# A TEXT TO SPEECH CONVERTER USING CASCADED POLE ZERO SYNTHESIZER

Dissertation

by

Nitin N. Raut
(92307014)

Guide : Dr. P. C. Pandey
Co-guide : Prof. S. D. Agashe

Submitted in partial fulfillment
of the requirements
for the degree of

Master of Technology



Department of Electrical Engineering
Indian Institute of Technology, Bombay

January 1994

# Dissertation Approval Sheet

Dissertation by Mr. **Nitin N. Raut** (Roll No. 92307014) titled "A Text to Speech Converter using Cascaded Pole Zero Synthesizer" is approved for the degree of Master of Technology in Electrical Engineering.

| | | |
|---|---|---|
| Guide | Dr. P. C. Pandey | PCPandey |
| Co-Guide | Prof. S. D. Agashe | S. D. Agashe |
| Internal Examiner | Dr. P.G. Poonacha | 13/2/94 |
| External Examiner | R. S. Sardesai | RSSardesai |
| Chairman | Prof. P.K. Pattanayak | PK Pattanayak |

19940213

## Abstract

Nitin N. Raut : **A text to speech converter using cascaded pole-zero synthesizer**, M. Tech. dissertation, Department of Electrical Engineering, I.I.T. Bombay, January 1994.

---

The use of a complex formant synthesizer such as cascaded pole zero synthesizer would be greatly simplified if one can specify the control parameters in some simpler form such as text. A text to speech converter would appropriately map these text symbols into the parameter tracks required for the synthesizer.

The Text To Speech converter described in here accepts text in Devnagri script using AKSHAR, a bilingual text editor. The synthesis is done by principle of " synthesis- by-rule". The basic phonemes in Marathi are used as references to generate initial and final targets and are generated by extensive hand tailoring. The rules for algorithmic coding are derived from observations during the generation of the phonemes.

# Acknowledgement

I wish to express my gratitude to my guides Dr. P. C. Pandey and Prof. S. D. Agashe for all the guidance and co-operation extended to me time and again in the hour of need. I am very much thankful to 'Effi' (Ravi Krishna), my wing mate, who helped me with C language programming. I also wish to thank Mr. T. G. Thomas and my colleague Yogesh Bhagwat, with whom I had many a rewarding discussions. I thank all people in Standards Lab for helping me out many times.

Date: $24^{th}$ January, 1994

Place: I.I.T. Bombay

(Nitin N. Raut)

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

A flexible speech synthesizer which can produce speech output if its various input parameters are controlled, finds applications in numerous areas such as those of speech sciences, psychoacoustics, linguistics, and also in testing and calibrating the various sensory aids such as hearing aids, visual aids and cochlear prosthesis ([1], [2]). The input parameters which are available for the control are determined largely by the model which is selected for synthesizing the speech and, depending on model selected the control parameters may or may not be directly related to the parameters of the speech signal. In most of the cases the parameters controlling the synthesizer are numerous and it becomes cumbersome and difficult to specify the parameters for long utterances like a sentence. In such a cases a text-to-speech converter (TTS) can be useful. The TTS would take the utterance in a simpler form such as text and provide a mapping to the parameters of the synthesizer.

## 1.1 Objectives

A formant synthesizer, cascaded pole-zero synthesizer, synthesizes the speech by simulating the spectral shaping characteristic of the vocal tract. This simulation is done by modeling the vocal tract as a filter being excited by glottal excitation. The parameters of this synthesizer are easy to understand and are directly related to the parameters of the speech signal. Our goal is to make a text-to-speech converter using this formant synthesizer. This converter would accept the input in the terms of the typed text, and would result in considerable simplicity in the synthesis of long sentence utterances. The flexibilty of the synthesizer can still be used by using different phoneme library.

For computerized implementation, the rules of pronunciations should be fairly rigid and the 'exceptions' (i.e. possibilities of different pronunciations for similar combination of the alphabets) dependent on context or otherwise should be minimum. In most of the Indian languages, the scripts can be considered as phonemic transcription, consequently, conversion to the phonemic form requires only a simple algorithm. Secondly the pronunciation should not disallow any combinations or clusters of basic phones. This would ensure

minimum difficulty in changing over to any other language, script may remain same or with minor modifications that too may be changed.

For our purpose we select Devnagri script for the input text and Marathi language for the synthesis and development of text-to-speech algorithm. We plan to synthesize all the phonemes in Marathi, and having done this, develop an algorithm to synthesize long sentence utterances.

The TTS works by converting the input text, keyed in using a software editor AKSHAR, to a phonemic transcription of the file. This phonemic transcription will then be converted to parameters for cascaded pole-zero synthesizer using synthesis-by-rule approach. The reference phoneme file for this synthesizer will be compiled by using the existing tools for generating the parameter tracks. These control parameter files are then passed to the cascaded pole-zero synthesizer, which then generates data for digital-to-analog converter. This stream of data is then presented using the presentation program.

## 1.2 Outline

The report is organised in two main parts, first three chapters are devoted to discussion of the algorithms, synthesizers and the options available for the final converter. These chapters also describe, in brief, the use of the analysis and synthesis tools. The second part, chapter 4 onwards, is dedicated to description of the work done. Chapter 4 reports on first hand observations during the synthesis of the phonemes with the cascaded pole-zero synthesizer. Chapter 5 deals with the development of the text-to-speech converter and issues relevant to formant synthesis and synthesis by rule as well as rules for the algorithm developed. The last chapter provides a summary of work done and some suggestions for further work.

# Chapter 2

# Overview of the synthesizers

The production of synthetic speech is a process of producing an acoustic signal by controlling the parameters of a selected model for human speech production mechanism. Various models exist and have been used more or less successfully. One of the earliest electrical synthesizers was reported by Dudley in 1939 ([3]), using principle of separation for excitation source and vocal tract. We shall in following sections review a few of more recent and popular models which use digital computers for speech synthesis and are fairly flexible.

## 2.1 Classification of speech synthesizers

The electrical speech synthesizers may be classified in two broad categories viz. articulatory and terminal analog. The articulatory synthesis models attempt to duplicate the geometric distribution of parameters of the human speech production apparatus itself. These kind of models however require fairly large number of parameters and pose a large computational load to the computer. The synthesizers are very flexible ([4]), however the synthesis parameters are not very well understood. The terminal analog synthesizers usually use temporal or spectral properties of the speech and actually model the end results, i.e. duplicate the terminal (input/output) properties of transmission from glottis through the vocal tract. The general structure of such a synthesizer shown in Fig. 2.1 essentially consists of source of excitation, followed by a vocal tract filter and a radiation load presented by the lips. The synthetic speech is produced by varying the parameters of the synthesizer with time.

A general formant synthesizer is depicted in Fig. 2.2 . A formant synthesizer consists of an excitation source exciting a cascade of resonators. These resonators simulate the spectral shaping of the vocal tract. Generally the various parameters of the resonators are under user control so that any kind of spectral shaping may be simulated. The load presented at lips is simulated in this case by the spectral function of lip radiation . The following sections describe two types of formant synthesizers. Here two formant

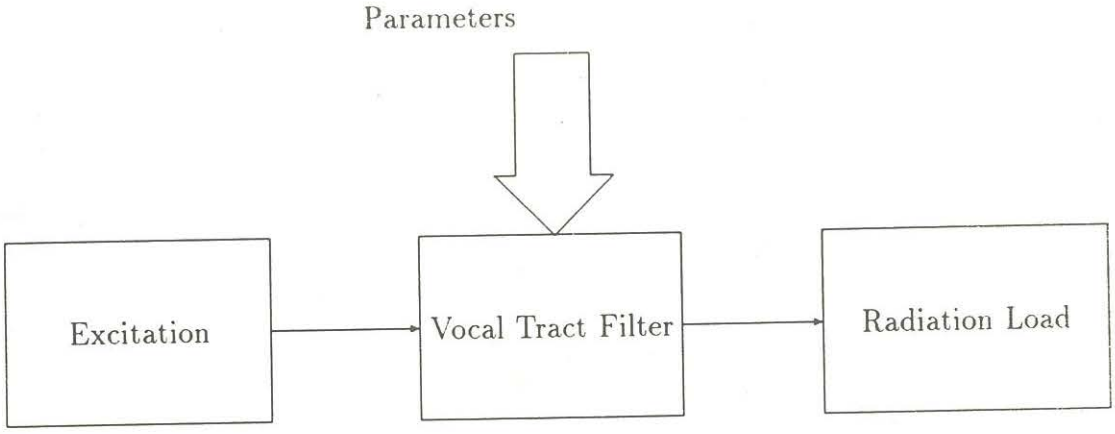



Figure 2.1: General model of terminal analog synthesizer

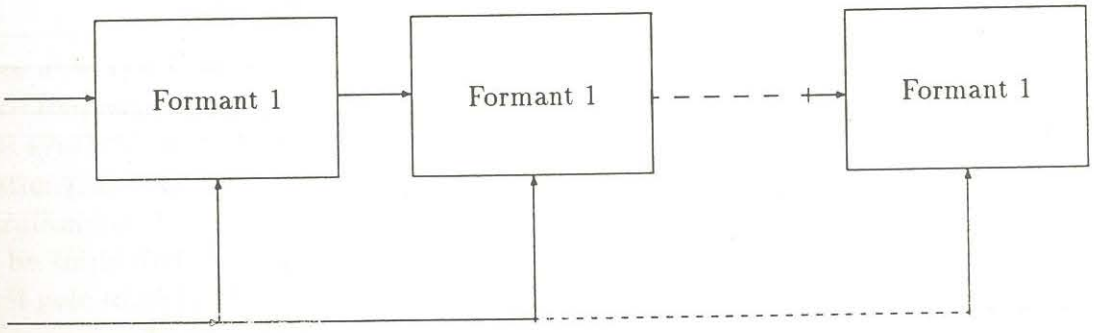



Figure 2.2: Formant Synthesizer Source: [5]

synthesizers will be briefly discussed. This will be followed by discussion of synthhesis strategies.

## 2.2 Cascade/Parallel synthesizer

A cascade/parallel synthesizer developed by [6], [16] is depicted in Fig. 2.3 (A). It contains 39 control parameters that go into determining the output and as many as 20 of these can be varied as a function of time. Additionally, it can be operated in two ways, either all parallel mode or cascade/parallel mode. The synthesizer is based on a digital resonator whose output $y(nT)$ in response to input $x(nT)$ is given as

$$y(nT) = a.x(nT) + b.y(\overline{n-1}T) + c \cdot y(\overline{n-2}T) \qquad 2.1$$

where,

$$\begin{aligned} a &= -exp(-2 \cdot c \cdot B_w \cdot T) \\ b &= 2 \cdot e^{(-c \cdot B_w \cdot T)} \cos(2 \cdot c \cdot F \cdot T) \\ c &= 1 - a - b \end{aligned} \qquad 2.2$$

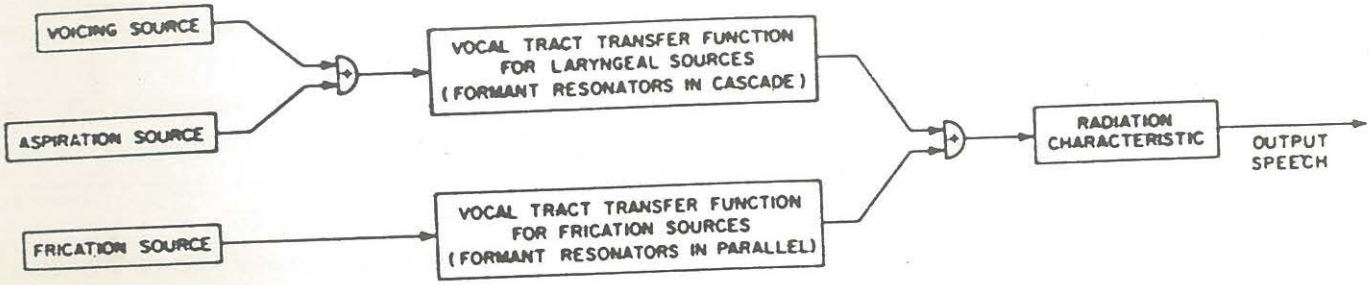
$T$ being the sampling rate, $F$ the centre frequency, and $BW$ the bandwidth. Similarly, the antiresonance is simulated by using

$$y(nT) = a' \cdot x(nT) + b' \cdot y(\overline{n-1}T) + c' \cdot y(\overline{n-2}T) \qquad 2.3$$

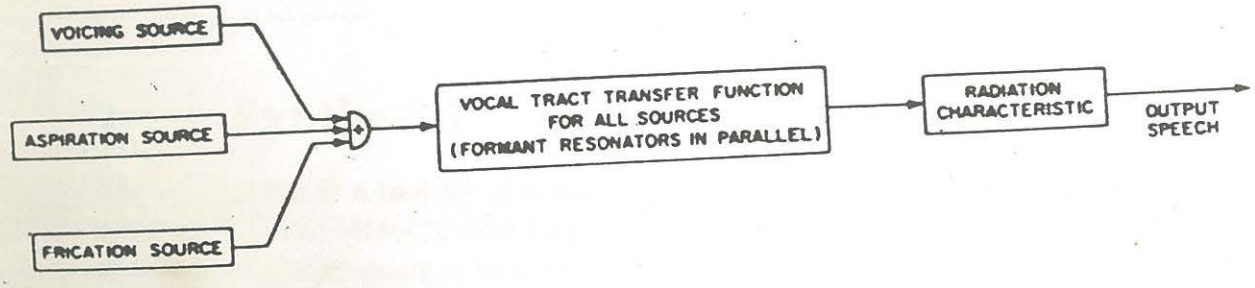
where $a' = 1/a; b' = -b/a; c' = -c/a$ and $a, b$,& $c$ are given by Eqn. (2.2) with antiresonance frequencies and bandwidths. The sources of excitations that can be simulated by Klatt synthesizer are glottal voicing, quasisinusoidal voicing, random noise (aspiration and frication), and mixed mode (voiced frication, etc. ). The amplitudes of voicing, frication, aspiration can be controlled using their respective parameters. Also the plosive bursts can be simulated. The representation of the vocal tract transfer function is achieved by an all pole model

$$T(n) = \prod_{n=1}^{6} \frac{a_n}{(1 - b_n z^{-1} - c_n z^{-2})} \qquad 2.4$$

where $a_n, b_n$, and $c_n$, are determined from values of $n^{th}$ formant frequency and bandwidth $B_W$ given by equation (2.2). Formant frequencies reflect the detailed shape of the vocal tract and first three formant frequencies vary substantially with changes in articulation, whereas the formant bandwidth is a function of energy losses, viscosity, cavity wall motions, radiation of sound from lips, and glottal impedance. For nasalization, nasal antiresonance is activated. The parallel set of digital resonators with amplitude controls and simulated frications are available and they control the amplitude of the formants. Zeroes are also accounted for by amplitude control of the formants. A sixth formant




(A) Cascade/Parallel formant configuration




(A) Special purpose all parallel formant configuration

Figure 2.3: Klatt's synthesizers Source: [6]

has been added to parallel branch specifically to synthesize high frequency noise in /s/ and /z/. Radiation characteristic of +6 dB/octave is used to simulate the radiation load presented by the lips.

## 2.3 Cascade pole-zero synthesizer

As a modification to cascade/parallel all pole model Klatt synthesizer, a cascade pole-zero synthesizer has been developed at IIT Bombay ( [7 ] , [8 ] ). In this model the individual poles and zeros are used to simulate the transmission in vocal tract by cascade connection of resonators and antiresonators. This scheme is depicted in Fig. 2.4. The voicing is simulated by impulse train of fundamental pitch period which is directed through (RGP and RGZ) the resonator and antiresonator. Amplitude of voicing is controllable . The frication is simulated by random number generator and a low pass filter. Amplitudes of frication and aspiration are controllable. There are six resonators (R1 - R6) connected in cascade and five antiresonators (RZ1 - RZ5) connected in cascade. These antiresonators simulate zeroes in segments involving frication. Vowels are simulated by using voiced excitation source of resonators in cascade. Unvoiced fricatives and bursts portions of unvoiced stops are simulated using frication excitation. For voiced stops and voiced fricatives (mixed mode excitation), the periodic impulse generator modulates random noise generator. Nasalization is achieved using additional resonator and an antiresonator, both of which are kept at some frequency (270 Hz for adult male) during non-nasalized sounds and the nasal zero frequency is increased during nasalization. For exact cancellation in non-nasalized utterances the frequencies as well as the bandwidths of the nasal poles and the nasal zeroes must be the same.

This program takes values of parameters from parameter files and generates synthesized files. It has 39 parameters, 34 of which can be time varying (however, not all parameters need to be time varying). Generation of parameters is achieved by another program PARTRC ([8]), which permits graphical specification of parameter tracks. Table 2.1 gives the synthesizing parameters. We will be using cascaded pole-zero method for speech synthesis.

## 2.4 Synthesizer strategies

The synthesizer is a tool for synthesizing utterances from parameters, however, for word synthesis it is important to select a proper algorithm to pass the parameters to the formant synthesizer. Before going in to the synthesis strategies, one must mention something about basic unit of the speech which is selected for storage. We may store the parameters for phones, i.e. the very basic units of speech. However, even larger units of speech have been used for synthesis. The size of the basic unit determines the size of the database (the

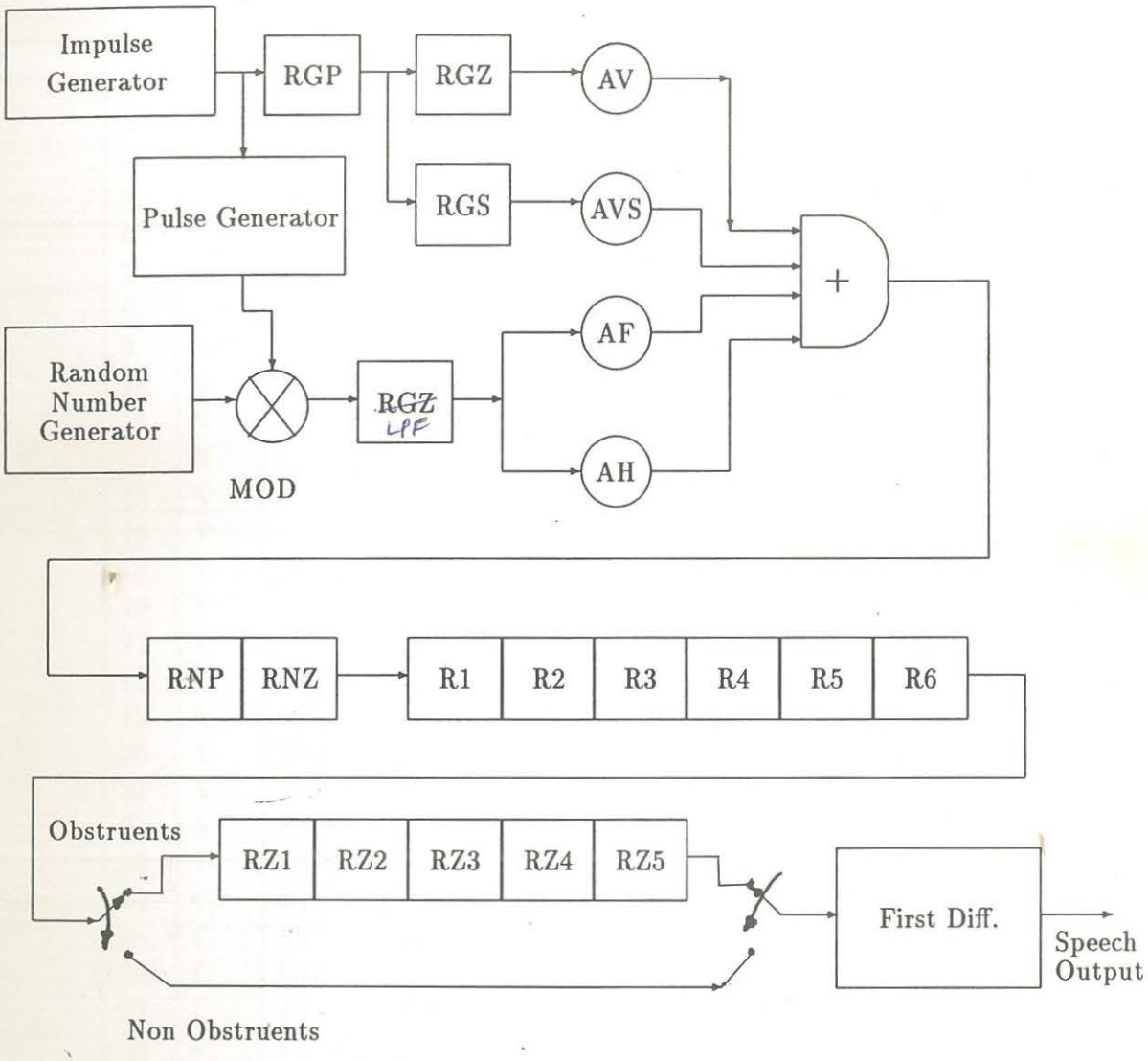



Figure 2.4: Cascaded pole-zero configuration. Source : [7]

Table 2.1: Control parameters of cascaded pole-zero synthesizer. Source : [7]
V/C = Variable/Constant

| No. | V/C | Symbol | Parameter Name | Parameter Values | | |
|---|---|---|---|---|---|---|
| | | | | Min | Max | Typ |
| 1 | C | NF | Number of formants | 4 | 6 | 4 |
| 2 | V | NZ | Number of zeroes | 0 | 4 | 0 |
| 3 | V | FO | Fundamental freq. of Voicing (Hz) | 0 | 500 | 0 |
| 4 | V | AV | Ampl. of Voicing (dB) | 0 | 80 | 0 |
| 5 | V | AF | Ampl. of frication (dB) | 0 | 80 | 0 |
| 6 | V | AS | Ampl. of sinusoidal Voicing (dB) | 0 | 80 | 0 |
| 7 | V | AH | Ampl. of aspiration (dB) | 0 | 80 | 0 |
| 8 | V | F1 | First formant freq. (Hz) | 150 | 900 | 450 |
| 9 | V | F2 | Second formant freq. (Hz) | 500 | 2500 | 1450 |
| 10 | V | F3 | Third formant freq. (Hz) | 1300 | 3500 | 2400 |
| 11 | V | BW1 | First formant bandwidth (Hz) | 40 | 500 | 50 |
| 12 | V | BW2 | Second formant bandwidth (Hz) | 40 | 500 | 70 |
| 13 | V | BW3 | Third formant bandwidth (Hz) | 40 | 500 | 110 |
| 14 | V | FZ1 | First zero freq. (Hz) | 150 | 5000 | - |
| 15 | V | FZ2 | Second zero freq. (Hz) | 150 | 5000 | - |
| 16 | V | FZ3 | Third zero freq. (Hz) | 150 | 5000 | - |
| 17 | V | FZ4 | Fourth zero freq. (Hz) | 150 | 5000 | - |
| 18 | V | BZ1 | First zero bandwidth (Hz) | - | - | - |
| 19 | V | BZ2 | Second zero bandwidth (Hz) | - | - | - |
| 20 | V | BZ3 | Third zero bandwidth (Hz) | - | - | - |
| 21 | V | BZ4 | Fourth zero bandwidth (Hz) | - | - | - |
| 22 | V | FMZ | Nasal zero freq. (Hz) | 200 | 700 | 250 |
| 23 | V | FMP | Nasal pole freq. (Hz) | 200 | 500 | 250 |
| 24 | C | UPDT | Parameter update int (ms) | 2 | 20 | 5 |
| 25 | C | SR | Sampling rate (Hz) | 5000 | 20000 | 10000 |
| 26 | C | GO | OVerall gain control (dB) | 0 | 80 | 0 |
| 27 | C | F4 | Fourth formant freq. (Hz) | 2500 | 4500 | 3300 |
| 28 | C | F5 | Fifth formant freq. (Hz) | 3500 | 4900 | 3750 |
| 29 | C | F6 | Sixth formant freq. (Hz) | 4000 | 4999 | 4900 |
| 30 | C | BW4 | Fourth formant bandwidth (Hz) | 100 | 5000 | 250 |
| 31 | C | BW5 | Fifth formant bandwidth (Hz) | 150 | 7000 | 200 |
| 32 | C | BW6 | Sixth formant bandwidth (Hz) | 200 | 2000 | 1000 |
| 33 | C | BWNZ | Nasal zero bandwidth (Hz) | 50 | 500 | 100 |
| 34 | C | BWNP | Nasal pole bandwidth (Hz) | 50 | 500 | 100 |
| 35 | C | FGP | Glottal res. 1 freq. (Hz) | 0 | 600 | 0 |
| 36 | C | BWGP | Glottal res. 1 bandwidth (Hz) | 100 | 2000 | 100 |
| 37 | C | FGZ | Glottal zero freq. (Hz) | 0 | 5000 | 1500 |
| 38 | C | BWGZ | Glottal zero bandwidth (Hz) | 100 | 9000 | 6000 |
| 39 | C | BWGS | Glottal res. 2 bandwidth (Hz) | 100 | 1000 | 200 |

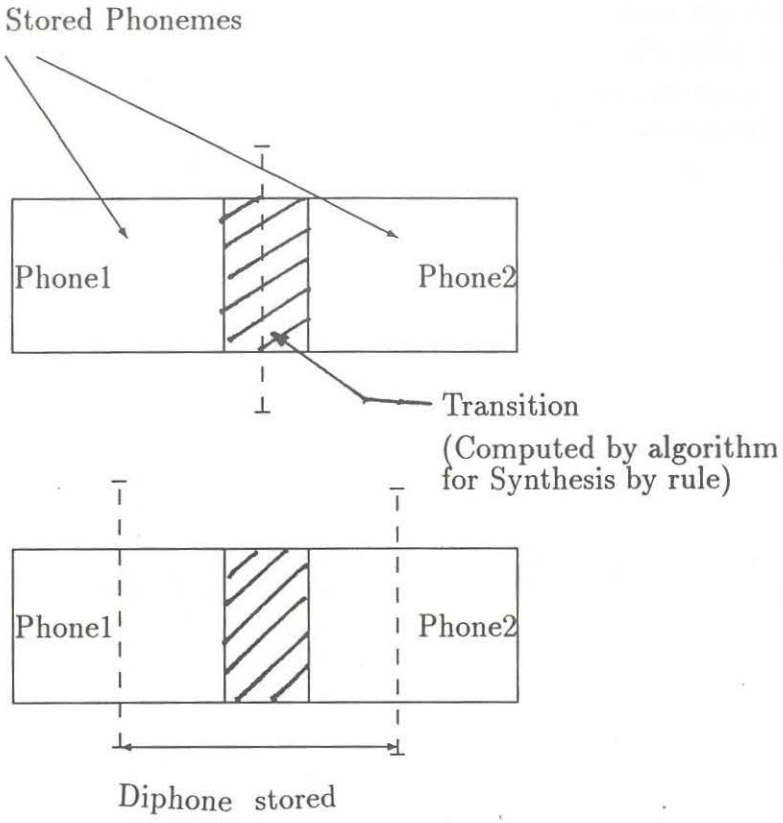



Figure 2.5: Difference between storing phonemes and diphones

look-up table), larger the unit, larger is the size of the database. However the decrease in the size of the unit means that more processing is required but the database is smaller.

In order to get a proper perspective, we briefly review the synthesizer strategies. The strategies that will be discussed here are

1. Diphone synthesis
2. Synthesis by Rule
3. Allophone synthesis

### 2.4.1 Diphone Synthesis

If parameters for the phones are stored then the synthesis strategy decides how to interpolate parameters between these units, but one may decide to slice a phone in middle rather that at boundary and store the transition of parameters as shown in Fig. 2.5. These transitions are stored for all the possible phone combinations for that particular

language. The resynthesis is done simply by concatenating the coded transitions. This is quite popular with synthesis by analysis systems, usually using the LPC analysis ([9]). This system however is not very flexible and can achieve context sensitivity only between adjacent phones and in many languages the sensitivity can extend to three or more consecutive phones. The storage of transitions also means a larger data base, but it also implies consequent reduction in computation. This kind of synthesis is used for real time implementation using digital hardware.

### 2.4.2 Synthesis by rule

This is a scheme for converting input string of phonemes to a set of parameters for a synthesizer by interpolating between target values for parameters [6], [16]. The target values for all basic units are stored. The algorithm consists of a set of rules to map a string of parameterized linguistic units such as phones into parameter values for driving a speech synthesizer. These rules take into account the context sensitivity, the prosodic information, intonation, etc. It also evaluates the durational information for the particular utterance. Generally, the rule set is complex and computation is involved. The storage data base is however considerably smaller than that required for diphone synthesis.

### 2.4.3 Allophone Synthesis

The allophone is the variation of the basic utterance depending on the context. This scheme stores the parameter values for all possible variations that are allowable in the language. The storage of allophones also carries an overhead in terms of the memory, however this is considerably lesser than the storage of the diphones which in most of the languages are more than the number of allophones. This technique tries to strike a compromise between diphone synthesis and phoneme synthesis. Here one stores the allophones to reduce the computational complexity of the algorithm. A trade off between increased memory and complexity of algorithm is achieved.

## 2.5 Closing remarks

This chapter gave us an overview of the synthesizers that can be used in conversion of the text to speech. Also discussed were a few of the synthesis strategies that seemed as likely candidates for the system being designed. The synthesis-by-rule approach is very flexible and we will attempt to implement it. While using synthesis-by-rule approach, one should remember that the synthesizer being used is not of prime importance. Most of the synthesizers would give a good result if parameter variations (called parameter tracks) are precise. The parameter tracks will be generated for cascaded pole-zero synthesizer (Kulkarni, 1992). In order to generate the target values for parameters of all

basic phonemes in Marathi one must have good analysis tools. In the next chapter, we discuss tools for generating the parameter tracks and methods to generate the tracks.

# Chapter 3

# Synthesis Tools

As mentioned in the previous chapter, generation of correct parameters hold a key to the success of the synthesis-by-rule system. In order to generate precise parameter targets, it is essential to have a good reference data. The reference unit is dependent on the synthesis strategy. The basic unit of speech selected here is a phoneme. Various aids are available for generating the parameter tracks. These parameters have to be finalized by using theory, analysis, and experimentation within the synthesis. Here, we look at the tools available for for generation of parameter tracks of phonemes, and on basis of these, a generalized method for generation of parameter tracks will be outlined. The suitability of parameter tracks for synthesizer and synthesis strategies require listening tests. A system will be described for outputing speech data files of arbitrary length.

## 3.1 Spectral analysis

An analysis package, `span`, was earlier developed by [10] and [8] , for analyzing a speech segment of user defined length. The log magnitude spectrum of the speech segment and the LPC (Linear Predictive Code) spectrum can be plot using this, it uses an auto regressive filter model of speech production. The LPC spectrum is used for determining the formant frequencies and their bandwidths. After the completion of the analysis, cursor can be used to locate the formant frequencies. The bandwidths will be evaluated automatically. All the parameters thus evaluated are tabulated and also displayed graphically. These can be stored in a file and can then be appended to the original parameter file. The method is slow and needs human intervention at every frame. This package is a good starting point for analyzing any general utterance. Additionally, programs for merging the tracks generated to old parameters is available and is called `mergefile`. These "natural" formant data generated by the analysis of the natural segment are updated every 25 ms. However the parameter tracks for the synthesizer program `partrc` are to be specified every 5 ms. The data in the intermediate range is generated by linear interpolation. Thus the analysis of natural speech segment can be used for generating parameters, in place direct graphical

specification using `partrc`. The natural segments that are used for analysis are from a male speaker.

## 3.2 Graphical generation of the parameter tracks

The procedure of generation of parameter tracks is a trial and error procedure but is based on some *a priori* knowledge of the parameters. The trial and error process would be very time consuming if there is no visual feedback. A program `partrc` implemented by [8] is capable of graphically editing the parameter tracks. The cursor controls can be used to mark the points graphically. All the parameters that are variable can be edited in graphical mode. The display is done by approximating the tracks with suitable line segments between the points marked by the user. The program can select either cascade/parallel or cascade pole-zero synthesizer. The program can modify the tracks from existing parameter files. There is, however, a small shortcoming with this program. The program does not use the information from the parameter track vectors from the parameter file for displaying the parameter tracks, but uses the information stored at the end of file for this purpose. These are the points marked in the graphical display by the user. This makes it possible to view only those parameter tracks which have been made on the `partrc`.

In the opening menu one specifies the parameters which are to be variable or constant depending on the requirement, also displayed are the extremal limits for the parameters. The option to modify the tracks is given after this selection is complete. Single or multiple tracks can be modified. The further details and options can be found in [5]. This is the single most important tool that has been used for the phoneme synthesis. The reference phonemes have been extensively tailored for good results on this program.

## 3.3 A digital spectrogram

A digital spectrogram developed by T. G. Thomas [11], is used for determining the approximate tracks for the natural speech segment. It also provides the option of observing either the wide band or the narrow band spectrogram. This program uses a 16 level gray tone scale in VGA mode to display the spectrogram. The program lets you select the segment to be analyzed by placing starting and ending markers in the appropriate positions. The segment is then resized to fit to the entire window and one is allowed to select the window length for computing the FFT. The window length is selectable from 10 to 512 samples. The amplitude of the FFT spectrum is mapped into 16 levels. The amplitude extremum are requested from the user so that he/she can specify the range he/she is most interested in. This is useful in analyzing utterances with different energies in different regions. In such cases one must supply limits that will highlight the desired region instead of blending it because of inadequate dynamic range.

The window specification option is for making observations on wideband (greater time resolution) or narrow band (better for observing transitions). The display consists of two windows: spectrogram, and time domain waveform.

This program has been modified a little to enable it to generate hard copies on normal dot matrix printer. This printing is, however, done by half toning and hence the resolution of the printer is only of 8 levels, and as such the quality of the output suffers. A major part of the analysis of the data segments is carried out on this package. The natural and the synthetic utterances can be compared very easily on basis of their spectrograms.

## 3.4 General procedure for generation of parameter tracks

To generate any parameter track values in terms of the various parameters required for the cascaded pole-zero synthesizer, the following procedure may be adopted.

1. If any *a priori* knowledge of the utterance is not available then a natural segment of the utterance is evaluated on a wide band digital spectrogram (this is done for couple of speakers so that any analysis errors may be averaged). From this analysis approximate formant tracks may be obtained. This would also give us a rough idea of the amplitude and bandwidth tracks. The tracks can also be found by the SPAN program.

2. Using these data as a rough guideline we create graphical tracks on PARTRC.

3. The tracks are manipulated by hand where ever necessary until the perception is close to that of a natural utterance.

4. Informal perception tests with many listeners are necessary to check the synthesis as a single listener tends to get biased by the knowledge of what the utterance is supposed to be.

5. In case we have some *a priori* information about the tracks that can be used as the starting point for the initial tracks. Even under these conditions it is necessary to check the natural tracks.

The trial and error procedure described herein outlines a general method to generate the tracks. The special issues for various kinds of utterances are discussed in next chapter.

## 3.5 Acquisition of natural utterences

The natural segments, used for anaylsis, are obtained from a data acquisition system interface to an IBM PC-AT compatible computer. The system is shown in Fig. 3.1 The

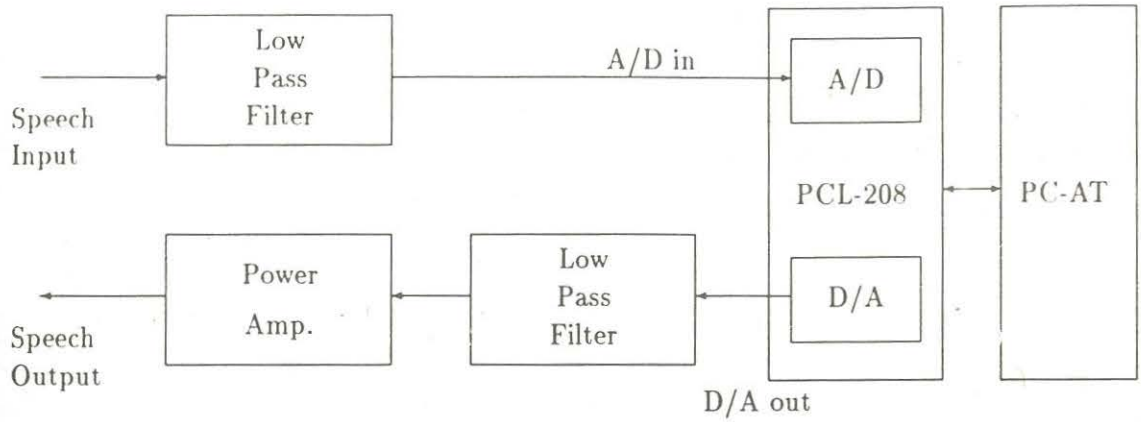



Figure 3.1: Hardware for speech input and output

system can acquire the data segments from either a microphone or an audio cassette player. A preamplifier is used to amplify the signal from microphone. The cassette player AIWA AD-R707 is used to record and play back the utterances. The signal to be acquired is passed through an antialiasing filter with low corner frequency of 4.65 kHz, [2], [18] before being digitized. The digitization is done by PC add on board PCL 208 [12].

The digitization is done with the help of a program, `mad` [8]. It can digitize the input waveform on a single channel at the rate of upto 50 kHz. The maximum number of samples that can be thus acquired is 16000 (Corresponding to 1.6 seconds of data at 10kHz). This data can be stored in a file in either binary or text format. The program allows the user to check the acquired data by presentation through D/A port before its storage. The first entry in the file indicates the number of data elements in the file. This is followed by sample integer values.

## 3.6 Presentation of synthesized speech

The synthesized speech files have to be output through D/A port of data acquisition board installed in the PC. The analog output from the D/A convertor is applied to a smoothening filter and audio amplifier and then listened through headphones or speaker. The data acquisition board used for this purpose is the same as that used for acquiring the segments for analysis. A program, developed earlier [8] was available for presenting speech signals upto duration of 1.6 seconds. A new program was written for handling speech data files of arbitrary length. In this section, we briefly describe this presentation program.

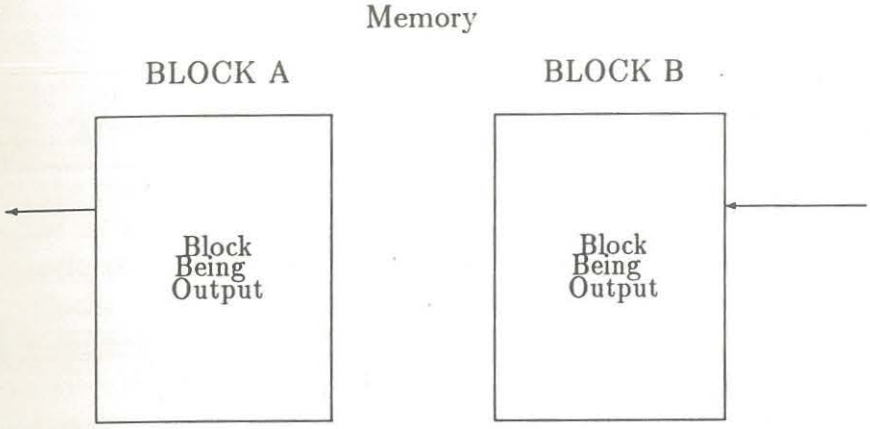



Figure 3.2: The scheme of using two blocks for presentation

### 3.6.1 Presentation program

The presentation of data to DAC port can be done by reading the data from a file into an array and then, under program control, output the data to the desired port. This approach [8] places a limit on the maximum number of items that can be presented. If one could output the data directly from the file then one need not have the limitation of number of data item. However, presenting the data directly from a file doesn't work because of overhead of reading the file is large when read requests are too frequent (like 10000 times a second). In order to overcome this problem, the data from the file was read into two blocks (as depicted in Fig. 3.2 ). Before the start of presentation the block A is filled with zero valued samples. An interrupt service routine (ISR) is used to output the data to the D/A port, from this block while block B is being filled by data from the input file. The data is output till the end of block and at this point the address of the block from which data is output is changed to that of block B. Meanwhile the block B may or may not be completely filled. In case, block B is filled the address of the block to filled is changed to that of block A and process continues till the end of the block. The last block, in general, may have a size different from the block size. This size is kept accessible to ISR through a global variable, value of which is computed as soon as the file is opened. Another global variable also keeps track of the number of blocks output.

The timing for the DA conversion is generated from the AD convertor. This is done because there is no method of generating the interrupts directly for DA conversion. The AD convertor is programmed by onboard timer (8254).

The required file should be in a binary format with first entry depicting the number of data items in the file. If the first entry were to be of integer type (16 bits) then the number of data items is limited to 64K ( if unsigned integer is used), so an unsigned long integer (32 bits) is used to ensure that the file size can be very large ($2^{32}$).

### 3.6.2 Testing of speech presentation program

To test the program a sine wave data with 100,000 samples, and 100 points per cycle was generated in a file. The continuity of this waveform was monitored on an oscilloscope. The waveform at 10kHz remained stable, but there was a slight flicker on the boundaries of the blocks. This flicker was visible only on oscilloscope. The monitoring of tone on speaker did not indicate this timing lapse. If the file is stored on a floppy disc, however, a perceptible glitch could be heard on the boundaries of the blocks. A very stable output was obtained when the file was saved on a virtual disk (RAMDRIVE) [13].

The file presentation program works well with virtual disk. The size of the virtual disk is the only limiting factor now. The original program however used to resize the data in order to use the dynamic range of the DAC completely, but since the new program does things online directly it is not possible for the program to find the appropriate scaling factor for the data, hence the dynamic range may remain under utilized. One can however do this dynamic resizing prior to presenting the data, and present the file with resized data with the presentation program.

# Chapter 4

# Synthesis of phonemes

The phonemes are used as basic units of speech. It is essential that these phonemes be free from context effects, as we plan to introduce the contextual effects by algorithm. The phones are, however, tailored by hand till the best possible results are achieved. This is a two step trial and error method of analysis and synthesis and until an acceptable quality of speech is achieved. The various Marathi phonemes with Devanagari symbols and I.P.A (or I.P.A like) symbols are given in Appendix A. These phonemes can be classified into seven groups according to manner of articulation: Vowels, Nasals, Glides, Stops, Fricatives, Affricates. All the non-vowels are collectively referred to as consonants (Tab. 4.1).

synthesized for the target vowel /ʌ/. The synthesis part is of extreme importance to generate precise targets for the text-to-speech converter. The experience gained in synthesis of the basic phones was used in making the rules for the converter.

## 4.1 Vowels

Vowels or voiced continuants are so called because they can be uttered continuously (i.e they are non transient in nature), as such their parameters are static in nature. The perception of vowels is defined by the location of first two formants. A variation of these is normally sufficient to yield the distinguishable vowels. Appendix B gives parameter values for all the vowels. The linear downward tilt of the F0 contour imparts a naturalness to the utterance. The F0 of vowel reduces, by about 25%, linearly till the end of the utterance. The durations shown in appendix A are for single vowel and their durations decrease in the context. The data from a male speaker (author himself) was used for analysis and parameters are listed in the appendix B.

Table 4.1: Classification of consonants

| Excitation | UNVOICED | | VOICED | | NASALS | Place of articulation |
|---|---|---|---|---|---|---|
| Group | Aspirated | Unaspirated | Aspirated | Unaspirated | | |
| STOP | k | $k^h$ | g | $g^h$ | $\eta$ | velar |
| AFFRICATE | $t\int$ | $t\int^h$ | dz | $dz^h$ | $\tilde{n}$ | alveopalatal |
| STOP | ṭ | $ṭ^h$ | ḍ | $ḍ^h$ | ṇ | alveolar |
| STOP | t̲ | $t̲^h$ | d̲ | $d̲^h$ | n | dental |
| STOP | p | $p^h$ | b | $b^h$ | m | bilabial |

## 4.2 Nasals

Nasalization of utterances is caused by coupling of nasal cavity with the oral cavity. This causes nasal antiresonances to decrease the amplitude near the antiresonance frequency. This effect may be achieved in two ways :

1. Activating the nasal pole at he desired frequency. This is done by shifting away the nasal zero so that the antiresonance is activated in the desired region. The effect is cancelled after this region by presence of the nasal zero.

2. Increasing the bandwidth of the pole nearest to the desired frequency. This will cause the amplitude to decrease in that region as the energy in that formant gets distributed. Its effect similar to that acheived by the nasal pole.

The nasals /m/ and /n/ are synthesized in this manner. The nasal /$\tilde{n}$/ was synthesized from the available track of /j/. The bandwidth of the first formant was increased so that it gave the effect of nasalization, further the nasal pole was also activated at 270 Hz. The transition time of various formants are 60, 50 and 60 ms respectively for F1, F2 and F3. There is an improvement of the phoneme $\eta$ by addition of frication for 25 ms with a sharp peak to account for the slight /g/ like perception. This may be because the nasal is generally targetted at the consonant in that particular row and makes the utterence sound closer to the natural speech for the listener. This can be seen in Fig. 4.1 which shows the amplitude tracks and Fig. 4.2 which shows the formant tracks of /$\eta$/.

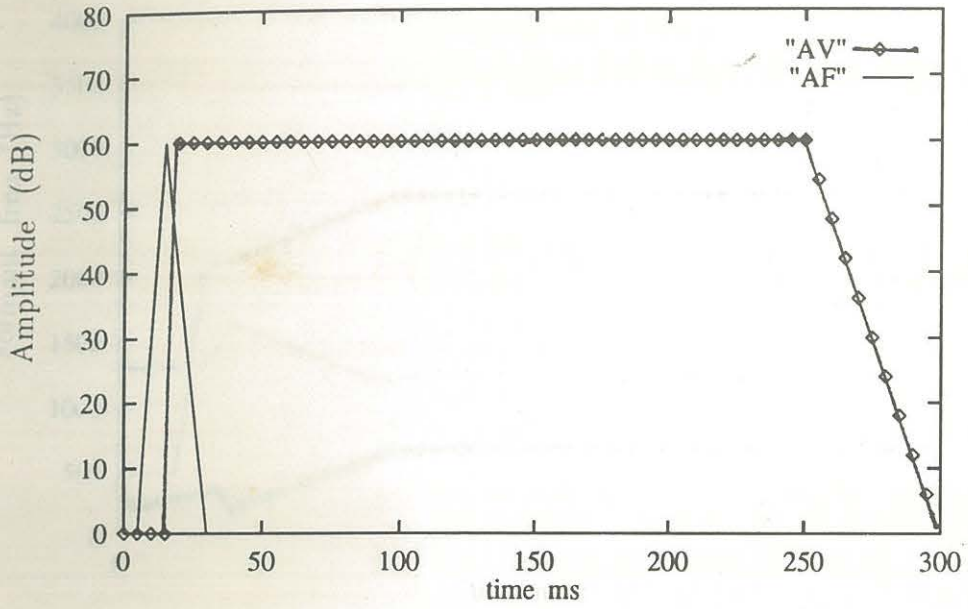



Figure 4.1: The Amplitude tracks of the nasal /$\eta$/

## 4.3 Glides

The glides /w/ and /j/ are synthesised from the analysis data. The formant tracks of the glides are clearly visible in the spectrogram. The glides are formed by smooth transitions from one continuant to other. The synthesized glides are very clearly distinguishable. The tracks of /j/ are given in Fig. 4.3 Fig. 4.3 and Fig. 4.4 for reference.

## 4.4 Liquids

The liquids /r/ and /l/ have been are characterised presence of quasi-sinusoidal voicing in the transition region. The analysis of /r/ shows that the formant tracks begin the transtion in 30 ms from the start of the phoneme, roughly at the same time as the amplitude tracks make the switch. The pitch acccent is at 30 ms. In the case of /l/ , however the formant transitions start later but they make more rapid transitions (in roughly 40 ms). The figures 4.5 and 4.6 depict the amplitude and formant tracks for /r/ and figures 4.7 and 4.8 show those for /l/.

## 4.5 Stops

Stops form a single largest group of phonemes in Marathi. There are 16 stops (see Table 4.1). The evaluation of formant tracks for stops is difficult because of low energy in

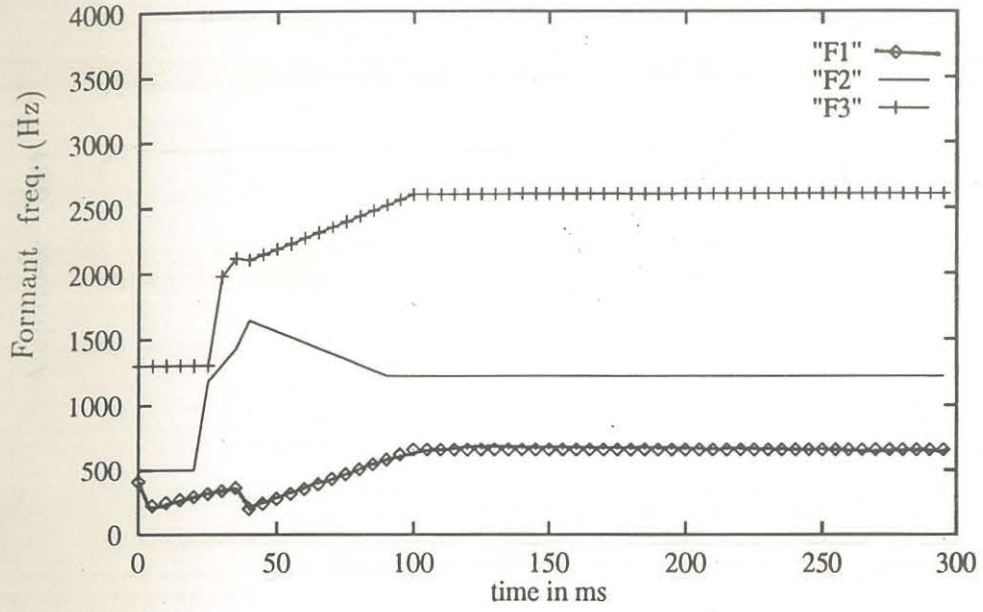



Figure 4.2: The Formant tracks of the nasal /$\eta$/

region of stops as compared to that in the following target vowel. The low energy is due to lack of voicing (in some cases) or due to low energy of voicing.

The bandwidth of the formants help in spreading or concentrating the frication energy present in that particular band. For example, after the evaluation of parameter tracks by spectrogram for /$k^h$/ and the corresponding synthesis, it was found that by increasing the bandwidths of first and fourth formants the utterance sounded as /k/. This is due to smoothing of frication energies due to even wider distribution. However when parameter tracks for /k/ were evaluated (they were slightly different) and /$k^h$/ synthesis was tried by reducing the said bandwidths, same effect was not quite achieved [1]. In general, wider bandwidths led to softer sounds and narrower ones led to sharper sounds.

The amplitude tracks of various excitations (frication, aspiration, voicing, etc.) also play a deciding part in the synthesis of a phoneme. This is so especially in the case of stops where onset and offset of various amplitude tracks may make all the difference in perception of the phoneme. The frication energy burst in /$k^h$/ resulted in a sharp glitch if the reduction was linear, however, the reduction in frication energy in an exponential form results in glitch free /$k^h$/ (see Fig. 4.9).

The major difference between third column stops and fourth column stops in Table 4.1 is in the aspiration energy. The /$g^h$/ was synthesized from /g/ by setting the aspiration on to a very high level (80 dB) for a short time (30 ms) in the /g/ parameter tracks. The timing for onset and offset was decided by analysis of /$g^h$/ segment. Again, exponentially

[1]This might be due to error in evaluating the tracks of low energy stops as compared to that of /$k^h$/ which has higher energy in initial region on account of aspiration present in the initial region

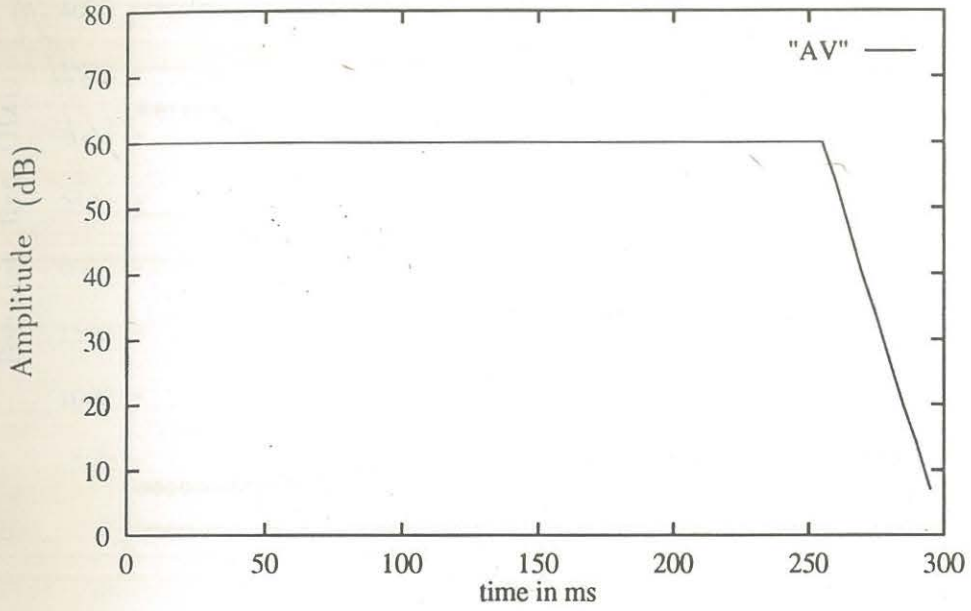



Figure 4.3: The Amplitude tracks of the Glide /j/

increasing and decreasing the aspiration energy resulted in glitch free phoneme, as can be seen from Fig. 4.10. If voicing energy is also high at the same time as the aspiration a whistling effect occurs. This could be due to crossing over of the dynamic range of the numbers in the filter stages of the synthesizer due to large excitation. This problem can be overcome by reducing the voicing, in case large voicing energy is used.

## 4.6 Affricates

Affricates /t ʃ/ and /dz/ have been synthesized. They are characterised by initial frication and quasisinusoidal voicing. The formant trajectories of both /dz/ & /t ʃ/ exhibit exponetial curves after a plateau. The F1 curve in /dz/ is increasing toward the target supplied by F1 value of /ʌ/ and F2 curve reduces exponentially to the respective target of F2 of /ʌ/.

## 4.7 Fricatives

Fricatives are characterised by presence of frication energy in the spectrum. This frication energy is concentrated in the spectrum depending on the fricative itself, and place of articulation. The fricatives ( /ʃ/,/ʃ̩/, /s/, /h/) are all unvoiced with different places of articulation. In /ʃ/ (Fig. 4.11) the amplitude of frication is high initially and it decreases towards the middle of utterance (roughly after 120 ms) in /s/ both onset and offset

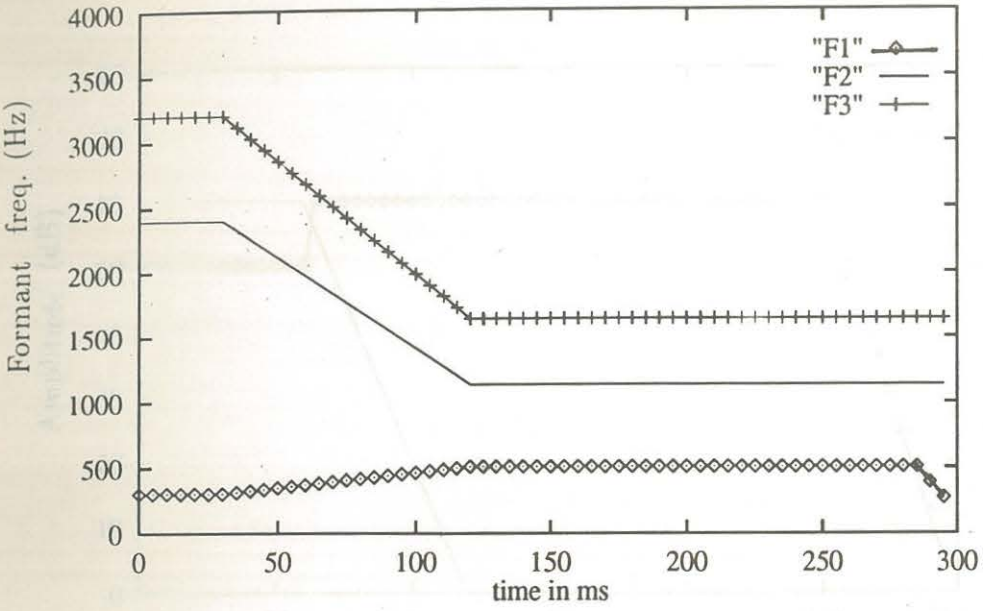



Figure 4.4: The Formant tracks of the Glide /j/

of frication is more gradual to account for smoother flow of air through wider opening as compared to that for /ʃ/ (also for /ʂ/). The (/h/) is aspirated for about 90 ms (Fig, 4.14) and the bandwidth tracks are kept variable to relocate the aspirations and in their spectrum. This energy is concentrated in the spectrum depending on the place of articulation of the fricative. All are unvoiced with different places of articulation. The transition in case of these utterences is rapid and begins only at the edge of the phonemes and the transition time is 30 ms. The utterence quality is remarkably clear, and distinguishable.

## 4.8 Closing Remarks

The synthesis of all 43 phonemes provided enough information about the synthesizer to be able to do reasonable synthesis by algorithm. Though formal listening tests have not been carried out, informal perception tests were carried out with native speakers of Marathi.

The vowels were very clearly intelligible. The nasals /m/, /n/, /ñ/ and /ŋ/ were also reasonably good. However /ɳ/ could not be synthesized well enough to be distinguishable. The unaspirated voiced stops /g/, /dz/, /ɖ/, /d̲/, /b/ are clear but aspirated voiced stops /$g^h$/, /$ɖ^h$/, /$d̲^h$/, and /$b^h$/ are not very clearly distinguishable. Affricates, glides and liquids are quite clear.

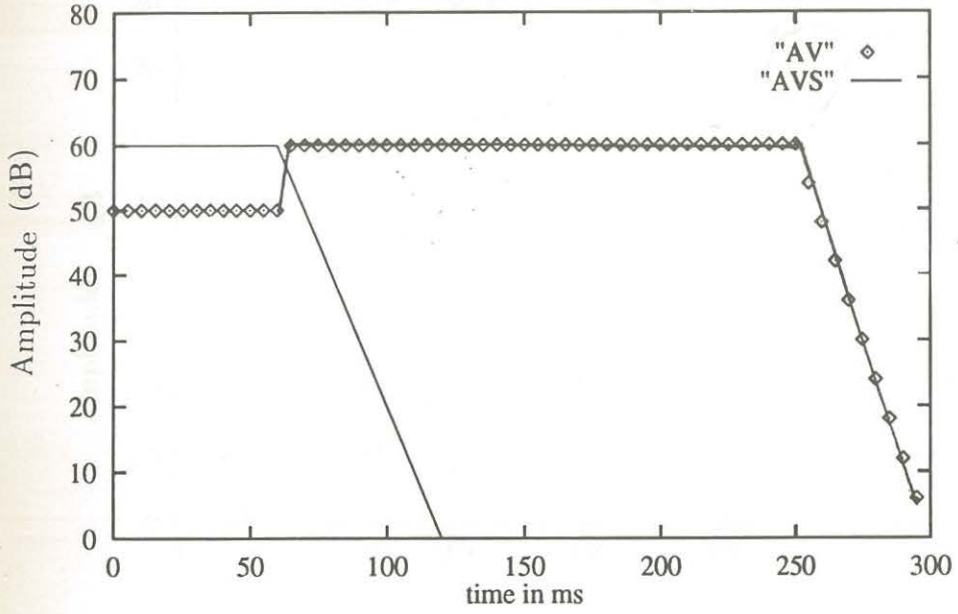



Figure 4.5: The Amplitude tracks of the Liquid /r/

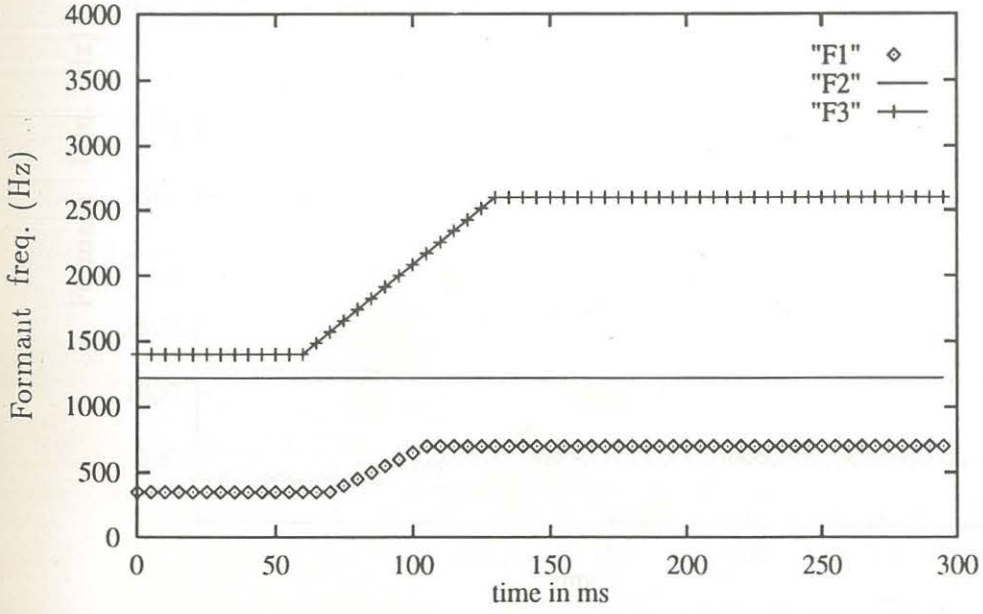



Figure 4.6: The Formant tracks of the Liquid /r/

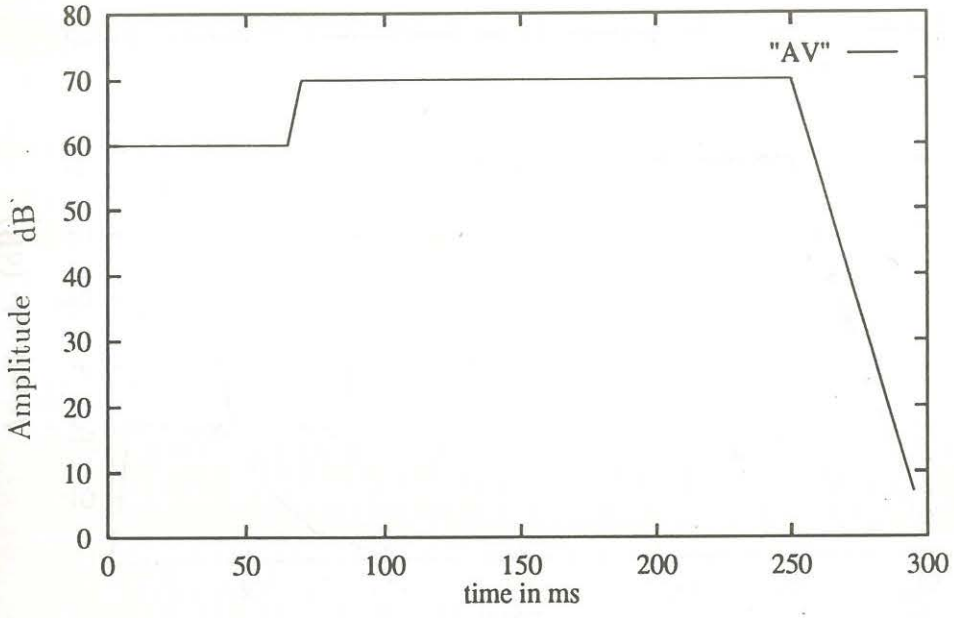



Figure 4.7: The Amplitude tracks of the Liquid /l/

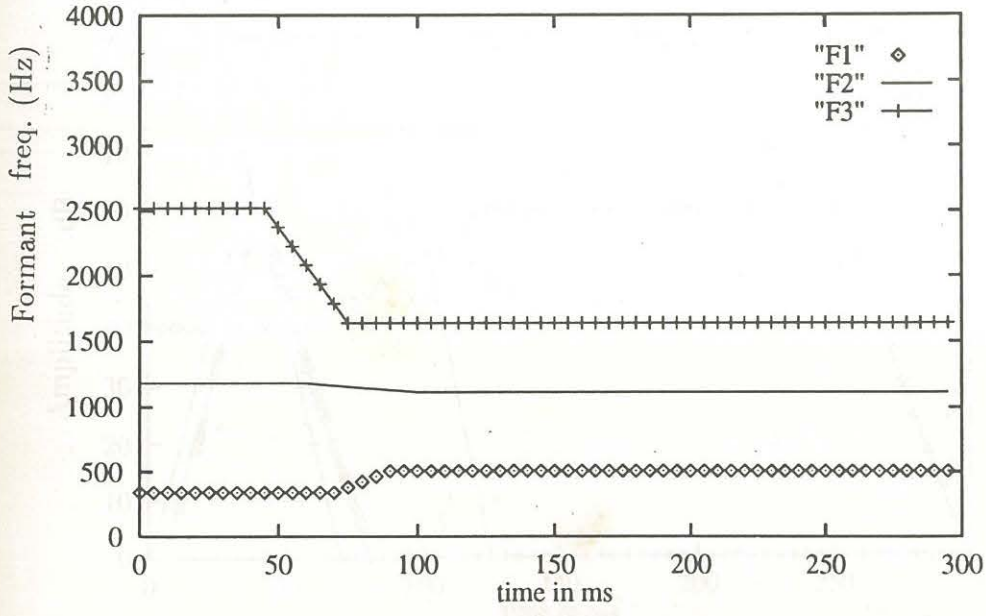



Figure 4.8: The Formant tracks of the Liquid /l/

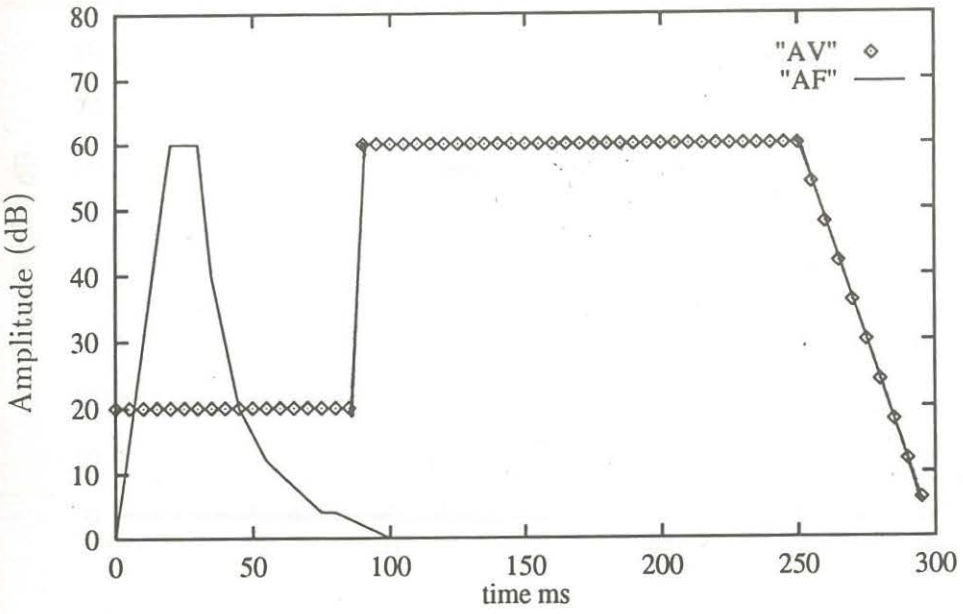



Figure 4.9: The Amplitude tracks of the Fricative /$k^h$/

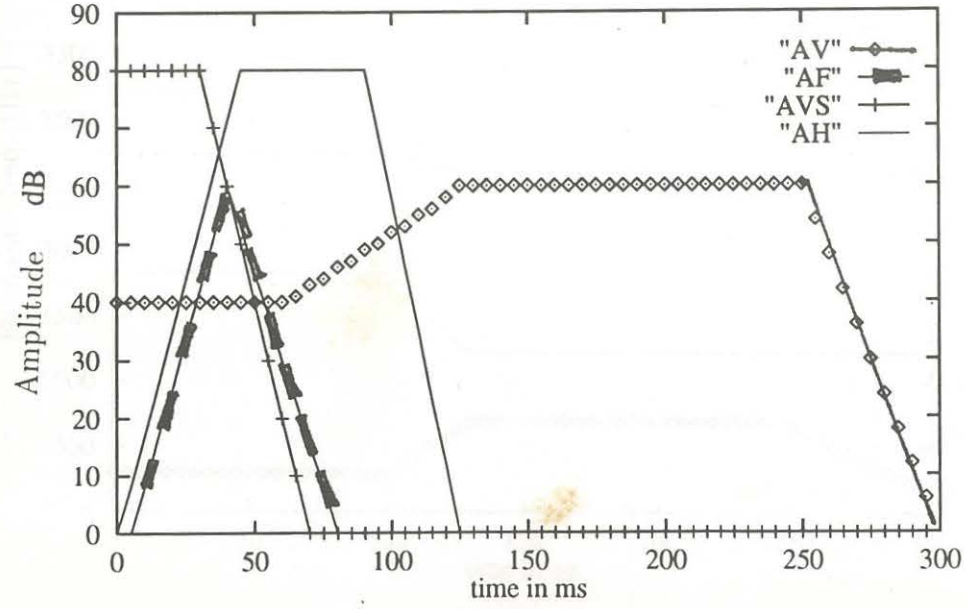



Figure 4.10: The Amplitude tracks of the Fricative /$g^h$/

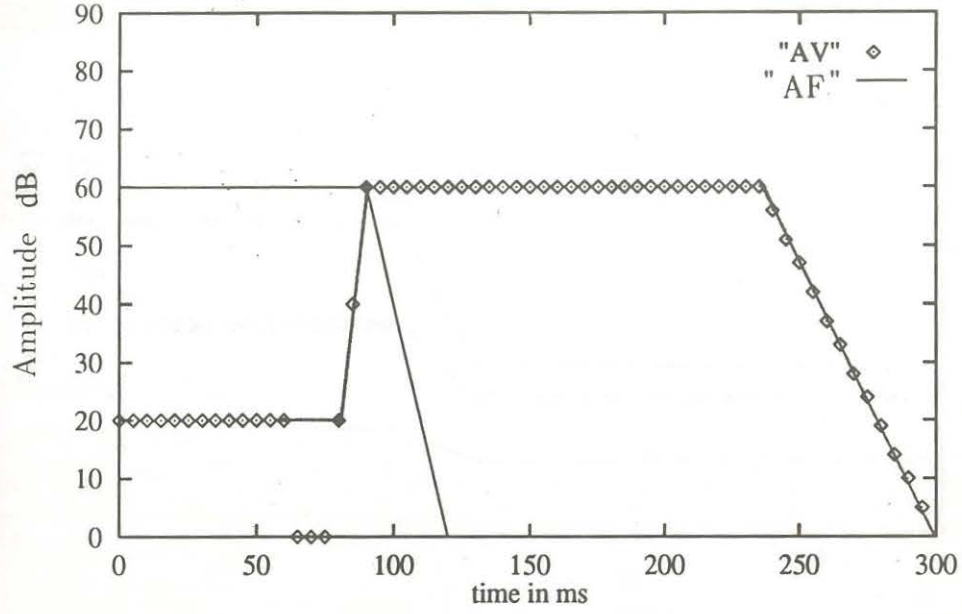



Figure 4.11: The Amplitude tracks of the Fricative /ʃ/

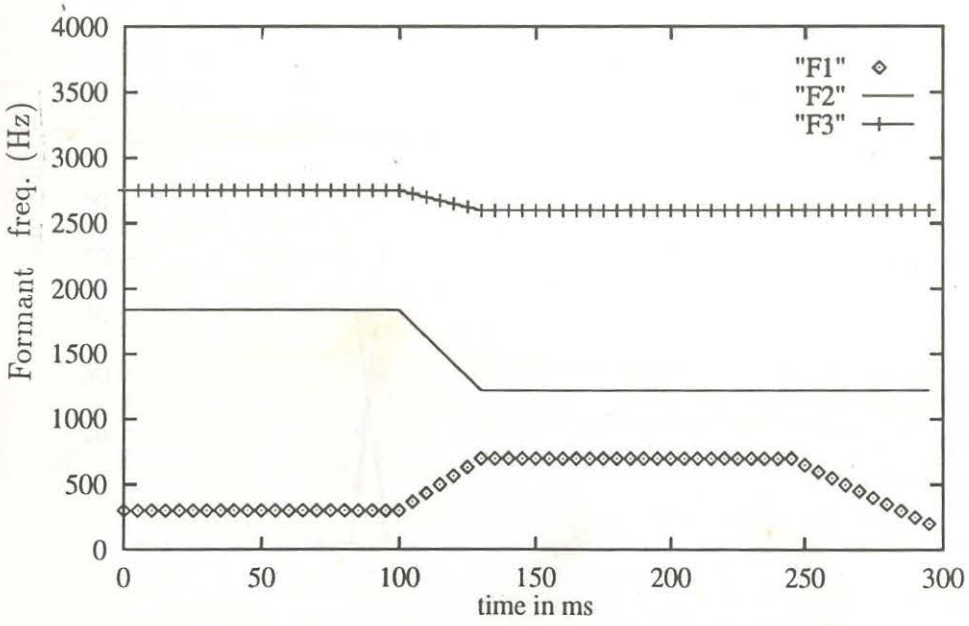



Figure 4.12: The Formant tracks of the Fricative /ʃ/

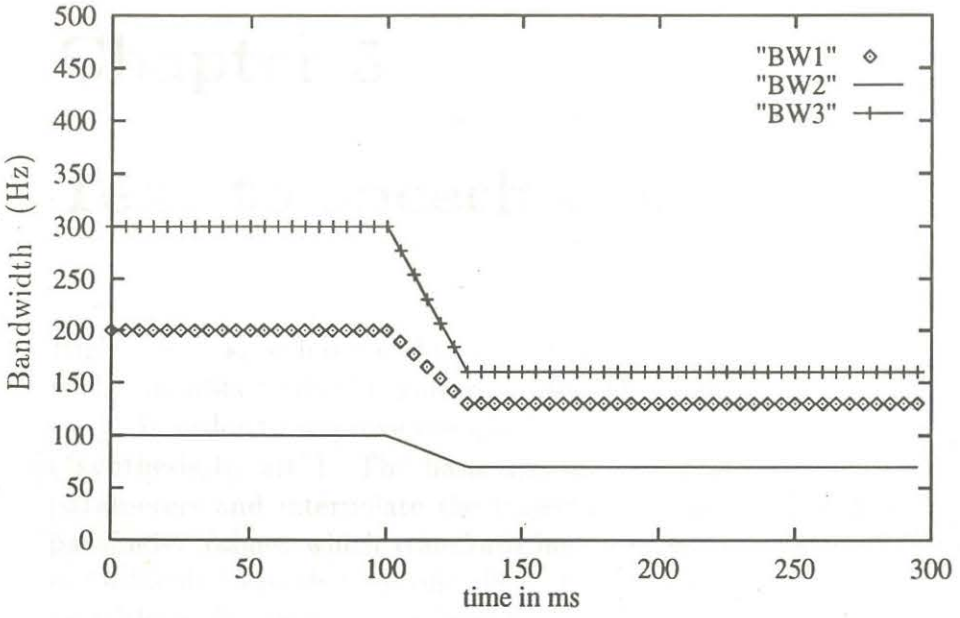



Figure 4.13: The Bandwidth tracks of the Fricative /ʃ/

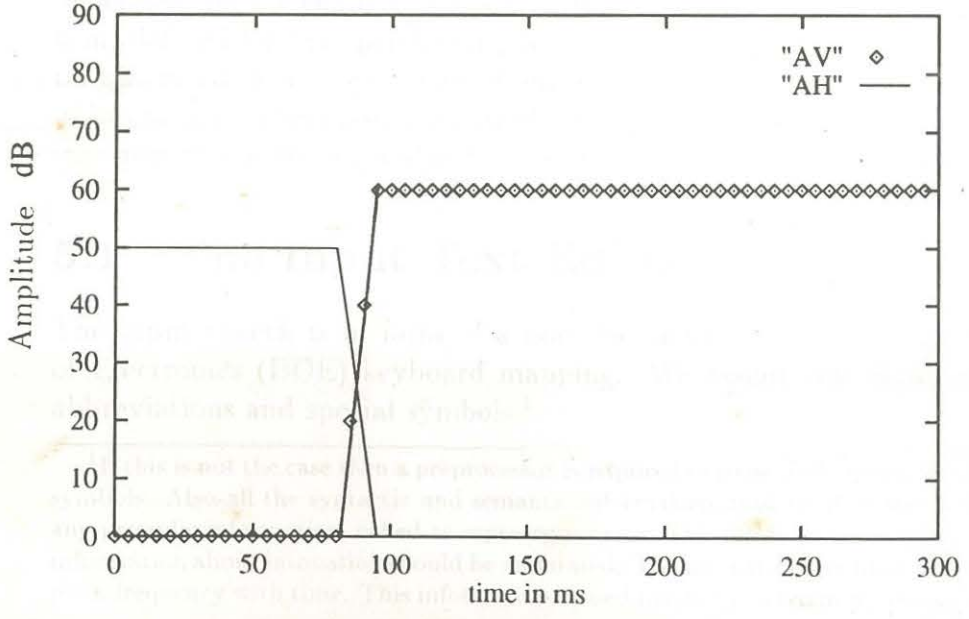



Figure 4.14: The Amplitude tracks of the Fricative /h/

# Chapter 5

# Text to speech converter

High quality speech is possible in most of the formant synthesizers, but the major difficulty is the inability to do this automatically. This high quality is possible only semiautomatically. In order to improve the quality, parameters are modified by hand frame by frame ("synthesis by art"). The basic approach to synthesis-by-rule has been to have target parameters and interpolate the trajectory between them by rule. The frame-by-frame parameter values, which transform into best speech output, exhibit discontinuities that are difficult to predict by rule ([9]). This result is not achieved in the synthesis-by-rule algorithms, because of oversimplification of the model used for generating these tracks, which result in the smoothing of the tracks. This trajectory will, in general, be different for various permutations of initial and final phonemes. This difference has to be simulated to take into account effects of co-articulation. The variation of intensity of a particular phone is again context sensitive. One must also consider aspects of durational change dependent on the context. These duration based rules modify the inherent or basic duration, defined for that particular phoneme, based on the contextual hypotheses that will be generated. For proper perception, it is also necessary to generate correct information of intonation. These issues are considered in detail and a frame work of the rules that are incorporated in the algorithm is constructed and is discussed in following few sections.

## 5.1 The Input Text Editor

The input speech is in form of a non document file (ASCII Codes) using Department of Electronics (DOE) keyboard mapping. We assume the input text to be free of all abbreviations and special symbols [1].

[1] If this is not the case then a preprocessor is required to parse and replace the abbreviations and special symbols. Also all the syntactic and semantic information must be included at this stage to incorporate any prosodic information, called as suprasegmentals. This information includes stress symbols also. Any information about intonation should be annotated. The intonation would be defined as the variation of the pitch frequency with time. This information is used to convey different meanings for same utterances. The

This input file is transcribed to its basic phonemes by a phonemic transcriptor. The input file is created using `akshar` bilingual text editor for Devanagari Script. Since Marathi uses a phonetic script the phoneme transcriptor is very simple. `akshar` saves its file in a simple format in which the symbols are saved as they are typed in (some phonemes may require more than one key strokes). The nasalization symbol is the only symbol with ambiguous phonemes. The nasalization phoneme depends on the target phoneme and the nasalization selected is the one which is closest to the place of articulation of the target phoneme, i.e. the nasal of the **row** to which that particular phoneme belongs (see Table 4.1). The complex symbols in Marathi are converted to their elementary phonemic form in the output file. The symbol '#' is used to mark the syllable boundaries in the parsed file. The white spaces are used as word boundary indicators. The output file created by this parser is presented to the text-to-speech converter as an input.

## 5.2 Overall strategy for cascade-pole-zero synthesizer

The phoemes which are hand tailored on `partrc` [8] are stored in memory and serve as look up data. This lookup data is used for generating the initial and target values for the TTS converter. The existing CPZ synthesizer will allow 34 variable tracks and 5 constants, however all of these 34 variables may be constant . In order to save on memory space requirement for stored phonemes they are not stored as they are made on `partrc`. To generate graphical information `partrc` generates a table which tells it the points of variable tracks marked during the hand tailoring. This information is deleted and then the tracks are saved in memory. This truncation of tail-end of parameters is done by a file adjusting program called as `fadjust`. This program accepts names of source file on command line in that order. The naming scheme for these parameter files is simple. Every phoneme keyed in from the `akshar` generated an ASCII code (Appendix A). The name of the file generated is simply C + $(\text{ASCII code})_{10}$. PAR e.g. if ASCII code of phonemes is decimal 168 then the parameter file name will be `c168.par`.

The number of tracks in an initial phoneme and target phoneme may be different and hence the region of co-articulation has to have a very general form i.e. all the tracks that can be possibly variable are kept variable.

The synthesizer generates an array of parameter files with different number of variable parameters. These parameters files are named `out0.dat`, `out1.dat`, etc. The original CPZ program has been modified so that it is a command line program rather than being an interactive one. This program `pz2` accepts output parameter file base name, i.e. OUT, output data stream base name and number outputs files. It generates a DAC data file for every parameter file. These DAC data files are merged into a single binary file and

logic to handle these has not yet been incorporated in the converter, present stress being on production of connected speech.

can be presented using program `present`. The merging program is called `merge` and is a command line program accepting the base name of files to be merged, name of merged file, and number of files.

## 5.3 Algorithm

We discuss in brief the rules used for various parameters before discussing the program implementation.

### 5.3.1 The pitch contour

The F0 or the pitch frequency in an utterance determines intonation of any sentence. Overall shape of pitch contour is varied in accordance with the type of sentence, i.e., a declarative sentence will produce a different pitch contour from an interrogative sentence. Incorporation of such logic into program requires that the meaning of a sentence be evaluated beforehand or atleast the location of accents be known. If we place markers to indicate the kind of sentence, we would need a 2-pass parsing to evaluate and then reshape the overall pitch contour. This aspect is important for correct perception but emphasis here has been on straight synthesis of utterances. F0 is linearly interpolated between the endpoints.

The phoneme symbols occupy space beyond decimal 127 in the the code, hence the entire set of codes used by the English alphabets is available to denote pitch markersand others prosodic markers.

### 5.3.2 The rules for amplitude tracks

Amplitude tracks govern the strengths of various excitation sources that represent the glottal input (frication, aspiration, voicing and quasi-sinusoidal voicing). A linear interpolation scheme works well with these tracks.

### 5.3.3 Rules For formant tracks

A simple scheme of linearly interpolating between initial and final values works well for amplitude tracks but does not work well with natural formants and a smooth interpolation is required to avoid slope discontinuities on phoneme boundaries (O'Shaughnessy, 1987). The exponential curves provide a very good fit to formant "motion" (with respect to time) of natural speech. The motion from initial target value Ai towards target value At can be described by the equation

$$a(t) = A_i + [V_i t + (A_i - A_t)(1 + t/\tau)]e^{(-t/\tau)} \qquad 5.1$$

where $V_i$ represents an initial parameter velocity, i.e. the slope of parameter at $t = 0$, (which in general would be non zero if parameter is not steady at $t = 0$) and $\tau$ is a time constant reflecting the duration of the transition. $\tau$ is typically a function of both the phonemes involved in the transition, it reflects the mass and the speed of articulation employed to produce the phonemes (Rabiner, 1968).

The time constant is also an indicator of speed of articulation and degree of difficulty of articulation. The data for time constant is compiled by performing extensive analysis on the transitional segments. It is found that the time constants are approximately the same for different members of the same group. Similar is the case with transition duration. Both time constant and transition duration are available as look-up data from one group to different groups.

### 5.3.4 Rules for bandwidth transitions

Bandwidth transitions are roughly linear and hence linear transitions are used. Similarly for both zeroes and their respective bandwidths the linear transitions seem to work.

### 5.3.5 Importance of duration in the synthesis

In natural speaking one always finds that the durational difference in an utterance can make a considerable difference in perception. The look up tables storing the parameters pertaining to a particular phoneme will have a duration associated with that particular phoneme. This duration will in general vary, depending on the context, for a natural speaker. For good synthesis it is therefore essential to vary the 'inherent' duration associated with the phoneme depending on the context. Appropriate pauses should be inserted between the words, phrases etc. Usually the speech pauses are different from the writing pauses, which are essentially intended to make reading easier but are not necessarily inserted during the actual utterance. Normally the pauses are inserted at the phrase boundaries. The segment duration is dependent on a large number of phonetic and structural constraints. They vary considerably depending on sex of the speaker, narration style (reading / conversational) and speaking rate. Presently we shall not worry about the subtleties introduced in the duration due to suprasegmental information about the emotional state of the speaker (sorrow, fear, excitement, etc.). The rate of speaking varies the pause length. We consider the following observations made ([14])

1. Positional Effect: A phoneme is more lengthened in a word final position than in a word beginning position, which in turn is longer than word medial position. The duration of a phoneme before a pause is increased. The pause may be due to a phrase boundary or a 'breath group' or a sentence ending. The breath group is feature to give synthesized voice a characteristic resulting from the finite lung volume of the speaker. This feature inserts pauses after a pre determined time to

make the speaker sound realistic while speaking long sentences. Duration of pauses (interword, intersentence, etc.) depend on their position in then input text.

2. Post Vocalic consonant effect : This effect states that the duration of the vowel changes depending upon the type consonant (post vocalic consonant or PVC) following it. The voicing aspiration, sonority and nasality of the post vocalic consonant, all affect the duration of the preceding vowel.

3. Place of articulation effect : If two adjacent characters (within as well as across the word boundaries) have same place of articulation (POA) then one or both the characters are shortened. This implies that if the parameter tracks do not change very much then duration of one of them or both may be shortened. This is attributed to relative ease of pronouncing the sequences of speech sounds with same POA.

4. Changes in cluster environments : In case of the cluster characters (CCV or CCCV), the duration of the various constituent basic units change due to presence of adjacent consonants. They may shorten due to cause (3) or sometimes lengthen due to difficulty in pronouncing certain sequences of consonants with conflicting articulatory requirements.

5. Polysyllabic shortening effect : If the number of characters in a word is greater than three then vocalic durations of various characters is reduced. This effect may relate to the efficiency in the communication, i.e. words with many units are much easier to identify than short words, hence speaker may spend less time per unit without the risk of errors in perception.

Additionally the changes in the duration also occur due to stress on some word which will be expressed as stress symbols preceding the word. Depending on the context one may decide to undershoot or overshoot the target by changing the target value itself.

### 5.3.6 Deciding the duration of the consonant

The stored phones have been targetted at /A/, so when following vowel is other than /A/ the vowel and the transition part needs to be appropriately truncated. The amount of truncation is specified as percentage of total duration. The truncation percentage is same for phones belonging to same group. Appendix C lists the percentage of phoneme to be truncated. The truncation ensures setting up of proper initial values for the synthesis.

### 5.3.7 Implementation

The program has been implemented on an IBM compatible PC-AT 386 machine. The program assumes a parsed textfile which has syllable boundary markers. The parsed textfile name is assumed to be `text.dat`. The program also needs some reference databases for

Table 5.1: Grouping of Phonemes

| Group | Group no. |
|---|---|
| None | 0 |
| Vowels | 1 |
| Nasals | 2 |
| Glides | 3 |
| Liquids | 4 |
| Stops | 5 |
| Fricatives | 6 |
| Affricates | 7 |

identification of groups and information related to transition and time constant information. The textfile `chargrp.dat` stores information about the grouping of the phoneme. The groups are enumerated in the order shown in Table 5.1.

This file just contains the group number on every line of the file. The line number corresponds to the ASCII code of that particular phoneme's symbol. The unused symbols are assigned code 0. This makes sure that any change in the editor or the codes or both can now be done simply by changing the character group information in textfile "chargrp.dat", without altering the executable code. The information about transition durations and time constants is stored in the file "ggparams.dat". This file saves these data for all possible permutations of groups. The information for any initial phoneme to a target phoneme is arranged into 3 main fields.

1. `tr` is an array of 39 integers and will store the transition periods for each of 39 parameters but all the parameters are not used.

2. `tau` is an array of 39 floating point numbers and will store the time constants for each of the variables.

3. `start_pct` is an integer storing the value indicating the percentage of initial phoneme from where the transition begins.

The transition data is in general the same for members belonging to the same group. The information from these files is read into arrays before the text file is opened for synthesis of parameter tracks. The fields of this database are read into a two dimensional array which can be indexed by the respective initial and target group indices. This data is used by the interpolating algorithm. After initialization, the mapping from phoneme symbol (also called as "grapheme") to the parameter tracks begins. The current phoneme and the target phoneme is read from the textfile and their corresponding data structures are updated with respective parameter files. The data structure has the following subfields.

1. ch: ASCII code of the character.

2. group: group number of that phoneme.

3. n_items: number of items in that phoneme.

4. dur: duration after which the parameters are updated.

5. vars: a two dimensional array of variables which is indexed by variable number (0 to 38) and the data item number (0 to n_items).

The stored phonemes have been targetted at /Λ/ and hence one has to clip the part involving the transition and the target vowel. This clipping of data is done by appropriately reassigning the data subfield "n_items" to a new value which is decided from group of the phoneme. The start of transition, the interval of transition of the time constant are available in array "curr_ggparams". The length of parameter tracks that is decided after this step is copied to output before the interpolation. After the interpolation, the tracks in which the coarticulation effect has been incorporated is copied to the output. The target phoneme tracks are now substituted into initial phoneme and next grapheme is brought in and continued. The syllable boundary markers are used to change the length of the phonemes according to the rules previously stated. The program listing is available separately [15].

The time constants and transition times stored in the file ggparams.dat are updated by a Modify-Group-to-Group-PARameters program called MGGPAR. This program will interactively update time constants, transition times and start percentages for any initial and target group.

## 5.4 Reduction Of data for algorithmic conversion

The success of our algorithm depends not only on the generation of accurate targets for the interpolation routines, but also on having precise duration in which to make the transition. We have attempted to normalize the values of transition across the members of a group. This normalization, i. e. assuming similarity of tracks in the transition region is not as gross as it may appear at first sight because the classification into groups is made on similarities in characteristics of the utterances. This step leads to a drastic reduction in data, from initial requirement of having transition times and time constants for all possible permutations of the entire phoneme set in that particular language, one now has to consider only all possible permutations amongst the groups. The target for any consonant is necessarily a vowel or a semivowel, hence only the combinations fromm all groups to vowels and semivowels should be considered. A provision has been made for alowing different transitory data for each of the 39 parameters. Thus a data field is available for each of the parameters. This field is not used by many paramters and has the space for 1 integer and 1 floating point number. Furthermore, it was also observed that starting of the transitions for phonemes in the same group was at the same locations for all the "observable" parameters [2]. On account of this, the start percentage field in transition look up table has only one field.

We now state the results of the analysis of transitory data of utterances that has been incorporated in to `ggparams.dat` data base.

## 5.5 Transition times and time constants

The formant transitions have been observed to be the prime factor in perception of the utterance. The converter lays emphasis on generation of the reasonably accurate tracks for the formants and the amplitude tracks, as they appear to play the most significant role in the perception. We have collected data for transitions from all groups to and from the vowels and the glides. A detailed table of time constants and start percentages is given in Appendix C.

The transition from one vowel to another vowel starts in the middle of the initial phoneme and continues till the middle of the target vowel. The transition starts in the middle of the phoneme and continues till the middle of the phoneme. The analysis of segment /a i/has shown that the transition duration is 180 ms of which 95 ms is in the /a/ part and remaining 85 ms in the /i/ part. The total duration is roughly 370 ms, which turns out to be the duration after retaining 50 % of the initial utterance. Hence we clip the initial and final 50 % of the phoneme to account for the reduction in the vowel durations. The generation of parameter tracks by algorithm compares well with the natural tracks The tracks of vowel utterence /a i/ shown in Fig. 5.1 smooth interpolation computed by

[2]Only a some of the parameters such as formant values and bandwidths have been observed

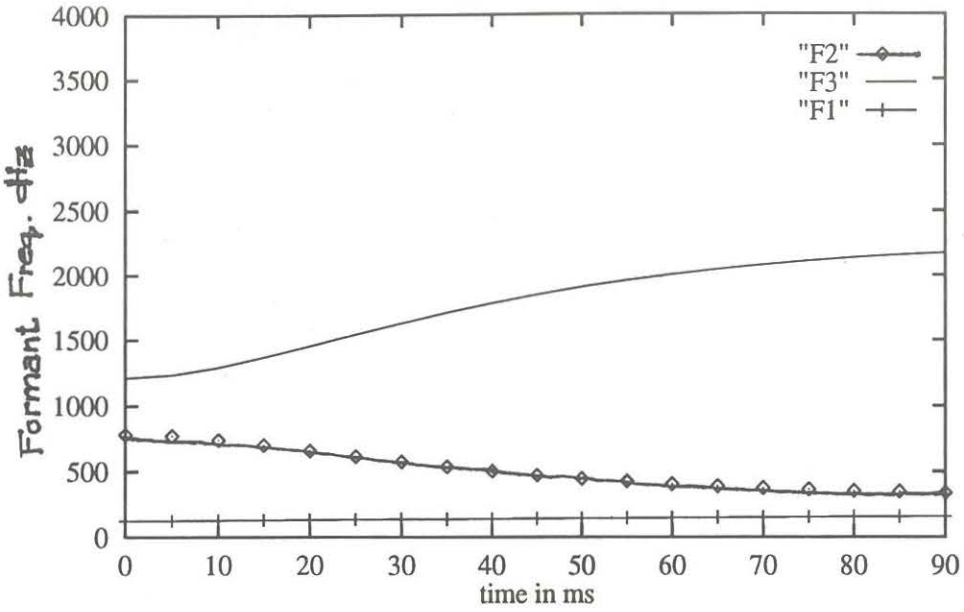



Figure 5.1: The transition track interpolated by the algorithm for /a i/

the TTS.

The transition from vowels to nasals begin at the middle of the vowels and the transition duration for all the formants is roughly 60 ms. The transition is only till the begining of the nasal phoneme. The value of the time constant for first three formants is 30 ms. The reduction in the intial phoneme end is 50 % but the length of the nasalised utterance remains constant. Hence we do not decrement the begining of the target phoneme. The vowel durations in these utterances can be directly clipped because vowels are a steady utterances and do not have a time varying parameter. The only time varying parameter is the pitch contour, but since we are not, at present much concerned with the intonation, we choose to ignore the effects caused by the clipping of the pitch contour.

The transition from the nasals to the vowels begins at roughly 80 % of the initial phoneme duration. The transition duration is of 75 ms and the time constant is 30 ms for all the formants. The transitions from the vowel to liquids start in the middle of the initial phoneme and continue till the beginning of the liquids. The transition duration for all the formants appear to be of 95 ms. The value of the time constant is 30 ms.The clipping of the target phoneme is not required, the initial phoneme however is is clipped to 50 % of its former value.

The transition from the liquids to the vowels begin at almost the end of the initial phoneme and continues well in to the middle of the target phoneme. The initial phoneme is trimmed by 80 % to free it from the effect of /a/. The initial values at this point give the correct values for the interpolation. The transition duration is 60 ms and the values of $\tau$ are 25, 30, and 9 ms respectively for formants F1, F2 and F3. The frictives

and affricates generated by the algorithm using the data in appendix C provided us with good results for both vowel consonant and consonant vowel sequences. The utterences are perceptually distinct.

# Chapter 6

# Summary and suggestions for further work

The text-to-speech converter using the cascaded pole-zero synthesizer using synthesis-by-rule has been developed. The synthesizer uses reference phonemes in a library for generation of initial and final target values. The reference phonemes have been synthesized by extensive hand tailoring on `partrc`, a graphical parameter track editor for the synthesizer parameter files. A presentation program, using interrupt driven approach has been developed for presenting the speech data of arbitrary durations.

## 6.1 Phonemes

The results on the phonemes have already been discussed in preceding chapter. The vowels are most easily recognizable utterances. The main factor which contributes to the naturalness of the vowels is the pitch contour or the F0 track. In case of vowels first two formants play an important role of differentiating it from other vowels. The experiments of changing only first two formant frequencies made the utterance to be perceived closer to the vowel to which the new formant data belonged to rather than the old utterance. The results on nasal /n̩/ is not very promising. It is not distinguishable from the utterance /n/. Other nasals like /$\eta$/, /m/, /$\tilde{n}$/ are clearly understandable. The liquids /r/ and /l/ too are of an acceptable quality. The affricates /dz/, /dz$^h$/ and /t$\int$/ are clear, but /t$\int^h$/ does not come out very distinct. The aspiration energy is gradually increased in order to reduce the 'glitch' but the utterance still seems to be closer to /t$\int$/ with heavy aspiration. In fact all the members of the fourth column (Table 4.1) showed these problems in listening tests. The stops such as /b/, /p/, etc. have a very good and distinct identity. The fricatives also are clearly distinguishable.

## 6.2 Presentation Program

The presentation program seems to work well with input and output directly from the disk, however the testing showed that some very minor timing faults occurred on the boundaries. The exact cause of this has not been pinpointed, however with so many factors to consider it seems that the disk accessing can cause problems sometimes depending on the fragmentation of disk memory space. However using huge memory model one can have a static data space as large as 1MB. In such a case the interrupt driven output can output at rates much faster than 10 kHz. The maximum data length using static data can be close to 10 seconds.

The program however works without any timing fault in case the output is done from a virtual disk, i.e., by assigning the RAM as a disk and accessing the file from this disk. The amount of errors in normal course of events does not actually preclude the use of this presentation program as it is. The program however has a small drawback as compared to the original presentation program. The old program used 16000 static memory locations for storing data. The program would resize the data to fit in the complete data window of the digital to analog converter. The new program does things on line and cannot possibly do the things as before by finding the scaling factor. The program was also tested at 20 kHz and from virtual disk it worked even at 20 kHz.

## 6.3 Text-to-speech converter

The text-to-speech converter for converting the input text to control parameter for cascade pole zero synthesizer has been developed. The ultimate success of this converter depends on the quality of the phoneme files that are used for conversion. The text-to-speech conversion was found to work very well with vowel utterances. The only problem with the utterances was that they seemed to have the accents at wrong points. This is because we do not manipulate the F0 contour at all, hence the accent points remain the same as they were for stand-alone utterances. The native listener then finds the utterance different. It is however difficult to control this contour without knowledge of the accent information.

Other utterances like those including fricatives also work well. Initial testing was carried out by testing the converter by synthesizing the consonant with different target vowel and targeting different vowels to an initial consonant. The intelligibility of the utterances remained consistently good. The affricates also gave anticipated results.

The nasalized utterances however created a lot of problems. They did not sound as good as they did without interpolation algorithm. This can be attributed to the errors in determining accurate transition width. It is also observed that the errors result in perception of utterances synthesized by the algorithm even though the original phonemes' result was good. This may be because of excessive difference in vowel to consonant energy in the utterance. One can overcome this problem in two ways. One simpler way is to

normalize the energies for the reference phonemes so that they come out 'just right' in context. The second method can be to normalize the excitation energies while synthesis by providing an energy control option. We have used the first method in our synthesizer.

## 6.4 Suggestions for further work

The quality of phonemes can be improved with better starting point for the synthesis. The synthesis quality will definitely improve with better tools for analysis. The presentation program can easily be modified for making presentations on two channels, which can be useful in studying the effects of partial deafness.

The text to speech program can be used to study prosody in a language. The converter can be further improved by having it analyze the prosody in language. Appropriate maskers can be inserted in order to make the converter insert appropriate accent information in synthesis. The converter only uses symbols of the ASCII code above decimal 127, hence the entire set of codes taken by the English alphabet is available for specifying the commands with prosodic information. The files produced by the converter are broken into parts and as such make it easier for the person to study the effect of the transitions. In fact with the development platform we have in the lab one can build an integrated environment which will 'compile' the speech from text directly. And additionally one can also control the parameter of the synthesizer directly. Such an integrated environment for analysis, synthesis and hearing would make a very good research tool.

# Bibliography


[1] H. Levitt , J. M. Pickette, and R. A. Houde, Eds., *Sensory Aids for the Hearing Impaired,* New York: IEEE Press, 1980.

[2] P. C. Pandey, *Speech Processing for Cochlear Prosthesis*, Ph.D. dissertation, Deptt. of Elec. Engg., University of Toronto, 1987.

[3] J. L. Flanagan, Ed., *Speech Analysis, Synthesis, and Perception,* New York: Springer Verlag, 1972.

[4] C. Coker,"A model of articulatory dynamics and control," *IEEE Proc.,* vol 64, pp 452-460, 1976.

[5] L. R. Rabiner and R. W. Shaffer, *Digital Processing of the Speech Signals,* Englewood Cliffs, NJ: Prentice Hall, 1978.

[6] D. H. Klatt, "A software for cascade/parallel formant synthesizer", J. Acoust. Soc. Am., vol 67(3), pp 971-995, 1980.

[7] S. A. Chafekar, *Speech Synthesis for Testing of Sensory Aids for the Hearing Impaired*, M.Tech dissertation, Deptt. of Electrical Engg., IIT Bombay, 1992.

[8] D. M. Kulkarni, *Development of a Cascade Pole-Zero Synthesizer*, M. Tech. dissertation, Deptt. of Electrical Engg., IIT Bombay, 1992.

[9] D. O'Shaughnessy, *Speech Communication : Human and Machine,* Reading, Massachussets: Addison Wesley, 1980.

[10] A. Sampath, *Speech Parameters for Formant Based Synthesis*, B. Tech. Project Report, Deptt. of Elec. Engg., IIT Bombay, 1991.

[11] T. G. Thomas, P. C. Pandey, and S. D. Agashe, "A PC-based spectrograph for speech and biomedical signals", *Proc. Int. Conf. on Recent Advances in Biomed. Eng.*, pp 7-10, 1994.

[12] *Dynalog Micro-Systems : User's Manual, PCL-208 Data Acquisition card*, Dynalog Micro-Systems: Bombay, 1989.

[13] *Microsoft MS-DOS User's Guide and Reference :ver 5.0*, Microsoft Corp., 1991.

[14] B. Yegnarayana et al., *Tutorial on Speech Technology, 6-8 December*, Dept. of Comp. Sci. and Engg., IIT Madras, 1992.

[15] N. N. Raut, *Program listings for A Text to Speech Converter*, accompanying M. Tech. thesis *A Text to Speech Converter using Cascaded Pole Zero Synthesizer*, Deptt. of Elec. Engg., I.I.T. Bombay, 1994.

[16] J. Allen, M. S. Hunnicut, and D H Klatt, *From Text to Speech : The MItalk System*, Cambridge University Press, 1987.

[17] L. R. Rabiner, "Speech Synthesis by Rule: an acoustic domain approach", *Bell Sys. Tech. J.*, vol 47, pp 17-37, 1968.

[18] S. Gracias, *A Speech Training Aid for the Hearing Impaired*, B. Tech. Project Report, Deptt. of Elec. Engg., IIT Bombay, 1991.

[19] E. J. Yannakoudakis and P. J. Hutton, *Speech Synthesis and Recognition Systems*, England: Ellis Horwood, 1987.

# Appendix A

# Phoneme Codes

Table A.1: Table of codes for the AKSHAR editor (DOE Keyboard)

| Akshar code | IPA | Devanagari |
|---|---|---|
| 166 | Λ | अ |
| 167 | a | आ |
| 168 | I | इ |
| 169 | i | ई |
| 170 | U | उ |
| 171 | u | ऊ |
| 178 | e | ए |
| 182 | o | ओ |
| 184 | k | क् |
| 185 | $k^h$ | ख् |
| 186 | g | ग् |
| 187 | $g^h$ | घ् |
| 188 | $\eta$ | ङ |
| 189 | $t\int$ | च् |
| 190 | $t\int^h$ | छ् |
| 191 | dz | ज् |
| 192 | $dz^h$ | झ् |
| 193 | $\tilde{n}$ | ञ् |
| 194 | $\dot{t}$ | ट् |
| 195 | $\dot{t}^h$ | ठ् |
| 196 | $\dot{d}$ | ड् |

Table of Codes (continued..)

| Akshar code | IPA | Devanagari |
|---|---|---|
| 198 | $\underset{\cdot}{\text{d}}^h$ | ढ् |
| 200 | $\underset{\cdot}{\text{n}}$ | ण् |
| 201 | $\underline{\text{t}}$ | त् |
| 202 | $\underline{\text{t}}^h$ | थ् |
| 203 | $\underline{\text{d}}$ | द् |
| 204 | $\underline{\text{d}}^h$ | ध् |
| 205 | n | न् |
| 207 | p | प् |
| 208 | $\text{p}^h$ | फ् |
| 209 | b | ब् |
| 210 | $\text{b}^h$ | भ् |
| 211 | m | म् |
| 212 | j | य् |
| 213 | r | र् |
| 215 | l | ल् |
| 218 | w | व् |
| 219 | ʃ | श् |
| 220 | $\underset{\cdot}{\text{ʃ}}$ | ष् |
| 221 | s | स् |
| 222 | h | ह् |

# Appendix B

# Transition table

Table B.1: Transition Data

| Initial Group | Target Group | F1 | | F2 | | F3 | | Start % | End Decr % | Begin Decr % |
|---|---|---|---|---|---|---|---|---|---|---|
| | | tr | tau | tr | tau | tr | tau | | | |
| Vowel | Vowel | | | | | | | | 50 | 10 |
| Vowel | Nasal | 60 | 30 | 60 | 30 | 60 | 50 | 50 | 50 | 0 |
| Nasal | Vowel | 60 | 30 | 60 | 30 | 60 | 70 | 80 | 50 | 0 |
| Vowel | Glide | 80 | 30 | 80 | 30 | 80 | 30 | 50 | 50 | 0 |
| Glide | Vowel | 80 | 45 | 80 | 45 | 80 | 45 | 100 | 90 | 50 |
| Vowel | Liquid | 95 | 30 | 95 | 30 | 95 | 30 | 50 | 50 | 0 |
| Liquid | Vowel | 60 | 25 | 60 | 30 | 60 | 90 | 100 | 80 | 50 |
| Vowel | Stop | 50 | 50 | 50 | 50 | 50 | 50 | 80 | 50 | 0 |
| Stop | Vowel | 30 | 30 | 30 | 30 | 30 | 30 | 25 | 85 | 0 |
| Vowel | Affricate | 60 | 80 | 60 | 70 | 60 | 80 | 70 | 50 | 0 |
| Affricate | Vowel | 60 | 50 | 60 | 30 | 60 | 30 | 100 | 80 | 0 |
| Vowel | Fricative | 95 | 25 | 95 | 50 | 95 | 80 | 50 | 50 | 0 |
| Fricative | Vowel | 90 | 30 | 90 | 65 | 90 | 90 | 100 | 70 | 0 |

# Appendix C

# Parameter data

**Note :** The units for duration is seconds and those for bandwidths and formant frequencies are Hertz.

## C.1 Vowels

## C.2 Nasals

## C.3 Glides

## C.4 Liquids

## C.5 Stops

## C.6 Affricates

## C.7 Fricatives

Table C.1: Vowels

| Phoneme | F1 | F2 | F3 | F4 | Duration |
|---|---|---|---|---|---|
| अ | 508 | 1133 | 1641 | 2305 | 365 |
| आ | 781 | 1211 | 2188 | 2734 | 365 |
| इ | 312 | 2266 | 3164 | 3750 | 340 |
| ई | 312 | 2266 | 3164 | 3750 | 345 |
| उ | 273 | 898 | 2266 | 3594 | 250 |
| ऊ | 391 | 2188 | 2734 | 3555 | 260 |
| ए | 480 | 1720 | 2600 | 3300 | 300 |
| ऐ | 371 | 977 | 2539 | 3242 | 365 |
| ओ | 700 | 1220 | 2600 | 3300 | 300 |

Table C.2: Nasals

| Phoneme | F1 | F2 | F3 | BW1 | BW2 | BW3 |
|---|---|---|---|---|---|---|
| ङ् | 410 | 500 | 1300 | 300 | 150 | 280 |
| न् | 200 | 1500 | 2500 | 300 | 70 | 160 |
| म् | 200 | 1180 | 2500 | 300 | 70 | 60 |

Table C.3: Glides

| Phoneme | F1 | | F2 | | F3 | | BW1 | BW2 | BW3 |
|---|---|---|---|---|---|---|---|---|---|
| | Start | End | Start | End | Start | End | | | |
| य् | 300 | 270 | 2400 | 1133 | 3200 | 1641 | 50 | 70 | 110 |
| व् | 273 | 508 | 898 | 1133 | 2266 | 1641 | 50 | 70 | 110 |

Table C.4: Liquids

| Phoneme | F1 | F2 | F3 | BW1 | BW2 | BW3 |
|---|---|---|---|---|---|---|
| र् | 350 | 1220 | 1400 | 50 | 70 | 110 |
| ल् | 340 | 1180 | 2520 | 50 | 70 | 110 |
| ळ् | 340 | 1180 | 2520 | 180 | 90 | 110 |

Table C.5: Stops-1

| Phoneme | F1 | F2 | F3 | BW1 | BW2 | BW3 | FZ1 | FZ2 |
|---|---|---|---|---|---|---|---|---|
| क् | 386 | 1387 | 2290 | 500 | 70 | 110 | 300 | 1990 |
| ख् | 386 | 1387 | 2290 | 50 | 70 | 110 | 300 | 1990 |
| ग् | 200 | 1630 | 2100 | 60 | 150 | 280 | 205 | 1990 |
| घ् | 200 | 1630 | 2100 | 60 | 150 | 280 | 205 | 1990 |

| Phoneme | FZ3 | FZ4 | FZ5 | BZ1 | BZ2 | BZ3 | BZ4 | BZ5 |
|---|---|---|---|---|---|---|---|---|
| क् | 3000 | 3671 | 4482 | 250 | 160 | 270 | 209 | 205 |
| ख् | 3000 | 3671 | 4482 | 250 | 160 | 270 | 209 | 205 |
| ग् | 3000 | 3671 | 4482 | 250 | 160 | 270 | 209 | 205 |
| घ् | 3000 | 3671 | 4482 | 250 | 160 | 270 | 209 | 205 |

Table C.6: Stops-2

| Phoneme | F1 | F2 | F3 | BW1 | BW2 | BW3 | FZ1 | FZ2 |
|---|---|---|---|---|---|---|---|---|
| ड् | 200 | 1700 | 2600 | 50 | 70 | 110 | 200 | 1700 |
| ठ् | 200 | 1700 | 2600 | 300 | 70 | 110 | 400 | 1600 |
| ड़् | 200 | 1600 | 2600 | 50 | 70 | 110 | 200 | 1700 |
| ढ़् | 200 | 1700 | 2800 | 50 | 70 | 110 | 200 | 1700 |

| Phoneme | FZ3 | FZ4 | FZ5 | BZ1 | BZ2 | BZ3 | BZ4 | BZ5 |
|---|---|---|---|---|---|---|---|---|
| ड् | 2800 | 4000 | 4900 | 200 | 200 | 200 | 65 | 100 |
| ठ् | 2540 | 2627 | 3363 | 300 | 120 | 256 | 266 | 487 |
| ड़् | 2800 | 4000 | 4900 | 200 | 200 | 200 | 65 | 100 |
| ढ़् | 2800 | 4000 | 4900 | 200 | 200 | 200 | 65 | 100 |

Table C.7: Stops-3

| Phoneme | F1 | F2 | F3 | BW1 | BW2 | BW3 | FZ1 | FZ2 |
|---|---|---|---|---|---|---|---|---|
| त् | 200 | 1700 | 2800 | 300 | 120 | 250 | 400 | 1600 |
| थ् | 386 | 1387 | 2290 | 50 | 70 | 110 | 300 | 1990 |
| द् | 200 | 1600 | 2600 | 60 | 100 | 170 | 200 | 1600 |
| ध् | 200 | 1600 | 2600 | 50 | 70 | 170 | 200 | 1600 |

| Phoneme | FZ3 | FZ4 | FZ5 | BZ1 | BZ2 | BZ3 | BZ4 | BZ5 |
|---|---|---|---|---|---|---|---|---|
| त् | 2540 | 2627 | 3363 | 300 | 120 | 256 | 266 | 487 |
| थ् | 3000 | 3671 | 4482 | 250 | 160 | 270 | 209 | 205 |
| द् | 2693 | 3496 | 4232 | 60 | 100 | 185 | 250 | 754 |
| ध् | 2693 | 3496 | 4232 | 60 | 100 | 185 | 250 | 754 |

Table C.8: Stops-4

| Phoneme | F1 | F2 | F3 | BW1 | BW2 | BW3 | FZ1 | FZ2 |
|---|---|---|---|---|---|---|---|---|
| प् | 150 | 1100 | 2150 | 200 | 40 | 40 | 200 | 1600 |
| फ् | 340 | 1110 | 2080 | 200 | 120 | 150 | 500 | 1875 |
| ब् | 210 | 1775 | 2800 | 60 | 80 | 270 | 260 | 1800 |
| भ् | 210 | 1775 | 2800 | 60 | 80 | 270 | 260 | 1800 |

| Phoneme | FZ3 | FZ4 | FZ5 | BZ1 | BZ2 | BZ3 | BZ4 | BZ5 |
|---|---|---|---|---|---|---|---|---|
| प् | 2693 | 3496 | 4232 | 60 | 100 | 185 | 250 | 754 |
| फ् | 3000 | 3750 | 4900 | 184 | 144 | 250 | 240 | 100 |
| ब् | 2882 | 3672 | 4420 | 60 | 80 | 268 | 215 | 896 |
| भ् | 2882 | 3672 | 4420 | 60 | 80 | 268 | 215 | 896 |

Table C.9: Affricates

| Phoneme | F1 | F2 | F3 | BW1 | BW2 | BW3 | FZ1 | FZ2 |
|---|---|---|---|---|---|---|---|---|
| च् | 300 | 1775 | 2800 | 200 | 90 | 300 | 350 | 1800 |
| छ् | 300 | 1775 | 2800 | 200 | 90 | 300 | 350 | 1800 |
| ज् | 210 | 1775 | 2800 | 60 | 80 | 270 | 260 | 1800 |
| झ् | 150 | 1700 | 2530 | 70 | 60 | 100 | 260 | 1400 |

| Phoneme | FZ3 | FZ4 | FZ5 | BZ1 | BZ2 | BZ3 | BZ4 | BZ5 |
|---|---|---|---|---|---|---|---|---|
| च् | 2882 | 3674 | 4420 | 200 | 90 | 295 | 215 | 895 |
| छ् | 2882 | 3674 | 4420 | 200 | 90 | 295 | 215 | 895 |
| ज् | 2882 | 3672 | 4420 | 60 | 80 | 268 | 215 | 896 |
| झ् | 2536 | 3300 | 3750 | 70 | 64 | 183 | 250 | 200 |

Table C.10: Fricatives

| Phoneme | F1 | F2 | F3 | BW1 | BW2 | BW3 | FZ1 | FZ2 |
|---|---|---|---|---|---|---|---|---|
| स् | 320 | 1390 | 2530 | 200 | 80 | 200 | 320 | 1390 |
| श् | 300 | 1840 | 2750 | 50 | 70 | 110 | 300 | 1840 |
| ह् | 508 | 1133 | 1641 | 300 | 70 | 160 | - | - |

| Phoneme | FZ3 | FZ4 | BWZ1 | BWZ2 | BWZ3 | BWZ4 |
|---|---|---|---|---|---|---|
| स् | 2531 | 3300 | 200 | 84 | 203 | 250 |
| श् | 3000 | 3652 | 200 | 100 | 260 | 218 |
| ह् | - | - | - | - | - | - |

# Appendix D

# Example of affricate tracks

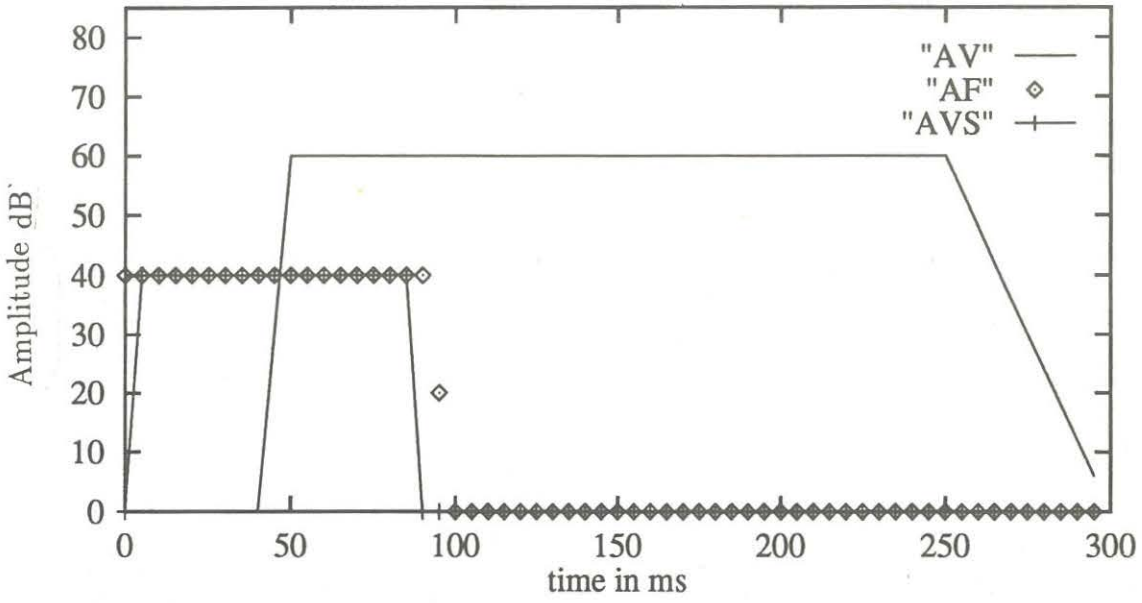



Figure D.1: The amplitude tracks of the affricate /t ʃ/

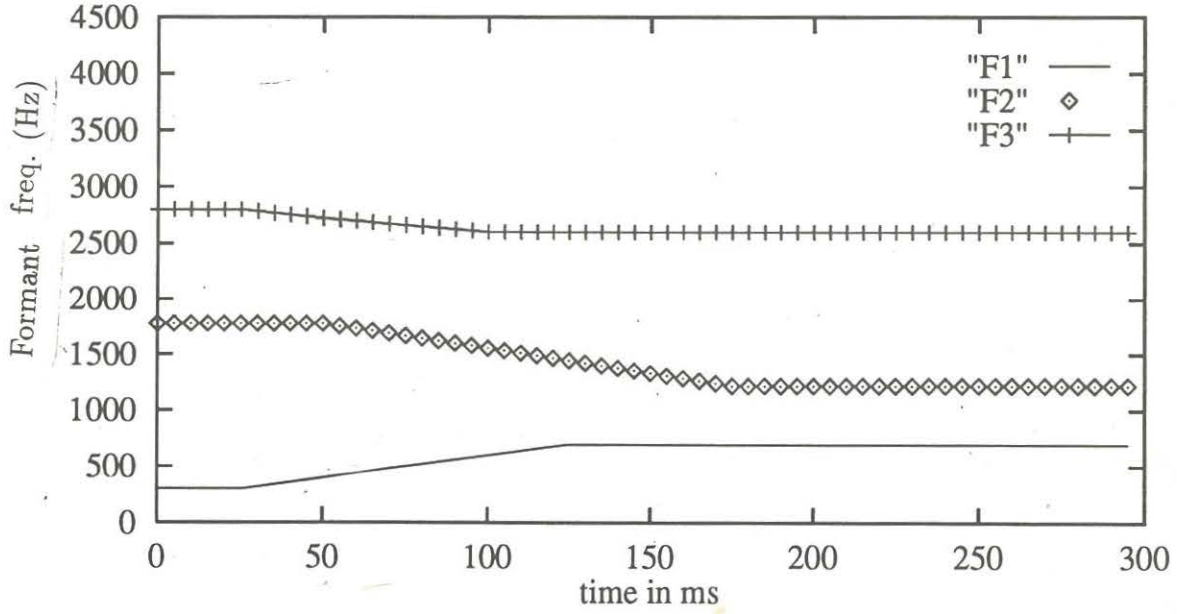



Figure D.2: The formant tracks of the affricate /t ʃ/

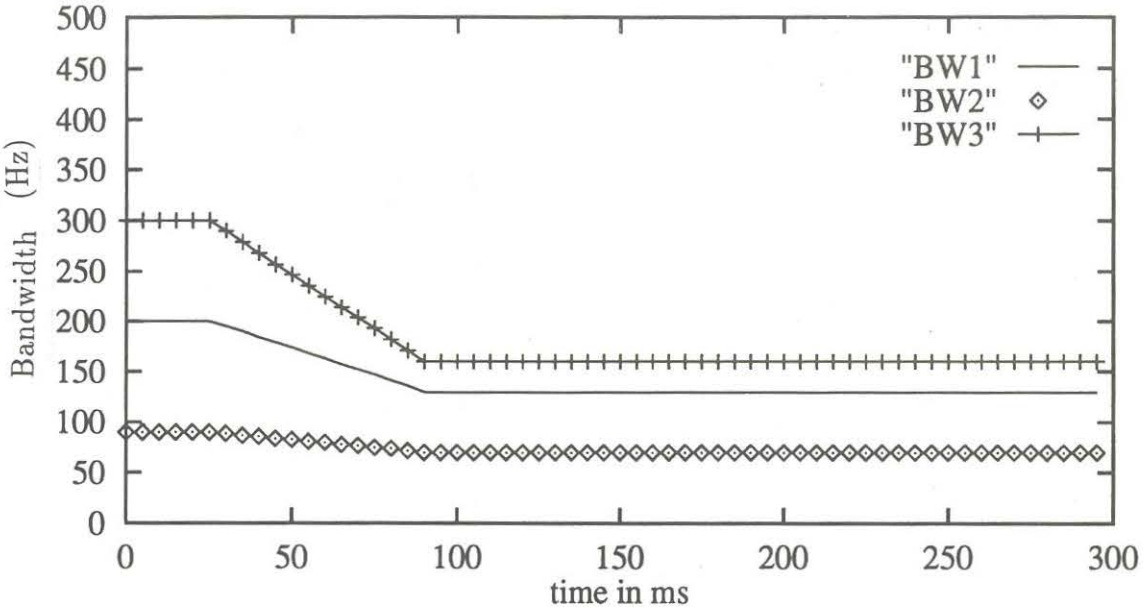



Figure D.3: The bandwidth tracks of the affricate /t ʃ/