APPLYING FALLBACK TO PROSODIC UNIT SELECTION FROM A SMALL IMITATION DATABASE

Joram Meron
Panasonic Speech Technology Lab., Santa Barbara, CA 93105
jmeron@research.panasonic.com

ABSTRACT

This paper presents an extension to a previous work [1], which used an imitation speech database and a prosodic unit selection algorithm, for improving the naturalness of synthesized speech.

The basic approach of the system is to combine a rule-generated prosody with a corpus based prosody module, trying to retain both the robustness of the rule prosody, and the naturalness of the human recorded speech units. This combination was achieved by using a database of imitation speech, enabling a higher level of annotation, which is used by a dynamic unit selection algorithm.

Although listeners have been shown to prefer the prosody generated with this method over that of the original rule generated prosody, the usual problems related to selection from an undersized training corpus were occasionally present.

Instead of increasing the size of the training database, a different solution is investigated here, which is to perform a controlled fallback to the rule prosody, but in a way which is compatible with the unit selection approach. The suggested method has a minimal effect on the required memory size and the amount of computation, and was shown to produce favorable results.

1. INTRODUCTION

Corpus based prosody generation methods have been shown to produce mostly natural-sounding prosody for TTS systems [2][3][4]. However, these methods are prone to produce unnatural prosody occasionally. Rule based prosody systems, on the other hand, achieve less natural prosody, but perform more consistently.

The Panasonic speech synthesizer, CyberTalk, is a small footprint, high quality synthesizer, aimed for use in embedded applications. One of its unique features is that it was designed to enable seamless integration between unlimited text synthesis, and pre-recorded messages, allowing either a new recording, or just a prosody template, to be implanted in a target sentence, over-riding the use of the default sound units and/or rule prosody. This ability to implant prosody templates into the synthesizer is instrumental for the work described here.

In a previous work[1], we have suggested a method to combine the robustness of the rule based method with a more natural (and speaker adaptive) corpus based method. This method is suited for our intended applications, as it does not require a significant increase in the needed memory size.

The prosody generation is achieved by using a database of imitation speech, on which a dynamic unit selection algorithm is run. The use of an imitation speech database solves some difficult problems associated with the collection and annotation of the training database, and allows a more accurate and more detailed annotation of the data, which improves the results of the unit selection process.

Although listeners showed a preference to the prosody generated by this selection method, there were still some problems with the consistency of the generated prosody. One obvious cause for these problems was the small size of the unit database. Instead of increasing the size of the database, a simpler solution is investigated - a controlled fallback to the rule prosody.

In section 2, a brief explanation of the basic prosody generation method and of the training database is given. Section 3 presents the extension to the basic method. Section 4 presents the results of a preliminary evaluation test, followed by conclusions, in section 5.

2. THE BASIC METHOD

In this section we will briefly review the prosody generation method which is described in [1], and the database that it, and this work, use (for full details see the original paper).

A basic motivation in the original work is the informal distinction between: 1) interpretation - the decision on the specific nuance that the utterance will convey (e.g. the types and locations of the intonation events), and 2) realization - the production of the final shape for these events.

Specifically - the interpretation is taken to be the output of the rule prosody-generation system (in the form of the set of produced intonation events and their parameters), and its synthetic realization is replaced by a natural one, obtained from a prosody unit database, by means of unit selection.

In order for the unit selection to produce good results, it needs a database of units with accurate and detailed annotation (in the above terms - an assigned interpretation). The difficult problem of annotation is solved by the use of imitation speech - having a human speaker imitate the synthesizer. In this way, the imitation can be automatically, and fairly accurately, annotated with an explicit interpretation (the rule-generated intonation events).

2.1. Database Construction

The imitation database consists of a set of 268 sentences, with between 10 to 18 words per sentence (avoiding long sentences which are more difficult to imitate correctly), including about 50 question, and 10 exclamation sentences. The total recording time for these sentences is relatively very short - about 12 minutes.

Our rule based prosody uses a tone sequence prosody model (related to ToBI [6]), consisting of accent, phrase, boundary, and auxiliary tones. Each of these tones can get various symbolic and
**Fig. 1.** The feature vector created for the middle syllable holds features for the current, and surrounding events and syllables. Filled circles represent syllables with events, blank ones represent syllables without events. The small blank circles on the top show the context used for syllable features, and the filled circles in the bottom show the context used for the event features.

quantitative values. Each of these tones is associated with one syllable (a syllable can have more than one tone associated with it).

The basic unit handled by the system is referred to as an intonation event. Each syllable, which is associated with a tone (or tones), is the nucleus of an intonation event. We construct a database by adding one entry for each of these intonation events. Each entry holds two types of features: Event features - which describe event properties, and Syllable features - which describe syllable properties. By having these two kinds of features we are able to describe both the local and the more global environment around each of the events. Each entry holds the 5 values before and after the current event/syllable - see figure 1. The features stored in each feature vector are:

- Event features: the type of tone (accent, phrase, boundary, or a combination in case one syllable is assigned more than one tone), part of speech (of respective word), the parameters of the event (type, and target amplitude), the declination value at the event, and the sentence type.
- Syllabic information: syllable segmental structure, syllable stress, part of speech, duration - synthetic and natural (obtained from the aligned recordings - see below), average F0 - synthetic and natural, and F0 slope.

After the training database is recorded, each recorded utterance is time aligned to its synthetic version. The time alignment allows to automatically obtain an approximate segmental labeling and syllable boundaries for the recorded speech. Using this alignment, intonation events (and their associated feature vectors) are projected onto the imitation speech, thus automatically producing an annotation for the recorded speech (avoiding the need for manual annotation). The projected feature vectors serve as the search keys for the database of natural prosody units.

### 2.2. Unit Selection

The selection algorithm is a standard Viterbi process [7], using a cost function which is a weighted sum of distortion costs (difference between a candidate and a target unit), and concatenation costs (difference between two consecutive candidates).

Before the selection is performed, the rule prosody system processes the text, decides where to place events, and creates the target feature-vector sequence corresponding to these events. The selection then runs, trying to find the best matching sequence of units from the database.

### 2.3. Unit Concatenation

The generation of the final prosody is done by transplanting the prosody from the selected recorded speech units into the target utterance.

Each syllable in the synthetic sentence is associated with an event. The prosody for the sequence of syllables associated with a target event, is taken from the sequence of syllables in the same relative position to the corresponding selected event. The transplantation of the prosody is done syllable by syllable (using a piecewise linear interpolation scheme inside each syllable, to avoid shifts in the relative location of pitch movements).

In order to avoid F0 discontinuities at concatenation points between two prosodic units, an F0 smoothing is performed by linear interpolation around the concatenation point.

For duration - a simple stretching of each selected syllable is performed to match target duration (except when the syllabic structure of the selected unit matches exactly that of the target unit - in which case, the duration of the selected natural unit is copied as is).

### 3. Applying Fallback

As in other selection systems, the size of the unit database has a crucial effect on the resulting quality. The database used in our experiment was clearly too small (12 minutes) to adequately represent the speaker’s prosody. However, as for other corpus selection methods, it is not practical to expect having a complete database, as its size grows exponentially.

It follows, that we should always expect to encounter target units for which no example is present in the database. The common practical approach is to find the “closest” unit in the database, assume it’s close enough (as the database size grows, this assumption slowly becomes more valid), and go ahead and use it.

### 3.1. Suggested Method

In this paper we investigate a simple alternative to this approach. In essence: instead of blindly trusting the selected units to be close enough to their targets, we check each unit if it really is, and if not, use the original rule-generated prosody for this unit. Instead of fully using the selection result, use it only partially - only the units which are close enough to the target.

In practice, however, this approach can not be implemented so simply: The selection process explicitly tries to maximize the continuity between adjacent units (through the use of the concatenation costs). Simply rejecting a unit and falling back to its original rule-generated unit, can violate the unit continuity, with adverse effects on the resulting prosody.

The suggested solution is to make the decision (to fall back to the rule prosody) inside the selection process itself: For a given sentence to be synthesized, we start by generating the rule prosody for the sentence, and its associated sequence of target events (same as in the original method).

Then, unlike the original method, we add the rule-generated target events to the training database, so that the database now consists of all the imitation events, plus the synthetic events for
the given sentence (in the current implementation, we actually use two database files - one for all the imitation sentences, and the other for the one synthetic sentence).

The final modification is the addition of a penalty in the selection algorithm, to be applied when using the synthetic units. This is done by adding a fixed value to the distortion cost of each of the synthetic units (which otherwise would be 0, since they are exactly the same as the target units).

Except for these differences, the selection process is run exactly as in the original method, and its results are used in exactly the same way.

3.2. Synthetic Unit Penalty

Figure 2 illustrates the effect of the value of the penalty, and the position of the unit inside the sentence, on the percentage of the selected units which are rejected by the suggested method (this percentage is referred to as the "Fallback Ratio").

As seen in the figure, when the value of the penalty is set higher (which is equivalent to allowing a higher dissimilarity between the selected target units), the fallback ratio gets lower (less of the selected units are rejected).

The figure also shows a strong dependence of the fallback ratio on the position inside the sentence - units close to the beginning or the end of the sentence are more likely not to be rejected.

This dependence can be explained by looking at the way the distortion cost is calculated, which is by summing differences between two context-inclusive windows (for each feature). For units which are close to the sentence edges, the parts of the context window which fall outside of the actual sentence match exactly, resulting in a lower cost, which is less likely to be rejected.

Another phenomenon which can be observed in figure 2 is a slight asymmetry in the shape of the graphs, with higher rejection towards the end of the sentence. This can be explained by the fact that the rule-generated prosody has a higher variability towards the end of the sentence than in the beginning of the sentence (e.g. differences between statements and questions, and different final declination values). Lower variability in the beginning of the sentence translates to a lower cost (on the average), and thus lower rejection.

Figure 3 shows the fallback ratio for the same penalty values, for the case a smaller (half size) training database is used. As expected, the fallback ratio is higher, as there are less available close candidates for each target unit.

In the listening experiment described below, figure 2 was used in calibrating the system - for choosing a value for the penalty that will result in the target fallback ratio.

3.3. Synthesis Example

Figure 4 shows example results of synthesizing the same sentence using different penalty values (increasing from top to bottom). The window at the top of the figure shows the result of a complete fallback to rule prosody, the window at the bottom shows the result using no fallback at all, and in the middle, two intermediate fallback ratios, in decreasing order.

It can be seen how more and more rule-generated units are being replaced by selected natural units as the penalty value increases.

As can be seen in this example in the two lower graphs, increasing the penalty does not only change the number of natural units to be used, but also their identity (the units indicated by arrows are natural units, which were used to replace the original rule-generated unit). The fallback, due to a lack of a close enough natural unit, of an adjacent unit to its rule-generated form, exerts influence on the surrounding units (through the requirement for continuity) and prevents bad examples from reinforcing each other. Thus, it pulls the selected unit sequence closer to the target sequence.

4. EVALUATION

A limited listening experiment was held, in which 7 listeners listened to 20 sentences. For each sentence, 3 versions were presented (in random order):

- Purely rule prosody.
- Purely selection prosody (using the method of [11]).
Using the fallback method described in this paper. The penalty was set according to figure 2, to the value corresponding to a fallback ratio of 0.3 in the middle of the sentence.

The listeners could listen to the different versions of each sentence as many times as they wanted, and were then asked to choose the version they preferred.

The results of this small experiment showed that in 48% of the cases, listeners preferred prosody generated by selection with fallback, in 32% of the cases rule prosody was preferred, and the pure selection was preferred in 20% of the cases.

Note that the set of test sentences for this experiment was chosen so that each sentence had at least two instances of fallback to the rule prosody (so that the listener would have to choose between three different versions). These are sentences which have some units with no close examples in the database, and thus are the ones where pure selection is prone to perform poorly. Without this pre-selection of test sentences, it is expected that in most sentences there would be no fallback, thus producing the pure selection prosody, which was preferred in [1]. In summary - the fallback method can produce more natural prosody than the rule prosody, and has the advantage of a more graceful degradation when adequate units are missing from the database.

5. CONCLUSION

Examining the method described in [1] for prosody generation by prosodic unit selection from an imitation speech database, some problems related to insufficient database size were found. Instead of increasing the database size, a complementary method was suggested, in which the selection falls back to rule generated prosodic units if no close natural unit exists in the database.

In order to ensure that the final unit sequence is as continuous as possible, even if fallback has occurred, the fallback mechanism is integrated into the dynamic unit selection algorithm, by augmenting the training database with the target rule-generated units, but penalizing the selection for choosing these units.

By varying the value of the penalty, it is possible to calibrate the system to a desired fallback ratio, which depends on the size of the database.

A listening experiment, using a small database, showed listeners preferred the results of the fallback method over those of pure rule-generated or pure selection prosody.

6. REFERENCES