CLUSTERGEN: A Statistical Parametric Synthesizer using Trajectory Modeling

Alan W Black

Language Technologies Institute
Carnegie Mellon University, Pittsburgh, PA, USA
awb@cs.cmu.edu

Abstract

Unit selection synthesis has shown itself to be capable of producing high quality natural sounding synthetic speech when constructed from large databases of well-recorded, well-labeled speech. However, the cost in time and expertise of building such voices is still too expensive and specialized to be able to build individual voices for everyone. The quality in unit selection synthesis is directly related to the quality and size of the database used. As we require our speech synthesizers to have more variation, style and emotion, for unit selection synthesis, much larger databases will be required. As an alternative, more recently we have started looking for parametric models for speech synthesis, that are still trained from databases of natural speech but are more robust to errors and allow for better modeling of variation.

This paper presents the CLUSTERGEN synthesizer which is implemented within the Festival/FestVox voice building environment. As well as the basic technique, three methods of modeling dynamics in the signal are presented and compared: a simple point model, a basic trajectory model and a trajectory model with overlap and add.

Index Terms: speech synthesis, statistical parametric synthesis, trajectory HMMs.

1. Unit Selection and Parametric Synthesis

The current preferred speech synthesis technique is probably unit selection, where appropriate sub-word units are selected from large databases of natural speech [1]. Over the last ten years this technique has been shown to produce high quality synthesis and is used for many applications. For its best examples it is hard to beat in quality, but it does have a limitation (and advantage) that the output speech will strongly resemble the style of the speech recorded in the database. As we require speech more varied in style and emotion, to retain the quality of unit selection we need to record larger and larger databases with different styles in order to achieve the synthesis we desire. However we have an alternative method for synthesis, which although at first seems to be a step backwards from unit selection may offer the ability to model different styles without requiring the recording of very large databases. Statistical Parametric Synthesis (SPS) is a new synthesis method, pioneered by HTS [2] where parametric models are trained from databases of natural speech. SPS falls firmly in the domain of corpus-based synthesis like unit selection, but distinguishes itself from older non-statistical parametric synthesis techniques as found in DECTalk [3], in SPS the parameters are trained from data and not constructed by hand.

Statistical Parametric Synthesis has the advantage of smoothing the data. The number of possible combinations of segments in concatenative synthesis is typically vast and some concatenations may introduce bad joins. Testing for this is very hard, and fixing such errors is not always simple. Statistical Parametric Synthesis addresses this issue by building what can be viewed as an average of a number of units, rather than in the unit selection case as a set of instances. There are however disadvantages too in Statistical Parametric Synthesis, the technique requires a parameterization of the speech that is both reversible, and has modelable properties, such as a Gaussian distribution. One such parameterization is Mel Frequency Cepstral Coefficients, as used in HTS, using the MLSA [4] filter for resynthesis. As with many speech parameterizations with no explicit excitation model the resulting resynthesized speech is vocoded and can have an unnatural buzziness, lacking the clear crispness typically found in unit selection synthesizers.

Other parameterization mechanisms have been shown to offer potentially better quality speech e.g STRAIGHT [5] in [6]. As any form of speech analysis/synthesis technique will undoubtedly introduce some signal processing artifacts, unit selection techniques have prided themselves on using no or very little signal processing to retain the crispness in the generated voices. The cost of this is the requirement for building larger and larger databases. [7] achieved stylistic variation in unit selection but at the cost of recording more data in different styles, something that is not necessarily easy for a voice talent to do. Although some users do not care about the size of a databases if it gives better quality synthesis, our goal is automatic speech output everywhere for everyone. Techniques that provide smaller models, and work with smaller databases will help us achieve our goal. The ultimate restriction is that it is clear that most people cannot consistently read thousands of utterance (nor have the time to do so). We still believe that resynthesis for parameterized speech is not good enough yet (even though its current quality is perfectly acceptable for some applications), and there is still work to be done there.

Statistical Parametric Synthesis has sometimes been called "HMM-generation synthesis", to distinguish it from HMM-state sized units in unit selection [8], however in this work (and in others) there is no actual requirement for HMMs. No HMMs are used at synthesis time, even though HMMs can be used to label the data. Therefore Statistical Parametric Synthesis seems a better term.

In this paper we will present CLUSTERGEN, a Statistical Parametric Synthesizer that has been created within the widely used Festival/FestVox voice building suite [9]. The first half describes the core technique while the second half will report results. Although the basic CLUSTERGEN system is only slightly different from HTS, the later results of trajectory modeling are substantially newer.

Building English voices for HTS with the FestVox suite of tools has been supported for some time [10]. But there were a number of hardwired aspects, particularly in the feature names and questions for the cluster methods in HTS. This work set out originally to make the link between HTS and FestVox more robust for different databases and languages, but grew into a complete synthesizer in itself (though it still uses exactly the same parameterization technique).

2. CLUSTERGEN Synthesizer

The CLUSTERGEN synthesizer is a method for training models and using these models at synthesize time within the Festival Speech Synthesis System. The training requires well recorded utterances, and text transcriptions of what has been said. The best databases are those that are phonetically balanced. For our experiments we have used the freely available CMU ARCTIC databases so that these experiments may be easily duplicated by others.

2.1. Training

The first stage, which is not technically part of the CLUSTERGEN synthesizer is to label the database using an HMM labeler. For the results presented here, we have used EHMM, [11], which is included within the latest FestVox release. It uses Baum Welch from a flat start to train context independent HMM models, which it then uses to force align the phonemes generated from the transcriptions with the audio. For this work we use 3-state models, that generate HMM state sized labels, three per phone. We have used other labeling techniques (SPHINX and JANUS), but the work presented here has all used EHMM.

Although in our other work [12] we typically analyze the signal in a pitch synchronous fashion, here we used a fixed frame advance of 5ms.

F0 is extracted using the Edinburgh Speech Tools **pda** program. Using the generated phoneme labels, the F0 is interpolated through unvoiced regions, thus there is a non-zero F0 value for all 5ms frames that contain voiced or unvoiced speech. This is following the F0 modeling techniques in [13].

24 MFCCs are combined with the F0 to give a 25 feature vector every 5ms.

For each of these vectors high level features are extracted, including phone context (with phonetic features), syllable structure, word position, etc. The extracted features are basically the same set used by the previous CLUNITS unit selection synthesizer [12], however in this case we extract them for each vector, rather than for each segment (phoneme).

Clustering is done by the Edinburgh Speech Tools CART tree builder **wagon**. It has been extended to support vector predictees. CART trees are built in the normal way with wagon to find questions that split the data to minimize impurity. A tree is built for all the vectors labeled with the same HMM state name. The impurity is calculated as

$$N * (\sum_{i=1}^{24} \sigma_i) \qquad (1)$$

Where N is the number of samples in the cluster and σ_i is the standard deviation for MFCC feature i over all samples in the cluster. The factor N helps keep clusters large near the top of the tree thus giving more generalization over the unseen data.

Initial studies built joint F0/MFCC models, but slightly better results are possible when separate F0 and MFCC models are built.

In these tests no delta features are used, initial studies did not give better results so that is left for later research.

An additional CART tree is built to predict durations for each HMM state. Unlike HTS, we predict each state duration independently, though do include features to identify the states position in its phoneme.

2.2. Synthesis

At synthesis time the phone string is generated from the text as is done in other synthesis techniques within Festival, then an HMM state name relation is build linking each phone to its three subphonetic parts. The duration CART tree is used to predict the length of each HMM state. A set of empty vectors is created to fill the length of the predicted state duration. Using the CART tree specific to the state name, the questions are asked and the means from the vector at the selected leaf are added as values to each vector.

Note, unlike HTS when a single vector is predicted for each state (though the treatment of dynamics does complicate this a little), in CLUSTERGEN we are predicting multiple vectors per state. This means that the predicted vector may be different through the state.

After prediction smoothing is done by a simple 3-point moving average to each track of coefficients.

$$s'_t = (s_{t-1} + s_t + s_{t+1})/3.0 \qquad (2)$$

Where s_n is the sample at time point n.

Then the speech is reconstructed from the predicted parameters using the MLSA filter [4]. Voicing decisions are currently done by phonetic type directly from the labels, rather than trained from the acoustics. A more elaborate model taking into to account acoustic information with respect to voice would probably give better results.

3. Experiments

Although other experiments have been done using CLUSTERGEN in the multilingual space [14], the work presented here concentrates on English, specifically the US English CMU_ARCTIC databases [15].

The base experiments were carried out on CMU ARCTIC SLT, a US female database. The database consists of 1132 phonetically balanced sentences. For testing we held out one tenth of the A set within the databases, this gives a test set of 59 sentences (actually every fileid that matches "arctic_a.*9") and 1073 for testing. Much of the testing was actually done on the A set alone, thus the training consisted of 534 utterances. The full 1132 utterance database consists of 41888 segments, around 56 minutes of speech, including less than 0.5 seconds of silence at the beginning and end of each utterance.

Unlike our unit selection work where we use short listening tests to evaluate different parameter settings in out systems [16], here we use an objective measure. For our held-out test set we use their defined state durations and predict F0 and MCEP features for each vector. We then calculate a distance measure between the predicted spectral features and the actual ones. This measure ignores duration modeling.

We choose to use Mel Cepstral Distortion (MCD) as a measure, which we have already used in our voice conversion work [17]. The measure is defined as

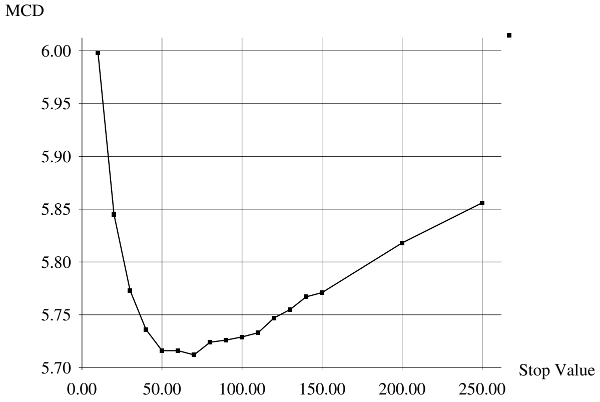
$$10/ln(10) * \sqrt{2 \sum_{i=1}^{24} (mc_i^t - mc_i^p)^2} \qquad (3)$$

Where mc_i is the ith MFCC coefficient in a frame, mc^t is the target MFCC we are comparing against and mc^p is the predicted MFCC

For voice conversion work we achieve values in the range of 4.0-6.0. In this work we achieve values in the range of 4.5-8.0 (and sometimes larger). Smaller numbers are better.

Note this measure does not weight the Mel Cepstral parameters as the range of Mel Cepstral Coefficients gets smaller for higher coefficients, thus this measure is more sensitive to minimizing the lower order ones.

The first experiment presented is for SLT with a training set of 1072 utterances and 59 test utterances. This experiment is to determine an appropriate stop value for training the CART trees. That is the minimal number of examples in cluster that are necessary before considering potential splits for that cluster.

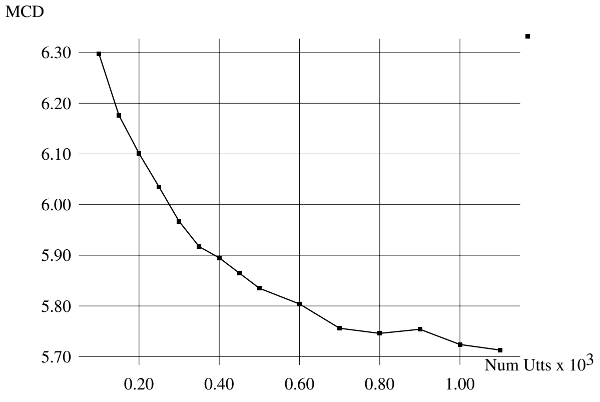

Stop value to MCD

Stop value for clustering

Using information from the above distribution we fixed the stop value at 70. Interestingly this gives approximately the same number of CART tree leaves over all as when using the CLUNITS unit selection [12] clusters that has a stop value of 20. As CLUSTERGEN is training on more data (a number of vectors per state) it appears we are still ending up with approximately the same number of spectral distinctions.

F0 modeling is also done by a CART tree predicting values for each vector, a larger stop size found to be better. The results give a RMSE for F0 as 14.09Hz, the same F0 model is used for the trajectory models described below.

The next experiment shows how varying the amount of data in the training set affects the synthesis results. It is notable that even modest amounts of data (200 utterances) can produce quite acceptable synthesis. The training subset was created by selecting the first N utterances from the complete 1073 utterance set. We start at 100 utterances, as that number is necessary to get full phonetic coverage.


Number of Utterances to MCD

Variation in size of database

4. Trajectory Modeling

The basic CLUSTERGEN method does not take into account the dynamic properties of the signal. Prediction is done in isolation to the prediction from the vectors before and after. However, unlike HTS as we are predicting potentially different vectors every frame we do model some of the desired dynamic variability (HTS achieves this through dynamic Cepstral features). But neither of these techniques really addresses the explicit modeling of dynamics in the segments we are trying to model.

Trajectory modeling [18] offers a potential solution here. Instead of modeling a single vector of Gaussians for each state, we should be modeling a sequence of vectors. In speech synthesis we are trying to model the variation of the speech over time, and a single Gaussian may be too gross a level to capture that variability. Here, we present two basic trajectory models. The first, we call **trajectory** where we model a HMM state-sized segment by a sequence of cepstral vectors. We optimize our CART tree to cluster to minimize the variance of the sequence of segments in the cluster.

This measure is very similar to the measure used in our previous CLUNITS technique [12], but there we keep a set of instances rather than representing the cluster in a model of means and variances.

One important issue is defining the number of vectors in the sequence. It should be related to the length of the samples in the cluster. At first we used the mean size, but after experimentation we settled on size 7 or the mean if greater. Each segment in the cluster is linearly interpolated to the size of the model for the cluster and sufficient statistics are updated. Note we do **not** use DTW to align these as the time differential through the segments is part of what we want to model, normalizing them through DTW would loose some of that information.

The second trajectory model presented is **trajola** a trajectory model with overlap and add. In this second model each instance for clustering consists of two parts the current segment and the previous segment. In an analogy to the motivation for using diphones for synthesis, here we are modeling the transitions between to HMM state sized units. The left and right parts are stored in separate models within the cluster, (again of size 7 or the mean which ever is bigger). At synthesis time the appropriate cluster is selected, the right portion is added to the signal, while the left is overlapped with the previous segment with a windowed weighting, with the sum of such weights adding to 1.0.

We tried both a triangular window and a Hanning window and found the triangular window gave better results. In addition to using the window and synthesis time while constructing the full predicted sequences of vector. We also found using the same window as weights in the impurity measure in the CART tree builder slightly improved results.

The stop value used for the trajectory and trajola modeling was 10, discovered by experiment. As trajectory modeling is done on a HMM-state sized units rather than the vectors within a state, there are less instances to train on, thus a smaller stop value is not unexpected..

The following table shows a comparison of the three presented statistical parametric techniques on 7 different ARCTIC databases. **cgp** is the simple single vector per leaf model. The databases all have approximated the same number of utterances, and they are split into training and test sets as with the SLT experiments above. Though as not all databases have exactly the same number of utterances these are not directly comparable. Although some speakers are not US English speakers, in all cases we used a US English front end (lexicon and phone set).

	Sex	Dialect	cgp	traj	trajola
AWB	M	Scottish	6.557	6.480	6.471
BDL	M	US	6.129	5.857	5.770
CLB	F	US	5.417	5.076	4.992
JMK	M	Canadian	6.165	5.934	5.872
KSP	M	Indian	5.980	5.823	5.733
RMS	M	US	5.731	5.437	5.394
SLT	F	US	5.713	5.525	5.472

In all cases the **trajola** models are better than the **traj** models which in turn are better than the **cgp** model. The number of vectors in each voice's test set varies from 32,000 to 45,000 (due to different speaker rate, and amount of silence at either end). Using two-sided paired T-tests the results for **traj** and **trajola** are statistically significant at the $p < 0.0015$ level, while the other results are statistically significant at the $p < 0.001$ level.

It is also worth mentioning the size of each model. The **cgp** model is much smaller than the trajectory ones. Although all have approximately the same number of leaves, there are more parameters in the leaves of the **traj** model and even more in the **trajola**. For SLT, there are 6455 vectors, for **traj** 73582 vectors in all sequences, and for **trajola** 198628 vectors. Although the above better results for trajectory models are partly due to more parameters, its also due to better modeling. As we can see from the stop value graph for **cgp**, allowing larger models does not improve the **cgp** model.

5. Conclusions

We see CLUSTERGEN as a first step forward for a new kind of synthesis, and there is still much work to make it better. Our current focus is on the area of signal representation and we are investigating both STRAIGHT [5] and HNM [19].

The updates to Edinburgh Speech Tools and Festival to build and run CLUSTERGEN voices are already released in the latest beta version 1.96. The scripts and accompanying Scheme support code (current version 0.8) will be released with the next version of FestVox. All code is released under a free software license.

6. Acknowledgments

This work is in part supported by the US National Science Foundation under grants number 00205731 ”ITR: Prosody Generation for Child Oriented Speech Synthesis” and number 0415021 ”SPICE: Speech ProcessingInteractive Creation and Evaluation Toolkit for new Languages.” Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. The author would like to thank Arthur Toth who helped in some presentation aspects of this paper.

7. References

[1] A. Hunt and A. Black, “Unit selection in a concatenative speech synthesis system using a large speech database,” in *ICASSP-96*, Atlanta, Georgia, 1996, vol. 1, pp. 373–376.

[2] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and Kitamura T., “Speech parameter generation algorithms for HMM-base speech synthesis,” in *ICASSP2000*, Istanbul, Turkey, 2000.

[3] J. Allen, S. Hunnicut, and D. Klatt, *Text-to-speech: The MITalk system*, Cambridge University Press, Cambridge, UK., 1987.

[4] S. Imai, “Cepstral analysis/synthesis on the Mel frquency scale,” in *ICASSP-83*, Boston, MA, 1983, pp. 93–96.

[5] H. Kawahara, I. Masuda-Katsuse, and A. Cheveigne, “Restructuring speech representations using a pitch-adaptive time-frequency smoothing and an instantaneous-frequency-based f0 extraction: possible role of a repetitive structure in sounds,” *Speech Communications*, vol. 27, pp. 187–207, 1999.

[6] H. Zen and T. Toda, “An overview of NITECH HMM-based speech synthesis system for Blizzard Challenge 2005,” in *Interspeech 2005*, Lisbon, Portugal, 2005.

[7] E. Eide, A. Aaron, R. Bakis, W. Hamza, M. Picheny, and J. Pitrelli, “A corpus-based approach to <AHEM/> expressive speech synthesis authors,” in *5th ISCA Speech Synthesis Workshop*, Pittsburgh, PA, 2004.

[8] R. Donovan and P. Woodland, “Improvements in an HMM-based speech synthesiser,” in *Eurospeech95*, Madrid, Spain, 1995, vol. 1, pp. 573–576.

[9] A. Black and K. Lenzo, “Building voices in the Festival Speech Synthesis System,” http://festvox.org/bsv/, 2000.

[10] K. Tokuda, H. Zen, and A. Black, “An HMM-based speech synthesis system applied to english,” in *IEEE TTS Workshop*, Santa Monica, CA, 2002.

[11] K. Prahallad, A. Black, and R. Mosur, “Sub-phonetic modeling for capturing pronunciation variation in conversational speech synthesis,” in *Proceedings of ICASSP 2005*, Toulouse, France, 2006.

[12] A. Black and P. Taylor, “Automatically clustering similar units for unit selection in speech synthesis,” in *Eurospeech97*, Rhodes, Greece, 1997, vol. 2, pp. 601–604.

[13] P Taylor, “Analysis and synthesis of intonation using the tilt model,” *Journal of the Acoustical Society of America*, vol. 107 3, pp. 1697–1714, 2000.

[14] A. Black and T. Schultz, “Speaker clustering for multilingual synthesis,” in *MultiLing 2006*, Stellenbosch, South Africa, April 2006.

[15] J. Kominek and A. Black, “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.

[16] J. Kominek, C. Bennett, B. Langner, and A. Toth, “The Blizzard Challenge 2005 CMU entry – a method for improving speech synthesis system,” in *Interspeech 2005*, Lisbon, Portugal., 2005.

[17] T. Toda, A. Black, and K. Tokuda, “Mapping from articulatory movements to vocal tract spectrum with Gaussian mixture model for articulatory speech synthesis,” in *5th ISCA Speech Synthesis Workshop*, Pittsburgh, PA, 2004, pp. 31–36.

[18] K. Tokuda, H. Zen, and Kitamura T., “Trajectory modeling based on HMMs with explicit relationship between static and dynamic features,” in *Eurospeech 2003*, Geneva, Switzerland, 2003.

[19] Stylianou, “Concatenative speech synthesis using a harmonic plus noise model,” in *3rd ESCA Workshop on Speech Synthesis*, Jenolan Caves, Australia., 1998, pp. 261–266.