BEAT-ALIGNING GUITAR LOOPER

Daniel Rudrich

Institute of Electronic Music and Acoustics,
University of Music and Performing Arts
Graz, Austria
rudrich@iem.at

Alois Sontacchi

Institute of Electronic Music and Acoustics,
University of Music and Performing Arts
Graz, Austria
sontacchi@iem.at

ABSTRACT

Loopers become more and more popular due to their growing features and capabilities, not only in live performances but also as a rehearsal tool. These effect units record a phrase and play it back in a loop. The start and stop positions of the recording are typically the player's start and stop taps on a foot switch. However, if these cues are not entered precisely in time, an annoying, audible gap may occur between the repetitions of the phrase. We propose an algorithm that analyzes the recorded phrase and aligns start and stop positions in order to remove audible gaps. Efficiency, accuracy and robustness are achieved by including the phase information of the onset detection function's STFT within the beat estimation process. Moreover, the proposed algorithm satisfies the response time required for the live application of beat alignment. We show that robustness is achieved for phrases of sparse rhythmic content for which there is still sufficient information to derive underlying beats.

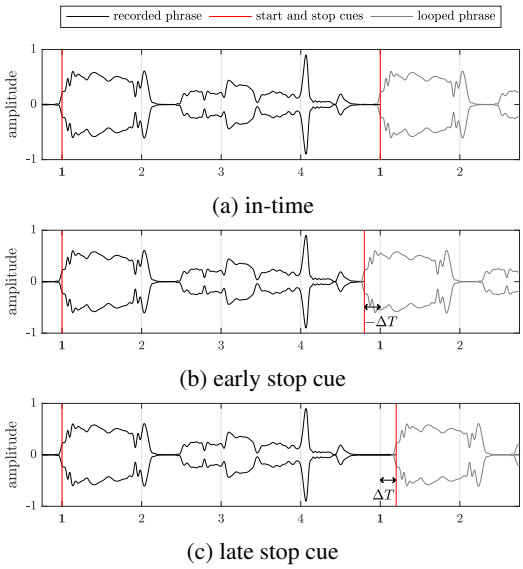
1. INTRODUCTION

Music can never have enough of saying over again what has already been said, not once or twice, but dozens of times; hardly does a section, which consists largely of repetition, come to an end, before the whole story is happily told all over again.

— Victor Zuckerkandl [1]

Repetition is an essential characteristic of music on different time scales. The compositional technique *canon* is solely based on a repeating structure. It elaborates by combining several layers of musical phrases. The technique of looping proceeds the same principles and establishes new perspectives and approaches to create and combine musical ideas. Since the invention of digital recording, the looping approach has been technically available.[1] However, especially for beginners it is often difficult to handle the timing precisely enough when controlling the looper.

Fig. 1a visualizes a looped phrase being repeated seamlessly as start and stop cues are in-time. The musical beats are generally unknown by the system and are depicted for visualization of the problem only. When hitting the stop button too early also the repetition starts too early, leading to a gap of length ΔT between the actual cue and the intended cue (Fig. 1b). Alternatively, a gap also occurs when the stop cue comes too late (Fig. 1c). Both cases also happen if the timing of the start cue is off, or a combination of both. Nevertheless, when both cues have the same offset, the loop gets repeated without any pauses or skips.



(a) in-time

(b) early stop cue

(c) late stop cue

Figure 1: Examples of a one-bar-phrase getting looped with different stop-cue timings. The abscissa holds the musical beats, the black line represents the envelope of the recorded signal, red lines indicate the start and stop cues and the grey line represents the repeated phrase setting in directly after the stop cue.

These artifacts can be audible and annoying, especially because of their continuous repetition. Some loopers solve this by quantizing the start and stop cues to a predefined click track. Here, the main objective was to develop a looper that estimates the underlying beats from the recorded phrase. In the proposed approach, the derivation of beats and tempo values is done in a final stage after gauging all possible candidates, in order to increase robustness and circumvent the general drawback of bottom-up approaches. As the recorded beat jitters with the accuracy and groove of the player, and the distinctness of transients depends on the style of play, this robustness is essential for a successful beat-alignment.

From the recorded musical phrase, we calculate a sparse energy distribution over time and compare it to given *tatum*[2] grids. In the following, the term *tatum* is used as a context-free measure of tempo, as it does not require a musical reference value as the musical measure *beats per minute* (bpm) does. The tatum τ embodies the time interval of the underlying grid and is given in seconds.

[1] With the analog predecessor of digital loop pedals, the *Time Lag Accumulator* [2], it was almost impossible to set the loop length while playing. First the use of digital memory instead of tape loops made it possible leading to a substantially different sound and use of live looping. [3]

[2] Bilmes coined this term in his thesis *Timing is of the Essence* [4], which describes the lowest metrical level in a musical piece as a fine underlying grid to which every note and rest can be assigned to.

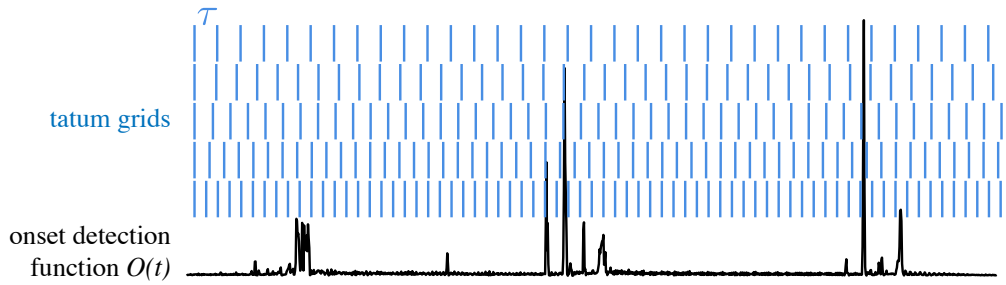


Figure 2: Qualitative illustration of the basic approach: Different tatum grids get placed over the onset detection function $O(t)$. The degree of matching provides information about the presence of different tatums in the signal.

The next section describes the structure of the proposed algorithm with its individual components. The subsequent sections then give a detailed description of these components followed by a performance evaluation of the algorithm including real time capability, perceptual evaluation, and performance examples.

2. OUTLINE OF THE BEAT-ALIGNMENT ALGORITHM

The beat-aligning looper should align the loop's start and stop cues so that no pauses or skips occur when the loop is repeated. The basic idea is to find the tatum of a recorded signal and align the start and stop cues to the corresponding beats. Tatum estimation is done by calculating an onset-detection-function (ODF) to which different tatum-grids are correlated to pick the one that fits best (see Fig. 2). Due to a possible variability of the tempo within the recorded phrase, the tatum estimation shouldn't be done for the entire sequence at once, but rather in blocks.

There is a variety of onset detectors. [5, 6] give a good overview of mainly energy based onset detectors. A more sophisticated approach based on recurrent neural networks is presented in [7]. However, as the audio signal from the guitar is expected to be clean without any effects (e.g. delay, reverb or noise) applied to it, we were able to choose a less complex onset detector. Therefore, the *spectral flux log filtered* approach proposed by Böck et al. [8] is used (Sec. 3). It is an advancement of the spectral flux algorithm by Masri [9]. The main differences are the processing in Mel frequency bands and the usage of the logarithm for the magnitudes to resolve loudness levels appropriately. The latter combined with calculating the difference between two consecutive frames (see Sec. 3.4) leads to an utilization of magnitude ratios instead of differences. This compensates for temporal variability (dynamic) of the audio signal as parts with high amplitudes do not get emphasized in comparison to parts with lower amplitudes.

Additionally, *Adaptive Whitening* suggested by Stowell and Plumbley [10] is used to compensate for variability in both time and frequency domain. It normalizes the magnitude of each STFT bin with regards to a preceding maximum value and compensates spectral differences as higher frequencies often have lower energy (spectral roll-off). Without the spectral whitening, lower frequency content tends to dominate over higher frequency content that contains important onset information, as the surprisingly good results of the simple HFC (high frequency content) ODF demonstrate [9].

As a basis of beat estimation, the tempo estimation approach proposed by Wu et al. [11] was employed (Sec. 4). In their proposal, an STFT of the ODF is calculated for a subset of frequencies, extending the tatum grids to complex exponential functions. This results in the tempogram, which holds a measure for the presence of a tatum in each ODF frame. Here, instead of picking the tatum with the highest magnitude for each block, a dynamic programming (DP) approach is used to find the optimum tempo path: a utility function is maximized with a gain for high magnitudes and a penalty for high tatum differences between two consecutive frames. This prevents paths with unlikely large variation in tempo. Different from existing DP approaches for beat estimation like [11, 12], we don't tie up beats to discrete onsets of the recorded signal as those onsets may be subject to the inaccuracy of human motor action. For this application it is advantageous to retrieve a beat grid without possible fluctuations. This is done by using phase information: the phase shift of the complex exponential function's periodicity (Fourier transformation) can be extracted by calculating the phase of the found optimum path. The phase directly measures the location of the beats as a beat occurs every time the phase strides a multiple of 2π. This approach is illustrated in Fig. 3. This leads to an increased precision similar to the instantaneous frequency tracking by [13]. As a result, without the necessity of a finely sampled set of tatum candidates efficiency is increased (see Sec. 5).

The following modifications were made regarding the enhancement of the tempogram and information extraction:

Modified tempogram The tempogram is enhanced by the information of how well the phase of a frame conforms with its expected value, calculated by the time difference between two consecutive frames and the tatum value.

Phase information In contrast to the tatum information of the optimum path, the phase of the path is used. It contains all the information needed to calculate the positions of the beats. Especially if the set of tatum grids is chosen to be coarse for low-cost implementation, the phase can be used to calculate the actual tatum more precisely.

Phase reliability A measure for the phase's reliability is used to spot frames, in which the phase information becomes meaningless. This happens when there are no significant onsets. For unreliable frames the beats will be discarded and replaced by an interpolating beat placement.

As a last step, the start and stop cues obtained by pressing a foot switch are aligned to the nearest estimated beat. There are no additional rules for the beat-alignment (e.g. loop length in beats has to be a multiple of 4) to keep enough responsibility for phrasing to the player.

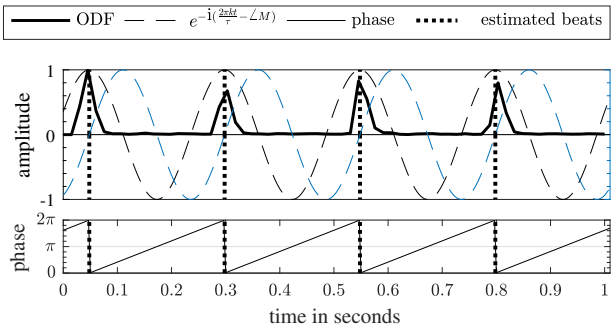


Figure 3: Visualization of the phase-based beat estimation. The cosine and sine components of the Fourier expansion for a tatum τ (dashed line) are shown in the graph on top. The Fourier transformation of the ODF yields a complex-valued tempogram value M, whose phase (bottom) carries the desired time-shift information of the corresponding tatum grid.

3. ODF: SPECTRAL FLUX LOG FILTERED

The *spectral flux log filtered* onset detection function consists of a block-wise transformation into the frequency domain followed by adaptive whitening. The signal then runs through a filter bank consisting of 50 Mel frequency bands. Last, after taking the logarithm, the temporal differences are summed up to obtain the ODF.

3.1. Short Time Fourier Transform (STFT)

Prior to the transformation into the frequency domain the signal $x(t)$ is blocked into N overlapping frames with a length of $K = 2048\,\text{samples}$ and the hopsize $h = 480\,\text{samples}$, resulting in an overlap of roughly $77\,\%$. With $f_s = 48\,000\,\text{Hz}$ subsequent frames are $10\,\text{ms}$ apart (resulting in an ODF sampling frequency of $f_{s,O} = 100\,\text{Hz}$). Each frame is windowed with a Hann window $w(t)$. The windowed signals $x_n(t)$ can be calculated with

$$x_n(t) = x(t + nh)w(t), \quad n \in [0, N-1],\ t \in [0, K-1]. \quad (1)$$

Each frame is transformed into the frequency domain using the discrete Fourier transform:

$$X(n,k) = \sum_{t=0}^{K-1} x_n(t) \cdot e^{-\mathrm{i}\frac{2\pi kt}{K}}, \quad k \in [0, K-1] \quad (2)$$

with n denoting the frame, k the frequency bin, and t the discrete-time index.

3.2. Adaptive Whitening

The *peak spectral profile*

$$P(n,k) = \begin{cases} \max\{|X(n,k)|,\ r\} & \text{for } n=0, \\ \max\{|X(n,k)|,\ r,\ \mu P(n-1,k)\} & \text{otherwise} \end{cases} \quad (3)$$

determines the spectral whitening in the division

$$X(n,k) \leftarrow \frac{X(n,k)}{P(n,k)}, \quad (4)$$

in which μ is the forgetting factor and r the floor parameter [10]. The choice $\mu = 0.997$ and $r = 0.6$ has proved suitable for the application.

3.3. Logarithmic-Magnitude Mel Spectrogram

The main difference to the regular spectral flux onset detection is the analysis in sub-bands. Consequently, the magnitude bins of the spectrogram $|X(n,k)|$ are gathered by filter windows $F(k,b)$ with $B = 50$ overlapping filters with center frequencies from $94\,\text{Hz}$ to $15\,375\,\text{Hz}$. Each window has the same width on the Mel-scale. The windows are not normalized to constant energy, which yields an emphasis on higher frequencies. The Mel spectrogram $X_{\text{filt}}(n,b)$ is given by:

$$X_{\text{filt}}(n,b) = \sum_{k=0}^{K-1} |X(n,k)| \cdot F(k,b), \quad b \in [0, B-1] \quad (5)$$

with b denoting the sub-band number. The logarithmic-magnitude Mel spectrogram is obtained by applying the logarithm to the Mel spectrogram

$$X_{\text{filt}}^{\log}(n,b) = log(\lambda \cdot X_{\text{filt}}(n,b) + 1) \quad (6)$$

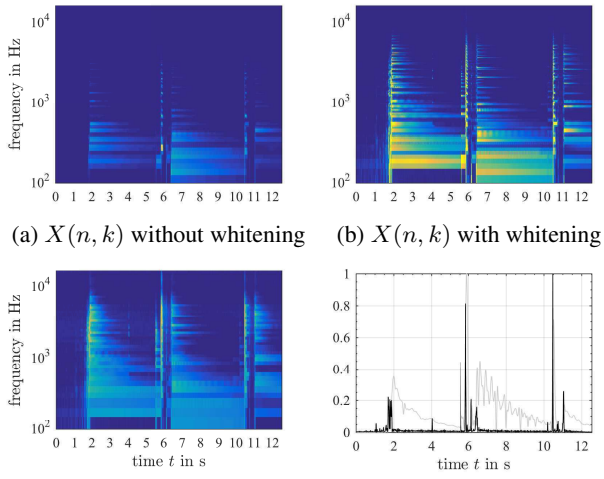
with the compression parameter λ. A value of $\lambda = 2$ yielded good results. The additive term $+1$ assures only positive values.

3.4. Difference

The final step to derive the onset detection function $O(n)$ is to calculate the difference between the frame n and its previous frame $n-1$, with a subsequent summation of all spectral bins. The half-wave rectifier function $H(x) = \frac{x+|x|}{2}$ ensures that only onsets are considered. Altogether, we obtain the following equation:

$$O(n) = \sum_{b=0}^{B-1} H(X_{\text{filt}}^{\log}(n,b) - X_{\text{filt}}^{\log}(n-1,b)). \quad (7)$$

Fig. 4 depicts the above-mentioned steps for the calculation of the onset detection function.



(a) $X(n,k)$ without whitening (b) $X(n,k)$ with whitening

(c) $X_{\text{filt}}^{\log}(n,b)$ (d) $O(n)$

Figure 4: Steps of calculation the *spectral flux log filtered* ODF: a) spectrogram b) adaptive whitening c) log-magnitude Mel spectrogram d) final ODF (black) and the audio signal (gray). The data of all four figures is normalized to a maximum value of 1.

4. BEAT ESTIMATION AND START/STOP ALIGNMENT

Out of the derived onset detection function the tempogram is calculated. A dynamic programming approach computes the most likely tatum path within the tempogram. Afterwards, beats are estimated by extracting the optimum path's phase information and a possible subsequent interpolation of areas with non-reliable phase information.

4.1. Tempogram

As described, the tempogram $M(j,\tau)$ is obtained by an STFT of $O(n)$, evaluated for specific frequencies corresponding to a set of tatums. This can be expressed as:

$$M(j,\tau) = \sum_{n=0}^{L-1} O(n+j,\tau) w(n) e^{-\mathrm{i}\frac{2\pi n}{\tau f_{s,O}}}, \quad j \in [0, J-1] \quad (8)$$

with j being the ODF frame index, L the ODF frame length, $w(n)$ the Hann window function, $f_{s,O}$ denoting the ODF sampling frequency (here 100 Hz) and $\tau \in \mathrm{T}$ denoting the tatum out of a set T with different tatum values between 60 ms and 430 ms. An ODF frame length of $L = 150$ yielded good results, meaning that one tempogram value represents a 1.5 s frame and the beginning of one frame is $\frac{1}{f_{s,O}} = 10$ ms apart from the beginning of its predecessor. The total number of time-steps j can be expressed as $J = N - L + 1$.

To emphasize on phase continuity the phase difference between two consecutive frames $d\Phi(j,\tau)$ is calculated and compared to the expected value $\hat{d\phi}(\tau)$. The difference of both results into the phase deviation matrix:

$$\Delta\Phi(j,\tau) = d\Phi(j,\tau) - \hat{d\phi}(\tau) \quad (9)$$

with

$$d\Phi(j,\tau) = \angle M(j,\tau) - \angle M(j-1,\tau) \quad \text{and} \quad (10)$$

$$\hat{d\phi}(\tau) = \frac{2\pi}{\tau f_{s,O}}. \quad (11)$$

$$\Delta\Phi_{\text{mapped}} = ((\frac{\Delta\Phi}{\pi} + 1) \bmod 2) - 1 \quad (12)$$

maps the values to a range of $[-1, 1]$, with a value of 0 indicating a perfect phase deviation of 0. The modified tempogram $M'(j,\tau)$ gets calculated as follows:

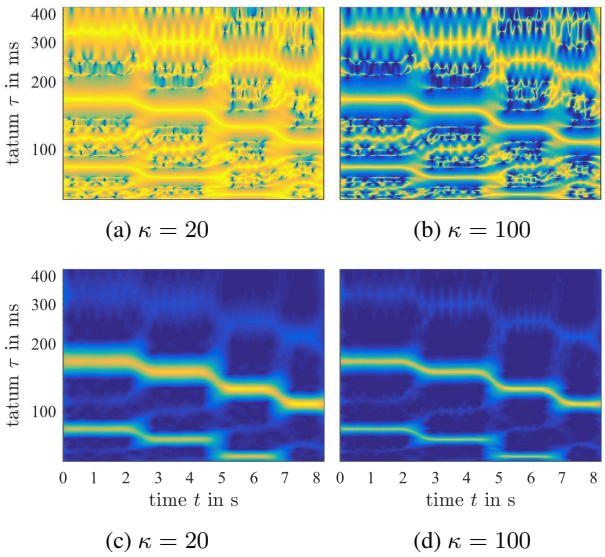
$$M'(j,\tau) = M(j,\tau) \cdot (1 - |\Delta\Phi_{\text{mapped}}(j,\tau)|)^{\kappa} \quad (13)$$

whereas κ denotes the degree of factoring in the phase conformance. A value of $\kappa = 100$ was ascertained experimentally and suited well for this application. Fig. 5 shows the sharpening of the tempogram due to this modification. The used signal is a hi-hat sequence with leaps in tempo. With κ the amount of sharpening can be adjusted.[3]

As a last step, $M'(j,\tau)$ gets normalized to a maximum absolute value of 1:

$$M'(j,\tau) \leftarrow \frac{M'(j,\tau)}{\max_{j,\tau} |M'(j,\tau)|}. \quad (14)$$

[3] In general, for a coarsely sampled set of tatums a lower κ value should be chosen. Otherwise, the phase nonconformance as a consequence of a non-sampled tatum would lead to a falsification of the tempogram.



(a) $\kappa = 20$ (b) $\kappa = 100$

(c) $\kappa = 20$ (d) $\kappa = 100$

Figure 5: Effect of different κ values on the modified tempogram. The range of the depicted values is between 0 and 1, with dark blue representing 0, yellow representing a value of 1. a) and b) show $(1 - |\Delta\Phi_{\text{mapped}}(j,\tau)|)^{\kappa}$, c) and d) show the modified tempograms $M'(j,\tau)$.

4.2. Optimum Tatum Path

The optimum tatum path can be extracted out of the modified tempogram by maximizing the utility function $U(\boldsymbol{\tau},\theta)$. This function is designed in order that high absolute tempogram values $|M'(j,\tau)|$ (tatum conformance) are advantaged and high leaps in tempo/tatum will result in a penalty (second term in equation (15)). The goal is to find a series of tatum values $\boldsymbol{\tau} = [\tau_0, ...\tau_j, ...\tau_{J-1}]$, with τ_j the tatum value for ODF frame j, which maximizes the utility function

$$U(\boldsymbol{\tau},\theta) = \sum_{j=0}^{J-1} |M'(j,\tau_j)| - \theta \sum_{j=1}^{J-1} |\frac{1}{\tau_{j-1}} - \frac{1}{\tau_j}|, \quad (15)$$

with θ denoting the penalty factor for a tatum difference between two consecutive frames. With $\theta = 0$ the maximization could be replaced by picking the tatum with the highest absolute tempogram value. The higher θ the smoother the tempo path due to a higher penalty for tempo changes. A value of $\theta = 20$ suited well for this application.

The search for the maximum of the utility function can be done efficiently with dynamic programming. Therefore, the maximum can be written as:

$$\max_{\boldsymbol{\tau}} U(\boldsymbol{\tau},\theta) = \max_{\tau} D(J-1,\tau) \quad (16)$$

with the recurrent equation

$$D(j,\tau) = \begin{cases} |M'(0,\tau)| & \text{if } j = 0, \\ |M'(j,\tau)| + \max_{\tau_{j-1}} \left(D(j-1,\tau) - \theta \left| \frac{1}{\tau_{j-1}} - \frac{1}{\tau_j} \right| \right) & \text{otherwise} \end{cases} \quad (17)$$

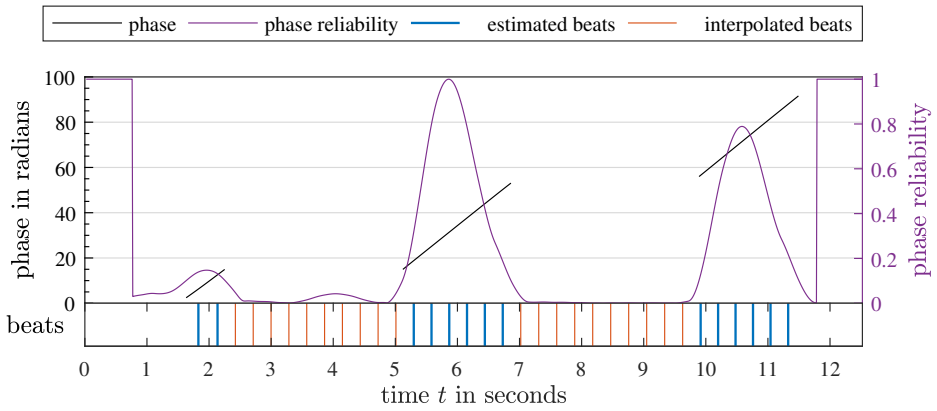


Figure 6: Example of filling gaps with low phase reliability. The black line represents the phase with gaps as the phase reliability (purple) drops below 0.1. The estimated beats (blue) are gathered with the phase information, the interpolated beats (orange) are filled in by interpolation. The impression of the phase keeping its value right before and after a gap is an artifact of phase-unwrapping.

Basically, after initialization the first frame $j = 0$, for every tatum τ_j of the frame j the algorithm looks for that tatum τ_{j-1} of the previous frame which yields the most rewarding transition $\tau_{j-1} \rightarrow \tau_j$. With memorizing $\tau_{j-1,\max}$ for every $D(j,\tau)$, the optimum path can be found by backtracking $\tau_{j-1,\max}$ starting with the tatum $\arg\max_{\tau} D(J-1,\tau)$ of the last frame.

The optimum path extracted for the previous shown tempogram is depicted in Fig. 7. The path (red line) follows the leaps in tempo.

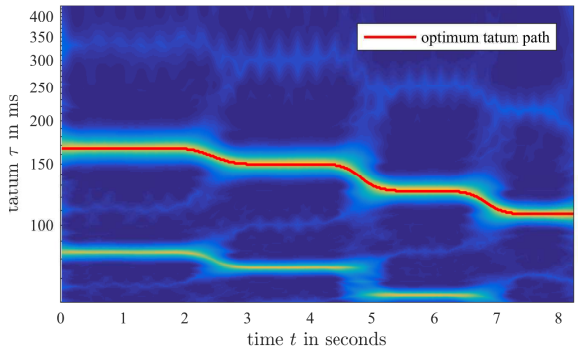


Figure 7: Resulting optimum tatum path for a hi-hat signal with leaps in tempo.

4.3. Beat Placement and Start/Stop Alignment

As described above, the beat placement uses the phase $\phi(n)$ of the optimum tatum path $\boldsymbol{\tau} = [\tau_0, ...\tau_j, ...\tau_{J-1}]$. The phase can be obtained from the tempogram $M(j,\tau_j)$ by calculating the angle of the complex values. Also the modified tempogram $M'(j,\tau)$ can be used here, as it holds the same phase information. To find a phase value for every time step n of the ODF, the tempogram time steps j have to be mapped to n:

$$j \rightarrow n = j + \frac{L}{2}. \tag{18}$$

The offset of $\frac{L}{2}$ issues from the phase information being calculated for a frame with length L (see equation (8)). So the center of each frame was chosen for the mapping. Nevertheless, the phase information is valid for the beginning of each frame, therefore, the phase itself also has to be adjusted to the frame center by the expected amount $\frac{L}{2}\hat{d\phi}(\tau)$. So the extraction of the phase can be formulated as follows:

$$\phi(n) = \phi(j+\frac{L}{2}) = \angle M(j,\tau_j) + \frac{L}{2}\hat{d\phi}(\tau_j), \text{for } j \in [0, J-1]. \tag{19}$$

The remaining phase information for $n < \frac{L}{2}$ and $n \geq J + \frac{L}{2} = N - \frac{L}{2} + 1$ has to be derived by extrapolation. The phase then can be used to place beats to the ODF. A beat occurs every time the phase strides a multiple of 2π. This search is equal to the calculation of positive[4] zero crossings of $\sin(\phi(n))$.

With the phase difference between two consecutive frames $d\phi(n)$ yielding the current phase step, the current tatum can be calculated analogous to equation (11):

$$\tau(n) = \frac{2\pi}{d\phi(n) f_{s,O}}. \tag{20}$$

The $\tau(n)$ values are not bound to those of the tatum set T and, as a consequence, are sampled with a higher precision.[5] Averaging these values results into the mean tatum $\bar{\tau}$, which can be used for the phase extrapolation and interpolation of beat gaps (described hereafter).

Additionally to the angle of the tempogram, the magnitude is used as a measure of phase reliability. If the magnitude is lower than a predefined threshold, the phase information is considered meaningless. Low tempogram magnitudes can occur in frames with only few or no onsets. In that case, the phase information gets discarded and the resulting gaps get filled with equally spaced beats corresponding to the mean tatum. An example of interpolated beats is shown in Fig. 6.

The last step is to align the start and stop cues to the estimated beat positions. This is easily done by shifting each cue the closest beat.

[4] transition from negative to positive values

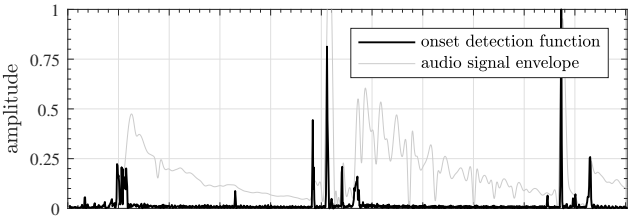
[5] see the example in Section 5.2 for demonstration

5. PERFORMANCE EVALUATION

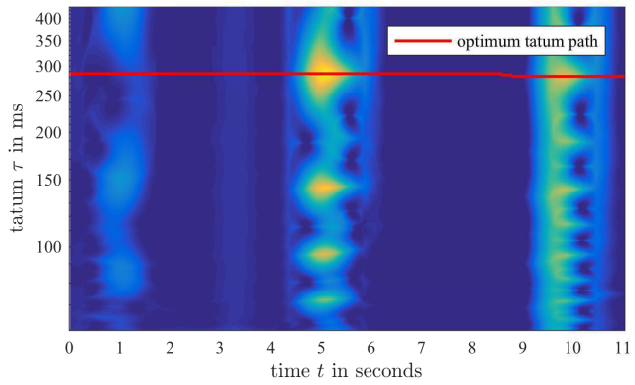
This sections shows exemplarily how the algorithm reacts to different signals. Also the performance concerning the real time capability is investigated. Additionally, the perceptibility of the remaining gaps is treated here.

5.1. Beat Interpolation of Low Magnitude Gaps

Fig. 8 shows ODF and tempogram of a recorded phrase with sustaining chords. Hence, only a few onsets exist at around $t = 5\,\mathrm{s}$ and $10\,\mathrm{s}$ as both ODF and tempogram reveal. Even the start of the phrase does not hold any useful tatum information. Nevertheless, the algorithm found the optimum path, which fits the audio signal best. Due to the vanishing magnitude of the tempogram between $t = 2\,\mathrm{s}$ and $4\,\mathrm{s}$ and between $t = 6\,\mathrm{s}$ and $9\,\mathrm{s}$, the phase reliability measure is not high enough to place reliable beats. As a consequence, two gaps emerge after discarding unreliable beats, which are filled with interpolated beats, as shown before in Fig. 6.



(a) Onset detection function



(b) Tempogram

Figure 8: Onset detection function (a) and tempogram (b) of a recorded phrase with only a few onsets.

5.2. High Tatum Precision despite Coarse Sampling

To demonstrate the higher tatum precision than the sampled tatum grids due to factoring in phase information, a semiquaver hi-hat signal at tempo $93\,\mathrm{bpm}$ is used. The resulting tatum is $\tau_0 = 161.29\,\mathrm{ms}$. The used tatum grids of the tempogram stage are sampled with a coarse resolution: the examined tatum values next to τ_0 are $155.01\,\mathrm{ms}$ and $165.89\,\mathrm{ms}$. The corresponding tempogram is depicted in Fig. 9.

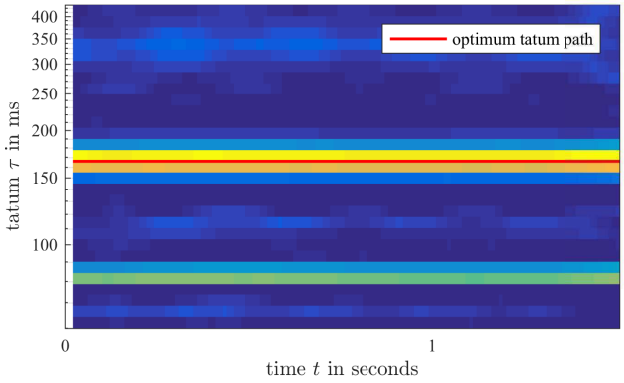


Figure 9: Tempogram with low tatum sampling.

The optimum path search yielded a constant tatum of $\tau = 165.89\,\mathrm{ms}$, as being closest to τ_0. However, the phase information yielded an average tatum of $\bar{\tau} = 161.40\,\mathrm{ms}$, which is remarkably closer, with a difference of just $0.11\,\mathrm{ms}$ instead of $4.6\,\mathrm{ms}$.

5.3. Greatest Common Divisor (GCD) of Existing Tatums

The algorithm tries to find that tatum which fits best to all occurring tatums. This mechanism is demonstrated with a signal consisting of alternating blocks of quaver notes and quaver triplets. At tempo $100\,\mathrm{bpm}$, the time difference between two successive quaver notes is $\tau_1 = 300\,\mathrm{ms}$ and for quaver triplets $\tau_2 = 200\,\mathrm{ms}$, respectively. As expected, these tatum values also occur in the tempogram with a high presence measure (see Fig. 10). However, the optimum tatum path was found for a tatum τ_3, which does not occur explicitly, but is implied by the two tatums τ_1 and τ_2 by being the greatest common divisor $\tau_3 = \gcd(\tau_1, \tau_2) = 100\,\mathrm{ms}$. All occurring events can be assigned to beats placed with that tatum.

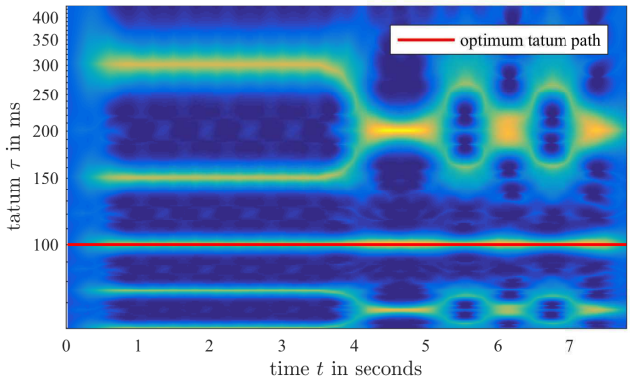


Figure 10: Effects of complex rhythms on the algorithm's performance. Optimum tatum path (red) does not follow the explicitly occurring tatums (150 ms and 300 ms), but their greatest common divisor (100 ms).

Table 1: Results of time profiling of two audio files with different durations and different number of tatums n_τ. The algorithm was implemented in Matlab and executed on a 2.3 GHz Intel Core i7 processor. ODF: onset detection function; TG: tempogram calculation; path: optimum path search; BE: beat estimation

duration (in s)	n_τ	computation time (in % of duration)				
		ODF	TG	path	BE	overall
9.3	60	1.76	0.08	2.51	0.10	4.45
9.3	120	1.83	0.16	5.93	0.14	8.06
9.3	240	1.76	0.26	16.48	0.14	18.64
22.2	60	1.02	0.09	2.85	0.07	4.03
22.2	120	1.02	0.17	6.80	0.06	8.05
22.2	240	1.00	0.23	17.02	0.07	18.32

5.4. Real Time Capability

The above described algorithm only embodies a useful tool for live looping if it is real time capable. At a first glance this means, that all the calculations have to be completed when the musician stops the recording and expects the phrase being repeated seamlessly. This actually is not possible as the back tracking process of the optimum tatum path estimation cannot start until the last frame, in which the stop cue occurs, was processed.

Fortunately, by accepting a possibly perceivable gap at the first iteration, real time capability can be easily achieved: now the algorithm has to be completed within the first unaltered iteration of the phrase, which usually has a duration between 3 s to 10 s.

Table 1 shows the needed time for computations in percentage of the duration of two different audio signals. The data shows that the algorithm's performance in regards of computation time depends strongly on the number of evaluated tatums n_τ as the computing effort of the tempogram increases linearly and that of the optimum path search quadratically with n_τ. The rather constant relative overall time shows a linear relationship with the signal's duration. Note that these results are gathered by a single execution of the algorithm for each combination of audio signal and number of tatums and, therefore, may be affected by different CPU loads. Also these results were gathered in an offline version of the algorithm. By computing the ODF and parts of the tempogram and the optimum path search during the recording phase, these values can even be lowered. It can therefore be concluded that the algorithm is real time capable and only needs a fraction of the recorded signal's duration for computation.

5.5. Perceptibility of Remaining Gaps

Hibi [14] and Rudrich [15] investigated the perception of rhythmic perturbances. Latter found a dependency on musical training onto the audibility of rhythm perturbances in a monotonic, isochronous sequence. The found threshold of subjects with musical background was remarkably lower with 5.4 % of the inter onset interval than that of subjects without musical training (9.0 %). We used these thresholds to validate the algorithm's performance regarding the reduction of perceivable gaps induced due bad timing of the start and stop cues. Actually, for a seamless performance these cues do not have to sit perfectly on the same musical measure. It is sufficient when both cues have the same offset to an arbitrary measure. As a consequence, only the difference of both offsets is used as a measure for the gap.

Table 2: Results of validation of the algorithm. Values are given in ms and show the resulting gaps introduced by the algorithm. Marked (*) $L_{16\text{th}}$ levels indicate the algorithm finding the tatum for quaver notes (300 ms), otherwise the tatum of semiquavers (150 ms) is found.

σ 1/f noise STD in samples / ms	$L_{16\text{th}}$ in dB				
	0	−10	−20	−30*	−∞*
0 / 0	0.23	0.29	0.44	1.92	1.88
100 / 2.08	0.17	0.15	0.42	0.10	1.54
200 / 4.17	−0.56	1.02	2.96	−0.75	1.23
300 / 6.25	−3.40	−1.73	−2.06	2.65	0.23
400 / 8.33	−6.31	1.21	0.85	−1.56	−4.35
500 / 10.42	5.02	4.25	1.00	0.73	−2.44

To validate the algorithm in an analytical and ecological valid way, a synthetic drum beat was used due to its advantages:

- easy reproducibility
- control of parameters (rhythm fluctuation and presence of the tatum)
- start and stop cues are easy to determine
- musical context.

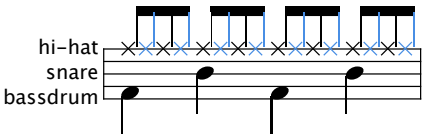


Figure 11: Score of the synthetic drum beat used for evaluation.

The drum beat's score is depicted in Fig. 11. The *tatum presence* was adjusted with the $L_{16\text{th}}$ level, representing the level of every other hi-hat note (highlighted blue). *Rhythm Fluctuation* is realized with an inserted 1/f noise jitter with standard derivation values between 0 ms and 10 ms.

For different combinations of the above mentioned parameters, the algorithm processes audio files with pre-defined and perfectly timed start and stop cues. The alteration of these cues leads to a gap introduced by the algorithm, which serves as a measure for performance as it shows how well the beats can be aligned at best.

The validation's results are depicted in Table 2. The data shows the gap introduced by the algorithm. The smaller the values, the better the algorithm's performance. Negative values indicate a negative gap, meaning the interval was shortened (otherwise lengthened). For $L_{16\text{th}} = -30$ dB and $L_{16\text{th}} = -\infty$ dB the tatum found was 300 ms corresponding to the quaver notes in the drumbeat. As can be seen for no introduced jitter ($\sigma = 0$) the algorithm creates a gap, especially when the tatum doubles. Nevertheless, when comparing to the results of Hibi [14] and Rudrich [15] all gaps are smaller then the found thresholds of perception.

6. CONCLUSIONS AND OUTLOOK

We proposed a real-time capable beat-aligning guitar looper. It is mainly characterized by its ability to support the temporally accurate handling of the looper without any further prior information and additional constrains. Start and stop cue alignment is done automatically during the first repetition of the recorded phrase. Independent of tempo, this adjustment guarantees that the resulting gap stays within rhythmic inaudibility. For traceability, this was evaluated within an ecologically valid setup.

It is obvious that the computation time of the described algorithm is dominated by the optimum path search algorithm. Informal tests showed, that a divide and conquer approach reduces the complexity and computation time. In this approach the optimum path search combines the output of separately analyzed tatum octave sets, instead of the entire tatum set.

As the beat-aligning looper retrieves all beats of the recorded phrase, it is a promising basis for further development of an automatic accompaniment (e.g. rhythm section) for practice or live-performance purposes.

A detailed view on the conducted listening tests and the evaluation of the algorithm can be found in [15].

7. REFERENCES

[1] V. Zuckerkandl, "Sound and Symbol," Princeton University Press, 1969.

[2] T. Baumgärtel, *Schleifen: zur Geschichte und Ästhetik des Loops*, Kulturverlag Kadmos, Berlin, 2015.

[3] M. Grob, "Live Looping - Growth due to limitations," http://www.livelooping.org/history_concepts/theory/growth-along-the-limitations-of-the-tools/, 2009, Last accessed on Jan 11, 2017.

[4] J. Bilmes, *Timing is of the Essence: Perceptual and Computational Techniques for Representing, Learning, and Reproducing Expressive Timing in Percussive Rhythm*, Ph.D. thesis, Massachusetts Institute of Technology, Program in Media Arts & Sciences, 1993.

[5] S. Dixon, "Onset detection revisited," in *Proceedings of the 9th International Conference on Digital Audio Effects (DAFx-06)*, Montreal, Canada, September 18-20, 2006, pp. 133–137.

[6] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. B. Sandler, "A tutorial on onset detection in music signals," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 5, pp. 1035–1047, 2005.

[7] S. Böck, A. Arzt, F. Krebs, and M. Schedl, "Online real-time onset detection with recurrent neural networks," in *Proceedings of the 15th International Conference on Digital Audio Effects (DAFx)*, York, UK, September 17-21, 2012.

[8] S. Böck, F. Krebs, and M. Schedl, "Evaluating the Online Capabilities of Onset Detection Methods," in *Proceedings of the 13th International Society for Music Information Retrieval Conference, ISMIR 2012*, Mosteiro S.Bento Da Vitória, Porto, Portugal, October 8-12, 2012, pp. 49–54.

[9] P. Masri, *Computer modelling of sound for transformation and synthesis of musical signals*, Ph.D. thesis, University of Bristol, 1996.

[10] D. Stowell and M. Plumbley, "Adaptive whitening for improved real-time audio onset detection," in *Proceedings of the International Computer Music Conference (ICMC'07)*, Copenhagen, Denmark, August 27-31, 2007, pp. 312–319.

[11] F. H. F. Wu, T. C. Lee, J. S. R. Jang, K. K. Chang, C. H. Lu, and W. N. Wang, "A Two-Fold Dynamic Programming Approach to Beat Tracking for Audio Music with Time-Varying Tempo," in *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR*, Miami, Florida, USA, October 24-28, 2011, pp. 191–196.

[12] D. P. W. Ellis, "Beat Tracking by Dynamic Programming," *Journal of New Music Research*, vol. 36, no. 1, pp. 51–60, Mar. 2007.

[13] B. Boashash, "Estimating and interpreting the instantaneous frequency of a signal—Part I: Fundamentals," *Proceedings of the IEEE*, vol. 80, no. 4, pp. 520–538, Apr. 1992.

[14] S. Hibi, "Rhythm perception in repetitive sound sequence," *Journal of the Acoustical Society of Japan (E)*, vol. 4, no. 2, pp. 83–95, 1983.

[15] D. Rudrich, "Timing-improved Guitar Loop Pedal based on Beat Tracking," M.S. thesis, Institute of Electronic Music and Acoustics, University of Music and Performing Arts Graz, Jan. 2017.