REALTIME SYSTEM FOR BACKING VOCAL HARMONIZATION

Adrian von dem Knesebeck, Sebastian Kraft and Udo Zölzer

Department of Signal Processing and Communications
Helmut Schmidt University
Hamburg, Germany
audio@hsu-hh.de

ABSTRACT

A system for the synthesis of backing vocals by pitch shifting of a lead vocal signal is presented. The harmonization of the backing vocals is based on the chords which are retrieved from an accompanying instrument. The system operates completely autonomous without the need to provide the key of the performed song. This simplifies the handling of the harmonization effect. The system is designed to have realtime capability to be used as live sound effect.

1. INTRODUCTION

The task to synthesize various voices from a solo singing voice has been realized in many different ways. Some approaches aim to synthesize a whole choir from a single singing voice. A system for the synthesis of natural sounding choir voices was presented in [1]. The singing voice is modified in pitch, time and timbre to synthesize a number of choir voices. Hence the choir voices are directly synthesized from the singing voice signal.

Another approach, presented in [2], extracts high level features from the singing voice and synthesizes the choir voices using a database of voices and timbres. The singing voice is morphed with the database voices to synthesize a choir which contains the features of the single voice, but also the smooth sound of a choir.

In this paper we present a system which synthesizes backing vocals. In contrast to the choir synthesis, where the aim is to synthesize a smooth and broad choir, we are more interested in synthesizing a number of distinguishable voices like in a typical band with one lead singer and one or more backing singers. Both of the referenced approaches implement systems which require additional information on how to harmonize the choir voices. We present a system which autonomously performs the harmonization task based on a harmony analysis of an accompanying instrument, e.g. a rhythm guitar or piano. An overview of the proposed system is given in Section 2. The signal analysis and feature extraction part, which includes the pitch detection and the chord detection, is described in Section 3. The synthesis of the backing vocals by modification of the singing voice and the harmonization is described in Section 4. Section 5 describes the evaluation of the system followed by a brief discussion of the performance. In Section 6 we conclude the paper.

2. SYSTEM DESCRIPTION

The presented system consists of three main blocks, i.e. the pitch detector, the chord detector and the voice synthesizer, as shown in Fig. 1. Two input signals are required for the processing. One input is the singing voice signal (*Voice*), which is fed to the pitch detector for pitch extraction and to the voice synthesizer. The other input is the accompanying instrument signal (*Instrument*), which is fed to the chord detector. The singing voice pitch and the chord information are fed to the voice synthesizer. The voice synthesizer processes the singing voice input signal in respect of rules for the harmonization. The particular blocks are described in more detail in the following sections.

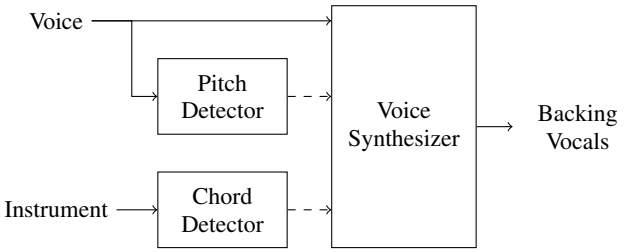


Figure 1: *System overview.*

3. SIGNAL ANALYSIS

The signal analysis section of the system consists of a pitch detector and a chord detector. The focus of this paper is on the chord detection particularly the multipitch detection required for the chord detector.

3.1. Pitch Detection

It is important to have robust information of the current lead vocal pitch to enable the succeeding blocks to work properly. We chose the YIN algorithm as presented in [3] for the pitch detection, because it returns robust and accurate results for monophonic harmonic signals [4].

3.2. Chord Detection

The task of the chord detector is to analyze the polyphonic instrument signal to extract multiple pitches and to derive from these pitches the corresponding chord symbol representation. The extraction of multiple pitches requires to reduce the rich harmonic signal produced by a musical instrument to the pitches. For most instruments the pitches can be expressed as the fundamental frequencies $F0$.

3.2.1. Multipitch detection

The multipitch detection algorithm is based on an approach proposed by Tolonen [5]. The Tolonen system is the auditory motivated system as shown in Fig. 2. The input signal is filtered by a whitening filter and split into two bands. The periodicity of the lower frequency band is calculated as main indicator of the fundamental frequencies. The signal in the higher frequency band is half-wave rectified and lowpass filtered. This models the mechanical to neural transduction of an inner hair cell according to Meddis [6]. The output of the model has an ac and a dc component. The ac component is of the same frequency as the input signal and the dc component is a monotonic saturating function of signal level, which provides the envelope of the signal. The periodicity of the higher band emphasizes the lower band's periodicity. The periodicities are calculated using the Fourier transform, which allows magnitude compression and speeds up the computation compared to the time domain autocorrelation. The sum of both paths builds up the sum autocorrelation function (SACF).

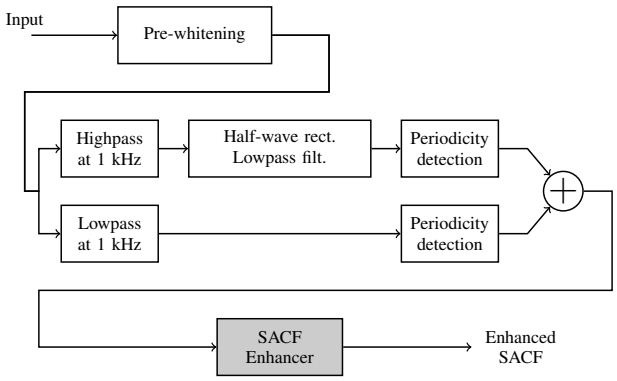


Figure 2: *Tolonen's multipitch detector [5].*

The Tolonen system continues with the SACF Enhancer which removes redundant and spurious information by stretching the SACF and subtracting the stretched SACF from the original one.

We replace this block and come up with a different approach to remove redundancy to be able to obtain the fundamental pitch candidates. Starting from the SACF we calculate a threshold as

$$th = \frac{\alpha}{L} \sum_{l=0}^{L-1} \max\left(\text{SACF}(l), 0\right), \tag{1}$$

which is the mean of the first $L = 700$ lag values l of the SACF, where the SACF is truncated to have only positive values. The constant α lowers the threshold by scaling the truncated mean. We used a factor of $\alpha = 0.4$. The SACF values below the threshold are omitted and the lags M of the peaks above the threshold are further considered. As the next step we group the peaks into harmonically related periods and calculate a rating value K for each group as

$$K = \sum_{i=0}^{4} 2^i \cdot \max(\text{SACF}(m_i)), \tag{2}$$

with

$$\left\lfloor 2^i M - (0.7 \cdot 2^i M)^{\frac{1}{3}} \right\rfloor \leq m_i \leq \left\lfloor 2^i M + (0.7 \cdot 2^i M)^{\frac{1}{3}} \right\rfloor. \tag{3}$$

m_i describes an observation range to find the actual peak location near $2^i M$ in the SACF, because the multiples must not be exact integer multiples. Starting from low lag values for each peak lag M above th the group of corresponding subharmonics is considered and the weighting factor is determined according to (2). The lag of the highest peak of the group with the highest value of K is regarded as most prominent pitch period candidate. The j-multiples of M are removed from the SACF with $j \in \{1, 2, 3, 4, 6, 8\}$. The pitch period determination is an iterative process with the number of iteration steps given by the maximum number of determinable fundamental pitches.

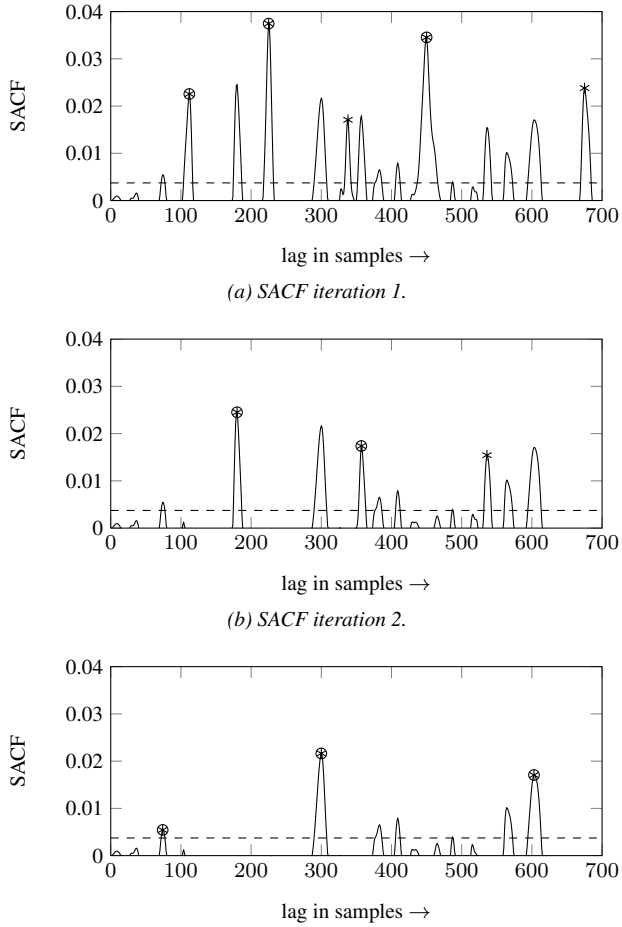
The schedule of the algorithm to distinguish the most prominent $F0$ is as follows:

1. calculate SACF of a 4096 samples time frame at $f_s = 44.1$ kHz
2. calculate threshold th
3. find maxima above th with maximum lags M
4. for each M search for multiples, i.e. subharmonics
5. sum the weighted values of the group of harmonically related maxima to get a rating of harmonicity
6. select the highest rating value as $F0$-candidate
7. eliminate the corresponding maxima of the $F0$-candidate from the SACF
8. continue iterating from step 3 with the remaining peaks
9. stop if desired number of $F0$-candidates is retrieved

Figure 3 shows an example of the removal process. In Fig. 3a the peaks of the SACF are shown. The group with the highest rating value K starts from $l = 112$. The peaks used for calculation of K are marked by circles. All corresponding lag values belonging to this group are marked by stars. The highest peak of the group is taken as the fundamental pitch candidate of this group, in this case the peak at $l = 225$. The marked peaks are removed as shown in Fig. 3b and the peaks of the second group, with lag values corresponding to $l = 178$, are marked. This procedure is repeated for the third group with lag values corresponding to $l = 74$ (see Fig. 3c). We end up with three lag values which represent the periods of the $F0$-candidates. Figure 4a shows the SACF over time of an example G major chord, played on a guitar. In Fig. 4b the results of the $F0$-candidate retrieval is shown. We see that this method is prone to octave errors, but mapping theses candidates to the corresponding tones of the 12-tone chroma vector shows that we obtain a robust pitch class representation of the chord. In Fig. 4c the discrete chromagram of the detected tones is shown.

3.2.2. Chord classification

The multipitch detection and pitch class determination, respectively, is the fundamental component of the chord detection. Once the chord tones are detected, as described in Section 3.2.1, a mapping from multiple tones to the corresponding chord representation is done. This mapping can be a quite complex task considering the amount of possible chords which could occur. The classification of chords can be done using statistical models, e.g. hidden Markov Models (HMM), as presented in [7, 8]. A realtime implementation of the HMM classifier is possible with a modified Viterbi [9] at a high computational cost adding some latency. We require a low complexity algorithm and therefore we applied a classification known as pattern or template matching [10, 11, 12, 13], which

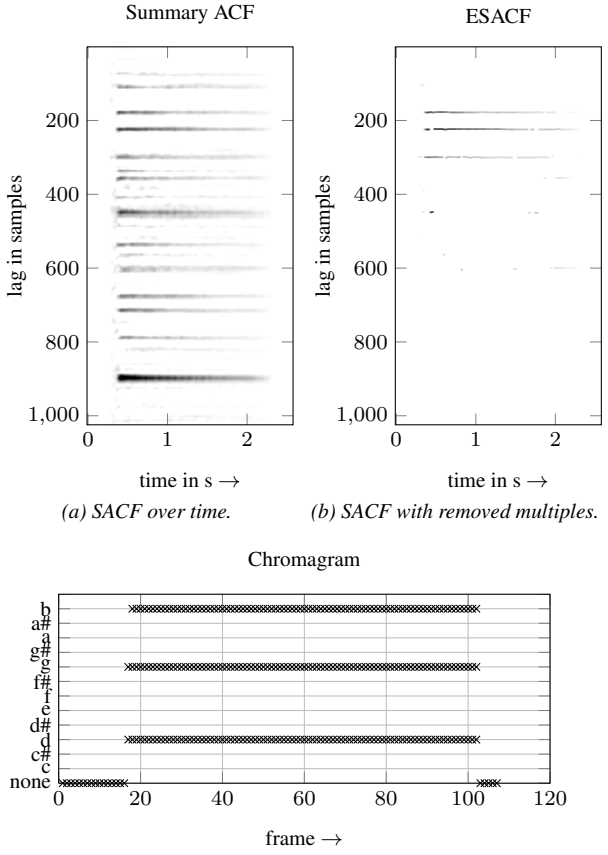


(a) SACF iteration 1.

(b) SACF iteration 2.

(c) SACF iteration 3.

Figure 3: *The three iteration steps of SACF peak removal are shown. The dashed line shows the peak detection threshold th, the stars indicate the peaks of the most prominent harmonics group and the circles indicate the peaks used for K calculation.*



(a) SACF over time.

(b) SACF with removed multiples.

(c) Chromagram of the detected $F0$-candidates.

Figure 4: *Results of the chromagram determination for G major guitar chord sample.*

is applicable in a frame-by-frame manner. The detected tones are represented by a bit mask which is a 12×1 vector having a 1 where a note is present and a 0 else. With the tones ordered as [c,c#,d,d#,e,f,f#,g,g#,a,a#,b], the G major chord of the example sample would be represented as [0,0,1,0,0,0,1,0,0,0,1]. The classification is done by calculating the hamming distance between the bit mask of the detected tones with the template bit masks of the possible chords. The bit mask with the minimum hamming distance is regarded as the chord candidate. We simplify the chord detection by restraining the possible chords to two triad modes, major and minor, with the common 12 key notes per octave. This leads to a total of 24 possible chords that can be detected.

3.2.3. Validation of proposed system

The comparison of the Tolonen system with the proposed modified system was done on recorded clean guitar samples. The test set included 14 samples of standard chords (major and minor, in different keys). Each sample presents one chord which was struck once and let ring to fade out. The compared quantities include the error of the detected chords (chord error), the amount of not detected tones (tone false negative) and the amount of wrongly detected tones (tone false positive). The errors were calculated as the mean frame error. For the tone detection we were not interested in the correct octave of the played note and therefore we did not consider octave errors as false detection. The robustness of both systems was increased in the same way by temporal smoothing of the detected tones and the detected chords.

error type	Tolonen	proposed system
chord error	30.8%	2.5%
tone false negative	5.8%	1.5%
tone false positive	87.5%	12.2%

Table 1: *Comparison of Tolonen approach to proposed system.*

The test results show that the investigated tone detection errors could be reduced with the proposed modified Tolonen system. Consequently the chord detection error was reduced as well.

4. VOICE SYNTHESIZER

Now that we have detected the pitch $F0$ of the singing voice and the chord of the accompanying instrument we can continue with

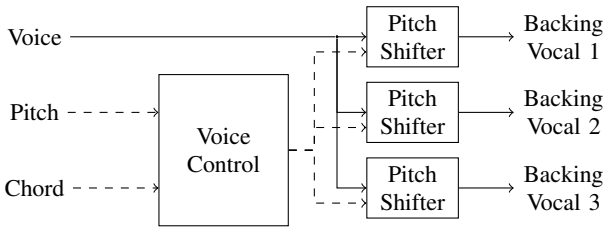


Figure 5: *Background vocal synthesizer.*

the synthesis of the backing vocals. The block diagram of the background vocal synthesizer is shown in Fig. 5. The synthesizer consists of a *Voice Control* block, which performs the control of the pitch shifting factors of the *Pitch Shifter* blocks. The Voice Control and the approach used for the pitch shifting are described in the following sections.

4.1. Pitch Shifter

The pitch shifters used are based on the Pitch Synchronous Overlap Add (PSOLA) pitch shifting algorithm [14, 15, 16, 17]. A block diagram of the PSOLA stages is shown in Figure 6. The al-

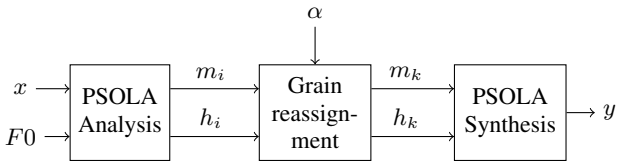


Figure 6: *Block diagram of PSOLA algorithm.*

gorithm segments the input signal x into short overlapping grains h_i with a length of twice the fundamental period time $1/F0$ in the analysis stage. The time instants marking the center of each grain are the analysis pitch marks m_i. Hence the time difference between two succeeding pitch marks represents on pitch period. The pitch is then shifted by a factor α in the grain reassignment stage by repositioning these grains, either reducing the distance in combination with occasional repetition of some grains to increase the pitch or expanding the distance in combination with omitting some grains to decrease the pitch. The pitch marks m_k indicate the time instants with the corresponding reassigned grains h_k for the overlap and add synthesis of the output signal y.

A robust pitch mark positioning algorithm which achieves high quality results is used as presented in [18]. The pitch mark positioning algorithm of the PSOLA analysis stage receives the fundamental pitch $F0$ from the pitch tracker described in Section 3.1. The pitch marks are positioned based on a center of energy approach, which ensures robust segmentation of the grains and reduces the occurrence of artifacts. The algorithm allows the positioning of the pitch marks in a frame-based manner to enable realtime application.

4.2. Voice Control and Harmonization

The task of the Voice Control block is to set the pitch shifting factors for the backing vocals synthesis in a way that a musically correct harmonization is achieved. We regard a harmonization as correct if the synthesized voices match the chord harmonies. At this stage of the work this means no dissonant voicings are desired.

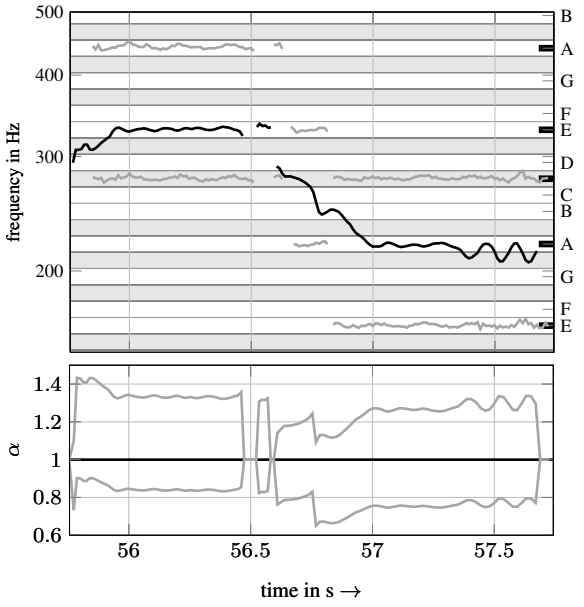


Figure 7: *Harmonization example for an A major chord. The upper plot shows the pitch contours of the recorded lead vocals (black curve) and a higher and lower synthesized backing vocal (gray curves). The lower plot shows the corresponding pitch shifting factors α.*

4.2.1. Harmonization

There are numerous approaches to harmonize additional voices to the lead vocals. In offline processing the voice leading for the synthesized voices can be manually provided as score transcription. This allows the user to individually adjust every note of each voice and hence offers the most creative possibilities.

An approach towards semi-autonomous harmonization is to provide the key of the song and to define the note interval of the synthesized voice related to the singing voice. This enables to synthesize the backing vocals with the correct pitch of the notes from the scale corresponding to the provided key. The major drawback of this approach is that it still requires to manually provide additional information about the musical content.

We achieve a completely autonomous harmonization without the requirement to supply information about the musical context. The musical information is determined using the chord detector which provides the instantaneous chord information. To make the valid note determination more robust the detected chords are observed over several frames. This ensures that short failures of the chord detection do not disrupt the harmonization of the backing voices. The harmonization starts by relating the singing voice to the current chord harmony. This can be regarded as a quantization of the singing voice pitch $F0_{\text{Lead}}$ to the chord notes. We consider an example of a singer accompanied by a guitar with an A major chord being the played harmony.The upper plot of Fig. 7 shows the lead vocal pitch contour of the example as black curve. On the right hand side the chord notes for the example A major chord

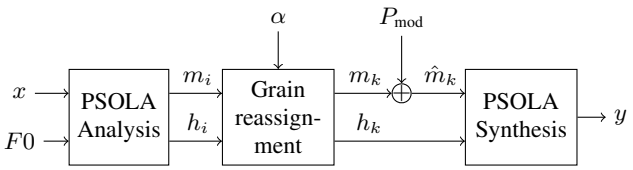


Figure 8: *Block diagram of PSOLA algorithm with modulation of the synthesis pitch marks.*

are marked with a black bar, i.e. A, C$^{\#}$ and E. The pitch shifting factors for the backing vocals are calculated as the ratio between the singing voice pitch and the intended backing vocal pitch as

$$\alpha_{\text{Back1}} = \frac{F0_{\text{Back1}}}{F0_{\text{Lead}}}, \tag{4}$$

$$\alpha_{\text{Back2}} = \frac{F0_{\text{Back2}}}{F0_{\text{Lead}}}. \tag{5}$$

The pitch shifting range is limited to synthesize voices which are close to the singing voice, because high pitch shifting factors result in disturbing artifacts. This leads to adding a backing voice which is pitched one chord tone higher than the lead vocal pitch and a backing voice one chord tone lower. The resulting pitch shifting factors are shown in the lower plot of Fig. 7. The corresponding pitch contours of the resulting backing vocals are shown by the gray curves in the upper plot of Fig. 7.

4.2.2. Humanization

To reduce artifacts and achieve a natural sounding synthesis the pitches of the backing vocals are slightly modulated. The PSOLA algorithm allows to efficiently realize a pitch modulation for vibrato simulation and a temporal modulation for varying delay simulation using the same modulation function but with different sets of parameters. This can be accomplished by modulating the synthesis pitch mark positions as shown in Fig. 8.

The pitch mark modulation function P_{mod} is given as

$$P_{\text{mod}}(t) = \frac{A_{\text{mod}}}{2} \cdot (1 + \sin(2\pi f_{\text{mod}} \cdot t)). \tag{6}$$

A relatively high modulation frequency f_{mod} of 5-10 Hz in conjunction with a relatively low modulation depth A_{mod} in ms results in a perceivable pitch modulation. In contrast a relatively low modulation frequency of 1 Hz or lower in conjunction with a higher modulation depth results in a varying delay. The effect of the later case is shown in Fig. 9. The example shows the synthesis pitch mark modulation for $A_{\text{mod}} = 45$ ms and $f_{\text{mod}} = 1$ s. The upper plot shows the synthesis pitch marks m_k and the resulting modulated synthesis pitch marks $\hat{m}_k$. The lower plot shows the modulation function $P_{\text{mod}}(t)$.

5. EVALUATION AND DISCUSSION

The described harmonization system is intended to support a lead singer of a small group or a solo artist with backing vocals. The current system is applicable for the typical singer songwriter kind of musical style. The development of the particular system modules was done in Matlab. The Matlab implementation of the algorithms was already done in a realtime manner. This allowed to port the algorithms to C++ functions to use them as VST plugins without much effort.

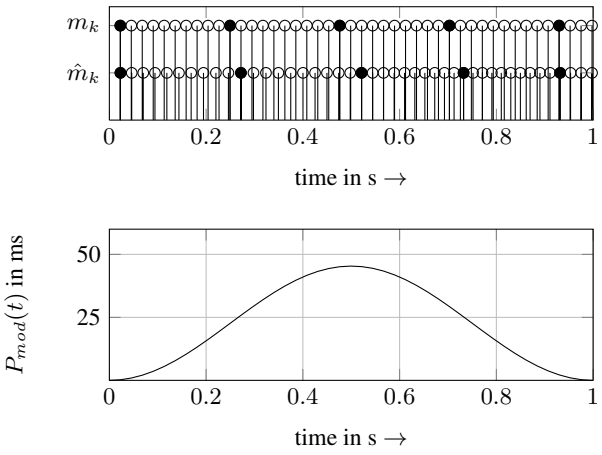


Figure 9: *Example of varying delay simulation by PSOLA synthesis pitch mark modulation.*

For evaluation we used a test song with a male singer accompanied by an acoustic guitar. The tracks were available as separate lead vocal and guitar signals. An objective assessment of such system's performance is hard to realize. Therefore a subjective assessment of the harmonization results was conducted by a group of musicians having experience in that style of music. The synthesized backing vocals were found to be in good accordance with the guitar harmonies.

There are some artifacts perceivable as glitches which are mainly caused by wrong lead vocal pitch detection. These artifacts could be further reduced by smoothing of the pitch transitions. The hard assignment of the lead vocal pitches to the chord harmonies may also lead to glitches. An algorithm which allows a soft decision region could resolve this problem.

Some audio clips of the harmonization results can be found at `http://ant.hsu-hh.de/dafx2011/harmonization`.

6. CONCLUSIONS

We presented a system which harmonizes backing vocals based on the detected chords of an accompanying instrument. We proposed a modification of the Tolonen multipitch detector. The results show that the accuracy of multiple pitch detection and consequently of the chord classification for recorded guitar samples could be increased. The harmonization is operating completely autonomously, which means no key has to be manually provided. The developed algorithms operate in realtime which allows the use of the harmonization as live effect. The achieved harmonization results are quite promising but there is room for further improvement.

Future work will concentrate on the autonomous harmonization, since the presented voice leading approach is rather simple compared to how a real musician could harmonize. Also the chord detection will be extended to be able to detect seventh chords to improve harmonization capabilities.

7. REFERENCES

[1] N. Schnell, G. Peeters, S. Lemouton, and X. Rodet, "Synthesizing a choir in real-time using Pitch Synchronous Overlap Add (PSOLA)," in *Proc. IEEE 1st Benelux Workshop on Model based Processing and Coding of Audio*, Leuven, 2002.

[2] J. Bonada, M. Blaauw, A. Loscos, and K. Hideki, "Unisong: A choir singing synthesizer," in *Proc. 121th Audio Eng. Soc. Convention*, San Francisco, 2006.

[3] A. de Cheveigné and H. Kawahara, "YIN, a fundamental frequency estimator for speech and music," *J. Acoust. Soc. Am.*, vol. 111, no. 4, pp. 1917–1930, 2002.

[4] A. von dem Knesebeck and U. Zölzer, "Comparison of pitch trackers for real-time guitar effects," in *Proc. 13th Int. Conf. on Digital Audio Effects (DAFx)*, Graz, 2010, pp. 266–269.

[5] T. Tolonen and M. Karjalainen, "A computationally efficient multipitch analysis model," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 6, pp. 708–716, 2000.

[6] R. Meddis and L. O'Mard, "A unitary model of pitch perception," *J. Acoust. Soc. Am.*, vol. 102, pp. 1811–1820, 1997.

[7] K. Lee and M. Slaney, "Acoustic chord transcription and key extraction from audio using key-dependent HMMs trained on synthesized audio," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 2, pp. 291–301, 2008.

[8] H. Papadopoulos and G. Peeters, "Simultaneous estimation of chord progression and downbeats from an audio file," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Las Vegas, NV, 2008, pp. 121–124.

[9] T. Cho and J. P. Bello, "Real-time implementation of HMM-based chord estimation in musical audio," in *Proc. Int. Computer Music Conference (ICMC 2009)*, Montreal, Canada, 2009.

[10] T. Fujishima, "Realtime chord recognition of musical sound: a system using common lisp music," in *Proc. Int. Computer Music Conference (ICMC 1999)*, Beijing, China, 1999, pp. 464–467.

[11] M. Cremer and C. Derboven, "A system for harmonic analysis of polyphonic music," in *Proc. 25th Int. Audio Eng. Soc. Conf.*, London, 2004.

[12] C. Harte and M. Sandler, "Automatic chord identifcation using a quantised chromagram," in *Proc. 118th Audio Eng. Soc. Convention*, Barcelona, 2005.

[13] A. M. Stark and M. D. Plumbley, "Real-time chord recognition for live performance," in *Proc. 2009 Int. Computer Music Conf. (ICMC 2009)*, Montreal, Canada, 2009.

[14] C. Hamon, E. Mouline, and F. Charpentier, "A diphone synthesis system based on time-domain prosodic modifications of speech," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, Glasgow, 1989, pp. 238–241.

[15] K. Lent, "An efficient method for pitch shifting digitally sampled sounds," *Computer Music Journal*, vol. 13, pp. 65–71, 1989.

[16] E. Moulines and F. Charpentier, "Pitch synchronous waveform processing techniques for text-to speech synthesis using diphones," *Speech Communication*, vol. 9, no. 5/6, pp. 453–467, 1990.

[17] R. Bristow-Johnson, "A detailed analysis of a time-domain formant-corrected pitch-shifting algorithm," *J. Audio Eng. Soc.*, vol. 43, no. 5, pp. 340–352, 1995.

[18] A. von dem Knesebeck, P. Ziraksaz, and U. Zölzer, "High quality time-domain pitch shifting using PSOLA and transient preservation," in *Proc. 129th Audio Eng. Soc. Convention*, San Francisco, 2010.