A NAME ANNOUNCEMENT ALGORITHM WITH MEMORY SIZE AND COMPUTATIONAL POWER CONSTRAINTS

Ze'ev Roth, Judith Rosenhouse
DSP Group, Technion

ABSTRACT

This paper describes an algorithm for name (surnames and personal names) announcement in American English implemented on DSP Group's SmartCores (registered trade-mark) digital signal processor (dsp) core. The name announcement module is targeted for low cost applications therefor the amount of memory that can be allocated for dictionaries, program code, and runtime parameters is limited. The required response time of 0.5 seconds limits the computations performed in the linguistic analysis phase of each name. The synthesis scheme is limited by the real time capacity of the processor (since this task may be performed in parallel with other real time tasks).

1. INTRODUCTION

Name announcement is the task of producing a signal that is the pronunciation of a name given a text string containing the spelling of that name. Name pronunciation differs from ordinary language to speech, due to the irregularities of names amongst other factors. Two prominent causes for this irregularity, are the varied orthographic conventions, and the diversity of languages of origin of names in American English, which may be attributed to the USA being a "melting pot". It is estimated that there are 1,500,000 different surnames in the US, this very large number of names (and human nature) ensures variety. Languages differ, amongst other things, in their phonetic and stress rules; e.g., Johnson (of English origin) is stressed on the first syllable while Perez (Spanish origin) is stressed on the second syllable.

Name announcement is a specialized text to speech task. An overview of the text to speech (TTS) systems is given in [5]. A detailed description of one of the early TTS systems called MITalk is given in [1]. In [4] an approach to solve the name announcement task is proposed (see also references therein).

We intend to incorporate the Name Announcement system in DSP Group's Telephone Answering Device (TAD) product line. In the TAD it will be hooked to the Caller Identity (CID) module. This module detects and decodes the signal sent by the central office to the subscriber telephone between the first and second ringing. This signal contains the telephone number of the calling party as well as the name of the person to whom the telephone (of the calling party) is registered. The goal of the name announcement system is to announce the name of the calling party while the phone is still ringing.

TAD systems being in the consumer electronics realm, are cost sensitive devices. Therefor there are limitations to the size of the memory that can be used by the announcement algorithm. This influences the design in that the exceptions list (usually found in this type of systems) must be kept relatively short (compared to the large number of names). Another constraint that was taken into account in the design is the algorithm's computational complexity, which is limited by the response time requirement and the processing power of the SmartCores (registered trademark).

The pronunciation strategy can at one extreme Americanize all names, and on the other extreme use the language of origin of each name at the other extreme. In the spirit of [3] we use a strategy that the system should pronounce the name like an educated native American speaker having some knowledge of foreign languages and foreign names [7].

The paper plan is as follows: in section 2, an overview of the system is shown, in section 3 the name announcement algorithm is described. Two modules are elaborated on in sections 4 and 5. Finally, in section 6 a short summary is given.

2. SYSTEM OVERVIEW

A block diagram of the name announcement system is shown in Figure 1. In the figure, the first block (Text to Phoneme) transforms the written text into phonetic representation; we refer to it as the linguistic block. The second block (Phoneme to Signal), given an input phonetic representation, synthesizes a speech signal; we refer to this block as the phonetic block (or the synthesizer). We adopted the phonetic representation units as proposed by [2].

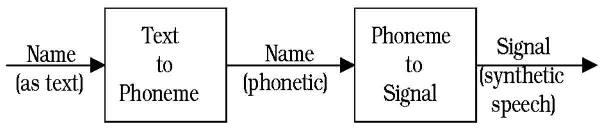

Figure 1: Name Pronunciation Task Block Diagram

3. RULE BASED NAME PRONUNCIATION

A block diagram of our rule based pronunciation method is shown in Figure 2.

- *Normalization* module removes punctuation from the input name, replaces titles by their full spelling (e.g., replace "Mr." by "mister"), removes spaces, and converts the input spelling into lower case alphabetic characters.
- *Exception List Search* module searches if a name is in the exceptions dictionary. The dictionary itself contains

exception names and their transcription. For non-exception names phonological processing is performed.

The following five modules are at the heart of the phonological processing:

- *Language* module decides what is the language of origin of the name; the default being English is replaced by an alternative (single language) only if there is compelling evidence to justify this. All of the following processes depend on this decision.
- *Morphology* module parses the name by identifying prefixes, roots, and suffixes of the name. Currently it is performed only for English origin names.
- *Transcription* module generates the sequence of phonetic segments for pronunciation. It is based on the specific language of origin transcription rules.
- *Syllable Structure* module partitions the name into syllables, the parsing is language of origin dependent.
- *Stress* module assigns stress values to each syllable. This module depends on the language of origin as well.

The language and morphology tasks are complex: both are usually unable to produce a unique answer [4]. Rather than submitting to the generation of multiple possible pronunciations of the same name, corresponding to the different choices of language and morphology, we constrain the language identification to yield a single deterministic result, i.e., the language of origin is taken as English unless compelling evidence exists for applying other than English rules, which is in accordance with the determined pronunciation strategy.

4. Language of Origin Identification

The purpose of this module is to identify the language of origin of the name. The system has several types of operators specially designed to discriminate between the main origin languages. Initially we're dealing with 5 languages of origin: English, Spanish, French, German, and Italian. Should the need arise additional languages of origin will be added on an as-needed basis. Potential languages of origin to complement the present set are: Russian, Polish, Japanese and Chinese. Since we're dealing with a fairly small number of languages the operators do not form a hierarchy. Thus for each input name there is a competition between the set of languages of origin which is the most probable to be the language of origin of that name.

The basic method used by the language identification module is a bi-directional search starting from both ends of the name, searching for prefix and suffix language identification rules. The forward direction of search finds all contiguous strings of morphemes (language identification rules) that match the spelling of the name starting from the beginning (for prefixes and mid rules). The backward direction of search does likewise working back from the end of the name to identify suffixes.

The database of morphemes that form the language identification rules were specifically designed to distinguish/differentiate between the various languages, thus morphemes common to several languages were omitted from this database.

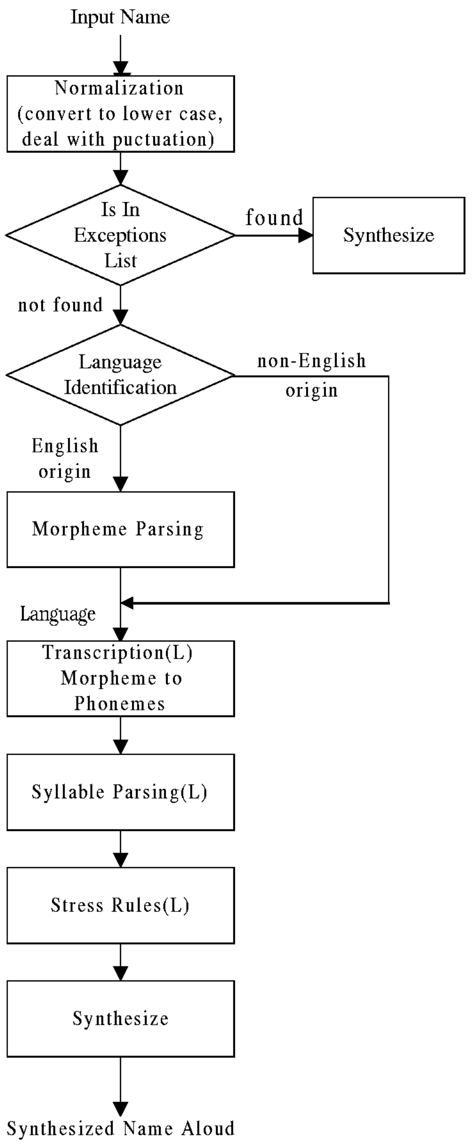

Figure 2: Name Announcement Algorithm

The procedure used by the language module consists of two steps described language score and language selection. At present the system has several hundred language identification rules obtained from [Golding] and books on surnames.

Language Score. The strength of a rule's recommendation is taken to be proportional to the length of its letter sequence, this is a crude heuristic based on [6]:" For all four languages (i.e., English, French, Italian, German –ZR, J.R.), the shorter the syllable the more frequent its occurrence". The strength of each matched rule is added to a score kept for the appropriate language analysis name. The procedure is outlined below.

```
a)  Init
b)  while language identification list end not reached;
        c)  get next rule from list
        d)  determine rule type (prefix, mid or suffix)
        e)  search for rule match in name (according to rule type)
        f)  if match was found
                g)  add search results to database
                h)  add rule length to language score
        i)  end if
j)  end while
```

Language Selection. The selection module takes into account the scores obtained by the Language Score module in its decision making. The scores are sorted in descending order. Then a heuristic decision rule based on empirical results is applied as outlined below.

```
a)  Δ= the difference between the first ranking language and the second ranking language
b)  if Δ >2 or the score of the second ranking language is greater than 2,
        c)  select the first ranking language as the language of origin
d)  otherwise,
        e)  selected language of origin is English
f)  end if
```

Modifications of this basic procedure are done according to test results.

5. Morphology

The system produces a particular morphological analysis of a name by applying a morpheme operator to it. Each morpheme operator has an associated morpheme. Application of the operator at some specified index of the name asserts that the name contains that morpheme starting at that index. Currently morpheme analysis is performed only for names for which the language of origin was decided to be English. This may be modified, as testing will progress. The Morphology module does not utilize the results of the language identification module, since the morphological operators are different from the language identification rules that were specially designed to differentiate between the various languages of origin.

The main incentives for performing morphological analysis are compound names, such as FOOTHILL and BRIDGEPORT. Such names would cause the transcription rules to generate an erroneous transcription due to the interaction of the two morphemes across syllable boundaries. In the first case, the "TH" would be considered a single phoneme, while in the latter case the "E" would not be silent since it is not easily recognized as a final "E". A second possible motivation is that parsing to morphemes can simplify the task for the syllable analysis module, though it takes its toll in memory space since one would need to keep the transcription of all morphemes.

The system has several hundred morpheme-operators, gathered from books on surnames. The morphemes break down into three classes: prefixes, roots, and suffixes.

The basic blocks comprising the Morphology module are:

- Morphology analysis
- Prune analysis results
- Generate morpheme covering
- Filter covering
- Score covering
- Select covering

For short names no morphological processing is done, rather the entire name is transcribed and parsed into syllables.

The Morphology module generates the morphological analyses of the name based on a database of morphemes in a similar manner as the language identification module works. This database is different from the one used by the language identification module since the latter was specifically designed to discriminate between the different languages, thus morphemes that exist in several languages and were omitted there were added to the morphological database.

The Pruning block reduces the analysis results to contain a single prefix morpheme at most, multiple middle morphemes, and a single suffix morpheme at most. Thus at the end of this step we have a set of morphemes that match the name. The Morpheme Covering block generates various coverings of the input name using different subsets of the morphemes obtained by the analysis and pruning steps. Each covering corresponds to one of the subsets of this set. The term covering describes the process of laying the letters of each of the morphemes of a subset on the letters of the input name. Therefore, in a covering, some of the letters of the name are accounted for by one or more of the morphemes while some of the letters are not. The coverings are filtered for legality by the Filter Covering block which tests that there is no morpheme overlap in the covering, and that letters in the name that are not accounted for, are reasonable linguistic units. The legal coverings have their fitness evaluated by the Score Covering block. The highest-ranking covering is chosen as the best morphological parsing of the name.

Generate Morpheme Covering

The output of the prune analysis results module can be inadequate for defining a morphological parsing of the input name, since it could contain incompatible morphemes, or not account for all the letters in the name. By incompatibility we mean that the morphemes could overlap (two morphemes in an analysis overlap if they contain the same letter in the name) or the gaps between morphemes could form unsuitable morphemes. Hence, it is desirable to find a "covering" of the input name by a subset of the morphemes found in the analysis.

Any subset of non-overlapping morphemes of the set of morphemes found by the analysis module, plus the remaining letters of the input name (those letters that are not accounted for by this subset), is called a *covering* of that name by the set of morphemes. We consider two examples shown in Table 1

	WHITAKER	BOTHAM
Prefix morphemes	WHIT	BOTH
Mid morphemes		
Suffix morphemes	ER	HAM
Use both	#WHIT#AK#ER#	
Only prefix	#WHIT#AKER#	#BOTH#AM#
Only suffix	#WHITAK#ER#	#BOT#HAM#
Use none	#WHITAKER#	#BOTHAM#

Table 1: Examples of the Covering process

The task of choosing the best possible coverings is left to other modules. This module concentrates on generating candidate coverings. Suppose that the maximal number of middle morphemes is m, then the number of possible coverings is: $2*2^m*2 = 2^{m+2}$ (prefix, m middle, and suffix). For m=4 this becomes inhibitive, as the scoring process includes transcription and parsing into syllables. Since a full search is not desirable we adopt the following strategy: choose no more than two middle morphemes. Assuming that there are no more than 4 middle morphemes one prefix and one suffix (6 altogether), the maximal number of coverings generated using this strategy is: 2*(1+4+4*3/2)*2= 44 (which is about 30% reduction in RAM requirements).

Thus, it is desired to form a covering of the input name using as many of the morphemes that were found, such that no overlap occurs between the morphemes and the remainder of the letters in the input name (those that were not matched by a morpheme) make up a reasonable morpheme.

Filter Covering

A covering of the input name generated by the previous module is checked by the filter-covering module for compliance to two regularity conditions:

- no overlap between morphemes of the covering
- the letters of the input name remaining unmarked, after marking those that belong to each of the morphemes in the analysis of the current covering, contain neither isolated consonant nor an isolated vowel. This can be stated as: A gap left by a covering must include an A, E, I, O, U, or Y, and at least one consonant.

Score Covering

The scoring mechanism is based on the observation that in English CVC syllables have preference over CV or VC syllables. Hence the score per covering is computed according to the following rule:

- Each CVC syllable adds 3 to the score, while each VC or CV adds 1 to the score.

As an example let us return to the following example: BOTHAM, which yields two different coverings:

1. #BOTH#AM# having a score of 4 (3+1)
2. #BOT#HAM# having a score of 6 (3+3)

Hence, in this case the second morpheme parsing is preferable. Notice however that to obtain the parsing into syllables, transcription is required.

Select Highest Scoring Covering

This is a simple module. Out of all legal coverings it picks the covering that yields the highest score.

6. SUMMARY

A system for name announcement is described, as well as the linguistic processing module. Two of the modules, language of origin determination and morphology analysis are described in detail. Our approach takes into consideration the constraints of the target implementation, and naturalness of the pronunciation.

ACKNOWLEDGEMENTS

The authors wish to acknowledge Dror Jerushalmi and George Gurevich of DSP Group for their support and help.

REFERENCES

1. Allen J., Hunnicutt M. S., and Klatt D., "From Text to Speech - The MITalk System", Cambridge University Press, 1987.
2. Elovitz H.S., Johnson R., McHugh A, Shore J.E., "Letter-to-Sound Rules for Automatic Translation of English Text to Phonetics", IEEE Transactions on Acoustics Speech and Signal Processing, Vol. ASSP-24, No. 6, December 1976, pp 446-459.
3. Fradkin R.A., "The Well Tempered Announcer", Indiana University Press, 1996.
4. Golding, A.R., "Pronouncing Names by a combination of Rule-Based and Case-Based Reasoning", PHD Thesis Stanford University, 1992.
5. Klatt D.H., "Review of Text-To-Speech Conversion of English", J. Acoustical Society of America, 82 (3), September 1987, pp 737-793.
6. Malecot A., "Cross-Language Phonetics", in Th. A. Sebcok ed., "Current Trends in Linguistics," vol. 12 pp 2507-2536, 1974, Mouton The Hague, Paris.
7. Rosenhouse J., "Phonetic and Other Factors Affecting the Pronunciation of Foreign Proper Names in Speakers of American English", IPS-98 Conference 1998.