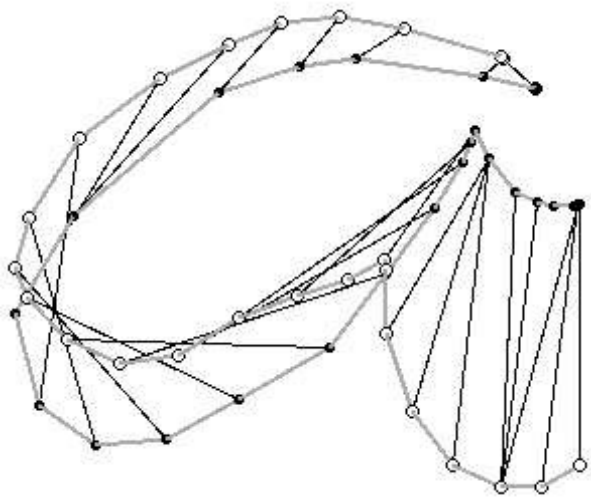
A Speaker Dependent Speech Recognizer for Isolated Hindi Digits Using Dynamic Time Warping

A Project Report



Arun Muralidharan

Ashutosh Modi

Shubhendu Trivedi

Anup Belsare

Aniruddha Bagool

T.E E&TC
Modern College of Engineering, Pune

I think that to keep trying new solutions is the best way to do everything.

-- Richard Feynman

Acknowledgements

The completion of this project work gives much satisfaction and fulfillment. The project could only see the light of the day with the gracious help of some very important people.

We are extremely grateful to **Prof Mrs Kanitkar** for the enormous encouragement and guidance and also making laboratories and facilities available in the face of power cuts.

We would also like to thank our Head of Department **Prof Dr (Mrs) K.R Joshi** and all other staff members for their kind support and guidance.

Also, we would like to thank **Mr Sandeep Kshirsagar** and **Mr Sumedh Kulkarni** for giving expert guidance and sharing hands on experience in developing such a project. The project would have been impossible had it not been for the duo we mentioned above. We are extremely grateful to them for giving valuable suggestions and also giving their valuable time.

Shubhendu Trivedi
Ashutosh Modi
Arun Muralidharan
Anup Belsare
Aniruddha Bagool

Contents

Project Abstract

A novel method for recognition of isolated spoken words on an 8-bit microprocessor is used. The method uses a less used but simple feature vector based on the zero-crossings of the speech signal. The feature vector is the histogram of the time-interval between successive zero-crossings of the speech signal. Dynamic time warping is used to calculate a time-aligned normalized distance between the feature vector and the reference templates.
The implementation needs only 1-bit A/D conversion and performs all its computations in integer arithmetic. A speaker-dependent recognition accuracy of 70% is obtained for Hindi digits spoken by 1 male speaker.

Speech being a natural mode of communication for humans can provide a convenient interface to control devices. Some of the speech recognition applications require speaker-dependent isolated word recognition. Current implementations of speech recognizers have been done for personal computers and digital signal processors. However, some applications, which require a low-cost portable speech interface, cannot use a personal computer or digital signal processor based implementation on account of cost.

The implementation of a speech recognizer on a fixed-point microprocessor could provide a possible solution to such applications. Standard algorithms based on hidden markov models (HMM) and artificial neural networks (ANN) cannot be used be on a fixed-point microprocessor because these algorithms require computation which cannot be done in real-time on an 8-bit microprocessor. Hence, there is a need for a simpler algorithm.
We have implemented a new and less used algorithm that uses only integer arithmetic and, hence, can be efficiently implemented on a fixed-point microprocessor in real-time. The algorithm is used for performing speaker-dependent recognition of isolated Hindi digits. Speech recognition algorithms employ a short time feature vector to take care of the non-stationary nature of the speech signal. Standard feature vectors Mel frequency

cepstrum coefficient (MFCC) or linear prediction coefficient (LPC) are computationally intensive. We designed a new but simple feature vector that uses only the zero crossings of the speech signal. The novel feature extraction requires 1-bit A/D conversion because it processes only zero crossings. The feature extraction is computationally very simple. It does not require any pre-processing of the speech signal. The feature vector preserves all information regarding the duration of the time intervals.

The short time feature vector is the histogram of time-interval between successive zero crossings of the speech utterance in a short time window. The feature vectors, extracted for each window in the speech utterance, are combined to form a feature matrix. The matrix is then normalized by multiplication with a weight vector. Dynamic time warping (DTW) is used to calculate the distance between the feature matrix of the input signal and the reference patterns. DTW finds a best time-aligned path for minimum distance under certain specified constraints. The pattern corresponding to the minimum distance is then identified to be the unknown input signal if the distance is less than a predetermined threshold.

The algorithm was implemented on Matlab to perform speaker dependent isolated word recognition on a vocabulary of 10 isolated Hindi digits. Simulation of the algorithm with recorded utterances gave an accuracy of 70%.

The algorithm was then implemented on 8051, an 8-bit microcontroller.

Chapter 1

Introduction

Speech recognition is the process of converting an acoustic signal, captured by a microphone or a telephone, to a set of words. The recognized words can be the final results, as for applications such as commands & control, data entry, and document preparation. They can also serve as the input to further linguistic processing in order to achieve speech understanding. There have been many science fiction movies and stories involving intelligent machines that can recognize speech and obey commands accordingly. Despite of all the attached glamour with speech recognition, it is a field with little luck so far. An intelligent recognizer has not yet nee designed. The reason for this is the interdisciplinary nature of speech recognition research. This project is a study in one of the domains in speech recognition i.e. of pattern recognition.

This project deals with the implementation of a speech recognizer both in Hardware (8051 based) and Software (MATLAB based) which may be used for simple machine control applications.

1.1 Problems with Speech Recognition Research

One of the most difficult aspects about speech recognition research is its interdisciplinary nature. A monolithic approach can not be applied to individual problems. The various disciplines that have been applied to various speech recognition problems are as follows.

1. Signal Processing: Extracting the information from the speech signal in an efficient and robust manner. Also it deals with the pre-processing part.

2. Physics (Acoustics): The science of understanding the physiological mechanisms and them producing different types of sounds.

3. Pattern Recognition: The set of algorithms used to cluster data and and to match a pair of patterns.

4. Communication and Information Theory: The set of modern coding and decoding algorithms (e.g dynamic programming, Viterbi algorithm)used to search a large but finite grid for a best possible path corresponding to the best recognized sequence.

5. Linguistics: The relationship between sounds (Phonology), words in a language (Syntax), meaning of spoken words (semantics), and sense derived from meaning (pragmatics). Also included in this domain is the idea of parsing and of grammar.

6. Physiology: Understanding of some higher order mechanisms in the human CNS (Central Nervous System) and that account for the speech production and perception process.

7. Computer Science: The efficient way to implement in software the algorithms so chosen for speech recognition.

8. Psychology: The science that enable technology to be used by humans for a specific task.

Thus the designing of a very robust recognizer requires a very inter-disciplinary approach, which makes research all more difficult.

1.2 Approaches to Automatic Speech Recognition

Broadly speaking there are three main approaches to research in speech recognition.
1. The Acoustic Phonetic Approach.

2. The Pattern Recognition Approach.
3. The Artificial Intelligence Approach.

The Acoustic phonetic approach is based on the theory of acoustic phonetics that postulates that there exist finite, distinctive phonetic units in spoken language and that these units may be characterized by a set of distinct properties.

The pattern recognition approach has been dealt with in some detail in the literature survey.

The AI approach focuses on the use of HMM and ANN.

1.3 Classification of Recognizers

Speech recognition systems can be characterized by many parameters and can be classified accordingly.
They can be classified according to the speaking mode, which is of two types: Isolated and Continuous. An isolated-word speech recognition system requires that the speaker pause briefly between words, whereas a continuous speech recognition system does not have that requirement. One more way of classifying the recognizers is speaking style. There can be a huge difference in the recognition if the speech to be recognized is generated impromptu than a sample that is read out. Yet another way is on the basis that some systems require speaker enrollment i.e. a user must provide samples of his or her speech before using them, whereas other systems are said to be speaker-independent, in that no enrollment is necessary. Recognizers are also classified as according to the vocabulary size. The above discussion may be summed up in table 1.1 below.

Table 1.1 Classification of Recognizers

Parameters	Range
Speaking Mode	Isolated words to continuous speech
Speaking Style	Read Speech to Spontaneous Speech
Enrollment	Speaker Dependent to Speaker Independent
Vocabulary	Large(>20000 words) to Small (<20 words)
Language Model	Finite State to Context Sensitive

This project primarily deals with the implementation of an isolated word speech recognizer on the 8051 and MATLAB using a pattern recognition approach called Dynamic Time Warping.

Chapter 2

Literature Survey

2.1 The Speech Signal

The speech signal is a slowly time varying signal. That essentially means that if examined over a sufficiently short period of time (5 to 100 ms), its characteristics are fairly stationery. However over longer intervals of time the signal is noticeably non-stationery, the signal characteristics change to reflect the various speech sounds being spoken.

Given an utterance by a male speaker of the phrase “It’s Time..”. The following five waveforms represent 100ms each of the total 0.5 s of the utterance.

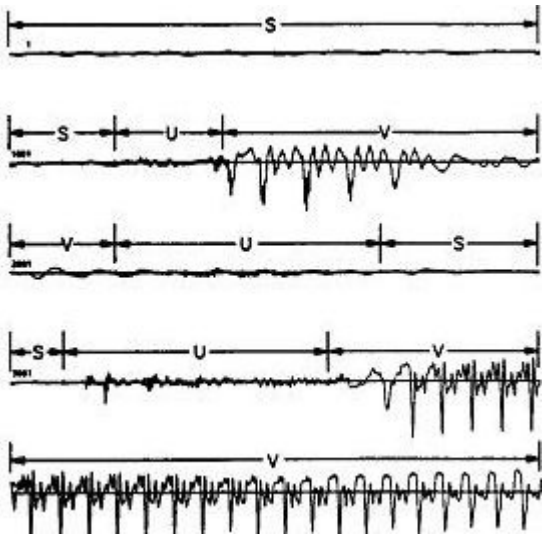


Figure 3.1 Waveform plot of utterance “It’s Time..”

There are several ways of classifying or labeling events in a speech utterance. The simplest is via the state of the speech production source, the vocal cords. It is a convention to use a three state representation. In which the states are 1. Silence (S), where no speech is produced. 2. Unvoiced (U), where the vocal cords are not vibrating,

hence the waveform produced is random or aperiodic ic nature. 3. Voiced (V), here the vocal cords are tensed and vibrate as air passes through them and therefore vibrate periodically. The resulting speech waveform is quasi-stationary in nature. In the figure 3.1 we see that the part before speaking actually begins is classified as Silence (S). A brief period of whispering or aspiration occurs before the actual utterance begins; this is called the unvoiced part. After that the actual voiced part (V) begins. Corresponding to the release of /t/ there is another unvoiced part.

Hence while considering the speech signal we consider a window length that is short enough for the signal to appear quasi stationary.

3.2 The Pattern Recognition Approach to Speech Recognition

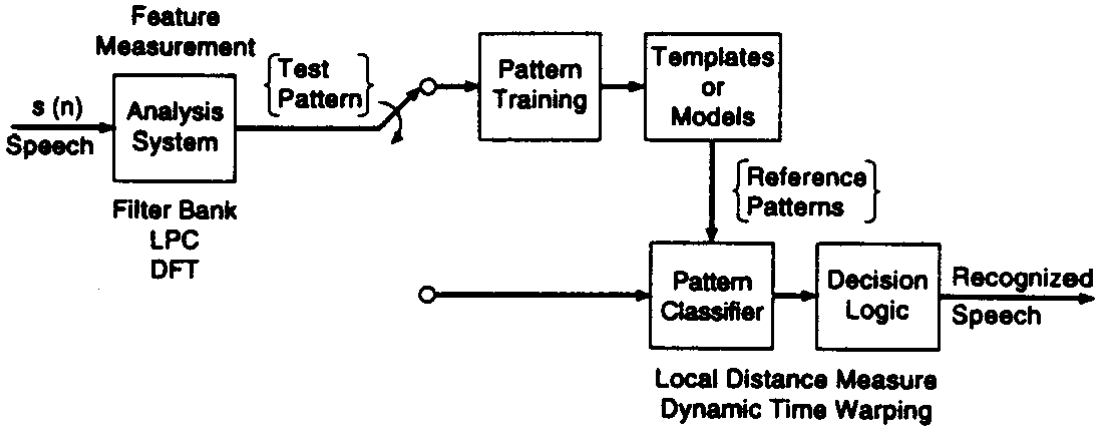


Fig 3.2 A basic Block Diagram of The Pattern Recognition Approach

The above is a very basic diagram for the approach to speech recognition using pattern recognition.

The pattern recognition paradigm has four basic steps:

1. Feature Measurement, in which a sequence of measurements is made on the input signal to define the “test course”. For speech signals the feature measurements are usually the output of some type of spectral analysis technique.

2. Pattern Training, in which one or more test patterns corresponding to the speech sounds of a particular class are used to create a pattern representative of the features of that class. The resulting pattern generally called a reference pattern, can also be called an exemplar or a template, derived from some kind of averaging technique.
3. Pattern Classification, in which the unknown test pattern is compared with each sound class reference pattern and a measure of similarity between the test pattern and each reference pattern, is computed. To compare speech patterns both a local distance measure can be used and a global alignment distance, which accounts for the difference in the rate of speaking.
4. Decision Logic, in which the reference pattern similarity scores are used to determine which reference pattern best matches the unknown test pattern.

The general advantages and the weaknesses of the pattern recognition paradigm are as:

1. The performance of the system is sensitive to the amount of training data available for creating the sound class reference patterns. Generally greater the training, better the performance of any task.
2. The reference patterns are sensitive to the speaking environment and the transmission characteristics of the medium used to create the speech signal.
3. This method is relatively insensitive to the choice of vocabulary, syntax etc.
4. The computational load is proportional to the number of references to be compared with. Thus this method is prohibitive for large number of references.
5. Since system is independent of sound class, the basic techniques can be applied to a wide variety of sound.

3.3 Distance Metrices

Each ‘n’ dimensional feature vector may be considered as a point in the ‘n’ dimensional vector space. Thus, a feature vector is mapped to a point in the n-dimensions. This mapping helps us to perceive the speech signal (represented by their feature vectors) as

high-dimensional points. The advantage of this representation is that one can now use different distance metrics for (i) finding similarity between two images and (ii) ordering a set of images based on their distances from a given image. This enables us to do a nearest neighbor search on a large database of images and retrieve a result set containing images that are closest matches to a user-specified query. It is evident that the speech signal and their ordering depend both on the feature extraction method as well as on the distance metric used.

Two commonly available distance metrics used in speech recognition are:

1. Manhattan Distance.
2. Euclidean Distance.

The Manhattan distance is also called the L_1 distance. If u =($x_{-1,}y_1$) and v=(x_2 y_2), then the Manhattan distance between them is given by MH(u,v)=$|x_1-x_2|$ + $|y_1-y_2|$

Instead of two dimensions if the point had *n* dimensions, such as a = (x_1, x_2, $x_{3,}$ $x_{4,}$…x_n) and b = (y_1, $y_{2,}$ $y_{3,}$ $y_{4,}$ … y_n). Then the Manhattan distance would be given by:

$$MH(a,b) = |x_1-y_1| + |x_2-y_2| + |x_3-y_3| + \ldots + |x_n-y_n|$$

For the same two points, the Euclidean distance will be given as:

$$EU(a,b) = \{(x_1-y_1)^2 + (x_2-y_2)^2 + (x_3-y_3)^2\ldots.\}^{1/2}$$

Out of the two distance matrices the Euclidean Distance measure is more robust as has been shown AK Majumdar *et.al*.

3.4 Use of Zero Crossings as an Effective Feature From Lipovac *et al.*

Following the study of a research paper by Lipovac and Sarajevo. We follow that zero crossings of a speech signal may be used as an effective feature for speech recognition.

Problem: Speech recognizers can be implemented using Hidden Markov Models or Artificial Neural Networks. There are plenty of such systems in place. However the problem with these algorithms is that they are computationally pretty intensive, and thus can not be implemented on a simple 8 bit fixed point micro-processor, and that is what we need for simple machine control applications. So there is a need for a simpler algorithm.

All these algorithms also employ a short term feature vector to take care of the non-stationary nature of speech. Generally the vector length is so chosen that the nature of the signal in this band is quasi-stationary. Feature vectors are an area of active research. Generally however at the university level, Mel Frequency Cepstrum Coefficients (MFCC) or Linear Predictive Coefficients are taken as features. These too require computations that are beyond the scope of a simple processor/ micro controller like the 8051.

Solution: We reviewed papers and were thinking what could be done to reduce this burden and choose a simpler feature so that it could be implemented on 8051. While researching on this we came across the lipovac paper which deals with this issue.

The researchers have used only zero crossings of the speech signal to determine the feature vector. Since this novel feature extraction method is based on zero crossings only, it just needs a one bit A to D conversion. This feature extraction is computationally very simple and does not require the speech signal to be pre-processed.

This feature vector is basically the histogram of the time interval between successive zero-crossings of the utterance in a short time window. These feature vectors for each window are then combined together to form a feature matrix. Since we are dealing with only small time series (isolated words), we can employ Dynamic Time Warping to compare the input matrix with the reference matrix' stored. To obtain this vector the following **steps** need to be followed.

1. The speech signal x(t) is band-pass filtered to give s(t).

2. s(t) is then subjected to infinite amplitude clipping with the help of a ZCD to give u(t).

3. u(t) is then sampled at say 8Khz to give u[n]. The feature extraction is carried out on u[n].

4. u[n] is divided in a number of short time windows for every one of the calculated W samples.

5. The histogram for each of this short time window is found. The histogram(or vector) is found as follows: The number of times ONLY ONE sample is recorded between successive zero crossings will constitute the element number 1 of the vector. The number of times ONLY TWO samples are recorded between successive zero crossings will constitute the element number two of the feature vector and so on. In this way we construct an histogram which is an appropriate feature vector.

These vectors then can be combined for all windows to get the feature matrix. These as i said earlier can be compared using DTW/DDTW/Fast DTW or some other algorithm.

The surface plot for one complete utterance by a male speaker for the matrix (where, as i have mentioned implicitly the rows represent the windows and the columns represent the histogram terms) prepared is as:

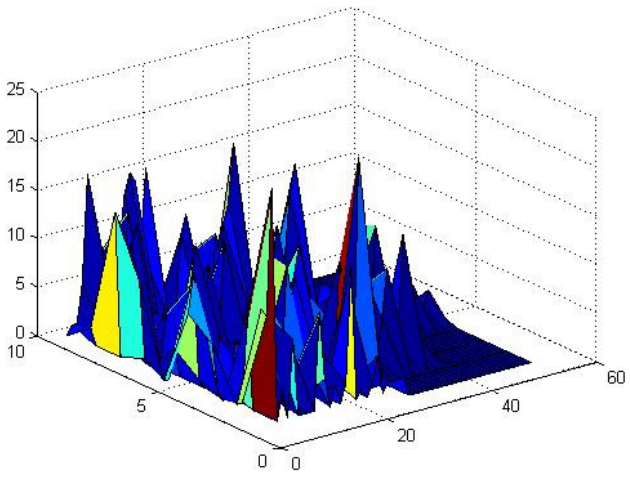


Fig 3.3 A surface plot for a single utterance by a male speaker

3.5 A Review of the Dynamic Time Warping Algorithm of Sakoe and Chiba *et al.*

Dynamic time warping (DTW) is a technique that finds the optimal alignment between two time series if one time series may be "warped" non-linearly by stretching or shrinking it along its time axis. This warping between two time series can then be used to find corresponding regions between the two time series or to determine the similarity between the two time series. Dynamic time warping is often used in speech recognition to determine if two waveforms represent the same spoken phrase. In a speech waveform, the duration of each spoken sound and the interval between sounds are permitted to vary, but the overall speech waveforms must be similar. In addition to speech recognition, dynamic time warping has also been found useful in many other disciplines [8], including data mining, gesture recognition, robotics, manufacturing, and medicine. Dynamic time warping is commonly used in data mining as a distance measure between time series. An example of how one time series is "warped" to another is shown in Figure 3.4.

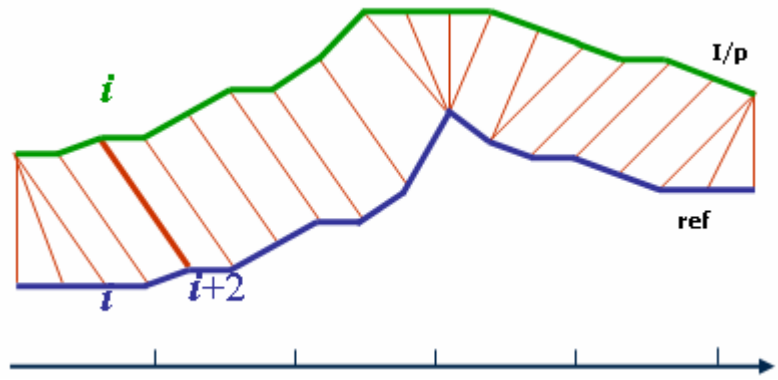


Figure 3.4

Each vertical line connects a point in one time series to its correspondingly similar point in the other time series. The lines actually have similar values on the *y*-axis but have been separated so the vertical lines between them can be viewed more easily. If both of the time series in Figure 1 were identical, all of the lines would be straight vertical lines because no warping would be necessary to 'line up' the two time series. The warp path distance is a measure of the difference between the two time series after they have been

warped together, which is measured by the sum of the distances between each pair of points connected by the vertical lines in Figure 3.4. Thus, two time series that are identical except for localized stretching of the time axis will have DTW distances of zero.

Despite the effectiveness of the dynamic time warping algorithm, it has an $O(N^2)$ time and space complexity that limits its usefulness to small time series containing no more than a few thousand data points.

A distance measurement between time series is needed to determine similarity between time series and for time series classification. Euclidean distance is an efficient distance measurement that can be used. The Euclidian distance between two time series is simply the sum of the squared distances from each *n*th point in one time series to the *n*th point in the other. The main disadvantage of using Euclidean distance for time series data is that its results are very unintuitive. If two time series are identical, but one is shifted slightly along the time axis, then Euclidean distance may consider them to be very different from each other. Dynamic time warping (DTW) was introduced [11] to overcome this limitation and give intuitive distance measurements between time series by ignoring both global and local shifts in the time dimension.

The dynamic time warping problem is stated as follows: Given two time series *X*, and *Y*, of lengths |*X*| and |*Y*|,

$X = x_1, x_2, x_3, \ldots x_n;$

$Y = y_1, y_2, y_3, \ldots y_n;$

construct a warp path *W*. Such that,

$W = w_1, w_2, w_3, \ldots w_k$

$MAX(|X|, |Y|) <= K <= |X| + |Y|$

Where k is the length of the warp path and the k^{th} element of the warp path maybe given as :

$W_k = (i,j)$

Where *i* is an index from time series *X*, and *j* is an index from time series *Y*. The warp path must start at the beginning of each time series at $w1 = (1, 1)$ and finish at the end of both time series at $wK = (|X|, |Y|)$. This ensures that every index of both time series is used in the warp path. There is also a constraint on the warp path that forces *i* and *j* to be monotonically increasing in the warp path, which is why the lines representing the warp path in Figure 1 do not overlap. Every index of each time series must be used. Stated more formally:

$W_k = (i,j)$ and $W_{k+!} = (i', j')$ where $i <= i' <= i + 1$ and $j <= j' <= j + 1$;

The optimal warp path is the warp path is the minimum-distance warp path, where the distance of a warp path *W* is
Dist (W) = Summation{Dist(w_{ki}, w_{kj})}.

Dist(*W*) is the distance (typically Euclidean distance) of warp path *W*, and *Dist*(*wki*, *wkj*) is the distance between the two data point indexes (one from *X* and one from *Y*) in the *k*th element of the warp path.

A dynamic programming approach is used to find this minimum-distance warp path. Instead of attempting to solve the entire problem all at once, solutions to sub-problems (portions of the time series) are found, and used to repeatedly find solutions to a slightly larger problem until the solution is found for the entire time series. A two-dimensional |*X*| by |*Y*| cost matrix *D*, is constructed where the value at *D*(*i*, *j*) is the minimumdistance warp path that can be constructed from the two time series *X*'=*x*1,...,*xi* and *Y*'=*y*1,...,*yj*. The value at *D*(|*X*|, |*Y*|) will contain the minimum-distance warp path between time series *X* and *Y*. Both axes of *D* represent time. The *x*-axis is the time of time series *X*, and the *y*-axis is the time of time series *Y*. Figure3.5 *D* shows an example of a cost matrix and a minimum-distance warp path traced through it from *D*(1, 1) to *D*(|*X*|, |*Y*|).

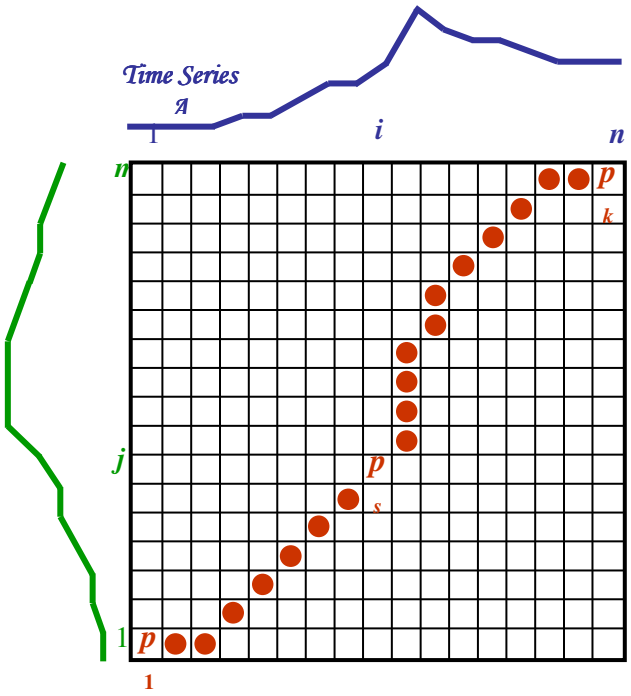


Fig 3.5 The DTW Alignment Path

The cost matrix and warp path in Figure 2 are for the same two time series shown in Figure 1. The warp path is {(1,1), (2,1), (3,1), (4,2), (5,3), (6,4), (7,5), (8,6), (9,7), (9,8), (9,9), (9,10), (10,11), (10,12), (11,13), (12,14), (13,15), (14,15), (15,15), (16,16)}. If the warp path passes through a cell $D(i, j)$ in the cost matrix, it means that the ith point in time series X is warped to the jth point in time series Y. Notice that where there are vertical sections of the warp path, a single point in time series X is warped to multiple points in time series Y, and the opposite is also true where the warp path is a horizontal line. Since a single point may map to multiple points in the other time series, the time series do not need to be of equal length. If X and Y were identical time series, the warp path through the matrix would be a straight diagonal line. To find the minimum-distance warp path, every cell of the cost matrix must be filled. The rationale behind using a dynamic programming approach to this problem is that since the value at $D(i, j)$ is the minimum warp distance of two time series of lengths i and j, if the minimum warp

distances are already known for all slightly smaller portions of that time series that are a single data point away from lengths i and j, then the value at $D(i, j)$ is the minimum distance of all possible warp paths for time series that are one data point smaller than i and j, plus the distance between the two points xi and yj. Since the warp past must either be incremented by one or stay the same along the i and j axes, the distances of the optimal warp paths one data point smaller than lengths i and j are contained in the matrix at $D(i\text{-}1, j)$, $D(i, j\text{-}1)$, and $D(i\text{-}1, j\text{-}1)$. So the value of a cell in the cost matrix is:

$D(i, j) = Dist(i, j) + \min[D(\tilde{i}\ 1, j), D(i, \tilde{j}\ 1),$

$D(\tilde{i}\ 1, \tilde{j}\ 1)]$

The warp path to $D(i, j)$ must pass through one of those three grid cells, and since the minimum possible warp path distance is already known for them, all that is needed is to simply add the distance of the current two points to the smallest one. Since this equation determines the value of a cell in the cost matrix by using the values in other cells, the order that they are evaluated in is very important. After the entire matrix is filled, a warp path must be found from $D(1, 1)$ to $D(|X|, |Y|)$. The warp path is actually calculated in reverse order starting at $D(|X|, |Y|)$. A greedy search is performed that evaluates cells to the left, down, and diagonally to the bottom-left. Whichever of these three adjacent cells has the smallest value is added to the beginning of the warp path found so far, and the search continues from that cell. The search stops when $D(1, 1)$ is reached.

Restrictions on the DTW Algorithm:

1. <u>*Monotonicity*</u>: $\boldsymbol{is\text{-}1} \leq \boldsymbol{is}$ and $\boldsymbol{js\text{-}1} \leq \boldsymbol{js}$.

The alignment path does not go back in "*time*" index.

Guarantees that features are not repeated in the alignment.

2. <u>*Continuity*</u>: $\boldsymbol{is} - \boldsymbol{is\text{-}1} \leq 1$ and $\boldsymbol{js} - \boldsymbol{js\text{-}1} \leq 1$.

The alignment path does not jump in "*time*" index.

Guarantees that the alignment does not omit important features.

3. <u>*Boundary Conditions*</u>: $\boldsymbol{i1} = 1, \boldsymbol{ik} = \boldsymbol{n}$ and $\boldsymbol{j1} = 1, \boldsymbol{jk} = \boldsymbol{m}$.

The alignment path starts at the bottom left and ends at the top right.

Guarantees that the alignment does not consider partially one of the sequences.

4. <u>*Warping Window*</u>: $|\boldsymbol{is} - \boldsymbol{js}| \leq \boldsymbol{r}$, where $\boldsymbol{r} > 0$ is the window length.

A good alignment path is unlikely to wander too far from the diagonal.

Guarantees that the alignment does not try to skip different features and gets stuck at similar features.

5. <u>*Slope Constraint*</u>: $(jsp - js0) / (isp - is0) \leq p$ and $(isq - is0) / (jsq - js0) \leq q$, where $q \geq 0$ is the number of steps in the x-direction and $p \geq 0$ is the number of steps in the y-direction. After q steps in x one must step in y and vice versa: $S = p / q \in [0, \infty]$.

The alignment path should not be too steep or too shallow.

Prevents that very short parts of the sequences are matched to very long ones.

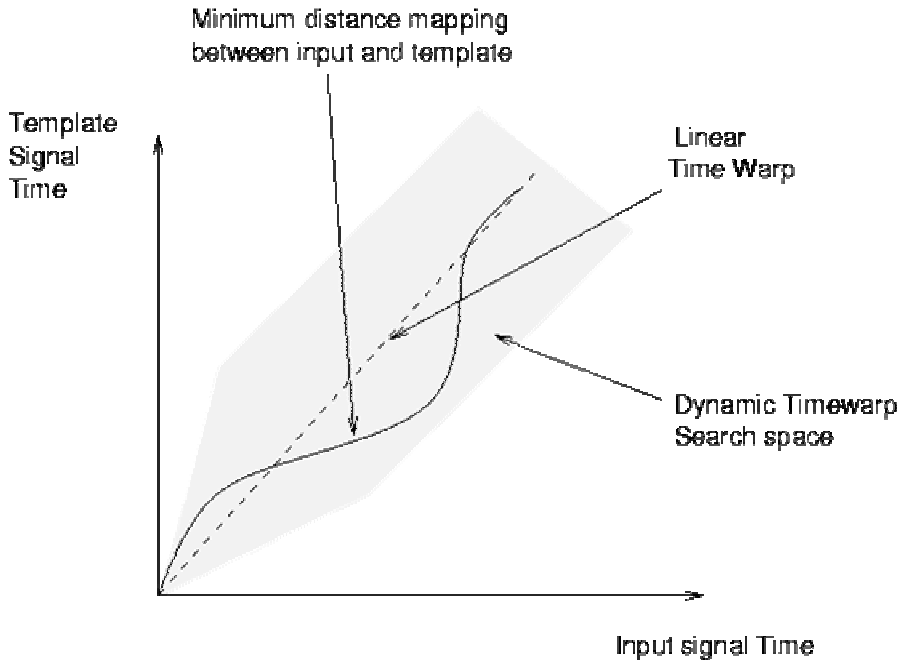


Fig 3.6 The DTW Function

Choice of the Weighting Coefficient:

There are two typical weighting coefficient definitions which enable this simplification. They are as follows.

1) Symmetric form:

$w(k) = (i(k) - i(k - 1)) + (j(k) - i(k - 1))$,

then

$N = I + J$,

where I and J are lengths of speech patterns A and B, respectively

2) Asymmetric form:

$w(k) = \boldsymbol{i(k)} - \boldsymbol{i(k} - 1))$, (1 **4)**

then , $\boldsymbol{N = I}$. (1 ***5*** *)*

(Or equivalently, $w(k) = (j (k) - j(k - 1))$, then $\boldsymbol{N = J.}$)

The basic concepts of the symmetric and asymmetric forms were originally defined by Sakoe and Chiba.

The DTW Algorithm at Work:

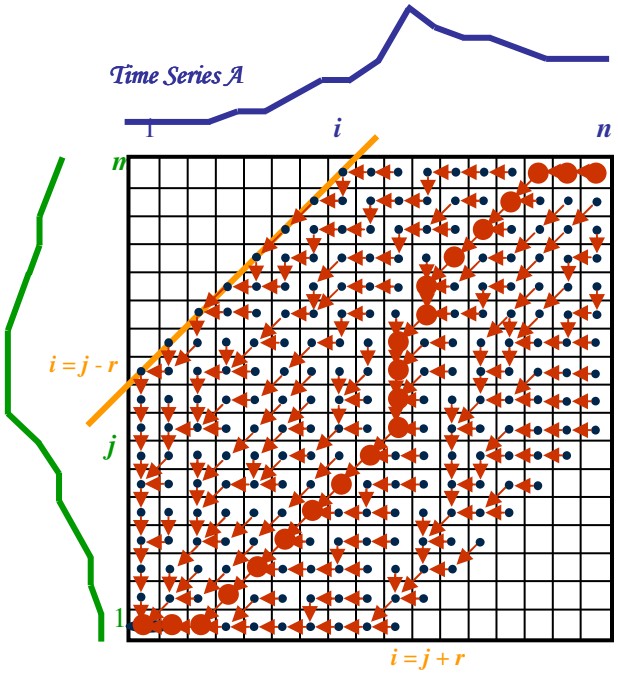


Fig 3.7 DTW at Work

1. Start with the calculation of g(1,1) = d(1,1).

2. Calculate the first row g(***i***, 1) = g(***i***–1, 1) + d(***i***, 1).

3. Calculate the first column g(1, ***j***) = g(1, ***j***) + d(1, ***j***).

4. Move to the second row g(***i***, 2) = min(g(***i***, 1), g(***i***–1, 1), g(***i*** – 1, 2)) + d(***i***, 2). Book keep for each cell the index of this neighboring cell, which contributes the minimum score (red arrows).

5. Carry on from left to right and from bottom to top with the rest of the grid g(***i***, ***j***) = min(g(***i***,***j***–1), g(***i***–1,***j***–1), g(***i*** – 1,***j***)) + d(***i***,***j***).

6. Trace back the best path through the grid starting from g(***n***, ***m***) and moving towards g(1,1) by following the arrows.

The Figure below gives an example of the local continuity constraints.

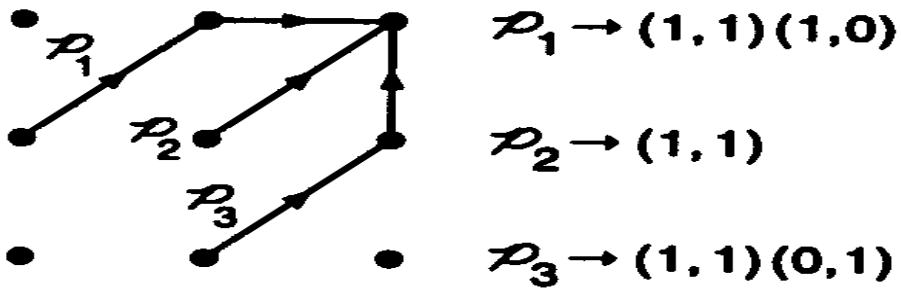


Fig 3.8 Local Continuity Expression

Conclusion:

This project involves use of use of zero crossings and their use as feature measures. Then, after the feature extraction process. The test utterance is tested against a reference template by the Dynamic Time Warping algorithm.

Chapter 3

Basic Block Diagram

The entire system may be condensed and represented in a simple block diagram:

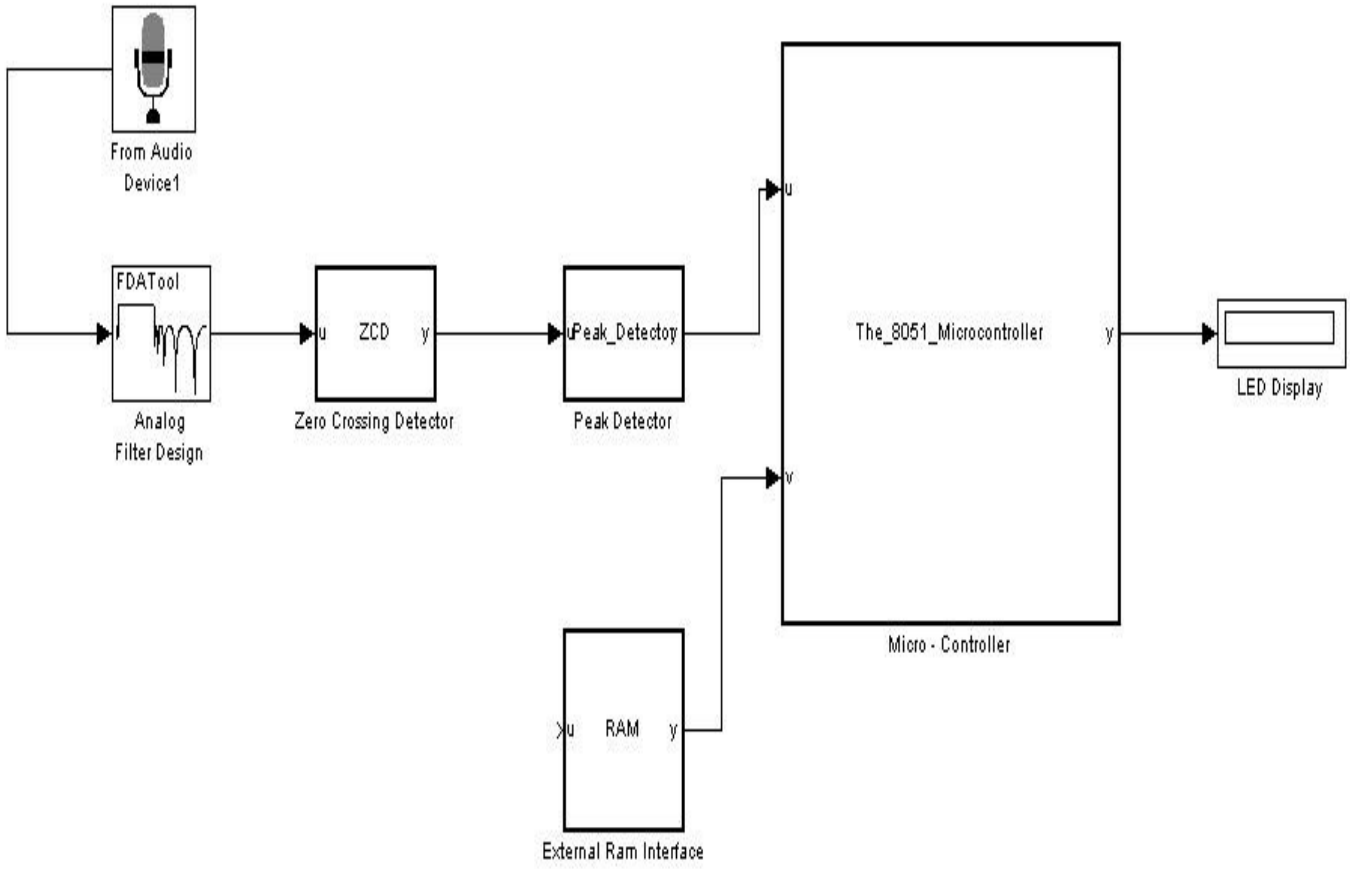


Fig 3.1: Complete elementary block diagram for the speech recognizer.

The system consists broadly of the blocks shown above. The basic blocks in the system are:

1. Audio Device and Signal Conditioning Circuit.
2. Band Pass Filter.
3. Zero Crossing Detector.

4. Peak Detector.
5. The 8051 development board with external RAM interfaced.
6. A display unit (LED/LCD).

The signal is input through the audio device, the signal is then band pass filtered. The band pass filtered signal is then subject to infinite amplitude clipping. This signal is then sampled at 8Khz (by software) to obtain the feature vector as already been discussed. The micro-controller does the processing and necessary comparison and then the output is put on a display unit. An external RAM is also interfaced with the micro-controller to store templates made by the above feature extraction procedure.

Chapter 4

Module Wise Description

4.1 Band-Pass Filter:

We have used Band Pass Filter to band limit the incoming audio signal in between 125 Hz to 3.4 KHz. Thus at the output of this stage we have eliminated the 50 Hz AC noise and the high frequency noise. This stage also does the signal conditioning. We have designed active band pass filters using operational amplifiers (LF -356).

The purpose of using band pass filters is:

1. To provide isolation between two stages. Therefore there is no chance of overloading of source, thus making cascading of stages simpler.
2. Flexibility in adjustment of gain and frequency.
3. Less noise at the output as compared to passive filter followed by an amplifier.

The response for a band pass filter of order 4 and of order 20 is given below. The order 4 filter is used in practical hardware implementation and the order 20 filter used in MATLAB as a "proof-for concept" simulation for the system.

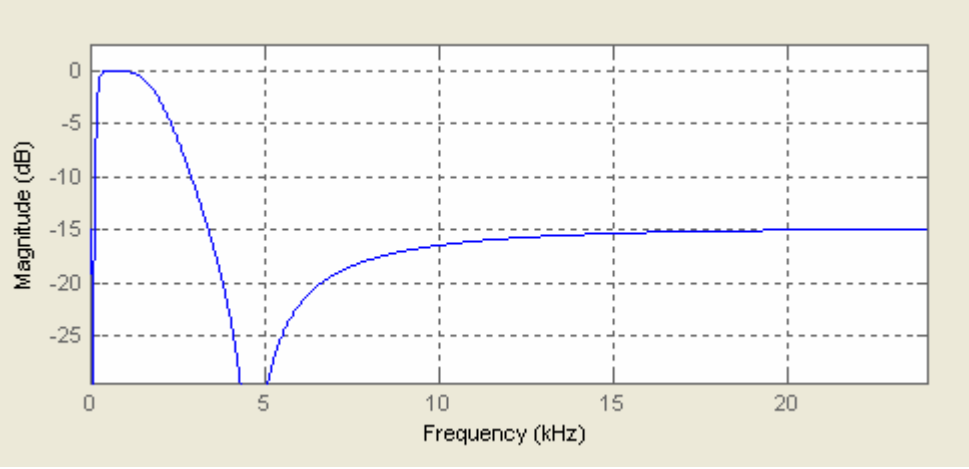


Fig 4.1: Frequency response of a 4th order band pass filter (125 Hz to 3.4 KHz)

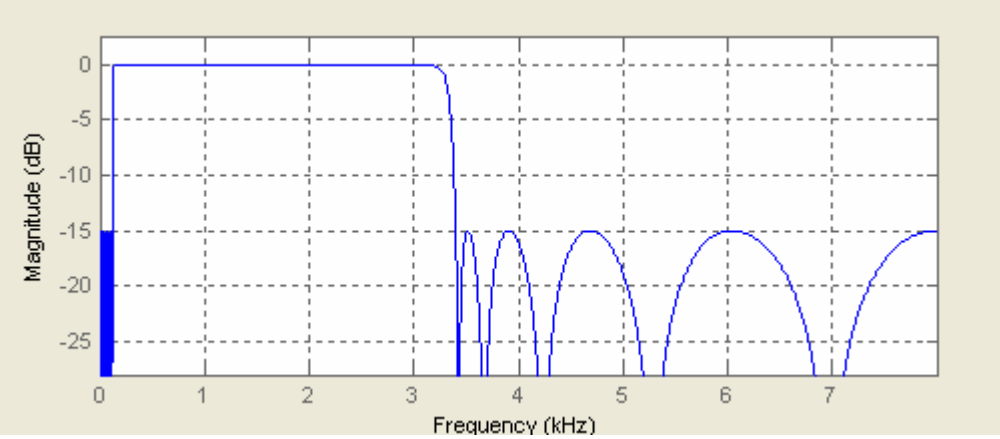


Fig 4.2 Frequency Response of a 20th order band pass filter (125 Hz to 3.4 KHz)

In practical implementation an order 4 filter has been used. In the MATLAB simulation testing has been done using both the 4^{th} order and the 20^{th} order band pass filter to compare the difference.

4.2 Zero Crossing Detector

The main purpose of implementing zero crossing detector is for 1 bit Analog to Digital conversion of the input speech signal.

The ZCD has been implemented using OP-AMP (LM 311).A pull up resistor has been used at the output as the op-amp used is open collector. We have used an additional 5V voltage source so as to get output voltage either 0V or 5V depending on the input condition. For the comparator we have kept a 1.5 V reference so as to minimize noise at the input.

4.3 Micro-Controller Unit

The microcontroller unit does the following things:

1. Feature Extraction by means of a software program
2. Implementing the Dynamic Time Warping Algorithm
3. Displaying the Output after the necessary polling with the stored templates

An external RAM (8KB x 8) is interfaced for storing the templates.

4.5 Buffer

The IC (74LS245) is a bidirectional buffer IC. The sole purpose of using the buffer IC is to drive the LED's. A 330 ohm resistor is connected in series with LED for the purpose of current limiting.

Chapter 5

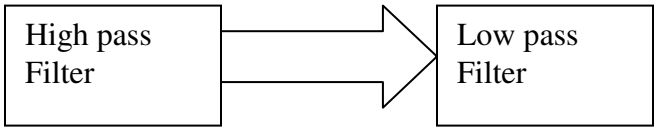
Design Steps

5.1 BANDPASS FILTER:

We have designed a fourth order Butterworth response bandpass filter with frequency range f_l=125Hz & f_h=3.4KHz.

The filter designed is an active filter using OP-AMP(LF 356).The purpose of using op amp is to introduce variable gain in the circuit & for inter stage isolation.

Now, the bandpass filter is designed using an high pass filter followed by an low pass filter.



To design a fourth order band pass filter we first designed a second order band pass filter & then cascaded it to another second order band pass filter.

The basic idea behind of such cascading is that on cascading the poles of the filter system gets added there by giving us the response of a fourth order filter.

HIGH PASS FILTER:-

A high pass filter is a circuit which passes only high frequency components above its cutoff frequency & attenuating all other frequencies lower than the
cutoff frequency.

The above described filter can be realized by a simple C-R circuit.

LOW PASS FILTER:

A low pass filter is a circuit which passes only low frequency components below its cutoff frequency & attenuates all other frequencies higher than the cutoff frequency.

The above described filter can be realized by a simple R-C circuit.

DESIGN CALCULATIONS:

The transfer function for a fourth order high pass filter is given as

H (jw) = $jw/w_c /[(s^2 + 0.765s + 1)(s^2 + 1.848s + 1)]$

And the transfer function for a fourth order low pass filter is given as

H (jw) = $1/[(s^2 + 0.765s + 1)(s^2 + 1.848s + 1)]$

FIRST STAGE:

1.<u>High pass filter</u>:

The expression for cutoff frequency is given by:

f= 1/(2*∏*R*C)

assume , **C=0.01µF**

Cutoff frequency, f=100Hz

We get ,

R= 160KΩ

Now, gain(A)=3-(1/Q)

where, Q:- Quality factor ; A=1+Rf/R1

1+Rf/R1= 3- 0.765

Rf=1.235R1

Select **R1=10KΩ**

Therefore, **Rf=12.35KΩ**

2.<u>Low pass filter:</u>

The expression for cutoff frequency is given by:

f= 1/(2*∏*R*C)

assume , **C=0.01µF**

Cutoff frequency, f=3.4KHz

We get,

R=4.7KΩ

Now, gain(A)=3-(1/Q)

where, Q:- Quality factor ; A=1+Rf/R1

1+Rf/R1= 3- 0.765

Rf=1.235R1

Select **R1=1KΩ**

Therefore, **Rf=1.235KΩ**

SECOND STAGE:

1. High pass filter:

The expression for cutoff frequency is given by:

f= 1/(2*∏*R*C)

assume , **C=0.01µF**

Cutoff frequency, f=100Hz

We get ,

R= 160KΩ

Now, gain(A)=3-(1/Q)

where, Q:- Quality factor ; A=1+Rf/R1

1+Rf/R1= 3- 1.848

Rf=0.152R1

Select **R1=10KΩ**

Therefore, **Rf=1.5KΩ**

2. Low pass filter:

The expression for cutoff frequency is given by:

f= 1/(2*∏*R*C)

assume , **C=0.01µF**

Cutoff frequency, f=3.4KHz

We get,

R=4.7KΩ

Now, gain(A)=3-(1/Q)

where, Q:- Quality factor ; A=1+Rf/R1

1+Rf/R1= 3- 1.848

Rf=0.152R1

Select **R1=10KΩ**

Therefore, **Rf=1.5KΩ**

Circuit Diagram:

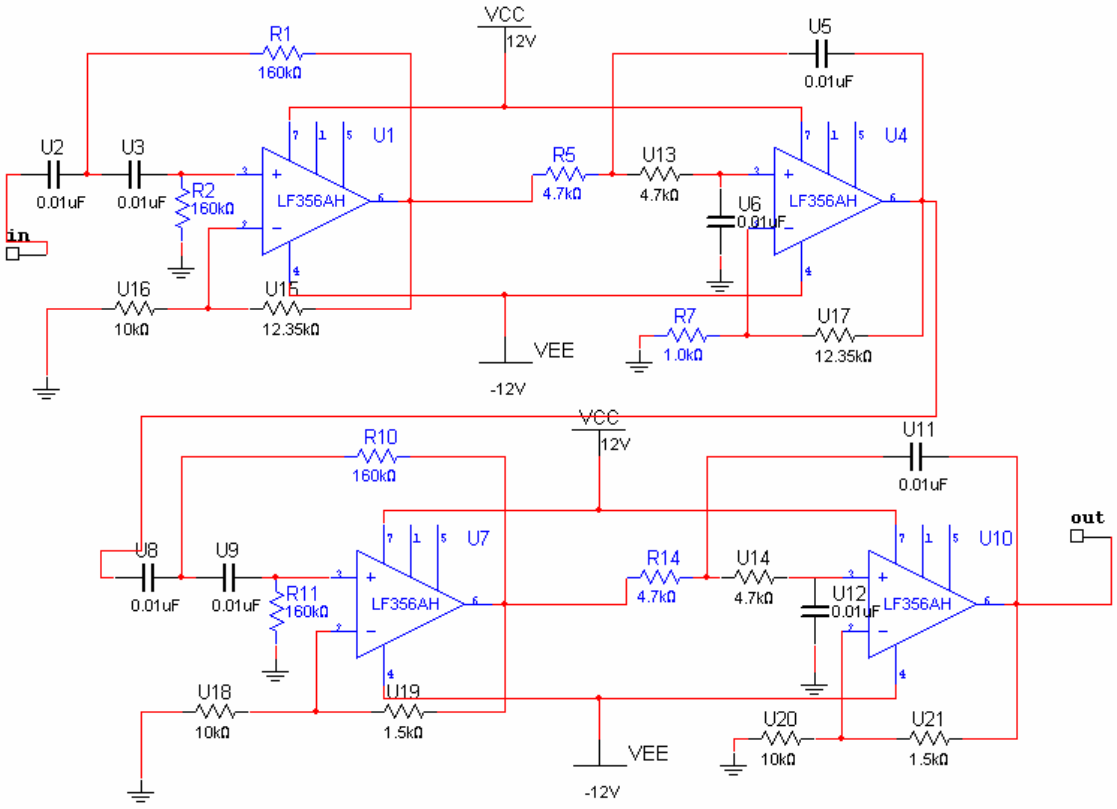


Fig 5.1 4th Order BPF

5.2 RAM INTERFACING:

External RAM IC has been interfaced with P89V51RD2 microcontroller.
The RAM IC used is HY6264, whose chip size is 64 KBits.

The port 0 of the microcontroller is multiplexed i.e. P0.0 to P0.7 pins is used as both data lines & lower address byte. The address & datalines can be separate out using a latch. The latch IC used is 74LS373. On receiving a high ALE(Address latch enable) pulse we get lower address byte at the output of the latch otherwise the port 0 acts as data line.

The Read(Rd)/Write(Wr) pin of the microcontroller is connected to the respective pins of the RAM IC as shown in the figure below:

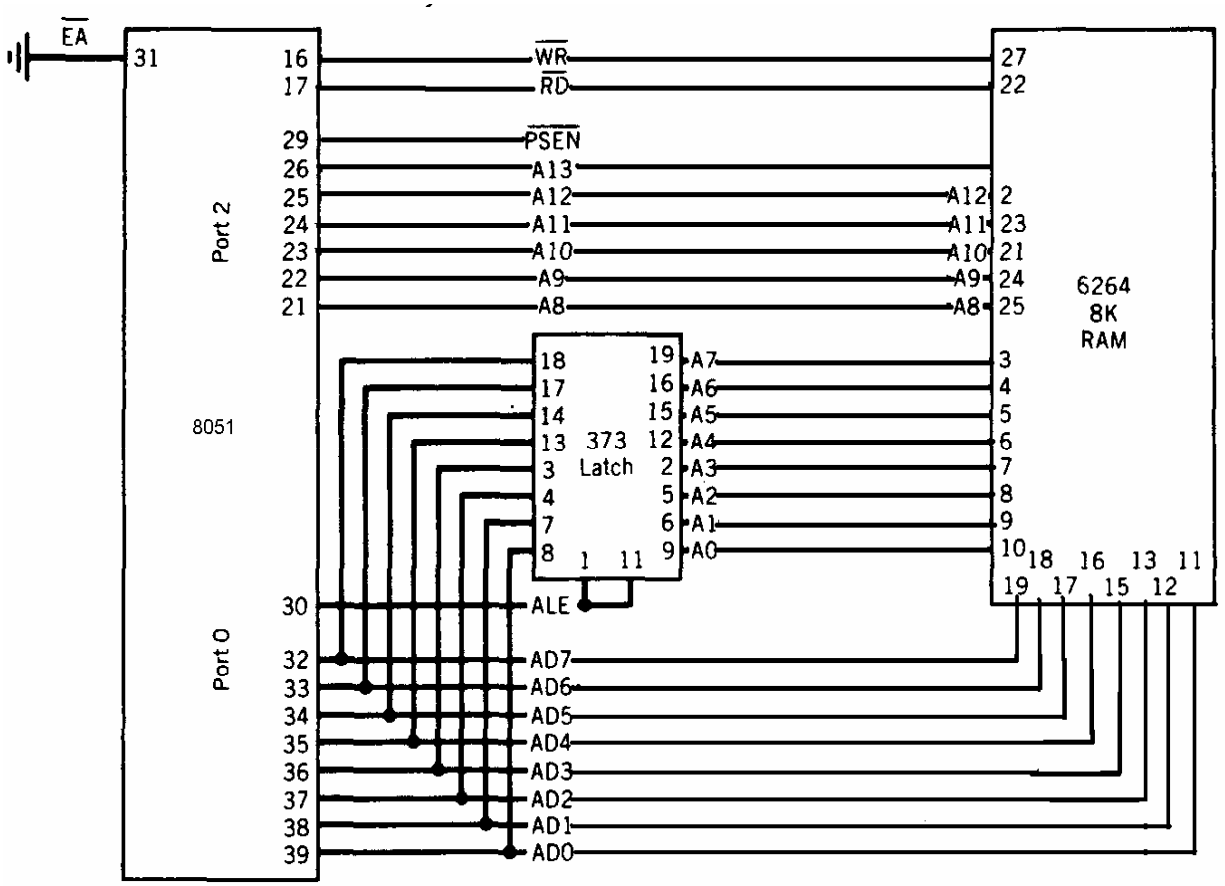


Fig 5.2 RAM Interface Diagram

5.3 Zero Crossing Detector

Zero Crossing Detector is nothing but the Infinite amplitude clipping. Our features are extracted only from these infinitely clipped signals. The signal coming from the filter stage is to be given directly to the ZCD stage. What we want at the output is only two voltage levels. By using lm311 we got the required output. But at the output pull-up resistor has to be connected. The circuit diagram is as shown.

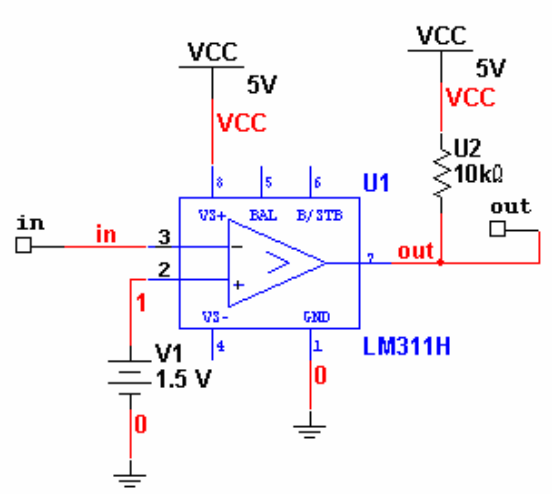


Fig 5.3 Zero Crossing Detector Implementation

5.4 LED interface

Output is in the form of LED indication. We have used buffer IC 74LS245. port 1 is used to give input to buffer. We have used 330Ω resistor in series with LED as shown.
For the IC pin no. 20 is +Vcc and 10 is GND.

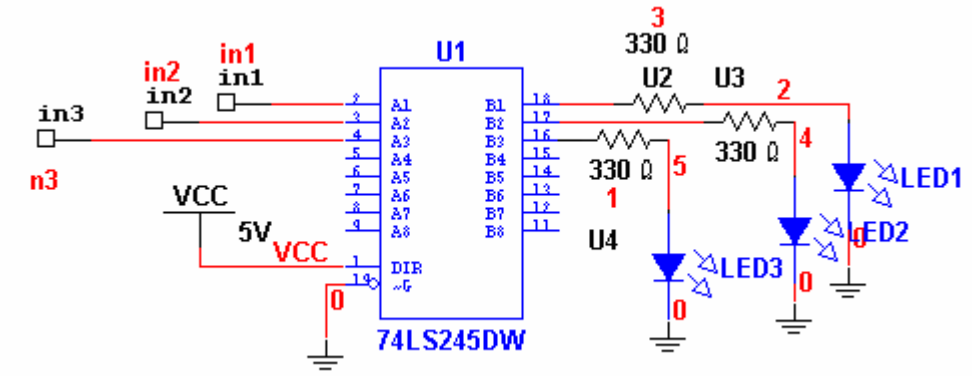


Fig 5.4 LED Buffer

Chapter 6

Debugging

6.1 Band pass filter

We had to encounter a good number of problems while implementing the filter. Initially we had planned for a sixth order filter but due to its lesser probability of working we settled for a fourth order filter.
Designing filters has always been a problem with every system, here we list few bugs that we found with the circuit:

1.The whole circuit was build on the bread board. The supply voltages & input was given to the circuit , but the output was saturated.

Sol:
i)We first tried reducing the gain (which would obviously affect the butterwoth response).Still we got the same saturated out put.

ii) For the time being the input was switched off, and we still got the output.
One possible reason would be that the circuit was acting like a oscillator and that definitely proves something wrong with the feed back. Then we found the real trouble which was the capacitor we had used 0.001µF capacitor instead of 0.1 µF.

2. Another problem was that when all the components are not perfectly grounded some spurious signals were shown at the output. If even a single component is not properly grounded the output will not be proper.

3. As lots of components are being used, there are chances that legs of two components may touch each other & get shorted thereby giving weird output.

4. The last fault with the filter circuit was with the PCB (Printed circuit board). By mistake, we got the component side same as the soldering side!

6.2 Ram Interfacing

While implementing the circuit on the bread board we did not face any problem with the circuit although there was large number of connections to be made. Also there was no problem when individual RAM locations were accessed to glow LED at the port pin.
While serially accessing all the memory locations through windows HyperTerminal we could access the first 760 bytes correctly but not the rest.
But when these locations were accessed individually we got the desired results.
The problem has not been debugged. We think RAM is working properly; there might be some problem with the windows hyper terminal (least chance).
So, to avoid any risks we have used IAP (In Application Programming).

6.3 Zero Crossing Detector

There was no problem with the working of zero crossing detector except for the first time when the input to the comparator was showing clipped.
The problem disappeared when implemented on another bread board. But we were implementing using op-amp 741. Since we are giving +12 V as Vcc and -12V as Vee we were getting output as +Vsat and –Vsat. And we want only 0 and 5volt as output. So we used lm311 with pull-up resistor at output.

6.4 LED Interface:

We thought it is the simplest part and it is. Important thing was to connect pin no 1 DIR to Vcc and pin no 19 to ground.

Chapter 7

Software

The following software have been used in the course of this project:

1. MATLAB 7.5.0: For simulation of the entire system as a proof of concept.
2. Multisim 7: For purpose of simulation of circuit diagrams involved.
3. PCB 123: For the purpose of the design of PCB layout.
4. Keil IDE: For programming the 8051 Micro-Controller.
5. 8051 IDE.

7.1 Algorithms

7.1.1 End Point Detection

End-pt detection based on ZCs

Program operates on a raw shifted signal of fixed length

1. First it traverses the signal from the beginning until the amplitude crosses a threshold

2. Now, it traverses further until length of successive intervals between ZCs crosses a threshold

3. The end point is taken as the sample where the signal first crossed the amplitude threshold

4. Same process is repeated in the reverse direction

7.1.2 Histogram Formation

1. Input infinite amplitude clipped signal u[n] is divided in a number of short time windows for every one of the calculated W samples.

2. The histogram for each of this short time window is found.

3. The number of times only one sample is recorded between successive zero crossings will constitute the element number 1 of the vector. The number of times only two samples are recorded between successive zero crossings will constitute the element

number two of the feature vector and so on. In this way we construct an histogram which is an appropriate feature vector.

7.1.3 Dynamic Time Warping

1. Start with the calculation of g(1,1) = d(1,1).

2. Calculate the first row g(***i***, 1) = g(***i***–1, 1) + d(***i***, 1).

3. Move to the second row g(***i***, 2) = min(g(***i***, 1), g(***i***–1, 1), g(***i*** – 1, 2)) + d(***i***, 2). Book keep for each cell the index of this neighboring cell, which contributes the minimum score (red arrows).

4. Carry on from left to right and from bottom to top with the rest of the grid g(***i***, ***j***) = min(g(***i***, ***j***–1), g(***i***–1, ***j***–1), g(***i*** – 1, ***j***)) + d(***i***, ***j***).

5. Trace back the best path through the grid starting from g(***n***, ***m***) and moving towards g(1,1) by following the red arrows.

7.2 Flowcharts

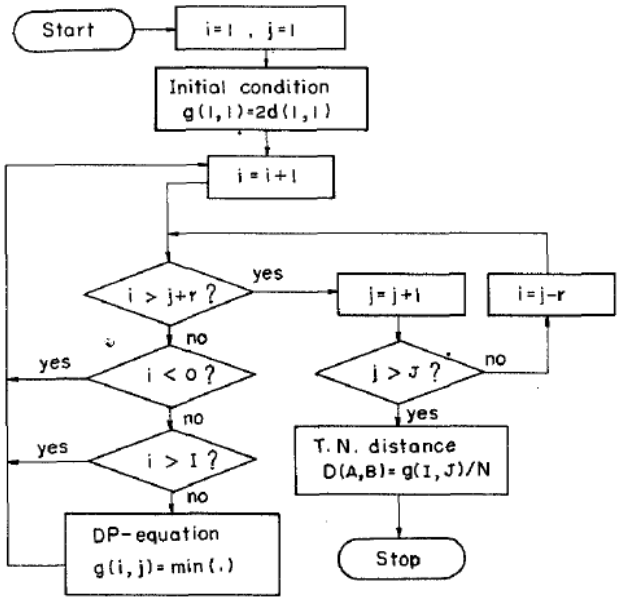
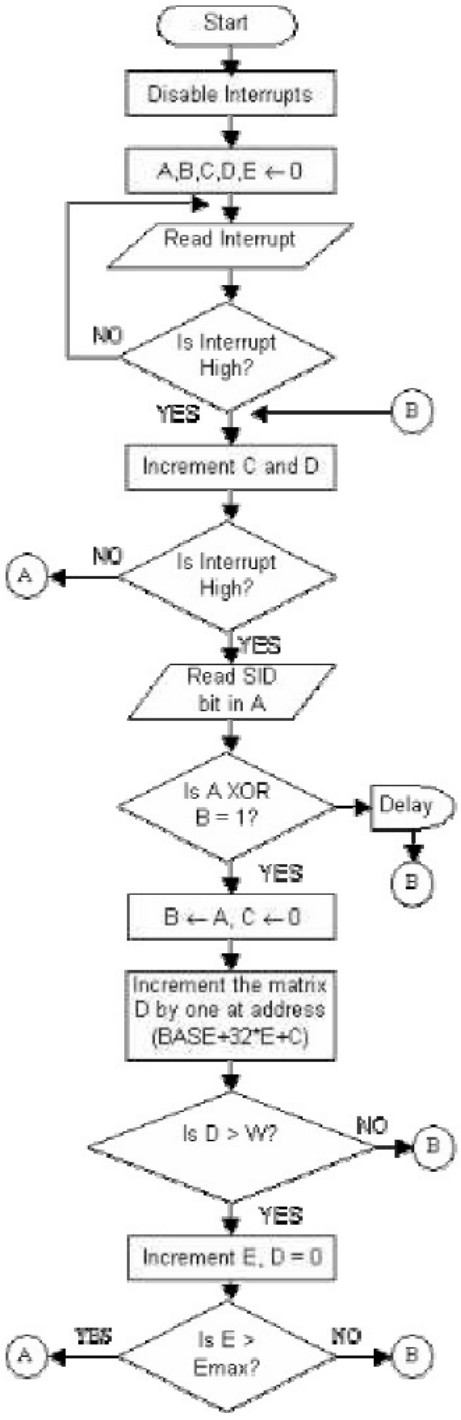


Fig 7.1 DP Matching Flowchart



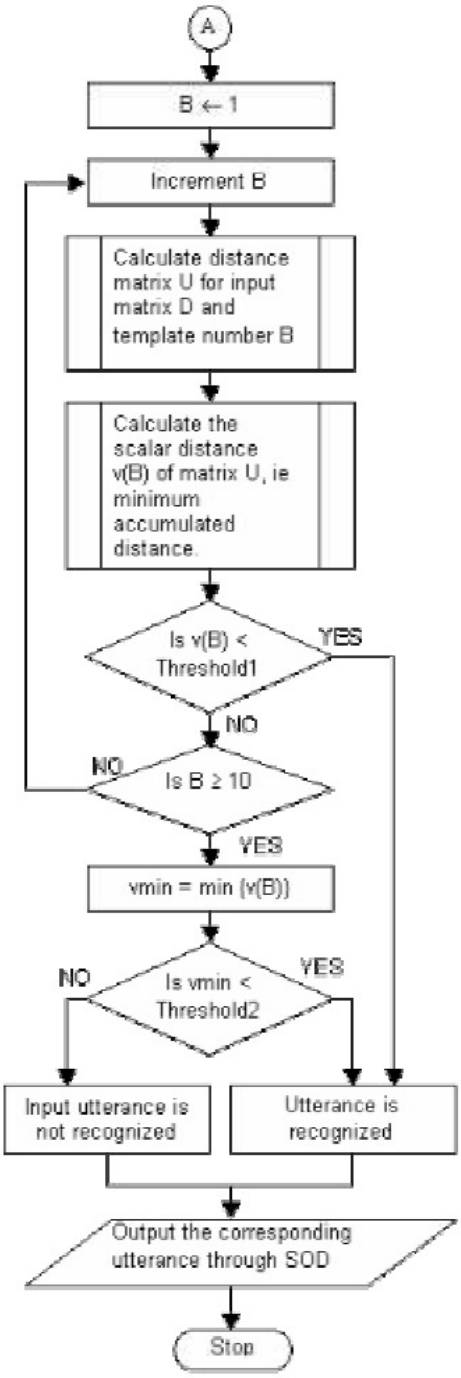


Fig 7.2 Flowchart for feature extraction. Fig 7.3 Flowchart for distance calculation and classifcation

Chapter 8

Performance Evaluation

The system was implemented on MATLAB, and evaluated for accuracy of recognition. The working methodology was simple. First an utterance was taken, and after repeated trials it was averaged and stored as part of a library. This procedure was repeated for all utterances. We created five libraries. The five libraries were as

1. With BPF of order 4
2. With BPF of order 20
3. With BPF of order 20 and Shubhendu's voice
4. With BPF of order 20 and Arun's voice
5. With BPF of order 20 and Ashutosh's voice.

The performance was evaluated in the following cases:

1. Input given with 4^{th} order BPF, enrolled speaker.
2. Input given with 20^{th} order BPF, enrolled speaker
3. Input given with 4^{th} order BPF with no enrolled library
4. Input given with 20^{th} order BPF with no enrolled library
5. Input without enrolled library, 20^{th} order BPF but no End Point detecting mechanism.

A sample example is given below:

A library for a case when the filter order is 20 has the following waveforms and histograms:

Utterance "Ek"

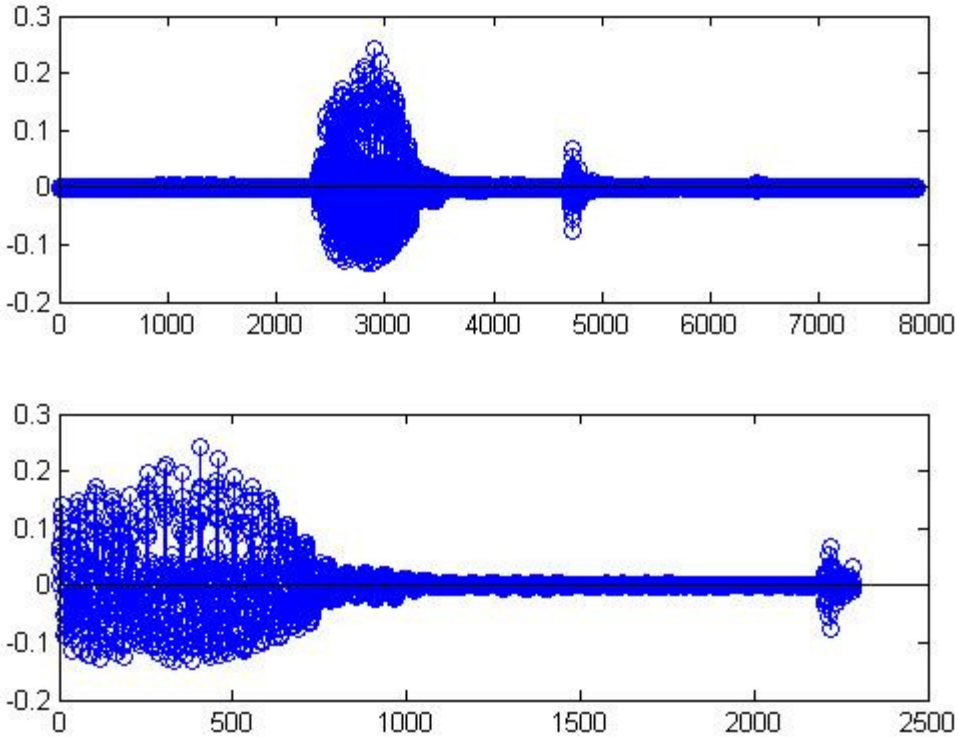


Fig 8.1 Plot of utterance “ek”, with and without end point detection

Histogram for utterance “ek”

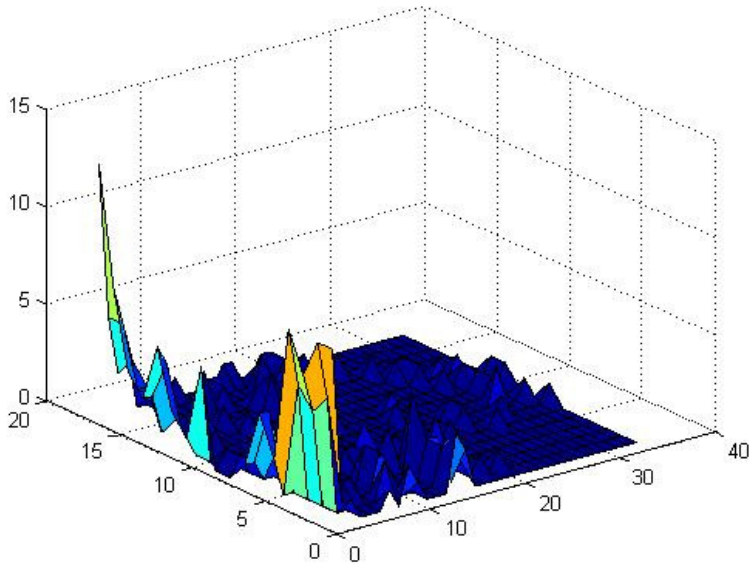


Fig 8.2 Histogram for “ek”

Utterance “do”

Plot:

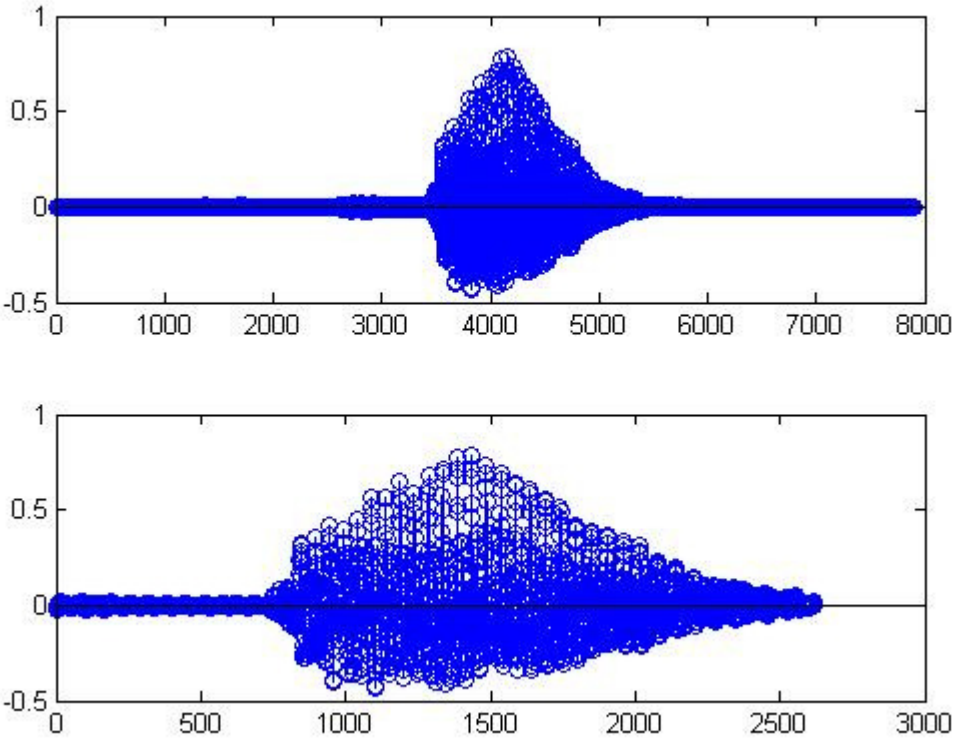


Fig 8.3 Utterance “do” with for without end point detection

Histogram:

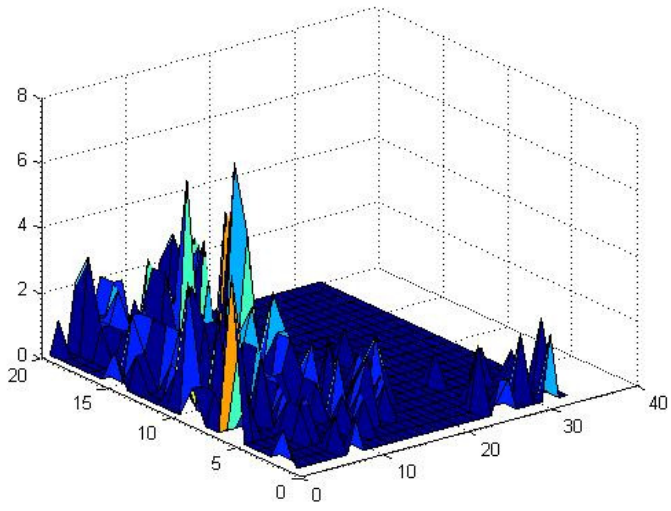


Fig 8.4 Histogram for “do”

Utterance "teen"

Plot

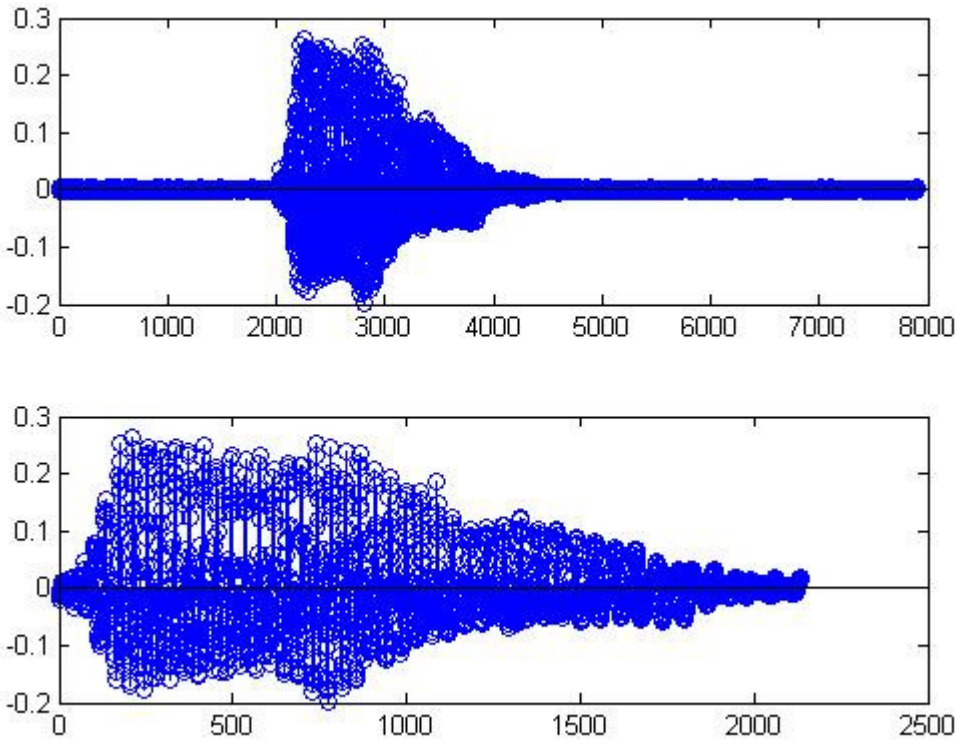


Fig 8.5 "Teen" with and without end point detection

Histogram:

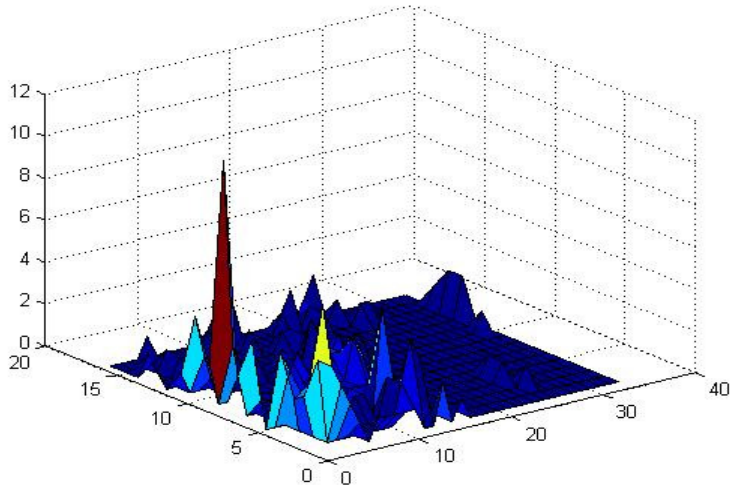


Fig 8.6 Histogram for "teen"

Similarly templates for all ten utterances are stored in the library and an exhaustive performance evaluation was done.

Performance in terms of accuracy of recognition was as follows:

1. Input given with 4^{th} order BPF, enrolled speaker: 65 %
2. Input given with 20^{th} order BPF, enrolled speaker: 73 %
3. Input given with 4^{th} order BPF with no enrolled library: 50 %
4. Input given with 20^{th} order BPF with no enrolled library: 56 %
5. Input with enrolled library, 20^{th} order BPF but no End Point detecting mechanism: 54 %
6. Input without enrolled library, 20^{th} order BPF but no End Point detecting mechanism: 48 %

Conlusion:

Out of the above results we conclude that recognition accuracy is the best with a mechanism for end point detection in place and a higher order filter with the speaker being already enrolled.

The performance significantly degraded with the reduction of filter order and end point detection.

The system gave a moderately good performance with a speaker who was not enrolled *a priori.*

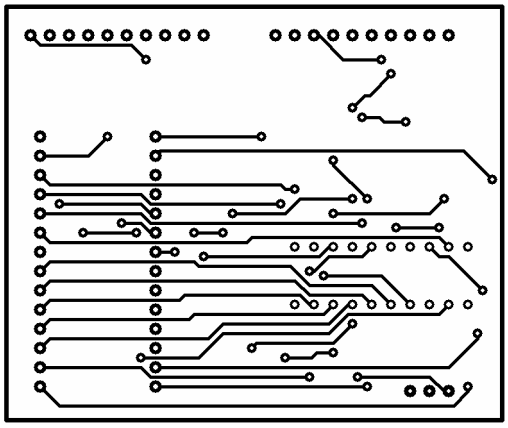
Performance was the worst with utterances which had similar waveforms such as “ek” and “saath”, “Do” and “Nau”. This problem can be eliminated by very specific utterances of the same. And in device control applications it may be combated by using very distinct sounds such as “On” and “Off” where there is no chance of a clash.

Chapter 9

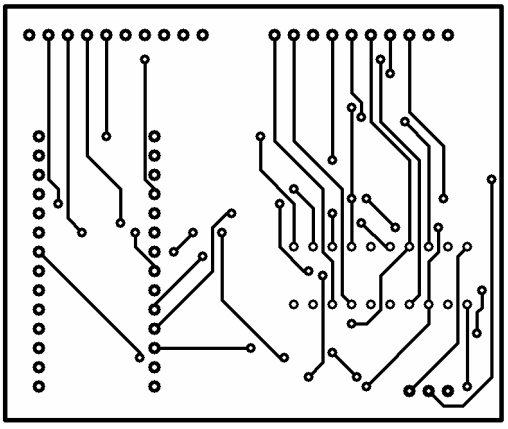
PCB Layout and Artwork Design

RAM Layout

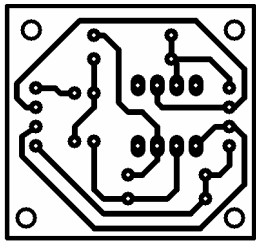
1. Top Layer



2. Bottom Layer



Filter Stage



Chapter 10

Possible Improvements and Future Scope

As a consequence of the comprehensive performance analysis, a great improvement could be made in recognition accuracy by the following:

1. Designing a **very high order BPF**, anything above order 10.
2. Designing an efficient algorithm and system for **effective end point detection**. This may be done by an idea of deciding a threshold amplitude.
3. The algorithm for DTW may be replaced by any of the two improved versions of the Dynamic Time Warping algorithm originally given by Sakoe Chiba *et al.*
    1. **Derivative Dynamic Time Warping**: This algorithm [11] was given by Pazzani *et al.* The DDTW algorithm can significantly improve performance as it addresses two major problems of the DTW algorithm.

    1.1 The DTW gives unintuitive alignments when a large number of points are mapped to a single point (this are called singularity points), this problem persists even after slope constraints are applied. Since DDTW deals with derivatives, it eliminates this problem.

    1.2 The DTW algorithm may fail to find obvious alignments in two sequences simply because the feature in one sequence is slightly higher or lower than the feature in another sequence. A review of DDTW suggests that it could eliminate this problem.

    2. Fast Dynamic Time Warping: One major problem with Dynamic Time Warping given by Sakoe Chiba *et al* is that it has O complexity of order 2, which can make it unsuitable for longer time series as it would be computationally intensive and would also take much greater time to complete. This problem could be solved by using a modified algorithm [4]. This algorithm promises to remove redundancy and eliminate the above issue.

Applications:

The speech recognizer designed could have wide applications in voice controlled appliance applications, robots, wheel chairs. This would be very efficient as generally the utterance in this case would be very different such as “ON“ or “OFF”, “LEFT” or “RIGHT”, or so on.

It could also have applications in simple voice controlled hand held computers.

The system so designed promises a cheaper alternative to the available system which gives satisfactory performance.

Chapter 11

Component List and Bill of Materials

S.No	Name Of The Component	Cost of the Item
1.	Development Board	Rs.200
2.	Power Supply Transformer – 12V, 1 Amp. Bridge IC – 2 Amp. Capacitor 2200 μf, 16V. IC – 7912, 7812, 7805. Heat Sink.	Rs 80
3.	Listening Bug Kit	Rs 77
4.	PCB (4 Nos.)	Rs 70
5.	LF 356 (4 Nos.)	Rs 24
6.	Resistors: 150K (4 Nos.),5.6K (2 Nos.),6.8K(1 Nos.),10K (4 Nos.)	Rs 4
7.	Capacitors: 103 (8 Nos.)	Rs 4
8.	Resistors: 4.7K(2 Nos). 10K(1 Nos)	Rs 2
9.	LM – 311 (1 Nos.)	Rs 13
10.	74LS245N(1 Nos.)	Rs 11
11.	Resistors: 330(4 Nos.)	Rs 2.5
12.	PCB, Double Sided (1 Nos.)	Rs 200
13.	Relimate: 4 pin (6 Nos.)	Rs. 52
14.	Relimate: 3 pin (2 Nos.)	Rs. 33
15.	Relimate: 2 pin (2 Nos.)	Rs 22
16.	89C51RD2 (1 Nos.)	Rs 250
	Total	**Rs 1044.50**

Bibliography

[1] H. Sakoe and S. Chiba, "Dynamic Programming Optimization for Spoken Word Recognition", *IEEE Trans. Acoustics, Speech, Signal Proc.,* ASSP-26(1):43-49, February 1978.

[2] Lipovac and V. Sarajevo, "Zero-crossing-based linear prediction for speech recognition", *Electronics Letters*, pages 9092, vol. 25 Issue 2,19 Jan 1989.

[3] Lawrence Rabiner, and Biing-Hwang Juang, "Fundamentals of Speech Recognition", *PTR Prentice Hall, Englewood Cliffs, New Jersey 07632*, 1993.

[4] "FastDTW: Toward Accurate Dynamic Time Warping in Linear Time and Space", Stan Salvador and Philip Chan.

[5] "Performance comparison of distance metrics in content-based Image retrieval applications", A Vadivel, A K Majumdar, Shamik Sural

[6] MIT OCW

[7] "8051 Microcontroller and Embedded Systems, The", Muhammad Ali Mazidi, Janice Gillispie Mazidi, *Prentice Hall of India*, 2003.

[8] "Op-Amps and Linear Integrated Circuits" Ramakant Gayakwad, *Prentice Hall of India*, 4th Edition.

[9] "Matlab and Simulink Student Version Release 14", The Mathworks Inc, *Prentice Hall of India*, 2006.

[10] “Getting Started with MATLAB 7: A Quick Introduction for Scientists and Engineers”, Rudra Pratap, *Oxford University Press Inc*, 2005.

[11] “Derivative Dynamic Time Warping”, Eamonn J. Keogh and Michael J. Pazzani.