Abstract — Nowadays, unit selection based text-to-speech technology is the mainstream approach for near natural speech synthesis systems. However, this is achieved at the expense of raised requirements in terms of computational resources. This work describes design and implementation approaches for the efficient integration of this technology in computational environments with limited resources, such as mobile devices, with no considerable speech quality degradation. In particular, the issues of database reduction, acoustic inventory compression and runtime computational load minimization are mainly addressed in this paper. Both objective and subjective assessments confirm the effectiveness of these approaches in terms of constructing a general purpose embedded unit selection TTS system and reducing the computational requirements while maintaining high speech quality.

Index Terms — Embedded Speech Synthesis, Unit Selection, Text-to-Speech, Mobile Devices, Mobile Phones.

I. INTRODUCTION

Text-to-Speech (TTS) technology deals with the production of synthetic voice output using textual information, thus serving as a more natural interface in human machine interaction. In order for TTS technology to be widely adopted, near-natural voice output quality has to be achieved. Over the last years, significant research progress in the field has contributed towards this goal. Nowadays, unit selection concatenative speech synthesis technology has become the dominant approach for building naturally sounding text-to-speech systems. This technique relies on runtime selection and compilation of speech units from a large speech database [1]. The speech database usually consists of a sufficient corpus of appropriately selected naturally spoken utterances, carefully annotated to the unit level. In most cases the speech units are phonemes or diphones. Each utterance comes from a text corpus designed to cover as many units as possible in different phonetic and prosodic contexts. The resulting repository of speech units may have little or great redundancy, on which speech variability and overall quality is significantly depended [1]. The drastic improvement in quality of synthetic speech, namely naturalness and intelligibility, over the years has led to the adoption of TTS as a mainstream technology. As a result, TTS technology is now employed in a wide range of applications, spanning from assistive technology and education, to telecommunications and entertainment [2]-[5]. However, this usually comes at the cost of large resource repositories and increased processing power, limiting the applications in desktop or server-based environments and therefore prohibiting its use on embedded or portable devices. Yet, the ever increasing demand for enhanced consumer applications and the extensive use of portable devices such as mobile phones or personal digital assistants (PDA) in everyday’s life, have intensified the need to efficiently adapt TTS technology in environments with limited computational resources. For example, application areas such as assistive aids and tools, speech-to-speech translation, robotics, mobile phones, household devices, navigation and personal guidance gadgets, can largely benefit from the more natural and intuitive means of human computer interaction (HCI) offered by speech [6]-[10].

In order to address the challenge of developing a high quality TTS system for embedded devices, several approaches have been considered, regarding not only the adaptation of unit selection TTS but also the underlying technology itself. In reference to the technology, there are several different approaches for building low footprint TTS systems. Solutions include diphone-based TTS where the speech stimuli are comprised of only one instance of every diphone, and parametric TTS, such as Formant-based or HMM-based [1], [11]-[13]. Even though both diphone-based and formant-based TTS systems are well suited for low resource TTS, they suffer from degraded speech quality which is often not acceptable for mainstream consumer products [11]. On the other hand, recent results have shown that statistical parametric speech synthesis based on Hidden Markov Models (HMM-based TTS) can deliver high quality synthetic speech with reduced demands for computational resources, and hence can efficiently be adopted for portable devices [12]. Nevertheless, unit selection TTS is still the dominant approach for high quality speech synthesis, and therefore its efficient adaption to environments with reduced computational resources is of great interest so long as the speech quality is preserved. Recent research in this field has been mainly concentrated on optimizing several aspects of the speech synthesis procedure. These aspects include speech database reduction and compression for unlimited domain speech synthesis, speech signal parameterization and runtime synthesis optimization [11]-[16]. Approaches for limited

1 This work was supported in part by E.U. and National funding. S. Karabetsos, P. Tsiakoulis, A. Chalamandaris and S. Raptis are affiliated with the Institute for Language and Speech Processing (ILSP) / R.C. Athena, Department of Voice & Sound Technology, Artemidos 6 & Epidavrou, Marousi, GR 15125, Athens, Greece (e-mail: {sotoskar, ptsiak, achalam, spy}@ilsp.gr).
domain speech synthesis (that is, domain-specific speech databases e.g., weather forecast) have also been proposed [17]. In summary, the most important issues of embedded unit selection TTS, relate to the objective of efficiently balancing the requirements regarding the computational load and the available resources together with delivering high quality speech.

In this paper, we describe design and implementation approaches for integrating generic domain unit selection TTS technology in environments with limited resources and computational capabilities such as mobile phones. More particularly, emphasis is given in three main issues. Firstly, we detail on the development of a specific methodology for the construction of a speech database for embedded devices, based on the existing databases utilized in the server-based or deskto-based versions of the corresponding TTS systems. The method relies on statistical analysis on the data derived from the unit selection stage on a large text corpus and employs, not only the selection frequencies, but also the unit selection scores of the units, leading to enhanced coverage and reduced redundancy. Secondly, we focus on the compression and coding of the speech units, aiming to efficient storage and retrieval, as well as to final signal quality during synthesis runtime. For this purpose, a code excited linear prediction (CELP) based approach is utilized and adapted to the particular needs of the TTS system. Finally, the last aspect we cover is the minimization of the computational requirements inherent to the unit selection module. The latter module performs a computationally demanding search to determine the optimal sequence among candidate speech units. To reduce the runtime computational load, we adopted a vector quantization (VQ) approach for the spectral join feature vectors of the same speech units and the offline computation of the corresponding distances. Evaluation results made clear that the aforementioned processes perform efficiently, leading successfully to a commercial TTS system for mobile phones of very high quality.

The rest of this paper is organized as follows. In section II, the unit selection concatenative speech synthesis technology is briefly reviewed and a description of the embedded TTS system architecture is given, highlighting its core modules. Section III provides details on the proposed design and implementation for database reduction and compression, as well as the minimization of the computational requirements related to the unit selection module. In section IV, both subjective and objective evaluation results are presented regarding the assessment of the followed techniques. Finally, a summary and some conclusive remarks are given in section V.

II. Embedded Text to Speech System Architecture

The general architectural diagram of a corpus-based TTS system is depicted in Fig. 1. There are two main components that comprise such a system, namely the Natural Language Processing (NLP) and the Digital Signal Processing (DSP). This diagram is valid for every data driven (that is, corpus-based) TTS system, regardless of the underlying technology (e.g., unit selection or parametric). The NLP component accounts for every aspect of the linguistic processing of the input text, whereas the DSP component accounts for speech signal manipulation and generation. For a unit selection TTS, besides the speech units (usually diphones) the speech database contains all the necessary data for the unit selection stage of the synthesis [1], [5].

In particular, the NLP component is mainly responsible for parsing, analyzing and transforming the input text into an intermediate symbolic format, appropriate to feed the DSP component. Furthermore, it provides all the essential information regarding prosody, that is, pitch contour, durations and intensity. It is usually composed of a text parser, a morpho-syntactic analyzer, a text normalizer, a letter-to-sound module and a prosody generator. All these sub-components are necessary for the disambiguation and proper expansion of all abbreviations and acronyms, for the correct word pronunciation, and also for the detection and application of the rich set of distinctive features of the signal, closely related to prosody.

The DSP component comprises of all the essential modules for the proper manipulation of the speech signal, that is, prosodic analysis and modification, speech signal representation and generation. Among various algorithms for speech manipulation, Time Domain Pitch Synchronous Overlap Add (TD-PSOLA), Harmonic plus Noise (HNM), Linear Prediction based (LPC-based) and Multiband Resynthesis Overlap Add (MBROLA) are the ones that are mostly employed [1]. The DSP component also includes the unit selection module, which performs the selection of the speech units from the speech database using explicit matching criteria. More details about this module are given later in this section.

It becomes apparent that a full scale deployment of a unit selection TTS system is either infeasible or impractical in embedded environments. The system architecture that we
The DSP component comprises of the unit selection module and the signal manipulation module, in our case one based on TD-PSOLA. The speech database of the embedded TTS system uses a sampling frequency of 16 KHz. The database includes diphones as principal speech units, derived from the recordings of a Greek female professional speaker.

The unit selection module is considered to be one of the most important components in a corpus-based unit selection concatenative speech synthesis system. It provides a mechanism to automatically select the optimal sequence of database units that produce the final speech output, the quality of which depends on its efficiency. The criterion for optimizing is the minimization of a total cost function which is defined by two partial cost functions, namely the target cost and the concatenation cost function [1], [5].

The target cost function measures the similarity of an applicant unit with its predicted specifications (from NLP) and is defined as,

\[ C'(t_i, u_i) = \sum_{j=1}^{p} w_j \cdot C_j'(t_i, u_i) \]  

where, \( u_i = \{ u_1, u_2, \ldots, u_n \} \) are the candidate (sequence) units, \( t_i = \{ t_1, t_2, \ldots, t_n \} \) are the target (sequence) units, \( C_j'(t_i, u_i) \) is a partial target cost, \( p \) is the dimension of the target feature vector and \( w_j \) is a weighting factor for every partial target cost. The target feature vector typically consists of spectral similarity measures, pitch similarity measures, context similarity etc. Hence, the total cost is defined as,

\[ C(u_{i-1}, u_i) = \sum_{j=1}^{q} w_j \cdot C_j(u_{i-1}, u_i) \]  

or based on (1) and (2) it can be written as,

\[ C(t_i, u_i) = \sum_{j=1}^{p} W_j \cdot C_j(t_i, u_i) + \sum_{i=2}^{n} W' \cdot C(u_{i-1}, u_i) \]  

where, \( W_j \) and \( W' \) are the weights that denote the significance of the target cost and the join cost, respectively. The goal of the unit selection module is to perform a (computationally demanding) search so as to find the speech unit sequence which minimizes the total cost, hence to specify,

\[ w^* = \min_{w} C(t_i, u_i) \]  

The selection of the optimal speech unit sequence incorporates a thorough search (usually a Viterbi search) which involves comparisons and calculations of similarity measures between all available units, often employing heuristics to guide and/or limit the search [1], [5].

III. ADAPTATION TECHNIQUES FOR EMBEDDED DEVICES

For the efficient adaptation and integration of unit selection TTS technology in embedded environments, a balance must be struck between the conflicting demands of minimizing the computational load while preserving a high-quality speech output. In most of the cases, computational load increases in...
A Speech Database Reduction

The method relies on statistical data produced by the full scale unit selection process on a large text corpus, initially proposed in [16]. It utilizes the selection frequency, as well as the actual score of each speech unit (diphone). As outlined in [15] the strategies usually fall in two categories: the top-down and the bottom-up approaches. According to the top-down approach, the unit repository is investigated for the reduction process and a clustering process is performed, based on prosodic and phonetic properties. By doing so, the search space of the unit selection algorithm is reduced as each target unit is searched within the corresponding cluster. On the other hand, the bottom-up approach is purely a data driven technique since it focuses on the statistical behaviour of the unit selection algorithm. The output of the unit selection stage is statistically analysed in order to reduce the unit repository. The statistical data is collected from the synthesis of a large text corpus, where the selection frequency of each unit is usually calculated and is used in the reduction process. For example, in [15] the removal of the less frequent units is proposed. A possible weakness of using only the selection frequency is that it does not help avoid redundancy. For example two very similar units that are alternatively selected frequency is that it does not help avoid redundancy. For example, in [15] the removal of the less frequent units is proposed. A possible weakness of using only the selection frequency is that it does not help avoid redundancy. For example two very similar units that are alternatively selected frequency is that it does not help avoid redundancy. For example two very similar units that are alternatively selected frequency is that it does not help avoid redundancy. For example two very similar units that are alternatively selected generally follow two categories: the top-down and the bottom-up approaches. According to the top-down approach, the unit repository is investigated for the reduction process and a clustering process is performed, based on prosodic and phonetic properties. By doing so, the search space of the unit selection algorithm is reduced as each target unit is searched within the corresponding cluster. On the other hand, the bottom-up approach is purely a data driven technique since it focuses on the statistical behaviour of the unit selection algorithm. The output of the unit selection stage is statistically analysed in order to reduce the unit repository. The statistical data is collected from the synthesis of a large text corpus, where the selection frequency of each unit is usually calculated and is used in the reduction process. For example, in [15] the removal of the less frequent units is proposed. A possible weakness of using only the selection frequency is that it does not help avoid redundancy. For example two very similar units that are alternatively selected generally follow two categories: the top-down and the bottom-up approaches. According to the top-down approach, the unit repository is investigated for the reduction process and a clustering process is performed, based on prosodic and phonetic properties. By doing so, the search space of the unit selection algorithm is reduced as each target unit is searched within the corresponding cluster. On the other hand, the bottom-up approach is purely a data driven technique since it focuses on the statistical behaviour of the unit selection algorithm. The output of the unit selection stage is statistically analysed in order to reduce the unit repository. The statistical data is collected from the synthesis of a large text corpus, where the selection frequency of each unit is usually calculated and is used in the reduction process. For example, in [15] the removal of the less frequent units is proposed. A possible weakness of using only the selection frequency is that it does not help avoid redundancy. For example two very similar units that are alternatively selected generally follow two categories: the top-down and the bottom-up approaches. According to the top-down approach, the unit repository is investigated for the reduction process and a clustering process is performed, based on prosodic and phonetic properties. By doing so, the search space of the unit selection algorithm is reduced as each target unit is searched within the corresponding cluster. On the other hand, the bottom-up approach is purely a data driven technique since it focuses on the statistical behaviour of the unit selection algorithm. The output of the unit selection stage is statistically analysed in order to reduce the unit repository. The statistical data is collected from the synthesis of a large text corpus, where the selection frequency of each unit is usually calculated and is used in the reduction process. For example, in [15] the removal of the less frequent units is proposed. A possible weakness of using only the selection frequency is that it does not help avoid redundancy. For example two very similar units that are alternatively selected generally follow two categories: the top-down and the bottom-up approaches. According to the top-down approach, the unit repository is investigated for the reduction process and a clustering process is performed, based on prosodic and phonetic properties. By doing so, the search space of the unit selection algorithm is reduced as each target unit is searched within the corresponding cluster. On the other hand, the bottom-up approach is purely a data driven technique since it focuses on the statistical behaviour of the unit selection algorithm. The output of the unit selection stage is statistically analysed in order to reduce the unit repository. The statistical data is collected from the synthesis of a large text corpus, where the selection frequency of each unit is usually calculated and is used in the reduction process. For example, in [15] the removal of the less frequent units is proposed. A possible weakness of using only the selection frequency is that it does not help avoid redundancy. For example two very similar units that are alternatively selected generally follow two categories: the top-down and the bottom-up approaches. According to the top-down approach, the unit repository is investigated for the reduction process and a clustering process is performed, based on prosodic and phonetic properties. By doing so, the search space of the unit selection algorithm is reduced as each target unit is searched within the corresponding cluster. On the other hand, the bottom-up approach is purely a data driven technique since it focuses on the statistical behaviour of the unit selection algorithm. The output of the unit selection stage is statistically analysed in order to reduce the unit repository. The statistical data is collected from the synthesis of a large text corpus, where the selection frequency of each unit is usually calculated and is used in the reduction process. For example, in [15] the removal of the less frequent units is proposed. A possible weakness of using only the selection frequency is that it does not help avoid redundancy. For example two very similar units that are alternatively selected generally follow two categories: the top-down and the bottom-up approaches. According to the top-down approach, the unit repository is investigated for the reduction process and a clustering process is performed, based on prosodic and phonetic properties. By doing so, the search space of the unit selection algorithm is reduced as each target unit is searched within the corresponding cluster. On the other hand, the bottom-up approach is purely a data driven technique since it focuses on the statistical behaviour of the unit selection algorithm. The output of the unit selection stage is statistically analysed in order to reduce the unit repository. The statistical data is collected from the synthesis of a large text corpus, where the selection frequency of each unit is usually calculated and is used in the reduction process. For example, in [15] the removal of the less frequent units is proposed. A possible weakness of using only the selection frequency is that it does not help avoid redundancy. For example two very similar units that are alternatively selected generally follow two categories: the top-down and the bottom-up approaches. According to the top-down approach, the unit repository is investigated for the reduction process and a clustering process is performed, based on prosodic and phonetic properties. By doing so, the search space of the unit selection algorithm is reduced as each target unit is searched within the corresponding cluster. On the other hand, the bottom-up approach is purely a data driven technique since it focuses on the statistical behaviour of the unit selection algorithm. The output of the unit selection stage is statistically analysed in order to reduce the unit repository. The statistical data is collected from the synthesis of a large text corpus, where the selection frequency of each unit is usually calculated and is used in the reduction process. For example, in [15] the removal of the less frequent units is proposed. A possible weakness of using only the selection frequency is that it does not help avoid redundancy. For example two very similar units that are alternatively selected generally follow two categories: the top-down and the bottom-up approaches. According to the top-down approach, the unit repository is investigated for the reduction process and a clustering process is performed, based on prosodic and phonetic properties. By doing so, the search space of the unit selection algorithm is reduced as each target unit is searched within the corresponding cluster. On the other hand, the bottom-up approach is purely a data driven technique since it focuses on the statistical behaviour of the unit selection algorithm. The output of the unit selection stage is statistically analysed in order to reduce the unit repository. The statistical data is collected from the synthesis of a large text corpus, where the selection frequency of each unit is usually calculated and is used in the reduction process. For example, in [15] the removal of the less frequent units is proposed. A possible weakness of using only the selection frequency is that it does not help avoid redundancy. For example two very similar units that are alternatively selected generally follow two categories: the top-down and the bottom-up approaches. According to the top-down approach, the unit repository is investigated for the reduction process and a clustering process is performed, based on prosodic and phonetic properties. By doing so, the search space of the unit selection algorithm is reduced as each target unit is searched within the corresponding cluster. On the other hand, the bottom-up approach is purely a data driven technique since it focuses on the statistical behaviour of the unit selection algorithm. The output of the unit selection stage is statistically analysed in order to reduce the unit repository. The statistical data is collected from the synthesis of a large text corpus, where the selection frequency of each unit is usually calculated and is used in the reduction process. For example, in [15] the removal of the less frequent units is proposed. A possible weakness of using only the selection frequency is that it does not help avoid redundancy. For example two very similar units that are alternatively selected generally follow two categories: the top-down and the bottom-up approaches. According to the top-down approach, the unit repository is investigated for the reduction process and a clustering process is performed, based on prosodic and phonetic properties. By doing so, the search space of the unit selection algorithm is reduced as each target unit is searched within the corresponding cluster. On the other hand, the bottom-up approach is purely a data driven technique since it focuses on the statistical behaviour of the unit selection algorithm. The output of the unit selection stage is statistically analysed in order to reduce the unit repository.
rate in all diphones. An explicit function is utilized to determine the number of units:

\[ M = \min(M_{\text{max}}, \max(M_{\text{min}}, \log_b(K))) \]  

where, the parameters \( M_{\text{max}} \) and \( M_{\text{min}} \) (\( M_{\text{max}} > M_{\text{min}} \)) explicitly define the maximum and minimum number of instance units per diphone, while parameter \( b \) determines a logarithmic reduction rate distribution among diphones. Evaluation results on this technique are presented in section IV.

B. Speech Database Compression and Coding

Speech database compression is considered as a vital problem in embedded unit selection speech synthesis since it facilitates for better and efficient adaptation of the technology in this domain [11]. The problem is not different than conventional speech coding although there are issues that are specific to TTS technology. The compression technique should not only ensure compression efficiency but also avoid introducing perceived signal degradations. Furthermore, it should provide random access capability and fast decoding. On the other hand, encoding complexity is not an issue since it is performed offline.

In this work, we have adopted Code Excited Linear Prediction (CELP) as the speech database compression technique in view of the fact that it is a well established and widely deployed coding scheme, capable of producing adequate speech quality [20]. The adaptation of CELP for the purpose of embedded unit selection TTS is depicted in Fig. 3. To cope with random access capability, every speech unit (diphone) is separately compressed and encoded. Hence, the speech database consists of CELP parameters representing diphones that are binary encoded for effective database organization. It is important to note that in this approach, neither the time limits nor the pitch marks of the diphones are affected. Also, no perceived spectral degradation occurs. Furthermore, a scalable bit allocation scheme is used for obtaining different compression ratios. In practice, informal listening tests have shown that compression ratios between 7 and 10 could be used.

At the synthesis stage, only the selected (best path) diphones are decompressed for TD-PSOLA, thus eliminating any overhead. The CELP decoder is implemented using fixed point arithmetic for performance optimization.

C. Reduction of the Computational Requirements of the Unit Selection module

One of the most demanding tasks during synthesis runtime is the unit selection. The unit selection process involves dynamically searching and deciding on the “optimal” unit sequence over a lattice of available units. The performance of the unit selection algorithm is vital since it heavily determines the response time of the system. Today’s speech databases with sizes ranging from a few MB to several GB, and incorporating hundreds of instances per speech unit, pose increased demands on CPU power. In large scale systems, such as in desktop- or server-based TTS, this is compensated, without loss in quality, by the available processing and storage power complemented by both heuristics (e.g. pruning) and clustering over similar units techniques [21][22]. However, in the case of embedded TTS, these techniques are not appropriate, since they rely on the plurality of remaining units. The latter assumption is not applicable for embedded unit selection where the databases used are already reduced and, therefore, the search space has already been sufficiently limited.

In this work, a vector quantization (clustering) approach is adopted in order to achieve lower computational and storage costs, for the purpose of spectral join cost calculation, since it is the most expensive task in the unit selection process. The approach is based on a within-type clustering of the spectral join feature vectors of the speech units (e.g. the clustering of the feature vectors of the same phoneme) and the offline computation of distances between the centers of each cluster. The approach is motivated by the idea presented in [22], although it differs significantly since it maintains the available search space. A similar approach has been mentioned in [23], but deals only with the case of a large scale TTS system and does not put focus on the particular characteristics for the deployment in embedded devices.

An example of the unit selection algorithm is illustrated in Fig. 4, where the synthesis of the utterance “ela” is depicted which is the Greek word for “come”. We consider the use of diphones so the speech units are \{/e/\_e/la/a_/\} each having a total of \( N, M, K \) and \( J \) number of instances in the
speech database respectively. Therefore, the corresponding lattice involves the computation of $N^M+MK+KJ$ concatenation cost, termed as $C^2$. It is important to notice that, among these metrics, the spectral join cost is one of the most predominant factors for high computational load and slow response time at synthesis runtime since unit selection is performed per utterance. It employs the retrieval of spectral feature vectors for each unit and the calculation of a distance which serves as the spectral join cost.

Specifically, if we let $P = \{d\}$ be the phoneme set having $|P| = N$ elements and $D = \{pq; p, q \in P$ and $pq$ is valid) be the diphone set having $|D| = M$ elements then the following statement is true, $N^2 \geq |D| = M - O(N^2)$. Furthermore, we consider the unit selection speech database as a repository of instances of speech units (diphones) that forms the set $R = \{u_{pq}^k: \text{the } k_{th} \text{ instance of } pq \in D\}$. If we let $K$ be the mean number of instances per speech then, a specific phoneme may form $N$ diphones as the left phoneme of a diphone and $N$ diphones as the right phoneme of a diphone, each of them having $K$ instances respectively. For example, the phoneme /a/ forms the diphone sets /aX/ and /Xa/ where $X \in P$ (e.g. can be any of $N$ phonemes). Since diphones for the type /Xa/ concatenate only with diphones of /aX/ and since there are $K$ instances for each of them, the possible joins for a phoneme (in the diphone set) is in the order of $N^2K^2$. Thus, since the phoneme set has $N$ elements, the order of (total) possible joins in $R$ is $O(N^3K^2)$. Moreover, for each $u \in R$ we need two pairs of (concatenation) feature vectors, that is, one for the left phoneme (left join $v_L(u)$) and one for the right phoneme (right join $v_R(u)$) and a distance measure, $d(v_L(u), v_R(u))$ or for simplicity $d(v(u), v(u))$, between two units to be used as the join cost. Therefore, the options for storage and computation are, either to store the feature vectors $v_L(u)$, $v_R(u)$ for every $u \in R$ and evaluate the distance $d$ at runtime, or compute every distance offline and store them (in tabular format) for every possible join in $R$. The former entails a high runtime computational cost while the latter a high storage cost.

In order to reduce the runtime computational and storage cost, a within-type clustering of spectral join feature vectors of speech units is proposed, as well as the offline computation and storage of distances between the centres of each cluster. More specifically, an offline clustering of all spectral feature vectors of the same phoneme is performed, followed by the computation of the distances between the centres (or representatives) of each cluster. Thus, the cluster distances are pre-computed and are used instead of the true distances for joining segments. This technique offers a low runtime computational and storage cost since it reduces the required number of concatenation costs calculations. However, this is achieved at the expense of possible degradation of the resolution of the spectral join cost which might affect the synthetic speech quality. Experimental evaluation shows that no significant degradation in quality is observed. The algorithmic description of the technique is illustrated in table II.

As a result, if $C$ is the cluster size per phoneme and the concatenation occurs per diphone, the number of total possible cluster joins is in the order of $O(NC^2)$ which is $O(NC^2) < O(N^2K^2)$ since $C$ can be chosen to be adequately small. Additionally, it is important to notice that the number of possible joins per diphone is $C^2$ for the case of clustering and $N^2K^2$ without clustering. Hence, the statement $C^2 < N^2K^2$ is true even for the case of embedded devices as long as the number of instances per diphone is sufficient and again if $C$ is adequately small. For example, the developed Text-to-Speech system for mobile phones utilizes a phoneme set for the Greek language that has $N = 34$ elements and the number of instances per diphone is at least 10. Hence, if the cluster size is $C = 32$ the above criterion is met.

<table>
<thead>
<tr>
<th><strong>Algorithm for Clustering Spectral Join Costs (Offline)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. ∀ $p \in P$ do steps 2 to 5</td>
</tr>
<tr>
<td>2. Find all instances of speech units that have $p$ as left or right phoneme i.e., find $R_{left}^p = {u^p: u \in R$ and $p=l}$ and $R_{right}^p = {u^p: u \in R$ and $p=r}$</td>
</tr>
<tr>
<td>3. Perform clustering of ${v_L(u): u \in R_{left}^p} \cup {v_R(u): u \in R_{right}^p}$ in $C$ clusters with centres $c_i, i = 1...C$</td>
</tr>
<tr>
<td>4. Compute $M(i, j) = d(c_i, c_j), i, j = 1...C$</td>
</tr>
<tr>
<td>5. Store $M(i, j)$ and the two cluster indexes per unit instance</td>
</tr>
</tbody>
</table>

At synthesis runtime the distance between pairs of diphones is retrieved and calculated as $M(a, b)$ instead of $d(v(u)^p), v(u)^q))$ where, $a, b$ are the corresponding cluster indexes that each phoneme of every diphone belongs to. The method does not reduce the available search space since it is clear that no clustering on the speech units themselves can be performed since the speech database is already reduced. Instead, the search space is kept the same while clustering is carried out for the features that constitute the spectral join cost. While this may lead to resolution degradation, it is assumed that since within-cluster costs have small differences between them together with implicit compensation due to other sub-costs, the reduction of the cost resolution can be tolerated.

**IV. Evaluation and Results**

The techniques addressed in this work are assessed using both objective and subjective criteria. For subjective evaluation, the most common approach in assessing the quality of TTS systems is through listening tests where a group of people is asked to express their opinion regarding the TTS quality namely, naturalness and intelligibility. The results, usually expressed in terms of mean opinion scores (MOS), reflect rather accurately the perceived quality of a
The experiments were carried out on a database of a Greek female speaker, which consists of a total of 1291 annotated utterances from a phonetically balanced corpus of modern Greek language. The resulting complete database has a total of 1098 unique diphones and contains about 115K instances. The final (embedded) database has approximately 11K diphones. The total resources are approximately 6MB and the memory footprint of the TTS is less than 2MB. There are no separate evaluation results for CELP encoding and decoding since this process is implicit in the following evaluation experiments. Furthermore, the mobile phone utilized in the experimental evaluation had a CPU of 220MHz.

A. Speech Database Reduction Evaluation

After benchmarking with various target embedded devices, we reached to the conclusion that reasonably high reduction rates, up to 95%, are both possible and necessary for the TTS system to run efficiently. At such high reduction rates, a degradation of output speech quality is almost inevitable, especially as far as variability in the speech is concerned. A large text corpus of no specific domain was collected for testing purposes. Hence, a total of about 12.5K sentences covering about 1.5M diphone instances were utilized. A 95% segment of the corpus was used to collect statistical data from the unit selection synthesis algorithm, and the rest was used for the objective evaluation process. The most obvious method for comparison is the “select most frequent units” method [15]. In order to have meaningful results we use the same number of units per diphone M across methods. Hereafter we refer to our method as PF and to the most frequent selection as PS. As shown in [16], both methods fully overlap for extreme reduction cases. For the evaluation of the database reduction technique, objective metrics derived from statistical parameters describing the behavior of the unit selection algorithm, are utilized. The commonly used statistics are, the mean values of target, join and total costs over the best path units. In addition, another set of objective metrics, also derived from the statistics of the unit selection algorithm, are introduced. In particular, the maximum target, join and total cost is considered. By taking into account the maximum cost per utterance, we try to identify glitches in the synthetic speech, since places of high cost are potential prosodic, spectral or other types of discontinuities. Such cases are usually avoided with the use of a large database, but this may be inevitable at high reduction rates. All the above statistical metrics are calculated per utterance and averaged over the whole test corpus. The comparison results of the objective evaluation of PF and PS are illustrated in Fig. 5. As a reference point, the corresponding measures for the complete database system are \( \{\text{total, join, target}\}_{\text{mean}} = \{0.15, 0.07, 0.07\} \) and \( \{\text{total, join, target}\}_{\text{max}} = \{0.50, 0.27, 0.34\} \). Although PS performs slightly better in terms of mean costs, PF has a far lower average maximum cost per utterance, which becomes more pronounced as the reduction rate increases. This behavior indicates two main presumptions. The PS method produces databases that result in synthetic utterances with good scores if averaged, but also having units with poor scores. On the other hand, PF produced databases resulting in utterances with far better target cost at the cost of a slightly higher join cost.

<table>
<thead>
<tr>
<th>MOS Results of the Database Reduction Method</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MOS</strong></td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td><strong>PF</strong></td>
</tr>
<tr>
<td><strong>PS</strong></td>
</tr>
</tbody>
</table>

In order to subjectively assess the method, listening tests with 35 short sentences (2 to 16 words long) selected from the test corpus were conducted. The sentences were synthesized with databases produced by PF and PS with a reduction rate of 93%. A group of 15 listeners, speech experts and listeners with no experience in synthetic speech were asked to evaluate each pair of sentences, presented in a shuffled order each time. The results are summarized in table III, where the mean opinion score (MOS) is shown together with the objective metrics (total, join, target costs). The results show that PF produces better synthetic speech than PS. Also, there is an agreement of the MOS values and the averaged maximum total cost per utterance. This seems to verify the initial hypothesis that PS could result in redundant units in terms of
target features, by selecting more similar units and leaving at the same time no room for other units to cover other less frequent but equally important cases met in general purpose TTS systems.

B. Clustered Spectral Join Cost Evaluation

The acoustic representation that is used is Mel-Frequency Cepstral Coefficients (MFCC) and the Euclidean distance between MFCC vectors is used as a spectral join cost. Thus, the feature vectors to be clustered are the MFCC vectors for every phoneme of every diphone. For all the experiments in this work, the number of clusters per phoneme is set to \( C = 32 \) and clustering was performed using the \( k\)-means algorithm utilizing the Euclidean distance measure as a classification metric among the feature vectors.

In order to evaluate the performance of the proposed technique a comparison between two versions of the unit selection algorithm namely, with \( C_{US} \) – clustered join cost unit selection) and without \( F_{US} \) – full unit selection) clustering, has been implemented for spectral join cost calculation. A total of 52 sentences were synthesized using both versions and the averaged times concerning the benchmarks of the processing time of the unit selection module were measured. The results are summarized in table IV. Obviously, a significant reduction in the computational load is observed since the proposed technique results in the reduction of the computational time, for the unit selection module on average by a factor of more than three and improves the overall performance of the TTS system on average by a factor of 29\% compared to the \( F_{US} \) version. Additionally, the \( C_{US} \) version accounts only for the 13.1\% of the total processing time. Moreover, a 2.4 real time factor is achieved, on the specific mobile phone. Consequently, the response time of the TTS, which heavily depends on the unit selection module, is greatly reduced achieving a mean value of approximately 0.25sec.

<table>
<thead>
<tr>
<th>TABLE IV</th>
<th>BENCHMARKS ON THE PROCESSING TIME OF CUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit selection speed improvement</td>
<td>&gt; 3.5 times</td>
</tr>
<tr>
<td>Total speed improvement</td>
<td>&gt; 29%</td>
</tr>
<tr>
<td>Mean response time</td>
<td>0.25sec</td>
</tr>
<tr>
<td>Real time factor</td>
<td>&gt; 2.4</td>
</tr>
<tr>
<td>Percentage of total TTS time</td>
<td>( F_{US} ) 32.5%</td>
</tr>
<tr>
<td></td>
<td>( C_{US} ) 13.1%</td>
</tr>
</tbody>
</table>

To assess the effect of the proposed approach in the overall speech quality, we conducted a small scale acoustical experiment. A total of 52 short sentences, having 4 to 16 words, were synthesized (on a mobile phone) using both \( F_{US} \) and \( C_{US} \). The sentences were no-domain specific and were not included in the speech database. A group of 15 listeners, comprised by both speech and non-speech experts, were asked to express their opinion for each sentence in terms of overall quality. Each sentence was presented in pairs \( (F_{US} \) and \( C_{US} \) version) and the subjects could listen to each sentence more than once. The order of each pair was random. The results are summarized in table V. The results depicts that the proposed technique performs slightly better, as far as overall quality is concerned, than its full version counterpart. However, the standard deviation shows that both versions can be considered equivalent. The main conclusion is that \( C_{US} \) results in a synthetic speech quality that is practically indistinguishable compared to the \( F_{US} \) version. On the other hand, the gain in computational time is significant. Additionally, the cost resolution degradation is well balanced since the clustering approach does not reduce the original search space, therefore any possible degradation is compensated by the target cost measures or other sub-costs involved in the concatenation cost calculation. This is in accordance with the experimental findings. Moreover, it is important to notice that the number of clusters causes a trade-off between processing time, storage and degradation in cost resolution. Indeed, as the number of clusters per phoneme increases the storage also increases.

The results are depicting that the TTS system is capable of real time operation with low response time and is sufficiently scaled for embedded environments.

V. CONCLUSIONS

In this paper, we have described the system architecture of a general purpose embedded unit selection TTS system and we have presented efficient techniques that successfully address the challenging problems arising in embedded environments, such as database reduction, database compression, and runtime load minimization. In particular, we have presented an algorithm which leads to small footprint.
speech databases with increased diversity and reduced redundancy. Sufficient compression ratios were achieved by appropriately adapting CELP to the synthesis process. Finally, a vector quantization approach was derived for the spectral joint cost calculation that significantly reduces the computational requirements of the unit selection module. Evaluation results provide clear evidence of substantial improvement in the computational resources exploitation while preserving the overall speech quality in terms of naturalness and intelligibility. All the concepts and approaches proposed in this paper have been employed in the development of a top-quality embedded unit selection TTS system for the Greek language. The system has been successfully adopted as part of a screen-reader solution for mobile phones.

ACKNOWLEDGMENT

The authors would like to thank all the persons involved in the listening tests or contributed to this work.

REFERENCES


Solitris Karabetsos received the M. Eng. degree in Electrical Engineering and Computer Science from the National Technical University of Athens (NTUA), in 2004 and the M.S. degree in Data Communications from Brunel University of London, in 2003. He has also received the BS degree in Electronic Engineering from the Technological and Educational Institution of Athens (TEI of Athens), in 1999. He is currently working towards the Ph.D. degree in Speech Synthesis at NTUA. From 2003, he is with the Institute for Language and Speech Processing (ILSP). He is also with the Technological and Educational Institution of Athens (TEI of Athens), Department of Electronics. His research interests are speech synthesis, signal processing, and telecommunications. He is a member of IEEE.

Pirros Tsiaikoulis received his M. Eng. degree in Electrical Engineering and Computer Science from National Technical University of Athens (NTUA), Athens, Greece. He is currently working towards the Ph.D. degree in Speech Synthesis at the National Technical University of Athens (NTUA). In 2000, he joined the Institute for Language and Speech Processing (ILSP). His research interests include NLP, speech synthesis and speech processing. He is a member of IEEE.

Aimilios Chalamandaris received his M.Eng. degree in Electrical Engineering and Computer Science from the National Technical University of Athens in 2000, and his M. Eng in Telecoms and Signal Processing from Imperial College in 2001. He is a PhD student at NTUA, and works at the Institute for Language and Speech Processing (ILSP), doing research on speech and signal processing. His research interests are NLP, speech synthesis, speech recognition, and signal processing.

Spyros Raptis received his M. Eng. degree in Electrical Engineering and Computer Science in 1994 and his PhD in hybrid computational intelligence for optimization, modeling and decision making in 2001, both from the National Technical University of Athens, Greece. He has been a lecturer at graduate and post-graduate level and has participated in a number of National and European RTD projects on speech technology, computational intelligence, robotics, and multimedia educational applications. He is currently a researcher at the Voice & Sound Technology Department at the Institute for Language and Speech Processing (ILSP) leading the speech synthesis team. His research interests include speech processing and applications, computational intelligence, software agents, hybrid systems and robotics.