SONIC: A System for Transcription of Piano Music

MATIJA MAROLT, MARKO PRIVOSNIK
Faculty of Computer and Information Science
University of Ljubljana
Trzaska 25, 1000 Ljubljana
SLOVENIA

Abstract: - This paper presents our system for transcription of polyphonic piano music – SONIC. SONIC takes an audio signal of a piano performance and tries to determine which notes were played by the performer, thus producing a list of notes and note onset times approximately matching the performance. SONIC is composed of three main parts: partial extractor, onset detector and note recognizer. We present these parts in more detail below, as well as some results obtained by using the system on recordings of piano music.

Key-Words: - Music transcription, polyphonic pitch recognition, neural networks, adaptive oscillators.

1 Introduction

Music transcription could be defined as an act of listening to a piece of music and writing down music notation for the piece. For each note its starting time, duration and loudness (dynamics) need to be determined.

Music transcription is a difficult cognitive task. It can be to some extent performed by trained humans, but it is a very difficult problem for current computer systems to solve. We could in a way compare it to speech recognition, where we convert an audio signal to phonemes, syllables and finally words; music transcription converts an audio signal into notes, their starting times, duration and loudness.

Many current transcription systems use some kind of a time-frequency transform and peak-picking algorithm to extract partial tracks from the audio signal and then use statistical methods to group these tracks into notes. Some systems (i.e. [6]) first calculate sound source models of instruments and then try to transcribe music performed by these same instruments. Others use no such models.

SONIC uses several types of neural networks to perform partial track extraction, onset detection and note recognition. The following sections present these tasks in more detail and also present some results obtained by using the system on recordings of piano music.

2 SONIC

SONIC attempts to correctly determine notes, their starting times, lengths and loudness in a polyphonic piano performance. It is composed of three main stages, depicted in figure 1 and described below.

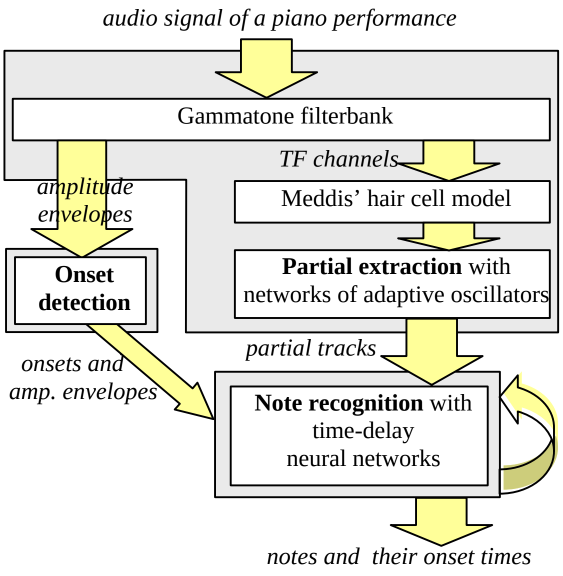
audio signal of a piano performance
Gammatone filterbank
TF channels
amplitude envelopes
Meddis' hair cell model
Onset detection
Partial extraction with networks of adaptive oscillators
partial tracks
onsets and amp. envelopes
Note recognition with time-delay neural networks
notes and their onset times

Fig. 1 Structure of the system

2.1 Partial extraction

The first stage of SONIC extracts partial tracks from the input audio signal. We chose a novel technique that uses a gammatone filterbank for splitting the signal into time-frequency tracks and adaptive oscillator networks for extraction of partials.

The audio signal is first fed through a commonly used auditory model, emulating the functionality of human cochlea. The model first uses a gammatone filterbank to split the signal into several frequency channels. The filterbank consists of an array of bandpass filters with near constant-Q bandwidth in middle and high frequencies. Center frequencies of filters are spaced logarithmically from 70 to 6000 Hz. We are using the filter implementation based on the model suggested by Patterson [5] and implemented by Slaney [7].

Subsequently, the output of each gammatone filter is processed by Meddis' model of hair cell transduction [4]. The hair cell model converts each gammatone filter output into a probabilistic representation of firing activity in the auditory nerve, incorporating well-known effects such as saturation and adaptation.

To successfully extract partial tracks from the resulting frequency channels, we first need to calculate a more accurate estimate of the dominating frequency in each channel. We could calculate more accurate frequency estimates with a correlogram and then extract partial tracks with some kind of a peak-picking algorithm, but we opted to use a different algorithm that implicitly produces partial tracks as its result.

Our model is based on networks of coupled adaptive oscillators. Adaptive oscillators are a class of oscillators that adapt their phase and frequency in response to external input. When a periodic signal is passed through an adaptive oscillator, it tries to synchronize to the signal by adjusting its phase and period to that of the input signal. By observing the frequency of a synced oscillator, we can make a more accurate estimate of the frequency of the driving input signal. Oscillators used in SONIC are a simplified version of the Large-Kolen adaptive oscillator model [1].

An oscillator has three variables that change with time: phase, frequency and output. It constantly oscillates through time according to its frequency and phase. If a periodic signal is presented to an oscillator, it tries to adjust its phase and frequency to match that of the input signal. The output value of an oscillator indicates how well it managed to sync to the input signal. If the oscillator fails to sync, it keeps oscillating, but its output value is low.

Each oscillator also has its so-called preferred frequency. This is the oscillator's initial frequency and an oscillator in our model is only allowed to sync to frequencies that are up to one semitone higher or lower than its preferred frequency. This prevents oscillators to drift away and sync to arbitrary input frequencies.

Each oscillator in SONIC gets its driving signal from one of the hair cell model's output channels. If a particular frequency (partial) appears in the audio signal, the output of the gammatone filter and consequently the hair cell model with center frequency near that frequency is quasi-periodic and the oscillator connected to the output syncs to that particular frequency. A synced oscillator therefore represents one of the partials in the audio signal. The oscillator's frequency is an approximation of the partial's frequency. Because oscillators are adaptive, they can also adapt to changes in frequency of their driving signal and stay in sync in cases of frequency modulations or beating in the input signal.

We took the model one step further and coupled harmonically related oscillators to form oscillator networks representing groups of partials. Each network consists of up to ten oscillators. The preferred frequency of the first oscillator in the network is tuned to frequency of one of the 88 piano notes (A0-C8). Preferred frequencies of other oscillators in the network are integer multiples the first oscillator's frequency (i.e. 220Hz, 440Hz, 660Hz, 880Hz,...). See also figure 2. Each oscillator in a network is internally coupled to all other oscillators to accelerate synchronization of oscillators in the network.

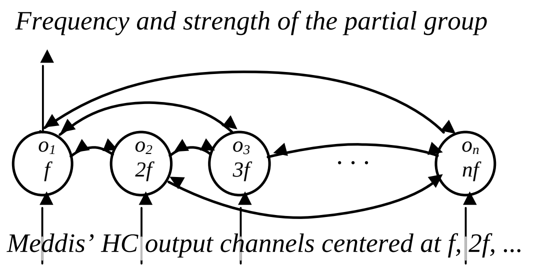
Frequency and strength of the partial group
o1 f
o2 2f
o3 3f
. . .
on nf
Meddis' HC output channels centered at f, 2f, ...

Fig. 2 A network of coupled oscillators estimating the strength of a group of partials at frequencies f, 2f, 3f,

The output of a network represents how successfully oscillators within the network synced to their input signals. The output is higher when more oscillators sync successfully (more partials are found). Such output therefore represents the strength (number of partials found) of a group of harmonically related partials. We are using 88 oscillator networks, each calculating the partial group strength of one piano note.

The final output of the partial extraction stage of SONIC (at any given time) is a set of 88 strengths of partial groups, one for each piano note. Because partial group strengths do not include amplitude information, we also extract amplitude envelopes from outputs of the gammatone filterbank and include them in the output. An example of such output can be seen in figure 3.

2.2 Onset Detection

The second stage of SONIC is the onset detector. Even though piano is an instrument with well-

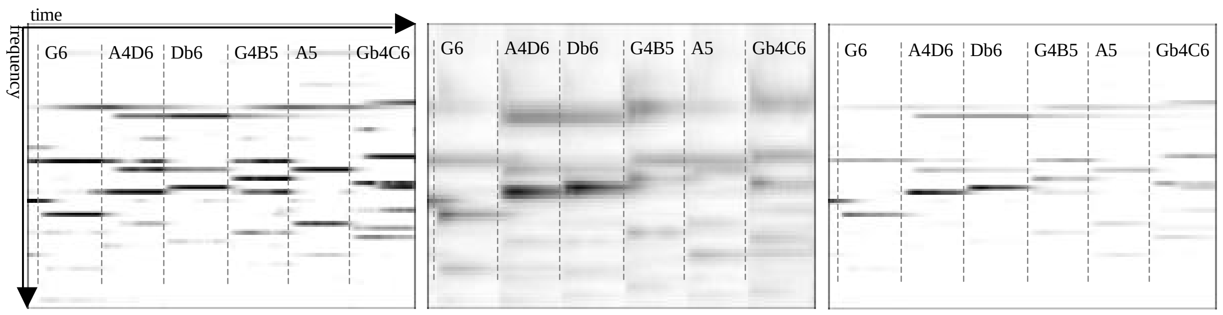

Fig. 3 Output of oscillator networks from transcription of Bach's French Suite No. 1 (BWV812): partial group strengths (left), amplitude envelopes (middle) and combined information (right).

pronounced onsets, a reliable onset detector is not easy to implement. After much experimentation, we chose to use an algorithm based on a combination of a network of integrate-and-fire neurons and a feed forward neural network.

SONIC first uses outputs of the gammatone filterbank and sums them into 22 overlapping bands, each covering half and octave. Two sets of amplitude envelopes are then calculated for each band, one using a 6 ms smoothing filter and the other using an 18 ms smoothing filter. The two sets are subtracted and the result passed through a network of integrate-and-fire neurons. Each neuron in the network integrates envelope differences in one of the frequency bands through time. When differences exceed a certain threshold in a period of time, the neuron fires and produces an indication of an onset within that channel. Neurons are connected to each other via excitatory connections that increase internal activations of all neurons, when one of the neurons in the network fires. This increases the prominence of weak onsets and brings onset activations across different bands closer together, thus improving onset detection.

Outputs of all neurons and amplitude differences are fed into a standard feed forward neural network, which has been trained previously on onset examples from various piano performances. The network has only one output, indicating whether an onset has occurred or not. The algorithm works quite well, most problems occur in very fast passages such as fast arpeggios (notes less than 50 ms apart) or thrills, where in our opinion a higher level cognitive mechanism would be needed to correctly extract all onsets.

2.3 Note Recognition

The part of the system that actually performs transcription consists of a set of 76 neural networks – one network for each piano note (we are currently ignoring the lowest octave from A0-Ab1). We tested several neural network models and finally settled for time-delay neural networks, which slightly outperformed standard feedforward networks in our tests.

The first half of inputs of each network consists of several time frames of outputs of oscillator networks, smoothed with an IIR filter with 25 ms time constant. The other half consists of smoothed amplitude envelopes (see left and center images in figure 3).

Each network is trained to recognize the occurrence of a single note in its input. Therefore, it only has one output: a high output value means that the target note is present in the input, a low output value means, that the note is not present.

Networks have been trained on a large database of piano pieces and chords. To obtain a set of segmented piano pieces that could be used for network training, we took a collection of over 100 midi files of piano songs, which we rendered with over twenty different piano samples obtained from commercially available piano sample CD-ROMs. We chose songs of various styles including classical from several periods, ragtime, jazz, blues and pop. Because very low and high notes were not frequent enough in the chosen pieces, we complemented the song set with a set of chords, ranging in polyphony from one to six and rendered with the same piano samples as songs.

The trained networks find a large percentage of notes, but produce quite a lot of additional notes, not present in the input, most of them octave errors. We therefore added a feedback mechanism, providing inhibition from the output of the note recognition stage to its input. When a network finds a note, this triggers inhibition that inhibits the found note's partials in the input of the recognition stage. The feedback decreases the number of octave errors made by note recognition networks, but also has a small negative effect on the number of correctly found notes.

2.4 Estimation of note lengths and dynamics

The note recognition stage yields a list of notes and their starting times. To produce the desired output, a midi file of the piano performance, note lengths and strengths also need to be determined. SONIC estimates the length of a note from the duration of positive activations of the note's neural network. The strength of a note is approximately calculated from the amplitude envelope at onset time.

3 Results

We tested the system by transcribing several solo piano performances. To make evaluation easier, we obtained the performances by rendering several midi files with piano samples different from those used to train the note recognition networks. The pieces ranged from very simple pieces, such as Bach's Two-part Inventions to more complex ones, such as excerpts from Tchaikovsky's Nutcracker Suite.

Transcription results obtained on some of the pieces are given in table 1. Results are given for transcriptions of five pieces: J.S. Bach's Contrapunctus 12 from Art of the Fugue, Partita No. 1 (BWV825), French Suite No. 1 (BWV812), Tchaikovsky's Nutcracker Suite Miniature Overture and Waltz of the Flowers. The second and third columns of table 1 represent the average and maximum polyphony of transcribed pieces. The fourth column (notes found) represents the percentage of notes in each piece that were correctly transcribed. The fifth column (extra notes) represents the number of additional notes that were found, but were not present in the input.

piano piece	*avg. poly*	*max. poly*	*notes found*	*extra notes*
Contrap. 12	1.8	5	95%	13%
Partita No. 1	2.6	6	94%	15%
French Suite	3	6	91%	14%
Nutcr. ovr.	3.1	6	90%	15%
Nutcr. waltz	5	15	81%	25%

Table 1 Transcription results

Most of the errors (either missing notes or extra notes) - over 50% - are octave or similar errors. The other most common source of errors are very short notes (less than 100 ms) or notes played very quickly one after another.

We also tested the performance of SONIC on several hand segmented commercial recordings of classical piano performances. As expected, results obtained on real recordings are not as good as those on rendered midi files, but are overall not bad. When analyzing Bach's Prelude from English suite no. 5 (BWV810) performed by Murray Perahia (Sony Classical SK 60277), SONIC correctly transcribed over 90 percent of all notes and produced 15 percent of extra notes with half of all the errors being octave errors. The performance decreases when polyphony increases; in Schumman's Traumerei performed by Cyprien Katsaris (TELDEC 75863), SONIC correctly found 78 percent of all notes and produced around 20 percent of extra notes; half of all errors were again octave errors.

4 Summary and Future Work

SONIC performs quite well on synthesized as well as real performances. Further work will be directed towards reducing the number of errors made by the system, especially the number of extra notes found. Most extra notes currently fall into one of the three categories; octave errors, repeated notes and misplaced notes. We are planning to add an extra postprocessing stage to the system, which would handle such errors by reexamining the found notes, detected onsets, amplitude envelope changes and some other features, such as common partial onsets. This stage would then remove some of the found notes or reassign them to new onset times.

References:

[1] E.W. Large, J.F. Kolen, Resonance and the perception of musical meter, *Connection Science*, 6(1), 1994.

[2] M. Marolt, Adaptive oscillator networks for partial tracking and piano music transcription, *Proceedings of the 2000 International Computer Music Conference*, Berlin, Germany, 2000.

[3] K.D. Martin, *Toward a Machine Listener: Recognizing Sound Sources*, Ph.D. thesis, MIT Department of Electrical Engineering, 1999.

[4] R. Meddis, Simulation of mechanical to neural transduction in the auditory receptor, *Journal of the Acoustical Society of America*, vol.79, no.3, p. 702-711, March 1986.

[5] R.D. Patterson, et. al., Complex sounds and auditory images, *Auditory Physiology and Perception*, Y. Cazals, L. Demany, K. Horner, Pergamon (Eds.), Oxford, 1992.

[6] L. Rossi, *Identification de Sons Polyphoniques de Piano*, Ph.D. Thesis, University of Corsica, France, 1998.

[7] M. Slaney, An Efficient Implementation of the Patterson-Holdsworth Auditory Filter Bank, *Apple Computer Technical Report* #35, 1993.