Isolated Words Recognition using a Low Cost Microcontroller

Clayder Gonzalez-Cadenillas
Department of Computer Science
Universidad Nacional de Trujillo
Trujillo, Perú
Email: claydergc@gmail.com

Nils Murrugarra-Llerena
Department of Computer Science
University of Pittsburgh
Pittsburgh, PA, USA
Email: nineil@cs.pitt.edu

***Abstract*—Currently, the field of automatic speech recognition is being widely used in commercial electronic devices such as TVs, phones, game consoles and computers. Thus, this article presents an evaluation of different isolated words recognition techniques on an embedded system using in the microcontroller dsPIC30F4013. So, the feature extraction phase is based on an adaptation of the Mel-frequency Cepstral Coefficients ($MFCC$) and the automatic recognition phase is based on the following techniques: Dynamic Time Warping (DTW), Artificial Neural Networks (ANN) and Principal Component Analysis (PCA). Related to the experiments setup, voice commands were evaluated in 3 different scenarios and the best accuracy rate was reached by a combination of PCA and ANN. It is also important to note that this implementation was carried out with the capacity constraints of the mentioned circuit.**

***Index Terms*—MFCC, DTW, ANN, PCA, dsPIC30F4013, isolated words recognition**

I. Introduction

Nowadays, there are several electronic devices capable of interact with users through the human voice. One example are operating systems which have several applications for speech recognition. Another example are cell phones, in which this kind of technology has been being used by wide-known brands through isolated words recognition software, even before the apogee of the smart phones.

In contrast to the software described before, there also exists special purpose hardware which only focus on automatic speech recognition. They exist in electronic modules [1], specialized integrated circuits [2] and digital signal processors (DSP [3]). These kind of devices are very useful for their usage with microcontrollers and for handling of robots.

When we talk about products and projects that work with speech recognition tasks, it is easy to relate this area with algorithms of artificial intelligence. However, characteristics such as the hardware architecture where those algorithms are going to run, are sometimes neglected. In other words, we should ask ourselves, would a speech recognition algorithm *X* be capable to run correctly over an architecture *Y*, with *Z* MIPS (Million Instructions Per Second) without depleting its performance? or Do we need additional components for it to run? or simply, they would not work?.

Before introduce our proposal, we have to consider that we are going to implement all the algorithms on architectures with limited resources of memory and processing. These architectures can be seen generally on microcontrollers, which are computers based on the Harvard architecture and with a reduced instruction set, low speed of processing and few capacity of data memory.

Despite the lack of resources that microcontrollers have, they are **very cheap and affordable**. In contrast to devices like DSPs, microcontrollers are available in almost all over the world in different brands and models. In South America one of the most used and cheapest microcontrollers' brand are the Microchip PIC microcontrollers. They can be found in a broad range in 8-bit, 16-bit and 32-bit devices. It is obvious that 8-bit devices are cheaper and have less hardware resources than 16-bit and 32-bit devices. Because of that and considering that our main purpose is to use a low cost microcontroller, first of all we have to begin our research by evaluating the algorithms with respect to 8-bit devices' features and step by step go toward higher architectures.

Related to the algorithms for speech recognition, there are several researches that perform that task; but, most of them work with algorithms such as Linear Predictive Coding (LPC) [4] [5], Linear Prediction Cepstrum Coefficients (LPCC) [6] and Hidden Markov Models (HMM) [5]. In spite of that, it is widely known that the feature extraction using Mel Frequency Cepstrum Coefficients (MFCC) is the most accurate feature extraction algorithm [7] [8]. However, there are few researches about its implementation on microcontrollers [9] due to the computational complexity that imply its implementation.

Thus, the purpose of this research is to adapt some techniques such as Mel Frequency Cepstrum Coefficients (MFCC), Dinamic Time Warping (DTW), Artificial Neural Networks (ANN) and Principal Component Analysis (PCA) in order to find out which algorithms are the most suitable for working correctly with a low cost microcontroller. Throughout the experiments we will focus on some indicators like the recognition accuracy and the time of response. Finally we will conclude giving the minimum requirements a microcontroller must have in order to perform the basic tasks for recognition of isolated-words.

For reaching our previous purpose, several voice commands audios for training and test were recorded. So, these commands are the following ones: *ARRIBA (UP)*, *ABAJO (DOWN)*,

TABLE I
MAIN FEATURES OF 16F877A, 18F4550, DSPIC30F4013

	PIC16F877A	**PIC18F4550**	**dsPIC30F4013**
Performance	5 MIPS	12 MIPS	30 MIPS
Instructions	14-bit	16-bit	24-bit
Program Memory	14 KB	32 KB	48 KB
Data Memory	368 Bytes	2 KB	2 KB
Additional Features	SPI, UART, 10-bit ADC	SPI, UART, 10-bit ADC	SPI, UART, 12-bit ADC
Cost (US$)	4.94	4.47	6.00

DERECHA (RIGHT), *IZQUIERDA (LEFT)*, *PARA (STOP)*, *AVANZA (FORWARD)* and *RETROCEDE (BACKWARD)*, each one of these commands were recorded in different scenarios: *without noise and unique speaker*, *with noise and unique speaker* and *without noise and different speaker*. It has to be considered that not all the voice commands work correctly with all the algorithms, because of that, the number of voice commands recognized by one algorithm may change while working with other algorithm.

Our main result is that we reach an accuracy rate of 100% in the 3 scenarios using the PCA+ANN techniques (with the other algorithms the number of recognized commands may vary).

The current article is organized as follows. Section II presents the implementation of the embedded system. In section III, our experiments with the different algorithms are shown. Finally, our conclusions are presented.

II. IMPLEMENTATION OF THE EMBEDDED SYSTEM

Voice is a pressure wave, which is afterwards converted through integrated circuits into numerical values in order to be digitally processed [10]. The circuits and the processing divide the analysis in 2 blocks, the first one is hardware block and the second one is the software block.

A. Hardware Design

As was told in the introduction, we had to find the correct microcontroller for the algorithms to run correctly. In this research, three PIC microcontrollers were used: PIC16F877A, PIC18F4550 and dsPIC30F3013, which correspond to 8-bit Mid-Range Architecture, 8-bit PIC18 Architecture and 16-bit dsPIC30F Architecture, respectively. The main reason for using these microcontrollers was because they can be easily found in almost all countries in South America, there are hundreds of examples of their use on the Internet and their cost is very low. Table I shows the main features of each microcontroller that correspond to each architecture mentioned before, including their prices according to Microchip Direct website [11]. It is important to consider that there exist other 8-bit and 16-bit architectures; however, for our specific purposes only one device per each architecture mentioned below were used.

When this research began, our main purpose was to get an isolated-word recognition with the cheapest and most common microcontroller, and as a consequence of the low price, we also understood that we were going to work with a lack of resources. Besides, we also knew that for an accurate processing of voice commands we needed to reach an audio recording with at least 8 KHz of sampling rate, which in other words meant 8 KB per second. This minimum requirement was a serious problem, because all the 8-bit and the majority of 16-bit microcontrollers does not have more than 8 KB of data memory. In order to overcome this problem, an external RAM of 32 KB was used. The external RAM was the 23K256, which communicates with the microcontroller through the Serial Peripheral Interface (SPI) protocol.

So, in order to evaluate the performance of each microcontroller, one algorithm for feature extraction (MFCC) and one algorithm for automatic recognition (DTW) were chosen.

The first microcontroller we used was the Microchip PIC16F877A. With its 14 KB of program memory only a half of the feature extraction routine was possible to code, so we concluded that it is almost impossible to implement highly accurate speech recognition algorithms with the Microchip PIC16XX family because of their lack of program memory.

The second microcontroller used in our research was the Microchip PIC18F4550. With this microcontroller, speech recognition algorithms were possible to code; however, in our first experiments the time that this microcontroller took for recognizing a voice command was 1 minute, approximately. This issue made us change the current microcontroller with a faster one.

Finally, we made a last intent with the microcontroller dsPIC30F4013. Using this MCU (Micro Controller Unit) the processing time was reduced. Thus, we decided to choose the dsPIC30F4013 in order to do a deeper research that led us to evaluate the behavior of other techniques running under the architecture of the chosen microcontroller. Those experiments will be described in the following subsections.

Once the dsPIC30F4013 was chosen, we decided to measure the time that the microcontroller takes to write and read its internal RAM and the 23K256 external RAM in order to see how much they differ in time and how much this time difference affect the feature extraction and recognition.

So, we did a simple experiment in which we wrote and read one byte 1024 times, in other words, we wrote and read 1024 bytes. The results of this little experiment can be seen in the table II, where we noticed that the time that the microcontroller takes to write/read its internal RAM is 50 times faster than the time it takes to write/read the external SPI RAM. These results will be considered for our conclusions, later.

After we decided to work with dsPIC30F4013 and an

TABLE II
TIME FOR WRITING AND READING RAMS

	Internal RAM	External RAM
Write Time (seconds)	0.000341367	0.016981367
Read Time (seconds)	0.000342667	0.016982667

external SPI RAM, we started the implementation of the embedded system by doing the block diagram shown in Figure 1. This block diagram is based on the dsPIC30F4013 MCU, additionally it has an amplifier circuit for the microphone, an external RAM block (23K256), a user interface (compounded by 4 buttons and one LCD 2x16 display) and finally the recognized voice command as its output.

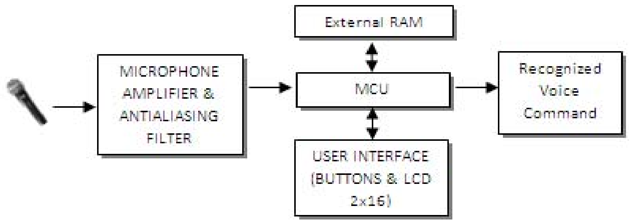
External RAM
MICROPHONE AMPLIFIER & ANTIALIASING FILTER
MCU
Recognized Voice Command
USER INTERFACE (BUTTONS & LCD 2x16)

Fig. 1. Isolated Words Recognition Embedded System's Block Diagram

As our purpose is to implement a low cost embedded system, the prices of the whole system can be seen in table III.

TABLE III
COST OF MATERIALS

Component	**Cost (US$)**
dsPIC30F4013	6.00
External SPI RAM	1.27
Microphone Amplifier	2.00
User Interface	2.20
Total	**11.47**

B. Software Design

The first step for all kind of speech recognition is the feature extraction. However, before focusing in the feature extraction, it is important to know that this process need approximately a voice signal with 100000 bits per second in order to keep all the transmitted information, which is greater than the average information of all fundamental phonemes [12]. So, the phonetic information transmitted with the correct frequency of bits will form the discrete signal, $x[n]$. However, the numeric values contained in $x[n]$ are not adequate for being processed by automatic speech recognition's algorithms [10]. Because of that, the voice signal must be previously preprocessed and later processed by some feature extraction's algorithm.

1) Preprocessing and Feature Extraction: The preprocessing of the discrete signal, $x[n]$, was done by using a filter and a windowing technique. In this research we used the pre-emphasis filter and the Hamming window. These preprocessing stages will let us to remove noise and compensate some frequencies.

Once the voice signal is preprocessed, now it is ready for being the input of a feature extraction algorithm. Among the most known we have: Fourier Transform Based Algorithms [13] [14], Linear Predictive Coding (LPC) [15] [4] [5], Mel Frequency Cepstral Coefficients (MFCC) [9], etc. In this research we used the Mel Frequency Cepstral Coefficients (MFCC), which is nowadays the most accurate and widely used algorithm for voice signals' feature extraction [7] [8].

The Mel Frequency Cepstral Coefficients are a defined representation such as the cepstrum of a windowed channel in the time, coming from a Fast Fourier Transform (FFT), however, they are in a non-lineal frequency scale. All of that produces an approximation widely similar to the human been auditive system [16]. The MFCC algorithm is divided mainly in 3 stages: Fast Fourier Transform, Logarithm of the energy and the Discrete Cosine Transform. The features used for automatic speech recognition are defined by 13, 26, or 39 coefficients per frame. The higher quantity of coefficients is, the better recognition precision that can be obtained [17]; however, we based our implementation only in the 13 first features per frame that the MFCC algorithm returns, because using 26 or 39 coefficients would slow significantly the time of response (given that all those coefficients would have to use the external RAM).

So, we started to implement them by adapting the preprocessing and MFCC algorithms to the dsPIC30F4013 architecture in spite of the lack of resources described before. Along with the Microchip C30 compiler, the library RxTx [18] was used to see the behavior of the different Preprocessing and MFCC's stages. Each stage of the Preprocessing and MFCC algorithms implemented in the microcontroller was compared with its computer-based version because some changes were done in the type of data used in the microcontroller in order to fit in the digital signal processing optimized functions requirements. So, this comparison was useful for us to see how much the functions vary throughout the different stages.

Figure 2 shows the comparison between the computer and the microcontroller implementation in each stage of the Preprocessing and MFCC's feature extraction. In this figure, we can see that the graphics corresponding to the pre-emphasis stages are almost equal, however from the Fast Fourier Transform to the Discrete Cosine Transform stages the differences start to increase. The reason for this is because the adapted algorithm implemented in the microcontroller consists in scaling the output of the hamming filter before entering the Discrete Fourier Transform function, so some bits of the phonetic information are lost. However, despite the lost of bits, the experiments will demonstrate that this issue do not affect the automatic recognition.

2) Automatic Recognition: Once the feature extraction is finished, the next step is the automatic recognition task. As was done in the feature extraction, the process of recognizing a spoken word by a person has been also a motivation for several researches. So, among most known algorithms for isolated words recognition, we have: the Dynamic Time Warping (DTW) [19] [20], Artificial Neural Networks [15] and Hidden

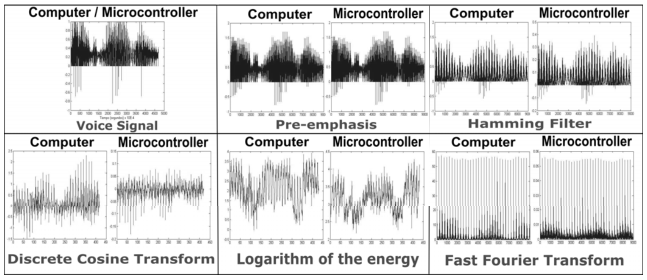

Fig. 2. Comparison of Mel Frequency Cepstrum Coefficients' Stages

Markov Models (HMM) [5]. Thus, in this research the first two techniques and a third one (Principal Component Analysis) have been implemented and compared.

Dynamic Time Warping is an algorithm proposed by Sakoe and Chiba [21] and it is originally based in the dynamic programming and the calculus of distances, which purpose is to do a time normalization with respect to the X axis. This feature help us to do a matching on similar voice commands, in spite of the different audios' lengths.

As the dynamic time warping requires the output from the feature extraction (MFCC), and given that the number of these features are greater than the internal RAM capacity, the majority of calculus were performed by reading and writing the external SPI RAM. This issue affected notably the time of processing (see table II), which will be shown in the following section.

For comparing the first technique, we implemented a second technique for the automatic recognition: Artificial Neural Networks (ANN); more exactly, we used a multilayer perceptron (MLP). A MLP neural network is a supervised machine learning technique that needs to be trained with previous data. Most of the time this training step take some time for processing, however, it is only done once. So, for the training task we used a computer program along with Matlab's *Pattern Network* [22] in order to copy the synaptic weights' matrix (θ) into the microcontroller. Matlab's Pattern Network uses the *Scaled Conjugate Gradient Backpropagation* to train the network and the *Hyperbolic Tangent Sigmoid Function* (g) as the transfer function. With the neural network trained, we only needed to implement the forward propagation algorithm ($h_\theta(x)$) in the microcontroller and set the input neurons x following the equation (1).

$$h_\theta(x) = g(\theta^T x) \tag{1}$$

The input for the ANN was supposed to be compounded by all the features returned by the MFCC function. We configured the MFCC algorithm to return at most 624 features, which correspond to 0.8 seconds of speech. As the number of features exceeded the microcontroller's capacity, we used the **arithmetic mean** in the last stage of the MFCC algorithm, so we compressed the data of each 13 features in order to obtain only 49 features per voice command. As a consequence, we had to configure the network with 49 input neurons, 5 hidden layer neurons and 5 output neurons (5 voice commands).

This second technique brought us better time results than the DTW, those results will be shown in the following section, however, we did not reach the DTW's accuracy. Consequently, we tried a third technique: The Principal Component Analysis (PCA), which is normally used to perform dimensional reductions. So, we used this algorithm along with the artificial neural networks in order to reduce the inputs from 624 to 2 dimensions. As a consequence, we reconfigured the network in order to work with 2 neurons in the input layer, 5 neurons in the hidden layer and 5 neurons in the output layer.

Before implementing the PCA algorithm, we used PEx [23] in order to project those features in 2 dimensions and to see how separable were them. So, in figure 3 we have the projections corresponding to 5 voice commands and 7 voice commands where each cluster of commands is labeled with a different shape and filling.

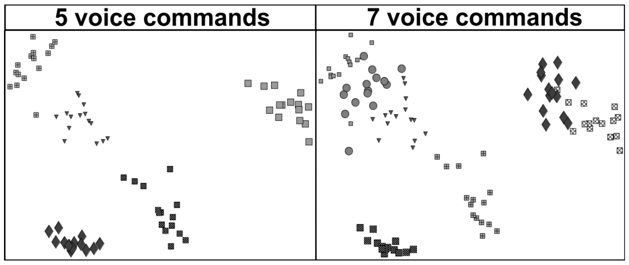

Fig. 3. Comparison between 2-Dimensional Projection of 5 and 7 Voice Commands

As seen in figure 3, the 5 voice commands 2D-projection shows the clusters very separated, however, when the commands are incremented to 7 the projection starts to look very crowded. So, as we wanted to work with 7 commands, we decided to work with 4 dimensions instead of 2. Those voice commands are: *ARRIBA*, *ABAJO*, *DERECHA*, *IZQUIERDA*, *PARA*, *AVANZA* and *RETROCEDE*. Finally, this addition of dimensions led us to use an ANN with 4 neurons in the input layer, 5 neurons in the hidden layer and 7 neurons in the output layer.

All the techniques we implemented before, brought us different accuracy results; nevertheless, we think that accuracy results must be always related with time results in order to reach a conclusion. Those results will be discussed in detail in the following section.

III. Experiments

To implement all the algorithms mentioned before in the microcontroller, the programming language C and the Microchip C30 compiler were used. These tools allowed us to optimize some digital signal processing functions such as the Fast Fourier Transform (FFT), which has its code specially implemented by Microchip in order to take advantage of the dsPIC30F architecture as much as possible. This feature increment the performance of the algorithms and it can be seen in similar researches [9].

Related to the experiments, we recorded one set of 161 audios divided among 7 voice commands. The voice commands recorded were: *ARRIBA*, *ABAJO*, *DERECHA*, *IZQUIERDA*, *PARA*, *AVANZA* and *RETROCEDE*, which are enumerated from 1 to 7 respectively in the following tables (names of commands were thought for using them in the control of a mobile device). The results obtained in the experiments were classified with the following criteria: *without noise and unique speaker* ($E1$), *with noise and unique speaker* ($E2$) and *without noise and different speaker* ($E3$). In relation to the scenario with noise, its Signal Noise Ratio (SNR) [24] was equal to 0.77 compared to signal obtained in the scenarios without noise.

A. Training Audios

The training set was formed of 105 audios (15 audios per voice command). In the case of the DTW algorithm, there is not an explicit training, so its 'training' consist on recording each command's features (coming from the MFCC) in the microcontroller's FLASH memory. Due to the large amount of data corresponding to features and considering that the FLASH memory was almost full, only 1 set of features for the first 4 voice commands (*ARRIBA*, *ABAJO*, *DERECHA*, *IZQUIERDA*) could be used for the DTW 'training'. For the ANN-based algorithms (ANN and ANN+PCA), 15 audios per voice command were transfered to the computer through the RS-232 protocol in order to train the artificial neural network with Matlab's help.

B. Test Audios

The test set consists on 56 audios (8 audios per each voice command). These audios were tested in real time using the embedded system in order to evaluate their results with the DTW, ANN and ANN+PCA algorithms. It must be considered that for the DTW algorithm only 32 audios were used (8 audios for the first 4 commands), as was done in the training set.

C. Experimental Results

Our first experiments were done in order to measure the accuracy of the algorithms. So, we started to evaluate the recognition rate of all the mentioned algorithms, using the test set. Table IV shows the recognition rate of the voice commands in the different scenarios and using the different algorithms. As told before, when using the DTW algorithm we could not record all the command's features because of the microcontrollers' lack of FLASH memory, so we evaluated only the first 4 voice commands. For the neural networks we did not obtain very good results using 7 commands, that make us think that data compressing using the arithmetic mean is not so useful. Finally for the PCA+ANN technique we obtained the best accuracy's results in the 3 scenarios; however, it is important to note that these results depends on the quantity of dimensions that we get from the PCA. So, in the implementation section we also got good results using PCA to reduce the dimensions to 2 but that configuration could only recognize accurately up to 5 voice commands due to the extreme dimensional reduction. Consequently, when we used PCA to reduce the dimensions to 4 we recognized up to 7 voice commands.

TABLE IV
RECOGNITION RATE OF THE VOICE COMMANDS DIVIDED BY ALGORITHMS AND SCENARIOS

	DTW(%)			ANN(%)			PCA+ANN(%)		
	E1	E2	E3	E1	E2	E3	E1	E2	E3
1	100	100	100	87.5	87.5	62.5	100	100	100
2	100	100	100	75	75	62.5	100	100	100
3	100	100	100	75	75	62.5	100	100	100
4	100	100	100	87.5	87.5	62.5	100	100	100
5	-	-	-	100	100	100	100	100	100
6	-	-	-	75	62.5	62.5	100	100	100
7	-	-	-	75	62.5	62.5	100	100	100
T	57.1	57.1	57.1	82.1	78.6	67.9	100	100	100

Next to the accuracy experiments, we proceed to measure the time of the algorithms MFCC, DTW, PCA+ANN. So, we divided MFCC's times in two groups, the first one corresponding to ANN with 49 neurons in its input layer and the second one used for the DTW and ANN-PCA. We did this division because the ANN writes only 49 features in the external SPI RAM and the second one write 624 features (all features) in the external SPI RAM.

Table V shows the processing time of the MFCC algorithm when its configured with DTW, ANN and PCA. So, there is a little difference in the time when we use the MFCC for ANN and when we use it for DTW and PCA, the reason for this is that the last ones access the external SPI RAM more times than the first one.

TABLE V
PROCESSING TIME (SECONDS) OF THE MFCC ALGORITHM

	1	2	3	4	5	6	7
DTW	5.41	5.41	4.92	6.99	-	-	-
ANN	5.18	5.32	4.69	6.67	1.90	5.27	6.10
PCA	5.41	5.41	4.92	6.99	1.94	5.53	6.47

Table VI shows the processing time of the 3 algorithms used for the automatic recognition. Here we can see that DTW's computational complexity along with the 624 features cause a big delay in the response. On the other hand, ANN and PCA+ANN algorithms are constant for all the voice commands, this is because in the case of the ANN it is only a sum of products of 49 features, and in the case of PCA+ANN it is the same technique but using the multiplication of PCA's covariance matrix with the 624 features as the ANN input.

TABLE VI
PROCESSING TIME (SECONDS) OF RECOGNITION ALGORITHMS

	1	2	3	4	5	6	7
DTW	29.64	29.50	30.97	39.96	-	-	-
ANN	0.04	0.04	0.04	0.04	0.04	0.04	0.04
PCA	0.06	0.06	0.06	0.06	0.06	0.06	0.06

It must be considered that the times shown in table VI must be added to its corresponding MFCC's time from table V in order to get the total recognition response time.

To finish our experimental results, we did a comparison with related works mentioned before considering: the technique used, the number of commands, the response time and the accuracy (recognition rate). Table VII shows that comparison done among the microcontrollers AT89C51RC [4], ATMega162 [5], ATMega128 [25], M16C [14] and our proposal dsPIC30F4013. Here we can see that our results are better than the first 3 ones (considering that Nitin Kandpal's article [25] did not report any response time) and very similar to M16C's proposal, with differences on response time and accuracy. Related to the M16C's response time, our proposal is slower; however, it must be considered that the microcontroller M16C has 20 KB of on-chip RAM, which is a disadvantage for us (as we saw earlier in the table II).

TABLE VII
COMPARISON WITH SIMILAR RESEARCHES

	[4]	[5]	[25]	[14]	30F4013
Technique	LPC/ED	LPC/HMM	ED	ANN	ANN/PCA
Commands	7	5	4	5	7
Response	19 s	17 s	-	347 ms	5 s
Accuracy	78.57%	87%	75%	92%	100%

IV. CONCLUSIONS

In order to reach our main purpose, we implemented a **very cheap** embedded system based on the microcontroller dsPIC30F4013 in order to evaluate 4 different algorithms. The first algorithm, used for feature extraction, was the Mel Frequency Cepstrum Coefficients (MFCC), the other 3 algorithms, used for the automatic recognition, were the Dynamic Time Warping (DTW), Articial Neural Networks (ANN) and Principal Component Analysis (PCA).

Also, we evaluated 3 different microcontrollers which gave us an idea of what kind of devices are the most suitable for dealing with speech recognition tasks. So, we can conclude that the 8-bit architecture is not appropriate for performing this kind of task. Moreover, microcontrollers under 30 MIPS, like PIC18F4550, may be very slow for processing large amounts of data. Thus, when we did experiments with the dsPIC30F4013, which is a 16-bit microcontroller, we could get a 7 commands recognition but with a time response of 5 seconds, approximately. This time of response could be improved if we use a microcontroller with the same characteristics but with more internal RAM memory or data memory.

Focusing on the results, we obtained an accuracy rate of 100% for the PCA+ANN in 3 different scenarios. In relation to the time of response, the fastest algorithm was the ANN based recognition, however, its accuracy rate was not the best one. Nevertheless, when ANN was used along with PCA, it was only a 0.02 seconds slower than the ANN.

Given that our results were better than related works [4] [5] [25], we confirmed that MFCC algorithm used along with ANN and PCA algorithms are very accurate for isolated words recognition.

Finally, considering the experiments done with PEx, we concluded that the more dimensions the PCA reduction has, the more broad range of recognized voice commands we can get. This fact will be useful for a future work in which we are planning to use a microcontroller from the dsPIC33 family.

REFERENCES

[1] (2012, Octubre) Easyvr. [Online]. Available: http://www.veear.eu/products/easyvr/

[2] (2012, Diciembre) Hm2007 speech recognition. [Online]. Available: http://www.datasheetcatalog.org/datasheets/2300/499674_DS.pdf

[3] M. B. B. Venkataramani, *Digital Signal Processors*, T. M.-H. Education, Ed., 2002.

[4] D. W. Thiang, "Implementation of speech recognition on MCS51 microcontroller for controlling wheelchair," in *International Conference on Intelligent and Advanced Systems*, 2007.

[5] D. W. Thiang, "Limited speech recognition for controlling movement of mobile robot implemented on atmega162 microcontroller," in *International Conference on Computer and Automation Engineering*, 2009.

[6] L. R. Shi Yuanyuan, Liu Jia, "Single-chip speech recognition system based on 8051microcontroller core," *IEEE Transactions on Consumer Electronics*, vol. 47, pp. 149–153, 2001.

[7] P. Davis S., Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 28, pp. 357 – 366, 1980.

[8] V. C. Sajjan S.C., "Comparison of dtw and hmm for isolated word recognition," in *International Conference on Pattern Recognition, Informatics and Medical Engineering (PRIME)*, 2012.

[9] U. A. Lizondo M, Agüero P, "Embedded speaker verification in low cost microcontroller," in *Congreso Argentino de Sistemas Embebidos*, 2012.

[10] P. L. Becchetti Claudio, *Speech Recognition*. John Wiley & Sons Ltd., 2002.

[11] Microchip. (2013, August) Microchip direct. [Online]. Available: http://www.microchipdirect.com/

[12] S. R. W. L. R. Rabiner, *Digital Processing of Speech Signals*. Prentice-Hall, Inc, 1978.

[13] S. G. Sarikaya R., Yuqing Gao, "Fractional fourier transform features for speech recognition," in *IEEE International Conference on Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP '04).*, 2004.

[14] B. M. d. B. Carlos Bernal Ruiz, Francisco Garcia Tapias, "Microcontroller implementation of a voice command recognition system for human-machine interface in embedded systems," in *10th IEEE Conference on Emerging Technologies and Factory Automation, 2005. ETFA 2005.*, 2005.

[15] K. M. Paul A.K., Das D., "Bangla speech recognition system using lpc and ann," in *Seventh International Conference on Advances in Pattern Recognition, 2009. ICAPR '09.*, 2009.

[16] J. L. G. Díaz, "Recuperación de información en textos hablados," Master's thesis, Escuela de Postgrado de la Universidad Nacional de Trujillo, 2010.

[17] F. Carranza-Athó, "Implementación de un reconocedor de palabras aisladas utilizando mfcc y dtw," Noviembre 2008.

[18] (2012, Agosto) Rxtx. [Online]. Available: http://rxtx.qbang.org/

[19] B. Q. Jing Zhang, "Dtw speech recognition algorithm of optimization template matching," in *World Automation Congress (WAC)*, 2012.

[20] L. L. Chun Wan, "Research and improvement on embedded system application of dtw-based speech recognition," in *2nd International Conference on Anti-counterfeiting, Security and Identification, 2008. ASID 2008.*, 2008.

[21] C. S. Sakoe H., "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing*, pp. 43–49, 1978.

[22] MatWorks. (2013, August) Pattern recognition network. [Online]. Available: http://www.mathworks.com/help/nnet/ref/patternnet.html

[23] F. V. Paulovich, M. C. F. Oliveira, and R. Minghim, "The projection explorer: A flexible tool for projection-based multidimensional visualization," in *Proceedings of the XX Brazilian Symposium on Computer Graphics and Image Processing - SIBGRAPI.* Belo Horizonte, Brazil: IEEE CS Press, 2007, pp. 27–36.

[24] F. Stumpers, "Theory of frequency-modulation noise," *Proceedings of the IRE*, vol. 36, pp. 1081 – 1092, 1948.

[25] A. P. Nitin Kandpal, Yashodhan Mandke, "Implementation of voice recognition in low power microcontroller," in *2012 IACSIT Hong Kong Conferences*, 2012.