9th ISCA Speech Synthesis Workshop
13-15 Sep 2016, Sunnyvale, USA

Development of a statistical parametric synthesis system for operatic singing in German

Michael Pucher[1], *Fernando Villavicencio*[2], *Junichi Yamagishi*[2,3]

[1]Acoustics Research Institute, Austrian Academy of Sciences, Vienna, Austria
[2]National Institute of Informatics, Japan
[3]Centre for Speech Technology Research, University of Edinburgh, UK
michael.pucher@oeaw.ac.at {villavicencio,jyamagis}@nii.ac.jp

Abstract

In this paper we describe the development of a Hidden Markov Model (HMM) based synthesis system for operatic singing in German, which is an extension of the HMM-based synthesis system for popular songs in Japanese and English called "Sinsy". The implementation of this system consists of German text analysis, lexicon and Letter-To-Sound (LTS) conversion, and syllable duplication, which enables us to convert a German MusicXML input into context-dependent labels for acoustic modelling.

Using the front-end, we develop two operatic singing voices, female mezzo-soprano and male bass voices, based on our new database, which consists of singing data of professional opera singers based in Vienna. We describe the details of the database and the recording procedure that is used to acquire singing data of four opera singers in German.

For HMM training, we adopt a singer (speaker)-dependent training procedure. For duration modelling we propose a simple method that hierarchically constrains note durations by the overall utterance duration and then constrains phone durations by the synthesised note duration. We evaluate the performance of the voices with two vibrato modelling methods that have been proposed in the literature and show that HMM-based vibrato modelling can improve the overall quality.

Index Terms: singing synthesis, opera singing, vibrato modelling

1. Introduction

By singing synthesis we refer to the task of generating an acoustic signal of a singing person. A text input and a musical score that is aligned with the text thereby control the synthesis output. The textual data is divided into sequences of syllables. This input data can be given as MusicXML [1] file encoding a musical score as shown in Figure 1. Several methods for synthesis of singing have been proposed in the literature, like articulatory singing synthesis [2], formant based singing synthesis [3], diphone-based singing synthesis [4], or HMM-based singing synthesis [5, 6, 7, 8]. We will work within the HMM-based paradigm and extend Sinsy [7] with a German front-end and acoustic models.

An articulatory singing synthesis system [2] takes gestural scores, which are fed into models of the vocal tract and the vocal folds that produce a dynamic feature trajectory. This trajectory can then be used to control the synthesiser. The gestural score can either be created by hand or by a rule-based system. One disadvantage of this kind of synthesiser is the difficulty to obtain

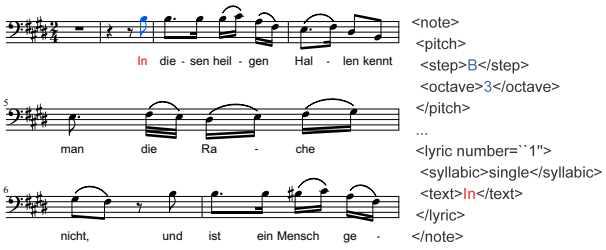

Figure 1: Musical score of Sarastros 2nd Aria "In diesen heilgen Hallen" from "Die Zauberflöte" by W. A. Mozart (left). Part of musical score in MusicXML (right).

enough articulatory data to train models automatically, as can be done with acoustic recordings.

The rule-based formant synthesis system proposed in [3] has the advantage that it can be used without any acoustic training data. The system takes text data and musical score as input and generates 28 parameters at 100 Hertz. These parameters are then used to control the formant synthesiser.

Waveform concatenation methods [4] have also been applied for singing synthesis. At recording time all possible combinations of consonant-vowel, vowel-consonant, vowel-vowel have to be recorded. This technology was developed by Yamaha and is licensed to other companies that sell commercial versions of singing synthesisers. In synthesis the pitch of the selected units has to be changed to the desired pitch and the timbre has to be smoothed at concatenation points. All this is done in the frequency domain. For changing the pitch the power spectrum is divided into different regions, which are then scaled to the desired pitch.

The method used in this paper is acoustic singing synthesis within the HMM framework [5, 6, 7, 8]. The main advantages of this method are that it is data-driven, flexible, and can use many techniques developed for HMM-based speech synthesis. HMM-based singing synthesis uses the maximum likelihood parameter generation algorithm [9] to generate the necessary parameters for singing. The context clustering is extended to take features derived from the musical score into account (current, previous, following notes, duration, etc.).

In such a corpus-based approach machine learning methods are used to learn a singing model from pre-recorded singing data. A general problem for corpus-based approaches to singing synthesis is the prediction of lyrics, phrases and expressions that are not covered in the training data. Operatic singing styles pose additional modelling problems due to the dynamic and largely

http://dx.doi.org/10.21437/SSW.2016-11

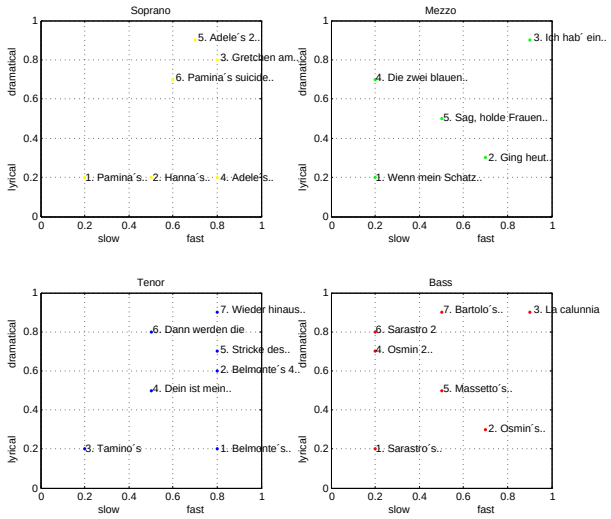

Figure 2: Classification of opera songs according to lyrical - dramatic and slow - fast dimension.

varying F0 and power trajectories, which would require very precise temporal modelling techniques. The mezzo-soprano and soprano may use very high F0 values, which also poses a challenge for spectral and excitation analysis and extraction [10]. A further challenge is the modelling of duration that can vary significantly between and within different opera pieces.

This paper describes our first attempts to develop a HMM based synthesis systems for such challenging operatic singing. The first steps which we report in this paper include the creation of a new opera singing database in Section 2, the development of a German front-end in Section 3, training of two operatic singing voices, female mezzo-soprano and male bass voices, using HMMs estimated on the new database in Section 4, and vibrato modelling and evaluation in Section 5.

2. Singer and song selection

For the recording of opera singers, we have consulted a professional opera-singing teacher who performed the selection of songs and singers. Differently to speech recordings the selection of songs and singers is tightly coupled. In standard speech synthesis we would select a corpus with an optimal phone coverage by finding an approximate solution for the associated minimum set-cover problem [11]. For the solution of this problem we can take different features like diphones, diphones in stressed syllables and so on into account [12]. For finding this corpus we need a large phonetically transcribed background corpus. A speaker that is selected in a separate selection process would then read this corpus.

For selecting opera songs, we could go the same way, provided that we have a large amount of opera songs in MusicXML format. However, with such a selection process we would end up with a selection of opera songs that no opera singer has available in his/her repertoire at the moment. This would possibly result in lower quality recordings. So the selection of singers and songs has to go hand in hand by using a different strategy.

Therefore we have decided to select a number of opera songs ($\approx$ 8-10) for each singer category that are in the repertoire of that singer at the moment and that cover the space of opera songs along the lyrical - dramatic and slow - fast axes. We also checked that these songs cover the F0 ranges of that

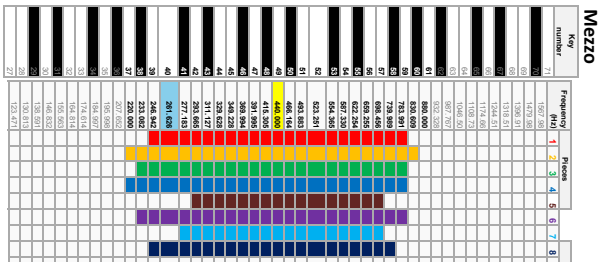

Figure 3: F0 range for mezzo and opera songs shown on the piano roll.

singer category.

Figure 2 shows the subjective classification of opera songs along the two dimensions lyrical - dramatic and slow - fast. The classification of opera songs was subjectively done by a professional opera-singing teacher. As can be seen in Figure 2 the opera songs that were then recorded cover the whole space for mezzo and bass but the slow and dramatic category is in general difficult to find and especially difficult for the soprano and tenor (although the opera teacher did his best consideration). A further restriction in the selection of songs was the fact that we were only looking for songs in the German language.

Figure 3 shows the general F0 range of the mezzo singer with bold numbers on the piano roll ranging from A3 with 220 Hz to A5 with 880 Hz. The coloured bars show the F0 ranges for our 8 selected opera songs. Song 1 for example has a range from B3 (=246 Hz) to G5 (=783 Hz). We can see from Figure 3 that our songs cover the F0 range almost completely with the exception of one note, the highest note with 880 Hz, which is not existent in the training data.

3. German front-end for Sinsy

The main extensions to the Sinsy system for German were the

- analysis of German input text (text analysis),
- conversion of input words into phonetic sequences (lexicon and letter-to-sound conversion), and
- duplication of syllables where syllables had more than one note (syllable duplication).

3.1. Text analysis

In parsing the data from the MusicXML file the task of text analysis consists in the reconstruction of words that can then be used to access the lexicon. The <begin>, <middle>, and <end> tag inside the MusicXML <lyric> tag are used to mark the specific syllables of the word. The <single> tag marks a word with just one syllable.

3.2. Lexicon and letter-to-sound conversion

Words are collected from the MusicXML files and the phonetic transcription and syllable boundaries are taken from the lexicon if possible. If the word is not found in the lexicon LTS rules are used directly on the syllables to avoid the need for syllabification.

We integrate a lexicon and rules for LTS conversion from an open-source synthetic voice for Austrian German [13]. The LTS rules consist of a set of decision trees, one tree for each letter, that are used to convert a given input character sequence (word, syllable) into the corresponding output phones.

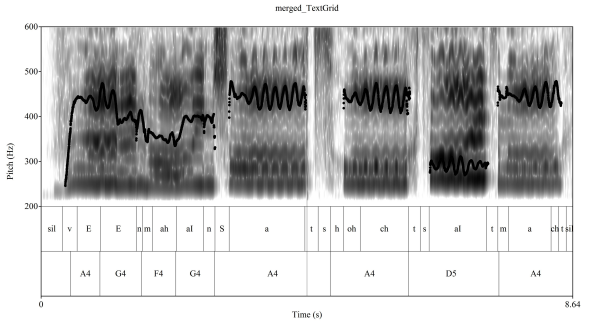

Figure 4: Alignment of musical score and phone labels on utterance level for the utterance "Wenn mein Schatz Hochzeit macht" from G. Mahler's "Lieder eines fahrenden Gesellen" from the mezzo-soprano voice.

If a word is not found in the lexicon, syllables from the MusicXML files are used directly for LTS conversion. For each syllable we are using the decision trees for predicting the corresponding phone sequence. Another approach would be to put syllables together into words, apply LTS conversion to words and split the resulting phone sequence again into syllables. By skipping the last syllabification step, where a sequence of letters is broken up into syllables we achieve a more robust prediction. Furthermore there are sometimes MusicXML files where the word level annotation is wrong, such that merging syllables into words leads to wrong or non-existing words. By starting directly from syllables, we can also alleviate this problem. Through the integration of LTS rules into the system we are able to synthesise from any German MusicXML file.

3.3. Syllable duplication

Syllable duplication was not implemented in the open-source Sinsy system, so we had to do this for German. We followed the method proposed in [14]. Figure 4 shows an example alignment for the sentence "Wenn mein Schatz Hochzeit macht", phonetically "v E n . m aI n . S a t z . h oh ch t s aI t . m a ch t" where '.' signifies word boundaries. The alignment also contains the silence symbol "sil" at the beginning and end of the utterance.

At the top we can see the spectrogram of the respective audio signal as well as the F0/pitch curve in Hz. Below we have the phonetic transcription aligned and the musical notes aligned. We see that the notes start with A4, which has 440 Hz. The F0 curve shows that this target is reached with vibrato around 440 Hz.

The transcription shown here is already after text analysis and letter-to-sound conversion. We can see that the word "wenn" ("v E n") is distributed across two notes A4 and G4 such that the syllable duplication turns it into "v E E n" phonetically. The word "mein" ("m aI n") is also distributed across two notes F4 and G4. Through syllable duplication it is turned into "m ah aI n" where we take into account that we do not duplicate diphthongs like "aI".

For the duplication of syllables with diphthongs we use simple duplication rules. When duplicating an "aI" n times for example, we generate $n-1$ times an "ah" (a long "a") followed by one "aI". Prefix and postfix are used from the original syllable. Our notation is closely related to the SAMPA standard [15].

Algorithm 1 shows the algorithm for syllable duplication. After pre-processing we have inserted pause symbols "pau" at the phones inside of slurs. For the example of the word "mein" mentioned above the sequence after pre-processing is "m aI pau n". The algorithm now goes through all syllables and phones within a slur (slur begin to slur end). The first vowel is replaced if it is a diphthong, so the sequence is "m ah pau n" after this step. If we are at the last pause in the slur, which we can check be checking if we are at the last note, we replace the pause by the respective diphthong. This leaves us with the final sequence "m ah aI n", which is the desired result.

Algorithm 1 Algorithm for duplicating syllables.

```
check that slur does not span across words
for note ← slur begin, slur end do
    for syl ← syl begin, syl end do
        for phone ← phone begin, phone end do
            if phone is the first vowel found then
                replace phone if it is a diphthong
            else
                if phone is a pause after vowel was found then
                    if not at the last pause in the slur then
                        replace pause by respective phone
                    else
                        replace pause by diphthong or phone
                    end if
                end if
            end if
            if last note and last syllable in slur then
                add remaining phones
            end if
        end for
    end for
end for
```

At the beginning the algorithm also checks if the slur does not span across words. In this case we cannot do syllable duplication with our algorithm. The algorithm does of course also work if there are more than two notes that are to be distributed across a syllable. If we have to sing "m aI n" with four different notes, the conversion would be from "m aI pau pau pau n" to "m ah ah ah aI n".

4. Mezzo-soprano and bass voice building

4.1. Corpora

The recording of the corpora was performed in a professional recording studio where the opera singer and the piano accompaniment player were recorded in two separate studios such that the singer was able to see the piano player. Since this is an uncommon recording setting for opera singers we had to perform some test recordings at the beginning.

For training the mezzo voices we had 8 different opera songs. After splitting the recordings into utterances we had 154 different utterances. 8 utterances were taken as test sentences and were not used for training. For the bass voice we had also 8 different songs, which were split into 358 utterances, with 8 test utterances. As shown in Table 1 we had songs from five different composers for the mezzo voice with a total duration of 25.8 minutes. For the bass voice we had 28.9 minutes, from three different composers as shown in Table 2. We have also corpora for a tenor (male) and a soprano (female) voice, which were however not used for the acoustic modelling experiments since we focused on the "simpler" singer categories at the beginning.

Table 1: Songs recorded for the mezzo voice. The table also shows the maximum and minimum F0 according to the MusicXML file.

Composer	Singer	Dur.	min(F0)	max(F0)
G.Mahler	Mez	214	246	783
G.Mahler	Mez	235	220	783
G.Mahler	Mez	201	246	783
G.Mahler	Mez	318	220	783
Mozart	Mez	170	261	698
Korngold	Mez	142	220	698
Humperd.	Mez	140	261	879
A.Mahler	Mez	128	246	698
		25.8 m.		

Table 2: Songs recorded for the bass voice. The table also shows the maximum and minimum F0 according to the MusicXML file.

Composer	Singer	Dur.	min(F0)	max(F0)
Mozart	Bass	146	87	261
Mozart	Bass	275	87	349
Lortzing	Bass	343	123	329
Mozart	Bass	223	73	329
Mozart	Bass	133	130	261
Mozart	Bass	226	87	261
Mozart	Bass	211	97	329
Beethoven	Bass	174	123	293
		28.9 m.		

4.2. Acoustic modelling

For training acoustic models for opera singing we adapted an existing training script for Japanese acoustic model training [16] that was released in December 2015. The model training follows the speaker dependent singing synthesis system using STRAIGHT [17] for feature extraction and synthesis. [18] analysed the analysis/re-synthesis quality of different parametric representations for singing speech, where STRAIGHT achieved a good performance. It would also be interesting to evaluate statistical parametric opera singing synthesis with different vocoders, which is however out of the scope of this paper.

To adapt the training script for German we had to generate clustering questions for German. Using the clustering questions from a speech synthesis training script for German [13] and adopting it to the Japanese singing training script we generated a training script. As a language dependent feature we also added lexical stress, which is not part of the Japanese training script.

Using the minimum F0 value from the MusicXML file produced severe F0 extraction problems with the RAPT [19] and the STRAIGHT F0 estimation algorithms. Therefore the minimum F0 value was set to 50 for the mezzo voice and the bass voice. The maximum F0 was extracted from the MusicXML utterance files with adding 30 semitones. Adding semitones to the extracted maximum F0 was necessary to optimise the F0 extraction.

We hypothesised that an increased frequency warping value compared to 0.55 for Mel-cepstral features would improve acoustic modelling for the bass by better representation of lower frequencies, while a lower or negative value would improve frequency representation for the mezzo voice. We could verify this by an informal listening test for the bass ($fw = 0.7$), but could not verify it for the mezzo, where the Mel-cepstrum was used ($fw = 0.55$).

In the future we plan to investigate adaptive frequency warping methods for feature extraction. In [20] for example a spectral model is used that emphasises the peak of the spectral envelope at the so called "singing formant", which was shown by [21] to lie near 3 kHz.

For generating durations for synthesis we used the overall original duration from the recorded utterance d_{orig} and the note durations d_i from the utterance MusicXML file. Final note durations $\hat{d}_i$ where computed as

$$\hat{d}_i = d_i \frac{d_{\text{orig}}}{\sum d_i} \tag{1}$$

by scaling the MusicXML note duration to the original utterance duration.

Phone durations $p_{i,j}$ for phone j in note i were generated from HMM decision trees and the final phone duration $\hat{p}_{i,j}$ was computed by scaling with the final note duration $\hat{d}_i$.

$$\hat{p}_{i,j} = p_{i,j} \frac{\hat{d}_i}{\sum_j p_{i,j}}. \tag{2}$$

With this duration modelling method we can control the overall duration externally while still having a full synthesis of note and phone durations, which takes information from the MusicXML score and the HMM decision tree into account.

5. Vibrato modelling

After synthesising the F0 trajectory from the HMM model, we first apply median filtering to cope with F0 errors and then apply a vibrato model. In the following subsections, we explain and compare two methods for modelling of vibrato.

5.1. Global OVP modelling

Like in [20] we modelled overshoot, vibrato, and preparation (OVP) by a second order system

$$H(s) = \frac{k}{s^2 + 2\zeta\omega s + \omega^2} \tag{3}$$

with $0 < |\zeta| < 1$ for overshoot and preparation and $|\zeta = 0|$ for vibrato. The 8 global parameters were estimated from the full training data sequence by using MATLAB's non-linear least-squares solver. In [20] they used a non-linear least-squared-error method to minimise errors between the generated F0 contours and the actual ones. We were estimating the OVP parameters that minimise the distance between original F0 curves and synthesised and OVP filtered F0 curves on the training data.

$$\underset{\text{OVP}}{\text{argmin}} (\text{F0}_{\text{orig}} - \text{F0}_{\text{syn}}(\text{OVP}))^2. \tag{4}$$

To generate the synthesised F0_{syn} trajectories for the training data we did a forced alignment of the data using HSMMAlign. Then we converted the monophone labels to full context ones and used the full context labels to generate F0 parameters using HMGenS [22], which where then OVP filtered.

The 8 parameters were constrained to the interval $[0, 1]$. In synthesis we then applied the OVP filter after generating F0 curves from HMMs and median filtering to add the OVP structure. For the bass we estimated the following parameter set: $\omega = 0.0654 * 2\pi, \zeta = 0.6046, k = 0.0053$ for overshoot, $\omega = 0.0346 * 2\pi, k = 0.0011$ for vibrato, and $\omega = 0.0306 * 2\pi, \zeta = 0.9023, k = 0.0289$ for preparation. For the mezzo voice the

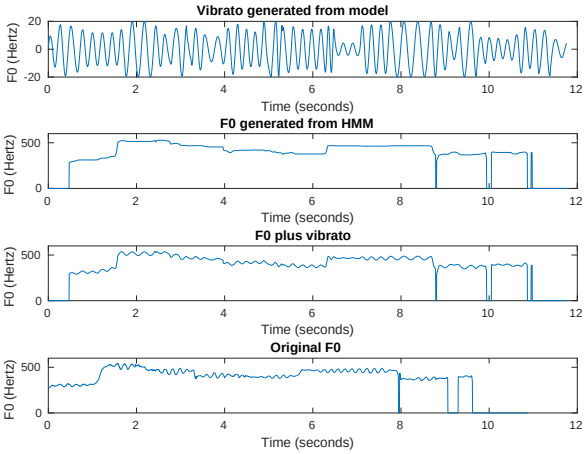

Figure 5: HMM-based vibrato modelling.

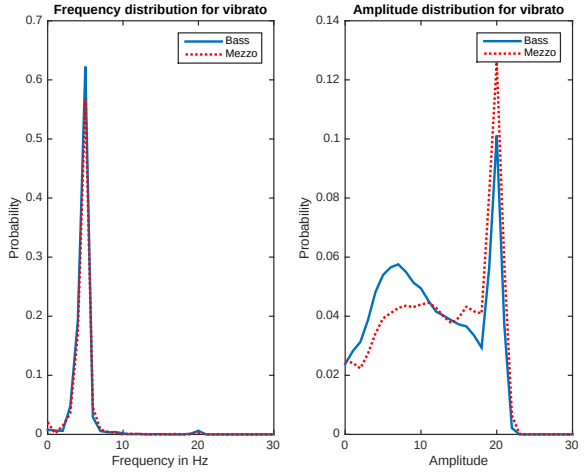

Figure 6: Frequency (left) and amplitude (right) distribution of estimated vibrato parameters for bass and mezzo voice.

values were $\omega = 0.0382 * 2\pi, \zeta = 0.1919, k = 0.0144$ for overshoot, $\omega = 0.0345 * 2\pi, k = 0.0002$ for vibrato, and $\omega = 0.0367 * 2\pi, \zeta = 1.0, k = 0.0385$ for preparation.

The advantage of this method is that we can estimate the OVP parameters fully automatic. A disadvantage is the averaging that is underestimating the vibrato at places where it is strong in the signal. Another possibility would be to estimate the parameters only for vibrato segments, which would however overestimate vibrato for the other segments and also lead to a less realistic modelling. Furthermore we would also need to detect vibrato first.

5.2. HMM vibrato modelling

The second method was to model vibrato as a time dependent periodic fluctuations of F0

$$v(m_a(t), m_f(t), i) = m_a(t) \sin(2\pi m_f(t) f_s(t - t_0)) \quad (5)$$

with $m_a(t)$ F0 amplitude in Cent, $m_f(t)$ F0 frequency in Hz, and f_s frame shift like it was proposed in [6].

The observation probability of the multi-stream HMM is

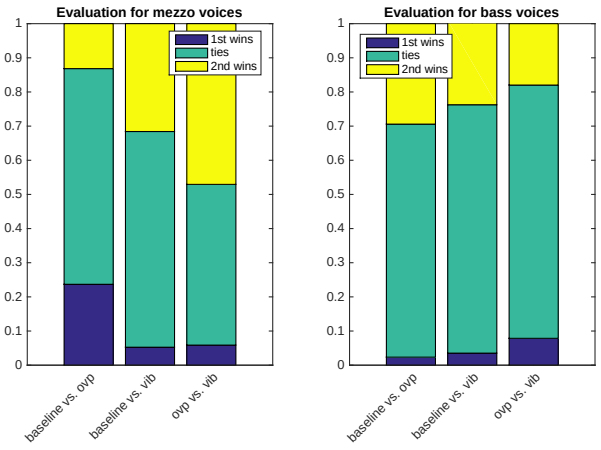

Figure 7: Evaluation results.

extended to

$$b(o_t) = p(o_t^{(spec)})^{\gamma_{spec}} p(o_t^{(bap)})^{\gamma_{bap}} p(o_t^{(F0)})^{\gamma_{F0}} p(o_t^{(vib)})^{\gamma_{vib}}. \quad (6)$$

by adding an additional stream for the vibrato model.

We optimised the frequency, phase, and amplitude parameter for sequences of 30 frames and used them to train a stream with frequency and amplitude parameters. The optimisation goal with the non-linear least-squares solver was again to minimise the distance between original F0 and synthesised F0 with the added frequency and amplitude varying sinusoids on the training data.

At synthesis time we created an amplitude and frequency varying sinusoid from the generated parameters like shown in Figure 5 (top), which was simply added to the generated F0 sequence (3rd from top). Figure 5 shows the frequency varying sinusoid, the F0 curve generated by the HMM, the HMM curve with the added amplitude and frequency varying sinusoid, and the original F0 curve. The difference between original and synthesised curve comes from additional pauses at beginning and end in the synthesised version.

Figure 6 shows the distribution for estimated frequency and amplitude parameters for the bass and the mezzo voice. Here we use a kernel density estimation with a normal kernel. Figure 6 left shows that the frequency is almost identically distributed for bass and mezzo voice around 5. Figure 6 right shows that there is a difference in the amplitude distribution between bass and mezzo with the mezzo having a higher probability of amplitudes around 20 and the bass having a higher probability of amplitudes around 7. The distribution depends of course on our assumption to model vibrato as an amplitude and frequency varying sinusoid.

5.3. Evaluation

For the evaluation we have used several mezzo and bass voices (mez_orig, mez_baseline, mez_ovp, mez_vib; bas_orig, bas_baseline, bas_ovp, bas_vib). Here *_orig are the original recordings, *_baseline is the voice with syllable duplication etc. but without any vibrato model, *_ovp uses the OVP based vibrato model with 8 global parameters, and *_vib uses the HMM-based vibrato model[1].

[1]Samples can be found on `http://speech.kfs.oeaw.ac.at/operassw16`.

We had 8 different listeners that had to perform a pair-wise comparison where they had to decide which voice is better. They were allowed to listen to the samples as often as they liked. All listeners were neither professional opera singers nor musicians. The task was to evaluate the overall quality of the singing voice.

Figure 7 shows the results for the evaluation of the mezzo (left) and the bass (right) voice. Since the original recordings always win over the synthesised ones we did not include these comparisons in the graph. This also shows that there is still a big room for improvement over the synthesised models.

In Figure 7 it can be seen that the HMM vibrato model wins over the baseline and the OVP model for the mezzo and the bass voice. We also see that the HMM-based vibrato model can improve over the global overshoot-vibrato-preparation model for both voices. The advantage of the HMM-based vibrato model lies in the context dependent estimation of singing vibrato.

6. Conclusion

We have described the development and evaluation of a Hidden Markov Model (HMM) based synthesis system for operatic singing in German. We have also discussed our procedure for singer and song selection and have described German front-end modules implemented for the Sinsy system including text analysis, LTS conversion, and syllable duplication.

Using this front-end, we have developed two operatic singing voices, female mezzo-soprano and male bass voices, based on a new database, which consists of singing data of professional opera singers based in Vienna. We described the details of the database and the recording procedure that was used to acquire singing data of four opera singers in German.

In acoustic modelling we have investigated two methods for vibrato modelling in opera singing synthesis. We show that the HMM-based vibrato modelling can improve the overall quality for the mezzo-soprano and the bass voice. We also saw that the HMM-based vibrato model could improve over the global overshoot-vibrato-preparation model where one global parameter set for the whole voice is estimated. For duration modelling we have proposed a method that hierarchically constrains note durations by the overall utterance duration, and constrains the phone durations by the synthesised note duration.

Although we need much further improvement before we can synthesise realistic opera singing, we showed how incremental improvements can be made in acoustic modelling for opera singing synthesis using a limited amount of training data. For further improvement we need to use more training data and incorporate more specific knowledge about operatic singing style into our models.

7. Acknowledgements

This research was partially funded by the Austrian Science Fund (FWF): P23821-N23. This work was also partially supported by EPSRC through Programme Grant EP/I031022/1 (NST) and EP/J002526/1 (CAF) and by the Core Research for Evolutional Science and Technology (CREST) from the Japan Science and Technology Agency (JST) (uDialogue project).

8. References

[1] Makemusic, "musicXML," http://www.musicxml.com/, 2014.

[2] P. Birkholz, "Articulatory synthesis of singing," in *INTERSPEECH 2007, 8th Annual Conference of the International Speech Communication Association, Antwerp, Belgium, August 27-31, 2007*, 2007, pp. 4001–4004.

[3] S. Ternström and J. Sundberg, "Formant-based synthesis of singing." in *INTERSPEECH*, 2007, pp. 4013–4014.

[4] H. Kenmochi and H. Ohshita, "Vocaloid - commercial singing synthesizer based on sample concatenation." in *INTERSPEECH*. ISCA, 2007, pp. 4009–4010.

[5] K. Saino, H. Zen, Y. Nankaku, A. Lee, and K. Tokuda, "An HMM-based singing voice synthesis system," in *Proc. Interspeech*, Pittsburgh, PA, USA, 2006, pp. 2274–2277.

[6] K. Oura, A. Mase, T. Yamada, S. Muto, Y. Nankaku, and K. Tokuda, "Recent development of the HMM-based singing voice synthesis system - Sinsy," in *SSW7*, Kyoto, Japan, 2010, pp. 211–216.

[7] Sinsy, "HMM-based singing voice synthesis system," http://sinsy.sourceforge.net/, 2013.

[8] K. Nakamura, K. Oura, Y. Nankaku, and K. Tokuda, "HMM-based singing voice synthesis and its application to Japanese and English," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, May 2014, pp. 265–269.

[9] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech parameter generation algorithms for HMM-based speech synthesis," in *Proc. ICASSP*, Istanbul, Turkey, June 2000, pp. 1315–1318.

[10] V. Fernando, B. Jordi, Y. Junichi, and P. Michael, "Efficient pitch estimation on natural opera-singing by a spectral correlation based strategy," *Information Processing Society of Japan SIG Technical Report*, vol. 2015, no. 1, pp. 1–6, 2015.

[11] C. Papadimitriou, *Computational Complexity*. Addison Wesley, 1994.

[12] M. Pucher, F. Neubarth, V. Strom, S. Moosmüller, G. Hofer, C. Kranzler, G. Schuchmann, and D. Schabus, "Resources for speech synthesis of Viennese varieties," in *LREC*, 2010, pp. 105–108.

[13] M. Toman, M. Pucher, and D. Schabus, "Austrian German voices for Festival," http://sourceforge.net/projects/at-festival/, 2013.

[14] K. Nakamura, K. Oura, Y. Nankaku, and K. Tokuda, "HMM-based singing voice synthesis and its application to Japanese and English," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 265–269.

[15] J. C. Wells *et al.*, "SAMPA computer readable phonetic alphabet," *Handbook of standards and resources for spoken language systems*, vol. 4, 1997.

[16] HTS, "Speaker dependent training demo - Japanese song," http://hts.sp.nitech.ac.jp/archives/2.3/HTS-demo_NIT-SONG070-F001.tar.bz2, 2015.

[17] H. Kawahara, I. Masuda-Katsuse, and A. de Cheveigné, "Restructuring speech representations using a pitch-adaptive time-frequency smoothing and an instantaneous-frequency-based F0 extraction: Possible role of a repetitive structure in sounds," *Speech Communication*, vol. 27, no. 3–4, pp. 187–207, 1999.

[18] O. Babacan, T. Drugman, T. Raitio, D. Erro, and T. Dutoit, "Parametric representation for singing voice synthesis: A comparative evaluation," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, May 2014, pp. 2564–2568.

[19] D. Talkin, "A robust algorithm for pitch tracking (RAPT)," *Speech coding and synthesis*, vol. 495, p. 518, 1995.

[20] T. Saitou, M. Goto, M. Unoki, and M. Akagi, "Speech-to-singing synthesis: Converting speaking voices to singing voices by controlling acoustic features unique to singing voices," in *Applications of Signal Processing to Audio and Acoustics, 2007 IEEE Workshop on*, Oct 2007, pp. 215–218.

[21] J. Sundberg, "Articulatory interpretation of the singing formant," *J. Acoust. Soc. Am.*, vol. 55, pp. 838–844, 1974.

[22] HTS, "HMM-based speech synthesis system (HTS)," http://hts.sp.nitech.ac.jp/, 2013.