Real-time temporal segmentation of note objects in music signals

Paul Brossier*, Juan Pablo Bello and Mark D. Plumbley
Centre for Digital Music, Queen Mary University of London
{paul.brossier,juan.bello-correa,mark.plumbley}@elec.qmul.ac.uk

Abstract

Segmenting note objects in a real time context is useful for live performances, audio broadcasting, or object-based coding. This temporal segmentation relies upon the correct detection of onsets and offsets of musical notes, an area of much research over recent years. However the low-latency requirements of real-time systems impose new, tight constraints on this process. In this paper, we present a system for the segmentation of note objects with very short delays, using recent developments in onset detection, specially modified to work in a real-time context. A portable and open C implementation is presented.

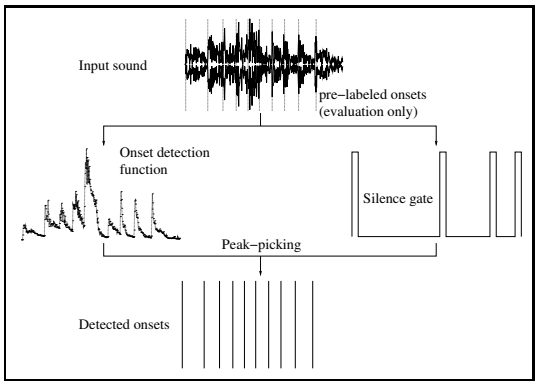

Figure 1: Overview of the segmentation process

1 Introduction

1.1 Background and motivations

The decomposition and processing of audio signals into sound objects are emerging fields in music signal processing. As well as allowing analysis of the content of an audio signal, it is in tune with accepted views on the human hearing process (Bregman 1990), and is particularly relevant to music, where musicians and musicologists have long proposed models based on musical objects (Schaeffer 1966). Sound-object taxonomies are at the core of novel research in music analysis (Ellis 1996) and frameworks have been recently proposed for the real-time transmission of object as audio content (Amatrian and Herrera 2002).

While many music-oriented applications require real-time functionality, little has yet been done to address the issue of real-time extraction of music objects, at least at levels higher than the composition of sinusoids. Note that real-time implementation is not only concerned with speeding up existing offline algorithms, but also with dealing with the constraints imposed by operating on a continuous, unknown and unpredictable stream of audio data. In (Brossier, Sandler, and Plumbley 2003), we presented a framework for the object-based construction of a spectral-model of a musical instrument. In the current paper, we concentrate on the temporal aspects of this process, investigating methods for the segmentation of note objects in real-time.

*PB is supported by the Department of Electronic Engineering at Queen Mary University of London, and by EPSRC grant GR/54620.

1.2 Temporal definition of note objects

In order to segment note objects, we need to identify the boundaries of a musical note, namely the exact times when the note starts – an *onset* – and finishes – an *offset*. These boundaries can be easily identified on the temporal envelope of an isolated musical note, that can be roughly characterised by the well-known ADSR (Attack Decay Sustain Release) linear approximation. However, a correct characterisation of onsets and offsets is not trivial, and depends on the notion of *transients* – transitional zones of short duration characterised by the non-stationarity of the signal spectral content.

Algorithms intended for the detection of onsets and offsets rely on observing those transients, a complex task not only because most notes are not present in isolation, but also because the nature of these transients changes from sound to sound – burst of energy across the spectrum for percussive sounds, large variation of the harmonic content for tonal or voiced sounds. This emphasises the difficulty of constructing a unique detection function that quantifies all relevant obser-

vations.

1.3 Overview of this paper

Fig. 1 gives an overview of the process of note object segmentation as implemented in this paper. First, we reduce the audio signal to an onset detection function at a lower sampling rate. Then, we perform temporal peak-picking on the detection function to obtain a sequence of onset times. These are combined with the output of a silence detector to produce the onset/offset pairs that define the boundaries of our note objects.

This paper is organised as follows: in Section 2 we explain a number of different techniques for the generation of onset detection functions, the temporal peak-picking and the silence detection and discuss their implementation in real-time; Sec. 3 discusses the details of our software library and presents quantitative results of the integration of the different parts of the system; our conclusions are presented in Sec. 4.

2 Techniques and Implementation

2.1 Onset detection functions

For a signal x at time n, let us define $X[n]$ as its Short Time Fourier Transform (STFT), calculated using the phase vocoder. $X_k[n]$, the value of the k^{th} bin of $X[n]$, can be expressed in its polar form as $|X_k[n]|e^{j\phi_k[n]}$ where $|X_k[n]|$ is the bin's spectral magnitude, and $\phi_k[n]$ its phase.

In (Masri 1996), a *High Frequency Content* (HFC) function is constructed by summing the linearly-weighted values of the spectral magnitudes, such as:

$$D_H[n] = \sum_{k=0}^{N} k|X_k[n]| \tag{1}$$

This operation emphasises the changes that occur in the higher part of the spectrum, especially the burst-like broadband noise, usually associated with percussive onsets, that is successfully characterised. However, the function is less successful at identifying non-percussive onsets – legato phrases, bowed strings, flute.

Other methods, reviewed in (Bello, Duxbury, Davies, and Sandler 2004), attempt to compensate for the shortcomings of the HFC by also measuring the changes on the harmonic content of the signal. One of such methods, known as the *spectral difference*, calculates a detection function based on the difference between the spectral magnitudes of two STFT frames:

$$D_s[n] = \sum_{k=0}^{N} |X_k[n]| - |X_k[n-1]|\,. \tag{2}$$

Alternatively, a function that measures the temporal instability of phase can be constructed by quantifying the *phase deviation* in each bin as:

$$\hat{\phi}_k[n] = \text{princarg}\left(\frac{\partial^2 \phi_k[n]}{\partial n^2}\right) \tag{3}$$

where princarg maps the phase to the $[-\pi, \pi]$ range. A useful onset detection function is generated as:

$$D_\phi[n] = \frac{1}{N}\sum_{k=0}^{N} |\hat{\phi}_k[n]| \tag{4}$$

Both approaches can be then combined in the *complex-domain* to generate a target STFT value $\hat{X_k}[n] = |X_k[n]|e^{j\hat{\phi}_k[n]}$, where $\hat{\phi}_k$ is the phase deviation function defined in Eq. 3. Then by measuring the complex-domain Euclidean distance between target and observed STFT we obtain:

$$D_C[n] = \frac{1}{N}\sum_{k=0}^{N} \left\| \hat{X}_k[n] - X_k[n] \right\|^2 \tag{5}$$

This function successfully quantifies percussive and tonal onsets.

For our experiments, we have implemented the four detection functions previously mentioned. Their offline implementations have proven to give good results on a variety of CD recordings, including percussive, purely harmonic signals and complex mixtures – pop and jazz recordings.

2.2 Temporal peak picking of note onsets

To obtain sequences of onset times, we need to process these detection functions through a temporal peak-picking algorithm. A number of peak-picking techniques have been proposed in (Kauppinen 2002). Intuitively peak-picking is reduced to selecting local maxima above a certain threshold value. However, in order to successfully perform this operation in a varied set of detection functions – and on a wide variety of signals – a number of processes are required.

Usual processes include the normalisation, DC-removal and low-pass filtering of the original function. This is done to maximise the success of the thresholding operation, by mapping functions to a limited range of values and by reducing noisiness in their profile that may result in spurious detections.

Also, dynamic thresholding is used to compensate for pronounced amplitude changes in the function profile. In this implementation we favour the use of the weighted median of a section of the detection function centered around the candidate frame:

$$\delta_t[n] = \lambda \cdot \text{median}(D[n_m]) + \delta \tag{6}$$

with $n_m \in [m-a, m+b]$ where the section $D[n_m]$ contains a spectral frames before m and b after. The scaling factor λ and the fine-tuning threshold δ are predefined parameters.

However, real-time implementation imposes more severe temporal constraints than offline implementations. Offline, the normalisation and DC-removal processes use information from a large time segment both before and after the current frame, allowing the use of fixed parameters for thresholding. In real-time we can only approximate this by using a long sliding window – thus causing long delays. We therefore propose an alternative thresholding operation using a small sliding window:

$$\delta_t[n] = \lambda \cdot \text{median}(D[n_m]) + \alpha\langle D[n_m]\rangle \qquad (7)$$

where α is a positive weighting factor and $\langle D[n_m]\rangle$ is the mean of $D[n]$ over the same window of spectral frames n_m. The introduction of the mean value attempts to replicate the effects of the normalisation and DC-removal processes, without the use of a long window, by using a dynamic value for the fine-tuning threshold. Onsets are then selected at local maxima of $D[n] - \delta_t[n]$. Experimental results confirm that, for small values of a and b, the modified threshold is robust to dynamic changes in the signal.

2.3 Silence gate

To reject false positives detected in areas of low energy, a simple envelope detector was built by measuring the mean energy of a frame of the signal. When loudness of a frame drops below a given threshold, typically -80 dB, it indicates the note offset. Onsets detected in the middle of a silent region are discarded. The threshold parameter can be adapted to the expected level of background noise.

3 Software library and results

3.1 Software library

We have implemented a small C library, providing device and file abstractions for both audio and MIDI, along with a set of processing units: phase vocoder, onset detection functions, peak-pickers. The library makes use of modern libraries such as FFTW and libsndfile. It also integrates with the Jack Audio Connection Kit (JACK). We can therefore reach low latency performances of modern Linux systems (MacMillan, Droettboom, and Fujinaga 2001).

A small application has been written to run segmentation experiments both real-time and offline. This ensures that the implementation is usable (decent overhead in real time mode) as well as correct (fast offline performance estimation). Using an overlap of 512 samples at $44100\ Hz$, the system can run on a standard desktop with a total latency of below 30ms. This breaks down into an $11\ ms$ delay caused by the forward analysis needed for the thresholding operation ($b = 1$), another $11\ ms$ introduced by the phase vocoder buffer and under $8\ ms$ for JACK and hardware latencies as tested by (MacMillan, Droettboom, and Fujinaga 2001). All the results above can be obtained using the JACK audio server at a mere 10% of processor usage on an AMD/Athlon 700 MHz. Offline testing for each function (and per set of peak-picking parameters) takes a few seconds of processing time per minute of audio.

3.2 Experimental results

We used a set of 23 monaural audio signals, sampled at 44100 Hz and representing a wide range of music styles and sound mixtures. In a previous step, the onset times of each of these files were carefully hand-labeled. The proportions of both correct and misplaced onsets were estimated by comparing detections against the database of 1066 hand-labeled onsets in the test set.

All detection functions have been peak-picked using a window of size $a = 5$ and $b = 1$ in Eq. 7, and are plotted for values of $\alpha \in [0.00, 1.15]$. The proportion of good detections against false positives after peak-picking is shown in Fig. 2. It can be seen that, in contrast to the offline case, the HFC outperforms the complex-domain onset detection. This is due to the effect that using short lengths of n_m has on smooth detection functions. Note that the complex-domain, phase-based and spectral difference approaches produce functions smoother than the HFC, as they operate on information from more than one frame.

Fig. 3 shows results when combining onset detection with the output of the silence gate. By using the silence detection to threshold onsets detected in low-energy conditions – where onsets are more likely to be produced by background noise – we obtain significant improvements on the detection accuracy. The simple gate reduces the average number of false positives by about 2% in all functions, while having a minimal effect on the percentage of correct detections. The gating level can be fine-tuned to recording conditions for better results.

Our segmentation is intended as a first step towards the real-time coding of note objects. Errors in the segmentation are inevitable (as the figures show), but we can attempt to minimise their effect on the final coded objects. In a musical scene, it is better to over-segment objects than to under-segment them, as the estimation of attributes such as pitch and loudness will be less affected – two notes with the same pitch is preferable to one note with an average pitch unrelated to those of the original notes. Moreover, we know from

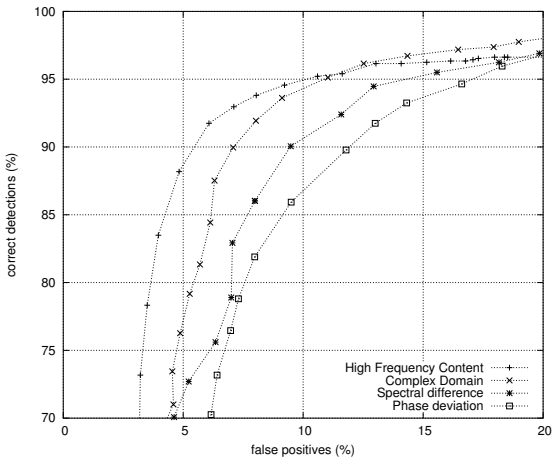

Figure 2: Correct detections against false positives for different α in Eq. 7 and for various detection functions.

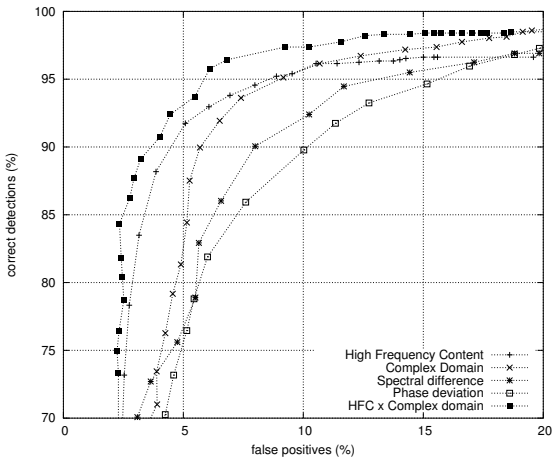

Figure 3: Correct detections against false positives as in Fig. 2 but using a silence gate.

the theory that while the HFC is well-suited for the detection of percussive onsets, spectral-difference methods, such as the complex-domain approach are well-suited for the detection of tonal – non-percussive – onsets. Therefore, to maximise the number of detections, we can combine these functions to produce a note segmentation algorithm tailored to the requirements of a real-time object-based coding system.

The design of our software library, allows for the easy implementation of various combinations. Fig. 3 (top curve) shows the benefit of multiplying the HFC and the complex-domain function. This combination consistently returns the best results for the whole set, increasing the overall reliability of the segmentation, and supporting the prevailing view that the different detection functions complement each other. This result is not surprising if we consider that both functions outperform the others, and that the spectral difference and the phase deviation can be seen as subsets of the complex-domain approach.

4 Conclusions and Future work

A complete system for real-time extraction of onsets from a live audio source has been described. Experiments confirm that combinations of the different detection functions along with a simple silence gate increase good detections and lower over-detections. The proposed peak-picking approach returns satisfactory results in a low-latency environment. Current development efforts are focused on the real-time estimation of attributes for the segmented note objects.

While the code would clearly benefit from some profiling, the design of the library allows the use of selected units from other plug-in systems such as Max, OSC, CLAM and LADSPA. Simple programs have already been written for use during live music performances – for instance to drive MIDI instruments.

References

Amatrian, X. and P. Herrera (2002). Transmitting audio content as sound objects. In *Proc. of AES22 Internat. Conf. on Virtual Synthetic and Entertainment*, Espoo, Finland, pp. 278–288.

Bello, J. P., C. Duxbury, M. Davies, and M. Sandler (2004, June). On the use of phase and energy for musical onset detection in the complex domain. *IEEE Signal Processings Letters*.

Bregman, A. S. (1990). *Auditory Scene Analysis: The Perceptual Organization of Sound*. Cambridge, MA: MIT Press.

Brossier, P., M. Sandler, and M. D. Plumbley (2003). Real time object based coding. In *Proc. of the AES 114th Convention*.

Ellis, D. P. W. (1996, June). *Prediction-Driven Computational Auditory Scene Analysis*. PhD dissertation, MIT, Departement of Electrical Engineering and Computer Science.

Kauppinen, I. (2002, July). Methods for detecting impulsive noise in speech and audio signals. In *Proc. of DSP-2002*.

MacMillan, K., M. Droettboom, and I. Fujinaga (2001). Audio latency measurements on desktop operating systems. In *Proc. of the International Computer Music Conference2001*.

Masri, P. (1996). *Computer modeling of Sound for Transformation and Synthesis of Musical Signal*. PhD dissertation, University of Bristol.

Schaeffer, P. (1966). *Traité des Objets Musicaux*. Paris, France: Éditions Du Seuil.