FAKULTÄT FÜR !NFORMATIK

Interpolation of Austrian German and Viennese Dialect/Sociolect in HMM-based Speech Synthesis

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Computational Intelligence

ausgeführt von

Dietmar Schabus, Bakk.techn.
Matrikelnummer 0147322

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung:

Betreuer:	Univ.Prof. Dipl.-Inf. Dr.rer.nat. Jens Knoop
Mitwirkung:	Dipl.-Ing. Dr.techn. Markus Kommenda,
	Mag.phil. Dr.techn. Michael Pucher

Wien, 29.04.2009

(Unterschrift Verfasser) (Unterschrift Betreuer)

Dietmar Schabus
1020 Wien, Untere Augartenstraße 38/16

„Hiermit erkläre ich, dass ich diese Arbeit selbstständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.“

Wien, 29.04.2009

Abstract

In contrast to widely used waveform concatenation methods, the presented approach to speech synthesis relies on a parametric analysis–re-synthesis technique, where the features extracted in the analysis stage are modeled by hidden Markov models (HMMs). Many important improvements in the last decade have helped this approach to reach impressive performance. Additionally, its inherent flexibility makes it suitable for advanced speech synthesis tasks, like speaker adaptation, speaker interpolation, emotional speech, etc.

In this work, a flexible multi-dialect HMM-based speech synthesis system for Austrian German and Viennese dialect/sociolect is presented. A novel contribution is the interpolation of dialects, where we have to deal with phonological processes that change the segmental structure of the utterance. Evaluation results show that listeners do perceive both continuous and categorical changes of varieties.

Kurzfassung

Im Unterschied zu weit verbreiteten Methoden der Sprachsignal-Verkettung basiert der hier vorgestellte Ansatz zur Sprachsynthese auf einer parametrischen Analyse–Resynthese-Technik, wobei die in der Analyse-Phase extrahierten Parameter mit Hidden Markov Modellen (HMMs) modelliert werden. Zahlreiche Verbesserungen dieses Ansatzes im letzten Jahrzehnt ermöglichen beeindruckende Ergebnisse. Darüberhinaus erlaubt es dessen inhärente Flexibilität, diesen Ansatz für weitergehende Aufgabenstellungen der Sprachsynthese einzusetzen, wie zum Beispiel Sprecher-Adaptierung, Sprecher-Interpolation, emotionale Sprache, etc.

Diese Arbeit stellt ein HMM-basiertes, flexibles multi-dialektales Sprachsynthesesystem für österreichisches Deutsch und Wiener Dialekt/Soziolekt vor. Neu eingebracht wird dabei die Interpolation von Dialekten unter Berücksichtigung von phonologischen Prozessen, die die segmentale Struktur der Äußerung verändern. Die Ergebnisse einer Evaluation zeigen, dass sowohl kontinuierliche als auch kategorische Veränderungen in der Varietät von den Hörern wahrgenommen werden.

Contents

Chapter 1

Introduction

The main means of human communication is speech. To enhance human-machine interaction, computers should be capable of speech communication. Among other things, this requires computers to be able to produce an acoustic speech signal for a given input text, i.e., pass information to a human user through “speaking”. The scientific field of speech synthesis deals with the development of text-to-speech (TTS) systems that satisfy this requirement.

Besides the attempt to make human-machine communication more natural, speech synthesis has applications wherever visual output has disadvantages. For example, acoustic output is suitable for a user steering a vehicle or aircraft (or other activities involving eyes and hands), it can be used over the well-established telephone network (for information, reservation or ordering services), and it easily attracts our attention (for public announcements, alarm systems, etc.). Furthermore, speech synthesis has long been a vital assistive technology for people with visual impairment or other reading difficulties (e.g., screen readers), as well as for people with speech impairment (e.g., voice output communication aids).

The most important measures of quality for speech synthesis systems are intelligibility and naturalness. Clearly, a system that produces unintelligible speech completely misses the target. Naturalness describes how closely the synthetic output resembles human speech. Usually, it is a goal to optimize both.

Many systems following quite different approaches have been developed, with an increasing tendency towards data-driven or corpus-based approaches over knowledge-based ones, due to the advances in the power and resources of computer technology. The most widely used technique relies on concatenation of segments of recorded speech which are selected from a large database. While this approach is quite successful in producing high-quality speech, its flexibility is limited: The resulting speech signal is a concatenation of segments taken from the recordings, therefore additional recording material is necessary to diversify the system’s output – e.g. in terms of speaker or voice characteristics, prosody, emotion, or dialect. However, the acquisition and annotation of large amounts of high-quality speech recordings is a time-consuming and costly task.

In the last decade, a statistical parametric approach based on hidden Markov models has gained increasing popularity. Although this method is generally considered not to reach the same signal quality as good concatenative systems, it was shown to produce acceptable, intelligible and natural speech. Its main strengths, however, derive from its inherent flexibility: Since speech is modeled in a *parametric* way, we can attempt to diversify the output by modifying the parameters. Techniques based on adaptation and interpolation have been developed and were shown to successfully produce speech with characteristics inexistent in the original speech corpus. Other advantages of this method in comparison with the concatenative approach include a greater robustness and a greatly reduced footprint regarding storage and computation requirements.

1.1 Original Contributions

This work describes speech synthesis based on hidden Markov models and its application to Austrian German and Viennese dialect/sociolect. Full-fledged TTS systems were built for different varieties using Austrian German resources (lexica, recordings) originally developed for a concatenative system [NPK08, KPM$^+$09], together with the HMM-framework from the "HTS-2007/2008" system [YZW$^+$08, YNZ$^+$09]. Several training strategies were pursued and evaluated in a user study with 40 test listeners.

Additionally, a new technique to interpolate between Austrian standard German and Viennese dialect was devised. The difference to existing interpolation techniques [YMT$^+$97, YTM$^+$00, TYMK04, TYMK05] is that we have to deal with changes in the segmental structure of the utterance. For example, the Austrian standard phrase *"mit viel Gewicht"* and the corresponding Viennese *"mid fü Gwicht"*[1] differ in the number of phonemes (the smallest linguistically distinctive unit of sound).

The interpolation technique was also evaluated in listening tests, which showed that in-between variants can be produced successfully, i.e., listeners perceived a continuous change from standard to dialect.

Although the building of the system and the corresponding publication [PSYN09] were a group effort, the realization of the dialect interpolation (i.e., the modification of the synthesis frontend, see Section 6.5) as well as the design, carrying out and analysis of the evaluations (see Sections 6.4 and 6.6) can be fully credited to the author.

1.2 Outline

This thesis is organized as follows. In Chapter 2 "What Is Speech?" we get a grasp on speech in general as well as on the technicalities required to bring speech to the digital world of computers. Chapter 3 "Speech Synthesis" gives an overview over different approaches to the task of speech synthesis. Chapter 4 "HMM-based Speech Synthesis in Depth" covers the approach used in this work in detail. In Chapter 5 "Austrian German and Viennese TTS", relevant differences between German German, Austrian German and Viennese dialect/sociolect are identified. The practical undertakings of this work are described in Chapter 6 "HMM-based Synthesis of Austrian German and Viennese Dialect", giving details about training strategies and the evaluation thereof, and about the dialect interpolation technique and its evaluation. Finally, in Chapter 7 "Conclusion" we summarize what has been described and achieved in this work and give an outlook to possible future work in this field.

[1] There is no defined orthography for Viennese dialect, the example shall merely illustrate the pronunciation. The phrase means "with much weight" in English.

Chapter 2

What Is Speech?

In making computers speak we are trying to imitate something that is used by humans every day as their primary means of communication. Therefore we will look at how speech is produced and perceived by humans. We will also look at how speech signals can be handled by computers, and at some other knowledge that is needed to bring these two views of speech together.

2.1 Production and Perception of Speech in Humans

2.1.1 The Human Vocal Apparatus

Figure 2.1 shows a diagram representing a sagital section through the human vocal apparatus. Its physiology and functionality are described in great detail by Flanagan [Fla65]. For our purposes, a more condensed discussion, as found in later works [RJ93, PK08], will suffice. The main parts of the vocal apparatus are the lungs, the trachea, the larynx with the vocal folds (or vocal cords), the soft palate (or velum), the tongue, the teeth and the lips. Also of importance are the pharyngeal, oral and nasal cavities.

In regard to the speech production process, the vocal apparatus can be divided into two components:

Sound Production While speaking, air is pushed from the lungs by muscle force through the bronchi and trachea. When the vocal folds are tensed, the air flow causes them to vibrate and periodically disrupt the flow of air, thus producing the so-called voiced speech sounds[1]. When the vocal folds are relaxed, in order to produce a sound, the air flow either must pass through a constriction in the vocal tract and thereby become turbulent, producing the so-called fricatives, or it can build up pressure behind a point of total closure within the vocal tract, and when the closure is opened, the pressure is suddenly and abruptly released, causing brief transient sounds called plosives.

Tone Shaping The acoustic signals from the vocal folds and from the constrictions are modified in tone by the pharynx, mouth and nasal cavity, which act as a filter. The motion of the so-called articulators (tongue, lips, jaws and velum) modifies the resonance frequency of this acoustic filter. The signal originating from the vocal folds is thus shaped into quite different sounds. The sounds resulting from air turbulences are also shaped in tone by the vocal tract, here the location of the constriction plays a major role.

2.1.2 Articulation of Speech Sounds

This section describes the relation between the vocal apparatus described in the previous section, and the elements of speech, the phonemes. These are commonly separated into two major groups:

[1]The oscillation frequency averages for male speakers around 120 Hz, for female speakers around 220 Hz.

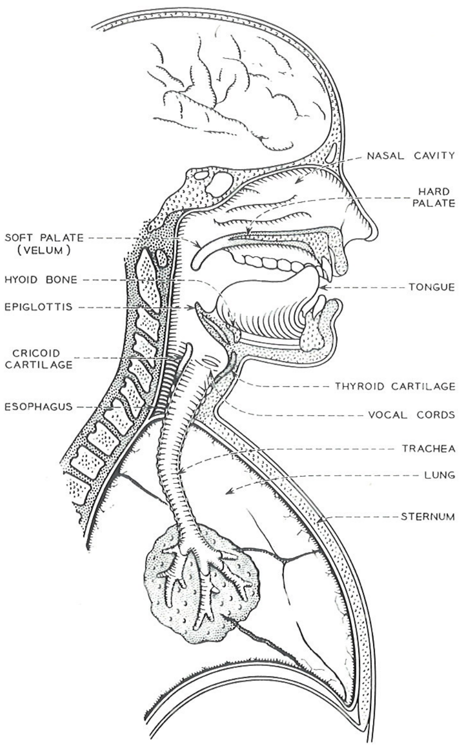


Figure 2.1: Schematic view of the human vocal apparatus (copy from [Fla65]).

vowels and consonants. They are discussed in the following using the example of the German language [PK08, RJ93].

Vowels

The resulting speech sound is called a vowel, when the vocal folds oscillate and the air is exhaled without constrictions in the vocal tract through the mouth (or the mouth and nose in the case of nasalized sounds). The positions of the articulators must not cause any too narrow points in the vocal tract. The tone of the sound depends mainly on the position of the dorsum linguae (the back of the tongue) and the lips.

The main distinction according to the position of the lips is that into rounded and unrounded vowels:

- unrounded: [iː ɪ eː ɛ aː a ə ɐ]
- rounded: [oː ɔ øː œ uː ʊ yː ʏ][2]

The relation of the position of the dorsum linguae and the resulting sound is illustrated in Figure 2.2.

Consonants

When some part of the vocal tract is narrowed such that the flow of air must pass through a constriction, or when there is a temporary total closure, the resulting sound is considered a consonant. For some phonemes, the vocal folds oscillate, for others they do not. According to this, they are called voiced or unvoiced consonants. Furthermore, the consonants can be categorized according to how

[2]We will use the symbols from the International Phonetic Alphabet to denote phones/phonemes throughout the text. See Appendix A for some explaining examples.

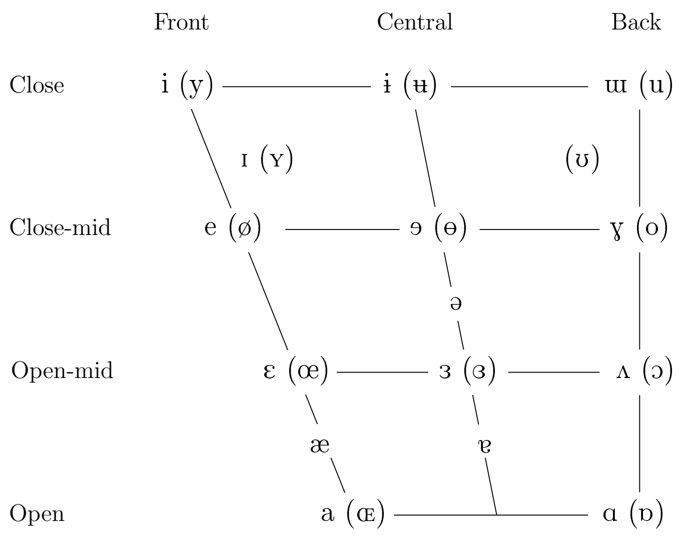


Figure 2.2: Vowel Diagram after [Ass99]. Vowels in parentheses are rounded.

and where in the vocal tract they are articulated. This gives rise to a table as the one depicted in Figure 2.3.

	bilabial		labiodental		dental		alveolar		palatal		velar		uvular		glottal	
	v.	uv.	v.	uv.	v.	uv.	v.	uv.	v.	uv.	v.	uv.	v.	uv.	v.	uv.
Plosives	b	p			d	t					g	k				ʔ
Nasals	m				n						ŋ					
Trills													ʀ			
Fricatives			v	f	z	s	ʒ	ʃ		ç		χ	ʁ			h
Laterals					l											
Approximants									j							
Affricates				pf		ts		tʃ				ks				

Figure 2.3: Consonants of Austrian German after Muhr [Muh07].

2.1.3 Human Perception

When it comes to the receiving part of human communication, i.e., hearing, authors of speech processing books take very different approaches. Flanagan [Fla65] again describes the physiology and functionality of the human ear in great detail. Lawrence and Rabiner [RJ93] touch on topics from the realm of the ear, hearing and perception at different points in their book, where it suits the respective topic being discussed in the context. Pfister and Kaufmann [PK08] explicitly state that they consider the anatomy of the human ear to be of minor interest for speech processing, what they do discuss are some relevant phenomena observed in psychoacoustic experiments. These shall be summarized here.

Perception of Intensity

Sound pressure is a physical quantity and thus measureable. Its values are usually given as sound pressure levels in decibels (dB re 20 μPa). Loudness, on the other hand, is a subjective perception that cannot be measured directly. To relate loudness and sound pressure, experiments on test subjects are required. Such experiments have shown, for example, that the perceived loudness depends heavily on frequency. This is depicted in Figure 2.4, which shows curves of equal perceived loudness (phon) plotted on a plane of the physical quantities frequency (kHz) and sound pressure level (dB). For example, for low frequencies, a much higher sound pressure level is required for a person to "hear" 20 phon than at higher frequencies. Our hearing is most sensitive to sound pressure between 2 and 5 kHz.

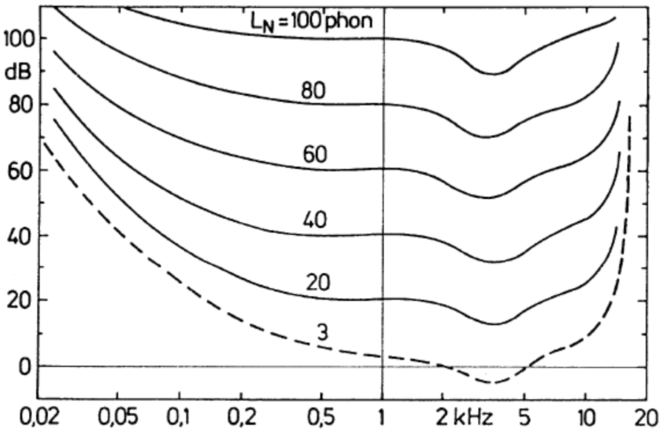


Figure 2.4: Curves of equal loudness, copy from [PK08].

Periodicity and Pitch

The human ear can hear sine waves in the frequency range from around 20 Hz to around 18 kHz as pitch. However, like most natural sources of sound, the human vocal apparatus does not produce pure sine waves. From the oscillating vocal folds as a source, a periodic signal is produced in the vocal tract, which humans perceive as having a distinct pitch, which is called the fundamental frequency F_0. Signals for which no pitch is perceived are noise-like or random in nature, as with the phones [s] and [ʃ], where the vocal folds are relaxed.

As for intensity, psychophysical studies have shown that human perception of the frequency content of sounds (both pure sine waves and speech signals) also does not follow a linear scale. This has lead to the perceptual scale of pitch called the "mel" scale [RJ93]. Frequency on the Hertz and mel scales are logarithmically related, as illustrated in Figure 2.5

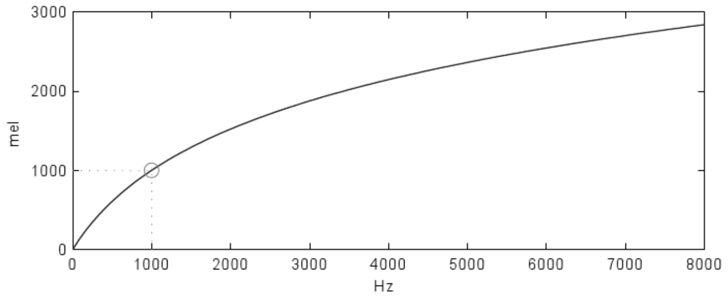


Figure 2.5: Relation of frequency on the Hertz and mel scales, copy from [PK08].

Phase Perception

Human hearing is insensible to small phase shifts, larger shifts are perceived as reverberations or even echoes. Phase plays an important role in binaural hearing, where the time difference between the ears is used to determine the direction of the sound source.

Auditory Masking

Auditory masking occurs when a more powerful sound prevents perception of a simultaneous less powerful sound. This effect depends mainly on the intensity and frequency of the masking signal.

Perception of Speech Sounds

In phonetics, speech sounds are categorized not only according to articulatory criteria, but also according to acoustic characteristics. Vowels, for instance, are commonly described using formants. The local maxima in the frequency spectrum of a sound signal are called its formants. They are described by the parameters formant frequency, bandwidth and amplitude. In the case of speech, formants result from resonances in the vocal tract, and since usually several of them occur they are denoted as F_1, F_2, F_3, etc., where lower index numbers indicate lower frequency.

The parameters of formants depend on the position of the articulators. As we have seen, the position of the articulators determines the spoken phoneme, thus there is a connection between formants and phonemes. This connection is illustrated in Figure 2.6 for the two lowest formants.

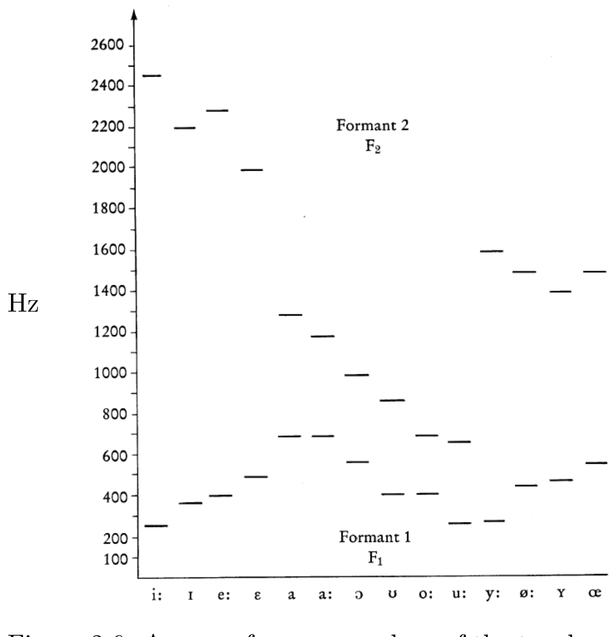


Figure 2.6: Average frequency values of the two lowest formants for German phonemes, copy from [PK08].

2.2 Representing Speech in Computers

Now that we have some understanding of how humans speak and perceive speech, we want to change to a more technical engineering perspective.

2.2.1 The Speech Signal in the Time and Frequency Domains

The speech signal is a slowly time varying signal in the sense that, when examined over a sufficiently short period of time (between 5 and 100 ms), its characteristics are fairly stationary. However, over long periods of time (on the order of 1/5 seconds or more) the signal characteristics change to reflect the different speech sounds being spoken [RJ93].

This is illustrated in Figure 2.7, which shows the time waveform corresponding to the spoken word "deshalb" (/ˈdɛshalp/, 'therefore') uttered by a male speaker. Following the graph on the horizontal time axis, we see that the utterance can be segmented into the following parts:

1. Initial silence
2. A short noise-like part, the transient sound of the plosive /d/
3. A quasi-periodic part corresponding to the vowel /ɛ/
4. A noise-like part, which is somewhat difficult to split into the two unvoiced phones /s/ and /h/
5. Another quasi-periodic part corresponding to the voiced sounds /a/ and /l/
6. A region of silence before the final plosive, in which air pressure is built up
7. A region corresponding to the transient sound resulting from the release of the pressure, the plosive /p/
8. Final silence

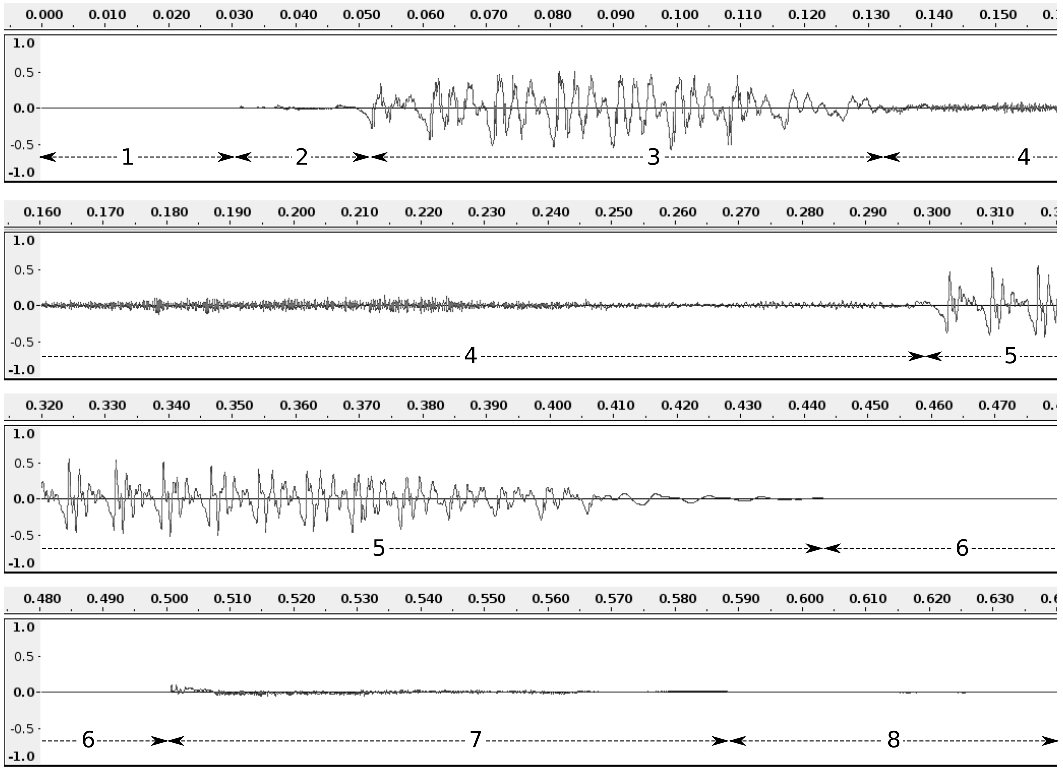
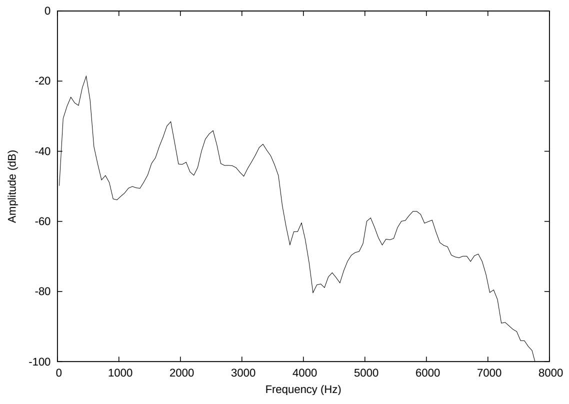


Figure 2.7: Waveform of the word /ˈdɛshalp/ uttered by a male speaker.

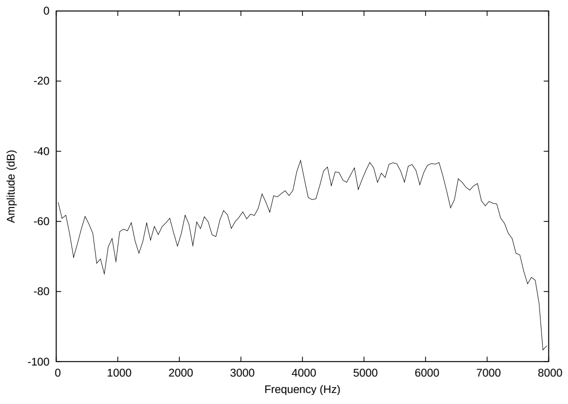
To compute and/or visualize the formants of a speech signal (at some point in time), we need to compute its frequency spectrum. This can be done using Fourier transformation on a short signal segment [Fla65, PK08]. The results are plots like the ones shown in Figure 2.8. Figure 2.8(a) shows the average spectrum of the part corresponding to the voiced /ɛ/ sound in the above example. Figure 2.8(b) shows the average spectrum of the part corresponding to the unvoiced /s/ sound.

In (a) we can see a distinctive peak around 500 Hz, and three somewhat less distinctive peaks around 1900, 2500 and 3300 Hz. These are the formants F_1 through F_4, and looking back at Figure 2.6 we find that these values are close to the ones given there for /ɛ/ ($F_1 \approx 500$ Hz, $F_2 \approx 2000$ Hz).

In (b) on the other hand, we cannot make out any distinctive peaks. If anything, we can say that higher frequencies (above 4 kHz) are a little stronger represented than lower frequencies.



(a)



(b)

Figure 2.8: Spectra of voiced (a) and unvoiced (b) speech signal segments.

Because there are such wide spectral differences at different points in time in speech signals, we wish to represent them as a function of *both* time and frequency. Such a three-dimensional representation is called a spectrogram. The values of a spectrogram are established by dividing the signal into short temporal sections and computing their spectrum using Fourier transformation. The most common form of graphical representation of a spectrogram is to have a two-dimensional plane with time on one axis and frequency on the other. Then the intensity or colour of each point in the plane represents the amplitude of that particular frequency at that particular point in time. Such a plot for the example utterance from above is shown in Figure 2.9.

In the voiced regions, the formants are clearly visible as horizontal dark bars, in the unvoiced regions the values appear to be random, with an emphasis on higher frequencies.

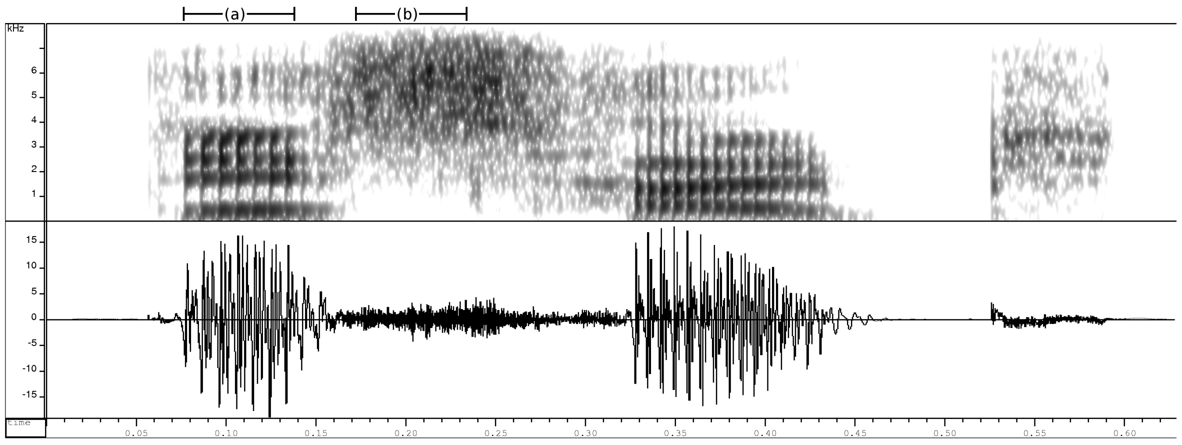


Figure 2.9: Waveform (bottom) and spectrogram (top) of the word /ˈdɛshalp/ uttered by a male speaker. The regions used to compute the spectra in Figure 2.8 are indicated at the top.

2.2.2 Digitizing the Speech Signal

When humans speak, they produce sound waves, which can be converted into a time-dependent electric signal by a microphone. The resulting signal is still analog (i.e., time- and value-continuous), and what has been said in Section 2.2.1 also holds for analog signals. However, our goal is to make computers produce speech and hence we need digital (i.e., time- and value-discrete) representations of originally analog signals.

To digitize an analog signal, it is first *sampled* at equidistant points in time and thus made time-discrete. Mathematically, sampling corresponds to multiplying the analog signal $x_a(t)$ with a sequence of Dirac pulses $s(t)$

$$x_s(t) = x_a(t) \cdot s(t) = x_a(t) \cdot \sum_{n=-\infty}^{\infty} \delta(t - nT_s). \tag{2.1}$$

where T_s denotes the sampling interval and $\delta(t)$ denotes the unity Dirac pulse:

$$\delta(t) = \begin{cases} 1 & \text{for } t = 0 \\ 0 & \text{otherwise.} \end{cases} \tag{2.2}$$

In the frequency domain, the multiplication in Equation 2.1 corresponds to a convolution with the Fourier transform of the pulse sequence:

$$X_s(\omega) = \frac{1}{2\pi} X_a(\omega) * S(\omega) = \frac{\omega_s}{2\pi} X_a(\omega) * \sum_{k=-\infty}^{\infty} \delta(\omega - k\omega_s). \tag{2.3}$$

I.e., the spectrum $X_s(\omega)$ results from a superposition of frequency-shifted $X_a(\omega)$ by integer multiples of $\omega_s = 2\pi/T_s$ (see Figure 2.10). To avoid aliasing, the sampling interval T_s needs to be shorter than $1/2B$, where B is the highest frequency occurring in the signal (Nyquist-Shannon sampling theorem) [PK08].

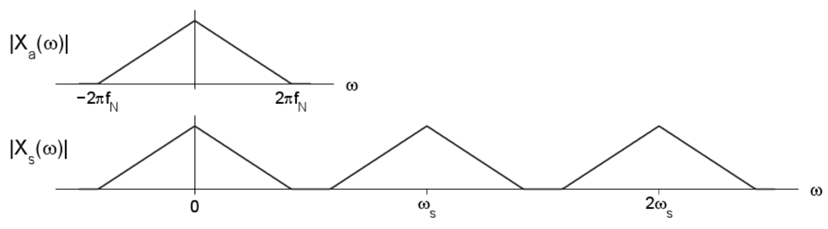


Figure 2.10: The bandlimited spectrum $X_a(\omega)$ of the analog signal $x_a(t)$ results in the unlimited spectrum $X_s(\omega)$ of the sampled signal $x_s(t)$ (copy from [PK08]).

The resulting signal is time-discrete and value-continuous. To discretize amplitude, the signal is then *quantized.* The (real) value of each sample needs to be represented in binary form. Using n bits, we obtain a codebook of 2^n so-called quantization levels. In the simplest case, they are chosen to be equidistant in a sufficiently wide range (uniform quantization). Each value is then represented by the nearest value from the codebook. To keep the introduced rounding error (called quantization noise) low, it is common to use 11 bits for speech representation, giving rise to 2048 quantization levels.

This process of discretizing a signal in both time and amplitude is called “pulse code modulation” (PCM) in the telecommunications jargon [PK08, Wit82].

2.2.3 Reconstructing the Analog Signal

To replay a signal that was digitized as described above, it needs to be converted back into a time- and value-continuous signal $x_a(t)$. However, the digital signal $x_d(t)$ is defined only at the discrete sampling points $t = nT_s$. A simple method common in practice is to obtain an analog signal by assuming the values to be constant over the sampling interval T_s, i.e., an analog signal $x_h(t)$ is computed as follows:

$$x_h(t) = x_d(t) * h_0(t) \tag{2.4}$$

where $h_0(t)$ denotes the zero-order hold function with

$$h_0(t) = \begin{cases} 1 & \text{for } -T_s/2 \le t \le +T_s/2 \\ 0 & \text{otherwise.} \end{cases} \tag{2.5}$$

The convolution of the sampled signal with $h_0(t)$ results in a step function in the time domain, as depicted in Figure 2.11. In the frequency domain, this corresponds to a multiplication of the respective spectra:

$$X_h(\omega) = X_d(\omega) \cdot H_0(\omega) = \frac{1}{2\pi}[X_a(\omega) * S(\omega)] \cdot H_0(\omega) \tag{2.6}$$

where $S(\omega)$ is defined as in Equation 2.3, and $X_a(\omega)$ denotes the spectrum of the desired analog signal $x_a(t)$. Under the assumption that no aliasing occurs (i.e., the Nyquist-Shannon sampling theorem was honored), we can reconstruct the analog signal from $x_h(t)$ using a low-pass filter (see Figure 2.12):

$$H_r(\omega) = \begin{cases} 1/H_0(\omega) & \text{for } -\pi/T_s < \omega < +\pi/Ts \\ 0 & \text{otherwise.} \end{cases} \tag{2.7}$$

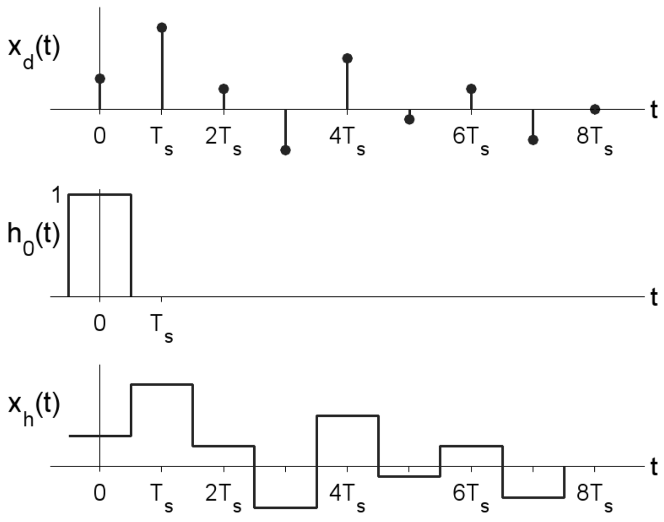


Figure 2.11: Convolving the sampled signal $x_d(t)$ with the zero-order hold function $h_0(t)$ results in the step function $x_h(t)$ (copy from [PK08]).

2.2.4 Symbolic Phonological Representation of Speech

Another way to represent speech in computers – one that is going to be of great importance later in this work – is one that is rather unrelated to the actual speech signal. A symbolic representation of speech is not aiming at describing the sound waves resulting from a person speaking, but at describing the conceptual aspect of speech that underlies the commands the speaking person is "sending" to the muscles involved in their vocal apparatus. Like the sound of music is described by musical score, we need a symbolic representation of speech that is independent of things like speaker, background noise, transmission channel, etc.

The first thing that comes to mind is to use text. However, there are several disadvantages in that approach. One problem is that in many languages (English in particular) a mapping from orthography

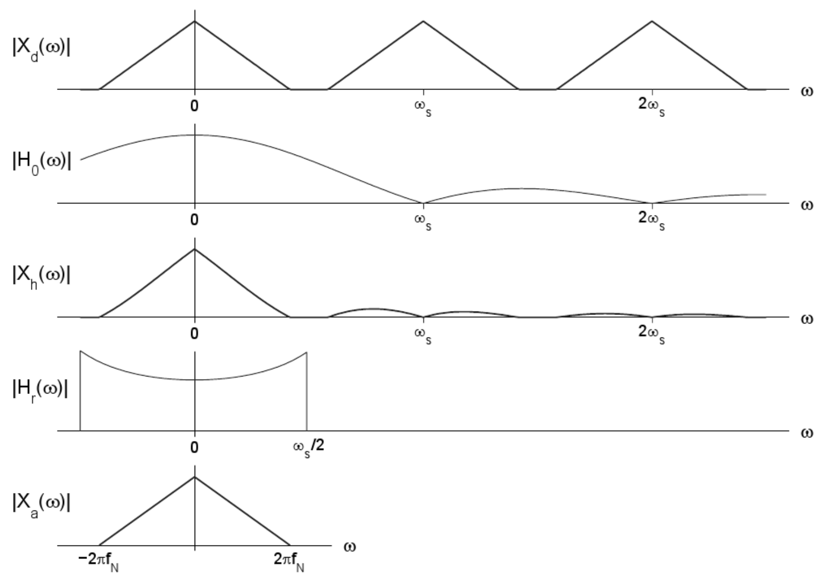


Figure 2.12: The analog spectrum $X_a(\omega)$ is reconstructed by multiplying the spectra of the sampled signal and the hold function ($X_d(\omega)$ and $H_0(\omega)$), and by then applying the low-pass filter $H_r(\omega)$ (copy from [PK08]).

to pronunciation of single words is already difficult.[3] The next natural step would be to use phonetic transcriptions using the IPA alphabet (see Appendix A), as we have already done for our example utterance (/ˈdɛshalp/). While this approach would clarify the pronunciation of each single word, it would for instance leave out things like rhythm, stress and intonation on the sentence level (these suprasegmental features of speech constitute what is called *prosody*). Furthermore, processing of exotic symbols like the IPA alphabet is unfortunately still somewhat cumbersome.

What we will use is a rich phonological representation containing a combination of orthographic transcription, phonetic transcription and several prosodic features, in a well-defined, machine-friendly format. Where appropriate, we will even include explicit durations of phones. An example of this representation is given in Appendix B, a text file in Festival utterance format [BTC99].

[3]Compare, for example, the pronunciation of the three English words heart, beard and heard.

Chapter 3

Speech Synthesis

Efforts of constructing a speaking machine date back as early as 1791, when Wolfgang von Kempelen presented his famous mechanical device (see Figure 3.1). Consisting mainly of a bellows, a reed and a leather "mouth", the machine could produce words and short utterances [Fla65].

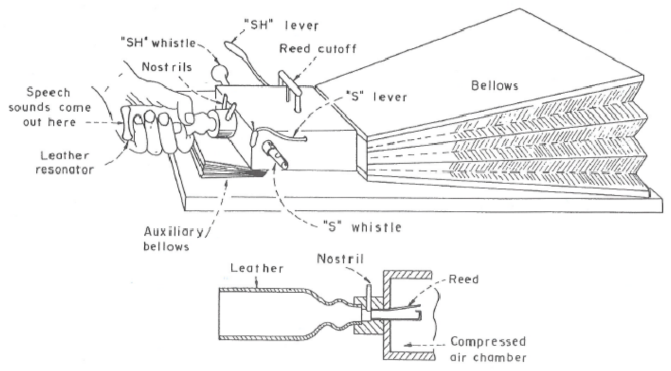


Figure 3.1: Wolfgang von Kempelen's speaking machine (copy from [Fla65]).

Centuries later, scientific interest in speech synthesis grew considerably with the development of electronics and even further with the upsurge of computer technology. Various approaches have been taken and are still developed, some of which will be discussed in this chapter. Before that, however, we want to clarify what it is that we are trying to achieve in speech synthesis.

3.1 Text-to-speech: The Task of Speech Synthesis

Using the methodology described in Section 2.2 (or – even simpler – an analog recording system), it is relatively unpretentious to record and replay spoken utterances. However, such an approach is subject to many restrictions: Everything that can be produced needs to be fixed and recorded beforehand, and it will be reproduced exactly in the way the original speaker spoke it during recording. A flexible speech synthesis system should be able to produce arbitrary utterances, potentially even containing previously unknown words (like proper names). Additionally, it would be desireable to have the possibility to diversify the produced speech in terms of speaker, speaking style, emotion, emphasis, etc.

In analogy to a person reading out a text, we would like a computer program to take written text as input and produce a speech signal as output. Such a program is called a text-to-speech (TTS) system, and from what has been said in Section 2.2.4 it soon becomes clear that the task of a TTS system can be described as consisting of two major parts: One that takes the text and computes a phonological representation, and one that takes these intermediate results and computes a speech signal from them. One major difference between the two is that the first part is independent of a voice, while the second is voice-dependent, as illustrated in Figure 3.2 [PK08].

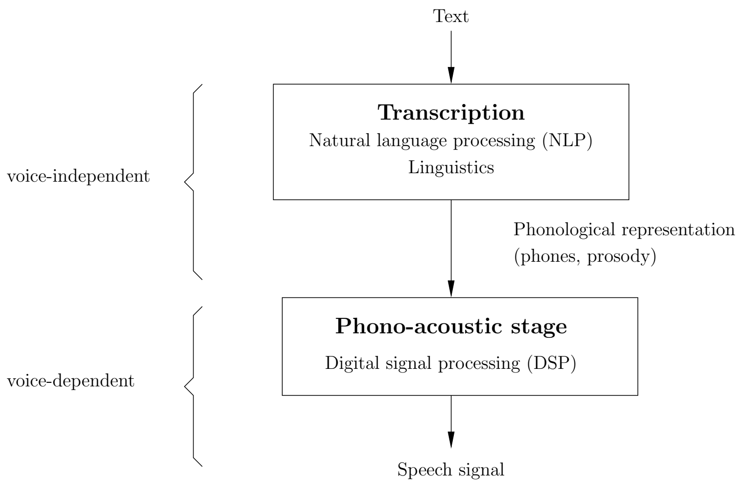


Figure 3.2: A TTS system can be divided into a voice-independent and a voice-dependent part (after [PK08]).

3.1.1 Transcription

One important task of the transcription stage is to determine the pronunciation of each word. In most cases, this can be done for each word individually. For some heterophonic homographs (words that are spelled the same way but pronounced differently), a syntactic analysis of the sentence can disambiguate pronunciation, e.g. for the two sentences "Please **wind** up this cable" and "The **wind** blows hard".[1] The two basic approaches to determine pronunciation are to use either a *lexicon* or *letter-to-sound rules*, or both [Dut97].

A lexicon is a collection of spelling–pronunciation pairs, for example the lexicon entry

```
("deshalb" ADV ((( d E s )1) (( h a l p )0)))
```

tells us that the word spelled "deshalb" is (as we already know) pronounced /ˈdɛshalp/, in a machine readable way: the phones of the word are given in a simple encoding (using `E` instead of ɛ in this example), syllables are parenthesized, the stress of the first syllable is indicated by a `1` as opposed to the `0` of the unstressed second syllable. Additionally, we are told that the word is an adverb by a so-called *part-of-speech tag* (`ADV` in the example). These tags indicate what part of speech, or lexical category, the entry belongs to (e.g. noun, verb, adjective, adverb, etc.). If syntactic analysis of the two sentences from above would determine that the word "wind" is used as a verb in the first sentence but as a noun in the second, the correct lexicon entries can be selected based on their part-of-speech tags.

As a pronunciation lexicon for a TTS system needs to contain an entry for every inflectional form of a word (e.g., plurals, conjugation, etc.), these lexica can become quite big already for a moderately inflected language like German. For example, the Austrian German/Viennese lexicon used in the system described later contains more than one million entries. This can be problematic if storage space is an issue, for instance in the realm of embedded systems and mobile devices.

In the other extreme case, a purely rule-based transcription system relies on a set of letter-to-sound (LTS) or grapheme-to-phoneme rules to transcribe the input text. This approach can result in greatly reduced storage requirements, but its success depends heavily on the language.

[1] An example for German would be "modern" which can mean both "fashionable" and "to rot", and is pronounced either /moˈdɛɐn/ or /ˈmoːdɐn/, respectively.

Most commonly, a combination of the two principles is used as a transcription strategy. There are different approaches possible [Dut97], for example:

- In a system with a very large dictionary containing inflectional forms, LTS rules are still useful as a fallback method, for example for proper names and other rare words. This also makes sure that the TTS system produces some speech output for any given input.
- To reduce lexicon size, many regular inflection forms can be covered by LTS rules, such that for most words, the lexicon needs to contain only the principal form.
- Similarly, in a mostly rule-based system, words with such a particular pronunciation that they constitute a rule of their own can be stored in a (small) lexicon of exceptions.
- In the development of LTS rules, a lexicon can serve as ground truth information, e.g. to evaluate a set of LTS rules, or as training data for automatic LTS generation through machine learning techniques.

Another task of the transcription module of a TTS system is prosody generation. The term prosody refers to certain properties of the speech signal such as audible changes in pitch (sentence melody or intonation), loudness and speech timing (mostly length and stress of syllables). Prosodic factors have a great impact on the perceived naturalness of a TTS system [Dut97]. We have said that we can determine pronunciation for each word individually in most cases. On the other hand, prosodic features of an utterance, like stressed words in a sentence, or grouping of a long sentence into phrases, can only be determined with information on the word interrelations. For instance, phrase borders within a sentence can be derived from the syntactic structure of the sentence [PK08]. Sometimes, generating correct prosody would require a semantic or even pragmatic understanding of the utterance in its context. As this poses a very difficult problem that is largely unsolved, TTS systems usually refrain of attempts in this domain and settle for morphological and syntactical analysis of the utterance.

The problems in transcription and different techniques to address them have been described in detail by Dutoit, including a detailed disquisition of prosody in TTS [Dut97].

3.1.2 Phono-acoustic Stage

As described, the output of a (successful) transcription is a still symbolic representation of the utterance to be synthesized, but one that is somewhat "closer" to the phonetic realization, i.e., the speech signal, than the original textual input. An example of such an intermediate result is given in Appendix B.

Next, we want to use these results as input to the phono-acoustic stage to compute a speech signal, thus entering the realm of digital signal processing. As mentioned earlier, this part of speech synthesis is voice-dependent, as the output is an actual speech signal. Some quite different approaches to this problem have emerged, which shall be briefly presented in the following sections. The road taken in this work, namely modelling speech parameters with HMMs, is discussed in detail in the following chapter.

3.2 Waveform Concatenation

A very common and successful method is to synthesize speech by concatenating segments of previously recorded speech material. One of the main advantages of this approach is that on the level of segments we obtain completely natural speech automatically (by design). As abrupt discontinuities are to be avoided, simply storing one segment for each phone would be a bad choice. Rather, the speech material to be used should contain all possible phone-to-phone *transitions* to account for effects like co-articulation. Thus, the smallest units that allow reasonable concatenation are so-called *diphones*, units that contain the transition from one phone to the next, with the cuts occurring in the middle of the two phones, where their signals tend to be more stationary.

The simplest form is thus diphone synthesis, where a database containing one instance of each possible diphone is used to build arbitrary utterances. Since a phone can sound quite differently depending on its position in the word and sentence in a natural speech signal, the diphones need to be prosodically modified before concatenation. However, this can have negative effects on the signal quality [PK08].

Unit Selection is a waveform concatenation technique that overcomes these problems. The idea here is not to have a collection of already cut-out speech segments with one representative for each diphone (or larger sub-word unit, like syllable, e.g.), but rather a (very) large collection of recorded utterances, providing a database of all units that occur in this corpus. For each unit, some phonetic and prosodic contextual features are computed and stored along with the unit itself.

To synthesize an utterance, units are selected from the database based on how well a candidate unit matches the required unit (*target costs*) as well as on how well two selected units "fit together" (*join costs*). The final synthetic utterance is built from the units that minimize the total cost.

Given a large enough speech corpus, unit selection systems can produce synthetic speech of very high quality. This technique has been very popular in the last decade, and is described in many works dealing with speech synthesis [Dut97, Fur01, PK08].

The main drawbacks of this approach are the large amount of high-quality data that is required and needs to be handled at runtime, and the inability to directly control the attributes of the synthetic speech, like prosody, emotion, speaking style, speaking rate, etc.

It should be clear that for speech synthesis tasks in restricted domains the waveform concatenation method can be applied very easily, e.g. for a talking clock the units to be concatenated can be whole words or even partial phrases.

3.3 Simulation of the Human Speech Production Mechanism

A totally different approach is taken in a family often called *articulatory synthesis* in the literature. Here, speech synthesis is realized by modeling the human vocal apparatus and simulating the speech production mechanism. The construction of the model relies on many measurements taken of a speaking person, for example by radiographic video or by placing sensors in the vocal tract of the speaker [Cok76, PK08].

Apart from helping to understand the human speech production mechanism, this approach is considered to be particularly effective in synthesizing transitional sounds such as consonants, since it can precisely simulate the dynamic manner of articulation in the vocal tract. Additionally, this method is considered to be easily related to the ponetic information conveyed by the speech wave. High-quality synthetic speech has not yet been obtained, however [Fur01].

3.4 Signal Formant Modeling

Instead of physiologically modelling the vocal apparatus, one can attempt to model the speech signal directly. As we have said in Chapter 2, an important characteristic of the speech signal are its formants. In what is often called *formant synthesis*, speech is synthesized following the division of the speech production process into sound production and tone shaping we have made in Section 2.1.1. Signal formant modeling systems can be divided into a *source* and a *filter* part. The source part is responsible for generating an excitation signal using either a pulse generator (for voiced sounds) or a random number generator (for unvoiced, noise-like sounds), or a mixture of the two. The pulse generator needs the fundamental frequency F_0 as input. The resulting signal is then passed on to a bank of filters, one for each formant. By varying the parameters (frequency, bandwidth and amplitude) of the filters, the F_0 input and the pulse/noise mixture over time, a speech signal is generated.

An important strength of this approach is that formant transitions can be explicitly specified, and thus naturally sounding phone transitions can be realized. Furthermore, formant systems have a small footprint in terms of storage and computational power. However, it takes enormous effort to provide all necessary parameters to allow acceptable quality speech synthesis [Dut97, Fur01, PK08].

3.5 Statistical Parametric Speech Synthesis

As unit selection, the statistical parametric approach is a data-driven one, exploiting the continuous increase in power and resources of computer technology to handle large amounts of speech data. This approach could be most simply described as generating the *average* of some sets of similarly sounding speech segments [ZTB09].

The general idea is to start with a large corpus of training data consisting of recorded speech samples and the corresponding phonological transcriptions; then extract representative features from the recordings by analyzing the waveforms, and use the feature–transcription pairs to train some generative probabilistic graphical models. Then, to synthesize speech, transcribe the input text, use the models to generate appropriate sound features for the transcription, and invert the waveform analysis to generate a speech signal from the features.

Although any generative model could be used, hidden Markov models (HMMs) have shown to be suitable and are widely used. HMMs can model sequences of speech spectra using well-defined algorithms, and have been applied with great success in speech recognition, i.e., to extract text from recorded speech, the inverse of speech synthesis. In a series of papers in the last decade, Tokuda et al. presented a new algorithm to generate speech parameters from HMMs using dynamic features, which enabled them to build an HMM-based synthesis system [FTKI92, TKI95, MTKI96]. Since then, many contributions have been made to improve HMM-based speech synthesis. It has been applied to many languages and various specialized tasks.

HMM-based speech synthesis is covered in detail in Chapter 4.

Chapter 4

HMM-based Speech Synthesis in Depth

In this chapter, we will try to get a thorough understanding of how HMM speech synthesis works. Not every concept used today in state-of-the-art systems can be covered in full detail, but the relevant sources are always given.

The chapter begins with some theoretical background on HMMs. Like *Kalman Filters* and *Particle Filters*, HMMs belong to a group of techniques using recursive Bayesian estimation, which in turn belong to the family of probabilistic graphical models, together with *Bayesian Networks* and their extension to *Dynamic Bayesian Networks* [DBM03].

4.1 Hidden Markov Models

In this section we will discuss the fundamentals of hidden Markov models (HMMs). The definitions and examples in this section are based on the classical tutorial by L. R. Rabiner [Rab89].

4.1.1 Definition

A hidden Markov model describes a doubly embedded stochastic process. At each point in time, the model is in one of N states $S_1, \dots, S_N$, and at regularly spaced (discrete) time intervals $t_1, t_2, \dots$, it undergoes a change of state (possibly back to the same state), according to the transition probabilities associated with the current state q_t. Which state the system is going to next is thus determined only by the current state, the history of preceding states is "forgotten" and the transitions are independent of time. This is called the *Markov property* and can be stated formally as

$$P(q_t = S_j \mid q_{t-1} = S_i,\ q_{t-2} = S_h, \cdots) = P(q_t = S_j \mid q_{t-1} = S_i) \tag{4.1}$$

The set of state transition probabilities is usually presented in the form of a matrix $A = \{a_{ij}\}$ with

$$a_{ij} = P(q_t = S_j \mid q_{t-1} = S_i) \quad \text{for} \quad 1 \le i, j \le N \tag{4.2}$$

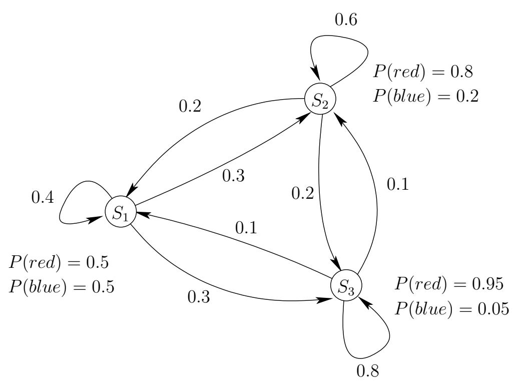
$$a_{ij} \ge 0 \quad \text{for} \quad 1 \le i, j \le N \tag{4.3}$$

$$\sum_{j=1}^{N} a_{ij} = 1 \quad \text{for} \quad 1 \le i \le N \tag{4.4}$$

The initial state probabilities are given by

$$\pi_i = P(q_1 = S_i) \quad \text{for} \quad 1 \le i \le N \tag{4.5}$$

So far, we have described the states of an HMM and the (stochastic) transitions among them. However, the *state sequence* $q_1, q_2, \dots$ is not directly observable; it is *hidden.* At each discrete point in time t,



$N = 3$

Set of states $= \{S_1, S_2, S_3\}$

$M = 2$

Set of observation symbols $= \{red, blue\}$

$$A = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.5 & 0.5 \\ 0.8 & 0.2 \\ 0.95 & 0.05 \end{bmatrix}$$

Figure 4.1: Example of a hidden Markov model. Graphical representation on the left, parameters on the right.

the current state q_t generates some output through a second stochastic process. For $t = 1, \ldots, T$, this gives rise to the *observation sequence* $O = (o_1, \ldots, o_T)$, where each observation o_t depends only on the current state q_t.

In the case of a finite alphabet of M discrete symbols $v_1, \ldots, v_M$ as possible observations, an HMM contains the observation symbol probability distribution $B = \{b_j(k)\}$, in which

$$b_j(k) = P(o_t = v_k \mid q_t = j) \quad \text{for} \quad 1 \le k \le M \tag{4.6}$$

defines the probability of emission for each symbol in state j, $1 \le j \le N$. Again, the general stochastic constraints

$$b_j(k) \ge 0 \quad \text{for} \quad \begin{matrix} 1 \le j \le N \\ 1 \le k \le M \end{matrix} \tag{4.7}$$

$$\sum_{k=1}^{M} b_j(k) = 1 \quad \text{for} \quad 1 \le i \le N \tag{4.8}$$

have to hold. An example HMM of this kind is depicted in Figure 4.1.1.

However, for some applications – among them speech recognition and synthesis – it is desirable to allow observations in the form of continuous feature vectors rather than discrete symbols from a fixed alphabet. Instead of listing the probability for each output symbol in state j, we then give a *probability density function* (pdf) that models the probability of the output $\vec{o} \in \mathbb{R}^n$ in state j. Usually, a finite mixture of the form

$$b_j(\vec{o}) = \sum_{m=1}^{M} c_{jm}\, \mathcal{N}(\vec{o}, \mu_{jm}, U_{jm}) \quad \text{for} \quad 1 \le j \le N \tag{4.9}$$

is used, where c_{jm} is the mixture coefficient and $\mathcal{N}$ is a Gaussian density with mean vector μ_{jm} and covariance matrix U_{jm}, for the m-th component in state j, respectively. The mixture coefficients satisfy

$$c_{jm} \ge 0 \quad \text{for} \quad \begin{matrix} 1 \le j \le N \\ 1 \le m \le M \end{matrix} \tag{4.10}$$

$$\sum_{m=1}^{M} c_{jm} = 1 \quad \text{for} \quad 1 \le j \le N \tag{4.11}$$

such that the pdf is properly normalized, i.e.,

$$\int_{-\infty}^{\infty} b_j(x)\, dx = 1 \quad \text{for} \quad 1 \le j \le N \tag{4.12}$$

Such a pdf can be used to approximate, arbitrarily closely, any finite, continuous density function. Hence it can be applied to a wide range of problems.

Summing up, an HMM is characterized by the number of states, N, the number of observation symbols, M (infinite for the continuous case), the state transition probability distribution, $A = \{a_{ij}\}$, the observation symbol (or vector) probability distribution for each state, $B = \{b_j(k)\}$, and the initial state distribution, π (compare Equations 4.2 through 4.12). For convenience, we use the compact notation

$$\lambda = (A, B, \pi) \tag{4.13}$$

to indicate the complete parameter set of an HMM.

4.1.2 The Three Basic Problems for HMMs

Now that we have defined the form of HMMs, we can next look at the three basic problems that need to be solved when dealing with them. The problems are stated in the following and then illustrated by an example task in speech recognition.

1. Given an observation sequence O and a model λ, how can we compute the probability of the model producing the observation sequence, i.e., $P(O|\lambda)$?
2. Given an observation sequence O and a model λ, how can we compute an "optimal" corresponding state sequence?
3. How do we adjust the model parameters $\lambda = (A, B, \pi)$ to maximize $P(O|\lambda)$?

Let us consider the example of a simple isolated-word speech recognizer, which uses one HMM for each word, with a vocabulary of 100 words. Assume that we have 10 recordings of each word by one or more speakers. After extracting appropriate feature vectors (i.e., an observation sequence) from the recordings, a solution to Problem 3 can be used to train for each of the 100 words one HMM, which maximizes the probability of the model producing the observation sequences we saw in the 10 recordings of the word.

In an attempt to understand and refine a model, we can use a solution to Problem 2 to uncover the hidden part of the HMM, i.e., determine the state sequence that leads to the observations. Of course there is no "correct" sequence that can be found, rather, a sequence that satisfies some optimality criterion is computed.

This completes the training part, and we can attempt to recognize a word from a new recording using a solution to Problem 1, by again extracting the feature vectors that form the observation sequence O, and then compute $P(O|\lambda_w)$ for each word w from our vocabulary, i.e., the probability of the word HMM producing the observation sequence. The model with the highest probability wins and we have (hopefully) recognized the word.

Solution to Problem 1

An efficient solution to Problem 1 is given by the *forward procedure*, where we use forward variables of the form

$$\alpha_t(i) = P(o_1 o_2 \dots o_t, q_t = i \mid \lambda) \tag{4.14}$$

to describe the probability of the partial observation sequence $o_1 o_2 \dots o_t$ until time t, and state i at time t, given the model λ. We can solve for $\alpha_t(i)$ inductively, as follows:

1. Initialization

$$\alpha_1(i) = \pi_i \, b_i(o_1) \quad \text{for} \quad 1 \le i \le N \tag{4.15}$$

2. Induction

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) \, a_{ij} \right] b_j(o_{t+1}) \quad \text{for} \quad \begin{matrix} 1 \le t \le T-1 \\ 1 \le j \le N \end{matrix} \tag{4.16}$$

3. Termination

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{4.17}$$

For the solution of Problems 2 and 3 we will also make use of similar backward variables of the form

$$\beta_t(i) = P(o_{t+1}o_{t+2}\dots o_T \mid q_t = i, \lambda) \tag{4.18}$$

to describe the probability of the partial observation sequence from $t+1$ to the end, given state i at time t and the model λ. Again, we can solve for $\beta_t(i)$ inductively, as follows:

1. Initialization

$$\beta_T(i) = 1 \quad \text{for} \quad 1 \le i \le N \tag{4.19}$$

2. Induction

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij}\, b_j(o_{t+1})\, \beta_{t+1}(j) \quad \text{for} \quad \begin{array}{l} T-1 \ge t \ge 1 \\ 1 \le i \le N \end{array} \tag{4.20}$$

Solution to Problem 2

As already stated, there exists no "correct" solution to Problem 2, we need to define some optimality criterion that we wish the uncovered state sequence to satisfy. One possible criterion is to choose the states q_t that are *individually* most likely at each time t to maximize the expected number of correct states. To do so, we define *a posteriori* probability variables of the form

$$\gamma_t(i) = P(q_t = i \mid O, \lambda) \tag{4.21}$$

to describe the probability of being in state i at time t, given the observation sequence O and the model λ. They can be expressed as

$$\gamma_t(i) = \frac{\alpha_t(i)\, \beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\, \beta_t(j)} \tag{4.22}$$

Now we can solve for the individually most likely state q_t^* at time t as

$$q_t^* = \operatorname*{argmax}_{1\le i\le N} \gamma_t(i) \quad \text{for} \quad 1 \le t \le T \tag{4.23}$$

However, maximizing individual states can be problematic, for example two neighboring states in the resulting sequence could have transition probability 0 between them, making the sequence in fact impossible.

For this reason among others, the most widely used criterion is to find the single best state sequence, i.e., to maximize $P(q \mid O, \lambda)$. The *Viterbi algorithm*, a dynamic programming method, realizes this, making use of variables

$$\delta_t(i) = \max_{q_1,q_2,\dots,q_{t-1}} P(q_1q_2\dots q_{t-1}, q_t = i, o_1, o_2 \dots o_t \mid \lambda) \tag{4.24}$$

to describe the highest probability along a single path, at time t, which accounts for the first t observations and ends in state i. By induction we have

$$\delta_{t+1}(j) = \left(\max_i \delta_t(i)\, a_{ij}\right) \cdot b_j(o_{t+1}) \tag{4.25}$$

To actually retrieve the state sequence, we keep track of the argument that maximized Eq. (4.25) via variables $\psi_t(j)$. We can now state the Viterbi algorithm:

1. Initialization

$$\begin{array}{l} \delta_1(i) = \pi_i \, b_i(o_1) \\ \psi_1(i) = 0 \end{array} \quad \text{for} \quad 1 \le i \le N \tag{4.26}$$

2. Recursion

$$\delta_t(j) = \left(\max_{1 \le i \le N} \delta_{t-1}(i) \, a_{ij} \right) \cdot b_j(o_t) \quad \text{for} \quad \begin{array}{l} 2 \le t \le T \\ 1 \le j \le N \end{array} \tag{4.27}$$

$$\psi_t(j) = \operatorname*{argmax}_{1 \le i \le N} \delta_{t-1}(i) \, a_{ij} \quad \text{for} \quad \begin{array}{l} 2 \le t \le T \\ 1 \le j \le N \end{array} \tag{4.28}$$

3. Termination

$$P^* = \max_{1 \le i \le N} \delta_T(i) \tag{4.29}$$

$$q_T^* = \operatorname*{argmax}_{1 \le i \le N} \delta_T(i) \tag{4.30}$$

4. State sequence backtracking

$$q_t^* = \psi_{t+1}(q_{t+1}^*) \quad \text{for} \quad T - 1 \ge t \ge 1 \tag{4.31}$$

Solution to Problem 3

Problem 3 can be solved by means of the *Baum-Welch algorithm*, in which we use

$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j \mid O, \lambda) \tag{4.32}$$

to describe the probability of making a transition from state i to state j from time t to $t+1$, given the observation sequence and the model. Previously we introduced $\gamma_t(i)$ as the probability of being in state i at time t, given O and λ, hence

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j) \tag{4.33}$$

Now we can give formulae to re-estimate the parameters of an HMM given a new observation sequence:

$$\bar{\pi}_i = \text{expected no. of times in state i at time } (t=1) \; = \gamma_1(i) \tag{4.34}$$

$$\bar{a}_{ij} = \frac{\text{expected no. of transitions from state } i \text{ to state } j}{\text{expected no. of transitions from state } i} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{4.35}$$

$$\hat{b}_j(k) = \frac{\text{expected no. of times in state } j \text{ and observing symbol } v_k}{\text{expected no. of times in state } j} = \frac{\sum_{\{t \mid o_t = v_k\}} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \tag{4.36}$$

If $\lambda = (A, B, \pi)$ denotes the current model and we compute a re-estimated model $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$ as stated above, then it can be proven that either $\bar{\lambda} = \lambda$ or $P(O|\bar{\lambda}) > P(O|\lambda)$.

The solutions to the three problems have been discussed for HMMs with discrete observation densities. The formulae can also be applied to continuous densities, with minor modifications that are not going to be discussed here. There are also some implementation issues to address when developing an HMM-based system (e.g. numerical underflow), these issues are also not covered here.

4.1.3 Using HMMs for Modeling Speech

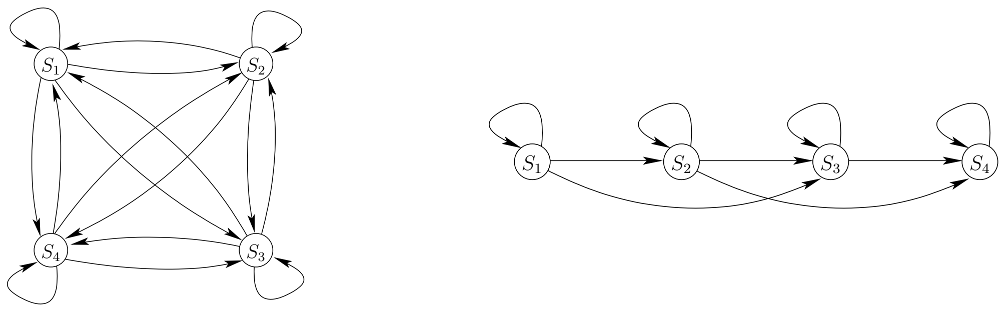
To model speech, HMMs with a topology that reflects the change of the speech signal over time have prevailed. So-called *left-right models* allow state transitions only from left to right, i.e., from states with lower indices to ones with higher indices. In other words, for the transition probability matrix A we have

$$a_{ij} = 0 \text{ for } j < i. \tag{4.37}$$

Furthermore, the initial state probabilities π have the property

$$\pi_i = \begin{cases} 0 & \text{for } i \neq 1 \\ 1 & \text{for } i = 1 \end{cases} \tag{4.38}$$

because the state sequence must begin in state 1 (and end in state N). Different types of HMMs are depicted in Figure 4.2.



(a)
all $a_{ij} > 0$

(b)
$a_{ij} > 0$ only for $i \leq j \leq i + 2$

Figure 4.2: (a) A general 4-state HMM, (b) A left-right 4-state HMM with maximum skip 2.

It remains to define feature vectors that can be used as suitable observations for speech HMMs. We have seen in Section 2.2 that spectral information of short time windows is suitable to digitally represent a speech signal. HMMs have been in use for speech recognition for quite some time, and experience in this area has led to the widespread use of *mel-frequency cepstral coefficients* (MFCCs) as features.

The *complex cepstrum* of a signal is defined as the Fourier transform of the logarithm of the Fourier transform of the signal. For a power spectrum (magnitude-squared Fourier transform) $S(\omega)$, which is symmetric with respect to $\omega = 0$ and is periodic for a sampled data sequence, the Fourier series representation of $\log S(\omega)$ can be expressed as

$$\log S(\omega) = \sum_{n=-\infty}^{\infty} c_n e^{-jn\omega} \tag{4.39}$$

where $c_n = -c_n \in \mathbb{R}$ are called the cepstral coefficients [RJ93]. For psychoacoustic reasons, cepstral coefficients for the mel-frequency scale – the MFCCs – are used in speech processing (see Section 2.1.3) [RJ93].

4.2 Mel-cepstral Analysis and Synthesis of Speech

Although MFCCs can be used for speech recognition, it is not possible to re-synthesize a speech signal from MFCCs alone – pitch information is also needed. The method in use today is based on Imai's results from 1983 [Ima83], when he proposed an analysis–synthesis method using a source-filter model. The excitation source is chosen by a switch which controls the voiced/unvoiced character of the speech signal. The excitation signal is modeled as a periodic pulse train for voiced signals, and as a random noise sequence for unvoiced signals. The source signal is then modified by a time-varying linear filter system which models the effects of the vocal tract on the speech signal, as shown in Figure 4.3.

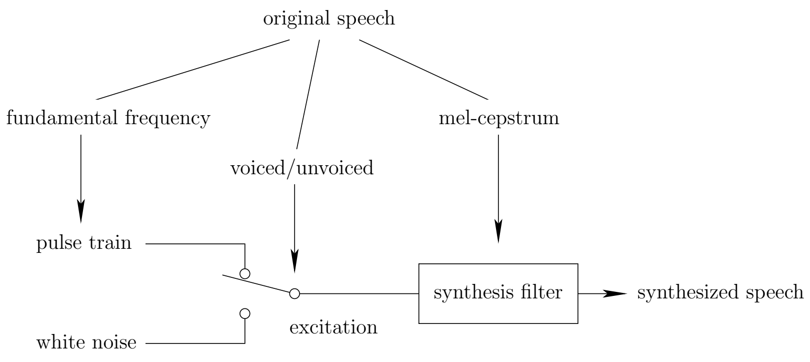


Figure 4.3: Source-filter model (after [Yos02]).

In 1992, Fukada, Tokuda, Kobayashi and Imai presented an improved algorithm for mel-cepstral analysis of speech [FTKI92] which is in wide use today.

The mathematical details of the *mel log spectrum approximation* (MLSA) filter, which realizes the source-filter re-synthesis, are given in the original papers [Ima83, FTKI92] and summarized in Yoshimura's doctoral dissertation [Yos02].

4.3 Generating Speech Parameters from HMMs

Masuko, Tokuda, Kobayashi and Imai developed a speech synthesis system based on HMMs using the source-filter model discussed in the previous section [TKI95, MTKI96]. A block diagram of the system is given in Figure 4.4. The system works as follows:

1. The training data consists of a database containing recorded speech and the corresponding phonetic transcriptions, with the latter also containing borders between phones on the time axis.

2. Speech parameters are extracted for each short-time window of speech (e.g., 25ms windows with a 5ms shift) to form the observation vectors.

3. One left-right HMM without skip is trained for each phone to model the observations seen in the training data. The observation densities are multi-dimensional Gaussian distributions.

4. To synthesize text, the input is converted to a phone sequence by means of natural language processing techniques.

5. The corresponding phone HMMs are concatenated to form one HMM for the whole utterance to be generated.

6. The most likely sequence of observations – speech parameters in this case – is generated from the concatenated HMM.

7. Using the MLSA filter technique, speech is synthesized from the speech parameters.

One key idea of the authors is to not only use MFCCs as spectral observations, but to also include dynamic features. Without them, the most likely observation at each state would always be the mean vector of the output pdf, and the generated speech parameter sequence would be a sequence of mean vectors.

Let c_t be the vector of mel-cepstral coefficients at frame t. Then its dynamic features Δc_t and $\Delta^2 c_t$ are computed as follows:

$$\Delta c_t = -\frac{1}{2}c_{t-1} + \frac{1}{2}c_{t+1} \tag{4.40}$$

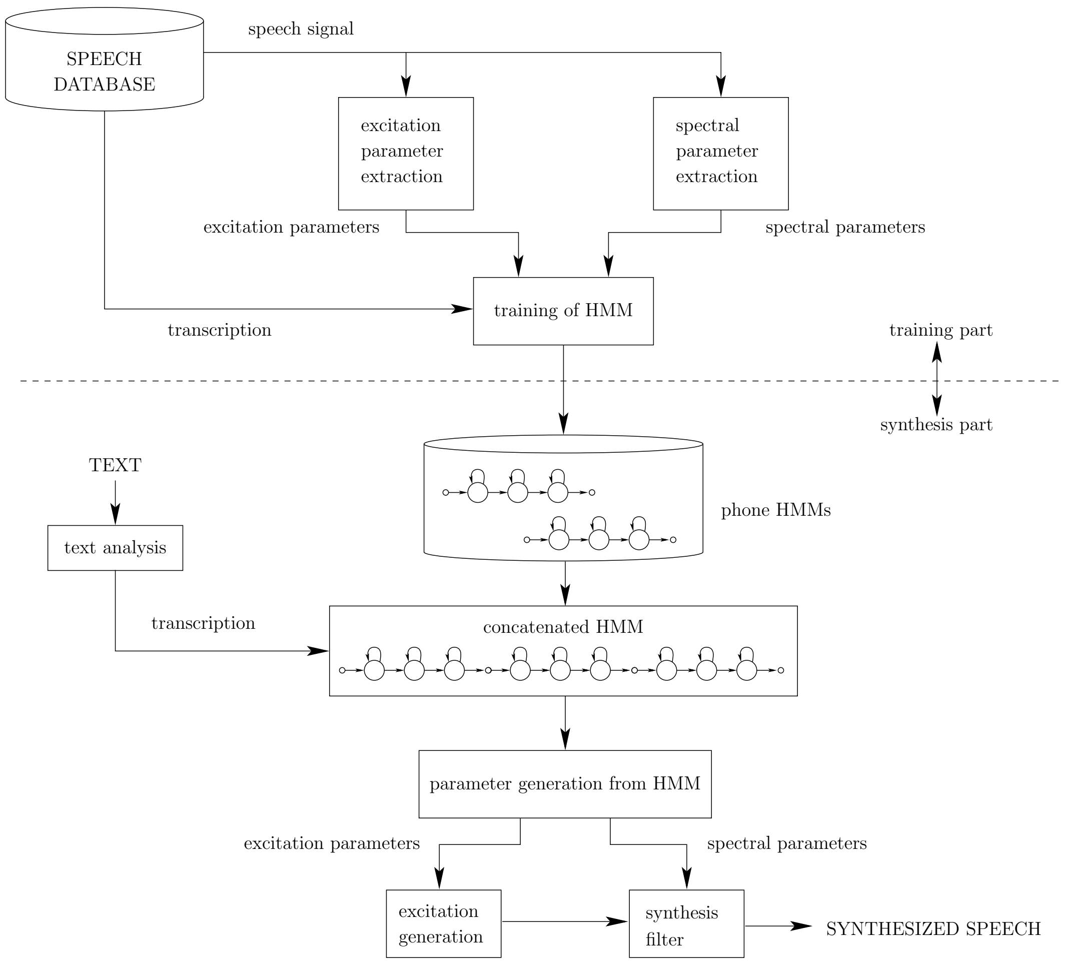


Figure 4.4: Overview of the HMM-based speech synthesis system (after [Yos02]).

$$\Delta^2 c_t = \frac{1}{4}c_{t-1} - \frac{1}{2}c_t + \frac{1}{4}c_{t+1}. \tag{4.41}$$

Furthermore, let $Q = (q_1, q_2, \ldots, q_T)$ be the state sequence and $O = (o_1, o_2, \ldots, o_T)$ be the output parameter sequence that are to be generated. Then maximizing $P(Q, O \mid \lambda, T)$ for an utterance HMM λ under the constraints 4.40 and 4.41 allows us to honor the gradual change of the spectral features over time, as illustrated in Figure 4.5. Hence, the spectral feature vectors that are extracted for each frame of training data consist for instance of 25 MFCCs, their deltas and delta-deltas, resulting in a 75-dimensional feature vector.

The authors have developed an algorithm to efficiently compute the most likely observation for an HMM subject to the constraints of the dynamic features [TKI95].

4.4 Context-dependent Models and State Tying

HMM-based speech parameter generation can be improved by the use of context-dependent models and state tying. Rather than training one HMM for every phone occurring in the training data, one HMM is trained for every occurring context. Possible contextual factors are the preceding, current and following phone (this is referred to as *triphone models*), preceding, current and following phone category (e.g., back vowel, plosive, etc.), prosodic factors like the position of the current phone in the current accentual phrase, etc. While one HMM still corresponds to one phone of the speech signal, two segments of training data with the same (current) phone are used to train two different HMMs if their

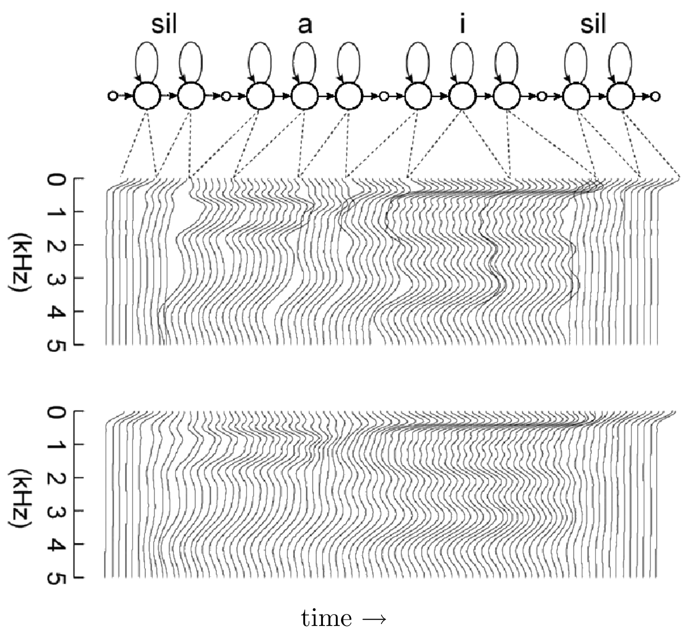


Figure 4.5: Example of generated spectral parameters from HMMs, without dynamic features (top) and with dynamic features (bottom) (copy from [TKI95]).

contexts differ. This allows for much more accurate modeling of speech, because contextual factors have a large impact on spectrum, pitch and duration [YOW94, Ode95]. Of course the transcriptions used for both training and synthesis will have to include this contextual information.

On the other hand, this approach results in a much higher number of HMMs to be trained, which is problematic as it means that for each model there is a smaller number of training samples from which the parameters can be estimated. Furthermore, it is impractical to have every possible context covered in the training data. To overcome this problem, a tied state HMM system is built in the following four steps (illustrated in Figure 4.6) [YOW94, YTM+98, YTM+99]:

1. An initial set of 3-state left-right monophone models with single Gaussian output probability density functions is created and trained.
2. The state output distributions of these monophones are then cloned to initialize a set of untied context-dependent triphone models which are then trained using Baum-Welch re-estimation. The transition matrix is not cloned but remains tied across all the triphones of each phone.
3. For each set of triphones derived from the same monophone, corresponding states are clustered. In each resulting cluster, a typical state is chosen as representative and all cluster members are tied to this state.
4. The number of mixture components in each state is incremented and the models re-estimated until performance on a development test set peaks or the desired number of mixture components is reached.

In the last step, re-estimation is performed as *embedded training*, i.e., as follows: for each speech sample in the training data, the HMMs for each occurring context are loaded and concatenated to form a composite HMM which spans the whole utterance. Observations are extracted from the speech signal and aligned to states of the sentence HMM using the Viterbi algorithm (solution to Problem 2 of Section 4.1.2). Then the output densities are re-estimated for each state.

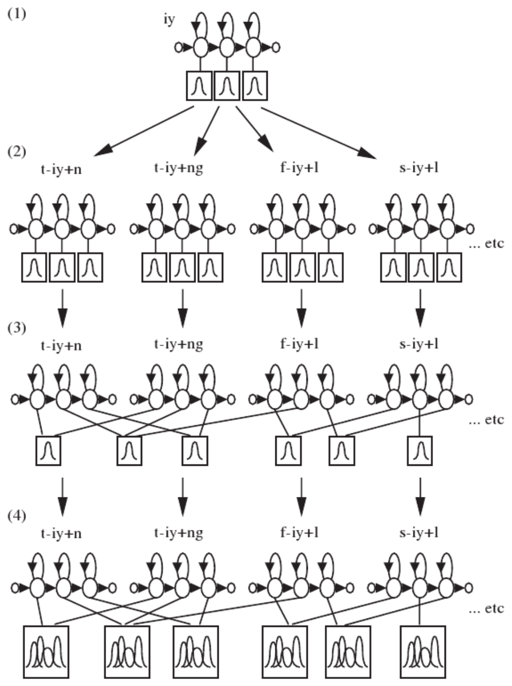


Figure 4.6: The tied-state HMM system build procedure (copy from [YOW94]).

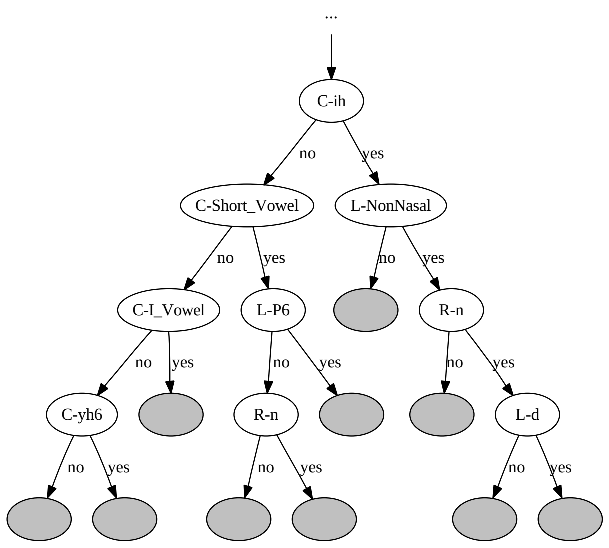


Figure 4.7: Part of an example decision tree.

4.4.1 Context Clustering Based on Decision Trees

The most straightforward way to cluster states of HMMs would be based on a distance metric regarding the output pdfs. However, it has turned out to be advantageous to build decision trees instead, using phonetic/linguistic questions about the current context [Ode95, YOW94, YTM+98, YTM+99].

Such a decision tree is a binary tree in which at each node a yes/no question splits contexts into two groups. For example, Figure 4.7 shows part of such a tree, where the topmost node represents the question "Is the current phone [iː]?". The set of contexts is split into two according to an affirmative or negative answer. In the example, we see questions about the current, preceding or following phone (C, L, R), asking for a specific phone (ih, n, P6, ...) or a phone category (Short_Vowel, NonNasal, ...). In the case of triphones, three trees are constructed. The first one is built to cluster all first states of all HMMs, etc. For example, the tree shown in Figure 4.7 will partition its states into ten subsets corresponding to its ten terminal nodes. The states in each subset are tied into a cluster to form a single state and the questions and the tree topology are chosen to maximize the likelihood of the training data, whilst ensuring that there is sufficient data to train each tied state.

One key advantage of this approach is that whatever context we are looking for, searching through the tree will always lead us to a representative terminal node (and thus an HMM), even if such a context was not part of the training data on which the tree was constructed.

Each tree is built using a top-down sequential optimization procedure [Ode95, YOW94]. Initially, all of the states to be clustered are placed in the root node of the tree and the log likelihood of the training data calculated on the assumption that all of the states in that node are tied. This node is then split into two by finding the question which partitions the states in the parent node so as to give the maximum increase in log likelihood. This process is repeated until the increase in likelihood falls below a threshold. To ensure that all terminal nodes have sufficient training data, a minimum occupation count is applied.

4.5 Simultaneous Modeling of Spectrum, Pitch and Duration

4.5.1 Explicit State Duration

In conventional HMMs, the duration of state occupancy is determined by the self-transition probability a_{ii}, and thus the probability to be in state S_i decreases exponentially with time. This type of state duration probability does not provide an adequate representation of the temporal structure of speech [Rab89, TKI95].

Instead, the state occupancy duration can be modeled explicitly by a duration probability density $p_i(d)$ = probability of d consecutive observations in state S_i. HMMs with explicit state duration probability densities are sometimes called hidden semi-Markov models (HSMMs), as the Markov property is not fully satisfied: The probability to be in a certain state at the next discrete point in time does not only depend on the current state, but also on the time that has been spent in the current state so far.

Yoshimura et al. improved their system to a unified framework of HSMMs which models spectrum, pitch and duration simultaneously [YTM$^+$98, YTM$^+$99, Yos02].

In their system, spectral and pitch parameters are put together into a common feature vector, as depicted in Figure 4.8. The distinct parts of the feature vector are referred to as *streams*, the corresponding HMMs are called *multi-stream HMMs*. During the re-estimation of the HMMs (as described in the last section), state durations are calculated on the trellis which is obtained in the embedded training stage, and modeled by n-dimensional Gaussian distributions, where n is equal to the number of states of the HMM in question, and the i-th dimension of the distribution corresponds to the i-th state of the HMM ($1 \leq i \leq n$).

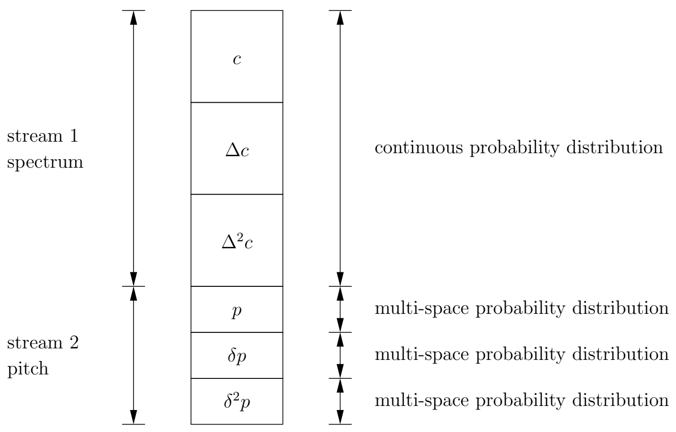

stream 1
spectrum
c
Δc
$\Delta^2 c$
continuous probability distribution
stream 2
pitch
p
δp
$\delta^2 p$
multi-space probability distribution
multi-space probability distribution
multi-space probability distribution

Figure 4.8: Feature vector (after [YTM$^+$99]).

State duration densities are estimated on the trellis which is obtained in the embedded training stage. For an utterance of total length T, the mean $\xi(i)$ and the variance $\sigma^2(i)$ of duration density of state i are determined by

$$\xi(i) = \frac{\sum_{t_0=1}^{T} \sum_{t_1=t_0}^{T} \chi_{t_0,t_1}(i)(t_1 - t_0 + 1)}{\sum_{t_0=1}^{T} \sum_{t_1=t_0}^{T} \chi_{t_0,t_1}(i)} \tag{4.42}$$

$$\sigma^2(i) = \frac{\sum_{t_0=1}^{T} \sum_{t_1=t_0}^{T} \chi_{t_0,t_1}(i)(t_1 - t_0 + 1)^2}{\sum_{t_0=1}^{T} \sum_{t_1=t_0}^{T} \chi_{t_0,t_1}(i)} - \xi^2(i) \tag{4.43}$$

where $\chi_{t_0,t_1}(i)$ is the probability of occupying state i from time t_0 to time t_1 and can be expressed as

$$\chi_{t_0,t_1} = (1 - \gamma_{t_0-1}(i)) \cdot \prod_{t=t_0}^{t_1} \gamma_t(i) \cdot (1 - \gamma_{t_1+1}(i)) \tag{4.44}$$

where $\gamma_t(i)$ is the occupation probability of state i at time t, and we define $\gamma_{-1}(i) = \gamma_{T+1}(i) = 0$.

4.5.2 F_0 Modeling with Multi-Space Distributions

Typical F_0 (fundamental frequency) patterns of speech contain voiced regions, where F_0 can be modeled by a continuous value as well as unvoiced regions, where there is no F_0, which should be modeled with a discrete switch (see Figure 4.9). Neither continuous nor discrete density HMMs are ideal to model such patterns. Tokuda et al. therefore introduced HMMs (and the corresponding formulae for training, re-estimation, etc.) for *multi-space probability distributions* (MSDs) [TMMK99].

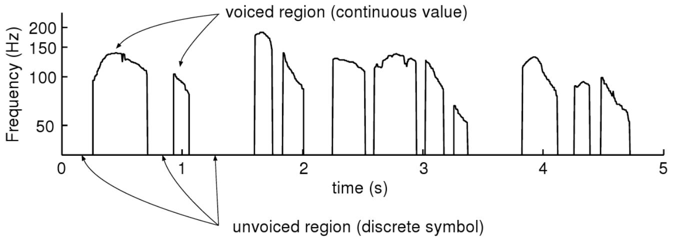


Figure 4.9: Example F_0 pattern (copy from [Yos02]).

The proposed MSD-HMMs include discrete and continuous mixture HMMs as special cases, so they can be used in place of both. Additionally, they can model observation sequences with variable dimensions, including zero-dimensional observations, i.e., discrete symbols. They are thus suitable to model F_0 patterns.

4.5.3 Independent Clustering of Spectrum, Pitch and Duration

Since each of spectrum, pitch and duration has its own influential contextual factors, the distributions for spectral parameters, pitch parameters and state duration are clustered independently (see Figure 4.10) [YTM+99].

In the experiments conducted by Yoshimura et al. [YTM+99], a system was built using 450 phonetically balanced sentences from a Japanese speech database for training. The speech signals were sampled at 16 kHz and windowed by a 25ms Blackman window with a 5ms shift. Then a feature vector was extracted from each window that consisted of 25 MFCCs including the 0-th coefficient, their deltas and delta-deltas, and log pitch with its delta and delta-delta. They used three-state left-right HMMs with single Gaussian output distributions. Decision tree based context clustering was applied as described, and resulted in trees with 6615, 1877 and 201 leaves in total for spectrum, pitch and duration, respectively. Examples thereof are shown in Figure 4.11, where "L_∗", "C_∗" and "R_∗" represent "preceding", "current" and "succeeding", respectively. "Silence" represents silence at the beginning and end of an utterance as well as during pauses. Questions concerning breath group and accentual phrase are represented by "∗_breath_∗" and "∗_accent_∗", respectively. "Pit_∗" and "dur_∗" represent leaf nodes.

We can see in these figures that spectrum models are much affected by phone identity, pitch models for voiced parts are much affected by accentual phrase and part-of-speech, and pitch models for unvoiced parts are clustered by a very simple tree. In the duration tree, silence and pause models are much affected by accentual phrase and breath group, while the other models are much affected by phone identity [YTM+99].

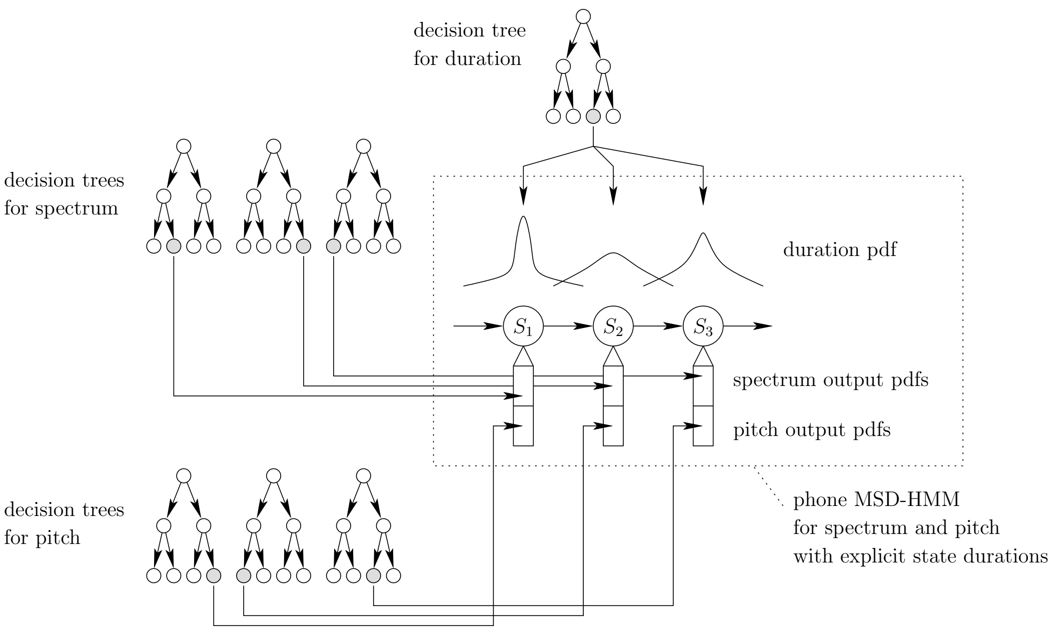


Figure 4.10: Distributions for spectrum, pitch and duration are clustered independently.

4.6 Average Voices and Speaker Adaptation

The HMM-based speech synthesis techniques described so far can produce natural sounding speech closely resembling the recorded speaker. However, it would be desirable to have the ability to produce speech with arbitrary voice characteristics and speaking styles.

To achieve this, a system based on average voices and speaker adaptation was developed [TMTK01, Yam06, YK07, YKN$^+$09]. An overview over the system is given in Figure 4.12. It works in four steps, which are briefly discussed in the following: Speech analysis, training of average voice model, speaker adaptation and speech synthesis [YKN$^+$09].

4.6.1 Speech Analysis

From a training corpus containing speech of several speakers, speech parameters are extracted using the analysis methods described earlier in this chapter. Multi-stream feature vectors containing MFCCs and $\log F_0$ plus their deltas and delta-deltas are computed for each frame [YKN$^+$09].

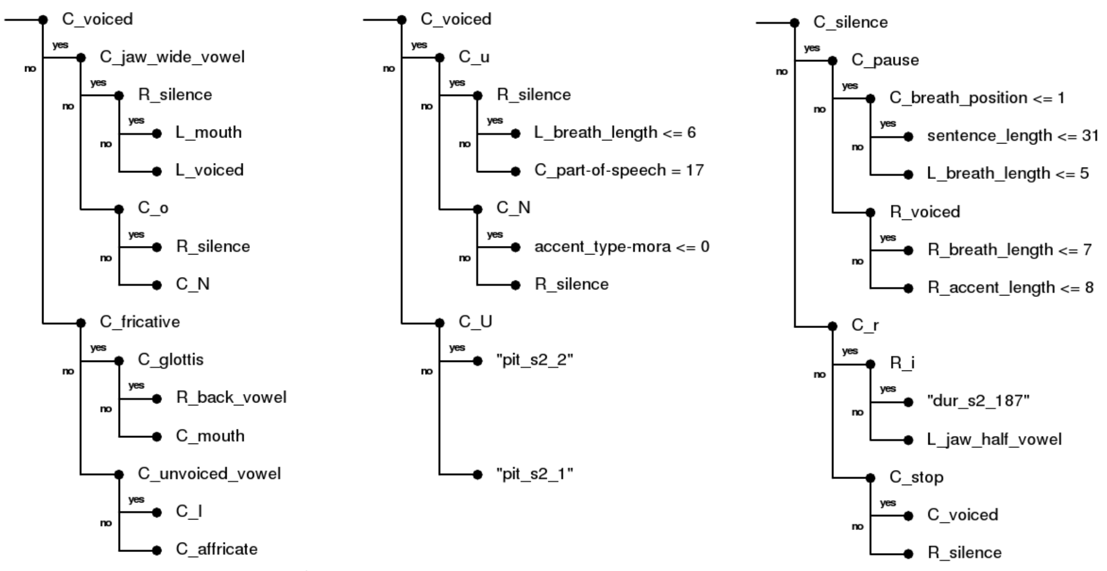
4.6.2 Training of Average Voice Model

The feature vectors obtained in the last step are used to train an average voice model in the form of a collection of context-dependent multi-stream MSD-HSMMs, similar to the single-speaker system described earlier [YKN$^+$09].

The training data for the average voice includes a lot of speaker- and/or gender-dependent characteristics. However, the goal is to adapt to a wide variety of target speakers, therefore the average voice model should avoid such speaker- and/or gender-dependent characteristics [YK07]. To overcome this problem, a *speaker-adaptive training* (SAT) algorithm is applied, which normalizes the influence of speaker differences in both state output (i.e., spectral and pitch parameters) and state duration. In this algorithm, the difference between each of the training speakers voices and the canonical average voice is assumed to be expressed by a simple linear regression function:

$$\mu_i^{(f)} = \quad \zeta^{(f)}\mu_i + \epsilon^{(f)} \quad = W^{(f)}\xi_i \tag{4.45}$$

$$m_i^{(f)} = \quad \chi^{(f)}m_i + \nu^{(f)} \quad = X^{(f)}\phi_i, \tag{4.46}$$



(a) Tree for spectrum model (first state)
(b) Tree for pitch model (first state)
(c) Tree for state duration model

Figure 4.11: Examples of decision trees (copy from [YTM$^+$99]).

where $\mu_i^{(f)}$ and $m_i^{(f)}$ are respectively the mean vectors of the state output and duration distributions for training speaker f and state i, μ_i and m_i are the corresponding means for the average voice model, and $\xi_i = [\mu_i^\top, 1]^\top$ and $\phi_i = [m_i, 1]^\top$ are their correspondants in homogeneous coordinates. Finally, $W^{(f)} = [\zeta^{(f)}, \epsilon^{(f)}]$ and $X^{(f)} = [\chi^{(f)}, \nu^{(f)}]$ are transformation matrices which indicate the difference between training speaker f and the average voice.

Let F be the number of training speakers, $O = \{O^{(1)}, \ldots, O^{(F)}\}$ be all the training data, and $O^{(f)} = \{o_{1_f}, \ldots, o_{T_f}\}$ be the training data of length T_f for speaker f. The HSMM-based speaker-adaptive training algorithm simultaneously estimates the parameter set of the HSMM λ and the set of transformation matrices $\Lambda^{(f)} = (W^{(f)}, X^{(f)})$ for each training speaker so that the likelihood of O is maximized. The problem of the HSMM-based speaker adaptive training based on the maximum likelihood criterion can now be formulated as follows:

$$\begin{aligned}(\tilde{\lambda}, \tilde{\Lambda}) &= \operatorname*{argmax}_{\lambda,\Lambda} P(O \mid \lambda, \Lambda) \\ &= \operatorname*{argmax}_{\lambda,\Lambda} \prod_{f=1}^{F} P(O^{(f)} \mid \lambda, \Lambda^{(f)})\end{aligned} \tag{4.47}$$

where $\Lambda = (\Lambda^{(1)}, \ldots, \Lambda^{(F)})$ is the set of the transformation matrices for all training speakers [YK07].

The problem is solved using an iterative re-estimation procedure based on the Baum-Welch algorithm, the details of which shall not be discussed here [Yam06, YK07, YKN$^+$09].

Another important technique in training average voice models is concerned with context clustering. Ideally, an average voice model would be trained on a very large corpus of speech recordings of many speakers, using the same sentence set for each of them. Such a speech database is difficult to establish in practice, however. It would be desirable that the amount of data for each speaker is as small as possible. Furthermore, it would be desirable to allow and even encourage varying sentence sets, in order to obtain a (common) speech database that is rich in phonetic and linguistic contexts. However, problems in context clustering arise when the contexts contained in each speaker's data are quite different: The nodes of the decision tree will not always have training data of all speakers, and some nodes will have data from only one speaker. This will cause a degradation of quality of the average voice, especially in prosody.

A solution to this problem was proposed and called shared decision tree context clustering (STC)

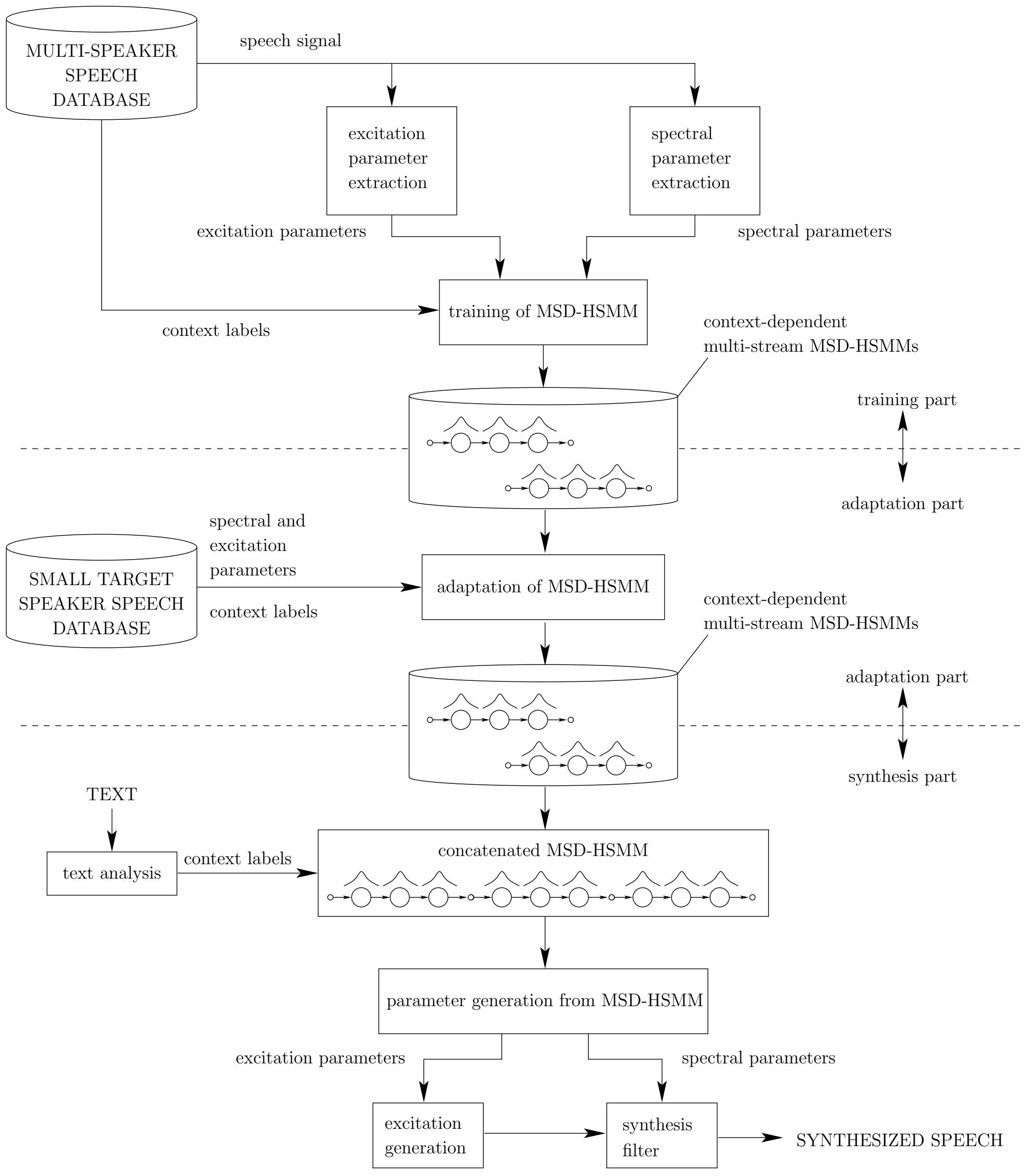


Figure 4.12: Overview of the average voice based speech synthesis system (after [YKN+09]).

[YTM+03]. In this technique, at first speaker-dependent models are trained for each speaker in the training corpus. Then one shared decision tree is created for context clustering. To split a node in the decision tree, only the contextual questions which are applicable to all speaker dependent models are considered. As a result, every leaf node of the decision tree always has training data of all speakers available. Using the common decision tree, all speaker-dependent models are clustered, i.e., an average voice model is obtained by combining the pdfs of all speaker-dependent models at each leaf node of the tree [YTM+03].

4.6.3 Speaker Adaptation

The average voice model obtained in the described way is then adapted using a small amount of training data from a target speaker. Similar to SAT (compare Section 4.6.2), this is achieved through maximum likelihood linear regression (MLLR). The final state output and duration distributions for the target speaker are obtained by linearly transforming the corresponding mean vectors of the average voice:

$$\begin{aligned} b_i(o) &= \mathcal{N}(o; \zeta\mu_i + \epsilon, U_i) && (4.48)\\ &= \mathcal{N}(o; W\xi_i, U_i) && (4.49)\\ p_i(d) &= \mathcal{N}(d; \chi m_i + \nu, \sigma_i^2) && (4.50)\\ &= \mathcal{N}(d; X\phi_i, \sigma_i^2) && (4.51) \end{aligned}$$

where μ_i and m_i are the respective mean vectors of state output and duration distributions of the average voice model, $W = [\zeta, \epsilon]$ and $X = [\chi, \nu]$ are the transformation matrices which transform the extended mean vectors $\xi_i = [\mu_i^\top, 1]^\top$ and $\phi_i = [m_i, 1]^\top$ of the target speaker model. Now we can again state the optimization problem of finding a set of transformation matrices $\Lambda = (W, X)$ such that the likelihood of the adaptation data O is maximized:

$$\tilde{\Lambda} = (\tilde{W}, \tilde{X}) = \underset{\Lambda}{\operatorname{argmax}}\, P(O \mid \lambda, \Lambda) \qquad (4.52)$$

Again, the details of the Baum-Welch-based procedure that solves the problem shall not be discussed here [TMTK01, Yam06, YK07].

4.6.4 Synthesis

Once the adapted target voice model is obtained, speech can be synthesized in the same way as with the speaker-dependent system discussed earlier. It should be noted here that speech closely resembling the target speaker can be synthesized by such a system with as little adaptation data as 10 sentences [Yam06], i.e., roughly one minute.

4.7 Advanced HMM Speech Synthesis Tasks

The flexibility of the speech synthesis method presented in the preceding sections allows for various advanced tasks which would be difficult if not impossible to address in a waveform-concatenation approach. Obviously, the adaptation technique alone already offers interesting possibilities. It not only allows us to build a voice which resembles a target speaker at low expense, we could also target specific speaking styles by recording an adaptation corpus containing speech of that style only (e.g., neutral, angry, happy, sad, fast, slow speech).

The HMM-based approach also allows for several ways of interpolation, which can in turn be very useful for attempts in the direction of speech with different speaking styles and emotions [YMK04, TYMK04, TYMK05].

Furthermore, the success of this approach has inspired researchers to use it in other analysis–re-synthesis tasks like, for example, singing voice synthesis [SZN$^+$06], speech-driven lip motion synthesis [HYS08], human walking motion synthesis [NYK05].

Only interpolation techniques shall be discussed here.

4.7.1 Model Interpolation

The goal in speaker interpolation is to synthesize speech with an untrained speaker's voice characteristics by interpolating the HMM parameters from some representative speakers' HMM sets. The proposed system [YMT$^+$97, YTM$^+$00] works very similar to the previously discussed system, as illustrated in Figure 4.13. The difference is that first several sets of phone HMMs $\Lambda_1, \ldots, \Lambda_F$ are trained for each of F speakers, and then a new set of phone HMMs Λ is generated by interpolating among them with an arbitrary interpolation ratio $a_1, \ldots, a_F$, $\sum_{f=1}^{F} a_f = 1$. Speech is then synthesized from the interpolated models in the usual way.

Interpolation among HSMMs is equivalent to interpolation among output probability and duration densities of corresponding states (under the assumption of equal HSMM topology, e.g., context-dependent three-state left-right MSD-HSMMs). If for each state the output and duration densities for

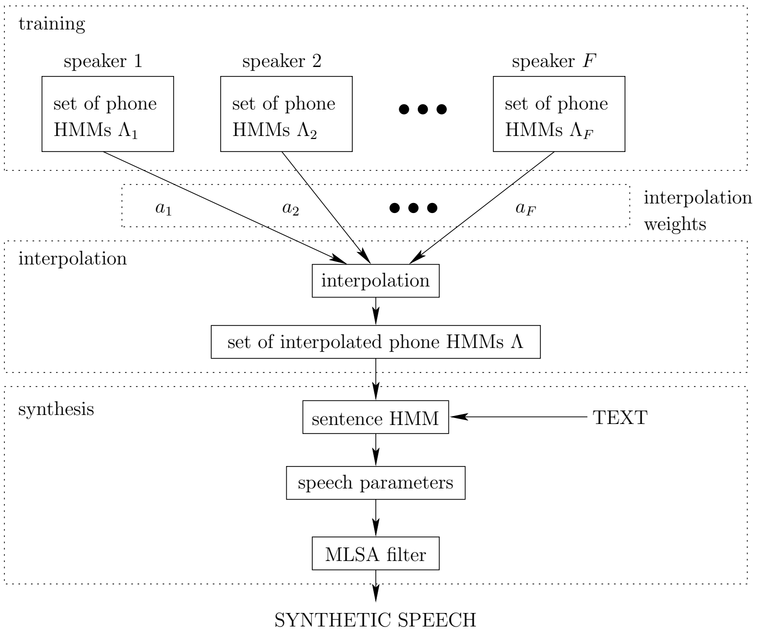


Figure 4.13: Overview of the speaker interpolation system (after [YMT+97]).

speaker f are single Gaussian pdfs $b_f(o) = \mathcal{N}(o; \mu_f, U_f)$ and $p_f(d) = \mathcal{N}(d; m_f, {\sigma_f}^2)$, respectively, and o and d are the speech parameter vector and the duration, then the problem is reduced to interpolation among the Gaussian pdfs. Three methods to interpolate among pdfs have been proposed [YTM+00]:

(a) Interpolation among observations: If we define the interpolated pdf $p(o) = \mathcal{N}(o; \mu, U)$ as a pdf of a random variable

$$o = \sum_{f=1}^{F} a_f o_f \tag{4.53}$$

where $\sum_{f=1}^{F} a_f = 1$, the mean μ and covariance matrix U are calculated as follows:

$$\mu = \sum_{f=1}^{F} a_f \mu_f \tag{4.54}$$

$$U = \sum_{f=1}^{F} {a_f}^2 U_f \tag{4.55}$$

(b) Interpolation among output distributions: Assume that mean μ_f and U_f are trained by using γ_f feature vectors of speaker f. If the interpolated pdf p is trained by using the feature vectors of all F representative speakers, this pdf is determined as

$$\mu = \frac{\sum_{f=1}^{F} \gamma_f \mu_f}{\gamma} = \sum_{f=1}^{F} a_f \mu_f \tag{4.56}$$

$$U = \frac{\sum_{f=1}^{F} \gamma_f U_f}{\gamma} - \mu\mu^\top = \sum_{f=1}^{F} a_f (U_f + \mu_f \mu_f^\top) - \mu\mu^\top \tag{4.57}$$

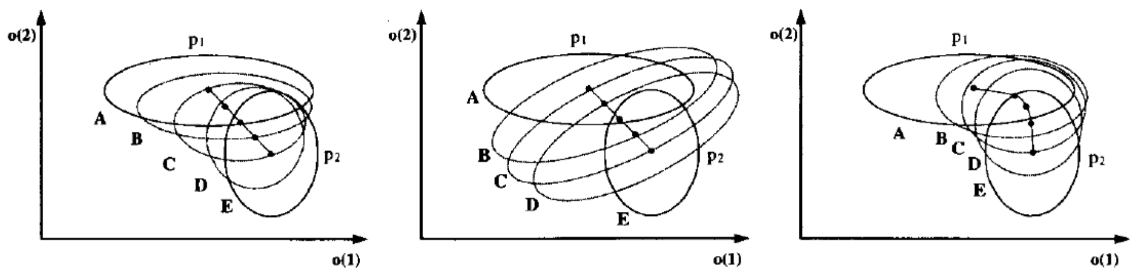
where $\gamma = \sum_{f=1}^{F} a_f \gamma_f$ and $a_f = \gamma_f / \gamma$.

(c) Interpolation based on KL-divergence: It is assumed that the similarity between the interpolated speaker S and each representative speaker S_f can be measured by the *Kullback-Leibler divergence*. For given pdfs $p_1, \ldots, p_F$ and interpolation weights $a_1, \ldots, a_F$, determine the interpolated pdf $p(o) = \mathcal{N}(o; \mu, U)$ by minimizing a cost function ε with respect to μ and U, where

$$\varepsilon = \sum_{f=1}^{F} a_f I(p, p_f) \tag{4.58}$$

and $I(p_1, p_2)$ is the KL-divergence from p_1 to p_2 [YTM+00].

An evaluation of the three methods by means of a simulated interpolation suggested that method (c) is most appropriate to interpolate HMMs which model speech spectra [YTM+00]. As illustrated in Figure 4.14, it can be seen that with methods (a) and (b) the interpolated mean vector is determined irrespective of the covariances of the representative distributions p_1 and p_2, whereas method (c) honors this additional information. Subjective listening tests showed that speech generated by means of model interpolation is indeed perceived as in between the representative speakers [YTM+00].



(a) (b) (c)

Figure 4.14: Comparison between methods (a), (b) and (c) with regard to interpolation between two Gaussian distributions p_1 and p_2 with the following interpolation ratios: **A**: $(a_1, a_2) = (1, 0)$, **B**: $(a_1, a_2) = (0.75, 0.25)$, **C**: $(a_1, a_2) = (0.5, 0.5)$, **D**: $(a_1, a_2) = (0.25, 0.75)$, **E**: $(a_1, a_2) = (0, 1)$

If the models of the representative speakers have a tied structure common to all models, it is possible to obtain an interpolated model Λ directly by interpolating among the Λ_f. In general, context clustering is applied independently for each speaker model, however, and in such cases it is not easy to obtain an interpolated model taking the structure of each representative style into account. To avoid this problem, interpolation can be applied on-line between two generated sentence HSMMs in the synthesis stage [TYMK04, TYMK05].

This interpolation technique can not only be applied to speaker interpolation in the sense of gradually changing from one person's voice characteristics to another, but also to interpolate between different speaking styles of the same person (or both). Tachibana et al. successfully interpolated between four different emotional speaking styles "neutral", "joyful", "sad" and "rough" [TYMK04, TYMK05]. Their subjective listening tests showed that speech from interpolated models is perceived as a speaking style in between two representative speaking styles, such that gradual changes from one emotion to an other are possible.

Furthermore, in expressive speech synthesis, it has been shown to be efficient to consider the speaking style/emotion as a contextual factor in the clustering [YMK04]. This leads to a compact model for all styles together, rather than a full model for each single style. These two methods could be shown to perform almost equally. As a drawback, to add a new style the whole acoustic model would have to be reconstructed.

One contribution made in this thesis is the adoption of these techniques to dialect interpolation, as discussed in Section 6.5.

Chapter 5

Austrian German and Viennese TTS

So far, we have considered speech synthesis independently of the target language. It should be clear that two TTS systems for different languages differ in the transcription part (mainly the lexica and letter-to-sound rules), and in the case of a data-driven approach also in the recorded speech.

However, speakers within the same language can differ to a quite large extent, depending on their region of origin. It is desirable to offer users regionalized TTS (as well as speech recognition) systems, in order to maximize intelligibility (or recognition performance) and naturalness of human-machine communication.

Using the example of Austrian German and Viennese dialects, Neubarth et al. [NPK08] have studied strategies for building adapted TTS systems for dialectal or regional varieties from a given standard source, with a focus on the unit selection approach. One key question was how much recoding is necessary for a transfer from one variety to an other (German German to Austrian German in this case), and to what extent it is possible to rely on the speech data alone. If, for example, one variety differs from an other only in vowel qualities and a neutralization of certain plosives, nothing has to be done beyond recording. However, this is rarely the case, especially with regard to non-standard[1] dialect or sociolect varieties. Rather, differences can be expected on all levels of representation. Figure 5.1 shows some identified differences between standard Austrian and Viennese dialect, along with the affected coding level.

Subsequently, an Austrian German unit selection TTS system was built [KPM+09] based on modifications to an existing lexicon for German German. Additionally, Viennese dialect unit selection voices were built (hitherto unpublished). The system is based on the open-source waveform-concatenation TTS system Festival [BL07, CRK07]. Linguistically, the applied modifications towards the Austrian variety are based on research by Muhr et al. [MS97, Muh07]. They are presented in the following.

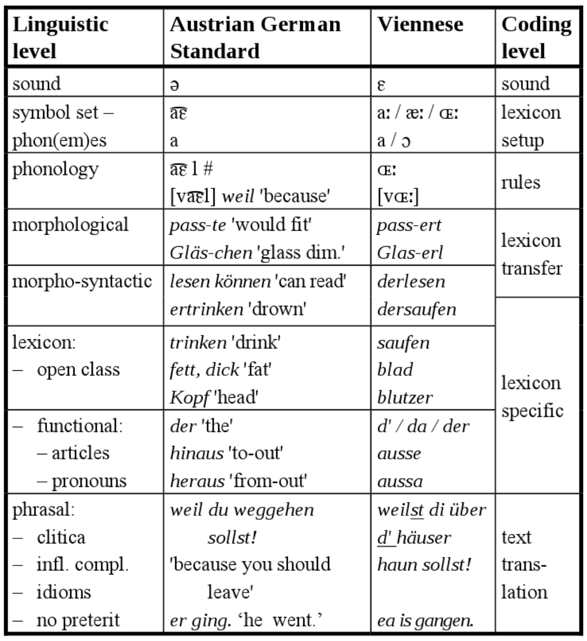
5.1 Differences between Austrian and German German

In the following, the most important differences between Austrian German and German German are listed [NPK08, KPM+09]. Note that there might be additional rules, which were omitted because the relevant phonetic properties will be unambiguously represented in the recorded speech data.

5.1.1 No voiced sibilants

In Austrian German, voiced sibilants (/z/ and /ʒ/) are realized in their unvoiced form (/s/ and /ʃ/). Examples:

[1] "non-standard" in this context means, e.g., that there is no defined orthography, there are no standard lexica, the variety is not used in "official" contexts (legislature, jurisprudence, etc.), not being taught in schools and not or barely used in broadcasting and print publications – all of which is true for Viennese dialect(s), but false for the standard Austrian variety of German.



Linguistic level	Austrian German Standard	Viennese	Coding level
sound	ə	ɛ	sound
symbol set – phon(em)es	a͡ɛ a	aː / æː / ɶː a / ɔ	lexicon setup
phonology	a͡ɛ l # [va͡ɛl] *weil* 'because'	ɶː [vɶː]	rules
morphological	*pass-te* 'would fit' *Gläs-chen* 'glass dim.'	*pass-ert* *Glas-erl*	lexicon transfer
morpho-syntactic	*lesen können* 'can read' *ertrinken* 'drown'	*derlesen* *dersaufen*	
lexicon: – open class	*trinken* 'drink' *fett, dick* 'fat' *Kopf* 'head'	*saufen* *blad* *blutzer*	lexicon specific
– functional: – articles – pronouns	*der* 'the' *hinaus* 'to-out' *heraus* 'from-out'	*d' / da / der* *ausse* *aussa*	
phrasal: – clitica – infl. compl. – idioms – no preterit	*weil du weggehen sollst!* 'because you should leave' *er ging.* 'he went.'	*weilst di über d' häuser haun sollst!* *ea is gangen.*	text trans- lation

Figure 5.1: Levels of representation concerning differences between Austrian standard and Viennese dialect (copy from [NPK08]).

word	gloss	German	Austrian
sechs	six	/zɛks/	/sɛks/
Genie	genius	/ʒeːˈniː/	/ʃeːˈniː/

5.1.2 /r/-vocalization

In Austrian German, the phoneme /r/ post-vocalically fully or partially vocalizes to /ɐ/. Example:

word	gloss	German	Austrian
werben	to solicit	/ˈvɛrbn̩/	/ˈvɛɐbn̩/

5.1.3 Schwa-elision

The e-schwa (/ə/) tends to minimize or completely disappear before a nasal in non-onset position in Austrian German. Example:

word	gloss	German	Austrian
fahren	to drive	/ˈfaːrən/	/ˈfaːn̩/

5.1.4 /ç/ versus /k/

Words ending in -ig are pronounced /ɪç/ in the German standard, whereas the Austrian pronunciation is /ik/. Furthermore, for foreign words starting with ch-, the German pronunciation is /ç/ while the Austrian standard is /k/. Examples:

word	gloss	German	Austrian
billig	cheap	/ˈbɪlɪç/	/ˈbɪlɪk/
Chemie	chemistry	/çeˈmiː/	/keˈmiː/

5.2 Additional Rules for Viennese Dialect

For the Viennese varieties, the following additional processes were identified [NPK08] and divided into three groups [PSYN09] (see also Section 6.5). The examples listed here are mostly from these two sources.

5.2.1 Neutralization of Plosives

In Viennese, plosives are neutralized in onsets, with the exception of /k/, which is only neutralized in complex contexts. Examples:

word	gloss	Austrian	Viennese
Tür	door	/tyːɐ/	/diːɐ/
Platz	place	/plats/	/blɑts/
Käse	cheese	/ˈkeːsə/	/kaːs/
kriegen	to obtain	/ˈkriːgŋ/	/ˈgriːɐŋ/

5.2.2 Lenition of Lenis Plosives

Intervocalically and before syllabic nasals and /l/, lenition of lenis plosives occurs. Examples:

word	gloss	Austrian	Viennese
jeder	everyone	/ˈjeːdɐ/	/ˈjeːðɐ/
Leber	liver	/ˈleːbɐ/	/ˈleːβɐ/

5.2.3 /l/-vocalization

Depending on the syllabic structure, /l/ is sometimes vocalized. Examples:

word	gloss	Austrian	Viennese
weil	because	/va͜ɪl/	/vœː/
Holz	wood	/hɔlts/	/hoɪts/
viel	much	/fiːl/	/fyː/
viele	many	/ˈfiːlə/	/ˈfyːlə/

5.2.4 Schwa-elision

The e-schwa is elided even more often than in the Austrian standard. Examples:

word	gloss	Austrian	Viennese
Gewicht	weight	/gəˈvɪçt/	/gviçt/
Hände	hands	/ˈhɛndə/	/hɛnd̥/

5.2.5 Monophthongization

Diphthongs are replaced by monophthongs. Examples:

word	gloss	Austrian	Viennese
zwei	two	/tsva͜ɪ/	/tsvaː/
deutsch	German	/dɔ͜ʏtʃ/	/dæːtʃ/
heute	today	/hɔ͜ʏtə/	/hɜːd̥/

5.2.6 Vowel Shifts

Some vowels are sometimes shifted from Austrian German to Viennese. Examples:

word	**gloss**	**Austrian**	**Viennese**
Schlag	beat	/ʃlaːk/	/ʃlɔːg̥/
lieb	dear	/liːp/	/lɪɐb̥/
bitte	please	/bɪtə/	/bitːə/
offen	open	/ˈɔfn̩/	/ˈofːm̩/

Chapter 6

HMM-based Synthesis of Austrian German and Viennese Dialect

We have built HMM-based TTS systems for Austrian German and Viennese dialect/sociolect [PSYN09]. This chapter first gives an overview over the system and describes the speech data used. Then various modeling approaches which have been taken are discussed, followed by an experimental evaluation of the same. Finally, our dialect interpolation system is described and evaluated.

6.1 System Overview

Our system uses the framework from the "HTS-2007/2008" system [YZW^+08, YNZ^+09], a speaker-adaptive system submitted to the Blizzard Challenge 2007 and 2008 [KKCM08]. The techniques used in the system have been described in detail in Chapter 4, Figure 4.12 gives an overview.

In the speech analysis part, feature vectors are extracted consisting of 40 mel-frequency cepstral coefficients, $\log F_0$ and average aperiodicity measurements, all with their respective deltas and delta-deltas. The feature extraction as well as the corresponding source-filter re-synthesis methods of this particular system are described in detail by Zen et al. [ZTNT07].

Average voices are trained on a multi-speaker multi-variety database (see Section 6.2), resulting in context-dependent multi-stream five-state left-to-right MSD-HSMMs without skip transitions, which model the acoustic features and state duration simultaneously (compare Section 4.5).

For clustering, the following contextual features on the phonetic, segment, syllable, word and utterance level are considered:

- Preceding, current and succeeding phones
- Acoustic and articulatory classes of preceding, current and succeeding phones
- Part-of-speech of the preceding, current, and succeeding morphemes
- The number of syllables and the type of accent in the preceding, current, and succeeding accentual phrases
- The position of the current syllable in the current accentual phrase
- The differences between the position of the current syllable and the type of accent
- The number of syllables in the preceding, current and succeeding breath groups
- The position of the current accentual phrase in the current breath group
- The number of words and syllables in the sentence
- The position of the breath group in the sentence

- The variety of the phone in the case of clustering of dialects

Speaker-adaptive training (SAT, compare Section 4.6.2) is applied to construct average voices, which are then adapted to target speakers by means of constrained structural MAPLR [YKN$^+$09]. Sentence HMMs are built from the adapted models in the synthesis stage, acoustic feature sequences are generated and used as input to an MLSA filter to create a speech waveform.

6.2 Speech Database

The database used for building the average voices and adapting to target speakers consisted of recordings from six different male speakers. Five of them are native speakers of the Viennese dialect. Table 6.1 lists the speakers, along with the number of utterances available for the Austrian German (AT) and Viennese dialect (VD) varieties.

				Number of utterances	
Speaker	Gender	Age	Profession	AT utterances	VD utterances
HPO	M	≈ 60	actor	219	513
SPO	M	≈ 40	radio speaker	4440	95
FFE	M	≈ 40	engineer	295	–
BJE	M	≈ 50	actor	87	95
FWA	M	≈ 60	language teacher	87	95
CMI	M	≈ 35	singer	–	95

Table 6.1: Data sources used for training and adaptation of standard Austrian German (AT) and Viennese dialect (VD) models.

For each utterance, the database contains two files: the recorded waveform and a transcription in the form of a text file in Festival utterance structure (see Appendix B). The speech data was recorded for different purposes and are thus of varying size and variety scope. BJE, FWA and CMI were recorded for unit selection test voices, FFE and SPO were recorded for a small and large unit selection voice, respectively, and HPO was recorded for the experiments described in this work (and in Pucher et al. [PSYN09]).

It would of course be preferable to have a more balanced database with equal numbers of AT and VD utterances. The collection and transcription of speech data is however a very time-consuming task, and a more balanced data set is not available at the moment. Therefore, the best modeling approaches using the data at hand, even though unbalanced, are explored.

The phone sets that were used to encode the data for the two varieties are shown in Table 6.2. For the AT variety, 60 different phones (including silence and pause) were used, whereas the VD variety was transcribed using only 40 phones. In the latter case, phones had to be clustered extensively due to the small amount of available data. Especially the very large SPO corpus justifies a more fine-grained distinction of phones for the AT variety.

6.3 Modeling Approaches

Speaker HPO was selected as the target speaker, as the data for this speaker is phonetically balanced for both varieties. Given the training data described in the last section, several modeling approaches are possible, which are summarized in Table 6.3.

6.3.1 Speaker-dependent and Dialect-dependent Voices

When a voice is speaker-dependent (SD), only data from one speaker is used. Likewise, when a voice is dialect-dependent (DD), only data from one variety is used. Therefore, the two SD-DD voices

	Austrian German	Viennese dialect
vowel	a aː ɒː eː ɛ ɛː iː ɪ ə ɐ oː ɔ uː ʊ yː ʏ œː ɶ	a ɒ e ɛ i ɪ ə ɐ o ɔ u ʊ y œ ɶ
diphthong/nasal	aɪ aʊ ɔʏ æ̃ː ɶ̃ː ɔ̃ː	æː ɔː œː
r-vocalized	ɛɐ ɛːɐ iːɐ ɪɐ oːɐ ɔɐ uːɐ ʊɐ yːɐ ʏɐ œːɐ ɶɐ	
plosive	b d g p t k	b d g p t k
fricative	f v s z ʃ ʒ h ç x	f v s ʃ h ç x
liquid/nasal/glide	r l m n ŋ j	
silence/pause/glottis	sil pau ʔ	

Table 6.2: Phones used in modeling Austrian German and Viennese dialect.

are directly trained from the AT and VD data, respectively, of speaker HPO, without any speaker adaptation.

6.3.2 Speaker-independent and Dialect-dependent Voices

For speaker-independent (SI) voices, an average voice using all speakers is built first and then adapted to the target speaker. In the case of dialect-dependent voices (DD), only data from one variety is used for both training of the average voice and adaptation to the target speaker.

6.3.3 Speaker-independent and Dialect-independent Voices

In the case of dialect-independent (DI) voices, an average voice is trained using data from both varieties. So for SI-DI, all data from the six speakers contributes to the average voice, which is then adapted to either the AT or the VD variety of target speaker HPO.

When dealing with data from multiple varieties of the same speakers, an additional possibility arises: The data from one speaker can be treated as coming from two separate speakers, split according to variety. As a result the SAT estimation while building the average voice will also normalize acoustical differences between the varieties (in addition to differences between speakers). The speaker-independent dialect-independent model with this technique applied is named SI-DN. The number of speakers contributing to the average voice here is thus 10.

6.3.4 Speaker-dependent and Dialect-independent Voices

When only data from target speaker HPO is used, but from both varieties, we speak of speaker-dependent dialect-independent voices. The nomenclature is analog to the last section, so for SD-DI an average voice is built from all HPO data and hence no SAT estimation is performed, as there is only one speaker. In the SD-DN approach, the data is again treated as coming from two distinct speakers, allowing for SAT estimation.

As briefly mentioned in Section 4.7.1, Yamagishi et al. [YMK04] found it to be advantageous to consider speaking style as a contextual factor in clustering. This idea can also be applied here by adding “Is the current context Viennese dialect?” to the pool of contextual questions, and adding a variety mark to all the training data.

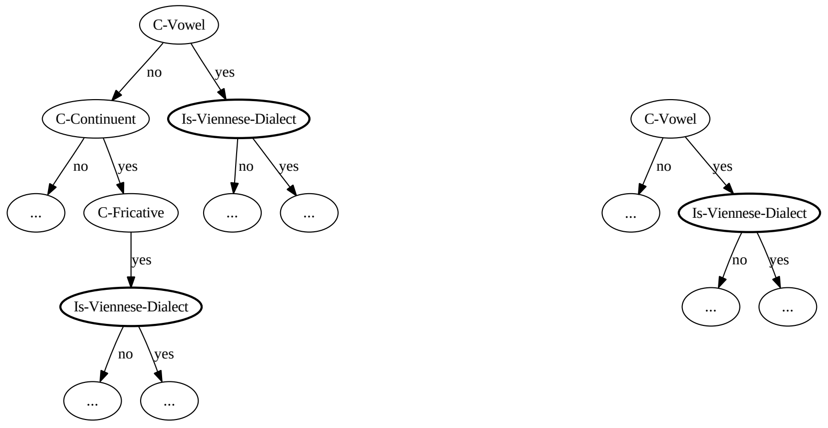
In SD-DC we used this concept, but not SAT normalization, and in SD-DM both the dialect clustering and normalization were applied.

Figure 6.1 shows the tops of the resulting decision trees for the SD-DM approach for mel-cepstral coefficients and state duration. We can see that the variety question is used at high levels in the tree.

			Data Dependency		Dialect	
Name	Target	# utt.	Speaker	Dialect	Clustering	Normalization
SD-DD (AT)	AT	219	✓	✓	×	×
SD-DD (VD)	VD	513	✓	✓	×	×
SI-DD (AT)	AT	5128	×	✓	×	×
SI-DD (VD)	VD	892	×	✓	×	×
SI-DI	AT/VD	6020	×	×	×	×
SI-DN	AT/VD	6020	×	×	×	✓
SD-DI	AT/VD	732	✓	×	×	×
SD-DN	AT/VD	732	✓	×	×	✓
SD-DC	AT/VD	732	✓	×	✓	×
SD-DM	AT/VD	732	✓	×	✓	✓

Table 6.3: Modeling approaches used, listing name, target variety, number of training utterances available, data dependency factors and applied dialect modeling techniques. ✓ means positive, × means negative for each factor.

In the case of MFCCs, different variety models are produced for vowels and fricatives. The duration of vowels is also influenced by variety.



(a) (b)

Figure 6.1: Tops of decision trees from the SD-DM system with dialect clustering for mel-frequency cepstral coefficients (a) and state duration (b).

To summarize, there is a total of 16 voices, 8 Austrian German and 8 Viennese. The two SD-DD systems do not use adaptation at all, the two SI-DD systems are adapted only to the corresponding variety, and the remaining 6 systems listed in Table 6.3 are adapted twice (AT and VD).

6.4 Evaluation of Modeling Approaches

To evaluate the presented systems, a subjective listening test with 40 test subjects was conducted. The goal of the evaluation was to determine one best voice for each of the varieties to carry out further experiments with the two winners.

The web-based evaluation was carried out on-line and consisted of two parts, which are discussed in the following.

6.4.1 Part One: Overall Quality Scores

In the first part, the listeners were presented sentences of synthesized speech and asked to give an overall quality rating on a five point scale, where 5 means “very good” and 1 means “very bad”. Five sentences for AT and five for VD, which were not part of the training data, were selected for this part of the evaluation. As we have 8 voices per variety, this gives rise to $8 \cdot 5 = 40$ synthesized samples per variety. To keep the evaluation procedure from getting too time-consuming, each test subject was not presented all 40 examples, but only 8. Presenting 8 examples to 40 subjects allows the whole 40 samples to be covered 8 times, and the samples were assigned to persons in a pattern based on random Latin squares, in order to spread voices and sentences evenly among the test subjects. The resulting scores are shown in the form of boxplots in Figure 6.2. The data was checked for significant differences using a Mann-Whitney-Wilcoxon test ($p < 0.05$), the results of that are also shown in Figure 6.2.

The results show that for the AT variety, there are three voices that are significantly better than most other voices, namely SI-DN, SI-DD and SI-DI. The differences among these three are not significant. The results for the VD variety are less clear, we only have a clear loser with SI-DD, which is significantly worse than most other voices. From the low performance of the speaker-independent dialect-dependent voice, where the average voice was trained on the limited amount of VD speech data, we may conclude that it is beneficial to include AT data into the training data for the average voice. System SI-DN, for example, does so and reaches much better results.

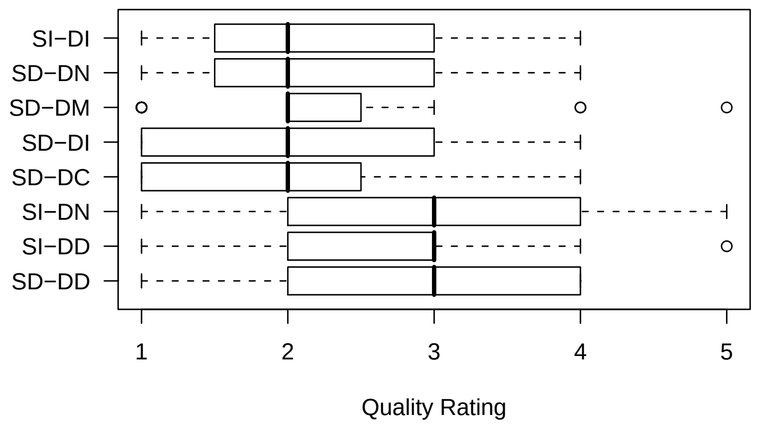
6.4.2 Part Two: Pairwise Comparison

In the second part of the evaluation, the listeners were presented a pair of synthesized sentences and asked which one they preferred. The two samples were always generated from the same input sentence but with different modeling approaches. Varieties were not mixed. Eight voices give rise to $8 \cdot (8 - 1)/2 = 28$ pairs that need to be compared. The same five sentences were used as in the first part, so a total of 140 sound file pairs were to be evaluated per variety. Again, not every subject was presented every pair but only 7. Thus the 140 pairs were each covered two times, and the assignment of pairs to persons again followed a random Latin square pattern. From the total of 560 comparisons (for both varieties), a score vector was computed for each voice in the following way. Voice i has a score s_{ij} for every other voice j from the same variety. This score is computed as $s_{ij} = w_{ij} - l_{ij}$, where $i \neq j$ and w_{ij} and l_{ij} are the numbers of comparisons won and lost, respectively, of voice i against voice j. In other words, for every comparison between voices i and j in which i won, the j-score of i was incremented by 1 and the i-score of j was decremented by 1. Undecided comparisons were not counted. The resulting data is shown as boxplots in Figure 6.3, along with the results of a Mann-Whitney-Wilcoxon test ($p < 0.05$) for significant differences.

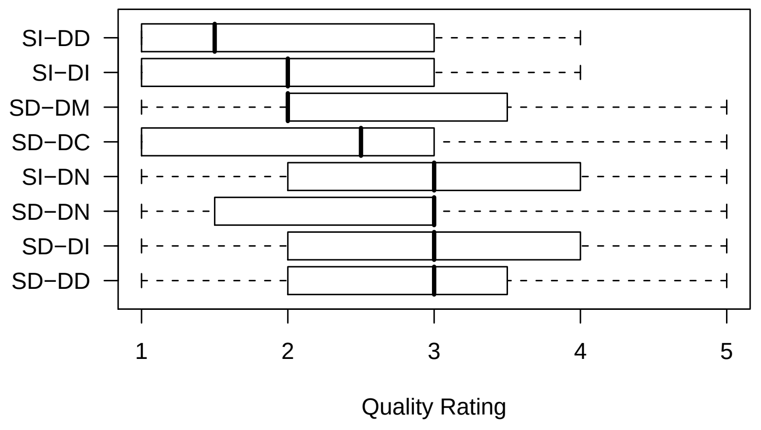
In this test, we can see that in the AT variety voice SI-DN is significantly better than all other voices except SD-DD. Since the speaker-dependent and dialect-dependent SD-DD differs from only two other voices significantly, we can nominate SI-DN as the best approach. In the VD variety, both SD-DD and SD-DI are significantly better than three other voices. The speaker-independent approaches did not perform very well, which could be explained by the relatively large amount of VD speech for target speaker HPO, in comparison to both the AT data for this speaker and to the amount of VD speech from other speakers.

6.4.3 Conclusions

From these results, we chose SI-DN for the AT variety and SD-DD for the VD variety as promising approaches. The fact that a dialect-independent voice won for AT is an interesting result: although we have enough data for AT speech and in particular a lot more than for VD speech, using data from both varieties simultaneously leads to better results (compare SI-DN to SI-DD in the figures). Furthermore, we can see that the best training strategy for dialect-independent voices is to split the

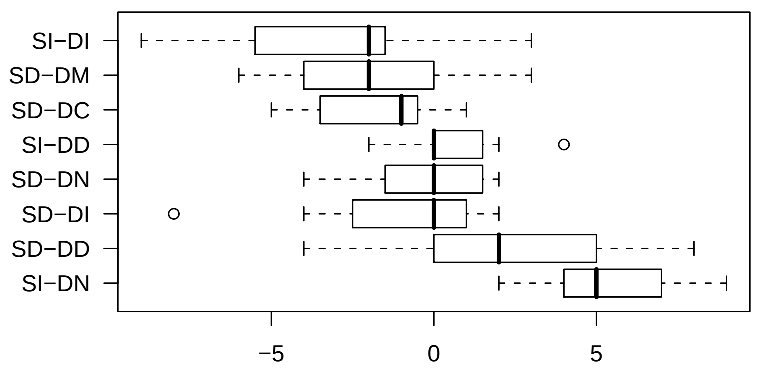


SI-DI	SD-DN	SD-DM	SD-DI	SD-DC	SI-DN	SI-DD	SD-DD	
					✓	✓	✓	SI-DI
				✓	✓			SD-DN
					✓	✓	✓	SD-DM
					✓	✓	✓	SD-DI
					✓	✓	✓	SD-DC
								SI-DN
								SI-DD
								SD-DD

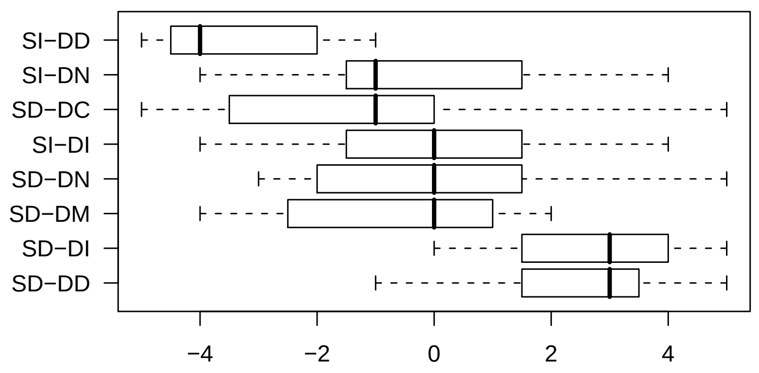


SI-DD	SI-DI	SD-DM	SD-DC	SI-DN	SD-DN	SD-DI	SD-DD	
		✓	✓	✓	✓	✓	✓	SI-DD
								SI-DI
								SD-DM
								SD-DC
								SI-DN
								SD-DN
								SD-DI
								SD-DD

Figure 6.2: Boxplots and significant differences for the overall quality ratings, for the AT (top) and VD (bottom) varieties. On the horizontal axis, 5 means “very good” and 1 means “very bad”. Significance was tested for using a Mann-Whitney-Wilcoxon test ($p < 0.05$).



SI-DI	SD-DM	SD-DC	SI-DD	SD-DN	SD-DI	SD-DD	SI-DN	
			✓			✓	✓	SI-DI
							✓	SD-DM
						✓	✓	SD-DC
							✓	SI-DD
							✓	SD-DN
							✓	SD-DI
								SD-DD
								SI-DN



SI-DD	SI-DN	SD-DC	SI-DI	SD-DN	SD-DM	SD-DI	SD-DD	
	✓		✓	✓	✓	✓	✓	SI-DD
								SI-DN
						✓	✓	SD-DC
								SI-DI
								SD-DN
						✓	✓	SD-DM
								SD-DI
								SD-DD

Figure 6.3: Boxplots and significant differences for the pairwise comparison. In the boxplots, the data for one voice i consists of seven scores $s_j = w_{ij} - l_{ij}$, where $j \neq i$ and w_{ij} and l_{ij} are the numbers of comparisons won and lost, respectively, of voice i against voice j. Significance was tested for using a Mann-Whitney-Wilcoxon test ($p < 0.05$).

training data of a speaker according to variety and treat the two sets as separate speakers, to enable speaker-adaptive training to normalize dialectal differences (compare SI-DN to SI-DI in the figures).

Unfortunately, the dialect clustering approach (SD-DC and SD-DM) is not very successful, although the generated decision trees look promising. One possible reason is the limited amount of training data, especially with regard to the AT-VD balance. Experiments with more balanced data sets could produce different results.

6.5 Dialect Interpolation

As discussed in Section 4.7.1, the HMM-based speech synthesis approach allows for interpolation between speakers or speaking styles, for example emotional speech. We would like to apply this methodology to a dialect interpolation between Austrian German and Viennese Dialect, to give users of TTS systems control over how close to the standard or to a dialectal variety the synthetic speech should be.

A major premise for this kind of dialect control is that language varieties form a continuous space [PSYN09]. The dialects of a language are related to one another in the sense of being linguistically close, which enables us to think of a continuum of varieties to lie between two varieties in general. There can be clear exceptions to this, however. From phonetic studies of the Viennese sociolects [Moo87], we know that some transitions of phones act as sociolect markers and are categorical. A speaker produces either the standard form of a phone or the dialect form, whitout the possibility of in-between variants. In such a case it is not appropriate to apply interpolation. A dialect interpolation system should therefore allow us to keep individual phones from being interpolated, depending on linguistic knowledge.

From this perspective, the differences between Austrian German and Viennese dialect that were discussed in Section 5.2 have been divided into three groups [PSYN09]:

1. **Minor shifts** that are phonetically close. This includes tension of vowels (e.g., /ɪ/ → /i/ and /ɔ/ → /o/), monophthongization (e.g., /ɔ͜y/ → /æː/) as well as neutralization and lenition of plosives (e.g., /t/ → /d/ and /b/ → /β/). These differences are gradual in a phonetic sense [Moo87] and are thus perfectly suitable for interpolation.

2. **Phonologically manifested differences** which are attributed to an “input switch” between standard and dialect or differences that involve different phonological processes. This includes certain instances of /l/-vocalization (e.g., /ɔl/ → /oɪ/) and vowel shifts (e.g., /aː/ → /ɑː/). For this group, interpolation is unproblematic from a technical point of view, but might not be linguistically meaningful.

3. **Differences affecting the segmental structure**, i.e., insertions or deletions of phones. This includes other instances of /l/-vocalization (e.g., /iːl/ → /yː/) and schwa-elisions (e.g., /gəv/ → /gv/). Since the segmental structure changes in this group, straightforward interpolation is not possible.

6.5.1 Phonological Constraints for HMM Interpolation

Moving further towards a technical view of things, we can interpret the three groups from the last section as phonological constraints on how interpolation should be applied.

As mentioned, the first group is the least problematic. We can straightforwardly apply interpolation of the output and duration distributions to each pair of phone HMMs. For example:

$$\begin{tikzcd}[row sep=small]
\textbf{AT} & \text{ɔ} \arrow[d] & \text{f} \arrow[d] & \text{n̩} \arrow[d] \\
\textbf{VD} & \text{o} & \text{f} & \text{m̩}
\end{tikzcd}$$

For the second group, in-between variants might not be meaningful, therefore the respective phones should not be interpolated. Rather, one or the other variant should be realized, as determined by thresholding on the interpolation ratio (0.5 in our experiments). For example:

AT	ʃ	l	aː	k
	↓	↓	×	↓
VD	ʃ	l	ɑː	g̥

For the third group the segmental structure of the word changes, which leads to a different number of phones. For this reason, we introduce null phones, depicted by /[]/, which are inserted in order to align the two phone strings appropriately. The null phone represents the deleted phone with zero duration. For example:

AT	g	ə	v	ɪ	ç	t
	↓	↓	↓	↓	↓	↓
VD	g	[]	v	ɪ	ç	t

6.5.2 Dialect Interpolation Implementation Issues

To simplify interpolation on the level of HMMs, we assume all phone HMMs to have the same number of states and the same output features, as well as state duration to be modeled by explicit duration distributions. The input to the interpolation are the two sequences of phones (with phonetic contexts) representing the extreme points (pure AT or VD). In our experiments, we assumed the input to already contain null phones where necessary, such that no phone alignment needs to be done. In practice, automatic alignment could be realized based on the linguistic knowledge, or the lexica could already contain null phones.

For the actual interpolation we chose *interpolation among observations*, the first and simplest method described by Yoshimura et al. [YTM$^+$00] (see also Section 4.7.1). This method was also used for the emotional speech interpolation experiments by Tachibana et al. [TYMK05].

From the examples above it should be clear that we cannot perform off-line interpolation, i.e., provide already interpolated voices at different interpolation ratios to synthesize from. Rather, the interpolation must be carried out on-line, in the synthesis process.

The procedure for interpolated synthesis is the following: The input are two sequences of phones, including all contextual information and null phones where appropriate, and an interpolation ratio. For each phone in the utterance, the respective models are loaded from the AT and VD voice. Then the two observation pdfs and the duration pdfs are interpolated. If one of the two is a null phone, then the output pdf of the other phone is used for both, and the duration of the null phone is set to a pseudo-pdf with mean 0 and variance 0. Thus, during interpolation, the spectral and excitation parameters of the phone stay the same, while duration is gradually reduced until the phone disappears.

As mentioned, it is sometimes not linguistically meaningful to interpolate. For these cases, we apply input switch rules. Together with the contextual information for the current phone, a threshold for the input switch is provided as input (0.5 in all our experiments). If such a threshold is given, and the interpolation ratio is below it, the current phone is not interpolated but rather the original lower extreme point model is used, as if the interpolation ratio were 0.0. If the interpolation ratio exceeds the threshold, the other extreme point (1.0) is used. Note that this is done phone by phone, so for neighboring phones it is possible that one is interpolated and the other is not.

When all pdfs are determined, speech is synthesized in the normal way using the interpolated pdfs.

To summarize, the following modifications were made to the synthesis frontend (hts_engine [ZTO09]):

- A *pair* of input label files is accepted, one containing the AT transcription and the other the VD transcription.
- For each phone, the two context-dependent models are loaded (from different locations in different decision trees).
- Linear interpolation is applied to the output pdfs for spectrum, pitch, duration and aperiodicity with a given interpolation ratio.
- The input can contain an interpolation threshold, whose presence suppresses interpolation. Instead, one of the extreme points is used, depending on the threshold (and the interpolation ratio).

- The input can contain special *null* phones. In these cases interpolation is applied to duration only, otherwise the parameters of the other, non-null phone are used.

6.5.3 Interpolated Examples

Figures 6.4 through 6.7 show spectrograms of interpolated examples[1] between the AT variety (top) and the VD variety (bottom). The rows correspond to interpolation ratios of 0.0, 0.2, 0.4, 0.6, 0.8 and 1.0, respectively.

Figure 6.4 shows the continuous transformation of /ɔ͜ʏ/ into /æː/. Interestingly, the 0.6 stage was categorized as "undefined" ("?" in the figure) by speech experts listening to the samples [PSYN09]. In Figure 6.5, on the other hand, an input switch was applied for that phone at 0.5, i.e., the AT model for /ɔ͜ʏ/ is used in the first three rows, and the VD model for /æː/ is used in the last three rows. All other phones are interpolated. This results in appropriate categorical transition of phones. Note, however, that even for the not interpolated phone, some change is visible from 0.0 to 0.4 and from 0.6 to 1.0. This is a result of taking dynamic features into account (compare Section 4.3), meaning that neighboring models have an influence on each others output.

Figure 6.6 shows another example of interpolation without switching rules, where /aː/ is transformed into /ɔː/. Even though the linguistics literature tells us that interpolation in this case is unnatural (i.e., people do not produce intermediate variants but realize either of the extreme points), from a technical standpoint the interpolation creates a smooth transition.

Figure 6.7 shows an example including a null phone, in this case a schwa-elision. It can be seen clearly how the /ə/ gradually disappears. Where the schwa segment gets really short, the speech experts judged the sample to sound somewhat unnatural.

6.6 Evaluation of Dialect Interpolation

To evaluate the dialect interpolation techniques described in the last section, again a listener test with 40 subjects was conducted. A carrier sentence was selected whose empty place was filled with words that instantiate some representative phonological processes discussed earlier. The carrier sentence "Und mit ... bitte (and with ... please)" has the following transcriptions in the two varieties:

AT	ʊ	n	t	m	ɪ	t	...	b	ɪ	t	ə
	↓	↓	↓	↓	↓	↓		↓	↓	↓	↓
VD	u	n	t	m	i	t	...	b	i	t	ɛ

This sentence was chosen in order to have only few and small differences in the fixed part, such that the focus is on the process instantiated in the respective word in the middle. The words were the following:

[1] These examples can be downloaded from `http://dialect-tts.ftw.at`

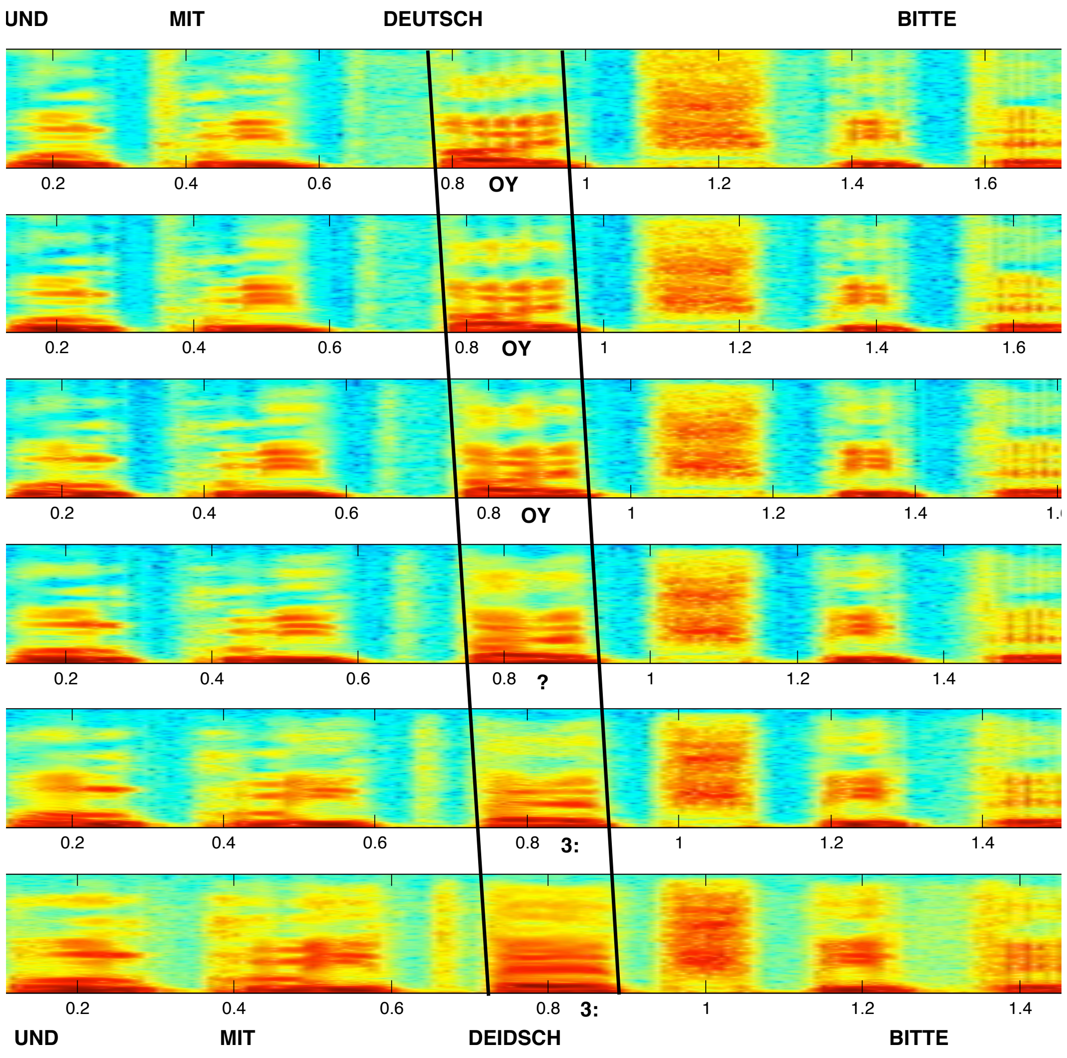


Figure 6.4: Spectrogram of an interpolation example between Austrian German (top) and Viennese dialect (bottom) for the sentence “Und mit Deutsch bitte (And with German please)”. No switching rules were applied, the interpolation ratio increments by 0.2 from one row to the next. “OY” represents /ɔ͜ʏ/ and “3:” represents /æː/.

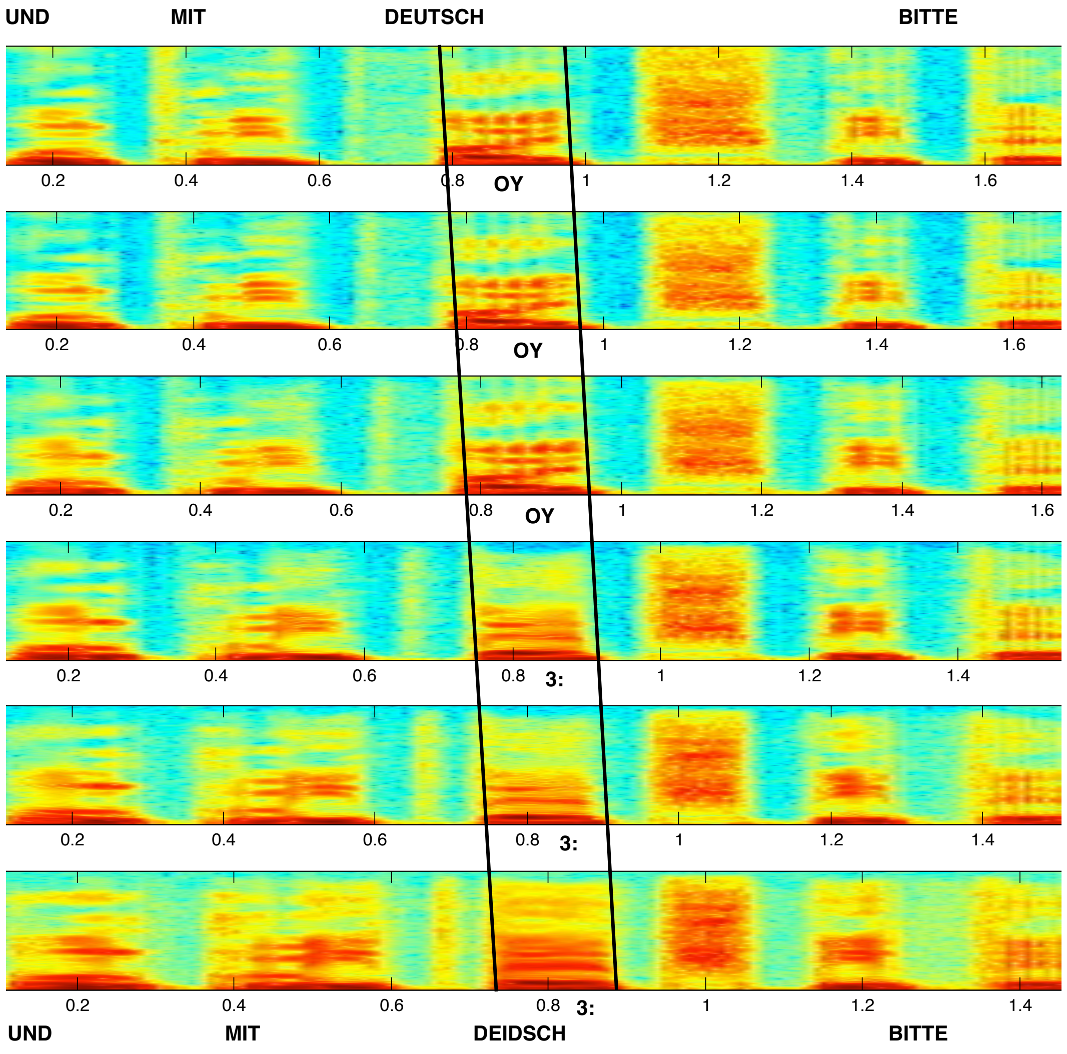


Figure 6.5: Spectrogram of an interpolation example between Austrian German (top) and Viennese dialect (bottom) for the sentence “Und mit Deutsch bitte (And with German please)” with switching rules applied. The interpolation ratio increments by 0.2 from one row to the next. “OY” represents /ɔʏ̯/ and “3:” represents /æː/.

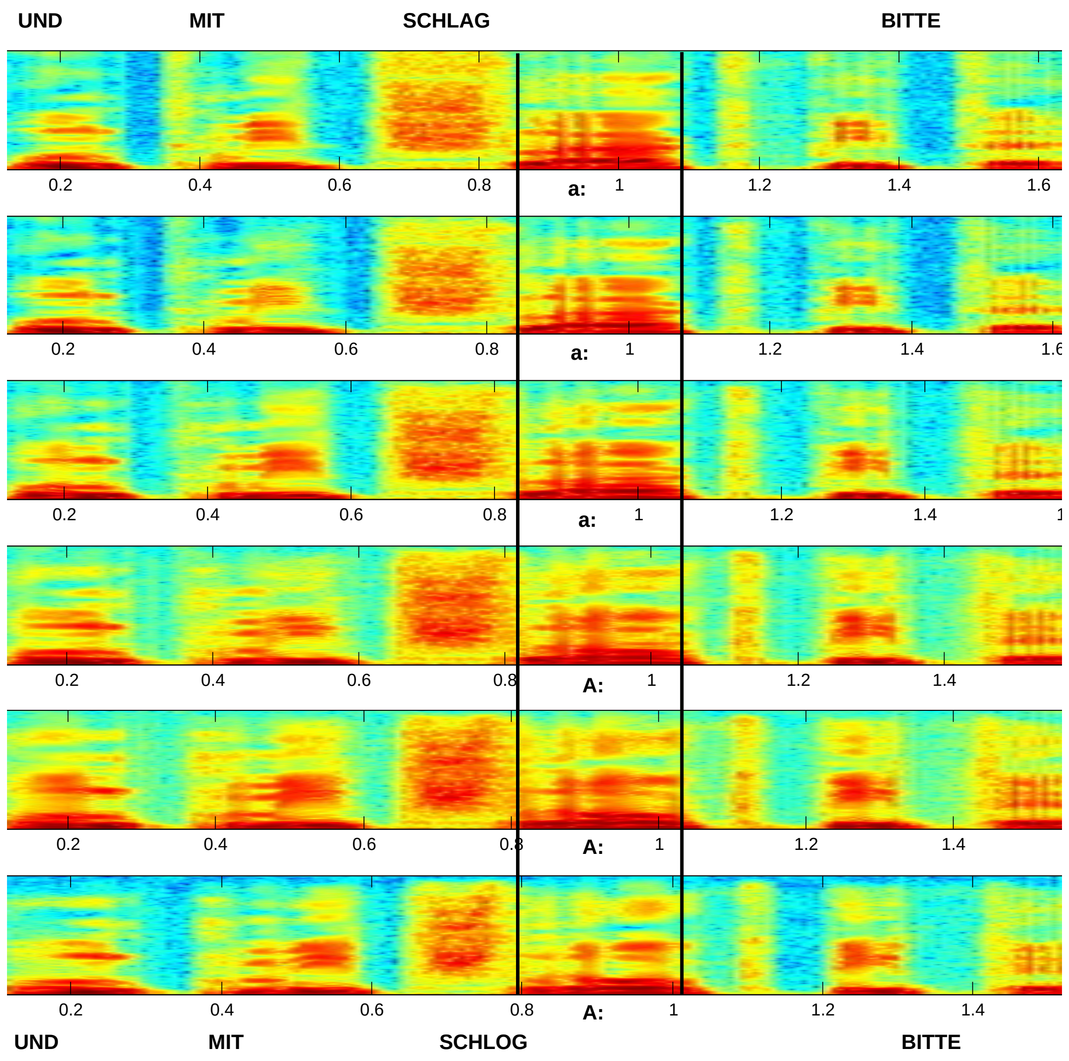


Figure 6.6: Spectrogram of an interpolation example between Austrian German (top) and Viennese dialect (bottom) for the sentence “Und mit Schlag bitte (And with beat please)”. No switching rules were applied, the interpolation ratio increments by 0.2 from one row to the next. “a:” represents /aː/ and “A:” represents /ɔː/.

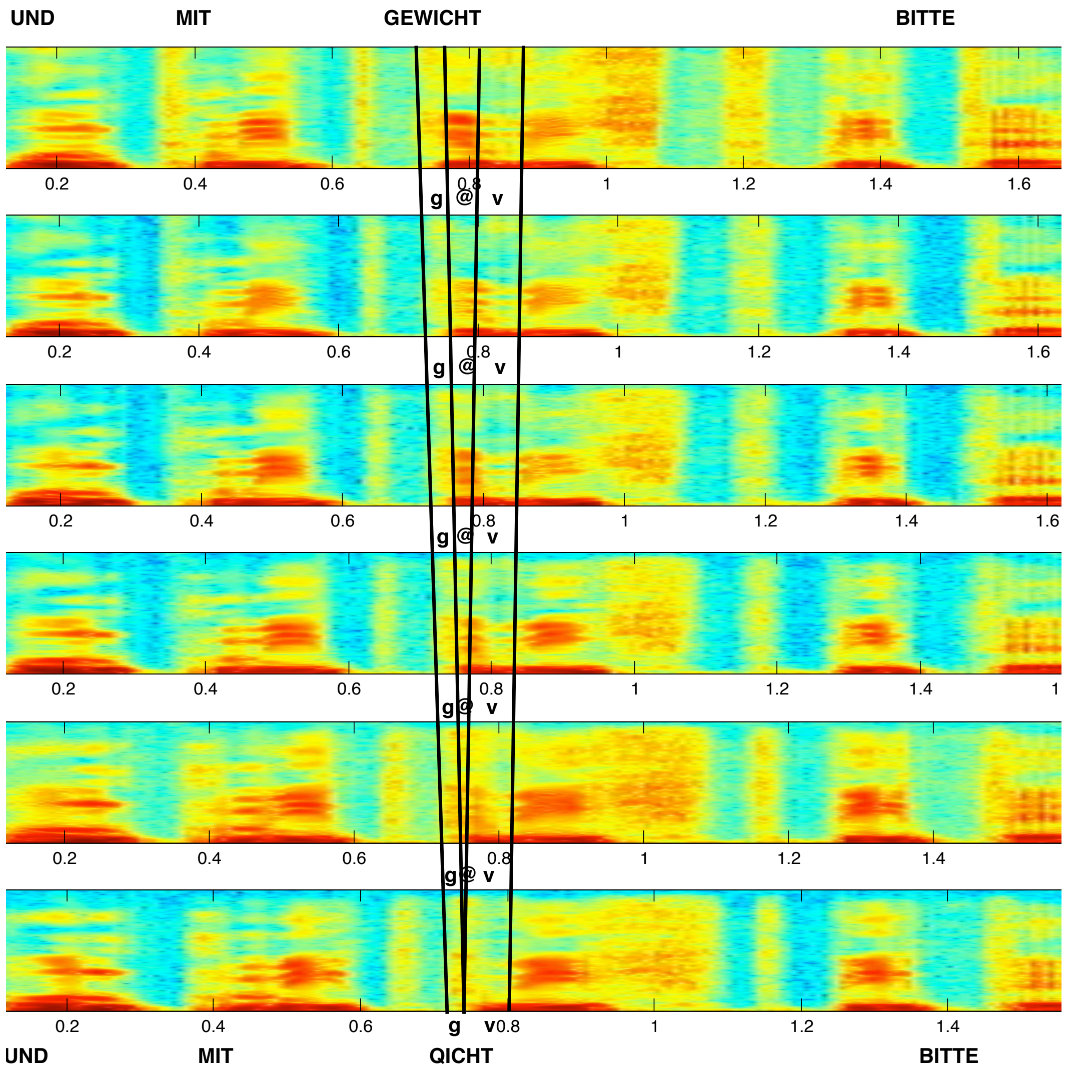


Figure 6.7: Spectrogram of an interpolation example between Austrian German (top) and Viennese dialect (bottom) for the sentence “Und mit Gewicht bitte (And with weight please)”. No switching rules were applied, the interpolation ratio increments by 0.2 from one row to the next. “g@v” represents /gəv/.

Word	Gloss	AT/VD	Process
Leber	liver	l eː b ɐ ↓ ↓ ↓ ↓ l eː β ɐ	lenition of lenis plosives
Holz	wood	h ɔ l t s ↓ ↓ ↓ ↓ ↓ h o ɪ t s	/l/-vocalization
Milch	milk	m ɪ l ç ↓ ↓ ↓ ↓ m yː [] ç	/l/-vocalization
Gewicht	weight	g ə v ɪ ç t ↓ ↓ ↓ ↓ ↓ ↓ g [] v ɪ ç t	schwa-elision
Hände	hands	h ɛ n d ə ↓ ↓ ↓ ↓ ↓ h ɛ n d̥ []	schwa-elision
Deutsch	German	d ɔ͜ʏ t ʃ ↓ ↓ ↓ ↓ d æː t ʃ	monophthongization
Schlag	beat	ʃ l aː k ↓ ↓ ↓ ↓ ʃ l ɔː g̊	vowel shift
Keller	cellar	k ɛ l ɐ ↓ ↓ ↓ ↓ k œː l ɒ	vowel shift

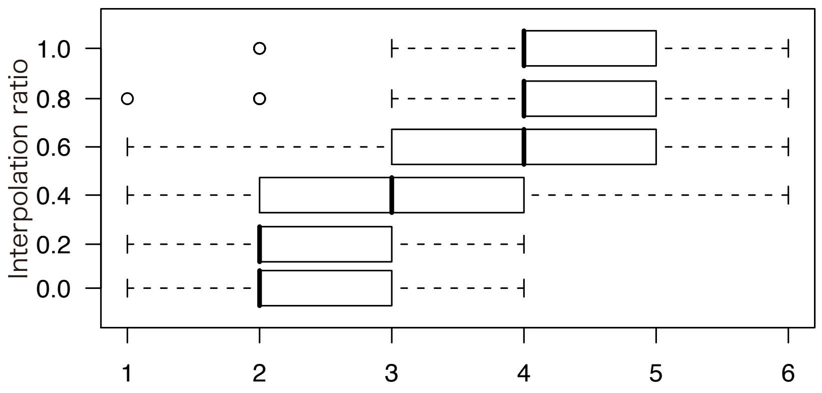
The evaluation was similar to the one carried out with the modeling approaches (see Section 6.4) and again consisted of two parts. For both parts, the listening examples were synthesized from the above eight words embedded in the carrier sentence. For each sentence, ten different interpolation ratios 0.0, 0.2, 0.2*s*, 0.4, 0.4*s*, 0.6, 0.6*s*, 0.8, 0.8*s* and 1.0 were used, where an *s* denotes that switching rules were applied to the central process, and 0.0 and 1.0 are the non-interpolated, original varieties.

6.6.1 Part One: "Vienneseness"

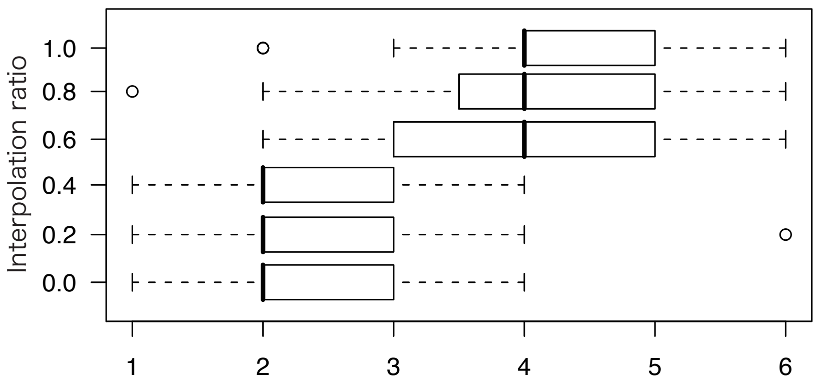
In the first part, the listeners were presented examples from the pool described above and asked to give a rating from 1 to 6, where 1 meant strong Viennese dialect and 6 meant high Austrian standard. As in the evaluation of modeling approaches, each subject was not presented all 80 samples but only 16. The samples were assigned to persons based on random Latin squares to ensure even distribution. The overall results (all sentences collectively) are shown in Figure 6.8 for interpolation without switching rules (top) and with switching rules applied (bottom).

The general picture is that a gradual change was perceived for the interpolation without switching rules and a categorical change was perceived with the switching rules added. Without switching rules, all differences are significant ($p < 0.05$) except $0.0 \leftrightarrow 0.2$ and $0.8 \leftrightarrow 1.0$, in particular, for example, along the path $0.2 \leftrightarrow 0.4$, $0.4 \leftrightarrow 0.6$, $0.6 \leftrightarrow 0.8$. With switching rules applied, the significance test splits the data nicely into two categories, 0.0, 0.2 and 0.4 on the one hand, and 0.6, 0.8 and 1.0 on the other hand. Within each of the two categories, there are no significant differences, but all inter-categorical differences are significant.

Figure 6.9 shows the results of the "Vienneseness" ratings for three sample utterances instantiating different phonological processes, monophthongization ((a) and (b)), vowel shift ((c) and (d)) and schwa-elision ((e) and (f)). We can clearly see the different behavior of these processes as dialect markers. In the case of monophthongization, a relatively continuous transition from dialect to standard is perceived in both settings. The vowel shift process generates a continuum in the setting without switching rules which does not match real phenomena, and with the switching rules a categorical shift is perceived. In the case of schwa-elision, a categorical shift is perceived at a certain point, regardless

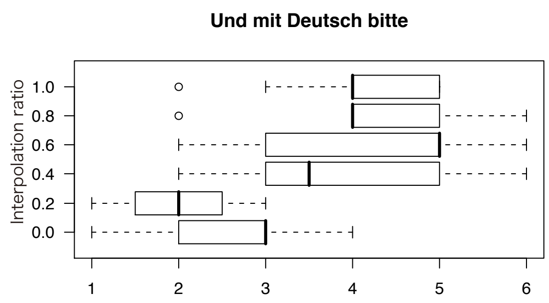


0.0	0.2	0.4	0.6	0.8	1.0	
		✓	✓	✓	✓	0.0
		✓	✓	✓	✓	0.2
			✓	✓	✓	0.4
				✓	✓	0.6
						0.8
						1.0

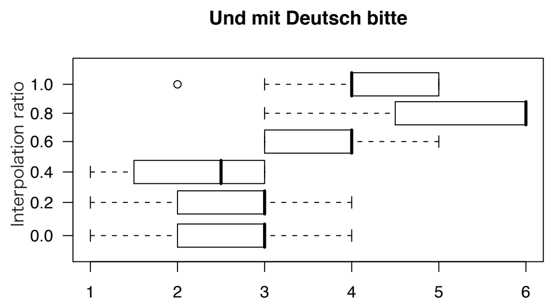


0.0	0.2	0.4	0.6	0.8	1.0	
			✓	✓	✓	0.0
			✓	✓	✓	0.2
			✓	✓	✓	0.4
						0.6
						0.8
						1.0

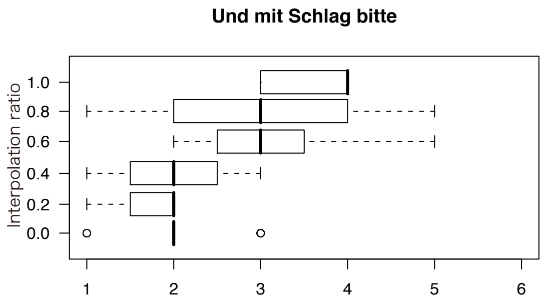
Figure 6.8: Boxplots and significant differences for the "Vienneseness" ratings, HMM interpolation only (top) and HMM interpolation plus switching rules (bottom). On the horizontal axis, 1 means strong Viennese dialect and 6 means high Austrian standard. Significance was tested for using a Mann-Whitney-Wilcoxon test ($p < 0.05$).



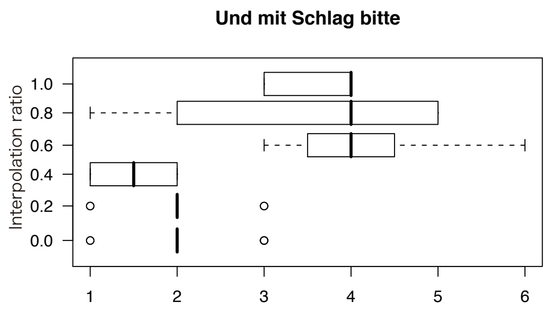
(a) HMM interpolation



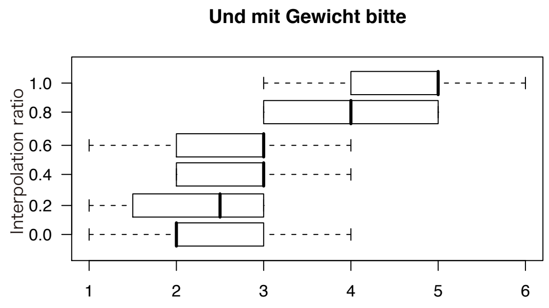
(b) HMM interpolation + switching rules



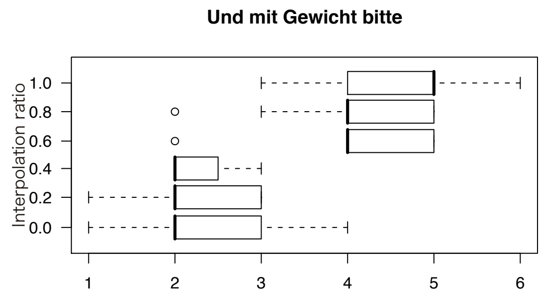
(c) HMM interpolation



(d) HMM interpolation + switching rules



(e) HMM interpolation



(f) HMM interpolation + switching rules

Figure 6.9: Boxplots for three different utterances chosen from three categories. On the horizontal axis, 1 means strong Viennese dialect, 6 means high Austrian standard.

of the use of input switch rules. This means that a categorical change is perceived even though there is a continuous interpolation of the signal in this case.

6.6.2 Part Two: Pairwise Similarity

In the second part of the evaluation, the listeners were presented pairs of synthesized samples and asked to judge the similarity of the two in terms of dialect on a scale from 1 “very similar” to 5 “very different”. The same sentences as for the “Vienneseness” ratings were used, in a pair the two utterances were generated from the same input sentence but with different interpolation settings (0.0, 0.2, 0.2s, ..., 0.8, 0.8s, 1.0). The results of this evaluation part are visualized using multi-dimensional scaling [CC01] in Figure 6.10.

Even from this figure, we can confirm several findings mentioned before. The HMM interpolation generates a continuum of dialectal varieties corresponding to the horizontal axis. On the other hand, when we apply switching rules, the samples below the switching threshold get clustered on the left side and the ones above the threshold get clustered on the right side of the graph. There is a big gap between 0.4 and 0.6 with switching rules applied at a threshold of 0.5.

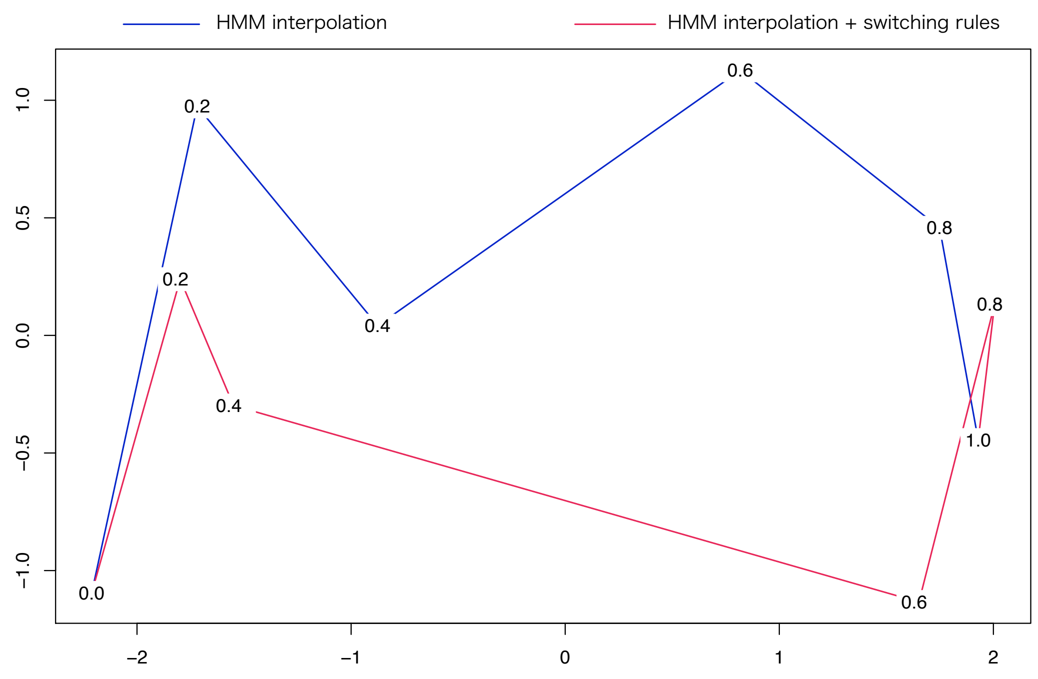


Figure 6.10: Evaluation of similarity in terms of dialect, visualized using multi-dimensional scaling.

The additional information we gain through the second dimension is related to the switching rules. The vertical axis allows for distances between switched and non-switched samples of the same interpolation ratio. We see a large gap between the switched and the non-switched version of 0.6, indicating that these samples are perceived very differently in terms of dialect.

Chapter 7

Conclusion

7.1 Summary

This work has presented hidden Markov model based speech synthesis and its application to Austrian German (AT) and Viennese dialect/sociolect (VD). In this approach, speech signals are generated in an analysis–re-synthesis manner: Suitable features (mel-frequency cepstral coefficients, fundamental frequency, phone duration) are extracted together with their dynamic features from a corpus of training data (annotated speech recordings). These features are then used to train context-dependent HMMs for phone-sized speech units. Context clustering based on phonetic decision trees addresses the problem of sparse covering of all possible contexts. Speech is then synthesized by concatenating context-dependent phone HMMs to an utterance HMM, from which the most likely observation sequence is generated, taking the dynamic features into account. A speech signal is created from the observations following a source–filter approach.

As the fundamental frequency is not applicable to unvoiced speech segments, F_0 is modeled with multi-space distributions (MSD). State durations are modeled explicitly with duration densities rather than through self-transition probabilities, thereby relaxing the Markov property (hidden *semi*-Markov models), leading to MSD-HSMMs.

Average voices can be trained from a multi-speaker database using the speaker adaptive training (SAT) algorithm to normalize the influence of speaker differences. An average voice can then be adapted to a target speaker using a very small amount of data.

Parameter interpolation allows synthesizing speech with characteristics in between those used for training different synthetic voices. This allows gradual changes between different speakers or speaking styles and emotions.

The interpolation techniques have been extended in this thesis to interpolation among dialectal varieties. To deal with changes in the segmental structure (insertions and deletions) which arise from phonological processes between Austrian standard German and Viennese dialect/sociolect, null phones were introduced to allow interpolation. Since it is not always linguistically justified to interpolate, the concept of input switch rules was incorporated in the interpolation technique, such that interpolation can be turned off for certain phones in an utterance.

Different modeling approaches for the two varieties (AT and VD) were pursued and evaluated in a test user listening study. It turned out that the outcome of the various modeling approaches was quite different for AT and VD, presumably due to the imbalance regarding amount and quality between the training sets.

The best voice of each variety was selected for an interpolation experiment between AT and VD. Again, the outcome was evaluated in a test user listening study. It could be shown that the test subjects did perceive gradual changes in terms of dialect, and thus that the interpolation experiments were a success. The switching rules helped to retain the (sometimes more appropriate) categorical switch, while still allowing the rest of the utterance to be interpolated.

7.2 Future Work

Some of the modeling approaches were less successful than expected, especially the dialect clustering technique, which showed promising results in the context clustering trees, but did not perform well in the perceptive evaluations. As more speech data becomes available, these experiments should be repeated.

For the dialect interpolation, the alignment of the utterances in the two varieties was assumed to be given, i.e., the input was assumed to already contain null phones where necessary. For the experiments, the alignment was manually established. This could be automatized using at least three distinct strategies: The simplest approach would be to provide a multi-variety lexicon whose entries already contain null phones. Secondly, a knowledge-based system could apply its information on the possible phonological processes to find a "justifiable" alignment between two utterances and insert null phones where appropriate. Finally, we could align the utterance HMMs rather than the phone strings, by means of a distance metric for HMMs and dynamic programming.

Furthermore, the interpolation techniques could be applied to more varieties, e.g., additional German dialects or varieties of other languages.

Appendix A

List of IPA Symbols for German

This appendix presents a list of the symbols of the international phonetic alphabet (IPA) that are needed to transcribe German speech. For each symbol, some German example words are given [wik09].

A.1 Vowels

	Symbol	Examples
Monophthongs	[aː]	Ameise /[ˈaːmaɪ̯zə]/, Vater /[ˈfaːtɐ]/
	[a]	Schatten /[ˈʃatn̩]/, wann /[van]/
	[ɐ]	Wasser /[ˈvasɐ]/, blechern /[ˈblɛçɐn]/
	[ɐ̯]	Fahrt /[faːɐ̯t]/, leer /[leːɐ̯]/, Bär /[bɛːɐ̯]/, wir /[viːɐ̯]/
	[eː]	geben /[ˈɡeːbm̩]/, ewig /[ˈeːvɪç]/
	[e]	in foreign words only: egal /[eˈɡaːl]/
	[ɛː]	Käse /[ˈkɛːzə]/, ähnlich /[ˈɛːnlɪç]/
	[ɛ]	Hecke /[ˈhɛkə]/, Bäcker /[ˈbɛkɐ]/, Ende /[ˈɛndə]/
	[ə]	bitte /[ˈbɪtə]/, Kanne /[ˈkanə]/
	[iː]	Knie /[kniː]/, Vieh /[fiː]/
	[i̯]	in foreign words only: radial /[ʀaˈdi̯aːl]/, Gradient /[ɡʀaˈdi̯ɛnt]/
	[i]	in foreign words only: motiviert /[motiˈviːɐ̯t]/
	[ɪ]	mit /[mɪt]/, Tisch /[tɪʃ]/
	[oː]	Bote /[ˈboːtə]/, Ofen /[ˈoːfn̩]/, roh /[ʀoː]/
	[o]	in foreign words only: Phonologie /[ˌfonoloˈɡiː]/, Motiv /[moˈtiːf]/
	[ɔ]	Gott /[ɡɔt]/, Post /[pɔst]/, offen /[ˈɔfn̩]/
	[øː]	schön /[ʃøːn]/, gewöhnlich /[ɡəˈvøːnlɪç]/
	[œ]	können /[ˈkœnən]/, öffentlich /[ˈœfn̩tlɪç]/
	[uː]	Blut /[bluːt]/, Kuhle /[ˈkuːlə]/
	[u]	in foreign words only: brutal /[bʀuˈtaːl]/
	[ʊ]	Schmuck /[ʃmʊk]/, Luft /[lʊft]/
	[yː]	über /[ˈyːbɐ]/, kühl /[kyːl]/, Typ /[tyːp]/
	[y]	in foreign words only: Physik /[fyˈsɪk]/
	[ʏ]	dünn /[dʏn]/, lüften /[ˈlʏftn̩]/, Symbol /[zʏmˈboːl]/
Diphthongs	[aɪ̯]	Leim /[laɪ̯m]/, Mais /[maɪ̯s]/
	[aʊ̯]	Haus /[haʊ̯s]/, Auto /[ˈaʊ̯to]/
	[ɛɪ̯]	in foreign words only: Lady /[ˈlɛɪ̯di]/, E-Mail /[ˈiːˌmɛɪ̯l]/, hey /[hɛɪ̯]/
	[ɔɪ̯]	Heu /[hɔɪ̯]/, Läufer /[ˈlɔɪ̯fɐ]/, neu /[nɔɪ̯]/
	[ʊɪ̯]	pfui /[pfʊɪ̯]/, Uigure /[ʊɪ̯ˈɡuːʀə]/, Ratatouille /[ʀataˈtʰʊɪ̯]/

A.2 Consonants

Symbol	Examples
[ʔ]	Theater /[teˈʔaːtɐ]/, beantworten /[bəˈʔantvɔʁtn̩]/
[b]	Biene /[ˈbiːnə]/, aber /[ˈaːbɐ]/
[ç]	ich /[ɪç]/, Pech /[pɛç]/
[d]	dann /[dan]/, Laden /[ˈlaːdn̩]/
[f]	Vogel /[ˈfoːgl̩]/, Physik /[fyˈzɪk]/, Wolf /[vɔlf]/
[g]	gehen /[ˈgeːən]/, Lager /[ˈlaːgɐ]/
[h]	Haus /[haʊs]/, Hund /[hʊnt]/
[j]	jung /[jʊŋ]/, Boje /[ˈboːjə]/
[k]	Katze /[ˈkatsə]/, Schreck /[ʃʀɛk]/, Qualm /[kvalm]/, sechs /[zɛks]/
[l]	Lamm /[lam]/, Öl /[øːl]/, Ball /[bal]/
[l̩]	schwindeln /[ˈʃvɪndl̩n]/, Fusel /[ˈfuːzl̩]/
[m]	Maus /[maʊ̯s]/, am /[am]/, Damm /[dam]/
[m̩]	großem /[ˈgʀoːsm̩]/, geben /[ˈgeːbm̩]/
[ɱ̩]	Ofen /[ˈoːfɱ̩]/, klopfen /[ˈklɔpfɱ̩]/
[n]	Nord /[nɔʁt]/, Land /[lant]/, Kanne /[ˈkanə]/
[n̩]	großen /[ˈgʀoːsn̩]/, Laden /[ˈlaːdn̩]/
[ŋ]	lang /[laŋ]/, singen /[ˈzɪŋən]/
[ŋ̩]	wiegen /[ˈviːgŋ̩]/, Augen /[ˈaʊ̯gŋ̩]/
[p]	Papier /[paˈpiːɐ̯]/, Mappe /[ˈmapə]/
[ʀ]	rot /[ʀoːt]/, drei /[dʀaɪ̯]/, Fahrer /[ˈfaːʀɐ]/
[ʁ]	dort /[dɔʁt]/, wird /[vɪʁt]/
[s]	Last /[last]/, Fass /[fas]/, sechs /[zɛks]/
[ʃ]	Schule /[ˈʃuːlə]/, Stier /[ʃtiːɐ̯]/, Spur /[ʃpuːɐ̯]/
[t]	Tag /[taːk]/, Vetter /[ˈfɛtɐ]/, Bett /[bɛt]/
[v]	Winter /[ˈvɪntɐ]/, Vase /[ˈvaːzə]/, Löwe /[ˈløːvə]/, Qualm /[kvalm]/
[χ]	lachen /[ˈlaχn̩]/, Dach /[daχ]/, Bucht /[bʊχt]/, doch /[dɔχ]/
[z]	sechs /[zɛks]/, Wiese /[ˈviːzə]/, Glaser /[ˈglaːzɐ]/
[ʒ]	in foreign words only: Garage /[gaˈʀaːʒə]/, Gendarmerie /[ʒãdaɐ̯məˈʀiː]/

Appendix B

Festival Utterance File

```
EST_File utterance
DataType ascii
version 2
EST_Header_End
Features max_id 32 ; type Text ; iform "\"Die Sonne lacht.\"" ; fileid BERLIN_002 ; filename utt/BERLIN_002.utt ;
Stream_Items
1 id _1 ; name Die ; whitespace "" ; prepunctuation "" ;
2 id _2 ; name Sonne ; whitespace " " ; prepunctuation "" ;
3 id _3 ; name lacht ; punc . ; whitespace " " ; prepunctuation "" ;
4 id _6 ; name lacht ; pbreak NB ;
5 id _7 ; name . ; pbreak BB ; bad_lex lts ;
6 id _5 ; name Sonne ; pbreak NB ;
7 id _4 ; name Die ; pbreak NB ;
8 id _8 ; name BB ;
9 id _9 ; end 0.042 ; name sil ; score -75.092 ; start F:standard+unisyn_start ;
10 id _11 ; end 0.132 ; name d ; score -77.8543 ; cl_end 0.048 ; start F:standard+unisyn_start ;
11 id _12 ; end 0.182 ; name schwa ; score -63.9348 ; start F:standard+unisyn_start ;
12 id _14 ; end 0.328 ; name z ; score -70.0005 ; start F:standard+unisyn_start ;
13 id _16 ; end 0.452 ; name O ; score -66.8023 ; start F:standard+unisyn_start ;
14 id _18 ; end 0.514 ; name n ; score -75.2894 ; start F:standard+unisyn_start ;
15 id _19 ; end 0.6 ; name schwa ; score -83.9436 ; start F:standard+unisyn_start ;
16 id _21 ; end 0.676 ; name l ; score -81.0053 ; start F:standard+unisyn_start ;
17 id _23 ; end 0.77 ; name a ; score -80.7124 ; start F:standard+unisyn_start ;
18 id _24 ; end 0.908 ; name ch ; score -71.8782 ; start F:standard+unisyn_start ;
19 id _26 ; end 1.046 ; name t ; score -80.4962 ; cl_end 0.93 ; start F:standard+unisyn_start ;
20 id _28 ; end 1.176 ; name sil ; score -71.5593 ; start F:standard+unisyn_start ;
21 id _29 ; stress 0 ;
22 id _30 ; stress 1 ;
23 id _31 ; stress 0 ;
24 id _32 ; stress 1 ;
End_of_Stream_Items
Relations
Relation Token ; "(" ")" ;
5 5 0 0 0 4
4 4 3 0 5 0
3 3 0 4 0 2
6 6 2 0 0 0
2 2 0 6 3 1
7 7 1 0 0 0
1 1 0 7 2 0
End_of_Relation
Relation Word ; "(" ")" ;
4 5 0 0 0 3
3 4 0 0 4 2
2 6 0 0 3 1
1 7 0 0 2 0
End_of_Relation
Relation Phrase ; "(" ")" ;
5 5 0 0 0 4
4 4 0 0 5 3
3 6 0 0 4 2
2 7 1 0 3 0
1 8 0 2 0 0
End_of_Relation
Relation Segment ; filename lab/BERLIN_002.lab ; seperator ";" ; nfields "1" ;
12 20 0 0 0 11
11 19 0 0 12 10
10 18 0 0 11 9
9 17 0 0 10 8
```

```
8 16 0 0 9 7
7 15 0 0 8 6
6 14 0 0 7 5
5 13 0 0 6 4
4 12 0 0 5 3
3 11 0 0 4 2
2 10 0 0 3 1
1 9 0 0 2 0
End_of_Relation
Relation Syllable ; "(" ")" ;
4 24 0 0 0 3
3 23 0 0 4 2
2 22 0 0 3 1
1 21 0 0 2 0
End_of_Relation
Relation SylStructure ; "(" ")" ;
8 19 0 0 0 7
7 18 0 0 8 6
6 17 0 0 7 5
5 16 4 0 6 0
4 24 3 5 0 0
3 4 0 4 0 2
12 15 0 0 0 11
11 14 10 0 12 0
10 23 0 11 0 9
14 13 0 0 0 13
13 12 9 0 14 0
9 22 2 13 10 0
2 6 0 9 3 1
17 11 0 0 0 16
16 10 15 0 17 0
15 21 1 16 0 0
1 7 0 15 2 0
End_of_Relation
End_of_Relations
End_of_Utterance
```

Bibliography

[Ass99] International Phonetic Association. *Handbook of the International Phonetic Association: A Guide to the Use of the International Phonetic Alphabet.* Cambridge University Press, June 1999.

[BL07] A.W. Black and K.A. Lenzo. Building synthetic voices, 2007. http://festvox.org/bsv/.

[BTC99] A. Black, P. Taylor, and R. Caley. The Festival Speech Synthesis System: System Documentation (1.4). Technical report, The Centre for Speech Technology Research, University of Edinburgh, 1999. http://www.cstr.ed.ac.uk/projects/festival/manual.

[CC01] T.F. Cox and M.A.A. Cox. *Multidimensional Scaling.* Chapman and Hall, 2001.

[Cok76] C.H. Coker. A model of articulatory dynamics and control. *Proceedings of the IEEE*, 64(4):452–460, April 1976.

[CRK07] R.A.J. Clark, K. Richmond, and S. King. Multisyn: Open-domain unit selection for the festival speech synthesis system. *Speech Communication*, 49:317–330, 2007.

[DBM03] J. Diard, P. Bessière, and E. Mazer. A survey of probabilistic models, using the Bayesian programming methodology as a unifying framework. In *Proceedings of the Second International Conference on Computational Intelligence, Robotics and Autonomous Systems (CIRAS 2003)*, Singapore, December 2003.

[Dut97] T. Dutoit. *An introduction to text-to-speech synthesis.* Kluwer Academic Publishers, Norwell, MA, USA, 1997.

[Fla65] J. Flanagan. *Speech Analysis Synthesis and Perception (Kommunikation und Kybernetik in Einzeldarstellungen, 3).* Springer-Verlag, 1965.

[FTKI92] T. Fukada, K. Tokuda, T. Kobayashi, and S. Imai. An adaptive algorithm for mel-cepstral analysis of speech. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP-92)*, volume 1, pages 137–140 vol.1, Mar 1992.

[Fur01] S. Furui. *Digital Speech Processing, Synthesis, and Recognition (Second Edition, Revised and Expanded).* Marcel Dekker, Inc., 2001.

[HYS08] G. Hofer, J. Yamagishi, and H. Shimodaira. Speech-driven lip motion generation with a trajectory HMM. In *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH-08)*, pages 2314–2317, Brisbane, Australia, September 2008.

[Ima83] S. Imai. Cepstral analysis synthesis on the mel frequency scale. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP-83)*, volume 8, pages 93–96, Apr 1983.

[KKCM08] V. Karaiskos, S. King, R.A.J. Clark, and C. Mayo. The Blizzard Challenge 2008. In *Proceedings of the Blizzard Challenge Workshop 2008*, Brisbane, Australia, September 2008.

[KPM$^+$09] C. Kranzler, F. Pernkopf, R. Muhr, M. Pucher, and F. Neubarth. Text-to-speech engine with Austrian German corpus. In *Proceedings of the International Conference on Speech and Computer (SPECOM-09) (accepted)*, St. Petersburg, Russia, 2009.

[Moo87] S. Moosmüller. *Soziophonologische Variation im gegenwärtigen Wiener Deutsch.* Franz Steiner Verlag, Stuttgart, 1987.

[MS97] R. Muhr and R. Schrodt. *Österreichisches Deutsch und andere nationale Varietäten plurizentrischer Sprachen in Europa.* Verlag öbv&hpt, 1997.

[MTKI96] T. Masuko, K. Tokuda, T. Kobayashi, and S. Imai. Speech synthesis using HMMs with dynamic features. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP-96)*, volume 1, pages 389–392 vol. 1, May 1996.

[Muh07] R. Muhr. *Österreichisches Ausspracheworterbuch, Österreichische Aussprachedatenbank.* Peter Lang Verlag, 2007.

[NPK08] F. Neubarth, M. Pucher, and C. Kranzler. Modeling Austrian dialect varieties for TTS. In *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH-08)*, pages 1877–1880, Brisbane, Australia, 2008.

[NYK05] N. Niwase, J. Yamagishi, and T. Kobayashi. Human walking motion synthesis with desired pace and stride length based on HSMM. *IEICE Transactions on Information and Systems*, E88-D(11):2492–2499, Nov. 2005.

[Ode95] J.J. Odell. *The Use of Context in Large Vocabulary Speech Recognition.* PhD thesis, Cambridge University, 1995.

[PK08] B. Pfister and T. Kaufmann. *Sprachverarbeitung: Grundlagen und Methoden der Sprachsynthese und Spracherkennung.* Springer Verlag, 2008.

[PSYN09] M. Pucher, D. Schabus, J. Yamagishi, and F. Neubarth. Modeling and interpolation of Austrian German and Viennese dialect / sociolect – towards flexible multi-dialect HMM-based speech synthesis. *Speech Communication (submitted)*, 2009.

[Rab89] L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, Feb 1989.

[RJ93] L.R. Rabiner and B.H. Juang. *Fundamentals of speech recognition.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[SZN$^+$06] K. Saino, H. Zen, Y. Nankaku, A. Lee, and K. Tokuda. HMM-based singing voice synthesis system. In *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH-06)*, pages 1141–1144, Sept. 2006.

[TKI95] K. Tokuda, T. Kobayashi, and S. Imai. Speech parameter generation from HMM using dynamic features. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP-95)*, volume 1, pages 660–663 vol.1, May 1995.

[TMMK99] K. Tokuda, T. Masuko, N. Miyazaki, and T. Kobayashi. Hidden Markov models based on multi-space probability distribution for pitch pattern modeling. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP-99)*, pages 229–232, Washington, DC, USA, 1999. IEEE Computer Society.

[TMTK01] M. Tamura, T. Masuko, K. Tokuda, and T. Kobayashi. Adaptation of pitch and spectrum for HMM-based speech synthesis using MLLR. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP-01)*, pages 805–808, May 2001.

[TYMK04] M. Tachibana, J. Yamagishi, T. Masuko, and T. Kobayashi. HMM-based speech synthesis with various speaking styles using model interpolation. In *Proceedings of the 2nd International Conference on Speech Prosody*, volume SP2004, pages 413–416, 2004.

[TYMK05] M. Tachibana, J. Yamagishi, T. Masuko, and T. Kobayashi. Speech synthesis with various emotional expressions and speaking styles by style interpolation and morphing. *IEICE Transactions on Information and Systems*, E88-D(11):2484–2491, 2005.

[wik09] Wiktionary: Lautschrift, Feb 2009. http://de.wiktionary.org/wiki/Wiktionary:Lautschrift.

[Wit82] I.H. Witten. *Principles of Computer Speech.* Academic Press, Inc. (London) LTD., 1982.

[Yam06] J. Yamagishi. *Average-Voice-Based Speech Synthesis.* PhD thesis, Tokyo Institute of Technology, 2006.

[YK07] J. Yamagishi and T. Kobayashi. Average-voice-based speech synthesis using HSMM-based speaker adaptation and adaptive training. *IEICE Transactions on Information and Systems*, 90-D(2):533–543, 2007.

[YKN$^+$09] J. Yamagishi, T. Kobayashi, Y. Nakano, K. Ogata, and J. Isogai. Analysis of speaker adaptation algorithms for HMM-based speech synthesis and a constrained SMAPLR adaptation algorithm. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(1):66–83, Jan. 2009.

[YMK04] J. Yamagishi, T. Masuko, and T. Kobayashi. HMM-based expressive speech synthesis – towards TTS with arbitrary speaking styles and emotions. In *Proceedings of Special Workshop in Maui (SWIM)*, Jan. 2004.

[YMT$^+$97] T. Yoshimura, T. Masuko, K. Tokuda, T. Kobayashi, and T. Kitamura. Speaker interpolation in HMM-based speech synthesis system. In *Proceedings of the European Conference on Speech Communication and Technology (EUROSPEECH-97)*, pages 2523–2526, 1997.

[YNZ$^+$09] J. Yamagishi, T. Nose, H. Zen, Z.-H. Ling, T. Toda, K. Tokuda, S. King, and S. Renals. A robust speaker-adaptive HMM-based text-to-speech synthesis. *IEEE Transactions on Speech, Audio & Language Processing*, 2009. (in press).

[Yos02] T. Yoshimura. *Simultaneous modeling of phonetic and prosodic parameters, and characteristic conversion for HMM-based text-to-speech systems.* PhD thesis, Department of Electrical and Computer Engineering, Nagoya Institute of Technology, 2002.

[YOW94] S.J. Young, J.J. Odell, and P.C. Woodland. Tree-based state tying for high accuracy acoustic modelling. In *Proceedings of the workshop on Human Language Technology (HLT-94)*, pages 307–312, Morristown, NJ, USA, 1994. Association for Computational Linguistics.

[YTM$^+$98] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura. Duration modeling for HMM-based speech synthesis. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP-98)*, pages 29–32, 1998.

[YTM$^+$99] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura. Simultaneous modeling of spectrum, pitch and duration in HMM-based speech synthesis. In *Proceedings of the European Conference on Speech Communication and Technology (EUROSPEECH-99)*, pages 2374–2350, 1999.

[YTM$^+$00] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura. Speaker interpolation for HMM-based speech synthesis system. *Acoustical Science and Technology*, 21(4):199–206, 2000.

[YTM$^+$03] J. Yamagishi, M. Tamura, T. Masuko, K. Tokuda, and T. Kobayashi. A context clustering technique for average voice models. *IEICE Transactions on Information and Systems*, E86-D(3):534–542, March 2003.

[YZW$^+$08] J. Yamagishi, H. Zen, Y.-J. Wu, T. Toda, and K. Tokuda. The HTS-2008 system: Yet another evaluation of the speaker-adaptive HMM-based speech synthesis system in the 2008 Blizzard Challenge. In *Proceedings of the Blizzard Challenge Workshop 2008*, September 2008.

[ZTB09] H. Zen, K. Tokuda, and A.W. Black. Statistical parametric speech synthesis. *Speech Communication*, In Press, Accepted Manuscript, 2009.

[ZTNT07] H. Zen, T. Toda, M. Nakamura, and K. Tokuda. Details of Nitech HMM-based speech synthesis system for the Blizzard Challenge 2005. *IEICE Transactions on Information and Systems*, E90-D(1):325–333, January 2007.

[ZTO09] H. Zen, K. Tokuda, and K. Oura. hts_engine, 2009. http://hts-engine.sourceforge.net/.