An Overview of the VUB Entry for the 2008 Blizzard Challenge

Lukas Latacz, Yuk On Kong, Wesley Mattheyses, Werner Verhelst

Laboratory for Digital Speech and Audio Processing (DSSP)
Department of Electronics and Informatics, Vrije Universiteit Brussel, Belgium
{llatacz, ykong, wmatthey, wverhels} @ etro.vub.ac.be

Abstract

In this paper, we describe the configuration of our synthesizer, as used for the Blizzard Challenge the first time. Two new UK English voices were built for the DSSP synthesizer, our in-house unit selection synthesizer, which uses non-uniform units and a symbolic description of target prosody. Listening tests indicate reasonable quality although there is still room for improvement.

Index Terms: speech synthesis, unit selection, evaluation of synthesized speech

1. Introduction

The Blizzard challenge [1] is a yearly speech synthesis challenge for evaluating synthesizers and advancing the technology. In 2008, the Vrije Universiteit Brussel (VUB) team participated in the challenge the first time. We built two UK English voices: voice A, using the complete Roger database and voice B, using the Arctic subset of that database.

Our system, tentatively called the DSSP synthesizer, is a synthesizer based on unit selection. It was developed as part of a computerized reading tutor for children with reading problems under a speech technology project funded by the Belgian government known as SPACE (SPeech Algorithms for Clinical and Educational applications) [2]. Although the quality of synthesized speech has improved a lot over the last decade, recordings of natural speech instead are still being used in most recent reading tutors (e.g., [3] and [4]). In order to meet the quality expected for such an application, we designed a hierarchical speech generation strategy which synthesizes speech as a sequence of non-uniform units. Recently, the synthesizer has been extended to support multi-modal unit selection, so to synthesize speech audio-visually.

Corpus-based concatenative speech synthesis (e.g. [5]) is the mainstream way to synthesize speech. In such synthesizers, a large speech database is first segmented into small units. To synthesize an input text, the best combination of speech units is selected from the database to match the utterances, based on the sum of weighted cost functions. The selected unit sequence is then concatenated to generate synthesis. Like any unit selection synthesizer, the DSSP synthesizer has two parts: a language-dependent front-end providing natural language processing, and a language-independent back-end providing unit selection. The system supports Dutch, and now UK English as well because of the challenge.

This paper is structured as follows; in section 2, we give an overview of the voice-building process. The front-end and the back-end are described in sections 3 and 4 respectively. The results of the listening tests are discussed in section 5. Our conclusions and future work are given in section 6.

2. Voice-building

The construction of a new voice for the DSSP synthesizer is mostly automated. Before building a voice, the recordings for the new voice need to be segmented and labeled. An orthographic transcription of each of these utterances must be available.

We used EHMM, which is part of the Festvox toolset [6], to segment the data for the two voices. EHMM is an HMM-based forced aligner. It can detect and insert pause labels that are missing in the *input* labeling. Festival utterance structures [7] containing such labels were provided by the Blizzard organizers. Since our front-end uses the same phone set and lexicon as those used for creating these utterance structures, we could use the labels in the utterance structure files as input to the EHMM script. Training was performed with the default settings on the Arctic data in order to align the data for both the full and Arctic voices.

One feature of the DSSP synthesizer is that a voice can be built adaptively, i.e. new data can be added to the system without the need to rebuild the existing part of the voice. The phonemic sequence of each utterance is stored in a tree-based index, allowing fast search of the speech database. Separate indexes are also constructed for words and syllables.

As in most other unit selection synthesizers, acoustic features needed for computing join costs, such as MFCC and f0, are extracted offline and stored beforehand. Based on the orthographic transcription of each utterance, the front-end generates symbolic information, which is used to calculate the target costs. Each segment, i.e. phoneme, of the database is labeled as such. Small mismatches between the front-end output and speech database labels, mostly due to pauses which are not predicted by the front-end, are resolved by performing dynamic time warping between the label sequence and the phonemic sequence generated by the front-end.

3. UK English Front-end

The UK English front-end of our synthesizer performs language-dependent natural language processing. Figure 1 shows an overview of the front-end. It uses some Festival [8] modules to perform its tasks.

The target prosody of the output speech is described symbolically only. Acoustic parameters, such as f0 and duration, are difficult to predict because of the natural variation of prosody. As prosody is described symbolically only, acoustic prosody models are not needed. This idea is also implemented in some other synthesizers, such as Multisyn [9].

Firstly, the input text is normalized into words, of which the pronunciation can be determined. A part-of-speech tagger determines the syntactic category of each word in the utterance. These words are then organized into phrases. The

algorithm for predicting phrase boundaries and pauses is described in the next section.

The word pronunciation module converts each word into segments (i.e. phonemes) and groups these segments into syllables. Lexical stress is assigned to each syllable. The pronunciation of a word can be looked up in a lexicon, in our case the Unisyn lexicon [10], with its orthographic transcription and part-of-speech tag as input. The Unisyn lexicon supports multiple regional pronunciation variants. The lexicon was set to its Received Pronunciation (RP) variant, which is close to if not the accent of the speaker itself. Out-of-vocabulary words are handled by the memory-based grapheme-to-phoneme conversion technique described in [11], implemented with TiMBL [12]. No post-lexical processing is performed.

The intonation module predicts a symbolic description of the intonational contour of the input utterance using a decision tree. Intonation is described as ToBI accents.

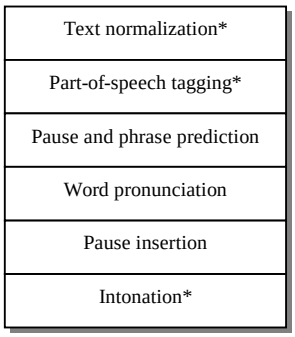


Figure 1: *Overview of the UK English Front-end. Modules with an * are provided by Festival.*

3.1. Pause and Phrase Prediction

An utterance consists of one or more prosodic phrases. It is a well-known fact in linguistic literature that phrase boundaries are optional. For example, the utterance "*Simon and Rob were seeing through the window.*" can be pronounced as

- Simon and Rob were seeing though the window.
- Simon and Rob | were seeing though the window.
- Simon and Rob were seeing | though the window.

(where "|" represents a phrase boundary)

This optionality makes it more difficult to predict phrase breaks and to evaluate phrasing algorithms because, in many cases, phrase breaks which do not match a reference may actually be judged as acceptable by human listeners [13]. Furthermore, pauses occur not only between phrases, but also within a phrase. Such pauses can also be found in the Roger and Arctic databases. Therefore, we need to predict not only those pauses between phrases, but also those within a phrase.

Over the last two decades, research in phrase boundary prediction has shifted from rule-based approaches to data-driven methods. These trainable systems are commonly trained on manually labeled data (text). The process is labor-intensive and inter-rater agreement is typically not very high. Silverman et al. reported an agreement of 69% among four labelers [14]. Ideally, the labeling should be carried out manually using the orthographic transcription and the speech database. This is not practical. Therefore we propose a different approach. We performed training on automatically labeled data. This results in speaker-dependent training. However, we need to assume that the speaking rate of the speech database does not vary too much, which is the case for the Roger and Arctic database. In our system, we defined three types of pauses:

- *Heavy*: long pauses, occurring between utterances
- *Medium*: shorter pauses, occurring between phrases of an utterance
- *Light*: short pauses, occurring between words of a phrase

The data is labeled iteratively as follows:

1. Label each pause as heavy, medium or light by referring to the punctuation. If there is a full stop, the pause is labeled as "heavy"; if there is a comma, "medium", etc. Calculate the mean duration of each type of pause.
2. For each pause, re-set the label of the pause as the type which has the closest mean duration.
3. Recalculate the mean duration for each type.
4. Go back to (2) until none of the pause labels needs to be changed.

After labeling, a machine-learning algorithm can be used to predict pauses. Each token of the text of the training database can now be labeled with one of four classes (heavy, medium, light and non-pause) by checking whether the speaker has inserted a pause after the token in the recording, and reading the type of pause from the iterative process above. The machine-learning part of our system is similar to the memory-based learning (MBL) approach described by Busser et al. [15]. The features that we use are the part of speech, pre-punctuation, punctuation and orthographic transcription of the tokens; the 2 tokens preceding and the 2 following. Machine learning is done by the IGTree algorithm, using TiMBL with default settings. For synthesis purposes, both medium and heavy pauses are actually taken as phrase boundaries because some pauses, though labeled as "heavy", are found to be phrase pauses.

In order to evaluate our system, we performed 10-fold cross-validation. For the Arctic voice, training was performed on the Arctic data set only. Results are shown in table 1, listing precision, recall and F-score. Accuracy is the total number of correct items.

Training set	Precision	Recall	F-score	Accuracy
Full	85.465 %	85.475 %	85.463 %	95.826 %
Medium	86.253 %	86.340 %	86.297 %	96.075 %
Arctic	68.515 %	68.559 %	68.537 %	94.978 %

Table 1: *The Results of Pause and Phrase Boundary Prediction. The "medium" data is a randomly selected subset of the full training set (28027 tokens).*

Precision	Recall	F-score	Accuracy
74.4 %	76.1 %	72.8 %	90.0 %

Table 2: *The Results as Reported by Busser et al. Training was carried out on a manually labeled corpus using a similar machine-learning technique.*

The full database is almost 10 times larger than the Arctic subset (93426 vs. 9836 tokens) Using such a large training database has a positive effect on the performance. The results of training a system on manually labeled data (39369 tokens) as reported by Busser et al. are shown in table 2. Note that

they used a different training corpus. These results seem to suggest that training on automatically labeled speech can yield better performance. Our labeling procedure probably makes it easier for the machine-learning algorithm to learn. The consistency of the speaker regarding phrase boundaries might be better than the agreement between manual labelers. Due to the optionality of phrase boundaries, the "real" results might be even better than these results here. Actually, we need to insert breaks so that our breaks match those in the speech database. Further analysis of these results is, however, out of the scope of this paper.

4. UK English Back-end

The back-end of the DSSP synthesizer consists of a unit selection framework, allowing several unit selection synthesizers to be implemented. Based on the output of the front-end, targets are constructed. These targets could be of any size. Besides the target cost based on extended phonemic identity matching mentioned in [16], several other target costs are defined, each describing a single symbolic feature. For each demiphone of a candidate unit, we check whether the value of a feature matches that of the corresponding demiphone of the target. The value of the target cost is the number of demiphones of which the value is different. Table 3 lists the targets costs used in our synthesizer for the Blizzard Challenge.

Units matching the phonemic description of the targets are searched for in the database. A simple pruning method is used. The N-best units only in terms of target costs were used in order to speed up selection (N is set at 200 units). If no units are found for a particular target, the default back-off strategy is to look for phones or demiphones instead. If still no suitable units are found, any missing phone is replaced by silence.

The search for the best unit sequence is performed by our implementation of the Viterbi algorithm. The cost function $c(u_1, u_2, ..., u_n, t_1, t_2, ..., t_n)$ is used to calculate the cost for selecting a sequence of n candidate units u_i, with their corresponding targets being t_i, based on k target costs c_j^{target} and m join costs c_j^{join}:

$$c(u_1, u_2, ..., u_n, t_1, t_2, ..., t_n) = \alpha * \sum_{i=1}^{n} \frac{\sum_{j=1}^{k} w_j^{target} c_j^{target}(u_i, t_i)}{\sum_{j=1}^{k} w_j^{target}} + \sum_{i=1}^{n-1} \frac{\sum_{j=1}^{l} w_j^{join} c_j^{join}(u_i, u_{i+1})}{\sum_{j=1}^{l} w_j^{join}} \quad (1)$$

The weight α allows the fine-tuning between join and target costs, and is currently set to 1. The weights w_j^{target} and w_j^{join} are set manually.

To measure the smoothness of a join, differences in pitch, spectrum and energy are taken into account. 4 join costs are used:

- The Euclidean distance between the MFCC's (12 coefficients including the first one) on the two sides of a join.
- The absolute difference in f0 (logarithmic) between the two sides of a join. If the phone at the join position is voiceless, this cost is 0.
- The absolute difference in energy between the two sides of a join.
- Adjacency (whether the demiphones on either side of the boundary are the left and right halves of the same particular instance of a phoneme in the database)

Units are then concatenated using a PSOLA-based algorithm with optimal coupling [17]. No further signal processing is performed.

Level	Target cost
Segment	Phonemic identity*
Segment	Pause type (if silence)*
Segment	Position in syllable
Syllable	Phoneme sequence
Syllable	Lexical stress*
Syllable	ToBI accent*
Syllable	Is_accented*
Syllable	Onset and coda type [18]*
Syllable	Onset, nucleus and coda size*
Syllable	Distance to next/previous stressed syllable, in terms of syllables
Syllable	Number of stressed syllables until next/previous phrase break
Syllable	Distance to next/previous accented syllables, in terms of syllables
Syllable	Number of accented syllables until next/previous phrase break
Word	Position in phrase
Word	Part of speech*
Word	Is_content_word*
Word	Has_accented_syllable(s)*
Word	Is_capitalized *
Word	Position in phrase*
Word	Token punctuation*
Word	Token prepunctuation*
Word	Number of words until next/previous phrase break
Word	Number of content words until next/previous phrase break

Table 3: *The List of Target Costs Used in the Synthesizer. Those with a * are also calculated for the neighboring segments, syllables or words. Neighboring syllables are restricted to the syllables of the current word. Three neighbors on the left and three on the right are taken into account.*

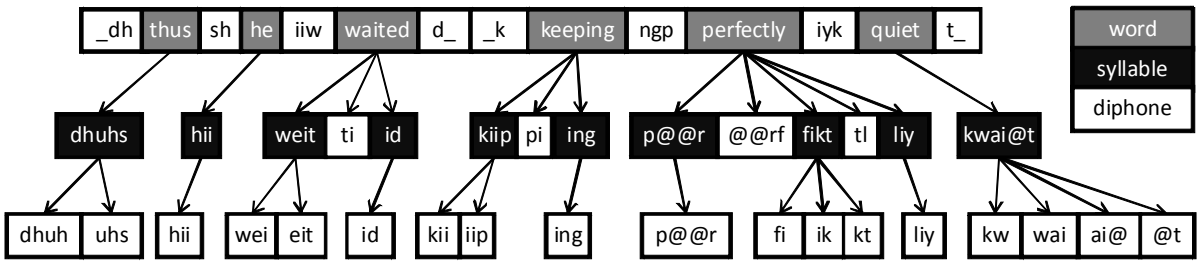


Figure 2: *Example of a Multi-level Selection Tree: "Thus he waited, keeping perfectly quiet."*

4.1. Multi-level Unit Selection

Since the speech database can be quite large (e.g. 15 hours for voice A), unit selection synthesizers face challenges in 2 areas: synthesis quality and speed. In order to reduce computational complexity and increase quality, we implemented a multi-level unit selection strategy in the DSSP synthesizer, which is a top-down approach. We aim at finding longer units first, before resorting to shorter ones.

A multi-level tree is constructed based on the output of the front-end. In figure 2, an example of such a tree is shown. Each level of the tree represents a type of unit, e.g. words, diphones. These levels can be adapted according to the target language. Each node of the tree represents a single target. For UK English, we implemented these levels: word, syllable and diphone. Units are joined at diphone boundaries, so additional diphones are inserted at this level to fill any gap. Note that targets representing words or syllables which contain only a single phoneme are not added to the tree, since these can be found directly at the diphone level. With "adjacency" as a join cost, combinations of candidate units which are neighbors in the speech database are favored.

4.2. Selection of a Synthesizer from our System for the Challenge

In order to compare the quality of the different synthesizers of our system (the "DSSP synthesizer"), we conducted an experiment in Dutch recently. As an initial evaluation, we had 6 subjects. Each listened to 100 synthesis files, i.e. 25 test utterances each synthesized by the four synthesizers as described below. The 25 test utterances are at 5 different so-called AVI levels [19], levels used for Dutch to reflect differences in lexical and syntactic complexities with level 1 being the simplest. 5 utterances were used at each level. In table 4, the columns represent the different synthesizers of our system:

1. Unit selection using long non-uniform units [15].
2. Homogeneous unit selection (diphones).
3. Multi-level unit selection (with word and diphone levels only);
4. Multi-level unit selection (with phrase, word and diphone levels).

The listeners were to listen to syntheses with high-quality headphones (Sennheiser HD555) in a quiet environment. They could listen to each file as many times as they wished. They were instructed to rate them using a scale from 1.0 to 5.0 (MOS, or mean opinion score) and to use up to one decimal point.

As we expected, synthesis quality falls as we go up the AVI levels (lexical and syntactic complexities). Multi-level (or hierarchical) unit synthesis performed the best while non-uniform unit selection the worst. (The two configurations of the multi-level (or hierarchical) unit synthesizers actually generated identical syntheses and also MOS ratings, because no phrase was found/selected from the speech database for synthesizing any of the test utterances.) ANOVA was conducted and the differences were found to be statistically significant both across rows, i.e. AVI levels (p=0.001) and across columns, i.e. synthesizers (p=0.05).

	1	2	3	4	mean
AVI1	3.0	3.5	3.5	3.5	3.4
AVI2	3.1	3.0	3.1	3.1	3.1
AVI3	3.1	3.1	3.1	3.2	3.1
AVI4	2.3	2.9	2.8	2.8	2.7
AVI5	2.5	2.6	2.9	2.8	2.7
mean	2.8	3.0	3.1	3.1	

Table 4: *Comparison of the Synthesis Quality of the Different Synthesizers of our System (ANOVA - rows: p=0.000183; columns: p=0.02842)*

On the assumption that the relative synthesis quality among the synthesizers of our system remains the same across languages, we decided to use multi-level (or hierarchical) unit synthesis for the Blizzard Challenge this year. Note that for the Blizzard Challenge, 3 levels, namely word, syllable and diphone, were used.

5. Results and Discussion

Figures 3 and 4 show the results of voices A and B, respectively. Our system is identified by the letter G. System A is actually natural recordings of the same speaker as in the speech database. Our calculation of the mean MOS values of all participating systems reported in these sections does not include systems A, or the two benchmark systems, B and C. System B is a Festival Benchmark system. This is a standard Festival unit-selection voice built using the same method as used in the CSTR entry for Blizzard 2007 [9]. System C is an HTS Benchmark system. This is a standard speaker-dependent HMM-based voice, built using a similar method to the HTS entry for Blizzard 2005 [20]. Statistically significant differences among systems were identified by applying Wilcoxon's signed rank with Bonferroni correction [21].

For voice A, the MOS indicates reasonable quality for our system (mean MOS: 2.8, which is just below the mean of the participating systems, 2.9). However, the quality of our voice A is not significantly different from that of the following

systems: C, E, F, H, L, M, Q, T and V. Our voice B was rated slightly better by listeners (mean MOS: 2.9). The mean score on the quality of voice B of all systems was 2.8. The quality of our voice B is not significantly different from that of systems B, C, F, L, M, O and T.

When comparing the quality of our voices A and B, the results are surprising, since voice B uses only a small subset of the data as in voice A. Based on the median MOS of those utterances which were used to test both voices A and B, we did not find any significant difference between our two voices (Wilcoxon's signed rank: $p = 0.23$). However, in half of these cases, voice B received a higher median MOS. When analyzing the selected units in syntheses, we found little overlap between our voices A and B even for those utterances which received the same median MOS for both voices.

We would expect that the similarity rating of the voice to the original recordings would be close to 5, but this is not the case. Raters probably also listened to the naturalness of syntheses when judging on similarity. Our system performed somehow better on similarity than naturalness (MOS for our voice A: 3.1 c.f. mean: 3.0; MOS for our voice B: 3.0 c.f. mean: 2.8).

Our system did not perform very well on the semantically unpredictable sentences since our mean word error rate (WER) for both voices is 45 %, which is slightly higher than the means (voice A: 40%, voice B: 44%). It must be noted that even if recordings of natural speech are used, WER remains relatively high (voice A: 22%, voice B: 25%). For voice A, the WER of our system does not differ significantly from those of systems E, F, H, K, L, M, N, O, Q and R. The WER of our voice B is significantly lower than that of system R, but higher than those of systems C, Q, T and V.

There is still room for improving our segmentation. The models for segmentation were trained on the Arctic subset of the data only. The DSSP synthesizer also contains many hand-tuned weights but current settings might be sub-optimal. Algorithms for training weights automatically could help improve the system. We are further analyzing the data to find other ways to improve.

6. Conclusions

In this paper, we described the DSSP synthesizer and the two UK English voices submitted to the Blizzard Challenge. Entering the challenge provided us with positive experience, resulted in the development of a UK English front-end and brought about various improvements to our synthesizer. The results of our syntheses indicate reasonable quality, especially considering that the DSSP synthesizer is still under development. Further work will focus on the improvements on quality, robustness and speed, and fully automating the voice-building process.

7. Acknowledgements

The research in this paper was supported by the IWT project SPACE (SBO/040102): SPeech Algorithms for Clinical and Educational applications [2] and a research grant from the Faculty of Engineering Science, Vrije Universiteit Brussel.

8. References

[1] Black, A. W., Tokuda, K. "The blizzard challenge - 2005: evaluating corpus-based speech synthesis on common datasets", In INTERSPEECH-2005, 77-80.

[2] http://www.esat.kuleuven.be/~spch/projects/SPACE

[3] Mostow, J., Aist., G. "Evaluating tutors that listen: An overview of project listen. In K. Forbus and P. Feltovich, editors, Smart Machines in Education: The coming revolution in educational technology., pages 169 – 234. MIT/AAAI Press, 2001.

[4] B. Wise, R. C. S. van Vuuren, S. Schwartz, L. Snyder, N. Ngampatipatpong, and J. Tuantranont. Learning to read with a virtual tutor: Foundations to literacy. In C. K. Kinzer and L. Verhoeven, editors, Interactive Literacy Education: Facilitating Literacy Environments Through Technology. Erlbaum Publishers, Mahway, NJ, 2005.

[5] A. J. Hunt and A. W. Black, "Unit Selection in a Concatenative Speech Synthesis System Using a Large Speech Database", in Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing 1996, pages 373-376, Atlanta, May 1996

[6] http://www.festvox.org

[7] P. Taylor and R. Caley and A.W. Black and S. King. Edinburgh Speech Tools Library, System Documentation. Centre for Speech Technology, University of Edinburgh, 1.2 edition, June 1999.

[8] Clark, Robert A. J. / Richmond, Korin / King, Simon (2004): "Festival 2 - build your own general purpose unit selection speech synthesiser", In SSW5-2004, 173-178.

[9] K. Richmond, V. Strom, R. Clark, J. Yamagishi, and S. Fitt. Festival multisyn voices for the 2007 blizzard challenge. In Proc. Blizzard Challenge Workshop (in Proc. SSW6), Bonn, Germany, August 2007

[10] Susan Fitt and Stephen Isard, "Synthesis of regional English using a keyword lexicon", in Proc. Eurospeech '99, Budapest, 1999, vol. 2, pp. 823-826.

[11] Daelemans, and A. Van den Bosch, "Memory-Based Language Processing", 2005, Cambridge, UK, Cambridge University Press

[12] Van der Sloot, K. (2007). TiMBL: Tilburg Memory Based Learner, version 6.1, API Guide. ILK Research Group Technical Report Series no. 07-09.

[13] M.C. Viana, L.C. Oliveira, and A.I. Mata, "Prosodic phrasing: Machine and human evaluation," in Proceedings 4th ISCA Tutorial and Research Workshop on Speech Synthesis, Perthshire, Scotland, 2001.

[14] Silverman, K., Beckman, M., Pierrehumbert, J., Ostendorf, M., Wightman, C., Price, P., AND Hirshberg, J. ToBI: A standard scheme for labelling prosody. In Proceedings of the International Conference on Speech and Language Processing 1992, 1992

[15] Busser, B., Daelemans, W., Van Den Bosch, A. Predicting phrase breaks with memory-based learning. In 4th ISCA Tutorial and Research Workshop on Speech Synthesis (2001).

[16] Latacz, L., Kong, Y. O., W. Verhelst, "Unit Selection Synthesis Using Long Non-Uniform Units and Phonemic Identity Matching", in Proceedings 6th ISCA Workshop on Speech Synthesis (SSW-6), Bonn, Germany, August 22-24 2007

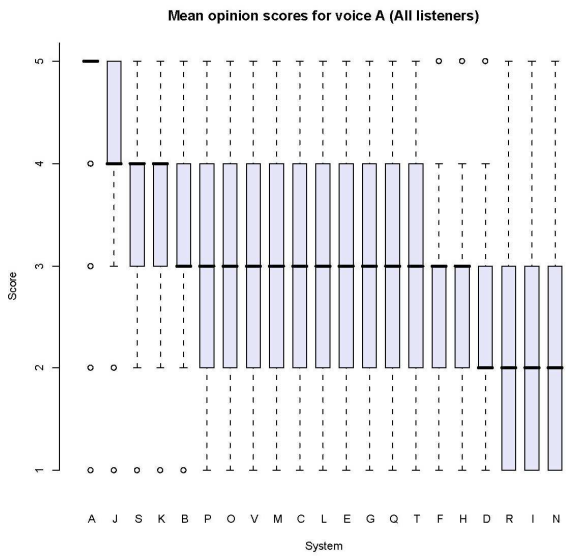
[17] Mattheyses, W., Latacz, L., Kong, Y.O., Werner Verhelst, "A Flemish Voice for the Nextens Text-To-Speech System", IS-LTC-06, Lublijana, Slovenia, October 2006.

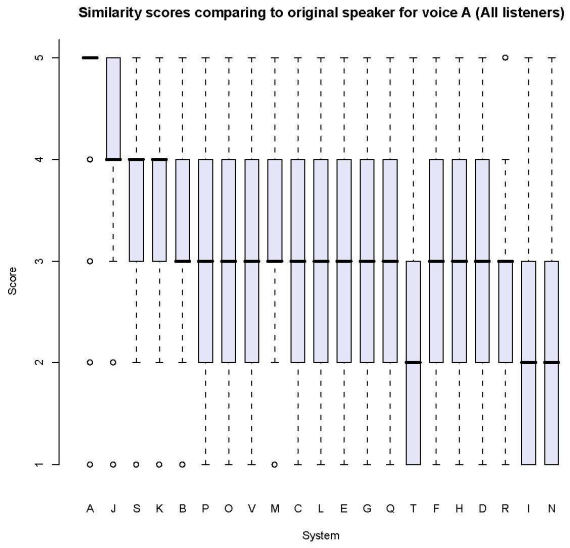
[18] J.P.H. van Santen and J. Hirschberg, "Segmental effects on timing and height of pitch contours," in ICSLP, Yokohama, 1994, vol. 2, pp. 719–722.

[19] Visser, J., Van Laarhoven, A. and Ter Beek, A. "AVI toetsenpakket. Handleiding", 1994.

[20] Zen, Heiga / Toda, Tomoki (2005): "An overview of Nitech HMM-based Speech Synthesis System for Blizzard Challenge 2005", In Interspeech 2005, 93-96.

[21] Robert A. J. Clark, Monika Podsiadlo, Mark Fraser, Catherine Mayo, Simon King. "Statistical Analysis of the Blizzard Challenge 2007 Listening Test Results," Proc. Blizzard Challenge Workshop, 2007, Bonn, Germany





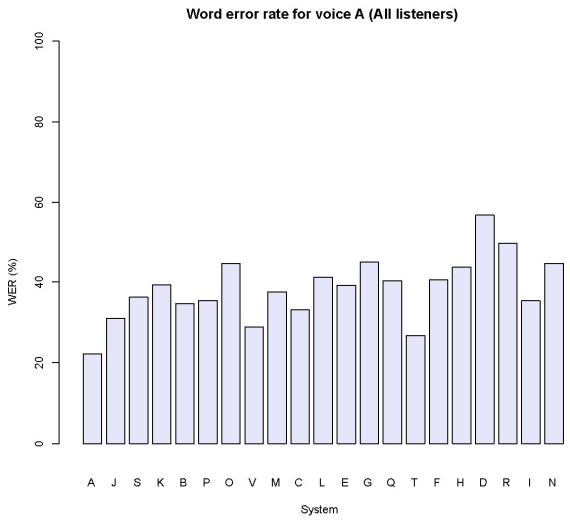
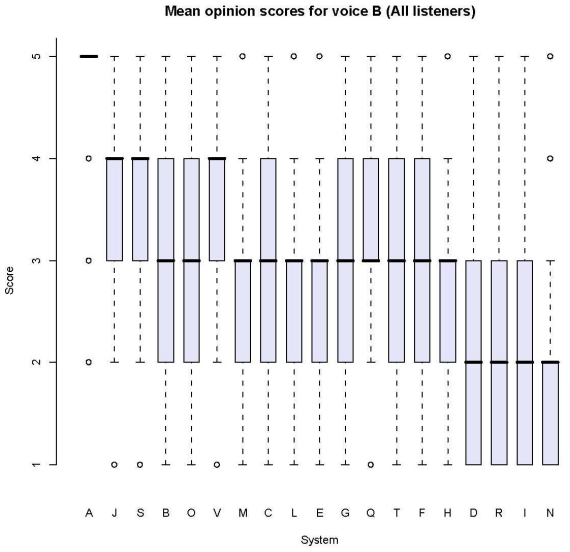
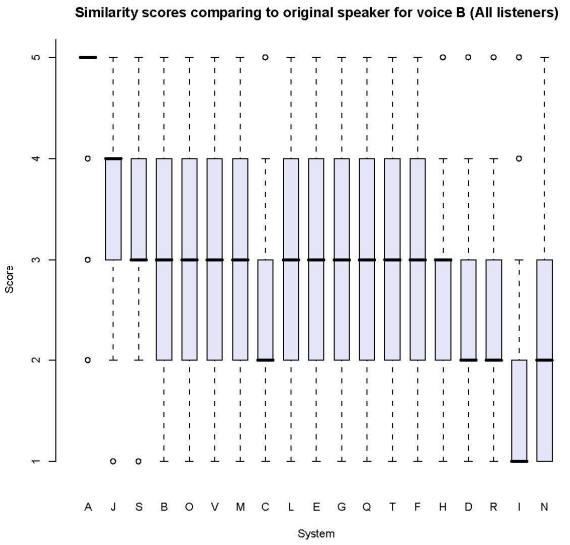


Figure 3: *Results for Voice A*





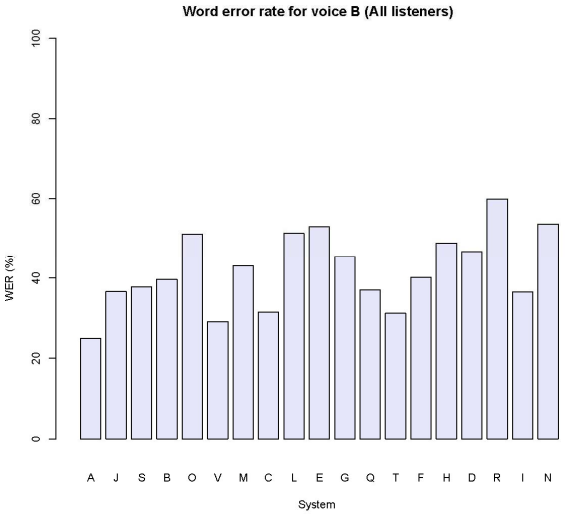


Figure 4: *Results for Voice B*