Beat tracking for multiple applications: A multi-agent system architecture with state recovery

João Lobato Oliveira*, *Student Member, IEEE,* Matthew E. P. Davies, Fabien Gouyon, *Member, IEEE,* and Luís Paulo Reis, *Member, IEEE*

***Abstract*—In this paper we propose an audio beat tracking system, IBT, for multiple applications. The proposed system integrates an automatic monitoring and state recovery mechanism, that applies (re-)inductions of tempo and beats, on a multi-agent-based beat tracking architecture. This system sequentially processes a continuous onset detection function while propagating parallel hypotheses of tempo and beats. Beats can be predicted in a causal or in a non-causal usage mode, which makes the system suitable for diverse applications. We evaluate the performance of the system in both modes on two application scenarios: standard (using a relatively large database of audio clips) and streaming (using long audio streams made up of concatenated clips). We show experimental evidence of the usefulness of the automatic monitoring and state recovery mechanism in the streaming scenario (i.e., improvements in beat tracking accuracy and reaction time). We also show that the system performs efficiently and at a level comparable to state-of-the-art algorithms in the standard scenario. IBT is multi-platform, open-source and freely available, and it includes plugins for different popular audio analysis, synthesis and visualization platforms.**

***Index Terms*—Audio beat tracking, musical rhythm, beat-synchronous applications.**

I. Introduction

A common and often unconscious response when listening to music is to tap one's foot in time to the beat. The computational task which aims to replicate this behavior is known as beat tracking. The identification of beat times from music signals can be addressed in one of two ways, either through *predictive* or *descriptive* beat tracking [1]. Predictive beat tracking is conceptually closer to the human behavior as beat times are estimated in a causal fashion, *i.e.*, predicted in real-time while listening/analyzing the musical input. Descriptive beat tracking on the other hand, places no such requirement of causality, permitting the algorithm to have access to the entire musical input prior to determining beat locations.

While the earliest beat tracking algorithms operated in real-time [2], [3], many algorithms developed since (*e.g.*, [1], [4], [5], [6], [7]) have opted for the descriptive route, since offline algorithms are empirically more accurate than predictive algorithms [8], [9], and since offline applications using beat trackers do not require beats to be estimated in a predictive manner.

Perhaps the most common use for descriptive beat trackers within music information retrieval (MIR) research is to enable temporal processing in musical time. This so-called "beat-synchronous" processing is a key component in music analysis tasks including chord recognition and musical structure estimation [10], music summarization [11] and cover song detection [12], to name but a few. Real-time beat tracking algorithms are applied in situations where a descriptive, "offline" algorithm would be ineffective, *i.e.*, when causal processing is a necessary requirement. Some example applications include: generative composition and remixing music on-the-fly [13], adaptive audio effects [14], or synchronization with live drumming [15]. For a review see [14, Ch.6].

Both offline and real-time systems share the same musical issues with regard to successful beat tracking, namely *i)* selecting a meaningful metrical level and phase at which to tap the beats, *ii)* being able to prevent unwarranted switching between levels (and phases), and *iii)* the ability to track changes in tempo and timing. However, real-time systems are subject to further "functional" issues. These include: *i)* noise robustness, *ii)* computational efficiency, *iii)* reaction time, and *iv)* causality. In this paper our interest is towards a specific application, that of general Robot Audition [16] in the context of interactive robot dancing [17]. Interestingly, a beat tracking application for robot dancing first appeared in Goto's early work – the virtual dancer Cindy [18]. Since all of the functional issues apply for a dancing robot interacting on a real-world scenario we consider this one of the most challenging situations for a beat tracking algorithm.

The standard practice in off-line beat tracking is to evaluate performance on as large a database of audio files as possible, where each file has ground truth beat locations annotated by a musical expert. While this scenario is of some interest for real-time systems, it is also useful to explore real-time specific situations for evaluating on-line beat trackers. One such method is the "streaming" scenario, first proposed by Collins [19]. It involves testing beat tracking performance on musical excerpts which are concatenated without any gaps. For our intended interactive robot dancing application the streaming task is important since it can not only demonstrate performance in terms of *beat accuracy*: the number of correctly estimated beats under some evaluation metric, but also allows for the measurement of *reaction time*: how quickly the beat tracker is

This work was supported by a PhD scholarship (ref. SFRH/BD/43704/2008) and through the projects "ShakeIt" (ref. PTDC/EAT-MMU/112255/2009) and "CASA" (ref. PTDC/EIA-CCO/111050/2009), all endorsed by FCT. J. L. Oliveira is with the Artificial Intelligence and Computer Science Laboratory (LIACC) at Faculty of Engineering of University of Porto (FEUP) and with the Institute for Systems and Computer Engineering of Science and Technology (INESC TEC), Rua Dr. Roberto Frias, 4200-465 Porto, Portugal, e-mail: (joao.lobato.oliveira@fe.up.pt). M. E. P. Davies is with INESC TEC, e-mail: (mdavies@inescporto.pt). F. Gouyon is with INESC TEC and FEUP, e-mail: (fgouyon@inescporto.pt). L. P. Reis is with LIACC and University of Minho, School of Engineering - DSI, Campus de Azurém, 4800-058 Guimarães, Portugal e-mail: (lpreis@dsi.uminho.pt).

able to forget musical information from the previous song and adapt to the next. Since the concatenation of music can lead to abrupt (and unnatural) changes in tempo and phase we believe it is a special beat tracking condition which must be accounted for in the design of the beat tracking algorithm. Our strategy is to incorporate an intelligent state recovery functionality in our agent-based beat tracker, which is able to detect when the beat structure has changed and hence to re-induce and recover the system as quickly as possible with more reliable hypotheses of beat and tempo.

This kind of application-oriented approach raises an interesting issue regarding the extent to which a beat tracker optimized for a particular task will fail in other situations. In the context of our paper, we consider whether a streaming-adapted version of our beat tracker will perform poorly when evaluated in a non-streaming situation and similarly how a "standard" version performs on the streaming task when this additional functionality is removed. Results indicate that, while this is the case, given appropriate knowledge of both types of application it is possible to modify our algorithm so that it is effective in both conditions. In this sense, our approach is towards a generic beat tracking algorithm applicable in diverse musical contexts.

The remainder of this paper is structured as follows: Section II describes the modules of the proposed audio beat tracking system architecture, and gives insights on its practical use and available implementations. Section III describes the proposed evaluation methodology for both standard and streaming scenarios. Section IV describes calibration of the proposed automatic monitoring mechanism (AMM). Section V present and discuss the overall results achieved for both streaming and standard evaluations. Finally, Section VI concludes the paper and presents directions for future work.

II. System Description

IBT (standing for INESC Porto Beat Tracker) is the proposed tempo induction and beat tracking algorithm, first described in [20]. It is inspired by the multi-agent tracking architecture of BeatRoot, where competing agents process parallel hypotheses of tempo and beat [1]. IBT differs from BeatRoot's strategy by identifying beat times in a *predictive* manner through *causal* decisions over incoming input data, instead of making *descriptive* decisions after the whole data has been analyzed. In order to improve the noise-robustness and efficiency of the algorithm, IBT processes *continuous* input data rather than *discrete* onsets. Further, IBT overcomes some of BeatRoot's limitations by considering *multiple types of rhythmic deviations* (*i.e.*, timing and/or tempo) within a musical piece instead of only considering tempo deviations; and applies a *penalizing scoring to bad beat predictions* to contradict the algorithm's tendency to favor faster tempi.

Hence, as depicted in Fig. 1, IBT's algorithm follows a top-down architecture composed of: *i)* an audio feature extraction module that parses the audio data into a continuous feature sequence assumed to convey the predominant information relevant to rhythmic analysis; followed by *ii)* an agents induction module, which (re-)generates a set of new hypotheses

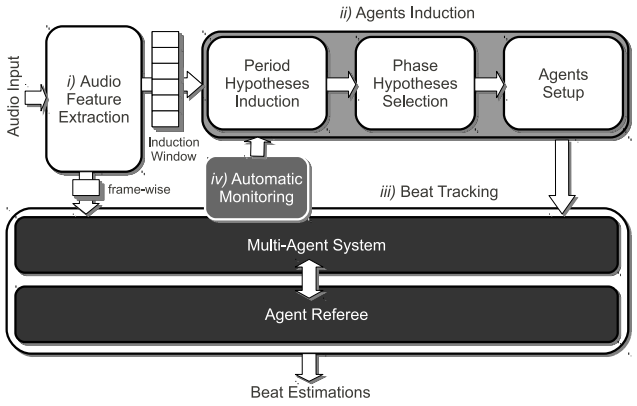


Fig. 1: IBT block diagram.

regarding possible beat periods and phases; followed by *iii)* a beat tracking module, which propagates hypotheses, proceeds to their online creation, killing and ranking, and outputs beats on-the-fly and/or at the end of the analysis. To handle abrupt changes in the musical signal more rapidly and robustly, in real-time contexts (*e.g.*, data streaming), the system also extends [1] and [20] by integrating *iv)* an automatic monitoring mechanism. This mechanism is responsible for supervising the beat tracking analysis of the signal to the necessity of recovering the state of the system through re-inductions of beat and tempo.

All the parameters of the algorithm, described in the following sections, were empirically chosen to maximize performance and stability of the system under different conditions.

A. Audio Feature Extraction

Estimating beat times from audio signals calls for an intermediate low-level representation exhibiting musical accentuation through time [8]. Previous beat tracking models [3], [1] attempted to infer beats from a discrete note onset layer requiring peak-picking algorithms to retrieve plausible rhythmic events from onset detection functions [21]. Considering that this post-processing may induce undesirable errors, besides being susceptible to noise distortions [22], recent beat tracking systems infer the beats from the onset detection functions themselves [8], [9], [4]. In order to emphasize salient note onsets most of these functions rely on spectral features such as energy/magnitude changes (*e.g.*, spectral flux), phase deviations, or combinations of both [9]; either referring to the whole spectrum or to specific psychoacoustic frequency ranges [2], [8]. Based on a comparative study, which evaluated different onset detection functions for beat tracking [23], we selected the spectral flux as our mid-level rhythmic descriptor. It provides a good trade-off between accuracy and computational demands. The spectral flux measures magnitude variations across all frequency bins, k, of the signal's spectrum, $X(n,k)$, along consecutive analysis frames, n. Our implementation computes the time-frequency representation of the signal through a Fast Fourier Transform (FFT), using a Hamming window envelope with w = 1024 samples (23.2 ms at a sampling rate of Fs = 44100 Hz) and 50% overlap. As

proposed in [24], the spectral difference is calculated using the L_1-norm over a linear magnitude, which is half-wave rectified, $HWR(x) = \frac{x+|x|}{2}$, to retain only increasing variations in the magnitude spectrum (emphasizing onsets rather than offsets):

$$SF(n) = \sum_{k=-\frac{w}{2}}^{\frac{w}{2}-1} HWR(|X(n,k)| - |X(n-1,k)|). \quad (1)$$

A circular buffer, with twice the length of the induction window, I, accumulates consecutive frame values of the spectral flux in order to keep an up-to-date continuous function, $SF \rightarrow [0, 2I]$. To remove spurious peaks while retaining the most salient, a low-pass second-order Butterworth filter (with a normalized cutoff frequency of ω_n = 0.28 rad/s) is applied to the accumulated SF at every time-step of the analysis. This filter is applied in both forward and reverse directions resulting in an $\tilde{SF} \rightarrow [0, 2I]$ window with zero-phase distortion.

B. Agents Induction

This module is responsible for (re-)inducing the system's agents with multiple hypotheses of beat positions and tempo. The process makes use of an induction window with fixed-length built of incoming values of spectral flux.

The induction modes of operation are user-definable and range from `single` to `reset` or `regen`. In `single` mode the induction is only run at the very beginning of the signal's processing to set up the first set of agents. In both `reset` and `regen` modes the system is induced at the beginning of the analysis and whenever requested to be re-induced with new hypotheses of beat and tempo. Moreover, in the `reset` mode all previously existing agents are killed from the system and no continuity is implied on the score of the newly created agents. These new scores are processed inside the considered induction window as if the beginning of a completely new musical piece is being analyzed. If the system is operating in `regen` induction mode, all previously exiting agents are kept and the new agents are scored in proportion to the score of the best agent at the time. This aims to minimize the effect of unnecessarily re-inducing the system by repopulating the pool of agents while still relying on the previous best hypotheses. However, since re-inductions of the system are primary oriented for tackling continuous music streams (build on a concatenated set of non-related musical pieces), no effort is made to avoid common beat tracking discontinuities (*e.g.*, switching of the chosen metrical level or switching between on- and off-beat) among excerpts of the same signal before and after new inductions of beat and tempo.

1) Period Hypotheses Estimation: The process of period (tempo) induction typically consists of extracting salient integer-related periodicities organized into a hierarchical metrical structure underlying in the mid-level audio representation [25]. Different methods have been proposed for discriminating periodicities from audio features, either directly selected from symbolic event lists [1], after peak-picking continuous periodicity functions; computed over discrete onset trains [3]; or calculated upon continuous onset functions [4].

According to [22] there are no solid conclusions on which periodicity function (*e.g.*, autocorrelation function (ACF), comb filterbank) is most effective for meter analysis, and whether one should process independent frequency bands, integrated *a posteriori*, or process the feature spectrum as a whole. Based on these considerations, and given the real-time requirements of our algorithm, IBT's period estimation uses the ACF, by taking advantage of its simplicity and efficiency. The decision on the "correct" metrical level was left to the tracking mechanism. We calculated the unbiased ACF of the spectral flux's induction window, $A(\tau)$, along time-lags τ, as:

$$A(\tau) = \sum_{n=0}^{I} \tilde{SF}(n)\tilde{SF}(n+\tau), \quad (2)$$

where $\tilde{SF}(n)$ is the smoothed spectral flux value at frame n, and I is the length of the induction window. The periodicity function is then parsed by an adaptive peak-picking algorithm to retrieve K global maxima, whose time-lags constitute the initial set of period hypotheses P:

$$\begin{cases} P_i = \arg\max_\tau \left(A(\tau)\right), & i = 1, ..., K \\ A(\tau) > \delta \cdot \frac{rms(A(\tau))}{T} \end{cases}, \quad (3)$$

where δ is a fixed threshold parameter, empirically set to 0.75, and T is the chosen tempo range, at a 6 ms granularity. As with most periodicity functions, the clarity of the ACF measurement depends on a rather stable tempo across a given induction window. If no salient periodicities can be extracted, K default periods are assigned (*e.g.*, 120, 100, 160, 80, 140 bpm (beats-per-minute)).

2) Phase Hypotheses Selection: For each of the P_i period hypotheses, M phase hypotheses, ϕ_i^j, (where j is the index of the phase hypotheses for the i-th period hypothesis) are considered among possible phase locations. In order to maximize the suitable starting offsets, these phases are assigned with fixed positions starting at the beginning of the induction window and spaced by $ceil(\frac{P_{max}}{M})$ (where P_{max} is the maximum admitted period) until the end of it. For each period hypothesis, P_i, we generate an isochronous sequence of beats (a "beat train template", Γ_i^j) of constant period for each possible phase, ϕ_i^j, such as $\Gamma_i^j = \phi_i^j + \gamma_i^j P_i \ : \ \gamma_i^j = 0, \cdots, \Upsilon_i^j$, where Υ_i^j is the total numbers of beats in Γ_i^j. For each P_i we then select the beat train template that best fits the spectral flux represented in the considered induction window. For this purpose, a raw score, $s_{i,j}^{raw}$, is computed for every Γ_i^j template by calculating the sum of $\Delta s(error_i^j)$ scores:

$$s_{i,j}^{raw} = \sum_{\gamma_i^j=0}^{\Upsilon_i^j} \Delta s(error_i^j) : error_i^j = m_{\gamma_i^j} - \Gamma_i^j(\gamma_i^j), \quad (4)$$

where $\Delta s(error_i^j)$ is calculated as defined in (13) of Section II-C. It measures the time deviations ($error_i^j$) between each beat time, $b_p = \Gamma_i^j(\gamma_i^j)$, in the chosen train template and local maxima, $m_{\gamma_i^j}$, in the *spectral flux* within a two-level tolerance window, represented by T_{in} and T_{out} (see Fig. 2).

By retrieving the highest $s_{i,j}^{raw}$ for each P_i we select the best suited phase per period hypothesis. This results in K period-phase, (P_i, ϕ_i), hypotheses and their respective s_i^{raw} scores.

3) Agents Setup: The final induction step is to compute and rank a score for each hypothesis. At first, and as proposed in [1], in order to favor candidates whose periods present metrical (*i.e.*, integer) relationships with others we defined a relational score, s_i^{rel}, to each agent, given by:

$$s_i^{rel} = 10 \cdot s_i^{raw} + \sum_{\substack{k=0 \\ k \neq i}}^{K} r(n_{ik}) \cdot s_k^{raw}. \quad (5)$$

The s_i^{rel} of each agent considers both the agent's own raw score, s_i^{raw}, weighted by 10, and the raw scores, s_k^{raw}, of all other $K-1$ agents weighted by $r(n_{ik})$:

$$r(n_{ik}) = \begin{cases} 6 - n_{ik}, & \text{if } 1 \leq n_{ik} \leq 4 \\ 1, & \text{if } 5 \leq n_{ik} \leq 8 \\ 0, & \text{if otherwise} \end{cases}, \quad (6)$$

where $n_{ik} = \frac{P_i}{P_k} : P_i >= P_k \vee n_{ik} = \frac{P_k}{P_i} : P_i < P_k$ is the integer ratio between each pair of period hypotheses, (P_i, P_k), with a tolerance of 15%. This weighting factor is intended to favor *single*, *duple*, *triple*, or *quadruple* metrical relationships among the agents' periods, to a maximum of $r(n_{ik}) = 6$. Ultimately, we define the final agents' scores s_i, for the `single` and `reset` induction modes of operation, as:

$$s_i = \frac{s_i^{rel}}{max(s^{rel})} \cdot max(s^{raw}). \quad (7)$$

In the `regen` induction mode of operation these s_i are additionally normalized by the score of the best agent, sb, at the time-frame n_r of the new induction request:

$$s_i = s_i \cdot sb(n_r). \quad (8)$$

The estimated hypotheses, (P_i, ϕ_i, s_i), can now be used to initialize a set of K new beat agents, which will start their beat tracking activity, as described in the following section.

C. Beat Tracking

The most common tracking strategies for beat estimation make use of oscillating filters [2], [8], probabilistic models [8], autocorrelation methods [4], [9], or multi-agent systems [3], [1]. The high performance of BeatRoot [1] on an important comparative evaluation study [26] combined with its computational efficiency and algorithmic simplicity made it a convincing methodology to pursue for IBT. However, given the better performance of more recent approaches (*e.g.*, [9], [27], [7]) we attempt to overcome some of BeatRoot's limitations (see Section II) while extending it to causal beat predictions.

1) Agents Operation: Using the initial (P_i, ϕ_i, s_i) induction hypotheses, an initial set of K beat agents, representing alternative hypotheses regarding beat positions and tempo, will start to causally propagate predictions based on incoming data. Each agent's beat prediction, b_p, is evaluated with respect to its deviation (*i.e.*, $error$) from the local maximum, m, in the observed $\tilde{SF}$ data within a two-level tolerance window around b_p; such that $error = m - b_p$. This two-level tolerance consists of an *inner* tolerance region, $T_{in} \in [b_p - T_{in}^l, b_p + T_{in}^r]$, $T_{in}^l = T_{in}^r = 46.4\,ms$, for handling short period and phase deviations, and an asymmetric *outer* tolerance region,

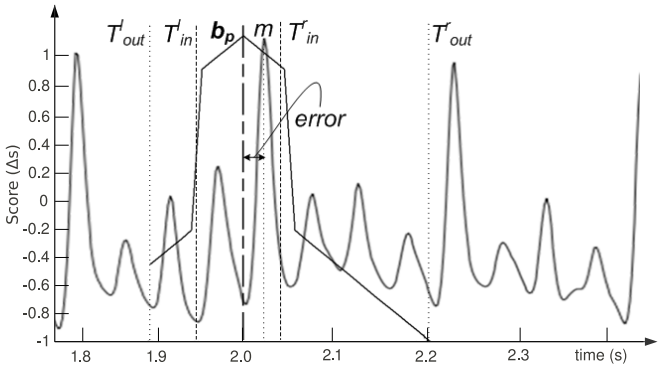


Fig. 2: Score function (thin line) around a beat prediction, b_p, with P_i = 120 bpm. Example of local maximum, m, in the spectral flux (thick line) found in the considered inner tolerance window, T_{in}.

$T_{out} \in [b_p - T_{out}^l, b_p - T_{in}^l[\cup]b_p + T_{in}^r, b_p + T_{out}^r]$, with a left margin $T_{out}^l = 0.2 \cdot P_i$ and a right margin $T_{out}^r = 0.4 \cdot P_i$ (see Fig. 2). This allows for sudden changes in tempo or timing to be followed. The asymmetry reflects the greater tendency of tempo to decrease than increase [1]. Consequently, two alternative scenarios arise. The first scenario corresponds to a local maximum found inside the *inner* tolerance window. In order for the agent's (P_i, ϕ_i) hypothesis to adapt to the observed prediction $error$, the agent's period, P_i, and phase, ϕ_i, are compensated by 25% of that error (limited by the minimum, P_{min}, and maximum, P_{max}, admitted periods):

$$\begin{cases} P_i = P_i + 0.25 \cdot error \\ \phi_i = (\phi_i + 0.25 \cdot error) + P_i \end{cases}, \quad \exists\, m \in T_{in}. \quad (9)$$

The second scenario considers larger deviations, with local maxima in the *outer* tolerance window. In this case, the agent under analysis keeps its period and phase but, in order to cope with sudden variations of tempo and timing, it generates three children $\{C_1, C_2, C_3\}$. These follow three alternative hypotheses [25], considering a possible timing deviation (C_1) or different degrees of tempo and timing (C_2 and C_3) deviations of its own current hypothesis:

$$C_1 : \begin{cases} P_{C_1} = P_i \\ \phi_{C_1} = (\phi_i + error) + P_{C_1} \end{cases}, \quad \exists\, m \in T_{out}, \quad (10)$$

$$C_2 : \begin{cases} P_{C_2} = P_i + error \\ \phi_{C_2} = (\phi_i + error) + P_{C_2} \end{cases}, \quad \exists\, m \in T_{out}, \quad (11)$$

$$C_3 : \begin{cases} P_{C_3} = P_i + 0.5 \cdot error \\ \phi_{C_3} = (\phi_i + 0.5 \cdot error) + P_{C_3} \end{cases}, \quad \exists\, m \in T_{out}, \quad (12)$$

where $P_{C_1}, P_{C_2}, P_{C_3} \in [P_{min}, P_{max}]$. To remain competitive, these new agents inherit 90% of their parent's current score.

Ultimately, different situations may terminate an agent's operation, at any point of the analysis. These include:

- *replacement* – an agent is killed if it is currently the worst in a pool of agents that has reached a maximum number (limited to 30 agents), and if its score is lower than a newly created agent;

- *redundancy* – to increase the algorithm's efficiency, an agent is killed if it is duplicating the work of a highly scoring agent *i.e.*, if their periods differ by less than 11.6 ms and their phases by less than 23.2 ms;
- *obsolescence* – an agent is terminated if the difference between its score and the best agent's is greater than 80% of the best score.
- *loss* – an agent is killed if it seems to be "lost", suggested by a high number (*i.e.*, 8) of consecutive beat predictions outside its inner tolerance window.

2) Agent Referee: To determine the best agent at each data frame, a central *Agent Referee* keeps a running evaluation of all agents at all times. This is conducted by scoring the beat predictions of each agent with respect to its "goodness-of-fit" for incoming spectral flux data.

The following score function, $\Delta s(error)$, is applied around each beat prediction b_p in order to evaluate the distance, $error$, between b_p and the local maximum, m, in the spectral flux inside either the *inner* or the *outer* window (see Fig. 2):

$$\Delta s(error) = \begin{cases} \left(1 - \frac{|error|}{T^r_{out}}\right) \cdot \frac{P_i}{P_{max}} \cdot \tilde{SF}(m), & \text{if } m \in T_{in} \\ \left(\frac{|error|}{T^r_{out}}\right) \cdot \frac{P_i}{P_{max}} \cdot \tilde{SF}(m), & \text{if } m \in T_{out} \end{cases}. \quad (13)$$

The $\frac{P_i}{P_{max}}$ ratio is used to normalize the score function by the agent's period, P_i, as a way to deflate the score of faster agents, which would otherwise tend to increase due to a higher number of beat predictions (leading to a higher number of score increments). The use of a negative score function when a prediction misses by T_{out} assumes a penalizing position (since there is always a maximum inside the tolerance window) responsible for inhibiting the potential overrating of slower agents caused by the period normalization by P_{max}. Therefore, good predictions are highly rewarded and bad ones are also highly penalized. To avoid false positives at best agent transitions (*i.e.*, at the moments when the current best agent A_b is replaced by another agent A'_b), the first beat of A'_b estimated as best agent is ignored if its time estimation is less than $0.6P_b$ ahead the timing of the last beat estimated by A_b (where P_b is the current period of A_b). Due to the causal operation of the algorithm no action can be made to prevent false negatives at transitions of the best agent.

3) Non-Causal Version: Whereas the causal processing of the system retrieves the beats of the *current* best agent, at any time-frame, in the non-causal version only the *last* best agent is considered. This longterm decision distinguishes the family of agents whose cumulative score prevails for the whole piece. In this way, every agent keeps a history of their beat predictions, attached to the one inherited from its parent, and transmits it to future generations. In the case of a re-induction of the system, all new agents inherit the history of the best agent at the time of the new induction request. To prevent false positives at transitions of the best agent, between agents of the same family, the first beat of each newly created child agent is ignored if its initial beat time prediction is less than $0.6P_f$ ahead of the last beat time estimated by its parent (where P_f is its parent's current period). In addition, to prevent false negatives the first beat of the new child is set as the next beat time predicted by its parent if its initial beat time prediction is $0.6P_f$ ahead of the last beat time estimated by its parent.

D. Automatic Monitoring Mechanism

To contend with situations that might require the state recovery of our beat tracking system we investigated an automatic monitoring mechanism to look for indications that the system has lost track of a reliable beat prediction. By looking into the system behavior while tracking a "hard" musical piece (see Fig. 3) we observed that although many children are created from the best agents throughout the analysis of the signal, which suggests high dynamics in the music, most of these new agents show a decrease in their scores (see Fig. 3b) from the moment they are created until they die and rarely prevail. In addition, there are moments where the best agent is repeatedly replaced (*e.g.*, Fig. 3 between 12-13 s and between 25-28 s) and others where it takes too much time to change the best agent while the analysis keeps losing reliability (*e.g.*, Fig. 3 between 36-40 s). These suggest that although the system demonstrates some ability to handle tempo/timing variations in a musical piece, greater measures might be required to recover from abrupt musical changes as those in a streaming context.

Hence, we created an AMM that looks for abrupt changes in the score evolution of the best agent. This monitoring runs at time increments of $t_{hop} = 1\,s$ and it looks for the variation, $\delta\overline{sb}_n$, of the current mean chunk of measurements of the best score, $\overline{sb}_n$, in comparison to the previous, $\overline{sb}_{n-t_{hop}}$, as follows:

$$\delta\overline{sb}_n = \overline{sb}_n - \overline{sb}_{n-t_{hop}} : \overline{sb}_n = \frac{1}{W}\sum_{w=n-W}^{W} sb(n-w), \quad (14)$$

where n is the current time-frame, $W = 3\,s$ is the size of the considered chunk of best score measurements, and $sb(n)$ is the best score measurement at frame n. A new *agents induction* of the system (see Section II-B) is requested if $\delta\overline{sb}_{n-1} \geq \delta_{th} \wedge \delta\overline{sb}_n < \delta_{th} : \delta_{th} = 0.03$. The optimized selection of these parameters is detailed in Section IV. To ensure the steady state of the analysis the AMM halts for one full induction window before considering new induction requests.

E. Practical Use

IBT was developed in C++ and is freely available, under GPL licensing, in MARSYAS (Music Analysis, Retrieval and Synthesis for Audio Signals)[1]. (At the date of writing, revision 4727.) The algorithm is multi-platform and includes four main modes of operation, executable with the following commands:

```
$ ./ibt input.mp3 [causal mode (default)]
$ ./ibt -mic [live mode (data captured by microphone)]
$ ./ibt -nc input.mp3 [non-causal (offline) mode]
$ ./ibt -a input.mp3 [play audio w/ clicks on beats]
```

To activate the AMM use the `-i` parameter, in any of the former modes of operation, as exemplified:

```
$ ./ibt -mic -i "auto-reset" [live mode with AMM
    and reset induction operation]
$ ./ibt -nc -i "auto-regen" input.mp3 [offline
    mode with AMM and regen induction operation]
```

[1]available at http://marsyas.info/.

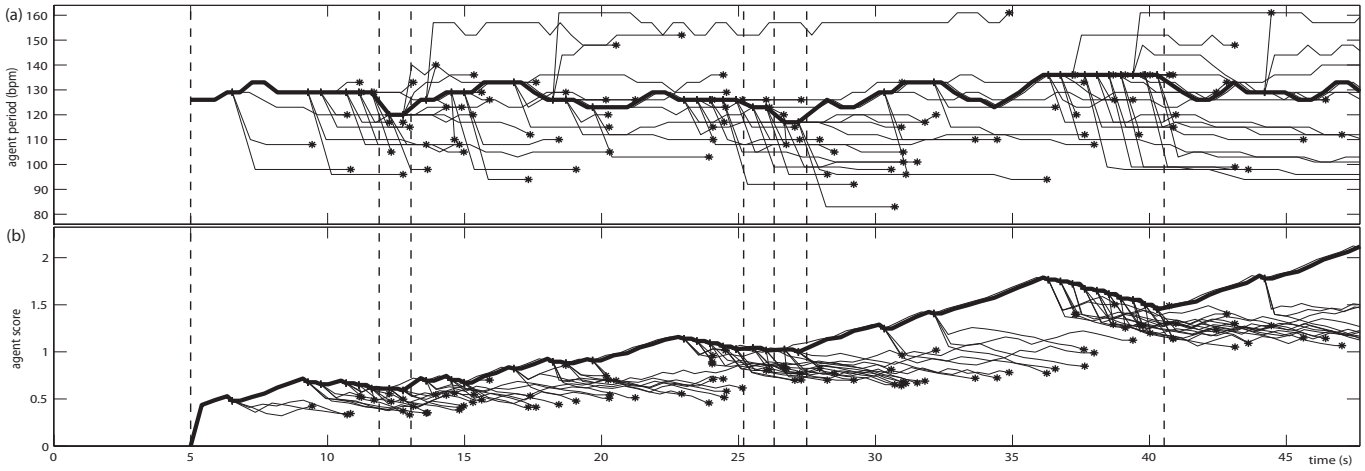


Fig. 3: Evolution of the best agents' proprieties (bold line) and children (branches) while causally tracking the beats of a piece of "Lost in the Flood" by Bruce Springsteen, in the `single` induction mode of operation: (a) Agents period; (b) Agents score. The starting points of the branches represent moments at which children were created from the current best agent, the asterisks the moments at which the same children were killed, and the vertical dashed lines represent changes of the best agent.

1) Plugins: Different plugins have been created to wrap IBT into audio analysis, synthesis and visualization platforms. These include a plugin[2] for Max/MSP [28] a Vamp plugin[2] for Sonic Visualiser [29]; and a plugin[3] for HARK (HRI-JP Audition for Robots with Kyoto University) [30].

2) Existing Applications: Current offline applications that make use of IBT include a DJ recommendation system for the iPad [31] and a music recommendation application for iTunes [32] that use offline tempo estimates to find rhythmic similarities in music playlists. Real-time applications that use IBT comprise a robot dancing system that synchronizes its steps to beats on-the-fly [33], and an interactive robot dancing system that simultaneously processes live music and speech auditory signals. The latter application is driven by IBT running with the AMM for rapidly adapting to continuous music stimuli that change on human request [17].

III. Evaluation Methodology

We report on the performance evaluation of real-time beat tracking in the context of a *streaming* scenario, and of on-line/offline beat tracking on a *standard* scenario using a large dataset of audio files. We compared different variants of IBT with two state-of-the-art algorithms.

A. Datasets

1) Standard Dataset: The *standard* evaluation was run on a dataset consisting of 1358 beat-labeled musical pieces, most with steady tempi ranging from approximately 50 to 250 bpm, and comprising 10 different genres [25]. All data was annotated by expert musicians. The audio data is not publicly available for copyright reasons.

2) Streaming Dataset: The *streaming* evaluation used a set of musical pieces from the former dataset concatenated into two audio data streams: *i)* one for calibrating the system, consisting of 51 concatenated pieces (*i.e.*, 50 music transitions) with a total length of 25 min, and *ii)* one for the actual system evaluation, consisting of another 101 concatenated pieces (*i.e.*, 100 music transitions) across 50 min. To focus the *streaming* evaluation on the specific ability of the system to cope with abrupt signal transitions caused by immediate changes between two musical pieces, the chosen pieces fulfilled a set of three conditions depicted in Fig. 4a. To avoid big signal variations (*e.g.*, expressive timing), beyond the transitions between two consecutive musical pieces, we first isolated data with stable tempi by defining condition d_1 – select data on which the maximum Inter-Beat-Interval (IBI) variation did not exceed the mean IBI of the piece by 40%:

$$d_1 \leq 0.40 : d_1 = \frac{max(IBI) - min(IBI)}{mean(IBI)}. \qquad (15)$$

In order to evaluate the system fairly in a streaming scenario, the second condition, d_2, consisted of selecting only data on which the default IBT scored 100% (with the selected evaluation measure – AMLt – see Section III-B1). Finally, a third condition, d_3, selected, from the remaining data, only musical pieces with tempi ranging from 81 to 160 bpm, to match the tempo limits stipulated in our system. This was to avoid switches of the metrical-level within the same piece of music, which would also degrade the focus of this evaluation.

For concatenating the selected musical pieces into continuous data streams two constraints were also applied. As illustrated in Fig. 4b, first each individual musical piece was trimmed between the time-point, t_i, of an arbitrary annotated beat-time, b_i, and the time-point, t_f, given by:

$$t_f = t_i + b_f + 0.25IBI_f, \qquad (16)$$

where b_f is the first annotated beat time 30 s after b_i, $IBI_f = b_{f+1} - b_f$, and b_{f+1} is the next annotated beat time after b_f. This procedure avoids swapping between the on- and off-beat at the transition of musical excerpts [19]. To provide different levels of difficulty in the reaction at transitions between excerpts, we randomly organized the data stream to ensure a minimum 10% tempo difference between consecutive excerpts (see Fig. 4c). In both the *calibration* and *evaluation*

[2]available at http://smc.inescporto.pt/research/demo_software/.

[3]available at http://winnie.kuis.kyoto-u.ac.jp/HARK/.

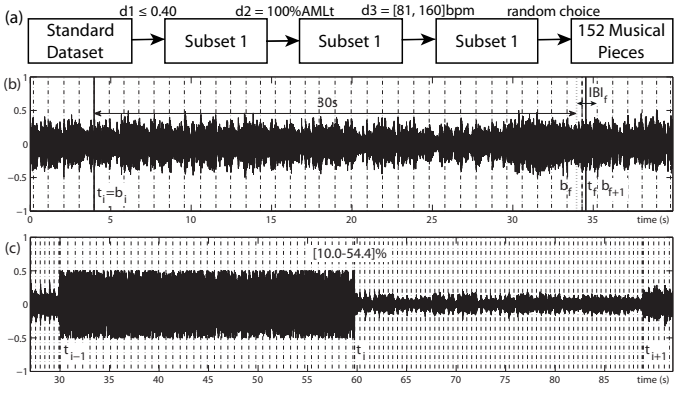


Fig. 4: Building the *streaming* datasets: (a) Data selection from the *standard* dataset; (b) Trimming a musical piece to constitute an excerpt of the data stream; (c) Concatenating the excerpts into a continuous data stream with some random tempo change at music transitions in the range of [10.0-54.4]%. The vertical dashed lines represent annotated beat times.

streams, the tempo deviations at excerpt transitions varied between [10.0-54.4]%. The annotation sequences for the streams were respectively mapped according to these operations.

B. Evaluation Measures

1) Standard Evaluation Measures: For the *standard* beat tracking evaluation, different metrics exist for measuring the performance of a beat tracker relying on groundtruth beat annotations [34]. To keep a broad evaluation with different levels of tolerance in terms of continuity and ambiguity of the algorithm's beat predictions, and for supporting a proper benchmarks of the system we considered four quantitative beat tracking evaluation measures [9]: CMLc (Correct Metrical Level, continuity required), CMLt (Correct Metrical Level, continuity not required), AMLc (Allowed Metrical Levels, continuity required), and AMLt (Allowed Metrical Levels, continuity not required). The CMLc and CMLt respectively quantify the longest number of continuously correct beat estimations and the total amount of correctly estimated beats, at the annotated metrical level and phase. The AMLc and AMLt respectively consider the same continuity conditions but allow for metrical and phase ambiguities, *i.e.*, they consider beat estimations at half and double the rate of the annotation, or at the off-beat (π-phase error) as also correct. These measures consider a tolerance window around each annotated beat time of $\pm 17.5\%$ the length of the considered IBI [8]. In addition, they discard the initial 5 s of the musical piece.

We also measure the algorithm's computational efficiency by calculating the ratio of the computational time, ct, to the length, len, of the file, f, across all F musical pieces in the test dataset. This so-called *real-time factor*, RT_f, is given by:

$$RT_f = \frac{1}{F}\sum_{f=1}^{F}\frac{ct(f)}{len(f)}, \tag{17}$$

2) Streaming Evaluation Measures: For the proposed *streaming* evaluation we used a measure of reaction time combined with a *standard* beat tracking evaluation measure. To quantify the standard performance of our system on the data streams we relied solely on AMLt, as the most permissive measure from those in Section III-B1 and therefore most adequate to assess the reaction time at music transitions. The AMLt discarded the first 5 sec after all excerpts' transitions in the data stream.

The reaction time at music transitions was measured as [19]:

$$r_t = b_r - t_t, \tag{18}$$

where b_r is the first beat-time, in seconds, of the first four consecutive correctly estimated beats in the current musical excerpt; and t_t is the timing of transition from the previous musical excerpt to the current (which matches the first beat-time, t_i, in the current excerpt). The identification of correct beats followed AMLt, at the allowed metrical levels and phases. A transition is considered successful if r_t is less than the duration of the considered musical excerpt, *i.e.*, if the system could recover track of the beats at some point after transiting to the considered musical excerpt.

C. Benchmark Algorithms

For the two proposed evaluation scenarios we compared the performance of different configurations of our beat tracking system with two existing state-of-the-art algorithms.

For the *streaming* evaluation, IBT was configured with different induction modes of operation, automatically requested via the AMM of the system or at pre-defined/random moments of the analysis. These IBT variants will be referred to as:

- `IBT-single`: IBT running on the `single` induction mode of operation;
- `IBT-reset@transitions`|`IBT-regen@transitions`: IBT respectively on the `reset` and `regen` induction modes of operation, requesting re-inductions of the system exactly one induction window after the time-points of each annotated music transition;
- `IBT-reset@random`|`IBT-regen@random`: IBT respectively on the `reset` and `regen` induction modes of operation, requesting re-inductions of the system at random time-points of the analysis in the range of [1.2-15] s;
- `IBT-reset@automatic`|`IBT-regen@automatic`: IBT respectively on the `reset` and `regen` induction modes of operation, requesting re-inductions of the system automatically when proposed by the AMM.

In addition, we selected one of the best reported real-time beat trackers for interactive musical systems: Stark *et al.*'s real-time beat-tracker [35], implemented in C/C++, which we will refer to as `SDP`. To minimize the risk of switching between metrical levels in the real-time analysis of each musical piece, we restricted every variant of IBT to one tempo octave in the range of 81 to 160 bpm, as in `SDP` [35]. Moreover, IBT's induction window length was set to 5 s.

In the *standard* evaluation we tested the same algorithms evaluated in the *streaming* scenario (except `IBT-reset@transitions` and `IBT-regen@transitions` which do not apply) but we additionally considered their non-causal variants for a more broad and generic assessment. For the non-causal version of `SDP` we used its offline Sonic Visualiser plugin [29]. We also tested Dixon's `BeatRoot` [36].

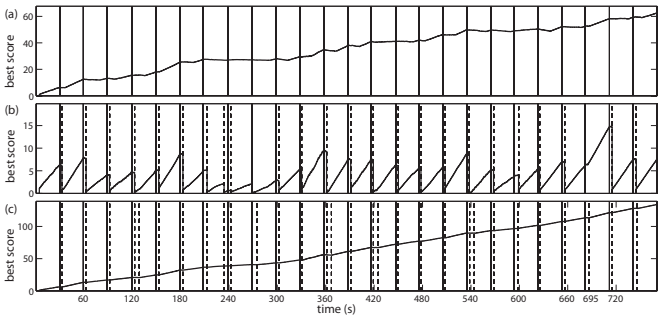


Fig. 5: Evolution of the best agents' score (bold line) throughout the analysis of the *calibration* data stream, for different induction modes of operation: (a) `single`; (b) `reset`; (c) `regen`. The full vertical lines represent the transition times between excerpts, and the dashed vertical lines represent re-induction request times.

IV. Automatic Monitoring Mechanism Calibration

The parameters for IBT's *automatic monitoring mechanism* were calibrated using the *calibration* data streaming. This calibration optimized the $W = 3\,s$, $t_{hop} = 1\,s$, and $\delta_{th} = 0.03$ parameters used in (14), by recurring to the AMLt results of the *streaming* evaluation as the objective function. These were tested under the following acceptable ranges: $W \in [1.0, 10.0]$ s and $t_{hop} \in [0.0, W]$ s, in increments of 1.0 s; and $\delta_{th} \in [0, 0.15]$, in increments of 0.01. A full-scale optimization of these parameters, through a more exhaustive search, is left for future work. Fig. 5 illustrates the evolution of the best score while beat tracking the *calibration* data streaming, for different induction modes of operation set with the optimal parameters: `single` (Fig. 5a), `reset` (Fig. 5b), and `regen` (Fig. 5c). The full vertical lines represent the transition times between each two concatenated musical pieces, and the dashed vertical lines represent request times for re-inductions of the system. Note that the time difference between the musical transitions and the actual induction requests are ideally separated by the length of the induction window, in order to re-induce the system only upon the data of the new musical piece.

IBT operating on the optimal `reset` induction mode (see Fig. 5b) successfully requested a re-induction of the system on 92% of the music transitions of the *calibration* streaming, with 4 false negative and 2 false positive requests. The missing 4 transitions are the result of the rapid self-recovery of the system which neglected a re-induction request, as depicted by the quasi-continuous increase of the best score before and after these transitions (*e.g.*, 695 s). On the other hand, IBT operating on the `regen` induction mode (see Fig. 5c), calibrated with the same parameters, successfully induced the system on all music transitions, but with 16 false positive re-induction requests. The higher number of false positive requests is the result of the system not resetting after a re-induction. This keeps the previous best agent in command and may result in a new re-induction if it is not rapidly replaced by one of the new agents.

V. Results and Discussion

This section presents and discusses the overall results achieved for both *streaming* and *standard* evaluations. All tests were run on a Core2Duo 2.8 GHz Linux 32-bit machine.

TABLE I: Streaming evaluation results on the data stream with 101 concatenated musical excerpts (100 transitions) with approximately 30 s each *vs.* the results on the same 101 individual excerpts (Excrp). The number in parentheses refers to the standard deviation of the mean reaction time, r_t. ST refers to number of successful transitions and NInds to the number of requested inductions of the system.

Data	Algorithm	ST	NInds	r_t(s)	AMLt(%)
Excrp	IBT-single	NA	NA	5.0 (0.0)	100.0
	SDP	NA	NA	5.1 (1.0)	97.5
Evaluation Data Stream	IBT-single	61	1	4.8 (6.3)	32.2
	SDP	99	1	5.4 (3.7)	80.5
	IBT-reset@transitions	100	101	3.8 (1.8)	98.7
	IBT-reset@random	100	382	3.8 (2.3)	93.3
	IBT-reset@automatic	100	107	3.2 (1.6)	97.2
	IBT-regen@transitions	97	101	4.7 (2.4)	89.1
	IBT-regen@random	100	376	7.0 (4.6)	79.8
	IBT-regen@automatic	100	142	4.7 (3.0)	91.3

Given the stochastic nature of the `IBT-reset@random` and `IBT-regen@random` variants, their results on both evaluation scenarios refer to the mean among ten runs of each test.

A. On Streaming Evaluation

Table I presents the results of the calibrated IBT variants on the *evaluation* data stream. These results are compared to the performance of `IBT-single` and `SDP` on the same 101 individual excerpts (*Excrp*), without being concatenated.

As expected, `IBT-single` performs poorly on the streaming data. It can only successfully handle abrupt music transitions of the *streaming* dataset in 61% of the transitions, and scores 32.2% in AMLt. This is a significant drop in performance when compared to the 100% it scored over the individual excerpts contained in this data. Its mean reaction time falls below the 5 s used for induction but it reveals a standard deviation of 6.3 s which would result in potentially high reaction times at music transitions. A similar scenario also arose with `SDP` which revealed a decrease of 17.0 percentage-points (pp) in AMLt when compared to its performance on the individual excerpts, and also showed high mean reaction time. When applying the system's state recovery at the moments requested by the AMM, we could enhance these results almost to the level achieved on the individual excerpts. In this condition, IBT's performance was 97.2% when reseting the system at detected transitions (*i.e.*, with `IBT-reset@automatic`) and 91.3% when regenerating it, with `IBT-regen@automatic`. These were only surpassed by 1.5 pp when given the precise timings of transition for the algorithm to reset itself (*i.e.*, with `IBT-reset@transitions`). The same improvement did not happen when giving the times of transitions for regenerating the system (*i.e.*, with `IBT-regen@transitions`), leading to a decrease of 3 successful transitions (97 of the total 100) and a mean 1.2 pp reduction in AMLt when compared to `IBT-regen@automatic`. This suggests that at some music transitions the previous best hypotheses still prevail to the end of the new piece. To contend with this situation, `IBT-regen@automatic` keeps requesting new inductions if the system cannot rapidly get reliable beat hypotheses, as suggested by its 42 false positive re-induction requests. Although

TABLE II: Standard evaluation results on the *standard* dataset.

Algorithm	Overall Scores (%)							
	Causal				Non-Causal			
	CMLc	CMLt	AMLc	AMLt	CMLc	CMLt	AMLc	AMLt
IBT-single	38.0	46.1	53.2	66.4	43.8	50.2	62.9	73.5
SDP	41.7	47.3	61.6	71.0	46.8	50.8	69.3	75.9
BeatRoot	NA	NA	NA	NA	29.0	35.7	64.8	70.8
IBT-reset@random	34.2	44.6	47.6	63.3	43.6	50.0	62.6	73.2
IBT-reset@automatic	36.8	45.3	51.2	64.5	40.3	47.8	57.3	69.4
IBT-regen@random	36.4	46.1	50.9	65.8	39.4	47.7	56.0	72.9
IBT-regen@automatic	37.9	46.4	52.8	66.4	42.9	49.5	62.0	72.9

the adaptation to new excerpts might get delayed the system can still handle all the music transitions.

In terms of reaction time, the highest results were obtained by `IBT-reset@automatic`, even surpassing `IBT-reset@transitions` by a mean of 0.6 s. This reveals that an automatic monitoring of the beat tracking analysis might generate re-induction requests more suitable to the needs of the system, decreasing the reaction time at abrupt transitions. For this purpose, IBT over-requests re-inductions if the system fails to get reliable beat hypotheses, without compromising its performance, and it does not request any re-induction if it can rapidly handle the given transition by itself. With such automatic reseting IBT could even surpass the reaction time obtained in the individual excerpts, by 1.8 ± 1.6 s, which also suggests that IBT does not require the whole 5 s induction to induce reliable beat hypotheses.

Ultimately, a random choice of moments for requesting new inductions of the system seems to lead to worse results in both AMLt and reaction time, in all conditions, besides the mean 279 more re-induction requests than actual music transitions.

B. On Standard Evaluation

Table II presents the standard evaluation results on the *standard* dataset. On this dataset, the causal variants of IBT could run the whole data with a mean $RT_f = 3.1\%$ of the total dataset duration, whereas the non-causal variants of IBT could run it with a mean $RT_f = 3.5\%$. On the other hand, the `SDP` could compute the same dataset with a $RT_f = 1.7\%$. Although IBT's computational time is slightly higher than SDP's, all the variants of its algorithm are still highly efficient making it suitable for generic real-time applications.

As observed, the causal version of `IBT-single` nearly matches the results achieved by `SDP` with a mean difference of 4.5 pp across all evaluation measures. Furthermore, the causal `IBT-single` seems to be more prone to discontinuities than `SDP` as suggested by the bigger differences between `IBT-single`'s CMLc and CMLt results, by more 2.5 pp than `SDP`'s, and between the AMLc and AMLt results, by more 3.8 pp. This is justified by the sequential operation of IBT without making any high-level effort to keep continuity on its causal beat predictions, contrary to `SDP` that makes use of overlapped windows of the input feature while enforcing dependency among successive estimates of the tempo [35].

Despite the significant operational differences, the non-causal version of `IBT-single` obtained similar results to `BeatRoot` in terms of AMLc and AMLt. Yet, `IBT-single` scored more 2.7 pp on AMLt which we argue to be the most suited measure for evaluating offline beat tracking systems (it seems more critical for offline applications to get the highest number of correctly estimated beats than keeping continuity within the whole beat sequence). Besides, the 1.9 pp decrease in AMLc can be justified by the `BeatRoot`'s bias towards faster metrical levels, which makes it less prone to miss beats (*e.g.*, by inhibiting the switching between on-beat and off-beat). These results were also competitive with `SDP`'s although with a mean decrease in performance of 4.4 pp across both measures. However, regarding the CMLc and CMLt results, `IBT-single` outperformed `BeatRoot` by a mean 14.8 pp, with a mean decrease of 1.8 pp against `SDP`. The big differences of both `IBT`'s and `SDP`'s CMLc and CMLt results in comparison to `BeatRoot`'s are justified by their restrictions to the 81-160 bpm octave. This covers a high percentage of the pieces' tempo from the *standard* dataset which leads to a bias in the determination of the correct metrical level.

Regarding the other variants of IBT we observed similar results but with slight differences between `IBT-reset@automatic` and `IBT-regen@automatic`, for both causal and non-causal operations. Both the causal and non-causal `IBT-reset@automatic` shown a mean reduction of 3.9 pp across all measures in comparison to `IBT-single`, whereas this difference was minimized to respectively 0.1 pp and 0.8 pp with `IBT-regen@automatic`. The differences are justified by the potential lost of continuity when completely reseting the system on unnecessary requests for recovering more reliable beat hypotheses. This undesired effect is reduced by the `IBT-regen@automatic` variant which regenerates the system when requested but keeps the existing agents, leaving major decisions to be taken *a posteriori* by the tracking mechanism.

Akin to the results on the *streaming* data, both `IBT-reset@random` and `IBT-regen@random` variants obtained the worst results on most evaluation measures. This reflects the importance of regenerating the system specifically in the moments where it indicates to be lost, otherwise it would be preferable to let it handle the data changes by itself.

C. Additional Results

Additional comparative results on audio beat tracking and tempo estimation accuracies of IBT can be found respectively

in [20], [27] and [20], [37]. An assessment of the noise-robustness of the algorithm under live conditions in the presence of motor noise from a dancing robot is described in [38]. Further, [17] evaluates IBT on a live streaming scenario also "contaminated" by speech noise from a human speaker.

VI. Conclusions and Future Work

We propose a beat tracking system for multiple applications by applying an automatic monitoring and state recovery mechanism to an agent-based beat tracking architecture. Different variants of the implemented system were evaluated in different musical contexts regarding standard and streaming scenarios, in both causal and non-causal conditions. Benchmarks of these variants reveal improvements in beat tracking accuracy and reaction time when applying the automatic recovery of the system on streaming data. The beat tracking accuracy and efficiency on standard data were with the level of the compared state-of-the-art algorithms, in both causal and non-causal operations. Moreover, specific variants of the developed algorithm are optimal to specific contexts of applications. However, the `IBT-regen@automatic` contends with different operational requirements by balancing reactivity, stability and performance of the system.

IBT is multi-platform, open-source and freely available and it includes plugins for different popular audio analysis, synthesis and visualization platforms. In the future we intend to investigate how the reaction time can be tuned by changing the length of the induction window according to the nature of the application and the type of musical input. Moreover, besides being able to more rapidly react to music transitions, we also plan to investigate if the online monitoring of the system can intrinsically act as a measure of running confidence that would reset the system whenever external noises (*e.g.*, robot motion) interfere with the real-time beat tracking predictions [39].

References

[1] S. Dixon, "Automatic extraction of tempo and beat from expressive performances," *J. of New Music Research*, vol. 30, pp. 39–58, 2001.

[2] E. Scheirer, "Tempo and beat analysis of acoustic musical signals," *J. of the Acoustical Soc. of America*, vol. 103, no. 1, pp. 588–601, 1998.

[3] M. Goto and Y. Muraoka, "A real-time beat tracking system for audio signals," in *Int. Computer Music Conf.*, 1995, pp. 171–174.

[4] D. P. W. Ellis, "Beat tracking by dynamic programming," *J. of New Music Research*, vol. 36, no. 1, pp. 51–60, 2007.

[5] S. Böck and M. Schedl, "Enhanced beat tracking with context-aware neural networks," in *Int. Conf. Dig. Audio Effects*, 2011, pp. 111–116.

[6] G. Peeters and H. Papadopoulos, "Simultaneous beat and downbeat-tracking using a probabilistic framework: Theory and large-scale evaluation," *IEEE Trans. on Audio, Speech, and Language Proc.*, vol. 19, no. 6, pp. 1754–1769, 2011.

[7] N. Degara, E. Argones-Ra, A. Pena, S. Torres-Guijarro, M. E. P. Davies, and M. D. Plumbley, "Reliability-informed beat tracking of musical signals," *IEEE Trans. on Audio, Speech, and Language Proc.*, vol. 20, no. 1, pp. 290–301, 2012.

[8] A. P. Klapuri, A. J. Eronen, and J. T. Astola, "Analysis of the meter of acoustic musical signals," *IEEE Trans. on Audio, Speech and Language Proc.*, vol. 14, no. 1, pp. 342–355, 2006.

[9] M. E. P. Davies and M. D. Plumbley, "Context-dependent beat tracking of musical audio," *IEEE Trans. on Audio, Speech and Language Proc.*, vol. 15, no. 3, pp. 1009–1020, 2007.

[10] M. Mauch and S. Dixon, "Simultaneous estimation of chords and musical context from audio," vol. 18, no. 6, pp. 1280–1289, 2010.

[11] T. Bertin-Mahieux, R. Weiss, and D. Ellis, "Clustering beat-chroma patterns in a large music database," in *11th Int. Soc. for Music Information Retrieval Conf.*, 2010, pp. 111–116.

[12] D. Ellis and G. Poliner, "Identifying cover songs with chroma features and dynamic programming beat tracking," in *Int. Conf. on Acoustics, Speech and Signal Proc.*, 2007, pp. IV–1429–1432.

[13] N. Collins, "BBCut2: Integrating beat tracking and on-the-fly event analysis," *J. of New Music Research*, vol. 35, no. 1, pp. 161–161, 2006.

[14] A. Stark, "Musicians and machines: bridging the semantic gap in live performance," Ph.D. dissertation, School of Electronic Engineering and Computer Science, Queen Mary University of London, 2011.

[15] A. Robertson and M. D. Plumbley, "B-Keeper : A beat-tracker for live performance," in *7th Int. Conf. on New Interfaces for Musical Expression*, 2007, pp. 234–237.

[16] H. G. Okuno and K. Nakadai, "Computational auditory scene analysis and its application to robot audition," in *Hands-Free Speech Communication and Microphone Arrays*, 2008, pp. 124–127.

[17] J. L. Oliveira *et al.*, "An Active Audition Framework for HRI: Application to Interactive Robot Dancing," submitted to IEEE Ro-Man, 2012.

[18] M. Goto, "An audio-based real-time beat tracking system for music with or without drum-sounds," vol. 30, no. 2, pp. 159–171, 2001.

[19] N. Collins, "Towards autonomous agents for live computer music: Realtime machine listening and interactive music systems," Ph.D. dissertation, Department of Music, University of Cambridge, 2006.

[20] J. L. Oliveira, F. Gouyon, L. G. Martins, and L. P. Reis, "IBT: a real-time tempo and beat tracking system," in *11th Int. Soc. for Music Information Retrieval Conf.*, 2010, pp. 291–296.

[21] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. Sandler, "A tutorial on onset detection in musical signals," *IEEE Trans. on Speech and Audio Proc.*, vol. 13, no. 5, pp. 1035–1047, 2005.

[22] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Pedro, "An experimental comparison of audio tempo induction algorithms," *IEEE TASLP*, vol. 14, no. 5, pp. 1832–1844, 2006.

[23] F. Gouyon, S. Dixon, and G. Widmer, "Evaluating low-level features for beat classification and tracking," in *Int. Conf. on Acoustics, Speech and Signal Proc.*, 2007, pp. IV–1309 – IV–1312.

[24] S. Dixon, "Onset detection revisited," in *9th Int. Conf. on Digital Audio Effects*, 2006, pp. 133–137.

[25] F. Gouyon, "A computational approach to rhythm description: audio features for the computation of rhythm periodicity functions and their use in tempo induction and music," PhD Thesis, Universitat Pompeu Fabra, 2006.

[26] M. F. McKinney, D. Moelants, M. E. P. Davies, and A. Klapuri, "Evaluation of audio beat tracking and music tempo extraction algorithms," *J. of New Music Research*, vol. 36, no. 1, pp. 1–16, 2007.

[27] MIREX, "Audio beat tracking contest," 2010. [Online]. Available: http://www.music-ir.org/mirex/wiki/2010:MIREX2010_Results

[28] Cycling '74, "Max/MSP." [Online]. Available: http://cycling74.com/

[29] C. Cannam, C. Landone, M. Sandler, and J. P. Bello, "The sonic visualiser: a visualisation platform for semantic descriptors from musical signals," in *ISMIR*, Victoria, Canada, 2006, pp. 324–327.

[30] K. Nakadai, T. Takahashi, H. G. Okuno, H. Nakajima, Y. Hasegawa, and H. Tsujino, "Design and implementation of robot audition system 'HARK' - open source software for listening to three simultaneous speakers," *Advanced Robotics*, vol. 24, no. 5, pp. 739–761, 2010.

[31] F. X. Aspillaga, J. Cobb, and C. H. Chuan, "Mixme: A recommendation system for DJs," in *Late-Breaking Session of 12th Int. Soc. for Music Information Retrieval Conf.*, 2011.

[32] J. Oliver, "Applying content-based recommendation to personal itunes music libraries," School of Music, College of Architecture, Georgia Inst. of Technology, MSc Degree Tech. Report in Music Technology, 2010.

[33] C. B. Santiago, J. L. Oliveira, L. P. Reis, and A. Sousa, "Autonomous robot dancing synchronized to musical rhythmic stimuli," in *6th Iberian Conf. on Information Systems and Technologies*, 2011, pp. 1002–1007.

[34] M. E. P. Davies, N. Degara, and M. D. Plumbley, "Evaluation methods for musical audio beat tracking algorithms," Queen Mary University of London, Centre for Digital Music, Tech. Rep. C4DM-TR-09-06, 2009.

[35] A. M. Stark, M. E. P. Davies, and M. D. Plumbley, "Real-time beat-synchronous analysis of musical audio," in *12th Int. Conf. on Digital Audio Effects*, 2009, pp. 299–304.

[36] S. Dixon, "Evaluation of the audio beat tracking system BeatRoot," *J. of New Music Research*, vol. 36, no. 1, pp. 39–50, 2007.

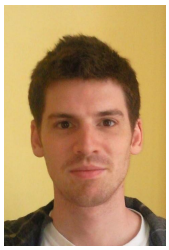
[37] J. Zapata and E. Gómez, "Comparative evaluation and combination of audio tempo estimation approaches," in *Audio Engineering Soc. Conf.: 42nd Int. Conf.: Semantic Audio*, 2011.

[38] J. L. Oliveira, G. Ince, K. Nakamura, and K. Nakadai, "Online audio beat tracking for a dancing robot in the presence of ego-motion noise in a real environment," in *IEEE ICRA*, 2012, pp. 403–408.

[39] J. L. Oliveira *et al.*, "Live Assessment of Beat Tracking for Robot Audition," submitted to IEEE/RSJ IROS, 2012.

João Lobato Oliveira (MSc Electrical and Computers Engineering, University of Porto, 2008) since 2008 is a PhD student in Computer Science at FEUP and a research member of LIACC and INESC TEC in Porto. During his PhD, in 2011 and 2012 he has enrolled as an internship research student at Honda Research Institute Japan Co. Ltd. (HRI-JP), in Tokyo. His main research interests are in autonomous robot dancing, the analysis of rhythm in musical audio, the analysis of human dance motions, and in music-interactive robotic systems.

Fabien Gouyon (PhD Computer Science, UPF Barcelona, 2005) is senior research scientist at INESC TEC in Porto where he leads (together with Prof. Carlos Guedes) the Sound and Music Computing research group. His main research and teaching activities are in Music Information Retrieval and Music Pattern Recognition. He has published over 60 papers in peer-reviewed international conferences and journals, published a book on computational rhythm description, and participated to the writing of the European Roadmap for Sound and Music Computing, published in 2007. He was general chair of the SMC 2009 Conference and the ISMIR 2012 Conference.



Matthew E. P. Davies (PhD Electronic Engineering, Queen Mary University of London, 2007) worked as a post-doc in the Centre for Digital Music before moving to INESC TEC in Porto, in 2011. His main research interests are in the analysis of rhythm, beat and groove in musical audio, evaluation methods in music information retrieval systems and reproducible research.

Luís Paulo Reis (PhD Electrical and Computers Engineering, University of Porto, 2003) is an Associate Professor at the University of Minho, Portugal and a member of the Directive Board of the Artificial Intelligence and Computer Science Lab at the University of Porto, Portugal. His research interests are on Artificial Intelligence, Intelligent Robotics, Multi-Agent Systems and Intelligent Simulation. He was principal investigator of more than 10 research projects on those areas, including FC Portugal robotic soccer team/project, eleven times World/European Champion in RoboCup. He also won more than 30 other scientific awards. He supervised 10 PhD theses and 75 MSc theses to completion. He is the author of more than 250 publications in international conferences and journals. He is the president of the general assembly of both SPR - Portuguese Society for Robotics and AISTI - Iberian association for Information Systems and Technology.