The Pulse: Embedded Beat Sensing Using Physical Data

Neil McGuiness
University of The West of England
Copenhagen, Denmark
neilmcguiness91@gmail.com

Chris Nash
Computer Science and Creative Technology
University of The West of England
Bristol, UK
Chris.Nash@uwe.ac.uk

ABSTRACT

This research investigates the utility of drum vibration data, as the input to beat-tracking algorithms. This approach presents a novel alternative to using either audio signals or MIDI/Virtual Score representations as a means of tempo following and subsequent tempo control. A prototype system: *The Pulse* has been developed as proof of concept for this approach. The system comprises one or more sensors connected to a microcontroller which runs beat detection and tempo tracking algorithms in real-time. This paper discusses scenarios where this approach benefits over existing approaches. As a means of quantitative evaluation, a methodology to compare the functionality of this sensor-based system to that of a contemporary audio signal-based system was also created in the form of a user study which was conducted with the results then analysed here. The conclusion of this project asserts that low-cost sensors, attached to either instruments or performers themselves during a live performance can be reliably used to detect the percussive onsets required by beat-tracking systems and the performance and accuracy of the prototype system is comparable with existing, audio-only systems.

CCS CONCEPTS

• **Applied Computing ~ Sound and Music Computing**

• **Software and its Engineering ~ Embedded Software**

• Human- centered computing ~ Gestural Input;

KEYWORDS

Beat Detection, Embedded Computing, Sensors, Physical Computing

ACM Reference format:

Neil McGuiness, Chris Nash 2019. The Pulse: Embedded Beat Sensing Using Physical Data. In *Proceedings of Audio Mostly (AM 2019). Nottingham, UK, 8 pages. ACM, New York, NY, USA, 2 pages.*

https://doi.org/10.1145/3356590.3356621

1 Introduction

Contemporary popular music increasingly features elements of electronic music in both recorded content and live performances; this results in the expanded use of electronic musical tools including hardware and software synthesizers, sequencers and drum machines etc. Whilst precise editing and overdubbing make this integration relatively straightforward within studio recording scenarios, when having to transpose these elements into a live performance, precise synchronisation between human performers and their electronic counterparts becomes more difficult to manage. Typically this problem is solved by providing the human performers with adequate rhythmic prompting regarding the output of the electronic elements of the performance. Minimally this includes intelligible monitoring of the electronic accompaniment so that their rhythmic influence on the performance can be followed, and usually a synchronised metronome, often provided to the live drummer via headphones to ensure they stay synchronised to the accompaniment. These measures sometimes apply to performers using other non-percussive instruments but fundamentally the drummer, who is typically responsible for providing the central rhythmic pulse of a group performance, particularly in the genres of rock and pop.[1]

This metronome solution however, places the responsibility of synchronisation on the human performer, and limits the dynamic capabilities of the performance [2] and also limits the performer's ability to engage in an immersive performance by requiring their focus to be divided between expressive performance and mechanical timekeeping.

Figure 1: Group rehearsal incorporating The Pulse

Audio-based beat-tracking systems for live performance have emerged over the past decade. These systems use a variety of techniques to determine onsets from audio signals. While these have been successful, an audio approach faces its own set of problems and limitations – primarily, the resolution of the onset detection function due to the computational load of real-time audio analysis, and the separation of different drums from the audio signal, which often requires more than one microphone. By using low-cost sensors attached to individual drums, the energy can be measured at source without the need for a feature extraction phase, ostensibly solving both these problems.

The Pulse is a proof of concept of a sensor-based beat-tracking system, which identifies musical onsets from the sensor data stream in order to infer the *beats* of the music. The onsets are then used to estimate the performer's intended tempo and can update and synchronize any MIDI compatible hardware or software during a real-time performance. This explores an alternative input modality to beat-detection systems that formerly have relied on the analysis of audio signals in order to detect percussive onsets.

Finally, the paper evaluates *The Pulse* primarily through a comparison to an existing system, Andrew Robertson's *BeatSeeker* [4]. [1] This is to determine the efficacy of this novel input method. In order to ensure a fair comparison, the core beat-tracking algorithm used by *The Pulse* is kept as close to *BeatSeeker* as possible, adapting only the signal processing and initial tempo hypothesis stages to suit an embedded system instead of a software plugin.

2 Background

Previous work in the area of tempo detection and beat tracking has mainly used audio signals as a means of detecting discrete musical events or onsets, which are then used to generate a tempo hypothesis. Popular digital audio workstations (DAWs) such as Ableton Live [5] and various examples of DJ software [6,7] perform this tempo analysis offline on entire audio clips, whereas other systems [2, 8] produce a continuously updating tempo estimate in real-time based on the current audio signal and a moving buffer of past onset information. This second approach is the one focused on within this project, as it is more applicable to the live performance use case for which *The Pulse* is designed to handle. More recent work in beat detection in recorded audio, such as Böck et al, [26] has explored using machine learning tools such as recurrent neural networks (RNNs) to aid in detecting beats using both lower level features such as onsets and higher level features such as chord changes to track beats, and downbeats.

2.1 Audio Based Onset Detection

To date there have been many audio-based onset detection algorithms developed and used within a number of different systems for both live and offline use cases. Collins' research [9] makes a comparison and evaluation of 16 different algorithms in which he posits that onset detection functions are typically split into two components:

- A detection function - a signal generated based on continuous causal operations performed on FFT bins of successive FFT frames derived from a Fourier transform of an audio signal. The sampling rate of this detection function is typically lower than standard audio playback (~100Hz) to maximize computational efficiency.
- A peak-picking algorithm - which uses either fixed or adaptive thresholds to find local maxima in the signal output from the detection function to determine higher resolution onset times which will form the basis of the overall tempo hypothesis.
-

Sophistication is added to this basic structure in several models such as Klapuri's approach [10], which divides the signal into eight non-overlapping frequency bands in order to approximate the human cochlea.

Another method is described by Bello et al, [11] in which a psychoacoustic model of loudness perception to inform the onset detection function, applying a logarithmically-adjusted difference function to interpret changes in energy closer to the way that listeners do. The level of detail which audio analysis provides is particularly useful when detecting onsets in polyphonic music with multiple and overlapping instruments. These techniques also provide better performance when detecting onsets in instruments that are less percussive and therefore do not have as clear an attack phase, described by Collins [9] as Pitched-Non-Percussion (PNP).

2.2 Live Beat Tracking Algorithms

Though not specifically developed for live performance, Davies and Plumbley's context-dependent beat tracker [12] presents the template that most recent real-time beat-trackers have followed. This is in the following respects:

- It is based on a causal solution – the system requires no future information on which to base expected beat times or beat periods on therefore all estimations must be made on past information such as previously detected onset times and the time interval between strong peaks in the autocorrelation function of the input audio signal.
- It features a two-pronged approach to the beat tracking, determining the *beat period* and *beat alignment* separately.

The two most salient examples of this method being applied within a dedicated live beat tracking system is Stark's Max/MSP [13] object *btrack~* [8] and Robertson and Plumbley's Max For Live [14] plugin *B-Keeper* [2].

Both systems, like *The Pulse* are focused primarily on musical styles that are based on the repetition of musical patterns, and both can be optimized by a user preconfiguring certain parameters prior to performance [8][2] However, the systems differ in several ways. The key difference being that *btrack~* was designed to work on arbitrary musical audio, while *B-Keeper* focuses primarily on drum audio. Moreover, while both systems favor tempo coherence, that is to say they favor detected beats that closest resemble those

[1] *BeatSeeker* being a commercial iteration of an earlier system: *B-Keeper*[2].

expected based on the current tempo estimation, [15] Starks's approach to bias the system to this interpretation is with the use of a *cumulative score* – a weighted sum of detected beats and expected beats, whereas Robertson and Plumbley use a Gaussian distribution function centered around the current system estimates to both determine the beat period and adjust *B-Keeper's* internal beat phase for the beat alignment.

2.3 Live Instrument Sensor Integration

With the surge of do-it-yourself electronics kits and open-source programming environments becoming more widely explored by computer hobbyists and multi-disciplinary technologists since the early 2000s, the field of music technology has integrated many of these tools to augment existing instruments and interfaces or create entirely new ones. These innovations have led to the advance of new types of musical performances and installations [16]. Several laboratories have pursued research and projects that integrate sensors and embedded technology into traditional musical instruments, such as The Augmented Instruments Laboratory at the Centre For Digital Music (C4DM)[17, 18], CCRMA[25] and IRCAM[26]

Medeiros and Wanderly's research into sensor usage in devices for musical expression [19] shows that accelerometers have been the most popular addition to existing instruments, most likely due to their affordability, ease-of-use and ability to sense movement in three axes, making it a useful tool to measure expressive movement of a performer. Piezo-electric discs on the other hand have not been adopted as enthusiastically into this field. One recent example of the use of piezo-electric sensors is the *Bi-Stable cymbal resonator* [20], which uses the piezo disc as a pickup in a coupled sensor-actuator feedback network, using a cymbal's resonance to create a musical drone.

2.4 Motivation For Sensors

Within this paper, kick-drum onset detection is the performance quality of interest. Such a gesture has a relatively simple nature and can be sensed using a piezo sensor, which accurately measures changes in pressure or force. This data stream provides apparent peaks when its attached drum is engaged, negating the need for a feature extraction phase. To explore a body-metrical approach, wearable sensors attached to the ankle or leg used to operate the kick drum were also tested and data recorded.

Recent work in onset detection in relation to tempo tracking typically sets the resolution of their respective detection function at 11.6ms [12], this is based on the assumption that the human ear cannot differentiate between transients less than 10ms apart [11]. Due to the nature of the windowing used in FFT that facilitates audio-based onset detection, the process inherently incurs a delay and therefore audio-based approaches are limited in how fine their resolution can be. A sensor-based approach has no such constraint and the resolution is limited only by the sample rate of the analog to digital converters on the platform's sensor inputs. Of course the detection logic used can also incur latency if windowing is used, but the possible strategies are more flexible and typically require less computation than FFT. Although this higher resolution may not mark any improvements regarding the detection of onsets, it may provide greater lead-time to any processing the beat tracking and tempo updating procedures, ultimately meaning a quicker system.

The Bela platform [3] used in this project has an analog sampling rate of 22050Hz and processes the data as well as executing the beat-tracking logic on the highest priority thread at audio rate, in order not to overload the thread, we down-sampled the sensor data to 1000Hz. This still provides an onset detection resolution of 1ms, more than ten times the rate of comparable audio-based systems.

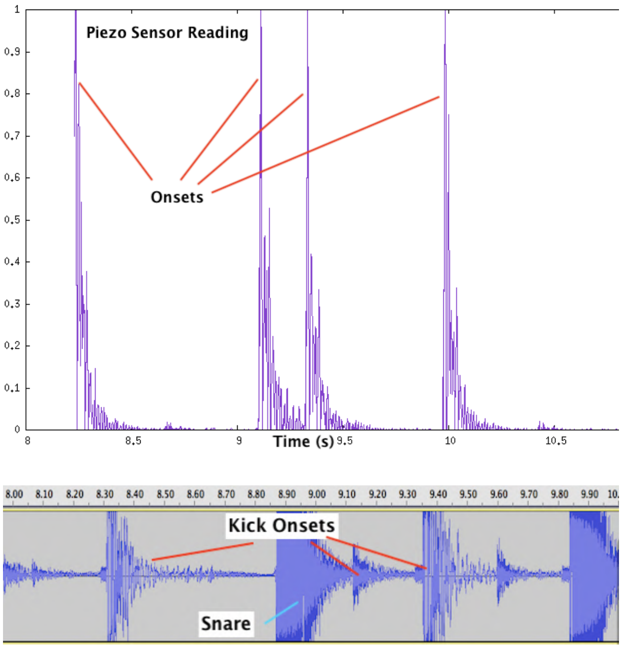

Figure 2: Top - Piezo sensor data, down- sampled to 1000Hz, Bottom- Audio wave of same performance (comparison not precisely to scale.)

3 Hardware

Developed as an embedded system, *The Pulse* comprises hardware sensors connected to the Bela platform [3] .The data from these sensors is analyzed and onset times are extracted from the signal using a simple threshold. Sensor peaks above this threshold will cause the embedded software to add its current CPU time to an array of onset times. The beat tracking algorithm then uses these onset times to estimate the current performance tempo and update any connected devices using MIDI protocol via the USB port on the BeagleBone Black. As *The Pulse* is an embedded device it has no graphical user interface (GUI) by which to provide the user critical feedback about the system's state and current tempo estimation. As a simple solution to this, a single LED is connected to the Bela that blinks at quarter note pulses at current BPM estimate, or is continuously on when awaiting an initial estimate.

3.1 Sensors

For this prototype design, only two sensors were used during the test performances: One piezo sensor, used to measure impact upon the drum kit, and one accelerometer to measure body-metric data of the performer.

Figure 3 shows the piezo sensor in a small enclosure, which is attached to the inner kick drum skin during performance. This position receives adequate force from the kick pedal's beater when played without risking damage to the sensor itself. The sensor is attached to an analog input of the microcontroller in series with a 1MΩ resistor to limit the voltage fluctuation upon impact, as the piezo disc is very sensitive to impulses.

Figure 3: Piezo sensor inside kick drum shell

Figure 4 shows the 3-axis ADXL 335 accelerometer embedded within an ankle strap, worn on the performer's right ankle, which is typically used to operate the kick pedal. Preliminary tests analyzing body-metric data involved experimenting with the location of accelerometers around the performers body, with the conclusion being that the y-axis of the ankle sensor being the most indicative of the performer's intended tempo, with additional sensors on the legs, or extra axes providing either complicating or redundant information.

Figure 4: Ankle sensor worn by performer

4. Implementation

In order to provide a fair comparison of *The Pulse* and *BeatSeeker*, the algorithm that is the basis of *The Pulse* is accordingly a C++ interpretation of *BeatSeeker* based on a detailed description of Robertson's implementation of *B-Keeper* [21]. As the input signals are different, an alternative method is used to detect onsets. As for the subsequent stages, *The Pulse* conforms to the approach and assumptions made in Robertson and Plumbley's work [22].

Figure 5 outlines the system architecture; excluding the output of a MIDI message via USB and the updating of the LED. This displays the two-state model approach [23] of tempo adjustment and beat alignment as being executed concurrently, when in fact the synchronisation is being performed before the global tempo adjustment in order to maximize accuracy and sync.

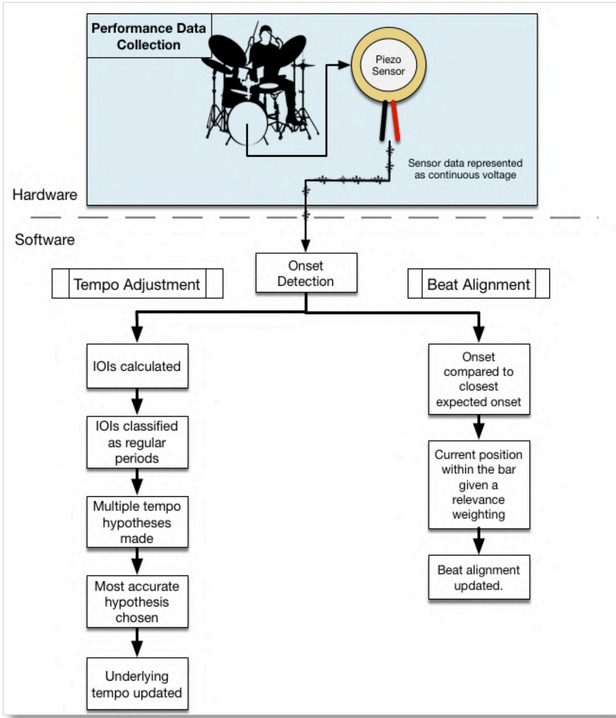

Figure 5: System Architecture

4.1 Onset Detection

Due to the clarity of the streams of sensor data with respect to the performance, only a simple method of programmatically detecting onsets was required. In this case, a debounced threshold, calibrated to the individual sensor ranges, is applied to an envelope following function on each of the sensors. Peaks in the sensor stream above an initial threshold trigger an onset flag, which is not switched off again until the sensor reading dips below a lower threshold, at which point new peaks can be detected, separating distinct onsets from noise or bouncing around the threshold. When a peak is successfully detected, the CPU time is stored in a circular buffer, capable of storing the eight most recent onsets.

An approach using sensor fusion was trialed during the early stages of the research in which a simple Markov model determines an onset ONLY if the ankle sensor exceeds the threshold followed by the drum sensor exceeding the threshold within a time window. However this approach exhibited no discernable improvement over the separate sensor approach for either spurious detections (false positives) or missing onsets (false negatives). Therefore, the ankle sensor data was not used in the final tests and comparison phase, although it could have been reliably used in place of the kick drum piezo sensor.

4.2 Tempo Adjustment

The tempo adjustment procedure determines the underlying tempo estimate using the following strategy:

- When a new onset is detected, the times between the most recently detected onset and the previous seven are calculated and stored as a buffer of inter-onset-intervals (IOIs) using the method:

  $$IOI_k = t_k - t_n$$

  Where t_n is the CPU time of the most recently detected onset and t_k is the CPU time of a previous onset.
- These IOIs are categorized to their closest integer multiple (IM) of an eighth note at the current tempo estimate, ostensibly classifying the IOI as a regular note duration. The categorization is performed by:

  $$IM_k = round(\frac{IOI_k}{\tau})$$

  Where τ is the duration of an eighth note at the current tempo estimate.
- The performance error E_k between the classified integer multiple and the actual IOI is calculated using:

  $$E_k = IOI_k - IM_k$$

- The overall accuracy A of the IOIs, in regard to the current tempo estimate, is then assessed using two parameters.
  - A non-normalized Gaussian window function $g(E_k, \sigma_{Tempo})$, centered around 0 and with a dynamic standard deviation σ_{Tempo} that adapts to the performance and is recalculated with each new onset. This means that if the standard deviation is larger, the window is wider, and larger error terms will be considered more accurate than if the window is narrower.
  - A series of weightings around the classified note durations, favoring durations that are heuristically determined [24] to be more relevant to beat tracking in the majority of use cases. In our case this means that quarter notes ($IM = 2$) and half notes ($IM = 4$) are set to the maximum weighting of 1, with eighth notes ($IM = 1$) and whole notes ($IM = 8$) being set close to 1 (~ 0.8) and all other durations such as dotted notes etc. are set to 0.
- Therefore, the accuracy is determined via:

  $$A = g(E_k, \sigma_{Tempo})W(IM_k)$$

- Finally, once the accuracies of all the IOIs are calculated, the highest accuracy is chosen, and if this accuracy is above the *update threshold*, its respective IOI and duration weighting are used to update the underlying tempo estimate of the system.

$$\Delta tempo = a.g(E_{win}).W(IM_{win}).\frac{g(E_{win})}{IM_k}$$

4.3 Beat Alignment

For a live beat tracking system, it is essential not only to maintain an accurate estimate of the global performance tempo, but also adequate synchronisation between the system's internal tempo grid and the drummer's possible timing deviations from it. To maintain synchronisation, when an onset occurs proximally to the system's expected onset, the internal grid will effectively be snapped to this new onset. Here *The Pulse* differs from *BeatSeeker* in that the latter receives a MIDI click from its host DAW (Live) whereas in the former an internal pulse is procedurally generated, effectively creating a MIDI click or expected onset

at every eighth note, in order to compare to the real onset when it occurs. The steps to synchronize the system are as follows:

- When a new onset t_n is detected, the CPU time will be compared to that of the closest expected onset $E[t_n]$. And the difference calculated:

  $$\bullet\, d_n = t_n - E[t_n]$$

- Similar to the weighting of the note duration in section 4.2, a weighting is applied to the metrical position within the bar of the closest expected onset. With typical kick drum placements of 0 eighth notes in (start of the bar, often said aloud as the *One* of the bar) and 4 eighth notes (*Three*) being weighted at 1 and the offbeat eighth notes in between weighted close to 0.
- The difference between the detected onset and expected onset is then passed through a similar yet separate Gaussian window function from the one specified in part 4.2. This assesses the need for synchronisation with respect to the current performance, the result of this is multiplied by the position weighting.
- The weighted result is then compared to a synchronisation threshold ξ, which if exceeded will synchronize the internal pulse to this onset by making a small tempo adjustment in addition to the global tempo adjustment in part 4.2. This logic is represented in the following rule:

$$\text{IF } g\left(d_n, \sigma_{sync}\right)W(E[t_n]) > \xi \text{ THEN sync}$$

5. Evaluation/Comparison

In order to precisely compare the beat detection capabilities of both *BeatSeeker* and *The Pulse* it was essential to conduct a study in which a singular performance was simultaneously monitored by both systems, whose continuous reported tempo estimations could later be compared and analysed. Rather than simply monitoring an improvised free performance, a quantitatively known tempo map was introduced as the primary layer from which the drummer is prompted. This creates a baseline that the two systems can be compared to, although it is acknowledged that the accuracy of the drummer's performance relative to the tempo map cannot fully be known. Nevertheless the principal correlation of interest is that of the two systems' tempo output, and not that of their outputs to the tempo map. Figure 6 illustrates the comparison study in which a performer was prompted via a click track following the tempo map, the performance is monitored by both systems, and their continuous tempo estimates (in Beats Per Minute) are recorded into time-stamped files at a rate of 1000Hz. This results in a high-resolution tempo map from each system that can be compared to the prompt tempo map.

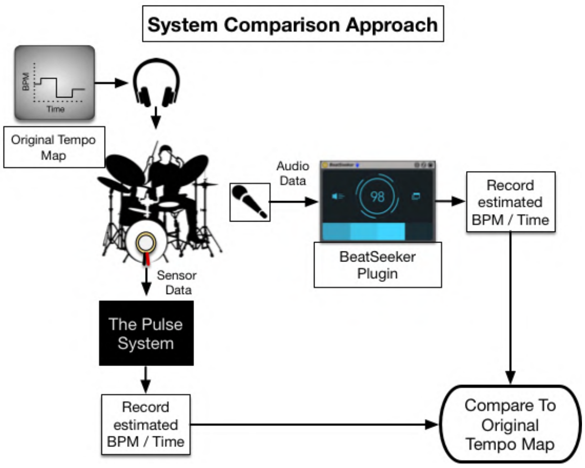

Figure 6: Comparison Study Approach

5.1 Results

Comparison of the three tempo maps illustrates the similarity of the systems' response to the performance. As figure 7 shows, both systems' maps closely resemble that of the prompt map, with ***P*** *(The Pulse)* seeming to respond quicker to sudden changes in tempo than ***B*** *(BeatSeeker).* This may be due to variances in the systems' tempo-update coefficient, which could be set lower in ***B***'s implementation, resulting in slower but more stable tempo tracking. This could also be due to P's potentially higher resolution onset detection specified in 2.4. The average difference of both systems' maps from the prompt map is noticeably similar (μ_B = 6.86, μ_P = 5.16), obtained by calculating the mean of the absolute deviation of each system's tempo estimation points from the corresponding points on the known tempo map. The average difference between the two systems' tempo maps was slightly larger ($\mu_d = 10.26$). Figure 8 demonstrates the continuous difference between both systems' tempo estimates and the prompt tempo.

The general trend shows that the systems' accuracies concurrently fluctuated, usually with a higher discrepancy from ***B***, although both systems had points of contrary performance. For instance, where 17s < t < 22s, ***P***'s tempo estimate is accurate to within < 5 bpm of the prompt map, whereas within the same window ***B***'s estimate drifts away from the map by up to 20 bpm. Comparably where 82s < t < 86s, a similar trend can be observed where ***B***'s estimate remains very close to the prompt map and ***P***'s estimate reaches its most inaccurate point at a difference of 20 bpm.

6.CONCLUSION & FUTURE WORK

The results of the comparison study support the hypothesis that beat-tracking systems that use sensor data as their input for onset detection can perform similarly to those that use audio-based detection functions. Although only a modification of the input modality, this alteration, along with the embedded nature of this project provide the possibility for other revisions to the system that improve its integration as a live music performance tool.

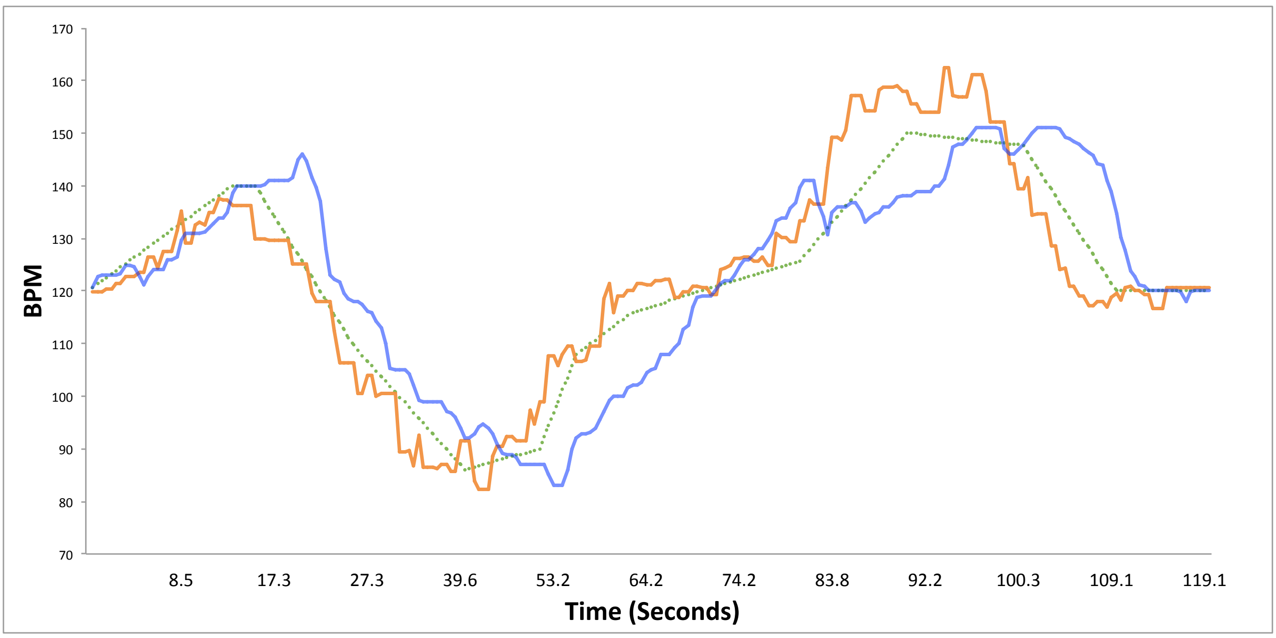

Figure 7 – BPM estimates over Time: Dotted Green = Prompt Tempo Map, Orange = The Pulse, Blue = BeatSeeker

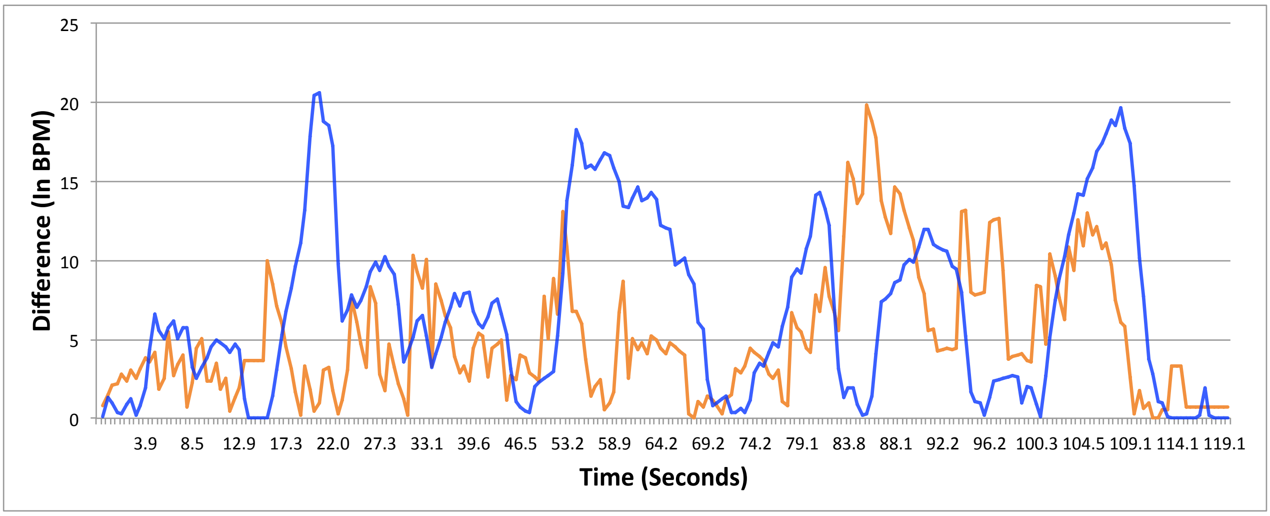

Figure 8 – Systems' absolute difference from the Prompt Tempo Map over Time: Orange = The Pulse, Blue = BeatSeeker

The Pulse has been designed as an embedded system, using a microcontroller to process sensor data, run the beat-tracking algorithms and output an updating MIDI clock pulse. For this prototype the Bela platform [3] was used for its convenient sensor integration and USB port. However, in the future the design could be encapsulated into an integrated circuit with a single-chip microcontroller such as the ATMEGA328.

Also, in future revisions to this design, more sensors could be used to obtain a more detailed representation of the performance, including a piezo sensor on each drum to be able to separate the signals and measure a complete sequence of beats from multiple drums. The Bela board has 8 analog inputs with an option of adding a multiplexing cape to increase this number by up to 64 inputs, with a corresponding division of the sampling rate. However, 8 inputs would be enough to connect a sensor from the board to each drum in a typical setup and the internal algorithm could be updated to treat each input as a specific drum that will contribute to the drum pattern with relative probability weightings to that drum's likely position within the beat, and a further weighting as to that drum's contribution to the overall estimate – for instance, the kick and snare drums would have more influence over the BPM estimate than a rack tom.

While this research required *The Pulse* to function as a 'Black Box' – with as little prior knowledge about the performance or its inputs as possible, such preparations can aid the system to be more accurate in its tempo and beat tracking [8]. For instance a button matrix located on the system's enclosure could be used to pre-select a known kick drum pattern, which would alter the period duration and beat position weightings used in the tempo adjustment and synchronisation procedures respectively, biasing the tracking towards this pattern.

As with *BeatSeeker*, we assume a common meter of 4/4, whereas Davies and Plumbley [12] used peaks in their onset-detecting autocorrelation function at a higher, metrical level to determine a binary (2 or 4) or ternary (3 or 6) meter. This same approach could be incorporated here if we hold with the assumption that the strongest onset will occur at the start of the bar, and note the strongest detected onsets within the sensor data. Relatedly, the amplitude of each onset could be considered and used to inform the weighting of each onset and its position, creating a more sophisticated beat tracker, rather than determining onsets only by a threshold. *BeatSeeker* has the feature of two functionality modes, tempo-lock and tempo-follow. Tempo-lock is a bypass mode where no tempo update will be made to Live, and tempo-follow is the usual mode that has been imitated in this research. Multiple modes such as these could be incorporated into *The Pulse*.

Finally, exploring the more qualitative aspects of the live performance user experience, *The Pulse* could include multiple peripheral switches to control parameters of the system. These switches could simply be MIDI drum pads that easily assimilate into a standard drum set up, but connected to the system could update user parameters such as switching between modes, adjusting the algorithms sensitivity or the width of the Gaussian windows etc. In order to control the system in this way a more sophisticated from of visual feedback would be required, possibly a seven-segment display that informs the user of the current mode, system parameters and current bpm estimate.

ACKNOWLEDGMENTS

This research was supported by the Impulse Enterprise Studio, at the University of the West of England (UWE).

REFERENCES

[1] M. Goto and Y. Muraoka. Music Understanding At The Beat Level. *In Proceedings of IJCAI '95* (Quebec, Canada, August 20-25 1995), 157-176
[2] A. Robertson and M. Plumbley. B-Keeper: A Beat tracker For Live Performance. *In Proceedings of NIME'07* (New York, USA, June 6-10 2007) 234-237
[3] A. McPherson, R. Jack and G. Moro. Action-Sound Latency: Are Our Tools Fast Enough. *In Proceedings of NIME ' 16* (Brisbane, Australia, 2016)
[4] https://www.ableton.com/packs/BeatSeeker
[5] https://www.ableton.com
[6] https://serato.com
[7] https://www.native-instruments.com/products/traktor/
[8] A. Stark. 2011. *Musicians and Machines: Bridging the Semantic Gap in Live Performance (Centre for Digital Music).* Ph.D. Dissertation. Queen Mary University of London.

[9] N. Collins. A Comparison of Sound Onset Detection Algorithms with Emphasis on Psychoacoustically Motivated Detection Functions. *In Proceedings of Audio Engineering Society (AES) 118th Convention* (Barcelona, Spain, May 28-31 2005)

[10] A. Klapuri. Sound Onset Detection By Applying Psychoacoustic Knowledge. *In Proc. IEEE International conference of Acoustics, Speech and Signal Processing (ICASSP)* (Tampere, Finland, 1999) 3089-3092

[11] J. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies and M. Sandler. A Tutorial on Onset Detection in Music Signals. *IEEE Transactions on Speech and Audio Processing, 13* (5), 1035-1047

[12] M. Davies and M. Plumbley. Context-Dependent Beat Tracking of Musical Audio. *IEEE Transactions on Audio, Speech and Language Processing, 15* (3) 1009 -1020

[13] https://cycling74.com/products/max/

[14] https://www.ableton.com/en/live/max-for-live/

[15] P. Allen and R. Dannenberg. Tracking Musical Beats in Real Time. *In Proc. International Computer Music Conference* (September, 1990) 140-143

[16] S. Waloschek and A. Hadjakos. Sensors on Stage: Conquering the Requirements of Artistic Experiments and Live Performances. *In Proceedings of NIME'15* (LA, USA, May 31-June 3, 2015) 351-354

[17] A. McPherson. The magnetic resonator piano: electronic augmentation of an acoustic grand piano. *Journal of New Music Research 2010, 39* (3), 189-202.

[18] A. McPherson and Y. Kim. Design and applications of a multi-touch musical keyboard. *In Proc. Sound and Music Computing*, (Padova, Italy, 2011)

[19] C. Medeiros and M. Wanderley. A comprehensive review of sensors and instrumentation methods in devices for musical expression. *Sensors*, *14*(8), 13556-13591.

[20] A. Piepenbrink and M. Wright. The bistable resonator cymbal: an actuated acoustic instrument displaying physical audio effects *In Proceedings of NIME'15* (LA, USA, May 31-June 3, 2015) 227-230.

[21] A. Robertson. 2009. *Interactive Real-time Musical Systems (Centre for Digital Music)*. Ph.D. Dissertation. Queen Mary University of London.

[22] A. Robertson and M. Plumbley. Synchronizing sequencing software to a live drummer. *Computer Music Journal*, *37*(2), 46-60.

[23] M. Davies and M. Plumbley. Beat tracking with a two state model. *In Proceedings of International Conference Acoustics, Speech, and Signal Processing, 2005 (ICASSP'05)* (March 23, 2005)

[24] E. Berdahl, H.C Steiner and C. Oldham. Practical Hardware and Algorithms for Creating Haptic Musical Instruments. *NIME 2008, 8* ,61-66

[25] J, Lerdahl, and Jackendorff. 1983. A Generative Theory of Tonal Music. Cambridge, Massachusetts: MIT Press.

[26] S. Bock, F. Krebs, and G. Widmer, "Beat and down- ¨ beat tracking with recurrent neural networks," in International Society for Music Information Retrieval Conference (ISMIR), 2016.