Character Recognition Utilizing Scratch Sound (March 2015)

Susan Xie and Heba AbuLebda

99

Excellent Report

***Abstract*—This project aims to recognize and differentiate between at least two English letters through the sound produced by scratching the letter on a rough surface. In order to do so, milestones for noise removal, feature extraction, and robust machine learning algorithm is necessary. Test results have shown that a large database of features extractions is will significantly lower the success rate. Thereby, we used a database of six letters: M, K, T, A, B, and C. We have observed a recognition rate of nearly 100% for five of the characters in the memory bank for six unique English alphabet letters and roughly 50% success rate for the sixth letter, respectively.**

***Index Terms*—Character recognition, digital signal processing, discrete cosine transforms, machine learning, noise removal, fast Fourier transforms, pattern recognition, supervised learning, text recognition, time series analysis**

I. Introduction

THE growing use of handheld devices such as smartphones and tablets have spurned growth in Human-Computer Interaction (HCI). Smartphone, tablet, and other touch-screen device users typically use a stylus or fingers to indicate their actions to the device. Many devices use complex systems to analyze the strokes made by the user to determine what action to take. In our project, the sound produced by scratching a pen against a rough surface when writing an alphabet character can allow us to determine the written character. Our idea is based on the research "Recognizing Text Through Sound Alone" by Li and Hammond from Texas A&M University. Instead of analyzing the speed of the strokes to determine the stroke patterns, we utilize the sound produced when writing characters on a rough surface.

Their inspiration for their research is from a game played where one person scratches a letter on paper and the other person has to guess the letter [1]. Research regarding the recognition of alphabet characters through sketch reorganization has been done before. In those research, speed of the strokes was an identifying characteristic of the characters to recognize the corners [2]. Additional studies of scratch inputs to recognize characters have utilized equipment such as digital stethoscopes to amplify sounds [2]. The results of these studies had shown 90% accuracy for six scratch gestures [3]. We are basing our project on the research conducted by Li and Hammand, who utilized a basic built-in laptop microphone to extract the features of the scratch signal to recognize 26 english alphabet characters [1].

Fig. 1 shows the unique stroke order for each capitalized letter of the English alphabet

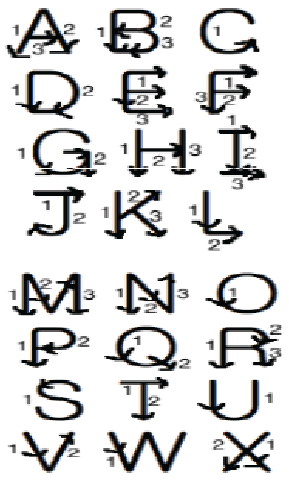

Fig. 1. Stroke orders of English Alphabet Characters [1]

Some societal and sustainability constraints to the results and implementation of this research for new technological devices are the various representations of a single character. In order to be truly helpful and easily implemented, the system must be robust to properly remove the environmental noise and perform fast feature extraction. In addition, the current machine learning algorithm must be supervised so calibration is necessary. However, with additional time and more robust implementations into our current project, the impact of these constraints may be reduced.

II. Motivation

We chose this project because it is challenging project that expands on the topics and applications of digital signal processing (DSP). Specifically, this project will expand on our knowledge on the use of MFCC and dynamic time warping. This project is interesting also because it is an alternative means of HCI. For example, people with disabilities could find this technology to be helpful in learning or daily life needs of communicating with others.

III. Approach

A. Team Organization

Both Heba and Susan have researched the material and process needed for this project. For example, Heba has researched and worked on the pseudo-code necessary for the noise removal of the scratch signal. Susan have researched and worked on the necessary algorithms for feature extraction and machine learning. Both Heba and Susan have worked on testing and fixing problems that were encountered throughout the progression of the project. Tasks were done in parallel, where Heba and Susan would perform independent research and tasks relating to separate sections of the theoretical basis of the project.

B. Development Plan

Our procedure started with completing the code for noise reduction by comparing energy of noise and energy of actual signal. Then we worked on verifying the robustness of the noise reduction code to identify noise versus actual scratch signal. Then we started working on the feature extraction and dynamic time warping where feature extraction was done by using Mel Frequency Cepstrum Coefficient (MFCC). Dynamic time warping was used to allow for a better analysis of the MFCC values due to the variance in speed that our scratch signal might have.

C. Standards

Since character recognition through scratch signals is not a standardized research, there are no de jure standards. Instead, we had utilized de facto standards in our project. These standards include segmenting a signal into frames of 20 to 40ms [4]. In addition, in characterizing the signal we have followed the de factor standards set by speech recognition research. This includes the use of a Hamming window prior onto the frame and use of only the first thirteen MFCC values in their feature vector when characterizing the frame [5].

In the machine learning algorithm, we had chosen to use dynamic time warping and followed the calculations by the de facto standards created through numerous research and use of this method.

D. Theoretical Basis

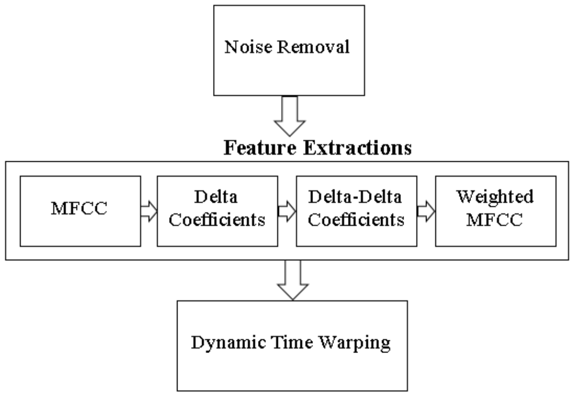

Fig. 2. Diagram of system blocks of scratch signal recognition

Noise Removal

We measured the energy of the noise at the first half second of the recording through calculating the average signal energy for 10 frames, where each frame is 50 ms. Refer to (1) below to calculate the energy of each frame.

$$E = \frac{1}{n}\sum_{i=1}^{n} S_i^2 \tag{1}$$

After the noise energy is determined, the energy of every 45ms frame is calculated and compared to the noise energy, which was measured in the first half second of the recording. When the energy of the signal is higher than the energy of the noise by a certain threshold, the program knows that this is the beginning of the desired input and eliminates all the noise before it.

In order to remove the noise from the end of the signal, we developed a counter to track the number of consecutive frames that has an energy lower than the threshold noise energy. If the number of consecutive frames is higher than the threshold number of frames, 12, then the end of our signal was detected and these twelve frames are considered to be noise.

Framing and Hamming Window

After start and end-point noise removal of the input signal, the resulting scratch signal is segmented into frames, each containing 256 samples. We cannot expect the scratch signal to contain a multiple of 256 samples, thereby zero padding is necessary. We chose to segment the signal into frames of this size, because of the calculations necessary for feature extraction, such as N-Point FFT, where N is a power of two. In addition, the sampling rate is 8000 samples per second, thereby, 256 samples represents a 32ms frame, following a typical de fact standard for a reliable spectral estimate [4].

Prior to performing the N-point FFT of a frame of a signal, a Hamming frame is applied to it. The purpose of the Hamming window is to prevent spectral leakage, shown as tails to form around the peaks of the FFT of the sine waves associated with the signal, shown in Fig. 3 [5].

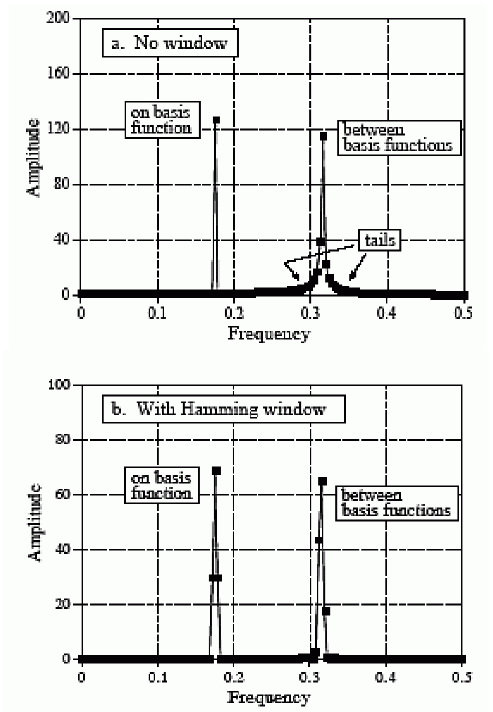

Fig. 3. (a) Magnitude frequency spectrum of signal with only two sine waves, without windowing. (b) Same magnitude frequency spectrum as (a) with Hamming window applied [5].

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 [4].

The Hamming window equation is displayed below in (2) with N = 256.

$$W(n) = 0.46 - 0.54 * \cos\left(\frac{2\pi n}{N-1}\right) \tag{2}$$

The Hamming window is applied by multiplying every element, n, in the signal frame by element n of W(n).

Feature Extraction

The feature extraction is performed on every frame of the scratch signal. In this project, frames are characterized through Weight Mel Frequency Cepstral coefficients (WMFCC). The WMFCC is calculated by first extracting thirteen Mel Frequency Cepstral Coefficients of the frame and the delta and delta-delta coefficients. The WMFCC is the resulting linear combination of the MFCC, delta, and delta-delta coefficients.

Mel Frequency Cepstral Coefficients (MFCCs)

In order to characterize the scratch signal, Mel Frequency Cepstral Coefficients (MFCCs) were used, similar to its application for speech recognition. In each frame, a feature vector of thirteen MFCC values is extracted to characterize each frame. Each element of the MFCC feature vector is the result of the Discrete Cosine Transform (DCT) of the energy at the output of 26 bandpass filter banks [6].

The periodogram estimate of the power spectral density is performed on the resulting frame after the Hamming window was applied. It is necessary for every frame to contain 256 elements, in order to perform a N-point Fast Fourier Transform (FFT) magnitude of. Every element of the resulting FFT frame is then squared and divided by the number of elements in the frame, 256. The first K elements (3) is the periodogram estimate of the power spectral density of the windowed frame.

$$K = \frac{N}{2} - 1 \tag{3}$$

The periodogram spectral estimate contains information about the audio signal that is unnecessary for our speech recognition project. So, we constructed the filter bank which is composed of M=26 bandpass filters, where each of their frequency responses are a triangular shape, as described in the piecewise function (4) below.

$$H_m(k) = \begin{cases} 0, & k < f(m-1) \\ \dfrac{k - f(m-1)}{f(m) - f(m-1)}, & f(m-1) \le k \le f(m) \\ \dfrac{f(m+1) - k}{f(m+1) - f(m)}, & f(m) \le k \le f(m+1) \\ 0, & k > f(m+1) \end{cases} \tag{4}$$

The filter banks will allow us to properly determine how much energy is present in the scratch signal at certain frequency regions. Following speech recognition research, the size of the filter bank is M=26 number of triangular bandpass filters [4]. In order to determine the spacing and width of the bandpass filters, the minimum and maximum frequencies of the scratch signal is converted from the frequency scale to the Mel scale using (5) below, where f is the frequency.

$$Mel\ Frequency = 1125 * \ln(1 + \frac{f}{700}) \tag{5}$$

The Mel scale conversion relates the actual measured frequency to the tone and pitch perceived by the human ear and is used to better match the features of our audio signal to what the human ear hears [4]. We defined the minimum and maximum frequency, μ (0) and μ (M +1), of the actual frequency range to be 0 Hz and 4000 Hz, respectively. By applying (5) to convert the lowest and highest frequency to Mel scale (ϖ), we can calculate 26 equally spaced Mel scale frequencies between ϖ (1) to ϖ(256) of each 256 samples frame. We then convert the 28 Mel scale frequencies consisting of the upper bound, lower bound, and 26 equally spaced Mel scale frequencies to actual frequencies in Hz. Using the boundaries of the actual frequencies in Hz, we can create the filter bank of 26 bandpass filters as previously described in (4), similar to Fig. 4.

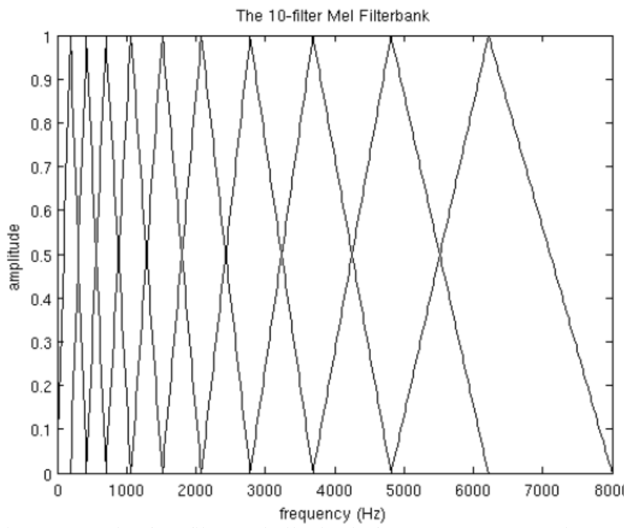

Fig. 4. Example of 10-filter Mel Filterbank, frequency range 0 to 8kHz [4]

We calculate the energy at the output of the filter, Y_m, by summing over all the frequency bins after applying the filter bank to the periodogram estimate of the power spectral density. To model after human hearing, which does not hear loudness on a linear scale, we take the logarithm of the energy outputs of the filter, resulting in X_m. We then perform a DCT of the twenty-six logarithm of X_m to compute the thirteen MFCC values for a single audio input signal. This calculation is described by (6) below.

$$MFCC(i) = \sum_{m=1}^{M} X_m \cos\left(i\left(m - \frac{1}{2}\right)\left(\frac{\pi}{M}\right)\right) \tag{6}$$

Deltas and Delta-Deltas Coefficients

Deltas and Delta-Deltas coefficients are also known as differential and acceleration coefficients. While the MFCC feature vector describes the power spectral of every frame, additional features such as the change in MFCC values over time is also helpful. The differential and acceleration coefficients of the MFCC feature vector is a method in which we extract additional characteristics of the MFCC feature vector and scratch signal [4]. To calculate the delta coefficients, (7) below is used, where c represents a MFCC element and d_t is the delta coefficient for element t of the MFCC feature vector. In order to calculation the delta-delta coefficients, (7) is further applied, but c represents a delta coefficient element and d_t is the delta-delta coefficient for element t of the delta coefficient vector.

$$d_t = \frac{\sum_{n=1}^{2} n(c_{t+n} - c_{t-n})}{2 * \sum_{n=1}^{2} n^2} \tag{7}$$

Weighted MFCC

Our independent research on speech recognition methods have considered the use of deltas and delta-deltas coefficients in addition to the MFCC feature vector [7]. The application of delta and delta-delta coefficients have been used in speech recognition research. Concatenating the delta and delta-delta coefficients feature vectors onto the MFCC feature vector, resulting a 39 element vector causes increased computation time. Thereby, we utilized the weighted MFCC (WMFCC) feature vector, where the elements are a linear combination of the respective MFCC, delta, and delta-delta coefficients as shown (8) below, and weight coefficients, ρ and ε.

$$WMFCC(i) = MFCC_i + \rho * \Delta MFCC_i + \varepsilon * \Delta\Delta MFCC_i \tag{8}$$

Following our research, we determined ρ and ε to be 1/3 and 1/6 respectively [7]. The resulting feature vector contains thirteen elements, instead of the 39 that would be too computationally heavy on the LCDK to compute in a reasonable period of time.

Dynamic Time Warping

Unlike the procedure to recognize phonemes in [6], the scratch signal may vary in speed and time. Thereby, instead of using the Euclidian distance as a measurement between the similarities of the template WMFCC feature vector and WMFCC feature vector of the new scratch signal, we utilize dynamic time warping (DTW). DTW allows us to calculate the minimal path between WMFCC elements of each frame as it accounts for changes in speed and time of the scratch [1].

First, we calculate the distance between all pairs of elements between the template and new WMFCC feature vector. This is calculated in (9), where $WMFCC_i$ and $WMFCC_j$ is the ith and jth element of the template and new WMFCC feature vector.

$$distance(i,j) = \left(WMFCC_i - WMFCC_j\right)^2 \tag{9}$$

We will utilize the resulting 2-by-2 matrix in the DTW matrix calculations. The DTW matrix allows us to determine the warping path that starts at indices (0,0) and finishes at (12,12) of the DTW matrix. The DTW matrix contains the accumulated minimum distances to reach each point from (0,0) to (12,12). The elements of the DTW matrix are calculated using (10), with the ith and jth element of the template and new WMFCC feature vector and the corresponding distance matrix [8].

$$DTW(i,j) = minimum \begin{cases} DTW[i-1, j-1] \\ DTW[i-1, j] \\ DTW[i, j-1] \end{cases} + distance[i,j] \tag{10}$$

In the special cases of the first row and column of the DTW matrix, is is the following (11) and (12).

$$DTW(i,0) = DTW[i-1,0] + distance[i,0] \tag{11}$$

$$DTW(0,j) = DTW[0,j-1] + distance[j,0] \tag{12}$$

An example of the DTW matrix that shows 16 of 169 elements of the complete 13-by-13 DTW matrix used in this project is shown in Table I, where TFV and NFV is the template feature vector and new feature vector, respectively.

TABLE I
4-by-4 DTW matrix

		New Feature Vector: NFV(j)			
i / j		0	1	2	3
Template Feature Vector: TFV(i)	0	A = |TFV(0) – NFV(0)|	E = A + |TFV(0) – NFV(1)|	F= E + |TFV(0) – NFV(2)|	G = F + |TFV(0) – NFV(3)|
	1	B = A + |TFV(1) – NFV(0)|	H = min(A,B,E) + |TFV(1) – NFV(1)|	K = min(F,E,H) + |TFV(1) – NFV(2)|	N = min(G,F,K) + |TFV(1) – NFV(3)|
	2	C = B + |TFV(2) – NFV(0)|	I = min(H,B,C) + |TFV(2) – NFV(1)|	L = min(K,H,I) + |TFV(2) – NFV(2)|	O = min(H,B,C) + |TFV(2) – NFV(3)|
	3	D = C + |TFV(3) – NFV(0)|	J = min(I,C,D) + |TFV(3) – NFV(1)|	M = min(I,C,D) + |TFV(3) – NFV(2)|	P = min(I,C,D) + |TFV(3) – NFV(3)|

Warping Path

Several rules are applied to ensure the reasonable calculation of that the warping path will be found: the path may only move forward in indices, either directly above, to the right of, or diagonally above [1]. A visual example of the warping path is shown in Fig. 5.

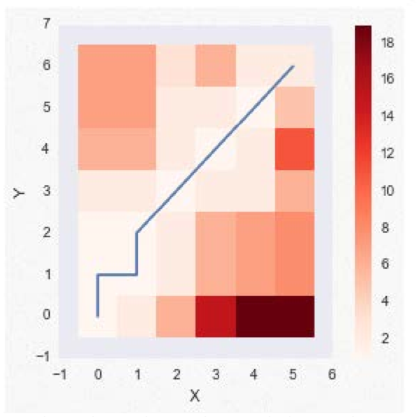

Fig. 5 DTW matrix and warping path. Y-axis is template vector of size 6 and X-axis contains new vector of size 5. Warping path is in blue [8].

The warping path allows us to determine the minimal distances and thereby similarity of the template and new feature vectors. The effective path cost of every frame is the summation of the values of the DTW matrix the warping path takes. The summation of the path costs for every frame is the scratch signals cost. Dynamic time warping is performed between every pair of template WMFCC feature vector of each character and new feature vector. The minimum resulting summed path cost of the test characters determines what the machine conceives the character of the scratch signal to be.

In addition, to utilizing the warping path to determine the identity of the scratch signal's character, it 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. However, due to the variance in time and speed, we cannot expect a one to one correlation between the elements. Thereby, we must backtrack the elements of the warping path to determine the best matched WMFCC feature vector elements to later add and average them into the template WMFCC feature vector.

E. *Software and Hardware*

This project had utilized the LCDK hardware and Code Composer Studio (CCS) 5 software. CSS also provided an adequate interface to debug the code. In addition, to CCS, MATLAB was utilized to verify calculations performed by the LCDK, analyze data, and compute FFT.

F. Procedures, Testing and Verification

Procedures

In the beginning of the program, human and computer interaction through the console of CCS5 is necessary, to indicate the number of characters being tested and the identity of the characters. This information is stored and processed in the area for the machine to communicate with the user easily.

After having known the test characters, the machine is available for memory of previous templates to be loaded in if necessary. The input signal is the recording of the scratching of each letter using a tip of a metal pen when the fifth switch of the LCDK is on.

The program runs the appropriate calculations to determine the best matching template and character that the new scratch signal represents and indicates it on the console. The user responds by indicating if the machine was correct or incorrect through the directions provided on the screen. If the machine was correct, the values of the WMFCC feature vector are averaged with the corresponding values in the template feature vector of the correct character. If the user indicates that the machine was incorrect, the machine will ask for the correct character, which the user will enter into the console, and the average WMFCC feature vector is recalculated.

As more input signals are sampled and processed, the WMFCC average vector for each character becomes a more accurate database; thereby, increasing the accuracy of our character recognition system. The testing process is completed by the program guessing what the letter is in each trial. If the program gets it right, it takes the user's input if it is correct or not. Switch 8 also keeps track of how many times we've had each letter correct out of how many time we input each letter.

Testing and Verification

In order to test the robustness of our system, both of us have tested the noise removal algorithm. Since the environmental noise may be different during every test, the noise removal is checked several times prior to further testing. In each theoretical segment of the project, the independent sections were tested and verified through the CCS5 variables display and MATLAB. We had verified the robustness of the noise removal in various levels of environmental noise. An example of the noise removal is shown in Fig. 6.

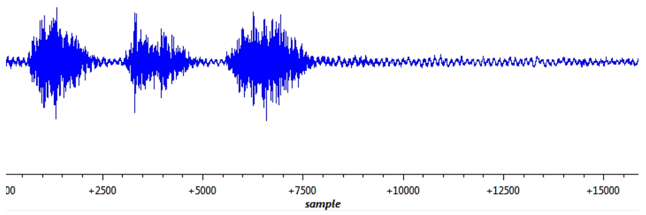

Fig. 6a. Original Input scratch signal for letter 'M' prior to start and end-point noise removal

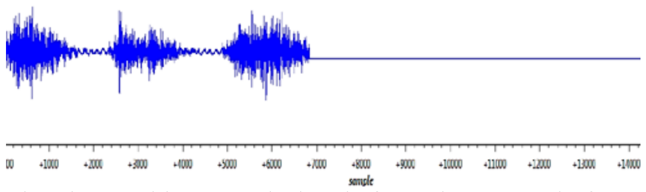

Fig. 6b. Resulting scratch signal after noise removal of above original input signal for letter 'M'

In addition, since different people write characters in differing stroke orders and speed, the machine learning system had to be tested and verified. Heba and Susan had both tested the system and saved the memory of the templates.

We had verified the calculations performed in our algorithm by exporting the distance, DTW, and WMFCC matrices and arrays in the MATLAB program. For example, the MFCC, delta, and delta-delta coefficients for a single frame of a scratch signal is shown in Fig. 7.

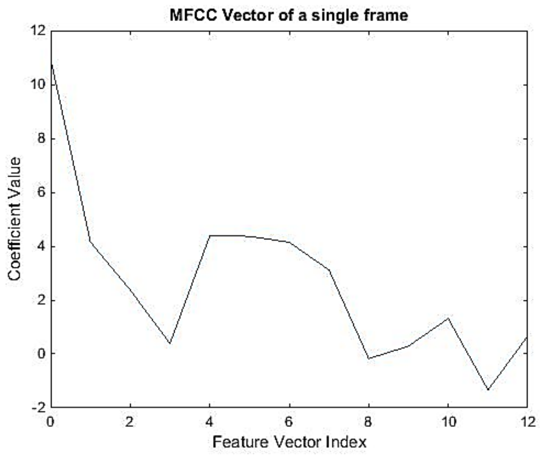

Fig. 7a. MFCC feature vector of a single frame in a trial for letter 'M'

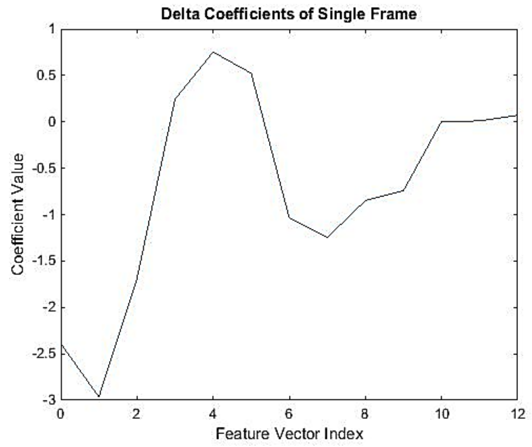

Fig. 7b. . Delta coefficients of the MFCC feature vector of a single frame in a trial for letter 'M'

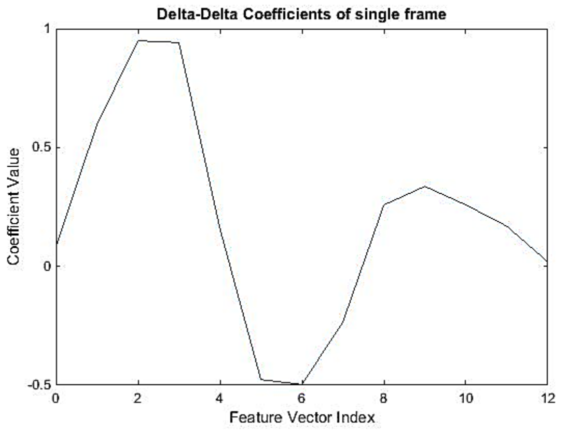

Fig. 7c. Delta-delta coefficients of the MFCC feature vector of a single frame in a trial for letter 'M'

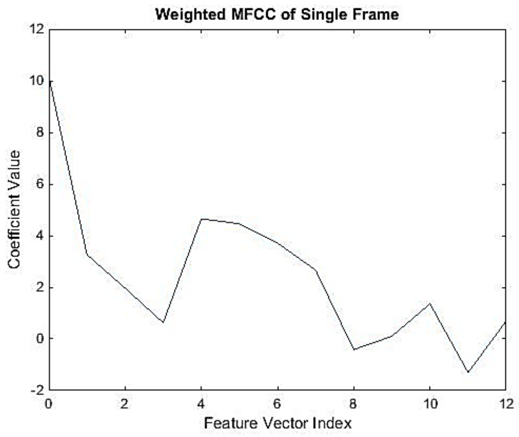

Fig. 7d. Weighted MFCC feature vector of a single frame in a trial for letter 'M', where the weights are 1/3 and 1/6 for delta and delta-delta coefficients, respectively.

IV. RESULTS

The noise removal system proved to be quite robust in removing the noise from the beginning and end of the input signal, resulting in the scratch signal alone, shown in Fig. 6.

The larger scheme of this project is to hopefully be able to differentiate and recognize the 26 letters in the English alphabet. We had attempted to do create a database of templates for all 26 letters, however, we noticed that as the number of characters are added to the database, the machine finds it much harder to differentiate among the characters in a reasonable number of trials and period of time. Thereby, we had chosen to six letters to differentiate among one another: A, B, C, M, T, and K.

Since the equipment we are using is only a standard microphone, we have accomplished our goal of differentiating at least one character. Through experiment, we determined that the characters are differentiable after an average of 5 to 15 trials for each character. Afterwards, we are able to achieve an accuracy ranging from approximately 52% to 100%, at an average of 86.8% accuracy, shown in Table II.

TABLE II
Experimental Results of test for letters: M, K, T, A, B, C

Letter	Trials untill learning complete		Accuracy
	Minimum	Maximum	
M	1	8	12/13= 92.3%
K	2	6	11/11= 100%
T	3	9	11/11 = 100%
A	4	9	10/11 = 90.9%
B	5	12	12/14 = 85.7%
C	12	40	14/27 = 51.9%

V. REFERENCES

[1] W. Li and T. Hammond, "Recognizing Text Through Sound Alone," in *AAAI Conference on Artificial Intelligence*, San Francisco, 2011.

[2] C. Harrison and S. Hudson, "Scratch input: creating large, inexpensive, unpowered and mobile finger input surfaces," in *ACM symposium on User interface software and technology (UIST)*, New York, 2008.

[3] J. E. Kim, J. Sunwoo, D. W. Lee and I. Y. Cho, "A gesteral input through finger writing on a textured pad," in *CHI Human factors in computer systems*, New York, 2007.

[4] J. Lyons, "Mel Frequency Cepstral Coefficient (MFCC) tutorial," Practical Cryptography:, 2013. [Online]. Available: http://practicalcryptography.com. [Accessed 7 March 2015].

[5] S. W. Smith, The Scientist and Engineer's Guide to Digital Signal Processing, Los Angeles: California Technical Pub, 1997.

[6] M. Briggs, *Mini-Project 2: Vowel Recognition,* EE113DA - Digital Signal Processing Design.

[7] S. V. Chapaneri, "Spoken Digits Recognition using Weighted MFCC and Improved Features for Dynamic Time Warping," *International Journal of Computer Applications,* vol. 40, no. 3, pp. 6-12, 2012.

[8] N. Batra, "Programatically understanding dynamic time warping," 31 July 2014. [Online]. Available: http://nipunbatra.github.io/2014/07/dtw/. [Accessed 1 March 2015].