PULSE-DEPENDENT ANALYSES OF PERCUSSIVE MUSIC

FABIEN GOUYON, PERFECTO HERRERA, PEDRO CANO

IUA-Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain
fgouyon@iua.upf.es, pherrera@iua.upf.es, pcano@iua.upf.es

With the increase of digital audio dissemination, generated by the popularization of personal computers and worldwide low-latency networks, many entertaining applications can easily be imagined to rhythmic analyses of audio. We herein report on a method of automatic extraction of a rhythmic attribute from percussive music audio signals: the smallest rhythmic pulse, called the "tick". Evaluations of the proposed scheme yielded quite good results. We then discuss the relevance of use of the tick as the basic feature of rhythmic analyses.

INTRODUCTION

This paper deals with percussive music of constant tempo. More precisely, audio signals of restricted polyphonic complexity, containing few sets of timbres, e.g. few seconds-long mixes of acoustic bass drums, snare drums, hi-hats, toms and cymbals. In the following, these signals will be referred to as *drum tracks*.[1]

Regarding these signals, some research topics that can be thought of interest are extraction of specific rhythmic attributes (e.g. the Beat [1]), automatic description [2], analysis of the rhythmic structures ([3], [2]) and analysis of the internal structures of sounds for synthesis purposes [4]. Our long-term objectives are to define a generic rhythm description scheme that would be compatible to the MPEG7 standard and to design retrieval and transformation tools for percussive musical signals, anchored in the musical contents of the audio. Thus, our interests will eventually embrace all the aforementioned research topics. We here focus solely on the two first issues.

Our precise goal is to extract the smallest rhythmic pulse and to investigate its relevance as a segmentation step for subsequent analyses. The smallest pulse is herein called the *tick*. This concept is referred to as the "*tatum*" by Bilmes in [3] and the *"attack-point"* by Schloss in [2]. It should be noted that, in contrast to the Beat, automatic extraction of this rhythmic attribute from audio is a topic that did not receive much attention. The sole algorithm we are aware of is quite recent [5]. We first present our rationale regarding the segmentation of drum track audio signals. We then propose a segmentation algorithm and detail evaluations of it. Subsequently, we discuss two intermediate applications of it: the characterization of percussive events in drum tracks and Beat extraction. Eventually, future lines of work are envisaged.

[1] It should be noted that we refer to a "track" as a mixture of several percussive instruments, and not a single percussive instrument sound file.

1 RATIONALE

Why would it make sense to start an analysis of drum tracks by segmenting them according to a pulse size? When seeking a meaningful representation of audio signals, one must address the issues of characterization and segmentation (i.e. the "what is when?" issue). These concepts are tightly linked, and it is never really clear which should come first. Indeed, segmenting temporally a signal in meaningful elements is obviously easier if we know what it's made of, and characterizing an event entails that it has boundaries. How could we meaningfully segment a signal before it has been categorized, or categorize it before it has been segmented? In this "chicken and egg" issue, focusing first on one task or the other implies explicit or implicit use of heuristics regarding the signal.

The objective in segmenting a signal is to provide a set of relevant boundaries. The "slices" of signal must isolate coherent events to the best. In our case, we focus on musical signals of constant tempi, thus we assume that there always exists a relevant segmentation grid in regularly spaced indexes that is inherent to the performance and corresponds to the smallest metrical pulse. The signals are solely made up of percussive instruments, onsets are expected to occur approximately on the tick grid indexes, thus if the grid gap and starting index are correctly determined, the resulting segments isolate coherent parts of percussive instrument signals around their indexes of occurrence in a scope that is determined by the smallest metrical pulse (typically 150 ms). We assume that "new events" occur in the signal when *energy transients* are present. Unlike in the general polyphonic multitimbral case, it seems acceptable to state that occurrences of onsets in percussive music are linked to abrupt changes in the

single energy parameter.[2] Therefore, we can focus on a method of detection of transients inspired by [6]. Transient times can be used to determine the tick and to segment drum tracks respecting to a regular frame grid that is rhythmically relevant and isolate significant parts of percussive events. This segmentation can serve as a first step towards subsequent analyses.

2 TICK EXTRACTION – SEGMENTATION ALGORITHM

The goal is to extract the smallest pulse implied in a performance on a set of drums: the tick. Following Bilmes' rationale [3], we believe that important musical features are communicated at this metrical level. Here we are not addressing the notion of Beat –perceptively most prominent pulse, or tactus–. According to [3] (p.22), the tick seems better defined as "the time division that most highly coincides with all note onsets" than as the shortest interval between notes. The algorithm is based on the measure of the recurrence of the different inter-onset intervals (IOIs) present in the signal. We do not limit the definition of IOI to time differences between *successive* onsets, as is done in many cases, but rather opt for a definition that takes into account *all* pairs of onsets. (See Figure 1.) We assume that expressive music shows approximate integer timing ratios between metrical pulses. This entails that histograms of IOIs should show peaks at approximately harmonic positions.

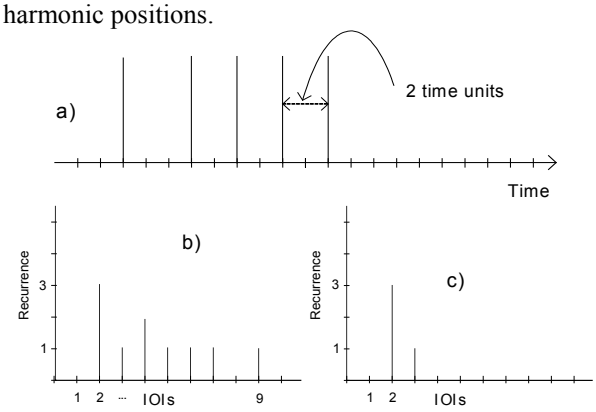

Figure 1: a) Example of onset sequence, b) and c) discrete IOI histograms. In b), IOIs are computed taking into account all pairs of onsets. In c), IOIs are computed taking into account solely successive onsets. Here, the tick –of 1 time unit– is not explicit. Rather, it is implied by the relationships between onset times.

If one extracts note-on timing data from quantized MIDI drum tracks, then the fact that the smallest pulse do contribute to the raising of peaks in the histogram at the exact positions of all of its multiples can be clarified visually, as in Figure 1 and 2.

[2] Therefore, in this paper, the terms "onsets" and "transients" will be used in the same manner.

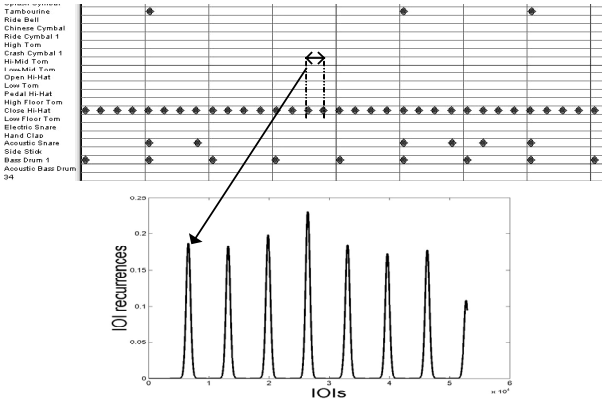

Figure 2: "Piano Roll" and IOI "smoothed" histogram of a MIDI drum track

Therefore, herein the tick is defined as *the gap of the IOI histogram harmonic series* –one could make an analogy with the notion of fundamental frequency–.

The segmentation algorithm is divided in the following steps:

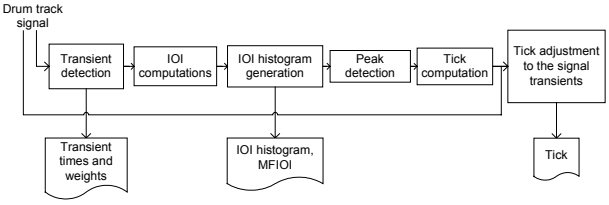

Figure 3: Flowchart of the tick extraction algorithm

- *Temporal envelope extraction*: The envelope is computed as the linear interpolation of maxima of the absolute value of successive short windows of the signal amplitude.
- *Short-time energy calculation*: The short-time energy is computed over overlapping short windows of the signal envelope.
- *Onset detection*: The frames that have an energy superior to the average of the energy of a fixed number of previous frames are considered as frames in which a transient occurs (see [6]). It is assumed that there are at least 68 ms in between 2 onsets. To each onset is associated a weight (i.e. a degree of confidence), corresponding to the number of after-onset successive frames whose energy is superior to the aforementioned averaged energy. The weight gives an indication on whether the onset should be considered as an actual one or an artifact of the onset detection scheme, which can be useful for subsequent uses of the onset list.
- *IOI computations*: As mentioned earlier, we take into account the time differences between all the onsets. A weight is associated to each time gap, corresponding to the smallest weight among the two onsets used for the gap computation.
- *IOI histogram generation*: Instead of considering

that a measured IOI participates solely to a single value in the histogram (as in Figure 1), it is considered that it should participate to a continuous region around its measured value. That way, IOIs that are slightly different participate to the raising of the same peak in the histogram. Indeed, it seems natural to consider that a drummer cannot perform the exact same time intervals, but rather performs this task with some degree of approximation. We generate a smoothed histogram (inspired by [2], p.90) by associating a Dirac delta function to each value of the IOIs and convoluting them with a Gaussian function which standard deviation was empirically adjusted (9 ms).

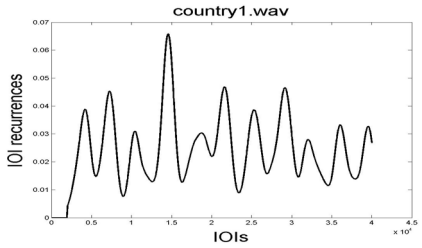

Figure 4: Acoustic data IOI histogram

- *Most frequent IOI (MFIOI) computation*: Given a set of IOIs, the MFIOI is computed as the maximum of the smoothed histogram of the IOIs (Schloss calls it the *"important duration"* in [2], p. 89).
- *IOI histogram peaks detection*: Peaks positions and heights are detected in the histogram with a three-points running window method: local maxima are detected at indexes whose values in the histogram are higher than that of their two direct neighbors.
- *Tick computation*: Here also, the IOI smoothed histogram is used. According to the previous definition of the tick, we seek the MFIOI integer divisor amongst the set of values [1,2,3,4,6,8,9] that best predicts the harmonicity of the IOI histogram. For each tick candidate (integer divisor of the MFIOI), a corresponding harmonic comb grid is generated and the two-way mismatch error function (TWM, see [7]) is computed between the grid elements and the histogram peaks: a first error function is computed, illustrating how well the grid elements explain the peaks, a second error function illustrate how well the peaks explain the grid elements. The TWM error function is a linear combination of these two functions –with equal weighting factors–. The tick-based grid that yields the TWM error function minimum is considered as corresponding to the good tick.
- *Tick adjustment*: The value of the tick is refined by achieving a matching in the time domain between several comb grids (the gaps of whose correspond to different tick candidates) and the onsets of the signal. For each tick candidate in a scope surrounding the first tick estimation, a comb grid is generated and the best "phase" for this tick grid is determined. That is, we seek the starting index of this grid that corresponds to the best matching between the signal onsets and the grid elements; this is done by computing a TWM error between
  - the signal onsets, and
  - a set of grids with fixed gap but different starting indexes

A function is built out of the values of the TWM error functions' minima of each tick candidate (i.e. each value corresponds to the best "phase" for a given tick grid). Eventually, the tick is chosen as the candidate that yields the minimum value of this function.

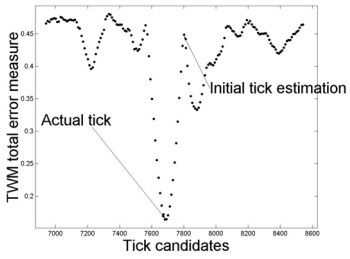

Figure 5: TWM error functions' minima. The local minimum corresponds to the actual tick

The reason to use this refinement of the value of the tick stands in the fact that the first aforementioned tick computation is not very accurate because of the smoothing of the histogram. The smoothing permits to agglomerate IOIs that should be considered jointly, even if not exactly equal; it necessarily entails a compromise between precision of the measure and amount of agglomeration of the IOIs. It must be stressed that the accuracy of the tick value should be considered as an important issue; indeed, even a very small error in this value does propagate in an additive manner in the generation of the tick grid.

The next figure shows an example of grids computed for both the MFIOI and the tick.

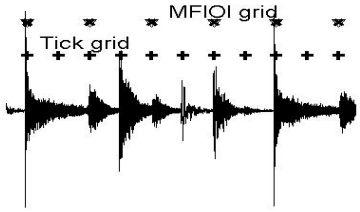

Figure 6: MFIOI grid and tick grid

3 EVALUATIONS AND RESULTS

3.1 Evaluation process #1

We developed an algorithm to generate random audio drum tracks, together with an exact score of these drum tracks. The resulting audio files don't really have a

musical meaning, however they constitute a useful audio material for automatic evaluation. Roughly, the algorithm first defines a pulse, the MFIOI, at each integer multiple position of which a percussive instrument is assigned in an empty audio signal. Instruments are randomly chosen from a percussive sound database. Then, a tick is defined as an integer divisor of the MFIOI, at each position of the tick grid is randomly assigned either an instrument or silence. To account for more realistic features, deviations of 1 to 10 ms from the exact tick grid positions are allowed and white noise is subsequently added to the signal.
The evaluation process is the following: (1) drum tracks and scores are generated, (2) the segmentation in ticks is achieved over the drum tracks and (3) the tick segmentation is automatically evaluated regarding the following criterion: the result is considered good if the computed tick indexes are the same as the ones given in the score of the track, with a possible deviation of ±1%. Three other categories are also considered: the ticks that are found twice smaller as they actually are (category 'halves'), those that are integer multiples (category 'multiples'), and eventually the rest, considered as the bad ones (category 'bad').
1000 5-seconds drum tracks have been generated following the previous algorithm, with a MFIOI of 500 ms (which corresponds to 120 BPM), four tick sizes being considered (250, 166, 124 and 83 ms). The results are: 77.3% of the computed ticks have been assigned to the category 'good', 0.7% have been assigned to the category 'halves', 10.4% have been assigned to the category 'multiples' and 11.6% of the tracks have been assigned to the category 'bad'.

3.2 Evaluation process #2

An analysis achieved over real audio drum tracks (that is, not generated automatically) has also been performed. The systematic evaluation is here more difficult as we don't have scores of the drum tracks that would provide the unambiguous knowledge of the tick. Here, the subjectivity of the listener enters obviously in the evaluation process, all the more if the number of excerpts to evaluate is important. Nonetheless, it is interesting to mention the following results: Over 57 drum tracks, ranging from 2 to 10 seconds, made up of different bass drums, snares, hi-hats, cymbals, toms, corresponding mainly to reggae, funk, hip-hop and rock styles, comparing subjectively extracted minimal pulses with tick gaps and starting indexes yielded by the algorithm, the determination of the tick was considered good in 86% of the cases, almost good (multiples or rationally-related values) in 7%, and bad in 7%.[3]

[3] A list of audio drum tracks used for this evaluation is available on the first author's web page.

4 ON THE USEFULNESS OF THE TICK

Bilmes (as detailed in [3]) found evidence for the fact that musical phenomenon (e.g. deviations) occurring within the scope of the smallest metrical pulse are very important for musical expressivity. We will here shortly discuss the relevance of use of the tick in two "classic" applications (though, still calling for algorithm improvements): (1) the automatic description of drum tracks and (2) the "smooth" sequencing or synchronization of musical signals (avoiding awkward temporal disparities). Eventually, we will comment the tick extraction in the context of polyphonic music.

4.1 Drum tracks percussive events characterization

Methods exist that automatically identify instrument tones by computers, a comprehensive review can be found e.g. in [8]. However, it is not clear yet how those methods can be applied to tones in their context –i.e. complex audio mixtures– without assuming a preliminary source separation, still not feasible. Nevertheless it is of great interest to tackle the issue of automatically describing audio files slightly more complex than single sounds or monotimbral excerpts, which is the case of drum tracks. Indeed, a simple look at musical magazines and web sites reveals many providers of such audio material.
The specific handling of drum tracks exist in commercial products, the state-of-the-art product being probably the software Recycle [9]. However, a functionality seeming of very first interest is absent of software packages: the identification of the tracks' percussive instruments, together with their occurrence instants. Such a functionality would open the way to many interesting processings, e.g. to automatically apply an effect on all the occurrences of a specific timbre.
In the task of classifying percussive sounds embedded in drum tracks, unavoidable steps are that of: (1) *segmenting* the drum tracks in coherent events, and (2) *choose a classification technique* to distribute these events among groups. It is obvious that we can't assume that a segmentation technique will provide us with isolated clean drum occurrences similar to those of a training set of percussive hits. A strategy to avoid this difficult issue is to use descriptors of signal frames rather than descriptors that would be relative to entire drum occurrences (as e.g. "log attack time", "temporal centroid of the tone", or "spectral centroid of the tone", the percussive perceptual timbre space reported in [10]). In this paper we shortly report on an off-line clustering method: for each excerpt, the frame size is set to its specific tick size and grouping behaviors are sought among frames. Here, no relation is supposed whatsoever between excerpts, nor between excerpts and the elements of a sound database. Setting the analysis frame size to the tick (typically 150 ms), we computed over

each frame different descriptors such as the spectral kurtosis, the temporal centroid or a measure of the temporal decay. The clustering algorithm chosen is a decision tree based on a hierarchical clustering scheme. More details on the algorithm and the features computation can be found in [11]. Also, different rationales regarding the classification of percussive sounds in complex mixtures can be found in [12] and [13].

The next figure illustrates a description of a drum track as generated by our algorithm. Here an 'a' corresponds to an occurrence of a kick, a 'b' of a snare, a 'd' of a hi-hat and a 'c' corresponds to "no new event". The second 'b' is the sole artifact of the example (it should be a 'c').

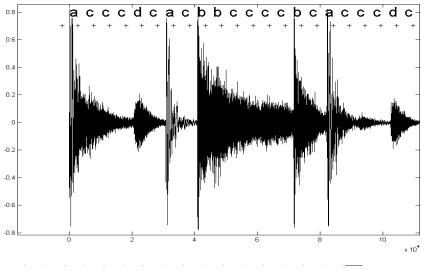

Figure 7: Computed sequence

To evaluate the goodness of fit of the characterization algorithm, we focused solely on drum tracks made up of non-simultaneous kicks, snares and hi-hats and generated 1000 of them with their exact scores. Drum tracks are segmented in ticks and symbolic sequences are generated subsequently with the clustering algorithm. These sequences are evaluated by comparison with the score sequences by a brute-force string-matching algorithm: For each possible permutation of a computed sequence (if N is the number of letters of the alphabet –the different elements constituting the sequences–, there are N! possible permutations) the percentage of elements that are not similar to those of the actual sequence is computed. Then, among the N! matching percentages computed for a given sequence, the best one is chosen to be the percentage of matching between the computed and actual sequences. By the non-supervised nature of our algorithm, the assignment of a letter to a frame is arbitrary, for instance, an 'a' doesn't have to correspond to the same instrument in different drum tracks. This is the reason for considering all the permutations of the computed sequences.

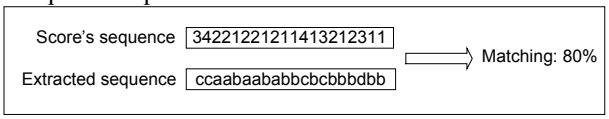

Figure 8: Example of matching evaluation between generated and extracted sequences

If we evaluate the sequences generated solely over the tracks that yield a good tick (that is, not considering 'multiples', 'halves' nor 'bad'), the best results showed 84.8% of matching between the structures of the scores sequences and the structures of the computed sequences. The evaluation over all the tracks (that is, 'good', 'halves', 'multiples' and 'bad') yields a 74.2% of matching between the structures of the scores sequences and the structures of the computed sequences. See [11] for more details.

4.2 Beat extraction

The automatic extraction of the perceived pulse rate in a musical flow is a research topic that has received much attention (see e.g. [1] for a review). The knowledge of the tempo and the beats positions would open the way to many applications as e.g. automatic "smooth" sequencing or synchronization of musical signals (see e.g. [14] or [15]), or concealment of transmission errors in streamed audio by beat-pattern based methods [16].

In the process of extracting the tick, the MFIOI is first computed. Naturally, one might wonder whether it corresponds to the Beat or not (e.g. Schloss qualified it as the *"important"* pulse in [2]). We performed a test over 80 drum tracks whose beats had previously been manually annotated. A refining of the MFIOI and the determination of its best phase were computed similarly as for the tick (see section 2). The results are that in 63.7% of the cases, the MFIOI corresponds to the Beat, 25% of the MFIOI correspond to integer multiples or divisors of the Beat, and 11.3% are not related with the Beat.

We believe that these results, specially the 25% of "almost good Beat", illustrate the fact that, as it is commonly thought, modeling Beat induction entails taking into account IOI data, but also dynamic, melodic, harmonic and timbral features of the signals. Seeking the Beat by applying signal processing techniques that extract such features could probably be improved by setting the size of the analysis frame to the smallest of the pulses that organize temporally musical signals.

4.3 Application to polyphonic music: first comments

An important issue that remains to be addressed is that of the handling of regular multitimbral polyphonic music as found in commercial CDs. Experiments concerning the tick extraction on excerpts of constant tempi were performed and gave encouraging results: a subjective evaluation reported that over 112 polyphonic excerpts –pertaining mainly to reggae, pop, funk, flamenco and Latin styles–, the algorithm tuned for drum tracks signals yielded 56% of correct ticks, 25% of multiples or rationally-related values and 19% of errors. Obviously, a proper systematic evaluation should be performed, but let us propose our first comments to this experiment: It seems obvious that the issue of accurately extracting onset times in multitimbral polyphonic audio calls for a different algorithm, not

focusing on the sole energy parameter. However, it seems that the general tick extraction scheme is quite robust to poor transient extraction. Indeed, as transients are solely used to generate a histogram, the tick extraction doesn't really call for the determination of *all* the transients, rather for an *accurate* determination of *some* instruments' onsets, would they correspond to the same instrument or not. Nevertheless, if we aim at increasing the success rate of the tick extraction in polyphonic audio, and also at addressing properly the intertwined issues of characterization and segmentation of such signals, we should certainly follow a different rationale regarding the transient detection.

5 CONCLUSIONS – FUTURE WORK

We have developed a technique to automatically associate to a given drum track a rhythmic attribute, the smallest rhythmic pulse: the tick. It is shown that this attribute is useful in the task of describing the timbral structures of such audio files. Future work concerns the improvement of a Beat extraction scheme based on the knowledge of the tick and the comparisons, over a large drum track database, of several pattern-matching techniques to improve the automatic characterization of drum track percussive events. Herein, we centered the debate on a physical property of sound; it would be interesting to pursue some investigations regarding the perceptual relevance of the tick, particularly regarding its variability with musical expertise. Also, we let for future work considering the usefulness of the tick for search-by-similarity issues.

ACKNOWLEDGMENTS

The authors would like to thank Güenter Geiger for his support in the porting of the code to the C++ Library for Audio and Music (CLAM) of the MTG. The work reported in this paper has been partially funded by the IST European project CUIDADO.

REFERENCES

[1] Scheirer E., *Tempo and beat analysis of acoustic musical signals*. Journal of the Acoustical Society of America *103,* (1998).

[2] Schloss A., *On the automatic transcription of percussive music - From acoustic signal to high-level analysis*. PhD Thesis Stanford Univ. (1985)

[3] Bilmes J., *Timing is of the Essence: Perceptual and Computational Techniques for Representing, Learning, and Reproducing Expressive Timing in Percussive Rhythm*. MS Thesis, MIT, (1993).

[4] Laroche, J. and Meillier, J.-L., *A simplified source/filter model for percussive sounds*. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, (1993).

[5] Seppänen J., *Tatum grid analysis of musical signals*. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (2001).

[6] Levine S., *Audio representations for data compression and compressed domain processing*. PhD Thesis, CCRMA Stanford University, (1998).

[7] Maher J. and Beauchamp J., *Fundamental frequency estimation of musical signals using a two-way mismatch procedure*. Journal of the Acoustical Society of America *95,* (1993).

[8] Herrera P., Amatriain X., Batlle E., and Serra X., *Towards Instrument Segmentation for Music Content Description: a Critical Review of Instrument Classification Techniques*. International Symposium on Music Information Retrieval, (2000).

[9] Propellerheads, *Recycle* http://www.propellerheads.se/products/recycle/

[10] Peeters G., Mc Adams S., and Herrera P., *Instrument sound description in the context of MPEG-7*. International Computer Music Conference, (2000).

[11] Gouyon F. and Herrera P., *Exploration of techniques for the automatic labelling of instruments embedded in audio drum tracks*. 1st MOSART Workshop on Current Research Directions in Computer Music, (2001).

[12] Herrera P., Yeterian A., and Gouyon F., *Automatic Classification of Drum Sounds*. (In preparation).

[13] Gouyon F., Pachet F., and Delerue O., *On the use of zero-crossing rate for an application of classification of percussive sounds*. Digital Audio Effects conference, (2000).

[14] Cliff D., *Hang the DJ: Automatic Sequencing and Seamless Mixing of Dance-Music Tracks*. Hewlett Packard technical report HPL-2000-104 (2000).

[15] Yamada Y., Kimura T, Funada T and Inoshita G., *An apparatus for detecting the number of beats*. US Patent [5,614,687], (1995).

[16] Wang Y., A *Beat-Pattern based Error Concealment Scheme for Music Delivery with Burst Packet Loss*. International Conference on Multimedia and Expo, (2001).