Microcontroller Implementation of a Voice Command Recognition System for Human-Machine Interface in Embedded Systems

Carlos Bernal-Ruiz, Francisco E. García-Tapias, Bonifacio Martín-del-Brío and Antonio Bono-Nuez
University of Zaragoza
Dept. of Electronics Engineering and Comm.
50018 Zaragoza SPAIN
cbernal@unizar.es bmb@unizar.es
antoniob@unizar.es

Nicolás J. Medrano-Marqués
University of Zaragoza
Facultad de Ciencias
50009 Zaragoza SPAIN
nmedrano@unizar.es

Abstract

Recent developments in signal processing, pattern recognition, neural networks and powerful (low-cost) embedded processors allow including speech processing for human-machine interaction in consumer equipment.

In this paper we present a compact system for vowels and small vocabulary recognition, its orientation being consumer applications, where cost is of paramount importance. In this case, the speech recognition method must be as simple as possible, in such a way that could be programmed onto a standard microcontroller device, with the typical requirements of low range embedded application, where memory (kilobytes of RAM) and computer power (some MHz) are very limited.

In this work we show both, algorithms and their implementation onto a low-cost microcontroller, and we study the final system performance.

1. Introduction

Nowadays computer based speech processing systems have reached complex structure and high accuracy [1, 2], due to sophisticated signal processing algorithms and powerful computers available. Nevertheless, the performance increase of embedded processors (also called microcontrollers) now allows the implementation of speech recognition in consumer products, where cost is of paramount importance [3].

The use of a low-cost control oriented embedded processor (microcontroller) for this task limits system complexity, making the selection of variables and methods a particularly challenging assignment.

In this paper, we present preliminary results of such a highly simplified system, which can be applied to non-critical menu control using very compact vocabulary and to speaker verification. Our work shows that low end microcontrollers can be programmed for this application by using standard tools to achieve reasonable accuracy, when certain considerations are taking and limited goals are considered.

In recent years several efforts have been carried out in this direction [4, 5], even proposing custom ASIC or dedicated chips [3, 6]. In this case vocabulary recognition is limited to a fixed inventory in the so called "text dependent" approach [3]. For speech or speaker recognition, power, pitch, and spectral envelopes are very important to represent the characteristics of speakers (with instantaneous and transitional features of speech).

Those features have been traditionally extracted in multiple time windows following the phonology phoneme breaking approach. In this work we limit the study to one time-framed buffer because of the required simplicity, although limiting application to a well stabilized vocabulary composed by simple phonemes. In this implementation we are going to consider the speaker and text dependent recognition scheme.

Fig. 1 shows the block diagram of a typical Automatic Speech Recognizer (ASR). Speech is first captured by an electrec microphone [9], being the signal band limited and amplified by an analog pre-processor, converted to digital and processed in the front-end analysis stage; details will be explained in Sect. 2.

Next, in Sect. 3 we present the speech classification stage, based on artificial neural networks; notice that two different operating modes are defined (Fig. 1): training, in which speech features are extracted and memorized as neuron weights, and recognition, involving real-time speech pattern classification.

In Sect. 4 we show the experimental setup and recognition accuracy achieved. Finally, some conclusions are provided in Sect. 5.

0-7803-9402-X/05/$20.00 © 2005 IEEE

2. Embedded speech recognition system

We have developed an ASR system (Fig. 2) keeping in mind that must be implemented in a standard microcontroller.

We have selected a Mitsubishi M16C, which integrates a 16-bit core, 20KB of on-chip RAM, FLASH, ADC/DAC, I/O ports and other typical peripherals of control applications. This can be considered a mid-range microcontroller, used in many consumer electronic products that cost about 5 euros when purchased in high quantities. Nevertheless, due to the simplicity of the techniques employed, the recognition system could be ported to other 8 or 16 bit microcontroller families, as Motorola/Freescale 68HC08/12, Microchip 16/18XX, Atmel AVR, etc.

By using the on-chip 10-bit analog to digital converter (ADC), a 114 ms speech frame is acquired at 4.5K samples per second providing a total of 512 samples per frame. Previously, an analog stage has amplified the microphone signal to the ADC range (0-5 volts), and limited to the band 20Hz-2KHz, where speech spectrogram shows significant components (Fig. 3). For our experimental setup, we have used as vocabulary the five Spanish vowels. The purpose of the digital front-end stage is to parameterize the speech signal. The reason is two fold: transform the signal into a more compact form, and extract representative acoustic features for the recognition stage.

Traditionally Mel Frequency Cepstral Coefficient (MFCC) or other complex feature extraction methods have been used [1, 2, 4], providing a good performance for word or digit oriented command recognition and 'word spotting' applications. In this paper we present a simpler processing scheme suitable to be implemented in low cost applications.

First, an automatic threshold detector (ATD) decides when a new acquisition starts, and then several processing stages are applied (Fig. 2); windowing, FFT and filter bank implement the spectrum estimation method including network pre-processing. Finally selected features are presented to the classifier.

In the next paragraphs we explain each part of this system.

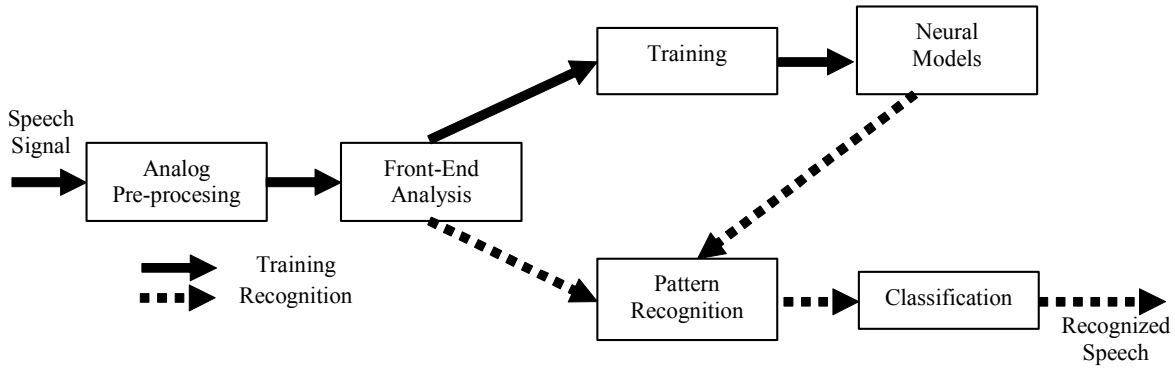


Figure 1. Stages involved in an automatic speech recognition system (ASR)

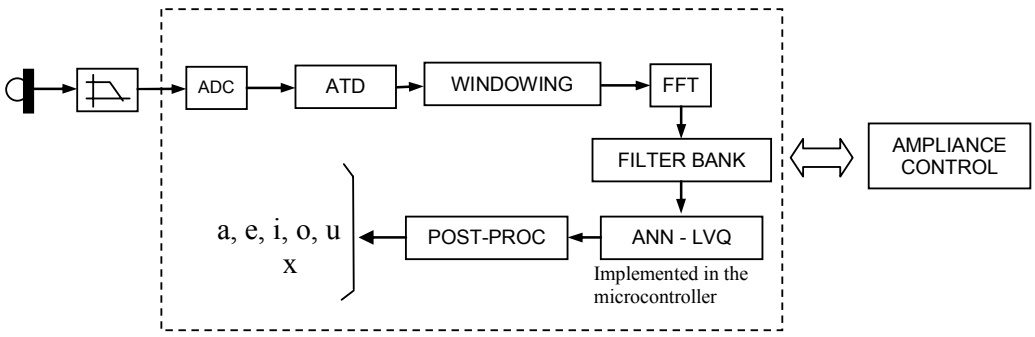


Figure 2. Block diagram of the automatic speech recognition system

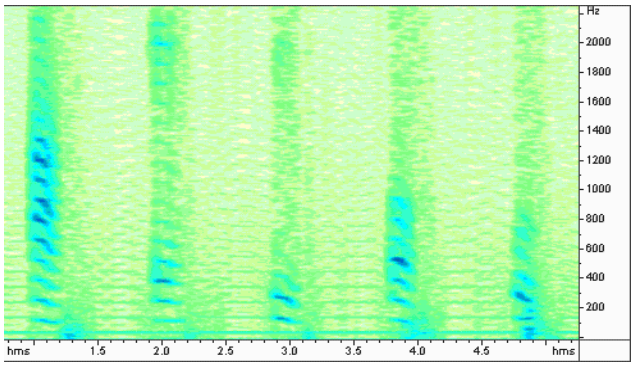


Figure 3. Spectrogram showing components of our application vocabulary

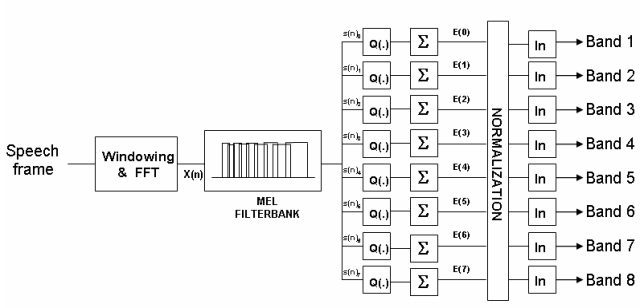


Figure 4. Spectral estimation diagram

2.1. Automatic threshold detection (ATD)

A subroutine (periodically called from the main program) samples audio data at very low frequency (330Hz), calculating increment from sample to sample by using both level and derivate threshold detection. A threshold trip event will trigger a 10 unit down counter clocked by the sample frequency. If the counter reaches zero the processor will begin a new acquisition and recognition cycle; in other case, the microcontroller will continue executing main control task.

2.2. Windowing

After acquisition is finished, the first procedure is to apply a window to the 114ms data buffer; this selected buffer length results from the maximum experimental value obtained with this vocabulary.

Windowing provides enough independence of sharp discontinuities in the window edges that would bring in artefacts at the window limits, thus permitting further dependence reduction on speaker inflection. Standard table-stored Hamming window has been selected for this purpose [8]; table implementation preserves the microcontroller of a high time-consuming task.

	WER (%)	Time (ms)
Mel-warping rectangular filter	8%	20
Mel-warping triangular filter	12%	35
Linear-warping rectangular filter	13%	22

Table 1. Bank filter influence in word error rate (WER) for the vocabulary selected, and processing time required for the microcontroller

2.3. Spectral estimator

Studying speech features usually involves one kind of spectrum estimation; in this case, subband processing thought FFT-implemented filter bank has been selected (Fig. 4).

As can be found in multiple references [1, 3, 7, 11], it provides a high potential estimation technique, involving standard well-known signal processing algorithms. Processing round-off error can be minimized by using double precision integer data type, and enough ADC input voltage must be provided to use most of the input range available in the microcontroller 10-bit ADC.

According to Fig. 4, Fourier transform is performed in a windowed buffer, followed by band filtering and power averaging. This provides as a result specific 8 band power of the speech frame. The following stage is normalization and natural logarithm compression.

Other common methods, as LPC analysis, provide good results with less memory usage [8], but due to the more processing power required for on-line processing (see, for instance, ref. [4]) these methods have been discarded. Selection of a power of two fixed buffer length avoid the zero padding FFT processing scheme increasing simplicity.

Several parameter selection considerations are now analyzed. In spectral estimation stage, the Discrete Fourier Transform (DFT) by means of N point Fast Fourier Transform (FFT) algorithm is calculated obtaining the speech spectrum for the N_S samples windowed buffer according to

$$S(f) = \sum_{n=0}^{N_S-1} s(n) e^{-j2\pi \frac{f}{f_S} n} \qquad (1)$$

Thus, the *S(f)* symmetric frequency output vector is obtained.

The spectrum estimation method is completed with power spectral information smoothing and compression by using a quarter overlapped rectangular Mel-warped filter bank [2] (Fig. 4).

Table 1 summarizes filter influence, showing different recognition results depending on the filter bank

and warping transformation employed. While presenting low differences, the filter bank method proposed is the lowest processing power consuming task due to the rectangular implementation requites only a few accumulate instructions.

In discrete form results on (being k the vector position, l the index of band filtering, and with L bands):

$$k = \frac{2f}{f_s} \;\; ; \;\; S(f) \Rightarrow S(k) \;\; ; \tag{2}$$

$$\hat{S}(l,n) = \sum_{k=0}^{N_S/2} S(k) M_l(k), \; l = 0,1,...L-1$$

Where $M_l(k)$ states for the k-Mel value (1-0) of the l-th filter.

In order to make speech recognition more acoustic level independent, power averaging (Equation 3) includes local application of quadratic operator, and computing natural logarithm of power normalized in each band:

$$x(l+1) = \ln\left(\frac{(E(l))}{MAX[E(1)...E(L)]} \right) \tag{3}$$

3. Learning Vector Quantization for pattern classification

After processing the speech signal, a pattern classification stage is carried out by means of a Learning Vector Quantization (LVQ) neural network (Fig. 5) [10]. We have selected LVQ because of its computational simplicity and good performance in pattern classification applications [10].

Our architecture (Fig. 5) consists of eight input neurons (eight speech bands) and five outputs (one for every Spanish vowel), requiring 40 synaptic weights.

The training phase involves first selecting representative samples $\mathbf{x}$ to training the neuron codebooks $\mathbf{w}_i$ (or vector weights). According to usual guidelines [10], we have used 100 training pattern vectors. Then, standard LVQ1 training algorithm [10] is carried out on a computer; and the 40 weights obtained will be stored as a table in the microcontroller flash memory.

Notice that previous to the LVQ, logarithm in Equation 3 introduces a non-linearity, which has revealed better recognition accuracy in our problem. At the recognition phase (recall) the Manhattan distance is used (Equation 4), because is simpler than standard Euclidian distance without degrading the results [10]. Then, the minimum neuron distance is detected, being recognized as the 'winner' neuron (c index). This neuron will be associated to the recognized pattern:

$$\left|\mathbf{x} - \mathbf{w}_C\right| = \min_i \{\left|\mathbf{x} - \mathbf{w}_i\right|\} \tag{4}$$

We have experimentally estimated a 'recognition' threshold (distance 250 in our case). When all distances computed for all neurons exceed that threshold, the pattern is considered not recognized.

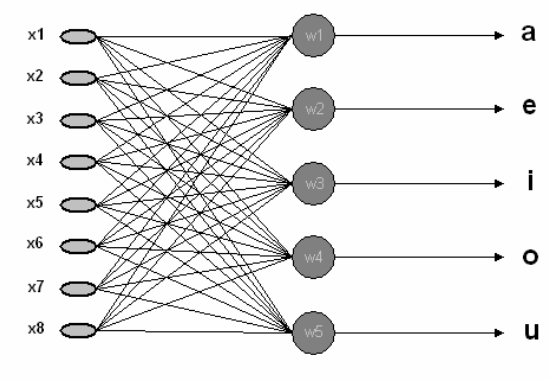


Figure 5. LVQ neural network

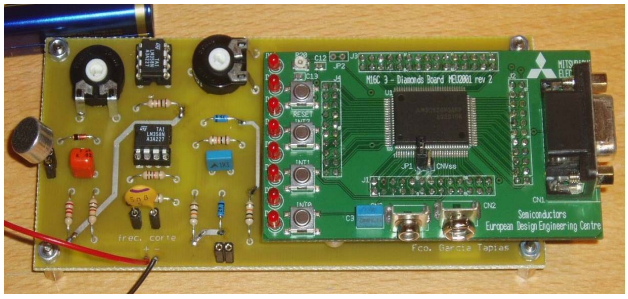


Figure 6. Embedded speech recognition prototype

WER vocabulary patterns	8% (92% pattern successfully recognized)
RAM/ROM usage	4.4Kb / 10Kb
Total processing time	347ms at 16MHz core clock
ANN (LVQ) recognition time	22ms (7%)
Spectral estimation proc. time	210ms (60%)
Other processes	115ms (33%)

Table 2. Prototype performance

4. Performance of the voice recognition system

Fig. 6 shows the implemented prototype, including analog pre-processing circuits (left) and the microcontroller device (right). All speech recognition stages described have been implemented in the microcontroller; a computer has only been used in training the LVQ.

Table 2 summarizes main characteristics and results derived by means of 200 recognition runs in an office quiet environment (SNR>10dB). Notice that a 92% of successful commands recognized has been achieved (word error rate, WER, in Table 2). A total processing time of 347 ms per speech command has been achieved; remark the little time required for the LVQ operation (22 ms, 7% of the total time). Other test conducted in a noisy environment (music at conversation level) shows similar results, while exact accuracy depends on specific vowel tested.

5. Conclusions

This paper shows that the proposed structure allows speech recognition features to be implemented in common embedded applications based on standard microcontrollers, by using standard tools and without a significant change in hardware (without additional cost). This can be useful in applications such as managing simple menus in consumer products.

While our prototype achieves a good performance in command recognition for a specific speaker, its performance degrades for different speakers. We think that this can be solved by modifying the LVQ architecture by including several neurons for every speech command, and training the system with speech from different speakers. We are now working in this direction.

Nevertheless, these speaker dependent recognition features [11, 12] present new possibilities in speaker verification and security features for embedded applications. Exhaustive exploration of all these characteristics is subject of future work.

6. Acknowledgements

This work has been supported by the Spanish CICYT and European Union/FEDER (project DPI2003-07216), and by the University of Zaragoza (UZ2002-TEC-05).

References

[1] N. Mirghafori, "Multi-band speech recognition: A summary of recent work at ICSI", *Technical Report TR97-051*. ICSI, Berkeley, USA, 1997

[2] H.G. Hirsch and D. Pearce, "The AURORA Experimental Framework for the Performance Evaluations of Speech Recognition Systems under Noisy Conditions", in I*SCA, ITRW ASR2000*, Paris, France 2000

[3] http://www.sensoryinc.com/

[4] Y.Y. Shi, Liu J. and Liu R.S, "Single-chip speech recognition system based on 8051 microcontroller core", *IEEE Transactions on Consumer Electronics*, Vol. 47, 1, pp. 149-153, 2001

[5] B. Burchard and R. Romer, "A single chip phoneme based human machine speech recognition system for consumer applications", IEEE Transactions on Consumer Electronics, Vol. 46, 3, pp. 914-919, 2000

[6] K. Nakamura et al., "Speech recognition chip for monosyllables", *Proceedings of the Asia and South Pacific Design Automation Conference* (ASP-DAC 2001), pp. 396-399, 2001

[7] T. Kinnunen, "Spectral Features for Automatic Text-Independent Speaker Recognition". PhD Thesis. University of Joensuu, Finland, 2004

[8] E. Karpov, "Real Time Speaker Identification". PhD Thesis. University of Joensuu, Finland, 2003

[9] L. Docio-Fernandez, N. Morgan, and D. Gelbart, "Far-field ASR on inexpensive microphones", *ICASSP*-2003, Hong Kong, 2003

[10] T. Kohonen, *Self-Organizing Maps*, 3ed edition. Springer-Verlag, Berlin, 2001

[11] J. Campbell, "Speaker Recognition: A Tutorial", *Proceedings of the IEEE*, Vol. 85, Sept., 1997

[12] A. Xafopoulos, "Speaker Verification, an overview", TICSP, Tampere University of Technology, Finland, 2001