Audio Engineering Society

Convention Paper

Presented at the 121st Convention
2006 October 5–8 San Francisco, CA, USA

Groovator - an implementation of real-time rhythm transformations

Jordi Janer[1], Jordi Bonada[1], Sergi Jordà[1]

[1] *MTG, Universitat Pompeu Fabra, Barcelona*

Correspondence should be addressed to Jordi Janer (jjaner@iua.upf.edu)

ABSTRACT

This paper describes a real-time system for rhythm manipulation of polyphonic audio signals. A rhythm analysis module extracts information of tempo and beat location. Based on this rhythm information, we apply different transformations: *Tempo*, *Swing*, *Meter* and *Accent* . This type of manipulation is generally referred as Content-based transformations. We address characteristics of the analysis and transformation algorithms. In addition, user interaction plays also an important role in this system. Tempo variations can be controlled either by tapping the rhythm with a MIDI interface or by using an external audio signal such as percussion or the voice as tempo control. We will conclude pointing out several use-cases, focusing on live performance situations.

1. INTRODUCTION

Digital music production has motivated the development of new tools for sound manipulation. However, the use (or abuse) of these tools may limit the musical expression. In sequencers, for instance, the generated music follows a predetermined rhythm pattern, lacking the expression of a real musician. Also in a performance situation, artists often use pre-recorded material that is then played back . It forces the artist to stick to the "machine-tempo", and thus avoiding any inspired variation, which constitutes the *raison d'être* of live performances. Our aim is contributing with a real-time tool that integrates several rhythm transformations, assuring at the same time sound quality and adequate user interaction. We propose transformations that are based on the rhythm analysis of an audio signal. These transformations consist mainly in altering the time axis with a time-scaling algorithm.

In the literature, the type of transformations addressed here are referred as *Content-based transformations* [1] or *Adaptive Digital audio Effects* [8]. Basically, it means that the transformation depends on the input signal analysis.

Several approaches proposed innovative rhythm manipulations, either altering prerecorded material [5] or as extensions of the Beat-Boxing retrieval [7, 3]. With this paper, authors contribute with a fully working system in real-time, offering on one side various transformations, and, on the other side, addressing explicitly *user interaction*, focusing on live performance. Our implementation works on general polyphonic audio, and is composed of two modules: rhythm analysis, and transformation.

2. RHYTHM ANALYSIS

A rhythm analysis module extracts tempo and time location of every beat in the beat sequence. In the areas of Music Cognition and Music Computing, *Rhythm Induction* has widely attracted the interest of researchers, describing different rhythmic representations and algorithms [6, 4].

The rhythm analysis algorithm implemented in the *Groovator* is based on [4]. It splits the input signal in eight perceptual frequency bands. For each frequency band, an envelope of the logarithmic energy derivative is calculated. These eight envelopes pass through a *Half-Wave Rectifier* in order to keep only the positive energy variations. The following step is to compute the autocorrelation of these envelopes, which highlights the most likely tempo. Along with the tempo value in *bpm*, this module outputs also a sequence of the position of the eighth notes.

Nevertheless, the implemented algorithms present some limitations, as we discuss next. First, although having a resolution of eighth-notes, we don't know the location of the downbeat (*accentuated beat*). Second, a common problem in beat induction algorithms is octave errors. This issue has been widely studied from a computational as well as perceptual points of view, showing that in many cases this is a subjective question[1]. And finally, in music with non-stationary tempo such as classical music, it is less likely that the rhythm is tracked properly, since the analysis module assumes a stationary tempo within an analysis window. For these situations, in addition to the automatic rhythm analysis, the system offers a *manual analysis* mode. The location of the beat is specified by tapping the tempo manually(see section5.1), allowing transformations such as *swing* to a classical piece.

[1]For a given song, two subjects may assign tempo values that are in different octaves

3. TRANSFORMATIONS

In this section, we describe the four proposed rhythm transformations: *tempo*, *swing*, *meter* and *accent*. These are applied to a general audio signal. As explained previously, we have to bear in mind that the rhythm analysis may give octave errors depending on the sound material, taking quarter notes instead of eight notes, resulting in a lower tempo. These errors will affect the last three transformations, which depend on the beat position. In order to overcome this issue, we let the user correct the tempo to a higher octave with the DOUBLE button in the interface (see fig. 1).

- **Tempo.** This transformation, also referred as *tempo conducting*, makes use of a high-quality time-scaling algorithm by J.Bonada[2]. The playback speed can be modified in a range from half-speed and double-speed. When the user controls the tempo, either directly by tapping or indirectly with an external audio stream. The time-scaling factor(TS) is computed as $TS = T_u/T_s$, being T_u the user tempo, and T_s the audio source estimated tempo. T_u may not be very stable, leading to fast changes of the time scaling factor that can be noticeable. In order to limit the variations of the time scaling factor, we can use the *Inertia* slider control. This control smooths the time scaling factor. Although there will be always a delay of one beat before the user tempo is applied, we observed that the user taps adapting continuously to the downbeats. It ensures the synchronization in a *tempo conducting* situation.
- **Swing.** In this transformation, based on [5], the global tempo is kept constant and time scale modifications are applied only within a quarter note, modifying inversely the duration of the first and the second eighth-note in order to control the swing. Since the rhythm analysis estimates eighth-note positions, it may occur that the swing is applied on the quarter-note instead of on the eighth-note. The OFFBEAT button in the interface allows to switch easily between the two options. Also, user can modify the amount

Fig. 1: *Groovator's* user interface for the streaming implementation. It is divided in three sections: Analysis, Control and Effects. On top, use set the look-ahead buffer size in seconds.

of swing ratio from the "no swing" 1:1 ratio, to the 3:1 ratio, i.e. first eighth-note lasting three times the second. The swing ratio is controlled with the *factor* slider on the interface. User can also apply it inversely, resulting in a swing removal effect.

Additionally to the initial transformations of *tempo* and *swing*, we propose two other transformations that are based on the rhythm analysis:

- **Meter.** In contrast to the swing effect where the transformations is applied within a quarter note, here we transform the meter. The process consists in repeating or deleting the last quarter-note in a measure. For example, to convert a 3:4 meter into a 4:4 meter, we repeat the last beat.
- **Accent.** It consists in changing the perceptual characteristics of one beat. Many effects could be applied (pitch shifting, reverb, etc.). In our first implementation, we used a bass-booster, consisting in equalization and level increment.

Note that in these last two transformations, the *downbeat* position within the measure has to be selected manually. We use the *factor* slider on the GUI, quantized to four steps for this task.

4. SYSTEM IMPLEMENTATION

The system runs in real-time as a VST[2] plug-in. There are two different implementations for the two operation modes: playback and streaming. The former is a VST-Instrument, while the latter is a VST-Effect. Next section describes the main differences of both implementations.

[2]VST is trademark of Steinberg http://www.steinberg.de

4.1. Playback and Streaming modes

Although, the system was initially conceived only for playback mode, we implemented two versions of the Groovator: as VST-Instrument (*playback*) and as VST-Effect (*streaming*) for an easier integration in diverse audio applications. The former processes and audio file, while the latter processes an incoming audio stream. Although the transformations are the same in both implementations, the VST-Effect version present some limitations as described below.

In streaming mode, tempo variations are limited to the length of a internal *look-ahead* buffer. In order to make the system robust, we implemented a self-recovering method that automatically initializes the buffer. It is achieved by changing the time scale factor smoothly until the buffer gets to its initial position (half the *look-ahead* buffer size). At that moment the user can change the tempo again. The recovering process is done in two-steps as shown in the Figure 2.

Also, in *streaming* mode, the Groovator introduced an initial latency of $284ms$ due to the time-scaling algorithm. We modified the time-scale algorithm, achieving a lower latency of $100ms$. However, this modification implied a lower quality in the time-scaling algorithm, mostly perceived in transients.

4.2. Other Technical aspects

The current implementation runs on any VST host application. In terms of computational load, the system uses 14% and 22% of CPU time, for the playback and streaming implementations respectively. Measurements were completed on a PC Pentium IV, $2.4GHz$ running Windows XP. The audio interface latency was set to 512 samples using ASIO drivers.

For tempo conducting, where MIDI input is required, we implemented a function for adjusting the MIDI interface latency. After activating the CALIBRATION button, the user taps to a sequence of synthesized impulses. The system computes automatically the delay, which is used later for synchronization issues. Besides to the MIDI input, the system offers the possibility of synchronizing with a beat

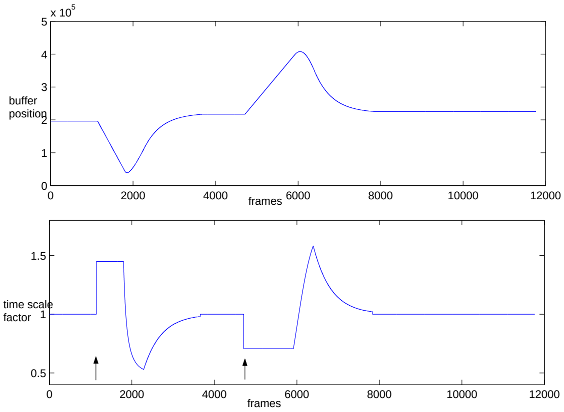

Fig. 2: The buffer recovering process applies two exponential decays to the time scale factor. The black arrows are the instants where the user modified the tempo. The top figure shows the evolution of the buffer reading position. The bottom figure shows the time-scale factor.

sequence using the Open Sound Control (OSC) protocol. On the GUI, user selects the listening port.

5. USER INTERACTION

An important characteristic of the presented system is the user interaction. In this section we enumerate some use-cases in which the presented system may be of great interest for musicians.

5.1. Use cases

- **Foot tapping.** By means of a MIDI foot controller, the user can drive the tempo of the Groovator. Other transformations parameters such as *Swing* factor is set manually in the GUI. A potential use case is a solo instrumentalist controlling his accompaniment with a foot pedal, thus freeing the musician from a fixed tempo.

- **Performance driven.** Here an input audio stream controls the tempo. Two typical use cases are: live percussion and voice control. In the former, a percussionist can change the tempo of an accompaniment audio file dynamically with his performance. This use case has its interest in the sense that it is not intrusive, since the percussionist can perform freely without bothering on computer interfaces. In the latter, the voice control use case, the rhythm is extracted from a beat-boxing performance. In fact, although the beat-boxing could be reduced to an impulse sequence, it is more engaging for the user to actually sing more complex rhythms along with the audio.

- **Loop-synchronization.** The Groovator may have its application as a DJ tool when we want for accomplishing loop synchronization. In this case, the beat sequence of a playback audio is synchronized to the control audio stream input. As discussed before, in the current implementation, the user decides whether to synchronize with the offbeat or with the downbeat using the OFFBEAT button on the GUI.

- **Manual tempo.** This is a special use-case where the tempo is not calculated automatically but rather determined by the user. By means of a MIDI interface, the user taps on the beat locations. In this mode, the Groovator is able to apply the above mentioned transformations to audio material with very unstable tempo such as classical music.

- **Multi-track.** A use-case of the *streaming* implementation is integrated in multi-track applications as insert effect. Despite the limitation described in the section 4.22, no audio file needs to be loaded, and thus the system becomes more flexible, processing any input stream.

- **Live audio stream.** Another, and more experimental use-case in *streaming* mode is the rhythm modification of a live audio stream, either a live performance, Internet jamming, or other sound sources such as TV or radio.

5.2. Preliminary user tests

Aiming at having a first user feedback, we asked some users to experiment with the system. All users had experience with digital audio tools and none was DJ. Among the two implemented versions (playback and streaming), users found the playback version far more useful for a live performance. Other users suggested to incorporate play-list management in the

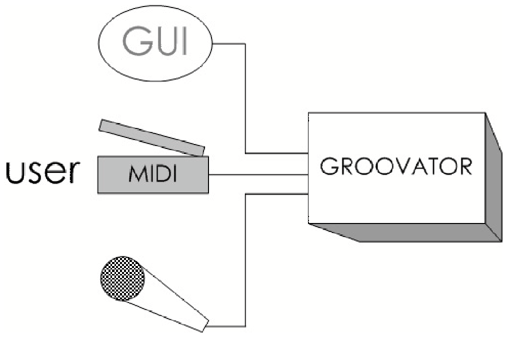

Fig. 3: User interaction with the *Groovator* for the tempo transformation: *top:* GUI controls; *middle:* foot-tapping with a MIDI controller; *bottom:* audio stream input (e.g beat-boxing).

GUI, instead of having to load audio files individually. Also, in the context of the Semantic HiFI project[3], the *Groovator* (in streaming mode) was integrated in the *Authoring Tool* prototype, developed by Native Instruments[4]. An evaluation experiment with users conducted by Native Instruments reported some feedback about the presented system. Users found the proposed transformations interesting, and mainly useful for loops. However they noticed artifacts due to the lower quality. As a result, we finally implemented the *streaming* version with the initial latency of $284ms$, which does not affect VST host application that supports the *Delay Compensation* feature.

6. CONCLUSIONS

The presented work offers ways of altering the rhythm characteristics of an audio mix. While several systems allow rhythm modification in an off-line manner, either at symbolic (MIDI) or at signal level, the main contribution of this implementation is to provide a real-time solution. Preliminary user evaluation pointed out that the system works particularly well on loops or songs with regular tempo, but showing to be less robust for complete songs. This is due to two facts. First, the beat analysis algorithms assumes a constant tempo within the analysis window (around 8 sec.); and second the beat tracker fails in some passages where a strong rhythm pattern is missing (e.g. solos). Methods to overcome this problem can be a confidence measure, so that for low confidence values, no rhythm transformation is applied in order to avoid artifacts. In terms of user interaction, two control modes have been proposed: tapping and input audio stream. Additionally, a GUI allow other configurations. To sum up, we argue that the proposed system can be used in live performances, enhancing the limited expression of sequencer-based live music.

[3]http://www.shf.ircam.fr

[4]http://www.nativeinstruments.de

7. ACKNOWLEDGMENTS

This research has been partially funded by the EU-FP6-IST-507913 project SemanticHIFI, and partially funded by the EU-FP6-IST project SALERO. The authors would also like to thank Ferran Relat for his previous work on tempo conducting.

8. REFERENCES

[1] X. Amatriain, J. Bonada, A. Loscos, J. L. Arcos, and V. Verfaille. Content-based transformations. *J. New Music Research*, 32(1):95–114, 2003.

[2] J. Bonada. *Audio Time-Scale Modification in the Context of Professional Post-Production.* PhD thesis, Pre-Thesis Work, Universitat Pompeu Fabra, Barcelona, 2002.

[3] O. Gillet and G. Richard. Drum loops retrieval from spoken queries. *Journal of Intelligent Information Systems*, 24:2/3:159–177, 2005.

[4] F. Gouyon. *A computational approach to rhythm description — Audio features for the computation of rhythm periodicity functions and their use in tempo induction and music content processing.* PhD thesis, Universitat Pompeu Fabra, Barcelona, 2005.

[5] F. Gouyon, L. Fabig, and J. Bonada. Rhythmic expressiveness transformations of audio recordings: swing modifications. In *Proc. Int. Conf. on Digital Audio Effects (DAFx-03), London*, pages 94–99, 2003.

[6] H. Honing. Structure and interpretation of rhythm and timing. *Dutch Journal of Music Theory*, 7(3:227–=232, 2002.

[7] G. Tzanetakis, A. Kapur, and M. Benning. Query-by-beat-boxing: Music retrieval for the dj. In *ISMIR-2004*, 2004.

[8] V. Verfaille. *Effets Audionumériques Adaptatifs : Théorie, Mise en Œuvre et Usage en Création Musicale Numérique*. PhD thesis, Université de la Méditerranée (Aix-Marseille II), 2003.