TECHNIQUES FOR AUTOMATIC MUSIC TRANSCRIPTION

Juan Pablo Bello, Giuliano Monti and Mark Sandler

Department of Electronic Engineering, King's College London,
Strand, London WC2R 2LS, UK
juan.bello_correa@kcl.ac.uk, giuliano.monti@kcl.ac.uk

Abstract

Two systems are reviewed than perform automatic music transcription. The first perform monophonic transcription using an autocorrelation pitch tracker. The algorithm takes advantage of some heuristic parameters related to the similarity between image and sound in the collector. The detection is correct between notes B1 to E6 and further timbre analysis will provide the necessary parameters to reproduce a similar copy of the original sound. The second system is able to analyse simple polyphonic tracks. It is composed of a blackboard system, receiving its input from a segmentation routine in the form of an averaged STFT matrix. The blackboard contents an hypotheses database, an scheduler and knowledge sources, one of which is a neural network chord recogniser with the ability to reconfigure the operation of the system, allowing it to output more than one note hypothesis at the time. Some examples are provided to illustrate the performance and the weaknesses of the current implementation. Next steps for further development are defined.

1. Introduction

Musical transcription of audio data is the process of taking a sequence of digital data corresponding to the sound waveform and extracting from it the symbolic information related to the high-level musical structures that might be seen on a score[1].
In a very simplistic way, all the sounds employed in the music to be analysed may be described by four physical parameters, which have corresponding physiological correlates [2]:

1. Repetition rate or fundamental frequency of the sound wave, correlating with pitch.
2. Sound wave amplitude, correlating with loudness.
3. Sound wave shape, correlating with timbre.
4. Sound source location with respect to the listener, correlating with the listener's spatial perception.

The latter is not considered determinant for music transcription, and will be discarded for this investigation. The other three generate the difference between the parts that can be defined in a musical track [3]: the orchestra and the score. The orchestra is the sound of the instrument itself, the specific characteristics of the instruments (timbre, envelope), which make it sound unique; the score consists of the general control parameters (pitch, onsets, etc), which define the music played by the instrument. In an academic music representation, just the latter can be described, *i.e. which notes to play and when to play them*. The purpose of the present work is to automatically extract score "features" from monophonic and simple polyphonic music tracks, using an autocorrelation pitch tracker and a computational reasoning model called blackboard system [4][5] and combining top-down (prediction-driven) processing with the bottom-up (data-driven) techniques already implemented in [6]. As the analysis of multitimbrical musical pieces and the extraction of expression parameters are not in the scope of the present work, just the parameters related with pitch and loudness will be considered.

2. Monophonic Transcription with autocorrelation

If the fundamental frequency of a harmonic signal is calculated, and the resulting track is visualised, it can be noticed that, for most of the duration of the notes, the pitch maintains approximately constant. This relation, so clear to the eyes, requires some comments. In order to implement some grouping criteria and rules for sounds, emphasis should be given to the similarity in human perception between image and sound [7]. Important clues can be obtained by observing carefully the plot of the pitch track. The current system doesn't use a conventional (energy based) onset detector, instead, it implements a pitch based onset detector, which is more robust with slight note changes (glissando, legato).
Monophonic music means that the performer is playing one note at a time. More than one instrument can be played, but their sounds must not overlap. This is a big limitation for the amount of input sounds that can be processed, however, it leads to fast and reliable results. Many commercial software tools are provided on Internet to help musicians in the difficult task that is transcription. Few of them dare to perform polyphonic transcription, but often the results are completely wrong.
Which information is needed?
The score is a sequence of note-events. Many music languages have been developed until now and a new standard is arising under the MPEG group [3]. The MIDI protocol [8] has been widely accepted and utilized by musicians and composers since its conception in 1982. It represents the most common example of a score file.
In order to define a note-event, three parameters are essential:

- Pitch
- Onset
- Duration

Every instrument is characterized by its own *timbre*, but the sounds created by different instruments playing the same note, will have the same pitch. Therefore, determining the pitch is equivalent to knowing which note has been played.
The onset time and the duration have also to be extracted in order to recreate the original melody.

2.1. Autocorrelation Pitch Tracking

In order to estimate the pitch in the musical signal, autocorrelation pitch tracking has been chosen, showing good detection and smooth values during the steady part of a note. The steady part of a note is just after the attack, where all the harmonics become stable and clearly marked in the spectrum.
The Autocorrelation function
An estimate of the Autocorrelation of an N-length sequence x(k) is given by:

$$r_{xx}(n) = \frac{1}{N}\sum_{K=0}^{N-n-1} x(k)\cdot x(k+n) \qquad (1)$$

Where n is the lag, or the period length, and x (n) is a time domain signal. This function is particularly useful in identifying hidden periodicities in a signal, for instance, in the weak fundamental situation. Peaks in the autocorrelation function correspond to the lags where periodicity is stronger. The zero lag autocorrelation $r_{xx}(0)$ is the energy of the signal. The autocorrelation function shows peaks for any periodicity present in the signal, therefore it is necessary to discard the maximum relative to the multiple periodicities. If the signal has high autocorrelation for a lag value, say K, it will have maximum for n*K as well, where n is a positive integer. Consequently, the first peak in the autocorrelation function, after the zero lag value, is considered as the inverse of the fundamental frequency, while the other peak values are discarded. The implementation takes advantage of some algorithms implemented by Malcolm Slaney in the 'Auditory toolbox' [9], a Matlab toolbox, freely available, implementing auditory models and functions to calculate the correlation coefficients.
Why autocorrelation ?
Autocorrelation is simple, fast and reliable. The equation (1) represents a very simple relation between the time waveform and the periodicities of the signal expressed by the autocorrelation coefficients.
The calculation of the autocorrelation is computed through the FFT, which has a computational complexity of N·log(N), where N is the length of the windowed signal. The calculation process, therefore, it is very fast. The simulations performed confirm the reliability of this method. In 1990, Brown published results of a study where the pitch of instrumental sounds was determined using autocorrelation [10]; she suggested this method to be a good frequency tracker for musical sounds.

2.2. Transcription

The transcription task is the translation from music to score. In the score all the notes played are listed in a time sequence, indicating the starting times, the durations and the pitches. The scheme of the monophonic transcription system implemented here, is illustrated in figure 1.
The outputs of the blocks are explained in the next figures. The Pitch Tracker is based on the autocorrelation method described in section 2.1. Its output is the instantaneous pitch of the signal. Beside the pitch tracker, a block calculates the envelope of the signal. This information goes to the pitch tracker, in order to skip the calculation of the pitch when the energy of the signal falls below the audibility threshold. This procedure avoids ineffective elaborations.

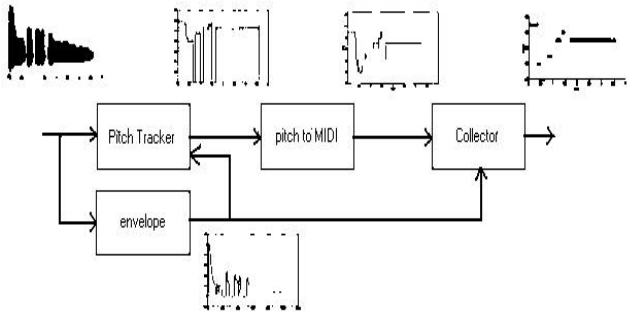

Figure 1. Scheme of the transcription system.

Figure 2 portrays the output of the pitch tracker. The pitch is set to 0 in the silence parts.

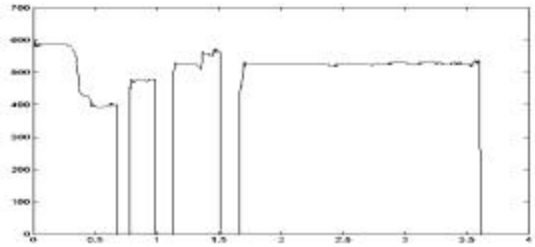

Figure 2. Pitch from autocorrelation

The conversion of the pitch (Hertz), to key number is the result of a rounding up to the nearest musical frequency. Unless the pitch, the key numbers keep the same value during the steady part of a note. The relation is given as follow [11]:

$$kn = 49 + \left(12 \times \frac{\log(f/440)}{\log(2)} \right) \qquad (2)$$

Where the $[\ ]$ operator calculates the nearest integer value. (Defined as piano keys from $A_1 = 1$, to $C_9 = 88$, with $A_5 = 49$ equivalent to the A at 440 Hz).

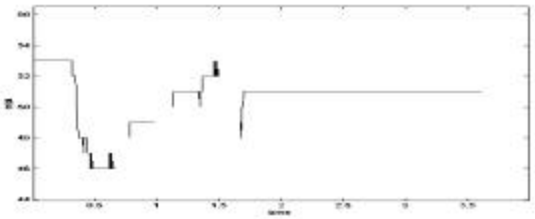

Figure 3. Pitch2MIDI conversion

If we consider a violin vibrato, in the rounding up process all the information regarding the frequency modulation are lost. However, the absence of frequency modulation in the synthesized sound has little effect on the perceptual response to

violin vibrato, while the absence of amplitude modulation causes marked changes in sound quality [12]. Moreover, an algorithm can extract the vibrato information from the signal envelope, after the sound has been segmented note by note [13].

The collector extracts the score, considering the pre-elaborated track and the signal amplitude, sorting: *onsets, pitch and offsets*.

If the short-time autocorrelation is calculated on a monophonic music signal and the results are plotted, the pitch information is almost constant during the steady state parts of the notes. The attack part of a note is usually noisy; therefore, the pitch can oscillate in a wide range of frequency before stabilizing. The transient part can last a few tenths of msec and varies depending on the instrument family [14]. In the attack part of the signal, the pitch tracker cannot provide useful information for the transcription system.

The collector recognises when the pitch maintains the same value, and proposes a note onset in the first value of the constant sequence. The onset is confirmed when the pitch lasts for the minimum note duration accepted. When a note is recognized, the system is able to write in the score file: the onset and the pitch of the note.

The *minimum note duration* is the main parameter in the collector. By modifying its value, the system adapts to the speed of the music, improving the performance of the transcription. If the minimum note duration is set for instance to 40 msec, all the pitch sequences, with constant values, lasting less than 40 msec are discarded. Hence, errors concerning spurious notes are eliminated.

The minimum duration parameter controls also the memory of the system: when a note is detected, the pitch can vary inside the 40 msec window before having again the same value, to be considered part of the same note. This is very similar to the consideration taken in sound restoration [15]: the human brain takes information from the cochlea, and interprets them with the knowledge of the previous samples; this behaviour is called streaming or integration process in psychoacoustics [7].

The termination of a note is determined by the start of a new note or by the recognition of silence. After an onset, the offset detector checks if the signal energy falls below the audibility threshold. The duration of the note in the score is calculated by the difference between its onset and the next onset/offset.

During the decaying part of a note, the pitch can slightly change. The collector allows the pitch to have different values, until a new note is predicted. However, if the conditions for a new note aren't met, the system keeps the last note.

2.3. Results

The number of lags considered in the autocorrelation determines the pitch range of the transcription system. The following table gives an idea of the relation between the autocorrelation coefficients considered and the pitch range covered (notes).

No. coeff.	From	to
256	E2	C6
512	B1	E6

Table 1. Relation between the number of autocorrelation coefficients and pitch range in the transcription system.

The configuration with 512 coefficients was chosen in the transcription. The wider pitch range covered was preferred to the faster computational time with 256 coefficients.

To verify that the pitch has been correctly tracked and the melody of the original file has not been modified, the system writes a Csound [16] score file. By providing an orchestra file, the score can be converted into wav format. The orchestra file contains the description of the instrument. Hence, from the same score, the same melody can be re-synthesised with different instruments specifying different orchestra files.

The test samples were obtained from a CD collection of brass instruments riffs. Comparative listening between the synthesised score and the original riffs, reveal the transparency of the transcription. By transparency, I mean that the tempo and the pitch are correctly extracted.

As shown in Figure 2, the matlab script also plots the segmentation of the signal (top); the black circles indicate onsets, the red circles indicate offsets. Then, the bottom figure portrays the midi notes of the score file in a "piano roll" form.

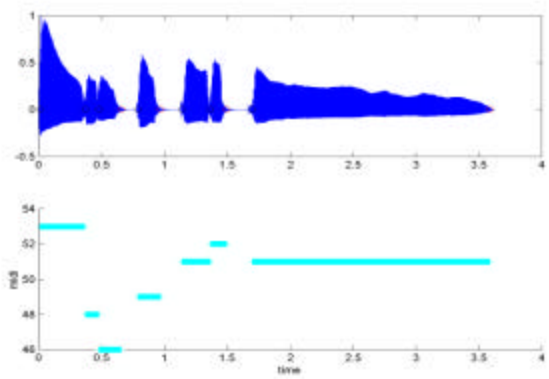

Figure 4. Original music (top) and score file (below).

It was interesting to compare this system with a commercial program downloaded from Internet, performing WAV2MIDI [17]. Even if no specification about the transcription system was given, the two systems seem to work in a very similar way. The minimum note duration can be modified in both the system. Finally, the simulations results are both fairly successful.

2.4. Conclusions

This part of the paper has reviewed a traditional method of performing pitch tracking, widely used in speech processing and has demonstrated to be also good for musical instruments. Furthermore, the implementation of a successful monophonic transcription system has been illustrated.

The transcription system described doesn't have an onset detector based on the signal waveform. The onset is recognised only at the beginning of the steady state part of the signal. As a result the onset time precision can fail of a few tens of msec. The great advantage of this approach, is that in glissando or legato passages, the onset is easily detected. This is because the new note is recognised analysing the pitch, instead of looking at the energy of the signal, which is usually ambiguous.

The pitch and time of notes are the main features in transcription. However, other features like amplitude envelope, timbre and vibrato are important to synthesize a close copy of the original sound. The spectral analysis and the signal envelope will be investigated in order to extract those parameters. Furthermore, in order to detect very low pitched notes in the range of 30-100 Hz, the pitch tracker has to be modified to provide high frequency resolution renouncing to the fast calculation of the autocorrelation function through the fft.

3. Simple Polyphonic Transcription: Blackboard System and neural network chord recogniser

The blackboard system is a relatively complex problem-solving model prescribing the organisation of knowledge and data, and the problem-solving behaviour within the overall organisation [5]. It receives its name from the metaphor of a group of experts trying to solve a problem plotted on a blackboard, each acting only when his specific area of expertise is required in the problem.

In opposition to the usual paradigm of signal processing algorithms, where algorithms are described by data flowcharts showing the progress of information along chains of modules [18], the architecture of the blackboard system is opportunistic, choosing the specific module needed for the development of the solution at each time step. Due to its open architecture different knowledge can be easily integrated into the system, allowing the utilisation of various areas of expertise. The basic structure of a blackboard system consist of three fundamental parts: *the blackboard*: global database where the hypotheses are proposed and developed, open to the interaction with all the modules present in the system; *the scheduler or opportunistic control system:* determines how the hypotheses are developed and by who; and *the knowledge sources or "experts" of the system:* modules that execute the actions intended to develop the hypotheses present in the blackboard.

The system operates in time steps, executing one action at the time. The scheduler, prioritise within the existing list of knowledge sources, determining the order in which these actions are executed. Each knowledge source consists of a sort of "if/then" (precondition/action) pair. When the precondition of a certain knowledge source is satisfied, the action described in its programming body is executed, placing its output in the blackboard. These knowledge sources can perform different kinds of activities, such as detecting and removing unsupported hypothesis from the blackboard or stimulating the search for harmonics of a given note hypothesis.

To achieve the transcription of a sound file the system can perform tasks such as:

1. The extraction of numeric parameters content in the original audio data-file, through the analysis of the output generated for signal processing methods such as the Short Time Fourier Transform (STFT), the Multiresolution Fourier Transform (MFT) [19] or the log-lag correlogram [18][20].
2. Elimination of non-audible or "irrelevant" data for the analysis performed, based on perceptual models of the ear and the brain. This helps the efficiency of the system, avoiding unnecessary computations and the generation of "impossible" hypotheses.
3. The use of musical knowledge to discern the presence of patterns or forms in the musical composition being analysed.
4. The use of "experience" for the recognition of musical structures in the audio file.

There are several implementations of blackboard systems in automatic music transcription [4][20][21], however part of the knowledge a human being use to transcribe music is based on his/her experience hearing music files and the inherent structures present on these, and in those systems this knowledge is ignored. As [18] specify, the structure of the blackboard makes little distinction between explanatory and predictive operations; hypotheses generated for modules of inference can reconfigure the operation of the system and bias the search within the solution space.

3.1 Top-Down and Bottom-up Processing

In bottom-up processing, the information flows from the low-level stage, that of the analysis of the raw signal, to the highest level representation in the system, in our case that of the note hypotheses. In this technique, the system does not know anything about the object of the analysis previous to the operation, and the result depends on the evolution of the data in its unidirectional flow through the hierarchy of the processor. This approach is also called data-driven processing. In contraposition, the approach when the different levels of the system are determined by predictive models of the analysed object or by previous knowledge of the nature of the data is known as top-down or prediction-driven processing [22].

Despite of the fact that top-down processing is believed to take place in human perception, most of the systems implemented until now are based on bottom-up processing, and just in the last years the implementation of predictive processing to recreate these perceptual tasks had become a common choice between researchers of this field [1][18][22][23]. The main reason for the implementation of top-down processing is the lack of abilities in the bottom-up systems to model important processes of the human perception; also in tasks such as the automatic transcription of music the "inflexibility" of these models make them unable to achieve results in a general context, in this particular case different types of sounds and styles of music.

In this work, the top-down processing is achieved through the implementation of a connectionist system. This kind of systems consists of many primitive cells (units), which are working in parallel and are connected via directed links. Through these links, activation patterns are distributed imitating the basic mechanism of the human brain, reason why these models are also called neural networks [24]. Knowledge is usually distributed throughout the net and stored in the structure of the topology and the weights of the links; the networks are organized by automatic training methods, which help the development of specific applications. If adequately trained, these networks can acquire the experience to make decisions in very specific problems presented. As extensive documentation of neural networks is available, no further explanation of this

topic will be developed here, just the basics of the implemented system are explained in section 3.2.3.

3.2. Implementation

3.2.1 Segmentation

As is not the focus of this paper, just a brief explanation of the system's front end is proportioned here. The onset detection aims to evaluate the time instants when a new note is played in a sound file. Whilst analysing the running spectrum of the sound it is possible to notice that when a new event occurs, the high frequency content is relevant. This property is exploited from the High Frequency Content method [25][26]. The measure of the high frequency content is given by:

$$HFC = \sum_{k=2}^{(N/2)+1} \left(|X(k)|^2 \bullet k \right) \quad (3)$$

Where N is the FFT array length ($N/2 + 1$ corresponds to the frequency $Fs/2$, Fs = sample rate) and $X(k)$ is the kth bin of the FFT. The power spectrum is weighted linearly emphasizing the high frequencies in the frame. The Energy function E is the sum of the power spectra of the signal in the specified range:

$$E = \sum_{k=2}^{(N/2)+1} \left(|X(k)|^2 \right) \quad (4)$$

In both equations the first bin is discarded to avoid unwanted DC bias. These equations are calculated on each frame and used to build the detection function:

$$DF_r = \frac{HFC_r}{HFC_{r-1}} * \frac{HFC_r}{E_r} \quad (5)$$

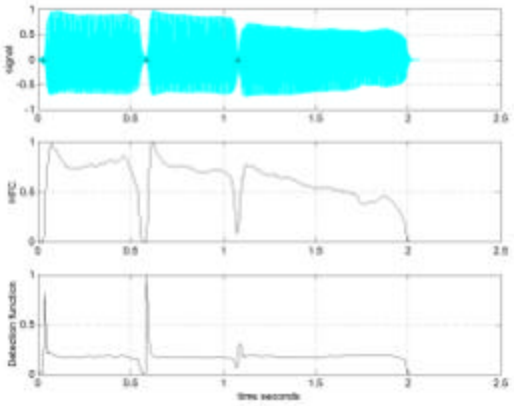

Figure 5: The original signal (a tenor sax riff) and the detected onsets and offsets (crosses) of this signal, the HFC and the Detection function.

As can be seen in figure 5, this function shows sharp peaks in the instant where the transient occurs. A criteria based on the slope of these peaks was used to determine the onset's time. After this process, the segmentation is performed averaging the signal's STFT between onsets. This is used as the input of the blackboard system.

3.2.2 Blackboard Implementation

The Blackboard system's architecture implemented is based on that of Martin's implementation [4] and is shown in figure 6.
At the lower level, the system receives the averaged STFT of the signal and identifies the peaks of the spectrum. Of this group just the peaks higher than an amplitude threshold are consider to build a *Tracks* matrix, containing the magnitude and frequency of each. This information is feed to the database and exposed to the evaluation of the knowledge's sources (KS) to produce new hypotheses.
There are three different levels of information present on the database: tracks, partials and notes. The *tracks* information is automatically provided at the beginning of the system operation, however the *notes* and *partials* information are the product of the knowledge's sources interaction with the database. It is the main task of the Scheduler to determine the need for a specific kind of information and to activate the corresponding knowledge source. In the present system a table of preconditions is evaluated at each time step and a rating is given to each knowledge source determining the order in which these will operate.

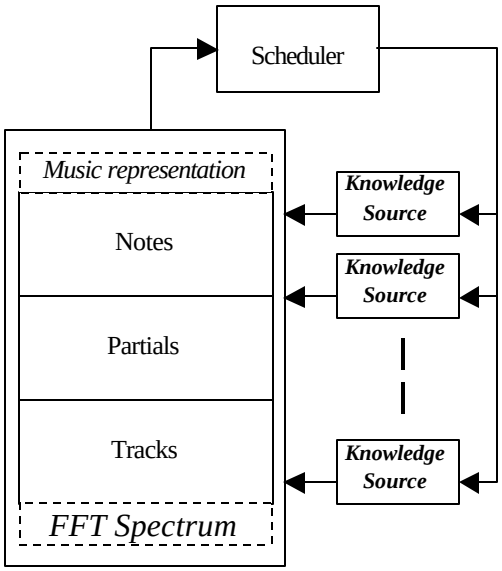

Figure 6: The control structure and the data hierarchy of the blackboard system

At *tracks* level, all the remaining peaks of the STFT have an equal chance of becoming notes, but as the operation of the system goes forward and new hypotheses are produced and evaluated by the KS, ratings are given to narrow the search for musical notes in the spectrum.
In the case of the *partials*, the rating is based on the magnitude of the nearest peak (within a specific range) to the ideal frequency of the hypothesis. For *notes*, rating is based on the presence and magnitude of peaks corresponding to the ideal

partials this note should have [27]. All this information is stored in a matrix called "Hypotheses".

3.2.3 Neural Network Implementation

In the neural network implemented, the information flows in one way from input to output. There is no feedback, which means that the output of any layer does not affect that same layer. This type of network is known as feed-forward.
The structure of this implementation consists of three layers: an input, an output and a hidden layer. The activation function implemented for all the neurons is the sigmoid transfer function. The learning is supervised. Training a feed-forward neural network with supervised learning consists of the following procedure [24]:

1. An input pattern is presented to the network. The input is then propagated forward in the net until activation reaches the output layer. This is called the forward propagation phase.
2. The output of the output layer is then compared with the teaching input. The error, i.e. the difference d_j between the output o_j and the teaching input t_j of a target output unit j, is then used together with the output o_i of the source unit i to compute the necessary changes of the link w_{ij}. To compute the deltas of inner units for which no input is available, (units of hidden layers) the deltas of the following layer, which are already computed, are used in a formula given below. In this way the errors (deltas) are propagated backward, so this phase is called backward propagation.
3. In this implementation offline learning is used, which means that the weights changes $\Delta\omega_{ij}$ are cumulated for all patterns in the training file and the sum of all changes is applied after one full cycle (epoch) through the training pattern file. This is also known as batch learning.

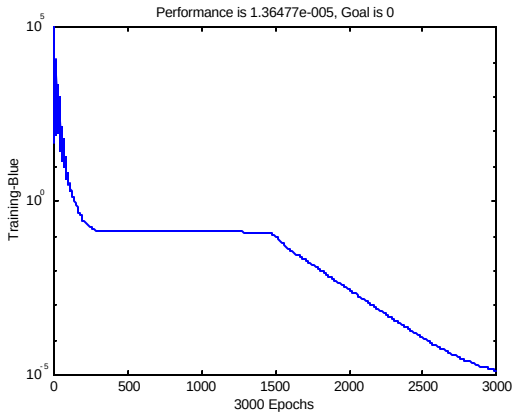

Figure 7: The learning performance of the neural network implemented.

Here, the input pattern consists of a 256 points spectrogram of a piano signal's segment (either a note or a chord), part of the batch of samples covering three octaves of the instrument. The target output is just represented for the absence "0" or presence "1" of a chord in the sample. The weight changes were calculated using the backpropagation weight update rule, also called generalized delta-rule, which reads as follows [24]:

$$\Delta w_{ij} = \eta \delta_j o_i \tag{6}$$

where,

$$\delta_j = f_j'(net_j)(t_j - o_j) \tag{7}$$

if unit j is an output unit or

$$\delta_j = f_j'(net_j)\sum_k \delta_k w_{jk} \tag{8}$$

if unit j is a hidden unit

where:

η	learning factor *eta* (a constant)
δ_j	error (difference between the real output and the teaching input) of unit j
t_j	teaching input of unit j
o_i	output of the preceding unit i
i	index of a predecessor to the current unit j with link w_{ij} from i to j
j	index of the current unit
k	index of a successor to the current unit j with link w_{jk} from j to k

The learning performance of the network is shown in figure 7, where the value of the error through the cycles can be seen.

3.2.4 Neural Network Interaction with the Blackboard

The network is trained off the process of automatic transcription until it obtains a set of parameters adequate to the task required, in this case the recognition of the presence of a chord in a spectrogram. When the overall system is running, the network receives as an input the same STFT data the blackboard system analyses. In the original blackboard's process, just the note hypotheses with rating bigger than a cut-off threshold remained as valid hypotheses [5], in this version of the system, the output of the neural network change the performance of the system allowing more than one note hypothesis to survive if necessary. This process reshapes the *Hypotheses* matrix and the routines that manipulate it, allowing the handling of a chord as a possible output of the system. In this first approach, just chords of two or three notes can be identified by the system.
After the selection of hypotheses is made, each of the frequencies obtained is rounded towards the nearest 'musical' frequency using equation (2) given in section 2.2. The key number obtained is rounded towards the nearest integer and introduced in equation (9) [11], where f_{note} is the nearest 'musical' frequency:

$$f_{note} = 440 \times 2^{(kn-49)/12} \tag{9}$$

The output is given in two different ways: a graphical representation and a score file in CSOUND™ language [28]. The graphical representation is in the form of a 'piano roll', which is a common way of representing musical events in most MIDI sequencers. The score file, is a text file written in CSOUND™ protocol, which can be compiled and rendered

with an Orchestra file (a sine wave sound for these experiments), obtaining an audio representation of the original sound.

3.3 Examples

Three examples are shown here to illustrate how the system is working and to define the next steps to follow. In the first example, illustrated in figure 8, a piano riff is plotted, consisting on a succession of four notes (C_5 D_5 E_5 F_5) followed by a C major chord (C_5 E_5 G_5). The notes and the chords are recognized successfully by the system, which plot the output according to the key number of each note. This example is intended just to show the main capabilities of the current system, notice that the notes and silences are well differentiated and the network identified the presence of a chord related with the last onset, causing the blackboard to output the three higher rated hypotheses of the segment.

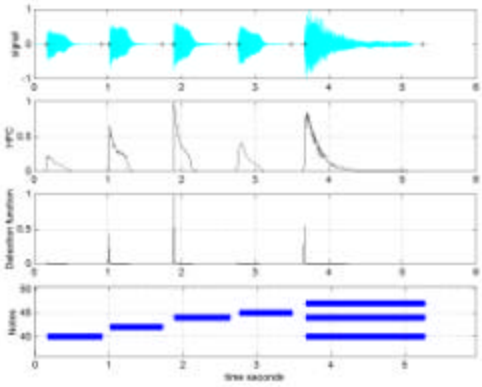

Figure 8: Example of automatic transcription of a piano riff.

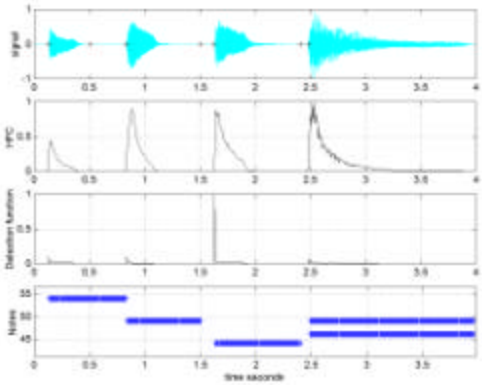

Figure 9: System's output of a piano riff with error in the chord transcription.

In figure 9 a note sequence (D_6 A_5 E_5) is represented followed by a D major chord (D_5 $F\#_5$ A_5). It is plotted here because an error is performed in the recognition of the chord showing one of the weaknesses of the current implementation. This error is due to the presence of a high rated hypothesis for the A_4 note, product of the strong harmonics of the A_5 in the D major chord. These make the hypothesis of the note A_4 better rated than the D_5 missing note. As this problem became repetitive in some of the experiments, an octave-error detection routine was implemented and placed after the output of the blackboard. In this example the error detection routine discarded the note A_4 in favour of its higher octave equivalent, which was already detected by the blackboard, leaving empty the output slot corresponding to the D_5 note of the chord. This extra routine disables the system's handling of octave intervals.

The last example shown in figure 10 represents a four measures section of a piano song, including several chords. For this example the octave error detector was disabled, to avoid restrictions in the kind of intervals the system can manage. Due to this, several mistakes are made in the transcription of the notes of three chords (the first three of the figure), where correct note hypotheses were discarded by the system in favour of their lower octave equivalents. In the plot can be noticed the presence of notes between the key numbers 18 and 29, when just notes of the key number 30 or higher were performed. The same error was detected in the note before the last chord, where the note C_6 was selected over the correct C_5. Another error in the transcription is related to the no detection of an onset in the seventh second of the song causing a wrong segmentation of the piece. The spectrogram of this segment was identified as a chord for the neural network, probably due to the presence of two strong fundamental pitches in the time window averaged. As can be seen in the figure 6 an inexistent chord was plotted between the times of 6.7164 and 7.3839 seconds, containing both the original notes played in that segment. The other twelve notes of the piece and the last chord were correctly identified by the system.

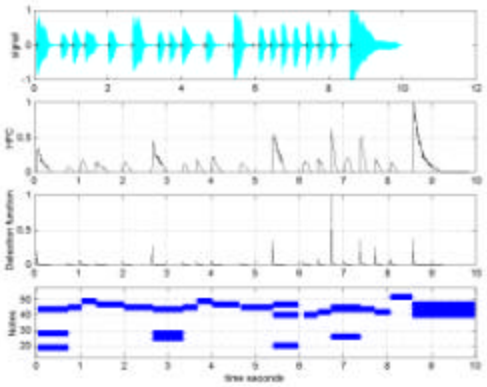

Figure 10: Example of automatic transcription of a simple polyphonic piano song of four measures.

3.4. Conclusions and next steps

The simple polyphonic system is achieving the automatic extraction of score parameters from simple polyphonic piano music, performed between the C4 and B6 notes, with up to three notes played at the same time and without the octave interval included. This is less general than the purpose defined on the introduction showing the necessity of some changes in the current system

First, to manage the octave detection problem, new knowledge sources should be added to the blackboard architecture based on the same principle implemented for the octave-error detection routine, but with the possibility of allowing the presence of an octave interval when it is truly present in the

input. To achieve that, more musical knowledge is necessary in the system.

The architecture of the blackboard will be modified, incorporating dynamic structures to handle different sized hypotheses, in this case, chords of more than three notes. Also, the training space of the network has to be expanded, contemplating the recognition of bigger chords and extending to all the octaves of the piano. As is showed in [6] the system is able to manage monophonic riffs of woodwinds and brass instruments, however, to define the next steps towards the handling of different instruments, more extensive testing has to be performed.

As a first approach, the results depicted here are very encouraging showing that further development of these ideas could be the way for more robust and general results.

4. References

[1] Eric Scheirer. "Extracting expressive performance information from recorded music". Master's thesis, MIT, 1995.

[2] R.F Moore. "Elements of Computer Music". Prentice Hall, Englewood Cliffs, New Jersey, 1990.

[3] Eric D. Scheirer, "*The MPEG-4 Structured Audio Standard*", IEEE ICASSP Proc., 1998.

[4] Keith Martin. "A Blackboard system for Automatic Transcription of Simple polyphonic Music". MIT Media Lab, Technical Report # 385, 1995.

[5] R.S Engelmore and A.J. Morgan. "Blackboard Systems". Addison-Wesley publishing, 1988.

[6] Bello J.P., Monti and Sandler,"*An implementation of automatic transcription of monophonic music with a Blackboard system*", Proc. of the ISSC, June 2000.

[7] Bregman A., "*Auditory Scene Analysis*", MIT Press, 1990.

[8] MIDI Manufacturers Association. "*The Complete MIDI 1.0 Detailed Specification*", 1996.

[9] Slaney M. "*Auditory Toolbox for Matlab*" available at URL http://www.interval.com/papers/1998-010/

[10] Brown , "*Musical frequency tracking using the methods of Conventional and Narrowed Autocorrelation*" J.A.S.A. , 1991.

[11] James H. McClellan, Ronald Schafer and Mark Yoder. "DSP First: A Multimedia Approach". Prentice Hall, USA. 1998

[12] Wakefield G.H., "*Time-frequency characteristic of violin vibrato: modal distibution analysis and synthesis*", JASA, Jan-Feb 2000.

[13] Bendor D, Sandler M., "*Time domain extraction of Vibrato from monophonic instruments*", to be published in Music IR 2000 Conference, October 2000.

[14] Martin K., "*Sound-Source recognition*", PhD thesis, MIT, ftp://sound.media.mit.edu/pub/Papers/kdm-phdthesis.pdf, 1999.

[15] Ellis D, "*Hierarchic models of sound for separation and restoration*",Proc. IEEE Mohonk Workshop, 1993.

[16] Csound web page URL: http://music.dartmouth.edu/ /~dupras /wCsound/ csoundpage.html.

[17] WAV2MIDI, URL: http://www.audiowork.com.

[18] Daniel Ellis. "Prediction-driven computational auditory scene analysis". PhD Thesis, MIT, June 1996.

[19] E.R.S Pearson. " The Multiresolution Fourier Transform and its Application to the Analysis of Polyphonic Music". PhD Thesis, Warwick University, 1991.

[20] Keith Martin. "Automatic Transcription of Simple polyphonic Music: Robust Front End Processing". MIT Media Lab, Technical Report # 399, December 1996.

[21] Daniel Ellis. "Mid-level Representation for computational auditory scene analysis". In Proc. Of the Computational Auditory Scene Analysis Workshop; 1995 International Joint Conference on Artificial intelligence, Montreal, Canada, August 1995.

[22] Anssi Klapuri. "Automatic Transcription of Music". MSc Thesis, Tampere University of Technology, 1998.

[23] Malcolm Slaney. "A critique of pure audition". In Proc. Of the Computational Auditory Scene Analysis Workshop, Montreal, Canada, August 1995.

[24] Stuttgart Neural Network Simulator. User Manual, version 4.1. University of Stuttgart, Institute for Parallel and Distributed High Performance Systems. Report No. 6/95.

[25] Tristan Jehan. "Music Signal Parameter Estimation". CNMAT Berkeley, USA. 1997.

[26] P. Masri and A. Bateman. "Improved Modelling of Attack Transient in Music Analysis-Resynthesis". University of Bristol. 1996.

[27] Randall Davis, Bruce Buchanan, and Edward Shortliffe. "Production Rules as a representation for a Knowledge-Based Consultation Program". Artificial Intelligence, 8:15-45, 1977.

[28] Barry Vercoe. "CSOUND A Manual for the Audio Processing System and Supporting Programs with Tutorials". Media Lab, M.I.T, Massachusetts, USA. 1992