

SEGMENT SELECTION IN THE L&H REALSPEAK LABORATORY TTS SYSTEM

Geert Coorman, Justin Fackrell, Peter Rutten, and Bert Van Coile

Lernout & Hauspie Speech Products NV, Flanders Language Valley 50, B-8900 Ieper, Belgium

(geert.coorman, justin.fackrell, peter.rutten, bert.vancoile)@lhs.be

ABSTRACT

The L&H RealSpeak Laboratory TTS (RSLab) system is a corpus based speech synthesis system comprising components that deal with linguistic processing, prosody prediction, segment selection, concatenation and modification. In this paper we focus on the segment selection process.

During segment selection, the units in a large database of speech are scored with a cost according to their prosodic/phonetic mismatch with the target description of the utterance to be synthesized. This prosodic/phonetic cost is computed on the basis of a combination of symbolic and numeric features. The candidate units from the speech database are then evaluated for the ease with which they can be concatenated. A dynamic programming algorithm, using additive costs, is used to find the optimal path of candidates to represent the spoken utterance. The chosen segments are then concatenated in the time domain to yield a smooth-sounding speech signal, with natural-sounding prosody.

One of the keys to the success of the segment selection component is the context dependent choice of cost functions, and the method of combining the costs from the various features. The RSLab system makes use of a family of complex cost functions that allows linguistic and perceptual knowledge to be incorporated in the segment selection process.

1 INTRODUCTION

The majority of synthesizers that have found use in commercial applications use the same basic principle – that of concatenation of stored speech units. After many years of determined effort by the research community the goal to produce natural sounding synthetic speech for any input, is finally within reach. A key factor in the success of concatenative systems is the increase in processing power and the availability of cheaper memory. Concatenative synthesis systems can now use thousands of speech units, and carry out a sophisticated selection of these units at run-time. Such systems are referred to as corpus based speech synthesis systems (e.g. [1], [3], [4], [6], [7]).

In the RSLab system, the linguistic modules and the feature extractor convert an input text into a multi-layer internal data sequence that describes the prosody and phonetic content of the target utterance as shown in Figure 1. The same multi-layer

description is used to represent the speech signals in the speech segment database. This database is consulted by a diphone-oriented unit selector. The task of the unit selector is to determine the best sequence of diphones from the database that can be concatenated to produce the target utterance. It does so by taking into account how well the diphones match the target diphone descriptors generated by the feature extractor, and how well they can be joined together in order to obtain natural sounding speech. Since the concatenation cost is zero for diphones that are next to each other in a recording, it is likely that polyphones are selected from the speech database.

The next element in the synthesizer is a polyphone concatenator, which joins the polyphone waveforms with minimal audible distortion.

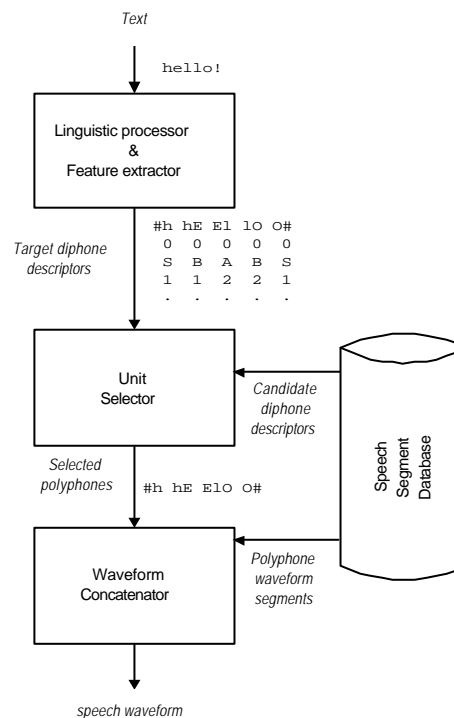


Figure 1 Basic building blocks of the RSLab corpus based speech synthesis system.

Since the unit selector operates in a diphone-oriented way, the data description is organized accordingly. Each diphone (target or candidate) is described by a diphone feature vector.

For each diphone in the target description the unit selector initially retrieves the feature vectors of a large number of diphone candidates that each match the diphone in the target description in a broad way. Each of these candidate diphones is scored by a multi-dimensional cost function that reflects how well it matches the target diphone – this is the *unit cost*. This cost is computed on the basis of a combination of symbolic and numeric features. A *concatenation cost* is calculated between each pair of diphone candidates that can be joined. This too is calculated by a multi-dimensional cost function, but in this case the cost reflects the cost of joining together two candidates. Most features used in the calculation of the concatenation cost are directly related to some acoustical property of the candidates. The *Dynamic Programming* (DP) [2] algorithm finds the lowest cost path through all the possible sequences of candidate diphones, taking into account both the unit and the concatenation costs.

2 COST FUNCTIONS FOR SEGMENT SELECTION

The main goal of the unit selector is finding the right combinations of units in such a way that the perceptual difference between the expected and the realized waveform is as small as possible. Therefore the DP cost must reflect the perceptual distance between a speech waveform determined by unit selection and a speech waveform as spoken by a human speaker. We assume that the cost operator is an additive operator that takes as input perceptual distances. The assumption that the arguments obey a distance metric means that the sum of the individual perceptual distances will be larger than the real perceptual distance (i.e. triangle inequality). In other words, the cost function that is being used in the DP scheme is a worst case estimator for the real perceptual cost.

2.1 Feature Vectors

A significant factor that contributes to the quality of the system is the level of detail of the description that is used to represent the speech signals in the database. For TTS systems these descriptions should be compatible with descriptions which can be generated from input text during synthesis.

We base the initial choice of the description on a study of the language and on our experience with existing systems for related languages. During the development process, the description is modified as part of an iterative procedure to improve the quality of the system.

Each diphone (target or candidate) is described by a diphone feature vector. This vector contains many elements. Some of them are numeric such as syllable position in phrase, prominence, pitch values, duration values, spectral vectors at diphone boundaries. Others are symbolic such as diphone identity and phonetic context. The representation of the features is chosen in such a way that the distance functions return

perceptually motivated results (e.g. semitones for pitch and mel-cepstrum for spectral vector).

Figure 2 illustrates how the unit and concatenation costs are calculated. The figure shows the three feature vectors that play a role in the evaluation of a diphone candidate given a certain predecessor diphone:

- the feature vector of the candidate itself,
- the feature vector of the target diphone,
- the feature vector of the predecessor.

In the following sections we will describe the cost function calculation, as depicted in Figure 2, in more detail.

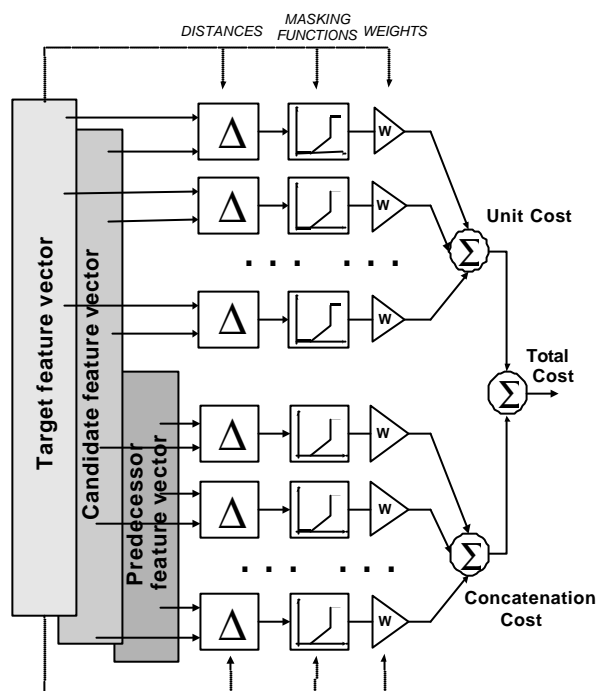


Figure 2 Overview of the cost calculation process

2.2 Distance calculation

The Δ symbols in Figure 2 represent distance functions that compare two values of the same feature and produce a distance value as output. These functions measure either the degree of match between a target feature and the corresponding candidate feature (for the unit cost) or the degree of match between a feature for two adjacent candidates (for the concatenation cost). There are three types of distance measure:

1. *Symbolic distance measures.* In its simplest form the distance between two symbolic values is 0 if the values are the same, and 1 otherwise. However, to incorporate linguistic knowledge in the system many symbolic features use more sophisticated distance functions that allow several degrees of precision in matching. For example, for the unit cost calculation, one feature which may be important is the

left-neighbor phone identity: if the target for this feature is /p/ a candidate with left neighbor /b/ typically represents a better match than one with left neighbor /s/. This sort of linguistic knowledge can be incorporated in the distance measure by using tables that specify the distance between each pair of symbolic feature values.

2. *Scalar distance measures.* These can be absolute difference functions (appropriate for pitch for example), or lop-sided functions (appropriate for duration, for example, in which duration shortening is penalized more than duration lengthening).
3. *Vector distance measures.* The calculation of the distance between multi-dimensional features such as the spectral envelope is time consuming. In order to reduce the computational complexity a combination matrix –containing the distances only- could be calculated in advance for all possible combinations occurring at the unit boundaries. Unfortunately the combination matrix grows quadratically with the size of the database. An efficient solution is to vector quantize (VQ) the feature space. Based on the results of this VQ, a distance matrix can be constructed whose size can be kept constant independently of the database size. Because the phoneme distribution is far from uniform it is appropriate to VQ on a phoneme-by-phoneme basis instead of performing a uniform VQ over the whole database. This process results in a set of phoneme-dependent VQ distance matrices.

2.3 Masking function

The unit selection is based on a DP search for the path with the lowest cumulative cost. The additive DP scheme, combined with the distance measures as described above, is not the ideal method to guarantee high quality synthesis. For example a path that contains only one big cost in combination with many small costs could be preferred over a path containing only acceptable costs. In order to avoid such behavior we apply a set of masking functions to the calculated distances as shown in Figure 2.

The masking functions are designed in such a way to facilitate the screening-out of “bad” unit candidates in the segment selection process, and to introduce a degree of tolerance for “near-miss” unit candidates. The masking function uses two thresholds: the transparency threshold and the quality threshold.

2.3.1 Transparency Threshold

It is well known that we are gifted with a hearing system that allows some tolerance on the perception of different stimuli [8]. The human auditory system will generally not perceive a difference between two utterances with two slightly different feature realizations. For example a small pitch discontinuity between two utterances will not be noticed if the pitch difference is below a certain threshold. The smaller the perceptual distance associated with feature values, the smaller the probability of hearing audible artifacts. We take this into

account in a distance masking function by introducing a *transparency threshold* T_T^k for the distance value associated with feature k . The masking function will map all perceptual distances smaller than the T_T^k to zero. This prevents the DP algorithm accumulating many small costs that would be tolerated by the listener.

The transparency thresholds used in the calculation of the concatenation cost can be determined by off-line perceptual experiments. For some acoustic features the region of transparency defined by T_T^k can be further increased by using high-quality signal modification algorithms. For example, the transparency threshold for pitch discontinuity is determined by the just noticeable pitch discontinuity plus the amount of pitch correction that can be done by the synthesizer without introducing audible artifacts.

2.3.2 Quality Threshold

The unit selection may lead to undesirable speech artifacts if the distance for a given feature exceeds a certain threshold. We will refer to this threshold as the *quality threshold* T_Q^k for feature k . The probability of selecting “bad” units (or combinations of units) can be minimized by increasing the distance to a very high value once the quality threshold is exceeded (See Figure 3). For example, if the pitch mismatch between two speech units exceeds the quality threshold, the cost becomes very high. As a result of this it is very unlikely that those diphones will be part of the best path.

It is advantageous to flatten the masking function above the quality threshold because in most practical circumstances it is useless to differentiate between different levels of unacceptable speech quality.

The calculation of the masking functions m_k can be made very efficient by using a piecewise linear function, shown in Figure 3.

$$m_k(\Delta) = \begin{cases} 0 & 0 \leq \Delta < T_T^k \\ \alpha(\Delta - T_T^k) & T_T^k \leq \Delta < T_Q^k \\ 1 & \Delta \geq T_Q^k \end{cases}$$

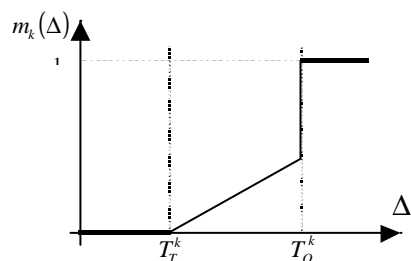


Figure 3 Piecewise linear masking function designed around the thresholds T_T^k and T_Q^k

2.4 Weights

The output of each masking function is multiplied by a weight value, as indicated in Figure 2, which describes the relative importance of each feature to the cost being calculated. While there has been interest in the research community in finding ways of automatically training weights [5] such approaches are hampered by a lack of good speech quality metrics. In our experience iterative fine-tuning guided by listening tests leads to better results.

2.5 Context Dependency

In describing the system of cost calculation above, it has been assumed that all features matter equally for all diphones. This is an oversimplification. To get around this, and to provide a way to introduce additional linguistic knowledge in the system, it is possible to specify context dependent rules that can affect the following components of the unit selection mechanism:

1. the distance measures. E.g. switch between different distance tables,
2. the masking functions.
3. the weights.

The rules are triggered by information in the target transcription – as indicated by the dotted lines in Figure 2. Many rules in our current system use the phonetic context, but context-dependent changes are also triggered by features such as sentence type (e.g. statement/question) or the prosody of the target.

The way in which linguistic knowledge has been incorporated in the RSLab system is best illustrated with some simple examples:

- *Coarticulation*. Because /r/ often colors vowels before and after it, a rule fires when a vowel in /r/-context is encountered. The rule increases the importance that the candidate units have the same phonetic context as the target unit.
- *Sonorants* are more sensitive to position and prominence than non-sonorants. Rules fire to increase the weight on position and prominence features for sonorants.
- *Pre-pausal lengthening*. It is well known that in many languages phonemes at the end of an utterance, i.e. the last syllable, tend to be longer than those elsewhere in an utterance. Therefore, when the dynamic programming algorithm searches for candidate speech units to synthesize the last syllable of an utterance, the segment duration

should preferably be longer and certainly not significantly shorter than specified in the target.

- *Sentence type*. It may be problematic to use segments extracted from a certain type of message in another type of message. Segments extracted from certain positions in isolated words or questions, could be inappropriate in statements, even though most features match. However, using units from statements in questions or isolated words, could be acceptable. This can be taken into account by using context dependent distance tables.

3 CONCLUSIONS

In this paper the unit selection component of the RSLab system has been described. This component has been successfully used for the development of TTS systems for 15 languages.

4 REFERENCES

- [1] M. Balestri, A. Pacchiotti, S. Quazza, P.L. Salza & S. Sandri, “*Choose the best to modify the least: a new generation concatenative synthesis system*,” Proc. Eurospeech ‘99, Budapest, Vol. 5, pp. 2291-2294, 1999.
- [2] R. Bellman, “*The theory of dynamic programming*,” Bulletin of the American Mathematical Society, 60, 503-515, 1954.
- [3] M. Beutnagel, A. Conkie & A.K. Syrdal, “*Diphone synthesis using unit selection*,” Proc. 3rd ESCA/COCOSDA International Workshop on Speech Synthesis, Jenolan Caves, pp. 185-190, 1998.
- [4] A.W. Black & N. Campbell, “*Optimizing selection of units from speech databases for concatenative synthesis*,” Proc. Eurospeech ‘95, Madrid, pp. 581-584, 1995.
- [5] A.J. Hunt & A.W. Black, “*Unit selection in a concatenative speech synthesis system using a large speech database*,” Proc. ICASSP ‘96, Atlanta, Vol. 1, pp. 373-376, 1996.
- [6] N. Iwahashi, N. Kaiki & Y. Sagisaka, “*Concatenative speech synthesis by minimum distortion criteria*,” Proc. ICASSP ‘92, San Francisco, Vol. 2, pp. 65-68, 1992.
- [7] P. Rutten, G. Coorman, J. Fackrell & B. Van Coile, “*Corpus based speech synthesis in the Lernout & Hauspie RealSpeak TTS system*,” Proc. IEE symposium on State-of-the-Art in Speech Synthesis, Savoy Place, London, pp. 16/1-16/7, 2000.

- [8] E. Zwicker & H. Fastl, "*Psychoacoustics. Facts and Models*," Springer Verlag, Heidelberg, New York, 1999.