Détection des positions de la battue dans les signaux musicaux à l'aide de la matrice "Delta-Phase"

Jan Weil[1], Thomas Sikora[1], Jean-Louis Durrieu[2], Gaël Richard[2]
[1] Teschnische Universität Berlin
Berlin, GERMANY
[2] Institut Télélcom, Télécom ParisTech, CNRS LTCI
37/39 Rue Dareau, 75014 Paris, FRANCE

August 17, 2009

Abstract

Nous proposons un algorithme simple mais efficace pour la détection des positions de la battue dans les signaux musicaux. Le concept de matrice “Delta-Phase” est présenté. Il s’agit de représenter l’évolution de la phase de la battue par rapport à la période correspondant à un tempo estimé au préalable. Le chemin optimal dans la matrice “delta-phase” est déterminé grâce à de la programmation dynamique, afin de s’assurer d’obtenir une battue relativement régulière. Le nombre d’erreur de phase est minimisé par un choix judicieux de périodes candidates. L’algorithme proposé est évalué sur une base de données contenant 474 extraits musicaux, couvrant divers genres. Nous obtenons des résultats du niveau de, voire supérieurs à, l’état de l’art.

BEAT TRACKING USING THE DELTA-PHASE MATRIX

Jan Weil, Thomas Sikora
Technische Universität Berlin
lastname@nue.tu-berlin.de

Jean-Louis Durrieu, Gaël Richard
Institut Télécom; Télécom ParisTech; CNRS LTCI
firstname.lastname@telecom-paristech.fr

ABSTRACT

We propose a simple yet efficient beat tracking algorithm. The Delta-Phase Matrix is introduced. It displays the progression of the pulse phase with reference to a period which is estimated before. We use dynamic programming to determine the optimum delta-phase path, which ensures a continuous beat grid. By carefully choosing appropriate period candidates we reduce the amount of phase errors. The presented algorithm is evaluated using a dataset which contains 474 music segments covering various genres. It compares favourably to two state-of-the-art systems.

1 INTRODUCTION

The various approaches to the analysis of rhythmic structures in music can be distinguished by several aspects. Although some of them work on symbolic data [11], most algorithms directly analyzing digital music signals depend on a signal representation which tries to detect musical events. Such a representation is usually referred to as an onset detection function (ODF). Usually, ODFs are designed to peak at note onsets. A tutorial overview on onset detection is given in [2]. The analysis of both constant and, in particular, varying tempo is one of the first tasks to tackle when analyzing rhythm. Beat tracking is usually understood as the process of finding those points in time when a human listener would tap his foot. This is different from the tempo estimation task since in addition to the beat period the beat phase, i.e., the actual position, is needed as well. Typical beat tracking applications include all forms of real-time synchronization to music, audio editing, and it is also an important part of automatic transcription systems. It is general consensus that, due to the presence of several metrical levels, the tempo is often ambiguous. The recognition of rhythmic patterns and, in particular, the meter [8], i.e., the time signature along with rhythmic sub-levels is even more challenging. The reader is referred to [7] for a thorough overview of topics related to the analysis of rhythm.

Laroche [9] proposed to use dynamic programming to estimate both the tempo and the exact positions of the beats in music signals. To this end, he splits the ODF into frames and chooses a subset of promising tempo candidates along with a corresponding subset of on-beat candidates. He then applies dynamic programming to determine the optimum pair of tempo and on-beat for each frame in a joint fashion. The pulse template used by Laroche is designed to fit typical 4/4 meters. Ellis [4] built on this approach and was able to reduce it to an elegant minimum assuming a relatively constant tempo which is estimated before. As also discussed by Laroche, tempo octave errors and phase errors are the main remaining problems of the beat tracking task.

In this paper we present an offline method which tries to avoid such phase errors and, at the same time, ensures a continuous beat grid. Similar to Laroche, we split the ODF into frames and estimate the phase for each frame. We do not assume a particular meter, though. The period has to be estimated before, which resembles the system of Ellis. However, there is no need for a constant tempo. In contrast to most other beat tracking systems we do not try to explicitly pick the tactus [8] level, i.e, the pulses which correspond to the beats in a measure. Instead we propose to favour a faster tempo whenever there is a chance for typical phase errors like picking the off-beat. The algorithm is evaluated using a dataset which covers several music genres. We compare it to two state-of-the-art systems, namely, Ellis's approach mentioned above [4] and Dixon's BeatRoot [3].

In the following section we present in short the onset detection function that was used for our experiments. After that we introduce the Delta-Phase Matrix and describe how the beat placement works. In Section 4 we illustrate the tempo estimation procedure for constant tempo and sketch how the beat placement works with varying tempo. We discuss the experimental results and their evaluation in Section 5 before we present our final conclusion.

2 ONSET DETECTION

We use a variant of the spectral flux [2] as the onset detection function (ODF). The input signal is downsampled to 11025 Hz and mixed to a single channel. We compute an overlapped Fourier transform with a window length of 256 and a hopsize of 32 samples. Each frame is windowed using a Hamming window. For each of the sub-bands we compute the logarithm and apply a finite impulse response (FIR) smoothing filter using an impulse response of 150 ms length. We then calculate a first-order difference from frame to frame to emphasize sudden changes in energy. Figure 1 depicts an example of this procedure. Each sub-band is half-

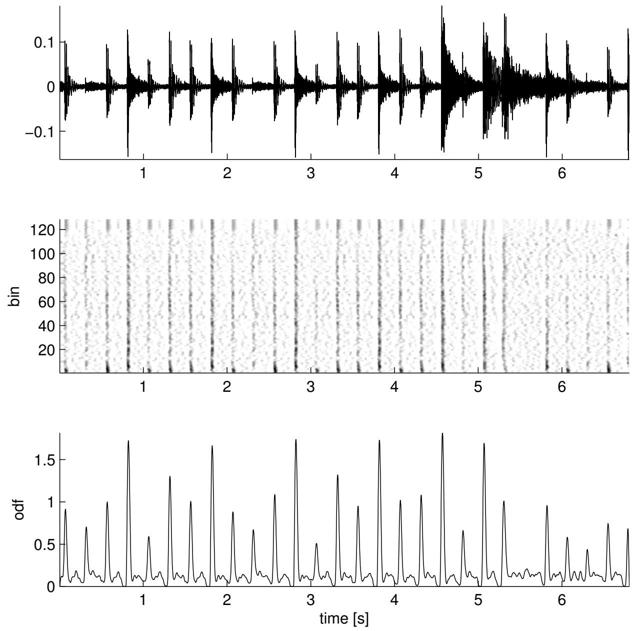

Figure 1. Computation of the onset detection function (ODF). Top: Mono waveform sampled at 11052 Hz. Center: First-order difference of the smoothed magnitude spectrum. Bottom: Proposed ODF.

wave rectified and all of them are summed up to form the ODF, which will be denoted by $\gamma(n)$ from now on. The resulting sampling rate of this ODF is 344.5 Hz.

3 THE DELTA-PHASE MATRIX

We first assume the tempo of the music signal is constant. Let b_i denote P beat positions ($i \in [0, P-1]$) and ρ the beat period. Let ϕ_0 be the initial phase, i.e., $b_0 = \phi_0$ and $b_i = \phi_0 + i \cdot \rho$. Suppose the ODF, $\gamma(n)$, is split into K overlapping frames $\Gamma(k)$ of length $L > \rho$ using the hopsize $h < L$; $k \in [0, K-1]$. If $\bar{\phi}(k)$ denotes the phase of the beat in frame k, i.e., the position of the first beat in this frame relative to its first ODF sample, the beat phase for the following frame is the result of

$$\bar{\phi}(k+1) = \bar{\phi}(k) - h \pmod{\rho}.^{1} \qquad (1)$$

Many algorithms in the literature determine the phase as the maximum of the cross-correlation function of $\gamma(n)$ and a pulse comb with the period ρ. We use the same principle for each ODF frame. The phase $\phi_c(k)$ of the pulse comb, however, is updated to reflect the relation described above. We arbitarily initialize $\phi_c(0) = 0$ and compute $\phi_c(k+1)$ according to Equation (1). The temporal evolution of the cross-correlation function for each frame forms the Delta-Phase Matrix (DPM) $\Phi_\delta(k, q)$. With the amplitude of the pulses equal to 1, we can compute

$$\Phi_\delta(k, q) = \sum_{j=0}^{\lfloor L/\rho \rfloor - 1} \gamma(\phi_c(k) + q + j \cdot \rho), \qquad (2)$$

where j iterates over all periods contained in a frame and $q \in [1, \rho]$. Each column of the DPM corresponds to one of the K frames. The comb phase $\phi_c(k)$ serves as an offset and the DPM column vector is used to determine the difference between the actual phase and this offset. We call this difference the delta phase $\phi_\delta(k)$. The optimum delta phase will in principle be given as q where $\Phi_\delta(k, q)$ is maximal. To enforce a regular beat placement, however, we apply dynamic programming [5] to determine the optimum delta-phase path. To this end, each column of $\Phi_\delta(k, q)$ is normalized by dividing it by its maximum. If ρ has been perfectly estimated and the tempo does indeed not vary we expect $\phi_\delta(k)$ to be constant. In this case the optimum delta-phase path is a straight horizontal line when $\Phi_\delta(k, q)$ is visualized. The bottom half of Figure 2 shows such an example. Sometimes, though, the tempo estimate is not perfect, which results in a sloped path and possibly $\phi_\delta(k)$ wrapping around ρ. See Figure 3 for an example. To reflect this possibility we add a cosine-shaped cost function:

$$c(d_q) = C \cdot \cos(2\pi d_q/\rho) \qquad (3)$$

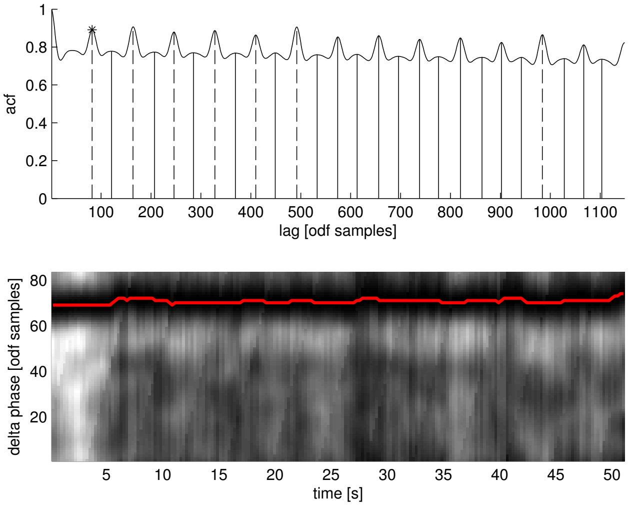

Figure 2. *Annie's Song*, The Golden Nightingale Orchestra. Top: Autocorrelation function of the ODF. The chosen beat period (marked with an asterisk) corresponds to a tempo of 252 bpm. In this case the eigth-note level of a 3/4 meter was picked. Bottom: The Delta-Phase Matrix. The optimum delta-phase path is marked with a solid line. The tempo is basically constant and well estimated which results in a horizontal orientation of that line.

[1] Note that $m \pmod{n} = m - \lfloor m/n \rfloor \cdot n$ is always positive for positive n as $\lfloor \ldots \rfloor$ rounds towards minus infinity.

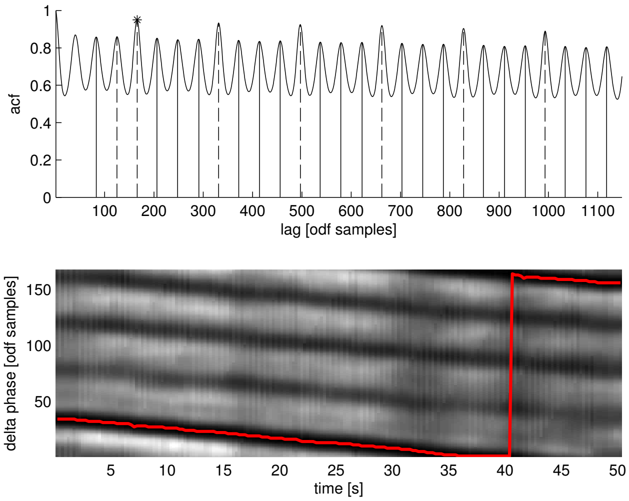

Figure 3. *Always On My Mind*, Pet Shop Boys. The chosen period (cf. Figure 2) corresponds to the tactus level of a 4/4 meter at 124.5 bpm. The tempo is again constant but not perfectly estimated. The optimum delta-phase path is thus sloped and wraps around. The presence of rhythmic sub-periods, corresponding to peaks of which the period is smaller than the chosen one, is clearly visible. As, however, the chosen period is salient enough, no phase error occurs.

d_q denotes the absolute difference between $\phi_\delta(k)$ and $\phi_\delta(k-1)$. This cost function penalizes phase jumps unequal to the period ρ. We apply dynamic programming to find the optimum delta-phase path $\phi_\delta(k)$, maximizing the gain given by $\Phi_\delta(k,q)$ and minimizing the cost given by $c(d_q)$. The cost weight C is used to tune the algorithm when confronted with tempo changes, which will be discussed in Section 4.2.

Eventually, we compute an absolute beat position for each frame as

$$b_f(k) = k \cdot h + \phi_c(k) + \phi_\delta(k).$$

Depending on the hopsize h and the beat period ρ, we have to add missing beats to find all beat positions for the piece under consideration and remove closely adjacent duplicates.

4 TEMPO ESTIMATION

Our goal in choosing a period candidate is not to explicitly find one of the metrical levels tactus, tatum, or measure [8]. We do not try to understand the rhythmic structure and, in particular, the time signature at this point. Instead we aim at picking a period which reduces the risk of phase errors. As will be shown, this period often is half the tactus period.

4.1 Constant Tempo

If, for any reason, the tempo is known to be more or less constant for the musical piece under consideration we can provide our beat-placement algorithm with a global tempo estimate. In that case the beat tracking performance strongly depends on the actual deviation of the tempo. This resembles the approach described by Ellis in [4].

We use a heuristically motivated peak picking algorithm to choose the optimum beat period candidate. We compute the biased autocorrelation function (ACF) of the onset detection function up to a lag corresponding to 20 bpm [2] . We then apply peak picking above a minimum period ($\sim$ 360 bpm) and choose the seven greatest peaks. We compute the corresponding inter-peak intervals and, finally, we choose the peak which is closest to the majority of the inter-peak intervals as the preferred period. The upper halves of Figures 2 and 3 depict two examples of the autocorrelation function along with the relevant peaks. The greatest peaks are drawn using dashed lines; the chosen beat period is marked with an asterisk.

The rationale behind our peak-picking approach is as follows. As noted above, we try to minimize the chance of phase errors. Picking the off-beat, for example, is such a typical phase error. If the off-beat is emphasized, e.g. in certain music styles like Reggae music, it will be clearly marked with a peak in the autocorrelation function of the ODF along with integer multiples. As illustrated in Figure 3, such sub-periods are reflected in the DPM along with the chosen period. The emphasis of the on-beat is strong enough in this example, and there is no phase error. However, the chance of picking the wrong delta-phase path increases if the off-beat, or any sub-period in general, is similarly emphasized. Therefore, our goal is to make sure that no such spurious sub-periods exist. Put simply, if there is a risk to pick the off-beat, cut the period in half. Where we would have picked the off-beat instead of the on-beat we now pick both. Figure 2 shows such an example. The tactus (i.e., quarter-note) period of the depicted 3/4 meter corresponds to the second peak at a lag of about 165 ODF samples. Given the salience of the eigth-note period, however, trying to find the correct phase for the tactus period is almost like gambling.

We compute the biased ACF, which does not compensate for the decreasing number of summands, because this adds a slight slope favouring smaller periods. This makes it more likely that we pick the greatest period peaks in increasing order and that they actually correspond to those salient peaks with the smallest period. We compute the ACF up to a log correspondng to 10 bpm to give enough room to pick peaks. We pick seven of them as we target peak periods corresponding to roughly 120 bpm and try to cover the available period range. Of course, this approach generally results in higher tempi and it depends on the application whether this is acceptable. However, the chance of missed beats is re-

[2] We used the entire piece for the given dataset, but would suggest to use a representative segment, e.g. the first 30 seconds, in the general case – if the tempo is assumed constant.

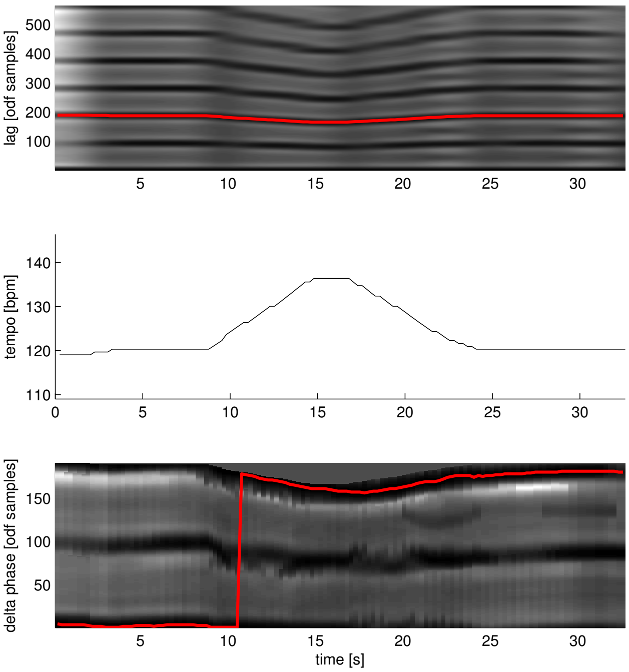

Figure 4. A synthesized drumloop with varying tempo. Top: The auto-correlogram of the ODF along with the optimum period path marked with a solid line. Center: The corresponding tempo map. Bottom: The Delta-Phase Matrix.

duced which is crucial for many segmentation applications.

4.2 Varying Tempo

Generally, the beat placement approach presented here is well suited for varying tempo as well. The difference is that ρ obviously is no longer constant but varies over time. Since the Delta-Phase Matrix is frame based, we can simply estimate the beat period once per frame and update Equations (1), (2), and (3) accordingly, replacing ρ by $\rho(k)$. We developed a simple dynamic tempo estimator, again based on dynamic programming, using the auto-correlogram of the ODF. A similar system has been presented by Alonso in [1]. In our case the hopsize has to be the same as for the DPM. Such a dynamic programming approach using a simple linear cost function will mostly yield a musically meaningful beat period path. Applying the same rationale as discussed above, trying to reduce phase errors, is non-trivial, though. The main challenge is to prevent jumps between different tempo levels. An example for our tempo estimator is shown in Figure 4. A synthesized drum loop was modified to follow a parametrized tempo map: starting from constant 120 bpm, linearly increasing to 140 bpm, linearly decreasing back to 120 bpm, followed by a period of constant 120 bpm again. In this simple case the tempo estimator was able to follow very well. Confronted with more realistic signals, our simple tempo tracker is prone to period jumps. Such tempo jumps make the evaluation even more challenging and the result is not suitable for many applications building on it. There are many potential solutions to this problem proposed in the literature, e.g. the template-based estimation described by Peeters in [10]. Note, though, that these tempo jumps may well reflect the musical content and the DPM will still ensure a continuous beat grid.

5 EVALUATION AND DISCUSSION

We compare the DPM beat tracker to two state-of-the-art algorithms, namely, Dixon's BeatRoot [3] and Ellis's beat tracking by dynamic programming [4]. BeatRoot uses a multiple agents architecture which simultaneously considers several beat hypotheses based on inter-onset intervals. Ellis's quite elegant approach is, to a certain extent, similar to ours. It is, however, not frame based and requires a relatively constant tempo. The source code for both systems is available online. We use the same dataset as Klapuri in [8] consisting of 474 music segments of various genres. Manually annotated beat positions for the tactus level are available for all pieces in this dataset. For some of them the tatum level has been annotated as well; we do not consider this information, though. We assume the tempo constant for all pieces even though this is actually not the case for all of them. The genre distribution is shown in the first two columns of Table 1. More details are available online [3] .

For our experminents, we set the cost weighting to $C = 6.0$. We chose the hopsize $h = 0.5\,s$. The window length was adapted to $L = 7.5\,\rho$ to make sure that enough peaks contribute to find the optimum delta phase. Note that this corresponds to roughly one bar, assuming we pick the eigthnote period of a 4/4 meter.

Figure 5 compares which of the metrical levels are detected by the three systems. BeatRoot performs best in picking the ground-truth tactus tempo. For about 60 % of the pieces the ratio of the detected tempo to the ground-truth annotation is 1; for the majority of the rest the tempo is twice the annotated. Ellis's beat tracker's tempo estimates are roughly equally distributed between ratio 1 and 2. The quite simple period picking approach proposed in this paper favours twice or even three or four times the ground-truth tempo. This is consistent with our aim to prevent phase errors as described in Section 4.1.

To evaluate the beat placement we use basically the same metric as Klapuri [8]: the performance measure is the portion of the longest continuous correctly analyzed segment in relation to the length of the entire signal. If a single beat in the middle of a piece is missed the performance value cannot be higher than 50 %. A segment is assumed to be analyzed correctly if the found beats deviate from the ground truth by

[3] http://www.cs.tut.fi/~klap/iiro/meter/

		DPM		Dixon		Ellis	
Genre	#	Perf.	Phs.	Perf.	Phs.	Perf.	Phs.
Classical	84	24.37	0.39	28.39	0.45	41.46	0.29
Electronic / Dance	66	**71.37**	**0.23**	62.26	0.32	44.98	0.40
Hip Hop / Rap	37	91.34	0.12	88.16	0.12	74.20	0.20
Jazz / Blues	94	62.58	0.23	53.68	0.26	61.19	0.26
Rock / Pop	124	77.90	0.17	80.28	0.14	74.78	0.21
Soul / RnB / Funk	54	82.62	0.15	80.30	0.14	69.94	0.23
Unclassified	15	61.16	0.24	52.70	0.32	61.33	0.26
Total / Average	474	65.52	0.23	63.05	0.25	61.01	0.26

Table 1. Genre distribution of the dataset, average performance values [%], and average phase error measures. The performance value is the longest continuous portion of the song for which all beats are detected. The phase error measure is $0 \leq \epsilon \leq 1$, where 1 corresponds to a systematic off-beat error.

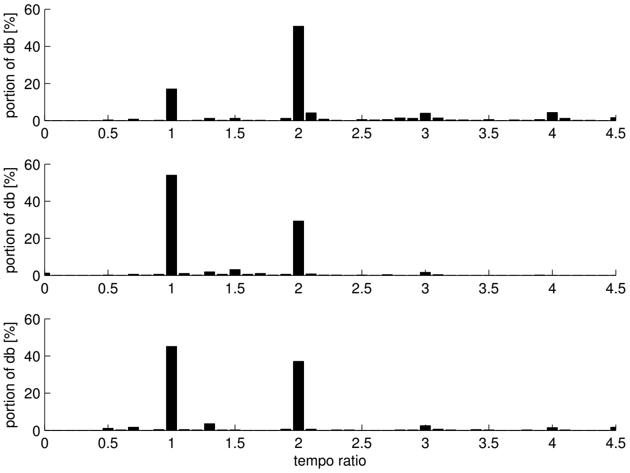

Figure 5. Histogram of the ratio *detected tempo / ground-truth tempo* over the entire database. Top to bottom: DPM, Dixon, Ellis. The peak picking approach presented (DPM) generally favours higher tempi.

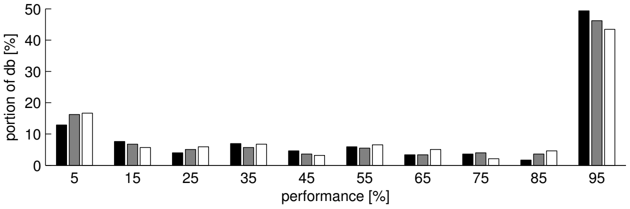

Figure 6. Histogram of the performance values over the entire database. Left to right: DPM, Dixon, Ellis. All three algorithms correctly track essentially all the beats for more than 40 % of the dataset.

less than $17.5\,\%$ of the period. If the (quantized) ratio r_q of the estimated tempo and the ground-truth tempo is one of $[2, 3, 4]$ we consider only every second, third, or fourth detected beat, respectively. In this case the maximum performance value for all r_q possible starting beats is chosen. Figure 6 displays the histogram of the performance measure over the entire database (left to right: DPM, Dixon, Ellis). All three systems track more than 90 % of the beats correctly for more than 40 % of the pieces. The total average precision is 65.9 %, 63.2 %, and 61.8 %, respectively. Genre-specific average performance values are given in Table 1. The distribution of the performance values largely complies with the findings reported by Klapuri in [8].

As noted above, most beat tracking systems working on audio data rely on a well performing onset detection front-end. As indicated in Table 1, our approach performs best for music containing clear percussive elements. Classical music, in particular, remains a challenge. The suboptimal performance of all three systems for classical music is a problem of onset detection. We did, however, not spend much effort in optimizing our ODF. Choosing a better frontend will probably further improve the results. With the BeatRoot program, it is possible to directly provide an ODF instead of the audio data. When we provided it with the ODF used in this paper its average performance dropped to 53.6 % (ca. -10 %) which confirms that our ODF can be improved. If we were able to similarly improve our system by tuning the ODF the average performance would be comparable to Klapuri's noncausal tactus tracker. This is certainly simplifying matters, though.

For Electronic, Hip Hop, or Funk music there is usually no problem with the detection of onsets. However, these are genres which are prone to typical tempo and phase errors. As proposed above, we tried to favour smaller periods, i.e., faster tempi, whenever phase errors are likely. We tried to assess phase error problems by computing a measure for each song as follows: Let b_i be the P known beat positions with $i \in [0, P-1]$ and ρ the ground-truth period. The phase error measure is

$$\epsilon = \frac{2}{\rho \cdot P} \sum_{i=0}^{P-1} \delta_b(i),$$

where $\delta_b(i)$ denotes the absolute distance of b_i to the closest detected beat. A systematic off-beat error, which leads to $\delta_b(i) = 0.5 \cdot \rho$ for all i, yields $\epsilon = 1$; $\epsilon = 0$ means that all beats have been detected correctly. As for the performance measure, we consider only every r_qth beat for $r_q \in [2, 3, 4]$. Examining the detailed results, we find that, as expected, high values of ϵ mostly go along with low performance values. Generally, the lower ϵ values for the DPM tracker seem to justify the design decision we made. A particularly good example is the Electronic genre (highlighted in bold) which for both Dixon's and Ellis's beat tracker seems to cause problems. Note that it would make sense to only compute the phase error measure if the tempo estimation did not fail. However, we did not try to filter out songs for which the tempo estimation failed as this is hard to assess, and the presented results would probably be a bit confusing. The average phase error measure thus does not only reflect true phase errors but also those cases for which the tempo estimation failed, e.g., the bigger part of the Classical tunes.

6 CONCLUSION

We introduced the Delta-Phase Matrix which, in combination with dynamic programming, forms a robust tool to determine the beat phase given a period which has to be estimated separately. The method presented works for constant as well as for varying tempo. By favouring smaller period candidates we could reduce the chance of phase errors. The system was evaluated using a large database covering several genres. The evaluation disclosed the weak performance of our onset detection function for classical music; this is, however, generally a challenging problem and was beyond the scope of this paper. The beat tracking performance was compared to two state-of-the-art systems, which were slightly outperformed. The apparent weakness of our onset detection function seems to leave room for further improvement.

The beat tracking problem is by now well understood. Remaining challenges, apart from hard-to-detect onsets, are mainly related to varying tempo and phase errors. Furthermore, estimating the time signature is a non-trivial task. We believe that the best approach to this problem cannot be exclusively based on an ODF but should also consider harmonic changes or other clues where applicable. An example of this has already been presented by Goto in [6].

7 ACKNOWLEDGMENTS

The authors would like to thank Anssi Klapuri for providing the dataset and Simon Dixon as well as Dan Ellis for making their source code available. This work was supported by the European Commission under contract FP6-027026 (KSpace), by the European Community?s Seventh Framework Programme [FP7/2007-2011] under grant agreement no 216444 (see Article II.30. of the Grant Agreement) and by OSEO, French State agency for innovation, as part of the Quaero Programme.

8 REFERENCES

[1] M. Alonso. Accurate tempo estimation based on harmonic + noise decomposition. *EURASIP Journal on Advances in Signal Processing*, 2007:1–14, 2007.

[2] 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, 2005.

[3] S. Dixon. Evaluation of the audio beat tracking system beatroot. *Journal of New Music Research*, 36(1):39–50, 2007.

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

[5] B. Gold and N. Morgan. *Speech and audio signal processing*. Wiley New York, 2000.

[6] M. 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] F. Gouyon. *A computational approach to rhythm description*. PhD thesis, Department of Technology of the University Pompeu Fabra, 2005.

[8] A. Klapuri, A. Eronen, and J. Astola. Analysis of the meter of acoustic musical signals. *IEEE Transactions On Speech And Audio Processing*, 14(1):342, 2006.

[9] J. Laroche. Efficient tempo and beat tracking in audio recordings. *Journal-Audio Engineering Society*, 51(4):226–233, 2003.

[10] G. Peeters. Template-based estimation of time-varying tempo. *EURASIP Journal on Advances in Signal Processing*, 2007:1–14, 2007.

[11] D. Temperley. *The cognition of basic musical structures*. Mit Press, 2001.