PORTABLE IMPLEMENTATION OF A TEXT-TO-SPEECH SYSTEM FOR PORTUGUESE

Rodrigo C. Torres[1], *José M. de Seixas*[1], *Sergio L. Netto*[1], *Diamantino R. da S. Freitas*[2], *Eduardo F. Brasil*[1]

[1]Signal Processing Laboratory
Federal University of Rio de Janeiro, Brazil
{torres,seixas,sergioln,efb}@lps.ufrj.br

[2]Laboratory of Signals and Systems
University of Porto, Portugal
dfreitas@fe.up.pt

ABSTRACT

Speech synthesizers are mostly developed for general purpose personal computers, which seems to be a problem for users who require mobility, as that is the case for disabled people. Therefore, a Portuguese speech synthesizing system inspired on the time domain pitch synchronous overlap and add technique was developed onto a digital signal processor due to its portability advantages. To allow full integration of the whole system, the original audio data, formed by 922 diphone units for the European Portuguese, was encoded using a code-excited linear prediction technique. By doing so, the database size dropped to less than 1% of its original size, with minimal loss in quality due to such coding process. The final system is able to operate with a personal digital assistant (PDA) or any custom device able to generate and send written text by means of the UART serial protocol.

1. INTRODUCTION

There is an important current demographic trend in Europe with respect of the aging of the population. The population over the age of 60 in the EU is expected to rise to 25% of the total population by the year 2020 [1]. The percentage of disabled people is currently about 11% in the EU and should rise to 17% by 2030 [2]. Nowadays, there is a clear claim for services and equipments that are designed taking the needs of disabled and older users.

An useful application for speech synthesizing systems is in the support of disabled people. People with severe visual deficiencies are unable to read a newspaper. Individuals with speech problems are incapable of speaking over the telephone.

Nowadays, there are several speech synthesizers implemented onto general purpose personal computers. However, disabled individuals can only take advantage of such systems while remaining near a computer. Therefore, a portable, cheap, and easy-to-use speech synthesizer system was developed, allowing its use in a wide range of situations, enhancing the independence of this kind of user.

Current digital signal processing technology allows the integration of very complex digital devices, like commercial PDAs and mobile phones. This level of compactation was achieved by exploiting specific features of the digital processing algorithms, in order to optimize their processing. Besides, developers can already count on extremely fast, small and power-saving digital devices. So, focusing on the mobility, a compact speech synthesizer system for Portuguese was implemented using digital signal processor (DSP) technology.

The speech synthesizing algorithm used in this work was inspired on the time domain pitch synchronous overlap and add (TD-PSOLA) [3], due to its simplicity and quality. In order to improve speech naturalness, a simple version of the Fujisaki model intonation contour [4] was used to incorporate some prosody to the synthesized speech. One problem, however, with the TD-PSOLA is that the amount of memory needed to store diphones, pitch marks, and voicing information tends to be very large. Hence, in order to reduce the overall cost of the developed system, the database needed to be compacted and resampled. For that purpose, the audio samples were coded using the code excited linear prediction (CELP) [5] speech coding system, so that the final system would be able to fit entirely (program plus the database) in only 512 Kbytes of non-volatile memory. The result was a portable text-to-speech conversion system for European Portuguese. Subjective evaluation of the synthesized speech indicated that the overall system achieved a good performance.

This paper is organized as follows: Section 2 presents the digital device used to implement the proposed system, as well as the DSP model and evaluation board chosen for the prototype development. Next, the overall implementation of the synthesizing system is shown in Section 3. Then, in Section 4, the steps performed to reduce the database to fit it in the 512 Kbytes flash memory available for the prototype are presented. Section 5 presents the results on system evaluation. Finally, Section 6 concludes the paper emphasizing its main contributions and future developments.

2. THE DIGITAL SIGNAL PROCESSOR

Digital signal processing requires some standard operations, like multiplication and accumulation, modular operations and high iterativity levels. So, a specific kind of processor was developed to explore the inherent operations of digital signal processing algorithms. Such processors, generally called digital signal processors (DSP) can optimally perform the required operations in order to achieve the real-time restrictions [6] which apply to this target application.

The DSP chosen for this speech synthesizer application was the SHARC ADSP-21160M [7]. Its inner structure is presented in Fig. 1. It is an 80 MHz high performance, 32-bit floating point processor, which executes every instruction in just one single clock cycle. It has an internal memory of 4 MWords and an additional processing element (with an additional multiplier, ALU, shifter and data register file) [8] for single instruction on multiple data stream (SIMD) [9] operations.

The overall system was implemented in the EZ-Kit 21160 evaluation board [10], which can be visualized from Fig. 2. This evaluation board contains a ADSP-21160M pro-

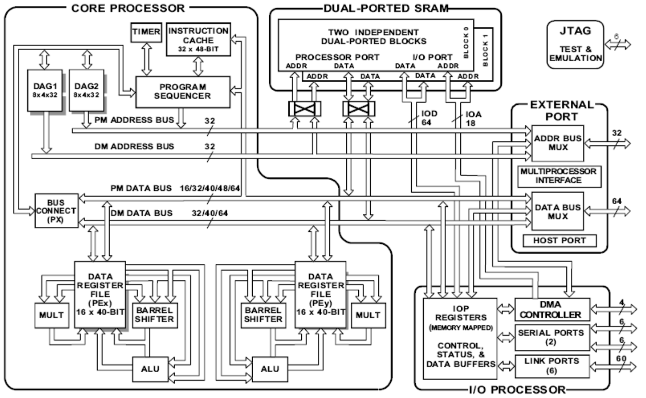

Figure 1: *Inner structure of the ADSP-21160M (extracted from [7]).*

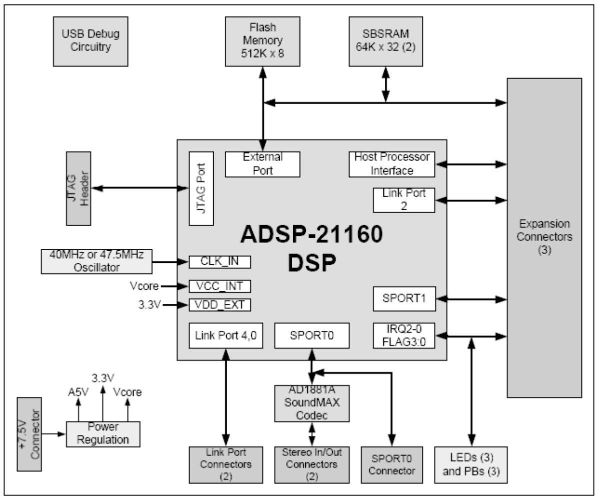

Figure 2: *Schematic diagram of the evaluation board used to implement the synthesizer (extracted from [10]).*

cessor. It acts as an interface between the DSP and external devices, reducing the complexity of prototypes development. Besides, it already provides some commonly used resources like:

- External memory modules (total of 512 Kbytes).
- Flash memory unit of 512 Kbytes for stand alone operations.
- CODEC for A/D and D/A conversions.
- Standard connectors attached to the DSP's serial ports.

3. THE TEXT-TO-SPEECH SYSTEM

The system was implemented in such a way that it can interface with any device capable to produce written text and send it serially using the UART protocol, with RS-232 voltage levels [11]. The block diagram of the synthesizer is presented in Fig. 3. First, the user types the text he wants to synthesize in a PDA, for instance. Then, the text is serially transmitted to the DSP by the UART protocol, triggering the synthesizing algorithm. At the end, the resulting audio samples are sent to the D/A converter and the generated analog signal is sent to a loudspeaker. On the DSP side, the algorithm runs as follows:

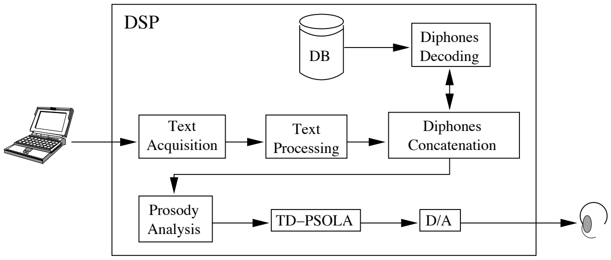

Figure 3: *Text-to-speech block diagram.*

1. The text to be synthesized is received by the DSP through its serial port.
2. Phonetic transcription and accentuation information are extracted.
3. Analyzing the phonemes, a list with the needed diphones is generated.
4. The information of each required diphone is read from the database. The diphone is then decompressed and concatenated to the previous diphones already synthesized.
5. A simple prosody analysis is performed on the text, generating, as result, the $F0$ contour to be used in the final synthesizing step.
6. The speech synthesis is executed and the synthesized audio samples are stored in a queue, to be sent to the D/A converter.
7. The system finally returns to step 1 and waits for the next text.

Since the synthesizing process is much faster than the audio reproduction (see Section 5), this algorithm would quickly exhaust the available memory. In order to avoid this effect, before allocating the memory needed to perform the synthesis of one portion of the text, the system first verifies whether it has enough memory for the task. If not, the system waits until some memory is released, since the synthesizing algorithm and the D/A conversion are executed in parallel. Once the amount of memory is available, the system resumes the synthesizing process.

4. DATABASE COMPRESSION

For the development of the database [12], a text containing all the phonemes for the European Portuguese language was reproduced by a professional reader. The text was recorded with a sampling rate of 22.05 kHz with 16 bit resolution. Then, the diphones were manually extracted and the pitch marks, as well as the voicing information, were inserted by automated methods. The final result was a high quality database, but requiring a high amount of memory ($\sim$23 Mbytes) for storage and processing.

The challenge was to reduce this database memory requirement, so it could fit to a memory size of 512 KBytes. To do so, the following steps were performed, where each step inherits the compression benefits of the previous steps:

1. The original version of the synthesizing algorithm used a vector with the same length of the audio samples to store the respective pitch marks and the voicing information. The first step in compressing the database was to store the indexes where a pitch mark was occurred, along with two voicing flags (one for each phoneme that formed the

diphone). With that approach, the database size was decreased by a factor of almost 3.

2. The audio samples were resampled to 8 kHz, reducing the number of audio samples by a factor of approximately 3.
3. The audio information of each diphone was coded using a CELP system, compressing the database by a factor of approximately 15.

The CELP coding system used [13] was operating with speech segments of 20 ms, corresponding to 160 speech samples per segment at the 8 kHz rate for telephone systems. For each segment, 10 linear prediction coefficients were determined and then transformed onto 10 line-spectrum coefficients [14] for a 32-bit quantization procedure. To determine the excitation information, each 20-ms block was sub-divided into four sub-blocks of 5 ms or 40 samples each. The excitation was constructed based on an extensive search over two codebooks: a fixed one, mainly composed by clipped white noise, and an adaptive one, with 4096 lines each. In that manner, both codebooks required altogether 96 bits for indexing the entire speech block. The codebook gains were uniformly quantized using 8 bits each, yielding a total of 64 bits to represent the gain coefficients for a given block. Hence, a 20-ms speech segment requires only $(32+96+64) = 192$ bits, after encoding, to represent all 160 original samples, while if the samples were stored in 32-bit words (the DSP default) it would require 5120 bits.

The results obtained in each compression step can be observed in Table 1. As it can be seen, the final database version dropped to less than 1% of the original database, with a final size of just 190 Kbytes. The main disadvantage of this compression level is the additional processing required for decoding the audio samples and recalculating the pitch and voicing marks for each sample. Such overhead, however, is easily overcome due to the high processing speed of the chosen DSP.

Table 1: *Absolute and percentual sizes obtained from each database compression version.*

Version	**Size (Kbytes)**	**% from the Original**
Original	23,330	100.00
Optimized	7,820	33.52
Resampled	2,870	12.30
Coded	190	0.81

5. SYSTEM EVALUATION

5.1 Time Analysis

The percentage of the required time for each step of the whole text-to-speech process was measured, and results can be observed in Fig. 4. As one can see, most of the time is spent in the decoding phase, but this overhead is perfectly acceptable in this DSP implementation. For instance, the total time elapsed from the moment the text is acquired to the time when the synthesized audio samples are stored in the queue is ∼34 ms for a small sentence, which fulfills most real-time requirements. Such delay can be considered negligible for the human hearing system for all practical purposes. Also, this time delay occurs only for the first sentence, since, by the time the algorithm sends all the audio samples to the D/A, more sentences will be already available in the output queue, thus, reducing this delay to virtually zero for further sentences.

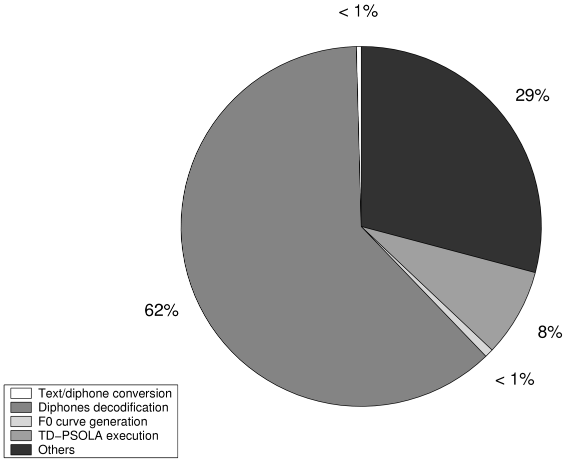

Figure 4: *Fraction of time spent in each step of the synthesizing system.*

5.2 Objective Test

The objective test was performed to verify the overall synthesizing quality. For that, the Itakura Distance (ID) method [15] was used. In such test, the higher the ID, the more affected the speech tends to be by the compression step. The distances were calculated between the same text, synthesized using the original 22 kHz sampling rate database, the 8 kHz version with no coding and the coded 8 kHz database used in the prototype implemented. A total of 35 randomly chosen sentences from a newspaper, with different lengths, was used for this test, and the results are presented in Table 2. It can be observed that the distance from the 22 kHz version to the others is large, when compared to the distance between the two 8 kHz versions. As the distance between the coded database and the 8 kHz database is small, the loss of quality was generated mainly by data resampling, and not by the coding process.

Table 2: *Itakura distance between the text synthesized using the original (22 kHz), the 8 kHz and the 8 kHz coded databases.*

	22 kHz	**8 kHz**	**8 kHz Coded**
22 kHz	$0,00 \pm 0,00$	$3,75 \pm 0,69$	$3,78 \pm 0,67$
8 kHz	$3,75 \pm 0,69$	$0,00 \pm 0,00$	$0,69 \pm 0,62$
8 kHz Coded	$3,78 \pm 0,67$	$0,69 \pm 0,62$	$0,00 \pm 0,00$

5.3 Subjective Test

Although the objective analysis provides a quality evaluation of the system, it is not capable to evaluate perceptual features of the proposed system. Hence, a subjective test was performed. In that test, 10 sentences (presented in Tab 3, with the corresponding english translation) were synthesized using both the 8 kHz and the 8 kHz coded databases. Each pair of sentence was reproduced, without repetition, to a group of 14 people, and after each pair of sentence, each tester had to decide which sentence achieved better quality. For better

Table 3: *The Portuguese sentences used for the subjective test.*

Portuguese	English
Ela saía discretamente	She went out discretely
Queremos discutir o orçamento	We want to discuss the budget
Hoje dormirei bem	Today I'll sleep well
Procurei Maria na copa	I looked for Maria in the kitchen
O inspetor fez vistoria completa	The inspector did a complete inspection
Desculpe se magoei o velho	Sorry if I hurt the old man
Ela tem muita fome	She is very hungry
Depois do almoço te encontro	I'll meet you after lunch
A pesca é proibida neste lago	Fishing in forbidden in this lake
Temos muito orgulho da nossa gente	We are very proud of our people

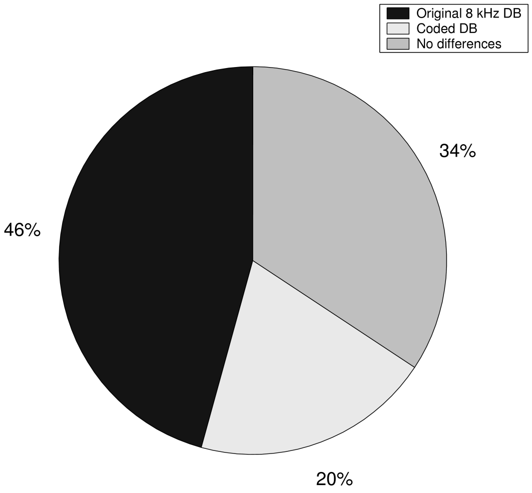

Figure 5: *Comparison between the 8kHz and the 8kHz coded databases.*

evaluation, the testers did not know which synthesizer version was used to produce each sentence. Results of such test can be observed in Fig. 5. One can note that, in 54% of the cases, the coded system was considered equal or better in quality to the 8 kHz version without coding, showing, once more, that the quality loss due to the coding process could be considered as small. However, this result was affected by the fact that the group of testers was composed mainly by Brazilians (only 2 Portuguese people), who were not fully used to the European Portuguese accent, which tends to influence negatively the obtained results.

6. CONCLUSIONS

This paper presented the prototype design of a portable speech synthesizer system for Portuguese language which was implemented using DSP technology. In order to achieve small memory requirements, the database was optimized, resampled and then coded using a CELP coder. Processing speed measurements have shown that the real-time requirements are sustained, and the quality test have shown that the coding process causes minimal impact in the synthesis quality.

The main design focus was on cost reduction of the final product. Better quality can be achieved by the expense of increasing the amount of memory used. Further studies are expected to be performed in order to improve the quality of the synthesis without increasing the memory requirements. Future studies will also be made in order to associate this system to a real-time, speech recognition system being developed [16], in order to produce a final system able to support synthesis and recognition tasks.

Acknowledgments

The authors would like to thank Maria J. Barros (FEUP) and João P. Teixeira (FEUP), for their support during this work, and also CNPq, CAPES, FAPERJ (Brazil) and GRICES (Portugal), for their financial support.

REFERENCES

[1] Eurostat, "Europa in zahlen," 1995.

[2] P. Roe, "Bridging the gap? access to telecommunciations for all people," www.tiresias.org/phoneability/bridging_the_gap, November 2001.

[3] T. Dutoit, *An Introduction to Text-to-Speech Synthesis*. Kluwer Academic, 1999, vol. 3.

[4] H. Mixdorff, "Intonation patterns of German - quantitative analysis and synthesis of *F0* countours," Ph.D. dissertation, Technische Universität Dresden, 1998.

[5] P. Kroon and K. Swaminathan, "A high-quality multirate real-time celp coder," *IEEE Journal on Selected Areas in Communications*, vol. 10, no. 5, pp. 850–857, June 1992.

[6] J. G. Ackenhusen, *Real-Time Signal Processing*. Prentice Hall, 1999.

[7] *ADSP-21160: SHARC DSP Hardware Reference*, 2nd ed., Analog Devices, May 2002.

[8] *Visual DSP++ 3.0 Manual: C/C++ Compiler and Library Manual For SHARC DSPs*, 4th ed., Analog Devices, January 2003.

[9] K. Hwang and F. A. Briggs, *Computer Architecture and Parallel Processing*, 5th ed. McGraw-Hill, 1989.

[10] *ADSP-21160 EZ-KIT Lite: Evaluation System Manual*, 3rd ed., Analog Devices, January 2003.

[11] G. Peacock, "Interfacing the serial / RS-232 port," www.beyondlogic.org/serial/serial.htm, August 2001.

[12] J. P. Teixeira, D. Freitas, D. Braga, M. J. Barros, and V. Latsch, "Phonetic events from the labeling the european portuguese database for speech synthesis, FEUP/IPB-DB," *Eurospeech*, pp. 1707–1710, September 2001.

[13] F. C. C. B. o. Diniz, "Implementation of a real-time CELP voice coding system (in Portuguese)," Federal University of Rio de Janeiro, Tech. Rep., May 2003.

[14] R. S. Maia, "CELP coding and spectral analysis of speech signals (in Portuguese)," M.Sc. thesis, Federal University of Rio de Janeiro, 2000.

[15] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*. Prentice Hall, 1993.

[16] M. R. Vassali, J. M. Seixas, and C. Espain, "Real-time speech recognition system for Portuguese language based on DSP technology," *IEEE South-American Workshop on Circuits and Systems*, 2000.