DOI: 10.1109/maes.2013.6477866

Design of an Embedded Speech Centric Interface for Applications in Handheld Terminals

Ascensión Gallardo-Antolín, Ana I. García-Moral, Yago Pereiro-Estevan, and Fernando Díaz-de-María, *Member*, IEEE

Abstract—**In this paper, we describe the design of an embedded speech centric interface, which includes automatic speech recognition and text-to-speech conversion, for mobile terminals. The speech interface is part of an application development platform for handheld devices using wireless technologies. The development platform provides support to make the voice-enabled component of the interface fully configurable from the remote server so it can be directly reused in different applications. As an example of the possibilities of the system, we present an application that allows access to real-time stock prices via wireless communications using a standard Personal Digital Assistant (PDA)[1].**

Index Terms—**Speech interface, automatic speech recognition, portable devices, wireless communication networks, real-time information system.**

I. Introduction

Nowadays, portable devices (in particular, mobile phones and Personal Digital Assistants (PDAs)) are part of our daily life. Furthermore, the potential connectivity of this type of devices with a wide range of services through wireless communication networks has triggered the creation of a rich variety of applications. However, the reduced size of portable devices actually makes it difficult to use many of these applications. In fact, there is very limited space to locate buttons or unfold menus.

Voice-enabled or speech interfaces are one effective solution to efficiently manage both, the possibility of taking advantage of the inherent potentiality of these devices and their small size. Some years ago, to embed speech recognition algorithms was inconceivable because of the high computational demands of such a technology. Today, embedded speech recognition is feasible, for constrained tasks ([1], [2] are good examples). However, a great effort must be exerted to achieve complete optimization of this kind of systems in terms of computational resources, so that embedded voice-enabled engines consume only a low percentage of the device capabilities to allow the usage of ASR in a variety of applications on various platforms.

An important advantage of speech communication is that it can be successfully combined with other natural input modes such as touch and vision in order to enhance the conventional pen/screen-based interfaces of portable devices [3]. Therefore, this kind of *speech centric* interfaces allows a friendly interaction between the portable device and the human user.

[1] Ascensión Gallardo-Antolín and Fernando Díaz-de-María are with the Department of Signal Theory and Communications, Universidad Carlos III de Madrid, 28911-Leganés (Madrid), SPAIN (e-mail: {gallardo, fdiaz}@tsc.uc3m.es). Ana I. García-Moral and Yago Pereiro-Estevan are now with Fonetic Solutions, S.L.

In this paper, we present a novel system consisting of a demonstration platform in which it is possible to implement new applications using wireless technologies and speech interfaces in handheld devices. As an example of the functionality of this prototype, an application has been designed and developed whose purpose is to access real-time information (in this case, stock prices) via the General Packet Radio Service (GPRS) using a standard PDA[2].

During the specification phase, several requirements were established with the aim of obtaining a real system that could be used in currently available devices. These requirements are as follows:

- The developed prototype should not be specific to a particular application. The system should be designed as a platform that allows easy deployment of new applications.
- Use of *IP data mobile networks*. The system should be fully operative with current GPRS networks and with the QoS and throughput provided nowadays. If the usability of the system is guaranteed with GPRS, better results can be expected on other wireless data networks as 3G, High Speed Downlink Packet Access (HSDPA) or WIFI.
- *Commercial available devices* with no special hardware requirements (only microphone and loudspeakers) should be used. The designed system should run on different mobile devices (mainly, PDAs and mobile phones) and with the corresponding operating systems (in particular, Pocket PC and Symbian).
- The interaction with the user should be supported through a *speech centric interface*. The speech interface should contain two main subsystems: Text-To-Speech (TTS) and Automatic Speech Recognition (ASR). Besides, both modules should be embedded in the local device and use a low percentage of its resources.
- The speech interface should allow *voice interaction of different speakers* without any specific training.

As it will be shown in the rest of the paper, the developed prototype fulfills all these conditions.

In this paper, we describe the whole system but focus on the voice-enabled interface, which has been so designed that special attention is given to the following issues:

- *Optimization of the computational and memory requirements*. This has been accomplished by means of the fixed-point implementation of the speech interface, dimensionality reduction of ASR feature vectors and the use of dynamic vocabularies.
- *Portability to other applications*. Although in this paper we focused on the application for real-time access to stock price information, the speech interface is flexible enough to support other kinds of tasks with minimum changes in its structure. In fact, we have designed and implemented on it other applications, as for example, a personal agenda manager. Portability is mainly achieved by the use of allophone-like acoustic units in the ASR subsystem, which allows the immediate inclusion

of new words in the dictionary, and the use of the TTS subsystem, instead of pre-recorded messages, which allows the reading of any text. In addition, the required data (acoustic models, dictionary, etc.) are provided (downloaded) from the server side during the initialization of the application, so the ASR system is easily adaptable to different applications.

- *Robustness*. To guarantee its usefulness, the ASR system has to achieve high recognition rates in real operation conditions. In this case, the ASR system should work properly in a relatively quiet office environment with the presence of non-speech sounds (such as door slams or keyboard typing) and with speech recorded with a low-quality microphone. When no enough training data recorded in the new environment is available, robustness can be accomplished by applying speaker or task adaptation techniques [4]. However, in our case, we have directly collected a specific speech database in PDA environment.

The rest of the paper is organized as follows. Sections II and III describe the whole system and the application selected for this development, respectively. The design of the speech interface and its fixed-point implementation are explained in detail in Section IV and V, respectively. Experiments and results for the evaluation of the ASR subsystem and the whole spoken interface are presented in Section VI and VII, respectively. Finally, conclusions are outlined in Section VIII.

II. System Description

The aim of the system described in this paper is to allow easy and friendly access to remote and real-time information through wireless networks using a standard, personal and mobile device, such as PDAs or cellular phones. As shown in Fig. 1, the whole system presents a client-server architecture, in which the client runs on a personal portable device. Server and client modules are connected via the wireless channel provided by GPRS, which allows to distribute the functionality between both modules.

One of the main characteristics of the platform is that it exhibits high flexibility in order to support different applications. In fact, the server offers to the mobile device the possibility of downloading applications. These applications are supported in a collaborative way by both the server and the client. Using an authentication procedure, the user can access a group of different applications, which are defined, deployed, and managed in the server.

Another main characteristic of system is that the interaction between the user and the application in the client is multimodal. Two input modes are allowed: audio (through an automatic speech recognizer) and touch (by using a pen for clicking in a touch-sensitive screen). Similarly, two different output modes are considered: audio (through the use of TTS) and vision (by means of the appearance of text and graphics in the device screen). In this paper we focus mainly on the speech interface which comprises the ASR and TTS subsystems.

[2] With the term *"standard PDA"*, we refer to a commercial available Personal Digital Assistant device with no special hardware requirements (only microphone and loudspeakers).

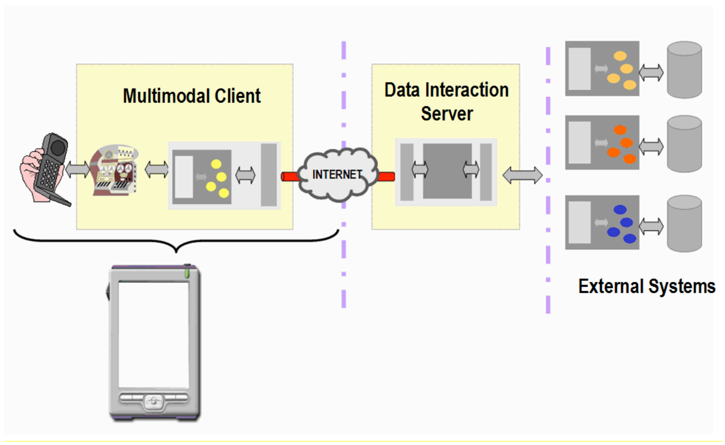


Fig. 1. Block diagram of the real-time information access system, showing its client-server architecture. The client, which includes the speech centric interface system, is embedded in a portable device.

In the following subsections we describe the main components of the system.

A. External systems

The external systems store databases, web publishing sources and other data sources that users want to access from their mobile devices. In our case, these data are related to the stock prices of the Spanish Market (in particular, IBEX-35 index). The external systems are connected to the main server of the system through Internet.

B. Server

The server of the system manages all the aspects related to the connection and interaction between the different data services (external systems) and the users and it contains the applications that can be downloaded by the client (as for example, the application for real-time access to stock prices described in this paper). The server is connected to Internet for accessing the external systems containing the required data sources and it is assigned to an IP address in order to be located by the portable device.

The main functions of the server are the following:

- *Data gateway with external systems.* The server eases the interaction of the system with different data gathering protocols and data sources, such as databases, messaging services, web services, web publishing sources, or unstructured data sources. In our case, the server is responsible of the real-time communication with the external system that contains the stock prices information.
- *Communication with the client and adaptation*. The server performs all the functions related to the communication with the client. As the server must be simultaneously reachable from different types of hand-held devices, it provides the adaptation of the services to these terminals without any intervention from the users.

- *Declarative application model engine.* The server supports the creation of new applications in a declarative way, thus easing the deployment of new solutions. A textual XML (eXtensible Markup Language) representation supports this model, which involves two main issues: the definition of the application structure and the flow representing the different user interactions, including those dealing with the TTS and ASR subsystems. The example application presented in this paper was designed and implemented using this engine.

C. Client

The client manages the dialogue with the user, handles the communication with the server through the wireless network and controls the information to be stored in the portable device. It runs on the mobile device (mobile phone or PDA) and it is composed of the following modules:

- *ASR module.* It takes a speech signal from the device microphone and automatically translates it into text, which is sent to the server.
- *TTS module.* It receives text information and converts it into synthesized speech, which is sent to the portable device loudspeakers.
- *Communication module.* The client system initiates the interaction with the server and coordinates the data flow with it according to the user interaction.
- *Application flow management module.* The client identifies the user commands coming through the keyboard, the pointing device or his/her voice. Using this information, the client determines the next action and launches it. In particular, it determines and coordinates the actions to be carried out by the TTS and ASR subsystems, interprets the application commands and dynamically builds a specific vocabulary for the ASR system according to the application state. Some illustrative examples of these actions are retrieving or sending data from/to the server, launching a TTS to request more information from the user, storing information locally in the device, etc. As mentioned before, all the logic that defines a specific application is retrieved from the server in XML and executed in the client according to the user profile.

D. Characteristics of the portable device

The portable device used is a commercially available PDA with a 16-bit, fixed-point CPU at 206 MHz with a 32 MB on-chip memory with no special hardware requirements (only microphone and loudspeakers).

E. Communication between server and client modules

The communication between the client and the server is wireless and works on actual IP data mobile networks. Our intention was to design the system as generic and independent of the wireless network as possible. Though European telecom operators are offering more advanced data services (as 3G) that can be exploited in mobile data solutions, we have focused on GPRS for wider availability than 3G or HSPA radio. Nevertheless, as the usability of the system is guaranteed with GPRS, it is plausible to assume that better results (specially in terms of data transfer rate) can be expected on 3G.

GPRS manages packet-switched data in GSM networks. Providing a maximum transmission rate of 115 kb/s, facilities such as web browsing and other services requiring data transfer become feasible. HTTP has been selected as the communication protocol between the PDA and the server.

The communication between the mobile client and the server is mainly textual. On one hand, from the client side the user makes the information request using his/her voice. The speech is transcript to text by the ASR subsystem (which runs locally in the portable device) so the client sends the query in text format to the server. On the other hand, the server obtains the required information (which is also textual) from the external system and sends it to the client. In the client, this information is presented to the user in the hand-held device display or it is uttered using the TTS subsystem.

F. Implementation

Main functions of the server and client were implemented in Java (J2EE and J2ME) with the exception of the ASR and TTS submodules of the speech interface that were implemented in C++ and compiled using EMbedded Visual C++ 4.0.

JDBC (Java Database Connectivity) was used for connecting Java programs to databases as it allows the use of SQL (Structured Query Language) for managing such databases. Finally, the application definition was made with XML.

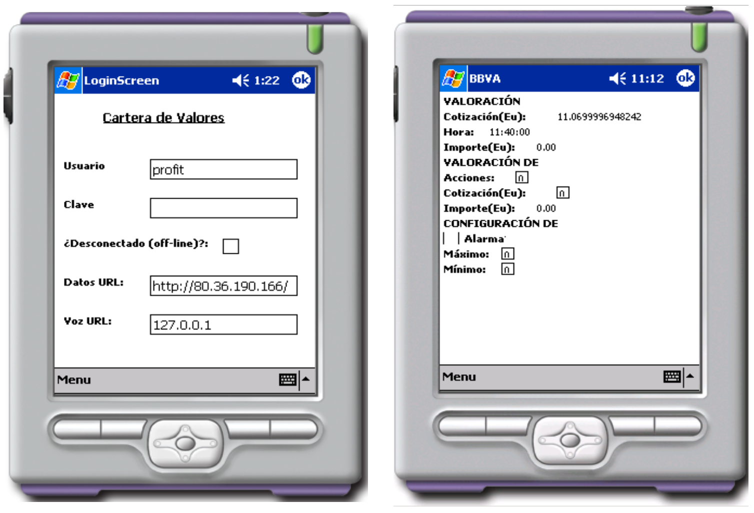
In the implementation stage of the system, we deal with two main difficulties. The first one was related to connection issues between the different modules of the system. The second one was the large computational requirements of the early versions of the ASR subsystem. In sections V and VI we describe the main strategies used for reducing such computational cost.

III. The Application: Real-Time Access to Stock Prices

The application selected for showing the performance of our system is called “Stocks Portfolio.” This application allows the user to access real-time stock prices of the Spanish Market (in particular, IBEX-35 index) by entering the name of a certain company either using the pen or her/his voice. This query is sent to the main server of the platform, which is connected to the external system that stores the database with the stock information. Once the requested information is available, the server sends it to the client and finally, the user can access it either through graphs or text in the PDA screen or by means of a synthetic voice using the TTS subsystem. In addition, the system is capable of managing the user’s stocks portfolio that contains the collection of the investments owned by him/her. Also, the system allows the user to perform several operations on the portfolio as consulting, selling or buying company stocks and shares.

Fig. 2a illustrates the initialization screen of the application. When the application starts up, the user is authenticated by entering his/her login and password using the pen. This could be also accomplished by using the user's voice and speaker identification techniques [5]; however this alternative is beyond the scope of this paper. Next, it is necessary to provide two IP addresses: that of the stock information database (*"Data URL"*) and that of the main server of the system which stores (among other items) the configuration information of the ASR and TTS subsystems (*"Voz URL"*). In normal operation, the system establishes a GPRS connection to these servers, from which several data are loaded or updated (for example, acoustic models or dictionaries for the ASR subsystem) in the PDA. Then, the user can start the interaction. In addition, for demonstration purposes, it is also possible to run the application without a remote connection by marking the box *"¿Desconectado (off-line)?"*. In this case, the system uses local databases previously loaded in the PDA.

Fig. 2b shows the screen obtained when the user asks for the information corresponding to the company *"BBVA"*. An overview of this information is displayed on the screen, but it is also possible to access certain additional details either using the pen or uttering the corresponding command word (for example, saying "*Cotización"* for accessing the real-time price of the company stock). The answer is read aloud by the TTS subsystem.



Cartera de Valores

Usuario: profit
Clave:
¿Desconectado (off-line)?: ☐
Datos URL: http://80.36.190.166/
Voz URL: 127.0.0.1

VALORACIÓN
Cotización(Eu): 11.0699996948242
Hora: 11:40:00
Importe(Eu): 0.00
VALORACIÓN DE
Acciones: ☐
Cotización(Eu): ☐
Importe(Eu): 0.00
CONFIGURACIÓN DE
☐ Alarma
Máximo: ☐
Mínimo: ☐

(a) (b)

Fig. 2. Two examples of application screen shots.

In this application, the basic dictionary of the speech recognizer is composed of 60 words: 25 Spanish command words (*"Principal"*, *"Abrir"*, *"Cerrar"*, …) and 35 proper names corresponding to the 35 companies included in IBEX-35. The speech recognizer considers multiple pronunciations or synonyms per word (for example, the company *"Red Eléctrica de España"* can also be named as *"REE"*), so the effective lexicon size increases. Nevertheless, it is worth mentioning that the full dictionary is not always active; in fact, the dialogue manager selects the allowable words at each application state. This mechanism achieves important improvements in both execution time and word recognition accuracy.

IV. Speech Interface

In this section, we describe the embedded speech interface designed for our system; in particular, the ASR and the TTS subsystems.

A. *ASR subsystem*

The ASR subsystem is an isolated-word, speaker-independent system based on Hidden Markov Models (HMMs). It is composed of two main modules: the parameterizer and the classifier, whose main functionalities are described in the next subsections.

1) *Parameterization module*

The parameterizer is a pre-processing stage that converts the speech signal into a set of feature vectors more suitable to perform automatic speech recognition. It involves two stages: acoustic parameter computation (in this case, Mel-Frequency Cepstrum Coefficients, MFCCs) and reduction of the feature vector dimensionality.

The MFCC computation algorithm is a modified version of the standard developed by ETSI [6] for Distributed Speech Recognition (DSR). MFCC are extracted every 10 ms. following the procedure illustrated in Fig. 3. First of all, a filter with a pre-emphasis factor of 0.97 is applied to the speech signal to compensate for the slope of voiced segments. Next, the pre-emphasized signal is windowed using a 25 ms Hamming window and its magnitude spectrum is computed via a Fast Fourier Transform (FFT) of 256 points. The magnitude spectrum is passed through a mel-scaled filterbank of 23 triangular band-pass filters and the logarithm of the energy corresponding to the output of each filter is computed. These log filterbank energies are decorrelated using a Discrete Cosine Transform (DCT) obtaining a set of 12 MFCCs. The normalized log-energy of each frame is also computed. Finally, the first time-derivative of the MFCC and the log-energy (denoted as □MFCC and □log-energy, respectively) are appended yielding to a feature vector of 26 components.

In order to minimize the computational cost and storage requirements of the ASR system, the feature vector dimensionality is reduced by means of a Principal Component Analysis (PCA) [7]. In practice, PCA is accomplished through the product of original feature vectors and a transformation matrix. If the original dimension is D and the target reduced dimension is D', the PCA transformation matrix is D' x D and is composed of the D' eigenvectors corresponding to the D' largest eigenvalues of the global covariance matrix of the training data. The optimum vale of D' has been evaluated in Section VI.

The calculation of the transformation matrix is computationally expensive. However, this process is carried out during an off-line training phase, being the resulting matrix loaded in the mobile device during the initialization of the application. Therefore, from the implementation point of view, the use of PCA only entails a product of a matrix and a vector per frame.

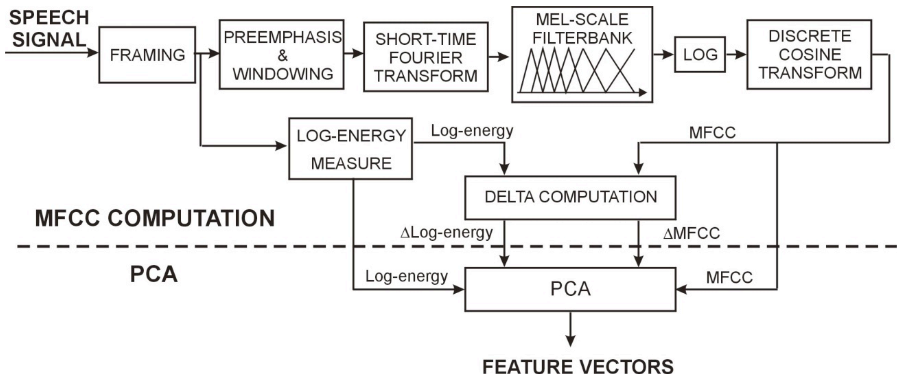


Fig. 3. Top-level diagram of the speech parameterization procedure.

2) *Classification module*

The purpose of the classification module is to decide which word or sentence has been uttered, according to a likelihood criterion: the probability of the observation given each acoustic model is computed and the acoustic unit represented by the winner model is selected.

In our case, the classification module is based on the well-known HMM paradigm, in which the acoustic decoding is performed by using the Viterbi algorithm [8]. The acoustic models (context-independent Continuous Density HMMs with three states and a mixture of 8 Gaussians per state) are built during an off-line training phase using the Baum-Welch algorithm [8] and the HTK toolkit [9].

We have considered an alphabet consisting of 30 allophone-like units plus one silence unit for reducing the effect of end-pointing inaccuracies. With this inventory of basic units, it is possible to build any word; consequently, the lexicon of the ASR system can be easily updated or totally changed without training and storing new acoustic models.

3) *Improvement of the ASR system robustness: speech data collection*

The performance of ASR systems working in real-world applications dramatically decreases when there is a mismatch between the acoustic conditions in which the training data are recorded and those in which the system actually operates. In our system, one of the main sources of mismatch is the handheld device microphone. In fact, while most of the commercial databases used for building the acoustic models are recorded with high-quality desktop or fixed-telephony microphones, for economic reasons, PDAs and other portable devices usually are provided with low-quality microphones.

This mismatch can be partially alleviated if the training database contains speech recorded with a microphone embedded in a PDA. Therefore, we have designed and collected a reduced speech database in a PDA environment. This database, called BD-PDA, was recorded directly in a PDA using its microphone and in a relatively quiet office environment (at a SNR between 15 dB and 20 dB), in which some undesired sounds, typical of office environment (such as door slams or keyboard typing), were present. It was recorded at 8 kHz with 16 bits per sample. The database consists of 6300 utterances pronounced by 50 Spanish speakers (25 males and 25 females), with up to three recordings from each speaker. The language is Spanish.

B. TTS Subsystem

Our TTS is a concatenative system that generates a male voice in Spanish language. To reduce its footprint for handheld device usage, the TTS subsystem uses a repertory of only 545 diphones recorded at 8 KHz. For prosody generation, the system uses a parametric model of the pitch frequency (F0) contour. The considered parameters are mainly the first tonic F0 value, the decline of F0-contour and the sentence length. For phonemic durations, a simple multiplicative model is applied.

Although this TTS system is very simple, its intelligibility and naturalness are quite reasonable, taking into account the limited resources of the PDA and the characteristics of its loudspeakers.

V. Fixed-point implementation of the speech interface

Portable devices, such as mobile phones or PDAs, are limited in computational capacity, memory, and power consumption. Most of them use a 16-bit, fixed-point CPU with only on-chip memory. Since ASR and TTS processes are computationally demanding tasks, their implementation in these devices should be in fixed-point arithmetic; otherwise, they would not be efficient enough to work in real-time. Although basic arithmetic operations, such as addition, subtraction, or multiplication can be easily performed in fixed-point arithmetic, it is necessary to deal with several difficulties, such as quantization and truncating noises, the possibility of under- and overflow, etc.

In the next subsections, we describe the fixed-point implementation of the main modules of the speech interface.

A. Fixed-point implementation of the ASR parameterization module

The speech processing algorithms are quite intensive in terms of arithmetic operations; however, they are not very demanding from the memory access point of view. Taking into account these characteristics, the conversion from floating-point to fixed-point arithmetic has paid special attention to the following issues:

1. *Algorithmic optimization*, which involves the substitution of most time-consuming algorithms by their corresponding fixed-point approximations.
2. Elimination of divisions, which can be implemented as a multiplication by the reciprocal of the divisor.
3. Elimination of global variables, as much as possible. All variables are either local or defined in a structure in the header file. Also, the use of large data structures is avoided.

4. Minimization of the number of parameters passed to a function. All functions receive only a pointer to a parameter structure.
5. Use of pre-calculated Look-Up-Tables (LUTs) for computational cost reduction. In particular, in the initialization of the parameterization process, the following tables are loaded in memory: Hamming window, twiddle factors in the FFT function, Mel-filter coefficients, and DCT cosine and PCA transformation matrix. All values in these LUTs are stored as 16-bit integers.
6. Optimization of the Qn precision (a fixed-point data in a Qn resolution has *n* bits to the right of the -hypothetical- decimal point) for every variable in each stage.
7. Normalization of variables after every mathematical operation as an overflow protection mechanism. This is accomplished through the appropriate down/up-scaling of the data, in order to not exceed the maximum/minimum integer limit.

In the next paragraphs, more details are given on steps 1 and 6.

1) Algorithmic Optimization Stage

The FFT function, the Square-Root function (required for the magnitude spectrum computation) and the Natural Logarithm function were replaced by their corresponding fixed-point approximations.

The fixed-point version of the FFT algorithm is an adaptation of the floating-point FFT code provided in the DSR standard [6], which can be adapted using conventional fixed-point operations such as addition, subtraction, multiplication and shifting.

To avoid using the C library function *sqrt(.)*, the Square-Root function was replaced with the algorithm described in [10], which returns the integer part of the square-root computation. Our experimental results show that this approximation was appropriate for our purposes.

For the Natural Logarithm function, the following factorization was used:

$$\ln(x) = \log_2(x)\ln(2) \tag{1}$$

where *ln(2)* is converted to a fixed-point integer number and stored as a constant. Furthermore, since x is a fixed-point number in *Qn* format, it can be expressed as $x = 2^y z$, with $1 \le z < 2$, where z and y can be efficiently computed by successively dividing x by 2 (through bit displacements). Thus,

$$\log_2(x) = \log_2(2^y z) = y + \log_2(z) \tag{2}$$

where, taking into account that $1 \le z < 2$, $\log_2(z)$ can be approximated by ($z - 1$) yielding to the following expression:

$$\log_2(x) \approx y + z - 1. \tag{3}$$

2) *Selection of the Qn precision*

Most of the input and output variables in the algorithm are stored as long integers, except for the input and output variables of the “Pre-emphasis” and “Hamming windowing” functions and the input variables of “FFT”, “Mel filtering” and “Log-energy measure” routines, which are short integers.

With respect to the selection of the Qn precision for each variable, two criteria were considered: the dynamic range of the data at each stage (for example in the “Mel filtering” routine, it is important to use Q15 given the dynamic range of the filterbank coefficients) and the minimization of the percentage of overflows that occurred on the training material.

Table 1 shows the Qn values used for the input, auxiliary and output variables of each stage of the ASR parameterizer. Special care should be taken with the choice of Qn for the MFCCs, log-energy and their corresponding first derivatives, since these parameters are the inputs to the PCA module, which, in turn, is the previous stage to the classification subsystem. For this reason, we tried three different values (Q7, Q9 and Q11) to observe the relationship between the precision used and the percentage of overflows occurring in the stages involved. Results obtained with the BD-PDA database are shown in Table 2. Q7 is the best value because it minimizes the number of overflows and it maximizes the recognition rate of the system (see Section VI).

TABLE 1. Qn VALUES FOR THE VARIABLES IN THE PARAMETERIZATION MODULE

Stage	**Qn**		
	Input	**Auxiliary**	**Output**
Pre-emphasis	Q0	Q15	Q0
Hamming windowing	Q0	Q15	Q0
FFT	Q0	Q15	Q0
Magnitude computation	Q0	Q15	Q0
Mel filtering	Q0	Q0/Q15	Q11
Logarithm	Q11	-	Q9
DCT	Q9	Q13	Q7/9/11
Log-energy measure	Q0	Q0	Q7/9/11
Log-energy normalizat.	Q7/9/11	-	Q7/9/11
Delta computation	Q7/9/11	Q15	Q7/9/11
PCA	Q7/9/11	Q7/9/11	Q7/9/11

TABLE 2. PERCENTAGE OF OVERFLOWS OCCURRING IN THE PARAMETERIZATION MODULE AS A FUNCTION OF THE FIXED-POINT PRECISION USED

Submodule	**Fixed-point**		
	Q7	**Q9**	**Q11**
Log-energy measure	0.001%	0.001%	0.001%
Log-energy normalizat.	0.047%	0.047%	0.061%
Delta-MFCC	0.000%	0.005%	1.609%
Delta-log-energy	0.000%	0.001%	3.019%

B. *Fixed-point implementation of the ASR classification module*

The logarithm version of the Viterbi search algorithm is used to reduce its computational requirements; thus, multiplications are converted into simple additions, the use of exponentials is avoided, and the probabilities involved can be represented by integer values [8].

C. *Fixed-point implementation of the TTS subsystem*

For the TTS fixed-point implementation, we used the same procedures described in subsection V.A for the ASR parameterizer. In addition, it is worth mentioning that both modules share several routines (such as Hamming windowing).

VI. Evaluation of the ASR subsystem

A. *Experimental set-up*

To evaluate the ASR subsystem, we used the BD-PDA speech database described in subsection IV.A.3). The dictionary is composed of 60 words chosen for fulfilling the requirements of the task described in Section III. Since this database is quite limited to achieve reliable results, we have used a 10-fold cross validation to artificially extend it. Specifically, we have split the database into 10 balanced groups: 9 of them for training and the remaining one for testing, averaging the results afterward. All the results were obtained following this procedure.

B. *Experimental results*

First, to compare the performance of both fixed- and floating-point implementations of the ASR systems, we carried out a set of experiments varying the resolution (Q7, Q9 and Q11) of the fixed-point feature vectors. Table 3 shows the word recognition rates achieved with these different implementations. In this case, PCA was not applied; therefore, the dimension of the acoustic vectors is 26.

From these experiments, it can be observed that the recognition rate obtained with the floating-point ASR system was very similar to those obtained with the different fixed-point versions of the ASR system. This fact confirms the validity of the approximations described in Section V. Regarding the particular value of Qn, although the performance of the ASR system for different resolutions was quite similar for all cases, it was slightly better for Q7. As shown in subsection V.A, Q7 was also the value that produced the minimum percentage of overflows, therefore we used this resolution in the final version of the system.

TABLE 3. RECOGNITION RATES FOR THE FIXED- AND FLOATING-POINT IMPLEMENTATIONS OF THE ASR SYSTEM

Floating-point	**Fixed-point**		
	Q7	**Q9**	**Q11**
96.81%	96.32%	96.09%	96.08%

Second, we carried out a set of experiments varying the number of components feature vectors after using PCA in order to study the influence of dimensionality reduction on system performance. Fig. 4 shows the results for the Q7 fixed-point ASR system.

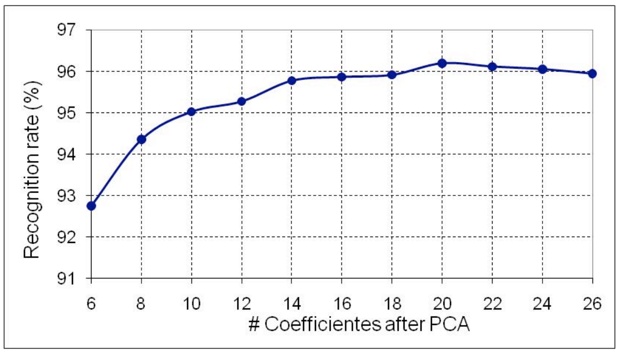


Fig. 4. Recognition rates for different feature vector dimensionalities after applying PCA.

This figure shows that it is possible to reduce the number of components of the feature vectors from 26 to 14–18 without a significant degradation in the system recognition rate. In the final version of the ASR system, we used PCA vectors with 18 components. In this case, the recognition rate only decreased by around 0.45% with respect to the system without PCA. In addition, PCA produced a significant memory reduction and speed improvement of the whole system in comparison with the system without PCA as shown in next subsection.

C. *Processing Time reduction*

Another important key factor for evaluating the performance of the different implementations of the ASR system is the execution time. Table 4 shows the CPU time required for recognizing 50 utterances in a 202 MHz processor for the floating point (without PCA), fixed point (without PCA) and fixed point (with PCA) code implementations. For each case, the CPU time (in ms. and in percentage over the total time) for the main modules of the ASR process (parameterization and classification) is also indicated. The floating point is an optimized code in which, among other improvements, a logarithm version of the Viterbi algorithm is used. In the fixed-point code with PCA, the time required for PCA operations is included into the parameterization module.

TABLE 4. PROCESSING TIME FOR THE DIFFERENT CODE IMPLEMENTATIONS

	Parameterization	**Classification**	**Total**
Floating-point – 26 components	649.3 ms. (6.23%)	9777.2 ms. (93.77%)	10426.5 ms. (100%)
Fixed-point (without PCA) - 26 components	131.3 ms. (4.77%)	2618.7 ms. (95.23%)	2750.0 ms. (100%)
Fixed-point (with PCA) - 18 components	131.6 ms. (6.24%)	1978.2 ms. (93.76%)	2109.8 ms. (100%)

It can be observed that in all cases the classification module consumes the most of the CPU time. The fixed-point implementation (without PCA) produces a processing time reduction of 73.6% with respect to the floating point version, which corresponds to a 79.8% reduction in the parameterization module and 73.2% in the classification one. Comparing the fixed point implementations without and with PCA it can be observed that PCA only increases the processing time in 0.2% in the parameterization module and, however, in the classification module the execution time is reduced by 24.5%. The total reduction is around 23.3% (a decrement of the execution time of 2.91% per parameter reduced).

In summary, the fixed-point implementation of the system with PCA decreases significantly the processing time of the whole ASR subsystem.

VII. Evaluation of the Speech Interface system

For testing the whole speech interface, four tasks of different complexity were designed to cover the main functionalities of the system. The tasks were formulated to the users as open scenarios, as follows:

- Task 1: To obtain the stock information (share price and date) of a certain company included in the stock database.
- Task 2: To buy shares of two different companies before consulting their corresponding information (share price and date) contained in the stock database.
- Task 3: To access the user's stock portfolio, consult the information regarding to the collection of the investments owned by the user (companies, share prices and dates) and sell all the shares.
- Task 4: To obtain the information (share price and date) of two different companies included in the stock database and buy shares of these companies. Subsequently, to consult the shares (companies, share prices and dates) owned by the user accessing his/her stock portfolio and sell them.

All tasks began by opening the application and finished by returning to the main menu and closing the application, using the corresponding voice commands. Names of the companies were balanced across tasks.

The evaluation was carried out in an office environment with 10 native Spanish speakers. Two of them had previous experience in using this kind of speech interfaces and the others did not have experience with these systems. Each of the speakers carried out the previously mentioned four tasks in complexity order. Before the evaluation itself, subjects received basic information about the capabilities of the system and viewed a demonstration. In order to avoid confusions, the user's stock portfolio was emptied in the beginning of the evaluation carried out by each subject.

For each task, several objective measures were taken as proposed in [11]: task completion, recognition rate (percentage of utterances correctly recognized by the ASR subsystem), the number of Voice User Interface (VUI) interactions and efficiency (percentage of the optimum number of steps required to complete a task with respect to the total number of interactions).

TABLE 5. RESULTS OF THE SPOKEN INTERFACE EVALUATION

Task	Task Completion	Recognition Rate	VUI	Efficiency
T1	100%	96.83%	6.30	95.24%
T2	100%	96.19%	10.50	95.24%
T3	90%	89.47%	13.30	82.71%
T4	100%	93.63%	15.70	76.43%
Average	97.5%	93.45%	11.45	85.15%

Results of the speech interface evaluation are shown in Table 5. As it can be observed, almost all the tasks were completed, being the average task completion rate across all tasks and users of 97.5%, which is a satisfactory result. The recognition rate of the speech interface was high (93.45% on average) and only slightly lower than the one achieved in the preliminary testing of the ASR subsystem (around 96%). With respect the efficiency, results are good on average, although there are significant variations between the different tasks. One possible reason is that tasks were presented to users as open scenarios, so they could achieve the corresponding goals using different strategies, which not always involved the optimum number of steps. This effect was especially remarkable in more complex tasks (T3 and T4).

VIII. CONCLUSION

In this paper, we have described an embedded speech centric interface for handheld wireless devices. It has been implemented on a commercially available PDA as a part of an application that allows real-time access to stock prices through GPRS. We have focused mainly in the optimization of the ASR subsystem for minimizing the use of the handheld computational resources. This optimization has been accomplished through the fixed-point implementation of all the algorithms involved in the ASR subsystem and the use of PCA to reduce the feature vector dimensionality. The influence of several parameters, such as the Qn resolution in the fixed-point implementation and the number of PCA components retained, have been studied and evaluated in the ASR subsystem, obtaining word recognition rates of around 96% for the best configuration. Finally, a field evaluation of the system has been performed showing that our design of the speech centric interface achieved good results in a real-life scenario.

ACKNOWLEDGMENT

The authors thank MOVILOK, S.L. for its collaboration and support. This work was supported in part by the Spanish Government grants TSI-020110-2009-103, IPT-120000-2010-24 and TEC2011-26807 and the Spanish Regional grant CCG08-UC3M/TIC-4457.

REFERENCES

[1] I. Varga, S. Aalburg, B. Andrassy, S. Astrov, J. G. Bauer, C. Beaugeant, C. Geibler and H. Höge, "ASR in Mobiles Phones–An Industrial Approach", *IEEE Trans. Speech and Audio Processing*, vol. 10, No. 8, pp. 562–569, Nov 2002.

[2] C. Lévy, G. Linarès, P. Nocera, and J. -F. Bonastre, "Reducing Computational and Memory Cost for Cellular Phone Embedded Speech Recognition System", *Proc. IEEE ICASSP 2004*, pp. 309–312, 2004.

[3] K. Kvale, N. D. Warakagoda and J. E. Knudsen, "Speech Centric Multimodal Interfaces for Mobile Communication Systems", *Telektronikk*, no. 2, pp. 104–117, 2003.

[4] R de Córdoba, J. Ferreiros, R. San-Segundo, J. Macías-Guarasa, J. M. Montero, F. Fernández, L. F. D'Haro and J. M. Pardo, "Air Traffic Control Speech Recognition System Cross-Task & Speaker Adaptation", *IEEE Aerospace and Electronic Systems Magazine*, Vol. 21, Issue 9, September 2006, pp. 12-17.

[5] M. Faúndez-Zanuy and E. Monte-Moreno, "State-of-the-art in Speaker Recognition", *IEEE Aerospace and Electronic Systems Magazine*, Vol. 20, Issue 5, March 2005, pp. 7-12.

[6] ETSI Standard: ETSI ES 201 108 v1.1.2, "Speech Processing, Transmission, and Quality aspects (STQ); Distributed Speech Recognition; Front-end Feature Extraction Algorithm; Compression Algorithms", 2000.

[7] X. Wang and D. O'Shaughnessy, "Improving the Efficiency of Automatic Speech Recognition by Feature Transformation and Dimensionality Reduction", *Proc. EUROSPEECH-2003*, pp. 1025–1028, 2003.

[8] X. D. Huang, Y. Ariki and M. A. Jack, "Hidden Markov Models for Speech Recognition", Chapter 9, *Edinburgh University Press*, 1990.

[9] S. Young, G. Evermann, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev and P. Woodland, "HTK-Hidden Markov Model Toolkit (ver 3.2.1)", *Cambridge University*, 2002.

[10] K. Piromsopa, C. Aportewan and P. Chongstitvatana, "An FPGA Implementation of a Fixed-point Square Root Operation", *International Symposium on Communications and Information Technology*, pp. 587–589, 2001.

[11] G. Branco, L. Almeida, N. Beires and R. Gomes, "Evaluation of a Multimodal Virtual Personal Assistant", *Proc. 20th International Symposium on Human Factors in Telecommunication*, 2006.