Extraction of Musical Performance Parameters from Audio Data

Simon Dixon

Austrian Research Institute for Artificial Intelligence
Schottengasse 3, A-1010 Vienna, Austria
simon@ai.univie.ac.at

Abstract

We present a system for the automatic extraction of musical content from audio signals containing polyphonic music. The system works off-line, taking data from audio files and producing MIDI output, representing the pitch, timing and volume of the musical notes. The initial signal processing stage is based on a STFT enhanced by a tracking phase vocoder, which interprets stable frequency components as partials of musical notes. Heuristic methods combine these partials, using a generic instrument model, to produce note estimates. The system is tested on a large corpus of professionally performed music from the standard classical piano repertoire.

1 INTRODUCTION

With the recent advances in multimedia capabilities of computers, the need for intelligent processing of multimedia content has risen greatly. Most work in this area has focussed on video and speech; relatively little has been achieved in the area of non-speech audio, such as music. This paper addresses the problem of extracting musical content from audio data. More specifically, we consider the task of ascertaining performance parameters from polyphonic music, that is, calculating which notes were played (*pitch*), when they were played (*timing*), and how loud they were played (*velocity*[1]). For piano music, these parameters almost completely characterize the performer's contribution to the performance. Other factors such as instrument and room acoustics are not considered in this paper.

A natural application of this work is to the problem of automatic transcription, that is, generating a representation in common music notation of the musical content of an audio signal. Although this is similar to the task performed in this work, we do not address the additional issues of rhythm understanding, quantization, key finding, note naming and page layout, which would be required for a transcription system. Instead, we focus on more precise determination of performance parameters in order to use the resulting output in studies of musical expression.

The signal processing techniques employed in this work are loosely based on the STFT and the tracking phase vocoder. The phase information obtained from adjacent FFT windows is used to resolve the ambiguity in the low end of the frequency spectrum, so that smaller windows can be used and a correspondingly better time resolution obtained. The input to the system is digital audio, taken from CD's or synthesised by a high quality software synthesizer. The output is a symbolic representation of the music, in MIDI (musical instrument digital interface) format, which is an event-based representation corresponding to synthesizer control information. Apart from its current use in performance analysis, the system could function as a front end for an automatic transcription system or for content-based indexing and retrieval.

In section 2, we briefly review the relevant literature, and then in section 3 describe the system itself. The following section outlines the testing methodology and preliminary results, and we conclude with a discussion of the results and further work.

2 RELATED WORK

Most related work consists of various attempts at automatic transcription in some limited domain [7, 8, 1, 6, 11, 12, 3, 9, 5, 4]. The current state of the art is that pitch and timing for a known instrument playing one note at a time can be detected quite reliably, and is commercially available in hardware and software realisations. Polyphonic transcription has only been performed successfully with severely restrictive conditions on the input data. Space does not permit reference to more than a small number of the approaches used (see [4] for a more complete review).

The pioneering work of Moorer [7] used comb filters and autocorrelation to perform transcription of very restricted duets. The input data was allowed to contain no more than two notes sounding simultaneously, and note combinations which shared common frequency components (e.g. octaves) were not allowed, so that the components could be interpreted unambiguously. The range of notes was restricted to two octaves. Schloss [11] developed useful time domain techniques for accurate estimation of onset times in his work on transcription of untuned percussion, but did not address pitch extraction. Martin [5] allowed up to 4 voices in the input data, but it was restricted to the chorale style of J.S. Bach, with all parts

[1] we will use MIDI terminology throughout this paper

played by synthesised piano. Furthermore, octave intervals were not allowed, and the note range was restricted to under 2 octaves (f_0 = 123–440Hz). Klapuri [4] allowed a 5 octave fundamental frequency range (65–2093Hz), but required example notes covering the complete range of each instrument in order to train the system. Good results were achieved for the stated test examples; it is not clear how the system would perform on more complex musical examples. The only work explicitly concerned with extraction of performance parameters is that of Scheirer [9, 10], who required that the musical score be provided to guide his system. The problem with most of these systems is lack of extensibility, due to overly restrictive assumptions made about the input data. Therefore, each new attempt starts from zero, rather than building on previous work.

3 SYSTEM DESCRIPTION

The basic design philosophy was to create an open system, avoiding design decisions which would commit the system to specific musical assumptions, such as the number of simultaneous notes, the frequency range or the instruments used. The generic system can then be specialized if necessary to take advantage of known features of particular input data. The modularity of the design also facilitates the replacement of system components with alternative implementations. The code is written in C++, and runs on Unix platforms (Linux and Solaris).

Figure 1 illustrates the stages of processing performed by the system. After low-pass filtering and downsampling the signal to a 12kHz sampling rate, the signal is windowed and converted to a frequency domain representation using a short-time Fourier transform. The specific parameter values (all adjustable from the command line) were a window size of 4096 samples (341ms), containing 230ms of signal shaped with a Hamming window and zero padded to fill the window, and hop size of 20ms.

The complex frequency domain data is then converted into magnitude squared (power) and phase values. An adaptive peak-picking algorithm finds spectral peaks, which give an initial estimate of the significant frequency components in each window of the signal.

Figure 2 shows the trade-off between time and frequency resolution inherent to Fourier analysis: the high frequencies are clearly resolved, but the lower frequecies are blurred by the use of a short time window and logarithmic frequency scale. A longer time window would help to solve this problem, but only at the expense of creating an alternative problem of insufficient resolution in time.

We use a method based on the phase vocoder [2] in order to obtain greater frequency resolution at low frequencies. Rather than using the centre frequency of FFT bins, a more accurate estimate of frequency is obtained by examining the rate of change of phase in each bin. In the bins surrounding a spectral peak, the rate of phase change corresponds with the true frequency in the signal, and is usually stable across a number of frequency bins. Figure 3 shows the relationship between bin centre frequency and rate of phase change (top), with the magnitude spectrum shown for reference at the bottom of the figure. A clear correspondence occurs between spectral peaks and horizontal segments in the graph, where the power from a single frequency component is spread across a number of FFT bins.

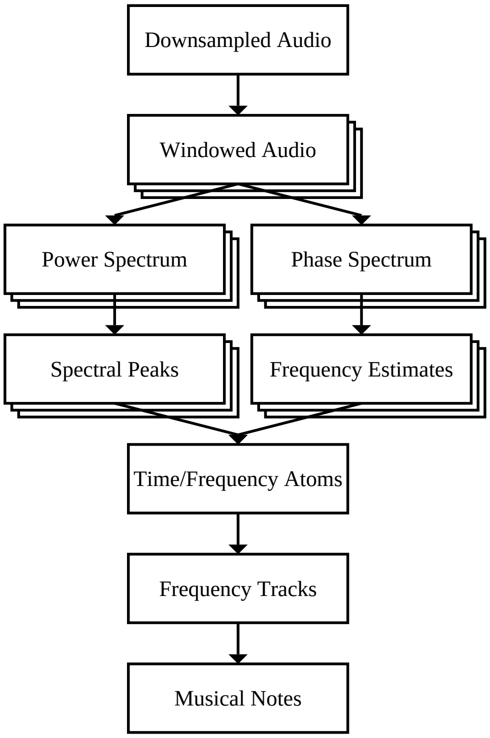

Figure 1: Data flow through the system

The peaks in the power spectrum and the frequency estimates from the rate of phase change calculation are combined to give atoms of energy localised in time and frequency. Atoms with significant energy are then traced in time, giving frequency tracks which represent partials or harmonics of the musical notes. The frequency tracks are updated by a few simple rules, for example to remove the tracks caused by transients at note onsets, which occur as frequency tracks with very short durations. The final step is to interpret the frequency tracks as musical notes, which is done by finding a set of fundamental frequencies which provides the best explanation for the observed frequency data, relative to an implicit generic model of musical instrument tones.

4 EVALUATION AND RESULTS

Evaluation of audio content analysis is hindered by a lack of suitable test data. In the field of speech recognition the availability of large corpora of tagged speech data enables both large scale testing and the use of statistical, machine learning and iterative improvement algorithms. There are

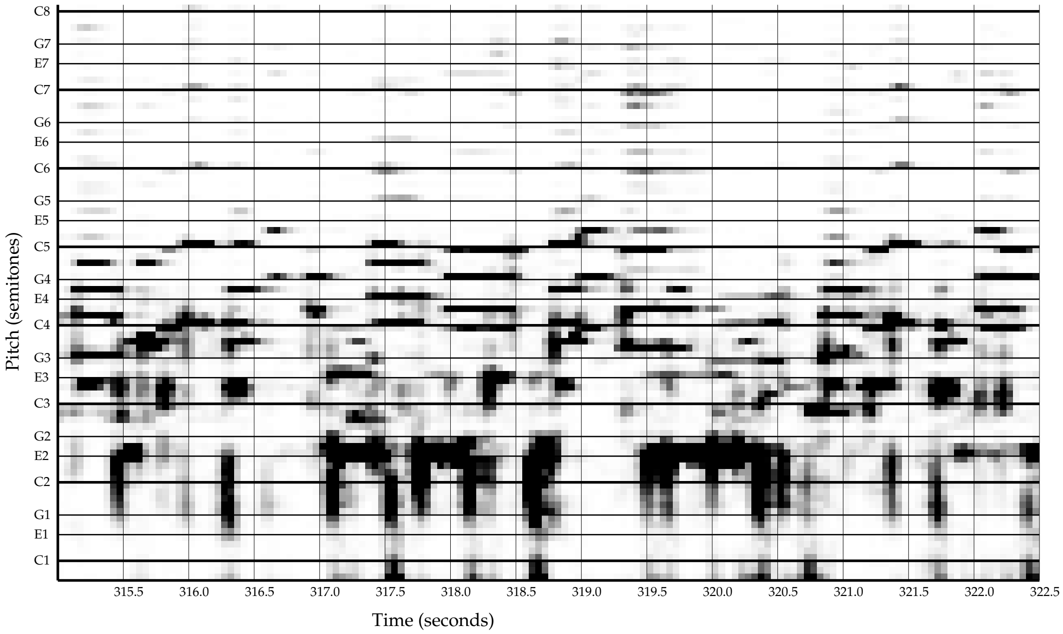

Figure 2: Power spectrogram from STFT with Gaussian window, showing poor resolution at low frequencies. The excerpt is 7.5 seconds of solo electric guitar music.

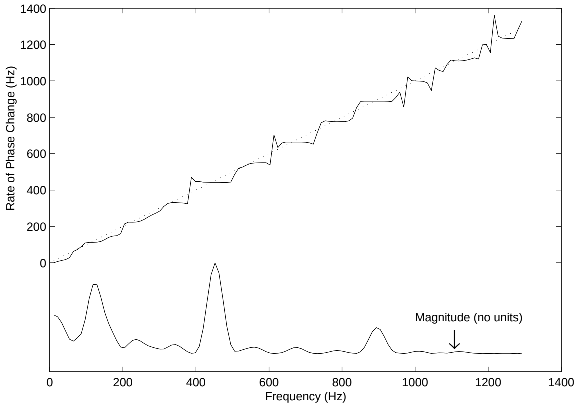

Figure 3: Rate of change of phase in FFT frequency bins (signal magnitude is shown below for reference)

no similar large corpora of musical data, so we chose to synthesize test data from MIDI representations of professional performances. Note that although the musical score provides sufficient information for the evaluation of a transcription system, it does not provide the expressive details of timing and dynamics which we require for testing our system. These are rarely, if ever, available in conjunction with audio recordings, but are represented in MIDI data.

In order to ensure that the system was tested with a wide range of musical situations, a data set of 13 Mozart piano sonatas (4 hours of music; over 100000 notes) was obtained in MIDI format, and audio data was generated from MIDI files using high quality software synthesis. The accuracy of the note recognition system was tested by comparison of pairs of MIDI files – the input files from which the audio data was generated, and the output files of notes detected by the system.

A matching algorithm is used to pair corresponding events in the input and output MIDI files. Events are judged to correspond if they have the same pitch and onset times differing by no more than a small error margin (70ms). The results are evaluated in terms of the number of paired notes (N), the number of false positives (FP = the number of notes reported by the system that were not played) and the number of false negatives (FN = the number of notes played that were not reported by the system). An incorrectly identified note (e.g. wrong pitch) is counted as both a false positive (the wrongly reported note) and a false negative (the note that should have been reported), which makes the evaluation metric relatively harsh on this type of error. The three figures are combined with the following formula into a single score (expressed as a percentage):

$$Score = \frac{N}{FP + FN + N}$$

To test the system with various instrument sounds, we selected several representative synthesizer voices from the General MIDI Specification. The results are shown in Figure 4. A default set of parameters were used for all voices except acpiano*, where the amplitude threshold parameters were adjusted to show the sensitivity of the system to instrument amplitude. Clearly the variation in results shows also that the system is very sensitivity to instrumental timbre.

Voice	N	FP	FN	Score
acpiano	95443	32053	11016	68.9%
acpiano*	95914	21433	10545	75.0%
britepno	87331	18185	19128	70.1%
honky	93777	8227	12682	81.8%
hrpschrd	93134	41136	13325	63.1%
marimba	92128	11306	14331	78.2%
violin	74528	146299	31931	29.5%
flute	89882	12109	16577	75.8%

Figure 4: Preliminary Results

5 DISCUSSION

Many aspects of the system are yet to be implemented, most notably the calculation of suitable thresholds and parameters. Nevertheless, the preliminary results are positive. The system currently uses no knowledge of the sound sources, being based on a very generic instrument model, which assumes only that notes are harmonic with most of the energy at the lower partials. Accurate sound source modelling will improve the system's performance considerably, as has been shown by [4].

Although synthetic data was used, its quality is sufficiently high that it is unlikely to affect results significantly when data from natural instruments is used. Other authors have used synthetic data even for small-scale tests [5].

Planned extensions of the work are to develop instrumental models dynamically, so that the system tunes itself to the instruments and acoustic conditions, much as a human listener does. A specific "hard-coded" system for acoustic piano is also under development, for use in the study of musical expression. Also planned for this study, where score information is often available, is an investigation of the incorporation of score knowledge into the system, as performed by [10]. The matching algorithm also needs further development, so that instead of using a fixed time tolerance with binary acceptance, a more graded evaluation function should be used which evaluates the extent of timing errors. Also, incorrectly identified notes should be classified into common error types (e.g. octave errors); these and other perceptually reasonable errors, such as those due to masking, should be penalized less harshly than they are currently. In further work we will also assess the accuracy of the dynamics and offset times reported by the system, which are more difficult problems, but perhaps less critical in their accuracy.

ACKNOWLEDGEMENTS

This research is part of the project Y99-INF, sponsored by the Austrian Federal Ministry of Education, Science and Culture in the form of a START Research Prize and support to the Austrian Research Institute for Artificial Intelligence. We also thank the L. Bösendorfer Company, Vienna, for the performance data used in these experiments.

References

[1] C. Chafe, D. Jaffe, K. Kashima, B. Mont-Reynaud and J. Smith. Techniques for note identification in polyphonic music. In *Proceedings of the International Computer Music Conference*. Computer Music Association, San Francisco CA, 1985.

[2] J.L. Flanagan and R.M. Golden. Phase vocoder. *Bell System Technical Journal*, Volume 45, pages 1493–1509, 1966.

[3] K. Kashino, K. Nakadai, T. Kinoshita and H. Tanaka. Organization of hierarchical perceptual sounds: Music scene analysis with autonomous processing modules and a quantitative information integration mechanism. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 1995.

[4] A. Klapuri. Automatic transcription of music. Master's thesis, Tampere University of Technology, Department of Information Technology, 1998.

[5] K.D. Martin. A blackboard system for automatic transcription of simple polyphonic music. Technical Report 385, Massachussets Institute of Technology Media Laboratory, Perceptual Computing Section, 1996.

[6] B. Mont-Reynaud. Problem-solving strategies in a music transcription system. In *Proceedings of the International Joint Conference on Artificial Intelligence*. Morgan Kaufmann, 1985.

[7] J.A. Moorer. *On the Segmentation and Analysis of Continuous Musical Sound by Digital Computer*. Ph.D. thesis, Stanford University, CCRMA, 1975.

[8] M. Piszczalski and B. Galler. Automatic music transcription. *Computer Music Journal*, Volume 1, Number 4, pages 24–31, 1977.

[9] E.D. Scheirer. Extracting expressive performance information from recorded music. Master's thesis, Massachusetts Institute of Technology, Media Laboratory, 1995.

[10] E.D. Scheirer. Using musical knowledge to extract expressive performance information from audio recordings. In H. Okuno and D. Rosenthal (editors), *Readings in Computational Auditory Scene Analysis*. Lawrence Erlbaum, 1997.

[11] W.A. Schloss. *On the Automatic Transcription of Percussive Music: From Acoustic Signal to High Level Analysis*. Ph.D. thesis, Stanford University, CCRMA, 1985.

[12] C.R. Watson. *The Computer Analysis of Polyphonic Music*. Ph.D. thesis, University of Sydney, Basser Department of Computer Science, 1985.