POLYPHONIC MUSIC ANALYSIS BY SIGNAL PROCESSING AND SUPPORT VECTOR MACHINES

Ruohua Zhou, Giorgio Zoia

Signal Processing Institute, LTS-3
Ecole Polytechnique Fédérale de Lausanne
{ruohua.zhou, giorgio.zoia}@epfl.ch

ABSTRACT

In this paper an original system for the analysis of harmony and polyphonic music is introduced. The system is based on signal processing and machine learning. A new multi-resolution, fast analysis method is conceived to extract time-frequency energy spectrum at the signal processing stage, while support vector machine is used as machine learning technology.
Aiming at the analysis of rather general audio content, experiments are made on a huge set of recorded samples, using 19 music instruments combined together or alone, with different polyphony. Experimental results show that fundamental frequencies are detected with a remarkable success ratio and that the method can provide excellent results in general cases.

1. INTRODUCTION

Analysis of real music content, not necessarily available in symbolic form, still remains a very challenging problem in spite of decades of very interesting research in the domain. One of the most promising analysis directions consists of the combination of consolidated signal processing techniques with intelligent agents, to improve the often ambiguous results of time-frequency analysis with smart decision systems trained by a consistent preliminary knowledge. The expected, introduced approximation in the results is acceptable as far as the performance is able to meet a quality of service that can be considered useful by users.
We are interested in providing fast (real-time) music analysis on audio content possibly without any kind of symbolic or metadata information being preliminary available, i.e. when the musical content is presented to the analysis tool in a pure digital sample format. In these cases, which better correspond to real world applications, extremely high precision and confidence in the results is very difficult to obtain but it will be shown that achieved results may be indeed interesting and useful for several practical purposes. In particular our research aims at real-time human machine collaboration in the musical domain and especially at content identification and classification for automatic processing control in the multimedia domain; integration with other forms of analysis (rhythmic pattern, instrument etc.) for mutual consolidation is envisaged.
This paper is organized as follows: the second section shortly presents the state of the art in the domain of our research; after a system overview the fourth and fifth sections introduce and explain the proposed time-frequency algorithm for music analysis, followed by a description of the approach used to exploit this method with support vector machines; the sixth section presents experimental results whereas the last sections concludes the paper with a short description of the project on which we are working and some final remarks about current new directions.

2. RELATED WORK

The main goal in polyphonic music analysis is to translate recorded polyphonic music into some meaningful symbolic representation such as note onsets, note durations, pitches, etc. Though polyphonic music transcription is still a very challenging task, some progress has been recently shown. Marolt use a neural network to extract polyphonic piano scores exploiting the auditory model and adaptive oscillator networks [1]. Goto extracts the melody line and baseline by estimation of the predominant harmonic in the different frequency regions [2]. Bruno et al. develop a system to support polyphonic transcription for different instruments based on audio models, pitch tracking and a neural network bank [3]; Bello constructs a blackboard system using both high level knowledge and data from bottom-up processing [4]. Based on spectrum smoothness and harmonicity, Klapuri uses the predominant frequency estimation and recursively removes the corresponding sound from polyphonic note mixtures for multiple frequency estimations, with excellent results [5]. The system in [6] makes chord analysis by a speech recognition tool. Monti and Sandler develop a blackboard system for piano polyphonic note recognition by fuzzy inference [7]. Cemgil designs a polyphonic transcription system based on Bayesian inference and probability models [8].

3. SYSTEM OVERVIEW

The proposed music analysis and transcription system is mainly composed by a signal processing block followed by a learning agent. Compared to the human listening system, the signal processing stage is a time-frequency signal analysis tool similar to cochlear filters, whereas the learning machine plays a role similar to the one of the human brain. A filter bank energy spectrum analyzer is used as signal processing component, and support vector machines (SVM) as intelligent agent. Figure 1 shows the main system block diagram. First, the music signal is processed by a complex resonator filter bank, and the energy spectrum is calculated in function of time and frequency. This spectrum is then smoothed by a low pass filter bank, and finally the peaks are picked from the smoothed energy spectrum to produce the input vector to the SVM. In the training phase, the extracted peaks are used with the target output to produce the SVM note recognizer;

in the application phase, the SVM note recognizer uses the peaks vector to perform multi pitch tracking. The SVM note recognizer consists of 88 two-classes classifiers for 88 notes (piano extension); each two-classes SVM classifier recognizes if an input sample includes the corresponding note or not.

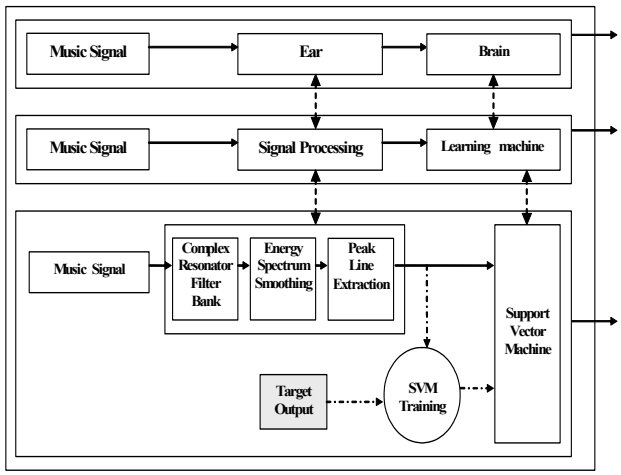

Figure 1: *The overall architecture of the proposed system*

4. TIME-FREQUENCY SIGNAL ANALYSIS

The first stage in the proposed polyphonic transcription system is a time-frequency energy spectrum analyzer. Music signal is time-varying within a wide frequency range, and the time-frequency analysis must be multi-resolution for optimal results. One commonly used time-frequency energy spectrum analysis method is a spectrogram based on Short Time Fourier Transform (STFT), which can be described as follows:

$$E(t,\omega) = \left| \int_{-\infty}^{\infty} s(\tau) w(\tau - t) e^{-j\omega\tau} d\tau \right|^2 \qquad (1)$$

In equation (1), *w* is the window function, which determines the time and frequency resolution of the STFT. An STFT-based energy spectrum can be obtained with a good computation-efficient FFT, but it pays a high price for such a fixed-length transform in musical terms. The window function is independent from the frequency ω, so the spectrum has the same time and frequency resolution for all frequency bands; this makes STFT-based analysis critical when different time and frequency resolutions are required, like in music.

A more general time-frequency energy spectrum analysis of time-varying signals is proposed as follows:

$$GE\ (t,\omega) = \left| \int_{-\infty}^{\infty} s(\tau) w(\tau - t, \omega) e^{-j\omega(\tau - t)} d\tau \right|^2 \qquad (2)$$

The window function *w* in equation (2) depends on the frequency ω, this means that time and frequency resolutions can be changed according to frequency. At the same time, equation (2) can also be expressed like:

$$GE\ (t,\omega) = \left| \int_{-\infty}^{\infty} s(\tau) I(t - \tau, \omega) d\tau \right|^2 = \left| s(t) * I(t,\omega) \right|^2 \qquad (3)$$

with

$$I(t,\omega) = w(-t,\omega) e^{j\omega t} \qquad (4)$$

Equations (1) and (2) are more suitable to express a transform-based implementation whereas equation (3) is more straightforward to implement a filter bank with impulse response functions expressed by equation (4).

In order to implement such an energy spectrum analyzer, an IIR filter bank is a reasonable choice; the order of the filter bank needs to be as low as possible to reduce the computation cost. In our polyphonic analysis system, a first-order complex resonator filter bank has been exploited.

The first-order complex resonator impulse response can be described as:

$$I(t) = e^{(-r + j\omega)t} \qquad (5)$$

The decay factor in (5) determines the exponent window length and the time resolution; at the same time it determines the frequency bandwidth (i.e. the frequency resolution).

Equation (3) can be further modified like:

$$GE(n,\omega_m) = \left| s(n) * h_m^E(n) \right|^2 * h_m^L(n) \qquad (6)$$

where $h_m^E(n)$ and $h_m^L(n)$ are the impulse responses of a first order complex resonator filter and real low pass filter bank; their transfer functions are defined as follows:

$$E_m(z) = \frac{1 - e^{-r_m}}{1 - e^{(-r_m + j\omega_m)} z^{-1}} \qquad (7)$$

$$L_m(z) = \frac{1 - e^{-r_m}}{1 - e^{(-r_m)} z^{-1}} \qquad (8)$$

with the exponent decay factor of the impulse response

$$r_m = map(\omega_m) \qquad (9)$$

Equation (6) shows how the energy spectrum is implemented by a first order complex resonator filter and real filter bank. The second convolution denotes the low pass filtering, which is used to smooth the energy spectrum.

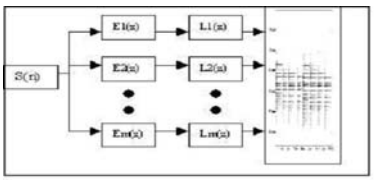

Figure 2: *The architecture of the time-frequency energy spectrum analyzer*

In Figure 2 the output of the complex filter is energy, which then passes through the low pass filter *L* to be smoothed. For a complex resonator, when the input only includes the frequencies around the resonator oscillation frequency, the magnitude of filter output is almost stable; otherwise the filter output shows some beats of input frequencies and oscillation frequency, and its absolute value oscillates. In order to maintain the time-frequency resolution, the low pass filter has the same impulse response exponential decay factor as the complex filter of the corresponding frequency channel.

The time-frequency resolution distribution can be set efficiently and flexibly through the *map* function between the frequency and

the exponential decay factor of the filter impulse response (equation (9)). When implementing the filter bank, it is first needed to define the center frequency ω of the complex resonator filter bank: in our system, ω for the m^{th} filter is defined as follows:

$$\omega_m = \frac{2\pi \cdot 440 \cdot 2^{\frac{StartNoteNum+m/r-69}{12}}}{f_s} \quad (9)$$

In equation (10), *StartNoteNum* is a parameter indicating the lowest note number (MIDI note numbers are used) in the considered range, and r is a *resolution* parameter used to denote how many filters are used to cover the frequency band of one semitone. This definition is convenient to map our multi-pitch tracking to western music notation. The *map* function can then be defined as like: $r_m=k\omega_m$. The parameter k is a constant for all the frequencies; this makes the ratio between the resonator filter bank bandwidth and center frequency a constant, so it is a constant-Q filter bank. The center frequency of the filter in the filter bank changes here according to an exponential law (10).

For practical applications, a fast implementation has been developed for the complex resonator filter bank. The basic idea is to reduce the redundancy in computation: it is not necessary to keep the same sampling frequency of the input for every filter in the filter bank. For the filters with lower center frequencies, the sampling rate can be decreased. At the same time, because the main frequencies of music notes vary according to the exponential law, the high frequency regions need only a lower frequency resolution. This means that a shorter duration signal frame is enough for the frequency analysis. To combine multi-resolution analysis and computation efficiency, other multi-rate filter banks have been proposed in music analysis [16,17,18].
In [17] and [18] the multi-rate filter bank is used to separate the signal into several octave-spaced subbands and then a sinusoids analysis is performed in every subband. In [16], similarly to [17] and [18], signal is first separated into several octave subbands by multi-rate filter bank and the following detailed frequency analysis is performed by FFT. On one hand, our multi-rate complex resonator filter bank uses a similar approach to first separate the signal into the several octave-spaced subbands; on the other hand differently from other cases, it still keeps the constant-Q frequency resolution for the detailed frequency analysis in every subband, whereas for example in [16] the FFT used in this phase has an equally spaced frequency resolution.

The proposed algorithm is especially conceived for multi pitch tracking based on short signal frames, which in a large majority of frames corresponds to a monotone or polyphonic stationary situation. In the spectrum extraction algorithm, first the signal is separated into 8 octave-spaced frequency bands by the dyadic sampling mulitrate filter bank according to the architecture shown in Figure 3. As mentioned before, similar ways have been used in [16,17,18]. At the same time, since the required frequency resolution at higher frequencies is lower, shorter time frames are used to compute the energy spectrum in order to reduce the computation cost further. Detailed information is shown in Table 1.
As shown in the table, the downsampling begins from the sixth frequency band; this is a reasonable choice to make the ratio between sampling rate and analysis frequency about 20 or more, according to an experimental rule of thumb. Finally the spectrum peaks are independently extracted in different frequency bands; a small overlap between neighbor frequency bands is used to find the spectrum peaks for the frequency bin at the edge of the frequency band. . Every frequency band includes 60 frequency bins (the overlapping frequency bins are not considered here). The introduced filter bank has several advantages. First the energy spectrum is extracted according to a logarithmic law, so the frequency resolution at low frequencies is high and up to about 1 Hz. Second, the fast implementation makes it practical for applications. Third, it is flexible and different time frames in different frequency bands can be chosen: this is useful in many cases.

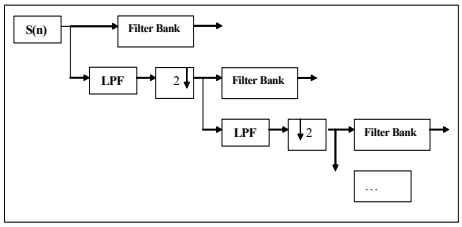

Figure 3: *Block diagram for the proposed filterbank implementation*

Table 1: *values used to reduce complexity in the analysis filter bank*

Band Number	Sampling Rate (Hz)	Frequency Range (Hz)	Used samples	Duration time (Sec)
1	689.06	25.96--55.00	680	0.9861
2	1378.12	51.91--110.0	680	0.4934
3	2756.25	103.8--220.0	1360	0.4934
4	5512.5	207.7--440.0	1360	0.2467
5	11025	415.3--880.0	2720	0.2467
6	22050	830.6--1760	2720	0.1234
7	44100	1661--3520	2720	0.0617
8	44100	3322--7040	2720	0.0617

As a final remark, wavelets can also provide another constant-Q filter bank solution, often preferred for audio compression and music synthesis because of the orthonormal or biorthornormal characteristics; but it has more rarely been used for music analysis because the commonly used dyadic sampling fast Discrete Wavelet Transform (DWT) only provides a very coarse frequency resolution, which is far from the requirements of music analysis. On the other hand the orthonormalization or biorthonormalization are not necessary for music analysis, and moreover a wavelet solution is implemented by FIR needing much more computation resources than IIR.

5. MACHINE LEARNING: SUPPORT VECTOR MACHINE CLASSIFICATION

As remembered, tracking multiple pitches in polyphonic music is still a very challenging task, especially because some harmonics of different music notes may overlap together. For such a problem, it is reasonable to consider a learning machine as classifier as this corresponds to an ear-brain scheme for trained humans (see again Fig. 1).

5.1. Support Vector Machine

Only a simple introduction to support vector machine (SVM) classification is given here. For more detail, please refer to some authoritative literature [9,10,11]. Two are the key concepts behind SVM classifiers: the first is to find an optimal hyperplane for linearly separable classification problem based on the structure risk minimization (SRM) inductive principle. The second is to nonlinearly map input data from an input space to a high-dimension feature space, where a nonlinear classification problem in the input space can become linearly separable.

To better explain SRM, we can start from the following machine learning problem: in the case of m samples with n-dimension input vector $\boldsymbol{x}$ and known output class label vector $\boldsymbol{y}$,

$$(x_1, y_1)...(x_m, y_{m1}) \quad x_m \in \Re^n \quad y_m \in \{-1,+1\} \qquad (11)$$

it is requested to find a learning machine f, with a decision function $y=f(\alpha,x)$ to classify the new samples. The learning process aims at adjusting the parameter α for minimizing the expected real risk $R(\alpha)$ on the new samples. If a function L is selected as loss function, $P(x,y)$ is the unknown probability distribution, and assumed the sample data are i.i.d. (identically drawn and identically distributed) it is possible to write ([9]):

$$R(\alpha) = \int L(y, f(\alpha, x))dP(x, y) \qquad (12)$$

Since $P(x,y)$ is unknown, $R(\alpha)$ can not be directly computed. Based on the empirical risk minimization (ERM) inductive principle, usually the empirical risk $R_{emp}(\alpha)$ is used to train the learning machine instead of $R(\alpha)$. For example if the least-squares method is selected as loss function, the empirical risk is ([9]):

$$R_{emp}(\alpha) = \frac{1}{m}\sum_{i=1}^{m}(y_i - f(\alpha, x_i))^2 \qquad (13)$$

The learning machine trained on ERM may show a very small training error on the training set but bad performance on the new samples and cause the so-called overfitting problem; in other words the ability of generalization of the machine is not good. To control the generalization ability of machine learning, Vapnik provides the structure risk minimization (SRM) inductive principle ([9]), on which a machine is trained in terms of both empirical risk and the generalization ability. Based on SRM, a linear SVM selects the maximum margin hyperplane classifier, which is considered to have the best generalization ability. The margin is defined as the distance between the hyperplane and the closest sample vector. Without entering too much into detail, it can be shown that the maximum margin hyperplane classifier for a linearly separable problem can be obtained by solving a quadratic optimization problem. To make a nonlinear classification, the input data are nonlinearly mapped from the input space to the high-dimension feature space, where a nonlinear classification problem in the input space can become linearly separable.

On one hand SVM transforms a nonlinear classification problem into a linear classification problem in a high-dimension feature space, on the other hand it does not need to operate in the feature space and all the necessary computation is directly performed in the input space by using the kernel function. Three commonly used kernel functions are radial basis function (RBF) kernel, sigmoid kernel and polynomial kernel ([11]).

Recently, SVMs have also been explored to resolve some music related issues such as music genre classification [19].

5.2. Support Vector Machine in polyphonic music

To define a learning system, it is first needed to specify the input vector and the type of machine. Spectrum scaled to a logarithmic axis was selected as the input vector; this is a reasonable choice for common western music. In fact, the fundamental frequency and corresponding partials of a music note can be described as

$$f_k^0 = 440 \cdot (2^{\frac{k-69}{12}}) \quad \text{and} \quad f_k^m = m \cdot f_k^0,\ k \geq 1 \qquad (14)$$

using again MIDI note numbers for note k. Supposing that the energy of every music note mainly distributes over the first 10 harmonics, and $Energy(f_k^m) \approx 0$ for $m \geq 11$, the frequency ratio between one note partials and the fundamental frequency of other music notes is as follows:

$$2f_k^0 = f_{k+12}^0, \quad 3f_k^0 / f_{k+19}^0 = 0.9989, \quad 4f_k^0 = f_{k+24}^0$$

$$5f_k^0 / f_{k+28}^0 = 1.0079, \quad 6f_k^0 / f_{k+31}^0 = 0.9989, \quad 7f_k^0 / f_{k+34}^0 = 1.018,$$

$$8f_k^0 = f_{k+32}^0, \quad 9f_k^0 / f_{k+38}^0 = 0.9977, \quad 10f_k^0 / f_{k+40}^0 = 1.0079$$

This means the first 10 partials always completely or nearly overlap with another fundamental frequency; as the fundamental frequencies follow an exponential law (14), so most of the energy is concentrated in frequency bins that are exponentially spaced and then equally spaced according to a logarithmic axis.

The note recognizer consists of 88 classifiers ranging from A0 to G8 (piano extension); every two-classes classifier recognizes if an input sample includes the corresponding note or not. For what concerns the machine type, neural networks have been tested in past experiences [12], but finally SVM has been chosen as it shows several important advantages. First, SVM is based on SRM and provides the theoretical support and related tools for controlling the ability of generalization, whereas a neural network design often depends on heuristics and easily leads to an overfitting problem. Secondly, SVM can achieve a global solution while the neural network can only converge to a local solution. Finally in our case, the input vector is composed by the extracted spectrum peaks that only exist in some frequency bins, so the input vector is sparse and high-dimension; SVM can process such sparse input vector very efficiently. In our experiments the input vector size is sometimes up to 960 and SVM can still work well; this property is very useful. On the contrary, neural networks with hidden layers training are very time consuming for high-dimension input vectors.

In practice, the LIBSVM [13] software has been used as major tool. To train a SVM classifier, it is first needed to select a kernel function; the library provides linear kernel, RBF kernel, sigmoid kernel and polynomial kernel. Indeed the linear kernel is a special case of RBF kernel [14], sigmoid kernel is not valid under some conditions [15], and polynomial kernel classifier needs too long training times for such a high-dimension classification problem; RBF has then been chosen as kernel function.

When using the RBF kernel, two parameters C and γ specify the function: C is the penalty parameter of the error term, and γ is the RBF kernel parameter ([14]); an optimal (c,γ) is needed to make the classifier perform well on unknown new samples. This is achieved by grid-search using the crossing-validation. We try several (c,γ) pairs and pick the pair, by which the trained classifier has the best crossing-validation accuracy. Hundreds of classifiers have to be trained and every classifier has more than 100,000 training samples (see next section for more details), so it

is impossible to run an extensive grid-search to find (c,γ). A subset is selected from the training samples, and this subset is used to find the good (c,γ) by grid-search and crossing-validation. Then the parameter pair (c,γ) is selected that will be used to train the classifier on the complete training set. Finally the trained classifier has to be tested. Experimental results are presented in the next section.

6. EXPERIMENTS AND RESULTS

In real cases, polyphonic music to analyze may be the combination of the sounds from several different instruments, as well as from a single instrument. We decided to design two kinds of multi-pitch analyzer, a general-purpose one used to analyze multitimbre polyphonic music, and another one assuming the polyphonic music limited to only a single defined instrument, such as piano or guitar. Two main experiments have been defined for the two kinds of recognizer.

A good validation requires thousands and thousands of polyphonic samples, which were produced by selected combinations of the recorded monophonic samples from RWC Music Instrument Sound Database. In total 1027 monophonic samples from 19 music instrument have been used, and every monophonic sample was pre-processed to normalize amplitude and faded to a one second duration time. Every polyphonic sample first needs to be processed by the fast filter bank to produce 960 bins-wide spectrum as the input vector to SVM. Three measures, note error rate (NER), chord error rate (CER) and note classification rate (NCR) have been used to evaluate the transcription performance. The NER is the ratio between the number of errors in recognized fundamental frequencies and total number of fundamental frequencies (i.e. notes) in the correct transcription. CER is the percentage of sound chords where one or more pitch identification errors occurred [5]. The recognizer consists of tens of classifiers for the different music notes; NCR is the rate of correct classifications in the single two-class note classifier.

In the first experiment, 88 two-class classifiers have been independently trained and tested for notes from A0 (MIDI note number 21) to C8 (MIDI note number 108) for the multi-timbre analyzer. The polyphonic samples in training and test sets are random instrument and pitch combinations. For instance to produce a two-note polyphonic sample, first two different notes between A0 and C8 are randomly chosen, the two selected notes being e.g. C2 and C3; among the 19 instruments, there are eight instruments including C2 in their range and twelve including C3. At this point a note C2 from one of the eight instruments and a note C3 from one the twelve instruments are randomly selected and added together to produce a two-note polyphonic sample. A total of 150,000 training samples and 200,000 test samples (with no overlap between the two sets) were produced with polyphonies from two to six notes, with a large percentage of musically meaningful combinations.

Table 2: *Test result of general-purpose recognizer*

	Note Error Rate (NER)	Chord Error Rate (CER)
2-note polyphonic samples	2.9%	
3-note polyphonic samples	5.8%	17%
4-note polyphonic samples	9.8%	31%
5-note polyphonic samples	14%	49%
6-note polyphonic samples	20%	

The NER and CER of the general-purpose recognizer are reported in Table 2.

The five curves in the top part of Figure 4 show instead the NCR for the 88 notes classifier under test from two- to six-notes polyphonic samples; the horizontal axis denotes the corresponding note according to MIDI note numbers. The bottom part of Figure 4 shows the number of instruments (in the performed experiments) containing a certain note in their extension; the horizontal axis denotes again the corresponding note according to MIDI note numbers. It can be noticed from the figure that when a note has more music instruments able to play it, the corresponding music note classifier has lower NCR.

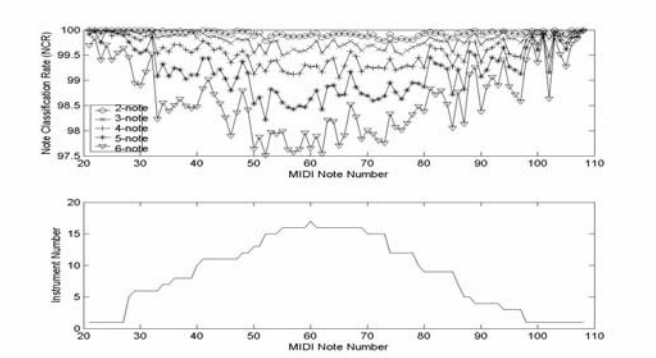

Figure 4: *Note Classification Rate and Instrument Number in Every music note*

When music instruments of the analyzed polyphonic music sequence are known in advance, it may be better to run the multi-pitch analysis by a specialized recognizer, which is trained by the samples of the corresponding instruments only. For example, it may happen (and it is rather intuitive) to obtain a better performance by making the multi-pitch analysis of piano music by a recognizer trained only by piano samples.

In the second experiment, a single instrument multi-pitch tracker was trained for piano, guitar and violin. The preparation of training and test data is almost same as in the first experiment. For each of the three instruments, samples from three different instrument producers are available. The setup information of the second experiment is presented in the following Table 3.

Table 3: *Setup information of the second experiment*

	Piano	Violin	Acoustic Guitar
Training Polyph. Samples	90000	60000	60000
Test Polyphonic Samples	80000	50000	50000
Note Classifiers	88	38	34
Instrument Samples	324	222	225

Tables 4 and 5 show the NER and CER for the 3 single instrument recognizers.

Table 4: *Note Error Rate (NER) of the 3 single instrument recognizers.*

	Piano	Violin	Acoustical Guitar
3-note polyphonic samples	3.7%	3.0%	2.7%
4-note polyphonic samples	6.8%	4.1 %	4.8%
5-note polyphonic samples	10%	6.6 %	8.6 %

Table 5: *Chord Error Rate (CER) of the 3 single instrument recognizers*

	Piano	Violin	Acoustical Guitar
3-note polyphonic samples	12%	5.0%	3.0%
4-note polyphonic samples	21%	13 %	10%
5-note polyphonic samples	32%	37 %	30%

Table 6: *Comparison with the state of art*

	NER (our result)	NER ([5])
2-note polyphonic samples	2.9%	3.9%
3-note polyphonic samples	5.8%	6.3%
4-note polyphonic samples	9.8%	9.9%
5-note polyphonic samples	14%	14%
6-note polyphonic samples	20%	18%

Compared with the state of art ([5]) the general-purpose recognizer has similar results, as shown in Table 6. The single-instrument recognizer shows an even better performance than general purpose recognizer (experiment 2 above).

7. STILE

The music analysis system presented above is part of a wider r&d Swiss project (partially funded by CTI 6893.2) named STILE (Sound for human-Tuned Interactive Living Environments). STILE aims at developing a novel class of sound and music rendering devices that are conceived for a seamless interaction with human beings in their living and working environments. The overall system will provide a transparent interface between conventional or specific sonic/musical content and listeners to highly improve their experience by enhanced listening conditions. The core technology of STILE is based on music oriented sound analysis (partially presented in this paper), perceptually relevant 3D signal processing, and non-invasive diffusion devices and control sensors.

8. CONCLUSION

In this paper we presented a new system for harmonic and polyphonic music analysis. The system is based on signal processing and machine learning. An efficient multi-resolution, fast time-frequency analysis method is introduced to extract energy spectrum in the signal processing stage; support vector machines are used as "intelligent" technology. Results show a state-of-the-art performance on a wider and unrestricted test scenario.
Future work includes a set of extensive tests on real music cases; at the same time the technology will be integrated with an already existing timbre (instrument) analysis tool to consolidate its results by more precise identification of event starts.

9. REFERENCES

[1] M. Marolt "A Connectionist Approach to Automatic Transcription of Polyphonic Piano Music," *IEEE Transactions on Multimedia,* vol. 6, no. 3, June 2004.

[2] M. Goto"A Predominant-F0 Estimation Method for Polyphonic Musical Audio Signals," *Proceedings of the 18th International Congress on Acoustics (ICA 2004),* pp. II-1085-1088, April 2004.

[3] I. Bruno, S. L. Monni, P. Nesi "Automatic Music Transcription Supporting Different Instruments," *Proceedings of the Third International Conference WEB Delivering of Music (WEDELMUSIC'03).*

[4] J.P. Bello "Towards the Automated Analysis of Simple Polyphonic Music: A Knowledge-based Approach," PhD thesis, Queen Mary, University of London.

[5] A. P. Klapuri "Multiple Fundamental Frequency Estimation Based on Harmonicity and Spectral Smoothness," *IEEE Trans. on Speech and Audio Processing*, vol. 11, no. 6, November 2003.

[6] A. Sheh and D. Ellis, "Chord segmentation and recognition using em-trained hidden Markov models," *in 4th Int. Symposium on Music Information Retrieval ISMIR-03*, October 2003

[7] G. Monti, M.Sandler "Automatic polyphonic piano note extraction using fuzzy logic in a blackboard system" *Proc. of the 5th Int. Conference on Digital Audio Effects (DAFx-02)*, Hamburg, Germany, September 26-28,2002.

[8] A.T. Cemgil "Bayesian Music Transcription" PhD thesis, University of Nijmegen.

[9] V. Vapnik, The Nature of Statistical Learning theory, Springer-Verlag, New York, 1995.

[10] N.Cristianini and J.Shawe-Taylor, An Introduction to Support Vector Machines and other Kernel-based Learning Methods, Cambridge University Press 2000.

[11] C.Burges, "A tutorial on support vector machines for pattern recognition", Data Mining and Knowledge Discovery, 2(2):121--167, 1998.

[12] G.Zoia, R.Zhou, D. Mlynek "A multi-timbre chord/harmony analyzer based on signal processing and neural networks", *IEEE Int. Workshop on Multimedia Signal Processing – MMSP2004*, Siena, Italy, September 2004

[13] Chang, C.-C and C.-J.Lin . LIBSVM: a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm, 2001.

[14] C-W.Hsu, C-C.Chang and C-J.Lin "A practical guide to support vector machine classification," available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[15] C. Cortes and V. Vapnik "Support-vector network," Machine Learning 20, 273–297, 1995.

[16] Keren, R., Zeevi, Y. Y., and Chazan, D "Multiresolution Time-Frequency Analysis of Polyphonic Music," *Proc. of the IEEE-SP International Symposium on Time-Frequency and Time-Scale Analysis,* Pittsburgh, PA, U.S.A., pp. 564-568, Oct 6-9, 1998.

[17] Levine, S. N. "Audio Representations for Data Compression and Compressed Domain Processing," *PhD thesis*, Stanford University, CA, U.S. A., 1998.

[18] Jang, H.K., and Park, J. S. "Multiresolution Sinusoidal Model with Dynamic Segmentation for Time-scale Modification of Polyphonic Audio Signals," *IEEE Trans. on Speech and Audio Signals*, Vol. 13, No. 2, pp. 254-262, March 2005

[19] Xu, C., Maddage, N. C., Shao, X., Cao, F., and Tian, Q "Musical Genre Classification using Support Vector Machines," *Proc. of ICASSP*, Singapore 2003.