An Online Tempo Tracker for Automatic Accompaniment based on Audio-to-audio Alignment and Beat Tracking

Grigore Burloiu

An Online Tempo Tracker for Automatic Accompaniment based on Audio-to-audio Alignment and Beat Tracking

Grigore Burloiu
Faculty of Electronics, Telecommunications and Information Technology
University Politehnica of Bucharest
gburloiu@gmail.com

ABSTRACT

We approach a specific scenario in real-time performance following for automatic accompaniment, where a relative tempo value is derived from the deviation between a live target performance and a stored reference, to drive the playback speed of an accompaniment track. We introduce a system which combines an online alignment process with a beat tracker. The former aligns the target performance to the reference without resorting to any symbolic information. The latter utilises the beat positions detected in the accompaniment, reference and target tracks to (1) improve the robustness of the alignment-based tempo model and (2) take over the tempo computation in segments when the alignment error is likely high. While other systems exist that handle structural deviations and mistakes in a performance, the portions of time where the aligner is attempting to find the correct hypothesis can produce erratic tempo values. Our proposed system, publicly available as a Max/MSP external object, addresses this problem.

1. INTRODUCTION

The task of online tempo tracking refers to the computation of a realistic tempo curve from an incoming audio stream in real time, relative to a reference. In this paper we focus on the application of tempo tracking in a musical context where a performer plays together with a responsive accompaniment track, with both the system reacting to live tempo fluctuations and acting as a steady backdrop for the musician to anchor herself to when needed. To this end, we examine two paradigms for tempo tracking—online audio-to-audio alignment and beat tracking—and propose a system that uses both, harnessing their respective strengths.

In the literature, audio alignment is generally part of *score following* systems [1–3], which gradually build a best-fit path matching the incoming live audio to a reference. This path is likely to contain intermittently unnatural slopes, which is why *tempo models* are needed to translate the path slope into a realistic relative tempo value [2]. A typical application for a score following task is a concerto simulation, where a solo performance and drives a machine-generated accompaniment, which faithfully responds to the musician's tempo progression. In cases of performance errors or structural differences such as jumps or repeats, online alignment algorithms can temporarily produce erratic local outputs. Tempo models can alleviate the problem by smoothing over minor errors, but the larger issue of systematic error detection and correction remains.

Meanwhile, beat tracking systems compute tempo curves by detecting the dominant metrical pulse in the live input [4, 5]. As such, they are geared towards scenarios with prominent periodic beats, where the live musician might be locked in with a rhythmic backing track. Wrong notes do not affect the tempo tracking performance, and even when the musician deviates from the rhythmic pattern, beat trackers produce a reasonably consistent tempo line. However, even though they might be configured to drive an external sequencer [4], beat tracking systems themselves do not have information about the material being played. Thus minor deviations can cause them to fall on an upbeat, and structural changes such as the omission of a bar of music are impossible to detect.

We introduce a system that tackles the issue of errors and structural deviations in music with a strong rhythmic pulse, by tracking tempo in two ways: a model based on audio-to-audio alignment by default, and a beat tracking-based model which takes over when the alignment path becomes erratic. The resulting machine can drive an accompaniment track based only on audio inputs, without the use of a score or any other symbolic ground truth information. The practical justification is that, for reasons of expediency, lack of access, or incompatibility with the material, a musician might rather record a reference performance of a part instead of plugging in a MIDI or MusicXML symbolic score.

An early version of our proposed system was presented in [3]. The application and its source are available [1] as a Max/MSP [2] external object; to our knowledge, it is the only online audio-to-audio alignment tool for Max to date.

The rest of the paper is organised as follows: section 2 is a brief review of relevant work. In section 3 we describe our alignment framework and in section 4 we introduce the beat tracking component and describe its integration into the system. Section 5 is a case study of the switching mechanism. We conclude with a discussion of the proposed system and future research directions in section 6.

[1] See https://github.com/RVirmoors/RVdtw-.

[2] Max is a state-of-the-art programming environment for realtime multimedia performance; see http://cycling74.com/.

2. RELATED WORK

Several studies have addressed the problems of performance variation and error, and of structural differences in the context of music alignment. In the case of alignment to a symbolic score, the problem of jumps is relatively easily solved by marking points of possible divergence [6–8]. For the audio-to-audio alignment of one performance to another, which is the focus of our paper, methods based on dynamic time warping (DTW) [9] and variations thereof [10] have shown good results in the offline case. For real-time situations, audio-to-audio alignment has been implemented using strategies based on online DTW [1, 2, 11] or particle filtering [12, 13].

Notably, structural differences are specifically addressed in [2, 13], which continuously monitor different positions in the score in parallel, to account for jumps. These methods, while performing well for tasks such as real-time page turning and annotation, are however not optimised for automatic accompaniment. There is no mechanism to ensure a musically useful tempo during time periods of incorrect alignment caused by mistakes, jumps, or improvisation.

Meanwhile, beat trackers have been at the core of "performance following" [14] tasks such as generative drum accompaniment [4] or tonal performance tracking [14, 15]. Such applications indicate that beat trackers are able to drive a performance forward over periods where the live musician strays from the original reference, which is the main insight driving this paper.

3. PERFORMANCE AUDIO ALIGNMENT

In this section we describe our pre-existing accompaniment framework, as expanded from [3]. It consists of an online audio-to-audio aligner and a tempo model.

3.1 Online DTW-based Follower

Our system computes the alignment path based on a variant of online DTW [1]. The basic algorithm aligns a *target* time series $X = x_1 \dots x_m$ to a *reference* $Y = y_1 \dots y_n$, where X is partially unknown: only the first t values are known at a certain point. The goal is for each $t = 1 \dots m$ to find the corresponding index h_t, so that the subsequence $y_1 \dots y_{h_t}$ is best aligned [3] to $x_1 \dots x_t$. The alignment path p is a sequence of (t, h_t) tuples connecting the origin with the current position for the lowest global match cost. For audio alignment tasks, all x_i and y_j are audio feature vectors; in this case we use chromagrams, as computed in [16].

The alignment path p is defined as being monotonous and continuous, with a local slope constraint that prevents the follower from getting stuck in a local minimum or making steep jumps. For each incoming frame x_t, the local match cost matrix is computed for the past $c \times c$ frames and the path advances by computing a new row, a new column, or both, depending on where the current minimum match cost is found on the search window's frontier: a row, a column or the top-right corner, respectively.

[3] In cases where a frame x_t is assigned to several frames in Y, we set h_t to point to the last of these.

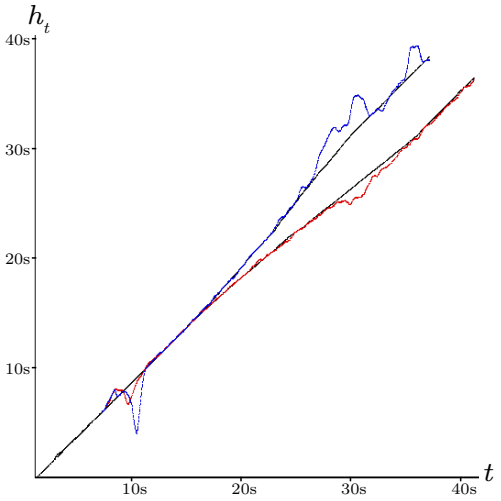

Figure 1: The online DTW-based audio alignment. Target time t progresses horizontally, reference time h_t vertically. Two test target alignment paths are shown in black. The deviations δ_t^{BACK} from the backwards path are added to each (t, h_t) tuple, in blue for the accelerating target and in red for the decelerating one.

Since the alignment engine is not the main focus of this paper, we will just briefly describe our main modifications to the classic online DTW method. For implementation details we refer the reader to [1, 3].

Our first change is to remove the path slope constraint, allowing the alignment path to trace along the X or Y axes for as long as necessary. To maintain stability, we compensate by adjusting the local match weighting coefficients to favour diagonal movement, and by adding a constant α to the local cost, minimising the influence of minor differences between target and reference feature vectors, which could have unpredictably diverted the path:

$$d(i,j) = \sqrt{\sum_k (x_{i,k} - y_{j,k})^2} + \alpha \ , \qquad (1)$$

Secondly, as inspired by [7], with each new input frame we compute a backwards, offline DTW path starting from the current (t, h_t) position, over a square-shaped match cost matrix covering around 5 seconds in the immediate past. Since both dimensions are now "known", the alignment is considered to be more accurate, and we are able to compute the distance δ_t^{BACK} between coordinates where the main alignment path and the backwards, corrective path reach the border of the window. Whenever this deviation δ_t^{BACK} exceeds a threshold $\epsilon = 50$ms, we adjust the weighting coefficients to favour a path in the respective direction. Figure 1 illustrates this behaviour for two test sequences: one gradually slows down, then carries on at 80% tempo before abruptly reverting to the original speed, and the other takes the opposite direction. It becomes evident how the deviation between the main and the backwards path fosters the different tempo regimes.

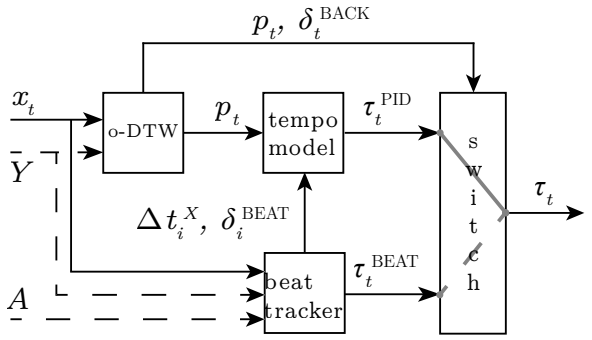

Figure 2: Framework architecture diagram. Dotted lines signify offline data pre-loading, regular arrows show online data flow. The beat tracker computes a tempo τ_t^{BEAT} based on the x_t input, and sends beat information to the alignment-based tempo model. The choice between τ_t^{BEAT} and τ_t^{PID} for tempo output is based on the alignment path p_t and its backwards error measure δ_t^{BACK}.

3.2 Tempo Model

In our system, for each time frame t the tempo model derives a relative tempo τ_t from the alignment path, so that:

$$\tilde{h}_t = \tilde{h}_{t-1} + \tau_t \ , \tag{2}$$

where $\tilde{h}_t \in \mathbb{Q}$ is the *accompaniment coordinate* in the reference series corresponding to the target frame t, and $\tilde{h}_0 = 0$.

Effectively, τ_t acts as a tempo *multiplier* in that it modulates the accompaniment playback speed. For each t, we define the *accompaniment error* ε_t as the difference between the alignment index and the accompaniment coordinate:

$$\varepsilon_t = h_t - \tilde{h}_t \ . \tag{3}$$

Our system contains four different tempo models, which the user can choose between. Hereon we describe and employ the *PID model*, inspired by the proportional-integral-derivative controller [17], which is a simple way to efficiently model adaptation and anticipation relative to trends in the alignment path. The model has the following output:

$$u_d(t) = K_P \varepsilon_t + K_I \sum_{i=0}^{t} \varepsilon_i + K_D \frac{\varepsilon_t - \varepsilon_{t-\Delta t}}{\Delta t} \ , \tag{4}$$

which produces the tempo multiplier:

$$\tau_t^{\text{PID}} = \frac{h_t - h_{t-\Delta t} + u_d(t)}{\Delta t} \ . \tag{5}$$

For the user-adjustable parameters, the Δt step has a default value of 500ms, and we found a good compromise between stability, response time and anticipation by setting: $K_P = 17 \times 10^{-3}$, $K_I = 3 \times 10^{-4}$, $K_D = 0.4$.

Finally, a *sensitivity* parameter S produces the threshold value ϵ^{S} which the accompaniment error ε_t needs to reach in order to activate the tempo model. For an appropriate sensitivity value, the system ignores minor fluctuations in the alignment path slope and holds the tempo steady at the last computed value. The threshold falls quadratically from one second to zero with the rise of sensitivity from 0 to 1:

$$\epsilon^{\text{S}} = (1 - S)^2 \times 100 \ . \tag{6}$$

The impact of the S setting is seen in Figures 4 and 6, where the time segments with a constant relative tempo have no background shading.

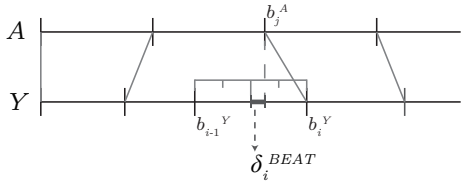

Figure 3: Computing the distance δ_i^{BEAT} between beats in the reference audio Y and their closest correspondents in the accompaniment track A. In this example, b_i^Y falls closer to an up-beat, so we translate it by $\frac{\Delta t_i^Y}{2}$ towards the closest upbeat b_j^A before measuring the distance.

4. BEAT TRACKING COMPONENT

We integrated the beat tracker in [5], which is publicly available [4] as a C++ class. The diagram in Figure 2 shows how the beat tracking module fits into the larger system framework.

We distinguish between the offline phase, where the reference audio to be matched and the accompaniment track are pre-loaded into the beat tracker and their beat positions marked as b_i^Y and, respectively, b_i^A, and the online phase, where the target audio beats b_i^X are detected in real time. Beat durations are measured as:

$$\Delta t_i^{[X,Y,A]} = b_i^{[X,Y,A]} - b_{i-1}^{[X,Y,A]} \ . \tag{7}$$

The Δt_i^X values replace the tempo update interval Δt from Equation (5) in real time, ensuring that new tempo values are computed by the PID tempo model with each new detected live beat.

Furthermore in the offline phase, for every reference beat b_i^Y, we compute the distance δ_i^{BEAT} to its corresponding accompaniment beat b_j^A. We take into consideration the possibility of the beats being in reverse phase, so if the distance is larger than a quarter of the current reference beat duration Δt_i^Y, then we translate b_i^Y by half of Δt_i^Y before again computing the distance:

$$\delta_i^{\text{BEAT}} = \min(|b_i^Y - b_j^A|, |b_i^Y \pm \frac{\Delta t_i^Y}{2} - b_j^A|) \ . \tag{8}$$

The actuation of the $\pm$ operation depends on whether b_i^Y, being the closest reference beat to b_j^A, comes before or after b_j^A. This process is depicted in Figure 3, where b_i^Y is translated backwards.

[4] See `https://github.com/adamstark/BTrack`.

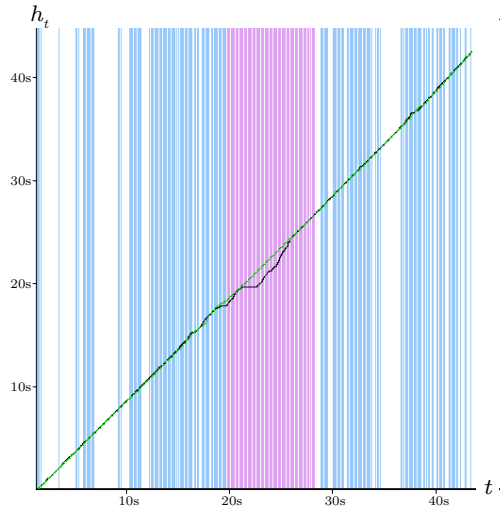

Figure 4: Online tempo tracking. The black line is the audio-to-audio alignment path; the green line traces the accompaniment coordinates produced by the system. Blue background shading marks time where τ_t^{PID} is in effect; purple marks τ_t^{BEAT} tempo; no background shading marks constant accompaniment tempo.

We use this reference-accompaniment beat distance δ_i^{BEAT} as a measure of how rhythmically "locked in" the reference part is to the backing track. This provides an indication of how closely we expect the live target to match the accompaniment beats, which makes it a good modulation factor for the tempo model's S sensitivity parameter. We rewrite Equation (6) as follows:

$$\epsilon_i^{\text{S}} = (1 - S)^2 \times 100 + \delta_i^{\text{BEAT}} \quad . \tag{9}$$

Now for each new b_i^Y beat reached, the tempo model's activation threshold moves depending on how tightly we expect the target to match the reference. Thus, for "loose" beats we raise the threshold, meaning that small deviations are ignored by the accompaniment and the tempo is kept constant. All the examples in this paper were realised using $S = 1$, which would have produced a global zero threshold under Equation (6).

The beat tracking module derives a local tempo BPM estimate from the observed beat durations [5]. To produce the relative tempo τ_t^{BEAT} that can drive the accompaniment, we divide the live tempo estimate by the tempo detected at the same beat in the accompaniment track. We define two situations where the system's relative tempo output switches from the alignment-based value τ_t^{PID} to the beat-based one τ_t^{BEAT}:

1. if the tempo produced is more than 3 times slower or faster than the reference, or

2. if the backwards DTW deviation δ_t^{BACK} exceeds a threshold $\epsilon^{\text{B}} = 174$ms, and the difference between the alignment slope [5] $\frac{dh}{dt}$ and τ_t^{BEAT} is greater than $\epsilon^{\text{T}} = 0.15$.

[5] smoothed over the last 40 frames, or 464ms.

Algorithm 1 Switching between the alignment-based tempo model τ_t^{PID} and the beat-based one τ_t^{BEAT}.

$calc \leftarrow$ NONE
$waiting \leftarrow 0$
$\tau_0 \leftarrow 1$
for all frames x_t in X **do**
  if $\varepsilon_t > \epsilon_i^{\text{S}}$ **then**
    if $(\delta_t^{\text{BACK}} > \epsilon^{\text{B}}$ **and** $|\frac{dh}{dt} - \tau_t^{\text{BEAT}}| > \epsilon^{\text{T}})$ **or**
    $(\tau_{t-1} \notin (\frac{1}{3}, 3))$ **then**
      $waiting \leftarrow \Delta t_i^A$
    end if
    if $waiting > 0$ **then**
      $\tau_t \leftarrow \tau_t^{\text{BEAT}}$ {*computed by the beat tracker.*}
      $calc \leftarrow$ BEAT
    else
      if $calc =$ BEAT **then**
        $\tilde{h}_t \leftarrow h_t$ {*jump back to alignment path pos.*}
      end if
      $\tau_t \leftarrow \tau_t^{\text{PID}}$ {*computed in Equation (5).*}
      $calc \leftarrow$ PID
    end if
  else if $\varepsilon_t \leq 1$ **and** $calc \neq$ NONE **then**
    $\tau_t \leftarrow \frac{dh}{dt}$ {*use current path slope.*}
    $calc \leftarrow$ NONE {*disable tempo model.*}
  end if
  if $waiting > 0$ **then**
    $waiting \leftarrow waiting - 1$
  end if
  $\tilde{h}_t \leftarrow \tilde{h}_{t-1} + \tau_t$
end for

When one of the two conditions is hit, the beat tracker drives the tempo for at least one Δt_i^A beat. Afterwards, if the conditions are inactive, the alignment-based tempo model regains control, and the accompaniment cursor $\tilde{h}_t$ jumps to the respective path position h_t. This entire switching algorithm is laid out in Listing 1.

We can follow the algorithm's execution through the example shown in Figure 4. For the first third of the run-through, the live target closely matches the reference. In the next third, the musician starts improvising, keeping the same tempo but diverging from the original pitches significantly. The $(\delta_t^{\text{BACK}} > \epsilon^{\text{B}}$ **and** $|\frac{dh}{dt} - \tau_t^{\text{BEAT}}| > \epsilon^{\text{T}})$ condition is activated and the tempo is now controlled by the beat tracker, $\tau_t \leftarrow \tau_t^{\text{BEAT}}$. The condition remains active until a few seconds after the musician has resumed playing the scored pitches. By that time, the online DTW path has rejoined the accompaniment path, so the transition back to the original tempo model is seamless.

In the next section we study a more challenging case and test the limits of our proposed accompaniment system.

5. CASE STUDY

The alignment performance of our online DTW engine has been evaluated in [3], with results on par with equivalent implementations. The beat tracker is evaluated in [5]. In order to thoroughly assess the performance of models presented in this paper, we would require experimental pro-

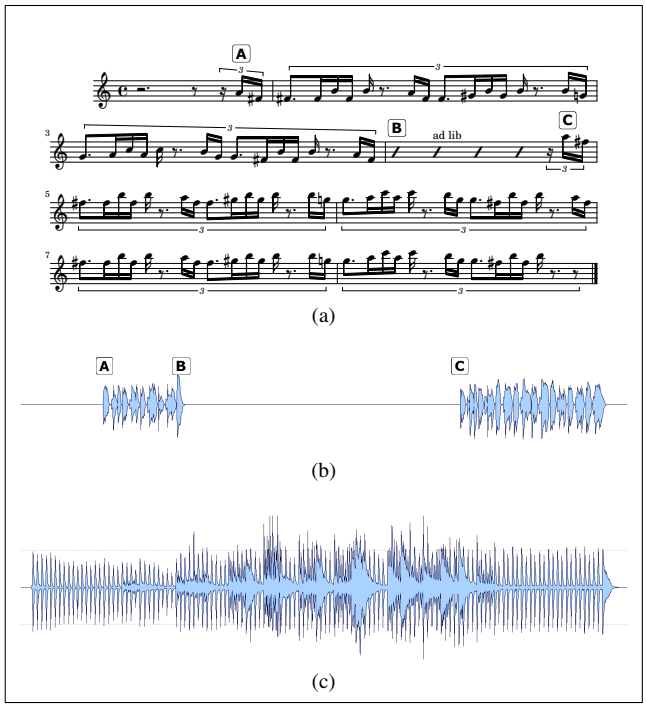

Figure 5: Example scenario. (a) lead instrument score, unknown by the system; (b) reference track; (c) accompaniment track. (b) and (c) are pre-loaded into the system.

cedures and reference benchmarks that exceed the current standard methods for score following or beat tracking tasks, which mostly measure keyframe-matching ability without specifically studying the musical quality of the resulting accompaniment.

The case for a wider test bench that includes not just several performances of a piece, but also the corresponding accompaniment tracks, is further strengthened by the beat-based tempo model we introduced in section 4, which strongly relies on backing track information. In the absence of such an evaluation database, a preliminary case study will demonstrate the qualitative improvements over our previous system and equivalent followers.

The materials we produced for this example scenario are presented in Figure 5. A backing track (drums) plays by itself for one measure to cue in the lead instrument (guitar). The two play together for one measure (section A), followed by an improvisation (section B) where the backing track keeps a steady beat. The lead instrument decides when to conclude the improvisation by entering the final two measures (section C).

There are two major questions that our system must answer, without referring to any symbolic information or programmed instructions: what to do when the musician starts improvising, and how to latch back on at the conclusion.

Figure 6 shows one realisation of the scenario, where the musician first misses the cue to start playing, which causes the alignment to remain stuck at the start of section A. Other online aligners might produce the same result, but our machine does not stop here: the beat tracker takes control, setting the tempo to 1, which allows the *waiting* variable to elapse during one beat. At the end of this beat, the accompaniment cursor goes back to the baseline, and the process is repeated. Thus, our musician (and the audience) is given a steady beat; once they start playing over it, the alignment-based tempo model regains control.

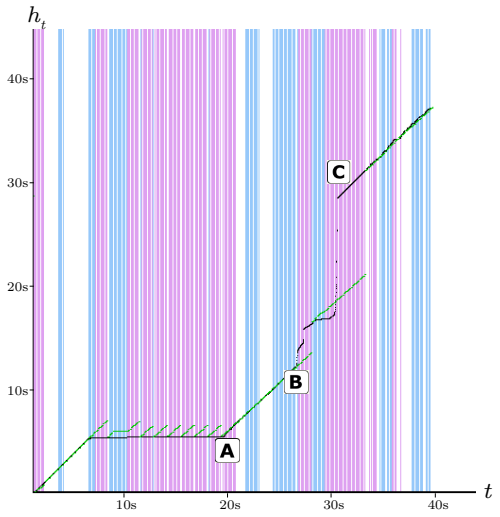

Figure 6: Example execution of the scenario in Figure 5. Notations in Figure 4 apply. See text for interpretation.

To model the space for improvisation, we found that filling the space in the reference audio with musical silence (actually the natural background noise of the instrument) gives the desired result. This way, the contrast between (target) activity and (reference) silence makes it instantly obvious when the improvisation begins: the online DTW starts evolving unpredictably, δ_t^{BACK} explodes and the beat tracker takes control.

The realignment in the final measures (as seen in the vertical jump in Figure 6) depends on one important condition: that the musician pause between the end of the improvisation and the start of the scored coda. This allows the DTW process to match one transition to the other. We found for our particular example [6] a pause of 1.1s to be sufficient, for an online DTW window of 1.49s [7].

6. CONCLUSIONS

We have presented an online audio-to-audio following and accompaniment system that combines a tempo model based on the alignment path produced by an online DTW process with a tempo model tied to beat tracking, without reference to any symbolic score data. The two models constantly inform each other, producing a synergistic relationship.

This framework is geared to a specific scenario, where a musician creates a reference audio track by playing along to a fixed backing track. This accompaniment track is then warped in real time to match a live target. Such a scenario applies more to popular beat-based music genres than the classical concerto or solo performances that followers such as [2,18] are designed around. Thus, while the added rigid-

[6] All test and demo files for this paper are available at `https://github.com/RVirmoors/RVdtw-/tree/master/_smc`

[7] $c = 128$ frames with a hop of 512 samples, at 44.1kHz sample rate.

ity serves beat-centred material well, it is less appropriate for strong rubato, where pure alignment-based models perform better. The user can address this by raising the ϵ^{B} and ϵ^{T} thresholds, causing the beat tracker to engage less easily.

We anticipate several avenues for future work. Firstly, an evaluation framework based on a cross-genre database of backing tracks and isolated performances is needed in order to ensure measurable progress. Our proposed embedding of beat tracking into the alignment process is certainly not the only possible method, and so far we have relied on piecewise experimentation to move forward.

Secondly, we are looking to develop new tempo models that make integral use of both performance alignment and beat tracking. The adaptation of the PID model introduced in section 4 is a start, but we might conceive models from the ground up with this configuration in mind.

Thirdly, we intend to further increase system robustness through parallel observations. Among the research directions worth considering are multi-agent following [19], asynchrony compensation [20] to capture the timing nuances of several musical facets, and parallel trackers for two or more musicians jointly driving the accompaniment.

Finally, we plan to move beyond warped backing tracks, to produce more dynamic accompaniments where the timing information extracted from live target performance(s) informs temporal deviations within the accompaniment's individual components.

7. REFERENCES

[1] S. Dixon, "Live tracking of musical performances using on-line time warping," in *International Conference on Digital Audio Effects (DAFx)*, 2005, pp. 92–97.

[2] A. Arzt and G. Widmer, "Simple tempo models for real-time music tracking," in *Sound and Music Computing conference (SMC)*, 2010.

[3] G. Burloiu, "An online audio alignment tool for live musical performance," in *Electronics and Telecommunications (ISETC)*, Nov 2014.

[4] A. Robertson and M. Plumbley, "B-keeper: A beat-tracker for live performance," in *International Conference on New interfaces for musical expression (NIME)*. ACM, 2007, pp. 234–237.

[5] A. M. Stark, M. E. Davies, and M. D. Plumbley, "Real-time beat-synchronous analysis of musical audio," in *International Conference on Digital Audio Effects (DAFx)*, 2009, pp. 299–304.

[6] C. Fremerey, M. Müller, and M. Clausen, "Handling repeats and jumps in score-performance synchronization." in *International Society for Music Information Retrieval conference (ISMIR)*, 2010, pp. 243–248.

[7] A. Arzt, G. Widmer, and S. Dixon, "Automatic page turning for musicians via real-time machine listening." in *European Conference on Artificial Intelligence (ECAI)*, 2008, pp. 241–245.

[8] A. Cont, "On the creative use of score following and its impact on research," in *Sound and Music Computing conference (SMC)*, Jul. 2011.

[9] M. Muller and D. Appelt, "Path-constrained partial music synchronization," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2008, pp. 65–68.

[10] M. Grachten, M. Gasser, A. Arzt, and G. Widmer, "Automatic alignment of music performances with structural differences," in *International Society for Music Information Retrieval conference (ISMIR)*, November 2013.

[11] R. Macrae and S. Dixon, "Accurate real-time windowed time warping," in *International Society for Music Information Retrieval conference (ISMIR)*, 2010, pp. 423–428.

[12] N. Montecchio and A. Cont, "A unified approach to real time audio-to-score and audio-to-audio alignment using sequential montecarlo inference techniques," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2011, pp. 193–196.

[13] B. Xiong and O. Izmirli, "Audio-to-audio alignment using particle filters to handle small and large scale performance discrepancies," in *International Computer Music Conference (ICMC)*, 2012.

[14] A. M. Stark and M. D. Plumbley, "Performance following: Real-time prediction of musical sequences without a score," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 190–199, 2012.

[15] P. Toiviainen, "Real-time recognition of improvisations with adaptive oscillators and a recursive bayesian classifier," *Journal of New Music Research*, vol. 30, no. 2, pp. 137–147, 2001.

[16] A. M. Stark and M. D. Plumbley, "Real-time chord recognition for live performance," in *International Computer Music Conference (ICMC)*, 2009.

[17] G. F. Franklin, J. D. Powell, and M. L. Workman, *Digital control of dynamic systems*. Menlo Park: Addison-Wesley, 1998, vol. 3.

[18] C. Raphael, "Music plus one and machine learning," in *International Conference on Machine Learning (ICML)*, 2010.

[19] A. Arzt and G. Widmer, "Real-time music tracking using multiple performances as a reference," in *International Society for Music Information Retrieval conference (ISMIR)*, 2015.

[20] S. Wang, S. Ewert, and S. Dixon, "Compensating for asynchronies between musical voices in score-performance alignment," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015, pp. 589–593.