Voice Recognition and Control System (May, 2015)

Zhuoqi Li (George) and Bum Joong (Ben) Kim

***Abstract* – This project aims to build a voice recognition and control system which could accurately recognize human spoken words, differentiate between different people's voices and behave accordingly to the command understood. Based on the TI DSP chip LCDK (L138 development kit) and with a microphone to take speech input, this voice recognition system, called "Cereal" is a C-language based project able to recognize vocabulary and their speakers out of a 70-word database. With each person speaking the same 10 words, the database contains 6 people's voice information. Those ten words are: Cat, all, no, fit, cute, share, crazy, thank, sky and cough. The first five words contain five basic vowel sounds and the last five words contain both vowel and consonants. With aid of supervised machine learning algorithm, the successful recognition rate of the system can reach as high as 85 percent or higher.**

I. INTRODUCTION

Voice recognition technology has been widely applied in modern lives nowadays. From the most famous 'Siri' on iPhone, voice control entertainment system installed on luxury vehicles to interactive voice system for handling phone calls, voice recognition and control has been increasingly popular in the recent years.

A. Societal context

Although voice recognition sounds novel, this technology has long been applied in multiple fields in modern life. The most famous speech recognition system in recent years must be Siri, a magic software embedded on Iphone. Rather than another avant-garde but useless gadget that could fascinate no one but nerdy geeks, Siri was surprisingly convenient and practical – it enables people to 'communicate' with their smartphone by simply speaking to it, just like what they would do to another human being. Beyond this well-known example, there are many other similar systems people get touch with on daily bases. In one of those popular fields, another application of speech recognition that possesses great significance lies in the security system regime. Security system based on voice identification technology is as safe as, if not more, the fingerprint identification system. Seeing this great potential, our goal of this project is to build a prototypical voice recognition and control home security system.

B. Motivation

"Engine Starts", after Yuri says with his heavy Russian accent, the control panel of Bentley suddenly lights up and John's family could finally escaped the wrecked plane sliding to the edge of cliff. Inspired by this scene from the famous apocalypse movie '2012', we decided to 'duplicate' a similar system seen in the movie; a system that can not only distinguish different people's voices, but can also understand their spoken commands and behave accordingly.

C. Technical Issues and solution

Designing this challenging project, we encountered many technical issues. Because the whole system is written in c languages, a language both of us are not very familiar with, general coding issues have been the most challenging and annoying part of this project. Thanks to the help of TA Yuyu and Dr. Briggs, and with hours of self-learning online, we finally managed to overcome almost all coding issues encountered.

Another technical issue arises from the lab equipment. We realize that for different brand of microphone, their resolution and sensitivity differ. Features extracted from one microphone may never match with those captured by another. Same thing happens for the LCDK. Although all DSP chip provided in the lab belongs to the same type, their performance is not identical. Therefore, those uncertainty in the lab equipment makes the system extremely unstable. We finally overcame it by always using the same microphone and LCDK chip throughout the design.

Furthermore, the system takes way too long to process the speech input. This happens because the computational ability of LCDK cannot satisfy the processing requirement of those complex algorithms implemented in this project. We eventually improved the processing speed of the system by changing the optimization level. More details will be covered in the optimization level section.

D. Report outline

Starting from the system overview and technical background section, the report will first show the reader how this system operates to simulate a home security system, and articulate current features and function of this system. In the technical background section, we will break the system design into multiple components and briefly introduces the specific algorithm employed in each component. After giving the reader a general picture about what system does and how it is designed, in the system design section, readers will see design details of the whole system. The next section followed is the optimization level, which is an independent section mainly talking about how we utilize a feature embedded in the complier to enhance our system efficiency. Followed the optimization level section are the result and conclusion. We will evaluate the experimental results and performance of the system and discuss its successes and non-successes.

II. SYSTEM OVERVIEW

As stated in the introduction, the goal of this system is to build a prototype of a home security voice recognition system which can accurately interpret spoken commands, differentiate users' voices and operate accordingly. Because of the computational capacity of the chip and our time limit, instead of being able to process consecutive speech, 'Cereal' could only process one word each time. And so far it has only a 70-word library built by six different people's voices. It could interpret ten specified words spoken by six different people. We choose the following ten words as the basic test bunch: 'cat', 'fit', 'cute', 'no', 'all', 'share', 'cough', 'sky', 'crazy', 'thank'. The first five words contain the five basic vowels in English: A, E, I, O, U and the other five contain not only vowels but also consonants such as /s/ and /k/ which are usually hard to detect. We believe those ten words could serve as a good test-bunch for our system to help us figure out its voice recognition capacity.

The operational process of 'Cereal' is displayed in the following state diagram:

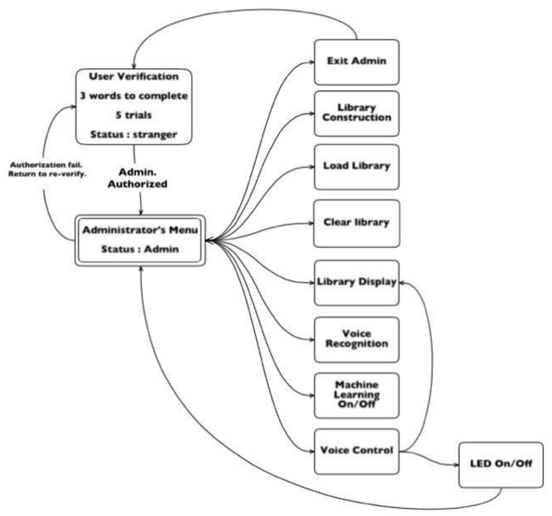


Figure 1. System operation flow chart

Entering the system, the user will first start from the 'user verification' stage. Similar to any real- life voice recognition security system, only the selected person, or say the host of the house can have access to modify or enter the system. In our project, one needs to pass three voice recognition tests in order to have access to system features. Users have to speak three words given by the system correctly and only when the system detects it is the admin speaking for all three cases will the user be authorized to enter the administrator menu.

In the admin menu, shown from the chart above, now the user has full access to all system features. He/she can then chooses to modify the library data, turn on/off machine learning algorithm or do voice control and so on. In the voice control function of our system, so far the user can only command the system to display its library and turn on/off LEDs on the DSP chip. But since the voice recognition part of the system is already built, theoretically the user can custom and add any voice control functions by storing new vocabulary commands in the library.

A. Technical Background

Although our system does have many features, the technical design can be mainly divided into two parts: voice recognition and voice control. The core of this project is the voice recognition process, which could be further divided into four sub-fields displayed in the diagram below:

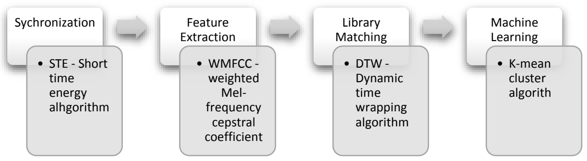

Sychronization
• STE - Short time energy alhgorithm
Feature Extraction
• WMFCC - weighted Mel-frequency cepstral coefficient
Library Matching
• DTW - Dynamic time wrapping algorithm
Machine Learning
• K-mean cluster algorith

Figure 2.Voice recognition processes and algorithms

Synchronization mechanism is introduced to differentiate the environmental noise from valid voice input. Because we want the system to only analyze valid speech input and filter out any environmental noise, synchronization is necessary as it makes the system to stay idle until speech comes in. To achieve this, we utilized the STE (short time energy) algorithm. It works to detect the valid voice input and filter out environmental noise by comparing their energy level in a short window.

After the synchronization process filters out the noise, feature extraction process starts. Input signal will be broke into frames and Mel-Frequency Cepstrum Coefficients algorithm will be applied to every frame to extract its power spectrum information in different frequency range. These extracted features of the speech input will be represented as MFCCs and WMFCCs (weighted MFCCs) which will be used in later processes.

Library matching process is where the system does the recognition. In this process, the system takes the MFCC feature values of the input calculated and compares with those in the library. The DTW (dynamic time wrapping) algorithm is employed to help find the best matched library data and display the matched library vocabulary as the result.

However, before we start to do data matching, we need to build up the library first. The database building process has the same first two steps as the voice recognition: the system first takes in speech input, extracts their frequency features in the form of MFCC values and then store those values in the database for later matching purpose.

But only with the library is not enough to achieve a high recognition rate because the way people speak, even for the same vocab, could vary. For example, users could change their intonation or speak quickly or slowly. In order to deal with this situation, a supervised machine learning algorithm, the k-mean cluster algorithm is implemented to help improve the library data. When the machine-learning option is turned on, users will need to judge the system's result and gives feedback. The library data will be updated accordingly and be more robust after every machine learning process.

B. *Standards Followed*

Throughout the design of this project, we followed several standards for voice recognition system. For example, we split the input signal into frames of 20 to 40ms, which is a very standard manipulation on speech input in voice recognition system [2]. Additionally, we applied the hamming window onto the frame and choose to use the first thirteen MFCC values as well as to use the dynamic time wrapping, which are also standards used in many other similar systems and researches [2].

C. *Development plan and work division*

This project is mainly built on the basis of Miniproject2, vowel detection system from EE113DA [4]. Start with the simple feature extraction and machine learning algorithm we already have in Miniproject2, we first add the synchronization mechanism to enable the system to handle longer speech input and filter out noise. Then we extend the MFCC feature extraction algorithm, enhance its efficiency and complexity in order to deal with longer and more complex data. Later, we implement library matching process using the DTW algorithm. At the same time, we make the system be able to load and write data from txt files. Machine learning, the mechanism we changed the least from the Miniproject2 was then added to close the voice recognition process of the system. But soon we realized only with MFCC algorithm is not enough to achieve high accuracy, therefore, we upgraded it to WMFCC. After the voice recognition were proven to be relatively robust, voice control features are gradually added to the system and the 'gui' (our interactive interface between user and the system) is facilitated. The last thing left to do is to sharpen the system library using our machine learning process.

The development of our system went through 10 stages. Start from version 1.0, where the system could only recognize simple vowels, it finalized at version 2.0, where the system is now able to accurately distinguish six people's voices, interpret spoken commands and operate accordingly. The following chart shows the historical version of our system and their features:

Version number	Features/Feature added
1.0	Synchronization, MFCC
1.1	Add machine learning
1.2	Add library loading, writing and clearing
1.3	Add DTW library matching
1.4	Add WMFCC
1.5	Machine learning algorithm modified
1.6	Add voice control, extend library to 60
1.7	10 more voice control commands added to library
1.8	System able to turn on LEDs and display Library
1.9	Gui facilitated
2.0	System design finalized

In order to finish this challenging project on time with quality, team work is necessary. To avoid wasting labors, we two have our own assigned portion of the system to design. For challenging sections, we would worked together. I am in charge of the synchronization, MFCC feature extraction and the machine learning mechanism while Ben is responsible for library matching, the DTW and upgrading MFCC to WMFCC algorithm. As for the voice control and debugging, we two worked together. Same thing applies for the report write-up.

III. SYTEM DESIGN

A. Synchronization

Synchronization is a very important step in this system as it differentiates noise from valid voice input. Because we do not want the system start to analyze the input as soon as we turn it on, not until valid voice input – spoken words come in, a mechanism to detect the start of the valid input needed to be implement. By differentiating the environmental noise from human voice, synchronization process enables the system to capture complete voice input without losing any data or adding extra noise.

The idea behind the synchronization process is very simple. Since we are using a microphone to capture the voice input, as observed in the following diagram, when there is no input (with environmental noise), the amplitude of captured input value is very smooth, exhibited a logarithm pattern.

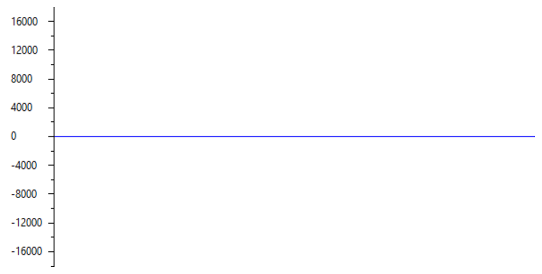


Figure 3.Amplitude diagram for no input

However, when speech input comes in, amplitudes between inputs samples will 'jump'. Between certain samples, amplitudes could differ by the order of three or even higher; and it is not just an occasional events, such phenomenon could be observed throughout the input samples.

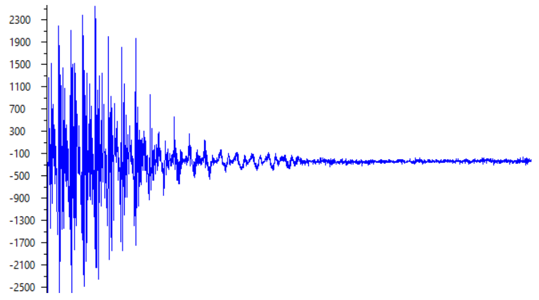


Figure 4.Amplitude diagram for word 'one'

As shown from two diagrams above, we can see that the amplitude of valid input could suddenly jumps to high as 2000 and drops to -2000 at the next sample. Therefore, for valid voice input, it must have high energy while noise or no input will has a relatively small energy.

Based on this observation, the short time energy algorithm is implemented to distinguish the noise from valid input:

$$E = \sum_{m=n-N+1}^{n} [x(m)w(n-m)]^2 \quad [1]$$

Labeled as *w(n)* in the formula, a small window containing three samples is implemented to and we calculate the short time energy for the three samples in the window. As the system is booted, it will continuously check the short time energy for every three samples; as the short time energy calculated exceeds certain threshold, the system knows that some valid input come in and it will initiate the voice recognition progress. The reason that we choose the window size to be as small as three samples is that it makes sure the calculating speed could catch up with the sampling rate and a short window size could ensure it does not lose any data (the valid input samples start to be recorded right after the check window).

By setting proper energy threshold, STE window can also filter noise. As the threshold is set to millions (8 million for general cases), any sudden jumps with the value change less than the level of thousand will be automatically ignored. According to the sensitivity of microphones used, the threshold can be adjusted to achieve optimal input capturing capacity.

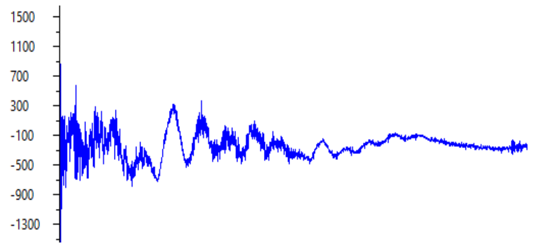


Figure 5.Example of environmental noise

The figure above exhibit a typical environmental noise (generate by moving the microphone). A threshold of 4 million filter this noise out totally as most of its amplitude unable to exceed one thousand, which would yield million-level energy.

B. Voice Recognition

As the core of the whole system, voice recognition contains many sub-processes:

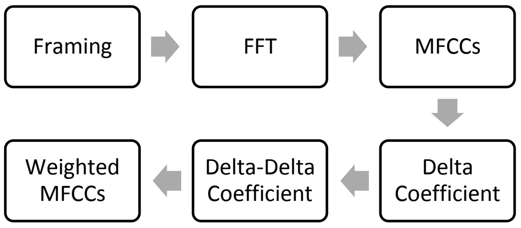


Figure 6.Voice recognition process flow chart

Start with framing, the input data will go through six processes and finally reaches MFCC values, of which represent inputs' features in different frequency range. And those values will be further used in the machine learning and library matching process.

Framing

The very first step in the process of speech analysis is to record the input speech samples. For our system, using a microphone as the receiver, the system samples the input at the rate of 8000 samples per second; in other words, it takes in a sample every 0.125 millisecond. For every analysis process, the system will only take in 8192 samples, or say a 1.024s voice input at a time and store those raw input data into its flash memory for later process. These 8192 data samples will be further divided into 32 frames with each frame it contains 256 samples (32ms). For each frame of data samples, same feature extraction manipulation will be applied individually and the sum of those frames' featured will be collected again at the end of the feature extraction process.

For each frame of input samples, a filtering window with the same size as the frame itself, the hamming window will be applied to prepare us for later transform and frequency domain manipulation:

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

Where N is the window size which in our case is 256 and n is from 0 to 255. Each sample in the frame will be multiplied by the value of w(n).

The hamming window is used to smooth out the audio signal at the beginning and end of the frames by weighing the samples within the window differently. This is important because applying the Fast Fourier Transform (FFT) directly to the sampled audio input would cause the sampled audio input to appear as though there was high frequency noise from what may be discontinuities and distortion in the audio signal [3] Also, computer cannot do Fourier Transform on infinite samples, therefore, a finite sample size has to be chosen by applying windows.

Containing 256 samples, each frame has a 32 micro-second length of voice input. This is actually a quite large window size in the time domain. In the frequency domain, however, a long time domain window yields a narrow band frequency window, which will make the pitch of human voice input more resolved and make voice recognition easier. That's the reason we choose a 256-sample frame.

Fast Fourier Transform (FFT)

In order to analyze the input in the frequency domain, the N point (256 in this case) FFT manipulation is necessary. The FFT algorithm is applied on the periodogram of input samples in each frame, which is the product of raw sample amplitude and the Hamming window function.

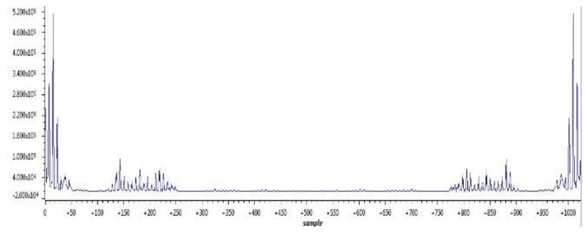


Figure 7.1024 samples FFT output

After the FFT manipulation, we would like to extract its power spectral density for later MFCC values calculation.

$$PSD(k) = \frac{|FFT|^2}{N}, k = 0 \dots . \frac{N}{2} - 1$$

As figure 9 demonstrates, the output after applying FFT is symmetrical. Therefore, half of the result is redundant, so we only need the first half of the output. By obtaining the periodogram, it helps the system to identify which frequency is present in the frame [3].

MFCCs (Mel Frequency Cepstral Coefficients)

The periodogram spectra still contains a lot of unnecessary information, we apply the filter bank to get an idea of how much energy exists in various frequency regions [3]. The filter bank is composed of 26 triangular shaped band-pass filter described below [4]:

$$H_m(k) = \begin{cases} 0 & k < f(m-1) \\ \frac{k - f(m-1)}{f(m) - f(m-1)} & f(m-1) \le k \le f(m) \\ \frac{f(m+1) - k}{f(m+1) - f(m)} & f(m) \le k \le f(m+1) \\ 0 & k > f(m+1) \end{cases}$$

Note that f (m) used in the formula above is all in the Mel-frequency domain which is calculated by the equation shown below [5]:

$$f_{mel} = 2595\log(1 + \frac{f}{700Hz})$$

The reason that we do the conversion because the Mel-scale relates perceived frequency, or pitch, of a pure tone to its actual measured frequency. By using thise scale, it makes our system more resemble to the human hearing [3].

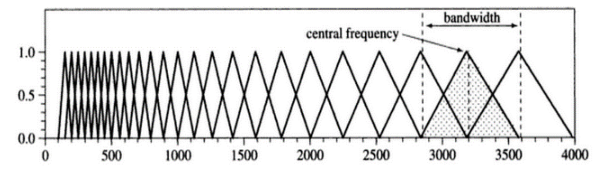


Figure 8.Triangular Filters for 26 frequency groups [5]

According to figure 10 above, the bandwidth of those triangular filters increases with the frequency. At low frequency, many filters crowd in a small range of frequency with each filter has a small bandwidth while at high frequency range, filters spread out. This makes senses because MFCCs system is based on human perception capabilities. The human ear resolves frequencies in different ways: the higher the frequency of the speech signal, the lower the resolution [5]. Since we care less about what happen in the high frequency region, the filter bank distribution is designed to be much sparse there.

To calculate the MFCCs value, we multiply the periodogram spectra with the 26 filter banks and sum up the resulted energy components in each filter, giving out 26 filter energy values. Then we take the logarithm on those 26 values to get their 26 log energy values labeling as X_m. After that, we perform the Discrete Cosine Transform (DCT) on X_m [5].

$$MFCC(i) = \sum_{m=1}^{M} X_m \cos\left(i\left(m - \frac{1}{2} \right) \frac{\pi}{M} \right), \quad i = 1 \dots Z.$$

In the MFCC equation above, M is the number of triangular filters, which in our case is 26. And Z = 13 because we need 13 MFCC values. MFCC value calculation is performed on every frame of the 32 frames for one voice input. That is to say, for every word the user says to Cereal, 13 times 32, 416 in total of MFCC values are extracted for later manipulation. Those four hundred MFCC values represent “features” of the voice input which are used by the system to do the differentiation.

Delta and Delta-Deltas Coefficients (WMFCC)

Delta and Delta-Deltas Coefficients are also known as the differential and acceleration coefficients. While MFCC coefficients describe the power spectral of each frame, additional features, such as the change of MFCC values over time is also helpful [2]. The trend of the speech signal is lost when we do frame by frame analysis. To recover the trend information, Delta and double Delta Coefficients are used. Although the location of the formant of the speech varies from person to person, the time trend of the formant is quite constant among different speakers. The trend information, represented by delta and double delta features, is therefore important for improving the robustness of the recognition [6]. To calculate the delta coefficient, the following equation is used:

$$d_t = \frac{\sum_{n=1}^{N} n(c_{t+n} - c_{t-n})}{2\sum_{n=1}^{N} n^2}$$

Where c represents MFCC element and dt represents delta coefficient for element t of the MFCC feature vector [2]. To calculate the double delta coefficient, the above equation will need to be applied again but this time c is the delta coefficient calculated.

Since we already have thirteen MFCC values, there will be thirteen delta and another thirteen delta-deltas coefficients for each frame which combines as the weighted MFCCs. So now, we have 39 feature coefficients for each frame and 1248 coefficients in total for one speech input.

C. Library Matching

Dynamic Time Warping

Uncertain nature of the voice varies signal in speed and time, so that the capturing correct information requires more sensitivity in capturing signal dimensions than of Euclidian distance between the two signals [2]. Thereby, to capture distortion in time axis between signals instead of using the Euclidian distance, we utilize Dynamic Time Warping (DTW) as a matching algorithm.

The Dynamic Time Warping algorithm is a technique vastly used in voice recognition since 1980s [6]. It serves to two of voice signal in time series that are similar but locally out of phase to align in a non-linear manner. In spite of its time complexity, The lower bound requires that the two sequences being compared are of the same length, and that the amount of warping is constrained.

In our design, DTW allows us to find the similarities between the template WMFCC feature vector in library database and WMFCC feature vector of the new input voice signal in a non-linear mapping of input voice and helps to minimize the distance between the two, of each frame as it accounts for changes in speed and time.

First distances between all pairs of elements between the template from the library database ($WFMCC_n$) and new input WMFCC feature vector ($WFMCC_m$) are calculated, this is calculated in equation below, where ith and jth element of the template and new WMFCC feature vector.

$$Distance\ (n, m) = (WFMCC_n - WFMCC_m)^2$$

After getting information about the difference in distance, in order to calculate the deviation of dimension, DTW matrix was utilized at each dimension of matching segment. The DTW matrix receives frequency information with accumulated minimum distances to calculate the warping path [6]. The path at the DTW matrix begins at coordinate (0,0) and ends at (12,12).

The warping path of distance matching in the DTW matrix are calculated using conditional below[8], with the nth and mth element of the template and new WMFCC feature vector and the corresponding distance matrix.

$$DTW\,(n, m) = min\begin{cases} DTW\ [n-1, m\ -\ 1] \\ DTW\ [n-1, m\ ] \\ DTW\ [n,\ m\ -\ 1] \end{cases} +\ gap[n, m]$$

In order to initiate the cumulative addition of the warping path, first row and column of the DTW matrix has initial axis values with the following math

DTW (n, 0) = DTW [n-1, 0] + distance [n, 0] (template)

DTW (0, m) = DTW [0, m-1] + distance [m, 0] (Input)

The following matrix diagram shows an simpler example of the 4 x 4 DTW matrix that represent 16 elements (instead of 169 elements of the complete 13 x 13 DTW matrix used) TFV and IFV are the template feature vectors and new Input feature vectors [2].

Template Feature Vector , TFV(n)	3	D=C+ [TFV(3)-IFV(0)]	J= min(I,C,D) + [TFV(3)-IFV(1)]	M= min(I,C,D) + [TFV(3)-IFV(2)]	P= min(I,C,D) + [TFV(3)-IFV(3)]
	2	C=B+ [TFV(2)-IFV(0)]	I= min(H,B,C) + [TFV(2) -IFV(1)]	L= min(K,H,I) + [TFV(2)-IFV(2)]	O= min(H,B,C) + [TFV(2)-IFV(3)]
	1	B= A+ [TFV(1)-IFV(0)]	H= min(A,B,E) + [TFV(1)-IFV(1)]	K= min(F,E,H) + [TFV(1)-IFV(2)]	N= min(G,F,K) + [TFV(1)-IFV(3)]

	0	A= [TFV(0)- IFV(1)]	E=A+ [TFV(0) - IFV(1)]	F=E+ [TFV(0)- IFV(2)]	G=F+ [TFV(0) - IFV(3)]
n/m		0	1	2	3
		Input Feature Vector : IFV(m)			

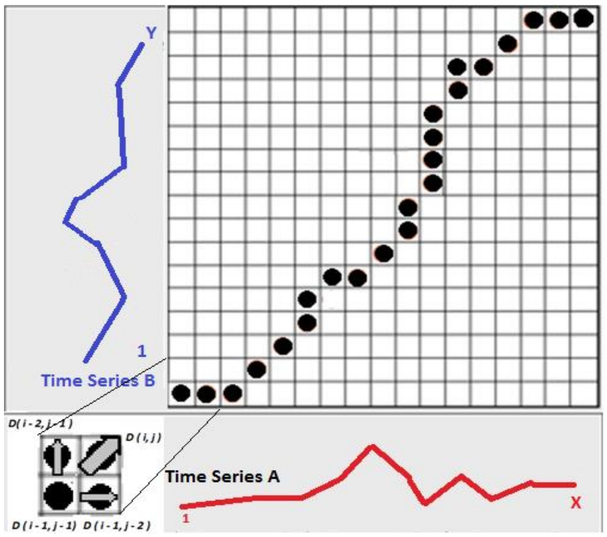


Figure 9.DTW matrix warping path [6]

Time series B on Y axis is template vector and Time series A on X axis is new input feature vector. Black dots indicates the warping path from the diagram left bottom corner, the path may move forward in indices, either in three directions (above, right, or diagonal(match)) [8]

The warping path allows us to set the minimum distances in newly spoken input and template library and thereby check similarity of both at the end of the path.

In addition, in order to utilize the warping path to determine the identity of the word spoken, model of the lowest level of deviation from the warping path should be located, and the effective path summation cost of DTW matrix the warping path takes has to be calculated. The summation of the path costs for every frame is how much the word spoken has deviated from the template. After calculating the deviation, we can locate the library index in the lowest deviating example.

D. Machine Learning

After the Library Matching with Dynamic Time Warping, the newly recognized WMFCC feature vectors from input is also used to allow the machine to learn. The machine learning algorithm in speech recognition of phonemes utilized simple averaging of the respective elements in the template and new feature vectors knowing as the K-mean cluster algorithm [4]. The equation is following:

$$WMFCC_y\,(i) = \frac{WMFCC_y\,(\mathrm{i})\mathrm{x}(\mathrm{L}-1) + WMFCC_{new}(i)}{L}\ \mathrm{i}$$

Our algorithm to capture the voice in each setting does not always provide us full information of the voice in each time of capture. Therefore, the warping path of newly determine the best matched WMFCC feature vector elements shall be counted with the path cost to later add and average them and write into the template WMFCC feature vector.

IV. OPTIMIZATION LEVEL

As we can see from the previous sections, in the voice recognition and library matching processes, there are quite a number of mathematical manipulations involved in multiple sophisticated algorithms we utilized. For each one-second speech input, 1248 feature coefficients including MFCCs and delta-delta coefficients are generated. Moreover, to get every one of those 1248 coefficients, countless mathematical calculation including FFT, DCT logarithm and square rooting were used. Besides the depth of the calculation level, the scope of the calculation level of our system is also monstrous. The system needs to deal with tens of thousands input samples and also humongous algorithm output (the DTW takes uses of four dimensional array). Although DSP chip is good at mathematical manipulation such as multiplication and division, however, facing the complexity of our system, the processing capacity of the LCDK is far from enough.

Optimization level	**Processing time (s)**
0	22.5
1	14.5
2	5.45
3	4.5

From the chart above, before we used the optimization level feature embedded in the Code composer studio (optimization level 0), for each speech input, which is only one second long, it takes the system 22.5 second to display the result. No doubt, this processing speed is too slow to be tolerated.

However, there is not much we can do on the code level to increase the processing speed because all the complicate mathematical operations are necessary and cannot be taken out. Luckily, we found we could use the ‘Optimization level’ feature embedded in the compiler CCS5 to improve our processing speed. By increase the optimization level, the compiler increase the compiling speed considerably but with some trade-off in the code file size [7], which is not much of a concern in our project. From the chart above, it shows that as the optimization level increases, the processing time decreases exponentially. We keep the optimization level of the compiler at level 3 from then on; although 4.5 second is still a pretty long processing time, it is definitely tolerable. Compared with the original speed, with optimization level 3, now our processing speed is 5 times faster.

V. EXPERIMENTAL RESULTS

Experimental procedure

As mentioned in the introduction section, 10 words, with one group of five with clear vowels and the other with obvious consonants, were chosen to test the accuracy rate of our program. Since the system already has a function to track the successful rate and is able displays it to the user, we can simply test the program by directly speaking those test words and copy down the accuracy values.

To investigate the impact of different algorithm on the accuracy rate, different versions of the system with different featured embedded are tested. We choose version 1.3, 1.4 and 1.8 to be tested and for each version, we let four different people go through those ten test words in sequence with a total of 40 trials.

Version 1.3 is the most basic version which only contains primitive MFCC and DTW algorithm. While in version 1.4, WMFCC were added into the feature extraction process. And in version 1.8, machine learning algorithm has been built and the library is already well sharpened. This way, from the result we can how WMFCC and machine learning (sharpen of library) will impact the system successes.

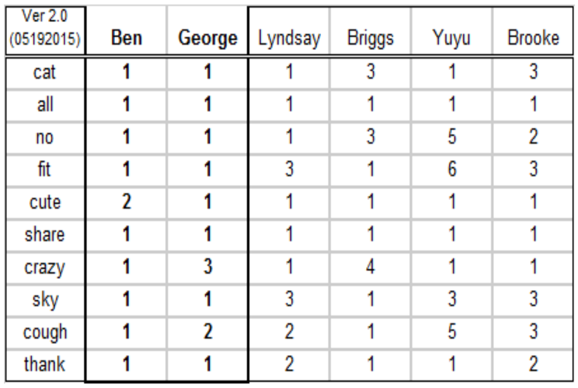
After we ran three 40-trial tests on these three versions, we test the final version of the system, version 2.0 with every word in the sixty-word library and record the times we try on each word to get both speaker and vocab right. Through this stricter test, we can know the system’s performance on each vocabularies and generalize a pattern to evaluate its reliability.

Experimental results

The chart below shows the experimental results of our voice recognition system with different versions:

	Version 1.3	**Version 1.4**	**Version 1.8**
Total Recognition rate	**65.0%**	**77.5%**	**85.7%**
Speaker Recognition rate	**75.0%**	**87.5%**	**85.7%**
Vocab Recognition rate	**90.0%**	**87.5%**	**100.0%**
Trial times	**40**	**40**	**42**

As we can see from the chart above, the accuracy of our system is pretty impressive. In primitive version 1.3, the system is already able to achieve a high rate; especially the vocab recognition rate, it reaches as high as 90%. The chart also reflects the WMFCC and machine learning algorithm definitely have a positive impact on the accuracy. All three recognition rate increase a bit as the algorithm is added to the system. In the version 1.8, total and speaker rate reaches higher than 85%. More impressively, the system gives out the right vocab recognition for more than 40 tests.



Ver 2.0 (05192015)	**Ben**	**George**	Lyndsay	Briggs	Yuyu	Brooke
cat	**1**	**1**	1	3	1	3
all	**1**	**1**	1	1	1	1
no	**1**	**1**	1	3	5	2
fit	**1**	**1**	3	1	6	3
cute	**2**	**1**	1	1	1	1
share	**1**	**1**	1	1	1	1
crazy	**1**	**3**	1	4	1	1
sky	**1**	**1**	3	1	3	3
cough	**1**	**2**	2	1	5	3
thank	**1**	**1**	2	1	1	2

Those six different people are randomly chosen and they differ with each in sex, age and races. The first two column shows the successful rate for two developers.

The diagram belows shows the recognition rate in graph, which is eassier for comparision:

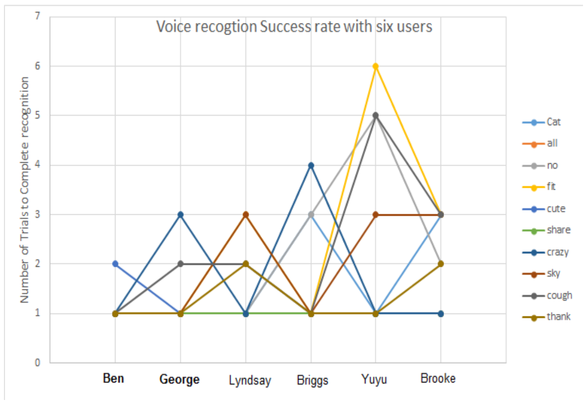


Figure 10.Test result of recording number of trials with six different speakers to complete a full identification of word and speaker.

From previous data on three historical versions, we learned that after the machine learning to sharpen library template, voice recognition and identification rate improved significantly. Same thing is reflected here as well. As shown in the graph view of Success rate, speakers from the left side of the diagram (Ben and George), who have more machine learning updates, has shorter time to complete the identification. While for people with less machine learning updates, it takes more times to achieve the recognition.

Moreover, another interesting fact we found from above is that vocabs with clear consonants, last five from the list, have higher vocab recognition rate, which tells our system better precision of extracting information from the consonants. However, in other similar systems, the opposite is true.

VI. CONCLUSION

According to the experimental results section, the voice recognition system is definitely a success. Our system succeeds to accomplish every objectives and goals we set at the very beginning of this project. Proved by the experimental result, the system can accurately identify the vocabulary and its speaker for most of the times (8 out of 10 trials) from a library containing more than 60 vocabularies spoken by six different people. The word interpretation rate is especially surprising as it achieves 100% accuracy for 40 trials. Beyond the voice recognition portion of the project, the voice control part is as successful. Although so far there are not much things users could command the system to do, but theoretically we can add as many voice control features as we can because the system is now able to 'interpret' spoken commands with voice recognition algorithm. Another success of this system is that we have a user friendly interactive interface that enable common people, even with no programming background, to use our system with little issue and confusion.

There are also non-successes in this project. Although we had been always trying to make the system more tolerant to speech input with different intonations, but even at the end of this project design, intonation of the speech still has a huge impact on the result. If the library vocab is stored with interrogative intonation, the system will hardly get it correct when the user says the same word but with an exclamatory intonation. Another non-success we consider in the project is that the machine learning algorithm we use is not robust enough. Despite the fact that the system eventually expresses a high accuracy rate, however, this is because we have sharpen the libraries for so many times using the machine learning algorithm, especially when have a large database. Even by doing that we still run into cases when the system has hard time giving out the right result.

Despite the non-successes, our system does have much potential and serves as a good prototype of voice recognition home security system. In order to put this system into production, although the accuracy is pretty high, however, the stability and reliability needs to be improved to deal with daily lives situations. To resolve this problem, a more advanced machine learning mechanism needs to be employed to replace the K-mean cluster algorithm to make the system more resistance to uncertainties. Also, the current 70 – word library is definitely not enough for a nature system. To deal with a much database, the resolution of voice recognition needs to improve to ensure accuracy. A shortcut to achieve that is to double the system sampling rate from the 8000 samples per second (the rate used in this project) to 16,000 samples per second. However, this improvement will bring heavy calculation burden to the DSP chip. Moreover, although we managed to improve the process speed by a factor of five, but five second is still quite too long and will cause inconvenience. To resolve these problems, the processing capacity of the DSP chip is the key in the sense that more advance algorithms need to be implement to enhance the system performance. Hence, instead of using LCDK chip with limited capacity, our system should move to a more advanced platform with much faster processing speed and also larger memory space.

VII. REFERENCE

[1] M. Jalil, F.A. Butt, A. Malik, "Short-time Energy, Magnitude, Zero Crossing Rate and Autocorrelation Measurement for Discriminating Voiced and Unvoiced Segments of Speech Signals." *IEEE Xplore*: 9 May 2013 http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6557272 [Accessed 11 June 2015]

[2] S. Xie, H. Abulebda, "Character Recognition Utilizing scratch sound." March 2015.

[3] J. Lyons, "Mel-Frequency Cepstral Coefficient (MFCC) tutorial," *Practical Cryptography*: 2013. http://practicalcerptography.com [Accessed 11 June 2015]

[4] M. Briggs, *Mini-Project 2: Vowel Recognition:* Feb. 2015. EE113DA – Digital signal processing design

[5] Paulus, Dietrich W. R., and Joachim Hornegger. "Mel Spectral and Cepstral Features." in *Applied Pattern Recognition: Algorithms and Implementation in C++. 4th ed.* GWV-Vieweg, 275 - 276. Print. [Accessed from Google books, 11 June 2015]

[6] V.chapaneri, Santosh. "Spoken Digits Recognition Using Weighted MFCC and Improved Features for Dynamic Time Warping." *International Journal of Computer Applications IJCA (2012)*: 6-12. Print.

[7] "Optimizer Assistant." - *Texas Instruments Wiki*: 23 Jan. 2014 http://processors.wiki.ti.com/index.php/Optimizer_Assistant [Accessed 12 June 2015]

[8] M Muller, "Information Retrieval for Music and Motion" Springer, vol. 16, no. 4, 69-76, 2007