IDIAP RESEARCH REPORT

Master Thesis

Integration of the Harmonic plus Noise Model (HNM) into the Hidden Markov Model-Based Speech Synthesis System (HTS)

June 2006

Coralie Hemptinne

IDIAP Research Institute **www.idiap.ch**
Rue du Simplon 4 P.O. Box 592 1920 Martigny − Switzerland
Tel: +41 27 721 77 11 Fax: +41 27 721 77 12 Email: info@idiap.ch

Master Thesis

Integration of the Harmonic plus Noise Model (HNM) into the Hidden Markov Model-Based Speech Synthesis System (HTS)

Coralie Hemptinne

June 2006

Abstract

In the present project, we developed and tested a new model-based text-to-speech (TTS) system, integrating the Harmonic plus Noise Model (HNM) into the Hidden Markov Model-based Speech Synthesis System (HTS). This integration leads to a TTS system that requires smaller development time and cost, in comparison with the usual state-of-the-art TTS systems typically based on automatic selection and synthesis of subword units (e.g., diphones), while also producing a better quality speech output (compared to HTS alone). This quality enhancement is achieved by replacing the source filter modeling approach typically used in HTS with the HNM model, which is known for being able to synthesize natural sounding speech under various prosodic modifications.

The basic idea behind HNM is to model speech as being composed of harmonic and noise parts. Voiced frames comprise a harmonic part and a noise part, separated by the time-varying maximum voiced frequency, whereas unvoiced frames are only composed of a noise part. The HNM algorithm consists in two steps: (1) the HNM analysis, i.e. the computation of the HNM parameters of every acoustic unit of the training database, and (2) the HNM synthesis, i.e. the speech waveform synthesis from the HNM parameters.

The HTS system comprises a training and a synthesis part. The training part consists in computing the parameters modeling the database and in training context-dependent HMMs. During the synthesis part, given the target prosodic and phonetic labels corresponding to the text to synthesize, the adequate context-dependent HMMs are concatenated to build a composite HMM. The most likely parameters are then estimated and used to synthesize speech waveform using a filter-based approach.

In the TTS system developed in this work, the utterance database is modeled by HNM parameters, which constitutes the first modification of the HTS system. These parameters and their dynamic features are used to train context-dependent HMMs, like in HTS. A composite HMM is then constructed and the HMM parameters are generated by maximum likelihood estimation. Finally, the speech waveform is obtained by HNM synthesis, which is the second modification of the HTS system. This constitutes the general description of the TTS system used in this work, which has been implemented in three different ways: (1) The HNM parameters are extracted at a fixed rate from the training database; they include the linear predictive cepstral coefficients (LPCC) and the fundamental frequency; (2) The same parameters are extracted pitch-synchronously; (3) The extraction is again pitch-synchronous, but maximum voiced frequency is also modeled. In conclusion, a slightly better speech waveform quality is obtained in the third case.

Contents

List of Figures

Acronyms and abbreviations

AR	Autoregressive
CART	Classification And Regression Tree
CDHMM	Continuous Density Hidden Markov Model
DTFT	Discrete Time Fourier Transform
FFT	Fast Fourier Transform
HMM	Hidden Markov Model
HNM	Harmonic plus Noise Model
HTS	Hidden Markov Model-based Speech Synthesis System
LPC	Linear Prediction Coefficients
LPCC	Linear Predictive Cepstral Coefficients
MBROLA	Multi-Band-Overlap-Add
MBE	Multi-Band-Excited
MFCC	Mel-Frequency Cepstrum Coefficients
MLSA	Mel Log Spectrum Approximation
MSD-HMM	Multi-Space Probability Distribution Hidden Markov Model
MOS	Mean Opinion Score
RELP	Residual Excited Linear Prediction
PSOLA	Pitch-Synchronous Overlap-Add
TD-PSOLA	Time Domain Pitch-Synchronous Overlap-Add
TTS	Text-to-Speech
V/UV	Voiced/Unvoiced

Acknowledgments

First I must thank Dr Thierry Dutoit, teacher of signal processing in the Faculté Polytechnique de Mons (FPMs), without who I wouldn't ever get interested in speech technologies.

This master thesis has been pursued through an exchange program between the Ecole Polytechnique Fédérale de Lausanne (EPFL) and the Faculté Polytechnique de Mons (FPMs), at the IDIAP Research Institute, a research institute affiliated with the EPFL and located in Martigny, Switzerland. I would then like to acknowledge Dr Hervé Bourlard, director of IDIAP, for welcoming me at IDIAP and for sharing his rich experience.

I have great thanks for my supervisors at IDIAP, Drs John Dines and Jithendra Vepa, whose enthusiasm, advice and availability gave me the necessary motivation for this master thesis.

Special thanks go to David Vandromme. My thesis being the continuation of his, I've had to use his results and his ideas were particularly useful.

My parents, my brother Jean-Louis, Seb and my friends are deserving of my thanks for their everyday's support.

I would also like to thank all my colleagues at IDIAP for their constant conviviality, as well as for all these great discussions during which they shared their ideas and personal experience.

Finally, I must acknowledge the developers of open source softwares. Their work was extremely useful in order to develop a complete text-to-speech synthesizer. Appendix A gives details of all the tools that have been used in this thesis.

Chapter 1

Introduction

1.1 Overview

In our society, where speed and efficiency are key qualities, a human computer interaction via speech is of great pertinence. Such an interaction involves speech recognition and speech synthesis. The first one consists in extracting the message information in a speech signal so as to control the actions of a machine in response to spoken commands, whereas the second one is the process of creating a synthetic replica of a speech signal so as to transmit a message from a machine to a person, with the purpose of conveying the information in the message [Rabiner94].

Speech recognition and speech synthesis have been used in numerous applications for several years, and new applications are appearing every year, with the improvement of the recognition rate and the quality of synthesized speech. Applications exist:

- **To help the disabled:** automatic reading of electronic and paper documents with an artificial voice;

- **In multimedia:** interactive games, educational softwares (foreign languages learning, reading learning, etc.), interaction with PDAs, pocket PCs, where the very small keyboard is time-consuming to use, information retrieval from an archive of BBC news broadcasts;

- **In telecommunication:** checking of one's emails or faxes with the telephone, listening to the synthesized speech corresponding to SMSs, voice access to databases (price list, cultural events, weather report, etc.).

1.1.1 Aims

In speech synthesis, which is the field of this work, the aim is to obtain a synthesized speech not only easily understandable, but also indistinguishable from that produced by a human, in other words, to create a system that equals the human performance. Thus, the two qualities required by a speech synthesis system are intelligibility and naturalness.

Most current text-to-speech systems are able to achieve acceptable levels of intelligibility. Naturalness is obtained if the waveform synthesis part of the TTS systems is able to render speech under various prosodic, phonetic, emotional and even semantic contexts. This is the basic idea behind continuous speech database-driven techniques, mainly unit selection. While providing more natural waveform synthesis, database-driven synthesis techniques have several drawbacks, mainly the requirement of a very large database, and the unability to produce unseen contexts. These disadvantages may be overcome with an alternative to the database-driven technique, which is the model-based speech synthesis technique, mainly the Hidden Markov Model-based speech synthesis system. Currently speech synthesized with HMM synthesis is not of high quality because the speech waveform is generated with a filter-based approach. A better quality could be obtained by replacing this filter-based approach by the Harmonic Plus Noise Model, which is known for being able to synthesize natural sounding speech under various prosodic modifications[1].

In conclusion, this thesis aims at improving the quality of the speech synthesized by a Hidden Markov Model-based approach.

1.1.2 Scope of research

Within the six months of this thesis the goals had to be prioritized. There remains several avenues of research to investigate, which are detailed in section 6.2. This work is based on the following assumptions:

- Statistical models have been trained using a database spoken by a Scottish English male speaker. It is expected that a HMM-based synthesis using training data including several speakers, from both genders, would lead to equal results in terms of speech waveform quality.
- HMM training has been performed by using two kinds of feature vector. They comprise the linear predictive cepstral coefficients and their the delta and delta-delta coefficients, the fundamental frequency and its dynamic features, and, only in one of the two feature vectors, the maximum voiced frequency and its dynamic features. Synthesis in this work

[1] See below for more details.

is based on the assumption of zero phase. Harmonic amplitudes are derived from the linear prediction coefficients. The quality of the synthesized speech waveform may be improved by using other kind of features. There may be better solution to model the harmonic phases. Harmonic amplitudes may be parameterized in a different way.

- The research in this thesis is limited to the integration of the Harmonic plus Noise Model into the Hidden Markov Model-based Speech Synthesis System, given:
  - The parameters provided by the HNM analysis module developed by D. Vandromme [Van Dromme05];
  - The phonetic and prosodic targets generated by The Festival Speech Synthesis System, which is an open source speech synthesis software [Black99];
  - The speech waveform synthesis carried out by the HNM synthesis module written by D. Vandromme [Van Dromme05].

1.2 Thesis organization

This report is concerned with text-to-speech synthesis, and more precisely with the integration of the Harmonic plus Noise Model (HNM) into the Hidden Markov Model-based Speech Synthesis System (HTS). It is then organized as follows:

Chapter 2 recalls briefly the general structure of a text-to-speech system. It explains first the natural language processing and describes then the main approaches to generate the speech waveform, namely rule-based synthesis, diphone based synthesis, unit selection synthesis and model-based synthesis. Unit selection synthesis is the state-of-the-art synthesis technique. It is compared from several points of view with the model-based synthesis, which is the technique used in this work. This chapter ends with a more detailed description of linear prediction techniques, time domain-methods and hybrid harmonic/stochastic approaches, as well as a comparison between these techniques.

Chapter 3 describes the Hidden Markov Model-based Speech Synthesis System (HTS), which is the only public implementation of model-based synthesis. First section describes the training part of the system. It details the features that are used in HTS to model speech, as well as their output distribution. It explains then the interest of using tree-based clustering techniques. Second section describes the synthesis part of HTS. Finally third section details the maximum likelihood estimation of parameters from HMMs.

Chapter 4 describes the Harmonic plus Noise Model. It comprises two sections, which detail the HNM analysis and synthesis, i.e. the computation of the HNM parameters of every acoustic unit of the utterance database and the speech waveform synthesis using these parameters.

Finally, chapter 5 details the integration of the HNM model into the HTS system. First section is devoted to the training part of the system. Three syntheses have been performed, making different different design choices, which are described in this section. Then some explanations are given about the function of all the software tools used for training context-dependent HMMs. Second section concerns the synthesis part of the system. It explains the operations that must be performed on the parameters generated by the composite HMM to meet the requirements of the HNM synthesis module. This chapter ends with a comparison of the speech waveform qualityof the three syntheses that have been performed.

Appendix A provides information about the open source softwares used in this work.

Appendix B gives mathematical details on Hidden Markov Model modeling.

Chapter 2

TTS summary

Text-to-speech synthesis refers to the automated generation of speech from any text, by mapping the sequence of characters of this text into another sequence of numbers representing the samples of the synthesized speech [Dines03, Dutoit02, Stylianou96]. This is achieved in two steps, as represented on Figure 2.1: the natural language processing, and the digital signal processing, which is in other words the speech waveform generation.

2.1 Natural language processing

The natural language processing takes raw text as input and outputs a symbolic linguistic representation. This leads to three major tasks:

- The **pre-processing** of the input text includes four phases: the detection of the end of sentences, the conversion of numbers, the expansion of abbreviations and acronyms into full text, the assignment of part-of-speech to every word, and the isolation of phrases to derive the intonation. These tasks are not straightforward; a TTS system must deal

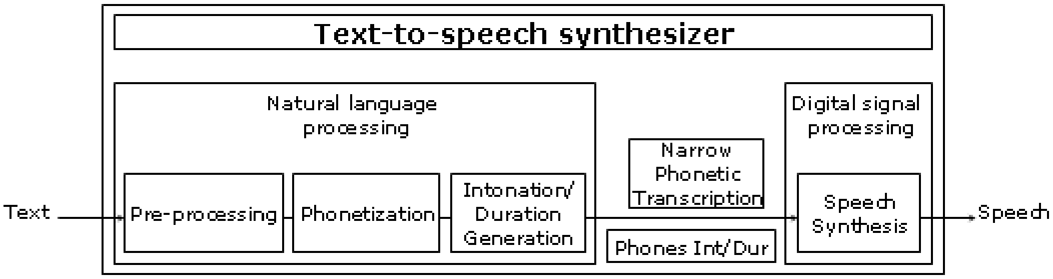

Text-to-speech synthesizer
Natural language processing
Digital signal processing
Narrow Phonetic Transcription
Text
Pre-processing
Phonetization
Intonation/ Duration Generation
Phones Int/Dur
Speech Synthesis
Speech

Figure 2.1: A text-to-speech synthesizer, source [Dutoit02]

with several linguistic ambiguities. For example, the dots don't necessarily indicate the end of a sentence; the pronunciation of a number depends on its context: if 1325 is part of an address (1325 Main St.) it should be read as "thirteen twenty-five", and, if it is the last four digits of a social security number, its pronunciation becomes "one three two five"; abbreviations can be ambiguous, such as "St." in "St. John St.", etc. The resulting data are sent to the next unit, that is, the phonetization.

- The **phonetization** establishes the phonetic transcription of the input sentences. The system must deal with words that are pronounced differently based on context ("*The* clouds" and "*The* army"), with heterophonic homographs ("Everyday, he records the weather records."), with phonetic liaisons (In French, liaisons can be compulsory: "très utile", forbidden: "plat exquis", or optional: "deux à deux".), with assimilation, with proper nouns, etc.

  Two approaches exist to perform the phonetization. The simplest one is based on a large dictionary containing all the words of a language and their pronunciation; it is then possible to replace each word by its phonetic transcription in the dictionary. The other approach relies on rules that derive the pronunciation of words from their spelling. In comparison to the rule-based approach, the dictionary-based technique requires more memory space, is more accurate (doesn't need to deal with irregular spellings or pronunciations) but is far less efficient if it is given a new word, i.e. a word which is not in its dictionary.

- The **generation of prosody** determines the sequence of prosodic characteristics, stored in the form of quantitative values. Prosody refers to intonation, rhythm, and vocal stress in speech; the prosodic features of a unit of speech (like a syllable, a word, a phrase or a clause) are the fundamental frequency (pitch), the duration and information the intensity. The prosodic characteristics of an utterance bring it additional meaning, not only related to emotion and naturalness, but also to intelligibility, as prosody is influenced by syntax and semantics, eg. in French, "On se quitte demain ?" compared to "On se quitte demain !".

2.2 Speech waveform generation

The second step of text-to-speech synthesis is the speech waveform generation. It uses the above symbolic linguistic representation, that is the phonetic transcription and the prosodic information, to produce the synthesized speech waveform. Several technologies have been developed to synthesize speech waveforms. They fall into two categories: rule-based synthesis and database-driven synthesis. The rule-based synthesis was invented in 1964. Despite a

Rule-based synthesis (1964)	Database-driven synthesis		
	Concatenative synthesis		Model-based synthesis (1999)
	Diphone based synthesis (1977)	Unit selection synthesis (1997)	

Figure 2.2: An overview of the speech waveform synthesis techniques

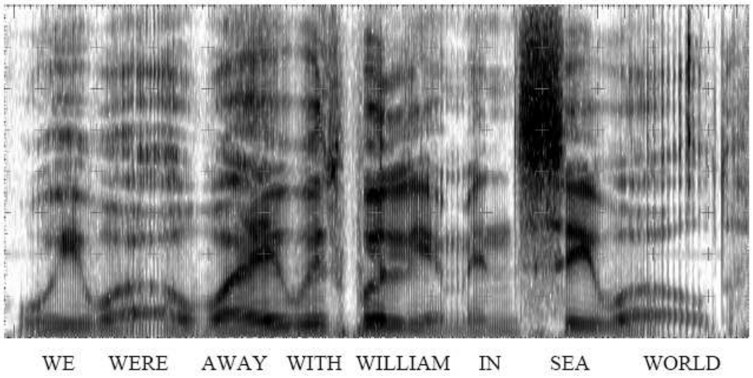


Figure 2.3: Coarticulation in the case of the phone [w], source [Odell95]

fairly average synthesized speech quality, rule-based synthesizers are still sold today. A better quality is obtained with the second approach, which can be also categorized into concatenative synthesis and model-based synthesis, which appeared in 1999. A last distinction can be made in concatenative synthesis between diphone based (1977) and unit selection based synthesis (1997). Table 2.2 gives an overview of all the existing approaches to the speech waveform generation. These approaches are detailed in the following sections.

2.2.1 Rule-based synthesis

The rule-based synthesis, also called "formant[1] synthesis", doesn't use any human speech samples but relies on rules written by linguists to generate the parameters that will permit the synthesis of speech, and to deal with the transition from one phoneme to another, that is, the coarticulation[2] To write these rules, linguists have studied spectrograms and derived the

[1] The formants are the peaks in the frequency spectrum.

[2] The coarticulation is the modification in the pronunciation of a sound because of its phonetic context, due to physiological constraints [Dutoit02]. Indeed speech is produced by physical articulators, which cannot react suddenly; actually they move continually and slowly from the position required to articulate the previous phone towards the position required for the next phone, via the position needed for the current phone [Odell95]. This means, for example, that the frequency spectrum of two [w] will be different if they're preceded and/or followed by different sounds. This is illustrated by the Figure 2.3, that represents the spectrogram of the sentence "We were away with William in Sea World."

rules of the evolution of formants. However we do not yet know the optimal rules to do this [Klatt87]. Moreover, the speech waveform is naturally produced in such a complex process that, currently, rules can only model the main features of the speech waveform.

Therefore, the synthesized speech has an artificial, robotic sound, and the goal of naturalness is not reached. However, the rule-based synthesized speech is very intelligible, even at high speeds, which is quite useful for the visually impaired for quickly navigating computers using a screen reader. Moreover, when memory and processing costs are limited, such as in embedded systems, these synthesizers are more interesting than database-driven systems because they don't have a database of speech samples.

The rule-based synthesis approach has been implemented in MITalk [Allen87, Allen79], in Klattalk [Klatt82], in DECTalk [Klatt90].

2.2.2 Database-driven synthesis

Database-driven synthesis techniques are based on a labeled database of utterances[3], from which acoustic units are extracted. Then, the units are either concatenated in the case of concatenative synthesis or they are used to train statistical models in the case of model-based synthesis. This is described below.

In database-driven synthesis, coarticulation is no longer described by rules because it is embedded in the acoustic units, which are diphones[4], triphones, etc. The choice of the size of the unit is an open issue dependent on the coverage of the synthesis database. Let's compare diphones and triphones. The number of diphones being smaller than that of triphones, the required speech corpus to enable a diphone based speech synthesis is smaller. But the synthesis of an utterance requires more concatenations in the case of diphones, which obviously damages the quality of synthesized speech. Therefore the size of the unit is a trade-off between the database size and the number of joins between concatenated units.

Concatenative synthesis

Concatenative synthesis consists of two main steps: 1) The inventory is built, by extracting acoustic units from the utterance database; 2) According to the text to synthesize, the required units are selected and concatenated.

It is unlikely that the target and recorded prosodies are the same, as the context of the

[3] The utterances can be either nonsense words, like in Festival, or actual words/sentences.

[4] A diphone is a pair of adjacent phones; more precisely it groups the second half of the first phone and the first half of the second phone.

target and recorded acoustic units are different. The desired prosody is then obtained by signal processing techniques. Finally the units' boundaries must be smoothed. These last two steps degrade the quality of the units. Examples of systems implementing concatenative synthesis are: Festival [Black99], CHATR [Campbell96], Next-Gen [Beutnagel99], RealSpeak [Coorman00].

Concatenative synthesis is categorized into diphone based synthesis and unit selection synthesis according to the size of the units' database, which is much larger in the case of unit selection synthesis. These two approaches are described in the following sections.

Diphone based synthesis In diphone based synthesis, it is assumed that the coarticulatory effect is limited, that a phoneme is only influenced by the last one. Consequently the units' database includes only one prosodically neutral instance of every diphone in the language. The number of diphones in a language is reasonable (English comprises around 1600 diphones). Therefore the database is rather small (about 3 minutes of speech, or $5Mb$). The selection of the diphones to concatenate is quite simple: after the phonetization process, we have a sequence of phonemes, which corresponds to a single sequence of diphones. Finally prosodic modifications smoothing of the units boundaries are performed using one of the following techniques : LPC [Makhoul75], PSOLA [Hamon89, Charpentier86], RELP [Hunt89], MBROLA [Dutoit93] and HNM [Stylianou96, Stylianou01, Stylianou98]. Details on these techniques are given in sections 2.3, 2.4 and 2.5.

Unit selection synthesis In the case of automatic unit selection, the coarticulatory influence isn't limited to the last phoneme. The database is much larger (1-10 hours, or $150Mb$-$1500Mb$) and comprises several occurrences of each acoustic unit, captured under various contexts (like its neighboring phonemes of course, but also its pitch, its duration, its position in the syllable, etc.). As a result, the sequence of phonemes to synthesize leads to a lattice of acoustic units, in which the best path must be found. For each phoneme, a unit is selected so that it not only best corresponds to the expected contexts (prosody, phonetics, etc.) but also minimizes the spectral and prosodic discontinuities. In other words, the best path, selected with the Viterbi algorithm, is the path that minimizes the sum of a "target" and "concatenation" cost. Consequently, automatic unit selection requires much less modification of the speech units, which leads to an overall quality of the synthesized speech much more natural than with diphone based synthesis.

Apart from this naturalness, unit selection techniques have several disadvantages:

- They rely on a very large database, which implies, on the one hand, considerable development time and cost to collect and label the data, and on the other hand, large memory resource requirements to store the data.

- Incorrect labeling and occurrence of unseen target contexts lead to fragments of synthesized speech of extremely poor quality. This phenomenon of unseen contexts may well never be fully overcome with concatenative synthesis as [Mobius01] suggest that rare events will always occur in language.

Great interest is therefore taken in the research related to model-based synthesis, which overcome these drawbacks.

Model-based synthesis

The model-based synthesis is an hybrid between rule-based and database-driven techniques. The speech database is described by a parametric model. The parameters (eg. the frequency spectrum, the pitch, the duration, etc.) are summarized by a set of statistical models, that is the Hidden Markov Models, that capture the significant patterns of the speech units. Finally, speech waveforms are generated by the HMMs based on a maximum likelihood criterion.

The Hidden Markov Model-Based Speech Synthesis System (HTS), which is the only public implementation of model-based synthesis, is described in chapter 3. Model-based synthesis has also been implemented by Microsoft in Whistler (Whisper Highly Intelligent Stochastic TaLkER) [Huang96], and by IBM [Donovan98, Donovan01].

In comparison with unit selection synthesis, model-based synthesis has the following advantages and drawbacks [Dines03, Tokuda02-2]:

- The automatic learning of model parameters, with a relatively small quantity of training data, reduces the storage requirements, and allows to develop a new voice in much less time.

- As far as the waveform quality is concerned, it must be recognized that model-based synthesis has a quality of "vocoded speech". However the use of HMMs and the introduction of dynamic features avoid the discontinuities observed in concatenative synthesis. Moreover, while waveform quality in unit selection synthesis can severely decrease in case of unseen target contexts and consequent bad unit selection, model-based synthesis provides means of generalization for unseen contexts, thanks to the decision-tree-based context clustering technique. This results in intelligible speech under numerous contexts.

- Model-based synthesis allows easy modification of the voice characteristics by using a speaker adaptation/interpolation technique, while unit selection synthesis generates speech whose style cannot be different from the style of the database, as the idea behind unit selection synthesis is to reduce as much as possible signal processing on the selected units.

In conclusion, model-based and unit selection syntheses have both advantages and disadvantages, as summarized in table 2.4.

	Unit selection synthesis	Model-based synthesis
Development time/cost to create a new voice	Significant	Small
Memory resource requirements	High	Small
Waveform quality	High	Vocoded speech (buzzy)
Presence of discontinuities	Possible	Less probable
Management of unseen target contexts	Bad	Good
Synthesis style	Fixed	Can be modified

Figure 2.4: Comparison of unit selection and model-based syntheses

The applications of statistical modeling go beyond speech synthesis; they comprise speech coding, speaker transformation and speaker imposture in speaker verification/identification.

2.3 Linear prediction synthesis

The concept behind linear predictive coding is that a sample of speech is approximately equal to a linear combination of past speech samples [Boite00, Dines03, Dutoit02, Quatieri01]. This implies that an autoregressive filter,

$$\frac{\sigma}{A_p(z)} \tag{2.1}$$

$$A_p(z) = 1 + \sum_{k=1}^{p} a_k z^{-k} \tag{2.2}$$

must be calculated so as to minimize the variance of the residual signal, that is, the difference between the speech samples and the samples predicted by the AR filter. This minimization is performed by solving the Yule-Walker equations:

$$\sum_{j=1}^{p} a_j \phi_x(i-j) = -\phi_x(i) \qquad (i = 1, \dots, p) \tag{2.3}$$

where:

- p, the prediction order, corresponds to the number of resonances created by the filter from 0 to the sampling frequency;

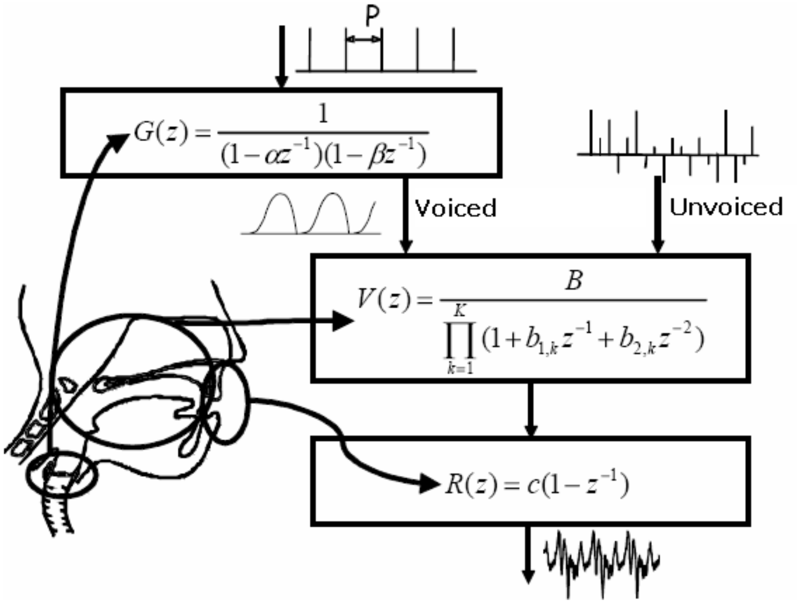

P
$G(z) = \frac{1}{(1-\alpha z^{-1})(1-\beta z^{-1})}$
Voiced
Unvoiced
$V(z) = \frac{B}{\prod_{k=1}^{K}(1+b_{1,k}z^{-1}+b_{2,k}z^{-2})}$
$R(z) = c(1-z^{-1})$

Figure 2.5: Model of the human speech production system, source [Dutoit02]

- a_j are the LPC coefficients, the first coefficient being given: $a_0 = 1$;

- $\phi_x(i)$ is the autocorrelation function of the signal x;

If this system of p linear equations with p unknowns is expressed in matrix form, a Toeplitz matrix is obtained, which enables the use of the Levinson or Schur algorithm, and therefore a reduction of the complexity from $O(p^3)$ to $O(p^2)$.

If the residual signal is applied to the AR filter, the original speech signal is recovered. This corresponds to a residual excited linear prediction (RELP) synthesis.

The LPC synthesis is actually based on a simplified modeling of the human speech production system, where the excitation and the vocal tract are modeled separately. LPC presupposes that the excitation can be categorized in two types: voiced parts are produced by a periodic train of Dirac pulses that models the periodic glottal opening/closure; unvoiced parts are generated by a white noise that models the turbulent airflow through the constricted glottis. The substitution of the residual signal for one of these excitations obviously degrades the quality of the synthesized speech. The excitation, either the train of pulses or the white noise, is then applied to a series of filters modeling the vocal tract, that is, successively, the glottis ($G(z)$), the oral and nasal cavity ($V(z)$), and the lips ($R(z)$), as represented on Figure

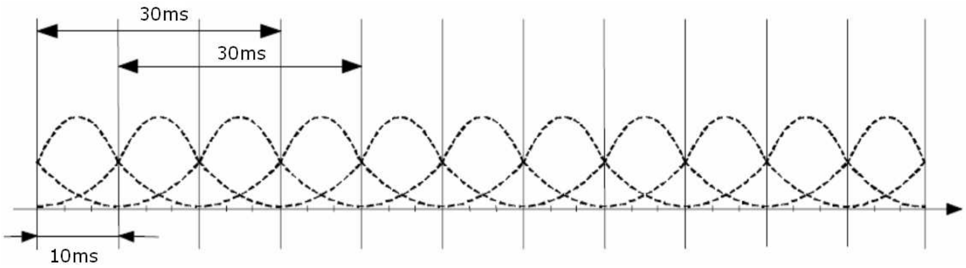


Figure 2.6: Overlapping and windowing of the short term speech signals, source [Dutoit02]

2.5. This series of filters is approximately equivalent to the AR filter defined above:

$$G(z)V(z)R(z) = \frac{1}{(1-\alpha z^{-1})(1-\beta z^{-1})}\frac{B}{\prod_{k=1}^{K}(1+b_{1,k}z^{-1}+b_{2,k}z^{-2})}c(1-z^{-1}) \quad (2.4)$$

$$\simeq \frac{\sigma}{A_p(z)} \quad (2.5)$$

Therefore, in the case of voiced speech, the number p of poles of the AR filter includes 2 poles for shaping the glottal waveform, and two poles per formant, that is, approximately, two poles per kilohertz of bandpass. For instance, a sampling frequency of 16kHz leads to a prediction order of 18.

The speech signal is not stationary otherwise it wouldn't carry any information, but the signal processing techniques used for LPC coding require signal stationarity. Therefore the speech signal is divided into overlapping frames and the short term signals are assumed to be stationary. The choice of frame size and frame shift influences the computational cost (a smaller frame shift leads to a larger number of frames to process) and the accuracy of extracted features (depending on whether speech segments are stationary or not, it is desirable to use larger or smaller frame sizes to carry out optimal feature extraction). In this work the frame size is of $30ms$, and the frame shift, of $10ms$, as represented on Figure 2.6. It is worth mentioning that there also exists variable frame rate and frame shift in the case of pitch synchronous frame analysis, which will be used in chapter 5.

The obtained segments of speech are then pre-emphasized and windowed, as illustrated in Figure 2.7. Pre-emphasis consists in compensating for the spectral tilt caused by the lips during speech production, by using a filter which boosts the higher frequencies of the speech signal:

$$H'(z) = 1 - az^{-1} \qquad (0.95 \leq a \leq 0.98, \text{ typically}) \quad (2.6)$$

The purpose of windowing is to dampen the effect of the Gibbs Phenomenon [Ifeachor96] so as to avoid ringing during spectral analysis. Windowing is performed by multiplying the window coefficients by the speech frame in the time domain. In this work, this work makes use of a

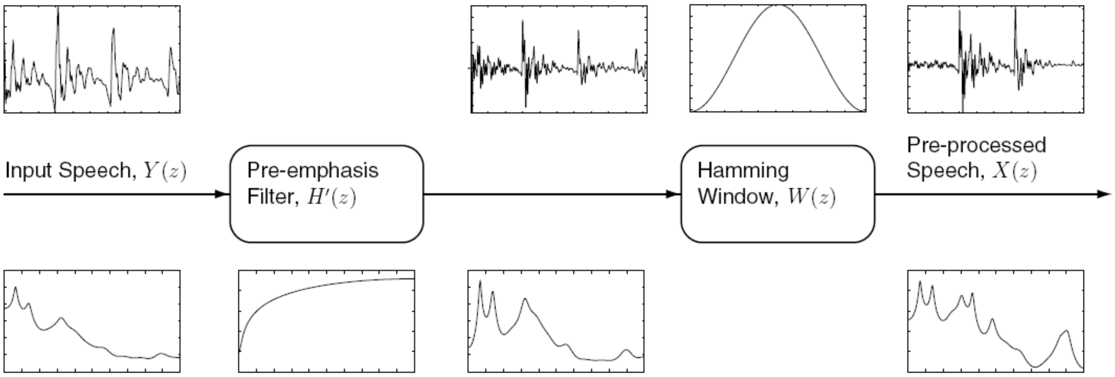


Figure 2.7: Pre-emphasis and windowing of a speech frame, source [Dines03]

Hamming window, defined by:

$$w(n) = 0.54 - 0.46 \cos\left(\frac{2\pi n}{L}\right) \qquad (0 \leq n \ L) \tag{2.7}$$

where the length of the analysis window is L samples.

Figure 2.8 summarizes the LPC synthesis: the excitation, either a train of pulses in the case of voiced frames or a white noise in the case of unvoiced frames, is input to the AR filter modeling the vocal tract. The parametric modeling then makes it easy to perform prosodic modifications. The pitch period, being the distance between consecutive impulses of the periodic train, can be set to any target value, T. To keep the same signal power, the gain, σ_0, must then be replaced by: $\sigma = \sigma_0\sqrt{\frac{T_0}{T}}$. The length of a frame is modified by adding or removing frames. Thanks to parametric modeling again, smoothing of spectral discontinuities at segment joins can be carried out by linear interpolation. Finally, the frames are overlapped and added to obtain the synthesized speech signal.

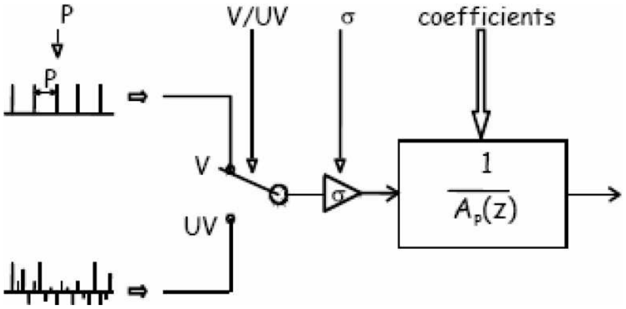


Figure 2.8: A LPC synthesizer, source [Dutoit02]

The main drawback of autoregressive models is their poor segmental quality[5]. Experience shows that this quality cannot be improved by increasing the model order, the sampling fre-

[5] Segmental quality refers to the quality of waveform synthesis.

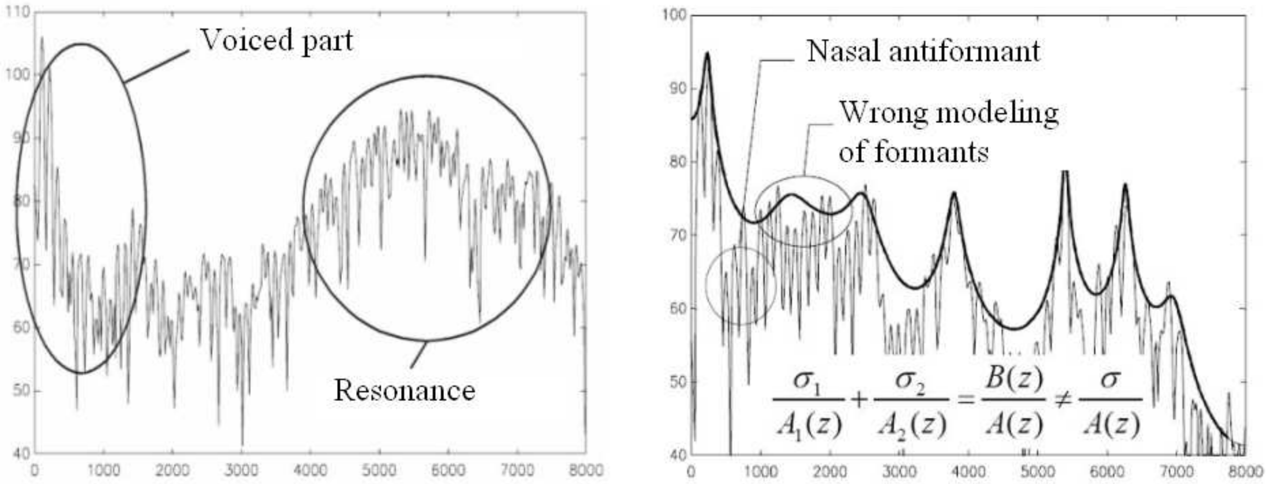


Figure 2.9: Drawbacks of the autoregressive model, source [Dutoit02]

quency or the refresh rate of the LPC parameters. First, separate modeling of source and filter can degrade the quality of synthesized speech if source and/or filter components are modified independently. Then the excitation, which is either voiced or unvoiced, cannot properly model mixed sounds (like [v]), i.e. sounds that comprise voiced and unvoiced components as the vocal cords vibrate, but don't completely close. A mixed excitation, as in hybrid synthesis, could therefore improve the quality. Finally, the nasal sounds (like [m] or [n]), whose spectrum includes antiformants around 1000Hz, are badly modeled, because the filter doesn't possess any zeros. These last two drawbacks are illustrated on Figure 2.9.

2.4 Time-domain algorithms

The common idea behind time-domain algorithms is that the speech waveform is modified in the temporal domain without using any parametric model. In other words, prosodic modifications are performed directly on the speech signal [Boite00, Dines03, Dutoit93, Dutoit02]. Several time-domain synthesis techniques exist, such as Time-Domain Pitch-Synchronous-Overlap-Add (TD-PSOLA), Multi-Band-Overlap-Add (MBROLA), Linear-Pitch-Synchronous-Overlap-Add (LP-PSOLA).

TD-PSOLA synthesis was developed by France Telecom in the beginning of the nineties. Its fundamental concept is that, if $x(n)$ is perfectly periodic from $-\infty$ to $+\infty$, it is possible to get a pitch-shifted version $s(n)$ of $x(n)$ by summing windowed frames $s_i(n)$, extracted pitch-synchronously from the original signal $x(n)$. The adequate localization of the pitch marks at the point of glottal closure is essential to implement a PSOLA based synthesis. The pitch modification is performed by changing the time-shift between successive windowed frames

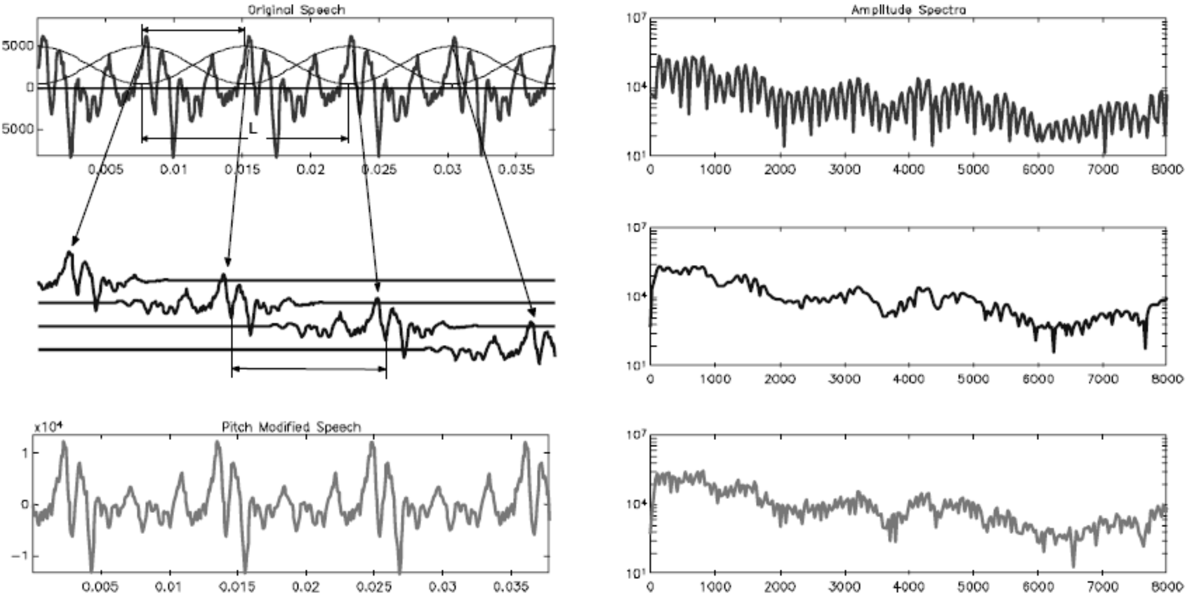


Figure 2.10: Pitch modification with TD-PSOLA: signals are represented in the left column, and the corresponding spectra in the right one; the spectral envelopes of the modified signal (bottom right graph) and of the original one (top right graph) are the same, but don't share the same fundamental frequency, source [Dutoit02]

from the original period T_0 to the target one T:

$$\begin{aligned} s_i(n) &= x(n)w(n - iT_0) && (2.8) \\ s(n) &= \sum_{i=-\infty}^{+\infty} s_i(n - i(T - T_0)) && (2.9) \end{aligned}$$

This is represented on the three left graphs of Figure 2.10.

It can be proved that, while having different fundamental frequencies (T_0 for the original signal and T for the modified one), the modified signal and the original one share the same spectral envelope, if the window $w(n)$ is chosen appropriately. From a practical point of view, we use a Hanning or a triangular window, whose length is twice the pitch period of the signal.

In addition to pitch shifting, duration warping can be performed by removing or repeating windowed frames and amplitude modification can be carried out by scaling the windowing function.

Segmental quality of TD-PSOLA synthesis is very good when performed inside a quasi-stationary segment, if the pitch and/or duration modification is not excessive. Indeed this quasi-stationary segment approximates the perfectly periodic signal used above. However segmental quality deteriorates when segments extracted from different words are concatenated:

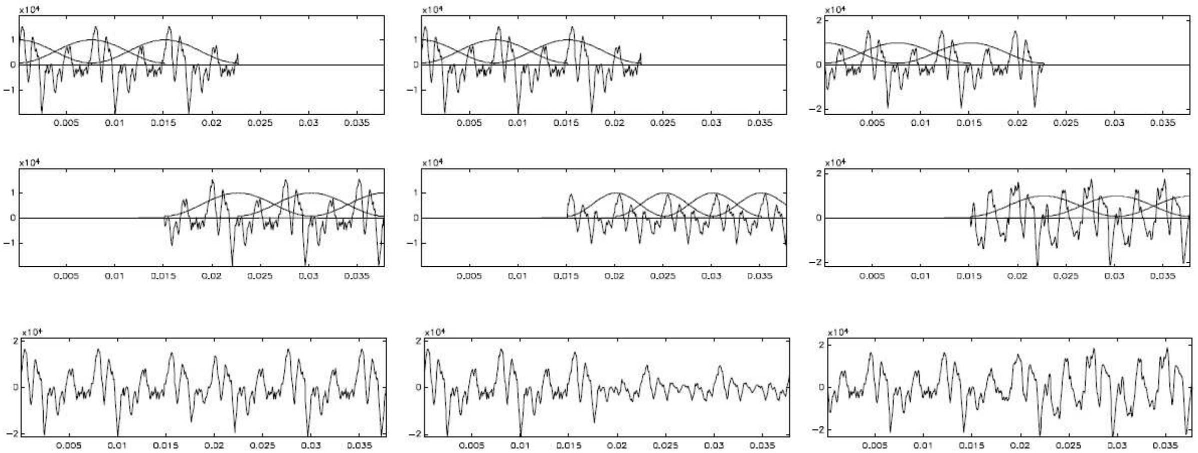


Figure 2.11: Phase (left), pitch (centre) and spectral envelope (right) discontinuities in TD-PSOLA synthesis in the case of the vowel [a] sampled at 16kHz; top graphs represent the last frames of the first segment to be concatenated, center ones, the first frames of the second segment, and bottom ones, the result after OLA, source [Dutoit93]

phase, pitch and spectral envelope discontinuities arise, as represented on Figure 2.11. Phase discontinuities occur when the concatenated frames are not centered on the same relative positions within the period. Pitch discontinuities are observed when the concatenated frames have not been pronounced with the same pitch. Spectral envelope discontinuities are due to coarticulation, whose influence on the right and left segments is different, as they've been extracted from different phonetic contexts.

Alternative time-domain synthesis techniques have been developed to counter these drawbacks, such as LP-PSOLA and MBROLA. LP-PSOLA consists in the association of TD-PSOLA and LPC modeling, which enables spectral smoothing in the parametric domain and offers good database compression thanks to the use of a parametric modeling. The basic idea behind MBROLA is to resynthesize the voiced segments of the database so as to meet the three following features: a constant pitch for all the segments, a constant position of the frame within the period for all the segments and the possibility to carry out spectral interpolation between concatenated frames easily.

More generally, compared to linear prediction synthesis techniques, time-domain algorithms don't enable spectral smoothing and permit only limited compression of the stored unit inventory. However, as no parametric modeling is used, most modeling errors are avoided. Moreover, reconstruction of the speech from raw speech segments leads to a small computational cost.

2.5 Hybrid harmonic/stochastic synthesis

The hybrid harmonic/stochastic models transfer the V/UV decisions from frames to frequency bands or even replace these bipolar decisions by frequency dependent V/UV ratios [Dutoit93], which enables a better modeling of mixed sounds. This hybrid approach was implemented notably in the Multi-Band-Excited (MBE) model [Griffin87], in the hybrid approach of Abrantes *et al.*[Abrantes91], in the Harmonic plus Noise Model [Stylianou96, Stylianou01, Stylianou98] and in the algorithm of d'Alessandro *et al.* [d'Alessandro98]. This section explains the key concepts of the Harmonic plus Noise Model. The HNM analysis and synthesis processes are detailed in section 4.

The Harmonic plus Noise Model represents the speech signal as being composed of harmonic parts and stochastic parts, respectively modeled by sums of time-varying harmonics and modulated noises. Voiced frames comprise a harmonic part and a noise part and unvoiced frames only consists of a noise part.

Figure 2.12a shows the waveform of a voiced speech segment, sampled at 16000Hz. On graph (b), which represents its amplitude spectrum, peaks at multiples of a fundamental frequency can be observed from 0 to 5000Hz, and the frequency range from 5000Hz to half the sampling frequency is dominated by noise. The lower band of the spectrum is the harmonic part $h_V(t)$ and the higher one, the noise part $n_V(t)$; the transition frequency between these two parts is called the maximum voiced frequency, $F_M(t)$, which is a time-varying parameter. Graphs (c) and (d) respectively represent the waveforms corresponding to the frequency intervals 0-5000Hz and 5000-8000Hz. Comparing the low-pass and high-pass signals, it can be observed that noise bursts of the high-pass signal are synchronized with the pitch period. This important property will influence the modeling of the noise part.

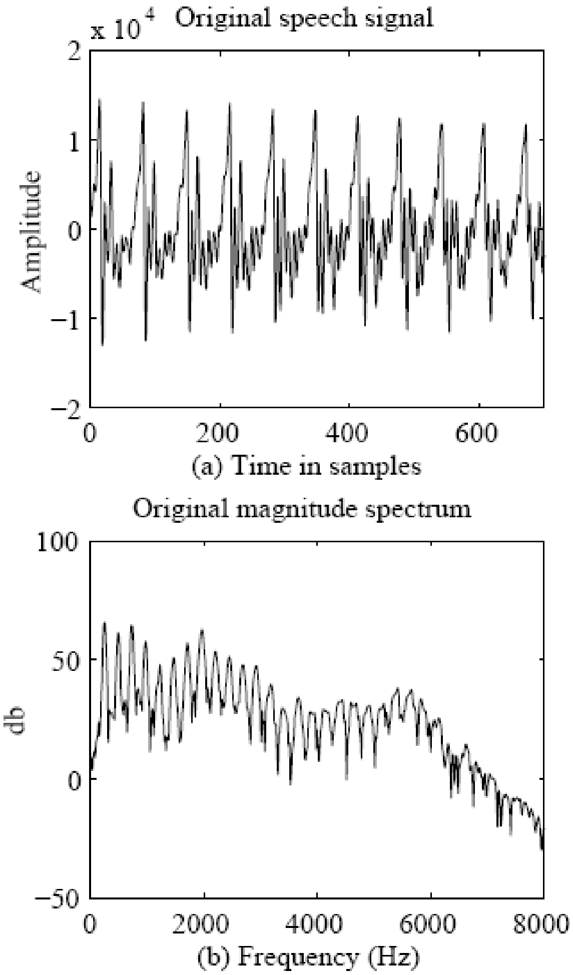
The harmonic part is represented by a sum of harmonics with continuously time-varying amplitudes and phases:

$$h_V(t) = \sum_{k=1}^{K(t)} a_k(t) cos\phi_k(t) \tag{2.10}$$

where:

- $a_k(t)$ and $\phi_k(t)$ are the amplitude and phase at time t of the k^{th} harmonic;
- $K(t)$ is the time-varying number of harmonics.

As far as the noise part is concerned, its frequency content is modeled by filtering a white Gaussian noise $u(t)$ by a time-varying, normalized all-pole filter $h(t)$ and its time-domain structure (the synchronization of the noise bursts with the pitch period) is represented by



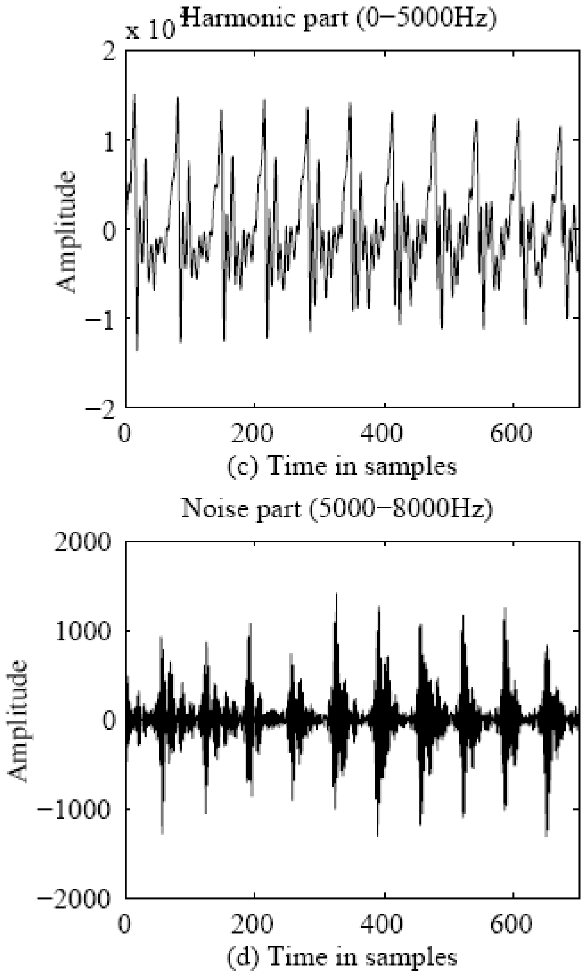


Figure 2.12: (a) Original speech signal (b) the magnitude spectrum of the speech signal (sampling frequency 16000Hz (c) the time-domain signal corresponding to frequency range 0-5000Hz (d) the time-domain signal corresponding to frequency range 5000Hz - 8000Hz, source [Stylianou96]

	LPC	TD-PSOLA	HNM
Segmental quality	Average[a]	High	High
Modeling of excitation and vocal tract	Separate	No modeling at all	Unified
Prosodic modifications	Easy	Limited	Easy
Smoothing of discontinuities at segment joins	Easy	Limited	Easy
Bit rate (speech coding)	Low	High	Low
Computation load (complexity)	Low	Low	High

Figure 2.13: Drawbacks and advantages of different speech waveform synthesis techniques

[a] Antiformants and mixed sounds badly modeled

multiplying the filtered noise by a piecewise linear energy envelope function $e(t)$:

$$n_V(t) = e(t)[h(t) * u(t)] \tag{2.11}$$

The synchronization must be properly performed otherwise two distinct sounds are perceived.

Finally, the voiced speech signal $s_V(t)$ is assumed to be the superposition of the harmonic part and the noise part: $s_V(t) = h_V(t) + n_V(t)$.

Unvoiced segments $s_{NV}(t)$ are only composed of a noise part $n_{NV}(t)$, without any specific time-domain structure:

$$s_{NV}(t) = n_{NV}(t) = h(t) * u(t) \tag{2.12}$$

The HNM analysis and synthesis are detailed in chapter 4.

Table 2.13 summarizes the characteristics of the three waveform synthesis techniques described, namely LPC, TD-PSOLA, and HNM.

Chapter 3

The Hidden Markov Model-Based Speech Synthesis System

Synthesis in HTS comprises two main steps [Dines03, Narayanan04, Tokuda02-1, Yoshimura99, Yoshimura01]: 1) The training of the HMMs that model the parameters extracted from the utterance database, taking into account contextual factors; 2) The construction of the composite HMM corresponding to the text to synthesize, and the estimation of the most likely parameters, from which the desired text is generated. Training and synthesis are respectively described in section 3.1 and 3.2. Section 3.3 details the maximum likelihood estimation of parameters. Details on HMM modeling are provided in the appendix B.

3.1 Training

During the training part, first, the spectrum parameters, the excitation parameters, and the state-duration densities are extracted from an utterance database. Then, these parameters are modeled by context-dependent phoneme HMMs, as shown in Figure 3.1.

More precisely, the spectrum parameters are the Mel-cepstral coefficients, plus the delta and delta-delta coefficients. The excitation parameters are the logarithm of the fundamental frequency, plus its dynamic features. The spectrum and excitation parameters are the two streams of the feature vector, represented on Figure 3.2. The state-duration densities characterize the temporal structure of speech.

The output probabilities of the spectrum parameters are single Gaussian distributions. As far as the excitation parameters are concerned, i.e. the fundamental frequency plus its dynamic features, they can be one-dimensional, continuous values in the case of voiced frames,

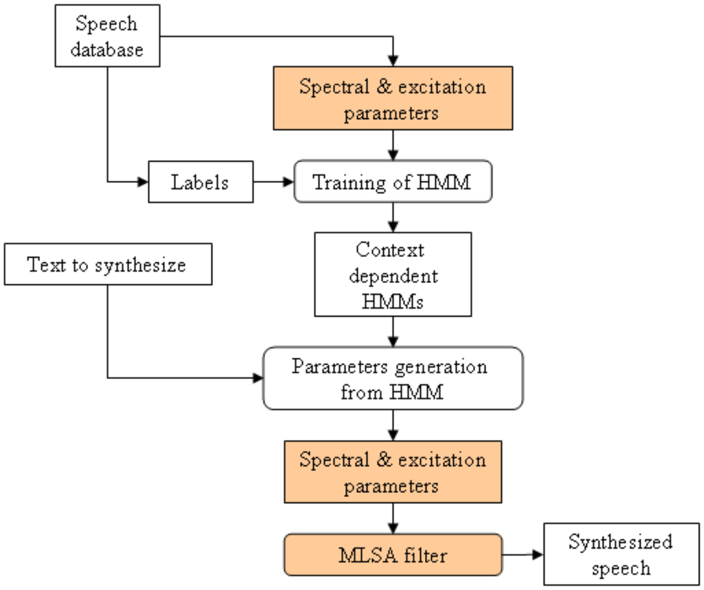


Figure 3.1: HMM-based speech synthesis system

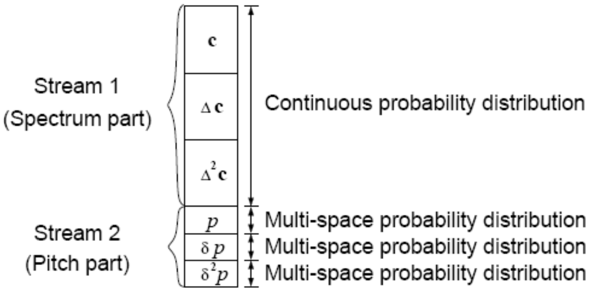

Stream 1
(Spectrum part)
Stream 2
(Pitch part)
c
Δc
Δ²c
p
δp
δ²p
Continuous probability distribution
Multi-space probability distribution
Multi-space probability distribution
Multi-space probability distribution

Figure 3.2: Feature vector, source [Yoshimura99]

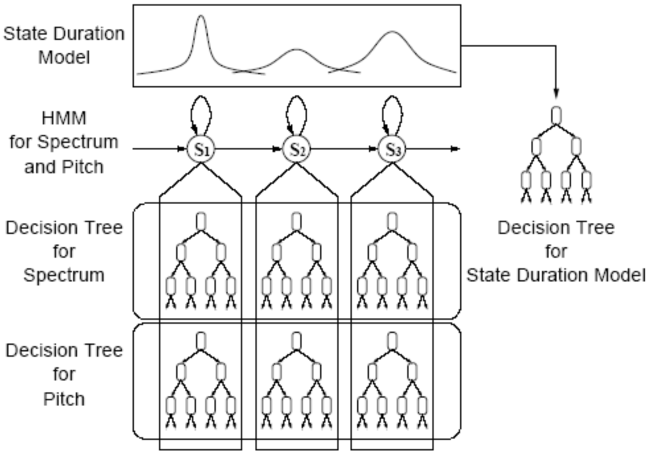

State Duration Model
HMM for Spectrum and Pitch
S1
S2
S3
Decision Tree for Spectrum
Decision Tree for Pitch
Decision Tree for State Duration Model

Figure 3.3: Decision trees, source [Yoshimura99]

and discrete symbols, that is a zero-dimensional observation vector, in the case of unvoiced frames. This kind of observation sequence is modeled by an HMM-based on a multi-space probability distribution (MSD-HMM) [Tokuda02-1]. In other words, the MSD-HMM can model a sequence of observations with variable dimensionality. Concerning the state-duration probability $p_q(d)$, which is the probability of d consecutive observations in state q, it is is supposed to be Gaussian, and the state durations of each phoneme HMM are modeled by a multivariate Gaussian distribution. The HMMs used in the system have a left-to-right structure with no skip. Therefore, the state-duration density of a phoneme HMM has a dimensionality equal to the number of states included in this phoneme HMM, with the i^{th} dimension of the state-duration density corresponding to the ith state of the HMM. As the simultaneous training of the HMMs and their state-duration densities requires a very large storage and computation load, the system HTS estimates the state-duration densities by using the probabilities obtained in the last iteration of embedded reestimation.

These parameters are influenced by many contextual factors, linked with the considered language. Contextual factors include co-articulatory effects, phone identity factors, stress-related factors, locational factors, etc. They are extracted by a text analyzer appropriate for the language, and the acoustic variations associated with these factors are captured by using context-dependent HMMs. The modeling of phones according to the context in which they occur greatly improves synthesis quality. Unfortunately the combinations of contextual factors increase as a power of the number of factors. This implies two restrictions; 1) the accurate estimation of the context-dependent HMMs would require a huge amount of training data; 2) it would take an unrealistic long time to record a database including all combinations of contextual factors. To circumvent these problems HTS applies a clustering algorithm

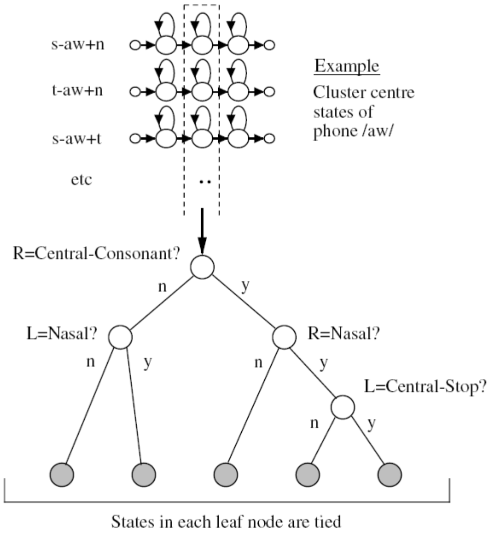


Figure 3.4: The phonetic decision tree clustering process, source [Young02]

[Dines03, Odell95] to the distributions of the spectrum, the fundamental frequency and the state durations. As the influences of each of these parameters are different, they are clustered independently, as represented on Figure 3.3. The idea behind clustering is that the acoustic realizations of a phone occurring in different contexts may be very similar. Thus, models (or parts of models) can be shared between these contexts. Therefore clustering allows to estimate the model parameters robustly without sacrificing the advantages of context dependent modeling. There exists two methods of clustering: a data-driven (or bottom-up) approach, and a tree-based (or top-down) approach.

In the data-driven approach, contexts are initially assumed to be all different, then a merging process is applied to produce more trainable but less specific models. Therefore this approach requires examples of each context to produce initial estimates of the model parameters used during clustering. Moreover accurate estimates require a sufficient number of examples. This means that the data-driven approach cannot estimate models for contexts that occur during synthesis but only occur a few times, or don't occur at all in the training data.

In the tree-based approach, contexts are initially assumed to be all the same and grouped, then a splitting process is used to produce more specific and more accurate models. As opposed to the data-driven approach, the tree-based technique is able to synthesize unseen contexts by using linguistic knowledge to determine which contexts in the training data are acoustically similar to the unseen ones. As a consequence, this approach, also called decision tree clustering, or Classification and Regression Tree (CART) clustering, is the approach used in HTS.

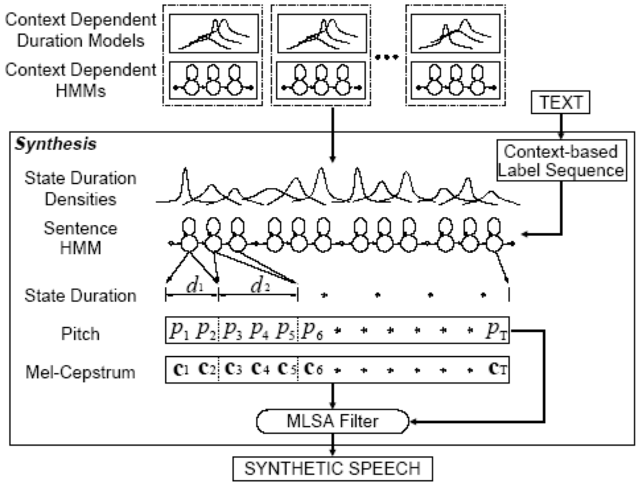

Context Dependent Duration Models
Context Dependent HMMs
TEXT
Synthesis
Context-based Label Sequence
State Duration Densities
Sentence HMM
State Duration
d_1
d_2
Pitch
p_1 p_2 p_3 p_4 p_5 p_6 p_T
Mel-Cepstrum
c_1 c_2 c_3 c_4 c_5 c_6 c_T
MLSA Filter
SYNTHETIC SPEECH

Figure 3.5: Synthesis part of the system, source [Yoshimura99]

The decision tree is a binary tree whose each node is associated with yes/no questions of the form "Is the model's context a member of the set X ?" where X may represent any prosodic (pitch, part-of-speech (POS), stress, duration, phrasing, etc.) or phonetic context. This is illustrated on Figure 3.4. The tree is constructed automatically, starting from a single root node representing all contexts, and iteratively splitting the data by using the question that provides the greatest modeling improvement (determined by a given cost criterion). Usually, two criteria must be met to create a leaf node: the modeling improvement must exceed a given threshold and their occupancy must reach a certain level.

3.2 Synthesis

The first step of the synthesis part is the concatenation of the context-dependent HMMs corresponding to the labels sequence derived from the text to synthesize, i.e. a part of the first chapter of "Alice's Adventures in Wonderland", by Lewis Carroll. Secondly, the state durations of the composite HMM are determined in such a way that their output probabilities are maximized. According to these durations, the spectrum and excitation parameters are obtained again by maximizing their output probabilities, as detailed in section 3.3. Finally, these parameters permit the generation of the speech waveform by using an excitation-generation module and a Mel Log Spectrum Approximation (MLSA) filter, as shown in Figure 3.5.

3.3 Speech parameter generation from an HMM

As explained above, an observation vector is emitted by each state according to its probability function. If we consider a continuous density Hidden Markov Model (CDHMM) comprising N states, with M mixture components per state and an observation vector $O = [o'_1, o'_2, \ldots, o'_T]'$, the emission likelihood of the observation o_t in the state S_j is [Dines03, Gosselin00]:

$$\begin{aligned} p(o_t|q_j = S_j) &= \sum_{m=1}^{M} c_{jm} b_{jm}(o_t) \\ &= \sum_{m=1}^{M} c_{jm} \mathcal{N}(o_t; \mu_{jm} \Sigma_{jm}) \end{aligned}$$

If we consider that the observation vector doesn't include any dynamic features but only static features, that is $o_t = c_t$, then the observations that maximize $p(O|\lambda)$ are the most likely emitted observation sequence:

$$p(O|\lambda) = p(O|Q, \lambda) P(Q|\lambda) \tag{3.1}$$

where Q is the state and mixture sequence: $Q = (q, i)$, $q = \{q_1, q_2, \ldots, q_T\}$, $i = \{i_1, i_2, \ldots, i_T\}$.

If Q is given, maximizing $p(O|\lambda)$ and $p(O|Q, \lambda)$ are equivalent:

$$\begin{aligned} \log p(O|Q, \lambda) &= \log \prod_{t=1}^{T} b_{q_t, i_t}(o_t) \\ &= -\frac{1}{2}(O - \mu)' \Sigma^{-1} (O - \mu) - \frac{1}{2} \sum_{t=1}^{T} \log |\Sigma_{q_t}| - \frac{1}{2} T D \log 2\pi \end{aligned} \tag{3.2}$$

where:

- $\mu = [\mu'_{q_1,i_1}, \mu'_{q_2,i_2}, \ldots, \mu'_{q_T,i_T}]'$;
- $\Sigma = diag[\Sigma_{q_1,i_1}, \Sigma_{q_2,i_2}, \ldots, \Sigma_{q_T,i_T}]$;
- T is the length of the observation vector sequence in frames;
- D is the dimensionality of the static feature vectors.

In equation (3.2), $\log p(O|Q, \lambda)$ is maximized if its derivative is equal to zero:

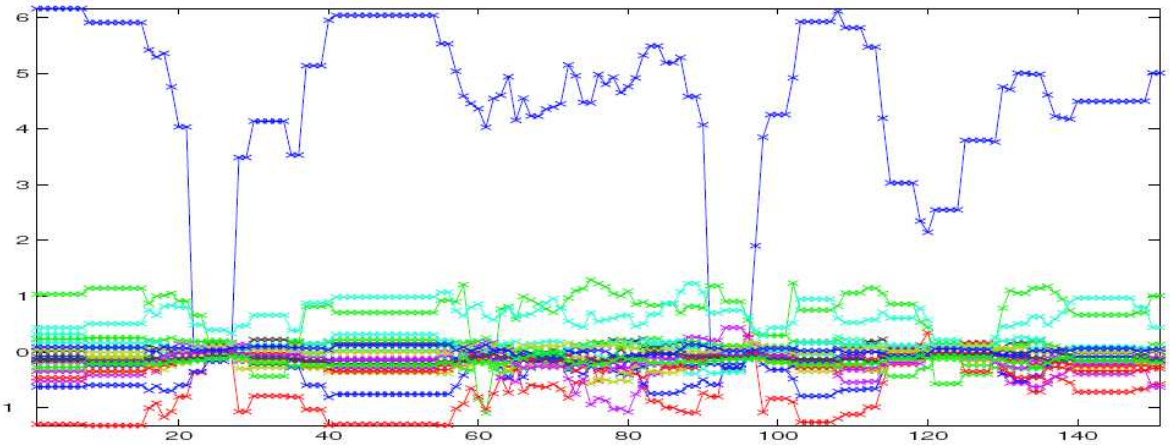


Figure 3.6: Feature trajectory sequence generated from the HMM for the utterance segment "... occasionally get the ..." trained from Mel-cepstral features without dynamic statistics, source [Dines03]

$$\frac{\partial(\log p(O|Q,\lambda))}{\partial c} = -\Sigma^{-1}c + \Sigma^{-1}\mu = 0 \tag{3.3}$$

This shows that the maximum is obtained when $c = \mu$, that is, in other words, the most likely emitted observation sequence is the mean vector sequence, independent of the covariance Σ.

Figure 3.6 represents the feature trajectory sequence generated from a single-mixture HMM constructed for the utterance segment "... occasionally get the ..." and trained from Mel-cepstral features without dynamic statistics. Feature transitions are sharp, which will lead to audible discontinuities during synthesis. This problem can be overcome by incorporating the dynamic features in the observation vectors [Masuko96, Plumpe98, Tokuda95-1, Tokuda95-2, Tokuda00].

By incorporating dynamic features, the rate of change of feature vectors over a time window, W, is taken into account. The feature vectors are now defined by $o_t = [c_t', \Delta c_t', \Delta^2 c_t']'$ where:

$$\Delta^{(n)} c_t = \sum_{\tau=-L(n)}^{L(n)} w^{(n)}(\tau) c_{t+\tau} \qquad (n = 0, 1, 2) \tag{3.4}$$

setting $\Delta^{(0)} c_t = c_t$, $\Delta^{(1)} c_t = \Delta c_t$ and $\Delta^{(2)} c_t = \Delta^2 c_t$.

With this new definition of feature vectors equation (3.2) becomes

$$\begin{aligned}
\log p(O|Q,\lambda) &= (O-\mu)'\Sigma^{-1}(O-\mu) - \frac{1}{2}\log|\Sigma| - \frac{3TD}{2}\log 2\pi && (3.5)\\
&= (Wc-\mu)'\Sigma^{-1}(Wc-\mu) - \frac{1}{2}\log|\Sigma| - \frac{3TD}{2}\log 2\pi && (3.6)\\
&= -\frac{1}{2}\epsilon(c) - \frac{1}{2}\log|\Sigma| - \frac{3TD}{2}\log 2\pi && (3.7)
\end{aligned}$$

where:

- μ and Σ are defined as in equation (3.2);
- $\epsilon(c) = (Wc-\mu)'\Sigma^{-1}(Wc-\mu)$;
- $W = [w_1, w_2, \ldots, w_T]'$;
- $w_t = [w_t^{(0)}, w_t^{(1)}, w_t^{(2)}]$;
- $w_t^{(n)} = [\underset{\mathbf{1^{st}}}{0_{M\times M}}, \ldots, 0_{M\times M}, w(n)(-L^{(N)})\underset{\mathbf{(t-L^{(N)})^{th}}}{I_{M\times M}}, \ldots, \underset{\mathbf{t^{th}}}{w(0)I_{M\times M}},$ $\ldots, w(n)(L^{(N)})\underset{\mathbf{(t+L^{(N)})^{th}}}{I_{M\times M}}, 0_{M\times M}, \ldots, \underset{\mathbf{T^{th}}}{0_{M\times M}}]$.

The minimization is performed by taking $\partial \log p(0|Q,\lambda)/\partial c = 0$:

$$(W'\Sigma^{-1}W)c - W'\Sigma^{-1}\mu = 0 \qquad (3.8)$$

This equation can also be written as:

$$Rc = r \qquad (3.9)$$

where:

$$\begin{aligned}
R &= W'\Sigma^{-1}W && (3.10)\\
r &= W'\Sigma^{-1}\mu && (3.11)
\end{aligned}$$

If this is solved directly $O(T^3M^3)$ operations are required. [Tokuda95-1, Tokuda95-2] show that there exists a fast algorithm to reduce significantly the number of operations required to determine c.

Figure 3.7 represents the feature trajectory sequence generated from maximum likelihood estimation of feature vectors including delta and delta-delta statistics for the HMM corresponding to the utterance segment "... occasionally get the ...". Comparison between Figures 3.6 and 3.7 show that incorporation of dynamic features has greatly reduced spectral discontinuities.

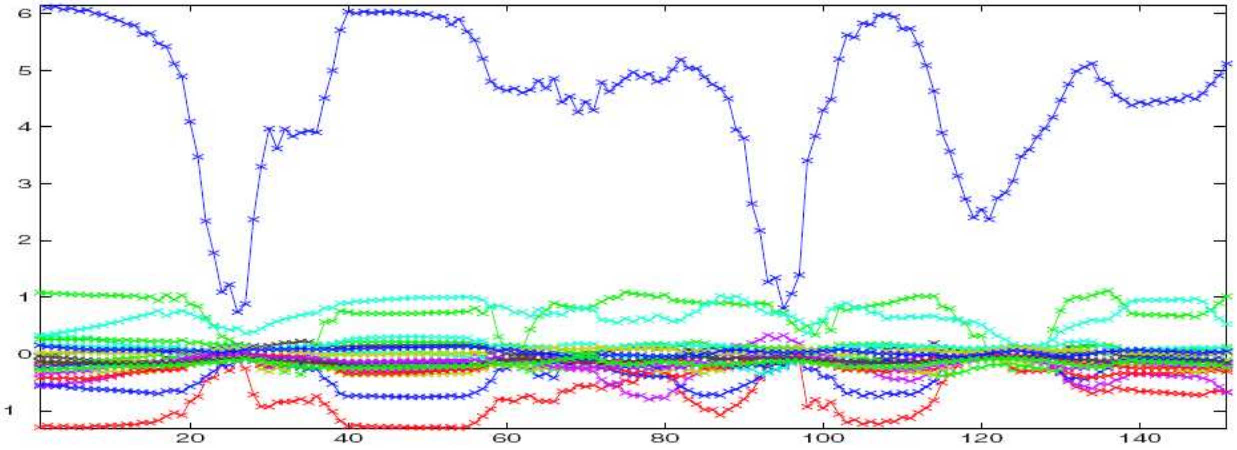


Figure 3.7: Feature trajectory sequence generated from the HMM for the utterance segment "... occasionally get the ..." trained from Mel-cepstral features including dynamic statistics, source [Dines03]

Chapter 4

A Harmonic plus Noise Model

The HNM algorithm comprises the HNM analysis and the HNM synthesis [Stylianou96, Stylianou01]. During the HNM analysis the HNM parameters of every acoustic unit of the recorded database are calculated and the original database is replaced by another database consisting of the calculated HNM parameters. The HNM synthesis consists in finding in the database the parameters corresponding to the text to synthesize, in possibly performing prosodic modifications and in synthesizing the speech waveform.

4.1 HNM analysis: estimation of the HNM parameters from the utterance database

The HNM analysis consists of the estimation of the parameters of the harmonic and noise part. The first parameter to be estimated is the fundamental frequency. This is followed by the voiced/unvoiced decision. In the case of voiced segments, the maximum voiced frequency can then be determined. Note that the accuracy of these three estimations greatly influences the quality of the synthetic speech generated by HNM synthesis. The remaining parameters are estimated pitch-synchronously, so the analysis time instants are defined. The amplitudes and phases of the harmonics are then calculated. Finally the parameters of the noise part are computed.

From a practical point of view, Festival provides a program "pitchmark" which uses electroglottograph (EGG) signal to derive pitch-marks, i.e. the locations of the short-time energy peak of each pitch pulse, which correspond to the glottal closure instants, as represented on Figure 4.1. The pitch-marks given for unvoiced frames are of fixed duration. The voiced/unvoiced decision is also provided by Festival. Examples of values issued by Festival are given below:

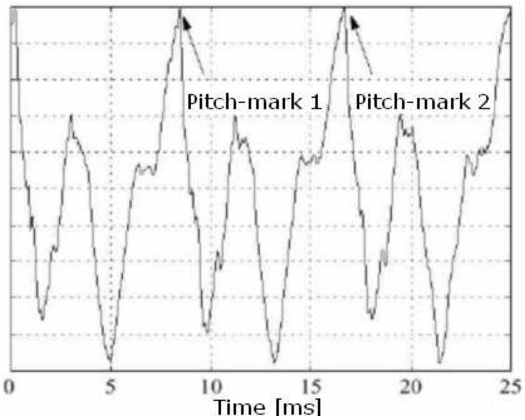


Figure 4.1: Pitch-marks of a speech waveform

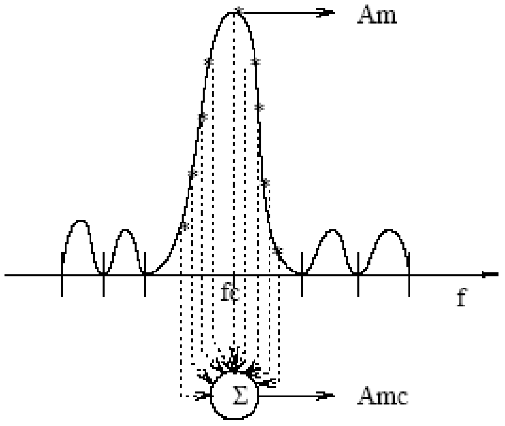


Figure 4.2: Cumulative amplitude, source [Stylianou96]

```
2.446312  1
2.454625  1
2.464062  1
```

In the case of voiced frames (value '1' in the second column), the fundamental frequency $\hat{f}_0$ is $(2.454625 - 2.446312)^{-1} = 120.2935$Hz for the first frame, and $(2.464062 - 2.454625)^{-1} = 105.9659$Hz for the second one.

The maximum voiced frequency, F_M, is estimated every $10ms$ using the following algorithm, first applied to the frequency range $[\frac{\hat{f}_0}{2}, \frac{3\hat{f}_0}{2}]$:

- Search in the frequency range for the largest amplitude, A_m, of frequency f_c;

- Calculate the cumulative sum, A_{mc}, of the amplitudes of all the samples situated between the minima surrounding the peak of frequency f_c, as represented on Figure 4.2;
- Search in the frequency range for the other peaks, of frequencies f_i, and determine $A_m(f_i)$ and $A_{mc}(f_i)$ for each peak;
- Calculate the mean of the cumulative sums, denoted by $\overline{A_{mc}}(f_i)$;, and determine the number, L, of the harmonic nearest to f_c;
- If $\frac{A_{mc}(f_c)}{\overline{A_{mc}}(f_i)} > 2$ or $A_m(f_c) - \max\{A_m(f_i)\} > 13dB$ then, if $\frac{|f_c - L\hat{f}_0|}{L\hat{f}_0} < 20\%$, the frequency f_c is labelled 'voiced'; otherwise it is labelled 'unvoiced'.

This algorithm is then applied to the frequency range $[\frac{3\hat{f}_0}{2}, \frac{5\hat{f}_0}{2}]$. The process is repeated throughout the spectrum of the frame.

Generally the separation between the voiced and the unvoiced parts of the spectrum is not precise. To circumvent this problem, a vector consisting of binary values is constructed, with the frequencies labeled "voiced" and "unvoiced" respectively represented by the values "1" and "0". This vector is then filtered by a three-point median smoothing filter, which enables the separation of the two parts. The maximum voiced frequency corresponds to the last "1" in the vector. Figure 4.3(a) illustrates the detection of voiced frequencies (marked by stars) in an amplitude spectrum; Figure 4.3(c) shows the estimated maximum voiced frequency for the speech waveform represented on Figure 4.3(b).

While the maximum voiced frequency has been estimated every $10ms$, other parameters are estimated pitch-synchronously. So they require the definition of the analysis time instants, t_a^i. For the voiced segments of speech, the instants are pitch-synchronous: $t_a^{i+1} = t_a^i + T_0(t_a^i)$, with $T_0(t_a^i)$ the pitch period at instant t_a^i; for the unvoiced segments, the instants are separated by $10ms$.

The next parameter to estimate is the complex amplitude of the harmonics. This estimation is based on the hypothesis that the amplitude of the harmonics, the pitch period and the maximum voiced frequency are constant around the analysis time instant t_a^i:

$$a_k(t) = a_k(t_a^i)$$

$$T_0(t) = T_0(t_a^i)$$

$$F_M(t) = F_M(t_a^i)$$

for small $|t - t_a^i|$.

Based on these stationary conditions, the instantaneous phase of k^{th} harmonic $\phi_k(t)$ in the neighborhood of t_a^i can be written as:

$$\phi_k(t) = \phi_k(t_a^i) + k2\pi f_0(t_a^i)(t - t_a^i) \tag{4.1}$$

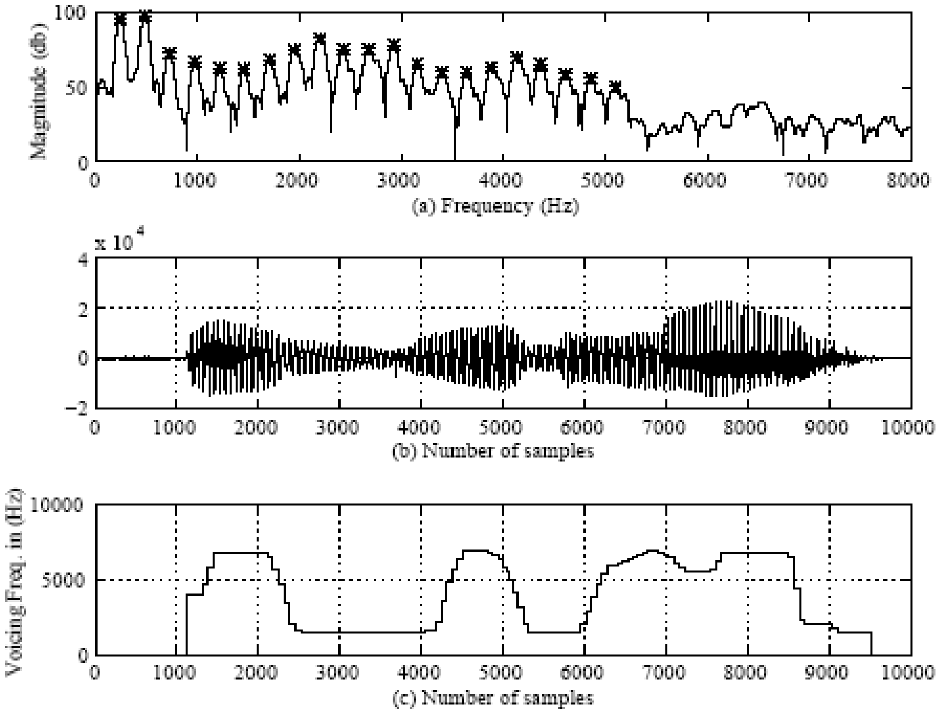


Figure 4.3: (a) Estimation of the maximum voiced frequency for a voiced frame (b) A voiced speech segment (c) Estimation of the maximum voiced frequency for the speech segment represented on (b), source [Stylianou96]

Then the harmonic part can be developed as:

$$\hat{h}(t) = \sum_{k=-L}^{L} A_k(t_a^i) e^{j 2\pi k f_0(t_a^i)(t - t_a^i)} \tag{4.2}$$

where $L = F_M(t_a^i)/f_0(t_a^i)$ is the number of harmonics and $A_k(t_a^i)$ is the complex amplitude of the k^{th}.

The complex amplitudes are evaluated so as to minimize the following weighted least squares criterion

$$\epsilon = \sum_{t=t_a^i-N}^{t_a^i+N} w^2(t)(s(t) - \hat{h}(t))^2 \tag{4.3}$$

where:

- $s(t)$ is the original signal;
- $\hat{h}(t)$ is the harmonic part defined in equation (4.2);
- $w(t)$ is a weighting function (typically a Hamming window), used to give more weight in the least squares process to the data around t_a^i;
- N is the integer closest to local pitch period $T_0(t_a^i)$.

Equation (4.3) shows that the harmonics are estimated on frames centered around the analysis instant t_a^i and of length equal to $M = 2N + 1$, called analysis frames.

This minimization is performed by using matrix notation. The harmonic part becomes simply

$$\hat{h} \quad = \quad Bx \tag{4.4}$$

where:

- B is a matrix defined by

$$\begin{pmatrix} e^{j(-L)2\pi f_0(t_a^i-N)} & e^{j(-L+1)2\pi f_0(t_a^i-N)} & \dots & e^{j(+L)2\pi f_0(t_a^i-N)} \\ e^{j(-L)2\pi f_0(t_a^i-N+1)} & e^{j(-L+1)2\pi f_0(t_a^i-N+1)} & \dots & e^{j(+L)2\pi f_0(t_a^i-N+1)} \\ \vdots & \vdots & & \vdots \\ e^{j(-L)2\pi f_0(t_a^i+N)} & e^{j(-L+1)2\pi f_0(t_a^i+N)} & \dots & e^{j(+L)2\pi f_0(t_a^i-N)} \end{pmatrix}$$

- x, the vector of unknown parameters, is defined by

$$\begin{pmatrix} A_{-L} \\ A_{-L+1} \\ \vdots \\ A_L \end{pmatrix}$$

The solution of this least-squares problem is finally obtained thanks to the normal equations [Lawson74]

$$(B^T W^T W B)x = B^T W^T W s \tag{4.5}$$

where:

- W is a diagonal matrix whose elements constitute the weighting window, defined by

$$\begin{pmatrix} w(-N) & 0 & \dots & 0 \\ 0 & w(-N+1) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & w(N) \end{pmatrix}$$

- s, the original signal, is defined by

$$\begin{pmatrix} s(-N) \\ s(-N+1) \\ \vdots \\ s(N) \end{pmatrix}$$

The equation (4.5) can also be written as:

$$Rx = b \tag{4.6}$$

where:

- The matrix R is defined by $R = B^T W^T W B$ and its element $r_{i,k}$ by

$$\begin{aligned} r_{ik} &= \sum_{t=t_a^i-N}^{t_a^i+N} w^2(t) e^{j2\pi(i-L-1)f_0 t - j2\pi(k-L-1)f_0 t} && (4.7) \\ &= \sum_{t=t_a^i-N}^{t_a^i+N} w^2(t) e^{j2\pi(i-k)f_0 t} && (4.8) \end{aligned}$$

- The vector b is given by $b = B^T W^T W s$ and its element b_k by

$$b_k = \sum_{t=t_a^i-N}^{t_a^i+N} w^2(t) s(t) e^{-j2\pi(k-L-1)f_0 t} \tag{4.9}$$

$$(i = 1, \dots, 2L+1; k = 1, \dots, 2L+1)$$

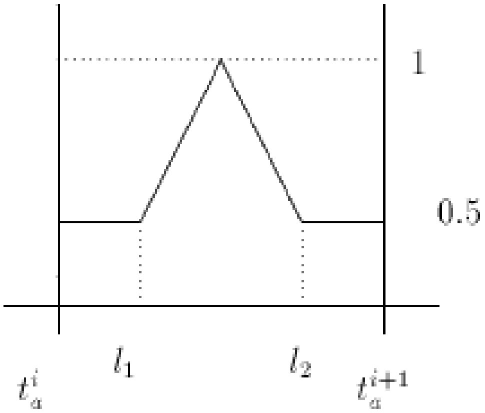


Figure 4.4: The time-domain envelope used for the time-behavior of the noise part. The t_a^i and t_a^{i+1} represent two successive analysis time-instants. Typical values for l_1 and l_2 are: $l_1 = 0.15(t_a^{i+1} - t_a^i)$ and $l_2 = 0.85(t_a^{i+1} - t_a^i)$, source [Stylianou96]

R is a Toeplitz matrix, that is a matrix only defined by its first column, as

$$\begin{aligned} r_{i+p,k+p} &= \sum_{t=t_a^i-N}^{t_a^i+N} w^2(t) e^{j2\pi((k+p)-(i+p))f_0 t} && (4.10) \\ &= \sum_{t=t_a^i-N}^{t_a^i+N} w^2(t) e^{j2\pi(k-i)f_0 t} && (4.11) \\ &= r_{i,k} && (4.12) \end{aligned}$$

Therefore, the equation (4.6) can be solved by fast algorithms like the algorithms of Levinson or Schur.

The last parameters to estimate are the parameters of the noise part for voiced and unvoiced frames. As explained above, an all-pole filter $h(t)$ of order 18 is used to model the spectral density function of the speech signal. It is estimated by a linear prediction analysis (cf. equation (2.3)) based on the correlation function of the whole analysis frame in the case of voiced frames, and of segments of $20ms$ of signal in the case of unvoiced frames. Note that, before prediction analysis, frames are pre-emphasized and windowed, as described in chapter 2.3. The filter is used to filter white Gaussian noise. If the frame to synthesize is voiced, the filtered noise is finally multiplied by an energy-envelope function $e(t)$, which is defined as a triangular-like time-domain envelope, as illustrated in Figure 4.4. This envelope doesn't require the estimation of any other parameter.

In short, the HNM parameters to extract from each analysis frame are listed in table 4.5.

For the voiced frames:	For the unvoiced frames:
The pitch and the maximum voiced frequency The linear prediction coefficients and σ[a] The amplitudes and the phases of the harmonics	The linear prediction coefficients and σ

[a] the standard deviation of the prediction error

Figure 4.5: Summary of HNM parameters

4.2 HNM synthesis: speech synthesis from the HNM parameters

In HNM synthesis, the harmonic and the noise parts are synthesized separately. Note that the HNM parameters involved in the synthesis process are supposed to be pitch-synchronous. This section describes the synthesis of the harmonic part, of the noise part, and finally of the resulting speech signal. The synthesis process is summarized on Figure 4.6.

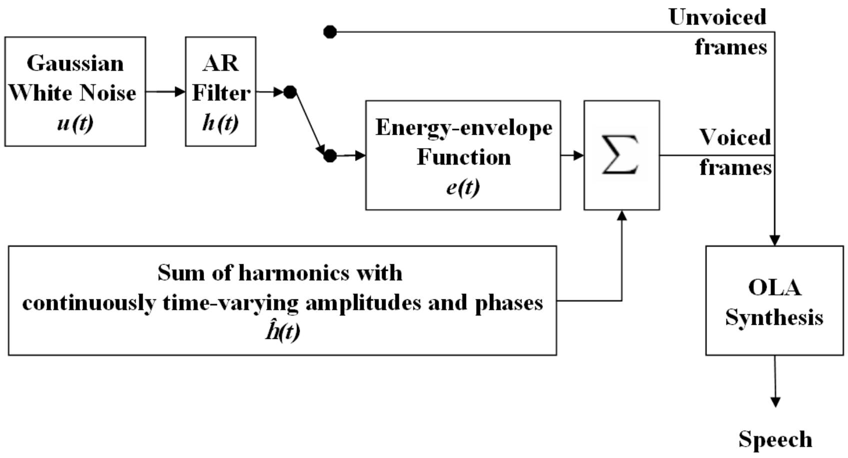

Gaussian White Noise u(t)
AR Filter h(t)
Unvoiced frames
Energy-envelope Function e(t)
∑
Voiced frames
Sum of harmonics with continuously time-varying amplitudes and phases ĥ(t)
OLA Synthesis
Speech

Figure 4.6: HNM synthesis

The synthesis of the harmonic part is performed based on the harmonic amplitudes a_k, the harmonic phases ϕ_k and the pitch frequency f_0 of the adequate frame. The harmonic part $\hat{h}(t)$ of the i^{th} frame is generated according to the equation:

$$\hat{h}(t) = \sum_{k=1}^{L(t_s^i)} a_k(t_s^i) \cdot cos(\phi_k(t_s^i) + k2\pi f_0(t_s^i)t) \tag{4.13}$$

where:

- $t = 0, 1, \ldots, N$, where N is the length of the synthesis frame, that is, also, the integer closest to the pitch period at instant t_s^i (expressed in terms of samples);

- t_s^i are the synthesis instants, which are pitch-synchronous and equal to the analysis instants when no prosodic modification is performed: $t_s^i = t_a^i \forall i$;
- $L(t_s^i)$ is the number of harmonics of the i^{th} frame.

The synthesis of the noise part starts from a unit-variance white Gaussian noise whose length is twice the local pitch period. This noise is then filtered by the all-pole filter whose denominator is defined by the linear prediction coefficients and whose numerator is the standard deviation of the prediction error, σ. The noise signal obtained at this point of the synthesis has the same variance as the original signal. Finally, and only in the case of the noise part of voiced frames, the filtered noise is multiplied by the energy-envelope function shown in Figure 4.4.

The synthesized speech signal is computing by overlapping and adding voiced and unvoiced frames. Unvoiced frames consist simply of the noise part, while voiced frames are obtained by adding the noise part and the harmonic part after having filtered the noise signal by a high-pass filter of cut-off frequency equal to the maximum voiced frequency, F_M.

Chapter 5

Integration of the Harmonic plus Noise Model into the Hidden Markov Model-Based Speech Synthesis System

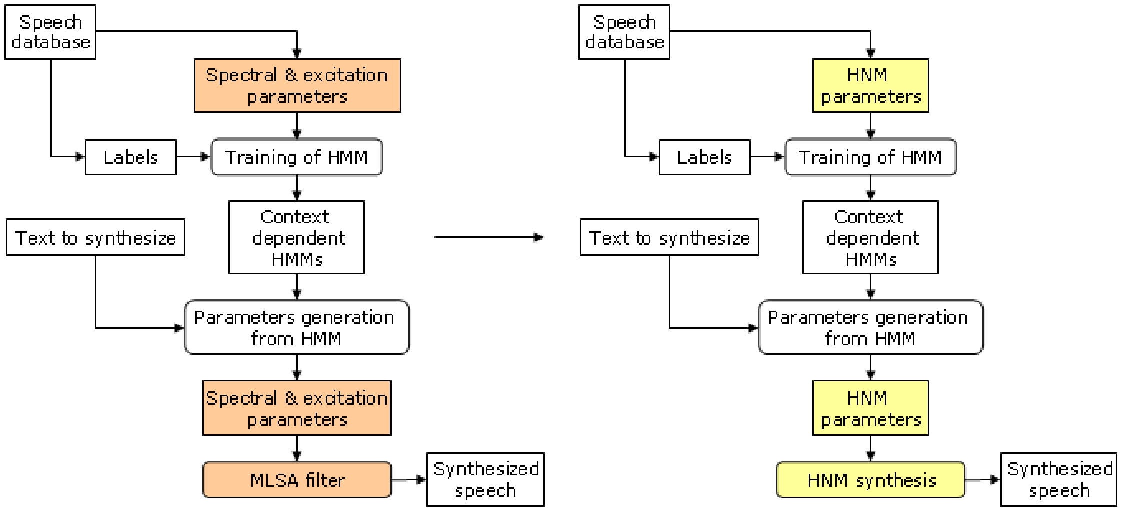


Figure 5.1: TTS in the HTS system (left scheme) and in this work (right scheme).

The general objective of this thesis is to perform a Hidden Markov Model-based speech synthesis using a Harmonic plus Noise Model of speech. This speech synthesis system would have the advantages of model-based speech synthesis techniques, described in section 2.2.2, and would overcome the main drawback of the baseline HMM-based speech synthesis system, i.e. HTS, that is the fact that the synthesized speech sounds artificial and metallic. Indeed this disadvantage is due to the source filter modeling approach, and the objective of this thesis is to replace this approach by the HNM synthesis, which sounds much more natural.

More precisely, this implies the following modifications of the HTS system: the substitution of the Mel-cepstral and pitch analysis for the HNM analysis, and of the MLSA synthesis for the HNM synthesis. Figure 5.1 illustrates the differences between the TTS systems used in HTS (left scheme) and in this work (right one).

This chapter describes the text-to-speech system used to integrate HNM and HMM syntheses, represented on Figure 5.2. Like HTS, this system includes a training part and a synthesis part, described in section 5.1 and 5.2. Several syntheses have been carried out, making different design choices, which are explained during training description. The qualities of the resultant syntheses are described after the synthesis description.

5.1 Training

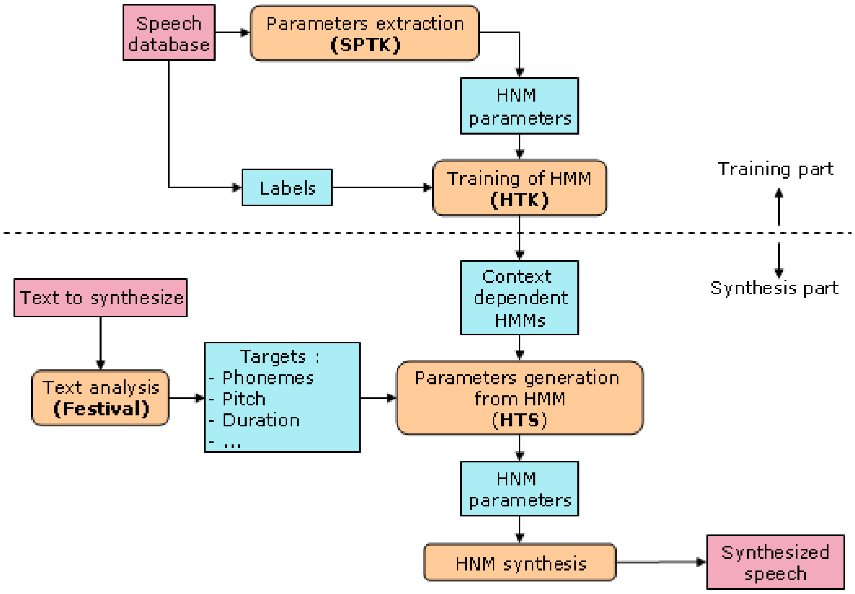


Figure 5.2: TTS synthesis with HNM parameters and HMMs

Training is based on a labeled database of utterances, i.e. the CMU_ARCTIC database spoken by a Scottish English male speaker [Festvox], from which HNM parameters are extracted, either with the HNM analysis module [Van Dromme05] or with SPTK [SPTK]. Several design choices are made at this level:

- Frame size and shift
  HNM parameters have been extracted at a constant rate of $5ms$ using SPTK and pitch-

synchronously using the HNM analyss module. Note that the pitch-synchronous extraction has been followed by a resampling at the rate of $5ms$, as it is required to train HMMs. This has resulted in two different syntheses, which are compared in section 5.2;

- Type of features
  - The fundamental frequency, and the voicing decision, extracted with Festival, as explained in section 4.1;
  - The maximum voiced frequency, extracted with the HNM analysis module, has been handled in two ways: in the simplified case, it has been assumed to be equal to 0 or half the sampling frequency (binary excitation), like in HTS; in a more precise modeling the value calculated during the HNM analysis has been included in the feature vector used for training;
  - The linear prediction coefficients and the standard deviation of the prediction error, σ, extracted with either SPTK or the HNM analysis module. In HTS, the spectrum was modeled by Mel-cepstral coefficients, which are widely used in speech recognition/synthesis for their ability to mimic the auditory mechanisms in humans, that is the non-linear perception of pitch. Linear prediction coefficients have been used in this work to derive the harmonic amplitudes, as explained in section 5.2. In practice, linear prediction coefficients, $a(n)$, are extracted from the utterance database, they are converted to linear predictive cepstral coefficients, $c(n)$, which present interpolative properties according to:

    $$c(n) = \begin{cases} \ln(\sigma) & \text{if } n = 0; \\ -a(n) - \sum_{k=1}^{n-1} \frac{k}{n} c(k) a(n-k) & \text{if } 1 \leq n \leq P; \\ -\sum_{k=n-P}^{n-1} \frac{k}{n} c(k) a(n-k) & \text{if } n > P. \end{cases} \tag{5.1}$$

    where P is the order of linear prediction coefficients;

    Finally, they are possibly resampled.
- Dynamic features are used, as they are known for increasing the synthesis quality [Dines03].
- Order of spectral feature analysis

  The orders of linear prediction and linear predictive cepstral coefficients are respectively 18 and 24.

Once the parameters have been extracted, training of HMMs is performed with the Hidden Markov Model ToolKit (HTK) [Young02], which is a software that provides a set of library modules and tools for building and manipulating HMMs.

Before HMM training, some other design choices must be made, namely:

- The structure of the observation vectors, which can be split into S independent data streams o_{st}, possibly weighted by a stream weight vector, γ:

  The vector comprises one stream of length 75 for the linear predictive cepstral coefficients and the delta and delta-delta coefficients, three streams of length 1 for the fundamental frequency and its dynamic features, and, in the case of the synthesis including the modeling of the maximum voiced frequency, three more streams of length 1 for the maximum voiced frequency and its dynamic features.

- The topology of HMMs, that is, essentially, the number of states (typically, three) and the number of mixture components per emitting state and per stream (in the case of CDHMM): HMMs consist of 7 states, whose first and last one are non-emitting states. For every state, every stream, except the stream of the LPCC coefficients, is modeled by a mixture of Gaussians having 2 components.

- The type of output distribution

  – The pitch is modeled by a multi-space probability distribution HMM (MSD-HMM), as explained in section 3.1 and in [Tokuda02-1].

  – The maximum voiced frequency may be also modeled by a MSD-HMM system, as the maximum voiced frequency takes its value among one-dimensional continuous values in the case of voiced frames, and it is not defined in the case of unvoiced frames. Instead of the maximum voiced frequency, we could have modeled the number of harmonics. The maximum voiced frequency is preferred because the number of harmonics is correlated with the pitch value, while the maximum voiced frequency and the pitch value may be assumed independent.

    As the distributions representing the pitch and the maximum voiced frequency have the same properties, these two parameters are both modeled by MSD-HMM.

  – The harmonic amplitudes, the linear prediction coefficients and σ can have a grouped modeling. Indeed, given the LPC coefficients and σ, the LPC envelope can be constructed and the harmonic amplitudes can be calculated by sampling this envelope at all the multiples of the pitch inferior to the maximum voiced frequency. This is described with more details in section 5.2. The LPC coefficients and σ are modeled by a multi-Gaussian distribution (GMM).

Introduction of streams slightly modifies the definition of the probability of observation o_t being emitted by state j, given in appendix B, which becomes, if GMM have been chosen as output distribution:

$$b_j(o_t) = \prod_{s=1}^{S} \left[\sum_{m=1}^{M_{js}} c_{jsm} \mathcal{N}(o_{st}; \mu_{jsm}, \Sigma_{jsm}) \right]^{\gamma_s} \tag{5.2}$$

where:

- M_{js} is the number of mixture components in state j for stream s;
- c_{jsm} is the weight of the m^{th} component;
- $\mathcal{N}(.;\mu,\Sigma)$ is a multivariate Gaussian with mean vector μ and covariance matrix Σ;
- γ_s is the weight of stream s, whose default value is one.

The HMM definition whose feature vector includes the maximum voiced frequency is given below in the HTK format :

```
~o <VecSize> 81 <USER> <MSDINFO> 7 0 1 1 1 1 1 1   → Observation vectors consisting of 81 elements, split into 7 streams, which are
                                                     all modeled by multi-space probability distribution ('1') except the first one ('0')

    <StreamInfo> 7 75 1 1 1 1 1 1                  → 7 distinct streams, the first comprising 75 components, and all the other ones,
                                                     1 component
<BeginHMM>
    <NumStates> 7                                  → 7 states, the first and last being non-emitting states
    <State> 2
        <Stream> 1                                 → Stream 1 includes the LPCC coefficients, the delta and delta-delta coefficients
            <Mean> 75
                 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
                 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
                 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
            <Variance> 75
                 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
                 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
                 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
        <Stream> 2                                 → Stream 2 consists of the logarithm of the fundamental frequency (log f_0)
        <NumMixes> 2
        <Mixture> 1 0.5
            <Mean> 1 0.0
            <Variance> 1 1.0
        <Mixture> 2 0.5
            <Mean> 0
            <Variance> 0
        <Stream> 3                                 → Δ log f_0
        <NumMixes> 2
        <Mixture> 1 0.5
            <Mean> 1 0.0
            <Variance> 1 1.0
        <Mixture> 2 0.5
            <Mean> 0
            <Variance> 0
        <Stream> 4                                 → Δ^2 log f_0
        <NumMixes> 2
        <Mixture> 1 0.5
            <Mean> 1 0.0
            <Variance> 1 1.0
        <Mixture> 2 0.5
            <Mean> 0
            <Variance> 0
        <Stream> 5                                 → Stream 5 consists of the logarithm of the maximum voiced frequency (log F_M)
        <NumMixes> 2
        <Mixture> 1 0.5
            <Mean> 1 0.0
            <Variance> 1 1.0
        <Mixture> 2 0.5
            <Mean> 0
            <Variance> 0
        <Stream> 6                                 → Δ log F_M
        <NumMixes> 2
        <Mixture> 1 0.5
            <Mean> 1 0.0
            <Variance> 1 1.0
        <Mixture> 2 0.5
            <Mean> 0
            <Variance> 0
        <Stream> 7                                    → Δ^2 log F_M
        <NumMixes> 2
        <Mixture> 1 0.5
            <Mean> 1 0.0
            <Variance> 1 1.0
        <Mixture> 2 0.5
            <Mean> 0
            <Variance> 0
            .
            .
            .
    <State> 7
        <Stream> 1
            <Mean> 75
                0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
                0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
                0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
            <Variance> 75
                1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
                1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
                1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
        <Stream> 2
        <NumMixes> 2
        <Mixture> 1 0.5
            <Mean> 1 0.0
            <Variance> 1 1.0
        <Mixture> 2 0.5
            <Mean> 0
            <Variance> 0
                .
                .
                .
        <Stream> 7
        <NumMixes> 2
        <Mixture> 1 0.5
            <Mean> 1 0.0
            <Variance> 1 1.0
        <Mixture> 2 0.5
            <Mean> 0
            <Variance> 0
    <TransP> 7                                        → Definition of the transition matrix
        0.000e+0 1.000e+0 0.000e+0 0.000e+0 0.000e+0 0.000e+0 0.000e+0
        0.000e+0 6.000e-1 4.000e-1 0.000e+0 0.000e+0 0.000e+0 0.000e+0
        0.000e+0 0.000e+0 6.000e-1 4.000e-1 0.000e+0 0.000e+0 0.000e+0
        0.000e+0 0.000e+0 0.000e+0 6.000e-1 4.000e-1 0.000e+0 0.000e+0
        0.000e+0 0.000e+0 0.000e+0 0.000e+0 6.000e-1 4.000e-1 0.000e+0
        0.000e+0 0.000e+0 0.000e+0 0.000e+0 0.000e+0 6.000e-1 4.000e-1
        0.000e+0 0.000e+0 0.000e+0 0.000e+0 0.000e+0 0.000e+0 0.000e+0
<EndHMM>
```

The parameters of the phone HMMs must then be initialized from the utterance database they are intended to model. The initialization can be achieved in two different ways. If the location of the phone boundaries is known for some utterances, these utterances are used as bootstrap data and the initialization is performed with the HTK tools HInit. When no labeled training data is available, a flat start is performed by the tool HCompV. So, in comparison with HInit, HCompV has the advantage of not requiring any information concerning the phone boundaries, but on the other hand it performs a far less precise initialization of the model parameters.

If bootstrap data are available, every phone HMM is initialized individually, by using all the segments of speech corresponding to the phone HMM that is being initialized. Initializa-

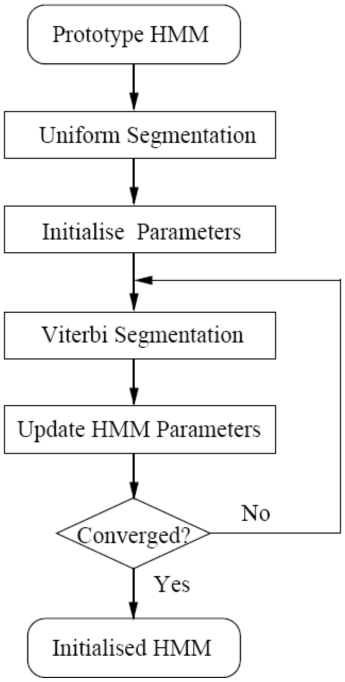


Figure 5.3: HInit operation, source [Young02]

tion is performed in several cycles. On the first cycle, HInit realizes a uniform segmentation of the training data, which associates successive speech segments with successive HMM states. Computation of means and variances is then possible. On the second and successive cycles, uniform segmentation is replaced by Viterbi alignment. HMM parameters are then estimated, as in the first cycle. Mathematical details of the Viterbi algorithm are given in appendix B. The whole estimation process is reiterated until convergence of the log likelihood of the training data, computed after each Viterbi alignment. This is summarized on Figure 5.3.

In the case of a flat start training, the global mean and variance of the speech training data is assigned to every Gaussian distribution in every phone HMM, as represented on Figure 5.4. It is important to note that the first cycle of embedded re-estimation will be carried out by uniformly segmenting each training utterance.

Once the HMM parameters have been initialized, they are refined using Baum-Welch re-estimation, which has been implemented in the tool HRest. Phone HMMs are trained individually, by using the forward-backward algorithm to find the probability of being in each state at each time frame and by computing then weighted averages for the HMM parameters, as depicted on Figure 5.5. The mathematical description of forward-backward and Baum-Welch algorithms is provided in appendix B.

The next step consists in creating context-dependent triphone HMMs from the set of

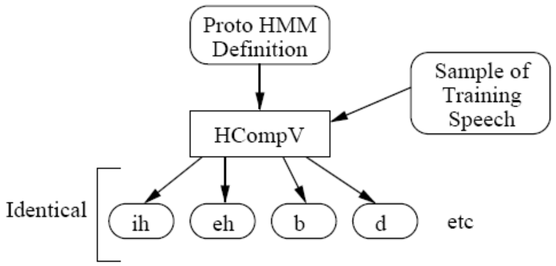


Figure 5.4: HCompV operation, source [Young02]

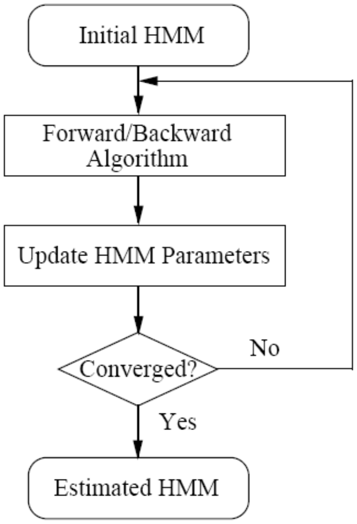


Figure 5.5: HRest operation, source [Young02]

trained monophone HMMs. This involves the tool HERest to perform embedded training. HERest uses continuously spoken utterances as training data to run a single Baum-Welch re-estimation of the whole set of phone HMMs. More precisely, HERest starts by loading all the HMM definitions. HERest needs a transcription, that is a list of the phones for every utterance of the database. Note that phone boundary information is not required. Therefore this list of phones can be automatically generated from the orthographic transcription of the utterances, by using a pronunciation dictionary. For every utterance, HERest constructs a composite HMM by concatenating the phone HMMs listed in the utterance transcription, and the forward-backward algorithm is used to accumulate the statistics of state occupation, means, variances, etc. Once all the training utterances have been processed, the accumulated statistics are used to compute re-estimates of the parameters of all of the phone HMMs. The Baum-Welch algorithm is recalled in appendix B Training of phone HMMs is summarized on Figure 5.6.

The training part ends with the tree-based clustering of the distributions of spectrum and

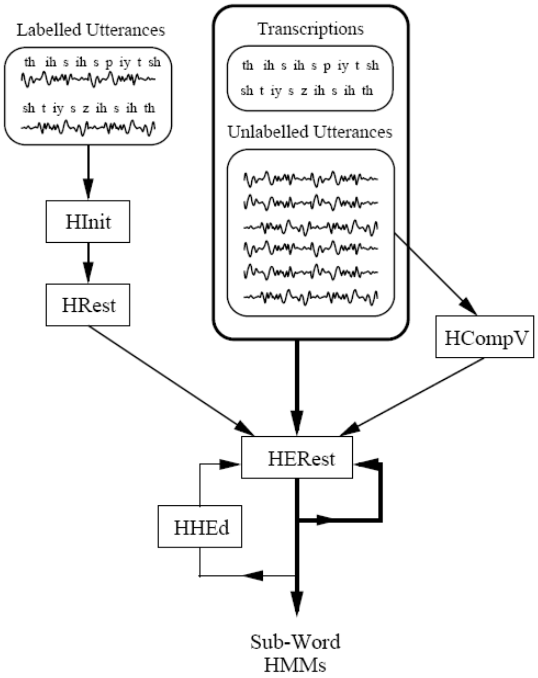


Figure 5.6: Training phone HMMs, source [Young02]

fundamental frequency, using the HTK tool HHed, enriched with functionalities provided by the HTS toolkit, the re-estimation of the parameters of the modified set of context-dependent models, using HERest, and the clustering of the distribution of state durations.

5.2 Synthesis

As in section 3.2, synthesis starts with the concatenation of the context-dependent HMMs corresponding to the labels sequence provided by Festival, given the text to synthesize. The state durations of the composite HMM are then determined, and using these durations, the HNM parameters are obtained by maximum likelihood estimation, as explained in section 3.3.

Several operations must then be performed on these parameters to meet the requirements of the HNM synthesis module:

1. HTS provides parameters computed every $5ms$, while the HNM synthesis programme requires pitch-synchronous parameters. So the parameters must be resampled according to the pitch contour.

2. Linear predictive cepstral coefficients, $c(n)$, must be converted back to linear prediction

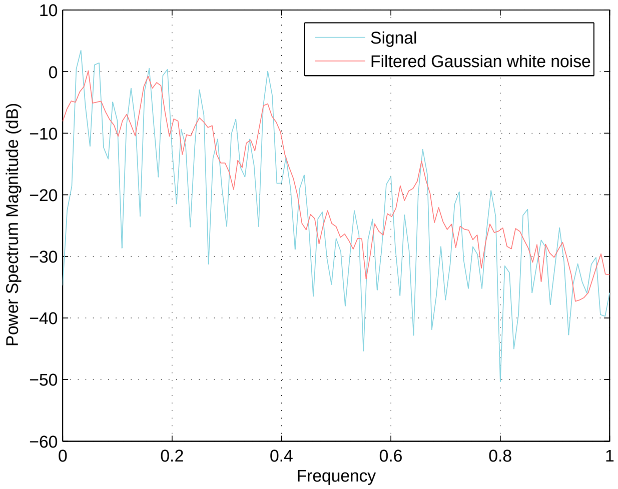


Figure 5.7: Signification of the linear prediction analysis

coefficients, $a(n)$, according to:

$$\sigma = \exp\{c(0)\} \tag{5.3}$$

$$a(n) = -c(n) - \sum_{k=1}^{n-1} \frac{k}{n} c(k) a(n-k) \qquad 1 \leq n \leq P \tag{5.4}$$

where P is the order of linear prediction coefficients.

3. The harmonic amplitudes are computed from the linear prediction coefficients and σ. Indeed a spectral envelope can be calculated from the linear prediction coefficients and the harmonic amplitudes are the values of this envelope at the multiples of the pitch.

   For more precise explanations, we must go back to the original derivations of the linear prediction coefficients. As explained in section 2.3, a linear prediction analysis of order p applied to a fragment of speech, y, provides its linear prediction coefficients, A_i, and the standard deviation of the prediction error, σ. These parameters permit to define an all-pole filter, whose numerator is σ and whose denominator is a polynomial whose coefficients are the linear prediction coefficients:

$$\mathrm{h(z)} = \frac{\sigma}{\sum_{i=0}^{p} A_i z^{-i}} \tag{5.5}$$

   where the first coefficient is given and equal to 1: $A_0 = 1$.

   The basic idea behind linear prediction analysis is that the power spectral density of the Gaussian white noise filtered by the all-pole filter defined by equation 5.5 is a least squares approximation of the power spectral density of the original signal, y. These two power spectral densities are drawn on Figure 5.7.

The power spectral density of a signal consisting of N samples, $x(n)$, is estimated by computing its periodogram defined by:

$$\frac{1}{N}|X_N(\phi)|^2 \tag{5.6}$$

where:

- $X_N(\phi)$ is the discrete time Fourier transform (DTFT) of $x(n)$;
- $\phi = 2\pi\frac{f}{f_s}$, with f representing the frequency, and f_s, the sampling frequency.

As the DTFT of a sinusoid comprising N samples and whose amplitude is A has a maximum of $\frac{N \cdot A}{2}$, the k^{th} harmonic amplitude of the signal y can be computed from its linear prediction coefficients by:

$$a_k = \frac{2}{\sqrt{N}} \frac{\sigma}{\sum_{i=0}^{p} A_i e^{-j\frac{i \cdot 2\pi \cdot k \cdot f_0}{f_s}}} \qquad (k = 1, \ldots, K) \tag{5.7}$$

where:

- f_0 is the fundamental frequency of the speech segment;
- K is the number of harmonics.

4. For the HNM synthesis the harmonic phases are assumed to be equal to zero. This supposition has been made after having carried out an HNM analysis on several speech samples, replaced the harmonic phase values obtained with the analysis by zero, and performed an HNM synthesis, based on the assumption of zero phase. Comparing the quality of the original and synthesized speech waveforms, this assumption can be considered as being reasonable.

Once these operations are performed, speech waveform can be synthesized using the HNM synthesis module.

As explained previously, three different syntheses have been carried out:

- In the first synthesis, the HNM parameters are extracted from the utterance database every $5ms$. The extracted parameters include the linear prediction coefficients and the fundamental frequency. The extraction is followed by the conversion from LPC coefficients to LPCC coefficients. Then the LPCC coefficients, the fundamental frequency and their dynamic features are input to the HTS system, which performs the training of the context-dependent models and outputs the HMM parameters generated by maximum likelihood estimation. The HMM parameters, i.e. the LPCC coefficients and the fundamental frequency, are then used to compute pitch-synchronous HNM parameters (as explained above) from which the HNM synthesis can be carried out.

- In the second synthesis, the same parameters are extracted pitch-synchronously using the HNM analysis module. Therefore the conversion from LPC to LPCC coefficients s followed by a resampling, as HTS requires parameters computed every $5ms$. The rest of the synthesis process is carried out like in the first case.

- In the third synthesis, the extraction is also pitch-synchronous but, besides the LPC coefficients and the fundamental frequency, the HNM analysis outputs the maximum voiced frequency. This is followed by the conversion from LPC to LPCC coefficients, the resampling, the training of context-dependent models and the generation of HMM parameters, which comprise the LPCC coefficients, the fundamental frequency and the maximum voiced frequency. They are converted to pitch-synchronous HNM parameters and the HNM synthesis is carried out.

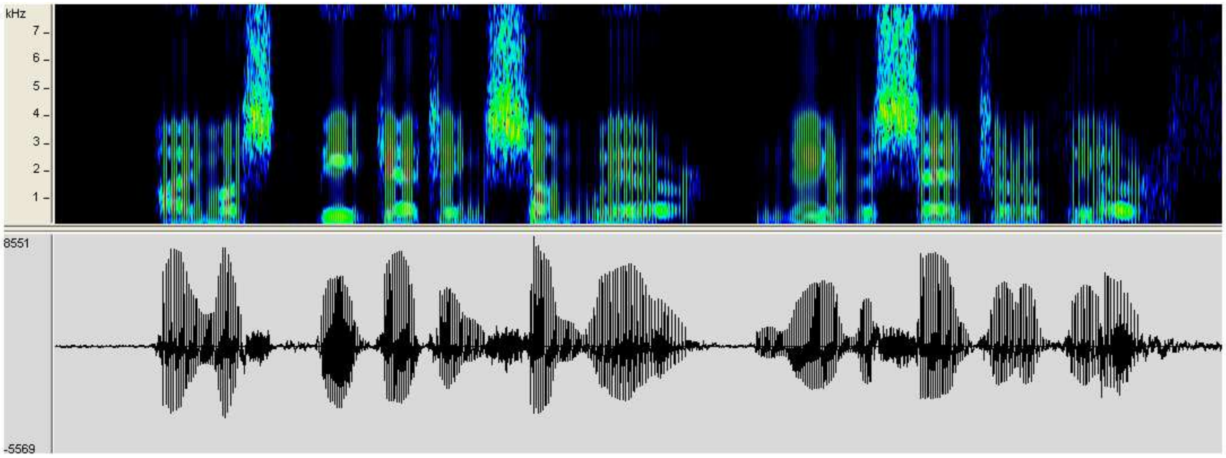


Figure 5.8: Waveform and spectrogram of the sentence "I must be getting somewhere near the center of the earth." synthesized with the first approach

Figures 5.8, 5.9, 5.10 show the speech waveforms and spectrograms obtained by synthesizing the sentence "I must be getting somewhere near the center of the earth." with the three approaches. It seems that a slightly better segmental quality is obtained in the third case. Future work would consist of performing listening tests, such as mean opinion score (MOS) test.

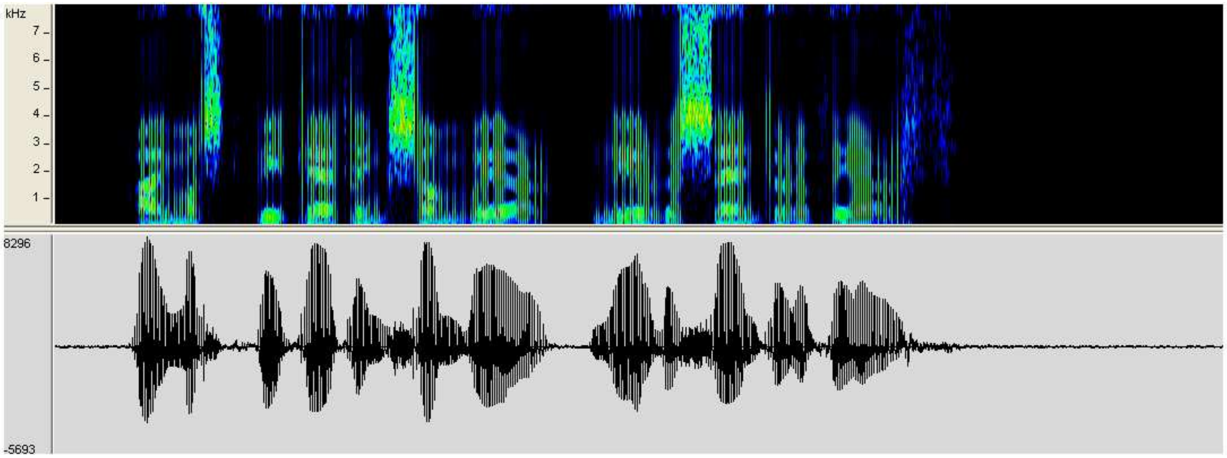


Figure 5.9: Waveform and spectrogram of the sentence "I must be getting somewhere near the center of the earth." synthesized with the second approach

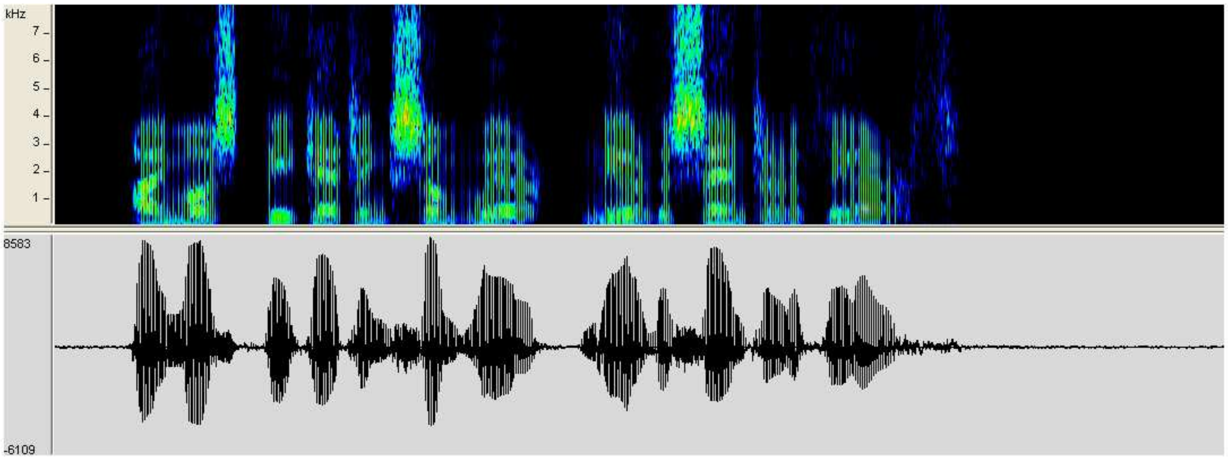


Figure 5.10: Waveform and spectrogram of the sentence "I must be getting somewhere near the center of the earth." synthesized with the third approach

Chapter 6

Conclusions and future work

6.1 Conclusion

The objective of this thesis was the integration of the Harmonic plus Noise Model (HNM) into the Hidden Markov Model-based speech synthesis system (HTS), in order to improve the speech waveform quality provided by the HTS system. This integration has been successful.

The text-to-speech system developed in this thesis has the advantages of the HTS system that is, essentially, the possibility to synthesize a new voice in a relatively small time, by using a relatively small amount of training data, and it overcomes its main drawback, i.e. the fairly average quality of the synthesized speech waveform, by substituting the source filter modeling approach for the Harmonic plus Noise Model. The possibility of creating a new voice with reduced development time and cost is a real advantage over the state-of-the-art synthesis technique, i.e. unit selection synthesis. Moreover, as opposed to unit selection techniques, model-based approaches permit to synthesize unseen target contexts thanks to tree-based clustering methods and allow easy modifications of the voice characteristics. The interest of using the HNM model as a parametric modeling of speech results from its advantages over other techniques. The HNM model represents speech as being composed of harmonic parts and noise parts. This decomposition leads to a higher quality of speech waveform than with a linear prediction approach, as it ensures the simultaneous conservation of low and high frequency energies. Moreover the HNM model represents the excitation and the vocal tract in a unified framework, as opposed to LPC.

The integration of the HNM model into the HTS system starts by extracting the HNM parameters from the training database. Like in HTS, this is followed by the training of context-dependent HMMs, the concatenation of HMMs and the estimation of the most likely HMM parameters. The final step consists in converting HMM parameters into HNM parameters and in synthesizing speech waveform, using the HNM synthesis. The HNM parameters used for synthesis have been computed making an assumption of zero phase, which seems to be reasonable.

The TTS system has been implemented in three different ways: 1) The HMM parameters include the LPCC coefficients and the fundamental frequency and they're extracted at a fixed rate; 2) The same parameters are extracted pitch-synchronously; 3) The HMM parameters include the LPCC coefficients, the fundamental frequency, and the maximum voiced frequency and they're extracted pitch-synchronously. In conclusion, a slightly better speech waveform quality seems to be obtained in the third case.

6.2 Future work

As explained in the preceding section, an assumption of zero phase has been made for the modeling of the harmonic phases. However, other approaches [Achan03, Chazan00, Shao03] could be investigated. There also remain avenues of research concerning the parametrization of the harmonic amplitudes, which have been derived from the linear prediction coefficients in this work. I hope that these open questions will constitute a starting point towards an improved speech waveform quality in the framework of model-based synthesis.

Appendix A

Open source software tools used in this work

A list of open source software tools used in this work:

- **HTK Ver 3.3** A set of tools for building and manipulating Hidden Markov Models provided by the Cambridge University Engineering Department (by permission of Microsoft). The tools enable isolated and embedded training of model parameters, decision tree clustering and Viterbi decoding.
- **HTS Ver 1.1.1** A HMM-based speech synthesis system developed by the Nagoya Institute of Technology.
- **Festival Speech Synthesis System Ver 2.0** A set of tools for building speech synthesis systems provided by the University of Edinburgh and Carneggie Mellon University including a full TTS system.
- **SPTK Ver 3.0** Speech signal processing toolkit from Nagoya Institute of Technology.
- **Wavesurfer Ver 1.8.5** Tool for waveform visualization and manipulation developed by the Centre for Speech Technology at KTH in Stockholm, Sweden.

Appendix B

Mathematical formulation of Hidden Markov Models

This chapter is devoted to the mathematical description of Continuous Density Hidden Markov Model (CDHMM). In the following developments the Hidden Markov Models are assumed to comprise N states. These HMMs are expected to describe a sequence of n-dimensional observation vectors, $O = [o_1, o_2, \ldots, o_T]$, o_t representing the observation vector at time t.

A Hidden Markov Model is described by three sets of parameters, $\lambda = (A, B, \pi)$:

1. Transition matrix A, whose $(i, j)^{th}$ element, a_{ij}, is the probability of transitioning from state S_i to state S_j, defined by

$$a_{ij} = P(q_{t+1} = S_j | q_t = S_i) \qquad (1 \leq i \leq n; 1 \leq j \leq n) \tag{B.1}$$

2. Matrix $B = \{b_j(o_t)\}$ contains the set of observation likelihoods, each representing the probability of an observation o_t being generated from a state S_j; if the observation vectors are modeled with a Gaussian Mixture Model (GMM), observation likelihoods are defined by

$$b_j(o_t) = \sum_{m=1}^{M} c_{jm} \mathcal{N}(o_t; \mu_{jm}, \Sigma_{jm}) \tag{B.2}$$

where:

- q_t represents the state occupied at time t;
- M is the number of mixture components;
- c_{jm}, is the weight of the m^{th} mixture component of the j^{th} state;

- $\mathcal{N}(.;\mu,\Sigma)$ is a multi-variate Gaussian with mean vector, μ and covariance matrix, Σ, defined by

$$\mathcal{N}(o_t;\mu_{jm},\Sigma_{jm}) = \frac{1}{\sqrt{(2\pi)^n|\Sigma_{jm}|}} e^{-\frac{1}{2}(o_t-\mu_{jm})^T\Sigma_{jm}^{-1}(o_t-\mu_{jm}))}$$

3. Vector $\pi = \{\pi_j = P(q_1 = S_j|\lambda)\}$, whose i^{th} element is the probability of the model initially being in state q_i.

There are three basic problems concerning Hidden Markov Models [Dines03, Gosselin00, Rabiner89]:

1. The estimation of the likelihood of an observation sequence given a model.
2. The determination of the best sequence of model states given a sequence of observations and the parameters of a model.
3. The estimation of the model parameters that best model the observed data, also called the training of the model.

The algorithms used to solve these problems are detailed in the three following sections.

B.1 Computation of the likelihood of an observation sequence given a model

The likelihood of the observation sequence O is obtained by summing, for every state sequence Q, allowed by the model λ, the joint likelihoods of the observation and the state sequences, given the model, that is

$$\begin{aligned} p(O|\lambda) &= \sum_{\text{all Q}} p(O,Q|\lambda) && \text{(B.3)} \\ &= \sum_{\text{all Q}} p(O|Q,\lambda)P(Q|\lambda) && \text{(B.4)} \end{aligned}$$

As the observations are supposed to be independent,

$$\begin{aligned} p(O|Q,\lambda) &= \prod_{t=1}^{T} p(o_t|q_t,\lambda) \\ &= b_{q_1}(o_1)\cdot b_{q_2}(o_2)\ldots b_{q_T}(o_T) && \text{(B.5)} \end{aligned}$$

and the probability of the state sequence, $Q = [q_1, q_2, \ldots, q_T], 1 \leq q_t \leq N$, is given by

$$P(Q|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \ldots a_{q_{T-1} q_T} \tag{B.6}$$

Then taking equations (B.5) and (B.6) into account equation (B.4) becomes

$$p(O|\lambda) = \sum_{q_1, q_2, \ldots, q_T} \pi_{q_1} b_{q_1}(o_1) a_{q_1 q_2} b_{q_2}(o_2) \ldots a_{q_{T-1} q_T} b_{q_T}(o_T) \tag{B.7}$$

In practice, this calculation is infeasible (of order $2T.N^T$) even for small numbers of observations, T, and of states, N. Nevertheless it is possible to compute $p(O|\lambda)$ recursively by the Forward-Backward algorithm. The forward and backward variables are respectively defined by equations (B.8) and (B.9) and calculated using equations (B.10) and (B.11).

$$\alpha_i(t) = p(o_1, \ldots, o_t, q_t = S_i|\lambda) \tag{B.8}$$

$$\beta_i(t) = p(o_{t+1}, \ldots, o_T|q_t = S_i, \lambda) \tag{B.9}$$

Forward recursion:

1. Initialization

$$\alpha_i(1) = \pi_i b_i(o_1) \qquad (1 \leq i \leq N) \tag{B.10a}$$

2. Recursion

$$\alpha_j(t+1) = \left[\sum_{i=1}^{N} \alpha_i(t) a_{ij}\right] b_j(o_{t+1}) \qquad (1 \leq t \leq T-1; 1 \leq j \leq N) \tag{B.10b}$$

3. Termination

$$p(O|\lambda) = \sum_{i=1}^{N} \alpha_i(T) \tag{B.10c}$$

Similarly, for the backward recursion:

1. Initialization

$$\beta_i(T) = 1 \qquad (1 \leq i \leq N) \tag{B.11a}$$

2. Recursion

$$\beta_i(t) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_j(t+1) \qquad (1 \leq t \leq T-1; 1 \leq j \leq N) \tag{B.11b}$$

3. Termination

$$p(O|\lambda) = \sum_{i=1}^{N} \pi_i b_i(o_1)\beta_i(1) \tag{B.11c}$$

The likelihood of the observation sequence O is finally obtained by combining equations (B.10) and (B.11):

$$p(O, q_t = S_i|\lambda) = \alpha_i(t)\beta_i(t) \tag{B.12}$$

$$p(O|\lambda) = \sum_{i=1}^{N} \alpha_i(t)\beta_i(t) \qquad (1 \leq t \leq T) \tag{B.13}$$

B.2 Estimation of the optimal state sequence

The algorithm used to determine the optimal state sequence is similar to the forward procedure. While the latter computes the likelihood of the observation sequence along all the possible paths, the former only calculates the likelihood along the best path, that is the state sequence maximizing the likelihood of the observation sequence:

$$\hat{p}(O|\lambda) = \max_{\text{all Q}} \pi_{q_1} b_{q_1}(o_1) a_{q_1 q_2} b_{q_2}(o_2) \dots a_{q_{T-1} q_T} b_{q_T}(o_T) \tag{B.14}$$

The algorithm uses two variables:

- $\delta_t(i)$: the maximum likelihood score along a single path among all the paths ending in state S_i at time t, defined by

$$\delta_t(i) = \max_{q_1, q_2, \dots, q_{t-1}} p(q_1, q_2, \dots, q_t = S_i, o_1, o_2, \dots, o_T|\lambda) \tag{B.15}$$

- $\psi_t(i)$: the state with the highest likelihood at time t, defined by

$$\psi_t(i) = \arg\max_{q_1, q_2, \dots, q_{t-1}} p(q_1, q_2, \dots, q_t = S_i, o_1, o_2, \dots, o_T|\lambda) \tag{B.16}$$

and is described by the following equations:

1. Initialization:

$$\delta_1(i) = \pi_i b_i(O_1) \qquad (1 \leq i \leq N) \tag{B.17a}$$

$$\psi_1(i) = S_0 \tag{B.17b}$$

where state S_0 is the non-emitting entry node

2. Recursion:

$$\delta_{t+1}(j) = \max_{1\leq i\leq N} [\delta_t(i)a_{ij}]\, b_j(o_{t+1}) \qquad (2 \leq t \leq T; 1 \leq j \leq N) \tag{B.17c}$$

$$\psi_{t+1}(j) = \arg\max_{1\leq i\leq N} [\delta_t(i)a_{ij}] \qquad (2 \leq t \leq T; 1 \leq j \leq N) \tag{B.17d}$$

$$\tag{B.17e}$$

3. Termination:

$$P^* = \max_{1\leq i\leq N} \delta_T(i) \tag{B.17f}$$

$$q_T^* = \arg\max_{1\leq i\leq N} \delta_T(i) \tag{B.17g}$$

4. Path backtracking: the best sequence of states is given by $Q^* = \{q_1^*, \ldots, q_T^*\}$, where

$$q_t^* = \psi_{t+1}(q_{t+1}^*) \qquad (t = T-1, T-2, \ldots, 1) \tag{B.17h}$$

B.3 Training

This third problem consists in finding the model parameters $\lambda = (A, B, \pi)$ that maximize the likelihood of the observation vector O, defined by $p(O|\lambda)$. There is no analytical solution to find the global maximum of the likelihood but there exists an iterative process which leads to a local maximum. The model parameters can be estimated by two techniques, based either on the Viterbi algorithm or the Baum-Welch algorithm, and described in the two following sections.

B.3.1 Viterbi estimation

The Viterbi estimation technique is an iterative approach. If there is no initial parameter estimates, the algorithm starts by uniformly segmenting the training data, that is by dividing up the training segments equally amongst the model's N states[1]. The segmented data is then used to estimate the parameters of each state's output distribution, $b_j(.)$. A second segmentation is performed by using the Viterbi algorithm and the the parameters are re-estimated. The last two steps of segmentation and re-estimation are repeated until convergence. The Viterbi estimation technique is described by equations (B.18):

1. Initialisation

$$[q_1, \ldots, q_{T/N} = S_1], [q_{1+T/N}, \ldots, q_{2\cdot T/N} = S_2], \ldots, [q_{1+[N-1]T/N}, \ldots, q_{N\cdot T/N} = S_N]$$

[1] Other approaches exist for HMMs whose topology is not left-right.

2. Parameter estimation

$$\hat{\mu}_{jm} = \frac{\sum_{t=1}^{T} \psi_{jm}(t) o_t}{\sum_{t=1}^{T} \psi_{jm}(t)} \tag{B.18a}$$

$$\hat{\Sigma}_{jm} = \frac{\sum_{t=1}^{T} \psi_{jm}(t)(o_t - \mu_{jm})(o_t - \mu_{jm})^T}{\sum_{t=1}^{T} \psi_{jm}(t)} \tag{B.18b}$$

$$\hat{c}_{jm} = \frac{\sum_{t=1}^{T} \psi_{jm}(t)}{\sum_{t=1}^{T} \sum_{l=1}^{M} \psi_{jl}(t)} \tag{B.18c}$$

where:

- $\hat{\mu}$ and $\hat{\Sigma}$ are estimates of the model parameters μ and Σ;
- ψ_{jm} is 1 if o_t is associated with the mixture component m of state j of the model λ and zero otherwise.

3. Viterbi segmentation

$$\delta_{t+1}(j) = \left[\max_i \delta_t(i) a_{ij} \right] \cdot b_j(o_{t+1}) \qquad (a \leq t \leq T-1) \tag{B.18d}$$

4. Termination

$$\max_{1 \leq i \leq N} \hat{\delta}_T(i) - \max_{1 \leq i \leq n} \delta_T(i) \leq \tau \tag{B.18e}$$

where τ is some predetermined convergence value.

The transition probabilities are calculated from the Viterbi segmentation of the final iteration of the algorithm:

$$\hat{a}_{ij} = \frac{A_{ij}}{\sum_{k=1}^{N} A_{ik}} \tag{B.19}$$

where A_{ij} is the total number of transitions from state i to state j.

B.3.2 Baum-Welch Estimation

The Viterbi estimation technique makes use of the variable ψ to define which state each training vector is associated with. However there are no hard boundaries between phones in real speech. In this context, the Baum-Welch estimation technique is considered to be better than the Viterbi technique because it takes a soft decision, by replacing the variable ψ by a variable L which represents the probability of an observation vector being associated with any Gaussian mixture component. The Baum-Welch technique has other advantages over the Viterbi techniques: the estimation of the model parameters is more robust as the parameters of each state are estimated using all of the available training data; moreover, in opposition to the Viterbi technique, the Baum-Welch technique enables the embedded training of model parameters, described in section 5.1.

Before applying the Baum-Welch algorithm, the model parameters must be initialized. As described in section 5.1, these initial estimates can be computed in two ways. If bootstrap data is available, the estimation is performed in two steps: 1) The training data is uniformly segmented and the model parameters are computed; 2) The training data is segmented again, but using the Viterbi algorithm, and the model parameters are re-estimated; these last two steps are repeated until convergence. The other approach for initialization consists of a flat start training, in which case the global mean and variance of the speech training data is assigned to every Gaussian distribution in every phone HMM.

The Baum-Welch algorithm makes use of two variables, $p_{ij}(t)$ and $\gamma_i(t)$, whose definition uses the forward and backward likelihoods:

$$\begin{aligned} p_{ij}(t) &= p(q_t = S_i, q_{t+1} = S_j | O, \lambda) \\ &= \frac{p(q_t = S_i, q_{t+1} = S_j, O|\lambda)}{p(O|\lambda)} \\ &= \frac{\alpha_i(t) a_{ij} b_i(t) \beta_j(t+1)}{\sum_{m=1}^{N} \sum_{n=1}^{N} \alpha_m(t) a_{mn} b_m(t) \beta_n(t+1)} \end{aligned} \tag{B.20}$$

$$\gamma_i(t) = \sum_{j=1}^{N} p_{ij}(t) \tag{B.21}$$

and the parameters of the new model $\hat{\lambda} = (\hat{A}, \hat{B}, \hat{\pi})$ are computed by using the following equations:

$$\begin{aligned}
\hat{\pi}_i &= E\{q_1 = i|\lambda\} \\
&= \gamma_i(1) && \text{(B.22a)} \\
\hat{a}_i j &= \frac{E\{q_t = i, q_{t+1} = j|O, \lambda\}}{E\{q_t = i|O, \lambda\}} \\
&= \frac{\sum_{t=1}^{T} p_{ij}(t)}{\sum_{t=1}^{T} \gamma_i(t)} && \text{(B.22b)} \\
\hat{\mu}_{jm} &= E\{o_t|q_t = j, k_t = m\} \\
&= \frac{\sum_{t=1}^{T} p(q_t = j, k_t = m|O, \lambda) o_t}{\sum_{t=1}^{T} p(q_t = j, k_t = m|O, \lambda)} \\
&= \frac{\sum_{t=1}^{T} L_{jm}(t) o_t}{\sum_{t=1}^{T} L_{jm}(t)} && \text{(B.22c)} \\
\hat{\Sigma}_{jm} &= \frac{\sum_{t=1}^{T} L_{jm}(t)(o_t - \mu_{jm})(o_t - \mu_{jm})^T}{\sum_{t=1}^{T} L_{jm}(t)} && \text{(B.22d)} \\
\hat{c}_{jm} &= \frac{\sum_{t=1}^{T} L_{jm}(t)}{\frac{1}{P}\alpha(t)\beta(t)} && \text{(B.22e)}
\end{aligned}$$

where:

$$\begin{aligned}
L_{jm}(t) &= \frac{1}{P}\alpha_i(t-1) a_{ij} c_{jm} b_{jm}(o_t) \beta_j(t) \\
P &= p(O|\lambda) \\
&= \sum_{i=1}^{N} \alpha_i(t)
\end{aligned}$$

Bibliography

[Abrantes91] A.J. Abrantes, J.S. Marques and I.M. Transcoso, "Hybrid Sinusoidal Modeling of Speech without Voicing Decision", *Proceedings of Eurospeech 91*, Paris, pp. 231-234, 1991.

[Achan03] K. Achan, S.T. Roweis and B.J. Frey, "Probabilistic Inference of Speech Signals from Phaseless Spectrograms", *NIPS*, 2003.

[Allen79] J. Allen, S. Hunnicutt, R. Carlson and B. Granström, "MITalk-79: The 1979 MIT text-to-speech system", *Speech Communications Papers Presented at the 97th Meeting of the Acoustical Society of America*, Cambridge, USA, pp. 507-510, 1979.

[Allen87] J. Allen, M.S. Hunnicutt and D.H. Klatt, "From Text-to-Speech: The MITalk System", Cambridge University Press, Cambridge, 1987.

[Bahl91] L.R. Bahl, P.V. de Souza, P.S. Gopalakrishnan, D. Nahamoo and M.A. Picheny, "Context dependent modelling of phones in continuous speech using decision trees", *Proceeedings of the DARPA Speech and Natural Language Processing Workshop*, Pacific Grove, California, pp. 264-270, 1991.

[Beutnagel99] M. Beutnagel, A. Conkie, J. Schroeter, Y. Stylianou and A. Sydral, "The AT&T Next-Gen TTS system", *Proceeding of the Joint Meeting of ASA, EAA and DAGA*, Berling, Germany, pp. 18-24, 1999.

[Black97] A.W. Black and P. Taylor, "Automatically clustering similar units for unit selection in speech synthesis" *Proceedings of the European Conference on Speech Communication and Technology*, vol. 2, Rhodes, Greecs, pp. 601-604, 1997.

[Black99] A. W. Black, P. Taylor and R. Caley, "The Festival Speech Synthesis System", Centre for Speech Technology Research, University of Edinburgh, England, 1.4 ed., June 1999.

[Boite00] R. Boite, H. Bourlard, T. Dutoit, J. Hancq and H. Leich, "Traitement de la parole", Presses Polytechniques et Universitaires Romandes, 2000.

[Campbell96] N. Campbell, "CHATR: A High-Definition Speech ReSequencing System", *Proceedings of the 3rd ASA/ASJ Joint Meeting*, pp. 1223-1228, 1996.

[Charpentier86] F.J. Charpentier and M.G. Stella, "Diphone synthesis using an overlap-add technique for speech waveforms concatenation", *Proc. IEEE Int. Conf. Acoust., Speech Signal Processing*, Tokyo, Japan, 1986.

[Chazan00] D. Chazan, R. Hoory, G. Cohen and M. Zibulski, "Speech Reconstruction from Mel Frequency Cepstral Coefficients and Pitch Frequency", *Proc. ICASSP-2000*, 2000.

[Coorman00] G. Coorman, J. Fackrell, P. Rutten and B. Van Coile, "Segment selection in the L&H Realspeak laboratory TTS system", *Proc. of ICSLP*, Beijing, China, 2000.

[d'Alessandro98] C. d'Alessandro, B. Yegnanarayana and V. Darsinos, "Effectiveness of a periodic and Aperiodic Decomposition Method for Analysis of Voice Sources", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 6, n°1, pp. 12-23, 1998.

[Dines03] J. Dines, "Model based trainable speech synthesis and its applications", Ph.D. Thesis, Queensland University of Technology, Brisbane, Australia, 2003.

[Donovan95] R.E. Donovan and P.C. Woodland, "Automatic speech synthesis parameter estimation using HMMs", *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Detroit, USA, pp. 640-643, 1995.

[Donovan98] R.E. Donovan and E.M. Eide, "The IBM Trainable Speech Synthesis System", *Proceedings ICSLP'98*, Sydney, Australia 1998.

[Donovan01] R.E. Donovan, A. Ittycheriah, M. Franz, B. Ramabhadran, E. Eide, M. Viswanathan, R. Bakis, W. Hamza, M. Picheny, P. Gleason, T. Rutherfoord, P. Cox, D. Green, E. Janke, S. Revelin, C. Waast, B. Zeller, C. Guenther and J. Kunzmann "Current Status of the IBM Trainable Speech Synthesis System", *Proceedings 4th ESCA Tutorial and Research Workshop on Speech Synthesis*, Atholl Palace Hotel, Scotland, UK, 2001.

[Dutoit93] Th. Dutoit and H. Leich, "MBR-PSOLA : Text-to-speech synthesis based on an MBE re-synthesis of the segments database", *Speech Commun.*, vol. 13, no. 34, pp. 167-184, November 1993.

[Dutoit02] Th. Dutoit, "Introduction au traitement de la parole", Faculté Polytechnique de Mons, 2002.

[Gosselin00] B. Gosselin, "Classification et Reconnaissance Statistique de Formes", Faculté Polytechnique de Mons, 2000.

[Griffin87] D.W. Griffin, "Multi-Band Excitation Vocoder", Ph.D. Thesis, MIT, Cambridge, 1987.

[Hamon89] C. Hamon, E. Moulines and F.J. Charpentier, "A diphone synthesis system based on time-domain prosodic manipulations of speech", *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, p. 238, 1989.

[Hunt89] M. Hunt, D. Zwierynski D. and R. Carr R, "Issues in high quality LPC analysis and synthesis", *Eurospeech89*, Paris, France, vol. 2, pp. 348-351, 1989.

[Ifeachor96] E. Ifeachor and B. Jervis, *Digital Signal Processing: A Practical Approach*, Addison-Wesley, 1996.

[Klatt82] D.H. Klatt, "The Klattalk text-to-speech conversion system", *Proceedings on the International Conference on Acoustic, Speech and Signal Processing*, Paris, pp. 1589-1592, 1982.

[Klatt87] D.H. Klatt, "Review of text-to-speech conversion for English", *Journal of the Acoustical Society of America*, vol. 82, pp. 737-793, September 1987.

[Klatt90] D.H. Klatt, "DecTalk user's manual", Digital Equipment Corporation Report, 1990.

[Lawson74] C.L. Lawson and R.J. Hanson. "Solving Least-Squares Problems", Prentice Hall, Englewood Cliffs, New Jersey, 1974.

[Makhoul75] J. Makhoul, "Spectral Linear Prediction: Properties and Applications", *IEEE Trans. ASSP*, vol. 23, no. 5, pp. 283-296, 1975.

[Masuko96] T. Masuko, K. Tokuda, T. Kobayashi and S. Imai, "Speech synthesis using HMMs with dynamic features", *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, 1996.

[Mobius01] B. Mobius, "Rare events and closed domains: Two delicate concepts in speech synthesis", *Proceedings of the 4th ESCA Workshop on Speech Synthesis*, Perthshire, Scotland, 2001.

[Narayanan04] S. Narayanan and A. Alwan, "Text-to-Speech Synthesis: New Paradigms and Advances", Prentice Hall, 2004.

[Odell95] J.J. Odell, "The Use of Context in Large Vocabulary Speech Recognition", Ph.D. Thesis, University of Cambridge, England, 1995.

[Plumpe98] M. Plumpe, A. Acero, H. Hon and X. Huang, "HMM-based smoothing for concatenative speech synthesis", *Proceedings of the International Conference on Spoken Language Processing*, Sydney, Australia, pp. 2751-2754, December 1998.

[Quatieri01] T.F. Quatieri, "Discrete-time speech signal processing", Prentice Hall, 2001.

[Rabiner89] L.R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition", *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257-286, 1989.

[Rabiner94] L.R. Rabiner, "Applications of Voice Processing to Telecommunications", *Proc. IEEE*, vol. 82, pp. 199-228, February 1994.

[Shao03] X. Shao and B. Milner, "Clean Speech Reconstruction from Noisy Mel Frequency Cepstral Coefficients using a Sinusoidal Model" *Proc. ICASSP-2003*, 2003.

[Stylianou98] Y. Stylianou, "Removing phase mismatches in concatenative speech synthesis", *Proc. 3rd ESCA Speech Synthesis Workshop*, pp. 267-272, November 1998.

[Stylianou96] Y. Stylianou, "Harmonic plus noise models for speech, combined with statistical methods, for speech and speaker modification", Ph.D. Thesis, Ecole Nationale Supérieure des Télécommunications, Paris, France, January 1996.

[Stylianou01] Y. Stylianou, "Applying the Harmonic Plus Noise Model in Concatenative Speech Synthesis", *IEEE Trans. Speech Audio Processing*, vol. 9, no. 1, pp. 21-29, January 2001.

[Tokuda95-1] K. Tokuda, T. Kobayashi, and S. Imai, "Speech parameter generation from HMM using dynamic features", *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Detroit, USA, pp. 660-663, 1995.

[Tokuda95-2] K. Tokuda, T. Masuko, T. Yamada, T. Kobayashi, and S. Imai, "An algorithm for speech parameter generation from continuous mixture HMMs with dynamic features", *Proceedings of the European Conference on Speech Communication and Technology*, Madrid, Spain, pp. 757-760, 1995.

[Tokuda00] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech parameter generation algorithms for HMM-based speech synthesis", *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Istanbul, Turkey, 2000.

[Tokuda02-1] K. Tokuda, T. Masuko, N. Miyazaki and T. Kobayashi, "Multi-space probability distribution HMM". *IEICE Trans. Information and Systems*, vol.E85-D, no.3, pp. 455-464, March 2002.

[Tokuda02-2] K. Tokuda, H. Zen and A. W. Black, "An HMM-based speech synthesis system applied to English". *Proc. IEEE Workshop on Speech Synthesis*, pp. 227-230, 2002.

[Van Dromme05] D. Van Dromme, "Développement d'une synthèse vocale harmoniques plus bruit", Travail de fin d'études, IDIAP, Martigny, Switzerland, 2005.

[Huang96] X. Huang, A. Acero, J. Adcock, H. Hon, J. Goldsmith and J. Liu, "Whistler: A Trainable Text-to-Speech System", *Proceedings of the International Conference on Spoken Language Processing*, Philadelphia, PA, 1996.

[Yoshimura99] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi and T. Kitamura, "Simultaneous modelling of spectrum, pitch and duration in HMM-based speech synthesis", *Proc. of European Conference on Speech Communication and Technology*, vol. 5, pp. 2347-2350, September 1999.

[Yoshimura01] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi and T. Kitamura. "Mixed Excitation for HMM-based Speech Synthesis". *Proc. of European Conference on Speech Communication and Technology*, Aalborg, Denmark, vol. 3, pp. 2259-2262, September 3-7, 2001.

[Young94] S.J. Young, J.J. Odell and P.C. Woodland, "Tree-based state tying for high accuracy acoustic modelling" *Proceeedings of the ARPA Human Language Technology Workshop*, Plainsboro, USA, pp. 307-312, March 1994.

[Young02] S. Young, G. Evermann, T. Hain, D. Kershaw, G. Moore, J.J. Odell, D. Ollason, D. Povey, V. Valtchev and P. Woodland, "The HTK Book", Cambridge University, England, December 2002. For HTK version 3.2.1.

[SPTK] `http://kt-lab.ics.nitech.ac.jp/~tokuda/SPTK/index.html`

[Festvox] `http://festvox.org/cmu_arctic/dbs_awb.html`