DRUMTRACK: BEAT INDUCTION FROM AN ACOUSTIC DRUM KIT WITH SYNCHRONISED SCHEDULING

Nick Collins
Centre for Music and Science
Faculty of Music, University of Cambridge,11 West Road, Cambridge, CB3 9DP, UK
http://www.cus.cam.ac.uk/∼nc272/

ABSTRACT

Technology is described to track the tempo and phase of a human drummer on an acoustic drum kit from analysis of the audio signal alone. The beat induction algorithm combines work by Laroche and Goto with original aspects, and is efficient for real-time operation. The inferred clock drives the scheduling of computer music which accompanies and processes the drummer. An algorithmic improvisation system entitled DrumTrack was built to exploit this tracking, efficient enough to run on a single laptop.

1. INTRODUCTION

Causal real-time beat induction from a musical audio signal is no easy task, and the subject of many recent papers[5, 4, 3, 1]. In the basic approach authors examine the energy signal using some form of exhaustive correlation search, whether by the use of comb filter resonators[5], an efficient cross correlation[4], or autocorrelation lags[1]. Whilst this may model a low-level neuronal mechanism, higher level knowledge about the signal is more rarely utilised. Masataka Goto in his beat induction work[3], however, has demonstrated some success in detecting certain features of (popular) music such as kick and snare patterns and chords and using these in rating the hypotheses of beat tracking agents. It would seem intuitively plausible that musicians make use of learnt stylistically relevant high-level features of music particularly in selecting the correct phase hypothesis for tracking. Recent psychological evidence has suggested that there may be both a low-level and a high-level beat induction facility consistent with these approaches[6].

The DrumTrack project described in this paper attempted to build a practical working system for the tracking of an acoustic drum kit, where the human player could exert control over the scheduling of computerised parts. This necessitated not just the tracking of tempo but the accurate determination of phase within a period, the arena of beat induction. In order to overcome limitations on the consistent determination of the phase found with correlation models alone, the author was drawn to Goto's ideas. The resulting system synthesises work by Laroche[4] and Goto[3] in a casual dynamic programming framework for beat induction (section 2). In order to drive the computer parts, a scheduler had to be built that could run from an external clock (section 3). Finally, an improvisation system is outlined (in section 4) which was premiered in a real concert setting.

2. BEAT INDUCTION ALGORITHM

A concert-proof causal real-time algorithm was required with accurate phase alignment. Whilst finding the correct tempo was relatively straight forward using a variety of beat induction models, and the efficient Laroche model provided a natural starting point [4], energy signal correlational search methods alone were found insufficient to consistently determine the correct phase. To overcome this problem, some higher level signal understanding adapted from work by Goto[3] was utilised to spot kick and snare drum patterns, and a heuristic was also introduced favouring cases where low frequency energy appears on the beat. This additional information was reconciled within a causal version of Laroche's dynamic programming framework, the drum pattern and low frequency information providing additional evidence to rank (period, phase) hypothesis pairs.

Figure 1 outlines the stages in the DrumTrack beat induction algorithm to be further detailed below.

2.1. Cross Correlation

Laroche provides a very efficient search procedure for (period, phase) hypotheses[4]. A memory holds an energy function of the last 3.4 seconds, which is calculated from an FFT of the audio signal input. This energy flux is searched by cross-correlation with impulse signals corresponding to a given (period, phase) pair, as illustrated in figure 2 for a quarternote impulse signal. Laroche suggests even sixteenth note spacing for 16 multiplications; it was found more robust in this project to use eighth notes (with weighting 1.0 for onbeats and 0.5 for off) to avoid any assumption of groove.

100 tempi are searched, from 90-190 bpm, with 20 phases tested per tempo. The highest scoring 10 tempi pass through to the dynamic programming stage, with the 2 best phases and their 2 antiphases, giving up to four phase hypotheses per tempo and thus 40 hypotheses in total out of the initial 2000. The rationale for always

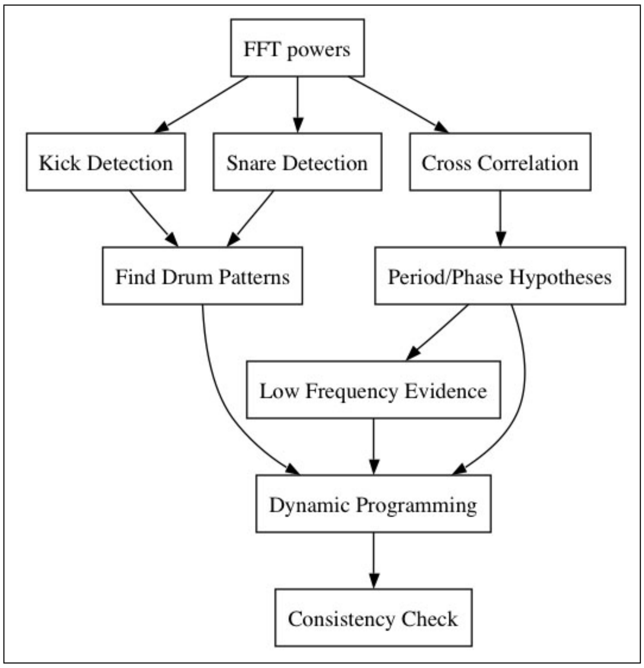

Figure 1. Overview of the beat induction algorithm

keeping the antiphases was that the pi-phase error was the most prevalent problem, and maintaining both hypotheses at this stage avoided such an error early in the assessment.

2.2. Detecting Drum Patterns

In a parallel step, the signal is searched for matches to an archetypal 4/4 drum pattern. This necessitates signal processing to detect kick and snare onsets, adapted from Goto's system[3, pp 162-3]; only differences are outlined here. A snare detection function is calculated as the product of values of the form $1 + x$ for each subband of 9 FFT components, rather than Goto's form x. This gives a much more continuous function than Goto's all or nothing system where the failure of any subband to be a noise component means a failure of snare detection. The bass drum detection is not calculated by Goto's more expensive histogram method but by using Goto's onset detection formula (equation (2), p161) on the three FFT bins above the zero bin. Sensible thresholds were found by examining the maxima and mean of the detection functions for real test signals.

Detected kick and snare signals are stored to a memory array of time resolution equal to the FFT hop size. This array can be searched for matches to a given drum pattern. Goto's publications do not give the full details of how he implements pattern matching for drum beats; he appears to use a beat hypothesis to establish a quantising grid for detected kick and snare onsets which are then matched against eight drum pattern templates (only two such templates are given in his papers). In this project, the choice was taken to search for matches without quantisation, though allowing some leeway on match location to allow for the detection latency and FFT resolution. The detection of a drum pattern would then provide evidence of the necessary period and phase of a winning hypothesis. Such a tactic demands a more exhaustive search; this could still be achieved relatively efficiently.

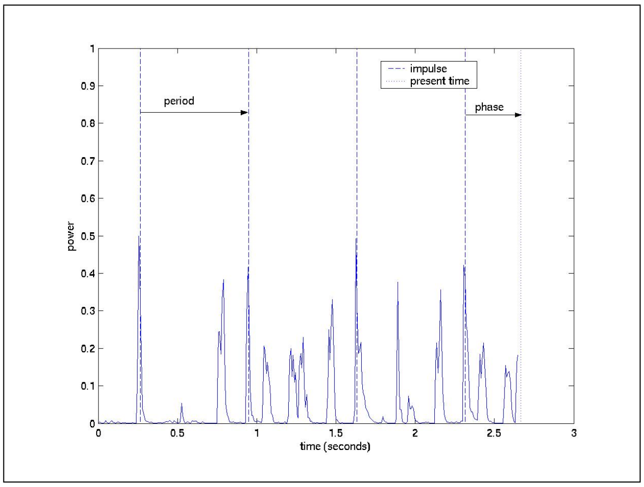

Figure 2. Cross correlation of an impulse signal representing a (period,phase) hypothesis with the source energy signal

The primary archetype is the classic 4/4 kick-snare-kick-snare on-beat alternating pattern. It is represented by weights such that the second kick is worth only 0.5 points whilst the other positions are all worth 1. A match requires a score of at least 1.75, thus disregarding single hits and the case of kicks on beat 1 and 3 which otherwise acted as a confound.

Figure 2.3 provides pseudocode for the search procedure. The reader is spared the modulo math to keep track of the circular onsets buffer and the cases that account for the type (kick or snare) of a starting onset. In the onsets memory the beginning of a bar (and hence a drum pattern) can begin in any position. The code is thus equipped to expect the archetype to appear in any of the four rotational forms.

2.3. Low Frequency Evidence

Given a (period, phase) hypothesis the proportion of on to off beat low frequency energy is assessed for the previous four beats located according to the hypothesis. The low frequency energy is calculated by summing the bottom five FFT bins (bin frequency $<=$172Hz for the specific FFT parameters in the implementation). To avoid inaccuracies in FFT time resolution a seven point average is taken around a given assessment frame position.

$$\text{basscost} = 1.0 - \text{scale factor} * \frac{\text{on-beats bass sum}}{\text{off-beats bass sum}} \quad (1)$$

2.4. Dynamic Programming Step

Various sources of evidence must be reconciled in the dynamic programming step. Laroche's original dynamic pro-

Figure 3. Pseudocode for drum pattern matching

```
now= current frame
for i= all starting onsets (where room for a later onset)
for j= all onsets later than i
        consider i as first beat,
        j as either second, third or fourth
        (The spacing must be plausible and there are
        various cases based on the type of the starting onset)
        Rate the archetype match such that period is diff(i,j),
        diff(i,j)/2 or diff(i,j)/3 respectively and phase is
        given by (now-i) mod(tempo)
        If rating best so far, store (period, phase) as best match
```

gramming scheme is not causal, so was adapted to calculate a step at a time. Programming step t proceeds by evaluating each of the 40 active hypotheses i with respect to the following equation, for each of the 40 previous hypotheses j from the last evaluation cycle.

$$\text{cost}_i(t) = \alpha\text{cost}_j(t-1) + \text{score}(i) + \text{trans}(i,j) + \text{evid}(i) \tag{2}$$

The Greek letters in these equations refer to weighting constants to be determined. In particular, α controls a leaky integrator on path costs from previous dynamic programming rounds. The score is the normalised score given by the cross correlation, and is assumed to have a constant of one; other constants are thus relative to this weight.

The transition cost is evaluated in a way similar to [4, p230]; tempo transitions above 6.3bpm have a fixed associated cost, and phase errors are scored by three times the difference of predicted beat times (giving a maximum cost 3*0.33=1 for the tempo range considered).

$$\text{trans}(i,j) = \beta\text{phaseerror}(i,j) + \gamma\text{tempochange}(i,j) \tag{3}$$

Finally, the evidence is incorporated:

$$\text{evid}(i) = \delta\text{basscost}(i) + \epsilon\text{pattern}(i) \tag{4}$$

A formula for the basscost was given in (1). The pattern score derives from a further transition cost (equation 3) but here from the current hypothesis to the period and phase suggested by the best pattern match (section 2.2). Optimal values of the constants were gained during trials and by feedback from comparative evaluation of performance with reference systems as detailed below.

2.5. Consistency check

The winning path (that with minimum cost) from the dynamic programming stage is not immediately accepted. A consistency condition requires a winning hypothesis to be selected over two iterations of the cost assessment. Because the phase is constantly updating, a further phase transition calculation takes account of the time elapsed between dynamic programming steps. Demanding two consistent results in a row is a compromise between the need to be sure of a hypothesis before making any phase and period shift, and the need to respond relatively quickly to the human drummer who may choose to change their beat at any time.

2.6. Implementation as a SuperCollider UGen

The beat induction algorithm is implemented as a SuperCollider UGen in C. The UGen assumes 44100 Hz sampling rate and 16 bit resolution, calculating a 1024 point FFT with 512 sample overlap (frame rate 86.1328 per sec). Dynamic programming rounds occur every 24 frames (0.28 seconds). The various computational loads are spread (amortised) amongst 64 sample control periods. The UGen was sufficiently efficient to run at 6% average CPU cost without any noticeable peaks on a 400MHz G4 Powerbook.

2.7. Evaluation

The influence of different weighting constants on beat tracking performance was assessed with respect to two other models from the literature. This gave feedback for (bisection) searches for appropriate parameter values for the model. A drum kit source example of 1 minute duration was prepared, combining a number of tempi and grooves in roughly ten second segments with abrupt phase jumps between them. 133 hand marked onsets constituted an ideal solution and a strict tolerance for matches was taken of 50mS.

A measure of longest continuous tracked segment as used in some beat tracking evaluations[1, 3] was inappropriate; even a human response would be disrupted by abrupt phase and tempo shifts, and this is exactly the sort of musical situation the algorithm would have to respond to in performance.

Two alternative evaluative criteria were explored. An evaluation formula from[2] gave the first, combining matches m, false positives F^+ and false negatives F^-.

$$\text{eval1} = \frac{m}{m + F^+ + F^-} \tag{5}$$

The second gave a score rating by iterating over the list of algorithm generated beats. A false negative scored -1, and a match scored 0 if isolated, or 1 if the following beat also matched to the corresponding next hand marked beat. This measure rewarded cumulative matches, but did not overly penalise drop out at phase jumps.

The reference systems were the Scheirer model[5], and a beat induction model kindly provided by Matthew Davies[1]. Neither of these is a practical real-time system, and both run around 2.5 times slower than real-time on the same computer used for testing the UGen's efficiency above. The Davies model provided a benchmark of the state of the art that a realtime system was not expected to surpass.

Table 1 lists results. It is readily seen that the best parameter settings combine the evidence and cross correlation scores but disregard the leaky integration dynamic programming. The consistency checks (section 2.5) are better at adapting than the dynamic programming controls over path consistency which showed too much lag. The

Table 1. Comparison of reference systems and DrumTrack systems with the given parameters [α,β,γ,δ,ϵ]

algorithm	eval1	m	$F^+ + F^-$	eval2
$Davies$	70.89	112	46	77=102-25
$[0, 0, 0, 0.025, 0.1]$	58.08	97	70	47=81-34
$[0.3, 0, 0.1, 0, 0.1]$	48.6	87	92	27=73-46
$Scheirer$	46.57	68	78	11=24-13
$[0, 0, 0, 1.0, 0]$	43.78	81	104	11= 63-52
$[0, 0, 0, 0, 0]$	36.979	71	121	-9=50-59

drum pattern matching was definitely required for good performance however, as the cross correlation alone performed worse than the Scheirer model. Performance was not as good as the Davies non-realtime model, due to various factors: the algorithm needed 3.4 seconds to initialise and did not have a prior tempo distribution. This was to equally favour faster tempi, a compositional choice which allowed the drummer in practise to work with such rates.

3. SCHEDULING FROM A BEAT INDUCTION CLOCK

For more complex rhythmic events which were to respect the tracked beat, SuperCollider code described various algorithmic agents. Some of these were algorithmic audio cutters which could utilise the anticipated beat positions to pre-schedule recording of the input source. Others were live synthesised machine drummers which would be locked to the external induced clock. Both forms of activity were supported by a scheduling system that prearranged any events due to play during the next beat on receipt of a beat tick from the UGen. In fact, due to the unknown network latency between the SuperCollider language application and the Server synthesiser, the prescheduling also took into account a minimum latency argument (of the order of 40 mS) so as to timestamp OSC messages and guarantee synchronisation.

Scheduling was achieved by an inner while loop which requested future events from client agents in small blocks until enough were available to fill the time to be prescheduled. Because the agents themselves often had to calculate more than needed at a given time (perhaps because they worked out their material by measures), the scheduler provided a queue to store any spare future events. All agents were compatible with this evaluation on demand set-up via a class hierarchy. Beat based scheduling covered long-term events but locations were converted to seconds for the next beat (where the tempo is known); this short-term time based scheduling queue could always be cancelled early on receipt of an unexpected premature beat signal from the tracker (perhaps corresponding to an accelerando or phase jump).

4. DRUMTRACK, AN IMPROVISATION SYSTEM FOR HUMAN AND ARTIFICIAL MUSICIANS

UGen development was targeted towards the production of a live improvisation system combining a human drummer with algorithmic drummers, called DrumTrack. Certain decisions taken in the programming of the beat induction algorithm betray compositional decisions, such as the 90-190 tempo range without mid biased tempo prior that supports drum and bass style 160bpm+ drumming. Assumptions of 4/4 eased the pattern matching task, and the handling characteristics at phase transitions were revised to fit feedback from the performer .

Ultimately, aesthetic considerations were considered alongside engineering ones in balancing the final tracker for performance. However, evaluation demonstrated competitive performance by the tracking software for its domain. A very efficient implementation was necessary to run synthesis and algorithmic agents on the same machine as the beat tracker, and the appropriate scheduling mechanisms were in place to support human drummer controlled algorithmic beats.

This beat induction project has shown the benefits of phase information provided by drum pattern matching. Work is underway on a combining a real-time port of the Davies autocorrelation model with drum pattern based phase alignment decisions as a future beat tracking solution; however, the system as described in the paper was sufficient to finish the improvisation system.

DrumTrack was premiered in a concert on Feb 21st, 2005, with Dave Ellis on drums.

5. REFERENCES

[1] Matthew E. P. Davies and Mark D. Plumbley. Beat tracking with a two state model. In *Proceedings of IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, 2005.

[2] Simon Dixon. Automatic extraction of tempo and beat from expressive performances. *Journal of New Music Research*, 30(1):39–58, 2001.

[3] Masataka Goto. An audio-based real-time beat tracking system for music with or without drum-sounds. *Journal of New Music Research*, 30(2):159–71, 2001.

[4] Jean Laroche. Efficient tempo and beat tracking in audio recordings. *J. Audio. Eng. Soc.*, 51(4):226–233, April 2003.

[5] Eric D. Scheirer. Tempo and beat analysis of acoustic musical signals. *J. Acoust. Soc. Am.*, 103(1):588–601, January 1998.

[6] K M Stephan, M H Thaut, G Wunderlich, W Schicks, B Tian, L Tellmann, T Schmitz, H Herzog, GC McIntosh, RJ Seitz, and V Hömberg. Conscious and subconscious sensorimotor synchronization–prefrontal cortex and the influence of awareness. *Neuroimage*, 15:345–52, Feb 2002.