CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

Voice command matching system is very useful in many applications and environments in our daily life. Applying voice control technology will help a lot in enhancing the security for certain situations such as using it for voice authentications. A different aspect of VCMS could be used to facilitate people with functional disabilities like blindness, thereby making their daily routine easier. With their voice, they could operate the computer in their own language. This leads to the discussion about intelligent human-computer interaction where these operations can be made available for common man.

Nowadays, people want to access computers without using traditional keyboard, mouse etc. What they expect is computer should be able to meet their needs. The use of human voice has significant potential to enhance the quality of everyday human-computer interactions. The primary obstacle of integrating voice commanding movements into today's interfaces is the availability of a reliable, low cost voice commanding system.

1.1.1 VOICE RECOGNITION

Voice recognition works by analyzing the features of speech that differ between individuals. Everyone has a unique pattern of speech stemming from their anatomy (the size and shape of the mouth and throat) and behavioral patterns (their voice's pitch, their speaking style, accent, and so on).

The applications of voice recognition are markedly different from those of speech recognition. Most commonly, voice recognition technology is used to verify a speaker's identity or determine known and unknown speaker's identity. Speaker verification and speaker identification are both common types of voice recognition.

Voice recognition can refer to:

- Speaker recognition, determining *who* is speaking
- Speech recognition, determining *what* is being said

1.1.2 TEXT INDEPENDENT SYETEM

Text-independent systems do not require a person to speak a specific pass phrase during enrollment and verification. The base voiceprint and sample voiceprint can be obtained in the background-even without the person's knowledge (ideals for forensic applications).

1.1.3 VOICE COMMAND MATCHING SYSTEM

Voice Command Matching System uses audio signal to execute the computer command. User of this system can access function commands of computer, such as opening notepads, opening Facebook, shutting down computer, etc. with human voice. People with disabilities that prevent them from typing have also adopted this system. If a user has lost the use of his hands, or for visually impaired users when it is not possible or convenient to use a Braille keyboard, the systems allow control of many computer tasks using human voice.

The main intention of developing this system is to operate computer with human voice without language barrier. First of all, this system has standardized way to operate commands, then after running it in user’s environment, s/he can modify these operating command as per their suite and in their own language.

1.2 MOTIVATION

The motivation about this project came from subject Digital Signal Processing. We, as a group, find it extremely interesting topic and the project itself involves the use of Signal Processing to a great extent. The purpose of VCMS is to make the system operate by voice. This system is a text independent system.

More over the concept of this project seemed very exciting and mind catching. Our basic idea is to develop some sort of computer command invoking system which is voice driven. A person interacting with our system would not need to use his hands for execution of commands.

1.3 STATEMENT OF PROBLEMS

In our daily life, using computer is very important in order to improve the quality of our life but such system is not friendly for those physically disabled user. To assist them, this project is proposed in order to create a system that can be execute computer command by using their voice. Those physically disabled users can easily operate the computer without touching all the computer.

Recently there are systems which can't exactly fulfill the following activities:

- Perform computer task using custom voice,
- Away from language barrier i.e. language independent,
- Must be operated in same environment setup condition with optimum efficiency,
- The general system should be modified able in own way the user wants, etc.

But, this system is totally not suitable for those who are experience the mute problem. Beside this most of the voice commanding system is developed for specific language by converting the language into text.

Generally, computer doesn't understand all the languages even it understand few sorts of languages. Basically, people usually want to operate computer in their language favorable to them.

1.4 OBJECTIVE

A. To develop voice command matching application.

1.5 APPLICATION

VCMS can be used in various fields where physical computer interactions can be minimized. For instance: For people with disabilities, the system allows personal voice to control and assist many computer tasks.

1.6 SCOPE AND LIMITATION

The main scope for this project is to create a user friendly GUI to perform the voice commanding. The GUI will consist of several functions and buttons of operation regarding to the voice command analysis. Users just need to speak to perform certain job. The GUI will display the desire result according to what its task is and have the ability to save the result.

1. Recording Tool with Playback Ability.
2. Saving file
3. Perform the respective computer task using voice command

No voice recognition system is totally perfect; several factors can reduce accuracy. Some of these factors are issues that continue to improve as the technology improves. Others can be lessened, if not completely corrected, by the user.

- Even the best speech recognition systems sometimes make errors. If there is noise or some other sound in the room (e.g. the television or a kettle boiling), the errors will increase.
- Speech Recognition works best if the microphone is close to the user (e.g. in a phone, or if the user is wearing a microphone). More distant microphones (e.g. on a table or wall) will tend to increase the number of errors.

1.7 STRUCTURE OF REPORT

This paper describes an implementation of voice command matching system. The report is divided into six parts.

First part, Chapter 1, deals with the introduction of Voice Recognition, Voice Command Matching System, etc.

Chapter 2 deals with literature review while Chapter 3 & 4 deals with Methodology & Project Development Life Cycle respectively.

Chapter 5 deals with Testing & Result Analysis. And the last chapter ends with a conclusion of the achieved results.

In the appendixes, Project Management; Use Case Diagram; State Diagram; Flowchart; NAudio Framework & Screenshots of the system shown.

CHAPTER 2

LITERATURE REVIEW

In previous days, similar projects were done mostly using text dependent system, Automatic Speech Recognition (ASR). In those systems, while extracting features from speech, it becomes difficult to recognize correct word due to noise and other environmental conditions. Windows speech recognition is efficient but it is like one way communication. When words are spoken, processing is done and reply is given by performing task or opening application. It is hardware or software response instead of voice. It is necessary to get voice feedback for the command given by user for any user friendly application. Noises, distortions, and unforeseen speakers seldom cause difficulty for human to understand speech signals whereas they seriously degrade performances of ASR systems. In ASR, voice should be in English language and should be meaningful. It detects character from speech and checks syntax. Finally, ASR executes the respective task. ASR wasn't complete in itself, because of several restrictions. It was only for English language with syntactically correct. In reality, users favor to work in their own languages as far as possible.

There are number of limitations like environment issues, personal characteristics and conditions.

All the projects are related to recognition of voice what they speak and identification of speakers by their voice. Generally, in this system there is either Text-to speech (TTS) or Speech-to Text (STT) recognition.

A *text-independent speaker recognition uses combined LPC and MFC coefficients* [1]. It was very simple approach to text independent recognition where the recognition is performed by using both LPC and MFC coefficients in parallel and the results of both methods are combined for best matching of the speaker. The system used ANN for classification. Our system is also based on text-independent & we calculated MFC coefficients so this paper was indeed best for such system.

A *feature extraction using MFCC* [2]. It was a new purpose of working with MFCC by using it for Hand gesture recognition. The objective of using MFCC for hand gesture recognition is to explore the utility of the MFCC for image processing. Above system is based on converting the hand gesture into one dimensional (1-D) signal and then extracting first 13MFCCs from the converted 1-D signal. Classification is performed by using Support

Vector Machine. Experimental results represent that proposed application of using MFCC for gesture recognition has very good accuracy and hence can be used for recognition of sign language or for other household application with the combination for other techniques such as Gabor filter, DWT to increase the accuracy rate and to make it more efficient. Above system uses 13 MFCCs for feature extraction so it was beneficial for development of our system.

An *Automatic Speech Recognition Technique for Bangla Words* [3]. They presented a technique for recognizing spoken words in Bangla. In that study they first derived feature from spoken words. The techniques MFCC, LPC, GMM and DTW for recognizing spoken words in Bangla. Above system used language other than English for speech recognition, & our system also language independent system so this paper is best for us.

A *Speaker Recognition System Based on MFCC and DCT* [4]. Authors examined and presented an approach to the recognition of speech signal using frequency spectral information with Mel frequency. In that paper the optimum values of above parameters are chosen to get an efficiency of 99.5 % over a very small length of audio file which corresponds to our system so it assisted for our system.

A *Voice Recognition Algorithms using Mel Frequency Cepstral Coefficient (MFCC) and Dynamic Time Warping (DTW) Techniques* [5]. This paper presented the viability of MFCC to extract features and DTW to compare the test patterns. Several methods such as LPC, HMM, ANN and etc. are evaluated with a view to identify a straight forward and effective method for voice signal. The extraction and matching process is implemented right after the Pre Processing or filtering signal is performed. The non-parametric method for modeling the human auditory perception system, Mel Frequency Cepstral Coefficients (MFCCs) is utilized as extraction techniques. The non-linear sequence alignment known as Dynamic Time Warping (DTW) introduced by Sakoe Chiba has been used as features matching techniques. Above system matches to our system for feature extraction & matching so it was advantageous for us.

A lot of speech aware applications are already there in the market. Various dictation software have been developed by Dragon[1], IBM and Philips[2]. Genie[3] is interactive speech

[1] http://www.nuance.com/dragon/index.htm visited on March 16, 2014

[2] http://www.speech.philips.com/ visited on March 16, 2014

[3] https://sites.google.com/site/aigeniesystem/ visited on March 16, 2014

recognition software developed by Microsoft. Various voice navigation applications, one developed by AT&T, allow users to control their computer by voice, like browsing the Internet by voice. Many more applications of this kind are appearing every day. The SPHINX speech recognizer of CMU [6] provides the acoustic as well as the language models used for recognition. It is based on the Hidden Markov Models (HMM). The SONIC [7] recognizer is also one of them, developed by the University of Colorado. There are other recognizers such as XVoice[1] for Linux that take input from IBM's via. voice which, now, exists just for Windows.

[1] http://xvoice.sourceforge.net/ visited on April 02, 2014

CHAPTER 3

METHODOLOGY

3.1 VOICE COMMAND MATCHING PROCESS

The process of voice command matching system typically consists of two phases:

1. Recording voice sample and
2. Operating system in user environment

In first phase the speaker has to provide sample of his voice so that the reference template model or database can be built. In this phase, each voice command for three sec duration repeatedly for five times. Then the features of these recorded samples are extracted and saved into a notepad.

The operating phase ensures the inputted test voice is matched with stored referenc voice samples and respective decision is made.

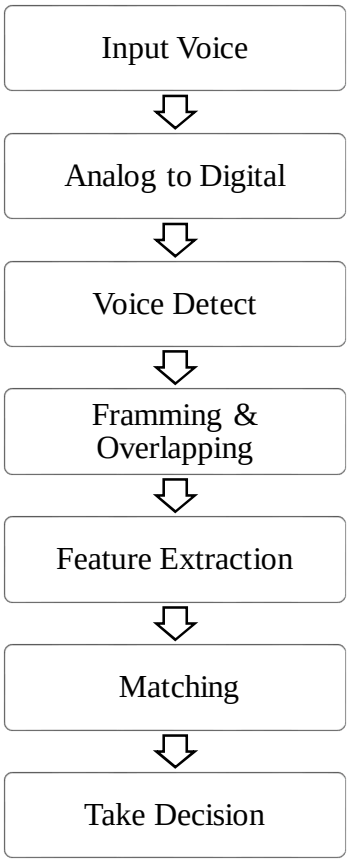

Figure 3.1: Different components of Voice Matching Process & their interactions

3.1.1 INPUT VOICE

Human's voice can be divided into two types, voiced and unvoiced. Voiced sounds are produced by the vibration of vocal cords. On the other hand unvoiced sounds are not produced by the vibration of vocal cords. Both voiced as well as unvoiced sound signal can be input to this system, which is recorded by NAudio (See Appendix 5).

3.1.2 ANALOG TO DIGITAL CONVERSION

Computer is a digital system, it cannot process analog signal directly. So inputted voice signal is then converted using Sampling and Quantization method.

3.1.3 VOICE DETECT

Each command is recorded for 3 seconds. But total 3 seconds do not contain voice activity, they contain some silence or background noise. So, this system suppress those silence parts and take only speech signal part. There are a lot of algorithms for speech detection and the scientists till now researching on it for effectively detect voice activity within a signal. Audio signals are trimmed by removing silence taking threshold value.

Figure 3.1.3.1: Waveform of recorded audio signal for command "Facebook Kholnuhos Mitra".

Figure 3.1.3.2: Waveform of trimmed audio signal for command "Facebook Kholnuhos Mitra".

3.1.4 FRAMING AND OVERLAPPING

When audio signal analyzed, it must be converted into a set of frames, because audio signals are more or less stable within a short period of time, say *20ms* or *30ms*. In our project *25ms* or *8192* sample points are taken as a frame. Usually the overlap is *1/2* of the original frame. The more overlap, the more computation is needed. For *3* sec of recorded audio *14* frames chosen & *7* overlapped frames there are total *21* frames. See footnote below for detail calculations:

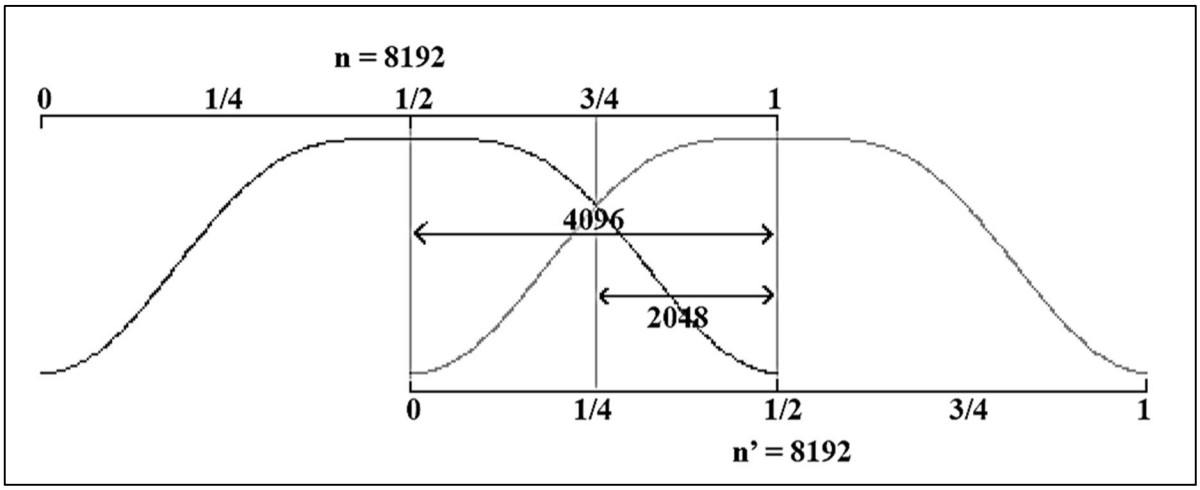

Figure 3.1.4: Frame Overlapping.

3.1.5 FEATURE EXTRACTION

The features of speech signal are amplitude of the signal, energy, intensity, velocity, acceleration, vibration rate, fundamental frequency etc. Feature extraction is process of obtaining different features such as power, pitch, and vocal tract configuration from the speech signals. The extraction of the best parametric representation of acoustic signals is an important task to produce a better recognition performance. For feature extraction, we decided to take MFCC which is based on human hearing perceptions that can perceive, frequencies over *1000 Hz*.

Audio file of *3* Seconds contains near about *180000* spectrum points of *16*-Bit PCM Mono. After removing some silence part it length becomes *1.5* to *2* Seconds. If there are points less than *114688* we append '*0*' and make it to *114688* & if greater than that amount we truncate remaining part. We divided *114688* points into *14* frames, each frame contains *8192* spectrum points and there are *7* overlapped frames. So we process data of *21* frames i.e. *172032* double precision points.

In other words, MFCC is based on known variation of the human ear's critical bandwidth with frequency. It has two types of filter which are spaced linearly at low frequency below *1000 Hz* and logarithmic spacing above *1000 Hz*.

The frequency range in FFT spectrum which is very wide and voice signal does not follow the linear scale. The Mel-filter bank is a nonlinear filter. The filter bank is constructed using *13* linearly-spaced filters. Each filter is constructed by combining the amplitude of FFT.

In a single frame *13* MFCC feature vectors are extracted. Also there are *14* frames for *3* Seconds of audio and *7* overlapped frames.

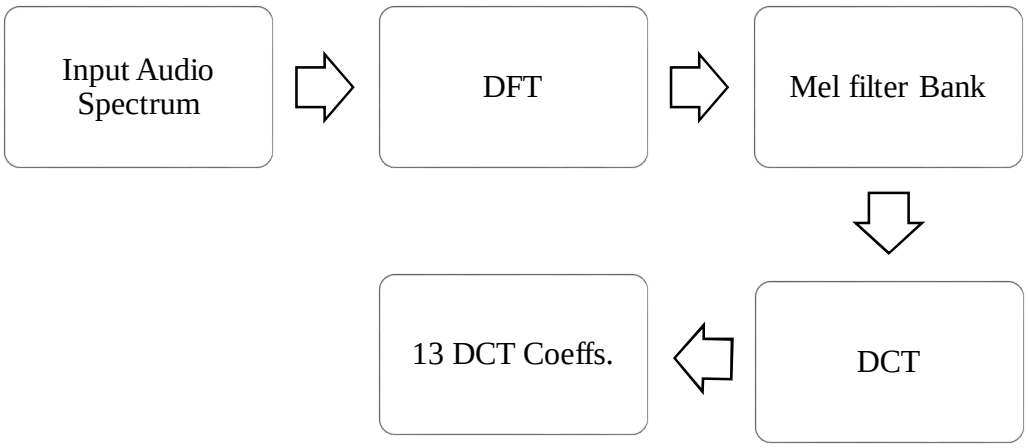

Figure 3.1.5: MFCC Block Diagram

MFCC consists of five computational steps. Each step has its function and mathematical approaches as discussed briefly in the following:

1. Input Audio Spectrum
2. Discrete Fourier Transform
3. Mel Filter Bank Processing
4. Discrete Cosine Transform and
5. Output *13* DCT coefficients

3.1.5.1 INPUT AUDIO SPECTRUM

The spectrum of recorded signal after trimming can be generated via Bit Converter to *16*-bit integer. Then it is used for further process.

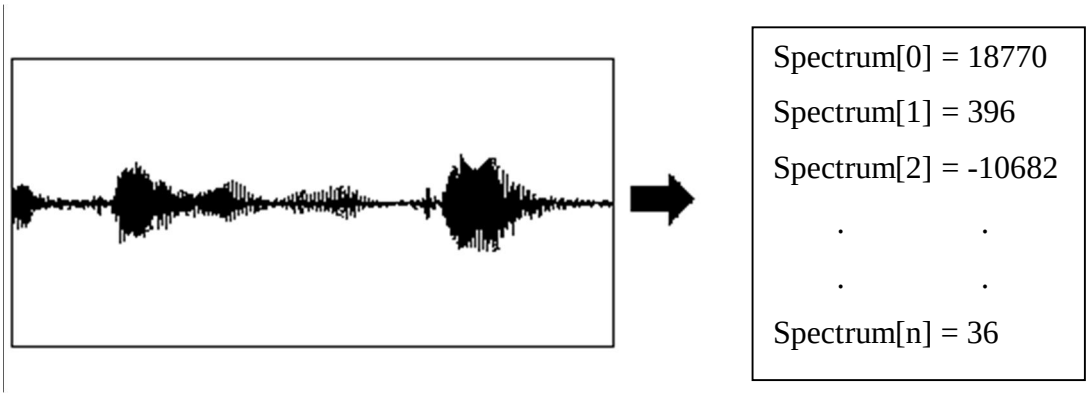

Figure 3.1.5.1: Audio to spectrum conversion

3.1.5.2 DISCRETE FOURIER TRANSFORM

The DFT is the most important discrete transform, used to perform Fourier analysis in many practical applications. In digital signal processing, the function is any quantity or signal that varies over time, such as the pressure of a sound wave, a radio signal, or daily temperature readings, sampled over a finite time interval.

The sequence of $\boldsymbol{N}$ complex numbers $x_0, x_1, x_2, \ldots, x_{N-1}$ is transformed into an $\boldsymbol{N}$-periodic sequence of complex numbers:

$$X_{\boldsymbol{k}} = \sum_{n=0}^{N-1} x_n . e^{-i2\pi kn/N}$$

3.1.5.3 MEL FILTER BANK PROCESSING

Filter bank Approach is method to transform the power spectrum, i.e. to compute a Mel-warped spectrum by interpolation from the original discrete-frequency power spectrum. The advantage is that the following triangular filters all have the same shape and can be placed uniformly at the Mel-warped spectrum. On the other hand, the discretization may be especially critical due to the large dynamic range of the power spectrum.

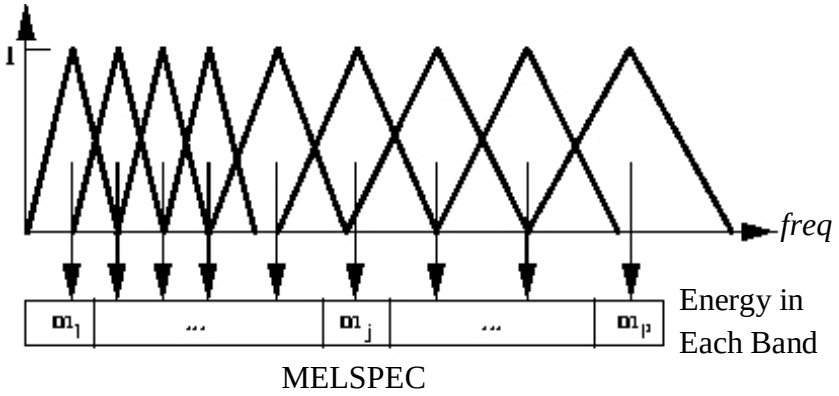

Figure 3.5.1.3: Mel-Scale Filter Bank[1]

A set of triangular filters are used to compute a weighted sum of filter spectral components so that the output of process approximates to a Mel scale. Each filter's magnitude frequency response is triangular in shape and equal to unity at the center frequency and decrease linearly to zero at center frequency of two adjacent filters. Then, each filter output is the sum of its filtered spectral components. After that the following equation is used to compute the Mel for given frequency f in Hz:

$$F(Mel) = 2595 * log_{10}(1 + \frac{f}{700})$$

3.1.5.4 DISCRETE COSINE TRANSFORM

This is the process to convert the log Mel spectrum into Discrete Cosine Transform coefficients. The result of the conversion is called Mel Frequency Cestrum Coefficient. Therefore, each input utterance is transformed into a sequence of acoustic vector. However, DFT is generally used for spectral analysis whereas DCT used for data compression as DCT signals have more information concentrated in a small number of coefficients and hence, it is easy and requires less storage to represent Mel spectrum in a relative small number of coefficients. This instead of using DFT DCT is desirable for the coefficients calculation as DCT outputs can contain important amounts of energy. The output after applying DCT is known as MFCC (Mel Frequency Cepstrum Coefficient).

[1]Image Source: http://www.ee.columbia.edu/ln/LabROSA/doc/HTKBook21/node54.html

3.1.5.5 OUTPUT 13 DCT COEFFICIENTS

Each input utterance is transformed into a sequence of acoustic vector. Generally, first *12* to *15* coefficients are best representation of amplitude of resultant spectrum. So, we choose first *13* coefficients as acoustic vectors per frame.

3.1.6 MATCHING

Template matching is the simplest technique and has the highest accuracy when used properly, but it also suffers from the most limitations. There are a lot of techniques used for matching feature matrix and reference matrix. In this project Euclidean distance and correlation are used for matching or measuring distance between feature matrix and reference matrix.

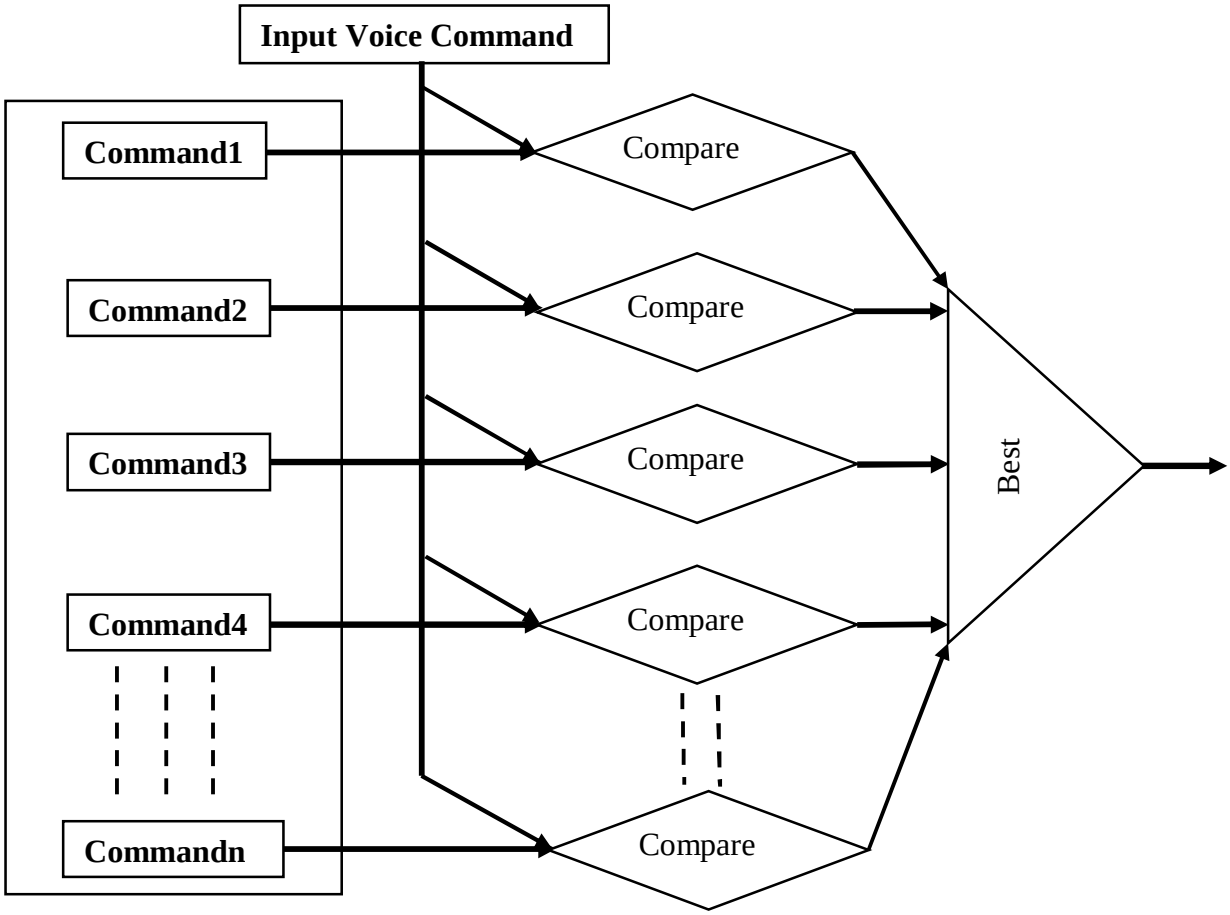

Figure 3.1.6: Matching Process

3.1.6.1 EUCLIDEAN DISTANCE

Euclidean distance is the ordinary distance between two points. The Euclidean distance between points *p* and *q* is the length of the line segment connecting them ($\overline{pq}$).

In Cartesian coordinates, if $p = (p_1, p_2,..., p_n)$ and $q = (q_1, q_2,..., q_n)$ are two points in Euclidean *n*-space, then the distance (*d*) from *p* to *q*, or from *q* to *p* is given by:

$$d(p,q) = d(q,p) = \sqrt{(q1 - p1)^2 + (q2 - p2)^2 + \cdots + (qn - pn)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (qi - pi)^2}$$

3.1.6.2 CORRELATION

Correlation is statistical relationship between two random variables or two sets of data. Correlations are useful because they can indicate a predictive relationship that can be exploited in practice. For example, an electrical utility may produce less power on a mild day based on the correlation between electricity demand and weather.

3.1.7 TAKE DECISION

The distance between feature matrix and all reference matrixes are calculated. Then the reference matrix is close to the feature matrix is selected and minimum distance for taking decision.

After successful matching respective computer command is invoked.

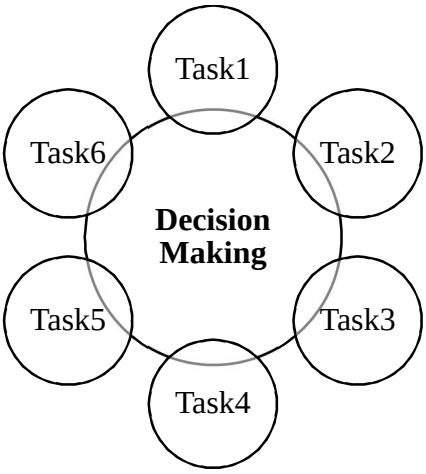

Figure 3.1.7: Decision Making

3.2 TOOLS & PLATFORM

We used C# as the development platform & *dot net framework 4.5* are used. The operating system for the model should be *Windows8/Windows7*. The coding was done in *Visual Studio 2012*. Various files such as .wav file, .txt file, and .xml files are used for storing data.

Tools used in this projects are as follows:

- IDE,
- Graphics Editor,
- Text Editor,
- Spreadsheet, etc.

3.3 SYSTEM REQUIREMENTS

To use VCMS, Laptop or PC that operate in *dot net 4.5* platform with operating system *Winwods7* or higher.

CHAPTER 4

PROJECT DEVELOPMENT LIFE CYCLE

The project VCMS contains various phases such as research, design, coding, testing etc. (see Appendix 1). User's interaction with the system is shown in the Appendix 2. The state diagram of the system is shown in Appendix 3 & its flowchart is shown in Appendix 4. And all designed & developed GUIs are in Appendix 6.

4.1 AGILE SOFTWARE DEVELOPMENT

Agile software development is a group of software development methods in which requirements and solutions evolve through collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development, early delivery, continuous improvement and encourages rapid and flexible response to change. It is a conceptual framework that focuses on delivering working software with the minimum amount of work.

4.2 SCRUM AS AGILE DEVELOPMENT

Scrum is the most popular way of introducing Agility due to its simplicity and flexibility. Because of this popularity, many organizations claim to be “doing Scrum” but aren’t doing anything close to Scrum’s actual definition. Scrum emphasizes empirical feedback, team self-management, and striving to build properly tested product increments within short iterations. Doing Scrum as it’s actually defined usually comes into conflict with existing habits at established non-Agile organizations.

Scrum is an agile methodology that can be applied to nearly any project; however the Scrum methodology is most commonly or used changing highly emergent requirements. Scrum software development progresses via a series of iterations called sprint, which last from one to four weeks. The Scrum model suggests each sprint begins with a brief planning meeting and concludes with a review. These are the basics of Scrum project management

Scrum has only three roles: Product Owner, Team, and Scrum Master. These are described in detail by the Scrum Training Series. The responsibilities of the traditional project manager role are split up among these three Scrum roles.

Scrum has five meetings:

- Backlog grooming (aka Backlog Refinement),
- Sprint Planning,
- Daily Scrum (aka 15-minute standup),
- The Sprint Review Meeting, and
- The Sprint Retrospective Meeting.

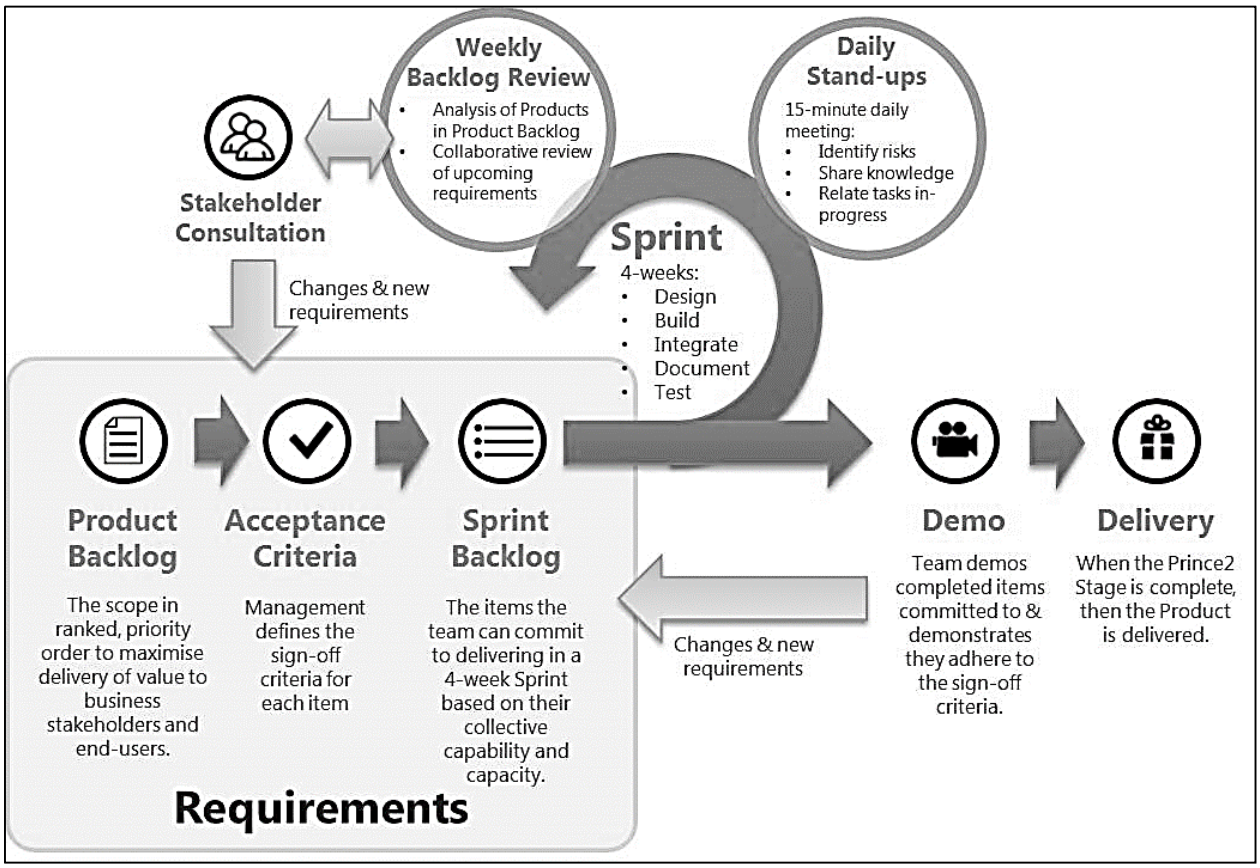

Figure 4.2: Scrum, Agile development Process[1]

[1]Image Source: http://zenexmachina.wordpress.com/2013/04/03/prince2-processes-vs-agile-development-methodologies/

CHAPTER 5

TESTING AND RESULT ANALYSIS

5.1 OVERVIEW

In order to guarantee a Voice Command Matching System, careful testing is needed. Different Snippets of audio like the ones starting from first, from middle and even of the last were taken and matched with the original snippet.

5.2 TEST CASES

A test case, is a set of conditions or variables under which a tester will determine whether an application, software system or one of its features is working as it was originally established for it to do. In this project, there are various test cases as listed below:

Table 5.2.1: Test Case - Audio Record	
VCMS_TESTCASE_01	**AUDIO RECORD**
Purpose:	Recording audio signal for both states i.e. Recording sample state & Operating State
Prereq:	Microphone, Soundcard
Test Data:	startRec = {Time Span: 3 Sec, Naudio Recording, fileName}
Steps:	**For Recording Sample:**
	1. Start VCMS
	2. Click + Sign (Menu)
	3. Click Record Sample Button
	4. Select the Command
	5. Click Start Button
	6. Click Record Button
	7. Speak
	8. Verify recording & check the location where respective files are saved.
	For Operating Purpose:
	1. Start VCMS
	2. Click on Mic Button
Expected result:	Recording audio in 16-Bit PCM, Mono Channel
Actual Result:	Audio Recorded with 16-Bit PCM, Mono Channel
Status:	Passed
Comments:	Tested for both states recording state as well as operating state

Table 5.2.2: Test Case - Silence Remove	
VCMS_TESTCASE_01	**SILENCE REMOVE**
Purpose:	Detecting voice signal by removing silence.
Prereq:	Audio Signal
Test Data:	trimWavFile = {.wav audio signal}
Steps:	1. Start VCMS
	2. Click on Developer Option
	3. Click on Trim Audio Button
	4. Verify trimmed audio signal
Expected result:	Get trimmed audio signal
Actual Result:	Silence removed
Status:	Passed
Comments	Length of trimmed audio signal is less than that of recorded audio signal

Table 5.2.3: Test Case - Waveform Generation & Play Back	
VCMS_TESTCASE_01	**WAVEFORM GENERATION AND PLAY BACK**
Purpose:	Generating waveform of audio signal & play back for verifying what the user want to spoke
Prereq:	Audio Signal
Test Data:	genWaveform = {Audio Signal}
	replay = {Audio Signal}
Steps:	1. Start VCMS
	2. Click on Developer Option
	3. Click on Generate Waveform Button
	4. Click on Play Button
	4. Verify waveform of audio signal & sound from that audio
Expected result:	Get trimmed audio signal
Actual Result:	Silence removed
Status:	Passed
Notes & Questions:	Are waveform of various samples of command similar?

Table 5.2.4: Test Case - MFCC Computation	
VCMS_TESTCASE_01	**MFCC COMPUTATION**
Purpose:	Compute MFCC of partially overlapped spectrum of trimed audio
Prereq:	1/2 Overlapped Spectrum
Test Data:	calcMFCC = {Spectrum Data}
Steps:	1. Start VCMS
	2. Click on Developer Option
	3. Click on Calc MFCC Button
	4. Verify MFC Coefficients

Expected result:	Get 273 MFC Coefficients
Actual Result:	MFCC Computed & got 273 MFCCs
Status:	Passed
Comments	There are 273 MFCCs per sample

Table 5.2.5: Test Case - Euclidian Distance & Correlation Computation	
VCMS_TESTCASE_01	**EUCLIDIAN DISTANCE AND CORRELATION COMPUTATION**
Purpose:	Finding Euclidian distance & correlation for matching purpose
Prereq:	Reference Feature Vector, Current Feature vector
Test Data:	CORREL = {Current MFCCs, Reference MFCCs}
	EUD = {Current MFCCs, Reference MFCCs}
Steps:	1. Start VCMS
	2. Click on Developer Option
	3. Click on Compute EudDist & CORREL Button
	4. Verify Euclidean Distance & Correlation
Expected result:	Get Euclidean Distance & Correlation between two feature vector
Actual Result:	Correlation & Euclidean distance computed
Status:	Passed
Comments	Respective command invoked if correlation is greater than 0.85 with minimum Euclidian distance

5.3 MATCHING SCENARIO

Table 5.3.1: Matching result for command "Facebook KholnuhosMitra"				
SN	**Voice Commands**		**ED**	**CORREL**
	Test Cmd	**Reference Cmd**		
1	Facebook KholnuhosMitra	Facebook KholnuhosMitra	34	0.905
2		Open Notepad	38	0.9
3		Shut Down	44.6	0.8
4		Open Wordpad	38.5	0.901
5		Close Notepad	36.5	0.911
6		Light On	44.3	0.8

Table 5.3.2: Matching result for command "Open Notepad"				
SN	**Voice Commands**		**ED**	**CORREL**
	Test Cmd	**Reference Cmd**		
1	Open Notepad	Facebook KholnuhosMitra	33.6	0.909
2		Open Notepad	29.9	0.937
3		Shut Down	48.92	0.792
4		Open Wordpad	33.5	0.92
5		Close Notepad	31.18	0.934
6		Light On	47.65	0.805

Table 5.3.3: Matching result for command "Shutdown"				
SN	**Voice Commands**		**ED**	**CORREL**
	Test Cmd	**Reference Cmd**		
1	Shutdown	Facebook KholnuhosMitra	57	0.713
2		Open Notepad	61.53	0.703
3		Shut Down	35.85	0.87
4		Open Wordpad	63.32	0.689
5		Close Notepad	61.45	0.709
6		Light On	36.31	0.857

Table 5.3.4: Matching result for command "Open Wordpad"				
SN	**Voice Commands**		**ED**	**CORREL**
	Test Cmd	**Reference Cmd**		
1	Open Wordpad	Facebook KholnuhosMitra	32.77	0.913
2		Open Notepad	31.71	0.929
3		Shut Down	47.54	0.802
4		Open Wordpad	33.8	0.928
5		Close Notepad	30.06	0.939
6		Light On	47.56	0.803

Table 5.3.5: Matching result for command "Close Notepad"				
SN	**Voice Commands**		**ED**	**CORREL**
	Test Cmd	**Reference Cmd**		
1	Close Notepad	Facebook KholnuhosMitra	35.2	0.902
2		Open Notepad	35.89	0.909
3		Shut Down	47.25	0.804
4		Open Wordpad	34.7	0.917
5		Close Notepad	30.36	0.93
6		Light On	47.1	0.804

Table 5.3.6: Matching result for command "Light On"				
SN	**Voice Commands**		**ED**	**CORREL**
	Test Cmd	**Reference Cmd**		
1	Light On	Facebook KholnuhosMitra	42.28	0.854
2		Open Notepad	50.06	0.815
3		Shut Down	35.54	0.867
4		Open Wordpad	48.88	0.83
5		Close Notepad	48.61	0.831
6		Light On	30.91	0.9009

5.4 SUMMARY OF MATCHING SCENARIO

Table 5.4.1: Summary of Matching Scenario			
SN	**Commands**	**Selecting Task**	
		From ED	**Form CORREL**
1	Facebook KholnuhosMitra	✔	✔
2	Open Notepad	✔	✔
3	Shut Down	✔	✘
4	Open Wordpad	✘	✔
5	Close Notepad	✔	✔
6	Light On	✔	✔

5.5 TEST ENVIRONMENT

For testing of the system in various environment, first of all samples of respective command are recorded in that environment & following are the result obtained from such operating environment.

Table 5.5.1: Test Environment - Calm Place(Closed Room at Midnight)			
SN	**Spoken Command**	**Invoked Computer Task**	**Estimation**
1	Facebook KholnuhosMitra	Facebook Opened	Correct
2	Open Notepad	Notepad Opened	Correct
3	Shut Down	Asked Confirmation for Shutdown	Correct
4	Open Wordpad	Notepad Opened	Correct
5	Close Notepad	Notepad Closed	Correct
6	Light On	Display Screen White	Correct

Table 5.5.2: Test Environment - Normal Place(Closed Room at Morning)			
SN	**Spoken Command**	**Invoked Computer Task**	**Estimation**
1	Facebook KholnuhosMitra	Facebook Opened	Correct
2	Open Notepad	Notepad Opened	Correct
3	Shut Down	Asked Confirmation for Shutdown	Correct
4	Open Wordpad	Notepad Opened	**Wrong**
5	Close Notepad	Notepad Closed	Correct
6	Light On	Display Screen White	Correct

Table 5.5.3: Test Environment - College Premises (At Midday)			
SN	**Spoken Command**	**Invoked Computer Task**	**Estimation**
1	Facebook KholnuhosMitra	Facebook Opened	Correct
2	Open Notepad	Notepad Opened	Correct
3	Shut Down	Asked Confirmation for Shutdown	Correct
4	Open Wordpad	Notepad Opened	**Wrong**
5	Close Notepad	Notepad Opened	**Wrong**
6	Light On	Display Screen White	Correct

Table 5.5.4: Test Environment - College's Noisy Environment (Operating Generator at Corner of College)			
SN	**Spoken Command**	**Invoked Computer Task**	**Estimation**
1	Facebook KholnuhosMitra	Facebook Opened	Correct
2	Open Notepad	Wordpad Opened	**Wrong**
3	Shut Down	Asked Confirmation for Shutdown	Correct
4	Open Wordpad	Notepad Opened	**Wrong**
5	Close Notepad	Notepad Closed	Correct
6	Light On	Display Screen White	Correct

Table 5.5.5: Test Environment - Noisy Environment (College Expo)			
SN	**Spoken Command**	**Invoked Computer Task**	**Estimation**
1	Facebook KholnuhosMitra	Facebook Opened	Correct
2	Open Notepad	Retry & Opened Facebook	**Wrong**
3	Shut Down	Asked Confirmation for Shutdown	Correct
4	Open Wordpad	Opened Facebook	**Wrong**
5	Close Notepad	Asked Confirmation for Shutdown	**Wrong**
6	Light On	Display Screen White	Correct

5.6 TEST VOICE

Table 5.6.1: Test Voice - Male (User - Shiva K. Shrestha)				
SN	**Spoken Command**	**Repetition**	**Correct**	**% Correct**
1	Facebook KholnuhosMitra	10	10	100
2	Open Notepad	10	9	90
3	Shut Down	10	10	100
4	Open Wordpad	10	9	90
5	Close Notepad	10	10	100
6	Light On	10	10	100
			Avg %Correct	96.66666667

Table 5.6.2: Test Voice - Male (User - Raj Kaji Shrestha)				
SN	**Spoken Command**	**Repetition**	**Correct**	**% Correct**
1	Facebook KholnuhosMitra	10	10	100
2	Open Notepad	10	9	90
3	Shut Down	10	10	100
4	Open Wordpad	10	8	80
5	Close Notepad	10	10	100
6	Light On	10	10	100
			Avg %Correct	95

Table 5.6.3: Test Voice - Female (User - Shanta Shrestha)				
SN	**Spoken Command**	**Repetition**	**Correct**	**% Correct**
1	Facebook KholnuhosMitra	5	5	100
2	Open Notepad	5	4	80
3	Shut Down	5	5	100
4	Open Wordpad	5	4	80
5	Close Notepad	5	4	80
6	Light On	5	5	100
			Avg %Correct	90

Table 5.6.4: Test Voice - Adult Male (User - Sandesh Danekhu)				
SN	**Spoken Command**	**Repetition**	**Correct**	**% Correct**
1	Facebook KholnuhosMitra	5	5	100
2	Open Notepad	5	2	40
3	Shut Down	5	4	80
4	Open Wordpad	5	1	20
5	Close Notepad	5	0	0
6	Light On	5	4	80
			Avg %Correct	53.33333333

Table 5.6.5: Test Voice - Adult Female (User - Anita Duwal)				
SN	**Spoken Command**	**Repetition**	**Correct**	**% Correct**
1	Facebook KholnuhosMitra	5	5	100
2	Open Notepad	5	1	20
3	Shut Down	5	5	100
4	Open Wordpad	5	2	40
5	Close Notepad	5	3	60
6	Light On	5	5	100
			Avg %Correct	70

5.7 SUMMARY OF TEST VOICE

Table 5.7: Summary of Test Voice				
SN	**Spoken Command**	**Repetition**	**Correct**	**% Correct**
1	Facebook KholnuhosMitra	35	35	100
2	Open Notepad	35	25	71.42857143
3	Shut Down	35	34	97.14285714
4	Open Wordpad	35	24	68.57142857
5	Close Notepad	35	27	77.14285714
6	Light On	35	34	97.14285714
			Avg %Correct	85.23809524

5.8 DISCUSSION AND RESULT ANALYSIS

The snippet of matching scenario is shown in table 5.4.1. In this summary table, we can see there are total six command. Among them, only for four matching is correct. These are correct because, these passes the correlation threshold which is equal to *0.9* & gives minimum Euclidian distance. Remaining two are error of true negative & false positive respectively. The first error is true negative because Euclidian distance selects the right command but correlation is *0.87* which is less than that of *0.9*. So it gives chance of retry for user instead of invoking command “Shutdown”. And next error is false positive because it selects the wrong command with correlation *0.913* which is greater than that of *0.9*. So it invoked false command “Facebook KholnuhosMitra” instead of command “Open Wordpad”.

For this snippet of testing, the efficiency of the system is *66.66%*, the efficiency of this system can be increased if we select the best correlation threshold. i.e. if we select threshold of the system to 0.87 then the efficiency of our system can be *83.33%* for this small snippet. And if we speak the command exactly as we record for sample then we can get optimum efficiency i.e. about *99%*.

We have done testing of our product in various environment. System perform 100% accurate at Midnight (See Table 5.5.1).The reason behind the correct estimation of the system are one the system is running in noise free environment & maybe run by the careful user. And all most voice signal's MFCCs have correlation greater or equal to the threshold *0.9* with minimum Euclidian distance. The system has one wrong estimation (See Table 5.5.2) at Morning. The possible reason for such type error can be more similar voice for the commands "Open Notepad" & "Open Wordpad". At midday, the system has two wrong estimation (See Table 5.5.3), it opened notepad instead of opening wordpad & closing notepad. The same reason for these error i.e. similar voice. In college's noisy environment i.e. operating generator at the corner of college, there are a lot of fluctuating noise. So here (See Table 5.5.4), we got two wrong estimation. The possible reason for such error can be operating system in that environment where noise level fluctuates & more similar voice. The system has more wrong estimation at College Exhibition Time (See Table 5.5.5). We got three wrong estimations, the possible reasons for such error can be using system in that environment where the noise level are fluctuating at both recording state as well as operating state, more similar voice, etc. As the result of various testing environment, the system can be operated with optimum efficiency if the sample was recorded in the same environment. And the operating efficiency will degraded if the user want to run the system different from recording environment.

Our system is tested by using voices of various age group. We can see summary of test voice in table 5.7. The summary is generated with the help of various user's testing repetition. For the command "Facebook KholnuhuosMitra", we got *100%* correct result. The reason behind this correct result can be the command itself longer than other command & users are more careful for this command in both state that is recording state as well as operating state. For other commands such as "Shut Down" & "Light On" also have more correct result because these are short command in speaking length. And other remaining have different correct result. The possible reasons for such result can be user's carefulness, testing environment, user's mic position, etc. Over all, we got efficiency is equal to *85.24%* from test voice.

CHAPTER 6

CONCLUSION

6.1 CONCLUSION

The text-independent speaker recognition is very difficult compared with the text- dependent speaker recognition because here the testing is performed with the new inputs which are not there in practice. So the new methods are necessary. VCMS, is a very challenging, yet very interesting project. The major objective of this project is to operate computer commands through custom voice is fulfilled.

The project is based on voice matching, which has been an area of interest for many developers. VCMS provide luxury to the general users and helps the physically disabled ones, the merit to use and access the world of Computers/Technologies.

We tried our best to provide higher accuracy and efficiency.

6.2 FUTURE ENHANCEMENT

- Robot navigation through voice command
- Extending the system for using more computer commands
- Improvement in the user interface
- Reducing noise using various noise filters
- For Gaming purpose
    - E.g. Puzzle game
- Security Enhancement – Voice Authentication
- Continuously running the system in background
- Developing special pack which can suite to general user as per their language considering the modification of the system

REFERENCES

[1] PPS Subhashini, Turimerla Pratap, "TEXT-INDEPENDENT SPEAKER RECOGNITION USING COMBINED LPC AND MFC COEFFICIENTS", International Journal of Research in Engineering and Technology eISSN: 2319-1163 & pISSN: 2321-7308

[2] Shikha Gupta, et al., "FEATURE EXTRACTION USING MFCC", An International Journal (SIPIJ) Vol.4, No.4, August 2013

[3] Md. Akkas Ali, et al., "Automatic Speech Recognition Technique for Bangla Words", International Journal of Advanced Science and Technology, Vol. 50, January, 2013

[4] Garima Vyas, Barkha Kumari, "Speaker Recognition System Based On MFCC and DCT", International Journal of Engineering and Advanced Technology (IJEAT), ISSN: 2249 – 8958, Volume-2, Issue-5, June 2013

[5] L. Muda, et al., "Voice recognition algorithm Using Mel Frequency Cepstral Coefficient (MFCC) and Dynamic Time Warping (DTW) Techniques," Journal of computing vol. 2, 2010.

[6] Kai-Fu Lee, Hsiao-Wuen Hon, and Raj Reddy, An Overview of the SPHINX Speech, Recognition System. IEEE Transactions on Acoustics, Speech and Signal Processing, IEEE Transactions on Acoustic Speech, and Signal Processing. Vol. 38, 1, Janaurary 1990

[7] Pellom, B., Sonic: The University of Colorado Continuous Speech Recognition System, Technical Report TR-CSLR-2001-01, March 2, 2001 Revised on May 31, 2005

APPENDIX 1

PROJECT MANAGEMENT

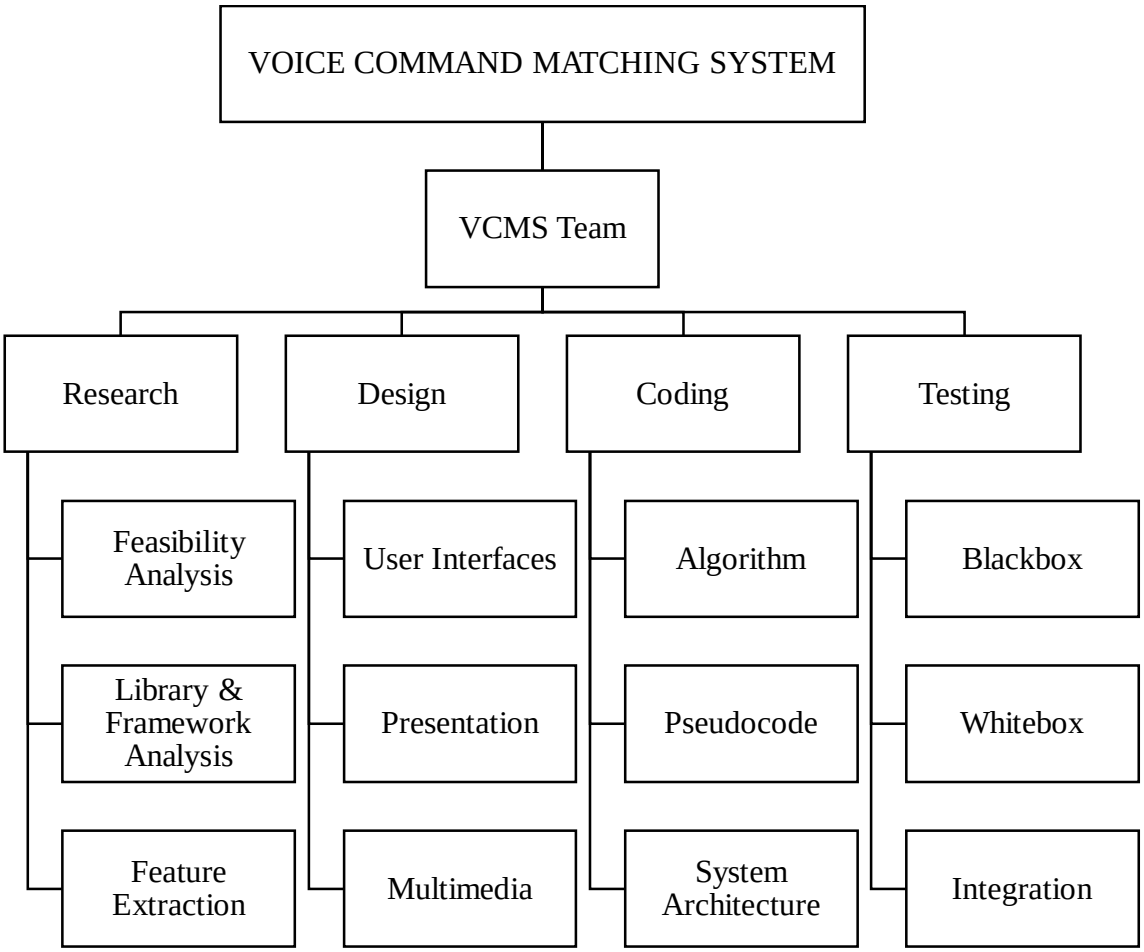

Figure A1.1: Work Breakdown Structure

APPENDIX 2

USE CASE DIAGRAM

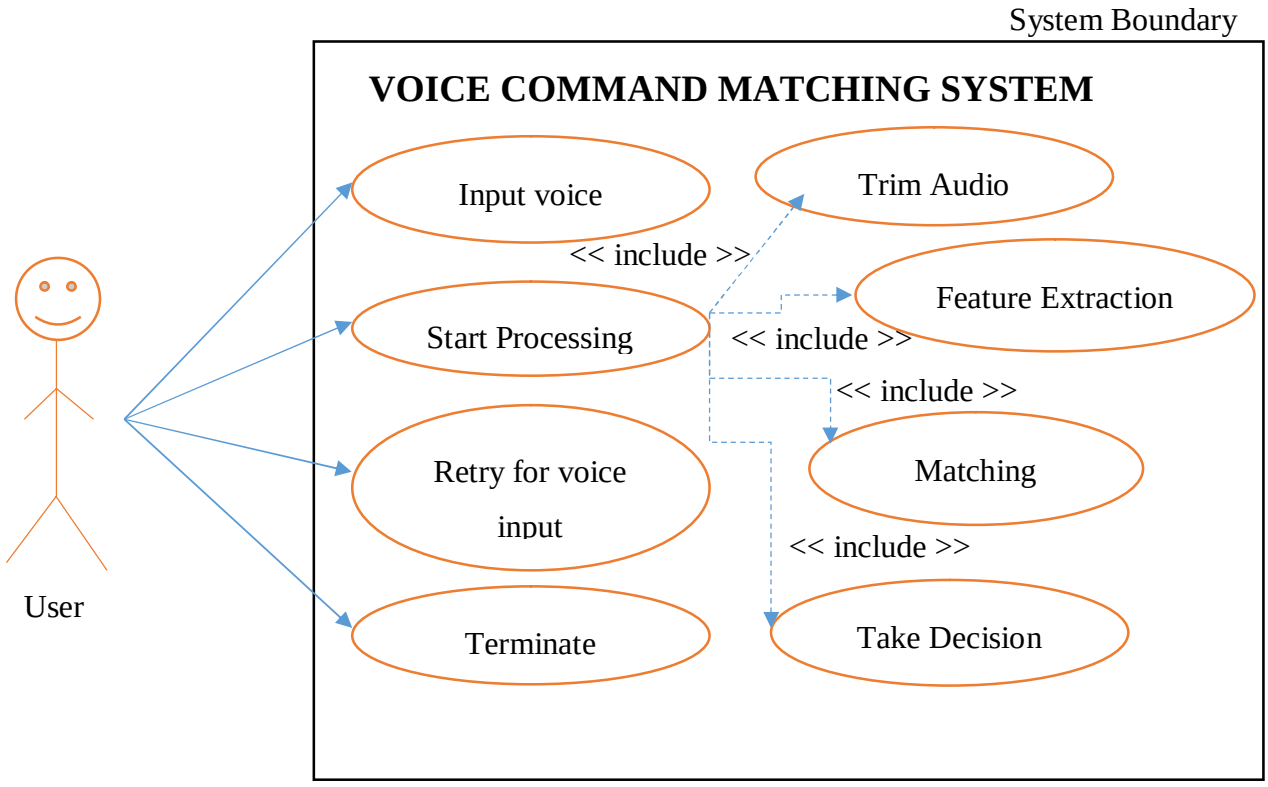

Figure A2.1: Use Case Diagram of VCMS

APPENDIX 3

STATE DIAGRAM

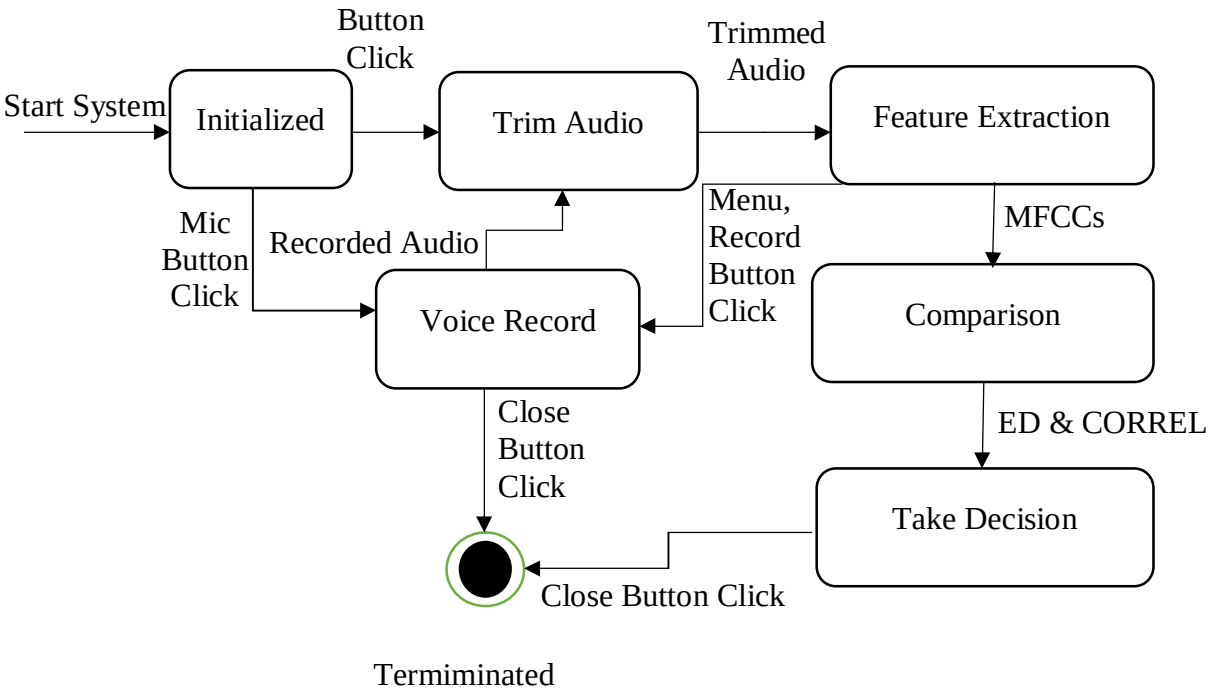

Figure A3.1: State Diagram of VCMS

APPENDIX 4
FLOWCHART

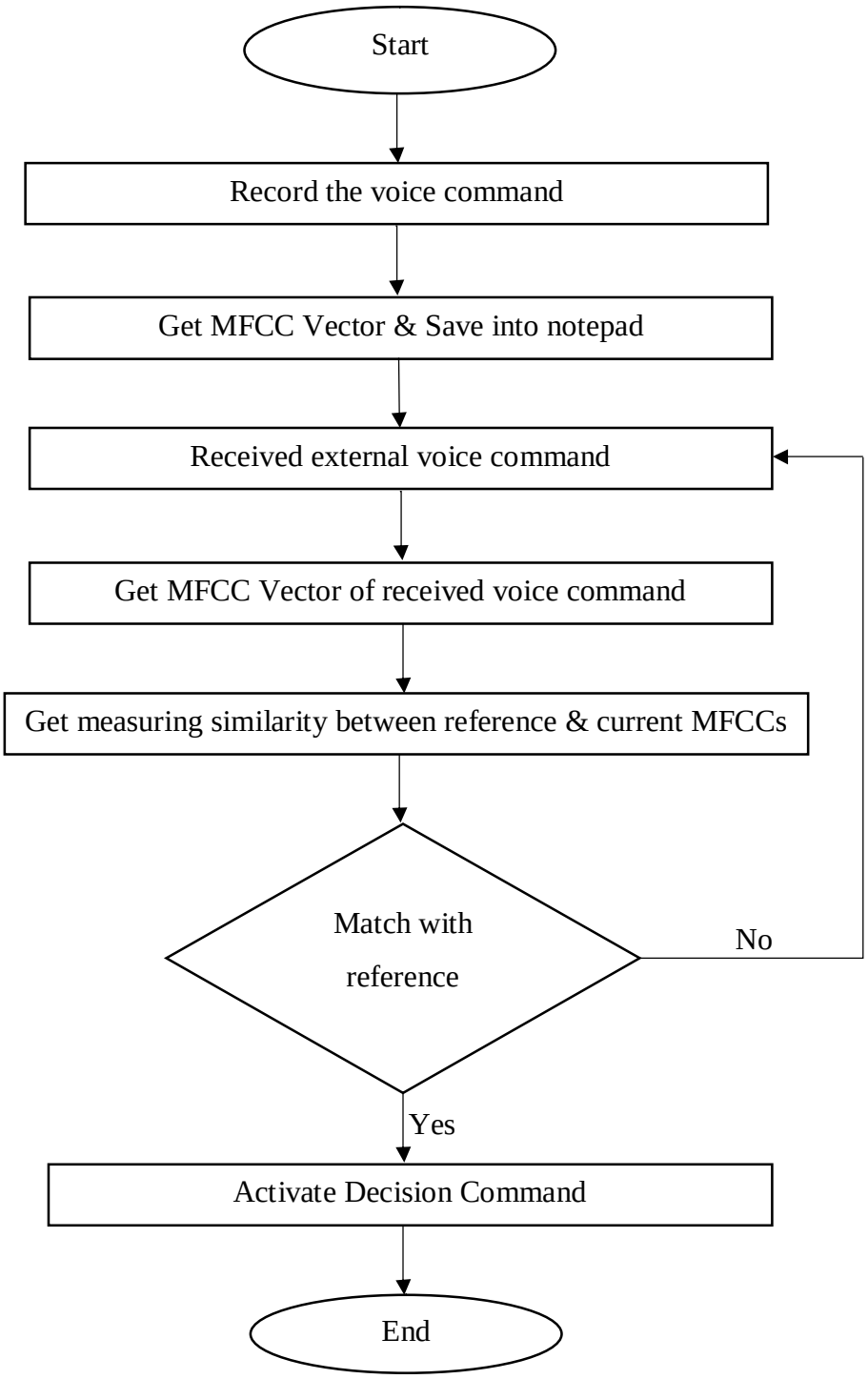

Figure A4.1: VCMS Flowchart

APPENDIX 5
NAUDIO FRAMEWORK

ABOUT NAUDIO

NAudio is an open source .NET audio and MIDI library, written in C# by Mark Heath, with contributions from many other developers, containing dozens of useful audio related classes intended to speed development of audio related utilities in .NET. It is intended to provide a comprehensive set of useful utility classes from which you can construct your own audio application. It has been in development since 2002 and has grown to include a wide variety of features. While some parts of the library are relatively new and incomplete, the more mature features have undergone extensive testing and can be quickly used to add audio capabilities to an existing .NET application. NAudio can be quickly added to your .NET application using NuGet.

WHY NAUDIO?

NAudio was created because the Framework Class Library that shipped with .NET 1.0 had no support for playing audio. The System Media namespace introduced in .NET 2.0 provided a small amount of support, and the Media Element in WPF and Silverlight took that a bit further. The vision behind NAudio is to provide a comprehensive set of audio related classes allowing easy development of utilities that play or record audio, or manipulate audio files in some way.

FEATURES:

- Play back audio using a variety of APIs
  - WaveOut
  - DirectSound
  - ASIO
  - WASAPI (Windows Vista and above)
- Decompress audio from different Wave Formats
  - MP3 decode using ACM or DMO codec
  - AIFF

- G.711 mu-law and a-law
- ADPCM
- G.722
- Speex (using NSpeex)
- SF2 files
- Decode using any ACM codec installed on your computer

❖ Record audio using WaveIn, WASAPI or ASIO

❖ Read and Write standard .WAV files

❖ Mix and manipulate audio streams using a 32 bit floating mixing engine

❖ Extensive support for reading and writing MIDI files

❖ Full MIDI event model

❖ Basic support for Windows Mixer APIs

❖ A collection of useful Windows Forms Controls

❖ Some basic audio effects, including a compressor

APPENDIX 6
SCREENSHOTS

Figure A5.1: VCMS Application UI

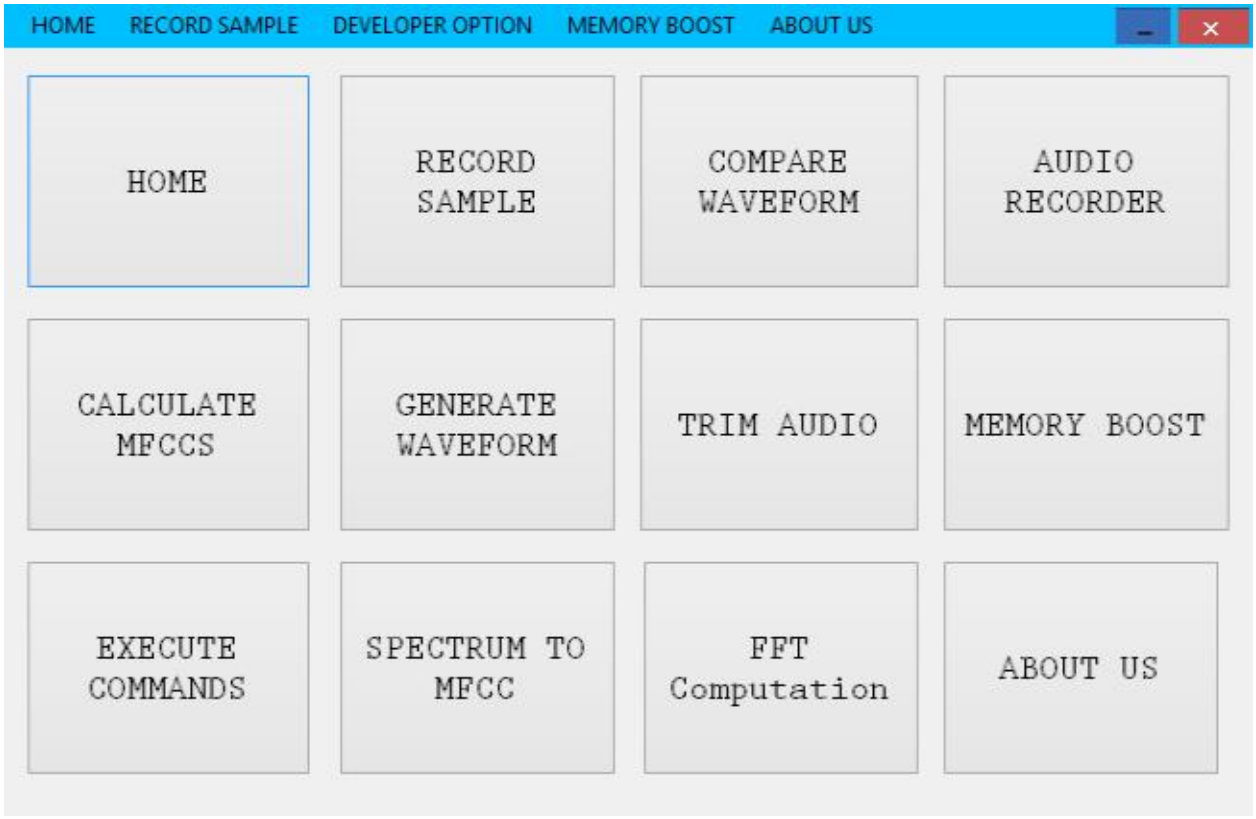

Figure A5.2: Menu

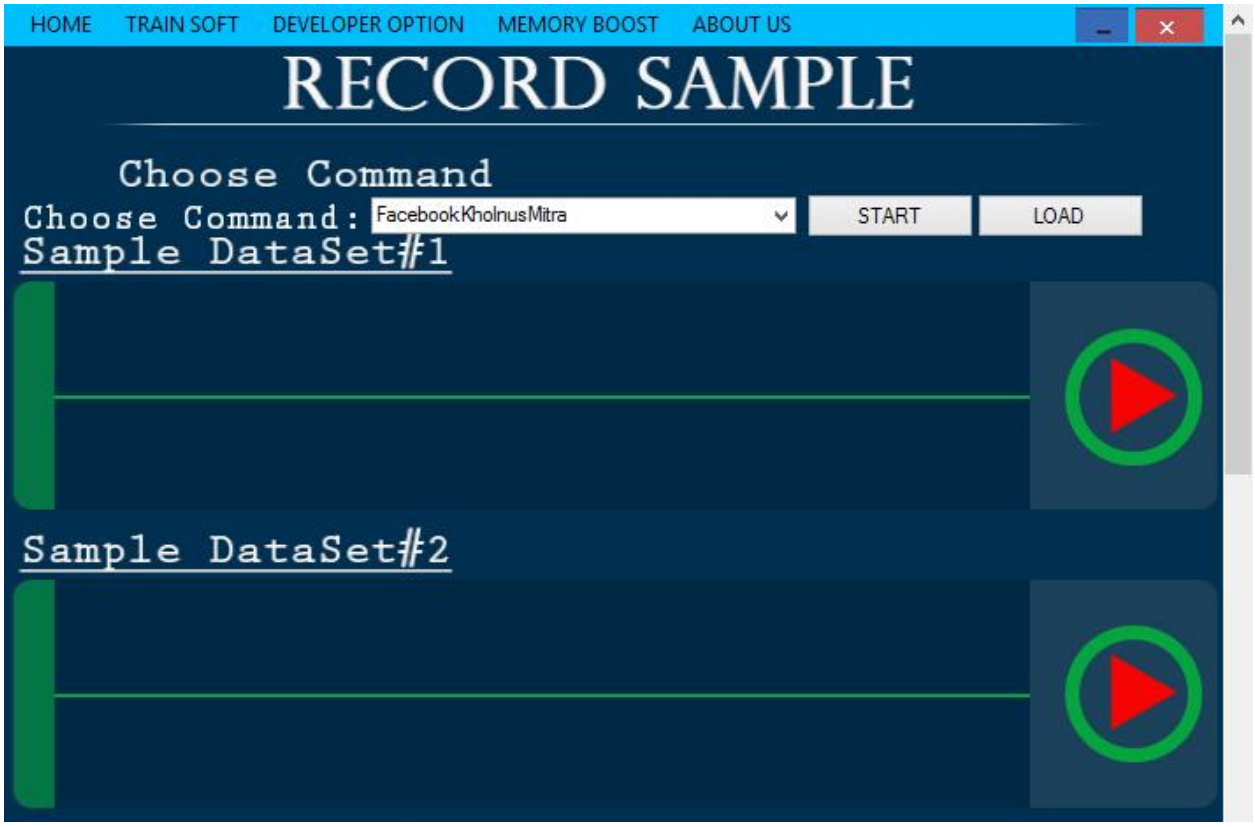

Figure A5.3: Sample Recording UI

Figure A5.4: Recording UI for Specific Command

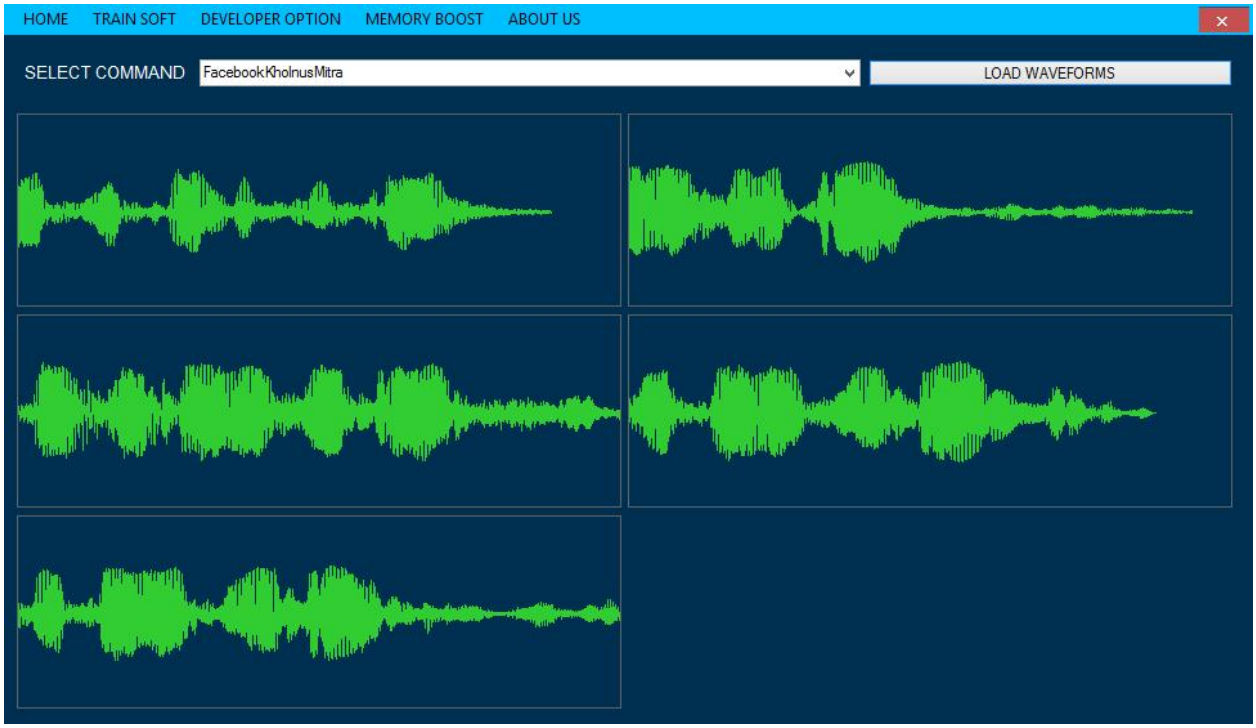

Figure A5.5: Waveform Layout

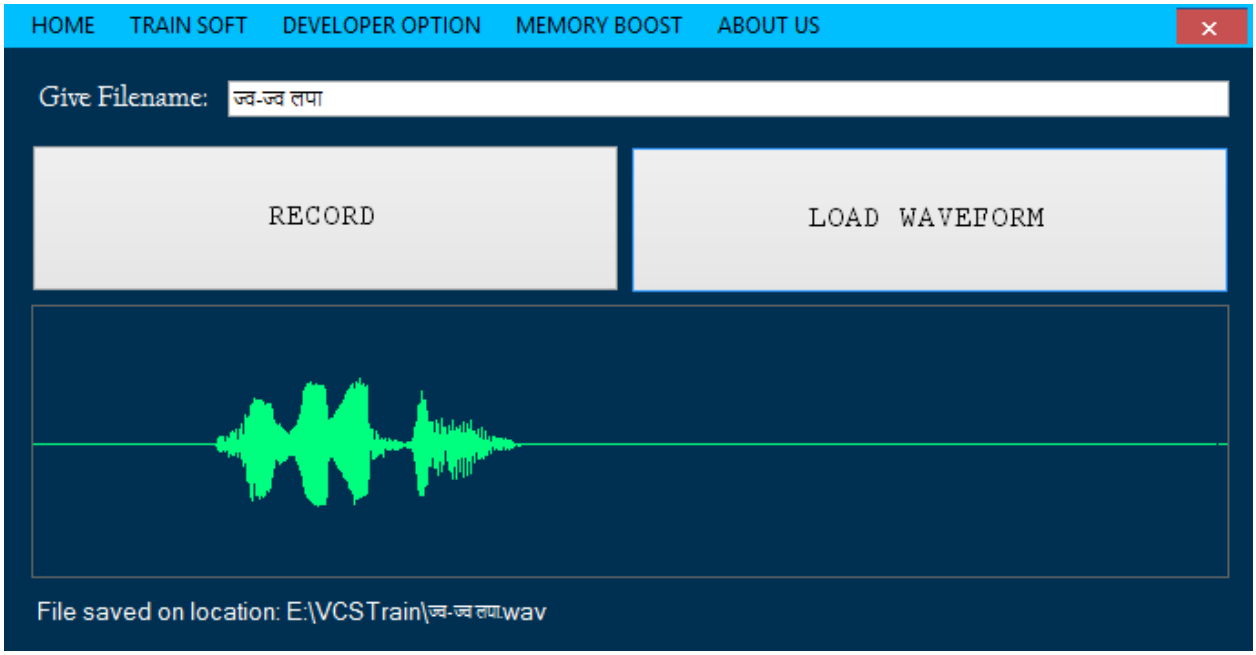

Figure A5.6: Audio Recorder

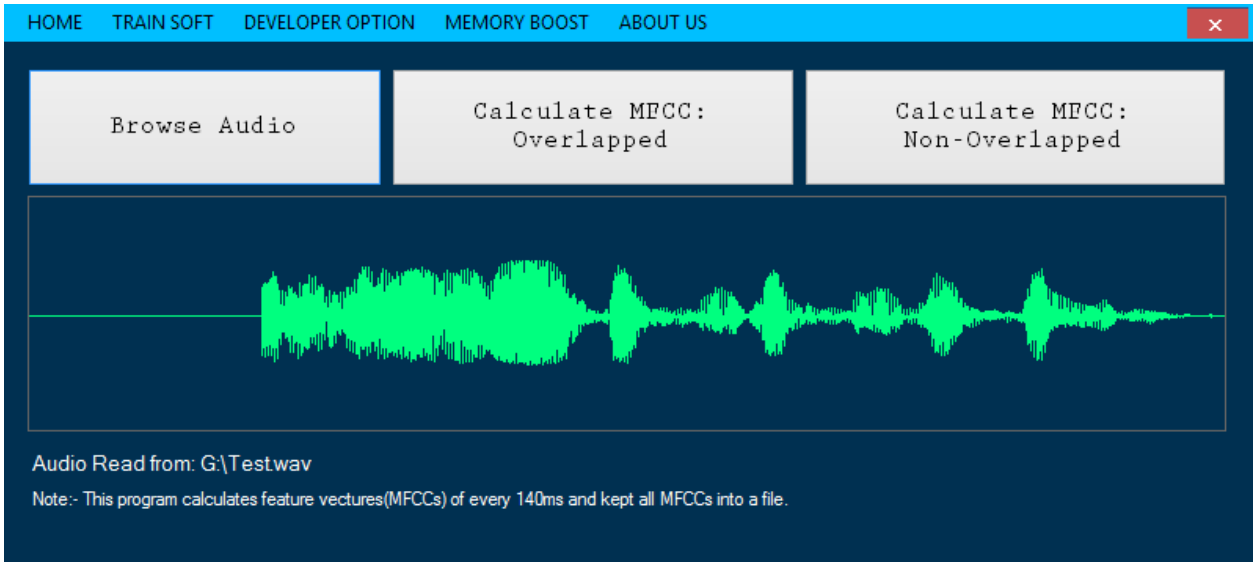

Figure A5.7: MFCC Calculation Panel

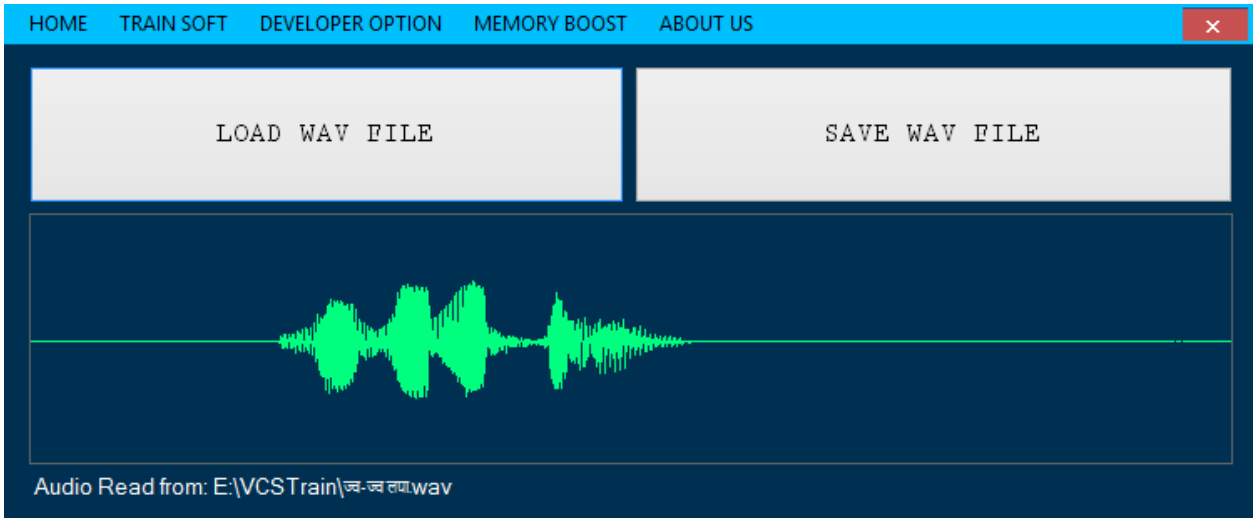

Figure A5.8: Generate Waveform

Figure A5.9: Trimming waveform

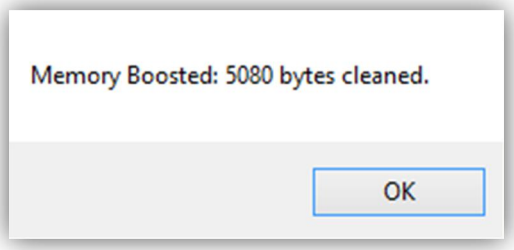

Figure A5.10: Memory boost by deleting temporary files