Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institut für
Technische Informatik und
Kommunikationsnetze

Implementation and evaluation of an HMM-based speech generation component for the SVOX TTS system

Simon Würgler

Semester Thesis SA-2011-02
Spring Semester 2011

Computer Engineering and Networks Laboratory

Advisor: Thomas Ewender
Co-Advisor: Sarah Hoffmann
Supervisor: Prof. Dr. L. Thiele

Abstract

Hidden Markov Model (HMM)-based speech synthesis has emerged in recent years as an alternative to speech synthesis methods like unit selection. There are two main advantages for this statistical approach to speech generation: the speech waveform is directly generated from the HMM parameters, so no extensive speech signal database needs to be stored. Furthermore, voice characteristics, speaking styles or emotions of the synthesized speech can easily be changed by adapting the HMM parameters.

In this thesis the integration of an HMM-based voice generation component to the text-to-speech system SVOX (developed at the ETHZ Speech Processing Lab) is presented. For this task another text-to-speech system called MARY is used, which already integrates HMM-based speech synthesis.

Acknowledgement

I would like to thank Thomas Ewender and Sarah Hoffmann who advised and supported this thesis with their patience and broad knowledge.

Contents

1 Introduction

This report is organized as follows. Section 1.1 introduces HMM-based speech synthesis. Section 1.2 describes the text-to-speech system MARY, which already integrates HMM voice building and HMM-based speech synthesis. In Section 1.3 we give an overview of SVOX, the text-to-speech system developed at the ETHZ Speech Processing Lab.

Section 2.1 describes the steps involved in building a new HMM-based voice, starting with a database of speech signals and their textual transcriptions. The proposed architecture to include HMM-based voice building and synthesis in SVOX is discussed in Section 2.2. The evaluation of the new voices is presented in Section 3.

1.1 HMM-based speech synthesis

Hidden Markov Model (HMM)-based speech synthesis has emerged in recent years as an alternative to speech synthesis methods like unit selection. Unit selection is still state-of-the-art, as high quality speech generation can be achieved selecting and concatenating acoustical units from previously recorded speech signals. Although the speech signal units are processed to control the prosody (duration and fundamental frequency), the waveform quality can still be preserved to give excellent synthesized speech quality. However if one wants to obtain various voice characteristics, speaking styles, or emotions, a very large amount of speech data needs to be collected, segmented and stored when using the unit selection approach. This is one of the main reasons for statistical speech synthesis to have grown so much in popularity lately.

An overview of a HMM-based speech synthesis system is shown in **Figure 1.1** [1]. The system contains two parts. In the training part spectrum (mel-cepstral coefficients) and excitation parameters (fundamental frequency) are extracted from the speech database and modeled by context dependent HMMs. A 5-state HMM is trained for each phoneme in every required context.

In the synthesis part, the context dependent HMMs are concatenated according to the text to be synthesized. For each phoneme the corresponding HMM model from the same context is selected to build a longer HMM for the whole sentence. Spectrum and excitation parameters are then generated from the HMM (the “observations”) according to speech parameter generation algorithm described in [2]. The waveform is eventually generated as in “vocoded speech”, with an excitation generation module and a synthesis filter set by the HMM generated parameters. With mixed excitation and postfiltering the speech synthesis quality may be improved further.

In this speech synthesis system only the HMM parameters and the context tree need to be stored, which is a much smaller amount of data compared to all speech waveforms stored in unit selection. Furthermore, if voice characteristics need to be changed, the HMM parameters can easily be adapted with some speaker adaptation techniques developed for speech recognition [3].

The core idea for HMM-based synthesis has been proposed by Tokuda from the Nagoya Institute of Technology in Japan. The HMM-based Speech Synthesis System[1] (HTS) is developed and maintained at the same institute and provides scripts for HMM training and parameter generation. The simultaneous modeling of spectrum, pitch and duration in a unified framework of HMM is described in [4]. The key to speech synthesis HMM modeling used in HTS is the introduction of multi-space probability distributions, which are described in [5]. Although there are some very interesting ideas and non-standard HMM algorithms in HTS, we will not discuss them in detail here. In this project we use HTS scripts as they are integrated in the MARY system, but the focus does not lie on the algorithms used.

[1] http://hts.sp.nitech.ac.jp/

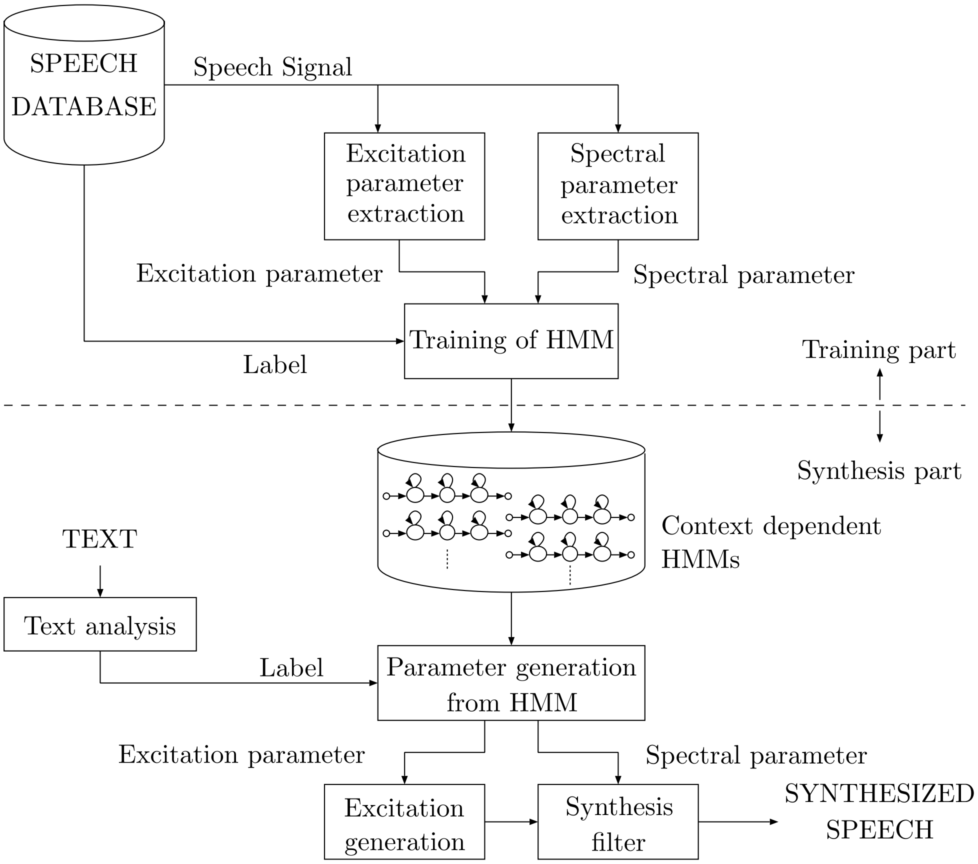


Figure 1.1: HMM-based speech synthesis system (from [1]).

1.2 The MARY Text-To-Speech System

The text-to-speech system MARY (Modular Architecture for Research on speech sYnthesis) is an open source speech synthesis platform. It is maintained by the German Research Center for Artificial Intelligence (DFKI).

The systems functionality includes everything needed for converting a written text into a spoken text (waveform). Text in, audio out, simple enough to get the picture. Of course the interesting part is what happens in between, and that is, as the bard would tell us, where the rub lies. With more than 260000 lines of Java code the system is really huge and complex. It is the result of years of development. Obviously it is out of scope of this project to give a detailed discussion on the whole system, the reader is referred to the publication [6] and the official homepage[2]. Actually, for more insights a look at the (pretty good documented) code is unavoidable. The focus here lies on giving an overview, with particular attention to the elements relevant to this project.

1.2.1 MARY components

The MARY TTS comes with the following components:

maryserver Main component, does all the processing.

maryclient GUI for speech synthesis, sends input data to the server and receives processing results (see Figure 1.2).

voiceimport GUI used for training and building a new voice (see Figure 1.3).

mary-component-installer GUI used to install new voices (see Figure 1.4).

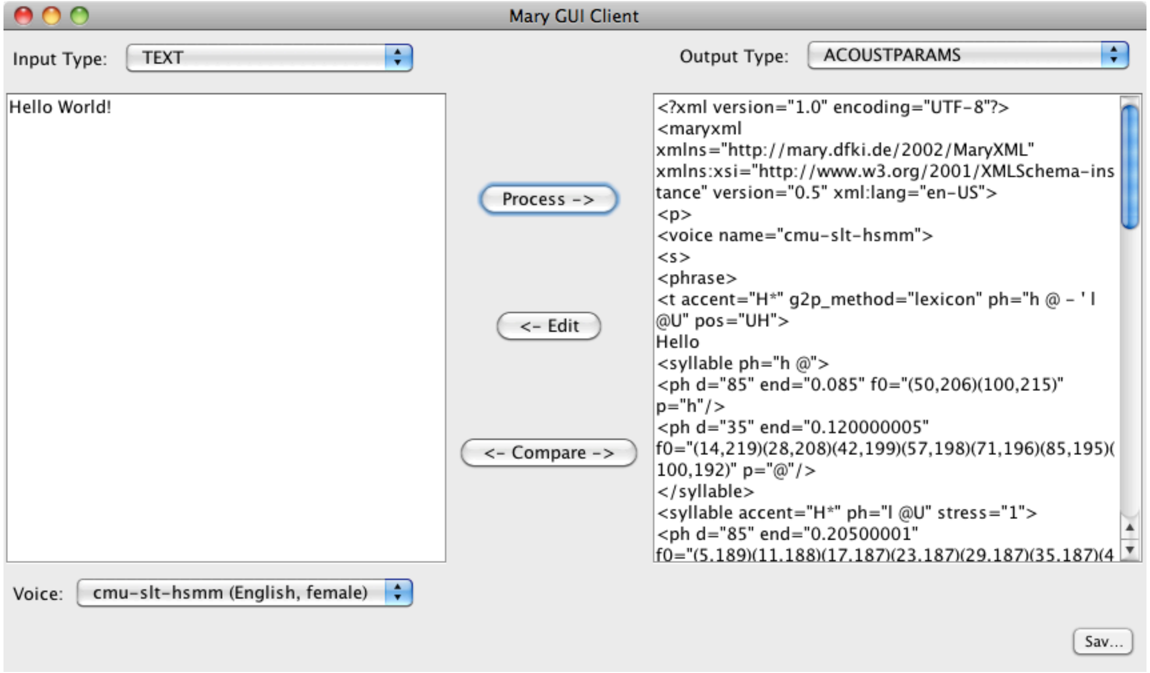
The client and the voice import component both connect to the server to process data. This means that the server must run when using the GUIs. The server is multi-threaded, which allows for more than one client or component to send requests to be processed.

The client allows to choose from different input types and output types. The two far ends are plain text as input and audio as output, however it is possible to output intermediate processing steps as well. This is described more detailed in section 1.2.2. It is possible to have a quick look at the client GUI without installing the whole TTS system: the developers have put online at `http://mary.dfki.de:59125/` a MARY

[2] http://mary.dfki.de/

Web client which connects to their MARY http server and actually offers the complete TTS functionality.

The voice import component provides a GUI to all the scripts and processing steps needed to build and train new voices. All is needed to train a new voice is a set of speech signals (the waveforms) and its corresponding text transcriptions. The component allows to build voices for unit selection synthesis or, as will be used in this project, for HMM-based speech synthesis. For HMM training the MARY voice import component uses HTS scripts (see Section 1.1). More on the voice import component is found in Section 2.1.



Input Type: TEXT

Output Type: ACOUSTPARAMS

Hello World!

```
<?xml version="1.0" encoding="UTF-8"?>
<maryxml
xmlns="http://mary.dfki.de/2002/MaryXML"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-ins
tance" version="0.5" xml:lang="en-US">
<p>
<voice name="cmu-slt-hsmm">
<s>
<phrase>
<t accent="H*" g2p_method="lexicon" ph="h @ - ' l
@U" pos="UH">
Hello
<syllable ph="h @">
<ph d="85" end="0.085" f0="(50,206)(100,215)"
p="h"/>
<ph d="35" end="0.120000005"
f0="(14,219)(28,208)(42,199)(57,198)(71,196)(85,195)(
100,192)" p="@"/>
</syllable>
<syllable accent="H*" ph="l @U" stress="1">
<ph d="85" end="0.20500001"
f0="(5.189)(11.188)(17.187)(23.187)(29.187)(35.187)(4
```

Voice: cmu-slt-hsmm (English, female)

Figure 1.2: MARY Client GUI

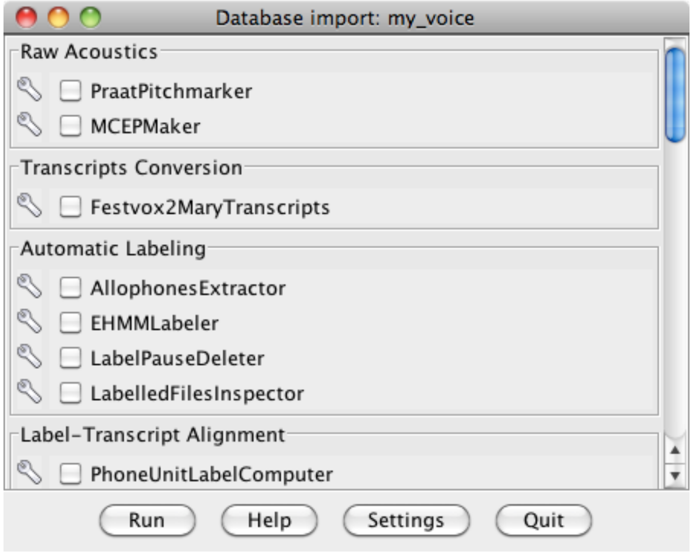
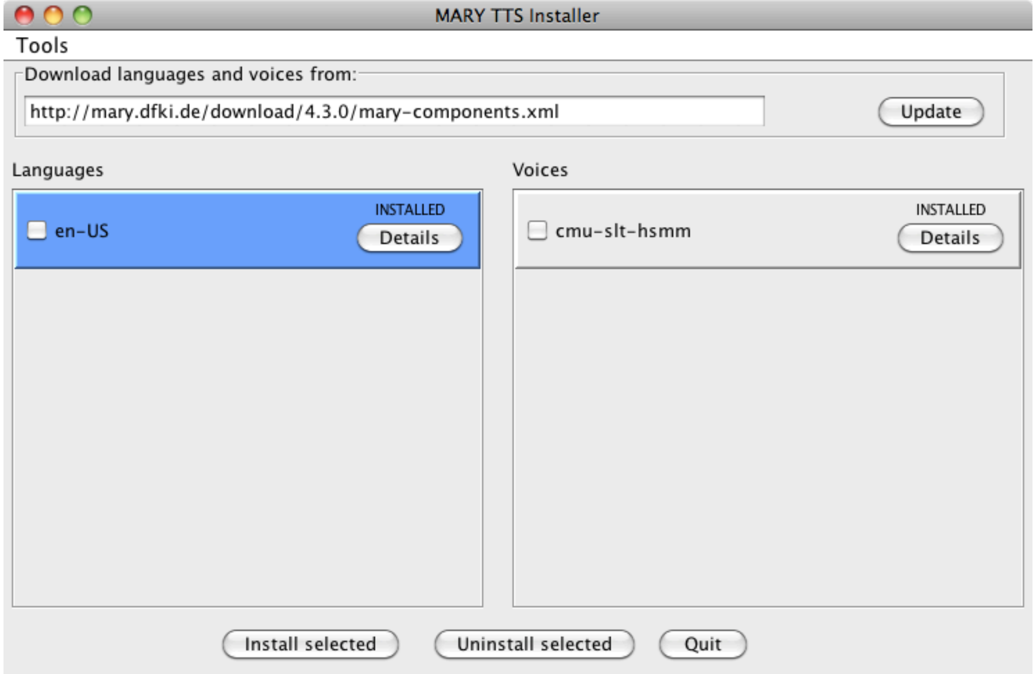


Figure 1.3: Voice Import GUI



Download languages and voices from: http://mary.dfki.de/download/4.3.0/mary-components.xml

Figure 1.4: MARY Component Installer GUI

1.2.2 MaryXML

As the acronym tells us, the MARY system is built with a modular architecture. The modules work sequentially, and the output of each module can be accessed to read out partial processing results. Moreover, such results can as well be changed and used as input to the following processing modules. This is key to this project, as it allows to break up the system and use only part of it.

The interface between the processing modules is given by MaryXML files. As XML files they have to be well-formed and valid. The XML Schema definition of MaryXML can be found at `http://mary.dfki.de/lib/MaryXML.xsd`. The definition is nice to have a look at, for instance if one needs to check which attribute/value pair some tag may contain. However the best way to understand the MaryXML format is to have a look at some examples.

Listing 1 shows a simple text line used as input to the system.

Listing 2 shows the MaryXML ALLOPHONES output when processing the input text. It contains a header with the XML Declaration and a document body enclosed by the `<maryxml>...</maryxml>` tags. Some general informations for speech synthesis are included, like the locale (language code `en-GB`) or the voice name `dfki-spike`. The main tags used are `<phrase>`, `<t>` (token), `<syllable>` and `<ph>` (phone), which are nested in the listed order. The `<t>` tag (for token) has some optional attributes (see line 7):

- `accent="H*"` indicates pitch accent in ToBI (Tones and Break Indices) annotation; here `H*` stands for peak accent.

- `g2p_method="lexicon"` indicates that the word has been found in the lexicon database, where the phonemic transcription is provided.

- `ph="h @ - ’ l @U"` lists the phones in this word, using the SAMPA phonetic symbols.

- `pos="UH"` indicates the part of speech in Penn Treebank annotation style; here `UH` stands for interjection.

The `<syllable>` tag may have some optional attributes as well (see line 13):

- `accent="H*"` indicates pitch accent as in the `<t>` tag.

- `ph="l @U"` lists the phones in this syllable.

- `stress="1"` gives information about the prosodic phrasing.

The `<ph>` tag just indicates the phonetic symbol of the given phone.

Further information about breaks and phrase tones are included into the `<boundary>` tag (see line 30). The required attribute to this tag is `breakindex` with value [0-5] as defined by ToBI. The `tone` attribute for phrase tone in ToBI annotation is optional though.

All these informations are the result of many processing steps implemented in the maryserver, including tokeniser (cutting text into words), numbers and abbreviations preprocessing, part-of-speech tagger, phonemisation (phonemic transcription, either using lexicon or some rather complex letter-to-sound conversion algorithm) and prosody modeling. The maryclient allows to output some intermediate steps, which then look very much like the ALLOPHONES file in Listing 2 without some of the tags and attributes. Basically with each processing step new tags and attributes are added to the existing xml structure.

Listing 3 shows the MaryXML ACOUSTPARAMS output. The acoustic parameters (duration and frequency) are calculated according to some rules based on the ToBI tones [7]. The resulting xml file contains new attributes in the phone tag `<ph>`:

- `d`: the duration with values in ms.
- `end`: the end time measured from start with values in s.
- `f0`: the fundamental frequency over the phone duration. The value of this attribute is a list of pairs, where the first entry is a value between 0 (phone start) and 100 (phone end) and the second entry is the frequency in Hz.

The ACOUSTPARAMS output gives the maximally rich MaryXML structure in the whole processing system. The next and last step before waveform synthesis is the phone unit feature computer, which outputs a file called TARGETFEATURES. This step is described in Section 2.1. For each phone it lists all phonetic features (vowel height, consonant type etc.) and contextual features (previous phone, next phone etc.). These features are used to choose the correct waveforms (for unit selection synthesis) or the correct context dependent HMM model (for HMM-based synthesis) of the given phone.

Listing 1: TEXT input

```
Hello World!
```

Listing 2: MaryXML ALLOPHONES

```
<?xml version="1.0" encoding="UTF-8"?>
<maryxml xmlns="http://mary.dfki.de/2002/MaryXML" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="0.5" xml:lang="en-GB">
<p>
<voice name="dfki-spike">
<s>
<phrase>
<t accent="H*" g2p_method="lexicon" ph="h @ - ' l @U" pos="UH">
Hello
<syllable ph="h @">
<ph p="h"/>
<ph p="@"/>
</syllable>
<syllable accent="H*" ph="l @U" stress="1">
<ph p="l"/>
<ph p="@U"/>
</syllable>
</t>
<t accent="H*" g2p_method="lexicon" ph="' w r= l d" pos="NNP">
World
<syllable accent="H*" ph="w r= l d" stress="1">
<ph p="w"/>
<ph p="r="/>
<ph p="l"/>
<ph p="d"/>
</syllable>
</t>
<t pos=".">
!
</t>
<boundary breakindex="5" tone="L-L%"/>
</phrase>
</s>
</voice>
</p>
</maryxml>
```

Listing 3: MaryXML ACOUSTPARAMS

```
<?xml version="1.0" encoding="UTF-8"?>
<maryxml xmlns="http://mary.dfki.de/2002/MaryXML" xmlns:xsi=" ↩
    http://www.w3.org/2001/XMLSchema-instance" version="0.5" ↩
    xml:lang="en-GB">
<p>
<voice name="dfki-spike">
<s>
<phrase>
<t accent="H*" g2p_method="lexicon" ph="h @ - ' l @U" pos="UH">
Hello
<syllable ph="h @">
<ph d="68" end="0.0681346" p="h"/>
<ph d="44" end="0.1125144" f0="(0,127) (50,97) (100,110)" p="@ ↩
    "/>
</syllable>
<syllable accent="H*" ph="l @U" stress="1">
<ph d="112" end="0.22480941" f0="(0,111)" p="l"/>
<ph d="153" end="0.37799042" f0="(50,141) (100,104)" p="@U"/>
</syllable>
</t>
<t accent="H*" g2p_method="lexicon" ph="' w r= l d" pos="NNP">
World
<syllable accent="H*" ph="w r= l d" stress="1">
<ph d="169" end="0.5470984" f0="(0,112)" p="w"/>
<ph d="145" end="0.69228137" f0="(50,96)" p="r="/>
<ph d="148" end="0.84076834" p="l"/>
<ph d="95" end="0.93528175" f0="(100,110)" p="d"/>
</syllable>
</t>
<t pos=".">
!
</t>
<boundary breakindex="5" duration="400" tone="L-L%"/>
</phrase>
</s>
</voice>
</p>
</maryxml>
```

1.3 The SVOX Text-to-Speech System

The SVOX system is a complete text-to-speech system developed at the ETHZ Speech Processing Lab. The concepts on which the system builds are described in [8]. The architecture is modular as in MARY, but different approaches are used for text analysis and determination of the physical parameters duration and frequency.

The SVOX System makes a clear distinction between all the processing that is speaker independent (called transcription stage), and those processing steps that are speaker dependent (called phono-acoustical stage). The interface between these two main modules is denoted with the term “phonological representation”.

1.3.1 Transcription stage

The transcription unit converts the input text to its phonological representation. In a first step a morphological analysis of each word and a syntax analysis of the whole sentence results in an annotated syntax tree. The syntax tree is then used to determine the distribution of accents and phrases of the sentence. More details on the transcription stage is found in Chapter 8.4 of [8] and in [9].

The output of the transcription stage is shown in **Listing 4**. It contains the following elements:

- `#{*:k}` curly brackets denote a phrase boundary. The first element in the brackets can be either `P` for a continuative phrase, `T` for a terminal phrase or `E` if the phrase is final. The index k has value 0 when the phrase marks the start or the end of the sentence. The value 1 is used when a punctuation mark is given in the input text, the value 2 just before a finite verb. The higher the index value k, the stronger the syntax coupling between the words that mark the phrase boundary.

- `\G\` is the German language tag. SVOX supports multilingual text-to-speech, thus the language tags `\E\`, `\F\` and `\I\` (for English, French and Italian, respectively) are possible. In this project we only consider German texts.

- `[*]` square brackets denote the accent, where the value 1 indicates a primary accent, and 4 no accent at all.

- `-` marks a syllable boundary.

- For the phonetic symbols ETHPA (ETH phonetic alphabet) annotation is used (see page 401-404 in [8]).

Listing 4: SVOX Phonological Representation

```
#{T:0}  \G\h[2]a:-?[4]aI-?[4]o:  v[1]EIt  #{E:0}
```

1.3.2 Phono-acoustical stage

The phono-acoustical stage converts the phonological representation to synthesized speech signal. In a first step some speaker dependent phonological processing is done, which considers differences between the phone sequence in natural speech and the phone sequence in the phonological representation. In particular assimilation and elision effects at word boundaries are taken into account.

The prosody control is the next step in the text-to-speech processing, and is the main component of the phono-acoustical stage. For each phone it generates the physical parameters duration and fundamental frequency, based on statistical models. The frequency is modeled by a neural network, which predicts several values per syllable.

The output of the prosody control unit is called "phone sequence" and is shown in **Listing 5**. It contains the following elements:

- language tag and phonetic symbols in ETHPA annotation style.
- phone duration in ms.
- fundamental frequency over the phone duration. It is given by a list of pairs with time points as first element (0 indicates the phone start, 100 is the phone end) and the frequency as second element.

With the given duration and fundamental frequency for each phone the sentence can eventually be synthesized to speech. SVOX supports unit selection synthesis and uses TD/PSOLA (Time-Domain Pitch-Synchronous Overlap-Add) to control duration and the fundamental frequency of the selected units. For more details on the phono-acoustical stage see Chapter 9 in [8].

Listing 5: SVOX Phone Sequence

```
\G\h    44.93    0:216  25:216  50:217  75:218  100:219
\G\a:   87.33    0:219  25:222  50:224  75:224  100:220
\G\?    45.76    0:220  25:216  50:212  75:207  100:203
\G\a    72.69    0:203  25:199  50:196  75:194  100:193
\G\l    63.83    0:193  25:192  50:191  75:191  100:191
\G\?    42.29    0:191  25:190  50:190  75:189  100:188
\G\o:   111.24   0:188  25:184  50:181  75:178  100:183
\G\v    51.05    0:183  25:187  50:192  75:195  100:197
\G\E    85.25    0:197  25:196  50:195  75:194  100:195
\G\l    82.02    0:195  25:195  50:195  75:195  100:195
>       54.79    0:195  25:196  50:196  75:196  100:196
\G\t    80.97    0:196  25:196  50:196  75:197  100:197
/       342.55   0:197  25:198  50:198  75:198  100:198
```

2 Approach

2.1 Building a new HMM-based voice for MARY

The first goal of this project is to build a new German HMM-based voice for the MARY TTS system. The ETH Speech Processing Lab holds a vast database of speech recordings of Ms. Heim with the corresponding transcriptions. This database has served here as training set to generate the new voice.

As seen in Section 1.2.1, the MARY installation comes with a voice import component, a graphical user interface to all the scripts needed for voice building. A full description of the voice import component is given in [10]. In this section we will give an overview of the steps involved to create an HMM-based voice. A very useful technical step by step tutorial is found on the MARY homepage[3]. Actually, this tutorial is not only very useful, but very much necessary.

The task of building a new voice is a stony road, as a big amount of steps are involved and usually nothing is straight-forward. Some steps are also time consuming, as they may take several hours. Lots of scripts are used, and they deal with almost every single aspect of speech synthesis.

The main processing steps are shown in **Figure 2.1** on page 21. All components can be called by checking the corresponding line in the voice import GUI and clicking on "run". The parameters and calling arguments can be set using the GUI as well. The main components are the following:

EHMM labeler is an external component called by MARY for automatic phonetic segmentation and labeling of the speech signals. It extracts cepstral coefficients from the wavefiles and trains HMMs using the Baum Welch algorithm to determine the phone boundaries. The output of the component are label files as shown in **Listing 6**. The first column contains the phone end times in ms, the last column the phone symbol.

Phone Unit Label Computer is used to convert the label format from EHMM to MARY; the MARY label format is shown in **Listing 7**.

Allophones Extractor processes the text transcription of the speech signals to generate a MaryXML ALLOPHONES file as shown in **Listing 2** and described in Section 1.2.2.

Transcription Aligner makes sure that the label and the allophone files are aligned, thus no mismatch between the phone sequence exists.

[3] `http://mary.opendfki.de/wiki/HMMVoiceCreation`

Feature Selection saves a file with the list of all features to be considered in the next step.

Phone Unit Feature Computer extracts linguistic target or context feature vectors from the allophone files. For each phone it basically lists all kind of features, mostly phonetic (vowel height, consonant type, etc) and contextual (the features of previous and next phone). **Listing 8** shows how such a phone feature file looks like. Each file starts with a list of the extracted features chosen with the Feature Selection component (in this case `phone`, `accented`, `accented_syls_from_phrase_end` etc) with their possible values. The second section (starting at line 13 in the given example) displays the sequence of phones of the processed sentence (`_`, `ts`, `aI`, `t` etc) and their feature values in the order they are listed in the first section. The last section (line 19) is basically the same as in the second section, now all values are encoded with integers.

Phone Label Feature Aligner makes sure that that the phone labels and the phone feature files are aligned, thus no mismatch between the phone sequence exists.

HMM Voice Data Preparation converts all wave files to raw and checks if the text files are available.

HMM Voice Configure configures some voice properties, e.g. its name, sampling rate, lower/upper frequency bound (depending on male or female voice) etc.

HMM Voice Feature Selection saves a list of features which are used to build the context tree for the HMM models. This could basically be the same list as above (see Feature Selection), however it doesn't make sense to consider every single feature for HMM training. Thus a subset of features is chosen.

HMM Voice Make Data This component basically calls external HTS scripts, which have been slightly adapted to the MARY system. These scripts extract mel-generalized cepstral coefficients (mgc), log F_0 (lf0), voicing strengths for mixed excitation (str) and Fourier magnitudes (mag) from the audio files. They are assembled to an acoustic parameter vector (mgc+lf0+str+mag).

HMM Voice Make Voice is used for training the HMM models using the adapted HTS scripts.

HMM Voice Packager collects all files necessary for speech synthesis. This package can then be installed with the MARY Component Installer GUI, and the new voice is ready to be used within the MARY Client for speech synthesis.

Listing 6: EHMM label file (.lab)

```
#
0.1 125 pau
0.21 125 ts
0.42 125 aI
0.5 125 t
...
```

Listing 7: MARY phone label file (.lab)

```
format: end time, unit index, phone
#
0.100000 1 _
0.210000 2 ts
0.420000 3 aI
0.500000 4 t
...
```

Listing 8: MARY phone feature file (.pfeats)

```
phone 0 2 2: 6 9 9~ ? @ C D E E: EI I N O OY R S T U Y Z _ a a: aI aU a~ b d e e: e~ f g h i i: j k l m n o o: o~ p pf r s t tS ts u u: v w x y y: z
accented 0 1
accented_syls_from_phrase_end 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
accented_syls_from_phrase_start 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
breakindex 0 1 2 3 4 5 6
edge 0 start end
gpos 0 in to det md cc wp pps aux punc content
next_accent 0 * H* !H* ^H* L* L+H* L*+H L+!H* L*+!H L+^H* L*+^H H+L* H+!H* H+^H* !H+!H* ^H+!H* ^H+^H* H*+L !H*+L
next_cplace 0 l a p b d v u g
next_ctype 0 s f a n l r
...

_ 0 3 0 0 0 0 L+H* a a ...
ts 1 2 0 0 0 0 L+H* 0 0 ...
aI 1 2 0 0 0 0 L+H* a s ...
t 1 2 0 1 0 0 L+H* g f ...
...

23 0 3 0 0 0 0 6 2 3 ...
53 1 2 0 0 0 0 6 0 0 ...
26 1 2 0 0 0 0 6 2 1 ...
51 1 2 0 1 0 0 6 8 2 ...
...
```

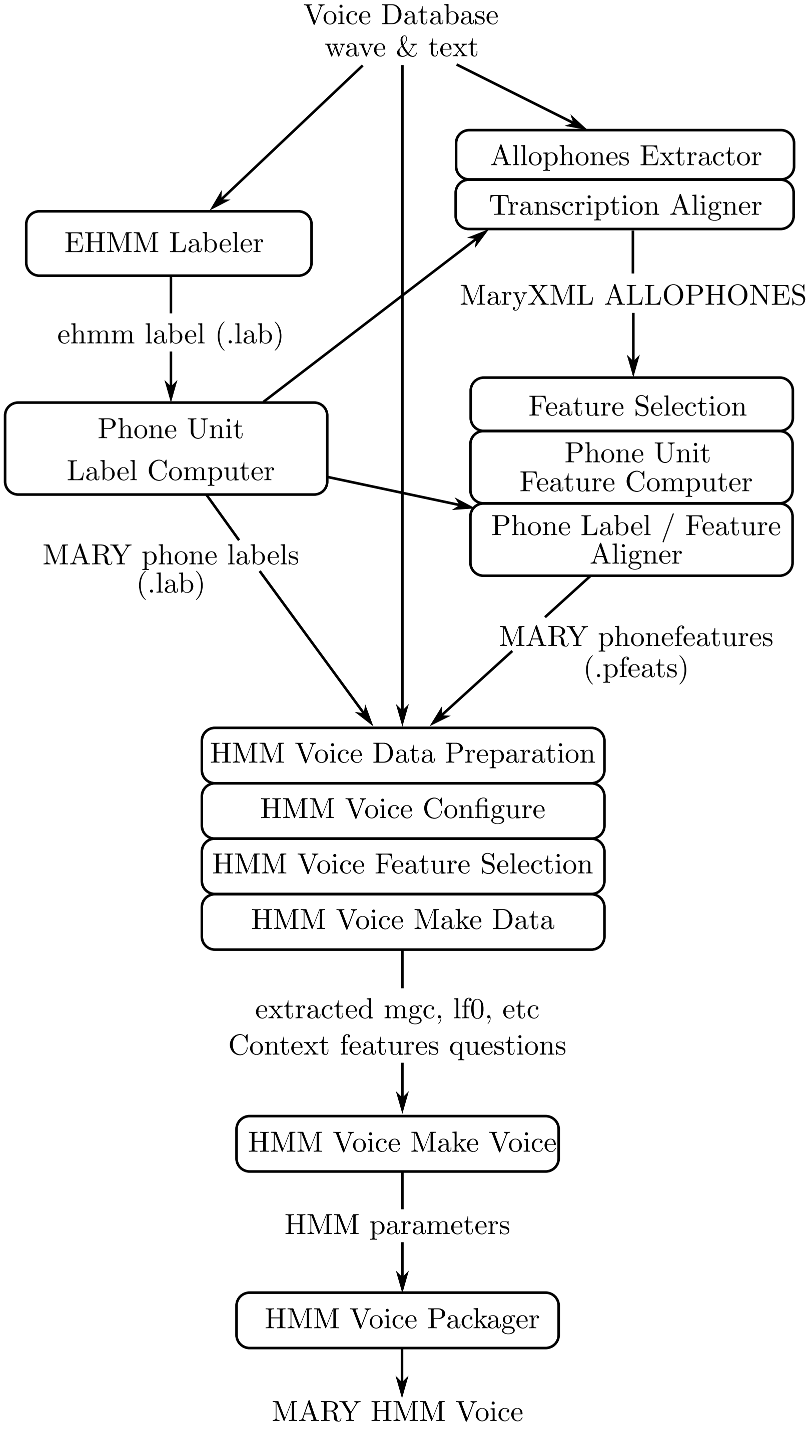


Figure 2.1: MARY HMM-based voice training.

2.2 Connecting SVOX to MARY

The second goal of this project is to build a German HMM-based voice to be used with the SVOX TTS system. As both HMM-based voice training (HMM parameter calculation) and HMM-based speech generation (from acoustical parameters to audio) are already implemented in MARY (see Section 2.1), it seems obvious to still use the same components and just adapt them for SVOX. An interface between the two systems needs thus to be designed which accounts for:

- Different phonetic symbols (ETHPA vs. SAMPA)
- Different label file format
- Different phonological representation format
  (.ptr vs. MaryXML ALLOPHONES)
- Different prosody elements file format
  (.phones vs. MaryXML ACOUSTPARAMS)

The SVOX adapted HMM voice training is described in Section 2.2.1, while Section 2.2.2 discusses the SVOX HMM-based text-to-speech system.

2.2.1 SVOX HMM Training

Figure 2.2 on page 27 shows the architecture of the implemented SVOX-MARY HMM-based voice training. The figure graphically sums up the detailed text description in this section.

A SVOX segmentation script is used to determine the phone boundaries in the training speech signals. The resulting files are SVOX labels, an example is shown in **Listing 9**. The first column indicates the phone start time in units of 100ns, the second column marks the end time and the last column contains the language tag and the phone symbol in ETHPA annotation. To be fed into the MARY voice building process, these lab files need to be converted to the MARY phone label format (see Listing 7). This is done with the Matlab function

```
SVOXlab2MARYlab(SVOXlabDir, MARYphonelabDir)
```

which converts all files in the directory `SVOXlabDir` and saves them to the directory `MARYphonelabDir`. The function processes the SVOX label files as follows:

- The language tag `\G\` is deleted.

- Since we're only considering German speech synthesis, when a foreign language tag (\E\, \F\ or \I\) is found, a warning is displayed and the file is not converted.

- The column containing start times is not considered, the values in the end times column are converted to ms.

- Preplosive pauses (*_c) are merged with subsequent plosive phones. In Listing 9 at line 5 for instance there is a preplosive pause (t_c) to the subsequent plosive phone t at line 6. These two lines would be merged to a single line with duration equal to the sum of preplosive pause and plosive phone durations. This merging is done to match the phonological descriptions (.ptr files) of the speech signals, which do not distinguish between preplosive pauses and the plosive phones.
  Yet there are situations in spoken language in which the plosive phone is omitted, leaving only the preplosive pauses (for instance t_c followed by =m). In this case the preplosive pause is converted to the omitted plosive phone (in the example t_c becomes t).

- Diphthongs are merged. The SVOX labels do distinguish between the first and the second vowel of a diphthong, whereas the phonological descriptions of the speech signals don't. All diphthongs are therefore merged to a single phone. Example: In Listing 9 the lines 3 and 4 (a=a_i and i=a_i) are converted to a single line with phone symbol a_i.

- A pause symbol at the beginning is forced, thus we may introduce one with duration 0. This is done in order to match the phone feature file described below, which always contains a pause at the beginning of a sentence.

Listing 9: SVOX label file (.lab)

```
1 0 1160000 \G\t_c
2 1160000 2070000 \G\t_s
3 2070000 3200000 \G\a=a_i
4 3200000 4230000 \G\i=a_i
5 4230000 4670000 \G\t_c
6 4670000 5120000 \G\t
7 ...
```

The text transcription files from the voice database are analyzed and converted to the phonological representation by the SVOX transcription stage (see Section 1.3.1). An interface between the SVOX phonological representation (cf. Listing 4) and the MaryXML ALLOPHONES (cf. Listing 2) file has been implemented as method `svoxPtr2maryAllophones` of the Java class `SVOX2MARY`. The shell script

`svoxmaryVoiceBuilding.sh`

shows how the method can be called. It converts all files in `/newVoiceDir/ptr/*.ptr` to `/newVoiceDir/allophones/*.xml`. Again, when foreign language tags are found in the phonological description, the files are not converted. The method will then write to the file `/newVoiceDir/basenames.lst` the name of those files that eventually have been converted. The `basenames.lst` file is read at the launch of the MARY voice import component and sets which files are used for the voice training.

Now it does not suffice to input the MaryXML ALLOPHONES files to the existent MARY system. Since the SVOX system uses a different phonetic alphabet (ETHPA vs. SAMPA), the phone unit feature computer (see Section 2.1) is not able to compute the phone features with the given German locale `de`. The MARY phoneme set for German is contained in

`$MARY_BASE/lib/modules/de/lexicon/allophones.de.xml`

and lists the phonetic features for each phoneme. This is a key file for the whole MARY system. It turned out that just by changing this phoneme set, i.e. adding the ETHPA symbols not contained in SAMPA, the mary server still cannot compute the features. It has been necessary to create a new language under the locale we called `de1`. The following files have thus been added to the MARY system:

- `$MARY_BASE/lib/modules/de1/lexicon/allophones.de1.xml` new directory `de1/lexicon/` with ETHPA adapted phoneme set (see **Listing 10**). The features and their possible values are as follows:
  `cplace`: 0-n/a `l`-labial `a`-alveolar `p`-palatal `b`-labio dental `d`-dental `v`-velar `g`-?
  `ctype`: 0-n/a `s`-stop `f`-fricative `a`-affricative `n`-nasal `l`-liquid `r`-r
  `cvox`: 0=n/a +=on -=off
  `vfront`: 0-n/a 1-front 2-mid 3-back
  `vheight`: 0-n/a 1-high 2-mid 3-low
  `vlng`: 0-n/a `s`-short `l`-long `d`-diphthong `a`-schwa
  `vrnd`: 0=n/a +=on -=off

Listing 10: allophones.de1.xml

```
<allophones name="ethpa" xml:lang="de1"
      features="vlng vheight vfront vrnd ctype cplace cvox">

   <silence ph="/"/>

   <vowel ph="i:" vlng="l" vheight="1" vfront="1" vrnd="-"/>
   <vowel ph="i" vlng="s" vheight="1" vfront="1" vrnd="-"/>
   <vowel ph="y:" vlng="l" vheight="1" vfront="2" vrnd="+"/>
   <vowel ph="y" vlng="s" vheight="1" vfront="2" vrnd="+"/>
   ...

   <consonant ph="t_h" ctype="s" cplace="a" cvox="-"/>
   <consonant ph="t_S" ctype="a" cplace="p" cvox="-"/>
   <consonant ph="t_s" ctype="a" cplace="a" cvox="-"/>

</allophones>
```

- `$MARY_BASE/java/marytts/language/de1/features/...` `/FeatureProcessorManager.java` new Feature Processor for the locale `de1`, which is a copy of the existing one with some minor changes (substitute `de` with `de1` where it happens, the pause symbol `_` with `/`, and `return Locale.GERMAN` with `return new Locale("de1")`).

- `$MARY_BASE/conf/de1.config` which sets the new locale and contains the paths to the two files above.

After a rebuild of the MARY system the moment has come to launch the voice import component and continue the voice building with the tools in there. In the global voice settings the locale variable must now be changed to `de1`. The processing steps are as follows:

Feature Selection as in Section 2.1.

Phone Unit Feature Computer computes the phone features from the MaryXML ALLOPHONES files using the new feature processor for the locale `de1`.

Phone Label Feature Aligner compares the phone labels with the phone features to find mismatches in the phone sequence. The mismatch here are due to

- glottal stops (phone symbol `?`) are ignored in the phonological representation (and thus in the phone feature file) but not in the SVOX labels. This can be solved using label files without glottal stops.
- longer sentences usually have pauses in the speech signal, and these pauses of course show up in the label file, but not in the phonological representation. By clicking on "solve all problems" the problematic files are neglected for further training.

HMM Voice components are used as in Section 2.1 to extract acoustic parameters from the speech files and to train and set up the HMM voice.

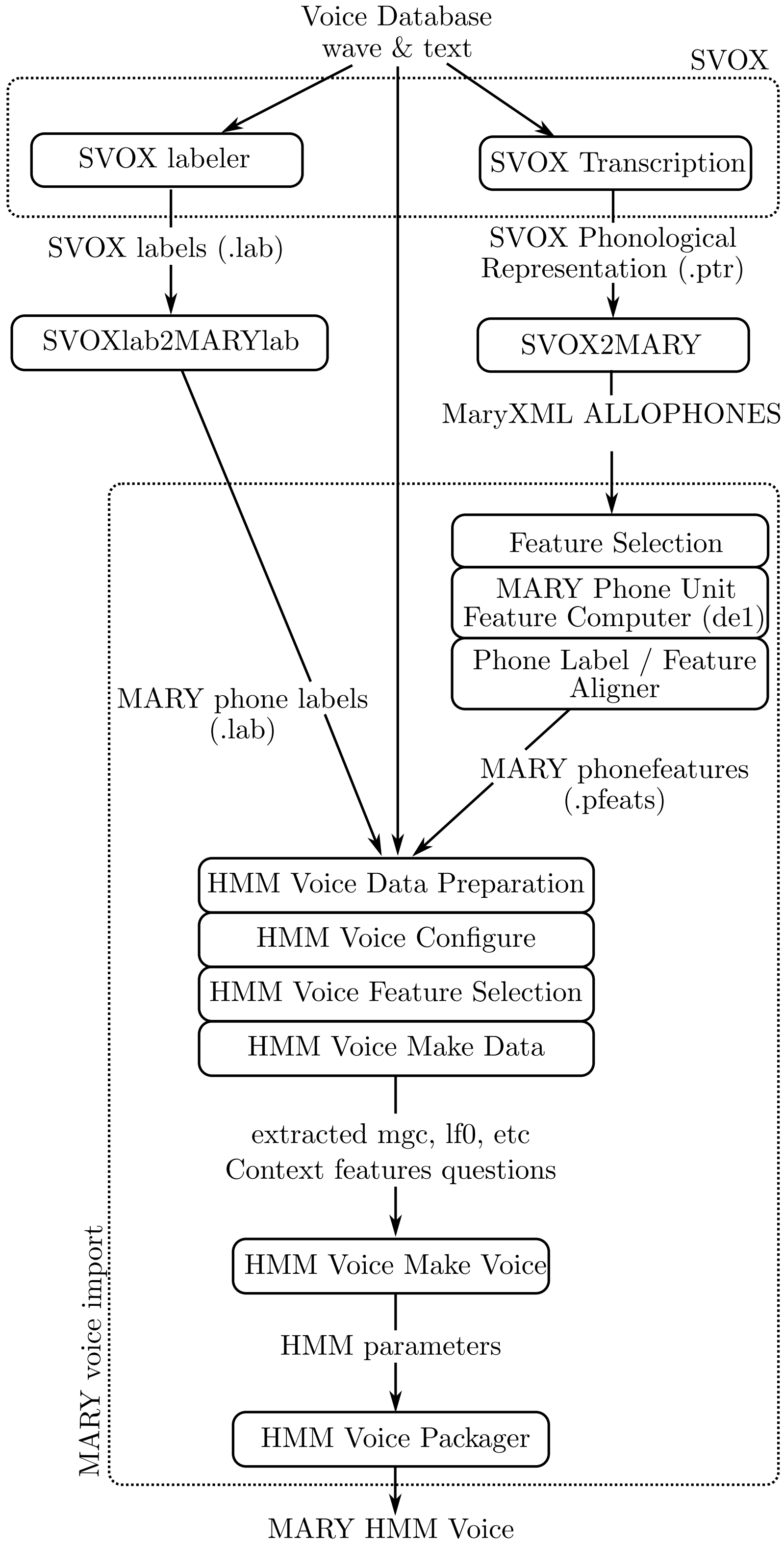


Figure 2.2: Combined SVOX-MARY HMM-based voice training.

2.2.2 SVOX HMM-based Synthesis

Figure 2.3 shows the architecture of the implemented SVOX-MARY HMM-based speech synthesis. The SVOX TTS system provides the text analysis which results in the phonological representation, and generates the prosodic parameters (duration and fundamental frequency), resulting in a phone sequence (see Section 1.3). An interface has been implemented in Java (method `svoxPtrPhones2maryAcoustparams` from the class `SVOX2MARY`) which converts the SVOX phonological description (cf. Listing 4) and phone sequence (cf. Listing 5) to a MaryXML ACOUSTPARAMS file (cf. Listing 3).

The method does some processing on the phone sequence: it merges the preplausive pauses (symbol `>`) with the subsequent plosive phone (as was done for the voice training). Otherwise the method is very similar to the `svoxPtr2maryAllophones` method used in the previous section. Remember that, compared to the ALLOPHONES format, the MaryXML ACOUSTPARAMS format has just some additional informations on phone duration and frequency. These informations are extracted from the SVOX phone sequence file.

The `SVOX2MARY` class then sends the ACOUSTPARAMS file to the MARY server and requests to write an audio wavefile. The MARY server must be running for this step, as it takes care of the HMM-based speech generation. The shell script

```
svoxmaryTTS.sh
```

lists all the necessary calls: twice SVOX (convert `.txt` to `.ptr` and `.txt` to `.phones`), and once `SVOX2MARY` to convert the SVOX files to ACOUSTPARAMS and eventually to get the audiofile.

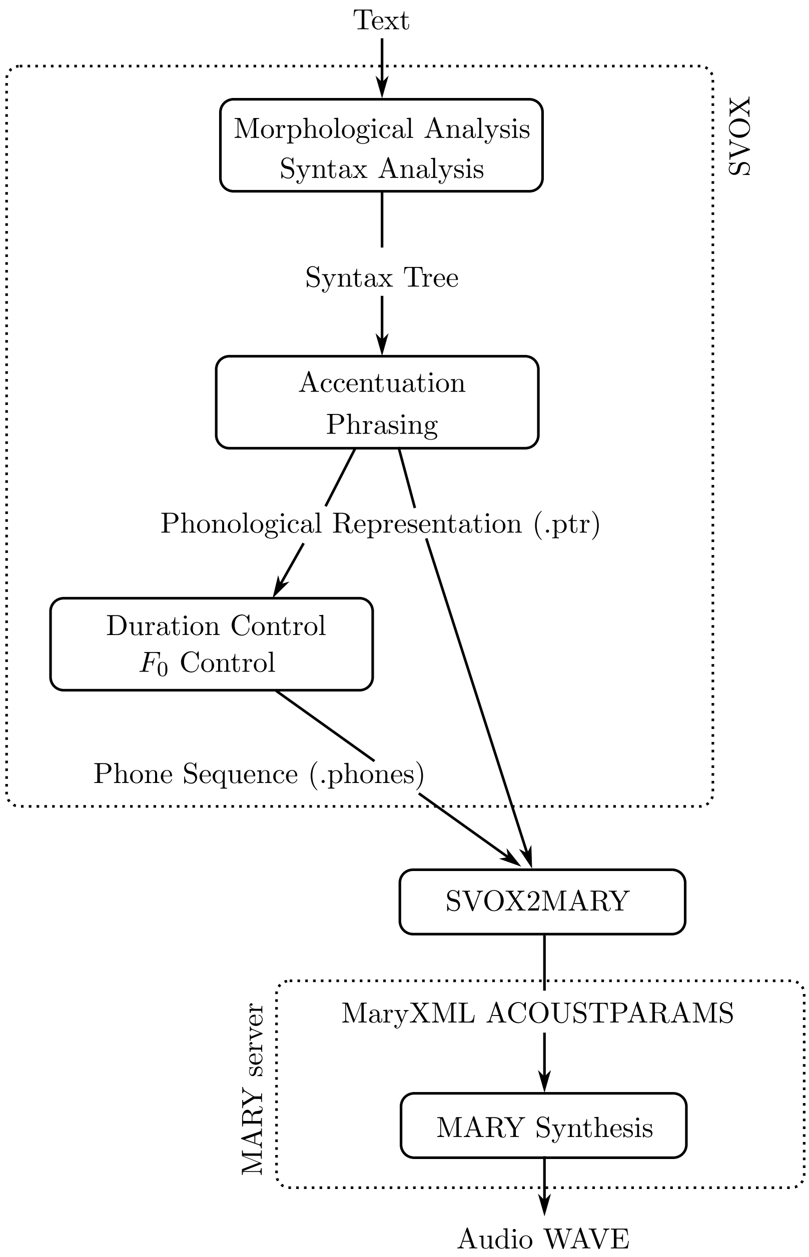


Figure 2.3: Combined SVOX-MARY speech synthesis.

3 Evaluation

Part of this project would have the evaluation and comparison of the two voices built, namely the one using just the MARY system (Section 2.1) and the voice built using a combined SVOX-MARY architecture (Section 2.2). Particularly the prosody control (duration and fundamental frequency) should have been investigated, as the MARY rules based computation of the acoustic parameters differs very much from the neural network approach used in SVOX.

To anticipate it right away: at the moment this thesis is being written the whole SVOX prosody control is being redesigned for multilingual support and is still under developement. Thus the SVOX transcription stage and the duration and frequency control is far from being optimal, which makes a thorough prosody comparison meaningless. This can be seen in Listing 4, where the phonological transcription of `Hallo` yields `ha:?al?o:`, which of course is not a correct transcription.

There are still interesting points to make though. For the HMM training in Section 2.1 a database of 1380 speech signals has been used. In order to speed up the training steps in Section 2.2 a subset of just 200 files has been chosen. The comparison between the two voices not surprisingly shows an improved sound quality for the first voice. More training data gives better HMM-parameters which reflects in better speech quality. What could be somehow surprising is a comparison between the SVOX unit selection synthesis and the new SVOX HMM-based synthesis. They both use the same acoustical parameter, a fair comparison can therefore be made. It turns out that even with only 200 training files the sound quality of the HMM-based voice is subjectively almost as good as the unit selection. Of course the quality depends on how often the phones that needs to be synthesized were present in the chosen training files. Still it shows that HMM-based synthesis may very much be a valuable speech generation approach, definitely catching up with the state-of-the-art unit selection.

4 Conclusion

In this thesis the integration of an HMM-based voice generation component to the text-to-speech system SVOX (developed at the ETHZ Speech Processing Lab) has been presented.
In a first part a new German HMM voice has been trained using the text-to-speech system MARY. In a second part an interface between SVOX and MARY has been designed, which allows to use the MARY HMM-based speech generation within SVOX. The designed architecture has been successfully implemented.

4.1 Some Technical Suggestions for Improvement

The focus of this project has been put on making the system work. The interface between SVOX and MARY has thus been kept simple enough. Now of course there are some ideas on how the system could be fine tuned, improving the overall performance. Here comes a list with some suggestions.

- As seen in Section 2.2.1, SVOX label files do separate the preplosive pause from the subsequent plosive phone, whereas the phonological transcriptions don't. Instead of merging the preplosive pause and the plosive phone in the label file, thus losing information, one could add preplosive pauses to the phonological transcription, in order to match the label files. This way HMM-models for preplosive pauses would be trained as well, and better synthesis of plosive phones could be expected.

- The selection of features used to build the context tree for the HMM models could be better adapted to the SVOX system. The feature list suggested by MARY includes various ToBI features, which are not provided by the SVOX transcription stage (and thus not considered for the HMM training). Instead, other features based on the SVOX accent and phrasing analysis could be computed. The feature computer in the MARY system would need to be adapted for this purpose. Further it may be necessary to adapt the HTS training scripts as well to the new context features.

- There is often a mismatch between the computed phone feature files and the label files (see Phone Label Feature Aligner in Section 2.2.1). This is because in natural speech pauses are introduced when the sentences are long; these pauses show up in the label file but not in the phonological representation. Instead of just ignoring all these problematic files, reducing the amount of training data, one could think of solving this issue

- by hand, adding pauses in the feature file to match the label files
- by processing the label files and merge the pauses to the subsequent phone
- by implementing some algorithm in the `SVOX2MARY` component, which from the phonological representation predicts where a speaking person would introduce a pause.

The first method is prohibitive for a large speech database. The second probably would give suboptimal training results, since the HMM training would include a pause to the actual phone. The third is the most interesting yet not so simple solution.

References

[1] K. Tokuda, H. Zen, and A. Black, "An HMM-based speech synthesis system applied to english," in *Proceedings of 2002 IEEE Workshop on Speech Synthesis.* IEEE, 2002, pp. 227–230.

[2] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech parameter generation algorithms for HMM-based speech synthesis," in *Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on*, vol. 3. IEEE, 2000, pp. 1315–1318.

[3] M. Tamura, T. Masuko, K. Tokuda, and T. Kobayashi, "Adaptation of pitch and spectrum for HMM-based speech synthesis using MLLR," in *Acoustics, Speech, and Signal Processing, 2001. Proceedings. ICASSP'01. 2001 IEEE International Conference on*, vol. 2. IEEE, 2001, pp. 805–808.

[4] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, "Simultaneous modeling of spectrum, pitch and duration in HMM-based speech synthesis," in *Sixth European Conference on Speech Communication and Technology*, 1999.

[5] K. Tokuda, T. Masuko, N. Miyazaki, and T. Kobayashi, "Multi-space probability distribution HMM," *IEICE Transactions on Information and Systems E series D*, vol. 85, no. 3, pp. 455–464, 2002.

[6] M. Schröder and J. Trouvain, "The text-to-speech synthesis system MARY: A tool for research, development and teaching," *International Journal of Speech Technology*, vol. 6, pp. 365–377, 2003.

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

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

[9] T. Ewender, "Information flow and architecture of the SVOX software." 2010, ETH Speech Processing Lab Internal Documentation.

[10] S. Pammi, M. Charfuelan, M. Schröder, and O. Türk, "Voice building tool for the MARY TTS platform."

A Encountered Issues

As a vast amount of time in this project has been spent in debugging, the following call of the MARY server has come in handy:

```
$MARYBASE/bin/maryserver -Dlog4j.logger.marytts=DEBUG,stderr’
```

This sets the level of logging informations to DEBUG.

B Task Description

Institut für
Technische Informatik und
Kommunikationsnetze

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich
Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo

Spring semester 2011
(SA-2011-02)

Task Definition

for

Simon Würgler

Main Reader: T. Ewender, ETZ D97.7
S. Hoffmann, ETZ D97.5

Issue Date: 21. February 2011
Submission Date: 3. June 2011

Implementation and evaluation of an HMM-based speech generation component for the SVOX TTS system

Background

Parametric speech synthesis based on hidden Markov models (HMMs) has emerged in the recent years as an alternative approach to concatenative speech synthesis techniques like diphone or unit selection synthesis.

Unit selection has been shown to synthesise high quality speech and is currently the most popular speech synthesis technique with many applications. It is very hard to surpass the quality of the best examples. However, unit selection is subject to the limitation that it does not allow flexible control over the speech variation.

Over the last few years, HMM-based synthesis systems have grown in popularity. In these systems, context-dependent HMMs are trained from databases of natural speech, and speech (in the form of a feature sequence) is generated from the HMMs themselves. The most attractive part of HMM-based synthesis systems is that its voice characteristics, speaking styles, or emotions can be modified by transforming HMM parameters without requiring the recording of very large databases.

One of the latest open source implementations of HMM-based speech generation systems is the HMM-based Speech Synthesis System (HTS) [1], which has first been published in 2002. The HTS API has been ported to Java and has been included into the MARY text-to-speech (TTS) system [2], which is also available as open source.

Problem Definition

The objective of this thesis is to adopt and evaluate HMM-based speech synthesis for the speech group's text-to-speech system SVOX. This task encompasses several steps:

- analyse the possibilities to integrate an existing HMM-based voice into the SVOX system
- analyse the interfaces of the software modules to be connected: the natural language processing component of SVOX and the speech generation component of MARY
- design an interface between the two modules at the phone description level (which encompasses the phone information and the acoustic parameters of the prosody)
- implement an HMM-based speech synthesis module based on the MARY TTS system in Java
- evaluate the quality (segmental/prosodic aspects) of the resulting synthesis system

Approach

Building a German voice for the MARY TTS system

The first part of this thesis is the preparation and training of a German voice that is to be used with the MARY TTS system. The preparation and training of a voice for HMM synthesis consists of various steps (illustrated in the upper part of Figure 1):

- Segmentation (automatic labelling)
- Excitation parameter extraction
- Spectral parameter extraction
- Training of HMMs

For the segmentation task and partly for the excitation parameter extraction tools of the speech processing group can be used. For the other tasks, speaker-dependent training scripts are provided by HTS (`http://hts.sp.nitech.ac.jp/?Download`, see also [3]).

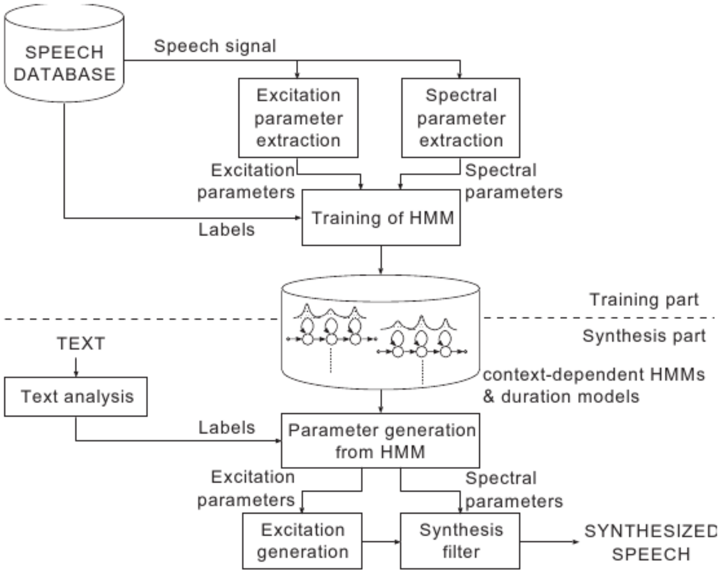


Figure 1: Overview of an generic HMM-based speech synthesis system (Fig. taken from [1])

Implementation of a HMM-based speech generation module for SVOX

In the second part of this thesis, the speech generation module of the MARY TTS system has to be adopted to be used with the language processing component of SVOX. More precisely, the natural language processing and the prosodic parameter generation should be done by the SVOX components. The output of the SVOX component at the phone description level, which encompasses the phone information and the acoustic parameters of the prosody, should then be used as input to the MARY speech generation module. Consequently, the MARY speech synthesis module has to be adapted to a new feature set based on the acoustic parameters, which replaces the context feature set usually used for the spectral parameter generation. This new feature set has also to be used to build a voice for that combined system.

Prosodic modelling for HMM-based synthesis is usually done using the context features of the HMMs. Hence prosodic parameters (duration and fundamental frequency) are implicitly modelled by the HMMs which are selected for synthesis. In our system, prosody is explicitly modelled by our neural network-based prosody models. By connecting the two systems at the phone description level as described above, an improved prosody generation for HMM-based synthesis can be realised.

The synthesis results obtained with the MARY TTS system for the voice built in the first part of this thesis are finally to be compared to the synthesis results obtained in the second part of this thesis that use the SVOX prosodic modelling.

The work performed and the results obtained have to be documented in a report (see [4]), which is to be delivered in electronic form (PDF file). In addition, two presentations are to be held in a colloquium: the work plan is to be presented about three weeks into the project, the results at the end of the project. The dates will be announced later.

Literature

[1] H. Zen, T. Nose, J. Yamagishi, S. Sako, T. Masuko, A.W. Black, and K. Tokuda. The HMM-based speech synthesis system (HTS) version 2.0. In *Proc. of Sixth ISCA Workshop on Speech Synthesis*, pages 294–299, 2007.

[2] M. Schröder and J. Trouvain. The German text-to-speech synthesis system MARY: A tool for research, development and teaching. *International Journal of Speech Technology*, 6(4):365–377, 2003.

[3] S. Pammi, M. Charfuelan, and M. Schröder. Multilingual voice creation toolkit for the MARY TTS platform. In *Proc. Int. Conf. Language Resources and Evaluation*, Malta, 2010.

[4] B. Pfister. *Richtlinien für das Verfassen des Berichtes zu einer Semester- oder Diplomarbeit.* Institut TIK, ETH Zürich, März 2004.
(http://www.tik.ee.ethz.ch/spr/SADA/richtlinien_bericht.pdf).

[5] B. Pfister. *Hinweise für die Präsentation der Semester- oder Diplomarbeit.* Institut TIK, ETH Zürich, März 2004.
(http://www.tik.ee.ethz.ch/spr/SADA/hinweise_praesentation.pdf).

[6] B. Pfister und T. Kaufmann. *Sprachverarbeitung: Grundlagen und Methoden der Sprachsynthese und Spracherkennung.* Springer Verlag (ISBN: 978-3-540-75909-6), 2008.

February 18, 2011