Published version: IEEE Transactions on Audio, Speech and Language Processing, 20(1): 190-199, Jan 2012. doi: 10.1109/TASL.2011.2159593

Performance Following: Real-Time Prediction of Musical Sequences Without A Score

Adam M. Stark, and Mark D. Plumbley, *Member, IEEE*

***Abstract*—This paper introduces a technique for predicting harmonic sequences in a musical performance for which no score is available, using real-time audio signals. Recent short term information is aligned with longer term information, contextualising the present within the past, allowing predictions about the future of the performance to be made. Using a mid-level representation in the form of beat-synchronous harmonic sequences, we reduce the size of the information needed to represent the performance. This allows the implementation of real-time performance following in live performance situations.**

We conduct an objective evaluation on a database of rock, pop and folk music. Our results show that we are able to predict a large majority of repeated harmonic content with no prior knowledge in the form of a score.

***Index Terms*—Performance following, online algorithms, beat-synchronous sequences.**

I. Introduction

THE DESIRE to create interactive automatic musical accompaniment systems for musicians has been the focus of much research in recent years. This technology has potential in several areas including facilitating rehearsal when no or limited human accompaniment is available [1] or easing the relationship between human performers and computer technology in live performances [2]. This technology has also often been used to create new forms of innovative and creative musical works [3].

One particular area that has received much attention is *score following* [4]. Score following is the automatic matching, in real-time, of notes played in a performance to those in a musical score. It allows us to discover the current position of the performance in the score. This positional information allows us to know the future of the performance and as a consequence a musical accompaniment can be played.

However, in some cases no score exists for the performance. This is not unusual for many forms of music, including rock and pop music. Furthermore, the music may be improvised, rendering the prospect of producing a score virtually impossible.

In this paper we address the problem of predicting the future of performances for which no score exists. Solutions to this problem, which we will refer to as *performance following*, can take advantage of the fact that much music contains repetitions of musical phrases [5, Ch. 1]. Indeed, it is certainly a skill of human musicians to be able to recognise repetitions in music and to be able to make predictions of the future based upon these repetitions.

Our challenge, then, is to contextualise recent musical information (the last few seconds) within longer term musical information (the last few minutes). Assuming the performance contains repeats, we should be able to identify any repeats and therefore make predictions about the future of the performance. This would allow an automatic musical accompaniment, such as a bassline or melody, to be generated to a live musical performance with no prior knowledge in the form of a score.

Our motivation for attempting to solve this problem is twofold. Firstly, we believe that automatic accompaniment systems, such as score followers, should be capable of dealing with spontaneity in music, such as is the case in improvisation. Secondly, we believe that there are a number of potentially interesting new applications based upon the idea of performance following. The prediction of future harmonic content opens up the possibility of content-informed audio effects or automatic musical performers that do not require human supervision to partake in musical performances.

A. M. Stark was supported by an EPSRC DTA Studentship. M. D. Plumbley was supported by an EPSRC Leadership Fellowship.

A. M. Stark and M. D. Plumbley are with Queen Mary University of London, London, E1 4NS, U.K. (e-mail: adam.stark@eecs.qmul.ac.uk; mark.plumbley@eecs.qmul.ac.uk)

Manuscript received July 9, 2010. Revised November 19th, 2010 & April 1st 2011.

II. Background

The repetition of musical patterns is an aspect commonly found in music, particularly western music [5, Ch. 1]. These patterns can easily be represented as sequences of notes [6], chords [7] or other musical features [8], [9].

Given the partially repetitive nature of these sequences a topic of interest in recent years has been the prediction of musical sequences.

A. Musical Prediction

Musical prediction is a topic that has been approached from a number of directions. Some have taken an information theoretic perspective where models are developed to reflect the perceptions of human listeners [10], [11]. The process of "anticipation", arising from prediction, has also been identified [10], [12], [13], a full discussion of which is beyond the scope of this paper. Others make no attempt to model human cognitive processes, employing prediction purely for the operation of some practical system [14]. In this present paper, as we are concerned solely with solving a practical problem, we also make no attempt to model human cognition, and focus on making musical predictions by identifying repetitions of current patterns in the past of the performance.

Early systems for musical prediction were developed to learn production rules from musical examples. For symbolic data, Kohonen [15] developed a context sensitive generative grammar that, given a number of sequences that presented two conflicting next symbols, dynamically expands the context length of one or more of the sequences until such conflicts are resolved. Later, Thom [14] made use of N-gram models to predict chord transitions in Jazz music.

More recently, Pachet [16] has introduced the *Continuator*, a system that learns stylistic information about a performance as it takes place. Operating upon symbolic data, and based on an extension of a Markov model, it represents sequences and sub-sequences of notes either from a performance or from data stored in a MIDI file. The Markov model is then used to generate new material in real-time by traversing a prefix tree according to transition probabilities.

Conklin [17] has argued that music generation and analysis are highly interconnected and highlighted the limitations of models that have highly specific contexts such as N-gram or Markov-based techniques.

Assayag and Dubnov [18] have suggested using the *Factor Oracle* structure [19] for musical analysis and generation. The Factor Oracle is a state automata which, for a given sequence of length N, is able to represent the presence and location of repeated sub-sequences of variable length using a linear number of states ($N + 1$) and transitions (at most $2N - 1$).

Factor Oracles have been used as the basis for the interactive system OMAX [20] for polyphonic MIDI and its equivalent for monophonic audio, OFON.

A difficulty associated with Factor Oracles is that they require a single discrete state for each symbol. This is appropriate when processing MIDI data, or for monophonic audio providing a reasonably reliable pitch detector can convert audio signals to discrete symbols. However, when considering polyphonic audio, it is more complicated to convert musical features, such as spectral vectors, into single discrete symbols.

Two suggestions relating to Factor Oracles are known to the authors. The *Audio Oracle* [21] is an extension of Factor Oracles for use with audio vectors, assigning transitions based upon a Euclidean distance function. A second suggestion was made in [22] that by using a “relatively severe quantisation”, spectral vectors can be reduced to a small number of classes.

Both these techniques rely on a form of thresholding of distance functions between spectral vectors, either in spectral comparison or in some form of clustering. However, spectral vectors extracted from musical audio that humans may consider harmonically similar can vary in their constitution depending on the instrumentation, intensity of performance, the register of the instrument as well as several other factors. As a result, any such thresholding is bound to be imperfect in some way. Indeed, the process of discretising harmonic vectors into a number of classes is related to the problem of chord recognition, which is still an unsolved problem.

We desire for our application a technique that is capable of 1) comparing sequences of spectral vectors extracted from polyphonic audio; and 2) allowing the comparison of sequences that are non-exact, possibly containing substituted, inserted or deleted elements. We turn our attention now to a widely used family of techniques for the alignment of musical sequences based upon similar sub-sequences.

B. Sequence Alignment in Music

Many techniques for the comparison of musical sequences are based upon several algorithms that have emerged from the field of computational biology. For the global alignment of two sequences of amino acids, Needleman and Wunsch [23] detailed a technique that first calculates a score matrix based upon the similarity of these sequences. In the simplest case, a value of 1 is assigned for matching elements and a 0 for mismatches. A dynamic programming algorithm is then used to calculate all of the possible pathways through the matrix. Finally a traceback step is performed to compute the best alignment of the two sequences. The algorithm is such that it allows for differences in the sequences such as inserted, deleted and substituted elements to be taken into account. Building on this work, Smith and Waterman [24] later presented a technique for computing the highest scoring alignments between sub-sequences of two longer sequences. These are referred to as local alignments. A faster approximation of local alignment without ‘gaps’ (originating from inserted and deleted elements) was presented in [25] with a gapped version following later [26].

These algorithms have been applied widely to music. Mongeau and Sankoff [6] used sequence alignment techniques to compute a value of similarity between two musical scores. Representing monophonic scores as ordered sequences of pitch-duration pairs, an alignment is calculated taking into account the concepts of insertion, deletion and substitution as in [23]. However, the concepts of replacement of one note by several, and several by one, are also considered.

In the field of music information retrieval, Hu et al [27] describe a technique that, given an audio file, attempts find the corresponding MIDI file or vica versa. Ferraro and Hanna [28] investigate various optimisations of alignment algorithms for music through the consideration of different pitch representations, different substitution costs for notes and consideration of note duration. Robine et al. [29] suggest improvements to alignment algorithms through the incorporation of aspects of music theory such as the intervals between notes.

For analysing the structure of musical pieces there has been an adaptation for musical data of the widely used genetic sequence alignment algorithm BLAST [25] by Kilian and Hoos [30] and the approach of Dannenberg and Hu [31] for creating structural descriptions of a piece of music.

C. Musical Sequence Alignment in Live Performances

The systems mentioned above largely process audio files in an offline fashion. However, we are proposing a real-time, live music performance system. Several existing real-time systems make use of the sequence alignment techniques discussed above.

Many score following systems have made use of sequence alignment of musical information. Dannenberg [32] compares a monophonic performance to a score, using pitch estimation to create a sequence of pitches from the audio and sequence alignment techniques to compare it to the score. A set of adaptations for handling polyphonic keyboard performances were later presented [33].

Pardo and Birmingham [7] compare polyphonic MIDI performances to partially specified scores or 'lead sheets' - musical notations that indicate the essential elements of a performance to musicians, for example chord sequences or the leading melody. From the MIDI information, a chord sequence is extracted and this is then compared to the lead sheet using a global sequence alignment.

Dannenberg and Hu [8] have suggested that their technique for comparing polyphonic audio to a symbolic MIDI file could be used as part a real-time performance system. They achieve this comparison by synthesising audio from the MIDI file and extracting sequences of chroma vectors from both audio streams before computing an alignment between the two.

Dixon and Widmer [9] present an approach for aligning polyphonic audio recordings of different performances of the same piece of music. This is intended to be a useful tool for musicologists and musicians as it allows the user to switch between time aligned recordings. Using a spectral representation linear at low frequencies and logarithmic at high frequencies, spectral difference vectors for each frame compared to its predecessor are calculated. A cost matrix is calculated using the Euclidean distance between vectors and an alignment is computed through a dynamic programming sequence alignment technique. This technique was later applied to the problem of tracking a live performance in real-time [34].

It should be mentioned that there are score following techniques that do not make use of sequence alignment. An example of this has been presented by Raphael [35] who uses hidden Markov models [36] to compare notes played by a soloist to a musical score. Based upon this analysis, the score itself and data from past performances, a probabilistic distribution is created which is used to schedule MIDI notes in the future, from a score, to create an accompaniment. Another example is Cont's *Antescofo* [37] which makes use of a probabilistic framework containing two audio and tempo agents that use each others predictions to estimate the position of a performance in a score and the current tempo.

D. Other Automatic Accompaniment Systems

The above techniques compare a musical performance to some form of score. However, we are attempting to track a musical performance without a score. Several other systems have been presented that track performances in some way that do not make use of a score.

Simon et al. [38] introduced *MySong*, an offline system for automatically choosing chords to accompany a vocal melody provided by the user. Aimed at musically untrained users, a hidden Markov model, trained on a music database, is used to select chords for each new melody.

Robertson and Plumbley [2] describe *B-Keeper*, a system for tracking a live drummer. An event based beat-tracker is used to detect tempo changes in a live performance and a pre-recorded accompaniment is adapted to maintain synchronisation with the drummer.

These previous approaches either do not operate in real-time [38] or do not attempt to predict harmonic information in the future of the performance [2]. We are proposing a real-time, audio based system for predicting future harmonic content in musical performances, with no access to a score.

III. Approach

We approach the problem of attempting to predict the future of a musical performance with no prior information in the form of a score. Our solution is to use the hypothesis that much music contains repeated themes and patterns. We wish to 'recognise' musical patterns being performed by referring to the past of the performance. If the current musical pattern has been previously performed in the piece, we can identify these previous repeats and use them to make predictions about the future of the current musical pattern.

The live performance systems mentioned in section II-C typically make use of a global alignment between the performance and some form of existing score. As we are solving the problem of tracking a performance without a score, we have no score to match against. Therefore, we instead compare the most recent few seconds of the performance to the most recent few minutes. The past information of the performance itself becomes the 'score'.

However, in comparing recent audio to longer term audio we are presented with some challenges. Firstly, the tempo of the performance may change, making audio-to-audio comparison non-trivial as the same pattern may be of different lengths. Secondly, the comparison of high quality audio signals in their raw form is computationally expensive.

In order to solve these problems, we calculate beat-synchronous sequences from the audio, with one symbol for each inter-beat interval. These are discussed further in section IV.

Once we have calculated these beat-synchronous sequences, we apply a sequence alignment technique, described in section V, based upon dynamic programming, to make predictions about the future of the performance based upon past information.

Overview of System

Before discussing the details of its implementation, we give a brief overview our system.

We use as an input signal the output of a single polyphonic instrument, such as an acoustic guitar or a piano. This is transformed, in real-time, in to a sequence of beat-synchronous chroma features using a chromagram analysis technique and either a beat-tracker or fixed tempo click. The chroma feature sequence is then fed into our performance follower which attempts to predict the content of the next chroma feature from past information. An overview of our approach is depicted in Figure 1.

IV. Beat-Synchronous Sequences

Beat-synchronous sequences are sequences calculated from music that has been segmented in some way related to the beat. We take the beat to be the dominant perceived metrical pulse in a piece of music [39]. The segmentation could be of the content between two beats, or the interval of half or double this length caused by performing beat-synchronous analysis at double or half the perceived tempo.

Through choosing to use beat-synchronous sequences, we are making the assumption that the music to be used as an input is based around a strong beat.

Beat-synchronous processing has been used previously by others to identify cover songs [40], it has been used in [41] and [42] for analysing the structure of music, and, in a real-time context, in a remixing and cross-processing tool [43].

A. Advantages of Using Beat-Synchronous Sequences

There are three advantages to representing music beat-synchronously for our application. Firstly, it drastically reduces the amount of information needed to represent an audio signal. This cuts the computation time of any subsequent sequence comparison. For example, to represent 60 seconds of audio at 44.1kHz, using a frame size of 512 samples, we need over 5000 features. But if we reduce this to one feature per beat, at 120bpm, we only need 120 features - around 2.5% of what we needed for frame-by-frame representation.

A second advantage is that the same signal will be represented by the same number of features regardless of tempo. This improves our ability to easily compare sequences.

Finally, as we are looking to represent harmonic information in the signal, beat-synchronous signals take advantage of the fact that harmonic changes often occur at beat-locations. This allows us to avoid the problem of attempting to analyse an audio frame that contains information from both before, and after, a harmonic change. Beat-synchronous techniques have been shown previously to increase performance in harmonic analysis [44].

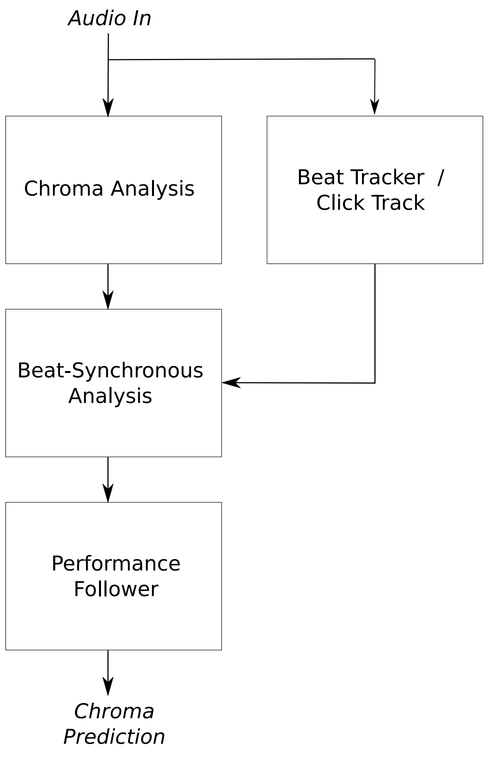

Fig. 1. a) An overview of the performance following technique.

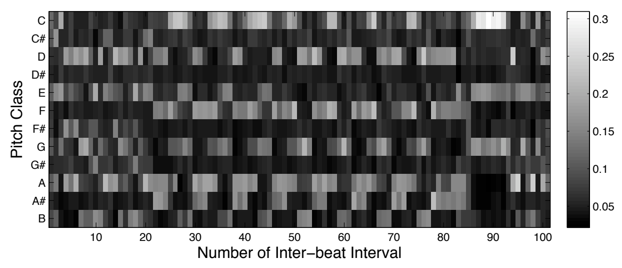

Fig. 2. A polyphonic beat-synchronous sequence - a sequence of chroma vectors. This shows the harmonic content for each inter-beat interval. By inspection we can see some repeated content. For example, the content of inter-beat intervals 29-36 looks visibly similar to the content of inter-beat intervals 37-44.

B. Calculation

We can achieve beat-synchronous segmentation using either some form of real-time beat tracker [45] or a fixed tempo click track. Due to the unreliability of existing beat tracking systems our emphasis is currently on the use of a click track (although we have also successfully used the system with a real-time beat tracker). It is quite possible to calculate beat-synchronous sequences at higher or lower metrical levels, but at present we choose the level of the tactus, the rate at which humans are most likely to tap beats.

At the time, in seconds, of the rth beat in the performance, γ_r, we calculate a single beat-synchronous feature vector, Δ_k, for the kth *inter-beat interval*, the time between the beats $[\gamma_{r-1}, \gamma_r]$. If, in the performance, we have R beats, then we will have $K = R - 1$ inter-beat intervals and therefore K beat-synchronous feature vectors (hence the subscript k for Δ and r for γ).

For a more detailed discussion of techniques for calculating beat-synchronous sequences, see [45].

C. Beat-Synchronous Sequences for Polyphonic Audio

In representing polyphonic audio, we use beat-synchronous sequences of chroma vectors, also known as pitch class profiles [46]. A chroma vector is a 12×1 vector whose values represent the energy present in each of the 12 semitone pitch classes found in western music. To calculate beat-synchronous chroma features, we downsample the 44.1kHz input signal to 11025Hz and make use of our previous chroma feature calculation technique [47]. We calculate as many chroma features as is possible for the inter-beat interval $[\gamma_{r-1}, \gamma_r]$ given the audio frame size of the chroma feature calculation technique. Then, for the kth inter-beat interval, we create a single chroma vector, Δ_k, by summing the Q vectors within the inter-beat interval. The vth bin of Δ_k is calculated by:

$$\Delta_k(v) = \sum_{q=1}^{Q} C_q(v) \tag{1}$$

where C_q is the qth chroma feature in the inter-beat interval, $q = 1, 2, ..., Q$, where Q is the number of chroma features calculated in the inter-beat interval $[\gamma_{r-1}, \gamma_r]$ and $v = 1, 2, ..., 12$. Finally, once the polyphonic beat-synchronous chroma feature has been calculated, we square it to increase the dynamic range and suppress the noise floor before normalising it so that each chroma vector sums to 1. An example of a beat-synchronous chroma vector sequence can be seen in Figure 2.

V. Performance Following: Sequence Prediction

In order to perform sequence prediction, we maintain two sequences. The first, $A = a_1, a_2, ..., a_N$, is a longer sequence containing the N most recent beat-synchronous vectors, which we refer to as the *long-term memory*. The second, $B = b_1, b_2, ..., b_M$, is a shorter sequence containing the M most recent beat-synchronous vectors, which we refer to as the *short-term memory*. Specifically:

$$A = \Delta_{k-N+1}, ..., \Delta_k \tag{2}$$

$$B = \Delta_{k-M+1}, ..., \Delta_k \tag{3}$$

where Δ_k is the kth, and most recent, beat-synchronous vector and $N > M$.

We choose the values of N and M according to a likely tempo of 120bpm. N should be large enough to allow the storage of recent developments in the music - although larger values will increase computation time. A value of 300 gives us a long-term memory length of 2 minutes and 30 seconds and seems to allow computation in real-time. The value of M determines the length of the longest sub-sequence match between the short-term and long-term memory. We discuss the choice of the value of M in section VII-A but for now suggest a value in the range of 5 to 50.

Our alignment technique is an adaptation of the Smith-Waterman algorithm [24]. In order to detect previous occurrences of information in the short-term memory, sequence B, within the long-term memory, sequence A, we compute an alignment matrix between the two.

The first step is to calculate a similarity score matrix, $s(i, j)$. We achieve this by calculating the inner product of the beat-synchronous chroma vectors in sequences A and B:

$$s(i, j) = \sum_{v=1}^{V} a_i(v) \times b_j(v) \tag{4}$$

where $a_i(v)$ is the vth chroma bin of the ith beat-synchronous chroma vector in sequence A, $b_j(v)$ is the vth chroma bin of the jth beat-synchronous chroma vector in sequence B and $V = 12$, the number of pitch classes in each chroma vector.

Next we calculate the alignment matrix H by dynamic programming. The value $H_{i,j}$ indicates the score for the alignment of two sub-sequences ending in a_i and b_j. Initially we set all the values $H_{i,0}$, $H_{0,j}$ and $H_{0,0}$ to zero. We then calculate the rest of the matrix H by:

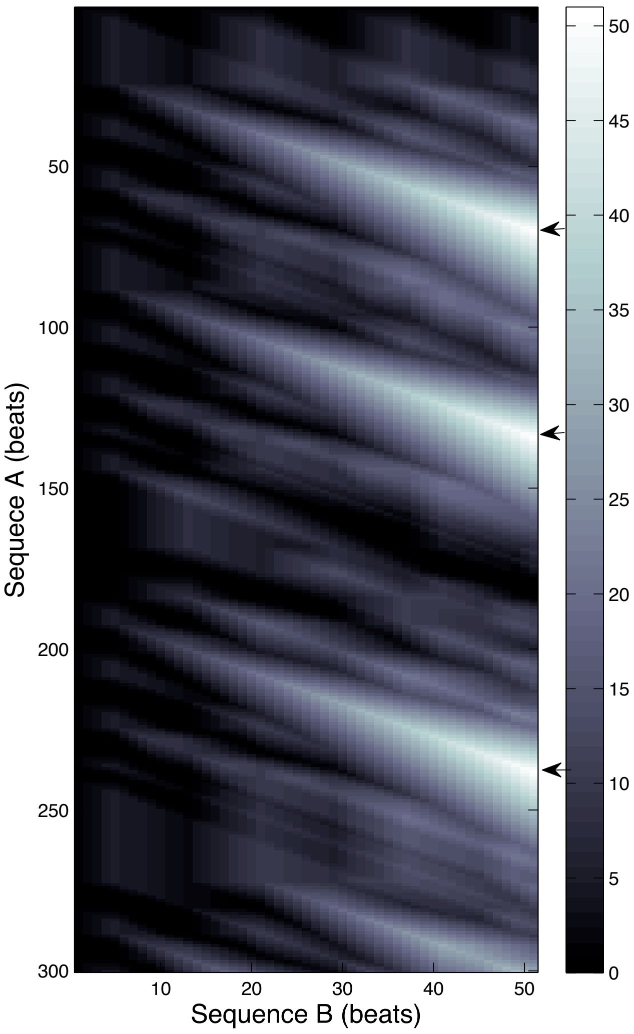

Fig. 3. *The matrix resulting from the comparison of the most recent 300 beats of a performance with the most recent 50. As can be seen in the final column of the matrix, there are three potential alignments (identified by arrows) for the fragment indicating that the fragment contains a repeated part of the performance.*

$$H_{i,j} = \max \begin{cases} H_{i-1,j-1} + s(i,j) \\ H_{i-1,j} - W \\ H_{i,j-1} - W \\ 0 \end{cases} \tag{5}$$

where $1 \leq i \leq N$, $1 \leq j \leq M$, and W is the gap penalty which penalises alignments that contain inserted or deleted elements. We choose $W = \frac{4}{3}$ in a similar way to [24]. Figure 3 shows an example of an alignment matrix resulting from the comparison of two sequences.

Our next step is to find a likely candidate for the next element in the beat-synchronous sequence. Using the alignment matrix H, this involves finding points where the last elements of sequence B align strongly with sequence A.

The strength of alignment is based upon the context of subsequence matches between the two sequences. In the example in Figure 3, there are three potential alignments shown by the strong diagonals and identified by arrows. To find these points of strong alignment we reduce our search to the final column of the alignment matrix and choose the value:

$$y = \arg \max_{1 \leq i < (N-\beta)} H_{i,M} \tag{6}$$

where β is a value that allows us to not search the most recent β elements in the long term memory for a strong alignment. This is because harmonic content can be slow in changing and so recent content may provide strong alignments which can be misleading. This essentially allows us to indicate that repetitions as recent as β beat-synchronous vectors ago are unlikely and so should not be favoured over other, longer term plausible alignments. Smaller values of β allow shorter repeats to be identified, larger values allow quicker identification of correct repeats, assuming they are repeated after a longer period than β. We choose $\beta = 10$.

Finally we predict the next element, $\hat{b}_{M+1}$, of the sequence B to be:

$$\hat{b}_{M+1} = a_{y+1} \tag{7}$$

It is possible that there will be two or more equally strong alignments and the choice of either the first, last or some other alignment is a design choice.

In addition to the polyphonic approach presented here, we have also previously presented a version of this technique for sequences of single discrete symbols such as chords or pitches [48] which could be used in a purely monophonic implementation.

VI. Evaluation

A. Musical Sequence Alignment Database

To evaluate our system, we asked a number of musicians to compose acoustic guitar pieces. The nature of the content, such as the repetition of themes, was not mentioned to the musicians. The result is a database of 32 pieces in rock, pop and folk styles comprising over 104 minutes of audio, recorded in mono, 16-bit audio at 44.1kHz. The mean length of each piece was 3'16 with a standard deviation of 36 seconds. This database has been made publicly available under a Creative Commons license, along with a real-time implementation in Max/MSP and Matlab source code both for our technique and for performing this evaluation.[1]

For each audio file, the beats were labelled using a combination of automated and human annotation to ensure that they were correct.

When assessing the similarity of predicted to observed harmonic vectors, our initial tests showed that distance measures such as the Euclidean distance did not necessarily show a small distance between vectors that were harmonically similar to the human ear. This may have been due to variations in the intensity of the performance or the octave at which a harmonic sequence was played.

[1]http://www.eecs.qmul.ac.uk/~adams/pf/

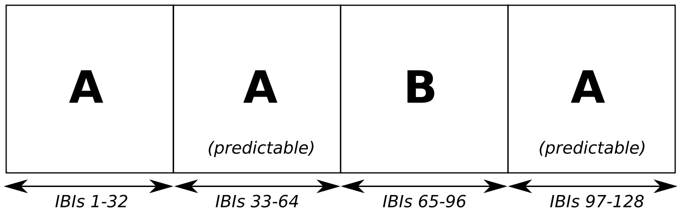

Fig. 4. The content of inter-beat intervals (IBIs) 1-32 is repeated from 33-64 and 97-128. This is shown as the three occurrences of section A. In this example, the content of inter-beat intervals 65-96, labelled B, is not repeated elsewhere and so is not 'predictable'. Overall in this example, 50% of content is 'predictable'.

As a result, in order to evaluate the performance of our system objectively, we have created a ground truth as follows. For each audio file, we have grouped repeated sections together. For example, the content of inter-beat intervals (IBIs) 1 to 32 may be repeated during 33 to 64 and 97 to 128. This is depicted in Figure 4. As an annotation, we record that information as follows:

[[1,32],[33,64],[97,128]].

From this, we can say that an acceptable prediction for IBI 33 is IBI 1, for IBI 34 we accept a prediction of IBI 2 and so on. For the later repeats, we can have two or more acceptable predictions - for IBI 97 we can accept predictions of IBI 1 or IBI 33. From this we develop a list of acceptable predictions (APs) for each IBI:

IBI	**APs**
1:	none
2:	none
⋮	⋮
32:	none
33:	1
34:	2
⋮	⋮
97:	1,33
98:	2,34
⋮	⋮

To perform an evaluation on the database, we assess predictions as either correct or incorrect depending on whether the prediction made is in the list of acceptable predictions for each IBI. We record, for each audio file, the percentage of correctly predicted IBIs. Some IBIs have no acceptable predictions and are therefore not 'predictable' - for example, the IBIs for the first instance of a pattern would not be 'predictable' as it would not have occurred before. We do not assess the performance of our system on these as the results for each audio file would vary greatly depending on the amount of repeated content.

While the stylistic content of the database is similar to much popular music, it is interesting to discover that 94.1% of the content is made up of sections that are repeated elsewhere in the piece and 76.2% is 'predictable' in the terms that we have described above.

TABLE I
THE RESULTS OF EVALUATING OUR PERFORMANCE FOLLOWING TECHNIQUE (*PF*), A FACTOR ORACLE (*FO*), AN N-GRAM MODEL (*N-Gram*) AND A RANDOM PREDICTOR (*Random*) ON A DATABASE OF 32 ANNOTATED AUDIO FILES. RESULTS ARE GIVEN AS MEAN PERCENTAGE SCORES (%) AND THE STANDARD DEVIATION OF THOSE SCORES (σ). THE PARAMETER M IS THE SHORT-TERM MEMORY LENGTH OF THE PERFORMANCE FOLLOWING TECHNIQUE, THE PARAMETER D IS THE NUMBER OF CLUSTERS USED IN THE K-MEANS CLUSTERING FOR THE N-GRAM AND FACTOR ORACLE MODELS, AND THE PARAMETER L IS THE LENGTH USED IN THE N-GRAM MODEL.

Model	Parameter	Score / %	σ / %
N-Gram	$L = 3, D = 13$	34.8	15.4
N-Gram	$L = 5, D = 6$	38.0	16.9
N-Gram	$L = 10, D = 3$	42.1	17.7
N-Gram	$L = 15, D = 3$	38.0	20.8
FO	$D = 4$	63.0	19.1
FO	Best D Per File	68.5	16.6
PF	$M = 5$	50.6	17.4
PF	$M = 10$	69.6	17.6
PF	$M = 15$	75.1	14.3
PF	**M = 20**	**75.3**	**13.9**
PF	$M = 25$	74.3	13.9
PF	$M = 30$	73.2	14.2
PF	$M = 35$	72.6	14.4
PF	$M = 40$	71.5	15.0
PF	$M = 45$	70.4	15.4
PF	$M = 50$	69.3	16.0
Random	-	2.2	1.4

B. Methodology

In evaluating our technique we were faced with the problem that if the beats are incorrectly tracked then it becomes impossible to tell whether errors are due to poor localisation of harmonic content or poor performance of our performance following technique itself. As a result, we decided to use the beat times from the database, rather than allow for the errors that may occur using only an automated beat tracker.

It is accepted that, should a beat tracker be used in a real-time context, inaccurate beat-tracking will have a detrimental affect on performance. However, reasonably accurate real-time beat-tracking models exist [45] and it is hoped that more accurate models will be produced in future. It is also perfectly possible to largely avoid this problem by playing to constant tempo click tracks.

We attempted to make predictions for each audio file as follows. If, for each audio file, we have R beats, γ_r, then we will have $R - 1$ inter-beat intervals. From these inter-beat intervals we calculate a sequence of beat-sychronous chroma features as described in section IV. We then use our performance following technique from section V to attempt to predict acceptable harmonic content from the past of the performance. We record the index number of the inter-beat intervals for predicted beat-synchronous chroma features. We achieve a score by comparing these predictions to the ground truth annotations of the database.

C. Comparison to Other Techniques

We also compared our technique to three others. As a baseline, we implemented a random predictor that would predict a random inter-beat interval from the past of the performance as a prediction of future content.

The second was based on a Factor Oracle (FO) implemented according to [18], using K-means clustering to group individual beat-synchronous chroma vectors into clusters so that a model with discrete states such as a FO could process the data. By following suffix links we identified previous repeated factors and used these to make predictions of past IBIs with similar harmonic content. We have presented the results for the best single number of clusters (4 clusters) and the results for a version where the best number of clusters is chosen for each file. As some random initialisation is involved with the K-means clustering, we ran the results 20 times and took the mean for each file.

The third technique was based on an N-gram model [49, Ch. 6]. We again used K-means clustering to create a sequence of discrete symbols. Using an N-gram model we modelled the sequences as they were processed in real-time - recording the best transition given a certain sequence. The recorded prediction was the inter-beat interval number from the past of the performance for the transition the last time that the given sequence occurred. Specifically, we chose lengths L of 3, 5, 10 and 15 for the N-gram model. We used the single best number of clusters for each value of L and as with the Factor Oracle approach, each result was the average over 20 runs to compensate for random initialisations in the clustering.

VII. RESULTS AND DISCUSSION

Our results are displayed in Table I. The best result is for our performance following technique (PF), with the value of the short-term memory length $M = 20$, which achieves a mean score of 75.3% of predicted IBIs. Given the standard deviation of 13.9 and the sample size, the 95% confidence interval [50] for this result is [75.3 $\pm$ 4.81]. In addition, we are able to predict a majority of content overall - 57.4% - given that our mean score is only on the 'predictable' 76.2% of the database.

Furthermore, compared to the other models, our performance following technique outperforms the Factor Oracle, N-gram model and the random predictor. From this we can conclude that our system works well.

A. Short-Term Memory Length (M)

There are, however, several potential areas for improvement. While the best value on average of the short-term memory length, M, was 20 in our evaluation, different pieces reached their maximum scores with different values of M. Figure 5 shows the number of examples achieving their maximum score for a given value of M. We can see that the majority of examples achieved higher scores with values of M other than 20. Ten of them achieved their best score when $M = 15$. But another 19 examples (59%) were best suited to values of M that were neither 15 or 20.

The implication of these findings is that improvements could be made by implementing M as an adaptive parameter that

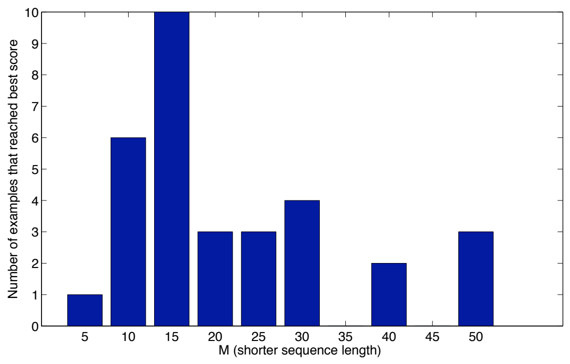

Fig. 5. The number of examples achieving their maximum score for a given value of M, the short-term memory length.

adjusts its value to suit the characteristics of the piece in question. Indeed, if we had managed to use the best version of M for each example, our performance would have increased by over 3% with a 95% confidence interval of [78.7% $\pm$ 4.08].

From informal use of the system in a real-time context, it appears that smaller values of M, i.e. smaller short-term memory lengths, cause the system to be more responsive to fast changes in the music. Meanwhile, larger values of M provide more context and therefore allow predictions based upon deeper analysis of the past of the performance.

Also, while our system can pick up on repetitions of varying lengths up to a size of M (due to the ability to match sub-sequences), our system may benefit from the combination of several performance following models with varying short-term sequence lengths M to capture information about changes happening at different rates in the performance.

B. The Selection of Features

The use of only harmonic features, such as chroma features, is a limitation on the system. In particular, there are many other aspects of music, such as rhythmic and timbral information, that are perceived as being repeated by humans and that are also useful in identifying repeated sections in musical performances.

The use of beat-synchronous chroma features has many benefits to the system. In addition to those mentioned in section IV-A, the beat-synchronous nature of these features acts as a form of temporal smoothing, while the chroma features, independent of octave, act similarly on harmonic data. The result is that we have been able to compare audio that may vary slightly but otherwise be considered similar and comparable by human ears. An example is the comparison of a strummed guitar chord progression compared to an arpeggiated version of that same chord progression – i.e. the same chord progression played in different styles.

There are, however, limitations to using beat-synchronous chroma features. We may wish to have segmentation at metrical levels that give us more detail in terms of the harmonic development of the performance as many harmonic events occur at rates faster than the tactus. However, in order to obtain a required frequency resolution, we must use an audio frame size of a certain size. If the inter-beat intervals are too short – due to a combination of the tempo and choice of metrical level – then we may be unable to achieve the required frequency resolution. While this could be partially solved using larger buffers with overlaps, the results would be to ‘blur’ representations across beat boundaries.

C. Evaluation of the Effect of Beat Tracking Errors

In order to migrate fully from the use of fixed tempo click tracks to a real-time beat tracker, it would be desirable to perform an evaluation that examined the effect of beat tracking errors on the system. Given our use of a ground truth based upon annotated beat locations, such a comparison is difficult as a beat tracker may erroneously identify beats and thereby make impossible a comparison using the ground truth. We have also mentioned problems encountered using distance measures such as the Euclidean distance in section VI-A.

Such an evaluation would also allow us to assess the useful values of the gap penalty, W. This is used rarely in the current evaluation as it is designed to deal with beat tracking errors - aligning sequences where there are 5 beats where there should be 4, for example. Given that we are using ground truth beat locations we do not encounter such errors in the current evaluation. We will assess ways of performing an evaluation using real-time beat trackers in future work.

D. The Annotation Process

An aspect of our evaluation that is worthy of more study is the annotation process. In the current study, repeated sections were identified through the subjective human perception of a musician. However, there are multiple potential interpretations of what can be considered to be a ‘repeated section’. A formal definition of a repeated section would be ideal, but due to the variability and complexity of music, it is complicated to define (and as complicated to annotate for a human).

For example, if one IBI was the same as the previous one, it could technically be called a repeated section. However, this would not be a definition of a repeated section that most humans would agree with due to its brevity. Likewise, the suggestion that a certain number of IBIs have to be repeated before we can label a section as ‘repeated’ is undermined by the potential for beat tracking at different metrical levels and the different time signatures of music. Another option would be to say that there must be a certain number of harmonic changes. However, if the music is more complicated than a simple chord sequence then harmonic changes will be more difficult to identify.

E. Scope of the Work

Our technique is specifically focused upon music that is based, at least partially, on the repetition of musical patterns, such as those that occur in the rock, pop and folk genres. Therefore we make no claim that our technique is extendable to all musical styles.

We also defend the use of a single instrument, such as a guitar, as the input signal. In a live performance context, it is much easier to access a good quality audio signal from an accompanying instrument such as a guitar than it is to acquire a good quality signal of an entire ensemble and to then deal with the multi-instrumental nature of that signal. Our use of only a single instrument, which may be part of a larger ensemble, has precedents in research on beat tracking of drums in live performances [2] and our own work on beat tracking informed audio effects for guitar [51].

F. Application: Automatic Bassline Accompaniment

Here we outline an application for our performance following technique. For our polyphonic performance following technique, we are given, at each beat, a prediction of the harmonic content of the next inter-beat interval in the form of a chroma vector. Our implemented application is an automatic bassline accompaniment. This involves playing the root note associated with each predicted chroma vector. We achieve this by classifying the chroma vector using our previous chord recognition technique [47]. We then use the root note of this chord to produce an accompaniment using a synthesiser.

VIII. Directions for Future Work

We have already outlined several areas for future study, including an adaptive short-term memory length M, the inclusion of musical features other than just harmonic features and a deeper study of the annotation process for evaluations of the kind we have performed.

While our system performed well in the evaluation, we believe that it must be improved further in order to be a truly robust live performance tool. Furthermore, as it currently only makes predictions based upon the presence of repetition, its integration into a larger system capable of making predictions in the absence of repetition would also be useful. Finally, integration of some form of external meta-data would be useful so that performing musicians could indicate imminent sharp changes in tempo or other unpredictable musical changes.

Another clear contender for future study is the generation of more complex accompaniments. The current real-time implementation only provides an accompaniment in the form of a synthesised root note bassline, as discussed in section VII-F. It would be interesting to examine more 'musical' accompaniments that take full advantage of the information contained in the sequence of chroma features predicted by the system - perhaps looking ahead by several beats to make more informed choices (although the accuracy of longer term predictions would need to be evaluated). Such accompaniments may involve arpeggiated chords, or perhaps long slow orchestral accompaniments, with harmonies complimenting the harmonic content of the piece.

It would also be interesting to make adjustments to the representations, or the comparisons between them, so that transposed versions of the same theme could be recognised by our technique.

IX. Conclusion

We have presented a technique that is capable of predicting harmonic content in musical performances based upon the detection of repeated musical themes. Using a combination of beat-synchronous chroma feature representations and a dynamic programming alignment algorithm, we have placed short term musical content in the context of the whole performance to allow predictions of future musical content to be made.

On an evaluation of polyphonic audio files, we have shown that we are able to predict the large majority of repeated content – and a majority of the content overall – with no additional information in the form of a musical score.

Reproducible Research

We have made available, under a Creative Commons license, both the data set and Matlab source code so that our results can be independently reproduced. We have also made available a real-time implementation of our technique for the Max/MSP environment. See section VI-A for more information.

Acknowledgment

The authors would like to warmly thank the following individuals for contributing compositions for use in the evaluation: Andrew Robertson, Claire Robbin, Jamie Dale, Peter Greliak, Philip Bazany, Simon Stark, Steve Willey and Tom Barton.

References

[1] C. Raphael and Y. Gu, "Orchestral accompaniment for a reproducing piano," in *Proceedings of International Computer Music Conference*, 2009.

[2] A. Robertson and M. D. Plumbley, "B-Keeper: A beat-tracker for live performance," in *New Interfaces for Musical Expression*, June 2007, pp. 234–237.

[3] IRCAM. (2010, July) Antescofo score following repertoire. [Online]. Available: http://imtr.ircam.fr/imtr/Scorefollowing_Repertoire

[4] N. Orio, S. Lemouton, and D. Schwarz, "Score following: State of the art and new developments," in *New Interfaces for Musical Expression*, 2003.

[5] A. Ockelford, *Repetition in Music: Theoretical and Metatheoretical Perspectives*, M. Structure, Ed. Ashgate Publishing Limited, 2005.

[6] M. Mongeau and D. Sankoff, "Comparison of musical sequences," *Computers and the Humanities*, vol. 24, no. 3, pp. 161–175, June 1990.

[7] B. Pardo and W. P. Birmingham, "Following a musical performance from a partially specified score," in *Proceedings of the 2001 Multimedia Technology and Applications Conference*, 2001, pp. 202–207.

[8] R. B. Dannenberg and N. Hu, "Polyphonic audio matching for score following and intelligent audio editors," in *Proceedings of International Computer Music Conference*, 2003, pp. 27–33.

[9] S. Dixon and G. Widmer, "MATCH: A music alignment tool chest," in *Proceedings of International Conference on Music Information Retrieval*, 2005, pp. 492–497.

[10] S. Dubnov, "Unified view of prediction and repetition structure in audio signals with application to interest point detection," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 16, no. 2, pp. 327–337, February 2008.

[11] S. Abdallah and M. D. Plumbley, "Information dynamics: Patterns of expectation and surprise in the perception of music," *Connection Science*, vol. 21, no. 2, pp. 89–117, 2009.

[12] A. Cont, S. Dubnov, and G. Assayag, "Anticipatory model of musical style imitation using collaborative and competitive reinforcement learning," in *Anticipatory Behavior in Adaptive Learning Systems*, M. Butz, O. Sigaud, and G. Baldassarre, Eds. Springer Verlag, 2007.

[13] A. Cont, "Modeling musical anticipation: From the time of music to the music of time," Ph.D. dissertation, University of Paris 6 and University of California in San Diego, October 2008.

[14] B. Thom, "Predicting chordal transitions in jazz: the good, the bad, and the ugly," in *Proceedings of IJCAI Workshop on AI and Music, International Joint Conference on Artificial Intelligence*, 1995.

[15] T. Kohonen, “A self-learning musical grammar, or ‘associative memory of the second kind’,” in *Proceedings of International Joint Conference on Neural Networks*, vol. 1, June 1989, pp. 1–5.

[16] F. Pachet, “The Continuator: Musical interaction with style,” in *Proceedings of International Computer Music Conference*, September 2002, pp. 211–218.

[17] D. Conklin, “Music generation from statistical models,” in *AISB 2003 Symposium on Artificial Intelli- gence and Creativity in the Arts and Sciences*, Aberystwyth, Wales, 2003, pp. 30–35.

[18] G. Assayag and S. Dubnov, “Using factor oracles for machine improvisation,” *Soft Computing*, vol. 8, no. 9, pp. 604–610, September 2004.

[19] C. Allauzen, M. Crochemore, and M. Raffinot, “Factor oracle: A new structure for pattern matching,” in *Proceedings of SOFSEM’99, Theory and Practice of Informatics*, ser. Lecture Notes in Computer Science, J. Pavelka, G. Tel, and M. Bartosek, Eds., 1999, pp. 291–306.

[20] G. Assayag, G. Bloch, and M. Chemillier, “OMAX-OFON,” in *Sound and Music Computing Conference*, 2006.

[21] S. Dubnov, G. Assayag, and A. Cont, “Audio oracle: A new algorithm for fast learning of audio structures,” in *Proceedings of International Computer Music Conference*, 2007.

[22] G. Bloch, S. Dubnov, and G. Assayag, “Introducing video features and spectral descriptors into the omax improvisation system,” in *Proceedings of International Computer Music Conference*, 2008.

[23] S. B. Needleman and C. D. Wunsch, “A general method applicable to the search for similarities in the amino acid sequence of two protiens,” *Journal of Molecular Biology*, vol. 48, no. 3, pp. 443–453, March 1970.

[24] T. F. Smith and M. S. Waterman, “Identification of common molecular subsequences,” *Journal of Molecular Biology*, vol. 147, no. 1, pp. 195–197, March 1981.

[25] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, “Basic local alignment search tool,” *Journal of Molecular Biology*, vol. 215, pp. 403–410, 1990.

[26] S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, “Gapped BLAST and PSI-BLAST: A new generation of protein database search programs,” *Nucleic Acids Research*, vol. 25, no. 17, pp. 3389–3402, 1997.

[27] N. Hu, R. B. Dannenberg, and G. Tzanetakis, “Polyphonic audio matching and alignment for music retrieval,” in *2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2003, pp. 185–188.

[28] P. Ferraro and P. Hanna, “Optimizations of local edition for evaluating similarity between monophonic musical sequences,” in *Proceedings of the 8th International Conference on Information Retrieval (RIAO)*, 2007.

[29] M. Robine, P. Hanna, and P. Ferraro, “Music similarity: Improvements of edit-based algorithms by considering music theory,” in *Proceedings of the ACM SIGMM International Workshop on Multimedia Information Retrieval*, 2007, pp. 135–141.

[30] J. Kilian and H. H. Hoos, “MusicBLAST - gapped sequence alignment for MIR,” in *Proceedings of International Conference on Music Information Retrieval*, 2004, pp. 38–41.

[31] R. B. Dannenberg and N. Hu, “Pattern discovery techniques for music audio,” in *Proceedings of International Conference on Music Information Retrieval*, 2002, pp. 63–70.

[32] R. B. Dannenberg, “An on-line algorithm for real-time accompaniment,” in *International Computer Music Conference*, 1984, pp. 193–198.

[33] J. J. Bloch and R. B. Dannenberg, “Real-time computer accompaniment of keyboard performances,” in *Proceedings of International Computer Music Conference*, 1985, pp. 279–289.

[34] S. Dixon, “Live tracking of musical performances using on-line time warping,” in *Proceedings of International Conference on Digital Audio Effects*, 2005, pp. 92–97.

[35] C. Raphael, “Music Plus One: A system for flexible and expressive musical accompaniment,” in *International Computer Music Conference*, Havana, Cuba, 2001.

[36] L. R. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition,” *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, February 1989.

[37] A. Cont, “A coupled duration-focused architecture for realtime music to score alignment,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 6, June 2010.

[38] I. Simon, D. Morris, and S. Basu, “MySong: Automatic accompaniment generation for vocal melodies,” in *Proceedings of CHI 2008 Conference on Human Factors in Computing Systems*, 2008.

[39] D. Moelants and M. F. McKinney, “Tempo perception and musical content: What makes a piece fast, slow or temporally ambiguous?” in *Proceedings of the 8th International Conference on Music Perception and Cognition*, 2004, pp. 558–562.

[40] D. P. W. Ellis, “Identifying ‘cover songs’ with beat-synchronous chroma features,” in *Music Information Retrieval Evaluation eXchange*, 2006.

[41] M. A. Bartsch and G. H. Wakefield, “To catch a chorus: Using chroma-based representations for audio thumbnailing,” in *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2001, pp. 15–18.

[42] M. Levy and M. B. Sandler, “Structural segmentation of musical audio by constrained clustering,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 16, no. 2, pp. 318–326, February 2008.

[43] N. Schnell, D. Schwarz, and R. Müller, “X-Micks - interactive content based real-time audio processing,” in *Proceedings of International Conference on Digital Audio Effects*, 2006.

[44] J. P. Bello and J. Pickens, “A robust mid-level representation for harmonic content in music signals,” in *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*, London, UK, 2005, pp. 304–311.

[45] A. M. Stark, M. E. Davies, and M. D. Plumbley, “Real-time beat-synchronous analysis of musical audio,” in *Proceedings of International Conference on Digital Audio Effects*, 2009, pp. 299–304.

[46] T. Fujishima, “Real-time chord recognition of musical sound: A system using common lisp music,” in *Proceedings of International Computer Music Conference*, 1999, pp. 464–467.

[47] A. M. Stark and M. D. Plumbley, “Real-time chord recognition for live performance,” in *Proceedings of International Computer Music Conference*, 2009.

[48] ——, “Performance following: Tracking a performance without a score,” in *Proceedings of International Conference on Acoustics, Speech and Signal Processing*, March 2010, pp. 2482–2485.

[49] D. Jurafsky and J. H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition*. Prentice Hall, 2000.

[50] A. Flexer, “Statistical evaluation of music information retrieval experiments,” *Journal of New Music Research*, vol. 35, no. 2, pp. 113–120, 2006.

[51] A. M. Stark, M. E. Davies, and M. D. Plumbley, “Rhythmic analysis for real-time audio effects,” in *Proceedings of International Computer Music Conference*, Belfast, 2008.

Adam M. Stark was born in 1983 in London, UK. He received the B.Sc. degree in Computer Science from Royal Holloway, University of London in 2005 and the M.Sc. degree in Digital Music Processing from Queen Mary University of London in 2006. He is currently studying for a Ph.D. in the analysis of musical audio in live music performances at Queen Mary University of London.

His research interests include beat tracking, harmonic analysis and musical systems for live performance such as beat tracking informed audio effects and score following systems.

Mark D. Plumbley (S’88–M’90) received the B.A. (Hons.) degree in electrical sciences in 1984 from the University of Cambridge, Cambridge, U.K., and the Ph.D. degree in neural networks in 1991, also from the University of Cambridge. From 1991 to 2001 he was a Lecturer at King’s College London. He moved to Queen Mary University of London in 2002, and where he is now an EPSRC Leadership Fellow and Director of the Centre for Digital Music. His research focuses on the automatic analysis of music and other audio sounds, including automatic music transcription, beat tracking, and audio source separation, and with interest in the use of techniques such as independent component analysis (ICA) and sparse representations. Prof. Plumbley chairs the ICA Steering Committee, and is a member of the IEEE SPS TC on Audio and Acoustic Signal Processing.