On the Computer Recognition of Solo Piano Music

Simon Dixon
Austrian Research Institute for Artificial Intelligence
Schottengasse 3, A-1010 Vienna, Austria
simon@ai.univie.ac.at
Phone: +43-1-533 6112 22 Fax: +43-1-533 6112 77

Abstract

We present work towards a computer system for the automatic transcription of piano performances. The system takes audio files containing polyphonic piano music as input, and produces MIDI output, representing the pitch, timing and volume of the musical notes. The aim of this work is not to reduce the performance data to common music notation, but to extract the performance parameters for a quantitative study of musical expression in piano performance. Standard signal processing techniques based on the short time Fourier transform are used to create a time-frequency representation of the signal, and adaptive peak-picking and pattern matching algorithms are employed to find the musical notes. In order to perform large scale testing, the test process is automated by synthesizing audio data from MIDI files using high quality sofware synthesis, and comparing results with the original MIDI data. The test data used is Mozart piano sonatas performed by a concert pianist.

1 INTRODUCTION

This paper addresses the problem of extracting musical content from audio data. More specifically, we consider the task of ascertaining performance parameters from polyphonic piano music, that is, calculating which notes were played (*pitch*), when they were played (*timing*), and how loud they were played (*velocity*[1]).

This is closely related to the task of automatic transcription, whereby musical notation is created from audio recordings of music. However, common music notation is not suitable for accurately representing a musical performance, as it does not describe the velocities of individual notes, nor the precise timing of onsets and offsets, using instead a more abstract discrete set of notated durations. In order to study musical expression, one of the goals of the current project, such extra details are required. In this sense, the accuracy required in this project is much greater than for a transcription system, which filters out the expressive details as if they were noise. On the other hand, several tasks necessary for transcription into common music notation are not addressed in this paper which would be required for a complete transcription system. Examples of these tasks are: rhythm understanding, quantization, key finding, note naming and the physical layout of musical symbols on a page[2].

The input to the system is digital audio, taken from CD's or synthesised by a high quality software synthesizer. Processing begins with time-frequency analysis based on the windowed short time Fourier transform, from which peaks in

[1] MIDI terminology is used throughout this paper.

[2] see (Dixon and Cambouropoulos, 2000; Cambouropoulos, 2000) for recent work on some of these tasks.

the time-frequency terrain are extracted, creating frequency tracks which are then grouped as partials of musical notes. The output is a symbolic representation of the music, in MIDI format, representing the musical events detected from the performance. Thus the system could function as a front end for an automatic transcription system, for content-based indexing and retrieval, or for a performance analysis system.

In section 2, we briefly review the previous work in this field, and then in section 3 describe the system itself. The following section outlines the testing methodology, and in section 5 we present the results obtained to date and an evaluation of the system's performance. The concluding section contains a discussion of the results, and lists possible directions of further work.

2 RELATED WORK

Most related work consists of various attempts at building an automatic transcription system, each of which tackles a different subset of the transcription problem (Moorer, 1975; Piszczalski and Galler, 1977; Chafe et al., 1985; Mont-Reynaud, 1985; Schloss, 1985; Watson, 1985; Kashino et al., 1995; Martin, 1996; Klapuri, 1998). 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. However, rhythm understanding even in this limited case is still quite poor in these commercial systems, most of which require a strict metrical performance synchronized with a pre-specified beat. Polyphonic transcription has only been performed successfully with restrictive conditions placed on the input data. Space does not permit reference to more than a small number of the approaches used[3].

The pioneering work of Moorer (Moorer, 1975) 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, in order that the frequency components could be interpreted unambiguously. The range of notes was restricted to two octaves. Schloss (Schloss, 1985) 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 (Martin, 1996) allowed up to 4 voices in the input data, but it was restricted to the chorale style of J.S. Bach, in which the notes have relatively long duration and change simultaneously, which made it possible for him to segment the signal in the time domain into "musically constant" sections. Furthermore, octave intervals were not allowed, and the note range was restricted to under 2 octaves (f_0 = 123 - 440Hz). Klapuri (Klapuri, 1998) 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 (Scheirer, 1995; Scheirer, 1997), who also dealt with solo piano music, and built a system which required that the musical score be provided to guide the system. Knowing in advance which notes were played, means that the system is only required to search for onsets, offsets and velocities of notes at known frequencies and within quite small time windows. Furthermore, the interactions between notes (e.g. shared frequency components) can be predicted and avoided (as is done in Scheirer's system). Thus, although the techniques developed in this system are relevant to the "blind" transcription problem, they are not immediately reusable, and some are not at all.

[3] see (Klapuri, 1998) for a more complete review.

3 SYSTEM DESCRIPTION

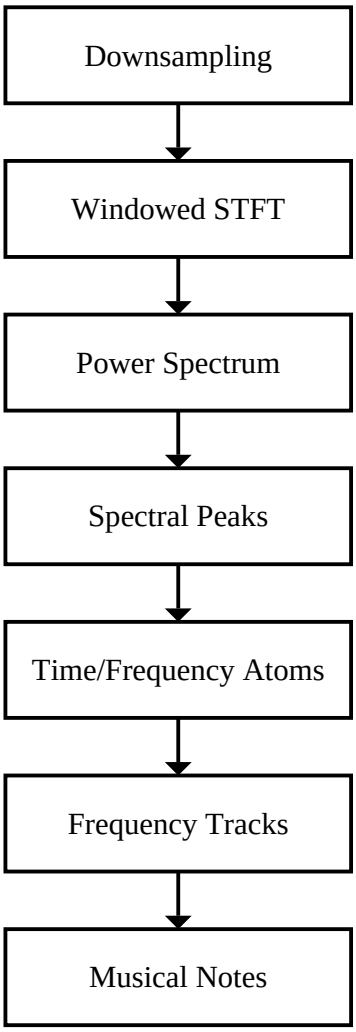

Figure 1: Data-processing stages

We now briefly describe the stages of processing performed by our system, shown diagrammatically in Figure 1. The first stage of processing consists of down-sampling the signal to a 12kHz sampling rate, in order to decrease the processing time of subsequent stages. The use of a lower bandwidth signal does not appear to affect results adversely. A time-frequency representation is then created using a windowed short-time Fourier transform. By default, a window of 4096 samples (341 ms) is used, containing 230ms of signal shaped by a Hamming window and zero-padded to fill the window. Command line parameters can be used to select windows with different sizes and shapes.

The complex frequency domain data is then converted into a magnitude squared (power) spectrum and an adaptive peak-picking algorithm finds spectral peaks, which give an initial estimate of the significant frequency components in each window of the signal. These peaks are represented on a logarithmic frequency scale, to conform with the musical representation of pitch in semitones.

Figure 2 shows the resulting power spectrum and a typical problem in time-frequency analysis of music: in order to get a sufficiently good frequency resolution to determine pitch accurately, the time resolution is poor, and a large amount of overlap is seen between notes in scale passages. This is not an insurmountable problem for the transcription of piano music, since the piano has a very sharp attack, and by using overlapping windows a reasonably accurate estimate of onset times can be obtained.

A current extension of the system, aimed at improving the time-frequency resolution tradeoff, uses the rate of change of phase in the FFT filterbank channels, rather than the centre frequency of the channels, in order to obtain a more accurate estimate of frequency, and in turn allow a smaller window size to be used. This idea was first used in the phase vocoder of (Flanagan and Golden, 1966), and has since been used in many computer music applications (Dolson, 1986).

The peaks in the power spectrum are then isolated by finding at each time point the local maxima in the frequency dimension which are above a minimum threshold and which contain at least 1% of the total power of the signal at that time, giving a set of atoms of energy localised in time and frequency. The time-frequency atoms are then searched in the time dimension, in order to find the peaks in time corresponding to note onsets, and these onsets are followed until the power drops below the minimum threshold, which determines the offset time of the note. Fi-

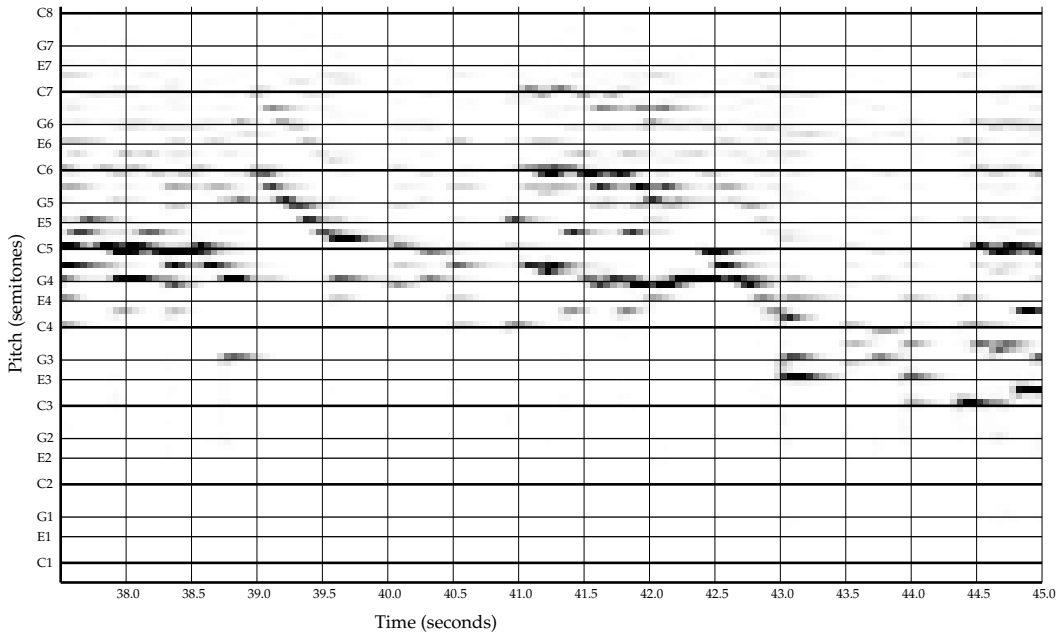

Figure 2: Power spectrogram from STFT with Hamming window, showing 7.5 seconds of Mozart's Piano Sonata in C major, KV279, 3rd movement.

nally, the velocity is determined from the peak power, which occurs at the onset time of the note. The frequency tracks calculated in this way represent partials (harmonics) of the musical notes. A few simple rules are employed to eliminate rogue frequency tracks, such as those with very short durations, which may be, for example, caused by transients at note onsets.

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. That is, we combine partials which occur simultaneously, are harmonically related, and do not fall outside the expected spectral envelope for piano tones. Due to considerable variability in the individual notes of a piano, this final condition cannot be particularly stringent, but is necessary in order to be able to detect intervals such as octaves. The resulting output, in MIDI format, is then evaluated as described in the following sections.

4 TEST METHODOLOGY

One of the greatest difficulties in building audio analysis systems is the lack of high quality tagged test data. In the field of speech recognition there are large corpora of tagged audio, making possible the use of statistical and machine learning methods such as hidden Markov models in analysis of audio with speech content. Music has no similar large corpora of data, so these methods cannot be used. Similarly, thorough quantitative testing is also made difficult by the lack of suitable test data. Although it is becoming easier to generate test data using, for example, a Bösendorfer SE290 computer-monitored piano or a Yamaha Disklavier, a further hindrance to gathering large data sets is the legal issues associated with copyright of professionally performed musical data.

In order to test our system, we require a symbolic representation of the audio content of the test data, which is more than just the musical score; we also need the expressive details

of timing and dynamics. These are rarely, if ever, available in conjunction with audio recordings. Manual transcriptions can be performed, but are extremely time-consuming to produce, and cannot guarantee a high degree of accuracy or precision. In the literature to date, testing is usually performed manually, using simple musical excerpts, normally no more than 30 seconds long. However, we have developed and tested our system using music from the standard classical repertoire, in this case Mozart piano sonatas, performed by a professional pianist. These provide a realistic source of noisy data, and a far more difficult data set than has been attempted elsewhere.

In order to ensure that the system was tested with a wide range of musical situations, a large data set was obtained, consisting of 13 complete Mozart piano sonatas (KV 279-284, 330-333, 457, 475 and 533). These were performed on a Bösendorfer SE290 computer-monitored piano, and were converted to MIDI format using conversion software. As the original audio data from the performances was not available, the audio data was then generated from the MIDI files using a software synthesis program, Timidity. A number of different synthesizer voices were chosen to test the sensitivity of the algorithm to the instrument timbre. 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 detected notes.

5 RESULTS

A matching algorithm was developed to pair events in the input file with corresponding events in the output file, under the constraints that the notes must have the same pitch and onset times differing by no more than a small error margin (70ms). The results are evaluated in terms of false positives (FP = the number of notes reported by the system that were not played) and 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 is perhaps an unnecessarily harsh evaluation metric. The error figures are combined with the following formula into a single percentage score (where N is the number of correctly identified notes):

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

Voice	N	FP	FN	Score
acpiano	95443	32053	11016	68.9%
britepno	87331	18185	19128	70.1%
honky	93777	8227	12682	81.8%
acpiano*	95914	21433	10545	75.0%

Figure 3: Results for 3 different piano sounds

In Figure 3 we present results computed for the complete test data set for three different synthesizer voices. The voices are labelled with their names from the General MIDI Specification, representing a standard acoustic piano (acpiano), a brighter sounding piano (britepno), and a honky-tonk piano sound (honky), respectively. These results were generated with a single set of parameters, and show a recognition accuracy of around 70 - 80%. The differences between these three results demonstrate a weakness of the current system, that it is sensitive to the amplitude and timbre of the instrumental sound which is used. To demonstrate this more clearly, we tuned the parameters to obtain better values for the acoustic piano sound, as shown in the last row of the table (acpiano*).

6 DISCUSSION

Although the system is far from complete, the preliminary results are positive. The stated aim of this paper is to recognise piano notes, but

as yet, the piano-specific assumptions have not been built into the system, and the software framework developed is suitable for a range of instruments. It is clear that results can be improved by modelling the sound source accurately, as is done by (Klapuri, 1998). One planned extension of this work is to build a recognition module that is specific to acoustic grand pianos.

But before tuning the system to particular instrument models, we intend to address the problem of sensitivity of results to the particular instrument by using dynamic modelling (Dixon, 1996) to automatically determine suitable parameter values from the audio data, rather than requiring the user to determine these values by trial and error. This is particularly important for the more general problem of transcription of music from unknown instruments, and is a more elegant approach than hard-wiring instrument parameters into the recognition algorithms. One way to implement dynamic modelling is the use of artificial intelligence iterative improvement algorithms, so that the system can learn automatically to improve its performance, using feedback from the evaluation part of the system.

At this point in time, the use of synthesised data was a pragmatic necessity, in order to make large-scale testing possible. Other authors have used the same method even for small-scale tests (Martin, 1996). The performance of the system on non-synthesized input data (i.e. recordings from acoustic piano) will be tested as the data becomes available.

Another area requiring further work is the evaluation function. Currently, the matching algorithm uses a fixed time tolerance, giving a binary result, which should be replaced by a more gracefully degrading accuracy value. Similarly, the note identification is an all-or-nothing value, which could be improved by classifying incorrectly identified notes into common error types (e.g. octave errors). Finally, the evaluation function should also take the perceptual strength of errors into account, as the input data contains a number of notes which are neither perceptually nor physically detectable from the audio signal, which are currently counted as errors when not detected by the system. For example, a key adjacent to a played key is sometimes brushed sufficiently hard to be detected by the monitoring system, even though no audible sound is produced.

Further development of the system is required to assess and improve the accuracy of the dynamics and offset times reported by the system. These are much more difficult problems than onset detection. For example, the physical release of a key may occur long after the note is no longer audible, making the offset time irrelevant. Similarly, the amplitude of a note, which is represented by the velocity of the hammer as it hits the string(s), is dependent on other factors not represented, such as interactions (coupling) between the vibrations of different strings and also of the piano frame and soundboard.

Finally, we plan to extend this work to investigate expressive timing. One approach would be to use time domain analysis to identify the onsets of notes more accurately. We also intend to link this work with recent work on beat tracking (Dixon and Cambouropoulos, 2000), in order to separate the different levels of expressive timing in the music, that is to separate local changes, such as the displacements of events from their nominal temporal position, from more general changes, such as an increase in the average tempo.

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

Cambouropoulos, E. (2000). From MIDI to traditional musical notation. In *Proceedings of the AAAI Workshop on Artificial Intelligence and Music: Towards Formal Models for Composition, Performance and Analysis*. AAAI Press. To appear.

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

Dixon, S. (1996). A dynamic modelling approach to music recognition. In *Proceedings of the International Computer Music Conference*, pages 83–86. Computer Music Association, San Francisco CA.

Dixon, S. and Cambouropoulos, E. (2000). Beat tracking with musical knowledge. In *ECAI 2000: Proceedings of the 14th European Conference on Artificial Intelligence*. IOS Press. To appear.

Dolson, M. (1986). The phase vocoder: A tutorial. *Computer Music Journal*, 10(4):14–27.

Flanagan, J. and Golden, R. (1966). Phase vocoder. *Bell System Technical Journal*, 45:1493–1509.

Kashino, K., Nakadai, K., Kinoshita, T., and Tanaka, H. (1995). 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*.

Klapuri, A. (1998). Automatic transcription of music. Master's thesis, Tampere University of Technology, Department of Information Technology.

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

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

Moorer, J. (1975). *On the Segmentation and Analysis of Continuous Musical Sound by Digital Computer*. PhD thesis, Stanford University, CCRMA.

Piszczalski, M. and Galler, B. (1977). Automatic music transcription. *Computer Music Journal*, 1(4):24–31.

Scheirer, E. (1995). Extracting expressive performance information from recorded music. Master's thesis, Massachusetts Institute of Technology, Media Laboratory.

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

Schloss, W. (1985). *On the Automatic Transcription of Percussive Music: From Acoustic Signal to High Level Analysis*. PhD thesis, Stanford University, CCRMA.

Watson, C. (1985). *The Computer Analysis of Polyphonic Music*. PhD thesis, University of Sydney, Basser Department of Computer Science.