SPEECH SYNTHESIS USING ARTIFICIAL NEURAL NETWORKS TRAINED ON CEPSTRAL COEFFICIENTS

Christine Tuerk *Tony Robinson*
Cambridge University Engineering Department, Trumpington Street, Cambridge, England.

ABSTRACT

Traditional synthesis systems often rely on a large set of rules and a hand-crafted set of synthesis parameters in order to produce output speech. Gathering the synthesis parameters and developing the rule set are very labour intensive tasks. This paper offers an alternative to these labour intensive tasks. A set of artificial neural networks (ANNs) are used to produce the filter parameters which drive a synthesiser. This set of ANNs is trained on data that is gathered fully automatically. The networks offer a storage-efficient means of synthesis without the need for explicit rule enumeration. The networks have the capability to produce temporal variation within a phonetic segment and differing outputs when input contexts are varied. Furthermore, the distributed architecture of the networks enables them to produce reasonable outputs when faced with novel inputs. In addition, a feedback mechanism incorporated into the architecture creates smooth transitions at segment boundaries.

1. INTRODUCTION

Developing an unlimited text to speech synthesis system is an enormous task. It requires converting the text of any desired utterance into an output waveform. Creating the actual waveform involves driving a speech synthesiser with appropriate control parameters. The determination of these parameters relies on combining knowledge from vast areas of speech research: acoustic-phonetic, lexical, syntactic, semantic, prosodic, and higher levels. Unifying this knowledge into a working system is a very labour intensive job.

Traditionally, the development of a synthesis system involves careful study of recorded speech. These recordings are sometimes stored parametrically for use in later synthesis. Alternatively, the recordings may be studied and generalisations in the form of a rule set created. Although synthesis systems built in this manner have been shown to produce quality output, their development involves incredible amounts of human effort.

In this paper we will explore an alternative method of performing some of the tasks required in a synthesis system. These alternatives include the fully automatic gathering of a synthesis parameter database and the use of a set of artificial neural networks (ANNs) for determining the control signals which actually drive the synthesiser. These alternatives are advantageous in that the amount of effort needed to create a synthesis system is drastically reduced. Furthermore, the ANN strategy offers a useful alternative for situations where a rule-enumerated system must fall back onto defaults.

The structure of the remaining sections is as follows. Section 2 gives an overview of the entire synthesis system. Section 3 then details the automatic data gathering procedure. The architecture and training of the ANN set are explained in Section 4 Synthesis results, evaluation and conclusions are given in Section 5

2. SYNTHESIS SYSTEM OVERVIEW

In our synthesis system, the synthesiser control parameters are determined by combining a phonetic string, an F0 contour, a duration pattern, and data from the synthesis parameter database into the signals required to control a linear prediction based synthesiser. This is illustrated in Figure 1. The work presented here has concentrated on building the synthesis control procedure via a set of ANNs and on providing the synthesis parameter database fully automatically. The phonetic string and F0 contour are obtained by integrating modules of the MITalk system [1] into our system. The text of the desired utterance is given to the MITalk FORMAT, DECOMP, PARSER and SOUND1 modules. This produces a proposed phoneme string. The output is also given to the MITalk PHONO1, PHONO2, PROSOD and F0TARG modules. This produces an F0 contour. The duration model used by the control procedure is described in [2].

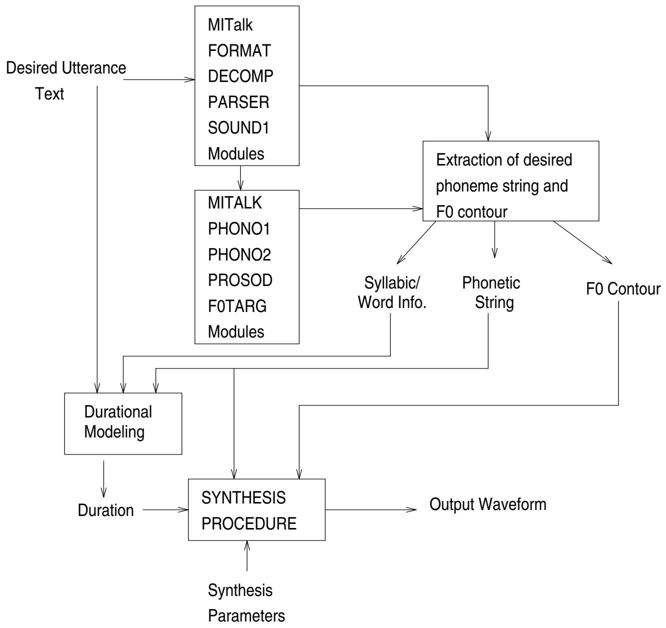

Figure 1. Overview of the proposed synthesis system

The proposed system utilises the phoneme as the basic synthesis unit. Although larger units tend to retain inherent naturalness, they sacrifice flexibility to create unlimited synthesis messages. Smaller units, such as the phoneme, allow great flexibility to produce unlimited messages but introduce significant coarticulation concerns and loss of natural flow between segments. Many speech synthesis systems use the diphone as the basic synthesis unit by claiming that joining synthesis segments at the middle of a phoneme reduces spectral mismatch. This relies on the assumption that each phoneme reaches a steady state target near its midpoint in time. However, if diphthongs and laterals are considered, spectral transitions through the duration of the realisation are expected, and thus, no steady state is reached. Without a defined steady state point, finding the location for segment concatenation becomes a significant problem. Phoneme boundaries have been chosen as the point to join synthesis segments because that is precisely where spectral transitions are expected to occur. This is not to say that coarticulatory concerns are being ignored. It is fully realised that transitions between phoneme segments must be smooth in order to produce intelligible, quality speech. These coarticulatory concerns are addressed in the way the synthesis parameter database is gathered. Although the phoneme is the basic unit, triphone context examples are gathered giving multiple examples in the database for a single phoneme. The artificial neural networks also use a feedback mechanism which smooths the boundaries between phonetic segments.

3. DATA GATHERING

The proposed synthesis database is gathered fully automatically from the TIMIT acoustic-phonetic database [3]. This automatic gathering means that a far greater amount of data can be analysed than would be humanly possible in a similar amount of time. Our analysis led to a total of 184,511 phoneme tokens from which to build our database. The database is divided into 50 subsets - one for each phoneme used as our basic synthesis unit. Figure 2 shows the sequence of steps involved in gathering synthesis data for each of these phoneme subsets.

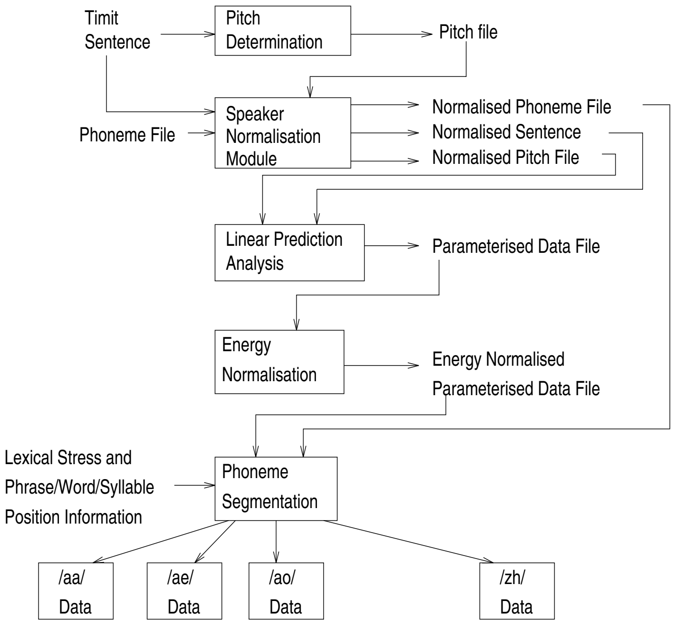

Figure 2. Flow of data gathering steps

Creating the data files for each phoneme begins by taking each of the TIMIT sentences and performing a pitch analysis algorithm. The waveform is then speaker normalised to average male characteristics according to a method described in [4]. The normalised sentence and pitch files are used in a pitch-synchronous linear prediction analysis module to create a parameterised data file. Each parameterised data file consists of pitch, probability of voicing, root mean square (r.m.s.) energy, and 20 cepstral coefficients for each analysis frame. In addition, the average sentence r.m.s. level is normalised to a reference level.

The energy normalised parameter files are divided into 50 phoneme subset files. For each example in the subset files, relevant information regarding the utterance from which it was taken must be included. The context of the left and right phonemes, the lexical stress of the left, current and right phonemes, and sentence position (including syllable, word and phrase information) are stored. In some cases, there are multiple phonetic examples having the same context, stress and position attributes. In these cases, it is desirable to replace these multiple examples by a "central" example representative of the entire group. Finding this representative example is somewhat complicated by the fact that each example may be made up of a different number of analysis frames (different durations). In addition, each analysis frame is made of multiple parameter values.

A representative "central" example is chosen by calculating a distance measure between each pair of phonetic examples. The example which has the least total distance between itself and all other examples is selected as the representative example. The distance measure chosen is a Euclidean distance between the cepstral coefficients which is a very good measure of spectral similarity [5]. Thus, the chosen representative example is the central example in spectral terms. Each pair of examples are time aligned by a dynamic time warping procedure.

Finding a central example in this manner enables the 184,511 examples to be reduced to 39,810 examples. This reduction leads to a set of templates that may be directly used in conjunction with a control strategy for synthesis. We will use these templates to train a set of artificial neural networks. The reduction in examples significantly decreases the amount of training time needed.

4. SYNTHESIS CONTROL BY ANNS

A synthesis control strategy must determine the exact parameters which drive a speech synthesiser. This usually entails an elaborate set of rules dictating the selection and concatenation of synthesis parameters stored in the synthesis database. The strategy adopted in our system differs. The duration model and F0 contour are used to determine the number of frames and the length of each for a given phonetic segment. The actual synthesis parameters are then found by using a set of trained ANNs. This section details the structure and use of these ANNs.

ANNs have been used in many speech related tasks. In speech recognition systems, these networks have been used as part of auditory preprocessors, vector quantisers and static pattern classifiers [6]. They have also been applied to the task of discriminating between voiced, unvoiced and silence portions of speech [7] and to the task of normalising input vectors of a novel speaker to a reference speaker [8]. Networks have also been applied to various synthesis-related tasks. Text-to-phoneme conversion has been done via networks [9]. Networks have been used to perform syntactic classification of isolated English words [10] and to predict pitch contours of entire sentences [11]. Neural networks have been used to learn typical formant values of CVC syllables [12]. Articulatory synthesis parameters have been learned through an assembly of ANNs [13]. The have also been used to convert formant values to articulatory parameters [14].

ANNs are not always the appropriate tool for a given problem. Their slow speed of model generation means wasted effort for simple problems. Situations where an ANN might be the appropriate tool include: problems needing a sophisticated smoothing algorithm when there are insufficient training examples to populate the input space, problems where the form of the solution is unknown and there is no obvious parametric way to deduce it, and problems where a series of transformations can exploit high order correlations in the input data. The task of producing synthesis parameters for different contexts, sentence positions and lexical stress attributes fits all of these situations.

We have decided to use a set of 50 ANNs (one for each phoneme). This choice gives each network a reasonably sized task in addition to the possibility of reducing overall training time by using different machines in parallel.

Figure 3 displays the architecture of each individual network. A 3 layer network is used, consisting of 60 input nodes, 65 hidden nodes and 22 output nodes. Including bias units, each network requires a total of only 5417 weights. The output of the final layer is linear while the output of the hidden layer is sigmoidal. In addition, the values of the output nodes are fed back directly into the input layer. A similar feedback structure was used by Jordan [15] to maintain serial order in network states.

The representation of information in a network's input layer is of great importance to its overall performance. We want our networks to produce the probability of voicing, the r.m.s. energy level and the 20 cepstral filter coefficients for each synthesis frame. The input given to produce each output frame includes the context, position, stress and timing of the frame. How this information is encoded can make a large difference in the performance of the network. Further, since we hope the networks will be able to generalise to novel situations, the input structure should easily reflect similarities among input classes. For example, the coding of a /bcl/ should be much closer to the coding of /pcl/ than to an /s/. A distributed coding scheme can accomplish this. Phoneme identities are encoded according

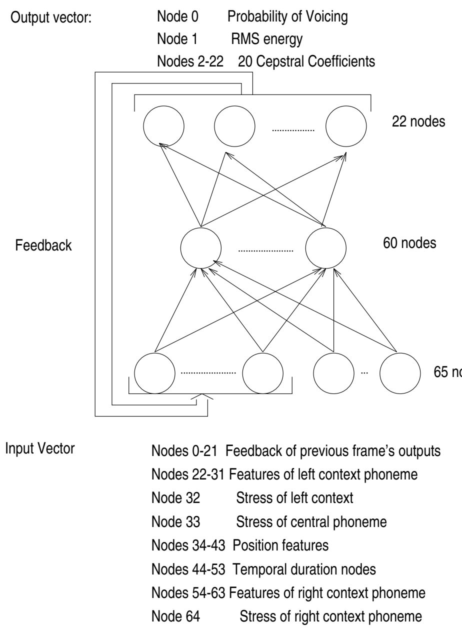

Figure 3. Neural network structure

the variables consonantal/vocalic, consonant position, consonant type, vocalic high versus low, vocalic front versus back, vocalic long versus short, vocalic glide type, schwa versus non-schwa, and silence type. Sentence position is also encoded in a distributed manner including binary values for syllable, word and phrase position.

The standard feed-forward artificial neural network is by nature static where a single input vector is related to a single output vector. However, our problem is not static. Transitions between neighbouring output vectors should be smooth. In addition, some indication of temporal variation within a phoneme is needed. These concerns are addressed in the network architecture. A feedback mechanism places the outputs of the previous frame into the input vector of the following frame. This helps to smooth transitions between neighbouring vectors, especially on segment boundaries. Temporal variation is addressed by dedicating a number of input units to coding relative position within a segment.

The above described architecture was used to build 50 networks, one for each phoneme to be trained. The training method used was a back propagation version similar to that of Jacobs [16].

5. RESULTS AND CONCLUSIONS

Evaluation of a synthesis system is always a subjective task. Without a panel of independent listeners (as used in Diagnostic Rhyme Tests) it is difficult to present an evaluation regarding synthesis quality without bias. Thus the following analysis attempts to evaluate the developed synthesis by looking more objectively at its capabilities and shortcomings. In the following analysis, some reference to audible output quality was necessary, however, attempts have been made to keep these minimal.

At a fundamental level, a synthesiser must first be capable of producing all the sounds which serve as its basic synthesis unit. This capability was tested by using the synthesiser to produce phonetically rich sentences and listening to the output. In addition, spectrograms of the synthetic utterances were studied to ensure that spectral characteristics matched those expected from established acoustic-phonetic studies. Although the spectrograms did not always clearly exhibit higher formants distinctly, lower formants were clear and located in expected regions. The formant tracks of at least the first two formants of vowels and semi-vowels were clear. Closures were clean and bursts were distinct. Furthermore, fricatives, both voiced and unvoiced, had appropriate high frequency energy patterns. Diphthongs and glides exhibited appropriate transitions throughout the duration of the segment.

Once it has been established that the networks are capable of producing the basic sound units, it is also important to show that the networks can produce different outputs when given different contexts, stress, and sentence position information as inputs. Furthermore, the networks should be able to produce variation within a phonetic segment, especially for diphthongs and glides. These capabilities are illustrated in Figure 4. The left side of the figure shows waveforms and spectrograms of [iy] examples in different contexts. Differences in the spectrum of each example may also be seen. The [iy] preceded by the [h] (the first example) has a second formant which remains above 2000 Hz during the entire duration of the phoneme. In the following example, [iy] in the context of [m] and [ix], the second formant remains around 2000 Hz throughout the phoneme. The final 2 examples show movement in the second formant as it initially starts low and moves upward. The right side of the figure shows examples of [w] and [ey] displaying appropriate formant transitions.

Another important capability of a synthesiser is the way coarticulatory concerns are addressed. This was explored by comparing the output of the network controlled synthesis with output produced by concatenating appropriate "central" templates. Immediate differences in the smoothness of transitions between phonetic segments was noticed. This is illustrated in Figure 5. The left portion of the figure shows the waveform and spectrogram of template based synthesis while the right portion gives the output from the networks. The template synthesis is quite "chunked". This results in speech that sounds very disconnected. The network speech, however, is much smoother and is preferable to the template speech.

Perhaps the greatest strength of the ANN control system is its ability to produce novel outputs (i.e. outputs not contained in the training set). In many rule based systems, novel outputs are coped with by reverting to default parameter sets. The network scheme, however, utilises similarities in input codes to produce viable output parameters. For example, one network's training vectors included an utterance in the context [p][ax] in an unstressed word initial position. When the novel input of [p][eh] in an unstressed word initial position was presented to the network, it was able to draw on similarities between the two examples to produce acceptable output. This was true in a wide variety of cases. Audibly, the novel outputs are intelligible and of the same quality as the learned outputs.

As discussed previously, an objective evaluation of a synthesis system is difficult. Attempts have been made to show that the neural based synthesiser can adequately perform essential synthesis tasks. First, it is capable of accurately producing the basic sound units. Second, it can produce various realisations of the same phoneme when different input contexts are supplied. Third, it exhibits variation within a phonetic example as one would expect when listening to actual speech. Fourth, it is able to produce reasonable synthesis parameters when faced with situations not included in the training data. Furthermore, the ANNs serve as a very efficient means of storing synthesis data. They also alleviate the burden of explicit rule generation for producing synthetic messages. Finally, the ANNs address the primary weaknesses of a template based system, poor smoothing and poor template selection in novel situations.

6. ACKNOWLEDGEMENTS

This work has been supported in part by the National Science Foundation and the UK Science and Engineering Research Council.

REFERENCES

[1] J. Allen, M.S. Hunnicutt, and D. Klatt. *From Text to Speech: The MITalk System.* Cambridge University Press, 1987.

[2] C.M. Tuerk and A.J. Robinson. A multiple speaker phoneme duration model. In *Proceedings of the Institute of Acoustics Autumn Conference*, 1992.

[3] L.F. Lamel, R.H. Kasel, and S. Seneff. Speech database development: Design and analysis of the acoustic-phonetic corpus. In *Proceedings of the DARPA Speech Recognition Workshop*, pages 26–32, 1987.

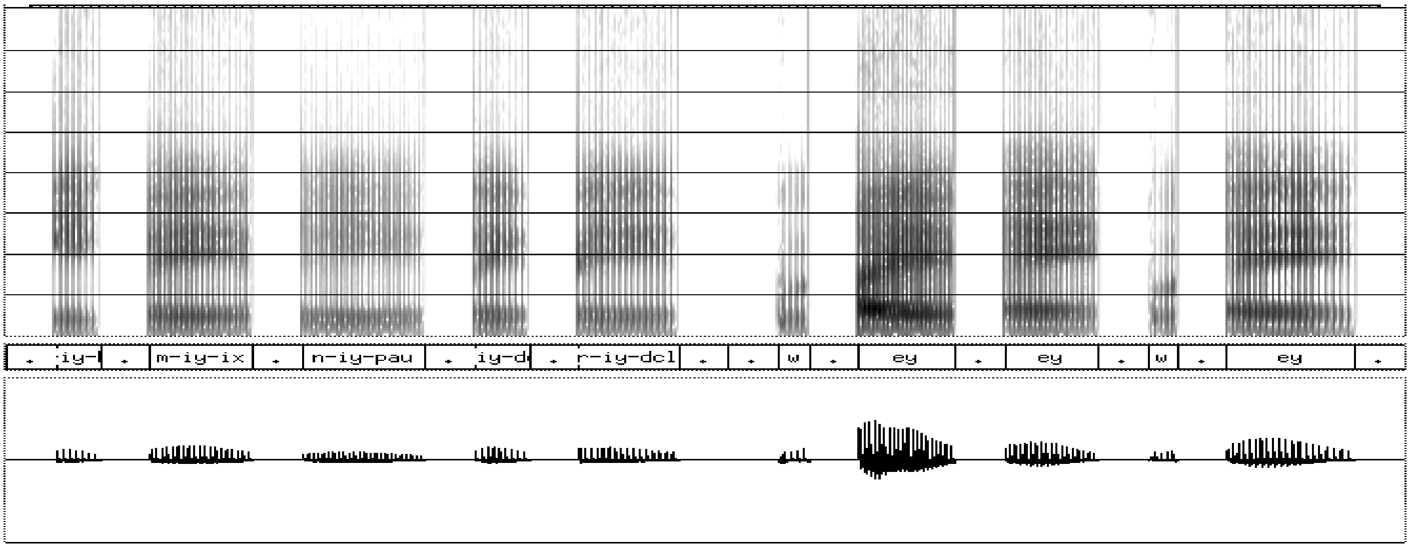

Figure 4. Examples of [iy], [w] and [eh] realisations in different contexts

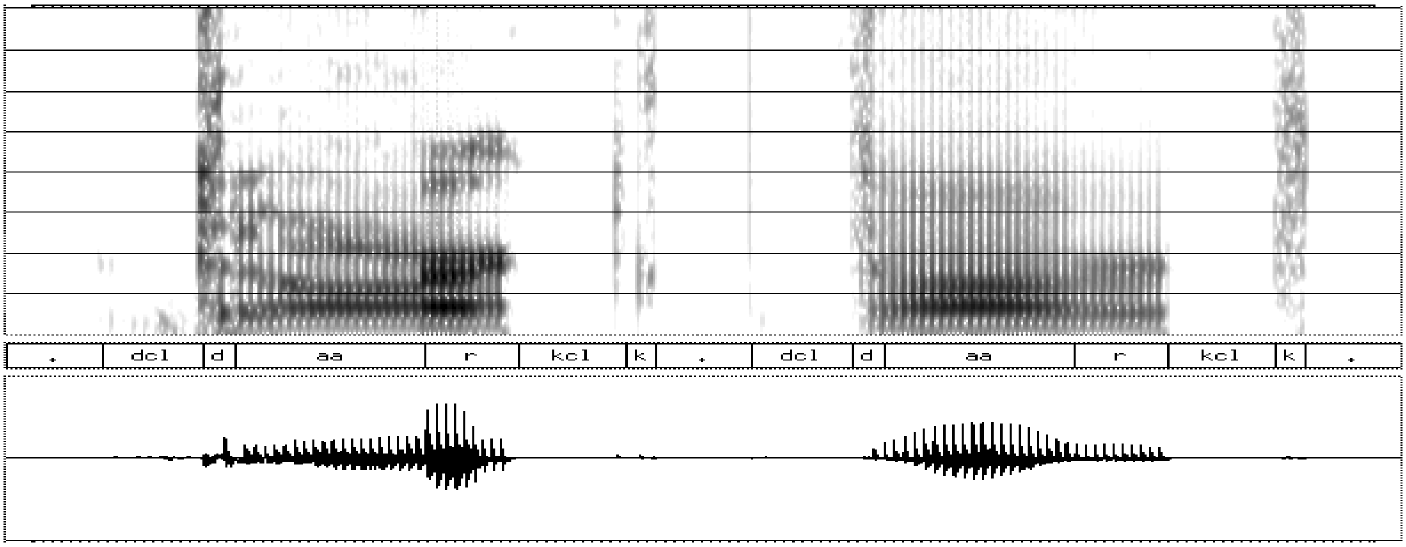

Figure 5. The word "dark" produced by template and network based synthesis

[4] C.M. Tuerk and A.J. Robinson. A new frequency shift function for reducing inter-speaker variance. In *Eurospeech 93*, 1993.

[5] A.H. Gray, Jr. and J.D. Markel. A spectral-flatness measure for studying the autocorrelation method of linear prediction of speech analysis. *IEEE ASSP*, ASSP-22:207–217, 1974.

[6] R.P. Lippmann. Review of neural networks for speech recognition. *Neural Computation*, 1:1–38, 1988.

[7] T. Ghiselli-Crippa and A. El-Jaroudi. A fast neural net training algorithm and its application to voiced-unvoiced-silence classification of speech. In *Proc. ICASSP*, pages 441–444, 1991.

[8] X.D. Huang, K.F. Lee, and A. Waibel. Connectionist speaker normalization and its applications to speech recognition. In *Proceedings of the IEEE Workshop on Neural Networks for Signal Processing*, pages 357–366, Princeton, N.J., October 1991.

[9] T.J. Sejnowski and C.R. Rosenberg. NETtalk: A parallel network that learns to read aloud. Technical Report JHU/EECS-86/01, The Johns Hopkins University Technical Report, 1986.

[10] N.P. Warren and W.A. Ainsworth. Automatic syntactic classification of isolated English words using connectionist architectures. In *Proc. ESCA Workshop on Speech Synthesis, Autrans*, pages 101–104, September 1990.

[11] C. Traber. F0 generation with a data base of natural F0 patterns and with a neural network. In *Proc. ESCA Workshop on Speech Synthesis, Autrans*, pages 141–144, September 1990.

[12] V.V. Kumar, S.C. Ahalt, and A.K. Krishnamurthy. Phonetic to acoustic mapping using recurrent neural networks. In *Proc. ICASSP*, pages 753–756, 1991.

[13] M.G. Rahim, W.B. Kleijn, J. Schroeter, and C.C. Goodyear. Acoustic to articulatory parameter mapping using an assembly of neural networks. In *Proc. ICASSP*, pages 485–488, 1991.

[14] A. Soquet, M. Saerens, and P. Jospa. Acoustic-articulatory inversion based on a neural controller of a vocal tract model. In *Proc. ESCA Workshop on Speech Synthesis, Autrans*, pages 71–74, September 1990.

[15] M.I. Jordan. Serial order: A parallel distributed processing approach. Technical Report ICS Report 8604, Institute for Cognitive Sciences, University of California, San Diego, 1986.

[16] R.A. Jacobs. Increased rates of convergence through learning rate adaptation. *Neural Networks*, 1:295–307, 1988.