A REAL-TIME APPROACH FOR ESTIMATING PULSE TRACKING PARAMETERS FOR BEAT-SYNCHRONOUS AUDIO EFFECTS

Peter Meier * *Simon Schwär* † *Meinard Müller* ‡

International Audio Laboratories Erlangen
Am Wolfsmantel 33, 91058 Erlangen
Germany
peter.meier@audiolabs-erlangen.de

ABSTRACT

Predominant Local Pulse (PLP) estimation, an established method for extracting beat positions and other periodic pulse information from audio signals, has recently been extended with an online variant tailored for real-time applications. In this paper, we introduce a novel approach to generating various real-time control signals from the original online PLP output. While the PLP activation function encodes both predominant pulse information and pulse stability, we propose several normalization procedures to discern local pulse oscillation from stability, utilizing the PLP activation envelope. Through this, we generate pulse-synchronous Low Frequency Oscillators (LFOs) and supplementary confidence-based control signals, enabling dynamic control over audio effect parameters in real-time. Additionally, our approach enables beat position prediction, providing a look-ahead capability, for example, to compensate for system latency. To showcase the effectiveness of our control signals, we introduce an audio plugin prototype designed for integration within a Digital Audio Workstation (DAW), facilitating real-time applications of beat-synchronous effects during live mixing and performances. Moreover, this plugin serves as an educational tool, providing insights into PLP principles and the tempo structure of analyzed music signals.

1. INTRODUCTION

Many audio effects and instruments, such as echo, flanger, tremolo, or synthesizers, are often strongly aligned with the rhythmic structure of the music. This is typically achieved by modulating effect parameters like amplitude, frequency, or phase in a beat-synchronized way with a Low Frequency Oscillator (LFO) [1]. LFOs are used to generate various waveforms such as square, sawtooth, or sinusoidal and typically operate at frequencies below 10 Hertz, so that they can enable rhythmic variations of parameters.

The challenge with LFO-modulated and beat-synchronous effects lies in the manual tuning of LFO frequency and phase to be in sync with the rhythmic structure of the music. In Digital Audio Workstations (DAWs), users often find themselves in need of manually adjusting parameters or drawing automation curves to achieve precise synchronization. This process can be time-consuming and requires careful attention to detail. Similarly, in live performance settings such as using a guitar effects pedal, musicians must manually enter the desired tempo through tapping, which can be cumbersome and prone to human error, while there is also no way to dynamically react to natural tempo variations. As a result, automating these processes can greatly enhance efficiency and accuracy in applying beat-related effects.

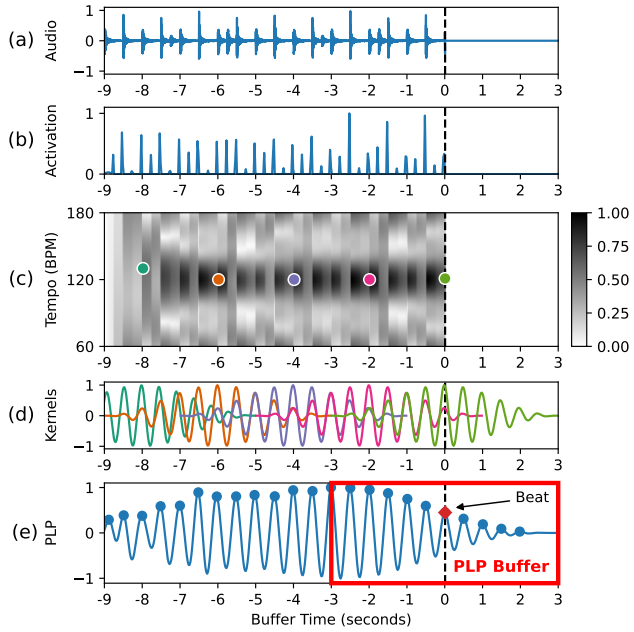


Figure 1: *The real-time PLP procedure is showcased utilizing a straightforward drum beat as follows: (a) Audio signal. (b) Activation function. (c) Tempogram. (d) Pulse kernels. (e) PLP function with buffer.*

One method of acquiring such beat-based control parameters is through beat tracking (or more generally *pulse* tracking), which is an essential task in the field of Music Information Retrieval (MIR). Beat trackers typically proceed in two steps to identify the temporal positions of beats within a music recording. They first compute an activation function to expose note onset information, followed by post-processing to determine beat positions [2]. In recent years, Deep Learning (DL) has driven significant advancements in beat tracking, with techniques like Temporal Convolutional Networks (TCN) [3], Transformer models [4], and SpecTNT-TCN [5]. These methods predominantly operate in an offline mode, requiring access to the complete music track for analysis.

* peter.meier@audiolabs-erlangen.de
† simon.schwaer@audiolabs-erlangen.de
‡ meinard.mueller@audiolabs-erlangen.de

With the increasing demand for real-time applications like interactive music systems [6] and live performance tools [7], there is a growing need for beat tracking algorithms capable of online operation. Online beat trackers have been utilized for beat-synchronous analysis in the past [8] and have recently attracted increased interest with contributions such as BeatNet [9], Novel-1D [10], and BEAST-1 [11]. However, when implementing online approaches for real-time applications or integrating them into larger interactive systems, previous online beat-tracking systems often lack explicit control over parameters such as the tempo range of the estimation or the latency of the system.

Drawing from the Predominant Local Pulse (PLP) concept [12], which was originally developed for offline applications, we introduced a real-time PLP tracking variant in our previous work [13]. This approach involves transforming the audio signal into an activation function, computing a Fourier tempogram to identify local periodic patterns, as well as selecting and overlap-adding windowed sinusoidal kernels that represent the local pulse structure, as outlined in Figure 1. The PLP buffer, which is updated with each new frame of audio analysis, serves as a central real-time component, reflecting both pulse oscillation and beat stability.

As the main contribution of this paper, we introduce a method to transform and normalize the real-time PLP output to derive multiple control signals. Given that PLP is based on oscillation, with sinusoidal kernels representing the local pulse structure of the audio input, the PLP output closely resembles an LFO signal. We introduce various normalization steps aimed at separating pulse oscillation from stability measures. Specifically, we develop two different confidence envelopes and two different oscillation signals: one for a beat-synchronized LFO signal with confidence amplitude, and another for a beat-synchronized LFO signal with constant amplitude, achieved by normalizing with the confidence envelope. We explore several case studies illustrating how the derived LFO and confidence parameters facilitate real-time applications of beat-synchronous effects in live performances and mixing environments. Through these case studies, we demonstrate how the derived control signals not only enable creative sound design but also provide valuable insights into the tempo structure of the audio recording.

The remainder of this article is organized as follows. In Section 2, we investigate the real-time PLP method and its central element, the PLP buffer. Section 3 presents the normalization techniques for disentangling beat structure and estimation confidence. In Section 4, we introduce four derived control signals for controlling DAW parameters, including low frequency oscillators and beat confidences. Section 5 demonstrates our method with various applications for the four control signal variants. Finally, in Section 6, we conclude our contribution. Additional materials and audio examples are available on a supplemental website[1].

2. PREDOMINANT LOCAL PULSE

In this section, we introduce the main idea of the PLP algorithm as first introduced in [12]. Specifically, we summarize the real-time PLP procedure, as initially described in [13], and discuss the main properties of the PLP buffer. This buffer serves as the centerpiece of the real-time PLP procedure, from which we aim to extract control signals.

[1] https://audiolabs-erlangen.de/resources/MIR/2024-DAFx-RealTimePLP

2.1. Real-Time PLP Procedure

This section summarizes the mathematical notation required for this paper. For a more detailed description, we refer to [12] and [13]. With Figure 1, we illustrate the basic idea of the real-time PLP procedure. First, the audio signal (Figure 1a) is transformed into an activation function $\Delta : \mathbb{Z} \to \mathbb{R}$, analyzing spectral changes over time positions $n \in \mathbb{Z}$. Note that the term *activation function* can refer to both novelty functions (such as spectral flux) and probabilistic machine learning models (such as a recurrent neural network). To detect local periodic patterns in Δ (Figure 1b), we calculate a Fourier tempogram $\mathcal{T}(n, \tau)$, which is a function over time positions n and tempo parameters τ, as depicted in Figure 1c (for details we refer to [12]). For this purpose, we use a window[2] function $\mathcal{W} : [-N : N] \to \mathbb{R}$ for $N \in \mathbb{N}$, which is normalized and centered around each time position n, yielding a total window size of $L = 2N + 1$. For any arbitrary but fixed time position $n \in \mathbb{Z}$, the window $\mathcal{W}$ specifies a neighborhood indexed by $m \in [n - N : n + N]$. For each $n \in \mathbb{Z}$, we select a windowed sinusoidal kernel $\kappa_n : [n - N : n + N] \to \mathbb{R}$, given by

$$\kappa_n(m) := \mathcal{W}(m - n) \cdot \cos\Big(2\pi\big((\tau_n/60) \cdot m - \varphi_n\big)\Big), \quad (1)$$

for $m \in [n - N : n + N]$, that best aligns with the periodic structure of the activation function Δ. The tempo parameter τ_n and the corresponding phase parameter φ_n maximize the tempogram $\mathcal{T}(n, \tau)$, as illustrated by colored dots in Figure 1c and can be obtained from the complex-valued Fourier representation underlying the tempogram [13]. In a real-time context, we consider n_0 as the current time position, where we only have access to the beat activation values $\Delta(n)$ for all time positions $n \leq n_0$. For the current time position n_0, we obtain a real-time PLP function $\Gamma_{n_0} : [-\infty : n_0 + N] \to \mathbb{R}$ with

$$\Gamma_{n_0}(n) := \frac{1}{C} \sum_{\ell=n-N}^{n_0} \kappa_\ell(n), \quad (2)$$

which is defined for all time positions $n \in [-\infty : n_0 + N]$ and has access to all kernels κ_ℓ for $\ell \in [n - N : n_0]$. The constant

$$C = \sum_{n=-N}^{N} \mathcal{W}(n) \quad (3)$$

ensures that the values of $\Gamma_{n_0}(n)$ lie within a range of $[-1 : 1]$. The PLP buffer, depicted in Figure 1e, displays only the section of $\Gamma_{n_0}(n)$ for $n \in [n_0 - N : n_0 + N]$, containing all the necessary information to compute the buffer for the subsequent time position.

2.2. Extracting Control Signals from PLP Buffer

The PLP buffer, as introduced in Section 2.1, serves as the central component of the real-time PLP procedure and is updated with the audio data for each new current time position n_0. In addition to encoding the local pulse structure as oscillations, it also exhibits varying amplitudes, reflecting the local tempo stability of the audio signal, see [12]. The reason for the varying amplitudes is that, depending on the local tempo structure and the predominant tempo kernels selected, different kinds of interference between kernels can occur, as illustrated in Figure 2.

[2] We use a Hann window for all the illustrations in this paper, as well as for our audio plugin prototype.

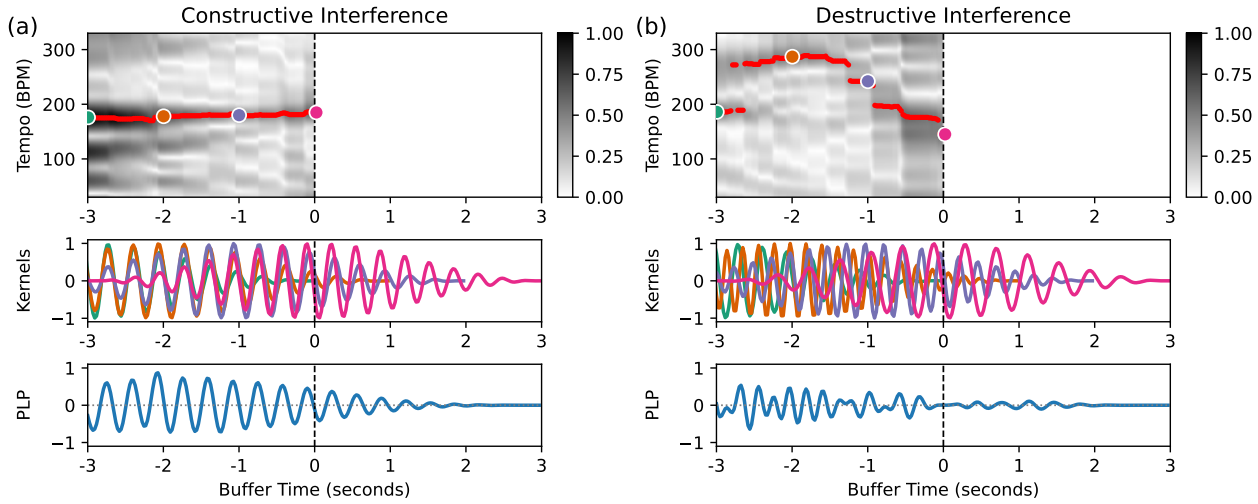


Figure 2: *The tempogram, kernels, and PLP buffer illustrate (a) constructive interference and (b) destructive interference.*

When the tempo situation is stable, as depicted in Figure 2a, the kernels κ_ℓ selected from the tempogram $\mathcal{T}$ have a similar frequency and overlap-add constructively, leading to a PLP function Γ_{n_0} with high amplitude values. Conversely, when the tempo situation is unstable, as depicted in Figure 2b, the neighboring kernels exhibit significant frequency variations and cancel each other out in the overlapping section, resulting in a lower overall amplitude of the PLP function. In this way, the PLP buffer contains not only pulse oscillation but also a beat stability measure in a single representation.

One inherent feature observed in the PLP buffer is that the values of its right half fade out to zero. This occurs because there are no kernels κ_ℓ available for overlap-adding at future time positions $n > n_0$, corresponding to diminishing confidence in pulse information the further one looks into the future. While this aspect is not crucial for beat detection, as it has been successfully employed in other real-time applications before [7, 6], normalizing the PLP buffer is especially necessary for extracting control signals to be independent from beat confidence.

In the following, we address two challenges for extracting control signals that arise from the varying amplitudes in the PLP buffer. First, to achieve a more consistent oscillation, compensation for the fading amplitude within the PLP buffer should be enforced. Second, the pulse information should be separated into pulse oscillation and beat stability measure. In Section 3, we will examine the normalization process to bring the PLP buffer into a more standardized form.

3. NORMALIZATION

To address the challenges outlined in Section 2.2, in this section, we explore various layers of normalization for the PLP buffer. First, in Section 3.1, we introduce a global kernel window function, followed by a PLP envelope function in Section 3.2. Subsequently, we utilize these functions to normalize the PLP buffer in multiple steps, as described in Section 3.3.

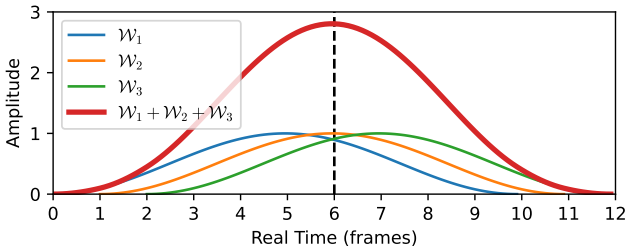


Figure 3: *An example demonstrating three shifted PLP kernel windows (Hann function) contributing to an overlap-added kernel window function.*

3.1. Overlap-Add of Kernel Window Functions

PLP is computed by adding overlapping kernels κ_ℓ, each obtained by multiplying a window function $\mathcal{W}$ with a maximum amplitude of one. To determine the maximum possible amplitude of the PLP function Γ_{n_0}, we assume perfect constructive interference of the shifted kernels and compute the overlap-add of the kernel window functions $\alpha_{n_0} : [n_0 - N : n_0 + N] \to \mathbb{R}$, defined by

$$\alpha_{n_0}(n) := \frac{1}{C} \sum_{\ell=n-N}^{n_0} \mathcal{W}(n-\ell), \tag{4}$$

for $n \in [n_0 - N : n_0 + N]$. The constant C, as defined in Equation 3, ensures that α_{n_0} lies within the range of values $[0 : 1]$. Since α_{n_0} from Equation 4 is shift invariant and does not change for different values of n_0, we can pre-calculate this normalization function based on the window configuration alone.

For demonstration purposes, in Figure 3, we illustrate only three shifted and overlap-added kernel windows $\mathcal{W}_1$, $\mathcal{W}_2$, and $\mathcal{W}_3$ that combine to form a sum of kernel windows with an overall higher amplitude. The exact maximum value of that sum varies depending on factors such as the hop size, window length, window type, and the total number of overlapping kernels. When running a real-time PLP procedure, all of these settings are known and can be pre-computed to remain constant during runtime.

In this way, α_{n_0} provides an important stage of the PLP buffer

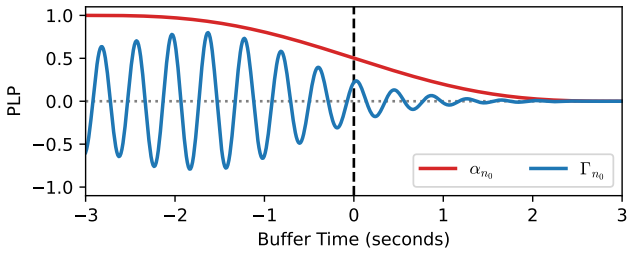


Figure 4: *The PLP function Γ_{n_0} with the overlap-added kernel window function α_{n_0} indicating maximum boundaries.*

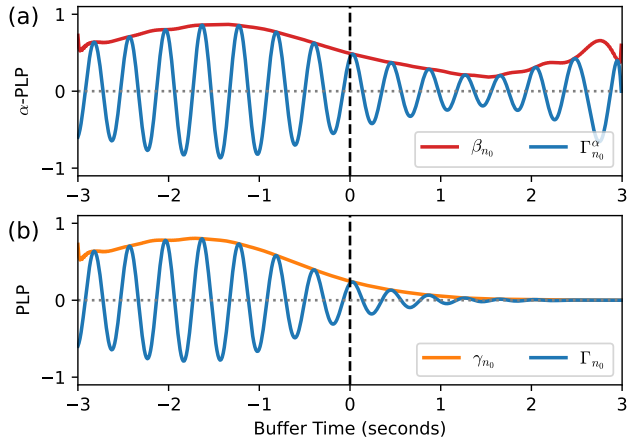


Figure 5: *(a) The α-normalized PLP function $\Gamma^{\alpha}_{n_0}$ with envelope β_{n_0}. (b) The PLP function Γ_{n_0} with envelope γ_{n_0}.*

normalization, as depicted in Figure 4. The overlap-add of kernel window functions α_{n_0} acts as the upper limit of the PLP function Γ_{n_0} across all time positions n and also dictates the shape of the fading amplitude within the PLP buffer for $n \in [n_0 - N : n_0 + N]$. The closer the PLP function Γ_{n_0} approaches the boundaries of α_{n_0}, the more stable the local tempo structure of the audio is. To utilize the PLP buffer without the influence of the fading amplitude, we can define an α-normalized variant of Γ_{n_0}, denoted by

$$\Gamma^{\alpha}_{n_0} := \frac{\Gamma_{n_0}}{\alpha_{n_0}}. \tag{5}$$

3.2. PLP Buffer Envelope

Another measure of stability is derived from the PLP function itself, or more precisely, from the **envelope** formed by its peak positions, as illustrated in Figure 5a. The envelope β_{n_0} of the PLP function $\Gamma^{\alpha}_{n_0}$ represents the magnitude of its oscillation, which can be computed using the Hilbert transform [14]. The higher the value of β_{n_0}, the more stable the beat structure of the audio, and vice versa. In this way, the envelope β_{n_0} serves as a valuable analytical tool for expressing the beat stability, particularly when the PLP function Γ_{n_0} is already α-normalized. Note that, even though $\Gamma^{\alpha}_{n_0}$ falls within the value range $[-1 : 1]$, this does not automatically guarantee that the envelope β_{n_0} is limited to the range $[0 : 1]$. To ensure the desired normalization, in practice, we simply clip all values to 1 where $\beta_{n_0} > 1$.

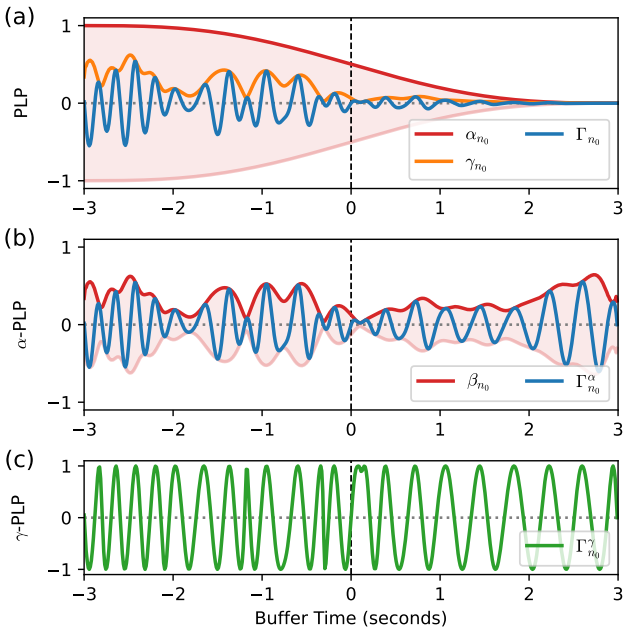


Figure 6: *The PLP buffer normalization in three steps: (a) PLP function Γ_{n_0} with normalization α_{n_0} and envelope γ_{n_0}. (b) α-normalized PLP function $\Gamma^{\alpha}_{n_0}$ with envelope β_{n_0}. (c) γ-normalized PLP function $\Gamma^{\gamma}_{n_0}$.*

Furthermore, the combination of α- and β-normalizations can also prove to be beneficial. For instance, when using future buffer time positions $n > n_0$, a descending slope fading to zero could be advantageous to express the uncertainty of the predictions. For this use case we define a second envelope

$$\gamma_{n_0} := \alpha_{n_0} \cdot \beta_{n_0}, \tag{6}$$

as illustrated in Figure 5b. While theoretically γ_{n_0} could be computed directly from the Hilbert transform of Γ_{n_0}, we found that cleaner signals are obtained when applying α-normalization first to prevent values from fading to zero. This approach can reduce unwanted edge effects when using the Hilbert transform.

3.3. PLP Buffer Normalization

With the overlap-added kernel window function α_{n_0} (Section 3.1) and the envelopes β_{n_0} and γ_{n_0} (Section 3.2), we have multiple layers for normalizing the PLP buffer Γ_{n_0}, as illustrated in Figure 6.

Starting with Figure 6a, both Γ_{n_0} and α_{n_0} already fall within the range of $[-1 : 1]$ and $[0 : 1]$, respectively, using the constant C from Equation 3. Γ_{n_0} exhibits a fading amplitude within the PLP buffer, descending to values near zero, which can be compensated by normalizing with α_{n_0}. The resulting α-normalized PLP function $\Gamma^{\alpha}_{n_0}$ is illustrated in Figure 6b. Note how the normalization process of the α-PLP is especially relevant for the right side of the PLP buffer, where oscillations for future time positions $n > n_0$ are now clearly visible instead of fading to zero. In this state, we can consider $\Gamma^{\alpha}_{n_0}$ as a pulse oscillation modulated in amplitude by its beat stability, which is depicted by the envelope β_{n_0} in Figure 6b. Combining α_{n_0} and β_{n_0} results in an envelope γ_{n_0} that matches

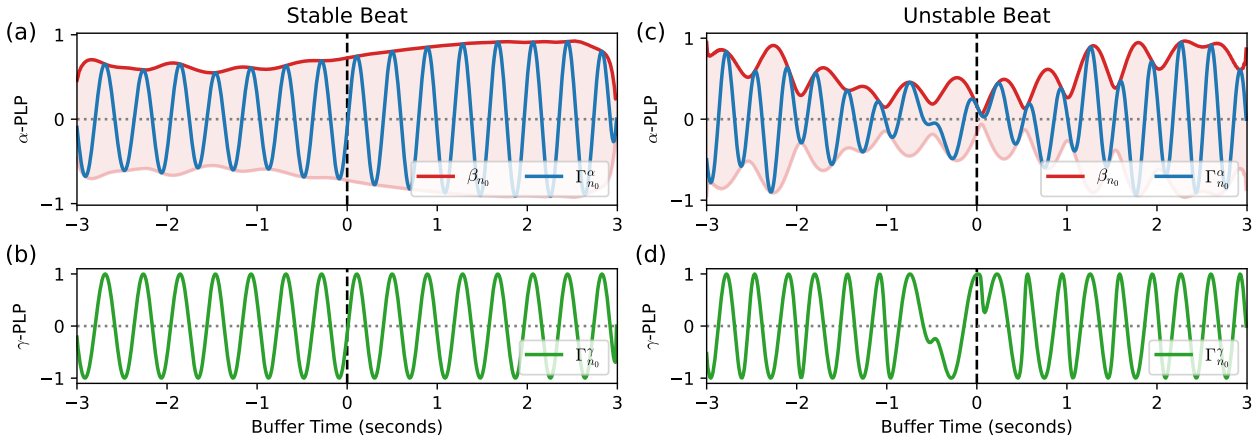


Figure 7: *Illustration of stable and unstable beat scenarios for α-normalized and γ-normalized PLP functions.*

Γ_{n_0}, as illustrated in Figure 6a. To separate pulse oscillation from beat stability measures, we can either normalize $\Gamma^{\alpha}_{n_0}$ with its envelope β_{n_0} or directly normalize Γ_{n_0} with its envelope γ_{n_0}, which yields identical outputs. The resulting γ-normalized variant of the PLP function, denoted as $\Gamma^{\gamma}_{n_0}$, exhibits a pulse oscillation between $[-1:1]$ with constant amplitude but without the influence of beat stability, as depicted in Figure 6c.

With α_{n_0}, β_{n_0}, and γ_{n_0}, we now have multiple separate PLP normalizations that offer insights into the local pulse structure in different ways. In Figure 7, we provide two examples of how these normalizations behave for stable and unstable beat tracking scenarios. A stable beat tracking scenario, as depicted in Figure 7a, is characterized by consistently high values for the envelope β_{n_0}, with only small variations close to one. As a result, for $\Gamma^{\gamma}_{n_0}$ in Figure 7b, we observe a highly consistent pulse curve resembling a clean sinusoidal waveform with a regular frequency oscillation. An unstable beat tracking scenario, as illustrated in Figure 7c, results in significant fluctuation in the envelope β_{n_0}, with peak values close to zero. As a result, for $\Gamma^{\gamma}_{n_0}$ in Figure 7d, we observe a more inconsistent pulse curve with phase shifts indicating tempo changes and a higher amount of non-regular frequency modulations.

With this transformation of the PLP buffer into normalized versions $\Gamma^{\alpha}_{n_0}$ and $\Gamma^{\gamma}_{n_0}$, we can derive a variety of different control signals, as we will discuss in Section 4.

4. CONTROL SIGNALS

In Figure 8, we present an example of a real-time PLP procedure using a short audio excerpt. For Figure 8a, we display the waveform of the audio with a black cursor fixed at the current time position n_0. At this particular time position, we observe the internal state of the PLP buffer, with the current values for tempogram $\mathcal{T}(n,\tau)$, kernels $\kappa_\ell(n)$, and PLP function $\Gamma_{n_0}(n)$. Note that we distinguish between time axis for *real time*, such as the audio waveform, and *buffer time*, which serves as an internal time measure within the PLP buffer, referencing the neighborhood of the current time position n_0. For each time position n_0, we apply the normalization methods α_{n_0}, β_{n_0}, and γ_{n_0}, as described in Section 3. From these normalizations, we can derive distinct control signals, as discussed in the next subsections.

4.1. Beat Confidence

The α-normalized PLP contributes a control signal by utilizing the beat stability of the PLP function $\Gamma^{\alpha}_{n_0}$. Specifically, we define *beat confidence* at time position n_0 as the value of the envelope β_{n_0} at the central buffer position. The resulting control signal, labeled β-confidence, is depicted in Figure 8b.

In parallel, we can derive a similar control signal γ-confidence, which is based on Γ_{n_0} and captures the value of the envelope γ_{n_0} for each real time n_0, as depicted in Figure 8c. The γ-confidence shares similarities with the β-confidence but exhibits a lower amplitude. This characteristic could prove especially beneficial if we opt to use the PLP look-ahead capability and adjust the buffer read position to future or past time positions, as we will discuss in Section 4.3. In such scenarios, the γ-confidence could effectively represent the integration of more uncertain future pulse information, encoded with a reduced amplitude.

To clarify the significance of beat confidence, let's examine the audio excerpt in Figure 8a. The lowest confidence value in the presented audio occurs around eight seconds of real time, where the music is fading out and no beat information is available. The subsequent section, spanning from seconds 8 to 12, is marked by consistent and rhythmic playing, resulting in a notably higher beat confidence.

4.2. Low Frequency Oscillator

The α-normalized PLP contributes another control signal by utilizing the pulse oscillation of the PLP function $\Gamma^{\alpha}_{n_0}$. Specifically, we define the *low frequency oscillator* α-LFO as the value of $\Gamma^{\alpha}_{n_0}$ at the central buffer position for each real time n_0. The corresponding control signal is shown in Figure 8d. Additionally, from $\Gamma^{\gamma}_{n_0}$ we can directly derive γ-LFO, which is depicted in Figure 8e. While the γ-LFO exhibits a clean, uniform oscillation between values of $[-1:1]$, the α-LFO combines both γ-LFO and β-confidence, with the confidence modulated onto the signal's amplitude.

To explain the different behaviours of the LFOs defined, we can again utilize the example in Figure 8a. Both the α-LFO and the γ-LFO clearly visualize the tempo structure of the provided song. At the beginning of the audio excerpt, we observe a mid-tempo oscillation, followed by a sudden change to a high-tempo

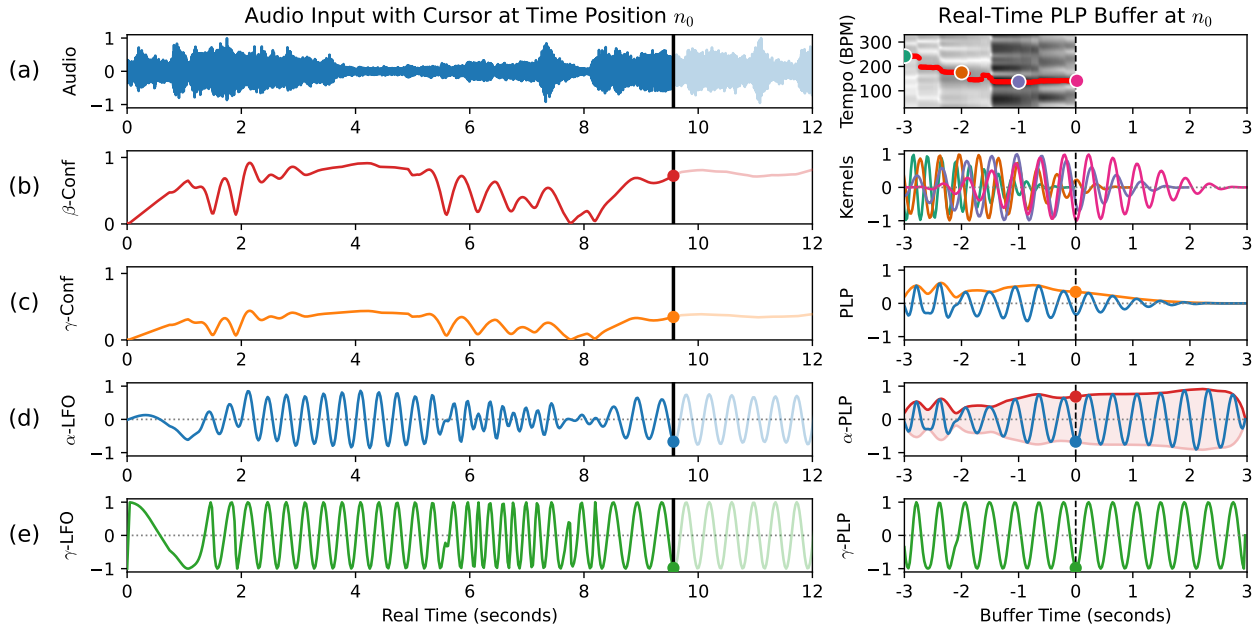


Figure 8: *An audio excerpt featuring: (a) Audio input and control signals, (b) β-Conf, (c) γ-Conf, (d) α-LFO, and (e) γ-LFO. The black cursor indicates the current time position n_0. The right side of the figure displays the inner state of the PLP buffer at that time position. An animation of this audio excerpt is available on our supplemental website.*

section around 5-8 seconds, and finishing with a low-tempo section towards the end of the audio track.

4.3. Look-ahead Capability

For each time position n_0, we can analyze a PLP buffer where the left half of the buffer shows past pulse information and the right half predicts future pulse information. This prediction is based on the centered pulse kernels that drive the real-time PLP procedure (see Section 2.1). The central buffer position corresponds to the current time position and is typically employed to retrieve values for control signals, as described in Section 4.1 and Section 4.2. However, for the buffer read position, any buffer time can be chosen, leading to some useful and interesting time-related effects. For example, by shifting the buffer read position to future time positions, we can trigger beats earlier and obtain control signals that operate ahead of time, as illustrated in Figure 9.

For Figure 9a, we see a PLP function Γ_{n_0} and two different buffer read positions labeled as `Cursor 1` and `Cursor 2`. While `Cursor 1` is positioned at the central buffer position, `Cursor 2` utilizes the look-ahead capability of real-time PLP and is shifted to a future time position. As a consequence, in Figure 9b, the γ-LFO for `Cursor 2` is consistently ahead of time in oscillation and reaches its peaks earlier than the `Cursor 1` version of the LFO. This can be advantageous for two distinct reasons. First, for technical purposes, it allows compensating for system latency effects and synchronizing the generated system output with the analyzed audio input even if the processing introduces an inherent delay. Second, for creative purposes, it enables time-related effects that need to start earlier to finish at the upcoming beat position, as detailed in Section 5.2. In this way, our system offers multiple adjustable control signals suitable for various applications, as we will discuss in Section 5.

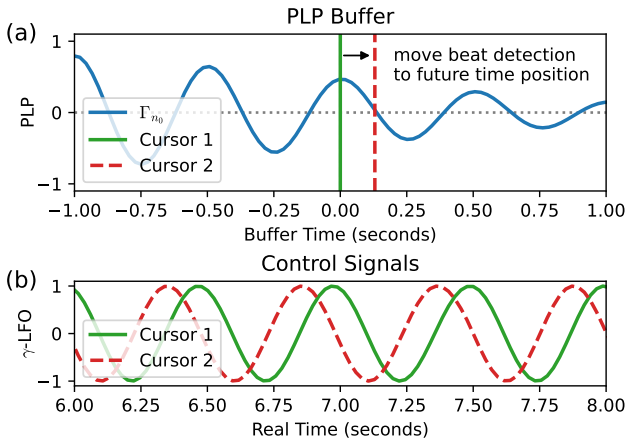
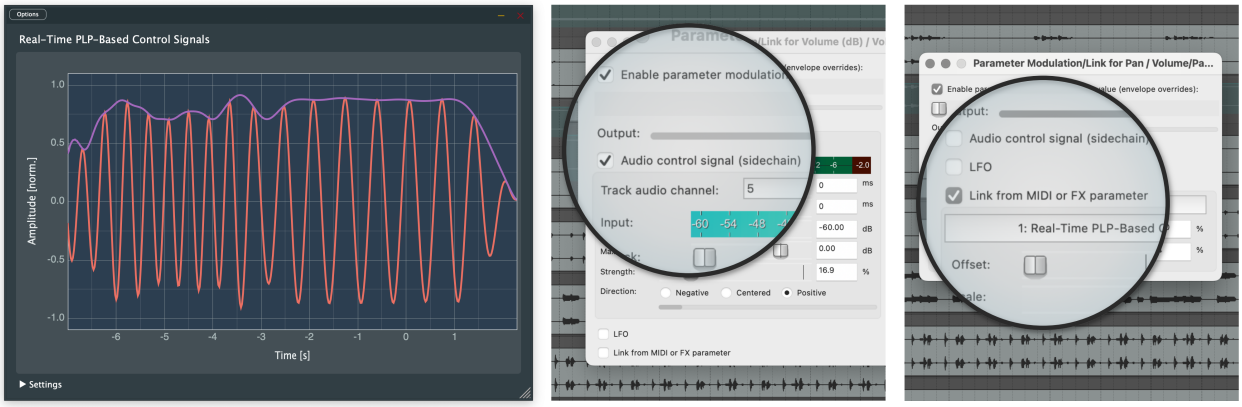


Figure 9: *Demonstration of the PLP buffer with look-ahead capability, shown with two different buffer read positions (cursors) for the γ-LFO control signal.*

5. APPLICATIONS

The control signals α-LFO, γ-LFO, β-confidence, and γ-confidence represent musical properties of the analyzed signal which can be utilized in different ways for creative mixing applications. To demonstrate some use cases in a practical setting, we developed an audio plugin prototype with JUCE [15] (Figure 10a) that can generate the control signals in real-time from any single-channel audio input directly in a DAW. Since many DAWs offer the possibility to use sidechain audio signals to control the modulation of effect parameters (see Figure 10b for an example using REAPER [16]), the control signals are provided as separate audio output



(a) (b) (c)

Figure 10: *The prototype implementation and integration in REAPER: (a) The audio plugin with visualization of the real-time PLP buffer. (b) A sidechain audio signal for effect parameter modulation in REAPER, accessible via the menu* `Param > FX parameter list > Param modulation/MIDI link`*. (c) Utilizing a time-varying parameter value for effect parameter modulation.*

channels of the plugin. Note that this method requires upsampling of the control signal from the frame rate (i.e., the sampling rate of Δ) to the audio sampling rate. Alternatively, the control signals can be made available as time-varying parameter values of the plugin itself, which can be updated at frame rate and may also be used as a source signal for parameter modulation in REAPER (see Figure 10c).

In addition, our implementation includes a real-time visualization of the current α-PLP buffer and its envelope β_{n_0}, which gives insight into the current tempo structure of the analyzed music signal. This provides both a visual indicator for current stability and tracked tempo octave, as well as educational value for exploring PLP settings. For instance, using a microphone input allows for a direct exploration of how parameters such as kernel size or tempo range affect PLP behavior in an intuitive way.

In the following, we will discuss three case studies for the control signals, highlighting musical implications, strengths and limitations of the presented approach for mixing. The examples are further complemented with audio excerpts on our supplemental website.

5.1. Case Study 1: Volume Control with Beta-Confidence

Musically, the β-confidence control signal is high when the beat is stable in a musical part and low where the beat is unstable. The α-confidence behaves similarly, but additionally factors in a reduced overall confidence when the buffer read position uses look-ahead and is moved into the future (since predicting future beat positions is inherently less reliable). In this way, both confidence signals are suitable for modulating effect parameters that should vary based on the presence of a stable beat. As an example, we may want to change the volume of a pad or drone sound during a bridge part which in popular music often coincides with a change in rhythmic patterns and a reduction of tempo stability. Here, we use the β-confidence to directly control the volume parameter of the channel.

We observed that a low tempo stability in the surrounding musical part often correlates with only short decreases in the confidence control signal. To mitigate this issue, we apply minimum filtering with a variable window size up to multiple seconds, so that a single PLP peak with low amplitude reduces the confidence value for a larger time span. However, since the minimum filter window can only be applied to past confidence values in a real-time setting, this may unintentionally delay the increase of confidence at the beginning of stable sections, so that the window size should be balanced accordingly (see Section 5.2). After applying such modifications to the confidence, we can also generate a new α-LFO signal by multiplying γ-LFO with the filtered confidence. In addition, while DAWs typically offer various ways to scale the control signal for the desired use case (for example, a confidence of 0 could be mapped to a fader value of -15 dB and a confidence of 1 to 0 dB), a non-linear transformation of the confidence (like exponentiation with an exponent > 0) may further help to achieve the desired effect.

5.2. Case Study 2: Noise Gate with Alpha-LFO

The α-LFO signal allows for rhythmically modulating an effect parameter based on the current beat estimates in sections where the beat is stable, while no modulation is done in unstable parts. Musically, this control signal is particularly suitable for effects where the beat-synchronicity is perceptually prominent, since in such cases it may be preferable to apply no modulation instead of a modulation that is out of sync with the music. As an example, we can trigger a noise gate using α-LFO, generating a shaker-like effect that is only present in stable beat parts. The look-ahead in the real-time PLP buffer additionally allows to adjust the phase of the α-LFO, so that possible latencies of the entire signal processing chain can be compensated for. Furthermore, larger look-aheads can be used to trigger an effect before the next estimated beat, e.g. to generate an off-beat or "reverse snare reverb" effect.

In this context, three limitations of the α-LFO signal may become relevant. First, since a stable beat can only be established after a few pulses have aligned, the α-LFO exhibits a "build-up" at the beginning of sections with a new rhythmic pattern, which is further intensified by minimum filtering the confidence as described in Section 5.1. Second, it may be desirable to freely choose the tempo octave of the oscillator to control effect parameters on the intended beat division level. This can be influenced to some degree by the tempo range of the local tempo estimation, but a narrow range may lead to unwanted instability when the desired tempo octave is not prominent in the analyzed input signal. Third, many parameter modulations benefit from non-sinusoidal wave-

forms (e.g., square or sawtooth) to create the desired effect. This may be achieved in future work through a non-linear transformation of the oscillator signal.

5.3. Case Study 3: Rhythmic Panning with Gamma-LFO

The γ-LFO is not scaled proportional to beat stability, so that this control signal always utilizes the full modulation range, but may have large and sudden frequency fluctuations in musical parts with low tempo stability. On the other hand, this LFO signal is not affected by the confidence build-up described in Section 5.2 and can thus be applied for effects where using the full modulation range is always desirable. As an example, we use the γ-LFO to modulate the pan parameter for a channel where the recorded instrument plays in a steady rhythm, causing each played note to alternate between the left and right channels. Since the panning effect is not as perceptually critical w.r.t. timing as for example the shaker in Section 5.2, some smaller inaccuracies of the γ-LFO synchronization do not negatively affect the outcome, but can rather lead to interesting variations in the mix.

6. CONCLUSIONS

In this article, we introduced a novel method for deriving pulse-synchronous LFOs and supplementary confidence-based control signals from a real-time PLP buffer. In this context, we discussed several normalization steps for the PLP output aimed at separating pulse oscillation from beat stability measures. We further developed a pulse tracking audio plugin prototype, demonstrating creative applications of our method, including its look-ahead capability, and offering educational insights into the real-time PLP procedure. With this, our goal is to provide a practical method for musicians and mixing engineers, allowing them to automate beat-synchronous audio effects in live performances or to creatively utilize real-time control signals for mixing applications. For future research, we aim to conduct subjective testing to validate the case studies described in this paper. Equally interesting would be an analysis of listener perception to identify useful parameter settings for our method.

7. ACKNOWLEDGMENTS

This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Grant No. 500643750 (MU 2686/15-1). The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and Fraunhofer Institute for Integrated Circuits IIS.

8. REFERENCES

[1] Udo Zölzer, *Digital Audio Signal Processing*, John Wiley & Sons, Hoboken, New Jersey, 2nd edition, 2008.

[2] Meinard Müller, *Fundamentals of Music Processing – Using Python and Jupyter Notebooks*, Springer Verlag, 2nd edition, 2021.

[3] Sebastian Böck and Matthew E. P. Davies, "Deconstruct, analyse, reconstruct: How to improve tempo, beat, and downbeat estimation," in *Proceedings of the 21th International Society for Music Information Retrieval Conference, ISMIR 2020, Montreal, Canada, October 11-16, 2020*, Julie Cumming, Jin Ha Lee, Brian McFee, Markus Schedl, Johanna Devaney, Cory McKay, Eva Zangerle, and Timothy de Reuse, Eds., 2020, pp. 574–582.

[4] Jingwei Zhao, Gus Xia, and Ye Wang, "Beat transformer: Demixed beat and downbeat tracking with dilated self-attention," in *Proceedings of the 23rd International Society for Music Information Retrieval Conference, ISMIR 2022, Bengaluru, India, December 4-8, 2022*, 2022, pp. 169–177.

[5] Yun-Ning Hung, Ju-Chiang Wang, Xuchen Song, Wei Tsung Lu, and Minz Won, "Modeling beats and downbeats with a time-frequency transformer," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2022, Virtual and Singapore, 23-27 May 2022*. 2022, pp. 401–405, IEEE.

[6] Peter Meier, Simon Schwär, Sebastian Rosenzweig, and Meinard Müller, "Real-Time MIR Algorithms for Music-Reactive Game World Generation," in *Mensch und Computer 2022 - Workshopband*, Bonn, 2022.

[7] Peter Meier, Gerhard Krump, and Meinard Müller, "A real-time beat tracking system based on predominant local pulse information," in *Demos and Late Breaking News of the International Society for Music Information Retrieval Conference (ISMIR)*, Online, 2021.

[8] Adam M. Stark, Matthew E. P. Davies, and Mark D. Plumbley, "Real-time beat-synchronous analysis of musical audio," in *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, Como, Italy, 2009.

[9] Mojtaba Heydari, Frank Cwitkowitz, and Zhiyao Duan, "Beatnet: Crnn and particle filtering for online joint beat downbeat and meter tracking," in *22th International Society for Music Information Retrieval Conference, ISMIR*, 2021.

[10] Mojtaba Heydari, Matthew McCallum, Andreas Ehmann, and Zhiyao Duan, "A novel 1d state space for efficient music rhythmic analysis," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 421–425.

[11] Chih-Cheng Chang and Li Su, "Beast: Online joint beat and downbeat tracking based on streaming transformer," 2024, Accepted by ICASSP 2024.

[12] Peter Grosche and Meinard Müller, "Extracting predominant local pulse information from music recordings," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 6, pp. 1688–1701, 2011.

[13] Peter Meier, Ching-Yu Chiu, and Meinard Müller, "A real-time beat tracking system with zero latency and enhanced controllability," *Transactions of the International Society for Music Information Retrieval (TISMIR)*, In Review.

[14] Julius O. Smith, *Spectral Audio Signal Processing*, W3K Publishing, `http://books.w3k.org`, 2011.

[15] Raw Material Software Limited, "JUCE framework for audio application and plug-in development," `https://juce.com`, 2024.

[16] Cockos Incorporated, "REAPER digital audio workstation," `https://www.reaper.fm`, 2024.