“ACCESSING VIDEOS AND IMPLEMENTING SPEAKER RECOGNITION SYSTEM USING SPEECH PROCESSING”

By

Sachin Malhotra, Anurag Chutani, Tarun Goel

Under the supervision of

Dr. Neeru Sharma

May-2014

Dissertation submitted in partial fulfilment

Of the requirement for the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS & COMMUNICATION ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY

WAKNAGHAT, SOLAN – 173234, INDIA

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY

(Established under the Act 14 of Legislative Assembly of Himachal Pradesh)

Waknaghat, P.O. DomeharBani. Teh. Kandaghat, Distt. Solan- 173234(H.P)

Phone: 01792-245367, 245368,245369

Fax-01792-245362

CERTIFICATE

This is to certify that the work titled **"Accessing videos and implementing speaker recognition system using speech processing"** submitted by **"Mr. Sachin Malhotra, Mr. Anurag Chutani, Mr. Tarun Goel"** in the partial fulfillment for the award of degree of Bachelor of Technology (ECE) of Jaypee University of Information Technology, Waknaghat has been carried out under my supervision. This work has not been submitted partially or wholly to any other university or institution for the award of this or any other degree or diploma.

Date:

Dr. Neeru Sharma

(Assistant Professor)

Department of Electronics and Communication Engineering

Jaypee University of Information Technology (JUIT)

Waknaghat, Solan – 173234, India

(Supervisor)

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY

(Established under the Act 14 of Legislative Assembly of Himachal Pradesh)

Waknaghat, P.O. DomeharBani. Teh. Kandaghat, Distt. Solan- 173234(H.P)

Phone: 01792-245367, 245368,245369

Fax-01792-245362

DECLARATION

We hereby declare that the work reported in the B. Tech thesis entitled "**Accessing Videos and Implementing Speaker Recognition System using Speech Processing"** submitted by "**Mr. Sachin Malhotra, Mr. Anurag Chutani and Mr. Tarun Goel"** at Jaypee University Of Information Technology, Waknaghat is an authentic record of our work carried out under the supervision of **Dr. Neeru Sharma**. This work has not been submitted partially or wholly to any other university or institution for the award of this or any other degree or diploma.

Mr. Sachin Malhotra

Mr. Anurag Chutani

Mr. Tarun Goel

Department of Electronics and Communication Engineering

Jaypee University of Information Technology (JUIT)

Waknaghat, Solan – 173234, India

ACKNOWLEDGEMENT

After the competitions of our thesis work, we feel to convey our indebtedness to all those who helped us to reach our goal. We take this opportunity to express our profound gratitude and deep regards to our guide **Dr. Neeru Sharma** for her exemplary guidance, monitoring and constant encouragement throughout the course of this project. The blessing, help and guidance given by her time to time shall carry us a long way in the journey of life on which we are about to embark. We are obliged to all our faculty members of JUIT, for the valuable information provided by them in their respective fields. We are grateful for their cooperation during the period of our project.

Contents

List of Figures

Abstract

Modern speech understanding systems merge interdisciplinary technologies from signal processing, pattern recognition, natural language, and linguistics into a unified statistical framework. These systems, which have applications in a wide range of signal processing problems, represent a revolution in Digital Signal Processing (DSP). Once a field dominated by vector-oriented processors and linear algebra-based mathematics, the current generation of DSP-based systems rely on sophisticated statistical models implemented using a complex software paradigm. Such systems are now capable of understanding continuous speech input for vocabularies of several thousand words in operational environments. We explored the core components of modern statistically-based speech recognition systems. The objective of this project is to implement a speech recognition engine and develop a system for speaker recognition using Mel Frequency Cepstrums and Vector Quantization. This would involve the design of an efficient MATLAB code on a PC. Throughout the development, measures will be taken to keep the memory requirement and the processing time of the software as small as possible. Every Speech Recognition system must be judged on two basic factors which govern its usability - accuracy and speed. Unfortunately, one of them almost invariably comes at the cost of the other. A higher accuracy rate implies a wider training sequence and a higher number of iterations in the learning algorithm. On the other hand, accuracy remains an important objective of our project. The precision of the above two mentioned algorithms that have been used depend almost entirely on the model parameters for every isolated word which needs to be calculated at the very outset. To improve accuracy, we calculate these parameters in a MATLAB environment deriving our results on a large number of test sequences recorded in a typical noisy environment.

Chapter 1

Introduction

The fundamental purpose of speech is communication, i.e., the transmission of messages. According to Shannons information theory, a message represented as a sequence of discrete symbols can be quantified by its information content in bits, and the rate of transmission of information is measured in bits/second (bps). In speech production, as well as in many human-engineered electronic communication systems, the information to be transmitted is encoded in the form of a continuously varying (analog)waveform that can be transmitted, recorded, manipulated, and ultimately decoded by a human listener. In the case of speech, the fundamental analog form of the message is an acoustic waveform, which we call the speech signal[4]. Speech signals, can be converted to an electrical waveform by a microphone, further manipulated by both analog and digital signal processing, and then converted back to acoustic form by a loudspeaker, a telephone handset or headphone, as desired. This form of speech processing is, of course, the basis for Bells telephone invention as well as todays multitude of devices for recording, transmitting, and manipulating speech and audio signals. Although Bell made his invention without knowing the fundamentals of information theory, these ideas have assumed great importance in the design of sophisticated modern communications systems. Therefore, even though our main focus will be mostly on the speech waveform and its representation in the form of parametric models, it is nevertheless useful to begin with a discussion of how information is encoded in the speech waveform.

Approaches

There are broadly three approaches of speech recognition, namely:

1. **The acoustic-phonetic approach**

   It is basically based on the theory of acoustic phonetics that there exist finite, distinctive phonetic units in the spoken language and that the phonetic units are broadly characterized by a set of properties that manifest in speech signal, or its spectrum over time. Even though the acoustic properties are of phonetic units are highly variable, both with the speakers and with the neighboring phonetic units, it is assumed are straightforward and can be readily learnt and appied in practical situations. Thus, the first step in acoustic-phonetic approach to speech recognition is called a segmentation and labeling phase because it involves segmenting the speech signal into discrete(in time) regions where the acoustic properties of the signal representative of one (or possibly several) phonetics units. To do speech recognition, we need a second step. This second step attempts to derive a valid word from the sequence of phonetic labels produced in the first step, which is consistent with the constraints of speech recognition task.

2. **The pattern-recognition approach**

   This approach to speech recognition is basically one in which the speech patterns are used directly without explicit feature determination (in the acoustic-phonetic sense) and segmentation. As with most pattern recognition approaches, the method has two steps- namely, training of speech patterns and recognition of patterns via pattern comparison. Speech knowledge is brought into the system via the training procedure. The concept is that if enough versions of a pattern to be recognized are included in the training set provided to the algorithm, the training procedure should be able to adequately characterize the acoustic properties of the pattern (with no regard for or knowledge of any other pattern presented to the training procedure). This type of characterization of speech via training is called pattern classification because the machine learns which acoustic properties of the speech class are reliable and repeatable across all training tokens of the pattern. The utility of the method is the pattern-comparison stage, which does a direct comparison of the unknown speech (the speech to be recognized), with each possible pattern learned in the training phase and classifies the unknown speech according to the goodness of the match patterns.

3. **The artificial intelligence approach**

   The artificial intelligence approach to speech recognition is a hybrid of the acoustic-phonetic approach and the pattern-recognition approach in that it exploits ideas and concepts of both methods. The artificial intelligence approach attempts to mechanize the recognition procedure according to the way a person applies its intelligence in visualizing, analyzing and finally making a decision on the measured acoustic features. In particular, among the techniques used within this class of methods are the use of an expert system for segmentation and labeling so that this crucial and most difficult step can be performed with more than just the acoustic information used by pure acoustic- phonetic methods; learning and adapting over time (i.e., the concept that knowledge is often both static and dynamic and that models must adapt to the dynamic component of the data) ; the use of neural networks for learning the relationships between phonetic events and all known inputs as well as for discrimination between similar sound classes.

   The use of neural network could represent a separate structural approach to speech recognition or be regarded as an implementation architecture that may be incorporated in any of the above three classical approaches.

Issues with acoustic phonetic approach

Many problems account for phonetic-acoustic approach to speech recognition. These problems, in many ways account for lack of success in practical speech recognition systems.

1. The method requires extensive knowledge of the acoustic properties of phoneme units. This knowledge is at best incomplete and at worst totally unavailable for all bit the simplest of situations.

2. The choice of features is made mostly based on ad hoc considerations. For most systems the choice of features is based on intuition and is not optimal in a well-defined and meaningful sense.

3. The design of sound classifier is also not optimal. Ad hoc methods are generally used to

construct binary decision trees. More recently classification and regression tree (CART) methods have been used to make the decision tree more robust. However, since the choice of features is most likely to be suboptimal, optimal implementation of CART is rarely achieved.

No well-defined automatic procedure exists for tuning the method(i.e. adjusting the thresholds) . In fact, there is not even an ideal way of labeling the training speech in a manner consistent and agreed on uniformly by a wide class of linguistic experts.

Flow Chart

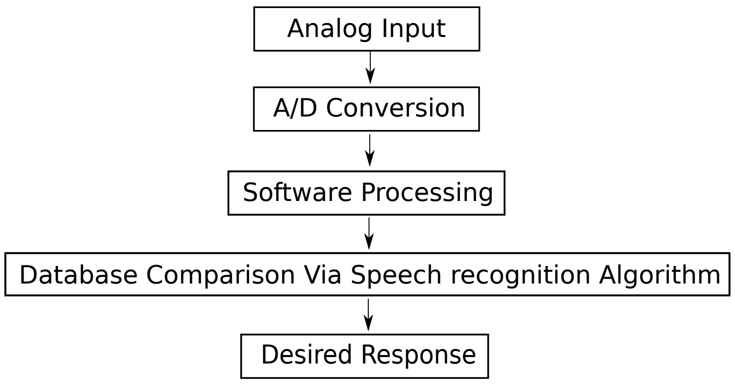


Figure 1.1: Dynamic time warping flow chart

Terms and Concepts

Following are a few of the basic terms and concepts that are fundamental to speech recognition. It is important to have a good understanding of these concepts when developing applications of Speech Recognition.

Utterances When the user says something, this is known as an utterance. An utterance is any stream of speech between two periods of silence. Utterances are sent to the speech engine to be processed. Silence, in speech recognition, is almost as important as what is spoken, because silence delineates the start and end of an utterance. Here's how it works. The speech recognition engine is "listening" for speech input. When the engine detects audio input - in other words, a lack of silence – the beginning of an utterance is signaled. Similarly, when the engine detects a certain

amount of silence following the audio, the end of the utterance occurs. Utterances are sent to the speech engine to be processed. If the user doesnt say anything, the engine returns what is known as a silence timeout - an indication that there was no speech detected within the expected timeframe, and the application takes an appropriate action, such as reprompting the user for input.
An utterance can be a single word, or it can contain multiple words (a phrase or a sentence). If the users pause too long between the words of a phrase, the end of an utterance can be detected too soon, and only a partial phrase will be processed by the engine.

Pronunciations The speech recognition engine uses all sorts of algorithms to convert spoken input into the desired response. One piece of information that the speech recognition engine uses to process a command is its pronunciation, which represents what the speech engine thinks a word should sound like. Words can have multiple pronunciations associated with them. For example, the word "the" has at least two pronunciations in the U.S. English language: "thee" and "thuh." As a application developer, you may want to provide multiple pronunciations for certain words and phrases to allow for variations in the ways your callers may speak them.

Grammars As an application developer, you must specify the words and phrases that users can say to your application. These words and phrases are defined to the speech recognition engine and are used in the recognition process. A grammar uses a particular syntax, or set of rules, to define the words and phrases that can be recognized by the engine. A grammar can be as simple as a list of words, or it can be flexible enough to allow such variability in what can be said that it approaches natural language capability. Grammars define the domain, or context, within which the recognition engine works. The engine compares the current utterance against the words and phrases in the active grammars. If the user says something that is not in the grammar, the speech engine will not be able to decipher it correctly.

Speaker Dependence vs. Speaker Independence Speaker dependence describes the degree to which a speech recognition system requires knowledge of a speakers individual voice characteristics to successfully process speech. The speech recognition engine can "learn" how you speak words and phrases; it can be trained to your voice. Speech recognition systems that require a user to train the system to his/her voice are known as speaker-dependent systems. If you are familiar with desktop dictation systems, most are speaker dependent. Because they operate on very large vocabularies, dictation systems perform much better when the speaker has spent the time to train the system to his/her voice. Speech recognition systems that do not require a user to train the system are known as speaker independent systems.

Accuracy The performance of a speech recognition system is measurable. Perhaps the most widely used measurement is accuracy. It is typically a quantitative measurement and can be calculated in several ways. Arguably the most important measurement of accuracy is whether the desired end result occurred. This measurement is useful in validating application design. For example, if the user said “yes,“ the engine returned “yes,” and the “YES” action was executed, it is clear that the desired end result was achieved. But what happens if the engine returns text that does not exactly match the utterance? For example, what if the user said “nope,” the engine returned “no,” yet the “NO” action was executed? Should that be considered a successful dialog? The answer to that question is yes because the desired end result was achieved. Another measurement of recognition accuracy is whether the engine recognized the utterance exactly as spoken. This measure of recognition accuracy is expressed as a percentage and represents the number of utterances recognized correctly out of the total number of utterances spoken. It is a useful measurement when validating grammar design. Using the previous example, if the engine returned “nope” when the user said “no,” this would be considered a recognition error. Based on the accuracy measurement, you may want to analyze your grammar to determine if there is anything you can do to improve accuracy. For instance, you might need to add “nope” as a valid word to your grammar. Recognition accuracy is an important measure for all speech recognition applications. It is tied to grammar design and to the acoustic environment of the user. You need to measure the recognition accuracy for your application, and may want to adjust your application and its grammars based on the results obtained when you test your application with typical users.

Digital Audio Basics Audio is inherently an analog phenomenon. Recording a digital sample is done by converting the analog signal from the microphone to an digital signal through the A/D converter in the sound card. When a microphone is operating, sound waves vibrate the magnetic element in the microphone, causing an electrical current to the sound card (think of a speaker working in reverse). Basically, the A/D converter records the value of the electrical voltage at specific intervals[5].

There are two important factors during this process. First is the “sample rate”, or how often to record the voltage values. Second, is the “bits per sample”, or how accurate the value is recorded. A third item is the number of channels (mono or stereo), but for most ASR applications mono is sufficient. Most applications use preset values for these parameters and user’s shouldn’t change them unless the documentation suggests it. Developers should experiment with different values to determine what works best with their algorithms.

So what is a good sample rate for ASR? Because speech is relatively low bandwidth (mostly between 100Hz8kHz), 8000 samples/sec (8kHz) is sufficient for most basic ASR. But, some people prefer 16000 samples/sec (16kHz) because it provides more accurate high frequency information. If you have the processing power, use 16kHz. For most ASR applications, sampling rates higher than about 22kHz is a waste. And what is a good value for “bits per sample”? 8 bits per sample will record values between 0 and 255, which means that the position of the microphone element is in one of 256 positions. 16 bits per sample divides the element position into 65536 possible values. Similar to sample rate, if you have enough processing power and memory, go with 16 bits per sample.

Chapter 2

Technology Background

2.1 MATLAB

Matlab is a numerical computing programming language. It allows easy matrix manipulation, plotting of functions and data, implementation of algorithm, creation of UI, and interfacing with programs in other languages. It allows for other toolboxes to be included for additional capabilities. Typical uses of Matlab include:

- Math and computation
- Algorithm development
- Modelling, simulation, and prototyping
- Data analysis, exploration, and visualization
- Scientific and engineering graphics
- Application development, including Graphical User Interface building

In this project we have used matlab as the main tool for designing the speech and speaker recognition model. We have used various inbuilt commands and functions to implement the algorithms. Many typical computations have been done quite efficiently and a good model for speech and speaker recognition have been created. We have also created the graphical user interface using the matlab commands so as to make the program user friendly. The main commands used in the programs were as follows:

2.1.1 Commands and their brief description

DISP(X) displays the array, without printing the array name. In all other ways it's the same as leaving the semicolon off an expression except that empty arrays don't display. If X is a string, the text is displayed.

PAUSE(n) pauses for n seconds before continuing, where n can also be a fraction. The resolution of the clock is platform specific. Fractional pauses of 0.01 seconds should be supported on most platforms.

EXIST Check if variables or functions are defined.

LOAD loads workspace variables from disk. LOAD FILENAME retrieves all variables from a file given a full pathname or a MATLABPATH relative partial pathname (see PARTIALPATH). If FILENAME has no extension LOAD looks for FILENAME.mat and, if found, LOAD treats the file as a binary "MAT-file". If FILENAME.mat is not found, or if FILENAME has an extension other than .mat it is treated as an ASCII file.

INPUT Prompt for user input.

ISEMPTY True for empty array. ISEMPTY(X) returns 1 if X is an empty array and 0 otherwise. An empty array has no elements, that is prod(size(X))==0.

NUM2STR Convert numbers to a string.

AUDIORECORDER(Fs, NBITS, NCHANS) creates an AUDIORECORDER object with sample rate Fs in Hertz, number of bits NBITS, and number of channels NCHANS. Common sample rates are 8000, 11025, 22050, and 44100 Hz (only 44100, 48000, and 96000 on a Macintosh). The number of bits must be 8, 16, or 24 on Windows, 8 or 16 on UNIX. The number of channels must be 1 or 2 (mono or stereo).

RECORD Record data and event information to a file.

GETAUDIODATA Gets recorded audio data in audiorecorder object. GETAUDIODATA(OBJ) returns the recorded audio data as a double array.

SAVE FILENAME saves all workspace variables to the binary "MAT-file" named FILENAME.mat. The data may be retrieved with LOAD. If FILENAME has no extension, .mat is assumed.

WAVWRITE(Y,FS,NBITS,WAVEFILE) writes data Y to a Windows WAVEfile specified by the file name WAVEFILE, with a sample rate of FS Hz and with NBITS number of bits. NBITS must be 8, 16, 24, or 32. Stereo data should be specified as a matrix with two columns. For NBITS ¡ 32, amplitude values outside the range [-1,+1] are clipped.

ISRECORDING Indicates if recording is in progress.

STRCAT Concatenate strings.

DELETE file_name deletes the named file from disk. Wildcards may be used. For example, DELETE *.p deletes all P-files from the current directory.

2.1.2 Graphical User Interface

Software that works at the point of contact (interface) between a computer and its user, and which employs graphic elements (dialog boxes, icons, menus, scroll bars) instead of text characters to let the user give commands to the computer or to manipulate what is on the screen. GUI elements are usually accessed through a pointing device such as a mouse, pen, or stylus. All programs running under a GUI use a consistent set of graphical elements so that once the user learns a particular interface, he or she can use all programs without learning additional or new commands. Pioneered

by Xerox and developed by Apple computers, GUI is now employed by all modern operating systems and application programs.
A GUI is a graphical user interface to a computer. The term came into existence because the first interactive user interfaces to computers were not graphical; they were text-and-keyboard oriented and usually consisted of commands you had to remember and computer responses that were infamously brief. The command interface of the DOS operating system is an example of the typical user-computer interface before GUIs arrived. An intermediate step in user interfaces between the command line interface and the GUI was the non-graphical menu-based interface, which let you interact by using a mouse rather than by having to type in keyboard commands.
Today's major operating systems provide a graphical user interface. Applications typically use the elements of the GUI that come with the operating system and add their own graphical user interface elements and ideas. Elements of a GUI include such things as: windows, pull-down menus, buttons, scroll bars, iconic images, wizards, the mouse, and no doubt many things that haven't been invented yet. With the increasing use of multimedia as part of the GUI, sound, voice, motion video, and virtual reality interfaces seem likely to become part of the GUI for many applications. A system's graphical user interface along with its input devices is sometimes referred to as its "look-and-feel."
When creating an application, many object-oriented tools exist that facilitate writing a graphical user interface. Each GUI element is defined as a class widget from which you can create object instances for your application. You can code or modify prepackaged methods that an object will use to respond to user stimuli.

Some of the GUI commands used in the programs is as follows:

MENU Generate a menu of choices for user input. CHOICE = MENU(HEADER, ITEM1, ITEM2, ...) displays the HEADER string followed in sequence by the menu-item strings: ITEM1, ITEM2, ... ITEMn. Returns the number of the selected menu-item as CHOICE, a scalar value. There is no limit to the number of menu items.

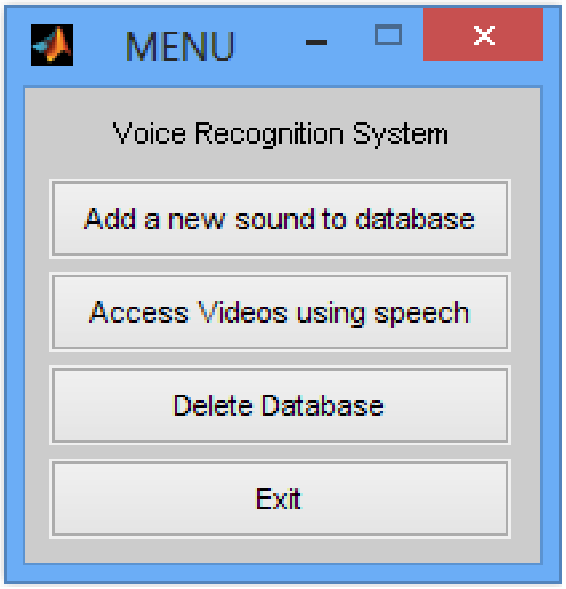
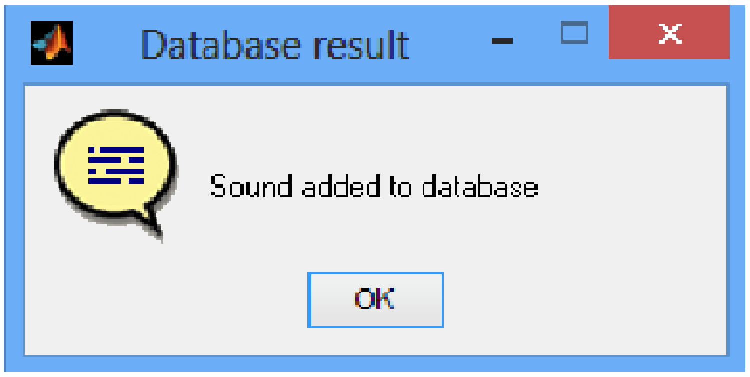


Figure 2.1: GUI - Menu

msgbox(Message) creates a message box that automatically wraps Message to fit an appropriately sized Figure. Message is a string vector, string matrix or cell array.



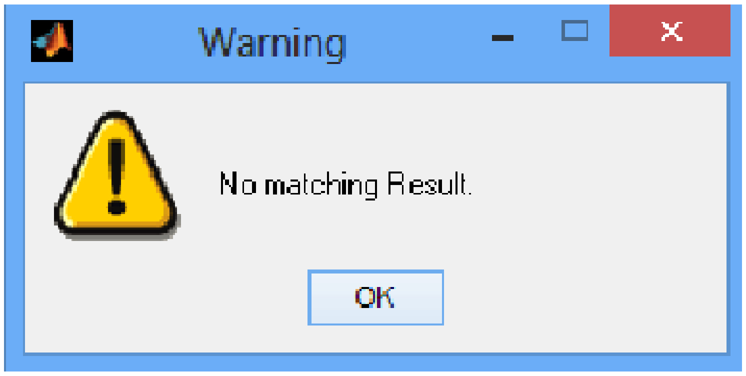
Sound added to database

Figure 2.2: GUI - Message Box

WARNDLG Warning dialog box.

HANDLE = WARNDLG(WARNSTRING,DLGNAME) creates an warning dialog box which displays WARNSTRING in a window named DLGNAME.

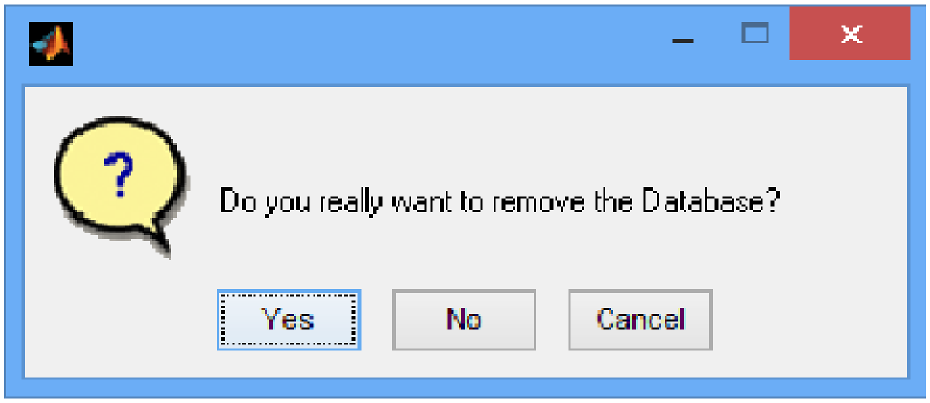
A push button labeled OK must be pressed to make the warning box disappear.



No matching Result.

Figure 2.3: GUI - Warning dialog box

QUESTDLG Question dialog box. ButtonName = QUESTDLG(Question) creates a modal dialog box that automatically wraps the cell array or string (vector or matrix) Question to fit an appropriately sized window. The name of the button that is pressed is returned in ButtonName. The Title of the figure may be specified by adding a second string argument:
ButtonName = questdlg(Question, Title)
QUESTDLG uses UIWAIT to suspend execution until the user responds. The default set of buttons names for QUESTDLG are “Yes”,“No” and “Cancel”.



Do you really want to remove the Database?

Figure 2.4: GUI - Question dialog box

Chapter 3

Algorithms

There are three approaches to speech recognition namely the acoustic phonetic approach, the pattern recognition approach and 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 the phonetic units are broadly characterized by a set of properties that are manifest in the speech signal or its spectrum over time. Pattern recognition approach to speech recognition is basically one in which the speech patterns are used directly without explicit feature determination and segmentation. Pattern recognition is concerned with the classification of objects into categories, especially by machine . A strong emphasis is placed on the statistical theory of discrimination, but clustering also receives some attention. Hence it can be summed in a single word: “classification”, both supervised (using class information to design a classifier i.e. discrimination) and unsupervised (allocating to groups without class information i.e. Clustering). Its ultimate goal is to optimally extract patterns based on certain conditions and is to separate one class from the others. Artificial Intelligence approach to speech recognition is a hybrid of the acoustic phonetic approach and the pattern recognition approach [1] in that it exploits ideas and concepts of both methods. The artificial Intelligence approach attempts to mechanize the recognition procedure according to the way a person applies its intelligence in visualizing, analyzing, and finally making a decisions on the measured acoustic features. One of the simplest and earliest approaches to pattern recognition is the template approach. Matching is a generic operation in pattern recognition which is used to determine the similarity between two entities of the same type. In template matching the template or prototype of the pattern to be recognized is available. The pattern to be recognized is matched against the stored template taking into account all allowable pose and scale changes. Dynamic Time Warping is a pattern recognition technique.

3.1 Dynamic Time Warping

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 nth point in one time series to the nth 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 to overcome this limitation and give intuitive distance measurements between time series by ignoring both global and local shifts in the time dimension.

3.1.1 Problem Formulation

The dynamic time warping problem is stated as follows:
Given two time series X, and Y, of lengths $|X|$ and $|Y|$, construct a warp path W where K is the length of the warp path and the kth element of the warp path is 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 $w_1 = (1,1)$ and finish at the end of both time series at $W_k = (|X|,|Y|)$. 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
reffig:timealign do not overlap. Every index of each time series must be used.

Stated more formally:
The optimal warp path is the warp path is the minimum-distance warp path, where the distance of a warp path W is $Dist(W)$ is the distance (typically Euclidean distance) of warp path W, and $Dist(w_{ki}, w_{kj})$ is the distance between the two data point indexes (one from X and one from Y) in

the kth element of the warp path.

3.1.2 Matching Patterns in Time

Speech is a temporal signal; by this we mean that the Linguistic information encoded in an acoustic speech signal is encoded in both the properties of the signal at any given instant and the way that those properties change over time. A case in point is a diphthongal vowel which can only be identified by looking at the way that it changes over time – moving from one target location to another. To recognize a continuous speech signal we must develop a method for comparing sequences of observations with stored patterns.

A fully function system will operate as follows:
The analog signal of this command will then be converted to a digital signal. After A/D conversion of the signal, software will process the signal and store it in memory. The stored information will then be compared to the information stored in a database of pre-recorded commands via a speech recognition algorithm. When a match is made a control signal will be issued to the output interface circuitry, which will control the appliances. This will occur in real time, optimized for minimum delay.

3.1.3 Block Diagram

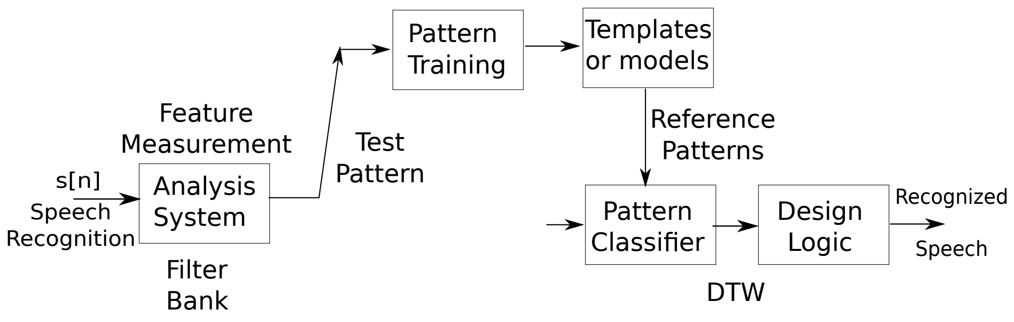


Figure 3.1: Block diagram for speech recognition

3.1.4 Distance between Two Sequences of Vectors

A classification of a spoken utterance would be easy if we had a good distance measure $D(X,W)$ at hand. To get a good distance measure, the distance measure must :

1. Measure the distance between two sequences of vectors of different length.
2. While computing the distance, find an optimal assignment between the individual feature vectors.
3. Compute a total distance out of the sum of distances between individual pairs of feature vectors.

3.1.5 Comparing the distance between two sequences of vectors of different length

In dynamic time warping method (DTW), when comparing sequences with different length, the sequence length is modified by repeating or omitting some frames, so that both sequences have the same length as shown figure 1 below. This modification of sequences is called time warping.

3.1.6 Linear Time Warping

As it can be seen from figure, the two sequences X and W consist of six and eight vectors, respectively. The sequence W was rotated by 90 degrees, so that the time index for this sequence runs from the bottom of the sequence to its top. The two sequences span a grid of possible assignments between the vectors. Each path through this grid (as the path shown in the figure) represents one possible assignment of the vector pairs. For example, the first vector of X is assigned the first vector of W, the second vector of X is assigned to the second vector of W, and so on. As an example the, let us assume that path P is given by the following sequence of time index pairs of the vector sequences.

$$P = \{(0,0),(1,1),(2,2),(3,2),(4,2),(5,3),(6,4),(7,4)\}$$

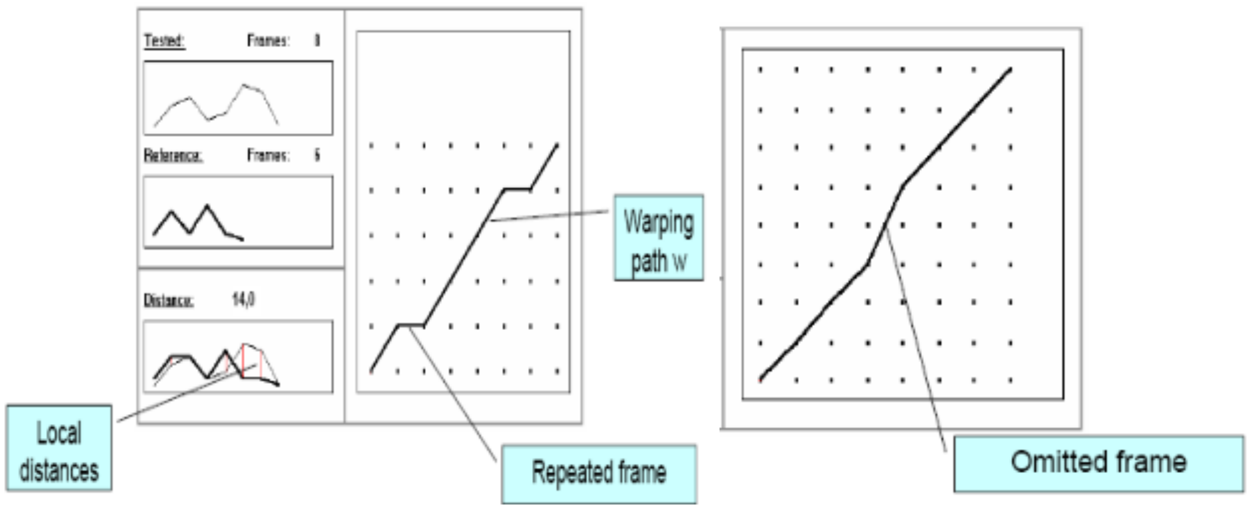


Figure 3.2: DTW time alignment

The length of path P is determined by the maximum of the number of vectors contained in X and W. The assignment between the time indices of W and X as given by P can be interpreted as "time warping" between the time axes of W and X. In our example, the vectors x_2, x_3 and x_4 were all assigned to w_2, thus warping the duration of w_2 so that it lasts three time indices instead of one. By this kind of time warping, the different lengths of the vector sequences can be compensated. And for the given path P, the distance measure between the vector sequences can be computed as the sum of the distances between the individual vectors.

3.1.7 Finding the optimal Path

Once we have the path, computing the distance becomes a simple task. DTW distance can be computed efficiently by using Bellmans principle of optimality. It states that If optimal path is the path through the matrix of grid points beginning at A and ending at B, and the grid point K is part of path, then the partial path from A to B is also part of optimal path[8]. From that, we can construct a way of iteratively finding our optimal path P.

According to this principle, it is not necessary to compute all possible paths P and corresponding distances to find the optimum path. Out of the huge number of theoretically possible paths, only a fraction is computed. To illustrate this concept further, we need to discuss what is called a local path alternative or local distance.

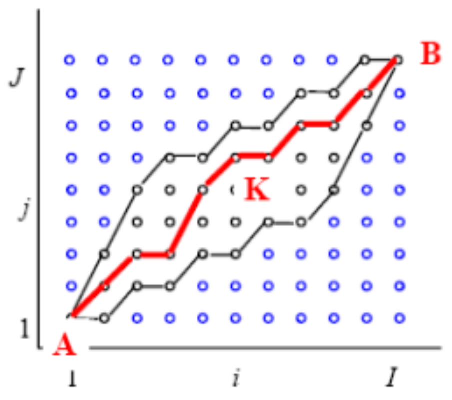


Figure 3.3: Optimal Path

3.1.8 Local distances

Since both sequences of vectors represent feature vectors measured in short time intervals, we can restrict the time warping to reasonable boundaries. The first vectors of X and W should be assigned to each other as well as their last vectors. For the time indices in between, we want to avoid any big leap backward or forward in time, but want to restrict the time warping just to the reuse of the preceding vector(s) to locally warp the duration of a short segment of speech signal. With these restrictions, we can draw a diagram of possible local path alternatives for one grid point and its possible predecessors. As we can see, a grid point (i, j) can have the following
Predecessors:

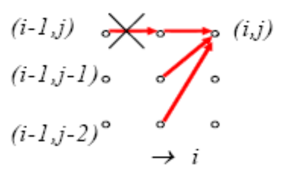


Figure 3.4: Local distance

$(i-1, j)$: horizontal local path

$(i-1, j-1)$: diagonal local path

$(i, j-1)$: vertical local path

$$A(i,j) = \partial(x_{i,}r_j) + min[A(i-1,j)*q, A(i-1,j-1), A(i-1,j-2)$$

All possible paths P, which we will consider as possible candidates for being the optimal path, can be constructed as a concatenation of the local path alternatives as described above. According to the local path alternatives diagram shown above, there are only three possible predecessor paths leading to a grid point (i,j): The partial paths from $(0,0)$ to the grid points $(i-1,j)$, $(i-1,j-1)$ and $(i,j-1)$). The (globally) optimal path from $(0,0)$ to grid point (i,j) can be found by selecting exactly the one path hypothesis among our alternatives which minimizes the accumulated distance $A(i,j)$ of the resulting path from $(0,0)$ to (i,j).
Starting from grid point $(0,0)$ to the vector distances defined by the grid point $(1,0)$ and $(0,1)$, we can compute $A(1,0)$ and $A(0,1)$. Now we look at the points which can be computed from the three points we just finished. For each of these points (i,j), we search the optimal predecessor point out of the set of possible predecessors. That way we walk through the matrix from bottom-left to top-right. Once we reached the topright corner of our matrix, the accumulated distance A (RefFrames, TestFrames) is the distance $D(W,X)$ between the vector sequences.

3.1.9 Optimizations

The major optimizations to the DTW algorithm arise from observations on the nature of good paths through the grid. These can be summarized as:

Monotonic condition: the path will not turn back on itself, both the i and j indexes either stay the same or increase, they never decrease.

Continuity condition: The path advances one step at a time. Both i and j can only increase by 1 on each step along the path.

Boundary condition: the path starts at the bottom left and ends at the top right.

Adjustment window condition: a good path is unlikely to wander very far from the diagonal. The distance that the path is allowed to wander is the window length r.

Slope constraint condition: The path should not be too steep or too shallow. This prevents very short sequences matching very long ones. The condition is expressed as a ratio n/m where m is the number of steps in the x direction and m is the number in the y direction. After m steps in x you must make a step in y and vice versa.

By applying these observations we can restrict the moves that can be made from any point in the path and so restrict the number of paths that need to be considered. For example, with a slope constraint of P=1, if a path has already moved one square up it must next move either diagonally or to the right.

The power of the DTW algorithm goes beyond these observations though. Instead of finding all possible routes through the grid which satisfy these constraints, the DTW algorithm works by keeping track of the cost of the best path to each point in the grid. During the match process we have no idea which path is the lowest cost path; but this can be traced back when we reach the end point.

3.1.10 The Weighting Function

A path through the grid is written in the paper as$F = c(1), c(2) \cdots c(K)$, the generalized element of the path is $c(k)$ and this consists of a pair of coordinates in the i (input) and j (stored) directions. The i coordinate of the kth path element is $i(k)$.

The weighting function $w(k)$ introduced into the overall distance measure is used to normalize for the path length. Two alternate weighting functions are presented: symmetric and asymmetric. Both functions are derived from the distance travelled (in grid units) in the last step of the path. The symmetric form combines the i and j directions while the asymmetric form uses just the i direction.

If the path has just made a diagonal step then i and j both increase by 1 and the symmetric w(k) = 1+1 = 2; the asymmetric w(k) = 1; The sum of this function over the length of the path gives a measure of how long the path is. This is used in normalizing the overall distance measures.

3.1.11 Endpoint detection

One issue that arises in isolated word recognition is separating speech from silence in the input signal. Each template and input sequence must correspond to a single word; the user is required to pause between words to make this segmentation possible.
One simple method of making a speech/no speech decision is to use a combination of zero crossings and RMS energy. Together these two features provide a reasonable separation of speech and silence since low energy speech (fricatives) tends to have high zero crossing rates and low ZCR speech (vowels) tend to have high energy. The speech signal is segmented into words before being passed to the front end of the recognition system.

3.1.12 Selecting Templates

So far I have assumed that a single recording of each word is used as the basis for the stored template in a DTW based recognizer. This approach will not be very robust since it takes no account of the variability of different utterances and does not ensure that the template is representative of the class as a whole. A number of methods have been proposed which seek to address these issues.

3.1.13 Continuous Speech

The DTW technique is obviously suited to isolated word recognition since the start and end of the words needs to be known before the match can proceed. Advanced versions of DTW were developed to match sequences of words in connected speech.

3.2 Mel Frequency Cepstrum Coefficients

The purpose of this module is to convert the speech waveform, using digital signal processing (DSP) tools, to a set of features (at a considerably lower information rate) for further analysis. This is often referred as the *signal-processing front end* . The purpose of this module is to convert the speech waveform, using digital signal processing (DSP) tools, to a set of features (at a considerably lower information rate) for further analysis. This is often referred as the *signal-processing front* *end*.

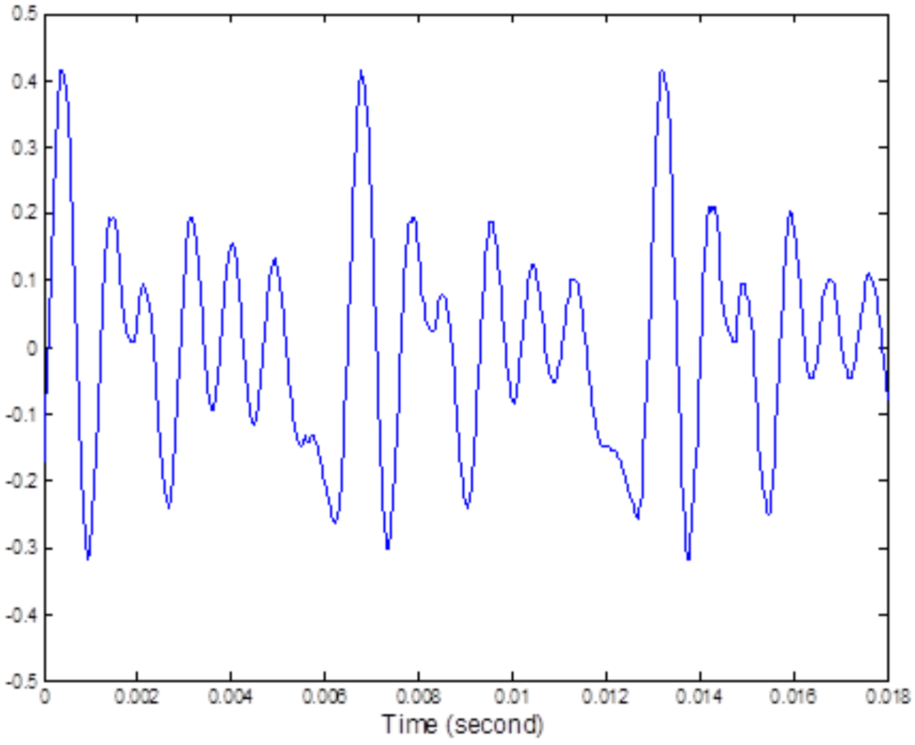


Figure 3.5: Example of Speech signal

A wide range of possibilities exist for parametrically representing the speech signal, such as Linear Prediction Coding (LPC), Mel-Frequency Cepstrum Coefficients (MFCC), and others. MFCC is perhaps the best known and most popular. MFCCs are based on the known variation of the human ears critical bandwidths with frequency, filters spaced linearly at low frequencies and logarithmically at high frequencies have been used to capture the phonetically important characteristics of speech. This is expressed in the *mel-frequency scale* , which is a linear frequency spacing below 1000 Hz and a logarithmic spacing above 1000 Hz.
The feature extraction is usually a non-invertible (lossy) transformation, Making an analogy with filter banks, such transformation does not lead to perfect reconstruction, i.e., given only the features it is not possible to reconstruct the original speech used to generate those features. Computational complexity and robustness are two primary reasons to allow loosing information. Increasing the

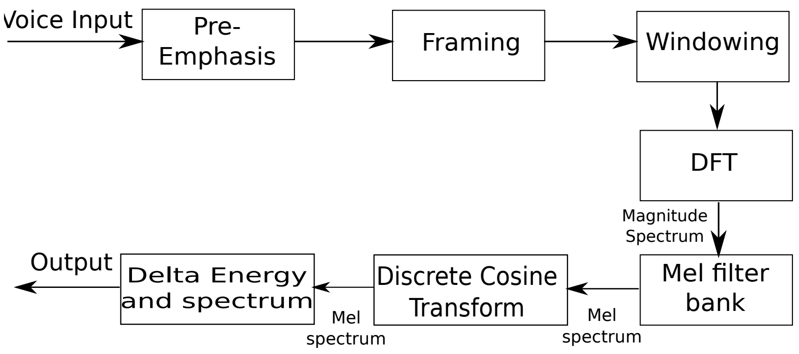


Figure 3.6: MFCC block diagram

accuracy of the parametric representation by increasing the number of parameters leads to an increase of complexity and eventually does not lead to a better result due to robustness issues. The greater the number of parameters in a model, the greater should be the training sequence.

3.2.1 Mel-frequency cepstrum coefficients processor

A block diagram of the structure of an MFCC processor is given in Figure 3.6. The speech input is typically recorded at a sampling rate above 10000 Hz. This sampling frequency was chosen to minimize the effects of *aliasing* in the analog-to-digital conversion. These sampled signals can capture all frequencies up to 5 kHz, which cover most energy of sounds that are generated by humans. As been discussed previously, the main purpose of the MFCC processor is to mimic the behavior of the human ears. In addition, rather than the speech waveforms themselves, MFFCs are shown to be less susceptible to mentioned variations.

Step 1: Preemphasis In processing electronic audio signals, pre-emphasis refers to a system process designed to increase (within a frequency band) the magnitude of some (usually higher) frequencies with respect to the magnitude of other (usually lower) frequencies in order to improve the overall signal-to-noise ratio by minimizing the adverse effects of such phenomena as attenuation distortion or saturation of recording media in subsequent parts of the system. That is the mirror of the de-emphasis. The whole system is called emphasis. The frequency curve is decided by special time constants.

$$Y[n] = X[n] - 0.95X[n-1]$$

In high speed digital transmission, pre-emphasis is used to improve signal quality at the output of a data transmission. In transmitting signals at high data rates, the transmission medium may introduce distortions, so pre-emphasis is used to distort the transmitted signal to correct for this distortion. When done properly this produces a received signal which more closely resembles the original or desired signal, allowing the use of higher frequencies or producing fewer bit errors.

Step 2: Framing In this step the continuous speech signal is blocked into frames of N samples, with adjacent frames being separated by M (M ¡ N). The first frame consists of the first N samples. The second frame begins M samples after the first frame, and overlaps it by N - M samples and so on. This process continues until all the speech is accounted for within one or more frames. Typical values for N and M are N = 256 (which is equivalent to 30 msec windowing and facilitate the fast radix-2 FFT) and M = 100.

Step 3: Hamming windowing Speech is non-stationary signal where properties change quite rapidly over time. This is fully natural and nice thing but makes the use of DFT or autocorrelation as a such impossible. For most phonemes the properties of the speech remain invariant for a short period of time (.5-100 ms). Thus for a short window of time, traditional signal processing methods can be applied relatively successfully. Most of speech processing in fact is done in this way: by taking short windows (overlapping possibly) and processing them. The short window of signal like this is called *frame*.

In implementational view the windowing corresponds to what is understoods in filter design as window-method: a long signal (of speech for instance or ideal impulse response) is multiplied with a window function of finite length, giving finite length weighted (usually) version of the original signal. In speech processing the shape of the window function is not that crucial but usually some soft window like Hanning, Hamming, triangle, half parallerogram, not with right angles. The reason is same as in filter design, sideband lobes as substantially smaller than in a rectangular window. Moreover in LPC-analysis (to be studied later on) the signal is presumed to be 0 outside the window, hence the rectangular window produces abrupt change in the signal, which usually distorts the analysis.

$$y_i(n) = x_i(n)w(n), 0 \leq n \leq N-1$$

The Hamming window has the form:

$$w(n) = 0.54 - 0.46cos\left(\frac{2\pi n}{N-1}\right), 0 \leq n \leq N-1$$

Step 4: Fast Fourier Transform The next processing step is the Fast Fourier Transform, which converts each frame of N samples from the time domain into the frequency domain. The FFT is a fast algorithm to implement the Discrete Fourier Transform (DFT), which is defined on the set of N samples xn, as follow:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-j2\pi kn/N}, k = 0, 1, 2, ..., N-1$$

In general $X_k s$ are complex numbers and we only consider their absolute values (frequency magnitudes). The resulting sequence X_k is interpreted as follow: positive frequencies. $0 \leq f < F_s/2$ correspond to values $0 \leq n \leq \frac{N}{2} - 1$, while negative frequencies $-F_s/2 < f < 0$ correspond to $N/2 + 1 \leq n \leq N-1$. Here, Fs denotes the sampling frequency.

A fast Fourier transform (FFT) is an algorithm to compute the discrete Fourier transform (DFT) and its inverse. Fourier analysis converts time (or space) to frequency and vice versa; an FFT rapidly computes such transformations by factorizing the DFT matrix into a product of sparse (mostly zero) factors. The DFT is obtained by decomposing a sequence of values into components of different frequencies. This operation is useful in many fields but computing it directly from the definition is often too slow to be practical. An FFT is a way to compute the same result more quickly: computing the DFT of N points in the naive way, using the definition, takes O(N2) arithmetical operations, while a FFT can compute the same DFT in only O(N log N) operations. The difference in speed can be enormous, especially for long data sets where N may be in the thousands or millions.

In practice, the computation time can be reduced by several orders of magnitude in such cases, and the improvement is roughly proportional to N / log(N). This huge improvement made the calculation of the DFT practical; FFTs are of great importance to a wide variety of applications, from digital signal processing and solving partial differential equations to algorithms for quick multiplication of large integers.

Step 5: Mel Filter Bank Processing The human auditory system doesnt interpret pitch in a linear manner. The human interpretation of the pitch reises with the frequency, which in some applications may be a unwanted Feature. To compensate for this the mel-scale was developed[11]. The mel-scale was developed by experimenting with the human ears interpretation of a pitch in 1940s. The sole purpose of the experiment were to describe the human auditory system on a linear scale. the pitch is linearly perceived in the frequency range 0-1000hz. Above 1000 hz, the scale becomes logarithmic. An approximated formula widely used for mel-scale is shown below:

$$F_{mel} = \frac{1000}{log(2)} \cdot [1 + \frac{F_{Hz}}{1000}]$$

where *Fmel* is the resulting frequency on the mel-scale measured in mels and FHz is the normal frequency measured in Hz.
With the mel-scale applied, coefficients from a LPC will be concentrated in the lower frequencies and only around the area perceived by humans as the pitch, which may result in a more precise description of a signal, seen from the perception of the human auditory system[13]. This is although not proved and it is only suggested that the mel-scale may have this effect. The mel-scale is, regardless of what have been said above, a widely used and effective scale within speech recognition, in which a speaker need not to be identified, only understood.

Step 6: Discrete Cosine Transform(Obtaining Cepstrums) This is the process to convert the log Mel spectrum into time domain using Discrete Cosine Transform (DCT). The result of the conversion is called Mel Frequency Cepstrum Coefficient. The set of coefficient is called acoustic vectors. Therefore, each input utterance is transformed into a sequence of acoustic vector.
Mel Frequency Cepstral Coefficients (MFCC) is usually derived using a filter bank, this is illustrated in figure. It has been found that the energy in a critical band of a particular frequency influence the human auditory systems perception. This critical band bandwidth varies with the frequency, where it is linear below 1 kHz and logarithmic above. Combining this with the mel scale, the distributions of these critical bands becomes linear. The critical band is a band pass filter, adjusted around the center frequency. Below 1 kHz critical bands are placed linear around

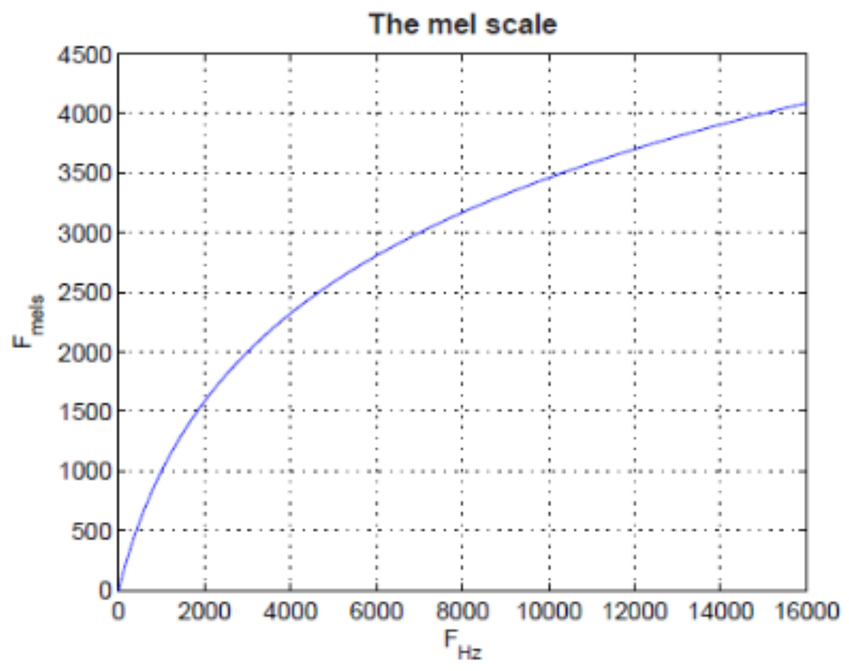


Figure 3.7: Mel Scale

100, 200, ... 1000 Hz. Above 1 kHz these bands are placed with the mel-scale. In the calculation of the MFCCs the total energy in each critical band is used.

The filter bank has a triangular band pass frequency response, and the spacing as well as the bandwidth is determined by a constant mel frequency interval. The number of mel spectrum coefficients, K, is typically chosen as 20. Note that this filter bank is applied in the frequency domain, thus it simply amounts to applying the triangle-shape windows as in the figure to the spectrum. A useful way of thinking about this mel-wrapping filter bank is to view each filter as a histogram bin (where bins have overlap) in the frequency domain[1]. Now we convert the log mel spectrum back to time. The result is called the mel frequency cepstrum coefficients (MFCC). The cepstral representation of the speech spectrum provides a good representation of the local spectral properties of the signal for the given frame analysis. Because the mel spectrum coefficients (and so their logarithm) are real numbers, we can convert them to the time domain using the Discrete Cosine Transform (DCT). Therefore if we denote those mel power spectrum coefficients that are the result of the last step are $\widetilde{S_0}, k = 0, 2, ..., K-1$ we can calculate the MFCC's, $\tilde{c}_n$, as

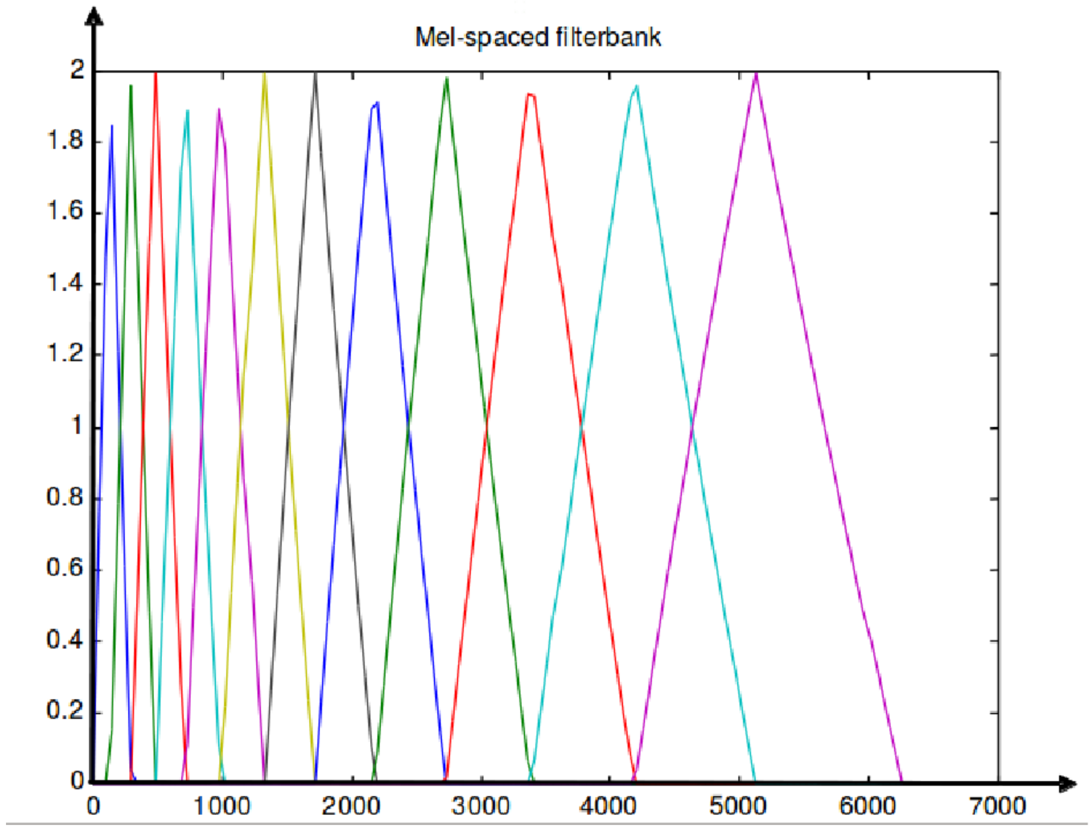


Figure 3.8: Mel-spaced filter bank

$$\tilde{c}_n = \sum_{k=1}^{K}(log\tilde{S}_k)cos\left[n(k-\tfrac{1}{2})\tfrac{\pi}{K}\right], n = 0, 1, ..., K-1$$

Note that we exclude the first component, $\tilde{c}_0$ from the DCT since it represents the mean value of the input signal.

3.3 Vector Quantization

The natural way of communication among human beings is through speech. Many human beings are exchanging the information through mobile phones as well as other communication tools in a real manner . The Vector Quantization (VQ) is the fundamental and most successful technique used in speech coding, speech recognition, and speech synthesis and speaker recognition. These techniques are applied firstly in the analysis of speech where the mapping of large vector space into a finite number of regions in that space. The VQ techniques are commonly applied to develop discrete or semi-continuous MFCC based speech recognition system[3]. In VQ, an ordered set of signal samples or parameters can be efficiently coded by matching the input vector to a similar pattern or codevector (codeword) in a predefined codebook. The VQ techniques are also known as data clustering methods in various disciplines. It is an unsupervised learning procedure widely used in many applications. The data clustering methods are classified as hard and soft clustering methods. In the hard clustering, each data point belongs to exactly one of the partitions in obtaining the disjoint partitioning of the data whereas each data point has a certain probability of belonging to each of the partitions in soft clustering. The parametric clustering algorithms are very popular due to its simplicity and scalability. The hard clustering algorithms are based on the iterative relocation schemes. The classical K-means algorithm is based on Euclidean distance and the Linde-Buzo-Gray (LBG) algorithm is based on the Itakura-Saito distance. The performance of vector quantization techniques depends on the existence of a good codebook of representative vectors. An efficient VQ codebook design algorithm is proposed known as Modified K-means LBG algorithm. This algorithm provides superior performance as compared to classical K-means algorithm and the LBG algorithm.

The main objective of data compression is to reduce the bit rate for transmission or data storage while maintaining the necessary fidelity of the data. The feature vector may represent a number of different possible speech coding parameters including linear predictive coding (LPC)coefficients, cepstrum coefficients. The VQ can be considered as a generalization of scalar quantization to the quantization of a vector. The VQ encoder encodes a given set of k-dimensional data vectors with a much smaller subset. The subset C is called a codebook and its elements Ci are called codewords, codevectors, reproducing vectors, prototypes or design samples. Only the index i is transmitted to the decoder. The decoder has the same codebook as the encoder, and decoding is operated by table look-up procedure. The commonly used vector quantizers are based on nearest neighbor called Voronoi or nearest neighbour vector quantizer. Both the classical K-means algorithm and the LBG algorithm belong to the class of nearest neighbor quantizers. A key component of pattern matching is the measurement of dissimilarity between two feature vectors. The measurement of dissimilarity satisfies three metric properties such as Positive definiteness property, Symmetry property and Triangular inequality property. Each metric has three main characteristics such as computational complexity, analytical tractability and feature evaluation reliability. The metrics used in speech processing are derived from the Minkowski metric. The Minkowski metric can be expressed as

$$D_p(X,Y) = \sqrt[p]{\sum_{i=1}^{k} |x^i - y^i|^p}$$

Where $X = \{x^1, x^2, ..., x^k\}$ and $Y = \{y^1, y^2, ..., y^k\}$ are vectors and p is the order of the metric. The City block metric, Euclidean metric and Manhattan metric are the special cases of Minkowski metric. These metrics are very essential in the distortion measure computation functions. The distortion measure is one which satisfies only the positive definiteness property of the measurement of dissimilarity. There were many kinds of distortion measures including Euclidean distance, the Itakura distortion measure and the likelihood distortion measure, and so on. The Euclidean metric is commonly used because it fits the physical meaning of distance or distortion. In some applications division calculations are not required. To avoid calculating the divisions, the squared Euclidean metric is employed instead of the Euclidean metric in pattern matching. The quadratic metric is an important generalization of the Euclidean metric. The weighted cepstral distortion measure is a kind of quadratic metric. The weighted cepstral distortion key feature is that it equalizes the importance in each dimension of cepstrum coefficients. In the speech recognition, the weighted cepstral distortion can be used to equalize the performance of the recognizer across different talkers. The Itakura-Saito distortion measure computes a distortion between two input vectors by using their spectral densities. The performance of the vector quantizer can be evaluated by a distortion measure D which is a non-negative cost
$D(X_j, \hat{X}_j)$ associated with quantizing any input vector X_j with a reproduction vector $\hat{X}_j$. Usually the Euclidean distortion measure is used. The performance of a quantizer is always qualified by an average distortion $D_v = E[D(X_j, \hat{X}_j)]$ between the input vectors and the final reproduction vectors, where E represents the expectation operator. Normally, the performance of the quantizer will be good if the average distortion is small. Another important factor in VQ is the codeword search problem. As the vector dimension increases accordingly the search complexity increases exponentially, this is a major limitation of VQ codeword search. It limits the fidelity of coding for real time transmission. A full search algorithm is applied in VQ encoding and recognition. It is a time consuming process when the codebook size is large. In the codeword search problem, assigning one codeword to the test vector means the smallest distortion between the codeword and the test

vector among all codewords. Given one codeword Ct and the test vector X in the k-dimensional space, the distortion of the squared Euclidean metric can be expressed as follows:
There are three ways of generating and designing a good codebook namely the random method, the pair-wise nearest neighbor clustering and the splitting method. There are three major procedures in VQ, namely codebook generation, encoding procedure and decoding procedure.

3.3.1 Feature Matching

The problem of speech recognition belongs to a much broader topic in scientific and engineering so called pattern recognition. The goal of pattern recognition is to classify objects of interest into one of a number of categories or classes. The objects of interest are generically called patterns and in our case are sequences of acoustic vectors that are extracted from an input speech using the techniques described in the previous section. The classes here refer to different words. Since the classification procedure in our case is applied on extracted features, it can be also referred to as feature matching[7].
Furthermore, if there exists some set of patterns that the individual classes of which are already known, then one has a problem in supervised pattern recognition. These patterns comprise the training set and are used to derive a classification algorithm. The remaining patterns are then used to test the classification algorithm; these patterns are collectively referred to as the test set. If the correct classes of the individual patterns in the test set are also known, then one can evaluate the performance of the algorithm.
The state-of-the-art in feature matching techniques used in speech recognition include Dynamic Time Warping (DTW), Hidden Markov Modeling (HMM), and Vector Quantization (VQ). In this project, the VQ approach will be used, due to ease of implementation and high accuracy. VQ is a process of mapping vectors from a large vector space to a finite number of regions in that space. Each region is called a cluster and can be represented by its center called a codeword. The collection of all codewords is called a codebook.
In the figure, only two words and two dimensions of the acoustic space are shown. The circles refer to the acoustic vectors from the word 1 while the triangles are from the word 2. In the training phase, using the clustering algorithm, a speaker-specific VQ codebook is generated for each known speaker by clustering his/her training acoustic vectors. The result codewords (centroids) are shown

in Figure by black circles and black triangles for word 1 and 2, respectively. The distance from a vector to the closest codeword of a codebook is called a VQ-distortion. In the recognition phase, an input utterance of an unknown voice is vector-quantized using each trained codebook and the total VQ distortion is computed. The word corresponding to the VQ codebook with smallest total distortion is identified as the word spoken in the input utterance.

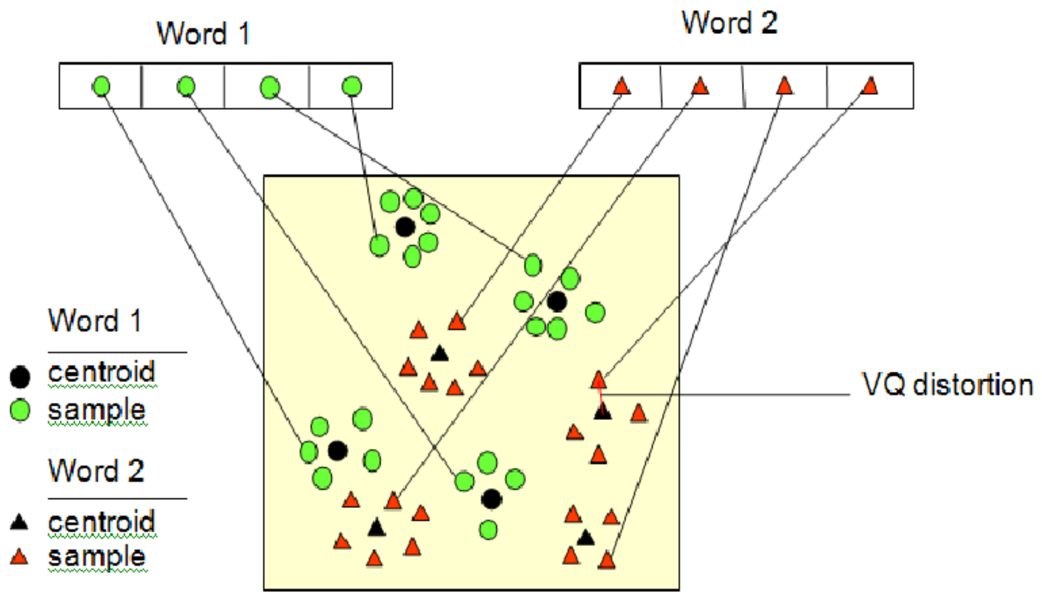


Figure 3.9: VQ codebook formation

3.3.2 Clustering the Training Vectors

After the acoustic vectors extracted from input speech of each speaker provide a set of training vectors for that word. As described above, the next important step is to build a speaker-specific VQ codebook for each speaker using those training vectors. There is a well-know algorithm, namely LBG algorithm[9], for clustering a set of L training vectors into a set of M codebook vectors. The algorithm is formally implemented by the following recursive procedure:

1. Design a 1-vector codebook; this is the centroid of the entire set of training vectors (hence, no iteration is required here).

2. Double the size of the codebook by splitting each current codebook yn according to the rule where n varies from 1 to the current size of the codebook, and is a splitting parameter (we choose =0.01).

3. Nearest-Neighbor Search: for each training vector, find the codeword in the current codebook that is closest (in terms of similarity measurement), and assign that vector to the corresponding cell (associated with the closest codeword).

4. Centroid Update: update the codeword in each cell using the centroid of the training vectors assigned to that cell.

5. Iteration 1: repeat steps 3 and 4 until the average distance falls below a preset threshold.

6. Iteration 2: repeat steps 2, 3 and 4 until a codebook size of M is designed.

Intuitively, the LBG algorithm designs an M-vector codebook in stages. It starts first by designing a 1-vector codebook, then uses a splitting technique on the codewords to initialize the search for a 2-vector codebook, and continues the splitting process until the desired M-vector codebook is obtained.

3.3.3 Block Diagram

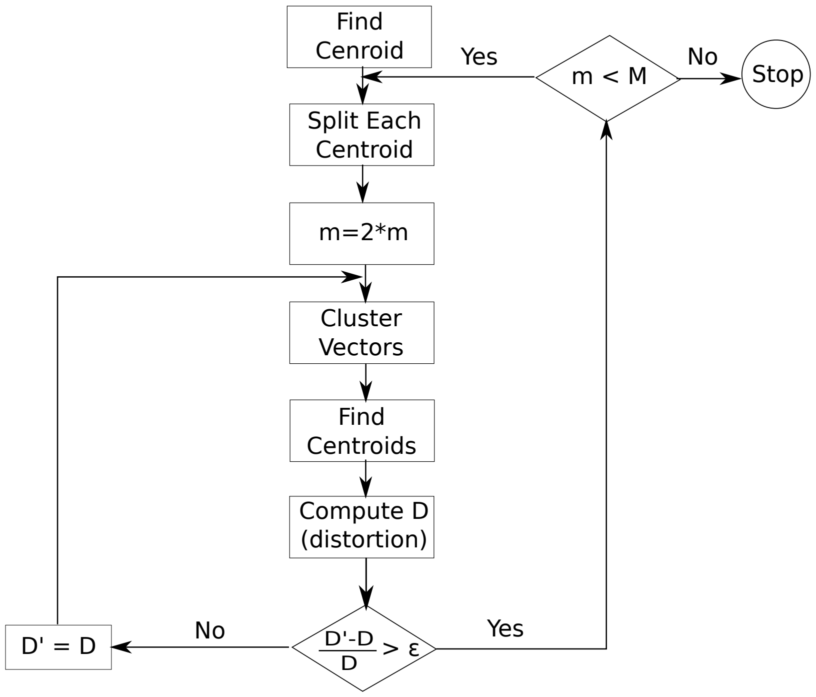


Figure 3.10: VQ flow chart

Chapter 4

Applications

4.1 Accessing videos using speech recognition

Speech recognition field is one of the most challenging fields that have faced the scientists from long time. The complete solution is still far from reach. The efforts are concentrated with huge funds from the companies to different related and supportive approaches to reach the final goal. Then, apply it to the enormous applications who are still waiting for the successful speech recognizers that are free from the constraints of speakers, vocabularies and environment. This task is not an easy one due to the interdisciplinary nature of the problem and as it requires speech perception to be implied in the recognizer (Speech Understanding Systems) which in turn strongly points to the use of intelligence within the systems. The bare techniques of recognizers (without intelligence) are following wide varieties of approaches with different claims of success by each group of authors who put their faith in their favorite way. The entire process of speech recognition can be broadly split into two subsequent phases -the training phase and the testing phase.

4.1.1 Basic Structure of Speech Recognition System

An Automatic Speech Recognition (ASR) engine tries to imitate the human auditory system. Speech recognition boils down to a matching problem matching the input speech signal to the reference speech signals stored in our brain or on a machine, as is the case for human and machine speech recognition, respectively. So, most of the latest speech recognition engines involve 2 parts training and recognition. Vector Quantization is a process of mapping vectors from a large vector space to a finite number of regions in that space. Each region is called a cluster and can be represented by its center called a codeword. The collection of all code words is called a codebook . Mapping or clustering the feature vectors can be done a number of clustering algorithms like the LBG algorithm or the K-means algorithm[2].

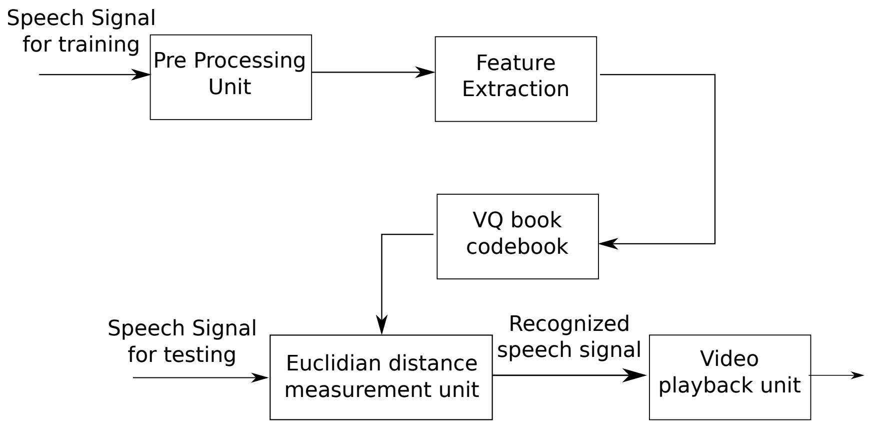


Figure 4.1: Video accessing block diagram

4.1.2 Pre-Processing Unit

This unit is a part of the training module of the application. Input training signals are preprocessed here so that they can be further used to extract the features.
The steps involved in pre processing are:

1. Segmentation and pre emphasis

   The speech signal is first segmented into lengths of 10/20 ms segments.Then it is subjected to pre emphasis. The pre-emphasizer is used to spectrally flatten the speech signal. This is usually done by a high pass filter.

2. Frame blocking and windowing

   The concept of short time analysis is fundamental to most speech analysis techniques. The assumption made is that, over a long interval of time, speech waveform is not stationary but that, over a sufficiently short time interval say about 10- 30msec, it can be considered stationary. This is due to that fact that the rate at which the spectrum of the speech signals changes is directly dependant on the rate of movement of the speech articulators. Since this is limited by physiological constraints, most speech analysis systems operate at uniformly spaced time intervals or frames of typical duration 10- 30 msec. In frame blocking, the continuous speech signal is blocked into frames of N Samples , with adjacent frames being separated by M (M ¡ N).The next thing to do is to apply a window to each frame in order to reduce signal discontinuity at either end of the block. The concept here is to minimize the spectral distortion by using the window to taper the signal to zero at the beginning and end of each frame. In other words, when we perform Fourier Transform, it assumes that the signal repeats, and the end of one frame does not connect smoothly with the beginning of the next one. This introduces some glitches at regular intervals. So we have to make the ends of each frame smooth enough to connect with each other. This is possible by a processing called Windowing. In this process, we multiply the given signal (frame in this case) by a so called Window Function. A commonly used window is the Hamming window.

3. FFT and Magnitude Response Extraction

   The next processing step is the Fast Fourier Transform, which converts each frame of N samples from the time domain into the frequency domain. The FFT is a fast algorithm to implement the Discrete Fourier Transform (DFT) which is defined on the set of N samples X_n.

$$X_n = \sum_{k=0}^{N-1} x_k e^{-\frac{\pi jkn}{N}}, n = 0, 1, 2, ..., N-1$$

   After the frames are converted to frequency domain from time domain using Fast Fourier Transform, we need to extract the magnitude response of the signals. This is so because while processing the speech signals we dont require the phase response of the signals. Thus we obtain the magnitude response of the speech signal.

4.1.3 Feature Extraction Unit

The purpose of this unit is to convert the speech waveform to a set of features for further analysis. This is often referred as the *signal-processing front end*. This method consists of two parts: the cepstrum calculation and a method called Mel scaling.

MFCC:

MFCCs are commonly calculated by first taking the Fourier transform of a windowed excerpt of a signal and mapping the powers of the spectrum obtained above onto the Mel scale, using triangular overlapping windows. Next the logs of the powers at each of the Mel frequencies are taken, Direct Cosine Transform is applied to it (as if it were a signal). The MFCCs are the amplitudes of the resulting spectrum. This procedure is represented step -wise in the figure below:

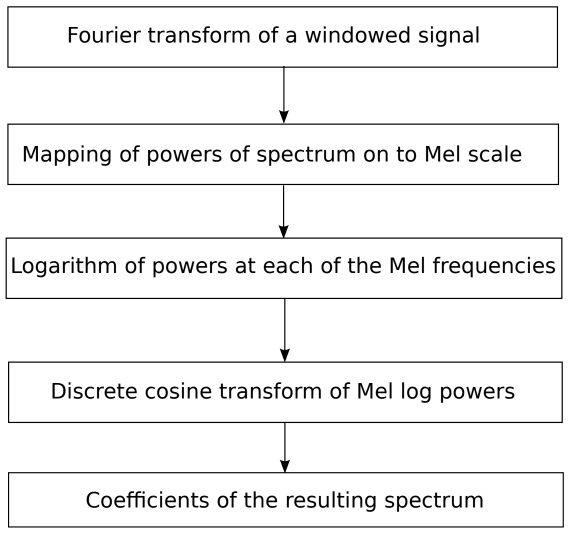

Fourier transform of a windowed signal
Mapping of powers of spectrum on to Mel scale
Logarithm of powers at each of the Mel frequencies
Discrete cosine transform of Mel log powers
Coefficients of the resulting spectrum

Figure 4.2: Procedure for forming MFCC

Codebook Generation Unit:

The problem of speech recognition belongs to a much broader topic in scientific and engineering so called pattern recognition. The goal of pattern recognition is to classify objects of interest into one of a number of categories or classes. The objects of interest are generically called patterns and in our case are sequences of acoustic vectors that are extracted from an input speech using the techniques described above. The classes here refer to different words. Since the classification procedure in our case is applied on extracted features, it can be also referred to as feature matching. Furthermore, if there exist some set of patterns that the individual classes of which are already known, then one has a problem in supervised pattern recognition. The remaining patterns are then used to test the classification algorithm, these patterns are collectively referred to as the test set. If the correct classes of the individual patterns in the test set are also known, then one can evaluate the performance of the algorithm. The VQ approach is used, due to ease of implementation and high accuracy. VQ is a process of mapping vectors from a large vector space to a finite number of regions in that space. Each region is called a cluster and can be represented by its center called a codeword. The collection of all code words is called a codebook.

Euclidean Distance Measurement Unit:

In this unit we would calculate the Euclidean distance between the test pattern and the codewords which are generated and recorded in the codebook. The codeword which has the minimum Euclidean distance will be the recognized word and the recognized word will be passed on to the video playback unit.

Video Playback Unit:

In this unit we get an input as a recognized word. This unit will look for a video named with the recognized word in the directory and will then play that video in the media player.

4.2 Speaker Recognition

Speaker recognition is the process of automatically recognizing who is speaking on the basis of individual information included in speech waves. This technique makes it possible to use the speaker's voice to verify their identity and control access to services such as voice dialing, banking by telephone, telephone shopping, database access services, information services, voice mail, security control for confidential information areas, and remote access to computers.

4.2.1 Principles of Speaker Recognition

Speaker recognition can be classified into identification and verification. Speaker identification is the process of determining which registered speaker provides a given utterance. Speaker verification, on the other hand, is the process of accepting or rejecting the identity claim of a speaker[6].

Figure shows the basic structures of speaker identification and verification systems. The system that we will describe is classified as text-independent speaker identification system since its task is to identify the person who speaks regardless of what is saying. At the highest level, all speaker recognition systems contain two main modules: feature extraction and feature matching. Feature extraction is the process that extracts a small amount of data from the voice signal that can later be used to represent each speaker. Feature matching involves the actual procedure to identify the unknown speaker by comparing extracted features from his/her voice input with the ones from a set of known speakers.

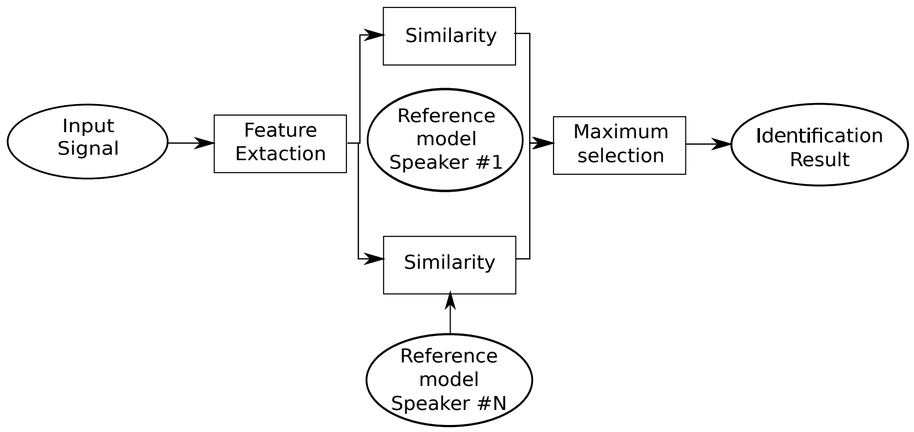


Figure 4.3: Speaker Identification

Now, I will take you through each system one by one. Feature extraction as the name suggest refers to extracting the important features of the speech signal for speech recognition. The analogue speech is converted into digital format by the microphone itself. After the speech is in digital format, it becomes easier to process. The first few values of the speech signal are taken into consideration for further processing. This is because after continuous evaluation it was noticed that all the major differences in the speech vectors are represented by the initial values. Once, we have the extracted features which are used further. We classify these into different classes; each class represents a different speaker. Each speaker or class is assigned a class number or speaker number, which is called as the reference model. These reference models are used while identifying the speaker. Now whichever class or speaker has the maximum matching with the input that reference model is chosen and the speaker ID corresponding to the reference model is shown.

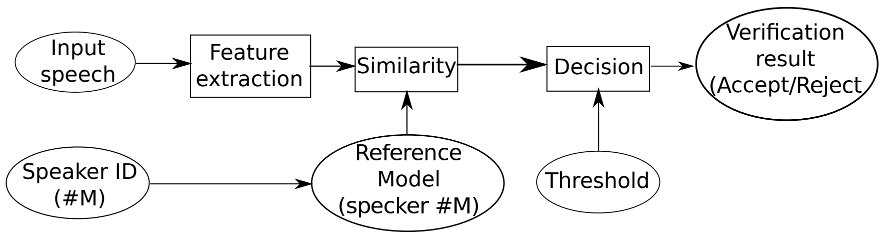


Figure 4.4: Speaker verification

For Feature Verification, the input speech and the speaker ID is passed to the system. The system checks for the similarity of the voice based on quality/duration/loudness/pitch[12]. The extracted features from the input are then compared to the models. A threshold has been set for the maximum allowable variation in the features, and a decision is made. If the value exceeds the threshold value the decision is made as rejection, i.e. the input feature does not match the trained features and no such speaker has been listed in their database. If the value is within the threshold limits the decision is made as accepted and the speaker details are flashed on the output window.

Speaker recognition is a difficult task. Automatic speaker recognition works based on the premise that a persons speech exhibits characteristics that are unique to the speaker. However this task has been challenged by the highly variant of input speech signals. The principle source of variance is the speaker himself/herself. Speech signals in training and testing sessions can be greatly different due to many facts such as people voice change with time, health conditions (e.g. the speaker has a cold), speaking rates, and so on. There are also other factors, beyond speaker variability, that present a challenge to speaker recognition technology. Examples of these are acoustical noise and variations in recording environments (e.g. speaker uses different telephone handsets).

Chapter 5

Simulations and Results

5.1 Speaker Recognition System

This is the graphical user interface (GUI) menu that we see after initializing the program:

- We can add a new sound or voice to the database.
- The speaker can be identified if his information is present in the database.
- We can retrieve the database info.
- The database can also be deleted.

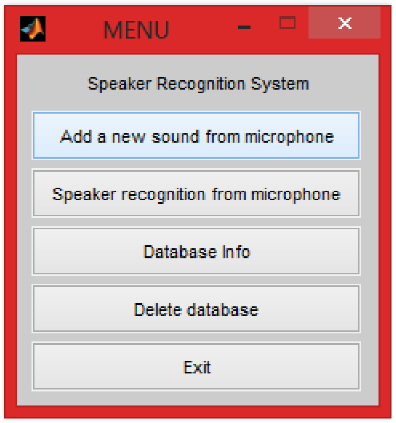
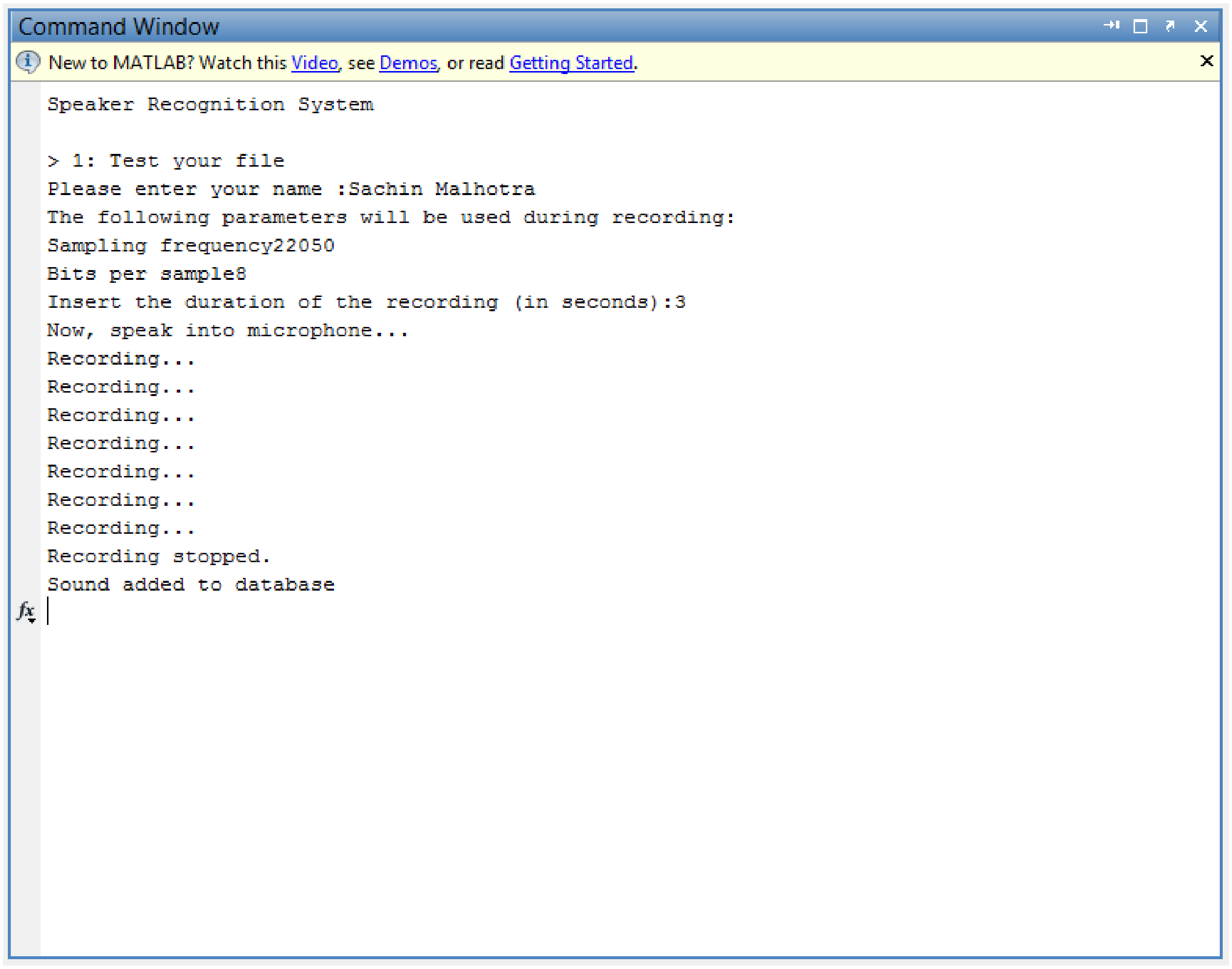

MENU
Speaker Recognition System
Add a new sound from microphone
Speaker recognition from microphone
Database Info
Delete database
Exit

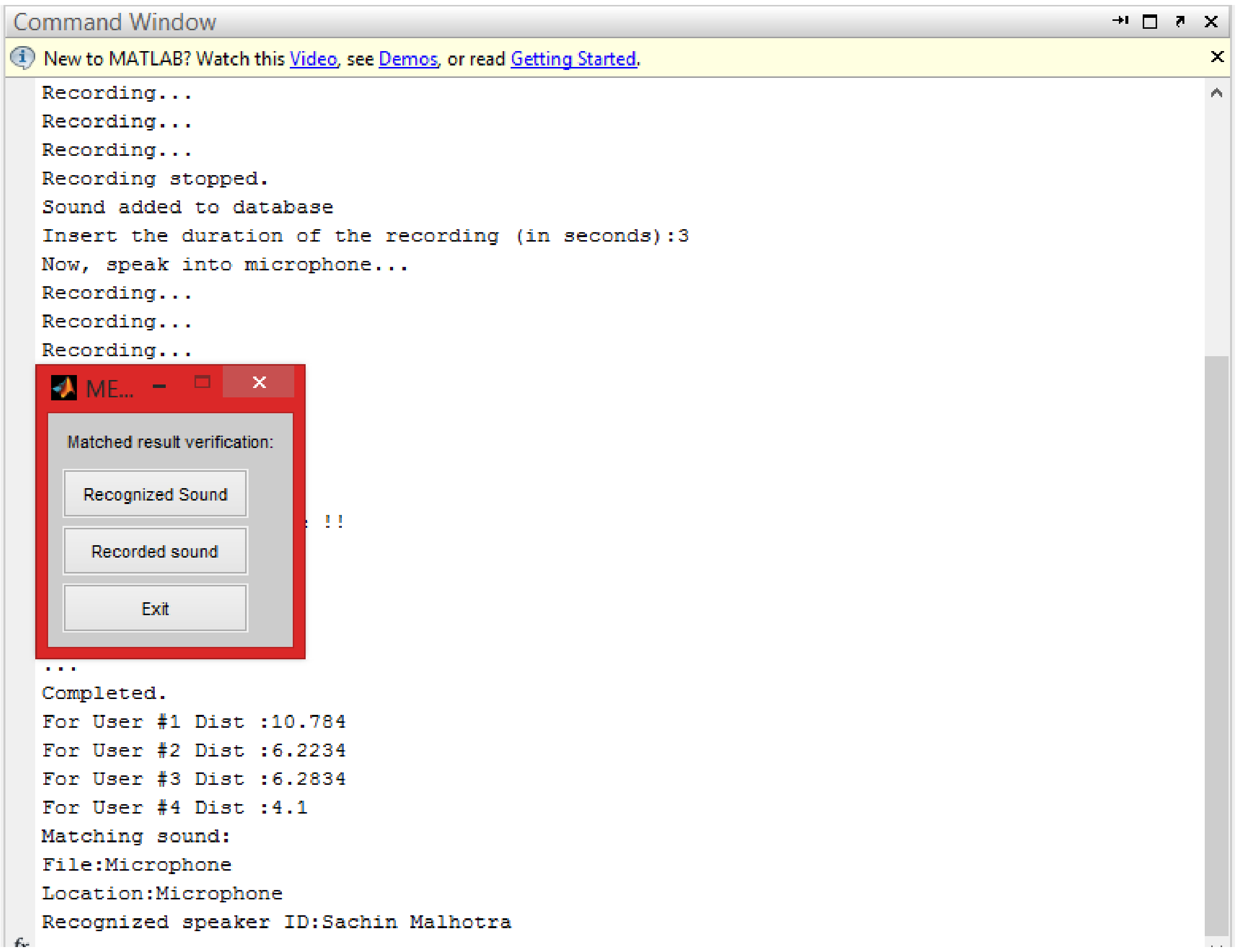
Figure 5.1: Speaker recognition menu

This is the command window that we see when we have to add a new speakers information to the system. It requires the speaker to enter his/her name and the duration of the speech to be recorded. When the sound has been added to the database a message is displayed in the command window. It gives the distance of the test sound vector from the trained vectors and displays the recognized speaker.


Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.

```
Speaker Recognition System

> 1: Test your file
Please enter your name :Sachin Malhotra
The following parameters will be used during recording:
Sampling frequency22050
Bits per sample8
Insert the duration of the recording (in seconds):3
Now, speak into microphone...
Recording...
Recording...
Recording...
Recording...
Recording...
Recording...
Recording...
Recording stopped.
Sound added to database
```

Figure 5.2: initializing speaker recognizing system

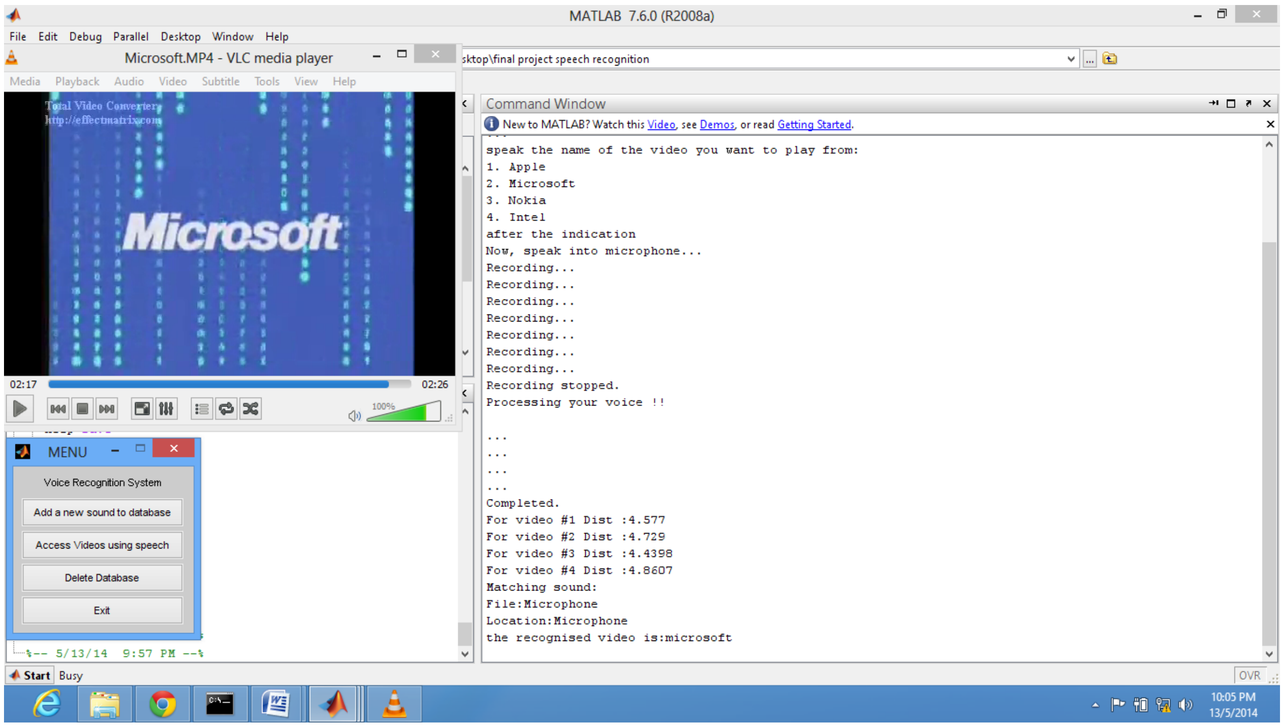

Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
ME...
Matched result verification:
Recognized Sound
Recorded sound
Exit

```
Recording...
Recording...
Recording...
Recording stopped.
Sound added to database
Insert the duration of the recording (in seconds):3
Now, speak into microphone...
Recording...
Recording...
Recording...

                       !!

...
Completed.
For User #1 Dist :10.784
For User #2 Dist :6.2234
For User #3 Dist :6.2834
For User #4 Dist :4.1
Matching sound:
File:Microphone
Location:Microphone
Recognized speaker ID:Sachin Malhotra
```

Figure 5.3: Command window recognizing speaker

5.2 Accessing Videos using Speech Recognition

This is the command window that we see when the voice recognition system is initialized. It prompts the user to choose from the given list of videos. The user can access the videos from the list through his voice. It then returns the distance of the users voice from the trained set. If the information is matched with the trained set, then the corresponding video is played through VLC player.


MATLAB 7.6.0 (R2008a)
File Edit Debug Parallel Desktop Window Help
Microsoft.MP4 - VLC media player
Media Playback Audio Video Subtitle Tools View Help
Microsoft
02:17
02:26
sktop\final project speech recognition
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
MENU
Voice Recognition System
Add a new sound to database
Access Videos using speech
Delete Database
Exit
Start Busy
OVR
10:05 PM
13/5/2014

```
speak the name of the video you want to play from:
1. Apple
2. Microsoft
3. Nokia
4. Intel
after the indication
Now, speak into microphone...
Recording...
Recording...
Recording...
Recording...
Recording...
Recording...
Recording...
Recording stopped.
Processing your voice !!

...
...
...
...
Completed.
For video #1 Dist :4.577
For video #2 Dist :4.729
For video #3 Dist :4.4398
For video #4 Dist :4.8607
Matching sound:
File:Microphone
Location:Microphone
the recognised video is:microsoft
```

```
%-- 5/13/14  9:57 PM --%
```

Figure 5.4: Video accessing system

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

On implementing the system for speech recognition and speaker recognition, we can assert with confidence that such systems can be used for a variety of applications. For instance speech recognition can be used to access videos i.e. Hands free accessing videos from a system. This system can be used for a variety of other applications such as in cars for assisting the driver for maneuvering, picking up a call, changing songs via voice while driving and many others to name a few.
Unique identification of voice is often of utmost importance in many sectors of industry. The speaker recognition system can be used for various security applications such as password authentication of bank account or homes or even high profile offices. The algorithms used in this project are uniformly accepted and recognized and are used for practical implementation of such systems. The market for voice implemented systems is continuously growing and need for systems like these are also increasing. Our recognition engines can be used by the common people without any difficulties due to the user friendly interface and the use of Graphic User Interface.
There is always a need to modify and add features according to the growing technology. We know there is profitability for modifying the technology to compete the market. Even though there are certain disadvantages and limitations to these systems due to lack of funds and limited scope and time available to us, they can be overcome by making certain changes and enhancements to our current system.

6.2 Future Scope

There are certain enhancements which would increase the efficiency of these recognition systems. Firstly, the accuracy of these systems can be increased by implementing and training in noisy and unfavorable environments. This would make the system more reliable and can be used in any conditions. Secondly, we can implement this system using a more practical algorithm know as Hidden Markov Model (HMM). Further these speaker recognition systems can be used to implement the following practical applications such as defense and security can be used to detect a suspect while searching for his voice. It can also be used in personalization as intelligent answering machine or voice-web/device. We can also use for speech data management such as voice mail browsing or searching audio archives. These additions would make the system more complete and more capable of being able to work in any kind of environment.

References

[1] Sankaran Panchapagesan, “Frequency Warping by Linear Transformation of Standard MFCC”.

[2] Samudravijaya K,Speech and Speaker Recognition: A Tutorial

[3] Vincent FONTAINE, Henri LEICH , Jean HENNEBERT, Influence of Vector Quantization on Isolated Word Recognition

[4] Lawrence Rabiner, Biing-Hwang Juang ,Fundamental of Speech recognition

[5] Lawrence R. Rabiner, Ronald W. Schafer, Introduction to Digital Speech Processing

[6] J o s e p h P . C a mp b e l l , J r., SPEAKER RECOGNITION

[7] Dipmoy Gupta, Radha Mounima C.,Navya Manjunath, Manoj PB, Isolated Word Speech Recognition Using Vector Quantization (VQ), IJAR Volume 2, Issue 5, May 2012

[8] A.Akila , Dr.E.Chandra, Slope Finder A Distance Measure for DTW based Isolated Word Speech Recognition, IJECS Volume 2 Issue 12, Dec. 2013, Page No.3411-3417

[9] Sunil Arya, David M. Mount, Algorithms for Fast Vector Quantization

[10] Balwant A. Sonkamble, D. D. Doye, Speech Recognition Using Vector Quantization through Modified K-meansLBG Algorithm ISSN 2222-1719 (Paper) ISSN 2222-2863 (Online) Vol 3, No.7, 2012

[11] Dimitrios Dimitriadis, Alexandros Potamianos, On the Effects of Filterbank Design and Energy Computation on Robust Speech Recognition, IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING, VOL. 19, NO. 6, AUGUST 2011

[12] Tomi Kinnunen, Md. Jahangir Alam, Pavel Matejka, Patrick Kenny, Frequency Warping and Robust Speaker Verication: A Comparison of Alternative Mel-Scale Representations

[13] S. Umesh, L. Cohen, and D. Nelson, Frequency Warping and the Mel Scale, IEEE SIGNAL PROCESSING LETTERS, VOL. 9, NO. 3, MARCH 2001