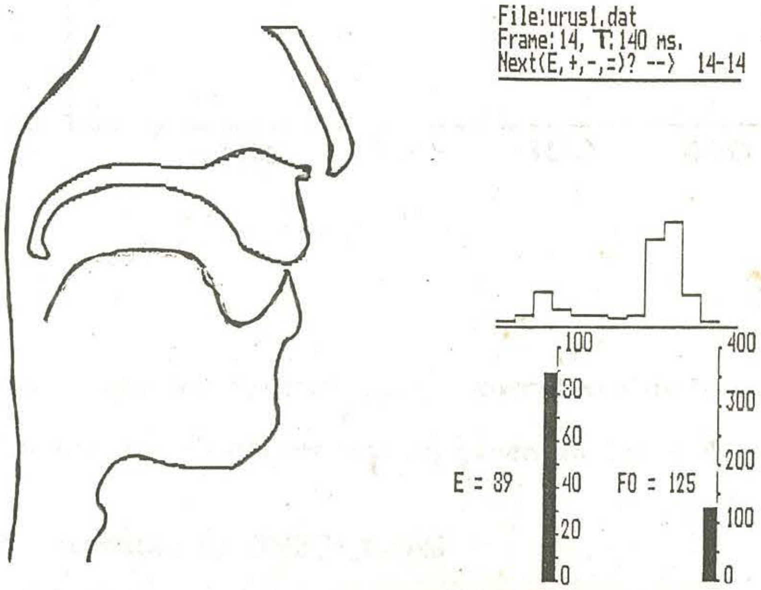


(b) vowel /e/

Fig. 4.3. Display of area functions realistic as well as staircase form along with the vartation of pitch and energy.



(c) vowel /i/



(d) vowel /u/

Fig. 4.3. Display of area functions realistic as well as staircase form along with the variation of pitch and energy.

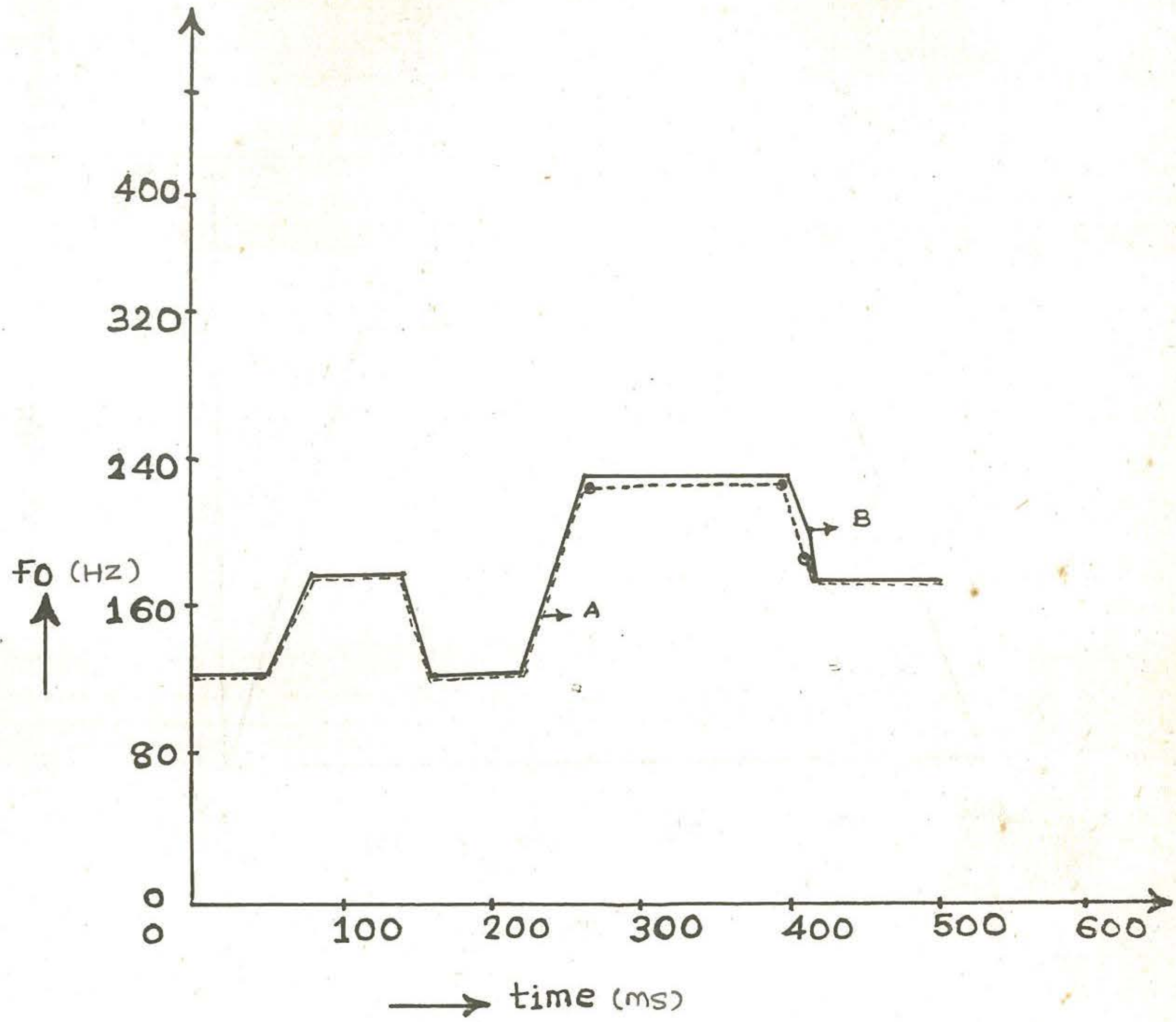


Fig. 4.4. Pitch variation for synthetically generated sample sound /ex1/, an /a/ like sound with parameters as given in Table 4.1.

(A) Reference (B) estimated by SPEECH_1.PAS

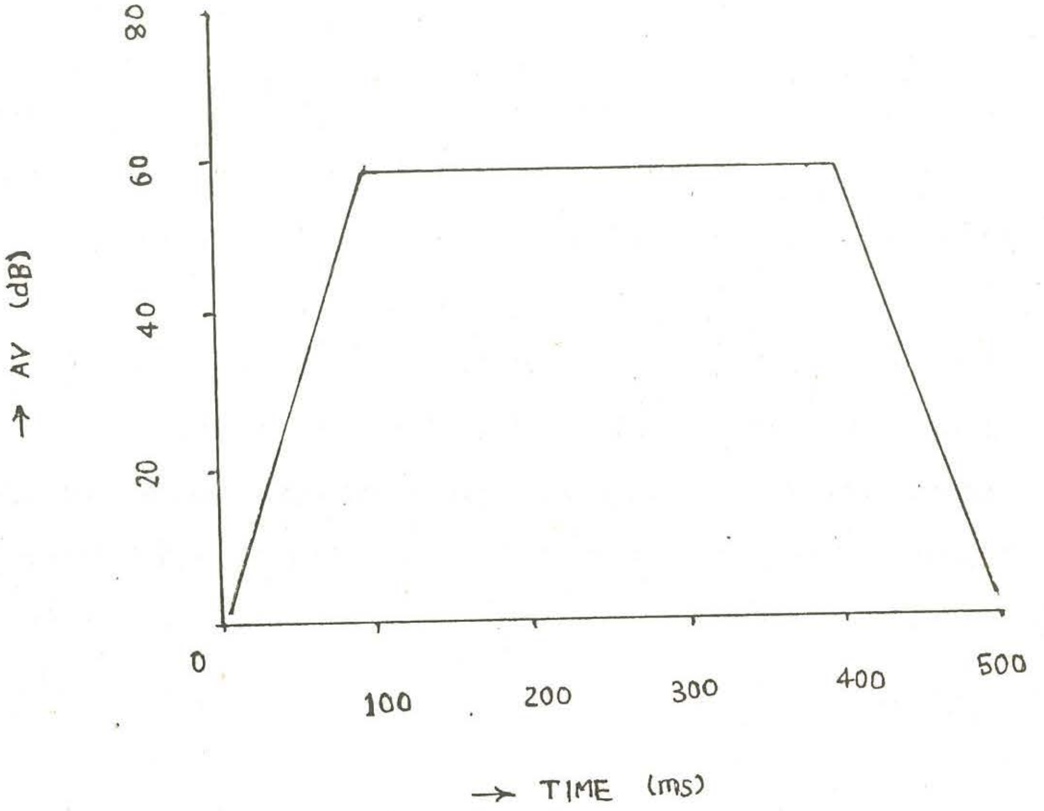


Fig. 4.5. Variation of the amplitude of voicing, AV, during the synthesis of isolated vowels.

CHAPTER 5

REAL-TIME SPEECH ANALYSIS AND DISPLAY

5.1 INTRODUCTION

For the real time analysis of the speech samples, the hardware developed by Gupte (1990) is used. Using this system and the modified display procedure one can speak into the microphone for around one second, and observe the energy and area function variation on the screen for that particular duration of speech. The system does not provide pitch information.This chapter describes the procedure for the hardware set-up and its testing. It also presents the description of the program developed for real-time analysis of speech, and display of realistic form of vocal tract shape and energy.

5.2 SYSTEM SET-UP & HARDWARE TESTING

The operating procedure for setting up of the speech analysis and display system is as follows,

(i) Connect the emulator cable of the EVM to the extension card through socket-U1, with the setting of the clock switches as follows, S1, & S2 OFF and S3, & S4 ON.

(ii) Connect the extension card to the PC interface through the connector C-1 using 40 pin flat ribbon cable.

(iii) Connect the proper supply voltages to TMS-32010, extension card, and anti-aliasing filter.

(iv) Set jumper J6 on the EVM to position 2-3 in order to use external $\overline{\text{BIO}}$ & $\overline{\text{INT}}$ but internal clock.

(v) Set the various jumpers on the extension card referring to Table 5.1.

(vi) By running KERMIT software (or some other terminal emulation program) on the PC, establish a serial communication link between the RS-232C connector of the PC and the connector C-1 of the EVM (parity is set to none).

(vii) Initialize EVM using INIT command, set CLOCK to EXTERNAL and PROGRAM MEMORY to INTERNAL.

After this initialization, the hardware system is tested as follows,

(a) Testing of A/D section

In order to get 10 kHz start of conversion pulses for ADC, the number 524 is sent to the cascade of bit rate multiplier ICs (7497) on the extension card.

$$fout = (M/64*64*64) * fin$$

for fout = 10 kHz and fin = 5 MHz, M = 524. Then the ADC is connected in the circuit and ADC data is read using hardware interrupt by running program TEST_1.TMS on TMS-32010. The status and interrupt pulses are observed on the CRO. For analog inputs of 0 and +10 volt, the ADC data is checked which is in the offset binary form. The respective values for 0 and +10 volt are +2040 & +4096.

(b) Testing of Host Interface

A program TEST_2.TMS is first run on TMS-32010. Then the program TEST_2.PAS is run on the PC which reads data from data memory of TMS-32010, using acknowledgement register of the extension card. Also, the other way communication from PC to TMS-32010 is verified using command register of the extension card. Simultaneously, the I/O decode, $\overline{WR}$, $\overline{RD}$, and $\overline{DEN}$ signals are verified.

(c) Testing of extended data memory (EDM)

A small program TEST_3.TMS is run on TMS-32010 which reads and writes data from and to the data memory respectively. Various control signals and the status of address and data lines are verified.

After checking of the hardware set-up, for the real-time analysis of speech and display following steps are carried out.

(i) Transmit the assembly language program STRAIN.TMS which is developed by Gupte (1990) to the EVM (the character 17H is transmitted with the command XON).

(ii) First run the program STRAIN.TMS on TMS-32010 and then run the modified PASCAL program SPEECH_2.PAS on PC.

5.3 DESCRIPTION OF THE PROGRAM SPEECH_2.PAS

The real-time speech data is acquired and processed by TMS-32010 by executing the program STRAIN.TMS. After every interrupt from ADC, the data is read by the TMS-32010 and stored in the program memory. The display procedue is handled by PC. The number of samples per frame, total number of frames to be acquired, offset value for the ADC, etc are set by software. The program SPEECH_2.PAS first reads all the above parameters and sends them to TMS-32010 data memory. After storing the data for each frame pre-emphasis and windowing is done by the assembly language program. Then the autocorrelation and reflection coefficients are computed using autocorrelation and L-G algorithm (these algorithms are already described in Chapter 4). After the processing of all frames, the reflection coefficients and zeroth autocorrelation coefficients are acquired by PC. The area coefficients are computed from the reflection coefficients using the Eqn. B.9. The area coefficients are scaled and interpolated to get a realistic form of vocal tract shape. Energy values are then computed from zeroth autocorrelation coefficients by normalization. A typical vocal tract shape is displayed on the screen, the data points for which are stored in the file COORD.PAS. The lower portion of the vocal tract shape is varied according to the new area coefficients. The upper portion of the vocal tract is fixed. Also, the energy for each frame is displayed along with the variation of originally developed staircase area functions. Thus the tongue, and lip movements for the given utterances (vowels) can be easily seen on the PC screen using the program SPEECH_2.PAS. Any random frame can be selected for display. By pressing keys '+' or '-', the succeeding or preceeding frames can be selected for display respectively. The area functions can be refreshed completely or segmentwise by pressing keys 'C' or 'S', respectively.

5.4 TEST RESULTS

The results are shown in Fig. 5.1 which provides a display of area functions and energy values for the sustained Hindi vowels spoken by the author herself. It can be observed from Fig. 5.1 (a) and (b), for vowel /a/, the constriction is in the back side and degree of constriction is low whereas for vowel /e/, the constriction is in front side and degree of constriction is medium. As shown in Fig. 5.1 (c) & (d), for vowel /i/, the constriction is in the front side but degree of constriction is high whereas for vowel /u/, constriction is in the back side and because of lip rounding the area between lips is coming out to be very small. Thus the degree of constriction and tongue hump position while speaking vowels match with the scheme given in Table 5.2 (Flanagan, 1972).

The energy values for the first and last few frames of the spoken vowels are very small compared to the maximum one. For rest of the frames energy is almost remains constant. Thus energy variation matches with the plot of voicing amplitude AV vs time as shown in Fig. 4.5.

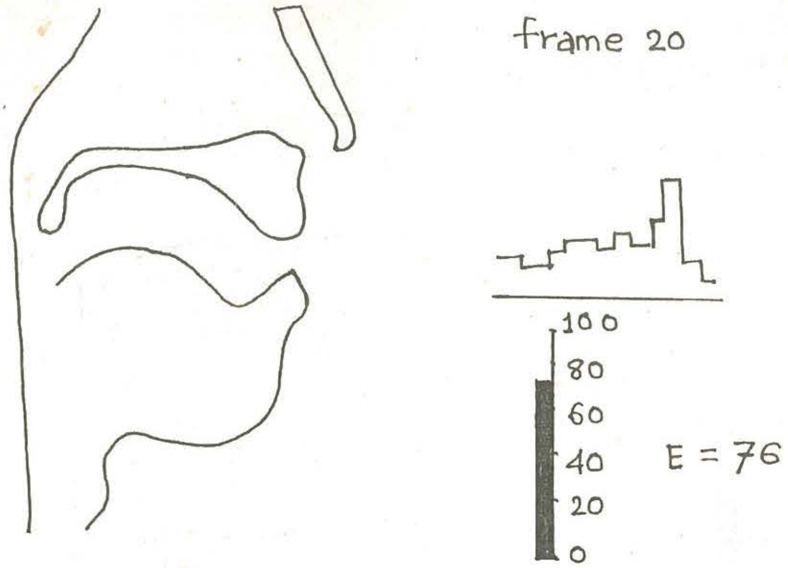
An extensive testing of the system with speech from normal and deaf, male and female, child and adult speakers with different languages needs to be carried out. Also, the system has so far been tested with sustained vowels only. This should be extended to sounds of other categories as well.

Table 5.1. Jumper settings for the extension card. Adapted from Gupte (1990).

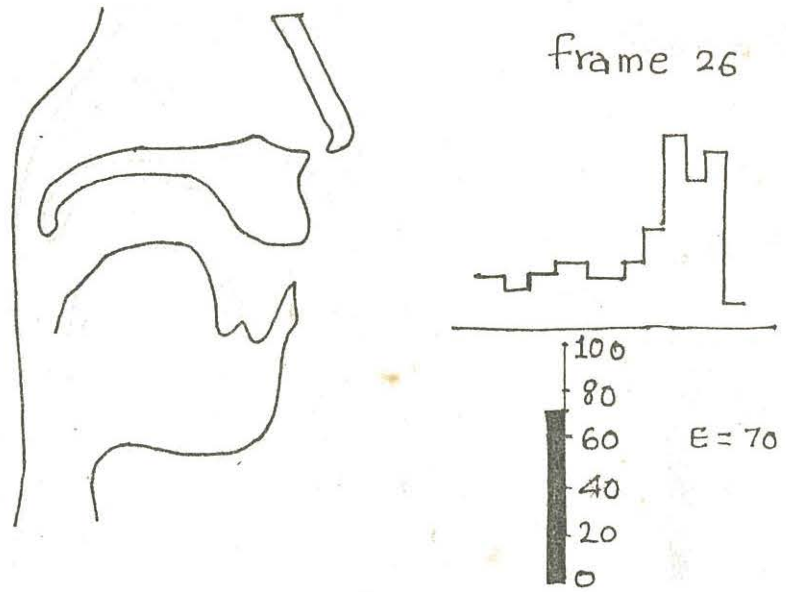
JUMPER	CONNECTION	FUNCTION
J-1	3-4	Directional signal for ADDRESS and DATA buffers.
J-2	2-4	For reading ADC data in offset binary form.
J-3	3-4	Provides clock syncronisation to $\overline{\text{INT}}$ signal.
J-4	1-2, 5-6, 9-10, 13-14	Provides HOST communication through $\overline{\text{BIO}}$ & ADC through $\overline{\text{INT}}$

Table 5.2. Classification of vowels according to the tongue-hump-position & degree-of-constriction. Adapted from Flanagan (1972).

Degree of constriction	Tongue hump position		
	front	central	back
high	/i/ eve	/ɝ/ bird	/u/ boot
	/I/ it	/ɚ/ over	/U/ foot
medium	/e/ hate	/ʌ/ up	/o/ obey
	/ɛ/ met	/ə/ ado	/ɔ/ all
low	/æ/ at		/a/ father

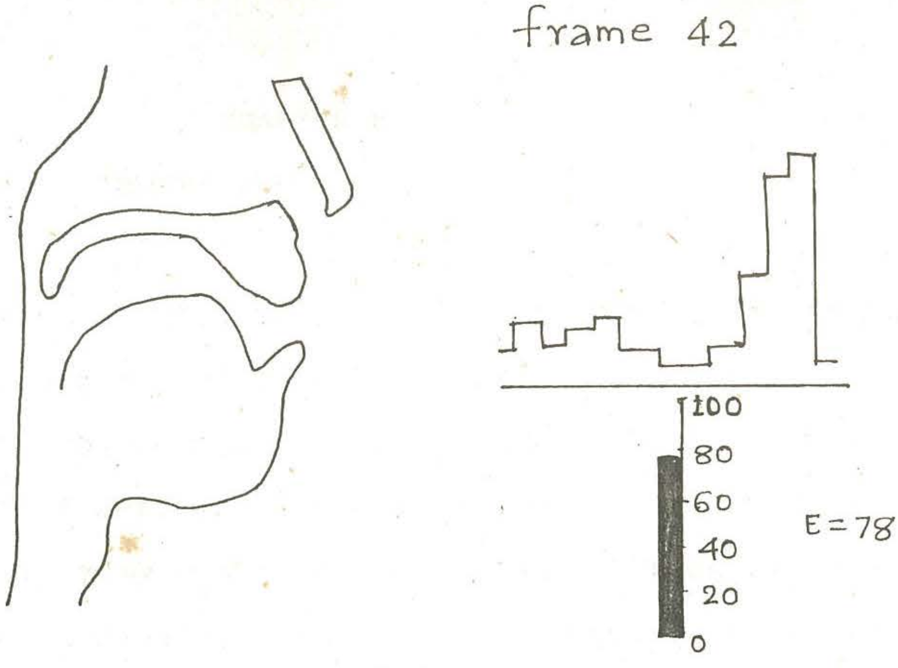


(a) vowel /a/

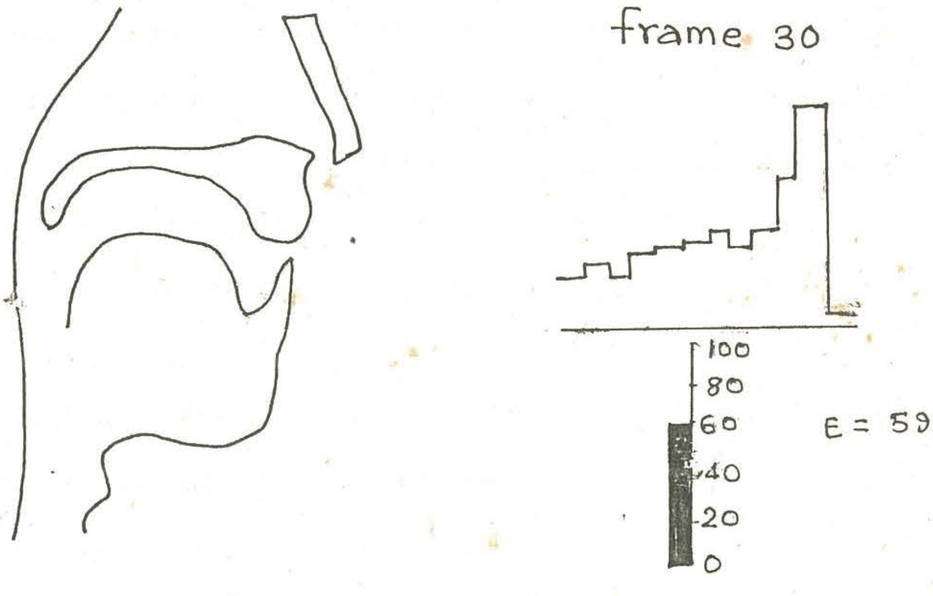


(b) vowel /e/

Fig. 5.1. Display of area functions and energy value for spoken vowels in Hindi.



(c) vowel /i/



(d) vowel /u/

Fig. 5.1. Display of area functions and energy value for spoken vowels in Hindi.

CHAPTER 6

SUMMARY AND CONCLUSIONS

The objective of this project is to develop a simple, computer based speech training aid using realistic vocal tract shape, pitch, and energy display. This can help deaf children to learn the movement of articulators while speaking, since realistic vocal tract shape display can give the actual position of tongue, lip, jaw, etc. while uttering speech sounds and pitch, and energy display can provide information about pitch, intonation, and stress patterns (the way in which the syllables are stressed).

In the first stage of development of the aid, a system for analysing the speech waveform in off-line mode and displaying a realistic vocal tract shape, pitch and energy simultaneously, for a visual feedback is implemented in software. A realistic vocal tract shape is estimated using reflections coefficients which are obtained from linear predictive coding. The energy and pitch values are obtained from autocorrelation analysis of short segments of the discretized input. The system was successfully tested for Russian vowels (/a, e, i, u/) generated by Klatt-synthesizer and area functions of which were available from Fant (using X-ray data).

In final stage, speech is analysed in real-time using a system which consists of a PC, a DSP TMS-32010 evaluation module, extension and interface hardware developed in an earlier project. Using this system, a realistic form of vocal tract shape as well as energy values for the spoken vowels of duration upto one second are estimated and displayed on the PC. The system was tested for the sustained vowels in Hindi spoken by the author herself.

Presently the system does not provide pitch information, but one could write a separate program for pitch estimation using the same system. In order to obtain a simultaneous display of vocal tract shape, pitch and energy values, one could possibly use a multiprocessor system, such as the one recently developed by Rampal (1991). In the existing system, eventhough analysis is achieved in real-time, display is non-real-time, due to delays in transferring data from the DSP board to the PC and then to the graphics interface of the PC. A possible solution may involve use of two D/A converters on the DSP board for directly controlling a CRT.

APPENDIX-A

ACOUSTIC PHONETICS

Sounds in American English can be generally described in terms of 42 phonemes as given in Table 2.1. The four broad classes of sounds are vowels, diphthongs, semivowels, and consonants. Each of these classes may be further classified into sub-classes which are related to the manner, and place of articulation of the sound within the vocal tract.

A.1 VOWELS, DIPHTHONGS, SEMIVOWELS, AND WHISPER

Vowels are produced by exciting a fixed vocal tract with quasi-periodic pulses of air caused by vibration of the vocal cords. Different vowel sounds are produced by different vocal tract configurations and therefore characterised (in a simple manner) by the vocal tract resonances or the formants. However, even for perceptually equivalent vowels, there is a great deal of variability in the formant frequencies among speakers, mainly due to the differences in the individual vocal tracts.

Diphthongs are usually defined as sounds that are produced by smoothly moving the vocal tract configuration from the articulatory position of one vowel to (or toward) that of another. Therefore, in spectrograms, the diphthongs are characterized by smooth glide of formant frequencies. Semivowels are generally characterized by a gliding transition in vocal tract area function between adjacent phonemes, and therefore, their acoustic characteristics strongly depend on the context in which they occur.

Phoneme /h/, also known as whisper, is produced by exciting the vocal tract by a steady air flow i.e. the turbulent flow at the glottis without vibration of vocal cords. The characteristics of /h/ are invariably those of the vowel which follows /h/ since the vocal tract assumes the position for the following vowel during the production of /h/.

A.2 NASALS, STOPS, FRICATIVES, AND AFFRICATES

The nasal consonants /m, n, and ŋ/ are produced by the glottal excitation with the total consriction at some point along the oral passage in the vocal tract. The velum is also lowered and the air flows through the nasal tract with the radiation of sound at the nostrils. The three nasal consonants are distinguished by the place along the oral tract at which a total constriction is made.

Stops are transient sounds produced by plosive excitation. The voiced stop consonants (/b, d, g/) are produced by building up pressure behind a complete closure somewhere in the oral tract and then suddenly releasing the pressure. Usually, no energy is radiated during the closure duration. The particular stop produced depends on the location of the constriction. Because of the dynamical nature of stop sounds, their properties depend upon the following vowels.

For unvoiced stops (/p, t, k/), the vocal cords do not vibrate during the period of total closure. Following the closure period, there is a brief interval of frication (turbulent) excitation of the vocal tract. This is followed by a period of aspiration (steady air flow without the vocal cord vibration) before the onset of voicing for the following vowel. Duration and frequency content of the frication and aspiration vary greatly with the stop consonant.

Unvoiced fricatives are produced by exciting the vocal tract by a steady air flow which becomes turbulent in the region of a constriction in the vocal tract. The location of the constriction determines the fricative sound produced. As the source of the excitation is at the constriction, the back cavity introduces antiresonances in the spectrum. For voiced fricatives, the excitation is different in that the vocal cords are vibrating. Thus, there is a quasiperiodic excitation source at the glottis and a noise like excitation source at the point of constriction.

The affricates are dynamically sounds that can be generally modelled as a stop followed by a fricative with the same place of constriction. The unvoiced affricate /tsh/ may be modelled as a concatenation of /t/ and /sh/, and the voiced affricate /dzh/ as a concatenation of /d/ and /zh/.

APPENDIX B

LINEAR PREDICTIVE CODING

B.1 INTRODUCTION

In this project linear predictive coding technique is used for speech analysis. Linear prediction is a mathematical model of the speech signal that assumes the speech signal as the output of a linear, time invariant, recursive filter excited by either a sequence of quasiperiodic pulses or a white-noise source. In this, the current sample of speech is predicted by a linear combination of previous known speech samples. Linear predictive coding technique involves the concept of short time analysis of speech to extract the relevant parameters of speech that remain constant over a short interval of time.

B.2 ALL-POLE MODEL OF VOCAL TRACT FILTER

Linear prediction can be used to obtain the transfer function of the vocal tract (Rabiner & Schafer, 1978). In the all pole model of vocal tract filter, the signal s(n) is assumed to be a linear combination of past output values and the present input u(n). This model is shown in Fig. B.1.

For applying time series analysis, continuous-time signal s(t) is sampled with a sampling interval T (= 1/Fo, where Fo is the sampling frequency) to obtain a discret-time signal s(nT). Here the signal s(n)= s(nT) is given as,

$$s(n) = -\sum_{k=1}^{p} a_k\, s(n-k) + G\, u(n) \qquad \text{(B.1)}$$

where G is the gain factor.

The transfer function of an all pole model is given by

$$H(z) = \frac{G}{1 + \sum_{k=1}^{p} a_k z^{-k}} \tag{B.2}$$

Assuming s(n) as a sample of a random process and using least square approach, the predictor coefficients a_k and the gain factor G can be determined (Makhoul, 1975). Let the predicted signal be s(n). Then

$$\tilde{s}(n) = - \sum_{k=1}^{p} a_k s(n-k) \tag{B.3}$$

The error between the actual value s(n) and the predicted value $\tilde{s}(n)$ is given as

$$e(n) = s(n) - \tilde{s}(n) = s(n) + \sum_{k=1}^{p} a_k s(n-k) \tag{B.4}$$

The error e(n) is also a sample of a random process. The expected value of the square of the error is minimized in the least squares method.

$$E = \mathcal{E}[e^2(n)] = \mathcal{E}[s(n) + \sum_{k=1}^{p} a_k s(n-k)]^2 \tag{B.5}$$

E is minimized by setting

$$\delta E / \delta a_i = 0 \qquad 1 \le i \le p$$

Hence we get

$$\sum_{k=1}^{p} a_k \mathcal{E}[s(n-k), s(n-i)]$$

$$= - \mathcal{E}[s(n), s(n-i)] \qquad 1 \le i \le p \tag{B.6}$$

The minimum average error is given by

$$E_p = \mathcal{E}[s^2(n)] + \sum_{k=1}^{p} a_k \mathcal{E}[s(n), s(n-k)] \tag{B.7}$$

Assuming that the speech signal remains constant for a short interval of time, for a stationary process s(n),

$$R(i-k) = E\,[s(n-k),\ s(n-i)] \qquad \text{(B.8)}$$

where R(i) is the autocorrelation of the process.

B.3 COMPUTATION OF PREDICTOR PARAMETERS

For a stationary and ergodic process, the autocorrelation can be approximated as a time average instead of ensemble average,

$$R(i) = \Sigma\ \ s(n)\ s(n-i) \qquad 1 \le i \le p$$

Hence Eqn. B.6 can be written as

$$-R(i) = \sum_{k=1}^{p} a_k\, R(i-k) \qquad 1 \le i \le p$$

The predictor coefficients a_k, $1 \le k \le p$ can be computed by solving a set of p equations with p unknowns.

The reflection coefficients can be computed from autocorrelation coefficients by Le-roux-Gueguen (L-G) algorithm without extracting polynomial coefficients of the transfer function. The intermediate variables are less than unity. Hence L-G algorithm can be implemented in the fixed point arithmetic (Leroux & Gueguen, 1977).

As shown in Fig. B.2, the central kernel is a lattice structure (Morris, 1983). This can be represented as

$$Y_1(I) = R(I) \qquad I = 1, \ldots., P.$$

$$B_1(I) = R(I) \qquad I = 0, \ldots., P.$$

$$F_K(I) = -\ Y_I(I)/B_I(I-1)$$

$$Y_{J+1}(I) = Y_J(I) + F_K(J)\ B_J(I-1)$$
$$B_{J+1}(I) = B_J(I-1) + F_K(J)\ Y_J(I)$$
$$\text{for } I = 1, 2, \ldots, P.$$
$$\text{for } J = 1, 2, \ldots, I-1.$$

$F_K(I)$ are the reflection coefficients.

The transfer function of an idealistic acoustic tube is equivalent to that of the linear prediction model. The acoustic tube is terminated at the lips by an infinite-length, infinite-area tube, and the transfer function of the inverse tube is defined as the ratio of the forward volume velocity at the point of excitation to the radiant volume velocity (Crichton & Fallside, 1974). The area A(n) at stage n is given by,

$$A(n) = A(n-1)\ [1+F_K(n)]\ /\ [1-F_K(n)] \qquad \text{(B.9)}$$

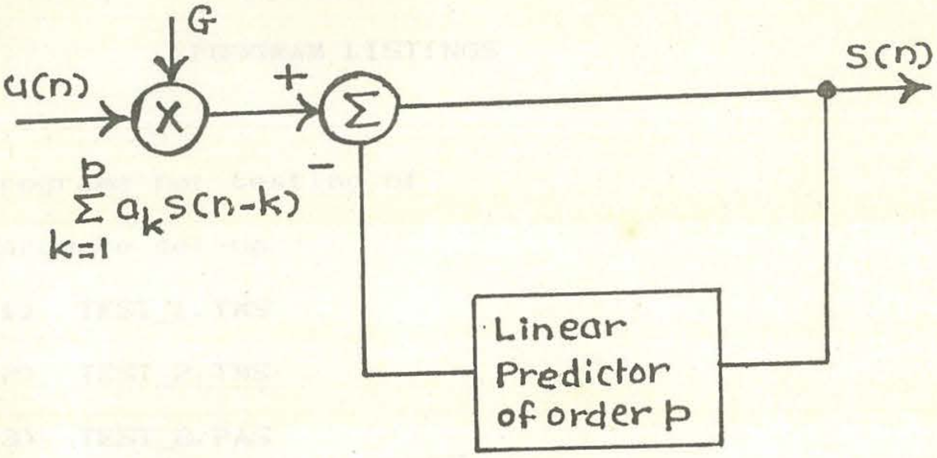


Fig. B.1. All-pole model of vocal tract filter. Adapted from Crichton & Fallside (1974).

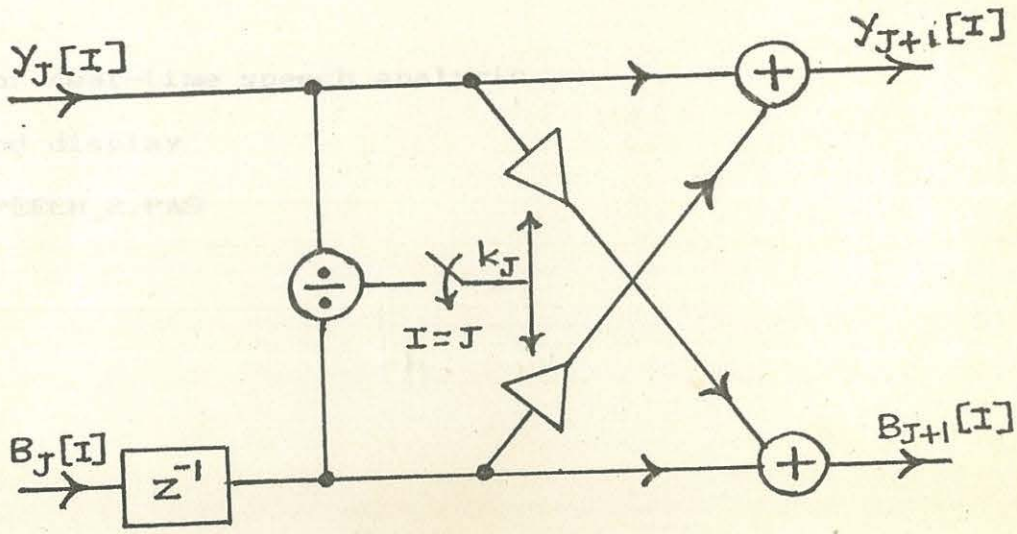


Fig. B.2. Kernel used in L-G algorithm. Adapted from (Morris, 1983).

APPENDIX C

PROGRAM LISTINGS

C.1 Programs for testing of Hardware set-up

(1) TEST_1.TMS

(2) TEST_2.TMS

(3) TEST_2.PAS

(4) TEST_3.TMS

C.2 For off-line speech analysis and display

INP.PAS

SPEECH_1.PAS

C.3 For real-time speech analysis and display

SPEECH_2.PAS

PROGRAMS FOR HARDWARE TESTING

```
*>
**PROGRAM TEST_1.TMS
ZERO EQU 0
ONE EQU 1
DATA EQU 2
ADDR EQU 3
COUNT EQU 4
PULSE EQU 5
*
    B LL1
    B ISR
ISR IN DATA,2       ;INPUT DATA FROM ADC
    ZALS ADDR
    TBLW DATA       ;STORE ADC DATA IN PROG MEMORY
    ADD ONE
    SACL ADDR
    ZALS COUNT      ;NO. OF DATA POINTS
    SUB ONE
    SACL COUNT
    BNZ LL2
    OUT ZERO,4      ;STOP CONVERSION PULSES
    RET
LL2 EINT
    RET
LL1 DINT
    OUT PULSE,4     ;START CONVERSION PULSES
    EINT
LL3 B LL3
*<
```

```
*PROGRAM TEST_2.TMS
*FOR DATA TRANSFER BETWEEN
*TMS-32010 AND PC
*>X0 EQU 0
  X1 EQU 1
  X2 EQU 2
  X3 EQU 3
**
*DATA TRANSFER FROM
*TMS-32010 TO PC
     ZALS X0   ;X0=COUNT
LL1  IN X1,0   ;CLEAR BIO
LL2  BIOZ LL3
     B LL2
LL3  OUT X2,2
     SUB X3
     BNZ LL1
**
*DATA TRANSFER FROM
*PC TO TMS-32010
     LAR 0,X2
     LARP 0
     ZALS X0
     IN X1,0   ;CLEAR BIO
LL4  BIOZ LL5
     B LL4
LL5  IN *+,0
     SUB X3
     BNZ LL4
     END
*<
```

```
PROGRAM TEST_2;
<FOR THE TRANSFER OF DATA BETWEEN DATA MEMORY
 OF TMS-32010 AND PC>
LABEL 1,2;
VAR
   I,J    :INTEGER;
   A,B    :ARRAY[1..10] OF BYTE;
BEGIN
    FOR I:=1 TO 10 DO
    BEGIN
        A[I]:=0;
        B[I]:=0;
    END;
    <DATA TRANSFER FROM TMS-32010 TO PC>
<    FOR J:=1 TO 10 DO
    BEGIN
        PORT[$319]:=0;
1:      I:=PORT[$320] AND ($2);
        IF  I=2  THEN
        A[J]:=PORT[$318]
        ELSE
        GOTO 1;
    END;>
    <DATA TRANSFER FROM PC TO TMS-32010 >
    FOR J:= 1 TO 10 DO
    BEGIN
        PORT[$319]:=0;
2:      I:=PORT[$320] AND ($2);
        IF  I=2  THEN
        PORT[$319]:=B[J]
        ELSE
        GOTO 2;
    END;
END.
```

```
*PROGRAM TEST_3.TMS
*FOR DATA TRANSFER BETWEEN
*EXTENDED DATA MEMORY (EDM) &
*DATA MEMORY OF TMS-32010
*>ADR EQU 0
  ONE EQU 1
  COUNT EQU 2           ;SET COUNT=COUNT1
  COUNT1 EQU 3
  DATA EQU 4
  BACK EQU 5
*
       OUT ADR,5        ;STARTING ADD FOR EDM
LOOP1  OUT DATA,6       ;DATA TRANSFER TO EDM WITH
                         INCREMENTATION OF ADDRESS
                         LOCATION
       ZALS COUNT
       SUB ONE
       SACL COUNT
       BNZ  LOOP1
       LAR 0,BACK      ;AR0=STARTING ADD FOR DATA
                         COMING FROM EDM TO DATA MEMORY
       LARP 0
       OUT ADR,5
       LACK COUNT1
LOOP2  IN *+,6          ;DATA TRANSFER FROM EDM TO
                         DATA MEMORY
       SUB ONE
       BNZ LOOP2
```

INP·PAS
SPEECH_1.PAS

```
PROGRAM INP;
VAR
      I                    :INTEGER;
      INVAR                :TEXT;
      OUTVAR               :FILE OF INTEGER;
      INFILE,OUTFILE       :STRING[14];
      S                    :ARRAY[0..4000] OF INTEGER;
 BEGIN
      WRITELN('input file name = ');
      READLN(INFILE);
      ASSIGN(INVAR,INFILE);
      WRITELN('output file name = ');
      READLN(OUTFILE);
      ASSIGN (OUTVAR,OUTFILE);
      RESET(INVAR);
      REWRITE(OUTVAR);
      FOR I:= 0 TO 4000 DO
      BEGIN
         READLN(INVAR,S[I]);
         WRITE(OUTVAR,S[I]);
      END;
      READLN;
         CLOSE(INVAR);
         CLOSE(OUTVAR);
 END.
```

```
PROGRAM SPEECH_1;
USES GRAPH,DOS,CRT;
LABEL 5,100,110,120;
TYPE
    DARRAY=ARRAY[1..15]OF REAL;
    SARRAY=ARRAY[-1..200]OF INTEGER;
CONST
     FRCNT=60;
VAR
   SP                    :ARRAY[0..300]OF REAL;
   LR,SR                 :ARRAY[0..300]OF REAL;
   S1,Y1                 :ARRAY[0..300]OF REAL;
   S                     :SARRAY;
   E                     :ARRAY[-1..300]OF REAL;
   WC                    :ARRAY[0..300]OF REAL;
   FK                    :ARRAY[0..FRCNT,1..15]OF REAL;
   AA                    :ARRAY[1..FRCNT,0..20]OF REAL;
   R,BI,BIM,BIM1,BII     :DARRAY;
   J,TS,GRAPHDRIVER,TL,NFMAX,
   GRAPHMODE,ERRORCODE,LAST,CODE,TIM,EXCODE,
   X,M,FR,FL,FORM,NF,DUMMY,TEMP,FS,SFNO,PAR,P,
   KK,WIND,PREMPH,Y      :INTEGER;
   I                     :LONGINT;
   K,C,KPRE,DSC,EMAX     :REAL;
   BIG1,BIG2,BIG3,PITS,D5,
   PITCHS,PITL,PITCHL    :ARRAY[1..FRCNT]OF REAL;
   BIG                   :ARRAY[1..FRCNT]OF INTEGER;
   PS,PL ,PS1            :STRING[40];
   YI                    :REAL;
   FILEPOS               :LONGINT;
   FILEVAR,OFILEVAR,OUTFILE,
   ENVAR,PIVAR,INFILE    :TEXT;
   DFILEVAR              :FILE OF INTEGER;
   DFILE,OFILE,ENFILE,
   PIFILE,COORD          :STRING[14];
   CH,C_H,CHH,CHHH,DDUMMY :CHAR;
   CHS,PREEMPH,CH1,CH2:STRING[5];
   D1,D2                 :ARRAY[1..200] OF INTEGER;
   D3,D4                 :ARRAY[0..50] OF INTEGER;

{**** PROCEDURE AUTOCORRELATION BEGINS *******}

PROCEDURE AUTOCORRELATION(FLAG1:INTEGER);
VAR
    II,MM        :INTEGER;
BEGIN
    IF FLAG1=1 THEN
    BEGIN
        BIG3[I]:=0;
        FOR II:=0 TO FL-1 DO
        BEGIN
            SR[II]:=0.0;
            FOR MM:=II TO FL-1 DO
            BEGIN
                SR[II]:=SR[II]+SP[MM]
                                    *SP[(MM-II)];
            END;
            IF (SR[II])>BIG3[I] THEN BIG3[I]:=SR[II];
        END;
        IF BIG3[I] = 0 THEN PITCHS[I] := 0
        ELSE
        BEGIN
            PITS[I]:=0;
            FOR II:=25 TO FL-1 DO
            BEGIN
```

```
            THEN
            BEGIN
                  PITS[I]:=(SR[II]);
                  PITCHS[I]:=II;
            END;
         END;
         IF PITS[I]/BIG3[I] >0.3
         THEN
            PITCHS[I]:=FS/PITCHS[I]
         ELSE
            PITCHS[I]:=0;
      END;
   END
   ELSE
   BEGIN
        BIG2[I]:=0;
        FOR II:=0 TO FL-1 DO
        BEGIN
            LR[II]:=0.0;
            FOR MM:=II TO FL-1 DO
            BEGIN
                 LR[II]:=LR[II]+E[MM]
                                  *E[(MM-II)];
            END;
            IF (LR[II])>BIG2[I] THEN BIG2[I]:=LR[II];
        END;
        PITL[I]:=0;
        IF BIG2[I]=0 THEN PITCHL[I]:=0
        ELSE
        BEGIN
            FOR II:=25 TO FL-1 DO
            BEGIN
                IF (LR[II])>PITL[I]
                THEN
                BEGIN
                    PITL[I]:=(LR[II]);
                    PITCHL[I]:=II;
               END;
          END;
          IF PITL[I]/BIG2[I] >0.3
          THEN
             PITCHL[I]:=FS/PITCHL[I]
          ELSE
          PITCHL[I]:=0;
        END;
    END;
END;

(***** PROCEDURE AUTOCORRELATION ENDS *******)

(PROCEDURE Le-roux-Gueguen ALGORITHM BEGINS ******)

PROCEDURE LGSOL;
VAR
   II,IM1,JJ      :INTEGER;
BEGIN
     IF SR[0]=0 THEN
     BEGIN
         FOR J:= 1 TO P DO
         BEGIN
             FK[I,J]:=FK[I-1,J]
         END;
     END
     ELSE
     BEGIN
```

```
        BIM1[1]:=SR[1];
        BIM1[2]:=SR[0]+FK[I,1]*SR[1];
        FOR II:=2 TO P DO
        BEGIN
            YI:=SR[II];
            BI[1]:=YI;
            IM1:=II-1;
            FOR JJ:=1 TO IM1 DO
            BEGIN
                BI[JJ+1]:=BIM1[JJ]+FK[I,JJ]*YI;
                YI:=YI+FK[I,JJ]*BIM1[JJ];
                BIM1[JJ]:=BI[JJ];
            END;
            FK[I,II]:=-YI/BIM1[II];
           {WRITELN('FK[',I,',',II,']=',FK[I,II]);}
            BIM1[II+1]:=BIM1[II]+FK[I,II]*YI;
            BIM1[II]:=BI[II];
        END;
    END;
END;

{***** PROCEDURE Le-roux-Gueguen ENDS ******}

{***** PROCEDURE LPCERROR BEGINS ***********}

PROCEDURE LPCERROR;
VAR
   II,KK,JJ    :INTEGER;
BEGIN
    BIM[1]:=SP[0];
    FOR KK:=2 TO P DO BIM[KK]:=0;
    FOR II:=1 TO FL-1 DO
    BEGIN
        YI:=SP[II];
        BII[1]:=SP[II];
        FOR  JJ:=2 TO P+1 DO
        BEGIN
            BII[JJ]:=BIM[JJ-1]+FK[I,JJ-1]*YI;
            YI:=YI+FK[I,JJ-1]*BIM[JJ-1];
            BIM[JJ-1]:=BII[JJ-1]
        END;
        E[II]:=YI;
    END;
    E[0]:=SP[0];
    BIG1[I]:=ABS(E[0]);
    FOR II:=1 TO FL-1 DO
    BEGIN
        IF ABS(E[II])>BIG1[I]
        THEN
        BEGIN
            BIG1[I]:=ABS(E[II]);
        END;
    END;
END;

{******* PROCEDURE LPCERROR ENDS *********}

{******* PROCEDURE DISPLAY BEGINS ********}
PROCEDURE DISPLAY;
LABEL 50,51,70,80;
VAR
   Z1,Z2,Z3,Z4,J1,J2,XX,YY1,XX1,XX2,SRR,XX11       :INTEGER;
   YY,YY11                                         :LONGINT;
PROCEDURE DRAW_LINES(YST,YEND,XST,COLOR:INTEGER);
BEGIN
```

```
    FOR J:=YST TO YEND DO
    LINE(XST,J,XST-10,J);
END;

BEGIN
    ASSIGN(INFILE,'COORD.PAS ');
    RESET(INFILE);
    FOR J := 1 TO 125 DO
    BEGIN
        READLN(INFILE,D1[J],D2[J]);
        D1[J]:=150-D1[J];
    END;
    FOR J := 126 TO 137 DO
    BEGIN
        READLN(INFILE,D3[J-125],D4[J-125]);
        D3[J-125]:=150-D3[J-125];
    END;
    CLOSE(INFILE);
    GRAPHDRIVER:= DETECT;
    INITGRAPH(GRAPHDRIVER,GRAPHMODE,'C:\');
    DIRECTVIDEO:=FALSE;
    ERRORCODE:=GRAPHRESULT;
    IF ERRORCODE<>GROK THEN
    BEGIN
        WRITE('GRAPHICS ERROR: ');
        WRITELN(GRAPHERRORMSG(ERRORCODE));
        WRITELN('PROGRAM ABORTED');
        HALT(1);
    END;

    I:=SFNO;
    FOR J := 1 TO 64 DO
    LINE(2*D1[J],D2[J],2*D1[J+1],D2[J+1]);
    FOR J := 66 TO 76 DO
    LINE(2*D1[J],D2[J],2*D1[J+1],D2[J+1]);
    FOR J := 78 TO 105 DO
    LINE(2*D1[J],D2[J],2*D1[J+1],D2[J+1]);
    FOR J := 107 TO 124 DO
    LINE(2*D1[J]+2,D2[J],2*D1[J+1]+2,D2[J+1]);
    LINE(450,117,450,197);MOVETO(580,117);LINETO(580,197);
    YY1:=117; XX1 := 583;
    FOR J:=0 TO 4 DO
    BEGIN
        STR((400-J*100),CH1);
        MOVETO(XX1+4,YY1-5);OUTTEXT(CH1);
        YY1 := YY1+20;
    END;
    YY1 := 117;
    FOR J := 0 TO 8 DO
    BEGIN
        PUTPIXEL(581,YY1+J*10,WHITE);
        PUTPIXEL(582,YY1+J*10,WHITE);
    END;
    YY1:=117;XX1:=450;
    FOR J:=0 TO 5 DO
    BEGIN
        STR((100-J*20),CH1);
        MOVETO(XX1+5,YY1-5);OUTTEXT(CH1);
        YY1:=YY1+16;
    END;
    YY1 := 117;
    FOR J := 0 TO 10 DO
    BEGIN
        PUTPIXEL(451,YY1+J*8,WHITE);
        PUTPIXEL(452,YY1+J*8,WHITE);
```

```
Z1:= ROUND((D5[I]/EMAX)*100);
{    Z1:= ROUND(D5[I]/100000); }
Z2:=ROUND(PITCHS[I]);
STR(Z1,PS);MOVETO(368,160);
OUTTEXT('E = ');OUTTEXT(PS);
STR(Z2,PS1);MOVETO(500,160);
OUTTEXT('F0 = ');OUTTEXT(PS1);
SRR:=ROUND(4*Z1/5);
Z2:=197-ROUND(Z2/5);
DRAW_LINES(Z2,197,579,1);
Z1:=197-SRR;
DRAW_LINES(Z1,197,449,1);
MOVETO(400,0);OUTTEXT('File:');OUTTEXT(DFILE);
MOVETO(400,25);
LINETO(400+(P+1)*15,25);
MOVETO(400,110);
LINETO(400+(P+1)*15,110);
XX:=400;
FOR J:= 1 TO 12 DO
BEGIN
    YY1:=109;
    YY:=YY1-ROUND(DSC*10*AA[I,P-J+1]/6);
    LINE(XX,YY,XX+15,YY);
    IF J <> 12 THEN
    BEGIN
        YY11:=YY1-ROUND(
        DSC*10*AA[I,P-J]/6);
        LINE(XX+15,YY,XX+15,YY11);
    END;
    XX:=XX+15;
END;
FOR J := 0 TO 7 DO
BEGIN
    IF J=0 THEN
    BEGIN
        XX1:=244;YY1:=105;
        END
        ELSE
        BEGIN
            XX1:=2*D3[J];YY1:=D4[J]+ROUND(DSC*AA[I,J]/(2.0));
        END;
        YY11 := D4[J+1]+ROUND(DSC*AA[I,J+1]/(2.0));
        XX11 := 2*D3[J+1] ;
        SETCOLOR(1);
        LINE(XX1,YY1,XX11,YY11);
   END;
FOR J := 8 TO 11 DO
BEGIN
    IF J=8 THEN
    BEGIN
        XX1:=2*D3[J];YY1:=D4[J]+ROUND(DSC*AA[I,J]/(2.0));
    END
    ELSE
    BEGIN
        YY1 := D4[J];
        XX1 := 2*D3[J]+2*ROUND(DSC*AA[I,J]/(2.0))+10;
    END;
    YY11 := D4[J+1];
    XX11 := 2*D3[J+1]+2*ROUND(DSC*AA[I,J+1]/(2.0))+10;
    LINE(XX1,YY1,XX11,YY11);
    SETCOLOR(1);
END;
Z4:=Z2;Z3:=Z1;
MOVETO(400,8);STR(I,CH1); SETCOLOR(1);
OUTTEXT('Frame:');OUTTEXT(CH1);
```

```
    OUTTEXT(', t:');OUTTEXT(CH1);OUTTEXT(' ms. ');J1:=0;
50: Z2:=Z4;Z1:=Z3;
51: MOVETO(400,16);SETCOLOR(1);OUTTEXT('Exit(Y/N)?');
    REPEAT
         BEGIN
         END;
    UNTIL KEYPRESSED;
    CH:=READKEY;
    IF (CH='Y') OR (CH='y')THEN
    BEGIN
        EXCODE:=1;EXIT;
    END;
    MOVETO(400,16);SETCOLOR(0);OUTTEXT('Exit(Y/N)?');
    MOVETO(400,16);SETCOLOR(1);OUTTEXT('Refresh:(''C'' or ''S'')?');
    REPEAT
         BEGIN
         END;
    UNTIL KEYPRESSED;
    C_H:=READKEY;
    MOVETO(400,16);SETCOLOR(0);OUTTEXT('Refresh:(''C'' or ''S'')?');
    IF (C_H<>'C') AND (C_H<>'c') AND (C_H<>'S') AND (C_H<>'s') THEN
    BEGIN
        GOTO 51;
    END;
    MOVETO(400,16);SETCOLOR(1);OUTTEXT('Next(E,+,-,=)?');
    STR(SFNO,CH1);
    OUTTEXT(' -->  ');OUTTEXT(CH1);
    STR((SFNO+NF-1),CH1);OUTTEXT('-');OUTTEXT(CH1);
80: REPEAT
         BEGIN
         END;
    UNTIL KEYPRESSED;
    CH:=READKEY;
    IF (CH='E') OR (CH<>'+') AND (CH<>'-') AND (CH<>'=') THEN
    BEGIN
    MOVETO(400,16);SETCOLOR(0);OUT.TEXT('Next(E,+,-,=)?');
    STR(SFNO,CH1);
    OUTTEXT(' -->  ');SETCOLOR(0);OUTTEXT(CH1);
    STR((SFNO+NF-1),CH1);SETCOLOR(0);OUTTEXT('-');OUTTEXT(CH1);
    GOTO 50;
    END;
    Z2:=Z4;Z1:=Z3;
    IF CH = '+' THEN
    BEGIN
         J1:=I+1;
         DDUMMY:='0';
    END;
    IF CH = '-' THEN
    BEGIN
         J1:=I-1;
         DDUMMY:='0';
    END;
    IF (CH='+') OR (CH='-') THEN
    BEGIN
         IF (J1< SFNO) OR (J1 > (SFNO+NF-1)) THEN
         BEGIN
              SOUND(800);
              DELAY(500);
              NOSOUND;
              IF CH='+' THEN J1:=J1-1;
              IF CH='-' THEN J1:=J1+1;
              GOTO 80;
         END;
    END;
    IF CH = '=' THEN
```

```
    WHILE NOT KEYPRESSED DO
    BEGIN
    END;
    CHH :=READKEY;{Z2:=Z4;Z1:=Z3;}
    VAL(CHH,J1,CODE);
    WHILE NOT KEYPRESSED DO
    BEGIN
    END;
    CHHH :=READKEY;{Z2:=Z4;Z1:=Z3;}
    VAL(CHHH,J,CODE);
    J1:=10*J1+J;
    IF (J1 < SFNO) OR (J1 > (SFNO+NF-1)) THEN EXIT;
END;
MOVETO(400,8);STR(I,CH1); SETCOLOR(0);
OUTTEXT('frame:');OUTTEXT(CH1);
TIM:=I*10;STR(TIM,CH1);
OUTTEXT(', t:');OUTTEXT(CH1);OUTTEXT(' ms. ');
MOVETO(400,8);STR(J1,CH1); SETCOLOR(1);
OUTTEXT('Frame:');OUTTEXT(CH1);
TIM:=J1*10;STR(TIM,CH1);
OUTTEXT(', T:');OUTTEXT(CH1);OUTTEXT(' ms. ');
SETCOLOR(0);
MOVETO(500,160);OUTTEXT('F0 = ');OUTTEXT(PS1);
MOVETO(368,160);OUTTEXT('E = ');OUTTEXT(PS);
Z4:=ROUND(PITCHS[J1]);STR(Z4,PS1);
MOVETO(500,160);SETCOLOR(1);OUTTEXT('F0 = ');OUTTEXT(PS1);
Z4:=197-ROUND(Z4/5);
IF PITCHS[J1] >= PITCHS[I] THEN
BEGIN
    DRAW_LINES(Z4,Z2,579,1);
END
ELSE
BEGIN
    DRAW_LINES(Z2,Z4,579,0);
END;
Z3:=ROUND((D5[J1]/EMAX)*100);STR(Z3,PS);
MOVETO(368,160);SETCOLOR(1);OUTTEXT('E = ');OUTTEXT(PS);
SRR:=ROUND(4*Z3/5);Z3:=197-SRR;
IF Z3 < Z1 THEN
BEGIN
    DRAW_LINES(Z3,Z1,449,1);
END
ELSE
BEGIN
    DRAW_LINES(Z1,Z3,449,0);
END;
WHILE NOT KEYPRESSED DO
BEGIN
END;
CHH:=READKEY;
FOR J := 0 TO 7 DO
BEGIN
    IF J=0 THEN
    BEGIN
        XX1:=244;YY1:=105;
    END
    ELSE
    BEGIN
        XX1:=2*D3[J];YY1:=D4[J]+ROUND(DSC*AA[I,J]/(2.0));
    END;
    YY11 := D4[J+1]+ROUND(DSC*AA[I,J+1]/(2.0));
    XX11 := 2*D3[J+1] ;
    SETCOLOR(0);
    LINE(XX1,YY1,XX11,YY11);
    IF (C_H = 'S') OR (C_H='s') THEN
```

```
        WHILE NOT KEYPRESSED DO
        BEGIN
        END;
        CH:=READKEY;
    END;
    IF J=0 THEN
    BEGIN
        XX1:=244;YY1:=105;
    END
    ELSE
    BEGIN
        XX1:=2*D3[J];YY1:=D4[J]+ROUND(DSC*AA[J1,J]/(2.0));
    END;
    YY11 := D4[J+1]+ROUND(DSC*AA[J1,J+1]/(2.0));
    XX11 := 2*D3[J+1] ;
    SETCOLOR(1);
    LINE(XX1,YY1,XX11,YY11);
    IF (C_H = 'S') OR (C_H='s') THEN
    BEGIN
        WHILE NOT KEYPRESSED DO
        BEGIN
        END;
        CH:=READKEY;
    END;
END;
FOR J := 8 TO 11 DO
BEGIN
    IF J=8 THEN
    BEGIN
        XX1:=2*D3[J];YY1:=D4[J]+ROUND(DSC*AA[I,J]/(2.0));
    END
    ELSE
    BEGIN
        YY1 := D4[J];
        XX1 := 2*D3[J]+2*ROUND(DSC*AA[I,J]/(2.0))+10;
    END;
    YY11 := D4[J+1];
    XX11 := 2*D3[J+1]+2*ROUND(DSC*AA[I,J+1]/(2.0))+10;
    SETCOLOR(0);
    LINE(XX1,YY1,XX11,YY11);
    IF (C_H = 's') OR (C_H='S') THEN
    BEGIN
        WHILE NOT KEYPRESSED DO
        BEGIN
        END;
        CH:=READKEY;
    END;
    IF J=8 THEN
    BEGIN
        XX1:=2*D3[J];YY1:=D4[J]+ROUND(DSC*AA[J1,J]/(2.0));
    END
    ELSE
    BEGIN
        YY1 := D4[J];
        XX1 := 2*D3[J]+2*ROUND(DSC*AA[J1,J]/(2.0))+10;
    END;
    YY11 := D4[J+1];
    XX11 := 2*D3[J+1]+2*ROUND(DSC*AA[J1,J+1]/(2.0))+10;
    SETCOLOR(1);
    LINE(XX1,YY1,XX11,YY11);
    IF (C_H = 'S') OR (C_H='s') THEN
    BEGIN
        WHILE NOT KEYPRESSED DO
        BEGIN
        END;
```

```
        END;
    END;
    XX:=400;
    FOR J:= 1 TO 12 DO
    BEGIN
        YY1:=109;YY11:=0;YY:=0;
        YY:=YY1-ROUND(DSC*10*AA[I,P-J+1]/6);
        SETCOLOR(0);
        LINE(XX,YY,XX+15,YY);
        IF J <> 12 THEN
        BEGIN
            YY11:=YY1-ROUND(DSC*10*AA[I,P-J]/6);
            SETCOLOR(0);
            LINE(XX+15,YY,XX+15,YY11);
        END;
        IF (C_H = 'S') OR (C_H='s') THEN
        BEGIN
            WHILE NOT KEYPRESSED DO
            BEGIN
        END;
        CH:=READKEY;
        END;
        YY:=0;YY11:=0;
        YY:=YY1-ROUND(DSC*10*AA[J1,P-J+1]/6);
        SETCOLOR(1);
        LINE(XX,YY,XX+15,YY);
        IF J <> 12 THEN
        BEGIN
            YY11:=YY1-ROUND(DSC*10*AA[J1,P-J]/6);
            SETCOLOR(1);
            LINE(XX+15,YY,XX+15,YY11);
        END;
        IF (C_H = 'S') OR (C_H='s') THEN
        BEGIN
            WHILE NOT KEYPRESSED DO
            BEGIN
            END;
            CH:=READKEY;
        END;
        XX:=XX+15;
    END;
    IF (C_H = 'S') OR (C_H = 's') THEN
    BEGIN
        SOUND(800);
        DELAY(500);
        NOSOUND;
    END;
    I:=J1;
    IF DDUMMY = '0' THEN
    BEGIN
        DDUMMY:='2';
    END;
    GOTO 80;
END;

{****** PROCEDURE DISPLAY ENDS *********}

PROCEDURE MENU;
VAR
   XM1,YM1,XM2,YM2       :BYTE;
BEGIN
    WINDOW(1,1,80,11);
    CLRSCR;
    GOTOXY(10,1);
    WRITELN('* SPEECH ANALYSIS & DISPLAY PROGRAM MENU *');
```

```
    GOTOXY(XM1,YM1);
    WRITE('1) DATA FILE:              ',DFILE);
    GOTOXY(XM1,YM1+1);
    WRITE('2) DATA FILE FORMAT:       ',FORM);
    GOTOXY(XM1,YM1+2);
    WRITE('3) SAMPLING FREQ:          ',FS);
    GOTOXY(XM1,YM1+3);
    WRITE('4) WINDOW TYPE:            ',WIND);
    GOTOXY(XM1,YM1+4);
    WRITE('5) PRE-EMP. COEFF:         ',K:4:2);
    GOTOXY(XM1,YM1+5);
    WRITE('6) PREDICTOR ORDER:        ',P);
    GOTOXY(XM1,YM1+6);
    WRITE('7) TO QUIT MENU');
    XM1 := 40; YM1 := 3;
    GOTOXY(XM1,YM1);
    WRITE(' 8) FRAME LENGH:           ',FL);
    GOTOXY(XM1,YM1+1);
    WRITE(' 9) FRAME RATE:            ',FR);
    GOTOXY(XM1,YM1+2);
    WRITE('10) PRE-EMPHASIS:          ',PREEMPH);
    GOTOXY(XM1,YM1+3);
    WRITE('11) STARTING FRAME:        ',SFNO);
    GOTOXY(XM1,YM1+4);
    WRITE('12) TOTAL FRAMES:          ',NF);
    GOTOXY(XM1,YM1+5);
    WRITE('13) DISPLAY SCALE:         ',DSC:4:2);
    GOTOXY(XM1,YM1+6);
    WRITE('14) TO EXIT PROGRAM');
    GOTOXY(1,YM1+7);
    WRITE('*********************************');
    WRITE('*********************************');
END;

(****** MAIN PROGRAM BEGINS ************)

BEGIN
    P:=12; DFILE:='SPEECH.DAT'; FS:=10000; FORM:=1;
    DSC:=1; WIND:=1; K:=0.9; FL:=200; FR:=100;
    SFNO:=1; NF:=3; PREEMPH:='ON';
5:  TEXTCOLOR(15);
    WINDOW(1,1,80,25);
    CLRSCR;
    X := 19; Y := 1;
    GOTOXY(X,Y+6);
    WRITELN(' * SPEECH ANALYSIS & DISPLAY PROGRAM * ');
    GOTOXY(X,Y+7);
    WRITELN(' * FOR OFF-LINE SPEECH DATA * ');
    GOTOXY(X,Y+9);
    WRITELN(' ****************************************');
    GOTOXY(1,25);
    WRITELN('* FOR MENU PRESS ANY KEY *');
    WHILE NOT KEYPRESSED DO
    BEGIN
    END;
    CH := READKEY;

(******* SELECTION OF PARAMETERS ********)

110:MENU;
    WINDOW(1,12,80,25);
    GOTOXY(1,13);
    WRITE('PARAMETER CODE TO BE ALTERED: ');
    GOTOXY(35,13);
    READLN(CHS);
```

```
    FOR I:=1 TO 8 DO
    BEGIN
        GOTOXY(1,I);
        CLREOL;
    END;
    GOTOXY(1,1);
    CASE PAR OF
         1 :BEGIN
               WRITE('DATA FILE : ');
               READLN(DFILE);
               IF DFILE='' THEN DFILE:='SPEECH.DAT';
            END;
         2 :BEGIN
               WRITELN('DATA FILE FORMAT (1/2) : ');
               WRITELN;
               WRITELN;
               WRITE('FORMAT 1: FIRST LINE CONTAINS ');
               WRITELN('TOTAL NO. OF SAMPLES');
               WRITE('          NEXT A DATA SAMPLE IS ');
               WRITELN('STORED PER LINE');
               WRITE('FORMAT 2: FIRST LINE CONTAINS ');
               WRITELN('TOTAL NO. OF DATA LINES ');
               WRITE('          NEXT EIGHT DATA SAMPLES ');
               WRITELN('ARE STORED PER LINE');
               GOTOXY(30,1);
100:           READLN(CHS);
               VAL(CHS,FORM,CODE);
               IF (FORM<>1) AND (FORM<>2) THEN
               BEGIN
                   GOTOXY(30,1);
                   CLREOL;
                   GOTO 100;
               END;
            END;
         3 :BEGIN
               WRITE('SAMPLING FREQUENCY : ');
               GOTOXY(30,1);
               READLN(CHS);
               VAL(CHS,FS,CODE);
               WHILE (FS<1000) OR (FS>20000) DO
               BEGIN
                   GOTOXY(30,1);
                   CLREOL;
                   READLN(CHS);
                   VAL(CHS,FS,CODE);
               END;
            END;
         4 :BEGIN
               WRITELN('WINDOW TYPES ARE:');
               WRITELN;
               WRITELN('1) :HAMMING WINDOW:');
               WRITELN('2) :               :');
               WRITELN('3) :               :');
               WRITELN('4) :RECTANGULAR    :');
               WRITELN;
               WRITE('SELECT WINDOW : ');
               GOTOXY(30,8);
               READLN(CHS);
               VAL(CHS,WIND,CODE);
               WHILE (WIND<1)OR(WIND>4) DO
               BEGIN
                   GOTOXY(30,8);
                   CLREOL;
                   READLN(CHS);
                   VAL(CHS,WIND,CODE);
```

```
        IF WIND=1 THEN
        BEGIN
            GOTOXY(1,8);
            CLREOL;
            WRITE('HAMMING WINDOW IS : ');
            WRITE('WC[I] = (0.54-0.46*COS(2*PI*I/');
            WRITE('(FL-1)))');
        END
    END;
 5 :BEGIN
        WRITE('PRE-EMP. COEFFICIENT :');
        GOTOXY(30,1);
        READLN(CHS);
        VAL(CHS,K,CODE);
        WHILE (K<0) OR (K>0.99) DO
        BEGIN
            GOTOXY(30,1);
            CLREOL;
            READLN(CHS);
            VAL(CHS,K,CODE);
    END;
    END;
 6 :BEGIN
        WRITE('ORDER OF LINEAR PREDICTOR : ');
        GOTOXY(30,1);
        READLN(CHS);
        VAL(CHS,P,CODE);
        WHILE (P<8) OR (P>20) DO
        BEGIN
            GOTOXY(30,1);
            CLREOL;
            READLN(CHS);
            VAL(CHS,P,CODE);
        END;
    END;
 8 :BEGIN
        WRITE('FRAME LENGTH IN NO. OF SAMPLES :');
        GOTOXY(35,1);
        READLN(CHS);
        VAL(CHS,FL,CODE);
        WHILE (FL<100) OR (FL>200) DO
        BEGIN
            GOTOXY(35,1);
            CLREOL;
            READLN(CHS);
            VAL(CHS,FL,CODE);
        END;
    END;
 9 :BEGIN
        WRITE('FRAME RATE IN NO. OF SAMPLES :');
        GOTOXY(35,1);
        READLN(CHS);
        VAL(CHS,FR,CODE);
        WHILE (FR<50) OR (FR>200) DO
        BEGIN
            GOTOXY(35,1);
            CLREOL;
            READLN(CHS);
            VAL(CHS,FR,CODE);
        END;
    END;
 10:BEGIN
        WRITE('PRE-EMPHASIS APPLIED (ON/OFF) :');
        READLN(PREEMPH);
        WHILE (PREEMPH<>'ON') AND (PREEMPH<>'OFF') DO
```

```
                GOTOXY(35,1);
                CLREOL;
                READLN(PREEMPH);
            END;
        END;
    11:BEGIN
            WRITE('STARTING FRAME NUMBER :');
            READLN(CHS);
            VAL(CHS,SFNO,CODE);
            WHILE (SFNO<1) OR (SFNO>60) DO
            BEGIN
                GOTOXY(30,1);
                CLREOL;
                READLN(CHS);
                VAL(CHS,SFNO,CODE);
            END;
        END;
    12:BEGIN
            WRITE('TOTAL NO. OF FRAMES :');
            READLN(CHS);
            VAL(CHS,NF,CODE);
            WHILE (NF<1) OR (NF>FRCNT) DO
            BEGIN
                GOTOXY(30,1);
                CLREOL;
                READLN(CHS);
                VAL(CHS,NF,CODE);
            END;
        END;
    13:BEGIN
            WRITE('DISPLAY SCALE FACTOR :');
            READLN(CHS);
            VAL(CHS,DSC,CODE);
            WHILE (DSC>20)AND(DSC<1) DO
            BEGIN
                GOTOXY(30,1);
                CLREOL;
                READLN(CHS);
                VAL(CHS,DSC,CODE);
            END;
        END;
    14:BEGIN
            GOTOXY(1,13);
            CLREOL;
            WRITE('ARE YOU SURE YOU WANT TO EXIT(Y/N)?');
            CH := READKEY;
            IF (CH='Y') OR (CH='y') THEN EXIT;
        END;
ELSE
  BEGIN
      CLRSCR;
      GOTOXY(1,13);
      WRITE('PARAMETER SELECTION OVER(Y/N)?');
      WHILE NOT KEYPRESSED DO
      BEGIN
      END;
      CH := READKEY;
      IF (CH='Y') OR (CH='y') THEN GOTO 120;
  END
END;
GOTOXY(35,13);
CLREOL;
GOTO 110;

{****** PARAMETER SELECTION OVER *******}
```

```
(****** APPLY PRE-EMPHASIS AND WINDOWING *****)

120: GOTOXY(1,13);CLREOL;
     IF PREEMPH='ON'
     THEN KPRE:=K
     ELSE
        KPRE:=0;
     FOR I:= 0 TO 300 DO
     WC[I]:=0;
     IF WIND=1 THEN
     BEGIN
         C:=0.5;
         FOR I:=1 TO FL DO
         WC[I]:=C*(0.54-0.46*COS(2*PI*(I-1)/(FL-1)));
         WC[I-1]:=WC[I];
     END;
     EXCODE:=0;
     FOR J:=1 TO FRCNT DO
     D5[J]:=0;
     FOR J:=1 TO P DO FK[0,J]:=0;
     ASSIGN(DFILEVAR,DFILE);
     RESET(DFILEVAR);
     FOR J:=-1 TO 1000 DO
     S[J]:=0;
     FOR J:=0 TO 1000 DO
     SP[J]:=0;
     FOR I:=SFNO TO (NF+SFNO-1) DO
     BEGIN
         GOTOXY(1,13);
         CLREOL;
         GOTOXY(43,13);
         CLREOL;
         GOTOXY(30,13);
         WRITE('** ');
         TEXTCOLOR(143);
         WRITE('READING & PROCESSING FRAME-');WRITE(I);
         TEXTCOLOR(15);
         WRITE(' **');
         SEEK(DFILEVAR , (100*(I-1)));
         FOR J:=0 TO FL-1 DO
         BEGIN
             READ(DFILEVAR,S[J]);
         END;
(         FOR J:=0 TO FL-1 DO
         S[J-1]:=S[J];)
         IF WIND = 1  THEN
         BEGIN
             S[-1]:=0;
             FOR J:=0 TO FL-1 DO
             BEGIN
                 SP[J]:=WC[J]*(S[J]-KPRE*S[J-1]);
             END;
         END
         ELSE
         BEGIN
             FOR J:=0 TO FL-1 DO
             BEGIN
                 SP[J]:=S[J]-KPRE*S[J-1];
             END;
         END;
         AUTOCORRELATION(1);
         LGSOL;
         LPCERROR;
         AUTOCORRELATION(2);
```

```
        AA[I,P+1]:=1;
        FOR J:= 0 TO P-1 DO
        BEGIN
            AA[I,P-J]:= (1+FK[I,P-J])/(1-FK[I,P-J])*AA[I,P-J+1];
        END;
    END;
    CLOSE(DFILEVAR);
    EMAX:=D5[SFNO];
    FOR I:=SFNO TO SFNO+NF-1 DO
    BEGIN
        IF D5[I]>EMAX THEN
        EMAX:=D5[I];
    END;
    WRITELN('name of energy file?');
    READLN(ENFILE);
    ASSIGN(ENVAR,ENFILE);
    REWRITE(ENVAR);
    FOR J:=SFNO TO (SFNO+NF) DO
    WRITELN(ENVAR,D5[J]);
    DISPLAY;
END.
```

SPEECH_2.PAS

```
PROGRAM SPEECH_2;
USES GRAPH,DOS,CRT;
LABEL 2,3,4,5,6,10,20,25,30,50,60;
CONST
    P1=14;N=200;FRMAX=90;P=12;
VAR
                                  :ARRAY[1..FRMAX,1..2*P1] OF BYTE;
   A                              :ARRAY[0..FRMAX] OF LONGINT;
   EN                             :ARRAY[1..FRMAX,1..P] OF REAL;
   FK                             :ARRAY[1..FRMAX,1..P+1] OF REAL;
   AR
   ENBIG,ENTH,TEMPENT             :LONGINT;
   ENTHL,ENTHH                    :WORD;
   X1,Y1,X2,Y2,X3,Y3,XX,YY,
   XX1,YY1,N1                     :INTEGER;
   C                              :ARRAY[0..2047]OF BYTE;
   D                              :ARRAY[0..200] OF INTEGER;
   I,J,FLAG,FLAG2,CODE            :INTEGER;
   LFILE,DFILE,S                  :STRING[14];
   LVAR,DVAR,FILEVAR,INFILE       :TEXT;
   IBPTR                          :^INTEGER;
   ICPTR                          :^LONGINT;
   NINI,OFFSET,WPTR,
   FR,NMFR,PAT1,CLOCK,APR,J11,
   GRAPHDRIVER,GRAPHMODE,ERRORCODE,
   PAT3,PAT6,FRCNT,FS,WIND,PAR    :INTEGER;
   FLTEST                         :BYTE;
   INI                            :ARRAY[1..220] OF WORD;
   WIN                            :ARRAY[1..220] OF INTEGER;
   PREEMPH,CHS,CH1,CH2            :STRING[5];
   SCR                            :REAL;
   DUMMY,TEMP,TIM,DSC             :INTEGER;
   PITCHS,PITL,PITCHL             :ARRAY[1..FRMAX]OF REAL;
   PS,PL ,PS1                     :STRING[40];
   DFILEVAR                       :FILE OF INTEGER;
   OFILE,LPFILE,PIFILE,COORD      :STRING[14];
   CH,CHH,C_H,CHHH,DDUMMY         :CHAR;
   D1,D2                          :ARRAY[1..200] OF INTEGER;
   D3,D4                          :ARRAY[0..200] OF INTEGER;

 {** INTERRUPT PROCEDURE, PC RECEIVES PARAMETERS FROM
     TMS-32010 ON AN INTERRUPT BASIS. **}

 PROCEDURE ADCINT;
 INTERRUPT;
 BEGIN
     FLAG:=1;
     IF FLAG2=0 THEN
     A[J,I]:=PORT[$318]
     ELSE
     C[I]:=PORT[$318];
     PORT[$20]:=$67;
 END;

 {** DISPLAY PROCEDURE BEGINS **}

 PROCEDURE DISPLAY;
 LABEL 50,70,80;
 VAR
    Z1,Z2,Z3,Z4,J1,J2,XX,YY1,XX1,
    XX2,SRR,XX11,SFNO,EXCODE      :INTEGER;
    YY,YY11                       :LONGINT;
 PROCEDURE DRAW_LINES(YST,YEND,XST,COLOR:INTEGER);
 BEGIN
     SETCOLOR(COLOR);
     FOR J:=YST TO YEND DO
```

```
END;
BEGIN
     ENBIG:=EN[1];
     FOR I:=1 TO FRCNT DO
     BEGIN
          IF EN[I]>ENBIG THEN ENBIG:=EN[I];
          AR[I,P+1]:=1;
          FOR J:=0 TO P-1 DO
          BEGIN
               AR[I,P-J]:=(1+FK[I,P-J])/(1-FK[I,P-J])*AR[I,P-J+1];
          END;
     END;
     ASSIGN(INFILE,'COORD.PAS ');
     RESET(INFILE);
     FOR J:=1 TO 125 DO
     BEGIN
          READLN(INFILE,D1[J],D2[J]);
          D1[J]:=150-D1[J];
     END;
     FOR J:=126 TO 137 DO
     BEGIN
          READLN(INFILE,D3[J-125],D4[J-125]);
          D3[J-125]:=150-D3[J-125];
     END;
     CLOSE(INFILE);
     GRAPHDRIVER:=DETECT;
     INITGRAPH(GRAPHDRIVER,GRAPHMODE,'C:\');
     DIRECTVIDEO:=FALSE;
     ERRORCODE:=GRAPHRESULT;
     IF ERRORCODE<>GROK THEN
     BEGIN
          WRITE('GRAPHICS ERROR: ');
          WRITELN(GRAPHERRORMSG(ERRORCODE));
          WRITELN('PROGRAM ABORTED');
          HALT(1);
     END;
     SFNO:=1;DSC:=20;I:=1;
     FOR J:=1 TO 64 DO
     LINE(2*D1[J],D2[J],2*D1[J+1],D2[J+1]);
     FOR J:=66 TO 76 DO
     LINE(2*D1[J],D2[J],2*D1[J+1],D2[J+1]);
     FOR J:=78 TO 105 DO
     LINE(2*D1[J],D2[J],2*D1[J+1],D2[J+1]);
     FOR J:=107 TO 124 DO
     LINE(2*D1[J]+2,D2[J],2*D1[J+1]+2,D2[J+1]);
     LINE(450,117,450,197);MOVETO(580,117);LINETO(580,197);
     YY1:=117;XX1 := 583;
     FOR J:=0 TO 4 DO
     BEGIN
          STR((400-J*100),CH1);
          MOVETO(XX1+4,YY1-5);OUTTEXT(CH1);
          YY1 := YY1+20;
     END;
     YY1:=117;
     FOR J:=0 TO 8 DO
     BEGIN
          PUTPIXEL(581,YY1+J*10,WHITE);
          PUTPIXEL(582,YY1+J*10,WHITE);
     END;
     YY1:=117;XX1:=450;
     FOR J:=0 TO 5 DO
     BEGIN
           STR((100-J*20),CH1);
           MOVETO(XX1+5,YY1-5);OUTTEXT(CH1);
           YY1:=YY1+16;
```

```
YY1:=117;
FOR J:=0 TO 10 DO
BEGIN
    PUTPIXEL(451,YY1+J*8,WHITE);
    PUTPIXEL(452,YY1+J*8,WHITE);
END;
Z1:= ROUND((EN[I]/ENBIG)*100);Z2:=ROUND(PITCHS[I]);
STR(Z1,PS);MOVETO(368,160);
OUTTEXT('E = ');OUTTEXT(PS);
STR(Z2,PS1);MOVETO(500,160);
OUTTEXT('F0 = ');OUTTEXT(PS1);
SRR:=ROUND(4*Z1/5);
Z2:=197-ROUND(Z2/5);
DRAW_LINES(Z2,197,579,1);
Z1:=197-SRR;
DRAW_LINES(Z1,197,449,1);
MOVETO(400,25);
LINETO(400+(P+1)*15,25);
MOVETO(400,110);
LINETO(400+(P+1)*15,110);
XX:=400;
FOR J:=1 TO 12 DO
BEGIN
    YY1:=109;
    YY:=YY1-ROUND(DSC*AR[I,P-J+1]);
    LINE(XX,YY,XX+15,YY);
    IF J <> 12 THEN
    BEGIN
        YY11:=YY1-ROUND(DSC*AR[I,P-J]);
        LINE(XX+15,YY,XX+15,YY11);
    END;
    XX:=XX+15;
END;
FOR J:=0 TO 7 DO
BEGIN
    IF J=0 THEN
    BEGIN
        XX1:=244;YY1:=105;
    END
    ELSE
    BEGIN
        XX1:=2*D3[J];YY1:=D4[J]+ROUND(DSC*AR[I,J]/(2.0));
    END;
    YY11 := D4[J+1]+ROUND(DSC*AR[I,J+1]/(2.0));
    XX11 := 2*D3[J+1] ;
    SETCOLOR(1);
    LINE(XX1,YY1,XX11,YY11);
END;
BEGIN
    FOR J := 8 TO 11 DO
    BEGIN
        IF J=8 THEN
        BEGIN
        XX1:=2*D3[J];YY1:=D4[J]+ROUND(DSC*AR[I,J]/(2.0));
        END
        ELSE
        BEGIN
            YY1 := D4[J];
            XX1 := 2*D3[J]+2*ROUND(DSC*AR[I,J]/(2.0))+10;
        END;
        YY11 := D4[J+1];
        XX11 := 2*D3[J+1]+2*ROUND(DSC*AR[I,J+1]/(2.0))+10;
        LINE(XX1,YY1,XX11,YY11);
        SETCOLOR(1);
END;
```

```
    MOVETO(400,0);STR(I,CH1); SETCOLOR(1);
    OUTTEXT('Frame:');OUTTEXT(CH1);
    TIM:=I*10;STR(TIM,CH1);
    OUTTEXT(', T:');OUTTEXT(CH1);OUTTEXT(' ms. ');J1:=0;
50: Z2:=Z4;Z1:=Z3;
70: MOVETO(400,8);SETCOLOR(1);OUTTEXT('Exit(Y/N)?');
    REPEAT
        BEGIN
        END;
    UNTIL KEYPRESSED;
    C_H:=READKEY;
    IF (C_H='Y') OR (C_H='y')THEN
    BEGIN
        EXCODE:=1;EXIT;
    END;
    MOVETO(400,8);SETCOLOR(0);OUTTEXT('Exit(Y/N)?');
    MOVETO(400,8);SETCOLOR(1);OUTTEXT('Refresh:');
    MOVETO(400,16);SETCOLOR(1);OUTTEXT('Complete(C)');
    OUTTEXT('/ Segmentwise(S)');
    REPEAT
        BEGIN
        END;
    UNTIL KEYPRESSED;
    C_H:=READKEY;
    MOVETO(400,8);SETCOLOR(0);OUTTEXT('Refresh:');
    MOVETO(400,16);SETCOLOR(0);OUTTEXT('Complete(C)');
    OUTTEXT('/ Segmentwise(S)');
    IF (C_H<>'C') AND (C_H<>'c') AND (C_H<>'S') AND (C_H<>'s') THEN
    BEGIN
        GOTO 70;
    END;
    MOVETO(400,8);SETCOLOR(1);OUTTEXT('Next(E,+,-,=)?');
    OUTTEXT(' -->');OUTTEXT('01');
    STR(FRCNT,CH1);OUTTEXT('-');OUTTEXT(CH1);
80: REPEAT
        BEGIN
        END;
    UNTIL KEYPRESSED;
    CH:=READKEY;
    IF (CH='E') OR (CH<>'+') AND (CH<>'-') AND (CH<>'=') THEN
    BEGIN
        MOVETO(400,8);SETCOLOR(0);OUTTEXT('Next(E,+,-,=)?');
        OUTTEXT(' -->');OUTTEXT('01');
        STR(FRCNT,CH1);OUTTEXT('-');OUTTEXT(CH1);
        GOTO 70;
    END;
    IF CH='+' THEN
    BEGIN
        J1:=I+1;
        DDUMMY:='0';
    END;
    IF CH='-' THEN
    BEGIN
        J1:=I-1;
        DDUMMY:='0';
    END;
    IF (CH='+') OR (CH='-') THEN
    BEGIN
        IF (J1<SFNO) OR (J1>(SFNO+FRCNT-1)) THEN
        BEGIN
            SOUND(800);
            DELAY(500);
            NOSOUND;
            IF CH='+' THEN J1:=J1-1;
            IF CH='-' THEN J1:=J1+1;
```

```
    END;
  END;
  IF CH='=' THEN
  BEGIN
    WHILE NOT KEYPRESSED DO
    BEGIN
    END;
    CHH :=READKEY;
    VAL(CHH,J1,CODE);
    WHILE NOT KEYPRESSED DO
    BEGIN
    END;
    CHHH :=READKEY;
    VAL(CHHH,J11,CODE);
    J1:=J1*10+J11;
    IF (J1<SFNO) OR (J1>(SFNO+FRCNT-1)) THEN EXIT;
  END;
  MOVETO(400,0);STR(I,CH1);SETCOLOR(0);
  OUTTEXT('Frame:');OUTTEXT(CH1);
  TIM:=I*10;STR(TIM,CH1);
  OUTTEXT(', T:');OUTTEXT(CH1);OUTTEXT(' ms. ');
  MOVETO(400,0);STR(J1,CH1); SETCOLOR(1);
  OUTTEXT('Frame:');OUTTEXT(CH1);
  TIM:=J1*10;STR(TIM,CH1);
  OUTTEXT(', T:');OUTTEXT(CH1);OUTTEXT(' ms. ');
  SETCOLOR(0);
  MOVETO(500,160);OUTTEXT('F0 = ');OUTTEXT(PS1);
  MOVETO(368,160);OUTTEXT('E = ');OUTTEXT(PS);
  Z4:=ROUND(PITCHS[J1]);STR(Z4,PS1);
  MOVETO(500,160);SETCOLOR(1);OUTTEXT('F0 = ');OUTTEXT(PS1);
  Z4:=197-ROUND(Z4/5);
  IF PITCHS[J1] >= PITCHS[I] THEN
  BEGIN
    DRAW_LINES(Z4,Z2,579,1);
    END
    ELSE
    BEGIN
      DRAW_LINES(Z2,Z4,579,0);
    END;
    Z3:=ROUND((EN[I]/ENBIG)*100);STR(Z3,PS);
    MOVETO(368,160);SETCOLOR(1);OUTTEXT('E = ');OUTTEXT(PS);
    SRR:=ROUND(4*Z3/5);Z3:=197-SRR;
    IF Z3 < Z1 THEN
    BEGIN
      DRAW_LINES(Z3,Z1,449,1);
    END
    ELSE
    BEGIN
      DRAW_LINES(Z1,Z3,449,0);
    END;
    WHILE NOT KEYPRESSED DO
    BEGIN
    END;
    CHH:=READKEY;
    END;
    FOR J := 0 TO 7 DO
    BEGIN
      IF J=0 THEN
      BEGIN
        XX1:=244;YY1:=105;
      END
      ELSE
      BEGIN
        XX1:=2*D3[J];YY1:=D4[J]+ROUND(DSC*AR[I,J]/(2.0));
      END;
```

```
        XX11 := 2*D3[J+1] ;
        SETCOLOR(0);
        LINE(XX1,YY1,XX11,YY11);
        IF (C_H='S') OR (C_H='s') THEN
        BEGIN
            WHILE NOT KEYPRESSED DO
            BEGIN
            END;
        CH:=READKEY;
        END;
        IF J=0 THEN
        BEGIN
            XX1:=244;YY1:=105;
        END
        ELSE
        BEGIN
            XX1:=2*D3[J];YY1:=D4[J]+ROUND(DSC*AR[J1,J]/(2.0));
        END;
        YY11 := D4[J+1]+ROUND(DSC*AR[J1,J+1]/(2.0));
        XX11 := 2*D3[J+1] ;
        SETCOLOR(1);
        LINE(XX1,YY1,XX11,YY11);
        IF (C_H='S') OR (C_H='s') THEN
        BEGIN
            WHILE NOT KEYPRESSED DO
            BEGIN
            END;
            CH:=READKEY;
        END;
   END;
   FOR J:=8 TO 11 DO
   BEGIN
       IF J=8 THEN
       BEGIN
           XX1:=2*D3[J];YY1:=D4[J]+ROUND(DSC*AR[I,J]/(2.0));
       END
       ELSE
       BEGIN
           YY1 := D4[J];
           XX1 := 2*D3[J]+2*ROUND(DSC*AR[I,J]/(2.0))+10;
       END;
       YY11 := D4[J+1];
       XX11 := 2*D3[J+1]+2*ROUND(DSC*AR[I,J+1]/(2.0))+10;
       SETCOLOR(0);
       LINE(XX1,YY1,XX11,YY11);
       IF (C_H='S') OR (CH='s') THEN
       BEGIN
           WHILE NOT KEYPRESSED DO
           BEGIN
           END;
           CH:=READKEY;
       END;
       IF J=8 THEN
       BEGIN
           XX1:=2*D3[J];YY1:=D4[J]+ROUND(DSC*AR[J1,J]/(2.0));
       END
       ELSE
       BEGIN
           YY1 := D4[J];
           XX1 := 2*D3[J]+2*ROUND(DSC*AR[J1,J]/(2.0))+10;
       END;
       YY11 := D4[J+1];
       XX11 := 2*D3[J+1]+2*ROUND(DSC*AR[J1,J+1]/(2.0))+10;
       SETCOLOR(1);
       LINE(XX1,YY1,XX11,YY11);
```

```
            BEGIN
                 WHILE NOT KEYPRESSED DO
                 BEGIN
                 END;
                 CH:=READKEY;
            END;
        END;
     XX:=400;
     FOR J:=1 TO 12 DO
     BEGIN
         YY1:=109;YY11:=0;YY:=0;
         YY:=YY1-ROUND(DSC*AR[I,P-J+1]);
         SETCOLOR(0);
         LINE(XX,YY,XX+15,YY);
         IF J<>12 THEN
         BEGIN
             YY11:=YY1-ROUND(DSC*AR[I,P-J]);
             SETCOLOR(0);
             LINE(XX+15,YY,XX+15,YY11);
         END;
         IF (C_H='S') OR (C_H='s') THEN
         BEGIN
             WHILE NOT KEYPRESSED DO
             BEGIN
             END;
         CH:=READKEY;
         END;
         YY:=0;YY11:=0;
         YY:=YY1-ROUND(DSC*AR[J1,P-J+1]);
         SETCOLOR(1);
         LINE(XX,YY,XX+15,YY);
         IF J<>12 THEN
         BEGIN
             YY11:=YY1-ROUND(DSC*AR[J1,P-J]);
             SETCOLOR(1);
             LINE(XX+15,YY,XX+15,YY11);
         END;
         IF (C_H='S') OR (C_H='s') THEN
         BEGIN
             WHILE NOT KEYPRESSED DO
             BEGIN
             END;
             CH:=READKEY;
         END;
         XX:=XX+15;
     END;
     IF (C_H='S') OR (C_H='s') THEN
     BEGIN
         SOUND(800);
         DELAY(500);
         NOSOUND;
     END;
     I:=J1;
     IF DDUMMY='0' THEN
     BEGIN
         DDUMMY:='2';
     END;
     GOTO 80;
     END;

(****** PROCEDURE DISPLAY ENDS *********)

PROCEDURE MENU;
VAR
   XM1,YM1          :INTEGER;
```

```
   WINDOW(1,1,80,12);
   CLRSCR;
   XM1:=1;YM1:=1;
   GOTOXY(XM1,YM1);
   WRITE('* SPEECH ANALYSIS & DISPLAY PROGRAM MENU *');
   WRITELN;
   GOTOXY(XM1,YM1+2);
   WRITE(' 1) FRAME LENGTH           : ',NINI);
   GOTOXY(XM1,YM1+3);
   WRITE(' 2) FRAME OVERLAP(%)       : ',FR);
   GOTOXY(XM1,YM1+4);
   WRITE(' 3) SAMPLING FREQUENCY     : ',FS);
   GOTOXY(XM1,YM1+5);
   WRITE(' 4) ADC OFFSET             : ',OFFSET);
   GOTOXY(XM1,YM1+6);
   WRITE(' 5) SELECT WINDOW TYPE     : ',WIND);
   GOTOXY(XM1,YM1+7);
   WRITE(' 6) QUIT MENU ');
   XM1 := 40; YM1:=1;
   GOTOXY(XM1,YM1+2);
   WRITE(' 7) WINDOW COEFF. POINTER : ',WPTR);
   GOTOXY(XM1,YM1+3);
   WRITE(' 8) PRE-EMPHASIS             : ',PREEMPH);
   GOTOXY(XM1,YM1+4);
   WRITE(' 9) PRE-EMPHASIS COEFF.      : ',APR);
   GOTOXY(XM1,YM1+5);
   WRITE('10) ENERGY THRESHOLD         : ',ENTH);
   GOTOXY(XM1,YM1+6);
   WRITE('11) NO. OF FRAMES ACQUIRED: ',PAT3);
   GOTOXY(XM1,YM1+7);
   WRITE('12) QUIT PROGRAM ');
END;

{***** MAIN PROGRAM STARTS HERE *****}

BEGIN
    NINI:=N; OFFSET:=2020; ENTHL:=$0;ENTHH:=0; WPTR:=1800;
    FR:=50; NMFR:=0; PAT1:=2400; CLOCK:=524; APR:=7;
    PAT3:=10; PAT6:=10;PREEMPH := 'OFF';FS := 10000;
    WIND := 2;ENTH := 0;
    WINDOW(1,1,80,25);
    CLRSCR;
    GOTOXY(22,3);
    WRITELN('* SPEECH ANALYSIS & DISPLAY PROGRAM *');
    GOTOXY(22,4);
    WRITELN('* FOR REAL-TIME SPEECH *');
    GOTOXY(22,5);
    WRITELN('*****************************************');
    GOTOXY(1,25);
    WRITE('* FOR MENU PRESS ANY KEY *');
    CH:=READKEY;
    GOTOXY(1,25);
    CLREOL;

 {*** SELECTION OF PARAMETERS STARTS *****}

 50: MENU;
    WINDOW(1,13,80,25);
    GOTOXY(1,12);
    WRITE('SELECT THE PARAMETER YOU WANT TO ALTER : ');
    READLN(CHS);
    VAL(CHS,PAR,CODE);
    FOR I:= 1 TO 8 DO
    BEGIN
        GOTOXY(1,I);
```

```
END;
GOTOXY(1,1);

CASE PAR OF

      1 :BEGIN
            WRITE('FRAME LENGTH IN NO. OF SAMPLES : ');
            READLN(CHS);
            VAL(CHS,NINI,CODE);
            WHILE (NINI<150) OR (NINI>200) DO
            BEGIN
                GOTOXY(34,1);
                CLREOL;
                WRITE(#7);WRITE(#7);
                READLN(CHS);
                VAL(CHS,NINI,CODE);
            END;
         END;

      2 :BEGIN
            WRITE('SUCCESSIVE FRAME OVERLAP(0% TO 50%) : ');
            READLN(CHS);
            VAL(CHS,FR,CODE);
            WHILE (FR<0) OR (FR>50) DO
            BEGIN
                GOTOXY(38,1);
                CLREOL;
                WRITE(#7);WRITE(#7);
                READLN(CHS);
                VAL(CHS,FR,CODE);
            END;
         END;

      3 :BEGIN
            WRITE('SAMPLING FREQUENCY : ');
            READLN(CHS);
            VAL(CHS,FS,CODE);
            WHILE (FS<1000) OR (FS>10000) DO
            BEGIN
                GOTOXY(22,1);
                CLREOL;
                WRITE(#7);WRITE(#7);
                READLN(CHS);
                VAL(CHS,FS,CODE);
            END;
         END;

      4 :BEGIN
            WRITE('ADC DATA OFFSET : ');
            READLN(CHS);
            VAL(CHS,OFFSET,CODE);
            WHILE (OFFSET<2000) OR (OFFSET>2060) DO
            BEGIN
                GOTOXY(19,1);
                CLREOL;
                WRITE(#7);WRITE(#7);
                READLN(CHS);
                VAL(CHS,OFFSET,CODE)
            END;
         END;

      5 :BEGIN
            WRITELN('WINDOW TYPES ARE :');
            WRITELN;
            WRITELN('1 : HAMMING          :');
```

```
        WRITELN('3 :                   :');
        WRITELN('4 :                   :');
        WRITELN;
        WRITE('SELECT WINDOW TYPE    : ');
        READLN(CHS);
        VAL(CHS,WIND,CODE);
        WHILE (WIND<1) OR (WIND>4) DO
        BEGIN
            GOTOXY(23,8);
            CLREOL;
            WRITE(#7);WRITE(#7);
            READLN(CHS);
            VAL(CHS,WIND,CODE);
        END;
    END;

 6 :BEGIN
        GOTOXY(1,12);
        CLREOL;
        WRITE('IS PARAMETER SELECTION OVER ?(Y/N) : ');
        CH:=READKEY;
        WRITE(CH);
        IF (CH='Y') OR (CH='y') THEN GOTO 60;
    END;

 7 :BEGIN
        WRITELN('STARTING ADDRESS OF THE WINDOW ');
        WRITE('COEFFICIENT TABLE IN TMS-32010 PROGRAM');
        WRITE(' MEMORY : ');
        READLN(CHS);
        VAL(CHS,WPTR,CODE);
        WHILE (WPTR<1750) OR (WPTR>2000) DO
        BEGIN
            GOTOXY(48,2);
            CLREOL;
            WRITE(#7);WRITE(#7);
            READLN(CHS);
            VAL(CHS,WPTR,CODE);
        END;
    END;

 8 :BEGIN
        WRITE('PRE-EMPHASIS (ON/OFF) : ');
        READLN(PREEMPH);
        WHILE (PREEMPH<>'ON') AND (PREEMPH<>'OFF') DO
        BEGIN
            GOTOXY(24,1);
            CLREOL;
            WRITE(#7);WRITE(#7);
            READLN(PREEMPH);
        END;
    END;

 9 :BEGIN
        WRITE('PRE-EMPHASIS APPLIED IS : ');
        WRITELN('X[N] = S[N] - A * S[N-1]');
        WRITELN('A = APR/8');
        WRITELN;
        WRITE('ENTER APR (0 TO 8)     : ');
        READLN(CHS);
        VAL(CHS,APR,CODE);
        WHILE (APR<0) OR (APR>8) DO
        BEGIN
            GOTOXY(26,4);
            CLREOL;
```

```
                    READLN(CHS);
                    VAL(CHS,APR,CODE);
                END;
            END;

        10 :BEGIN
                WRITE('IT INDICATES THE LOWER LIMIT FOR THE');
                WRITELN(' SHORT TIME ENERGY IN THE FIRST FRAME');
                WRITELN;
                WRITE('ENTER ENERGY THRESHOLD : ');
                READLN(CHS);
                VAL(CHS,ENTH,CODE);
                WHILE (ENTH<0) OR (ENTH>99999) DO
                BEGIN
                    GOTOXY(24,3);
                    CLREOL;
                    WRITE(#7);WRITE(#7);
                    READLN(CHS);
                END;
            END;

        11 :BEGIN
                WRITE('ENTER TOTAL NO. OF FRAMES TO BE ACQUIRED :');
                READLN(CHS);
                VAL(CHS,PAT3,CODE);
                WHILE (PAT3<1) OR (PAT3>90) DO
                BEGIN
                    GOTOXY(43,1);
                    CLREOL;
                    WRITE(#7);WRITE(#7);
                    READLN(CHS);
                    VAL(CHS,PAT3,CODE);
                END;
            END;

        12 :BEGIN
                GOTOXY(1,12);
                CLREOL;
                WRITE('ARE YOU SURE YOU WANT TO EXIT ?');
                CH := READKEY;
                IF (CH='Y') OR (CH='y') THEN EXIT;
            END;
    ELSE

    BEGIN
        GOTOXY(1,12);
        CLREOL;
        WRITE('IS PARAMETER SELECTION OVER ?');
        CH := READKEY;
        IF (CH='Y') OR (CH='y') THEN GOTO 60;
    END;
    END;
    GOTOXY(41,12);
    CLREOL;
    WRITE(#7);
    GOTO 50;

(** PARAMETER SELECTION OVER **);
60: TEMPENT := ENTH AND $0000FFFF;
    ENTHL   := TEMPENT;
    TEMPENT := ENTH AND $FFFF0000;
    ENTHH   := TEMPENT DIV 65536;
    CLOCK   := TRUNC(FS/5000000*64*64*64);
    INI[6]  := TRUNC(NINI*(100-FR)/100);
    PAT6    := TRUNC(((NINI*PAT3)-(NINI-INI[6]))/INI[6]);
```

```
   NMFR    := NINI-INI[6];
   PAT1    := WPTR + 2*NINI;

   IF WIND=1 THEN
   BEGIN
       FOR I:=1 TO NINI DO
       BEGIN
           SCR :=4096*(0.54-0.46*COS(2*PI*(I-1)/(NINI-1)));
           WIN[I] := ROUND(SCR);
       END;
   END
   ELSE
   BEGIN
       FOR I:=1 TO NINI DO
       WIN[I] := 4096;
   END;

{**  WRITELN('DO YOU WANT TO CHOOSE HAMMING WINDOW ?');
     READLN(CH);
     IF (CH='Y')OR(CH='y') THEN
     BEGIN
         WRITELN('ENTER WINDOW FILE NAME');
         READLN(WFILE);
         ASSIGN(WFILVAR,WFILE);
         RESET(WFILVAR);
         FOR I:=1 TO NINI DO
         BEGIN
             READLN(WFILVAR,WIN[I]);
         END;
         CLOSE(WFILVAR);
     END
     ELSE
     FOR I:=1 TO NINI DO
     BEGIN
         WIN[I]:=4096;
     END;
**}
     IF (PREEMPH='OFF') THEN APR:=0;

     INI[1]:=NINI; INI[2]:=OFFSET; INI[3]:=ENTHL; INI[4]:=ENTHH;
     INI[5]:=WPTR; INI[6]:=TRUNC(NINI*(100-FR)/100);
     INI[7]:=NMFR; INI[8]:=PAT1;
     FOR I:=1 TO NINI DO
     BEGIN
         INI[I+8]:=WIN[I];
     END;
     INI[209]:=CLOCK;
     INI[210]:=APR; INI[211]:=PAT3; INI[212]:=PAT6;

     CLRSCR;
     FOR I:=1 TO 9 DO WRITELN(INI[I]);
     FOR I:=208 TO 212 DO WRITE(INI[I],' ');


{******* TRANSFER PARAMETERS TO TMS-32010 **********}

     PORT[$319]:=INI[1] AND $000000FF;
2:   FLTEST:=PORT[$320] AND ($2);
     IF FLTEST=2 THEN
     PORT[$319]:=HI(INI[1])
     ELSE
     GOTO 2;
     FOR I:=2 TO 209 DO
     BEGIN
3:       FLTEST:= PORT[$320] AND ($2);
```

```
        PORT[$319]:=LO(INI[I]) ELSE GOTO 3;
4:      FLTEST:= PORT[$320] AND ($2);
        IF FLTEST=2 THEN
        PORT[$319]:=HI(INI[I]) ELSE GOTO 4;
    END;
    FOR I:=210 TO 212 DO
    BEGIN
6:      FLTEST:= PORT[$320] AND ($2);
        IF FLTEST=2 THEN
        PORT[$319]:=INI[I] ELSE GOTO 6;
    END;

(********* TRANSFER OF PARAMETERS COMPLETE ***********)

    FRCNT := PAT6;
5:  CLOSEGRAPH;
    INLINE($FB);FLAG :=0;FLAG2:=0;
    I:=1;J:=1;
    A[1,1]:=PORT[$318];
    SETINTVEC($F,@ADCINT);
    PORT[$21]:=PORT[$21] AND $7F;
    PORT[$20]:=$67;
    GOTOXY(1,1);
    CLREOL;
    WRITELN('PRESS ANY KEY TO START');
    WHILE NOT KEYPRESSED DO
    BEGIN
    END;
    CH:=READKEY;
    PORT[$319]:=0;
    WHILE FLAG<>1 DO
    BEGIN
    END;
    WRITELN('ACQUIRING LPC COEFF.FROM TMS-320');
10: BEGIN
        IF FLAG =1 THEN
        BEGIN
            FLAG :=0;
            I:=I+1;
            IF I>2*P1 THEN
            BEGIN
                J:=J+1;
                I:=1;
            END;
            IF J>FRCNT THEN GOTO 20;
            PORT[$319]:=0;
        END;
        GOTO 10;
    END;
20: FLAG2:=1;
    I:=1;
    PORT[$319]:=0;
25: IF FLAG = 1 THEN
    BEGIN
        FLAG:=0;
        I:=I+1;
        IF I>2*NINI THEN GOTO 30;
        PORT[$319]:=0;
    END;
    GOTO 25;
30: PORT[$21]:=PORT[$21] OR $80;
    FOR J:=1 TO FRCNT DO
    BEGIN
        ICPTR:=@A[J,1];
        EN[J]:=ICPTR^;
```

```
        BEGIN
            IBPTR:=@A[J,2*I-1];
            FK[J,I-2]:=IBPTR^/32768;
        END;
    END;
    DISPLAY;
    SETCOLOR(BLACK);
    CLEARVIEWPORT;
    CLOSEGRAPH;
    WRITELN('DO YOU WANT TO RESTART THE MAIN PROGRAM ?');
    WHILE NOT KEYPRESSED DO
    BEGIN
    END;
    CH :=READKEY;
    IF (CH='Y') OR (CH='y') THEN
    BEGIN
        PORT[$319] := 1;
        GOTO 5;
    END
    ELSE
    PORT[$319] := 0;
    CLRSCR;
    WRITELN('ENTER NAME OF THE LPC COEFF.+ENERGY FILE TO BE CREATED:');
    READLN(LFILE);
    ASSIGN(LVAR,LFILE);
    REWRITE(LVAR);
    WRITELN(LVAR,FRCNT);
    WRITELN(LVAR,P);
    FOR J:= 1 TO FRCNT DO
    BEGIN
        FOR I:=1 TO P DO
        BEGIN
            WRITELN(LVAR,FK[J,I]);
        END;
    END;
    WRITELN(LVAR,ENBIG);
    FOR I:=1 TO FRCNT DO
    BEGIN
        WRITELN(LVAR,EN[I]);
    END;
    CLOSE(LVAR);
    WRITELN('ENTER NAME OF THE DATA FILE TO BE CREATED');
    READLN(DFILE);
    ASSIGN(DVAR,DFILE);
    REWRITE(DVAR);
    WRITELN(DVAR,N);
    FOR J:=1 TO N DO
    BEGIN
        IBPTR:=@C[2*J-1];
        D[J]:=IBPTR^;
        WRITELN(DVAR,D[J]);
    END;
    CLOSE(DVAR);
END.
```

REFERENCES

Atal BS & Hanauer L (1971). Speech analysis & synthesis by linear prediction of the speech wave. J. Acoust. Soc. Am., vol 50, pp 637-655.

Chafekar SA (1990). Speech Synthesis for the Testing of Sensory Aids for the Hearing Impaired. MTech dissertation, School of Biomedical Engineering, I.I.T. Bombay.

Crichton RG & Fallside F (1974). Linear prediction model of speech production with applications to deaf speech training. Proc. of the IEE, Control & Science, vol 121, pp 865-873. Reprinted in Levitt, Pickett, & Houde (1980), pp 399-407.

Fant G (1959). The acoustics of speech. Proc. 3rd Inter. Cong. Acoust., pp 188-201. Reprinted in Schafer & Markel (1979), pp 17-30.

Flanagan JL (1972). Speech Analysis, Synthesis & Perception. 2nd edn. New York: Springer-Verlag. *

Grant KW, Ardell AH, Kuhl PK, & Sparks DW (1985). The contribution of fundamental frequency, amplitude envelope and voicing cues to speechreading in normal hearing subjects. J. Acoust. Soc. Am., vol 77, pp 671-677.

Gulian E, Fallside F, & Hinds P (1984). Can deaf children interact * with computers? Evidence from speech-acquision training. Interact, 1st IFIP Conf. on Human-computer Interaction. Elsevier Science Publishers, North Holand, vol 1, pp 164-168.

Gupte MS (1990). *A Speech Processor & Display for Speech Training of the Hearing Impaired*. MTech dissertation, Electrical Engineering Department, I.I.T. Bombay.

Klatt DH (1980). Software for a cascade/parallel formant synthesizer. *J. Acoust. Soc. Am.*, vol 67, pp 971-995.

Kushler CA, Misu T, Isomura T, Funakubo H, & Komeda T (1985). A microprocessor and signal processor based speech training aid for hearing impaired. *Proc. 8th Ann. Conf. on Rehabilitation Tech.*, Memphis, Tennessee, pp 311-313.

Ladefoged P (1982). *A Course in Phonetics*. New York: Harcourt Brace Jovanovich.

Leroux J & Gueguen C (1977). A fixed point computation of the parcor coefficients. *IEEE Trans. ASSP*, vol 25, pp 257-259.

Levitt H, Pickett JM, & Houde RA, Eds. (1980). *Sensory Aids for the Hearing Impaired*. New York: IEEE Press.

Makhoul J (1975). Linear prediction: a tutorial review. *Proc. IEEE*, vol 63, pp 561-580.

Martony J (1968). On the correction of the voice pitch level for severely hard of hearing subjects. *Am. Ann. Deaf*, vol 113, pp 195-202.

Martony J & Spens KE (1972). The transposer as a lipreading aid. *Proc. Conf. Speech Comm. & Processing*, pp 246-248. Reprinted in Levitt, Pickett, & Houde (1980), pp 224-226.

Mermelstein P (1967). Determination of the vocal tract shape from measured formant frequencies. *J. Acoust. Soc. Am.*, vol 41, pp 1283-1294. Reprinted in Schafer & Markel (1979), pp 321-332.

Morris LR (1983). Digital Signal Processing Software. Ottawa, Canada: Digital Signal Processing Software Inc. & Carleton University.

Nickerson RS & Stevens KN (1972). An experimental computer based system of speech training aids for the deaf. Proc. Conf. Speech Commun. and Processing, Newton, Mass., pp 238-241.

Pandey PC (1987). Speech Processing for Cochlear Prostheses. PhD thesis, Electrical Engineering Department, University of Toronto, Canada.

Pickett JM (1963). Tactual communication of speech sounds to the deaf: comparison with lipreading. J. Speech & Hearing Disorders, vol 28, pp 315-330. Reprinted in Levitt, Pickett, & Houde (1980), pp 262-277.

Potter RK (1945). Visible patterns of sound. Science, pp 463-470. Reprinted in Levitt, Pickett, & Houde (1980), pp 252-259.

Rabiner LR & Schafer RW (1978). Digital Processing of Speech Signals. Englewood Cliffs, New Jersy: Prentice Hall.

Rampal S (1991). Design & Implementation of a Parallel System of DSPs. MTech dissertation, Electrical Engineering Department, I.I.T. Bombay.

Risberg A (1968). Visual aids for speech correction. Am. Ann. Deaf, vol 113, pp 178-194. Reprinted in Levitt, Pickett & Houde (1980), pp 357-361.

Schafer RW & Markel JD, Eds, (1979). Speech Analysis. New York: IEEE Press.

Texas Instruments (1985). TMS-32010 Evaluation Module, User's Guide. Houston, Texas: Texas Instruments Technical Publications.

Upton HW (1968). Wearable eyeglass speechreading aid. Am. Ann. Deaf, vol 113, pp 222-229. Reprinted in Levitt, Pickett & Houde (1980), pp 322-323.

Wakita H (1973). Direct estimation of the vocal tract shape by inverse filtering of acoustc speech waveforms. IEEE Trans. Audio & Electroacoust., vol 21, pp 417-427. Reprinted in Schafer & Markel (1979), pp 333-343.

Wakita H (1979). Estimation of vocal tract shapes from acoustical analysis of the speech wave: the state of the art. IEEE Trans. ASSP, vol 27, pp 281-285.