Increased Diphone Recognition for an Afrikaans TTS system

Francois Rousseau and Daniel Mashao
Department of Electrical Engineering, University of Cape Town, Rondebosch,
Cape Town, South Africa, frousseau@crg.ee.uct.ac.za, daniel.moshao@ebe.uct.ac.za

Abstract – In this paper we discuss the implementation of an Afrikaans TTS system that is based on diphones. Using diphones makes the system flexible but presents other challenges. A previous effort to design an Afrikaans TTS system was done by SUN. They implemented a TTS system based on full words. A full word based TTS system produces more natural sounding speech than when the system is designed using other techniques. The disadvantage of using full words is that it lacks flexibility. The baseline system was build using the Festival Speech Synthesis System. Problems occurred in the baseline due to the mislabeling of diphones and the diphone index. The system was improved by manually labeling the diphones using Wavesurfer, and by changing the diphone index. Wavelength comparison tests were done on the diphone index to show how much of the diphones are recognized during synthesis. For the diphones tested results show an average improvement of 38% in the recognition of diphones compared to the baseline. These improvements improve the overall quality of the system.

Key words**:** Festival Speech Synthesis, diphones, labels, diphone index

1. INTRODUCTION

Afrikaans is the first language to approximately six million people in South Africa. The language originates from seventeenth century Dutch and is influenced by English, Malay, German, Portuguese, French and other African Languages [2]. Together with English it first became the official language in 1925 according to the *Act of 1925*. Previous work on an Afrikaans Synthesizer was made by SUN [1]. The system is embedded within a system called AST (African Speech Technology) which is a hotel reservation booking system that works for Afrkaans, Zulu, Xhosa and English.

TTS (Text-to-speech) in the simplest words is the conversion of text to a speech output using a computerized system. It therefore allows for the communication between humans and machines through synthetic speech [5]. TTS consists of two phases. The first is called high level synthesis also known as the front-end [8]. This is where text analysis and the linguistic analysis are done on the input text. The second phase is called the low level phase, also known as the back-end. This is the phase where prosody is added to the phonetic information gained at the front and where the speech waveform is generated [4]. These two phases are shown in Figure 1.

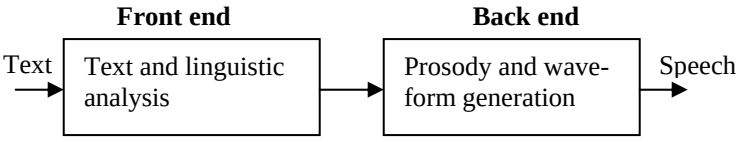


Figure 1: Two phases involved in TTS

Section 2 discusses different techniques of achieving TTS. The most flexible and natural sounding TTS systems are achieved by concatenating short prerecorded speech samples such as phones and diphones to produce synthetic speech. The Festival Speech Synthesis System is a system based on this technique [3]. The system was designed at the Centre for Speech Technology Research (CSTR), at the University of Edinburgh, Scotland [3]. It is an open source system with the ability to be a workbench for the development of new TTS systems [8]. The labeling of diphones together with the diphone index gives the system crucial on the duration of the diphones. This is where problems occur in Festival. The problems are:

(i) the automatic labeling technique label's the diphones incorrectly
(ii) the diphone index is set up to only recognize portions of the diphones.

The methods involved in solving these problems are discussed in Section 3. Section 4 discusses the results of the baseline system and the improvements made by the methods discussed in Section 3. Conclusions on the improved system are discussed in Section 5.

2. TEXT TO SPEECH SYNTHESIS

The first speech synthesis system was build by Christian Kratzenburg in 1779 [5]. The system was able to produce five long vowel sounds using resonators activated by vibrating reads. This breakthrough has lead to the various synthesis techniques available today. The three main techniques are articualtory synthesis, formant synthesis and concatenative synthesis.

The articulatory synthesis method models the human articulators and vocal cords. The advantage of this method is that the vocal tract models allow for accurate models of the transients due to abrupt area changes [4]. The disadvantage is that it is incredibly difficult to implement and hence very rarely used in practice.

Formant synthesis models the formant frequencies of human speech. The advantage is that it has an infinite number of sounds which makes its more flexible than other methods [4].

The disadvantage is the lack of natural sounding synthetic speech. This is due to the fact that usually up to five formants are required for good synthetic speech [4].

Concatenative synthesis connects prerecorded speech units derived from natural speech for synthesis [4]. The sizes of the units vary from phones to diphones to even full words. Using full words has the advantage that it produces very natural synthetic speech. Such systems are however limited to a specific database and hence not flexible, as in the case of the AST system [1]. Building full blown TTS systems based on this method is expensive and time consuming. Using diphones has the advantage that the system is very flexible. Instead of using long prerecorded speech units such as words this method uses diphones, which are the possible phone-to-phone transitions for the language. For example the word '*hello'* would be made up of the diphones '*h-e'* and '*l-o'*. By theory the amount of diphones present in a language is the square of the number of phones [8]. The disadvantage of this method is that there is no pronunciation variation in the diphones. This leads to unnatural sounding synthetic speech and information on segment duration and prosody must to be added to gain naturalness [10]. This includes information on stress levels and phone durations for the desired output.

The aim of this paper is to present a full blown Afrikaans TTS system that is flexible and natural. Therefore we implemented the concatenative synthesis technique based on diphones to build the Afrikaans synthesizer.

3. IMPLEMENTATION OF AFRIKAANS SYSTEM

Building the Afrikaans synthesizer using the Festival Speech Synthesis System is faced with the problems of automatic labeling and an undesired diphone index. The baseline system was constructed using the methods provided by Festival and was improved by manually labeling the diphones and rebuilding the diphone index.

3.1 The baseline system

The system is built using the Festival Speech Synthesis System which runs in a UNIX environment under Linux. The packages required are:

1. Festival-1.4.3
2. Festvox-2.0
3. Speech_tools-1.3

These packages are freely available for download from the CSTR website [3]. A diphone database and a lexicon database are required for building a new voice in Festival. Modules written in Scheme (a Festival specific language) are provided for these two requirements and are to be manipulated to suite the language.

Constructing the diphone database

The diphone database was constructed using *Die Groot Woorde Boek*, Afrikaans dictionary [9]. In total we found 64 phones therefore the amount of diphones were 4096. The diphone database was generated automatically by the system using the phone-to-phone transition rules for Afrikaans. These are the consonant-consonant, consonant-vowel, vowel-vowel and vowel-consonant transition rules for Afrikaans. The generated diphones are placed within non-sense words. These words are used for the extraction of the speech units for concatenation. Table 1 shows a list of diphones located within non-sense words.

Table 1: Examples of diphones located within non-sense words

Diphones	Non-sense word	Diphones with-in non-sense word
'b-a' 'a-b'	tababa	t *a-b-a-b* a
'sj-a' 'a-sj'	takasjata	t a k *a-sj-a* t a
'kn-o' 'o-kn'	takoknota	t a k *o-kn-o* t a
'tj-e' 'e-tj'	taketjeta	t a k *e-tj-e* t a

Recording the speaker

The objective of recording is to get the uniform set of diphone pronunciations. For this research my own voice was used. Recording was done using *na_record*, part of the *speech_tools-1.3* package. This recording system creates wave files of the recorded non-sense words and places them into a log file that stores them as * .wav files.

Labeling the non-sense words

The labeling of non-sense words is important because it labels the positions of the diphones within the non-sense words. At minimum the start of the preceding phone to the first phone in the diphone, the changeover and the end of the second phone should be labeled [8]. Festival provides an automatic labeler called *make_labs* to automatically label the diphones.

Building the diphone index

The diphone index is needed for the extraction of diphones from the acoustic non-sense words. During synthesis the system looks at this index to see where in the recorded non-sense words the diphone should be extracted from. The index is built by taking the diphone list and finding the occurrence of each diphone in a label [8]. By default the diphone will be extracted from the middle the first phone to the middle of the second phone. This is done by using *make_diph_index* a module provided by Festival.

Extracting the pitchmarks

Festival requires information on the pitch periods in an acoustic signal for synthesis and therefore the pitchmarks in each speech waveform must be extracted [8]. The technique used to get this information is called Residual Excited Linear-Predictive Coding (LPC). Linear prediction works on the basis that a current speech sample x(n) can be predicted from a finite number of previous *p* amount of samples x(n-1) to x(n-k) by a linear combination with an error e(n) [4]. This error term is the residual signal. And therefore

$$x(n) = e(n) + \sum^{p}_{k=1} a(k)x(n-k), \quad (1)$$

and

$$e(n) = x(n) - \sum^{p}_{k=1} a(k)x(n-k) = x(n) - \tilde{x}(n) \quad (2)$$

where $\tilde{x}(n)$ is the predicted value, p is the linear predictor order and a(k) are the linear prediction coefficients which are found by minimizing the sum of the squared errors over a speech frame [4].

The best way to find the pitchmarks in a waveform is to extract them from an EGG (electroglottograph) recording of the signal [8]. The EGG records the electrical activity in the glottis during speech, which means the pitch moments, can be found more easily and are more precise [8]. For this research no EGG was available so the pitchmarks were extracted automatically from the waveforms using methods provided by Festival.

Building the LPC parameters

Due to the natural changes in the recording environment and because of human fatigue the ideal recordings could not be realized. These factors made it impossible for all recorded diphones to be at the same power level. These fluctuations in power levels produce bad synthesis [8]. To overcome this power normalization was done on all the recorded non-sense words using a method provided by Festival. The method used finds the mean power for each vowel in each of the non-sense words and then finds the power factor with respect to the overall mean vowel power [8]. Using the calculated power factors the LPC coefficients and residuals for LPC analysis were generated.

Building lexicon support database and prosody

The lexicon database consists of the letter-to-sound rules and pronunciation guides for the system. Unpronounceable words and abbreviations are also given definition here.
Certain phones and diphones are not always as required when trying to pronounce certain words. Take the word "*Francois"* as an example. The first syllable of the word can be pronounced just by using the information of the phones. The second syllable is not pronounced correctly in the context of how the full word should be pronounced. For this reason the system needs to be told how to pronounce this syllable. Below is an example taken from the lexicon database that shows how the syllable is pronounces.

```
(lex.add.entry
 '('Francois' nil (((f r a n) 0) ((s w a) 0))))
```

This now gives the system a definition to how the word "Francois" should be pronounced.

Problem statement

Problems occur in the baseline system due to the mislabeling of diphones by the automatic labeler, and due to the basis on which the diphone index is built. The quality of a concatenative TTS system is directly related to the accuracy with which the underlying acoustic inventory is labeled [13]. Therefore because the diphones are mislabeled the performance of the system is undesired. The problem with the diphone index is that it is set up to only recognize the portion of the phone-to-phone transitions. This means that the entire transition is not used during synthesis which is also undesirable.

3.2 Improving the baseline system

By manually labeling the diphones and by changing the basis on which the diphone index is built the baseline system is improved.

The manual labeling the diphones fixes the errors made by Festival's automatic labeler by placing the labels in the correct positions. Figure 1 shows an example where the non-sense word "*a-c-i-c-a*" is labeled incorrectly.

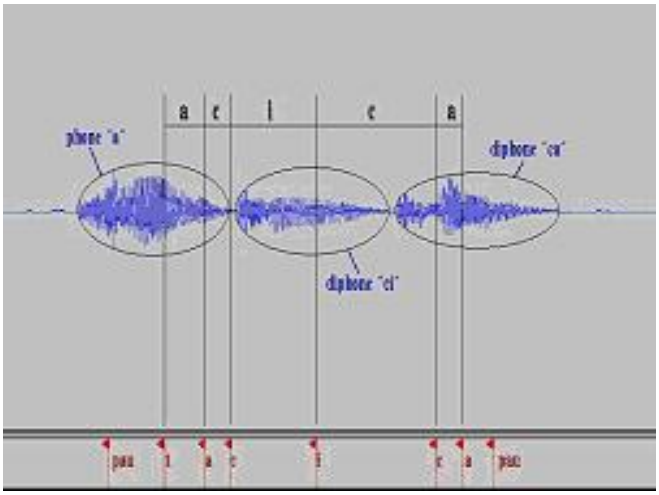


Figure 1: "*a-c-i-c-a*" labeled incorrectly

As seen from Figure 1, the diphone "*ci*" is labeled in such a way that it contains a portion of phone "*a*" and a portion of "*ci*". When the system calls "*ci"* for synthesis, it will pronounce the portion of "*a*" together with the portion of "ci", which is not desired. To solve this problem we re-label all the the non-sense words using Wavesurfer [7]. Figure 2 shows the correct labeling for the non-sense word "*a-c-i-c-a*"

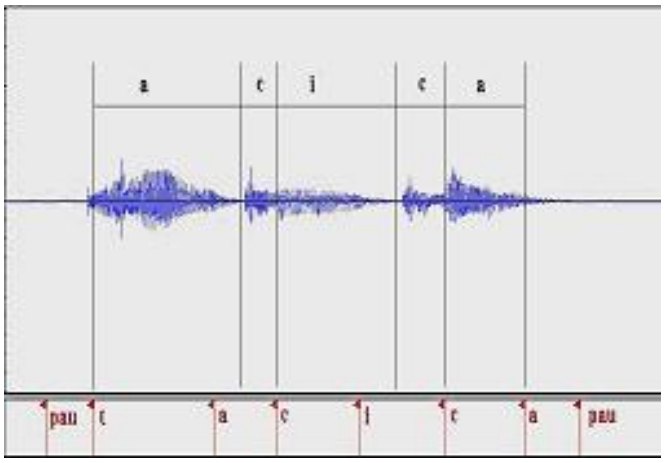
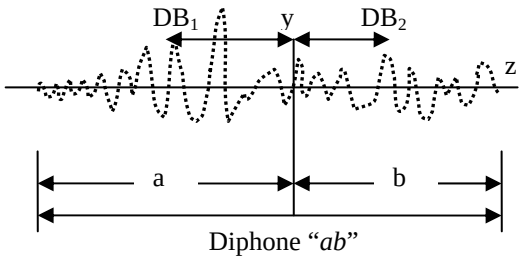


Figure 2: Correct labeling of *"a-c-i-c-a"*

Now the non-sense word is labeled such that it only contains the portion of "*ci*" that is needed and nothing else. When the system now calls on "*ci*" it will only pronounce what was labeled as "*ci*".

By default the diphone index is build in such a way that the portion from the middle of the first phone to the middle of the second phone is used for synthesis [8]. This is because diphone boundaries (DB) are positioned as shown in Figure 3. This is not desirable since full diphones are needed in order for the synthesized words that make sense.



Diphone "*ab*"

Figure 3: Diphone boundary of diphone "*ab*"

Festival's *make_diph_index* program uses the equation

$$DB_2 = (y+z)/2.0 \qquad (3)$$

where y- the mid point in diphone

z- the end point of second phone, to calculate the diphone boundary of the second phone.

To stretch the DB to the end of the diphone this line of code in *make_diph*_index was changed to:

$$\text{Let } DB_2 = z, \quad \text{instead of (3)}$$

This changes the definition the DB by placing it at the end of the diphone. The code is mirrored so the same applies for the first phone.

By applying these two methods to the baseline system, it will ensure that the full diphones are called at synthesis and not portions of it. This will increase the overall performance of the system.

4. EXPERIMENTAL RESULTS

The measures used for testing are based on how much of the diphones are recognized at synthesis. The system is tested by checking the periods of the diphones in Wavesurfer in comparison to the diphone index. For full diphone recognition these two periods should be the same. Table 2 shows the improvements made by the methods discussed in Section 3. It shows the true length of each diphone, the length recognized by the baseline and the improvements made in seconds.

Table 2: Percentage improvements on the baseline

Diphone	True wave-length (s)	Baseline wave-length (s)	Improve-ments made (s)	% Impr.
a-b	0.4432	0.3307	0.1125	34.01
b-a	0.3005	0.198	0.1025	51.76
a-c	0.5575	0.375	0.1825	48.66
c-a	0.411	0.281	0.13	46.26
a-d	0.482	0.392	0.09	22.95
d-a	0.337	0.207	0.13	62.80
a-f	0.4285	0.316	0.1125	35.60
f-a	0.3605	0.248	0.1125	45.36
a-g	0.638	0.54	0.098	18.14
g-a	0.498	0.352	0.146	41.47
a-h	0.5403	0.4303	0.11	25.56
h-a	0.395	0.285	0.11	38.59
a-j	0.6165	0.514	0.1025	19.94
j-a	0.4955	0.378	0.1175	31.08
a-k	0.414	0.294	0.12	40.81
k-a	0.362	0.212	0.15	70.75
a-l	0.5725	0.455	0.1175	25.82
l-a	0.455	0.335	0.12	35.82
a-m	0.5815	0.459	0.1225	26.68
m-a	0.432	0.317	0.115	36.27

As seen from Table 2 the improvements made are up to almost 50% in some cases. On average for these twenty diphones that were tested an improvent of 37.9% was made. Therefore more of the diphones are recognized during synthesis. This table also gives evidence to why the baseline system did not perform as desired. The majority of the portions recognized in the baseline were at the start of the diphone which means that the percentages lost at the end, held crucial information regarding the second phone in the diphone.

5. CONCLUSIONS

From the results shown is Section 4 it can be concluded that by manually labeling the diphones and changing the diphone index the overall quality of the TTS system will be improved.

Future work is to be done on completing the re-labeling process and changing the entire diphone index. This will ensure that all diphones are recognized correctly and hence should improve the overall quality of the system to such a point that it can synthesize full words and sentences accurately.

REFERENCES

[1] Prof J. Roux, Prof L. Botha, Prof J du Preez "African Speech Technology", Online Resource: www.ast.sun.ac.za, Last accessed 7 October 2004

[2] J. Oliver "Afrikaans", Online resource: www.geocities.co.za/users/~jako/lang/afrwrd.html, Last accessed 7 October 2004

[3] A. W. Black, R. Clark, K Richmond, S King "The Festival Speech Synthesis System" University of Edinburgh, Scotland www.csrt.ed.ac.uk/projects/festival, Last accessed 15 October 2004

[4] S. Lammetty, "Review of Speech Synthesis Technology", Master's Thesis, Department of Electrical Engineering, Helsinki University of Technology, March 1999, Available at http://www.acoustics.hut.fi/~slemment/dipp/index.html, Last accessed 5 August 2004

[5] A. Conkie, "Robust unit selection system for speech synthesis", Proc. Joint Meeting of ASA, EAA and DEGA, Berlin, Germany, March 1999.

[6] O. Salor, B. Pellom, M. Demirekler, "Implementation and Evaluation of a Text-To-Speech Synthesis System for Turkish", INTERSPEECH-2003/Eurospeech-2003. pp 1573-1576, Geneva, Switzerland, Sept. 2003

[7] K. Sjolander , J Beskow "Wavesurfer" , Audio Editing Software 2004, www.speech.kth.se/wavesurfer, Last accessed 25 September 2004

[8] A. W. Black, K. Lenzo "Building Synthetic Voices", unpublished document, Carnegie Mellon Universtiy, Available at http://festovx.org.bsv, Last accessed 5 November 2004

[9] Kritzenbeurg, M. S. B.(Matthys Stefanus Benjamin), Groot Woordeboek, Pretoria, Vanschaik 1972,

[10] N. Rochford, "Developing a new voice for Hiberno-English in The Festival Speech Synthesis System", Final Year Thesis Project, Trinity College Dublin. Available at http://www.cs.tcd.ie/courses/csll/projects4.html, Last accessed 7 June 2004

[11] T. Dutoit, "Introduction to Speech Synthesis Systems", Kluwer Academic Publishes, Dordrecht, 1997

[12] M. Chu, H. Peng, E. Chang, "A concatenative Mandarin TTS system without prosody model and prosody modification", Proceedings of 4th ISCA workshop on speech synthesis, Scotland, 2001.

[13] M.J Makashay, C.W Wightman, A.K Syrdal, A Conkie, "Perceptual Evaluation Of Automatic Segmentation In Text-To-Speech Synthesis", In Proc, ICSLP, volume 2, pp. 431-434, 2000