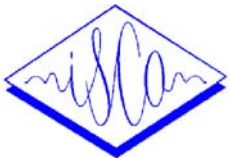

5th ISCA Speech Synthesis Workshop
Pittsburgh, PA, USA
June 14-16, 2004

FESTIVAL 2 – BUILD YOUR OWN GENERAL PURPOSE UNIT SELECTION SPEECH SYNTHESISER.

Robert A.J. Clark, Korin Richmond and Simon King

CSTR, The University of Edinburgh

ABSTRACT

This paper describes version 2 of the Festival speech synthesis system. Festival 2 provides a development environment for concatenative speech synthesis, and now includes a general purpose unit selection speech synthesis engine.

We discuss various aspects of unit selection speech synthesis, focusing on the research issues that relate to voice design and the automation of the voice development process.

1. INTRODUCTION

Festival [1] is a well established speech synthesis platform, providing all the modules necessary for a text-to-speech system, along with an environment suitable for researching all areas of speech synthesis. Recent developments in unit selection speech synthesis mean that Festival's diphone approach has become somewhat dated. Unit selection has been available in Festival for some time, but only for *limited domains* (and not-so-limited domains) in the form of *cluster unit* synthesis [2, 3]. However, this technique is not necessarily suitable for many applications or as a framework for researching other aspects of general-purpose unit selection. This paper reports recent Festival developments designed to address this problem. It discusses the design and implementation details of a general purpose unit selection engine for Festival called *multisyn*. The multisyn engine was designed to meet the following criteria:

1. the system is designed to be robust enough to be used as a real world synthesiser, rather than just a research tool which works for a few restricted examples;

2. the system is designed to be flexible enough to be useful as a development environment for further research into all aspects of speech synthesis and related fields such as natural language generation and dialogue system building;

3. the voice building process is designed to be simple enough that only very limited specialist knowledge is required to build new voices.

In section 2 we outline the general unit selection procedure implemented in Festival 2, which is fairly conventional. In sections 3 and 4 we describe aspects of voice design, concentrating on developments that are designed to simplify and automate the voice building process, from text selection and script design to automatic labelling.

2. SYNTHESIS PROCEDURE

The *multisyn* unit selection algorithm implemented in Festival 2 is conventional and reasonably straightforward. A target utterance structure is predicted and suitable candidates from the inventory are proposed for each target unit, then the best candidate sequence is found by minimising target and join costs.

There are a number of options for what size these units can be, the main contenders being phones, half phones, diphones or larger units (e.g. units matching prosodic structures [4]). Large or variable sized units are not used in our implementation, as we believe that the selection should be performed by the search, rather than by some pre-selection criteria that may restrict the search in a non-optimal way (section 2.2). Phones are also not used as they are notoriously difficult to join.

Diphones have the advantage that they are relatively easy to join. However, obtaining the necessary coverage for a full diphone inventory is hard [5, 6, 7, 8]. The distribution of diphone occurrence in context means that a large number of diphone-in-context types occur very infrequently, making it difficult to design a compact inventory, while any given sentence to be synthesised has a reasonably high probability of containing at least one rare diphone, highlighting the need for good coverage.

Using half phones would alleviate this problem somewhat by allowing joins at phone boundaries as well. However, we have used diphones in preference to half phones to make rapid voice development easier. This is because a greater degree of phone labelling accuracy is required to make phone boundary joins, whereas diphone joins (mid-phone) are less sensitive to label misalignment. As automatic phone alignment is generally more consistent than it is accurate, it is easier to guarantee good joins when dealing

only with diphones.

The advantage of being able to automate the labelling (and therefore being able to label more data) is that it is generally easier to collect a larger data set than it is to collect a smaller dataset and guarantee the accuracy of labels through hand correction. This said, the system is still implemented in such a way that using units other than diphones would only require a small amount of fairly straightforward work.

2.1. Target construction

A target utterance structure is constructed from text (or marked-up text). One of the advantages of unit selection synthesis over diphone synthesis is that a lot of the information that has to be predicted for diphone synthesis is not strictly necessary for unit selection synthesis. Many properties of the speech, including segment durations and prosody, do not need to be explicitly modelled. Instead, the natural segment durations and prosody inherent in the database are used. As a result, the basic linguistic resources that are needed are a simple phrasing model and a pronunciation lexicon or other grapheme to phoneme conversion routine.

2.2. Pre-Selection

The result of the linguistic phase of the synthesis process is a sequence of target phones with an appropriate linguistic structure attached. The selection phase of the synthesis then proceeds in the conventional way: the target phone sequence is converted to a list of target units (diphones, in Festival 2); a list of candidates for each target unit is retrieved from the inventory – with an optional pre-selection step to limit the length of each list – and each is scored according to a target cost; joins are scored with the join cost, and finally the optimal candidate sequence is found by Viterbi search.

Pre-selection can be employed to limit the number of candidates for each target unit to only those that are suitable. Methods of pre-selection vary from the complex, such as phonological structure matching [4], to the simple, such as only including units which are appropriately stressed/unstressed. Pre-selection has the ability to make the speed up the search significantly by restricting the search space.

Pre-selection can be seen as a sub-component of the target cost; in our implementation, it was decided that any form of pre-selection was effectively attempting to second guess the target cost in a way which could not necessarily be guaranteed to be optimal. Any speed gain obtained by restricting the initial search space should also be achievable by pruning. For example, if stress is deemed to be sufficiently important that phones carrying the wrong level of stress should not be used, beam pruning with a suitable beam width can ensure that if one suitably stressed candidate is available then all unsuitably stressed candidates are not considered. Folding the pre-selection into the target cost also simplifies the overall system architecture.

So, an initial candidate list for each target diphone contains all the diphones of that type from the inventory; clearly, the inventory must have at least one token of each diphone type for this to work. However, this may not be the case, in which case some backing-off must be performed.

2.3. Backing-off

The long-tailed Zipf-like distribution of diphone types makes it difficult to guarantee complete coverage of all necessary diphones in the inventory during the script design phase. Diphones may also be missing for other reasons, such as instances where the speaker has spoken a word with a pronunciation different to that predicted during script design (and where the labelling has been adjusted appropriately), or where an existing dataset has been used as a voice, and the planned coverage cannot be controlled at all. An example of using existing data is described in section 5.1.

To deal with missing diphones, a back-off procedure has been implemented. If a diphone cannot be found, a manually-written, ordered list of possible substitution rules is consulted and an attempt is made to find an appropriate replacement diphone.

In initial experiments, the back-off procedure altered the target sequence appropriately to find not just a replacement for the missing diphone, but to then substitute the surrounding diphones to preserve continuity of the phone sequence. For example, to synthesise the word “team” the diphone sequence t-ee ee-m is required. If the diphone t-ee was missing and substituted for the diphone t-schwa, an attempt would then be made to substitute the target diphone ee-m with schwa-m to keep the phone sequence consistent.

It quickly became apparent that this in itself was a difficult search problem, and that unless the substitution rules were written very carefully, it was difficult to obtain a suitable substitute phone sequence. The procedure was therefore simplified and the current back-off procedure does not correct adjoining diphones, so any substitution that occurs means that there will be a mismatch in phone sequence either with the diphone preceding or following the diphone being substituted. An appropriately specified join cost ensures the join is as good as it can be. Obvious substitution rules include: reduced vowels for full vowels (in which case there are probably instances of the full vowels and reduced vowels which are spectrally close enough to join reasonably well); substitutions like [n] to replace a missing [n!] (syllabic [n]) , where there will be little difference at the join point.

After candidate substitution to account for any missing diphones, the best candidate sequence is found using a stan-

dard Viterbi search to minimise the sum of all the target costs and join costs.

2.4. Target Cost and Join Cost

The target cost is the sum of a user definable set of weighted functions, each of which adds a penalty cost if some feature of the candidate diphone doesn't match the target, or if some default penalty feature is set in a candidate (which can be used to penalise candidates with poor labelling or bad pitch marking).

Both the weights and the functions can be user defined, but a reasonable default target cost is provided in the current implementation which includes components for stress; left and right phonetic context; position in the syllable, word and phrase; part of speech. Of the comparison features, stress is weighted most highly, followed by phrase position. The other penalty features have weights similar to stress.

The join cost employs three equally weighted subcomponents for pitch, energy and spectral mismatches. Spectral discontinuity is estimated by calculating the Euclidean distance between two vectors of 12 MFCCs from either side of a potential join point. Euclidean distance between two additional coefficients for f0 and energy are likewise used to estimate the pitch and energy mismatch across the join. All three sets of acoustic parameters used for the join cost are pre-normalised to lie within the range [0:1] during voice building.

Pitch discontinuity incurs the maximum penalty for a join between voiced and unvoiced speech. The weightings for the current target and join costs have been derived empirically to provide a baseline acceptable performance, but these can easily be changed to values based on statistical training or perceptual evaluation, should data be available.

3. VOICE DESIGN

A crucially important aspect of unit selection speech synthesis is voice design. However well a system is implemented, the resulting speech can only ever be as good as the data that makes up the voice inventory. Moreover, many of the important implementation details are tied up in the voice specification. The voice contains a database of structurally annotated speech data.

3.1. Script Design

The fundamental concern of script design is to provide sufficient recorded units to ensure good quality synthesised speech. A minimal requirement here is obviously at least one example of each unit type (diphone) that the system needs; this is complicated somewhat if a strict pre-selection is performed at synthesis time – we have avoided this by folding pre-selection criteria into the target cost as weighted sub-costs rather than strict requirements. As diphones are going to be scored by the target cost according to how well they match a specific target context, having a wide variety of each diphone type in as many of these different contexts as possible is desirable. Context minimally includes stressed or unstressed condition, position in syllable, word and phrase. The rare diphone problem makes choosing other features to be part of the context harder, as rare diphones become even rarer, and a compromise has to be met between utilising a very finely detailed context and having to record a very large dataset to cover it.

Using units in contexts as described above results in reasonable quality synthesis. The selection criteria (join cost and target cost) that determine which units are selected and combined for synthesis are strictly *local*. These guarantee consistency at a local level between any two or three consecutive units, but cannot guarantee any *global* consistency. For example, in synthesising a list of numbers, each number in the sequence will probably sound reasonably natural in its own right, but the list structure as a whole will not receive appropriate prosody. If units making up numbers or parts of numbers are taken from different global contexts and put together in a list, they are likely to sound unnatural, as there is nothing to ensure we get the prosodic structure that is associated with a list.

One way of dealing with this (and with related problems) is to add additional constraints. The inventory can be annotated with specific syntactic or semantic structure, which can disambiguate units in wider linguistic structures, but to do so requires a lot of extra manual work. Doing this in a consistent manner is difficult, and being able to predict the appropriate structure for a target sentence at synthesis time, where there is little or no pragmatic context, is harder still.

While there is no restriction on what can be annotated in the inventory, or on what information can be used in target cost, it cannot be assumed that the type of information required for syntactic or semantic annotation would be readily available to the voice builder. A alternate solution is to try to design the inventory in such a way that the existing local constraints are sufficient. A simple example would be to ensure that the types of structures that the voice is likely to be required to say are present in the dataset. For example, to have a system that reads lists of numbers well, one should include lists of numbers in the original recordings; then, when synthesising lists of numbers, there is a reasonable chance that the most suitable candidates will come from lists of numbers, and the combination of target and join cost constraints should result in a reasonable list structure being produced. Examples which show this approach to be reasonably effective are demonstrated in section 5.2. It is also worth including uncommon phrases that require particular stress patterns that you expect to need in the dataset,

your research group or company name is probably a good example of this!

3.2. Recording Conditions

Once a voice script has been designed, there are two major issues concerning actually recording the voice.

1. The quality of the recording environment.
2. The quality of the speaker.

Opinions differ as to the needs of the recording environment. The ideal environment is a near anechoic recording studio. Clean speech with no reverberations present is much easier to manipulate. However, we have made a reasonable sounding voice from CMU_ARTCIC [9] data that was recorded using a laptop computer and a cheap microphone in a quiet room.

The quality of the speaker is equally important. The ideal speaker has a clear sounding voice and can read large quantities of text in a natural sounding way. They should be able to keep their voice quality consistent over a long period of time. Voice talents often have the right voice qualities, but their ability to read from a script consistently may be lacking as they are usually accustomed to speaking for shorter periods where they can memorise their lines. Additionally, however good the quality of a voice is, if it is perceived as sounding annoying, any synthetic voice that is built from it will probably sound annoying too.

The script needs to be presented in a suitable way; forcing the speaker to read text from the same position on a screen for a number of hours is not good for the speaker's sanity. However, presenting it from different positions also has its problems. For example, when presenting ten sentences to a page, we have found clear correlations between some cepstral coefficients and the position of a sentence on the page, probably due to the change in position of the speakers head. Spectral averaging also clearly shows where breaks in each recording session were taken and where sessions start and end. However, there does not seem to be a straightforward relationship between spectral properties and synthesis quality, and it is not yet clear the extent to which these effects reduce the quality of synthesis. The bottom line is to be as consistent as you can afford to be, but find a suitable compromise between cost and quality.

4. VOICE BUILDING TOOLS

Many of the tools needed to build voices for Festival, including general purpose unit selection voices, are freely available as part of the FestVox project [10]. Additional tools, often based upon FestVox ones, are provided with Festival 2 to aid the building of voices for the new unit selection engine.

4.1. Automatic Labelling

Automatic labelling of the speech data can be carried out in a number of ways, but we employ a forced alignment procedure using the HTK HMM tool kit [11], for which we provide a number of scripts. To provide a robust labelling procedure a number of issues are addressed, described below.

Unlike recognition, forced alignment starts from a known sequence of segment labels. There are two factors which determine what these labels should be. First, these labels need to be an accurate representation of what was spoken by the speaker. If the labels are not accurate, then synthesis resulting from using them may sound bad. This is easily demonstrated for the case of vowel reduction. Suppose an instance of the word "were" was spoken by the speaker with a reduced vowel. If this were labelled as a full vowel and then used in a context where a reduced vowel was inappropriate, like in the word "worthy", the result would be bad. This is a particular problem with diphones because one half of the synthesised vowel may be inappropriately reduced, while the other is a full vowel.

The second factor is that the labelling of the database should be consistent with the segment sequences that will be generated as part of the target utterances at synthesis time. If this were not the case, inappropriate and often unnecessary joins will be made where there are inconsistencies. For example if the word "were" is always specified with a full vowel at synthesis time, then it should always be labelled as such in the dataset, otherwise a series of two consecutive diphones from the dataset will never be used to synthesise it.

It is apparent that these two factors conflict with each other. Our early synthesis attempts during development of the system suggested that the first criterion – labelling the speech according to what was actually said – was more important because, most of the time, a segment that sounds wrong is usually worse than an extra join or two.

With this in mind, our labelling procedure generates an initial label sequence using the linguistic analysis phase of the synthesis process – i.e. lexical lookup, letter-to-sound, and post-lexical rules. A few additions are then made to this sequence: closure labels for stops and affricates, sentence initial and final silence and optional short pauses between words. The forced alignment procedure will determine whether these short pauses are present in the speech, and the procedure is also allowed to make certain substitutions to improve the alignment log likelihood. This is currently restricted to vowel reduction, but could be extended to other substitutions.

The alignment procedure has been designed to require from the user only the text for each utterance, a list of the phone set and a set of possible phone substitutions. It is carried out using standard left-to-right monophone HMMs

with three emitting states and mixture densities with eight Gaussian components. Triphone models have not been used, as the performance of the monophone models was deemed good enough not to warrant the additional complexity of building the triphone models. Our requirement is a consistent alignment of phone boundaries rather than a reduction in word error rate. The models are trained first using a single phonetic transcription; then an intermediate forced alignment is carried out, with substitutions being allowed. The models are then retrained, and the densities converted to mixture distributions using HTK's standard "mixing up" procedure. A final alignment step then produces the labelling for the inventory. The speech is parameterised as 12 Mel-scale cepstral coefficients, energy, deltas and delta deltas. A relatively short window size of 10ms is used with a short 2ms shift. Initial results suggest that this generates more consistent boundary positions and fewer gross labelling errors than using a larger frame shift or longer window.

Once the alignment is done, the label times are reconciled with the linguistic structure that is generated by the synthesiser. This process deals with insertions and deletions of pauses, substitutions made by the automatic labelling process and the merger of the closure and release portions of stops and affricates into a single label. Substitutions that have occurred are marked as such in the linguistic structure (for possible later use in unit selection) and the end of the closure portion of stops and affricates is added to the linguistic structure and used as the diphone join point for these segments.

Other information is stored on individual phones in the linguistic structure to enable the target cost to incorporate a component which indicates bad labelling. This includes a normalised version of the log likelihood score for each segment and a flag which marks a segment as too short to have meaningful pitch-marking.

The result of this alignment procedure is a segmental labelling that is very consistent and reasonably accurate. Judging accuracy in phone boundary labels is very difficult because of the transitional nature of phone boundaries and co-articulation effects. A comparison to some reference hand labelled data can be made, but results suggest consistency is more important for speech synthesis. [12] may really be suggesting that the levels of inconsistency in hand labelled data means that it cannot be considered as an accurate baseline.

5. DISCUSSION

5.1. Real World Voice Performance

The performance of a given voice depends upon the amount of speech it contains, and how well that speech matches the type of speech being synthesised. Increasing the inventory size will generally increase the quality of the output if the additional units are appropriate for what is being synthesised. However, adding units increases computational load and there is probably a limit on the amount of speech which can be obtained from a speaker over a short enough period of time to avoid voice quality changes. The n-squared relationship between the lengths of candidate lists and the number of join costs to be computed means that a compromise between database size and speed needs to be found.

Voice	Phones	Time to Synthesise
cstr_nina	175000	9.25
cstr_awb	36000	1.96
cstr_fsew0	14000	1.00

Table 1. Time taken to synthesise and play: "*Comprehension usually precedes production. Quite often contextual cues are strong enough for the child to get the gist of an utterance without perhaps being able to understand the details.*" using voices with various inventory sizes. Times are expressed as a proportion of the time taken by the smallest voice.

We present the performance of three voices of very different sizes. The first voice, *cstr_nina*, is a CSTR prototype voice, it consists of just fewer than 175000 phones (5.8 hours of speech), all of which is read newspaper text[1]. The second voice, *cstr_awb*, is a smaller voice that was built from ARCTIC data [9]. This voice contains 36000 phones (1.4 hours of speech) of text from out of copyright books. The third voice, *cstr_fsew0*, is a single set of 460 TIMIT [13] sentences for British English that were recorded as part of the Mocha [14] project. This voice contains 14000 phones (about 55 minutes of speech) and was recorded in an EMA machine. This voice is primarily designed for evaluating the usefulness of articulatory information as a contribution to the join cost. The obstructions of the instrumentation in the mouth make the segmental quality of the voice somewhat unnatural, but the voice provides a useful comparison to the others as it comprises a relatively small number of phones. Table 1 shows the time taken to synthesise and play a single sentence.

As the number of diphones increases, so does the time taken to synthesise an utterance. The *cstr_awb* voice currently synthesises in about one fifth real time, whereas the *cstr_nina* voice is slower. (We expect to be able speed up both voices by code optimisation with no change in synthesis quality.) The *cstr_fsew0* voice is the fastest, but there are noticeable problems with the resulting synthesis, strongly suggesting that there are insufficient phones here. The time increase required to synthesise with increasing numbers of diphones is less than linear, showing that the pruning is working well. The quality of both the *cstr_nina* and *cstr_awb*

[1] We are grateful to The Herald for this data (www.theherald.com)

voices is very good and in general both of these voices sound reasonably natural. The *cstr_nina* voice is usually better than the *cstr_awb* voice but is maybe not sufficiently better to justify the use of the significantly larger number of phones. This suggests that the ARCTIC dataset size is about right. However, our plans to consider prosodic context in more detail may dictate the need for a larger dataset.

5.2. Domain Specific Data

Evaluation examples showed that the *cstr_nina* voice was not very good at providing flight information, and the prosody of synthesised utterances in this domain varied from fair to terrible. This was often down to the durations of words being quite inappropriate for the context. In an attempt to address this problem, a variant of the *cstr_nina* voice was created, which contained one quarter of the original data (392 sentences designed for minimal diphone coverage) plus 1000 utterances from the flight booking domain. The resulting synthesis was found to far higher, showing how domain-specific speech can greatly improve speech synthesis.

6. FUTURE WORK

A number of ongoing projects are aimed at improving different aspects of unit selection speech synthesis, including: addressing the need for long term consistency measures, particularly with respect to prosody, enable natural language generation systems to be able to convey specific meaning through intonation; improving join cost and join smoothing by using information about underlying articulation.

7. ACKNOWLEDGEMENTS

We thank all those that have supported Festival over the years, both developers and users. Special thanks go to Alan Black, without whom there would be no Festival.

We greatly appreciate funding provided by EPSRC GR/R94688/01 and the EC project IST-1999-29078 which has supported aspects of this research.

8. REFERENCES

[1] Paul Taylor, Alan Black, and Richard Caley, "The architecture of the Festival speech synthesis system," in *Proc. The Third ESCA Workshop in Speech Synthesis*, 1998, pp. 147–151.

[2] A. Black and P. Taylor, "Automatically clustering similar units for unit selection in speech synthesis.," in *Proc. Eurospeech 97*, Rhodes, Greece, 1997, vol. 2, pp. 601–604.

[3] Alan Black and Kevin Lenzo, "Limited domain synthesis," in *Proc. ICSLP2000*, Beijing, China, 2000.

[4] Paul Taylor, "Concept-to-speech by phonological structure matching," *Philosophical Transactions of the Royal Society*, 2000, Series A.

[5] Jan van Santen and A. Buchsbaum, "Methods for optimal text selection," in *Eurospeech97*, 1997, vol. 2, pp. 553–556.

[6] Mark Beutnagel and Alistair Conkie, "Interaction of units in a unit selection database," in *European Conference on Speech Communication and Technology*, 1999, vol. 3, pp. 1063–1066.

[7] Alan W. Black and Kevin A. Lenzo, "Optimal data selection for unit selection synthesis,," in *4th ISCA Workshop on Speech Synthesis*, 2001, pp. 63–67.

[8] Bernd Möbius, "Rare events and closed domains: Two delicate concepts in speech synthesis," in *4th ISCA Workshop on Speech Synthesis*, 2001, pp. 41–46.

[9] J. Kominek and Black A., "The CMU ARCTIC speech databases for speech synthesis research," Tech. Rep. CMU-LTI-03-177 http://festvox.org/cmu_arctic/, Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA, 2003.

[10] Alan Black and Kevin Lenzo, *Building Voices in the Festival Speech Synthesis System*, http://www.festvox.org.

[11] S. Young, G. Evermann, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, *The HTK Book (for HTK version 3.2)*, Cambridge University Engineering Department, 2002.

[12] M. Makashay, C. Wightman, A. Syrdal, and A. Conkie, "Perceptual evaluation of automatic segmentation in text-to-speech synthesis," in *Proc. ICSLP2000*, Beijing, China, 2000.

[13] J. S. Garofolo, *Getting started with the DARPA TIMIT CD-ROM: An acoustic phonetic continuous speech database*, National Institute of Standards and Technology (NIST), Gaithersburgh, MD, 1988.

[14] Alan A. Wrench, "A new resource for production modelling in speech technology," in *Proc. Workshop on Innovations in Speech Processing*, 2001.