OpenMary –
Open Source Unit Selection as the Basis for Research on Expressive Synthesis

Marc Schröder, Anna Hunecke, Sacha Krstulović

DFKI GmbH, Saarbrücken, Germany
`firstname.lastname@dfki.de`

Abstract

This paper describes the unit selection component of the open source text-to-speech system OpenMary. It is a generic unit selection component with parameterisable target and join cost functions followed by optional signal post-processing, and it has been created for the Blizzard challenge 2006. We describe the creation process, the core properties of the system and the preparation of a synthesis voice from the Blizzard data. We also point out the directions in which we are currently developing the system towards expressive unit selection.

Index Terms: speech synthesis, unit selection, expressive speech, target cost, signal processing

1. Introduction and background

The text-to-speech (TTS) synthesis platform MARY [1] is a "Modular Architecture for Research on speech sYnthesis", based on a robust client-server architecture written in Java. This architecture is modular, so that new processing modules or module chains for existing or new languages can be easily added as Java code or as native binaries. The main data representation format within the system is an XML language called MaryXML.

Originally developed for German, the platform has been extended to English and, as the result of a student project, Tibetan. It has been used to implement a flexible approach to expressive synthesis [2], in which gradual changes of emotional states, represented as emotion dimensions, are realised as gradual changes of acoustic parameters using diphone synthesis.

In early 2006, we released the core of the MARY system as open source – the OpenMary system. It has been deployed and tested on Windows, Linux and MacOS X. An installer can be downloaded from `http://mary.dfki.de`; the source code is available via a software development environment based on Trac and Subversion at `http://mary.opendfki.de`.

The current release version 3.0 of OpenMary contains some preliminary unit selection code, through an implementation of the cluster unit selection algorithm [3] inherited from Flite [4] via its Java port FreeTTS. Using this code and the Festvox toolkit, we have created several limited domain voices, including an expressive football announcer. This voice, consisting of a moderate and an excited database produced by the same speaker, is based on the same approach as previous work on expressive unit selection [5, 6, 7]: Several databases are produced by the same speaker, each in a different expressive style, and the expressivity in the synthetic speech results from the choice of the appropriate database at runtime.

In our current work in the projects HUMAINE (`http://emotion-research.net`) and PAVOQUE ("PArametrisation of prosody and VOice QUality for concatenative synthesis in view of Emotion expression" – `http://mary.dfki.de/pavoque`), we aim at increasing the flexibility of expressive unit selection synthesis, by combining selection based on expressive targets with signal modification. As a basis for this work, we require a generic unit selection system. The amount of control over the unit selection process provided by the cluster unit selection algorithm is too limited for our purposes; in particular, the suitability of units for a given synthetic target is assessed exclusively through a Classification and Regression Tree (CART) making binary yes/no decisions at its decision nodes. In contrast, generic unit selection, as it is implemented, e.g., in Festival 2 [8], considers a target cost function jointly with a join cost function in the dynamic programming step of unit selection. One main advantage of this approach is that target costs can be "traded against" join costs, i.e. a unit that does not fit the target very well may be selected if this can avoid very high join costs. A second advantage is that the target cost function is capable of including continuous feature cost functions in addition to discrete class membership decisions, which is advantageous if we want to experiment with acoustic targets for expressivity. Finally, during experimentation, manually modifying feature weights in the target cost function is easier than re-training a CART selection tree.

With these research directions in mind, we implemented a generic unit selection algorithm and submitted it to the Blizzard challenge, as described in the main body of this paper. First, we describe the implementation, giving some detail about the preselection, target cost, join cost, concatenation and signal post-processing methods used in the system, and describing the creation of a voice database from the Blizzard data. Then, we discuss the system in the light of the Blizzard listening test results. Finally, we report on current work on preparing a release of a system implementing these algorithms in a robust and fast way, and outline the next steps towards expressive unit selection.

2. OpenMary unit selection for the Blizzard challenge 2006

2.1. System creation

To enable the OpenMary system to process unit selection voices, we started from the FreeTTS implementation (`http://freetts.sourceforge.net`). With FreeTTS, voices suited for cluster unit selection, such as the "clunit" version of the Arctic voices, can be built in Festvox and converted into a large binary file which can be read by FreeTTS. We imported the FreeTTS source code implementing the cluster unit selection algorithm into OpenMary, and we performed some re-organisation of the code. This reorganisation aimed at a more general and reusable class structure, so that OpenMary is not restricted to cluster unit selection, but can implement various unit selection algorithms in the future. The following paragraphs describe the significant divergences from FreeTTS.

Most notably, we have replaced the method of selecting units for a given target: a target cost function has been added to the dynamic programming phase, as motivated above, and the CARTs, which in FreeTTS are the only means to find a set of suitable candidate units for a target, are used in OpenMary as a mere preselection step. The target cost function uses the same feature set as used for training the CARTs, but applies weights to compute the similarity of a unit to a target. In OpenMary, the target features and weights are contained in a configuration file, so that the weights can be changed independently of the binary voice file. This makes tuning the weights more flexible.

In the original Festival and Flite code implementing the cluster unit selection algorithm, an "optimal coupling" algorithm can be used to compute the join costs between two units. This algorithm browses the context of the labelled unit boundaries until it finds the audio frames that best fit together. In OpenMary, our attempts to get the existing optimal coupling code to work yielded worse results than the default algorithm which joins units at the labelled boundaries. Therefore, we used the latter algorithm in the current system.

Regarding the synthesis of an utterance's waveform, we moved from a purely sequential frame-based synthesis to an overlap-add approach to joining the audio signals for different units, which seems to reduce audible discontinuities at the joins. We also added a first step in the direction of signal post-processing, in the form of an algorithm smoothing the F0 curve.

More detail is given in the next section.

2.2. System details

The system performs a selection of phone-sized units, consisting of a CART-based pre-selection of candidates, a dynamic programming phase in which target and join costs determine an optimal unit chain, and a concatenation and signal post-processing stage in which the audio for the selected units is generated, concatenated and post-processed to smooth the F0 contour.

2.2.1. *Pre-selection*

We use CART trees for pre-selecting a medium-sized set of potentially suitable candidate units. In the classical cluster unit algorithm, the CARTs select small sets of 20-or-so candidates. This strict selection in terms of linguistic and phonetic target features entails a risk of preventing possibly interesting units from participating in the dynamic programming phase. In contrast, our system uses CARTs to pre-select larger sets of 500 candidate units. This pre-selection is a means to reduce the complexity of the dynamic programming, by imposing only loose linguistic and phonetic constraints during the pre-selection.

It remains an open question whether this pre-selection is best made using trees that are automatically trained based on the criterion of acoustic homogeneity of sub-clusters [3], as in the cluster unit selection algorithm, or whether manually defined preselection criteria would be more appropriate.

2.2.2. *Target costs*

In selecting the list of features to use as target costs, we started with the set of features used for training the CART trees for the Arctic voices in Festvox. These features are mostly phonetic (e.g. vowel height, stress, pitch etc.). We have modelled the weights after Festival *multisyn*, and adapted them to our current system where necessary. For example, the target duration and F0 values, which are given considerable weight in Festival, could not be used in our current system because our F0 and duration models predict generic values, which are not adapted to any specific voice database.

However, we have prepared the code for the use of such continuous target features, so that they can be used as soon as appropriate targets can be defined. We consider this feature to be of essence in view of defining expressive targets, such as emotion-related changes in prosody or spectral characteristics.

As a result of having neither sentence type nor F0 values in the available feature set, the selection algorithm had no means of predicting question intonation. We therefore added ToBI accents and boundary tones to the set of features, annotating ToBI tones in the database automatically using our system's predictions. Even if these may not always be the ones realised by the database speaker, they are suitable for distinguishing questions from statements.

2.2.3. *Join costs*

Join costs measure spectral and F0 continuity between adjacent units. For measuring spectral continuity, we re-used the existing FreeTTS implementation which computes the Euclidian distance of the second to twelfth Mel-frequency cepstral coefficients (MFCCs). This choice of feature set will be the topic of future investigation; open questions include the issue of discarding or not discarding the first MFCC, the possibility to extend the feature set with delta features, and the possible addition of other parameters such as energy.

The F0 continuity measure was re-programmed in view of the application of a smoothing algorithm. The F0 median over five periods is computed at unit boundaries. Given the fact that small F0 deviations can be fixed by the smoothing algorithm (see section 2.2.6 below), a step function was used as the cost function, giving zero cost to small deviations, and large cost to deviations larger than a given threshold (20%).

2.2.4. *Dynamic programming*

For the sake of time efficiency, a beam search was used in the dynamic programming phase, by pruning the number of intermediate paths down to the most likely ones. In computing the cost associated with adding a unit to a given path, the target costs for that unit and the join costs for appending the unit to the path were added. We aimed at a globally equal contribution of target and join costs to the overall cost, which was achieved by selecting the weights of the individual feature costs according to appropriate heuristics.

2.2.5. *Unit concatenation*

The audio data is created using pitch-synchronous residual-excited LPC synthesis. The individual units are re-synthesised separately, and concatenated using overlap-add of one period at the unit boundary, with a Hann window. This approach helps reducing discontinuities at join points, compared to the previous implementation taken from FreeTTS, where all residuals and LPC frames were re-synthesised in one loop, causing audible clicks at some unit boundaries.

2.2.6. *Signal post-processing*

On the resulting audio, we apply a shift-only F0 smoothing algorithm closely modelled after Bozkurt et al.'s work [9]. Their idea was to maintain micro-prosody while smoothing by shifting entire units up or down, rather than reducing intonation to stylised curves at join points. We re-implemented their algorithm, which is based on solving a system of equations which simultaneously minimises pitch discontinuities and unit shifts. A "shift propensity" parameter determines the relative importance of minimising shifts vs. minimising discontinuities. In our implementation, these shifts are deltas in the log F0 domain, i.e. F0 factors in the Hertz domain. We are not sure whether this is the same as in the original work by Bozkurt et al. – we could not find this detail in their paper –, but given the fact that humans perceive log F0, it seemed like an appropriate choice. PSOLA [10] is used to realise the requested pitch factors.

2.3. Voice creation with the Blizzard data

The creation of the unit database for the Blizzard voice was conducted with Festvox, Sphinx, FreeTTS and OpenMary. The actual voice creation was performed fully automatically; most work was invested in getting the tools to work with the large amount of data.

As a first practical step, the audio data and the transcriptions were fitted to a directory structure which complies with the directory structure assumed in the Festvox scripts. Then, in order to ensure consistency between the corpus annotations and the OpenMary predictions, we discarded the automatically generated phonetic transcriptions and re-ran the automatic transcription with the OpenMary lexicon and tools.

The phonetic labels created from the OpenMary transcriptions were subsequently force-aligned with the audio recordings using CMUSphinx with the Sphinxtrain scripts coming with Festvox. This labelling phase took most of the time, as we fist had problems with units that were too long – they were actually spanning entire words. This turned out to be due to the fact that the initial utterances were created with the assumption that each wave file contained just one sentence, which is not the case for all files in the test data. After this was corrected and the labelling redone, the overall quality of the labels seemed acceptable. We did not manual inspect, select or correct any of the labelled units; however, an automatic filter was included which discarded all units that were shorter than 2 or longer than 50 pitch periods.

At several steps, the Festvox tools had to be used in non-standard ways to accommodate the important size of the Blizzard database. For example, the CART-building algorithm, implemented in the EST Wagon tool, ran into a memory problem when creating distance tables for Schwa and pause units; this was circumvented by using only the first 13300 instances of each unit, which resulted in distance table files of less than 2 GB size. Similarly, the shell scripts taking the list of audio files as input failed with "Argument list too long" errors, and needed to be run in several chunks.

The last step in voice creation was to import the voice into FreeTTS and from there to OpenMary. We are now working on a solution to avoid the detour over FreeTTS.

As the system was still under heavy development during the preparation of the voice database, we only created one voice, from the full data set of 4273 utterances; we did not create a voice from the Arctic subset.

2.4. Listening test results and discussion

The Blizzard challenge organisers carried out listening tests with the voices created by all Blizzard challenge participants, using three groups of listeners: speech experts (S), undergraduates (U), and random participants on the web (R). Procedure and overall results are presented elsewhere in these proceedings; here, we only discuss the results for our system compared to the average of all systems built from the full data set.

Our system was rated with a word error rate (WER) of 23.9 (S), 24.7 (U) and 33.9 (R), which is slightly worse than the average of all voices built from the full data set: 23.5 (S), 22.3 (U), and 31.1 (R). Mean opinion scores (MOS) for our system were 2.5 (S), 2.5 (U) and 2.4 (R), below the average of all voices built from the full data set: 2.9 (S), 3.0 (U), 2.8 (R).

Given the preliminary state of our system and the fact that it was our first participation to the Blizzard challenge, we consider these results as promising, even if they cannot yet be considered totally satisfactory. Improvements to our system can be expected at various levels. Regarding database labelling, we are not certain that we get optimal results with the out-of-the-box Sphinxtrain algorithms. For example, an algorithm capable of taking into account various pronunciation options for a given word could be expected to improve the labelling; manual correction of the label chain before doing forced alignment would be desirable but seems unrealistic in the context of this challenge, given the size of the database and the licensing issues preventing the use of the resulting voice in other contexts. Regarding the creation of a symbolic+parametric representation of a synthesis target, we assume that the quality would be improved if explicit duration and F0 targets appropriate for the given voice could be computed and used as target features and/or in signal post-processing. The list of linguistic target features will need careful scrutiny, and important but missing features such as the sentence type (question vs. statement) must be added. The choice of the unit type (phones at the moment) should also be re-considered – diphones were invented precisely because phone boundaries were found *not* to be the best places to cut a speech signal.

Finally, it is an open question whether the F0 smoothing algorithm which we used actually improved the overall impression, or if MOS ratings would have been better if we had not performed any F0 smoothing. The fact that our MOS ratings are further below the average than our WER ratings can be interpreted as an indication that this might be the case, although it would have been more interesting and conclusive to be able to compare these two variants in the context of the Blizzard evaluation.

3. Towards expressive unit selection

As stated in the previous sections, the developments currently under way on the OpenMary TTS system include more flexible and efficient database formats and more flexible selection and concatenation processes, in addition to providing an open source and portable synthesis runtime to the research community. The common goal of these developments is to make the OpenMary platform more suitable to testing new ideas, in particular for enhancing the expressivity of state-of-the-art speech synthesis.

Starting from a selection-based synthesis framework, the rationale is to enable the system to characterise and adapt the expressivity of the speech units. The expressivity is primarily understood here as a set of characteristics of prosody and voice quality. In view of selection, such expressive characteristics should be described in a way that is sufficiently independent from linguistic and phonetic features, so that it becomes possible to "trade" high expressivity against high linguistic/phonetic relevance, e.g. by selecting a unit with the intended expressivity even though it does not stem from the right linguistic context.

To enable such a selection of expressively relevant units, we plan to introduce an explicit Acoustic Target Cost (ATC) function into the dynamic programming phase of the selection, in addition to the classical Linguistic Target Cost (LTC) and Join Cost (JC) functions. The definition of this new Acoustic Target Cost function raises several research issues, regarding:

1. how to measure the voice quality (e.g., the vocal effort) through some reliable acoustic cues;
2. how to specify the expressive characteristics of prosody independently of the linguistic contents;
3. how to calibrate and weight the contribution of the ATC to the overall selection process.

To help tackling these issues, we will record an acted speech database which will cover a range of expressive speaking styles. Using this experimental material, we will compare the ability of several acoustical measurements to characterise the prosody and the voice quality, with the view of including these measurements into the definition of the ATC function. Regarding voice quality, for instance, we forecast that measurements based on some characteristics of the glottal flow [11] or, more coarsely, of the LPC residual, may act as relevant cues.

In addition to the introduction of the ATC into the selection process, we plan to introduce expression-related signal modification methods into the concatenation phase, to better adapt the selected units to a target expression style – in the same way that join-related F0 corrections have been applied so far. Here again, our new database will help us assess the relevance of state-of-the-art speech modification methods with respect to the modification of expressivity, for modifying the prosody and, experimentally, the voice quality. The degree of performance of the most relevant modification methods may in turn lead to the definition of a notion of modification cost which could become a part of the ATC.

4. Conclusions

We have presented a first version of unit selection with OpenMary, an open source synthesis platform written in Java. The listening tests carried out as part of the Blizzard challenge have shown that the resulting quality is not far below the average of the systems that participated in the challenge. Given the preliminary state of our system and the obvious improvements that can be made, this is an encouraging result.

When the current process of stabilising the implementation in view of more speed and robustness is completed, the system will be made available for download with a number of German [12] and English [13] voices.

The system will serve as the basis for research in adding parametrisation capabilities to unit selection in view of expressive speech synthesis, combining selection of expressive units with signal modification aspects.

5. Acknowledgements

This work was supported by the DFG project PAVOQUE and by the EC project HUMAINE (IST-507422).

6. References

[1] M. Schröder and J. Trouvain, "The German text-to-speech synthesis system MARY: A tool for research, development and teaching," *International Journal of Speech Technology*, vol. 6, pp. 365–377, 2003.

[2] M. Schröder, "Expressing degree of activation in synthetic speech," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 4, pp. 1128–1136, 2006.

[3] A. W. Black and P. Taylor, "Automatically clustering similar units for unit selection in speech synthesis," in *Proceedings of Eurospeech 1997*, Rhodes/Athens, Greece, 1997, vol. 2, pp. 601–604.

[4] A. W. Black and K. A. Lenzo, "Flite: a small fast run-time synthesis engine," in *Proceedings of the 4th ISCA Workshop on Speech Synthesis*, Perthshire, Scotland, 2001.

[5] A. Iida and N. Campbell, "Speech database design for a concatenative text-to-speech synthesis system for individuals with communication disorders," *International Journal of Speech Technology*, vol. 6, pp. 379–392, 2003.

[6] W. L. Johnson, S. S. Narayanan, R. Whitney, R. Das, M. Bulut, and C. LaBore, "Limited domain synthesis of expressive military speech for animated characters," in *Proceedings of the 7th International Conference on Spoken Language Processing*, Denver, Colorado, USA, 2002.

[7] J. F. Pitrelli, R. Bakis, E. M. Eide, R. Fernandez, W. Hamza, and M. A. Picheny, "The IBM expressive text-to-speech synthesis system for American English," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 4, pp. 1099–1108, 2006.

[8] R. Clark, K. Richmond, and S. King, "Festival 2 – build your own general purpose unit selection speech synthesiser," in *Proceedings of 5th ISCA Speech Synthesis Workshop*, Pittsburgh, PA, USA, 2004, pp. 173–178.

[9] B. Bozkurt, T. Dutoit, R. Prudon, C. d'Alessandro, and V. Pagel, "Reducing discontinuities at synthesis time for corpus-based speech synthesis," in *Text to Speech Synthesis: New Paradigms and Advances*, S. Narayanan and A. Alwan, Eds., pp. 1–17. Prentice Hall, Upper Saddle River, NJ, USA, 2004.

[10] E. Moulines and F. Charpentier, "Pitch synchronous waveform processing techniques for text-to-speech synthesis using diphones," *Speech Communication*, vol. 9, pp. 453–467, 1990.

[11] C. d'Alessandro and B. Doval, "Voice quality modification for emotional speech synthesis," in *Proc. Eurospeech' 03*, Geneva, Switzerland, 2003, pp. 1653–1656.

[12] T. Ellbogen, F. Schiel, and A. Steffen, "The BITS speech synthesis corpus for German," in *Proc. 4th Conference on Language Resources and Evaluation (LREC)*, Lisbon, Portugal, 2004, pp. 2091–2094.

[13] J. Kominek and A. W. Black, "CMU ARCTIC databases for speech synthesis," Tech. Rep. CMU-LTI-03-177, Carnegie Mellon University, 2003.