A framework to provide fine-grained time-dependent context for active listening experiences

Joren Six Marc Leman
IPEM, University Ghent
joren.six@ugent.be

March 2017

Abstract

This work presents a system that is able to provide fine-grained time-dependent This work presents a system that is able to provide fine-grained time-dependent context while listening to recorded music. By utilizing acoustic fingerprinting techniques the system recognizes which music is playing in the environment and also determines an exact playback position. This makes it possible to provide context at exactly the right time. The design of the system can be used to augment listening experiences with lyrics, scores, tablature or even music videos. To test the concept, a prototype has been built that is able to give feedback that coincides with the beat of music playing in the users environment. The system is evaluated with respect to timing and is able to respond to beats within 16 ms on average.

1 Introduction

The ability to identify which music is playing in the environment of a user has several use cases. After a successful recognition meta-data about the music is immediately available: artist, title, album. More indirect information can also be made available: related artist, upcoming concerts by the artist or where to buy the music. Such systems have been in use for more than a decade now.

A system that is able to not only recognize the music, but also determine a sufficiently *precise* playback time opens new possibilities. It would allow to show lyrics, scores or tablature in sync with the music. If the time resolution is fine enough it would even allow to play music videos or visuals synced to the environment. In this work we focus on active listening experiences where precise timing is required.

Commercial applications such as SoundHound and Shazam already incorporate real-time synchronized display of lyrics. However, lyrics do not need a very precise time resolution. Dan Deacon, an electronic music artist created, a smartphone application that *"turns each phone into a source of synchronized light and sound depending on your location within each venue"*. Precise timing is needed for this to work but this application operated by specific queues in the music. So it only supports a few predefined pieces of music.

In this work a design of a system is proposed that i) is able to find the playback time of the music in the environment precisely and ii) provide feedback synchronized to the environment. The paper focuses on yet another type of time-dependent contextual data: beats. A prototype is developed that provides feedback exactly on the beat for the following three reasons:

1. For its *inherent value*. Humans are generally able to track musical beat and rhythm. Synchronizing movement with perceived beats is a process that is natural to most. Both processes develop during early childhood[7]. However, some humans are unable to follow musical

beat. They fall into two categories. The first category are people that suffer from hearing impairments which have difficulties to perceive sound in general and music in particular. Especially users of cochlear implants that were early-deafened but only implanted during adolescence or later have difficulties following rhythm[5, 17]. In contrast, post-lingually deafened CI users show similar performance as normal hearing persons[9]. The second category are people suffering from beat deafness[11, 8]. Beat deafness is a type of congenital amusia which makes it impossible to extract music's beat. Both groups could benefit from a technology that finds the beat in music and provides tactile or visual feedback on the beat if they want to confidently partake in dance.

2. For *evaluation purposes*. Using discrete events - the beats - makes evaluation relatively straightforward. It is a matter of comparing the expected beat timing with the timing of the feedback event.

3. For *pragmatic reasons*. The contextual data - the beat lists - are available or can be generated easily. There are a few options to extract beat timestamps. The first is to manually annotate beat information for each piece of music in the reference database. It is the most reliable method, but also the most laborious. The second option is to use a state of the art beat tracking algorithms e.g. the one available in Essentia[2]. The third option is to request beat timestamps from specialized web services. The AcousticBrainz project[1] provides such a service. AcousticBrainz currently has information for more than two million songs. It is similar to the Echo Nest API[1] but AcousticBrainz' inner workings are well documented and completely transparent. In the prototype AcousticBrainz is used to provide beat timings. If the audio is not present in AcousticBrainz, the beat timings are extracted using the previously mentioned beat-tracker and stored locally.

[1]Information on the AcousticBrainz project is available at its website `http://acousticbrainz.org/`

The following sections present the system and its evaluation. The paper ends with the discussion and the conclusions.

2 System overview

A system that provides time-dependent context for music has several requirements. The system needs to be able to recognize audio or music being played in the environment of the user together with a precise time-offset. It also needs contextual, time-dependent information to provide to the user e.g. lyrics, scores, music video, tablature or triggers. The information should be prepared beforehand and stored in a repository. The system also needs an interface to provide the user with the information to enhance the listening experience. As a final soft requirement, the system should preferably minimize computational load and resources so it could be implemented on smartphones.

Acoustic fingerprinting algorithms are designed to recognize which music is playing in the environment. The algorithms use condensed representations of audio signals to identify short audio fragments in vast audio databases. A well-designed fingerprinting algorithm generates similar fingerprints for perceptually similar audio signals, even if there is a large difference in quality between the signals. Wang [18] describes such algorithm. The algorithm is based on pairs of spectral peaks which are hashed and compared with the peaks of reference audio. Wang's algorithm is able to recognize short audio fragments reliably even in the presence of background noise or artifacts introduced by A/D or D/A conversions. The algorithm is computationally efficient, relatively easy to implement and yields precise time-offsets. All these features combined make it a good algorithm to detect which music is being played and to determine the time-offset precisely. Figure 2 shows fingerprints extracted from two audio streams using Panako[14], an implementation of the aforementioned algorithm. The reference audio is in the top, the query in the bottom. Using fingerprints that match (in green), the query is aligned with the reference audio.

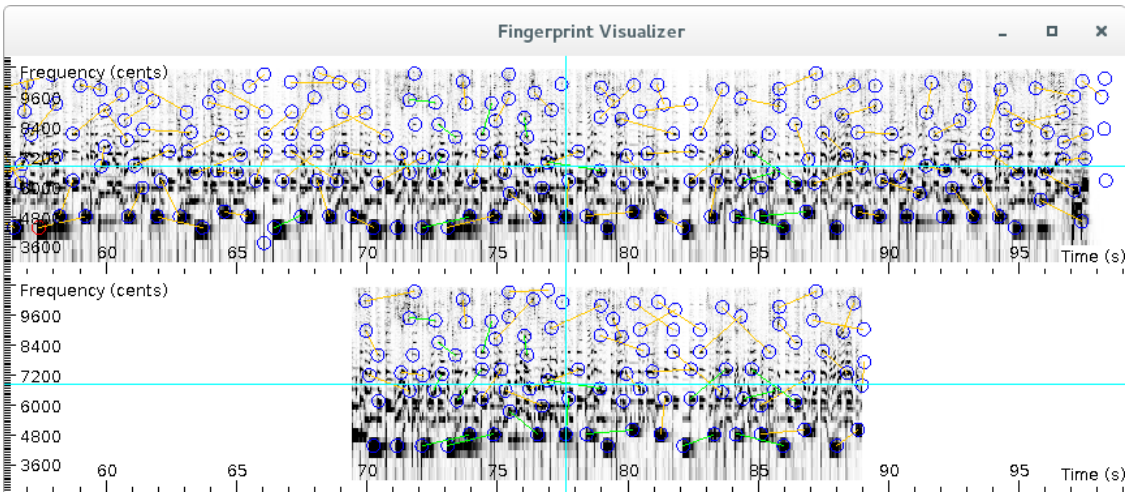

Figure 1: Two streams of audio with fingerprints marked. The reference audio is the stream on top, the bottom stream is the possibly noisy query. Some fingerprints are present in both streams (green) while others are not (yellow). Matching fingerprints have the same relative time offset in the query with respect to the reference.

For the prototype, the complete process is depicted in Figure 2. A client uses its microphone to register sounds in the environment. Next, fingerprints are extracted from the audio stream. The fingerprints are sent to a server. The server matches the fingerprints with a reference database. If a match is found, a detailed time-offset between the query and the reference audio is calculated. Subsequently, the server returns this time-offset together with a list of beat timestamps. Using this information the client is able to generate feedback events that coincide with the beat of the music playing in the users environment. This process is repeated to make sure that the feedback events remain in sync with the music in the room. If the server fails to find a match, the feedback events stop.

With the list of beat events available the system needs to generate feedback events. The specific feedback mode depends on the use case. Perhaps it suffices to accentuate the beat using an auditory signal: e.g. by a loud sound with a sharp attack. A bright flash on each beat could also help some users. Haptic feedback can be done with vibration motors attached to the wrist using a bracelet. A commercially available wireless tactile metronome - the Soundbrenner Pulse - lends itself well for this purpose. A combination of feedback modes could prove beneficial since multisensory feedback can improve sensorimotor synchronization[4, 12].

3 Evaluation

The evaluation makes clear how synchronized context can be delivered to ambient audio or music. The evaluation quantifies the time-offset between the beats - annotated beforehand - and the time of the feedback event that should correspond with a beat. For an evaluation of the underlying fingerprinting algorithm with respect to robustness against noise and digital-analog / analog-digital conversions readers are referred to [18].

The evaluation procedure is as follows: a device plays a piece of music. The device also knows and updates the current playback position accurately in real-time[2]. A client listens to this audio, extracts

[2]In a conventional computing environment this is not triv-

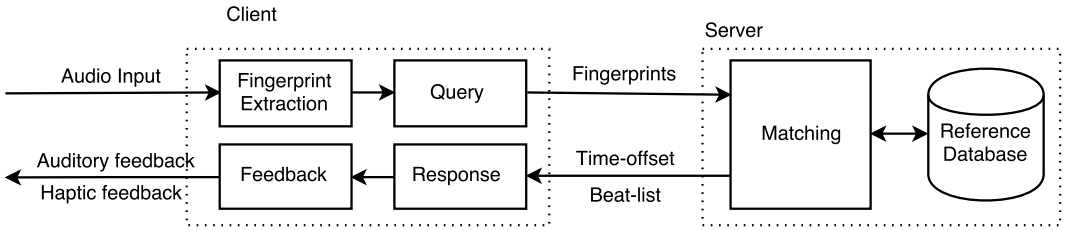

Figure 2: Schema of a client/server architecture to match an audio query and recieve synchronized feedback on the beat.

fingerprints and sends these to a server (see 2). If the server finds a match, a time-offset is returned together with a list of beat events. The client uses the beat-list and offset to send events that are in sync with the beat to the playback device. The playback device responds with an accurate representation of the current playback position. Finally, the reported playback position is compared with the beat-list. The difference between the beats and the feedback events are used as a synchronicity measure. Note that when the beats are incorrectly labeled and do not fall on the actual beat of the music, the evaluation remains correct. Only the difference between the time of the feedback event and the labeled event is evaluated, not the time of the label (beat) itself. Since we are primarily interested in how precise in time feedback can be given this suffices.

To counter problems arising with soft real-time thread scheduling and audio output latency the evaluation was done using a microcontroller with a hard real-time scheduler and low-latency audio playback. An extra benefit is that timing measurements are very precise. Cheap, easily programmable microcontrollers come with enough computing power these days to handle high-quality audio. One such device is the Axoloti, a microcontroller for audio applications

ial. Audio travels through layers of software abstractions or mixers before it arrives at the audio hardware output. At each point it can potentially be buffered. The time reported by a software audio player and the actual audio being played by the speakers differs by the total audio output latency which quickly amounts to over 20ms and is only constant if carefully configured.

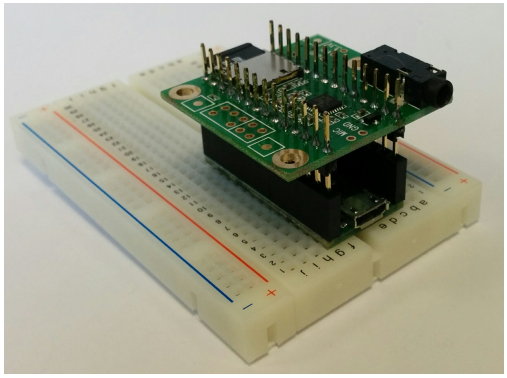

Figure 3: Teensy with Audio Board. The microcontroller is capable of high-quality low-latency audio playback from an SD-card and precise timing measurement.

that can be programmed using a patcher environment. Another is the Teensy equipped with a Teensy Audio Board. Here we use a Teensy 3.2 with Audio Board for audio playback. It supports the Arduino environment which makes it easy to program it as a measuring device. It has an audio-output latency of about 1 ms. It is able to render 16 bit audio sampled at 44100 Hz and is able to read PCM-encoded audio from an SD-card. In the experimental setup, the Teensy is connected to a Behringer B2031 active speaker.

Audio enters the client by means of the built in

BPM	Artist - Title
82	Arctic Monkeys - *Brianstorm*
87	Pendulum - *Propane Nightmares*
90	Ratatat - *Mirando*
91	C2C - *Arcades*
95	Hotei - *Battle Without Honor or Humanity*
95	Skunk Anansie - *Weak*
100	Really Slow Motion - *The Wild Card*
105	Muse - *Panic Station*
108	P.O.D. - *Alive*
111	Billie - *Give Me the Knife*
121	Daft Punk - *Around The World*
128	Paul Kalkbrenner - *Das Gezabel de Luxe*
144	Panda Dub - *Purple Trip*
146	Digicult - *Out Of This World*
153	Rage Against the Machine - *Bombtrack*
162	Pharrell Williams - *Happy*

Table 1: The dataset used during the evaluation has a wide BPM range and a stable easy to follow audible beat.

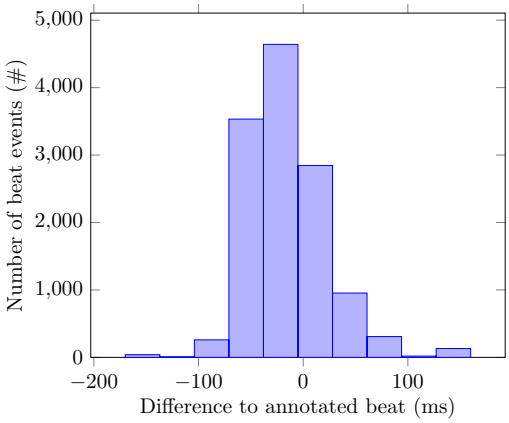

Figure 4: The difference (in ms) between feedback events and beats. The feedback events arrive 16ms before the beat to allow to perceive beats together with feedback. The data is centered around the average and bins of 32ms are used, the precision of the system.

microphone. The client part of Figure 2 is handled by a laptop: a late 2010 Macbook Air, model A1369 running Mac OS X 10.10. The laptop was placed two meters from the speaker in a reasonably quite office environment[3]. Next the audio is fed into the fingerprinting system. The system presented here is based on Panako [14]. The source of Panako is available on-line and it is implemented in Java 1.8. Panako was modified to allow a client/server architecture. The client is responsible for extracting fingerprints. A server matches fingerprints and computes time offsets. The server also stores a large reference database with fingerprints together with beat positions. The client and server communicate via a web-service using a JSON protocol. JSON is a standard that describes a way to encode data, it allows communication between computers.

When the time-offsets and the beat list are available on the client feedback events are generated. To evaluate the system the feedback events are sent over a USB-serial port to the Teensy. The Teensy replies over the serial port with the current playback position of the audio. The playback position is compared with the expected position of the beat and the difference is reported in milliseconds. Negative values mean that the feedback-event came before the actual audible beat, whereas positive values mean the opposite. The system is tested using a the data set presented in Table 1. It features music in a broad BPM range with a clear, relatively stable beat.

The results are depicted in Figure 4. The system responds on average 16 ms before the beat. This allows feedback events to be perceived together with the beat. Depending on the tempo (BPM) of the music and type of feedback it might be needed to schedule events later or even sooner. This can be done by adapting the latency parameter which modifies the timing of the scheduled feedback events. However, there is a large standard deviation of 42 ms. The current system is limited to an accuracy of 32 ms:

[3]An analysis on how the fingerprinting algorithm handles noise falls outside the scope of this article. Consult [18] for such analysis.

the size of a block of 256 audio samples, sampled at 8000 Hz. The block size used in the fingerprinter. In Figure 4 each histogram bin is 32 ms wide and centered around -16 ms. The results show that the system is able to recognize the correct audio block but is sometimes one block off. The main issue here is the unpredictable nature of scheduling in Java: Java threads are not guaranteed to start with predictable millisecond accurate timing. Garbage collection can cause even larger delays. Larger delays are due to incorrectly recognized audio. Repetition in music can cause the algorithm to return an incorrect absolute offset which makes the beat drift. The results, however, do show that the concept of the system is very promising and can deliver timing dependent context.

In its current state the system listens to 12 seconds of audio and sends the fingerprints of those 12 seconds to the server. The state is reset after that. The system does not employ use-case dependent heuristics. If it is known beforehand that the user will most likely listen to full tracks the current state, time-offset and beat-lists could be reused intelligently to improve accuracy, especially in the case of repeating audio. This could also be optimized by running a real-time onset detector and correlating the detected onsets list with the beat list returned by the server, this would however make the system more computationally expensive.

5 Discussion

The proposed system only supports recorded music. Supporting live music is challenging but could be done. The MATCH algorithm[3] for example supports tracking live performances in real time via dynamic time warping. The basic song structure however needs to be kept intact during the live rendition, otherwise the pre-computed contextual data becomes useless. Another challenge is DJ-sets. Although recorded music is used, during DJ-sets the tempo of the music is often modified to match a previous piece of music. To support such situations a more involved acoustic fingerprinting algorithm is needed. Currently there are two algorithms described in the literature that report both time-offsets and tempo modifications accurately[15, 14].

Repetition is inherent in music. Especially in electronic music the same exact audio can be repeated several times. A fingerprinting algorithm that only uses a short audio excerpt could, in such cases, return an incorrect absolute offset. To alleviate this problem, context could also be taken into account. Also the type of data returned needs to be considered. Lyrics could be incorrect while tablature or beats could still be aligned correctly since they do not depend as much on an absolute offset.

Since the system uses a computationally inexpensive algorithm, it can be executed on a smartphone. The implementation used here is compatible with Android since it depends only on two Android compatible libraries[14, 13]. If only a small reference music library is used, all server components of the system could be moved to the smartphone. An app that offers aligned music videos for the music for one album could run all components easily on a smartphone without the need for an external server.

For the prototype a database with pre-computed beat-position is created *off-line* using all the acoustic information of the song. However, it is possible to determine beat positions with a real-time beat tracking algorithm[6]. Unfortunately, this poses several problems. Beat-tracking involves an important predictive and reflective element. To correctly model beats based on a list of onsets extracted in real-time, musical context is needed. This context may simply not be available. Another issue is that the computational load for a beat-tracking systems is often high. In [10] there is an overview of beat tracking techniques which are challenging to implement on smartphones. A third problem is that feedback needs to be provided before the actual acoustic beat is perceived by the user. Tactile feedback e.g. takes around 35 ms to be processed[4]. Feedback based on a real-time beat-tracker - which introduces a latency by itself - would be always late. Generating feedback based on real-time beat-tracking algorithms is impractical especially in the context of smartphones with low-quality microphones and restricted computational resources.

To further develop the prototype into an assistive technology, more fundamental research is needed to pinpoint the optimal type of feedback for user groups.

The early deafened late implanted CI user group is recognized as an *'understudied clinical population'*[5] for which models on auditory rhythm perception are underdeveloped. Insights into tactile or multi-modal rhythm perception for this specific group seem to be lacking from the academic literature. There is however a study that suggests that multi-sensory cues improve sensorimotor synchronization[4]. In Sowinski et al. (2013) [16] another fundamental issue is raised. In the study two participants seem to be able to perceive small timing deviations in audio but are unable to move accordingly. As the authors put it *"This mismatch of perception and action points toward disrupted auditory-motor mapping as the key impairment accounting for poor synchronization to the beat"*. The question remains whether this holds for tactile-motor mappings especially in the late implanted CI user-group.

6 Conclusion

A system was described that employs acoustic fingerprinting techniques to provide augmented music listening experience. A prototype was developed that provides feedback synchronized with music being played in the environment. The system needs a dataset with fingerprints from reference audio and pre-computed beat-lists. Since it offers fine-grained context awareness the systems design can be used to also show lyrics, scores, visuals, aligned music videos or other meta-data that enrich the listening experience. The system could also be used to trigger events linked to audio during e.g. a theater performance.

7 Acknowledgment

This research was funded by a Methusalem grant of the Flemish government.

References

[1] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.

[2] D. Bogdanov, Nicolas Wack, Emilia Gómez, Sankalp Gulati, P. Herrera, O. Mayor, G. Roma, J. Salamon, J. Zapata, and Xavier Serra. ESSENTIA: an Audio Analysis Library for Music Information Retrieval. In *Proceedings of the 14th International Symposium on Music Information Retrieval (ISMIR 2013)*, pages 493–498, Curitiba, Brazil, 04/11/2013 2013.

[3] Simon Dixon and Gerhard Widmer. Match: A music alignment tool chest. In *Proceedings of the 6th International Symposium on Music Information Retrieval (ISMIR 2005)*, pages 492–497, 2005.

[4] Mark T Elliott, AM Wing, and AE Welchman. Multisensory cues improve sensorimotor synchronisation. *European Journal of Neuroscience*, 31(10):1828–1835, 2010.

[5] Christina Fuller, Lisa Mallinckrodt, Bert Maat, Deniz Baskent, and Rolien Free. Music and quality of life in early-deafened late-implanted adult cochlear implant users. *Otology & Neurotology*, 34(6):1041–1047, 2013.

[6] Masataka Goto. An audio-based real-time beat tracking system for music with or without drum-sounds. *Journal of New Music Research*, 30(2):159–171, 2001.

[7] Erin E Hannon and Sandra E Trehub. Tuning in to musical rhythms: Infants learn more readily than adults. *Proceedings of the National Academy of Sciences of the United States of America*, 102(35):12639–12643, 2005.

[8] Marie-Andrée Lebrun, Patricia Moreau, Andréane McNally-Gagnon, Genevieve Mignault Goulet, and Isabelle Peretz. Congenital amusia in childhood: a case study. *cortex*, 48(6):683–688, 2012.

[9] Hugh J McDermott. Music perception with cochlear implants: a review. *Trends in amplification*, 8(2):49–82, 2004.

[10] Meinard Müller. *Fundamentals of Music Processing - Audio, Analysis, Algorithms, Applications*. Springer, 2015.

[11] Jessica Phillips-Silver, Petri Toiviainen, Nathalie Gosselin, Olivier Piché, Sylvie Nozaradan, Caroline Palmer, and Isabelle Peretz. Born to dance but beat deaf: a new form of congenital amusia. *Neuropsychologia*, 49(5):961–969, 2011.

[12] Joren Six, Laura Arens, Hade Demoor, Thomas Kint, and Marc Leman. Regularity and asynchrony when tapping to tactile, auditory and combined pulses. In *Proceedings of ESCOM 2017*, 2017.

[13] Joren Six, Olmo Cornelis, and Marc Leman. TarsosDSP, a Real-Time Audio Processing Framework in Java. In *Proceedings of the 53rd AES Conference (AES 53rd)*. The Audio Engineering Society, 2014.

[14] Joren Six and Marc Leman. Panako - A Scalable Acoustic Fingerprinting System Handling Time-Scale and Pitch Modification. In *Proceedings of the 15th ISMIR Conference (ISMIR 2014)*, 2014.

[15] R. Sonnleitner and G. Widmer. Robust quad-based audio fingerprinting. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, PP(99):1–1, 2016.

[16] Jakub Sowiński and Simone Dalla Bella. Poor synchronization to the beat may result from deficient auditory-motor mapping. *Neuropsychologia*, 51(10):1952–1963, 2013.

[17] Lydia Timm, Peter Vuust, Elvira Brattico, Deepashri Agrawal, Stefan Debener, Andreas Büchner, Reinhard Dengler, and Matthias Wittfoth. Residual neural processing of musical sound features in adult cochlear implant users. *Frontiers in human neuroscience*, 8, 2014.

[18] Avery L. Wang. An Industrial-Strength Audio Search Algorithm. In *Proceedings of the 4th International Symposium on Music Information Retrieval (ISMIR 2003)*, pages 7–13, 2003.