TEMPO DETECTION OF URBAN MUSIC USING TATUM GRID NON-NEGATIVE MATRIX FACTORIZATION

Daniel Gärtner
Fraunhofer Institute for Media Technology IDMT
daniel.gaertner@idmt.fraunhofer.de

ABSTRACT

High tempo detection accuracies have been reported for the analysis of percussive, constant-tempo, Western music audio signals. As a consequence, active research in the tempo detection domain has been shifted to yet open tasks like tempo analysis of non-percussive, expressive, or non-western music. Also, tempo detection is included in a large range of music-related software. In DJ software, features like beat-synching or tempo-synchronized sound effects are widely accepted in the DJ community, and their users rely on correct tempo hypothesis as their basis. In this paper, we are evaluating both academic and commercial tempo detection systems on a typical dataset of an urban club music DJ. Based on this evaluation, we identify octave errors as a problem that has not yet been solved. Further, an approach based on non-negative matrix factorization is presented. In its current state it can compete with the state of the art. It further provides a foundation to tackle the octave error issue in future research.

1. INTRODUCTION

Tempo detection on percussive music with constant tempo has been extensively investigated by the music information retrieval community throughout the last 30 years, and high accuracies have been reported. Therefore, researchers have moved on to related tasks like tempo detection of non-percussive music, dealing with soft onsets, or tempo/beat tracking of expressive performances, that are more difficult to analyze correctly.

Several comparative evaluations of tempo detection algorithms have been published. The results of the ISMIR tempo induction contest of 2004 are summarized in [9]. In a recent study [19], another 12 algorithms are investigated. In both studies, [11] outperforms the competing algorithms in most of the cases. Another comparative study is presented in [14], where algorithms of seven groups are analyzed. It is shown, that the genre has an effect on the tempo detection performance. Also, algorithms performed quite differently within different tempo ranges. Furthermore, some algorithms performed much worse on songs with ternary meter compared to songs with binary meter, while in general, percussive music returned higher scores than non-percussive music.

To motivate our work, a pre-study has been conducted, in which several algorithms have been evaluated on a urban club music dataset of 1000 songs size (more details can be found in Section 3). This study revealed, that the leading academic tempo-detection algorithms reach up to 70% of accuracy on urban club music, which is less than expected (100% on reggae, soul, and rap are reported in [1], [8] achieved over 95% on constant rock and pop music).

Several metrics are commonly used for tempo detection evaluation. The fraction of songs, for which the tempo has been correctly identified, is an intuitive measure. Often, an additional metric is used [9, 19], in which tempo estimates that are an integer multiple or divisor of the ground-truth tempo are also counted as correct estimates. This metric is motivated by the fact that even human listeners will not agree on a single tempo (another approach dealing with this fact is the metric used in [14]). This is surely true for a large quantity of music from several styles. However, as reported in [13], there is a high agreement in tempo perception of urban club music amongst listeners, and it can be assumed that the agreement is even higher amongst urban club music DJs, since it is their job to mix songs with the same tempo. However, this assumption remains to be proven.

From the perspective of the user of a DJ software, it is absolutely mandatory that the tempo is annotated correctly. The so called octave errors are unacceptable. Although not really related to the origin of the word octave in music, they refer to tempo estimates that are different from the correct tempo by a factor that is a power of two. Songs need to have the same tempo in order to be mixed with clean transitions, which is requested by the dancing audience. Further, many audio effects also rely on correct tempo hypotheses. And of course, additional processing like beat tracking, which can be used for automatic mixing, also strongly depends on a correct tempo estimation.

[13] shows that there is a significant effect of music genre on the most salient tempo, which is consolidated in [16], where a style detection method is used to improve tempo detection of ballroom dance songs. Further, [4] lists several cues that beat-tracking might profit from a style-specific analysis.

Besides a limited tempo range of about 60 to 140 bpm, urban club music is percussive, has a constant tempo, and is composed of repeating drum patterns. Further, the drums are often sampled and triggered by a sequencer. This reduces tempo fluctuations and the variety in sound of multiple occurrences of a certain drum.

The remainder of this paper is organized as follows. In Section 2, the proposed system is explained, followed by the experimental setup and results section (3). The paper concludes with a summary and outlook in Section 4.

2. APPROACH

As [3] states, ”Perceptually, musical metric structure comprises beats and hierarchies. Beats constitute the framework in which successive musical events are perceived”. The number of beats per minute is used to quantify the tempo of a musical piece, the beat grid specifies the position of the beats in a musical piece. The beat grid can be further subdivided, leading to the tatum grid (or just tatum), which is the ”lowest regular pulse train that a listener intuitively infers from the timing of perceived musical events” [10]. Also, beats can be grouped in bars. In urban music, a bar mostly consists of four beats. On each metrical level we will use the term grid to address the positions of the pulses, and the term period for the time or frame interval between two successive pulses.

An overview of our approach is shown in Figure 1. From the spectrogram of the audio data, quantized event bands are calculated from which the bar period is estimated, that directly leads to the tempo hypothesis. In order to extract the quantized event bands, a tatum tracker is applied to an onset detection function. Next, the spectrogram is sampled at tatum grid positions and then factorized in event bands. This is done by means of non-negative matrix factorization (NMF [12]), aiming at also isolating different drum classes (bass drum, snare drum) in separate bands. Calculating NMF on just a subset of spectrogram frames is a major difference to our approach in [7].

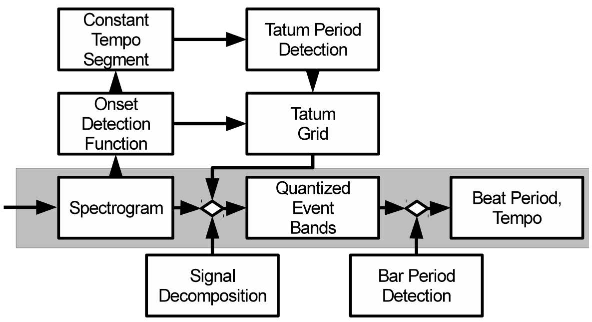

Figure 1. System Overview.

2.1 Onset Detection Function

At the beginning of the analysis, events that are supposed to contribute to the perception of beats in the musical piece are determined. As this study focuses on urban music, percussive drum events are investigated, and an onset detection function that is sensitive to percussive events is chosen. An absolute spectrogram S is calculated from the 44.1 kHz audio input data, using a window-size of 4096 samples size with a hop-size of 512 samples, which corresponds to a spectrogram sample rate of 86.1 Hz.

From S, an onset detection function d_o is calculated. The onset components detection function developed in [8] is used as onset detection function.

2.2 Constant Tempo Segment

In this component, the longest segment with approximately constant tempo is identified. Analyzing audio data with several distinct tempi will harm the tatum period detection.

d_o is convolved with a set of 601 comb grids, which correspond to pulse trains of 20 s length from quarter notes between 40 bpm and 640 bpm (sixteenth notes at 160 bpm). Next, each convolved function is cut into segments of 1 s length, and the maximum value inside each segment is stored in a so called comb response matrix.

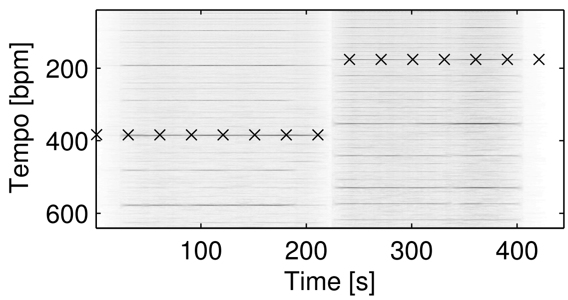

Figure 2. Comb response matrix of a song with multiple tempi, including the Viterbi path.

An example of a comb response matrix is shown in Figure 2. The tempo is constant over the first half of the song, and then changes to another tempo. A strong response for the grid corresponding to about 380 bpm, which equals a pulse train of 16th notes at 95 bpm, can be observed in the first half. The second half is a little slower.

Next, the entries in the comb response matrix are normalized to unit sum for each column. Now, each entry in the matrix can be seen as likelihood for observing a bpm class at a given time in a hidden Markov model (HMM). The Viterbi algorithm [18] is then used to track the most likely path through the comb response matrix of observations. From the path, the segment where the tempo stays roughly constant for the longest time is determined and further investigated during tatum period detection.

2.3 Tatum Period Detection

Since d_o is an onset detection function sensitive to percussive events, it can be used for tatum period detection in the following way. An accent function g_o is calculated from the constant-tempo excerpt of d_o by keeping only the local maxima of d_o while setting all other values to zero, and convolving the resulting signal with a hann window (11 samples width). The tatum period detection function g_t is calculated: $g_t = R(R(g_o))$, where $R(f)$ is the autocorrelation of function f. The first 400 values of g_t can be seen in Figure 3. Now, for each local maximum position on the lag

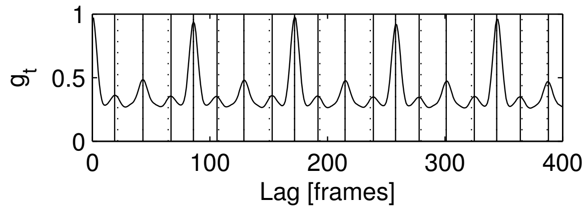

Figure 3. The tatum period detection function g_t. Values close to 1 indicate a high periodicity with a period of the corresponding lag axis index. Solid vertical lines denote local maxima positions, dotted vertical lines denote multiples of the tatum period.

axis as tatum period candidate c_i, we determine the smallest multiple m_i of c_i that does not correspond to a local maximum in g_t. The tatum period candidate c_i with the largest m_i is chosen as tatum period if it is also the local maximum with the smallest lag. Otherwise, it's lag value is divided by 2 until it is smaller than 14, which still allows tracking sixteenth notes at 180 bpm. This additional step is required, since not even all the rhythmic events necessarily fall exactly on the tatum grid. If a certain drum event in a bar is systematically played early or late, this will lead to slightly shifted locations of local maxima in g_t. This can also be observed in Figure 3, where the odd local maxima (solid lines) do not exactly overlap with the tatum period and its multiples (dotted lines).

2.4 Tatum Grid

After having determined the tatum period, the onset detection function d_o is convolved with a comb grid where the combs are tatum period spaced. This will strengthen accents in d_o that lie on the not yet determined tatum grid, and extenuate all other accents. Now, all dominant local maxima positions and their pairwise differences are determined. Only those pairs, for which the difference is approximately 1, 2, or 3 times the tatum period, are kept. Then, dynamic programming [2] is used to find the longest path over the remaining pairs. Using this approach, variations around a center tempo can be compensated. However, if a song contains multiple different tempi, the path will only cover the region that has a beat period that is approximately a multiple of the tatum period hypothesis. In [6], dynamic programming is used for beat tracking. The approaches differ mostly in the fact that we only allow local maxima in the underlying detection function as potential tatum grid anchors.

2.5 Signal Decomposition

The determined tatum grid positions are refined to the locations of close local maxima in d_o. This way, also early and late played events can be incorporated. Then, the spectrogram is subsampled at the tatum grid positions, and the resulting matrix is factorized using NMF [12]. In NMF, a non-negative matrix V is factorized in matrix factors W and H, $V \approx WH$. Applied to an absolute spectrogram, it is factorized in a set of components where the characteristic spectrum of each component is stored in W and the activation of each component is stored in H. From factorization, we expect to separate different drum classes in different bands, and also separate additional, sources in additional bands (e.g., separate bass drum and bass even though they might be overlapping in the spectrogram). Since the drum tracks in urban club music are often generated using drum machines, samplers, and sequencers, NMF seems to be a good choice for decomposition, since drum sounds are not supposed to vary over time. In [7] we observed better results in tempo recognition on urban music using NMF based decomposition compared to a filter bank approach.

NMF is used to factorize the subsampled spectrogram in 24 components. Both bases and activations are randomly initialized, and 50 iterations of multiplicative update rules using Kullback-Leibler divergence are performed. These parameters have not yet been quantitatively optimized in any way.

NMF has been shown to be able to separate drum events in, e.g., [15], where NMF is used for drum transcription in polyphonic music.

Factorizing a spectrogram that is subsampled at tatum grid positions instead of the full spectrogram reduces the computational load. A more general advantage of working on a tatum grid level lies in removing tempo variations.

However there are disadvantages as well. The decomposition is dependent on the quality of the tatum grid. If the tatum grid does not contain the drum event onsets, the drums will not be analyzed at all. Further, in cases where different drum events fall, for example, on the same beat but at slightly different times, the spectral frame that is meant to capture these events might not capture them all.

2.6 Bar Period Detection

The 24 bands obtained by the NMF analysis are again analyzed using a comb grid. Each of the bands is convolved with a set of comb grid with combs spaced 1, 2, ... , n, where n corresponds to the largest possible distance between 4 beats at 65 bpm (see Section 3.5.2), considering the underlying tatum grid. For each band, from the corresponding n filtered functions, the index of the one with the largest variance is determined.

From all 24 collected indices, a histogram is calculated and the most frequent index is chosen as tatum-period-to-bar-period factor z. The bar period is calculated by multiplying Z and the tatum period. As final steps, the determined bar period is first divided by four to retrieve the beat period, since 4/4 is the most common measure in urban club music. As a final step, the retrieved beat period is transformed in the bpm range of 65 bpm to 130 bpm (see Section 3.5.2) by doubling or halving.

3. EVALUATION

In this section we describe the experiments, present the results, and discuss them.

3.1 Measures

For each evaluation run, we report the relative number of songs with a tempo hypothesis that differs less then 4% from the ground-truth tempo (Acc1). To gather further insights on the performance of the evaluated algorithms, we further report the relative number of songs with two times or three times the correct tempo (Acc2), and one half or one third of the correct tempo (Acc3). The fraction of the remaining songs is denoted Acc4.

3.2 Dataset

2324 songs have been collected from an urban music promotion platform exclusively for DJs. Labels provide songs to DJs over this kind of platforms at no charge, in return the DJs will help to promote these songs and make them popular by playing them in their sets in the club. It is an authentic set that well represents the kind of music, urban club music DJs are working with. Each of the songs has been tempo-annotated by an experienced urban music DJ. The set has been randomly split into a development set (1000 songs, denoted *dev*) and a test set (1324 songs, denoted *test*).

Dev has been used for the development of the algorithm, which includes the design of the components and parameter setting. The final algorithm has then been evaluated using *test*, the results for *dev* are also reported.

Figure 4 shows a histogram over the observed tempi in *dev*.

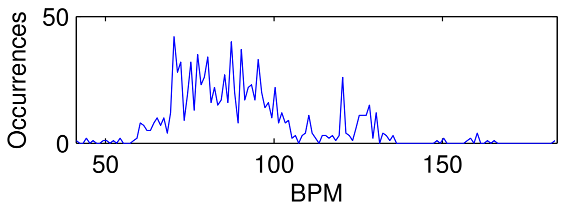

Figure 4. Histogram over the ground truth tempi in *dev*

Although it has been argued, that urban club music in general is strongly percussive and songs have a constant tempo, there are exceptions. A few songs in the dataset do not contain any or only soft percussions. Some are played by live bands (e.g., The Roots) which leads to varying tempo. Another source for varying tempo is the use of sampling in the music, where the samples vary according to their tempo. And there are even a few songs that are cut together in a way that the segments are not concatenated on beat, or even the tempo is different for the segments. Further, a few songs in the database do not belong to the urban music genre, like some pop rock songs, that are also responsible for the upper tempo outliers in Figure 4. We decided to keep them in the database, since we wanted to keep it exactly the way it was obtained. However, genre detection algorithms could be used to identify songs like that prior to tempo analysis.

3.3 Preprocessing

The data consists of MP3 files of different sample rates and bit rates. FFmpeg has been used to convert the files to 44.1 kHz mono PCM wav files.

3.4 Algorithms

The approaches from Dixon [5], Ellis [6], and Klapuri [11] have been selected to represent the academic systems. [6] returns two tempo hypotheses, from which the stronger one is selected. [11], and [5] return a beat grid. We calculated histograms over inter-beat-intervals, and then determined the mean of all inter-beat-intervals that contributes to the most frequent inter-beat-interval, from which the tempo in bpm can be derived. All implementations were obtained from the authors.

In addition, several commercial DJ systems have been evaluated. Each one offers ways to parameterize the tempo analysis algorithm. Cross 1.7.0, denoted Cross [1], offers three different tempo ranges for bpm analysis, each one covering exactly one octave. 75-150 has been selected for analysis. Scratch Live 2.4.1, denoted SL [2], offers five different tempo ranges for analysis, each one covering one octave. 68 - 135 has been selected for analysis. Torq 2.0.3, denoted Torq [3], offers several genre-specific tempo ranges. In our experiments, the default settings have been used, returning bpm values from 60 to 160 bpm. Traktor Pro 2.6.0, denoted TraA+B [4], offers 9 different octaves, from which 68-135 (TraB) has chosen. Further, Traktor also offers a single range covering more than a octave (60-200), which will be denoted TraA. Virtual DJ Home 7.0.5, denoted VDJ, offers an option to allow also bpm values smaller than 80 bpm, which was activated. It returned tempi between 60 and 170 bpm. It is worth noticing, that some of the investigated tools only offer tempo ranges of exactly one octave, which is a simple but (as can be seen in the Section 3.5) working approach to reduce octave errors at least for urban club music, since a large amount of urban club music is located inside a single octave.

3.5 Results

In this section the results of the conducted experiments are presented and discussed. All experiments have been performed in Matlab.

3.5.1 Evaluation of the reference systems

Table 1 lists the results returned from the evaluation of the state of the art.

Directly comparing the results for *dev* and *test*, one can see that the performances are similar, which indicates that both sets are comparably difficult. This is also true for the

[1] http://www.mixvibes.com
[2] http://serato.com/scratchlive
[3] http://www.torq-dj.com/
[4] http://www.native-instruments.com/traktor

	Acc1 1	Acc2 2+3	Acc3 1/2+1/3	Acc4 other
Cross	73.2/75.6	23.2/22.9	1.2/0.5	2.4/1.0
SSL	89.4/89.4	8.2/8.5	1.3/1.2	1.1/1.0
Torq	85.6/84.3	2.9/4.4	4.3/3.4	7.2/7.9
VDJ	81.0/78.5	17.0/20.4	1.1/0.9	0.9/0.2
TraA	77.6/79.1	15.3/14.7	5.1/4.5	2.0/1.7
TraB	90.3/90.7	6.1/6.2	1.5/1.4	2.1/1.7
Dixon	25.3/24.5	69.8/70.1	0.0/0.0	4.9/5.4
Ellis	57.5/51.5	4.5/5.4	19.3/26.5	18.7/16.5
Klapuri	68.7/71.7	28.8/27.2	1.8/0.8	0.7/0.3

Table 1. Results for the state of the art algorithms on *dev* / *test*, accuracies in %.

results of the proposed approach, listed in Table 2, which shows, that even though it has been optimized using *dev* it still generalizes well.

With an accuracy of 73.2%, Cross is the worst of the commercial algorithms. This is mainly caused by the limited choices of the tempo range, of which no one really fits our data well. Since all commercial algorithms do a pretty good job in choosing the correct tempo (as long as octave errors are still accepted as correct), the performance mainly depends on the prior choice of the bpm-analyzing octave. Traktor offers both a large tempo range (60-200, TraA) and a suitable octave tempo range (68-135, TraB). TraB has an accuracy of about 12.7% higher than TraA, which shows, that automatically picking the right tempo octave is still an open issue.

Dixon often returns twice the correct tempo (69.8%) and could be simply tuned by halving the returned tempo estimate. For Ellis, almost 20% of the songs in *dev* are neither correct nor do they belong to one of the octave error classes. The most common cases in "other" of Ellis are 4/3 (15%) and 2/3 (4%). In accordance with the mentioned MIREX benchmarks, Klapuri is the best performing academic algorithm.

In the first category (Acc1), the commercial approaches outperform the academic ones. In this category, the commercial approaches can benefit by the fact that most of the data is located in a single octave. However, in the fourth category (other), Klapuri's algorithm is among the best.

3.5.2 *Best Tempo Octave*

Based on the data from *dev*, an experiment has been conducted to determine the best tempo octave settings, assuming that an algorithm performs perfectly but transforms its results in a specified tempo range of exactly one octave by doubling or halving the tempo several times. The accuracy depending on the given tempo range is plotted in Figure 5. The best performance (92.2%) is achieved for a bpm range of 65 - 130 bpm. Therefore, in the presented approach, a tempo hypothesis is transformed in this range by doubling and halving.

3.5.3 *Evaluation of the proposed approach*

Table 2 contains the results for the evaluation of the presented algorithm. For both *dev* and *test*, the algorithm returns the highest accuracies retrieved in the whole study.

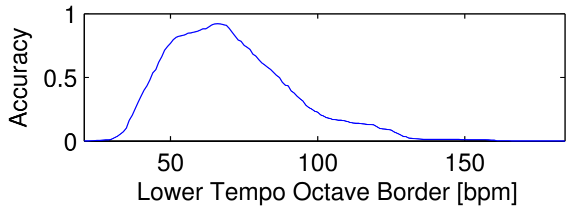

Figure 5. Accuracy of a perfect algorithm with octave restriction depending on the allowed range.

	Acc1 1	Acc2 2+3	Acc3 1/2+1/3	Acc4 other
Own rel.	91.9/92.5	4.9/4.5	2.9/2.7	0.3/0.2
Own abs.	919/1225	49/60	29/36	3/3

Table 2. Results for the presented algorithm on *dev*/*test*, accuracies in %, and absolute number of songs in each category.

For both *dev* and *test*, only three songs return a tempo that is not 2^n times the correct tempo. The three failing songs in *dev* all have a 3/4 measure, but a 4/4 measure is assumed when going from bar period to beat period. Two of the failing songs in *test* have no or almost no drum tracks, the third one has also a measure of only 3 beats length.

The chosen tempo range (that has been determined from the bpm distribution in *dev*) induces the assignment to one of the accuracies Acc1, Acc2 and Acc3. An Acc1 of 92.5% confirms the choice of the bpm range.

	1	2	3	4	6	8	9	12	other
dev	1	188	8	761	6	32	1	3	0
test	0	279	7	992	5	41	0	0	0

Table 3. Distribution of the songs in tatum classes.

Table 3 shows the performance of the tatum period detection component. The songs are differentiated into tatum classes, where the class name denotes the ratio of ground truth beat period and determined tatum period. For all instances, the beat period is an multiple of the tatum period. The most common multiple is 4, which corresponds to 16th notes. Assuming 16th notes as tatum for each song, and therefore estimating the beat period as four times the tatum period, the tempo could be correctly determined for 76.1% (*dev*) and 74.9% (*test*) of the songs without making any assumptions on a bpm range, which outperforms any of the academic approaches.

The distribution of the songs in bar classes is listed in Table 4. For both sets, for most of the songs a bar-length of 4 beats is returned. Assuming the determined bar period to be four times the beat period, the tempo could be determined correctly for 81.5% (*dev*) and 82.0% (*test*) respectively, without making any assumptions on a bpm range, which again outperforms any of the academic approaches.

Both tables reveal the strength of the algorithm, but at the same time also show limits regarding to octave determination, as also shown in [7].

	1	2	3	4	6	8	other
dev	4	166	3	815	0	12	0
test	0	214	0	1085	1	22	2

Table 4. Distribution of the songs in bar classes.

4. CONCLUSION AND OUTLOOK

In this paper, it is shown that finding the correct octave is still an issue for even urban club music. This claim is consolidated by evaluating several academic and commercial tempo detection algorithms on a urban club music data set. The presented algorithm, developed specifically for urban music, outperforms all the other algorithms evaluated in this study, estimating the correct tempo for 92.5% of the *test* set. The remaining songs are a 2^n multiple of the correct tempo except for 3 out of 1324 songs.

Therefore, the proposed algorithm provides a good basis for further processing, in which the correct octave has to be determined. In the current approach, all tempo values are forced to be in the range of 65 to 130 bpm. Further experiments will be conducted where the octave is chosen based on the musical structure. A first investigation on the tatum-quantized activations indicated, that they still capture the dominant drum events, contributing to the rhythm of a song. Therefore, drum pattern analysis, as as performed in, e.g., [17] could be carried out on the activations, and then be incorporated in tempo detection. Finding the characteristic drum pattern of a song also offers additional opportunities like drum pattern similarity or urban music sub-genre classification.

5. ACKNOWLEDGEMENT

We want to thank Simon Dixon, Daniel P.W. Ellis, and Anssi Klapuri for providing implementations of their algorithms.

6. REFERENCES

[1] Miguel Alonso, Bertrand David, and Gael Richard. Tempo and beat estimation of musical signals. In *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR)*, 2004.

[2] Richard Ernest Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[3] Jeff A. Bilmes. *Timing is of essence*. PhD thesis, Massachusetts Institute of Technology, 1993.

[4] Nick Collins. Towards a style-specific basis for computational beat tracking. In *Proceedings of the 9th International Conference on Music Perception & Cognition*, 2006.

[5] Simon Dixon. Evaluation of the audio beat tracking system beatroot. *Journal of New Music Research*, (36), 2007.

[6] Daniel P. W. Ellis. Beat tracking by dynamic programming. *Journal of New Music Research*, 36(1):51–60, 2007.

[7] Daniel Gärtner. Tempo estimation from urban music using non-negative matrix factorization. In *Proceedings of the 42th AES International Conference*, pages 208–215, Ilmenau, 2011.

[8] Masataka Goto. A real-time beat tracking system for audio signals. In *Proceedings of the International Computer Music Conference*, 1995.

[9] Fabien Gouyon, Anssi Klapuri, Simon Dixon, Miguel Alonso, George Tzanetakis, Christian Uhle, and Pedro Cano. An experimental comparison of audio tempo induction algorithms. *IEEE Transactions on Speech and Audio Processing*, 14:1832–1844, 2006.

[10] Tristan Jehan. *Creating Music by Listening*. PhD thesis, Massachusetts Institute of Technology, 2005.

[11] Anssi Klapuri, Antti Eronen, and J. T. Astola. Analysis of the meter of acoustic musical signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1):342–355, 2006.

[12] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. *Advances in neural information processing systems*, 2001.

[13] Martin F. McKinney and Dirk Moelants. Ambiguity in tempo perception: What draws listeners to different metrical levels? *Music Perception: An Interdisciplinary Journal*, 24(2):155–266, 2006.

[14] Martin F. McKinney, Dirk Moelants, Matthew E. P. Davies, and Anssi Klapuri. Evaluation of audio beat tracking and music tempo extraction algorithms. *Journal of New Music Research*, 36(1):1–16, 2007.

[15] Jouni K. Paulus and Anssi Klapuri. Drum transcription with non-negative spectrogram factorisation. In *Proceedings of the 13th European Signal Processing Conference (EUSIPCO)*, 2005.

[16] Björn Schuller, Florian Eyben, and Gerhard Rigoll. Tango or waltz?: Putting ballroom dance style into tempo detection. *EURASIP Journal on Audio, Speech, and Music Processing*, 2008(6):1–12, 2008.

[17] Christian Uhle. *Automatisierte Extraktion rhythmischer Merkmale zur Anwendung in Music Information Retrieval-Systemen*. PhD thesis, Technische Universität Ilmenau, Ilmenau, 2008.

[18] Andrew J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 1967.

[19] Jose R. Zapata and Emilia Gómez. Comparative evaluation and combination of audio tempo estimation approaches. In *Proceedings of the 42th AES International Conference*, Ilmenau, 2011.