Trajectory Mixture Density Networks with Multiple Mixtures for Acoustic-articulatory Inversion

Korin Richmond

Centre for Speech Technology Research
Edinburgh University, Edinburgh, United Kingdom
korin@cstr.ed.ac.uk

Abstract. We have previously proposed a trajectory model which is based on a mixture density network (MDN) trained with target variables augmented with dynamic features together with an algorithm for estimating maximum likelihood trajectories which respects the constraints between those features. In this paper, we have extended that model to allow diagonal covariance matrices and multiple mixture components in the trajectory MDN output probability density functions. We have evaluated this extended model on an inversion mapping task and found the trajectory model works well, outperforming smoothing of equivalent trajectories using low-pass filtering. Increasing the number of mixture components in the TMDN improves results further.

1 Introduction

Mainstream speech technology, such as automatic speech recognition and concatenative speech synthesis, is strongly focused on the acoustic speech signal. This is natural, considering the acoustic domain is where the speech signal exists in transmission between humans, and we can conveniently measure and manipulate an acoustic representation of speech. However, an articulatory representation of speech has certain properties which are attractive and which may be exploited in modelling. Speech articulators move relatively slowly and smoothly, and their movements are continuous; the mouth cannot "jump" from one position to the next. Using knowledge of the speech production system could improve speech processing methods by providing useful constraints. Accordingly, there is growing interest in exploiting articulatory information and representations in speech processing, with many suggested applications; for example, low bit-rate speech coding [1], speech analysis and synthesis [2], automatic speech recognition [3, 4], animating talking heads and so on.

For an articulatory approach to be practical, we need convenient access to an articulatory representation. Recent work on incorporating articulation into speech technology has used data provided by X-ray microbeam cinematography and electromagnetic articulography (EMA). These methods, particularly the latter, mean we are now able to gather reasonably large quantities of articulatory data. However, they are still invasive techniques and require bulky and expensive experimental setups. Therefore, there is interest in developing a way to recover an articulatory representation from the acoustic speech signal. In other words, for a given acoustic speech signal we aim to estimate the underlying sequence of articulatory configurations which produced it. This is termed acoustic-articulatory inversion, or the inversion mapping.

The inversion mapping problem has been the subject of research for several decades. One approach has been to attempt analysis of acoustic signals based on mathematical models of speech production [5]. Another popular approach has been to use articulatory synthesis models, either as part of an analysis-by-synthesis algorithm [6], or to generate acoustic-articulatory corpora which may be used with a code-book mapping [7] or to train other models [8]. Much of the more recent work reported has applied machine learning models to human measured articulatory data, including artificial neural networks (ANNs) [9], codebook methods [10] and GMMs [11].

The inversion mapping is widely regarded as difficult because it may be an ill-posed problem; multiple evidence exists to suggest the articulatory-to-acoustic mapping is many-to-one, which means that instantaneous inversion of this mapping results in a one-to-many mapping. If this is the case, an inversion mapping method must take account of the alternative articulatory configurations possible in response to an acoustic vector.

In previous work [12, 9], we have successfully employed the mixture density network (MDN) [13] to address this problem. The MDN provides a probability density function (pdf) of arbitrary complexity over the target articulatory domain which is conditioned on the acoustic input. In [14], we began to extend this work to provide a statistical trajectory model, termed the Trajectory MDN, along similar lines as the HMM-based speech production model of [15] and the GMM-based inversion mapping of [11]. This was achieved by augmenting the static articulatory target data with dynamic delta and deltadelta features and incorporating the maximum likelihood parameter generation (MLPG) algorithm [16]. This allows to calculate the maximum likelihood estimate of articulatory trajectories which respect the constraints between the static and derived dynamic features.

This paper seeks to further the work in [14] with three specific aims: 1) to evaluate an extension to the TMDNs in [14] (which were limited to using spherical covariance matrices) that allows mixture models with diagonal covariance matrices. 2) to evaluate the new implementation of TMDN on the full set of articulator channels, and in comparison with a low-pass filtering approach previously reported. 3) to evaluate TMDNs with multiple mixture components.

2 The Trajectory Mixture Density Network Model

We give here a very brief introduction to the MDN, and describe how it may be extended with the MLPG algorithm to give a trajectory model. For full details of the MDN and MLPG, the reader is referred to [13] and [16] respectively. To avoid introducing unnecessary confusion, we have attempted to retain the original notation as far as possible.

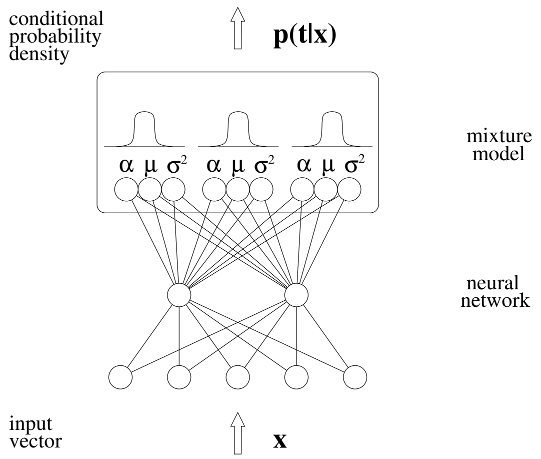


Fig. 1. *The mixture density network combines a mixture model and a neural network.*

2.1 Mixture density networks

The MDN combines a mixture model with an ANN. Here, we will consider a multilayer perceptron and Gaussian mixture components. The ANN maps from the input vector $\mathbf{x}$ to the control parameters of the mixture model (priors α, means μ and variances σ^2), which in turn gives a pdf over the target domain, conditioned on the input vector $p(\mathbf{t}|\mathbf{x})$. The toy-example MDN in Figure 1 takes an input vector $\mathbf{x}$ (dimensionality 5) and gives the conditional probability density of a vector $\mathbf{t}$ (dimensionality 1) in the target domain. This pdf takes the form of a GMM with 3 components, so it is given as:

$$p(\mathbf{t}|\mathbf{x}) = \sum_{j=1}^{M} \alpha_j(\mathbf{x}) \phi_j(\mathbf{t}|\mathbf{x}) \tag{1}$$

where M is the number of mixture components (in this example, 3), $\phi_j(\mathbf{t}|\mathbf{x})$ is the probability density given by the jth kernel, and $\alpha_j(\mathbf{x})$ is the prior for the jth kernel.

In order to constrain the GMM priors to within the range $0 \leq \alpha_j(\mathbf{x}) \leq 1$ and to sum to unity, the *softmax* function is used

$$\alpha_j = \frac{\exp(z_j^\alpha)}{\sum_{l=1}^{M} \exp(z_l^\alpha)} \tag{2}$$

where z_j^α is the output of the ANN corresponding to the prior for the jth mixture component. The variances are similarly related to the outputs of the ANN as

$$\sigma_j = \exp(z_j^\sigma) \tag{3}$$

where z_j^σ is the output of the ANN corresponding to the variance for the jth mixture component. This avoids the variance becoming ≤ 0. Finally, the means are represented directly:

$$\mu_{jk} = z^{\mu}_{jk} \tag{4}$$

where z^{μ}_{jk} is the value of the output unit corresponding to the kth dimension of the mean vector for the jth mixture component.

Training the MDN aims to minimise the negative log likelihood of the observed target data points

$$E = -\sum_{n} \ln \left\{ \sum_{j=1}^{M} \alpha_j(\mathbf{x}^n)\phi_j(\mathbf{t}^n|\mathbf{x}^n) \right\} \tag{5}$$

given the mixture model parameters. Since the ANN part of the MDN provides the parameters for the mixture model, this error function must be minimised with respect to the network weights. The derivatives of the error at the network output units corresponding separately to the priors, means and variances of the mixture model are calculated (see [13]) and then propagated back through the network to find the derivatives of the error with respect to the network weights. Thus, standard non-linear optimisation algorithms can be applied to MDN training.

2.2 Maximum likelihood parameter generation

The first step to an MDN-based trajectory model is to train an MDN with target feature vectors augmented with dynamic features (i.e. deltas and deltadeltas), derived from linear combinations of a window of static features. For the sake of simplicity we will first consider MDNs with a single Gaussian distribution and a single target static feature c_t at each time step. Given the output of this MDN in response to a sequence of input vectors, in order to generate the maximum likelihood trajectory, we aim to maximize $P(\mathbf{O}|\mathbf{Q})$ with respect to $\mathbf{O}$, where $\mathbf{O} = [\mathbf{o}_1^T, \mathbf{o}_2^T, ..., \mathbf{o}_T^T]^T$, $\mathbf{o}_t = [c_t, \Delta c_t, \Delta\Delta c_t]$ and $\mathbf{Q}$ is the sequence of Gaussians output by our MDN. The relationship between the static features and those augmented with derived dynamic features can be arranged in matrix form

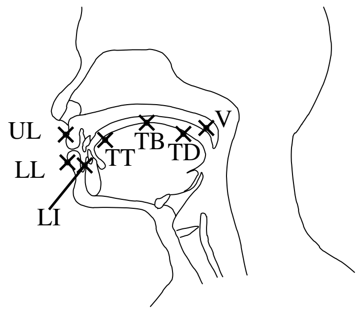
$$\mathbf{O} = \mathbf{W}\mathbf{C} \tag{6}$$

where $\mathbf{C}$ is a sequence of static features and $\mathbf{W}$ is a transformation matrix composed of the coefficients of the delta and deltadelta calculation window and $\mathbf{0}$. Under the condition expressed in Eq. 6, maximising $P(\mathbf{O}|\mathbf{Q})$ is equivalent to maximising $P(\mathbf{WC}|\mathbf{Q})$ with respect to $\mathbf{C}$. By setting

$$\frac{\partial \log P(\mathbf{WC}|\mathbf{Q})}{\partial \mathbf{C}} = 0 \tag{7}$$

a set of linear equations is obtained (see [16] for the details)

$$\mathbf{W}^T\mathbf{U}^{-1}\mathbf{W}\mathbf{C} = \mathbf{W}^T\mathbf{U}^{-1}\mathbf{M}^T \tag{8}$$



label	articulator	label	articulator
UL	Upper lip	TT	Tongue tip
LL	Lower lip	TB	Tongue body
LI	Lower incisor	TD	Tongue Dorsum
V	Velum		

Fig. 2. *Placement of EMA receiver coils in the MOCHA database for speaker* `fsew0`. *Coil placement abbreviations may be suffixed with "_x" and "_y" to designate the x- and y-coordinate for a given coil in the midsagittal plane respectively.*

where $\mathbf{M}^T = [\mu_{q_1}, \mu_{q_2}, ..., \mu_{q_T}]$ and $\mathbf{U}^{-1} = diag[\mathbf{U}_{q_1}^{-1}, \mathbf{U}_{q_2}^{-1}, ..., \mathbf{U}_{q_T}^{-1}]$ (μ_{q_T} and $\mathbf{U}_{q_t}^{-1}$ are the 3×1 mean vector and 3×3 (diagonal) covariance matrix respectively). Solving Eq. 8 for $\mathbf{C}$ computes the maximum likelihood trajectory.

To extend the MLPG algorithm to the case of a sequence of pdfs with multiple mixture components, we make use of an iterative EM method described in [16]. Essentially, for each time frame at each iteration, instead of using the means and covariances of a single Gaussian in (8), we use a weighted sum of these parameters of the multiple mixture components, weighed by their occupancy probabilities (i.e. posterior probability of each component given the augmented observation sequence $\mathbf{O}$). Hence, we first choose an initial static feature trajectory $\mathbf{C}$ and use this to calculate the occupancy probabilities using the forward backward algorithm. We can then solve (8) using the weighted means and covariances to obtain an updated feature trajectory $\mathbf{C}'$, which is then used to calculate updated occupancy probabilities. This two stage iteration is repeated until convergence.

3 Inversion mapping experiment

3.1 MOCHA articulatory data

The multichannel articulatory (MOCHA) dataset [17] used for the experiments in this paper gives the acoustic waveform recorded at the same time as electromagnetic articulograph (2D EMA) data. The sensors shown in Figure 2 provide x- and y-coordinates in the midsagittal plane at 500Hz sample rate. Speakers were recorded reading a set of 460 short, phonetically-balanced British-TIMIT sentences. Female speaker `fsew0` was used for the experiments here. This is the same data set as used previously [9, 14], and so enables comparison with those and similar results reported in the literature (e.g. [11]).

Data processing The acoustic data was converted to frames of 20 melscale filterbank coefficients using a Hamming window of 20ms with a shift of 10ms.

These were z-score normalised and scaled to the range [0.0,1.0]. The EMA trajectories were downsampled to match the 10ms shift rate, then z-score normalised and scaled to the range [0.1,0.9] using the normalisation method described in [12]. Frames of silence at the beginning and end of the files were discarded, using the labelling provided with MOCHA.

368 utterances were used for the training set, and the validation and test sets contained 46 utterances each (the same subsets as [9, 14]). A context window of 20 consecutive acoustic frames was used as input to the TMDN, which increased the order of the acoustic vector paired with each articulatory vector to 400.

3.2 Method

We trained TMDNs with 1, 2 and 4 mixture components for each of the 14 EMA channels, making a total of 42 models trained. In [14], we trained separate MDNs for the static, delta and deltadelta features for each articulatory channel, because the implementation limited output pdfs to spherical covariance. Here, in contrast, our implementation has been extended and now allows diagonal covariance matrices, and so the three feature streams for each articulator channel were trained in a single network. All networks contained a hidden layer of 80 units. The scaled conjugate gradients non-linear optimisation algorithm was run for a maximum of 4000 epochs, and the separate validation set was used to identify the point at which an optimum appeared to have been reached. The validation error was calculated in terms of RMS error between the target trajectories and those resulting from the Trajectory MDN. This differs from using simply the likelihood of the target data given the pdfs output by the MDN. To generate output trajectories from the TMDN, we simply ran the input data for an utterance through the TMDNs for each articulatory channel, and then ran the MLPG algorithm on the resulting sequences of pdfs over the static and dynamic feature spaces.

To evaluate the Trajectory MDN, we compared the resulting trajectories with those of the output units corresponding to the mean of the static feature alone. This output is in theory approximately equivalent to that of an MLP (with linear output activation function) trained with a standard least-squares error function[1]. In this way, we can directly observe the effect of using the augmented features without considering the effects of two systems having been trained differently. Finally, we also low-pass filtered the static mean trajectories as a smoothing step which has been shown in the past to improve inversion results [12, 11], and compared those smoothed trajectories with the TMDN output.

4 Results

Table 1 lists the results of 14 TMDNs trained on each articulatory channel separately, using an output pdf containing a single Gaussian. Two error metrics have been used: correlation between the target and output trajectories, and root

[1] although the MLP component of the TMDN here has been trained with augmented target features, which from comparison with previous results, e.g. [9], seems beneficial

Table 1. *Comparison of results for Trajectory MDNs (TMDN) with a single Gaussian with the MLP described in [9]. Exactly the same training, validation and testing datasets have been used. Average RMSE(mm) in [9] was* 1.62*mm, compared with* 1.4*mm here.*

	Correlation		RMSE(mm)		RMSE(mm)
Channel	MLP	TMDN	MLP	TMDN	reduction %
ul_x	0.58	0.68	0.99	0.90	9.5
ul_y	0.72	0.79	1.16	1.05	9.9
ll_x	0.60	0.69	1.21	1.10	9.2
ll_y	0.75	0.83	2.73	2.27	16.8
li_x	0.56	0.63	0.89	0.82	8.1
li_y	0.80	0.85	1.19	1.03	13.3
tt_x	0.79	0.85	2.43	2.12	12.9
tt_y	0.84	0.90	2.56	2.08	18.7
tb_x	0.81	0.85	2.19	1.96	10.4
tb_y	0.83	0.89	2.14	1.76	17.6
td_x	0.79	0.84	2.04	1.85	9.5
td_y	0.71	0.82	2.31	1.89	18.2
v_x	0.79	0.86	0.42	0.35	15.6
v_y	0.77	0.83	0.41	0.37	10.2

mean square error (RMSE) expressed in millimetres. The table also lists the results previously reported in [9], which used an MLP with exactly the same dataset, for comparison. It can be seen that the improvement is substantial. By way of further comparison with other studies, [11] reported an average RMS error of 1.45mm for MOCHA speaker `fsew0`.

Table 2. *Channel-specific cutoff frequencies used for low pass filtering.*

	ul	ll	li	tt	tb	td	v
_x	3 Hz	3 Hz	3 Hz	6 Hz	6 Hz	7 Hz	5 Hz
_y	5 Hz	8 Hz	7 Hz	9 Hz	7 Hz	6 Hz	5 Hz

In order to investigate the effect of using dynamic features and the MLPG algorithm within the Trajectory MDN, we have compared these results for TMDNs with a single Gaussian with those obtained using low-pass filtering, as described in [12, 11]. Table 3 compares three conditions: *"TMDN"*, *"static only"* and *"static lpfilt"*. For the *"static only"* condition, we have used the TMDN's output corresponding to the mean for the static target feature as the output trajectory. For the *"static lpfilt"* condition, we have further low-pass filtered the static mean above using the cutoff frequencies listed in Table 2. These channel-specific cutoff frequencies were determined empirically in [12], and are very similar to those given in [11]. As expected, it can be seen that low-pass filtering improves results for all channels. However, using the dynamic features and the MLPG algorithm in the Trajectory MDN results in the best performance, with improvements varying between 0.6 and 2.4% over low-pass filtering.

Table 3. *Comparison of correlation and RMS error (in millimetres) for Trajectory MDN model ("TMDN") with the static mean MDN output only ("static only") and low-pass filtered static mean ("static lpfilt"). The TMDN here has a single Gaussian.*

	Correlation			RMSE(mm)			RMSE(mm)
Channel	static only	static lpfilt	TMDN	static only	static lpfilt	TMDN	reduction %
ul_x	0.63	0.67	0.68	0.93	0.90	0.90	0.6
ul_y	0.74	0.77	0.79	1.13	1.06	1.05	1.5
ll_x	0.64	0.69	0.69	1.17	1.11	1.10	1.0
ll_y	0.81	0.83	0.83	2.40	2.31	2.27	1.6
li_x	0.57	0.62	0.63	0.88	0.84	0.82	2.4
li_y	0.83	0.84	0.85	1.07	1.05	1.03	1.5
tt_x	0.82	0.84	0.85	2.26	2.14	2.12	1.0
tt_y	0.88	0.89	0.90	2.19	2.12	2.08	1.8
tb_x	0.83	0.85	0.85	2.05	1.99	1.96	1.2
tb_y	0.87	0.89	0.89	1.88	1.80	1.76	1.8
td_x	0.81	0.83	0.84	1.95	1.88	1.85	2.1
td_y	0.78	0.81	0.82	2.04	1.92	1.89	1.7
v_x	0.84	0.85	0.86	0.37	0.36	0.35	1.2
v_y	0.80	0.82	0.83	0.39	0.37	0.37	1.6

The improvements over using low-pass filtering shown in Table 3, although consistent, are not huge. However, in contrast to low-pass filtering, the TMDN is able to make use of multiple mixture components, which can potentially increase performance further. Table 4 performs this comparison, by the addition of results for TMDNs with 2 and 4 mixture components. We see that in the majority of cases increasing the number of mixture components improves results, e.g. by up to 8.3% in the case of the tt_y channel using 4 mixture components.

Finally, Figure 3 gives a qualitative demonstration of the nature of this improvement. In these plots we compare the tt_y trajectory estimated by the TMDN with 4 mixture components (bottom plot) with the low-pass filtered static mean (top plot). It can be seen in several places that the TMDN with 4 mixtures is substantially closer to the real target trajectory.

5 Conclusion

The results of this paper show we have successfully extended the Trajectory MDN first described in [14] to allow diagonal covariance matrices. For all 14 articulator channels tested, the TMDN with a single Gaussian output pdf performed better than low-pass filter smoothing. Increasing the number of mixture components improved results further. This is a unique advantage of the TMDN model over smoothing single trajectories using, for example, low-pass filters.

6 Acknowledgments

Many thanks to Junichi Yamagishi for useful discussion pertaining to implementation. This work was supported by EPSRC grant EP/E027741/1.

Table 4. *Comparison of RMS error (in millimetres) between using the low-pass filtered static feature mean ("static lpfilt") and Trajectory MDNs with 1, 2 or 4 mixture components. Average (min) RMSE=*1.37*mm*

	static	TMDN			opt #	% best
Channel	lpfilt	1 mix	2 mix	4 mix	mixes	reduction
upper lip x	0.90	0.90	0.90	0.91	1	0.6
upper lip y	1.06	1.05	1.03	1.06	2	3.3
lower lip x	1.11	1.10	1.10	1.12	1	1.0
lower lip y	2.31	2.27	2.20	2.22	2	4.7
lower incisor x	0.84	0.82	0.80	0.81	2	4.2
lower incisor y	1.05	1.03	1.04	1.03	1	1.5
tongue tip x	2.14	2.12	2.09	2.10	2	2.1
tongue tip y	2.12	2.08	1.98	1.94	4	8.3
tongue body x	1.99	1.96	1.97	1.98	1	1.2
tongue body y	1.80	1.76	1.73	1.78	2	3.5
tongue dorsum x	1.88	1.85	1.81	1.83	2	4.1
tongue dorsum y	1.92	1.89	1.85	1.88	2	3.6
velum x	0.36	0.35	0.35	0.35	2	3.3
velum y	0.37	0.37	0.36	0.37	2	2.8

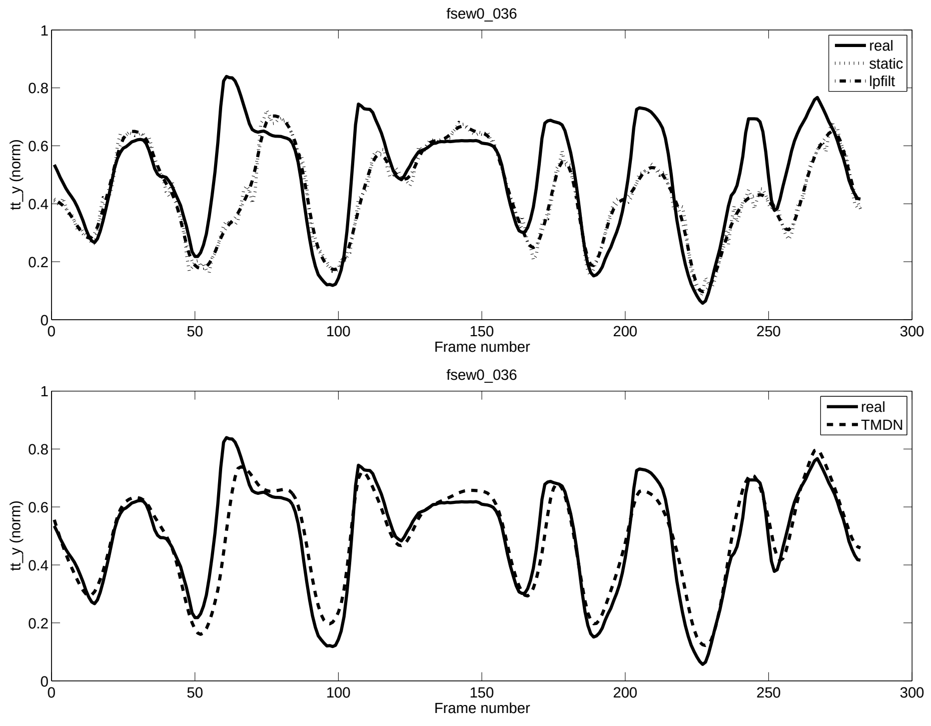


Fig. 3. "Only the most accomplished artists obtain popularity." (fsew0_036). Comparison of TMDN output trajectory with low-pass filtered static mean output trajectory.

References

1. Schroeter, J., Sondhi, M.M.: Speech coding based on physiological models of speech production. In Furui, S., Sondhi, M.M., eds.: Advances in Speech Signal Processing. Marcel Dekker Inc, New York (1992) 231–268
2. Toda, T., Black, A., Tokuda, K.: Mapping from articulatory movements to vocal tract spectrum with Gaussian mixture model for articulatory speech synthesis. In: Proc. 5th ISCA Workshop on Speech Synthesis. (2004)
3. King, S., Frankel, J., Livescu, K., McDermott, E., Richmond, K., Wester, M.: Speech production knowledge in automatic speech recognition. Journal of the Acoustical Society of America **121**(2) (February 2007) 723–742
4. Wrench, A., Richmond, K.: Continuous speech recognition using articulatory data. In: Proc. ICSLP 2000, Beijing, China (2000)
5. Wakita, H.: Estimation of vocal-tract shapes from acoustical analysis of the speech wave: The state of the art. IEEE Trans. Acoust. Speech Signal Process. **ASSP-27** (1979) 281–285
6. Shirai, K., Kobayashi, T.: Estimating articulatory motion from speech wave. Speech Communication **5** (1986) 159–170
7. Atal, B.S., Chang, J.J., Mathews, M.V., Tukey, J.W.: Inversion of articulatory-to-acoustic transformation in the vocal tract by a computer sorting technique. J. Acoust. Soc. Am. **63** (1978) 1535–1555
8. Rahim, M.G., Kleijn, W.B., Schroeter, J., Goodyear, C.C.: Acoustic-to-articulatory parameter mapping using an assembly of neural networks. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing. (1991) 485–488
9. Richmond, K., King, S., Taylor, P.: Modelling the uncertainty in recovering articulation from acoustics. Computer Speech and Language **17** (2003) 153–172
10. Hogden, J., Lofqvist, A., Gracco, V., Zlokarnik, I., Rubin, P., Saltzman, E.: Accurate recovery of articulator positions from acoustics: New conclusions based on human data. J. Acoust. Soc. Am. **100**(3) (September 1996) 1819–1834
11. Toda, T., Black, A., Tokuda, K.: Acoustic-to-articulatory inversion mapping with Gaussian mixture model. In: Proc. 8th International Conference on Spoken Language Processing, Jeju, Korea (2004)
12. Richmond, K.: Estimating Articulatory Parameters from the Acoustic Speech Signal. PhD thesis, The Centre for Speech Technology Research, Edinburgh University (2002)
13. Bishop, C.: Neural Networks for Pattern Recognition. Oxford University Press (1995)
14. Richmond, K.: A trajectory mixture density network for the acoustic-articulatory inversion mapping. In: Proc. Interspeech, Pittsburgh, USA (September 2006)
15. Hiroya, S., Honda, M.: Estimation of articulatory movements from speech acoustics using an HMM-based speech production model. IEEE Transactions on Speech and Audio Processing **12**(2) (mar 2004) 175–185
16. Tokuda, K., Yoshimura, T., Masuko, T., Kobayashi, T., Kitamura, T.: Speech parameter generation algorithms for HMM-based speech synthesis. In: Proc. ICASSP, Istanbul, Turkey (June 2000) 1315–1318
17. Wrench, A.: The MOCHA-TIMIT articulatory database. http://www.cstr.ed.ac.uk/artic/mocha.html (1999)