A SPEECH TRAINING AID FOR THE DEAF

A Dissertation
submitted in partial fulfillment of the
requirements for the degree of

Master of Technology

by

Ashok Baragi B.N.

(94307023)

Guides : Dr. P. C. Pandey
Prof. S.D. Agashe

Department of Electrical Engineering,
Indian Institute of Technology, Bombay.
January 1996.

INDIAN INSTITUTE OF TECHNOLOGY, BOMBAY

DISSERTATION APPROVAL SHEET

Dissertation entitled "**A Speech Training Aid for the Deaf**" by Ashok Baragi B.N., is approved for the award of the degree of Master of Technology in Electrical Engineering.

Guide	:	_____	(Dr. P.C. Pandey)
Co-Guide	:	S. D Agashe	(Prof. S.D. Agashe)
Internal Examiner	:	Vikram M Gadre	(Dr. V.M. Gadre)
External Examiner	:	R.S. Sardesai	(Sri ~~S.K.~~ Sardesai)
Chairman	:	1/2/96	(Prof. B.P. Kashyap)

Abstract

Ashok Baragi B.N., "A Speech Training Aid for the Deaf", M.Tech. dissertation, Dept. of Elec. Engg., Indian Institute of Technology, Bombay, Jan 96.

Providing a visual feedback of various speech parameters, can help deaf children in learning to produce intelligible speech. A training aid of this type can also have a provision for comparing the speech parameters of teacher with that of the person being trained.

Earlier efforts at IIT Bombay have resulted in a training aid which estimates and displays vocal tract shape, pitch, and energy in real-time. In this project, this aid was tested for accuracy. Various curve fitting algorithms for smoothing the vocal tract area, were examined for their suitability for real-time implementation. After selection of a suitable algorithm, the same was implemented in real-time. The software of the aid was modified to freeze the vocal tract shape during stops to the shape just before the stops period, thereby eliminating the random vocal tract shape during this period.

A PC add-on DSP board based on TMS320C25 is used for acquiring and processing the speech signal. A PC/AT, based on 80286 processor is used for providing the user interface and for displaying the parameters. The fast data transfer between DSP and the PC is through a shared memory space on the DSP board. The system also provide facilities for storing the parameters for 100 frames (equivalent to 1.28 s) and displaying them either in review mode or as an "areagram", a two-dimensional representation of variation of vocal tract area, (from glottis to lip) with time.

The system was tested for accuracy and for consistency of energy, pitch, and vocal tract shape estimation with both synthetic and natural speech and results were satisfactory.

ACKNOWLEDGMENTS

I thank my guides Dr. P.C. Pandey and Prof. S.D. Agashe for their guidance and valuable suggestion through out the project work.

Any working place is full of excitement, tension, and depression. It is a friend circle where you can share your joy and sorrow. I thank my friend-cum-labmates Pravin, Prasad, Satish, Prakash, Mandar, Shivi, Tushar, Rahul, and last but not least Harshal for their timely help.

I would also like to thank Prashant Gavankar, Niranjan Khambete, and Yogesh Bhagwat for their suggestions and Kush Shinde for his help.

I thank god for giving me necessary strength and determination to complete my project work.

I thank all the people who have helped me directly or indirectly.

IIT Bombay.
January 1996

Ashok Baragi B.N.
(94307023)

Contents

List of Figures

Chapter 1

Introduction

1.1 Problem Overview

The deaf people, often find it difficult to produce normal speech, even if they do not have any physiological disorder in their speech production system. The obvious reason is a lack of feedback of their own speech. Such persons can be trained to produce intelligible speech, if proper feedback is provided. Visual feedback is one of the effective methods of feedback.

A training aid should provide visual feedback of speech parameters which are easily perceivable, as well as controllable by the person being trained. Phoneme utterances can be, generally characterized (except for nasals) by the vocal tract shape, voicing pattern, pitch and energy variations. Pitch provides the information regarding voicing pattern and helps in distinguishing between voiced sounds and unvoiced sounds. Energy gives an idea of the intensity level of the speech signal and also helps in identifying the closure duration of stops. So, vocal tract shape along with pitch and energy can be used as feedback parameters, as they provide sufficient perceivable information regarding speech production.

A PC based aid can be used to extract and display speech parameters, in real-time. The deaf person who is being trained, can speak in to the microphone of such an aid and can see the speech parameters on the screen. Such an aid can also provide option for slow motion review mode and a facility to compare his speech parameters with that of the person training him.

For some utterances like fricatives, stops, constriction or complete closure of the vocal tract takes place. During this period, the signal level is low, making the estimation unreliable. However the vocal tract shape just before and after such a period gives an idea of the vocal tract shape during this period. During such a period, the vocal tract shape can be frozen to the vocal tract shape of the previous frame. Such a frame can be detected by the energy level of the speech signal.

1.2 Development of Speech Training Aid at IIT-Bombay

There has been efforts, at IIT Bombay, to develop a speech training aid [1, 5, 7, 20, 8, 6]. The hardware set up is as shown in Fig. 1.1. The speech is sensed by a microphone and conditioned by the analog signal conditioning circuit. This speech signal is then processed on the signal processor, and the extracted parameters are transferred to the PC for displaying. The need for use of separate signal processor arises because achieving real-time operation is not feasible using a PC itself. Gupte [7], selected a particular algorithm for vocal tract shape estimation and for pitch estimation. He implemented them first off-line and then on a TMS320C10 EVM board, and developed hardware for interfacing with PC. Even though processing could be achieved in real-time, transfer of data from the DSP board to PC for real-time operation could not be achieved due to memory limitation of the DSP board.

Subsequently Taklikar [20], worked on a system for off-line implementation, to produce a more realistic display of vocal tract shape. Grecious [6], worked on refining the pitch estimation and its implementation, off-line as well as in assembly language of TMS320C10.

Khambete [8], implemented the speech training aid that operated in both real-time and slow motion review modes. He used signal processor board based on TMS320C25 for extracting the vocal tract area (i.e., acquiring and processing of speech input) and PC for displaying vocal tract as well to provide user interface. But he was not able to display the pitch along with vocal tract.

Bhagwat [1] was able to provide simultaneous graphical display of vocal tract shape, pitch, and energy, in real-time by modifying the system developed by Khambete. He achieved the high speed display(which made it possible to display pitch and energy along with vocal tract shape) by directly writing the image on to the EGA RAM. He also reduced the frame length from 300 samples to 256 samples. So the total delay was 25.6 ms (at 10 k Sa/s).

Gavankar [5], tested the system developed till then. He also devised a scheme for performing the overlapping of the window. In this report the system developed by Gavankar will be referred to as Speech Training System 1, STS-1.

1.3 Project Objective

The project aims at developing a system which estimates and displays vocal tract shape, pitch and energy, in real-time, based upon the input speech. The system STS-1 (as explained in sec. 1.2), apparently achieves this for vowels. But this system

has to be tested for accuracy and consistency, both for synthetic and natural speech input.

The displaying of vocal tract shape in STS-1 is done as a set of straight lines, making the interpretation difficult. So a suitable curve fitting algorithm has to be found and implemented. This should operate in real-time.

Further the prediction of vocal tract shape during closure duration of stops in STS-1 is not reliable, as the energy during this period is low. Hence the vocal tract shape during this period has to be decided based upon vocal tract shape, pitch, and energy just before and/or after stops. So for analyzing vocal tract shape, pitch, and energy variations, a suitable software has to be developed.

1.4 Report Outline

In the second chapter, estimation of energy, pitch, and vocal tract shape are discussed. The linear predictive coding, used in vocal tract shape estimation and its implementation on fixed point machine is described. The schemes for estimating the pitch and energy are also explained.

To help the testing, a feature which gives a two-dimensional representation of the variation of vocal tract shape, called 'areagram', was added to STS-1 and this modified system will be referred to as STS-2.In the third chapter, the hardware setup, the software features, and testing of STS-2, are briefly explained. The results of testing, are stated. The fourth chapter describes the software that were developed during the course of the project to achieve the objectives of the project. The division of tasks between PC and the DSP modules of the software, areagram, the curve-fitting algorithm for interpolation of vocal tract area function and the modified window overlapping technique, are explained. The system STS-2 after addition of these features will be referred to as STS-3.

The testing and results of STS-3 are explained in fifth chapter. Details of a software which generates spectrogram, that was developed to help the analysis is given in Appendix-A. The program listings and areagrams of some of the VCVs are given in Appendix-B

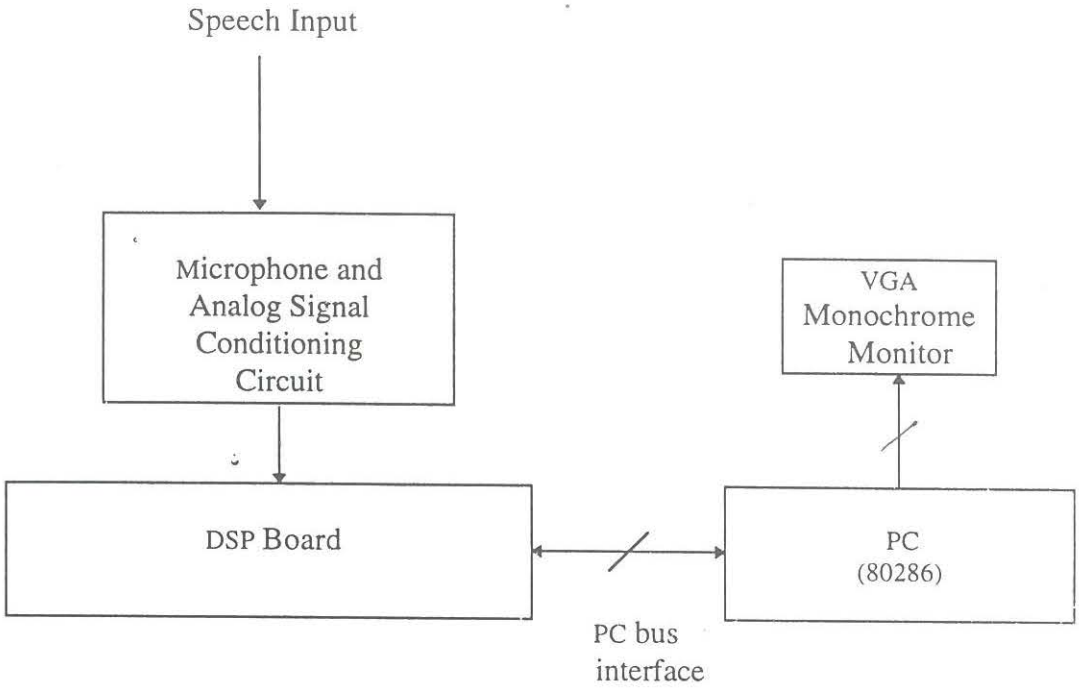


Figure 1.1: Block diagram of hardware setup

Chapter 2

Estimation of Energy, Pitch, and Vocal Tract Shape

2.1 Introduction

In this chapter, techniques for estimation of energy, pitch, and vocal tract shape will be presented. Estimation of the vocal tract shape from the speech input involves, obtaining solutions to a set of equations, of an assumed mathematical model of human speech production system. The mathematical model of human speech production system and two algorithms, which give the solutions to these mathematical equations are discussed.

2.2 Energy Estimation

Short time energy function is a convenient way of representing the amplitude variations of speech signal, and is defined as,

$$E_n = \sum_{m=0}^{N-1} (x(n-m)w(m))^2 \tag{2.1}$$

where E_n is the energy of the sequence when the window is placed at sample n. $x(.)$ is the speech signal and $w(.)$ is the window function of length N. The window length has to be carefully chosen. It can not be too large, as the estimated energy will then change slowly and will not indicate the sub-phonemic intensity variations of speech. At the same time it can not be too small (smaller than pitch period), as the estimated energy then fluctuates rapidly over individual pitch period. This problem is complicated by variation of pitch from about 20 samples for a high pitch female and child voice up to 250 samples (at 10 k Sa/s) for a low pitch male voice

[18]. So no single value gives satisfactory results for the entire range. But, generally a value of 100-200 samples (for 10 k Sa/s) gives acceptable results.

2.3 Pitch estimation

An accurate estimation of pitch, is an important problem in speech processing. Pitch detectors are widely used in vocoders, speaker identification and verification systems. They also find application in aids-to-the handicapped. There are several reasons which make the estimation a difficult task, like,

- The glottal excitation varies with time .
- The true periodic structure may be masked due to the vocal tract filter.
- The voiced/unvoiced distinction becomes difficult at low signal levels.

Because of its importance, many solutions have been proposed, like, spectrographic method, linear predictive analysis, and time domain algorithms [18]. In this section, two important time domain algorithms are described. Time domain algorithms are considered because of the need for real-time operation of the system.

2.3.1 Short time auto correlation method

The short time auto correlation is defined as,

$$\overline{r}_n(k) = \sum_{m=0}^{N-1-k} [x(n-m)w(m)][x(n-m+k)w(m-k)] \tag{2.2}$$

$\overline{r}(k)$ has the following important properties,

1. If the input is periodic with p period then, $\overline{r}(k) = \overline{r}(k+p)$
2. It is an even function, i.e.,$\overline{r}(k) = \overline{r}(-k)$
3. It has maximum at $k = 0$ and $\mid \overline{r}(k) \mid \leq \overline{r}(0)$, for all k.

The equality occurs when $k = p$, the period of the input signal. The properties indicate that the short time auto correlation peaks at shift equal to integral multiple of pitch period and hence can be used for pitch detection. However because of windowing, the value of auto correlation tapers down as the value of k increases. This may lead to false detection of pitch due to secondary peaks which occur because of vocal tract response. So the signal has to be pre-processed to eliminate these

secondary peaks. One such method is center clipping [18]. In center clipping, the output is made equal to zero if the input is below a certain threshold and made equal to input if the input is above the threshold. The threshold can be chosen to be a constant value or can be chosen as a percentage (typically 30%-50%) of the peak signal value in that window. The later method has an advantage in that it adapts to the signal level variation but is slightly more complex.

2.3.2 Short time average magnitude difference method

The drawback of short time auto correlation method is that it involves considerable amount of computation. A function called average magnitude difference function, which eliminates multiplications is defined below and can be used for pitch estimation.

$$d(k) = \sum_{m=0}^{N-1} \mid x(n-m)w(m) - x(n-m+k)w(m-k) \mid \tag{2.3}$$

If the signal is periodic, then $d(k)$ will be minimum for $k = 0, \pm p, \pm 2p \ldots$ where p is the period of the signal $x(.)$. This method has definite advantage, especially when estimation is to be done in real time and the machine on which it is done consumes more time for multiplication.

2.4 Vocal Tract Shape Estimation

The estimation of vocal tract shape is essentially the estimation of cross-sectional area along the vocal tract. The vocal tract area can be estimated using Linear Predictive Coding (LPC). In this section a model for speech production system and LPC are explained. The vocal tract area can be related (as will be shown in this section) to the 'reflection coefficients of speech production model. These reflection coefficients can be found from 'PARCOR' (PARtial CORrelation) coefficients of LPC. The relation between PARCOR (PARtial CORrelation) co-efficients of LPC and reflection co-efficients of speech production model is also explained.

2.4.1 Model for speech production system

A block diagram of speech production model is shown in Fig. 2.1. In this model, the speech signal is modeled as the output of a cascade of three filters driven by an impulse train or white noise,

$$S(z) = G(z)\ V(z)\ R(z)\ U(z) \tag{2.4}$$

where $S(z)$ is the Z-transform of speech signal, $G(z)$ and $R(z)$ are the source and radiation characteristics and $V(z)$ is the vocal tract transfer function. $U(z)$ is the excitation and usually modeled as train of impulses or Gaussian white noise for voiced and unvoiced sounds respectively.

The vocal tract can be thought of as concatenation of loss less tubes from glottis to lips [18]. The area function of the vocal tract relates the vocal tract cross-sectional area to the distance (from glottis to lip). If number of loss less tubes is assumed to be sufficiently high, then it approximates continuously varying area function. This model is shown in Fig. 2.2.

The discussion is based on [23]. Let $u_m(t,d)$ and $p_m(t,d)$ represent the volume velocity and pressure, respectively, in section m. Here t indicates time variable and d the distance variable. The wave equation for the velocity potential $\Phi_m(t,d)$, for section m is given by

$$\frac{\partial^2}{\partial d^2}\phi_m(t,d) = \frac{1}{c^2}\ \frac{\partial^2}{\partial t^2}\phi_m(t,d) \tag{2.5}$$

where c is the velocity of sound.

The velocity potential, volume velocity, and pressure are related by,

$$u_m = -A_m\ \frac{\partial}{\partial d}\Phi_m(t,d) \tag{2.6}$$

$$p_m = \rho\ \frac{\partial}{\partial t}\Phi_m(t,d) \tag{2.7}$$

where A_m is the cross sectional area of the m^{th} section, and ρ is the density of air. A solution to eq. 2.5 is

$$\Phi_m(t,d) = \Phi_m^+(t,d) + \Phi_m^-(t,d) = r_+ e^{jw(t-d/c)} + r_- e^{jw(t+d/c)} \tag{2.8}$$

where r_+ and r_- are constants, and w is the angular velocity. From eq. 2.6, 2.7, 2.8 we can get

$$u_m(t,d) = u_m^+(t,d) - u_m^-(t,d) \tag{2.9}$$

$$p_m(t,d) = p_m^+(t,d) + p_m^-(t,d) = \frac{\rho c}{A_m}\{u_m^+(t,d) + u_m^-(t,d)\} \tag{2.10}$$

where $u_m^+ = (jwA_m r_+/c)e^{jw(t-d/c)}$ and $u_m^- = (jwA_m r_-/c)e^{jw(t+d/c)}$
As volume velocity and the pressure are continuous, we can write, for the boundary between mth and $m+1$th section,

$$u_{m+1}(t,d_m) = u_m(t,d_m) \tag{2.11}$$

$$p_{m+1}(t,d_m) = p_m(t,d_m) \tag{2.12}$$

Since it is assumed that the tubes are loss less, $u^+_{m+1}(t, d_m)$ is equivalent to the component of volume velocity that started at d_{m+1} at time $\Delta t = \Delta l/c$ earlier. Similarly $u^-_{m+1}(t, d_m)$ is equivalent to volume velocity that will arrive at d_{m+1} at time Δt later. Hence the notation can be simplified by eliminating the distance variable, i.e., $u^+_m(t, d_m) = u^+_m(t)$. The continuity equations eq. 2.11 and 2.12 become,

$$u^+_{m+1}(t - \Delta t) - u^-_{m+1}(t + \Delta t) = u^+_m(t) - u^-_m(t) \tag{2.13}$$

$$\frac{\rho c}{A_{m+1}}\{u^+_{m+1}(t - \Delta t) + u^-_{m+1}(t + \Delta t)\} = \frac{\rho c}{A_m}\{u^+_m(t) + u^-_m(t)\} \tag{2.14}$$

By rearranging eq. 2.13 and eq. 2.14we get,

$$u^+_{m+1}(t - \Delta t) = \frac{1}{1 + r_m}\ \{u^+_m(t) - r_m u^-_m(t)\} \tag{2.15}$$

and

$$u^-_{m+1}(t + \Delta t) = \frac{1}{1 + r_m}\ \{-r_m u^+_m(t) + u^-_m(t)\} \tag{2.16}$$

where, r_m is referred to as reflection coefficient and is given by,

$$r_m = \frac{A_m - A_{m+1}}{A_m + A_{m+1}} \tag{2.17}$$

By taking Z transforms of eq. 2.15 and 2.16 and combining them, we can write,

$$\begin{bmatrix} U^+_{m+1}(z) \\ U^-_{m+1}(z) \end{bmatrix} = \frac{z^{1/2}}{1 + r_m} \begin{bmatrix} 1 & -r_m \\ -r_m z^{-1} & z^{-1} \end{bmatrix} \begin{bmatrix} U^+_m(z) \\ U^-_m(z) \end{bmatrix} \tag{2.18}$$

This is because z is chosen to be $e^{jw2\Delta t}$ instead of $e^{jw\Delta t}$. Now let us assume that the front end (lip end) is connected to a tube of infinite area (i.e., $r_0 = 1$) and that the back end (glottis end) is connected to a tube of area A_{M+1} that is terminated with characteristic impedance of $\rho c/A_{M+1}$. Also let us assume that the tube is excited at the back end by a source with this characteristic impedance, and with forward going volume velocity of $U^+_{M+1}(z)$. We can write

$$\begin{bmatrix} U^+_{m+1}(z) \\ U^-_{m+1}(z) \end{bmatrix} = z^{m+1/2} g_m \begin{bmatrix} D^+_m(z) \\ D^-_m(z) \end{bmatrix} \{U^+_0(z) - U^-_0(z)\} \tag{2.19}$$

where,

$$g_m = \prod_{i=0}^{m} \frac{1}{1 + r_i} \tag{2.20}$$

and

$$\begin{bmatrix} D_m^+(z) \\ D_m^-(z) \end{bmatrix} = \begin{bmatrix} 1 & -r_m \\ -r_m z^{-1} & z^{-1} \end{bmatrix} \begin{bmatrix} 1 & -r_{m-1} \\ -r_{m-1} z^{-1} & z^{-1} \end{bmatrix} \cdots \cdots \begin{bmatrix} 1 & -r_1 \\ -r_1 z^{-1} & z^{-1} \end{bmatrix} \begin{bmatrix} 1 \\ -z_{-1} \end{bmatrix} \tag{2.21}$$

now let us normalize $U_{m+1}^+(z)$ and $U_{m+1}^-(z)$ by g_m,

$$\begin{bmatrix} \hat{U}_{m+1}^+(z) \\ \hat{U}_{m+1}^-(z) \end{bmatrix} = z^{m+1/2} \quad \begin{bmatrix} D_m^+(z) \\ D_m^-(z) \end{bmatrix} \{U_0^+(z) - U_0^-(z)\} \tag{2.22}$$

where $\hat{U}_{m+1}^{+/-}(z) = U_{m+1}^{+/-}(z)/g_m$.
The inverse transfer function between section m and front end is defined as,

$$\begin{aligned} C_m^+(z) &= \frac{\text{forward going volume velocity component at the end of } m \text{ sec.}}{\text{volume velocity at the front end}} \\ &= \frac{\hat{U}_{m+1}^+(z)}{U_0^+(z) - U_0^-(z)} \end{aligned} \tag{2.23}$$

similarly we can define

$$C_m^-(z) = \frac{\hat{U}_{m+1}^-(z)}{U_0^+(z) - U_0^-(z)} \tag{2.24}$$

so, from eq. 2.22, 2.23, 2.24 we can arrive at

$$\begin{aligned} \begin{bmatrix} C_{m+1}^+(z) \\ C_{m+1}^-(z) \end{bmatrix} &= z^{m/2+1} \begin{bmatrix} D_{m+1}^+(z) \\ D_{m+1}^-(z) \end{bmatrix} \\ &= z^{m/2+1} \begin{bmatrix} 1 & -r_{m+1} \\ -r_{m+1} z^{-1} & z^{-1} \end{bmatrix} \begin{bmatrix} D_m^+(z) \\ D_m^-(z) \end{bmatrix} \end{aligned} \tag{2.25}$$

Since $z^{m/2+1}$ is just a delay, the inverse transfer function can be written as the following recursive equation,

$$\begin{bmatrix} D_{m+1}^+(z) \\ D_{m+1}^-(z) \end{bmatrix} = \begin{bmatrix} 1 & -r_{m+1} \\ -r_{m+1} z^{-1} & z^{-1} \end{bmatrix} \begin{bmatrix} D_m^+(z) \\ D_m^-(z) \end{bmatrix} \tag{2.26}$$

2.4.2 Linear predictive coding

In sec. 2.4.1, a recursive equation for inverse transfer function of the speech production model was explained. In this section, the estimation of the reflection coefficients, using LPC, is discussed. The speech analysis model is shown in the Fig. 2.3. In this model , impulse train at the input is analogous to the glottal excitation and filter is analogous to the vocal tract with glottal and radiation characteristics. Also the filter is assumed to have only poles. This implies that the inverse filter is having only zero's. Hence the Z-transform of the inverse filter of order M is given by,

$$A(z) = \sum_{i=0}^{M} a_i z^{-i}, \qquad a_0 = 1 \tag{2.27}$$

The error is given by,

$$\in_n = \sum_{i=0}^{M} a_i x_{n-i} - \eta\delta_{n0} \tag{2.28}$$

where x_i is the input signal to inverse filter and η is the amplitude of the impulse, which is input to the filter, as shown in Fig. 2.3
Least mean square error technique is applied to the eq. 2.28 i.e.,

$$\frac{\partial E}{\partial a_j} = 0 \qquad \text{for} j = 1, 2, \ldots, M \tag{2.29}$$

where E is the sum of $\in_n^2$ over $0 \leq n \leq N+M-1$
The above equation is equivalent to,

$$2\sum_{i=0}^{M} a_i \overline{r}_{i-j} - 2n \sum_{n=0}^{N+M-1} x_{n-j}\delta_{no} = 0, \qquad \text{for } j = 1, 2, \ldots, M \tag{2.30}$$

Where $\overline{r}_k$ represents the k^{th} auto correlation of the sequence x_n. As it is assumed that x_n is non-zero for $0 \leq n \leq N-1$, the above eq. 2.30 reduces to,

$$\sum_{i=0}^{M} a_i \overline{r}_{i-j} = 0 \tag{2.31}$$

Now for arriving at a recursive algorithm, we use inner product approach [12]. The inner product is defined as,

$$< F(z), G(z) > \quad \triangleq \quad \frac{1}{2\pi}\int_{-\pi}^{\pi} R(e^{j\theta}) F^*(e^{i\theta}) G(e^{j\theta}) d\theta \tag{2.32}$$

where $R(e^{j\theta})$, $F^*(e^{j\theta})$,$G(e^{j\theta})$, are the Z-transforms, evaluated at
$z = e^{j\theta}$, of $\overline{r}_k, f_k^*, g_k$. Now let us assume that the filter co-efficients are found for

m degree. From this we shall find out the filter co-efficients for $m+1$ degree. Let $A_m(z)$ represent the inverse filter of degree m, given by,

$$A_m(z) = \sum_{i=0}^{m} a_{m,i} z^{-i}, a_{m,0} = 1 \qquad \forall m \tag{2.33}$$

from eq. 2.31, 2.32, 2.33 it can be concluded that

$$< A_m(z), z^{-l} >= 0 \qquad \text{for } l = 1, 2, \ldots, m \tag{2.34}$$

this can be written as

$$< z^{-m-1+l}, z^{-m-1} A_m(1/z) >= 0 \tag{2.35}$$

now define,

$$B_m(z) \triangleq z^{-m-1} A_m(1/z) \tag{2.36}$$

by putting $k = m + 1 - l$ in eq. 2.34, 2.35 and using eq. 2.36, we can write

$$< z^{-k}, B_m(z) >= 0, \qquad \text{for } k = 1, 2, \ldots, m \tag{2.37}$$

also from eq. 2.33 and 2.36

$$B_m(z) = \sum_{k=1}^{m+1} b_{m,k} z^{-k} \tag{2.38}$$

Now we have got two polynomials of degree m, $A_m(z)$ and $m+1$, $B_m(z)$, which are orthogonal to $z^{-1}, \ldots, z^{-m}$. We have to obtain a polynomial $A_{m+1}(z)$ of degree $m+1$, such that it is orthogonal to $z^{-1}, \ldots, z^{-(m+1)}$. This can be obtained as a linear combination of $A_m(z)$ and $B_m(z)$ i.e.,

$$A_{m+1}(z) = A_m(z) + k_m B_m(z) \tag{2.39}$$

As $A_{m+1}(z)$ is orthogonal to $z^{-(m+1)}, k_m$, which is referred to as PARCOR(PARtial CORrelation) coefficients in literature, can be found using the following relations,

$$\beta_m \triangleq < z^{-(m+1)}, A_m(z) > \tag{2.40}$$

$$\alpha_m \triangleq < z^{-(m+1)}, B_m(z) > \tag{2.41}$$

$$k_m = \frac{\beta_m}{\alpha_m} \tag{2.42}$$

Since k_m can be computed from already known values, we can find out $A_{m+1}(z)$, and hence $B_{m+1}(z)$. So the filter co-efficients of $m+1$ stage can be calculated using the following equations,

$$\begin{aligned} a_{m+1,l} &= a_{m,l} = 1 && l = 0 && (2.43)\\ &= a_{m,l} + k_m a_{m,m+1-l} && l = 1, 2, \ldots, m && (2.44)\\ &= k_m && l = m+1 && (2.45) \end{aligned}$$

Computation of α_{m+1} and β_{m+1} can be done using the relations,

$$\alpha_{m+1} = \alpha_m(1 - k_m^2) \tag{2.46}$$

$$\beta_{m+1} = \sum_{l=0}^{m+1} a_{m+1,l}\overline{r}_{m+1-l} \tag{2.47}$$

The initial conditions are $a_{0,0} = 1$, $\alpha_0 = \overline{r}_0$, $\beta_0 = \overline{r}_1$

To obtain the relation between these kms and the reflection coefficients of the sec. 2.4.1, we can re-write eq. 2.39 as,

$$A_{m+1}(z) = A_m(z) - k_m\hat{B}_m(z) \tag{2.48}$$

$$\text{where } \hat{B}_m(z) = -B_m(z)$$

$$\text{and} \qquad \hat{B}_{m+1}(z) = \tfrac{1}{z}\{\hat{B}_m(z) - k_m A_m(z)\} \tag{2.49}$$

eq. 2.48 and 2.49 can be combined and written in matrix form as,

$$\begin{bmatrix} A_{m+1}(z) \\ \hat{B}_{m+1}(z) \end{bmatrix} = \begin{bmatrix} 1 & -k_m \\ -k_m z^{-1} & z^{-1} \end{bmatrix} \begin{bmatrix} A_m(z) \\ \hat{B}_m(z) \end{bmatrix} \tag{2.50}$$

Comparing eq. 2.26 and 2.50 we can arrive at the relation between reflection coefficients and PARCOR coefficients,

$$r_m = k_{m-1} \tag{2.51}$$

2.5 Fixed Point Algorithm for LPC

The estimation of reflection co-efficients has to be done in real-time, to achieve the goal of this project. One possible way of doing this is to do the processing in a DSP processor. Khambete [8] selected a particular hardware setup which is based on TMS320C25, a fixed point DSP processor. So the algorithm that estimates the reflection co-efficients must be implementable on a fixed point processor. One such algorithm which can be implemented on a fixed point processor is explained in this section.

2.5.1 Recursive algorithm

The discussion is based on [19]. The approach is similar to the approach discussed in sec. 2.4.2. The problem with the algorithm, discussed in sec. 2.4.2, is that the range of variation of a_i is not known. Hence it is difficult to implement that algorithm on a fixed point arithmetic machine. Here, a new intermediate variable, defined as,

$$e_i^m = \quad < A_m(z), z^{-i} > \quad = \sum_{k=0}^{m} a_{k,m} r_{i-k} \qquad \forall i \tag{2.52}$$

is introduced. using eq. 2.34 we can write $e_i^m = 0$ for $i = 1, 2, \ldots, m$
The recursive algorithm is obtained as follows,

$$B_m(z) \triangleq z^{-m-1} A_m(1/z) \qquad \Rightarrow \qquad < B_m(z), z^{-i} >= e_{m+1-i}^m \tag{2.53}$$

we now obtain $A_{m+1}(z)$ as a linear combination of $A_m(z)$ and $B_m(z)$ i.e.,

$$A_{m+1}(z) = A_m(z) + k_m B_m(z) \tag{2.54}$$

As $A_{m+1}(z)$ has to be orthogonal to $z^{-1}, \ldots, z^{-(m+1)}$, we can arrive at the following recursive equations,

$$k_m = -e_{m+1}^m / e_0^m \tag{2.55}$$

$$e_0^{m+1} = e_0^m (1 - k_m^2) \tag{2.56}$$

$$e_i^{m+1} = e_i^m + k_m e_{m+1-i}^m \qquad \forall i \tag{2.57}$$

The algorithm can be understood from the flow graph given in Fig. 2.4.

2.5.2 Fixed point implementation

From Cauchy-Schwartz inequality,

$$| e_i^m |^2 \quad = \quad |< A_m(z), z^{-i} >|^2 \quad \leq \quad < A_m(z), A_m(z) >< z^{-i}, z^{-i} > \tag{2.58}$$

$$\text{but } < z^{-i}, z^{-i} >= \overline{r}_0 \text{ and } < A_m(z), A_m(z) >= e_0^m \tag{2.59}$$

using eq. 2.58 and 2.59, we can write

$$| e_i^m |^2 \quad \leq \quad e_0^m \overline{r}_0 \qquad \forall i \tag{2.60}$$

Also it can be proved that [12] ,

$$e_0^m = \overline{r}_0 \prod_{i=0}^{m-1} (1 - k_i^2) \qquad | k_i | \leq 1 \tag{2.61}$$

Hence $| e_i^m | \leq \overline{r}_0$. So if we assume that $\overline{r}_0 \leq 1$, then $| e_i^m | \leq 1 \; \forall i$, and hence can be implemented on fixed point arithmetic machine.

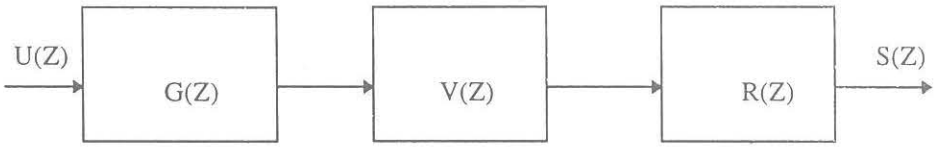
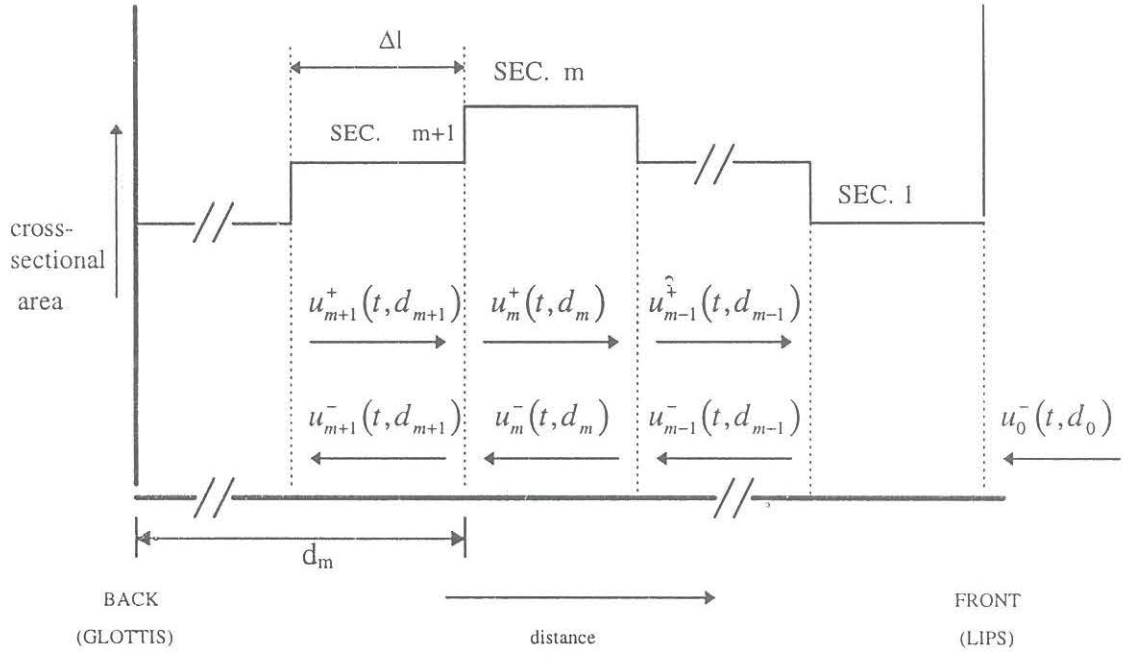


Figure 2.1: Block diagram of speech production model



Where d_m is the distance between the boundary of m th and $m+1$ th sections, and the glottis, and Δl is the length of each section.

Figure 2.2: A non-uniform acoustic model of voal tract

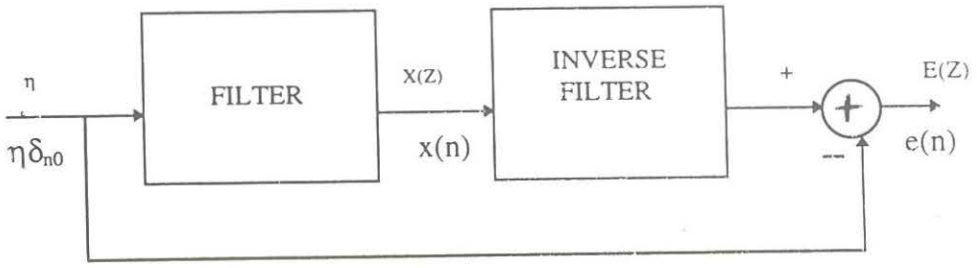


Figure 2.3: Speech analysis model

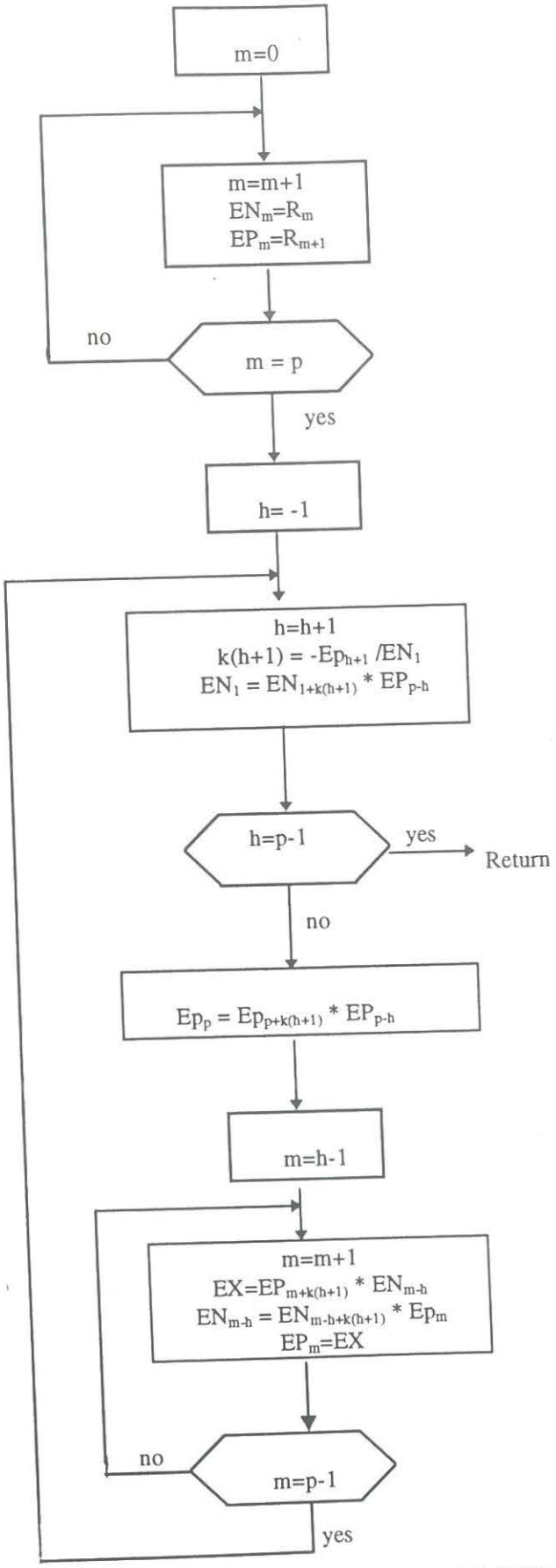


Figure 2.4: Computation of PARCOR coefficients

Chapter 3

A Speech Training System, 'STS-2'

3.1 Introduction

In section 1.2, the development of speech training system, at IIT Bombay, was discussed. To achieve the objectives of this project, it was decide to retain the same hardware set up as that of STS-1, and modify the software. The block diagram of the hardware setup is shown in Fig. 3.1 The speech signal from an electret microphone is amplified and filtered by the signal conditioning circuit. The output of the signal conditioning block is fed to the on-board ADC of the DSP board. The DSP board is used for acquiring (at a sampling rate 10 k Sa/s) and processing the data. A PC is used for providing the user interface and for displaying the speech parameters (read from the shared memory space on the DSP board) on a VGA monochrome monitor. The individual blocks of the hardware setup is explained in sec. 3.2

The software of STS-1 has basically two modules. One runs on the DSP board and the other on the PC. The tasks between the two modules are divided so as to exploit the fast arithmetic operations and powerful instruction set of the DSP board and floating point arithmetic operations as well as display capabilities of the PC. The software is explained in sec. 3.3

As pointed out earlier the system STS-1 was tested by Gavankar only for synthetic vowel inputs. Hence the system needs to be tested for natural as well as synthetic and natural speech corrupted with white noise. To help the testing of the system, a feature which generates areagram (a two-dimensional representation of the variation of vocal tract shape, from glottis to lip, with time) was added. This modified system will be referred to as STS-2 in the report. The section 3.4 explains the testing and results of STS-2.

3.2 Hardware Setup

The important blocks of the hardware setup, shown in Fig. 3.1, viz. microphone and analog signal conditioning circuit, DSP board and PC & monitor, are explained in this section.

1. Microphone and Analog Signal Conditioning Circuit.

The electret microphone generates a voltage which is typically ten's of mV. This is amplified by the preamplifier of the analog signal conditioner to $\pm$ 10 V to make full use of the dynamic range of the ADC. The block diagram of the analog signal conditioning unit is shown in Fig. 3.2. As can be seen, this unit can be used for both inputting and outputting analog signals. The input part has a pre-amplifier (as mentioned earlier), gain of which can be varied. At the input of this preamplifier, a simple RC first order high pass filter is provided, with cutoff frequency of 60 Hz., for isolating the dc offset voltages. This pre-amplifier is followed by an active seventh order elliptic low pass filter (anti-aliasing filter). The filter has pass band up to 4.6 kHz with a pass band ripple of 0.3 dB, and stop band starts at 5 kHz with a minimum attenuation of 40 dB. The filter specifications were chosen for avoiding aliasing due to sampling at the rate of 10 k Sa/s.

The output board consists of 4 Modules. They provide,

- $\pm 5V$ output as a test signal.
- $\pm 300mV$ output as recording signal, for tape recorder.
- $\pm 20mV$, which can act as output to a head phone.
- output of power amplifier, which can drive a loud speaker $(0.5W, 4\Omega)$

The module requires a supply of $\pm 12V$ dc.

2. PCL DSP Board

This board, from dynalog micro systems, is based on TMS320C25 digital signaling processor, and operates at 40 MHz. This can be used as an add on board to the PC mother board. This board has on board 64 k word program memory and 64 k word data memory. This memory area can be accessed directly by the PC (shared memory) and hence the data transfer between C25 and the PC can be very fast. It has on board ADC (with maximum conversion time of 35 μs), a 16 bit programmable timer, clocked at 5 MHz [17] It also has an on-board DAC.

The TMS320C25 processor has 544 words on chip RAM (256+32 words of data ram and another 256 words which can be programmed either as program or as data memory). It provides a very powerful instruction set for digital signal processing applications. The processor has an instruction cycle of 100 ns. and most instructions

are of single cycle.
3. PC and Monitor

A PC/AT is used for the purpose of displaying the speech parameters (vocal tract shape, pitch, and energy), providing user interface, and for the calculation of the area function (as this requires floating point arithmetic). A VGA monochrome monitor is used for displaying the parameters.

3.3 Software

As stated earlier, the software is divided into two modules. The division of tasks between the two modules is shown in Fig. 3.3. The tasks are divided so as to exploit the fast arithmetic operation of TMS320C25 processor and floating point as well as display capabilities of the PC. The modules are explained in this section.

DSP module

This program is written in assembly language of TMS320C25 processor. The signal acquisition has to be done continuously at a rate of 10 k Sa/s, and data blocks of 256 samples, with 50% overlap, are to be processed for parameter estimation. The sampling is done with the help of an on-board timer, which generates interrupts corresponding to 10 k Sa/s. The samples are stored in a circular buffer of length 256. 50% overlap is performed on this data, which is explained in sec. 4.5 . The data block of 256 is Hamming windowed (as it gives better results [18]) and the estimation of reflection coefficients is done using Le Roux and Gueguen algorithm as explained in sec. 2.5 The energy is calculated as zeroeth auto correlation. The input signal is then center clipped with a thresh hold of 50% of the peak value in that frame. Pitch is then estimated from the center clipped data using auto correlation method, described in sec. 2.3.1

PC module

This program is written in C. This provides the user interface and also initializes the DSP board apart from displaying the speech parameters. The reflection co-efficients along with pitch and energy are read from the DSP board and the area of the m'th tube, A_m is calculated using the relation,

$$A_m = \frac{1+r_m}{1-r_m} A_{m+1}, \qquad 0 \leq m < 12 \tag{3.1}$$

where $A_{12} = 1$ and r_m is the m'th reflection co-efficient. The speech parameters are then displayed by directly writing the images on to the video RAM of VGA.

Also entire screen is not modified, only the required portions of the previous image are modified. This method increases the speed of display. The program provides facilities like, capturing image, displaying the captured image in slow motion along with the image generated in real time.

3.4 Testing and Results

The system STS-1 was tested by Gavankar [5], only for synthesized vowels. Hence the system needs to be tested for natural speech and synthetic/natural speech corrupted with noise. To help the testing, a new feature which generates areagram of 100 frames was added to STS-1. Testing of this modified system (which will be referred to as STS-2), was carried out. Areagram, as mentioned earlier, helps in analyzing the variation of vocal tract shape with time. The vertical axis represents the distance (from glottis to lip). This is divided in to 12 sections (as the vocal tract is modeled as a 12 tube model). The horizontal axis represents the frame number (each frame is equivalent to 12.8 ms). The vocal tract area is coded as the intensity of the pattern.

The pitch estimation was tested with sinusoidal wave and natural speech. The estimator was also tested for these inputs with different signal to noise ratios (The resolution of displayed pitch is 5Hz, because 0-500Hz is mapped onto 100 pixels). The minimum input peak to peak voltage for the pitch estimator to work satisfactorily with in this resolution is tabulated in tab. 3.1. As can be observed, the minimum peak-to-peak voltage required, decreases as the frequency increase. Also the minimum peak-to-peak voltage required increases when the signal is corrupted with noise.

For sinusoidal wave, and for synthetic speech [11], the estimated pitch was compared with that of the input. For natural speech, the estimated pitch was compared with that observed in spectrogram. Fig. 3.4 shows the estimated pitch for synthetic vowel /a/ of pitch 140 Hz. The energy of the input speech was made initially to increase and then to remain constant at a particular level(for some time) and then to decrease again. The Fig. 3.4 shows the minimum energy required for consistent estimation. The testing was repeated with these speech signals affected with white noise. It was observed that the minimum energy required for accurate estimation increased (e.g., approximately 16% increase was needed for vowel /a/ with 12 dB SNR).

The testing of vocal tract shape can be done using an 'articulatory synthesizer', which generates speech sounds depending on the shape of articulators, input to it. The speech produced by such a synthesizer can be given as input to STS-2 and the vocal tract shape estimated by STS-2 can be compared with the input to the

synthesizer.

Due to the non-availability of such a synthesizer, the system could not be tested in this method. However the estimated shapes were compared with that given in Rabiner [18], for vowels. It was found that the results were comparable. The vocal tract areagrams for synthetic vowels /a/ and /i/ are given in Fig. 3.4 and Fig. 3.5 respectively. As can be seen it is consistent above a certain energy level. The system was again tested for the same synthetic vowels, but this time affected with noise. It was observed that, the minimum energy required for consistent estimation increased with increase in noise.

Table 3.1: Minimum peak to peak voltage(v) required for pitch estimation, for sine wave input

Frequency (Hz)	with out noise	with oise 12dB SNR	with noise 9dB SNR
100	3.6	4	4.8
200	2.8	3.6	4
300	2	2.8	3.6
400	1.6	2.4.	2.8

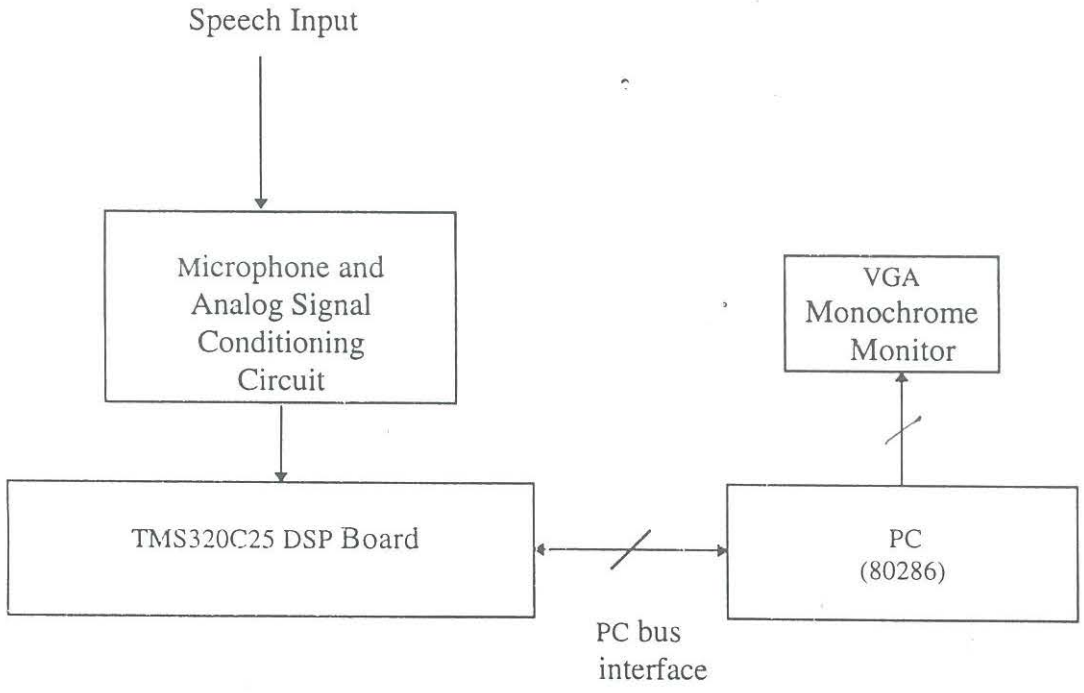


Figure 3.1: Block diagram of hardware setup of STS-1

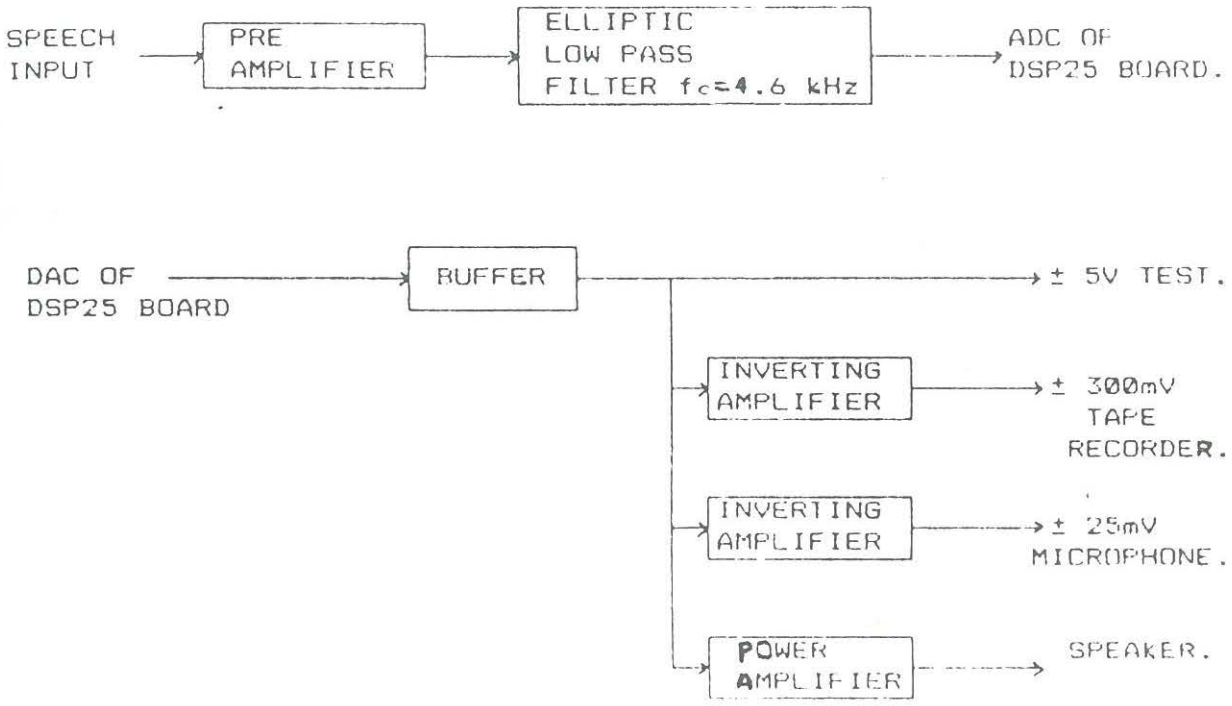


Figure 3.2: Block diagram of analog signal conditioning circuit of STS-1

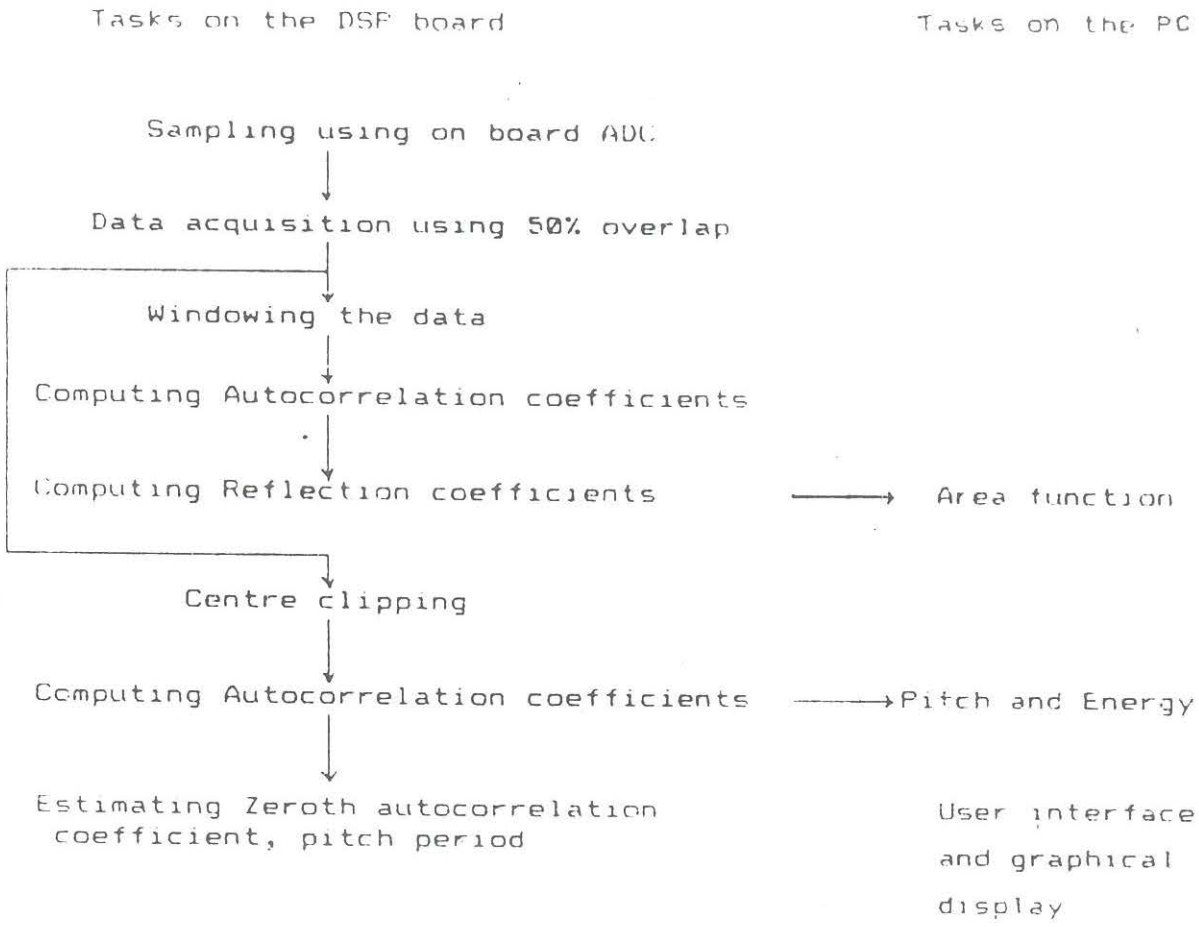


Figure 3.3: The allocation of tsaks between PC and DSP board

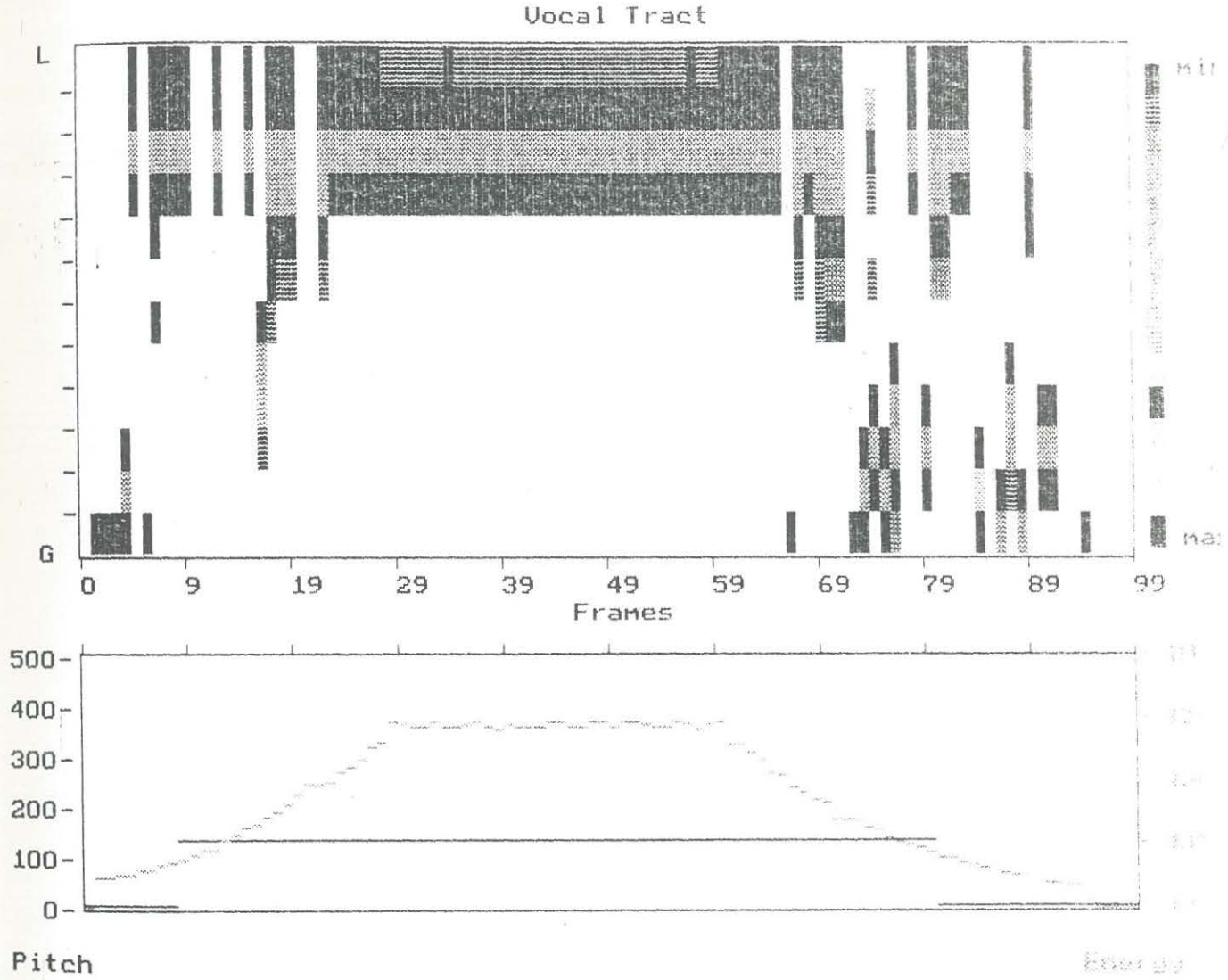


Figure 3.4: Estimated vocal tract shape, pitch, and energy for synthetic vowel /a/ of pitch 140 Hz, without noise

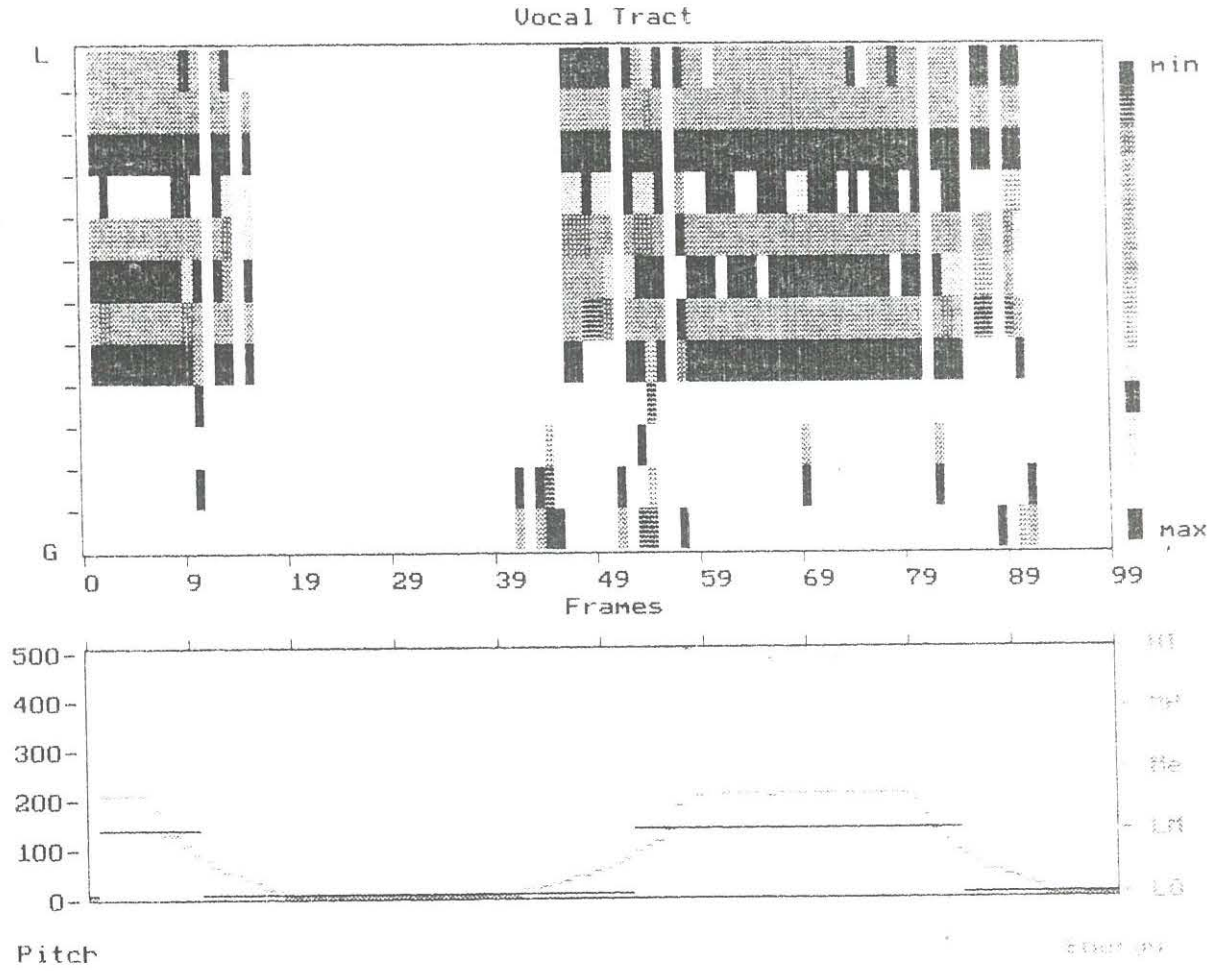


Figure 3.5: Estimated vocal tract shape, pitch, and energy for synthetic vowel /i/ of pitch 140 Hz, without noise

Chapter 4

Software Developments

4.1 Introduction

To achieve the aim of this project, the software of STS-1 was modified. As explained in sec. 3.1, a feature which generates areagram was added to STS-1 and the modified system is called as STS-2. STS-3 which is a modified version of STS-2, basically has the same software structure, i.e., it has two modules, one running on the DSP board and the other running on the PC. But in this system, a curve fitting algorithm for vocal tract shape smoothing has been incorporated and the vocal tract shape during the closure period is frozen to the vocal tract shape just before the closure period. The Fig. 4.1 shows the division of tasks between the two modules. The division of tasks, is similar to that of STS-2. All the computationally intensive operations are done by the DSP module while the displaying, user-interface and memory operations (like storing the vocal tract shape) are done by the PC module. In this chapter, the two modules of STS-3, the areagram and the curve-fitting algorithm which have been implemented in real-time, are explained.

While testing the STS-2, it was observed that neither the vocal tract shape nor the pitch and energy estimation were consistent. The reason for this was found to be loss of some data while window overlapping was being performed. Sec. 4.5 describes the modification that was carried out in the implementation of overlapping, to rectify this problem.

Spectrogram is one of the important tools in speech analysis. It was decided that a software which generates spectrogram was needed for analysis. A program has been developed, earlier at IIT Bombay [21], for generating spectrogram from the speech input. However, in this program all calculations (FFT, log magnitude) and displaying are done on the PC, making the process slow. So to enhance the speed of generation, it was decided to develop a software, which exploits fast number crunching capabilities of TMS320C25 processor and display capabilities of the PC.

Such a software was developed along with Prasad [16] and is explained in Appendix-A.

4.2 DSP Module

This module (v_ear.asm) is written in the assembly language of TMS320C25 processor. The tasks that are to be carried out by this module are selected, based on computational complexity involved. Both pitch and reflection coefficient estimations require auto-correlation coefficients, while the curve-fitting requires, approximately 704 multiplications and 528 additions per frame, and hence are computationally intensive. Hence these estimations/interpolation are done by the DSP module. The other important tasks carried out by this module, are, speech data acquisition and windowing.

The signal acquisition has to be done continuously at a rate of 10 k Sa/s, and data blocks of 256 samples, with 50% overlap, are to be processed for parameter estimation. The sampling is done with the help of an on-board timer, which generates interrupts corresponding to 10 k Sa/s. The samples are stored in a circular buffer of length 256. After the acquisition of every 128 samples, it combines 128 previous samples and these 256 samples are copied as a block to another memory area for processing. The signal acquisition and the data processing proceed in parallel. The details of overlapping are given in the sec. 4.5

The data block of 256 samples, is Hamming windowed and the auto-correlation coefficients are calculated. Then Le Roux-Gueguen algorithm is used to estimate the reflection coefficients. The predictor order used is 12, which gives satisfactory results for a sampling rate of 10 k Sa/s. The zeroeth auto-correlation coefficient, which is calculated for reflection coefficient estimation, gives an estimate of energy and hence no additional computation is involved for energy estimation. The program running on PC reads these reflection coefficients and the energy, calculates the vocal tract area, and then writes back on the DSP board. While the PC program is calculating the vocal tract area, the speech signal is center-clipped with a threshold of 50% of the peak value in that frame. Again auto-correlation coefficients are calculated and pitch is estimated using the auto-correlation method.

On the 12 area coefficients obtained from the PC, the curve-fitting algorithm, Bezier form [4], is used to interpolate the area function. The details of this algorithm are given in sec. 4.6. This generates a total of 176 points which are read and displayed by the program running on PC. The time taken for calculation of reflection coefficients is 1.8 ms, for calculation of pitch and curve fitting, it is 8.4 ms. So a total of 2.6 ms is free in a period of 12.8 ms.

4.3 PC Module

This module (v_ear.c), is written in C. The important tasks that are allotted to this module, are, providing user interface, initializing the DSP board, storing of various speech parameters, calculation of vocal tract area. from the reflection coefficients, generation of areagram, and displaying. The calculation of vocal tract area needs floating point arithmetic and hence can be done using the floating point capabilities of PC instead of doing it on the fixed point DSP processor. Similarly displaying and storing of various parameters are basically memory operations and are easily and efficiently handled by the PC.

The module initializes the DSP board, writes the window function, and the coefficients of curve fitting polynomial, onto the DSP board. These are one-time operations and will not be part of real time operations. The real time operations can be started by selecting the proper option of from the main menu. The real-time operation which begins then, is explained here. The module displays the speech parameters of previous frame (for the very first frame, the vocal tract shape is initialized to zero) by directly writing onto the VGA RAM, and waits for the DSP module to estimate the reflection coefficients of the present frame. It reads the reflection coefficients and energy from the DSP board, calculates area function and writes the normalized vocal tract area back to the DSP board, for curve fitting. If the energy is below a certain threshold, the vocal tract shape is not modified, i.e., freezed to the present shape. The vocal tract shape is frozen, till the energy again goes back above this threshold or the number of frames for which the energy was below this threshold exceeds a limit, (this is to ensure that the vocal tract shape is not frozen during the silence period) which is fixed to 10 frames. Both this limit and the energy threshold can be changed. It has to be noted that these levels were selected, after observing areagrams of different speech utterances. The freezing eliminates the randomness of the vocal tract shape during closure period of stop consonants. If the energy is above the threshold, then the speech parameters of previous frame is erased. The program then waits for the DSP module to estimate the pitch and perform curve-fitting. The interpolated vocal tract area function and pitch of the present frame are read from the DSP board after the DSP module indicates the completion of pitch estimation and interpolation. This process is repeated.

The module allows to store the speech parameter of 100 frames without affecting real time operation. This stored parameters can be used for review mode or for generating areagram (areagram is explained in sec. 4.4). In review mode, the stored parameters are displayed along with the parameters estimated in real time. This helps in training the deaf as, the speech parameters of teacher can be stored and that of the student can be compared with the stored parameters. At the time of

generation of areagram, the real time operation is suspended. Real time operation is restored after returning from the areagram display. The real time processing of the PC module takes approximately 8 ms.

4.4 Areagram

The estimation of vocal tract shape during the closure duration of stops, is particularly difficult or almost impossible using linear prediction as the energy during this period is very low. So, the vocal tract shape has to be determined from the vocal tract shape, pitch, and energy, just before and/or after the stops. It is therefore necessary to know the variation of these parameters, with time. A program which generates areagram and displays it along with pitch and energy, for 100 frames (equivalent to 1.28 s) was developed for this purpose.

Areagram is a two-dimensional representation of the variation of vocal tract cross-sectional area from glottis to lips, with time. The time (frame number) and distance from glottis to lips, are indicated by horizontal and vertical axes, the cross-sectional area is coded as the intensity of the pattern.

The areagram is 500 pixel wide and 176 pixel high. The width of the areagram is chosen to be 500, because it then can be compared with the spectrogram. In case of areagram 5 pixels correspond to one frame, i.e., 12.8 ms. Also this means that areagram displays area function of 100 frames (equivalent to 1.28 s) at a time. This period is sufficient to capture the variation of vocal tract shape for VCV syllables (like /apa/).

Areagram can be generated in two ways. One way is to capture the vocal tract shape, when the V_EAR program is running. This can be done by pressing function keys, F2 to start capturing and F4 to stop capturing. This captured vocal tract shape can be transformed in to an areagram by pressing the function key F3 after entering 'use-image' mode. This mode can be entered by pressing the function key F3 after capturing. The areagram can be stored in a file and later displayed using a program 'DISPY'.

The other way of generating areagram is particularly useful when the analysis involves the comparison of areagram with spectrogram on the same time scale. This requires the data to be in a data file (either binary or text) and should be of length 12800 samples. This file should be first downloaded on to the DSP board using 'F_LOAD' function. Then the 'OFF_LINE' program can be used to generate areagram, and can be stored in a file, which can be displayed using the program DISPY. The program DISPY provides option for storing the areagram as an image file, which can be then converted to different formats like, .ps, .gif, using 'xv' of X-windows, for making hard copies using a laser printer..

4.5 Modification of Window Overlapping Technique

Overlapping of the data window, intuitively would improve the estimation, as then, there may not be a sudden jump of data between the successive windows. A scheme for window overlapping was developed and implemented by Gavankar [5] and an improvement was observed in the estimation. However it was observed while testing STS-2, that neither the pitch estimation nor the vocal tract shape was consistent, even when the input was periodic. This was because of loss of some data, while overlapping was performed. In this section the implementation of window overlapping by Gavankar [5], and how the data loss occurs is explained. A modified method which, eliminates this loss of data was implemented and is explained in this section.

The window overlapping technique used in STS-2 is shown in Fig. 4.2. Initially the data is written in the location 4000h. After collecting 256 samples, this data (from 4000h thro' 4100h), is copied to the location 4200h for processing. After this again collection of data is started and this time it is stored in location starting from 4100h. After 256 samples are collected, data from locations 407fh thro' 417fh are moved to memory location starting from 4200h, and storing of new data is done from 4000h. After 256 samples are captured, the data from location 4100h thro' 41ff is transferred to 4200h for processing and new data is collected from 4100. Now, here the data gets mixed up as the data from 417fh thro' 41ffh gets overwritten by latest data and the locations 4000h thro' 40ffh would still have the old data.

So a modified version of overlapping is now used. This is explained using the Fig. 4.3. Initially the data is collected in location 4000h thro' 407fh. After 128 samples are collected, 256 samples starting from 4080h thro' 40ff and again from 4000h thro' 407fh are transferred to location 4200h for processing. While the processing is done, new data is collected in location starting from 4080h. Again after 128 samples are collected, data starting from 4000h thro' 40ffh is copied to location 4200h for processing and this process repeats.

4.6 Curve Fitting for Vocal Tract Area Function

The vocal tract shape, in STS-2 is displayed as a set of 12 straight lines. To make vocal tract shape look narural, this has to be properly interpollated using a curve fitting algorithm. Such a curve fitting algorithm, should be computationally efficient, enabling real-time implementation. This section describes a particular algorithm which was selected for this purpose and describes its real-time implementation.

4.6.1 Algorithm

Linear interpolation involves finding a suitable polynomial, from the given set of points. This polynomial is then used for finding the intermediate points. One of the well known methods of linear interpolation is Lagranges algorithm [10]. In Lagrange's interpolation method, a polynomial is found that passes through all the given points. In the present case, vocal tract shape estimator generates 12 points. One way of generating intermediate points is to find a polynomial which passes through all the 12 points. However this does not produce good results as the 11th degree polynomial can have large peaks and the interpolation may contain large errors (even if it is implemented on a floating point machine). So a better method would be to divide these 12 points into groups of 4 points and performing interpolation separately on each group. The equation for third degree Lagrange's polynomial is

$$x(t) = \sum_{i=1}^{4} \frac{\prod_{j=1,j\neq i}^{4}(t-t_j)}{\prod_{j=1,j\neq i}^{4}(t_i-t_j)} P_i \qquad 0 \leq t \leq 1 \tag{4.1}$$

This was implemented off-line on PC in C language (using floating point operations). The results were recorded for vowels. But this algorithm has a disadvantage that the implementation on a fixed point machine is difficult, as x(t) can have any value and also the value can be less than zero even when all the four control points are positive (this is unrealistic as the area cannot go to a negative value).

So an algorithm, Bezier Form [4] was considered for the purpose of interpolation. This algorithm basically finds a third degree polynomial, which passes through two of the four given points. This is illustrated in Fig. 4.4 P_1, P_2, P_3, and P_4 are the four control points. The derivative (slope) at the points P_1 and P_4, R_1 and R_4 respectively, are calculated as $R_1 = 3(x(t_2) - x(t_1))$ and $R_2 = 3(x(t_4) - x(t_3))$. The intermediate points are calculated using the relation,

$$x(t) = (1-t)^3 x(t_1) + 3t(t-1)^2 x(t_2) + 3t^2(1-t)x(t_3) + t^3 x(t_4) \qquad 0 \leq t \leq 1 \tag{4.2}$$

As can be seen from the equation the value of x(t) always lies between 0 and 4, if all the control points lie between 0 and 1. Hence this can be implemented on a fixed point machine. This was implemented off-line on a PC in C language. The results were recorded for vowels and compared with that produced by Lagranges method and were found to be more realistic even if it does not pass through all the points. Hence it was decided to use this algorithm for real-time implementation.

It has to be noted that the interpolation is computationally intensive, and hence for real time operation, it has to be done on the DSP board. Therefore the algorithm should be implementable using fixed point arithmetic.

4.6.2 Real time implementation

The length of vocal tract representation on the monitor, was decided to be 176 pixels. This length was selected based upon time required for computation and the perceivability of the resulting vocal tract shape. If the length is more, computational time will be more and if it is less, the interpretation could become difficult. The cross-sectional area of the 12 tubes, hence will have to be suitably interpolated to 176 points. As pointed out earlier, Bezier form algorithm can be implemented on a fixed-point processor, and gives satisfactory results. Hence this algorithm was chosen for interpolating the cross-sectional area of the 12 tubes, in real-time.

As stated in sec. 4.6.1, the interpolation of 12 points can be done by dividing them into groups of 4 and performing interpolation on each groups separately. The 12 points A_1 (lip end) ... A_{12} (glottis end) are divided into four groups as follows,
Group1 = A_1, A_2, A_3, A_4
Group2 = A_4, A_5, A_6, A_7
Group3 = A_7, A_8, A_9, A_{10}
Group4 = A_9, A_{10}, A_{11}, A_{12}
The division into four groups implies that the number of points in each group is 48. Hence the eq. 4.2 gets modified as,

$$x(t) = \frac{1}{48^3}[(48-t)^3x(t_1)+3t(t-48)^2x(t_2)+3t^2(48-t)x(t_3)+t^3x(t_4)] \qquad 0 \leq t \leq 48 \tag{4.3}$$

By observing equation 4.3 we find that the co-efficients of $x(t_1)$, $x(t_2)$, $x(t_3)$, and $x(t_4)$ can be computed beforehand and need not be done every time. So effective number of multiplications per point is 4 and effective number of additions per point is 3.

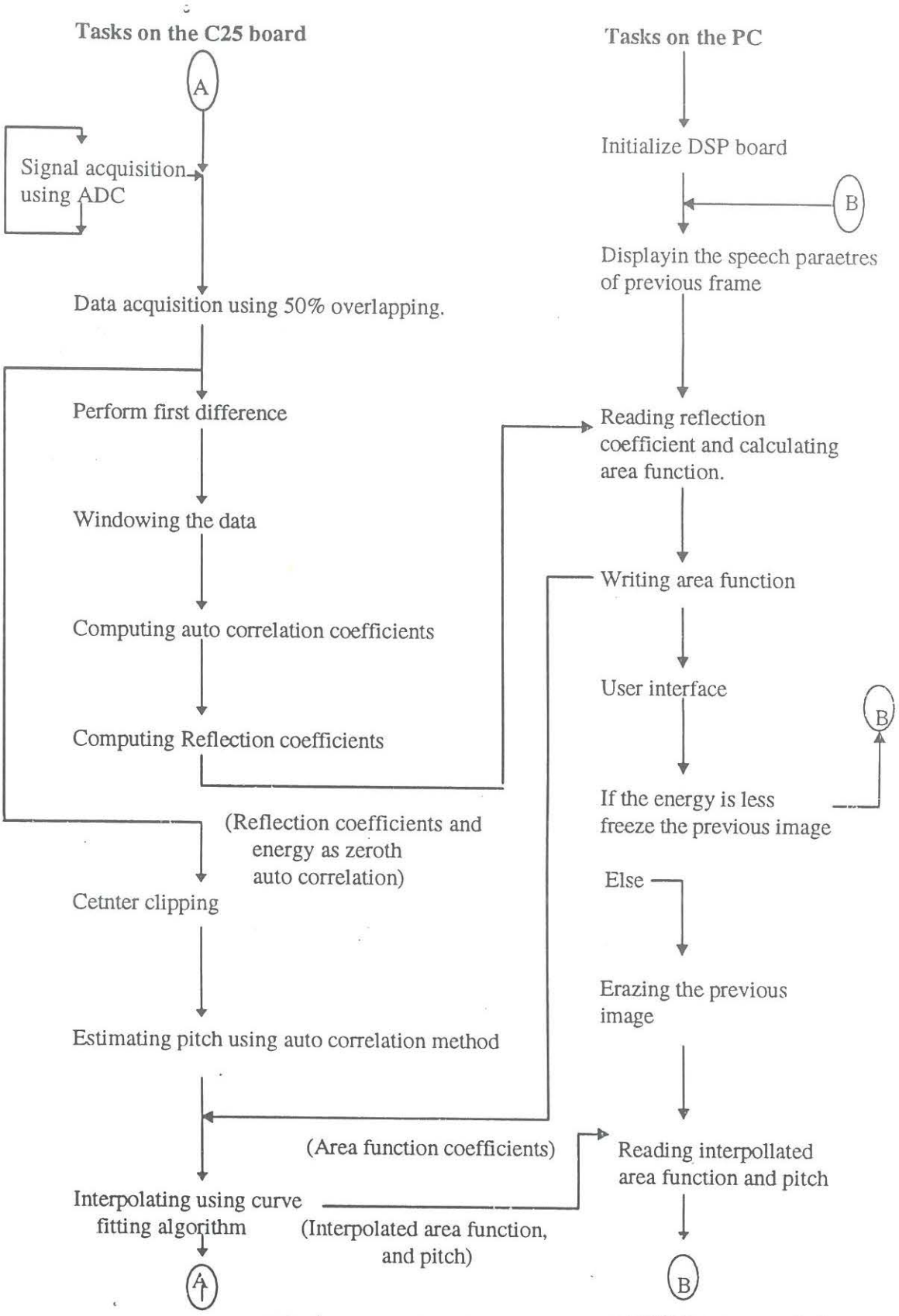


Figure 4.1: The allocation of tasks between PC and DSP board

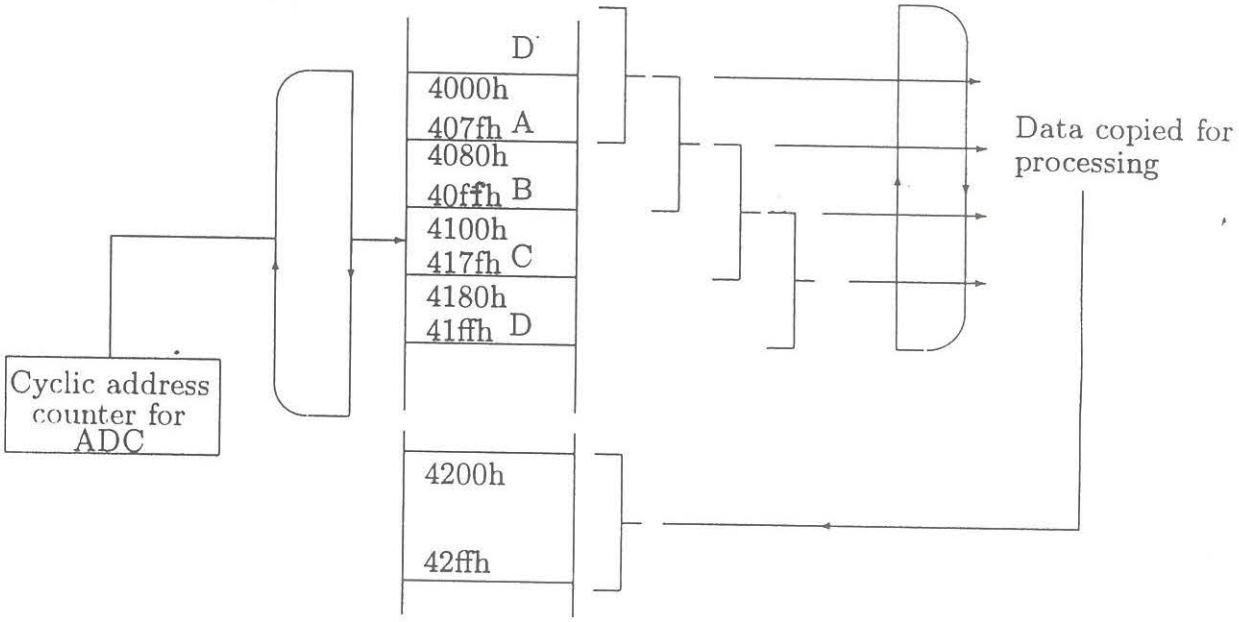


Figure 4.2: Functional diagram of window overlapping technique

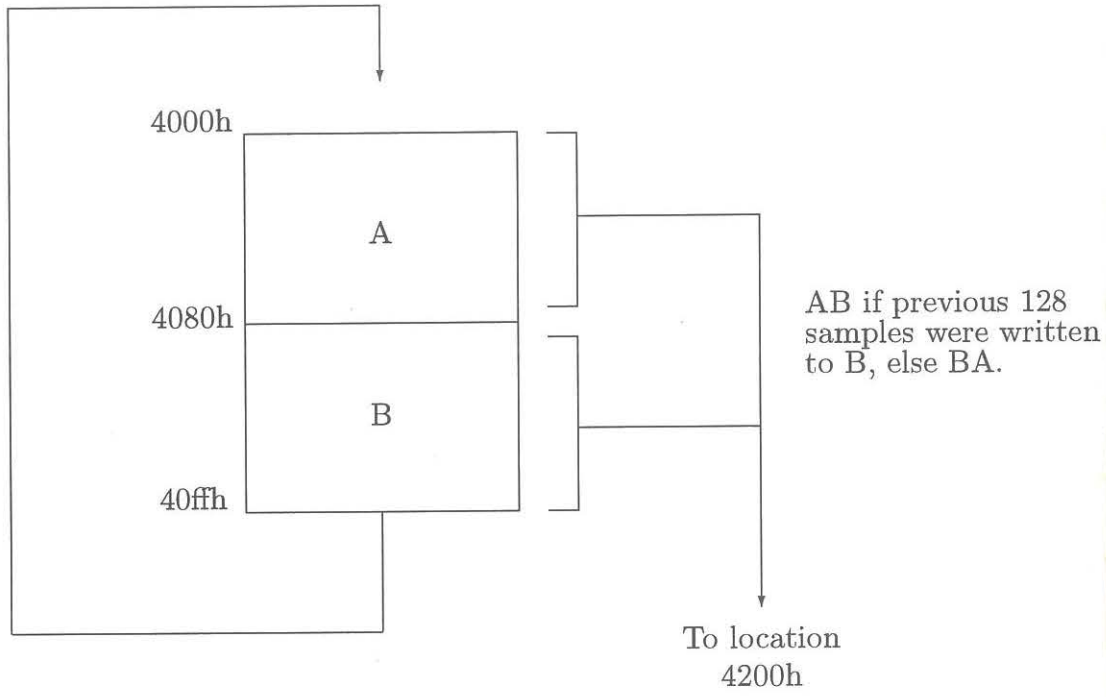


Figure 4.3: Functional diagram of modified window overlapping technique

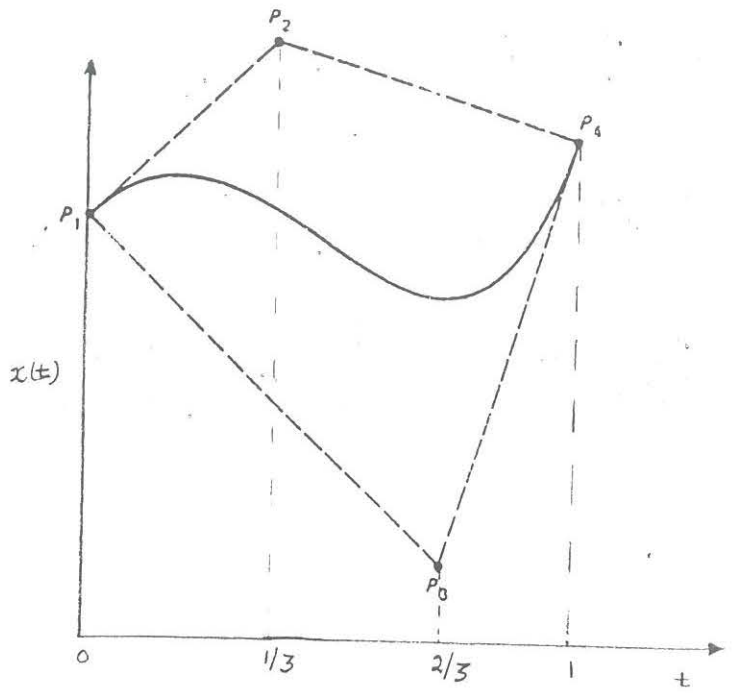


Figure 4.4: Bezier curve and its four control points

Chapter 5

Testing and Results

5.1 Introduction

The system STS-3, has the same scheme for estimation of energy, pitch, and vocal tract shape, as that of STS-2. These schemes were tested for accuracy and consistency and the results have been tabulated in sec. 3.4. So the estimation schemes of STS-3 need not be tested again. However the additional features, viz. interpolation of vocal tract area function and the freezing of the vocal tract shape during closure duration of stops, need to be tested. The curve-fitting algorithm can be tested by comparing the interpolated vocal tract area function, produced by STS-3 with that produced by a program implemented on PC (using floating point arithmetic). The sec. 5.2 describes the testing done on STS-3. The results are listed in sec. 5.3.

5.2 Testing

The Bezier form algorithm was implemented off-line, in both C and the assembly of TMS320C25. The interpolated vocal tract shape produced by the two were compared for different vowels and were found to be matching. The vocal tract shape for the vowel /a/, after interpolation by the C program, is shown in Fig. 5.1 and that by the C25 assembly is shown in Fig. 5.2. As can be seen the difference between the two is negligible. This is expected as, the algorithm (sec. 4.6.1) is suitable for fixed point implementation, and hence the error due to fixed point implementation, is negligible.

After incorporating the curve-fitting algorithm in DSP module of the system, it has to be made sure that the system still operates in real-time. The real time operation is possible, if the total time taken by the module for processing is less than 12.8 ms. The assembly program of STS-2 takes approximately 9.8 ms for the

entire process. The curve fitting takes approximately 0.4ms, so the total time taken for processing by STS-3 is 10.2 ms (the timings were measured on CRO). Hence the real time operation of the system is not affected by the addition of this feature.

The displaying of interpolated vocal tract shape requires more memory operations, as the number of points to be displayed (for vocal tract only) in STS-3, is 176 (in STS-2, only 12 points were displayed). Also displaying one pixel requires one memory read and one memory write operations [22]. The time taken by the PC module to display all the parameters, perform floating point operations and provide user-interface, should be less than 12.8 ms, for real time operation of the system. The time taken to perform all these tasks, was found to be 8 ms and hence real time operation is possible. The time required for these calculations was found by finding out the time required for 10000 such operations and then this time was divided by 10000. This was done as, the 'time' function, which is used for finding the time required for calculations, returns time in seconds.

5.3 Results

The typical display (when in review mode) of the energy, pitch, and vocal tract shape, is shown in Fig. 5.3. The darker vocal tract shape is the stored shape, for vowel /a/, spoken by a male speaker. The lighter one is produced in real time, for speech input, vowel /i/. As pointed out earlier this mode can be very helpful in training, as the teacher can store his parameters and the student (deaf person) may try to emulate him.

Fig. 5.4, shows the areagram of the speech utterance /ai/. The areagram clearly shows the changes that take place in vocal tract shape during the transition from one vowel to the other.

The Fig. 5.5, shows the areagram for /ada/, generated before the freezing of the vocal tract shape facility was incorporated and Fig. 5.6 shows the areagram for the same input but this time areagram was generated after this freezing facility was incorporated. As can be seen the freezing clearly avoids the randomness of vocal tract shape during the low energy period, and gives an idea of the vocal tract shape during the closure period of the stops.

The areagrams and spectrograms for the vowel-consonant-vowel (VCV)'s /apa/, /ata/, /aka/, /aba/, and /aga/ are shown in Fig. 5.7, 5.8, 5.9, 5.10, and 5.11, respectively. (the areagrams were generated without 'freezing' facility). As mentioned earlier the time scale of both areagram and spectrogram are made same, enabling comparison. The spectrogram shows the variation of formant frequencies just before and after the closure duration of the VCV's. Also we can observe that the vocal tract shape varies during this period and the variation is different for

different VCV's. The variation is expected as the vocal tract shape is related to formant frequencies. This information can be used to estimate the vocal tract shape during the closure period of stops. However the real time implementation of any such estimation method is difficult, as the duration of the stops varies from person to person and the energy levels during this period also varies. It was observed that the duration for which the energy levels were below the threshold was around 7 to 10 frames (corresponding to 64 ms to 89.6 ms) for voiced stops to 10 to 14 frames (corresponding to 128 ms to 179.2 ms) for unvoiced stops. This makes the prediction of vocal tract shape during closure period of stops, difficult.

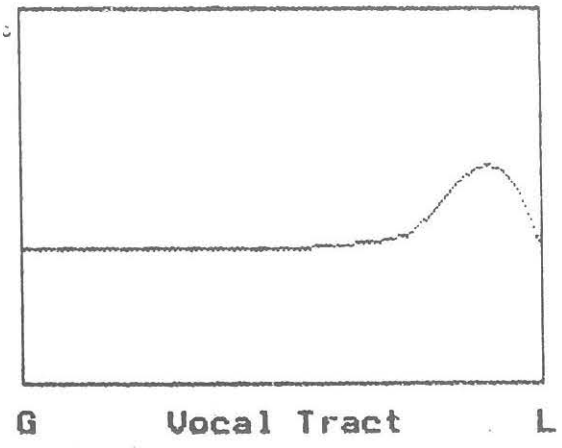


Figure 5.1: Interpolated vocal tract shape produced by C program, for natural vowel /a/

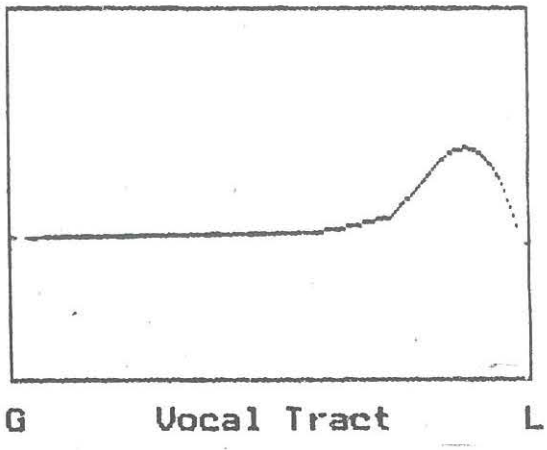


Figure 5.2: Interpolated vocal tract shape produced by DSP program, for natural vowel /a/

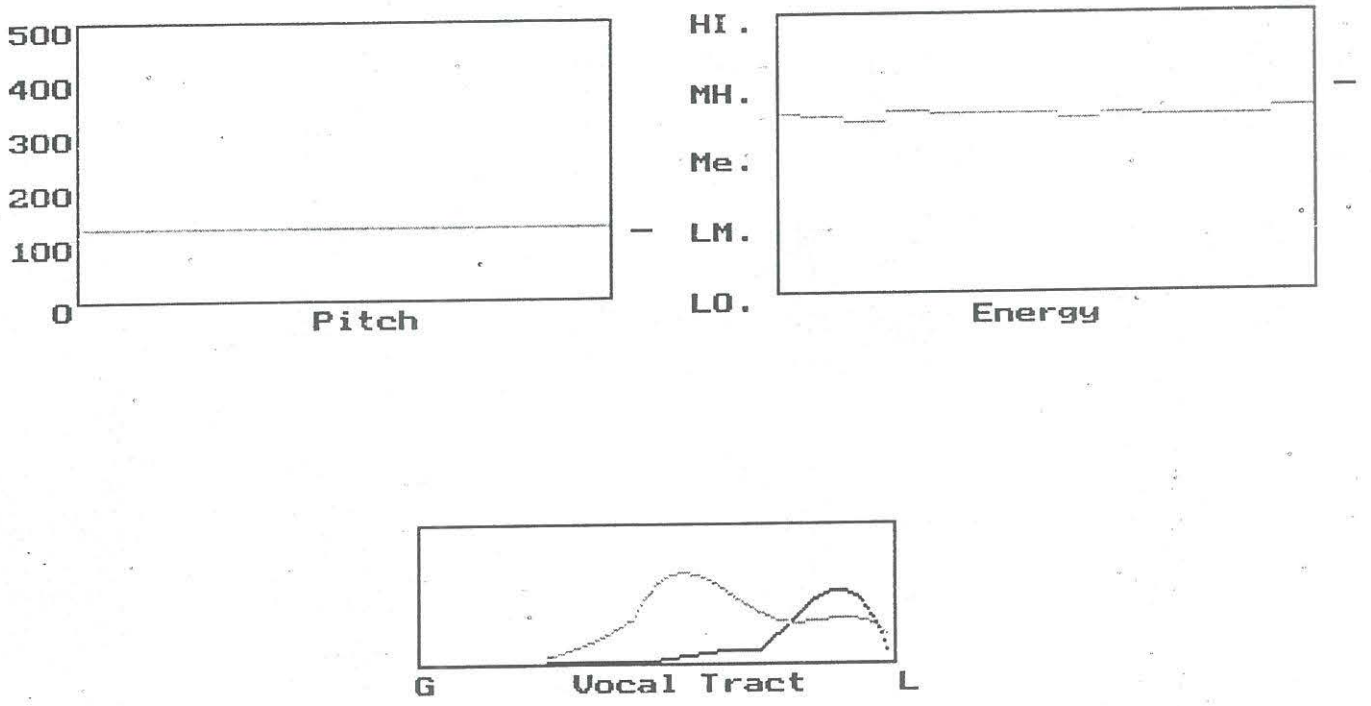


Figure 5.3: Typical display of STS-3, showing energy, pitch, and vocal tract shape

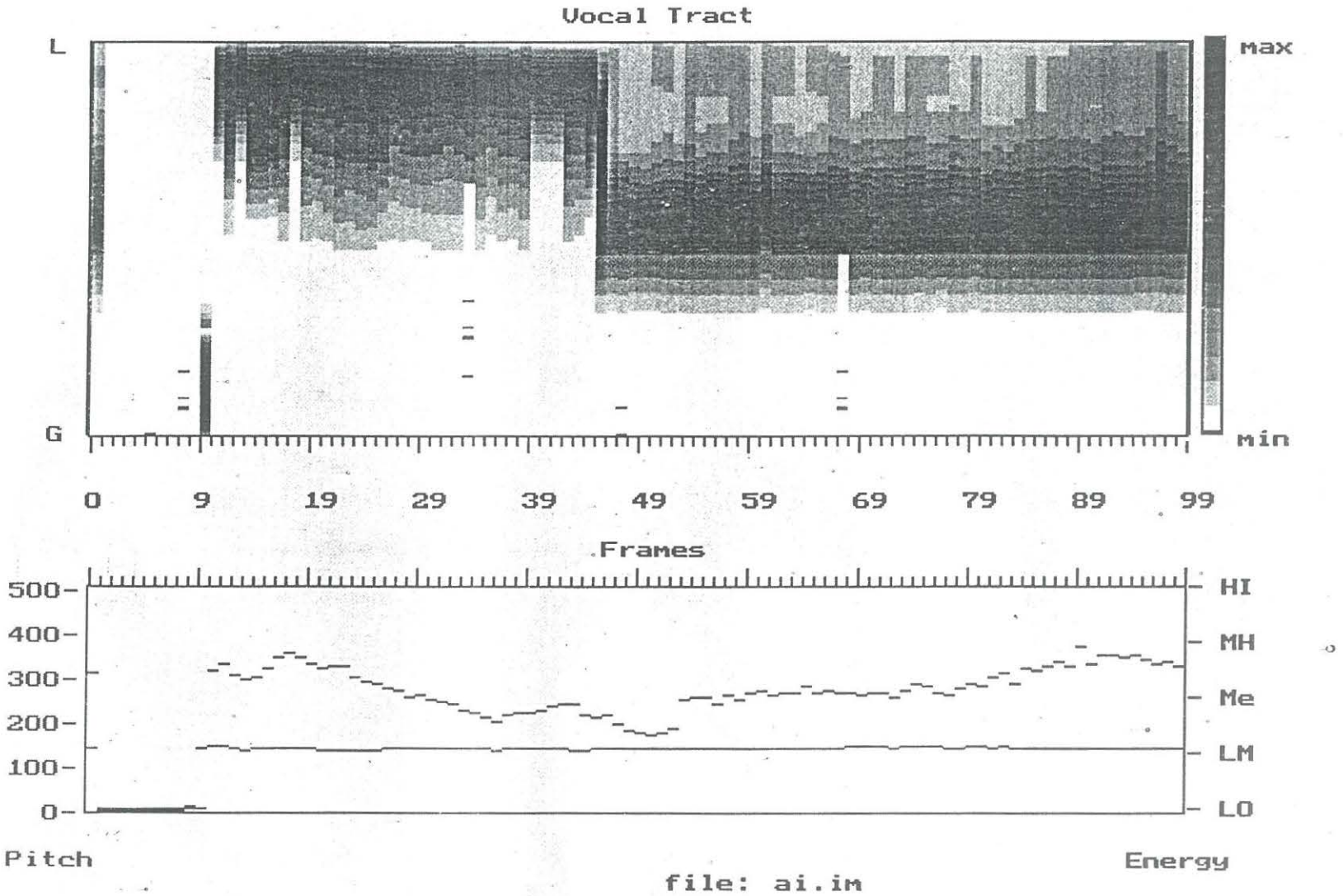


Figure 5.4: Areagram along with pitch and energy, showing the variation of vocal tract shape, pitch, and energy, for the input /ai/

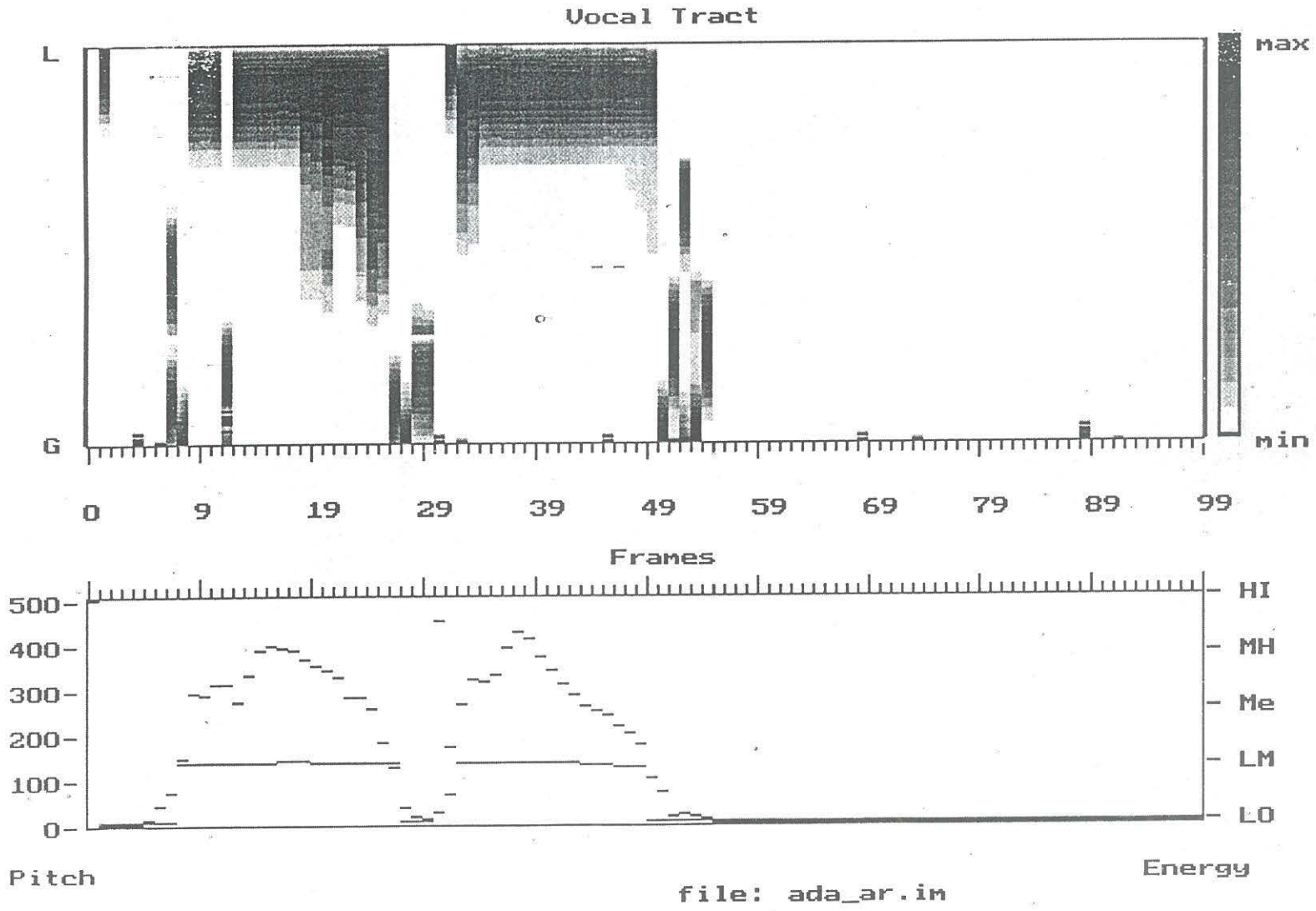


Figure 5.5: Areagram along with pitch and energy, showing the variation of vocal tract shape, pitch, and energy, for the input /ada/, without 'freezing'

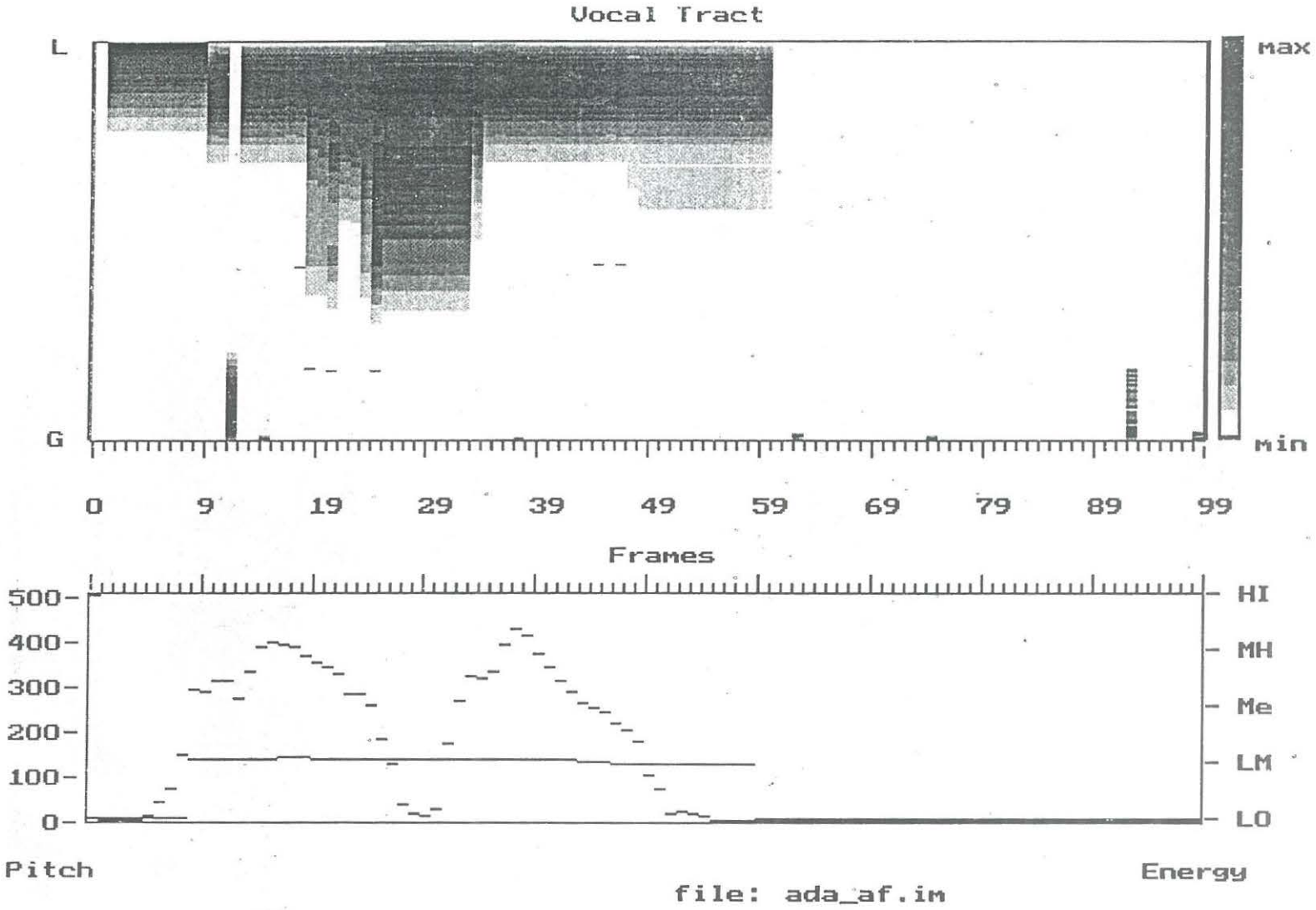


Figure 5.6: Areagram along with pitch and energy, showing the variation of vocal tract shape, pitch, and energy, for the input /ada/, after 'freezing'

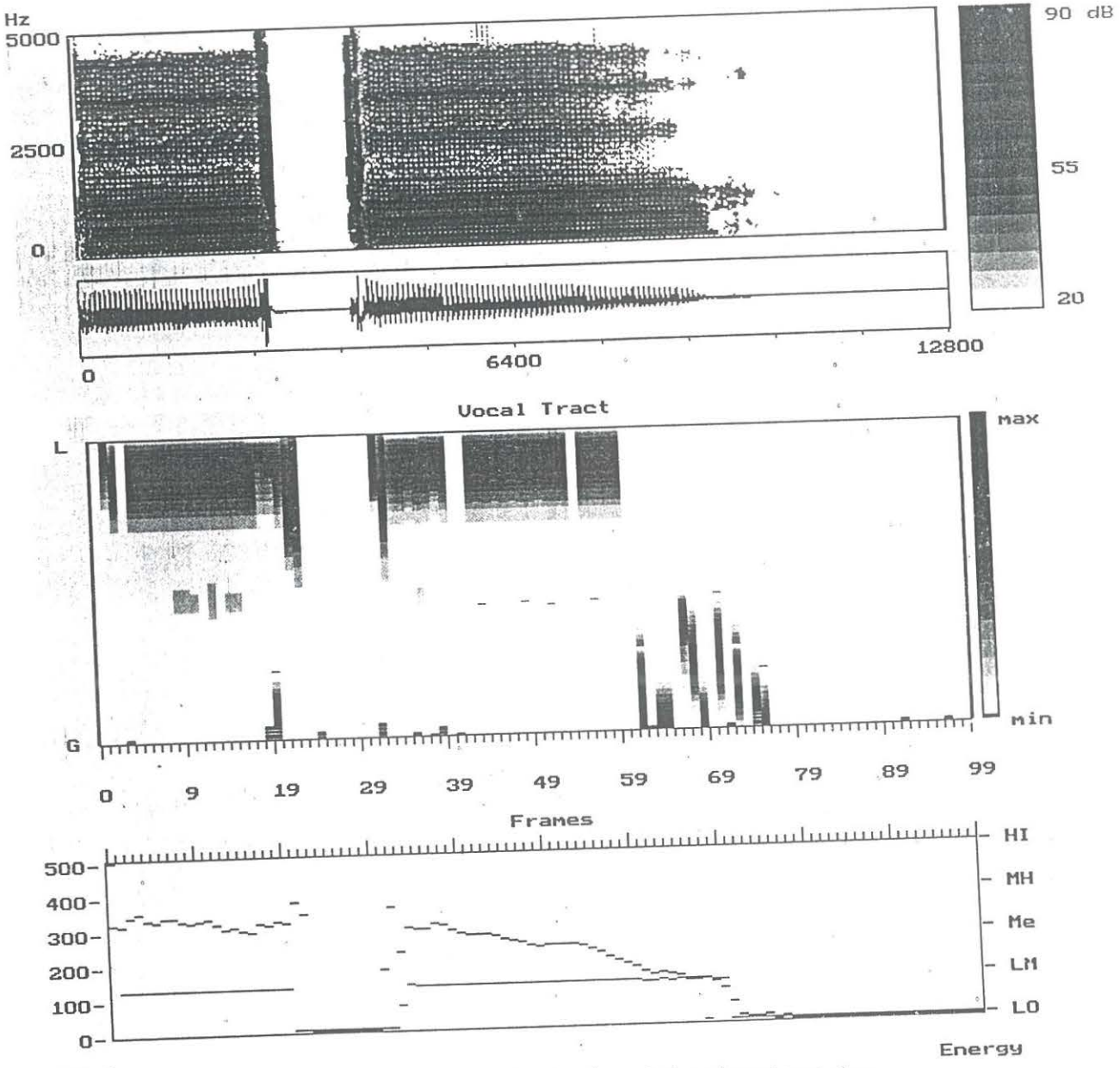


Figure 5.7: Spectrogram and Areagram, showing the variation of vocal tract shape, pitch, and energy, for the input /apa/

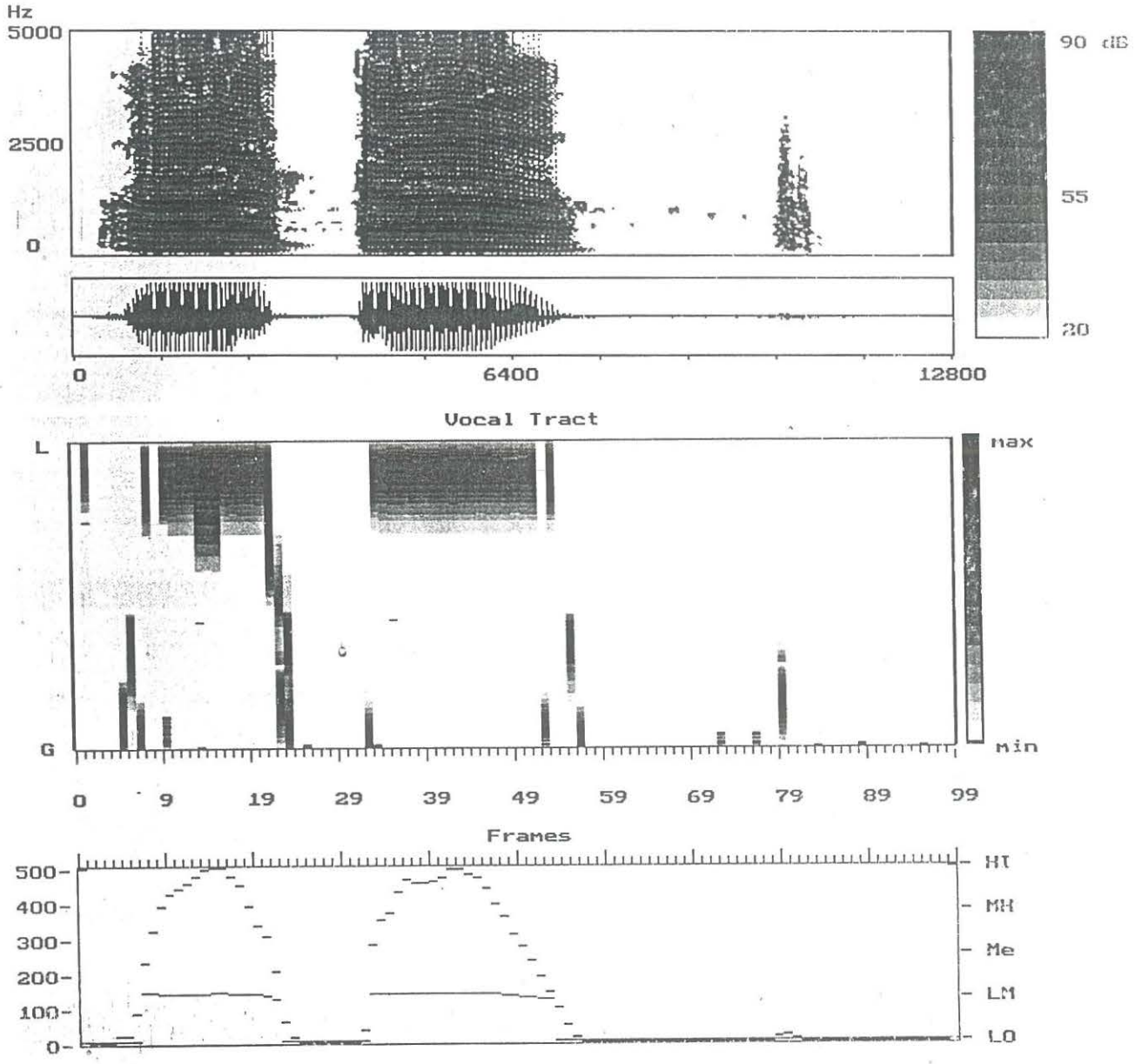


Figure 5.8: Spectrogram and Areagram, showing the variation of vocal tract shape. pitch, and energy, for the input /ata/

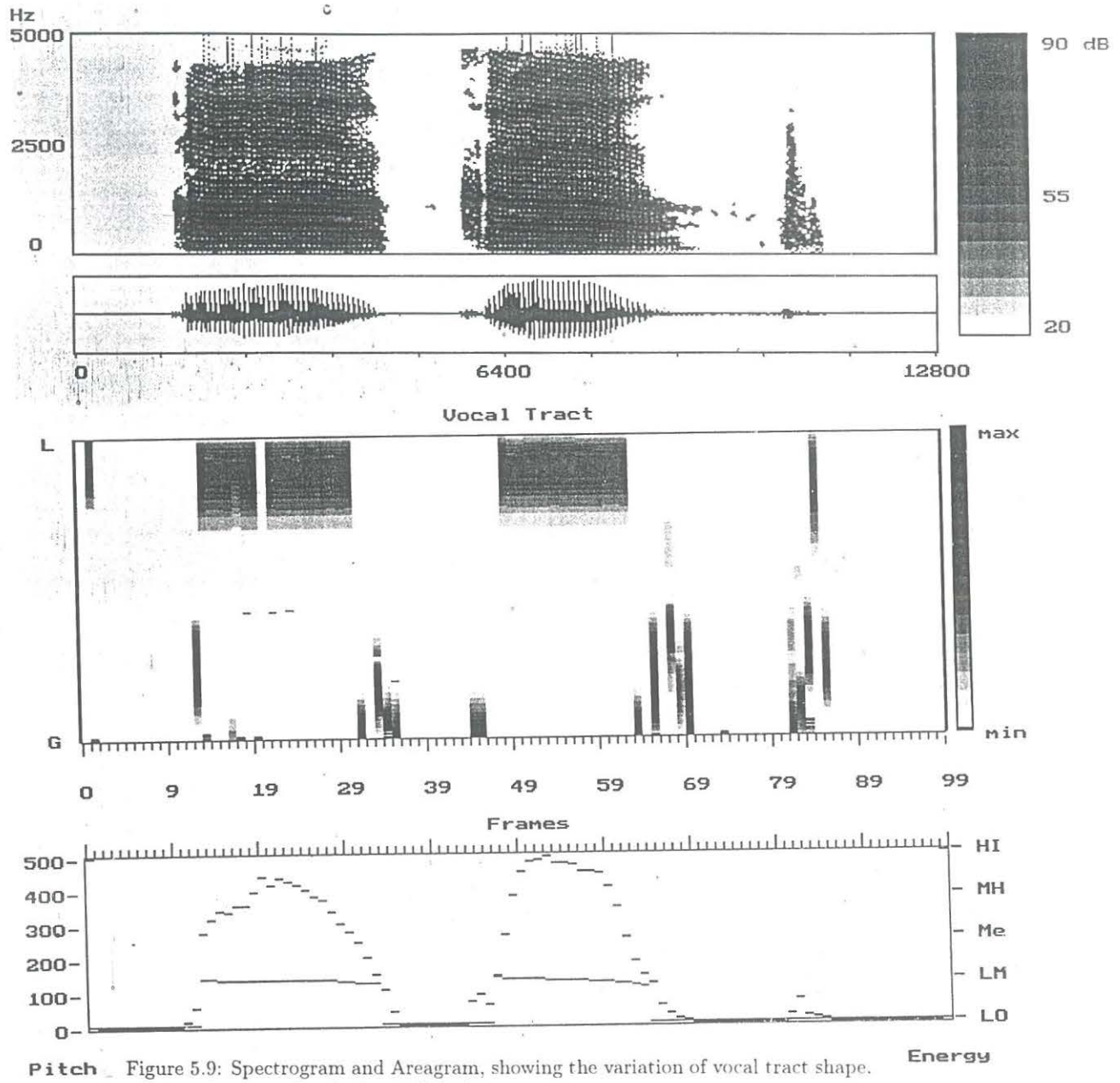


Figure 5.9: Spectrogram and Areagram, showing the variation of vocal tract shape. pitch, and energy, for the input /aka/

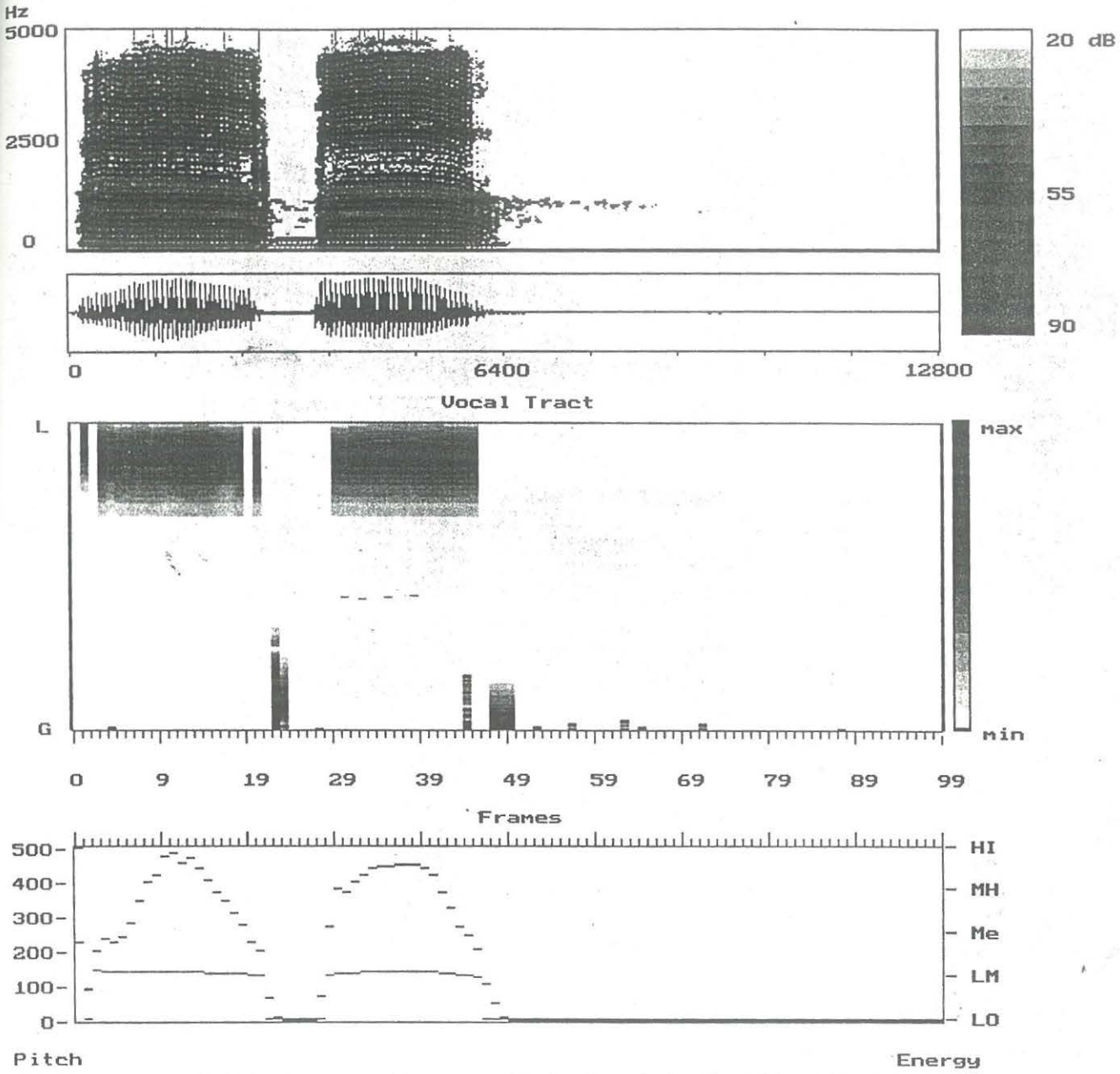


Figure 5.10: Spectrogram and Areagram, showing the variation of vocal tract shape, pitch, and energy, for the input /aba/

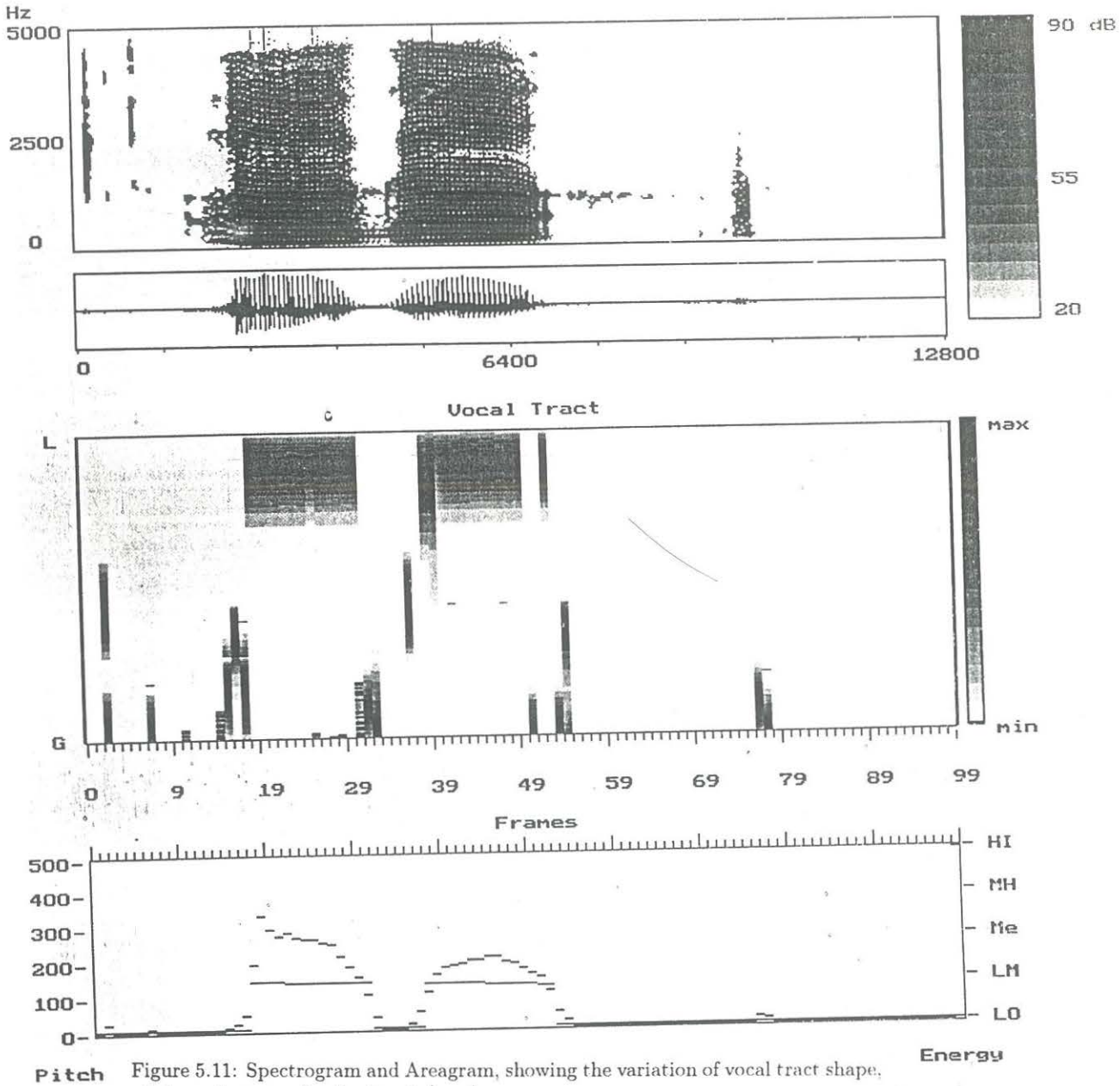


Figure 5.11: Spectrogram and Areagram, showing the variation of vocal tract shape, pitch, and energy, for the input /aga/

Chapter 6

Summary and Suggestions for Further Work

6.1 Summary

The objective of the project is to develop a speech training aid for the deaf, which may help in training the deaf children to learn to produce intelligible speech. The aid should provide visual feedback of speech parameters, like energy, pitch, and vocal tract shape, in real time, from the speech input. The aid should also provide facilities like recording these parameters and displaying the stored parameters in slow motion review mode.

An aid of this type has been developed earlier at IIT Bombay [1, 5, 8]. The aid uses Le Roux-Gueguen algorithm for vocal tract shape estimation, auto correlation method for pitch estimation and energy is estimated as zeroeth auto correlation. The system however, has to be tested for accuracy and consistency. The vocal tract shape in this system is displayed as a set of straight lines. This has to be properly interpolated, to make the shape look more natural. The estimation of these parameters during closure period of stops becomes unreliable because of the low energy. So the vocal tract shape during this period has to be decided, based upon the speech parameters just before and after the stops. For analysis purpose, suitable software has to be developed which show the variation of these parameters, with time.

As the project is continuation of ongoing efforts at IIT Bombay, the existing system, STS-1 was studied. A software, which generates areagram (a two-dimensional representation of the variation of vocal tract shape from glottis to lip, with time), for analyzing the variation of speech parameters, with time, was added to STS-1. This modified system is referred to as STS-2. The system was tested for accuracy

and consistency with both synthesized as well as natural speech (also corrupted with noise) inputs. The implementation of window overlapping technique was modified which was causing loss of some data during window overlapping.

Different algorithms were considered for interpolating the vocal tract area function and one particular algorithm, 'Bezier form' algorithm was selected for this purpose, because of its suitability for implementation on a fixed point processor. The algorithm was implemented for real time operation. To eliminate randomness of vocal tract shape during closure period of stops, the vocal tract shape was frozen to the shape just before closure period of stops. The threshold for this was decided after studying the areagrams of various speech utterances. To help the analysis, a software which generates spectrogram, using the fast arithmetic operations provided by the DSP processor.

6.2 Suggestions for Further Work

The system presently, freezes the vocal tract shape during closure duration, to the vocal tract shape just before this period. A better method would be to identify the duration for which the estimation is not reliable and interpolate the vocal tract shape during this period based on the variation of vocal tract shape just before- and/or after this duration. The areagram and spectrogram of different VCV's indicate that it might be possible to interpolate the vocal tract shape from vocal tract shape just before and after the closure period. However real-time implementation of any such interpolation scheme could be difficult, as the the duration for which, the estimation is unreliable, varies over a wide range. Also the delay involved could be as large as 150-200 ms.

The estimation of speech parameters could be done on a floating point processor. In this case the minimum energy required for reliable estimation may be much lower compared to the present value and 'freezing ' technique could give much better picture of the vocal tract shape during the closure period of the stops.

The display of the vocal tract shape could be made more realistic, by showing outline of face on the monitor, whose vocal tract shape could be made to vary according to the speech input.

Appendix A

Spectrogram

A.1 Introduction

A spectrograph is an instrument, which translates a given signal into a visual representation of its frequency components, as a function of time. Time and frequency are indicated by horizontal and vertical axes respectively, and the spectral magnitude is indicated by darkness of the pattern. These representations are known as spectrogram, and are very useful in speech analysis.

In one of the methods of generating spectrogram [9], the spectrogram of a short duration (2 s) speech utterance is recorded by an electro-mechanical instrument onto heat-sensitive teledeltos paper. In this system the speech signal repeatedly modulates the output of a variable frequency oscillator. This modulated signal is input to an analog bandpass filter. The average energy output of the bandpass filter is recorded on teledeltos paper as a spectrogram. The time and frequency resolutions depend upon the bandwidth of the filter. The spectrogram will have good time resolution and poor frequency resolution if the band pass filter has a wide band width (300 Hz). On the other hand it will have good frequency resolution and poor time resolution if the band pass filter has narrow band width (45 Hz). This process takes about 10 min and has a very small dynamic range and hence not widely used these days. Spectrograms can be digitally generated by spectral analysis of digitized waveform, either by using digital filter or by DFT analysis, and displaying the time-frequency plots on a monitor, or making hardcopy on a printer. The implementation can be done on a dedicated digital hardware or a general purpose computer [14, 13].

Short time fourier transform of the sampled waveform x(n), is given by,

$$X(n,k) = \sum_{m=0}^{N-1} w(m)\ x(n-m)e^{-j2\pi km/N} \qquad \text{(A.1)}$$

where, n represents the discrete time samples, k the discrete frequency and N the DFT size. The window $w(m)$ is L-point ($L < N$) Hamming window. The window length L plays an important role, as it is related to the frequency resolution, by the relation,

$$f_{rs} \quad = \quad 2f_s/(1.5L)$$

where, f_{rs} is the frequency resolution and f_s is the sampling frequency. The factor 1.5 is due to the Hamming window [15]. Hence spectrogram of required resolution can be generated by properly choosing L, e.g., wideband (300 Hz) spectrogram can be generated using L=43 (speech signal digitized at 10 k Sa/s) and narrowband (45 Hz) spectrogram can be generated using L=289 (signal sampled at 10 k Sa/s). The DFT size N is maintained constant in order to have same number of spectral samples for different values of L.

Attempts have been made to obtain a spectrogram-like representation, with good time and frequency resolution, simultaneously [2]. Most of the schemes which produce such representation are computationally intensive and also the resulting displays are difficult to interpret. One of the simpler methods for providing the features of both wideband and narrowband spectrogram [3], is to obtain "Combined spectrogram" X_{cb}, as the geometric mean of wideband spectrogram X_{wb} and narrowband spectrogram X_{nb}, given by,

$$| X_{cb} | \quad = \quad [| X_{wb} | . | X_{nb} |]^{1/2} \tag{A.2}$$

The valleys (lighter levels) of both the spectrograms are preserved by geometric mean operation, and hence the horizontal and vertical features will be visible in the combined spectrogram.

As mentioned earlier, spectrogram can be digitally generated on a dedicated hardware or a general purpose computer. Because of easy availability of computers, the later method has become popular. Programs have been developed, that generate spectrogram and can be run on a PC. One such program was developed at IIT Bombay [21].The program in addition to providing trade-off between time and frequency resolutions, can generate "combined spectrogram". It plots the log magnitude of short time fourier transform, instead of the magnitude itself, there by increasing the dynamic range and also making the display easily understandable. The program allows the user to vary the dynamic range. In this program, all the calculations (FFT, log magnitude) and displaying are done on the PC, making the process slow. So to increase the speed of generation of spectrogram, it was decided to use a DSP PC add-on board based on TMS320C25 for calculation of FFT and the PC for displaying the spectrograms. A program was developed along with Prasad [16], for this purpose. Here, first this program itself will be described, followed by its operation.

A.2 Spectrogram program

The program has basically two modules, one runs on the DSP board and the other runs on the PC. The PC module handles the user interface, calculation of log magnitude and display functions. The DSP module computes the FFT. Generation of spectrogram is made faster by doing these things in parallel, i.e., while the DSP program is calculating FFT of the present block, the PC displays the log magnitude of FFT of the previous block. The DSP program is written in TMS320C25 assembly language and PC program in C.

The spectrogram program "spectro.c", provides a provision for acquiring the input either from a file or from ADC, on the DSP board. The program needs a VGA card along with a 640x480 pixel resolution monochrome monitor, and the DSP25 board (from Dynalog Micro systems, Bombay based on TMS320C25 processor). The spectrogram is 500 pixel wide and 128 pixel high, and is displayed above the waveform segment being analyzed. The display also shows a strip providing the gray scale for magnitude, cursor readouts and user directives. There is a provision for displaying the spectrum of a particular window along time axis, which is selectable. The program provides the user, the option of giving or not giving pre-emphasis to the signal. Pre-emphasis emphasizes high frequency components, and is needed, in case of speech signal, as the higher frequency components, otherwise would not be visible because of limited dynamic range of magnitude scale.

As mentioned earlier the input to the program can be from a data file containing digitized signal waveform. Alternatively, the signal can be first captured (sampling rate and the number of samples specified by the user) by using the ADC of the DSP board, stored in a file, and then used for spectrographic analysis. First difference is performed if the pre-emphasis of the signal is required. The program uses 256 point DFT for spectrographic analysis. The signal is Hamming windowed with a window length L,that is selected by the user (<256) to form a data block. The block is extended to a length of 256 by padding zeros. This block of length 256 is then downloaded to the DSP board, where FFT is calculated. While FFT of this block is being calculated on the DSP board, the program running on the PC, calculates the log magnitude of FFT of the previous block. This log magnitude of FFT is mapped to 16 grey levels (highest magnitude is mapped to grey level '0' and lowest to '15'), linearly, and then displayed on the monitor. Only 128 samples are displayed as the magnitude spectrum is symmetric. After displaying, the PC program reads the FFT of the new block (the completion of FFT is indicated by setting of a flag, in the shared memory space, by the DSP program). This process is repeated 500 times (as the width of the spectrogram is 500 pixels). The overlap between successive windows depends on two factors, one is the length of the sequence for which spectrogram is to be found and the other on the length of the window.

After generating the spectrogram, the program provides facility to move the cursor to the required location on both time and frequency axes. If the movement is along the time axis, then the magnitude of that cursor location can be seen by pressing the return key. If the cursor movement is along the frequency axis, the return key need not be pressed. The reason for this is that the program stores the spectral magnitude of only one block, ending at the cursor point along the horizontal axis. As the cursor is moved horizontally, new magnitude spectra need not be computed. Therefore, in order to avoid unnecessary slowing down of cursor movement, spectrum is calculated only after return key has been pressed.

The program also offers facilities for storing the spectrogram as an image file which can be viewed and converted to required format (like fname.ps, fname.gif) using 'xv' of X-windows.

The program does not provide the facility of "combined spectrogram" [21], but it permits change of resolution by changing the Hamming window length for the time segment being analyzed.

A.3 Operation

Certain precautions have to be taken while running this program as the FFT is calculated on a fixed-point DSP processor. The program may otherwise terminate with floating point error or may give wrong results. The program can be started by typing 'spectro'. The program checks for the presence of the properly set DSP board and will inform the user of any errors encountered. It requires the assembled DSP program file 'FFT256.mpo' in the same directory, and automatically loads it onto the DSP board. In this section, the information, regarding, the acquisition of input signal, the input data file, the frequency resolution, the magnitude dynamic range required, scaling factor and the pre-emphasis that are asked by the program are explained. The order in which the program asks these inputs is maintained to help the operation.

- Acquire data for spectrogram (y/n) : Typing 'Y' will make the program to capture the signal from ADC of the DSP board. The captured data will be stored in a file before spectrogram is generated. The program requires following inforamtions for capturing and storing,

    1. Sampling frequency : It can not be less than 77 Sa/s because of hardware configuration of on board timer of the DSP board.
    2. File type: can be binary or text
    3. Data files : Data file can have any name. The file will be created in the present directory (overwritten if existing). It is better to use extensions

.txt and .bin for text and binary files even though it is not essential, as it helps in identifying the file type.

4. Number of samples : It has to be a multiple of 128, otherwise the nearest (lower) integer multiple of 128 samples will be stored. The program starts capturing after getting this inputs.

Typing "n" will make the program to get the data from an input file. The data file should be in the following format. The first item should be number of samples in the file. The samples should be stored from second item onwards .The file can be either binary or text format. Both the samples and the number of samples should be stored as integers. The program require the following information for getting the data from file correctly.

1. Sampling frequency : This information is used for frequency scale calibration and hence does not affect the generation of spectrogram, however it is better to give the frequency at which the data was sampled.
2. Data file : The name of the data file, with the extension should be given . Program terminates if the file does not exist or it is unable to open the file.
3. File type : It can be either "bin" or "txt".

- Zero intensity level and maximum intensity level : The purpose of this is to give variable magnitude resolution. The spectral magnitude level between these two limits are linearly mapped to 16 gray levels. The spectral magnitude outside this range are made equal to the limits. If nothing is known about the signal, one can start off with zero level of 20 dB and maximum level of 90 dB and change them as per the requirements later. The program then plots the signal waveform of the entire file. Any portion (entire file also) of this can be selected for generating the spectrogram. Waveform of this selected segment is again plotted over the entire range (i.e., zooming in!). The selection can be started by moving the cursor to the required position and hitting the return key. Then the cursor can be moved to the end position and the return key hit to mark the end position. For slow movements arrow keys can be used and for faster movements 4,6 keys can be used.

- Window length: This is the length of the Hamming window and the frequency/time resolution depend on this input. It can have any value that is less than 256. Typically one can select 43 samples (corresponding to 300 Hz bandwidth with 10 k Sa/s) for wide band and 159 samples (for 84 Hz) for narrow band . Selecting higher window length, even though gives a better

frequency resolution, may lead to overflow and selecting lower window length, even though gives a better time resolution, may result in under flow.

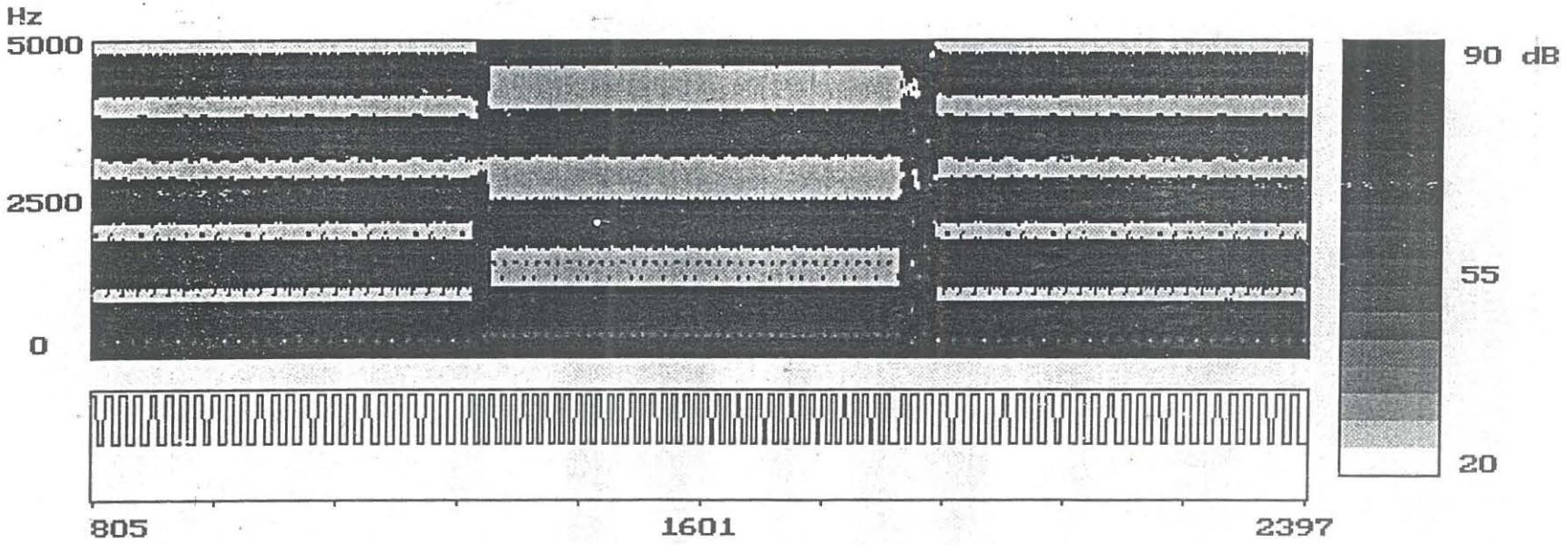
- First difference: Typing 'y' will pre-emphasize the input signal. Typing 'n' will not modify the input signal. For speech signals, it is better to give 'y' as this gives a boost to the higher frequency signals.

- Scale factor: Only 16 bits of the square magnitude of FFT can be stored, as the memory is length 16 bits. The magnitude of FFT can be 30 bit long (worst case). So certain bits (LSB's) have to be truncated before storing. 'Scale factor' is the number of LSB's that are truncated before storing. So this factor depends upon the signal level. If nothing is known about the signal level, one can start off with scale factor equal to 15 and change according to the requirements later. If this value is less, then the program may terminate with 'log10' error. It has to be noted that, choosing a proper scaling factor, does NOT ensure overflow-less operation, as the scaling is done at the time of storing. So in case of overflow (which sometimes can be detected by white spots on black strips of the pattern) the signal level has to be decreased. This can be done with the help of the program 'scale' [16].The maximum level for overflow-less operation when the input is a 1 k Hz sinusoidal wave (sampled at a rate 10 k Sa/s) is 0.44 V rms. This is equivalent to 2000 after digitizing the signal on a 16 bit ADC with $\pm$ 10 V input.

The program, after generating spectrogram, gives option of moving the cursor to required position for readouts or plotting spectrum. The movement can be done using arrow keys(for slow movements) or 2,4,6,8 keys(for fast movements). The function key F3 can be used for storing the entire screen in a file and F4 for storing only the spectrogram. The file is stored in text form and hence occupies a large space.

A.4 Results

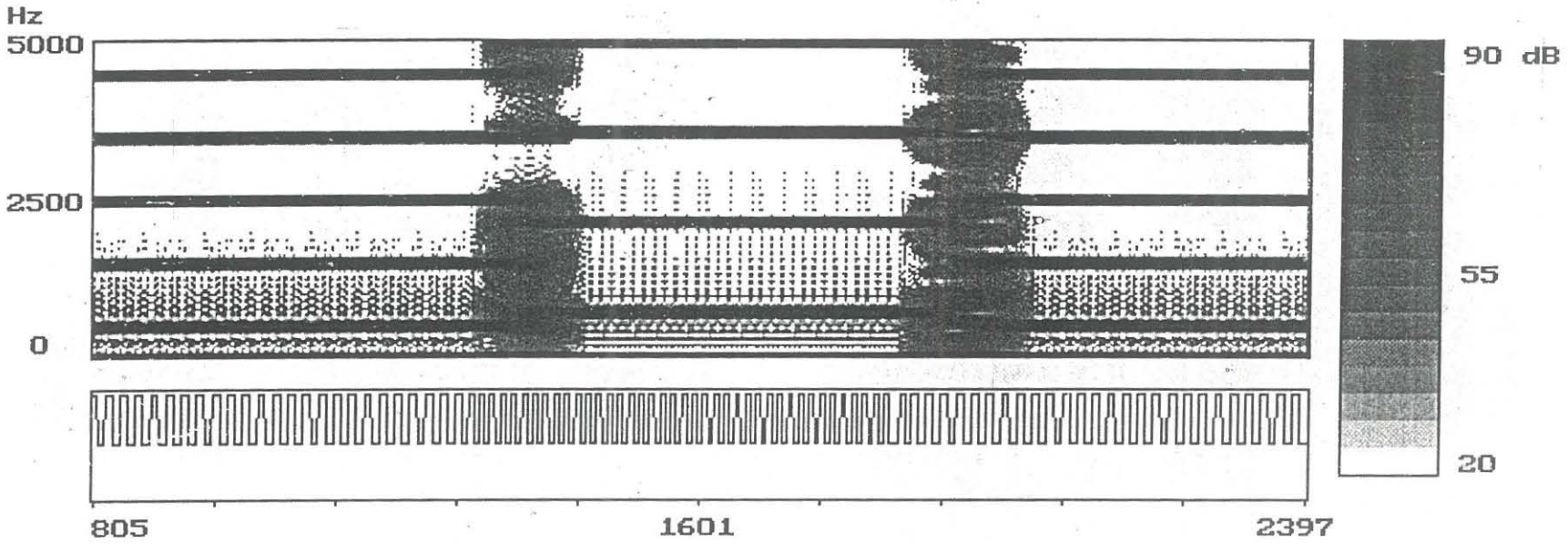
Fig. A.1 shows the wideband spectrogram of a square wave, whose frequency has a step variation from 500 Hz to 750 Hz and back to 500 Hz. The spectrogram clearly shows the abrupt variation of frequency, as the time resolution is good. The Fig. A.2 shows the narrowband spectrogram of the same squarewave. In this case the funadamental frequency and its harmonics can be clearly seen, as the frequency resolution is good. However, the time resolution has suffered as is seen from the smears at the points of frequency changes.



file:sqrw.im

```
fine movement    : arrow keys
coarse movement  : 2,4,6,8 (arrows with numlock)
measurements     : Enter
cursors off/on   : F1
band analysis    : F2
quit             : Esc
```

Figure A.1: Wideband spectrogram for a square wave input, with step change in frequeny



file:sqrn.im

```
fine movement    : arrow keys
coarse movement  : 2,4,6,8 (arrows with numlock)
measurements     : Enter
cursors off/on   : F1
band analysis    : F2
quit             : Esc
```

Figure A.2: Narrowband spectrogram for a square wave input, with step change in frequeny

References

[1] Y. Bhagwat, "Real-time vocal tract shape and pitch estimator", M.Tech. Dissertation, Dept. Elec. Engg., I.I.T., Bombay, 1993.

[2] L. Cohen, "Time-frequency distributions - a review," *Proc. IEEE*, Vol. 77, pp 941-981, 1989.

[3] S. Cheung and J.S. Lim, "Combined multiresolution (wideband/narrowband) spectrogram," *IEEE Trans. Sig. Proc.*, Vol. 40, pp 975-977, 1992.

[4] J.D. Foley and A. VanDam, *Fundamentals of interactive computer graphics*, pp. 514-536, Addison-wesley, New york, USA, 1983.

[5] S. Gavankar, "A speech training aid for the deaf", M.Tech. Dissertation, Dept. Elec. Engg. I.I.T., Bombay, 1995

[6] S. Gracias, "A speech training aid for the hearing impaired", B.Tech. Project report, Dept. Elec. Engg., I.I.T., Bombay, 1991.

[7] M. Gupte, "A speech processor and display for the speech training of the hearing impaired", M.Tech. Dissertation, Dept. Elec. Engg., I.I.T., Bombay, 1990.

[8] N. Khambete, "A speech training aid for the deaf", M.Tech. Dissertation, School Biomed. Engg. I.I.T., Bombay, 1992.

[9] R. Koenig, H.K. Dunn, and L.Y. Lacey, "The sound spectrograph", *J. Acoust. Soc. Amer.*, Vol. 18, pp 19-49, 1946.

[10] E.V. Krishnamurthy, S.K. Sen, *Numerical algorithms*, East-West Press, New Delhi, India, 1986.

[11] D. Kulkarni, "Development of a cascade pole-zero synthesizer", M.Tech Dissertation, Dept. Elec. Engg., I.I.T. Bombay., 1992

[12] J.D. Markel and A.H. Gray, "On auto correlation equations as applied to speech analysis", *IEEE Trans. Audio & Electro-acoustics*, Vol.21, pp. 69-79., April 1973.

[13] L.R. Morris, "A PC-based digital speech spectrograph," *IEEE Micro*, Vol. 8, pp 68-85, 1988.

[14] A.V. Oppenheim, "Speech spectrograms using the fast Fourier transform," *IEEE Spectrum,*. Vol. 7, pp 57-62, 1970.

[15] P.C. Pandey, "Speech processing for cochlear prosthesis", Ph.D. thesis, Dep. Elec. Engg., University of Toronto, 1987.

[16] V.V.S.R. Prasad, "Speech processing for single channel sensory aid", M.Tech Dissertation, Dept. Elec. Engg., I.I.T., Bombay, Jan 1996.

[17] PCL-DSP25, *User's manual*, Bombay : Dynalog Microsystems.

[18] L. Rabiner and R. Schafer, *Digital signal processing for speech signal*, Englewood Cliffs, New Jersey : Prentice hall, 1978.

[19] L. Roux and C. Gueguen, "A fixed point computation of PARCOR coefficients", *IEEE Trans. Acoust. Speech & Signal Process.*, Vol.25, pp. 257-259., 1977.

[20] K. Taklikar, "A speech training aid for the deaf", M.Tech. Dissertation, Dept. Elec. Engg. I.I.T., Bombay, 1990.

[21] T.G. Thomas, P.C. Pandey, and S.D. Agashe, "A PC-based spectrograph for speech and biomedical signals," in *Proc. Intl. Conf. Recent Advances in Biomedical Engineering*, (Hydrabad), Jan 1994, pp. 6-8.

[22] G. Sutty, S. Blair, *Programmer's guide to the EGA/VGA*, New Delhi : BPB, India, 1990.

[23] H. Wakita, "Direct estimation of vocal tract shape by inverse filtering of acoustic speech waveforms", *IEEE Trans. Audio & Electro-acoustics*, Vol.21, pp. 417-427., Oct 1973.