DIGITALLY ENHANCED DRUMS: AN APPROACH TO RHYTHMIC IMPROVISATION

Matteo Amadio
Conservatory of Music "C. Pollini", Padua
matteoamadio@outlook.com

Alberto Novello
Conservatory of Music "C. Pollini", Padua
albynovello@gmail.com

ABSTRACT

This paper presents a set of real-time modules that digitally enhance the performance of a drummer. The modules extract rhythmic information from the multichannel audio acquired using simple microphones onto the different drum parts. Based on the predicted tempo, the modules generate complex patterns that can be manually controlled through high-level parameters or can be left automatic while the system adapts to the playing condition of the drummer. Such an interactive system is intended mainly for an improvised solo performance confronting a human drummer with a computer; however, it could be effectively employed in improvisations with larger ensembles or installations.

1. HISTORICAL BACKGROUND AND RELATED WORK

In the early years of electronic music, when the new technologies began to be considered as valid and innovative means for musical expression, percussion instruments had a fundamental role in the transition between the traditional, *keyboard-influenced* musical language and the *all-sound music of the future* [1]. Several composers considered percussion a great match with electronics because of their common characteristics. Both instrument families can generate a broad spectrum of unpitched sounds which encouraged composers to focus on timbre and rhythm. Another aspect is the modularity that characterises both setups of the percussionist and the electronic musician: the possibility to swap parts of an instrument chain has an impact on the creation process, and opens new possibilities in the performative approach [1] [2] [3].

In the following years, the diffusion of digital technologies, computers and real time digital audio processing allowed the birth of new strategies for composition and live electronics. In the late 70's Chadabe started experimenting with digital *interactive composing systems*: algorithms capable of responding in complex ways to the actions of the performer [4]. In his pieces *Solo* and *Rhythms*, for example, the performer provides high-level input data to the system which elaborates them in order to produce the whole musical output. The performer influences the final result but is unable to control each single event, placing himself in a position where he needs to listen and react to the gestures of the computer. In 1983 George Lewis was working at IRCAM developing an interactive software that could automatically generate instrumental music and also analyse the performance of human musicians in order to play along with them [5]: this work established the base of his *Voyager* system. The idea of independence of the computer from the human performer led to the abolition of the "human leader/computer follower" hierarchy, in order to create the possibility to communicate only using the musical language [6]. As a result, some intentions and emotions expressed by the human performer could also be found in the electronic performance, confirming the achievement of an authentic man-machine musical interaction. This last concept was pointed out also by Robert Rowe in the description of one of his early works in the field of human-machine musical interaction, *Hall of Mirrors* [7]. Rowe describes the feedback loop generated by mutual imitation as two (or more) mirrors facing each other. Rowe developed his own interactive system named *Cypher*. In *Cypher* the user has to decide how the listener will interact with the player, allowing a high-level control on the actions of the software [8]. In the '90s both *Cypher* and *Voyager* were modified to include the use of MIDI data as input and output, in order to easily process all the necessary data [9].

In recent years, great progress has been made in the development of new interactive music systems and digital tools for the analysis of human performances. *B-keeper* is a real-time beat tracker specifically designed for live performance [10]. The software analyses the audio coming from the kick microphone of a drum set and syncs to the drummer's tempo. It creates the possibility to play pieces with pre-recorded audio or MIDI sequences without the constrictions imposed by the "fixed media", allowing the musicians to shift the performance tempo. Every temporal interval analysed by the system is weighted using probability distributions in order to discard non relevant data or reduce the impact of human errors on the stability of the system. By doing that, the algorithm privileges regularity over sudden changes and, as a consequence, it is less reactive in some musical situations with drastic tempo changes. Jeff Gregorio follows a completely reversed approach by considering the drum set as a whole polyphonic resonant system [11]. By applying electromagnets on the drum heads he sends synthesized sounds to activate the vibration of the drums: the software allows direct control over the tones that will be generated by the drums. The system can listen to the performance of a melodic instru-

ment and harmonize the notes that are played in order to generate an always varying musical accompaniment.
In *Agumenta* by Alessandro Guerri an electronic drum kit is used to send continuous event information to a software that extracts general parameters like tempo, density and dynamics [12]. This data is then used to control the generation of rhythmic patterns related to the performance of the drummer. In these drum kits all the information about the notes played by the drummer are sent as MIDI messages, so their analysis can produce very accurate results. On the other hand, the use of an electronic drums has the downside of the drastic reduction of the timbral expressivity of the instrument.
The augmented drum kit developed by Christos Michalakos is based on an acoustic jazz drum kit with drum triggers, contact microphones and a speaker mounted on it [13]. The software includes sound processing and synthesis modules that can be turned on and off in various ways: manually by using a MIDI controller, automatically after the recognition of specific acoustic elements or through a pre-programmed cue. The variety of sound processing modules and the presence of different modalities of interaction with the software allows the drummer to control a rich electronic section, capable of enhancing the performance in several different ways, but requires a complex set of transducers thus a long setup time.

In this paper we present our approach for a digitally enhanced acoustic drum kit combining and expanding the ideas of the aforementioned systems. Our main goal is to sample the acoustic sound of the drum kit with all its nuances, then extract high-level features such as tempo, cue onsets and density to rhythmically enrich the performance by counterbalancing the drummer style. In the design process we privileged the ease of setup and transportation for the system, requiring only few common microphones and a sound card.

2. SYSTEM STRUCTURE AND DESCRIPTION

2.1 Hardware

We use the acoustic drum kit in two ways: as acoustic source and as control interface for the software. Beyond the drums, the performer can use a self-built MIDI foot controller which is based on an Arduino Uno [1] and has six buttons that can be mapped to control the desired parameters. We chose an acoustic drum kit instead of an electronic one because, even if some parts of the software (e.g. onset detection) would have requested a minor amount of work, an important drawback would have been a drastic reduction of the possibilities of interaction between the performer and the sound of the drums. The acoustic sound of the drums is captured using four microphones: one placed inside the kick drum, one close to the snare and the two overheads that can be placed over the kit to capture its sound with a stereo microphone technique, or closer to some elements that are mostly desired to be included in the digital processing. The number and the position of the overhead microphones can be modified, but kick and snare

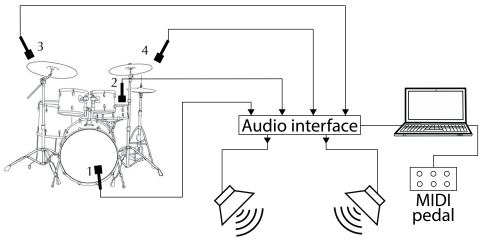

Figure 1: Scheme of the hardware used.

microphones are essential for the software to work properly because their signals are analysed in order to retrieve important data regarding the performance of the drummer.

2.2 Software

The software is programmed in Max/Msp [2] and is divided in two main sections: the first one analyses the inputs to extract rhythmical information regarding the performer's style in order to generate control signals, while in the second one the audio coming from the microphones is digitally processed to produce rhythmical effects related to the data retrieved. This allows the drummer to easily influence the action of the effects by just modifying, for example, the groove or by changing the metric subdivision of a fill.

The electronics is completely controlled by the data extracted from the performance except for a few manual controls: the *Reset* button which immediately sets the patch to its initial state, and a *Performance end* message that causes the electronics to slowly fade out and can be used whenever the drummer wants to end its exhibition. These messages can be sent through the use of the midi pedal and are not subject to any kind of variation operated by the software.

2.2.1 Features Extraction

In the first section the sound captured by the kick and snare microphones is analysed in order to detect the onsets of the single strokes and to send a message whenever one is identified. This is done by calculating the RMS amplitude over a 64 samples window and then routing the resulting signal to the *thresh*$\sim$ object, which acts as a Schmitt trigger: an onset is identified whenever the signal exceeds the higher threshold, but the object won't be able to detect another one until the input falls beneath the lower threshold. After this data is extracted, it will be used to control the modules of the second section, but also reanalysed in order to retrieve information about the rhythmical aspects of the performance, in particular its tempo and density.

Tempo is estimated by measuring the distance in milliseconds between each onset and the previous one and then arranging this data in a histogram which has on the X axis the time interval in milliseconds, and on the Y axis the number of repetitions of the corresponding value. The higher "peaks" are identified as useful data and are then analysed to generate the tempo value.

[1] Arduino Uno: https://store.arduino.cc/arduino-uno-rev3

[2] Max/Msp by Cycling '74: https://cycling74.com

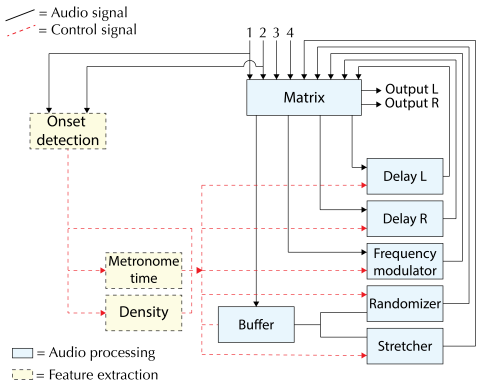

Figure 2: The structure of the software.

Density is estimated as the number of kick and snare onsets received over a two-seconds temporal window. The onset messages are routed into a counter: every time a message is received, the carried count is increased by one. At the same time, each message is delayed by 2 seconds and then routed back to the counter, this time causing it to decrease the count by one. By doing that we obtain a parameter that is highly reactive to the actions of the drummer and computationally cheap. The resulting signal is then smoothed and normalised in order to reduce the value to a floating-point number varying between 0 and 1. This parameter is very important for the musical interaction with the software because, when used as control signal, it helps creating a strong and musical link between the drummer's and software actions. As will be explained later, Density often controls the Probability parameter of the effects.

2.2.2 *Audio Routing and Effects*

All the calculated data is sent to the elements of the second section of the patch, which is formed by an audio routing matrix and five sound processing modules that we named *Delay*, *Frequency Modulator*, *Buffer*, *Stretcher* and *Randomizer*. In order to easily route all the audio signals and define their way from the microphones to the stereo output, we arranged a matrix that can be modified manually or in an automatic way.

The *Delay* is made of a time varying delay line with two feedback loops, one of which has a transposer in the middle that shifts the signal of a transposition ratio randomly chosen between 0.062 and 4 times its original pitch. This effect generates ascending or descending "quantised glissandos", creating rhythmical and timbral variety whenever the effect is activated. The presence of this double feedback allows to manipulate the timbre of the effect by choosing the desired mix between transposed and non-transposed signal. The incoming audio signal is analysed using the *bonk*~ [3] object in order to detect the presence of onset transients. If an onset is detected, the Delay effect can be triggered depending on a *Probability* value set by the user. The delay time and the length of the amplitude envelope of the effect are chosen probabilistically from multiples or submultiples of the estimated tempo.

The *Frequency Modulator* uses the audio input as the modulator signal for the frequency of a square wave oscillator. This signal is amplified and then the central frequency value is added; these two operations transform the input in order to bring it in a range where the modulation can be audible and effective on the carrier oscillator. The output of this effect is not continuous but, similarly to the delay, its amplitude envelope is triggered whenever the randomly generated number exceeds the gate imposed by the *Probability* value. The effect is triggered by snare onset messages.

The *Randomizer* and *Stretcher* use a monophonic audio buffer which is the recording of the signal coming from one of the audio outputs of the matrix, and can operate in a manual or automatic way. When in automatic mode, the recording is armed by the kick onsets with a probability of 3% and the recording starts on the next tempo tick. Manual commands override the automatic mode. While recording to buffer we temporarily deactivate the other effects that use the audio buffer from playback in order to avoid clicks. At the end of each recording, the *Randomizer* analyses the buffer through the use of the *slice*~ [4] object that divides the buffer in various segments based on the detection of transients in the signal. The buffer is rhythmically scrambled to create a playing queue where all the segments are permutated in time. The reproduction of this queue is automatically managed by comparing the *Density* parameter with a threshold set by the user. The queue starts to be played whenever the *Density* parameter goes beneath the threshold and it is paused when the threshold is exceeded or, if that does not happen, after 5 seconds. This method keeps the performance balanced and avoids confusion.

The *Stretcher* responds to the need of balancing all the short and percussive sounds with longer and softer ones. The audio output of this effect is mostly located in the lower region of the spectrum and is characterised by slow and often undefined attacks that create continuity in the performance. It uses the results of the analysis performed by the *slice*~ object and stretches the audio segments by playing them at much lower speed, causing them to be transposed and lengthened. The effect is triggered by the snare onset messages and the *Probability* parameter is controlled by the *Density* value. The probabilistic gate is analogous to the one described for the *Frequency Modulator* and whenever it is exceeded, the segment to be played and its new speed are randomly chosen.

2.3 System setup

Setting the system correctly is fundamental for the performance and it has to adapt to the playing style of the drummer. The gains for the kick and snare microphones are of critical importance as they influence the corresponding onset detection algorithms. Once verified the correct functioning of the onset detection algorithm, it will be sufficient to reset the patch and recall the desired preset in order to

[3] *bonk*~ by Miller Puckette: https://cycling74.com/forums/64-bit-versions-of-sigmund-fiddle-and-bonk

[4] *slice*~ by Nao Tokui: http://naotokui.net/2014/10/maxmsp-objects-on-github/

start the performance. The whole procedure usually takes few minutes.

3. APPLICATIVE SCENARIOS

The probabilistic gates make the interventions of the software unpredictable for a drummer. This aspect suggests an improvisational approach to the performance, in which the musician has to constantly modify his musical material to adapt to the electronics. The solo performance is the first and more immediate way to use this system, but interesting experiments could also be done by improvising with other musicians. They could play independently from the augmented drums system, or their sound could also be routed into the audio processing unit of the software, considering it as a sound source equal to the drum microphones. Another variation could be to use the audio from the drums microphones just to generate the control signals to process the sound of the other musicians.

Another possible application of this project is the realisation of an interactive installation where the audience through sensors or an interface could interact with the software and, as a consequence, with the performer. The audience could modify the parameters of the processing engine to add more or less disturbance elements to the drummer playing.

4. CONCLUSION AND FUTURE WORK

We described an augmented drum system for interactive electroacoustic improvisations between a drummer and the computer. The computer listens to the acoustic sound of the drums and generates effects and rhythmical patterns based on the analysis of the performance. We wanted to establish a close relationship between the actions of the drummer and the software in order to create a strong sensation of interaction between drums and electronics, also from the audience's point of view. However, to avoid a one to one correlation thus allowing for more complex interaction between acoustic and electronic events, we decided to implement an architecture using probabilistic gates. This last strategy suggests new directions to the drummer that can decide to follow or disregard the musical gestures generated by the electronics. Also due to this probabilistic architecture, the approach utilised in our software for the organisation of the electronic section can be considered similar to the concept of "sound-masses" as used by Varése.

At the current status the system gives a lot of musical freedom to the performer that can explore new possibilities without being constrained by technical limits or instabilities of the software. The effects interact between them in different ways, creating a various and well-balanced electronic section which fits the goal of enhancing the rhythmic patterns played by the drummer. The presence of an audio routing matrix and the possibility to store presets of the patch allow to create different sonic situations which can be utilised, for example, as different moments of a performance. From the performer's point of view the interaction with the software is smooth and doesn't require any extra-musical gesture. In other words, after completing some basic operations, the drummer will only have to sit on the drum kit and start his performance.

For the current implementation, the hardware used is just a basic drum kit and few common microphones: instruments and materials that could be easily found in the concert venue. We conceived it in this way so that the musician only has to bring the software, the relative computer and sound card. However, improvement can be done in order to create more variety in timbre by adding new elements to the kit. Also experimenting with different types of drum sticks or mallets and the use of other objects to excite the various drum parts is recommended. These new additions of course won't require modifying the software architecture. An important aspect we will be studying is the type and positioning of the microphones. The first idea that will be tested is the addition of two room microphones that will be placed far from the drum kit in order to capture and process the ambient response. Another possible improvement could be the use of drum triggers that will replace the onset detection section in the software, giving it more stability and a simpler structure.

On the software side, we are thinking of new mapping possibilities, also through the predisposition of another routing matrix for control signals. This revision will also introduce the possibility to easily set the action of the electronics to manual or automatic. In manual mode there will also be the possibility to give the control of the software to a second performer through the use of a MIDI controller, for more classic electro-acoustic concert setup. We consider fundamental the implementation of new features extraction algorithms that will provide a more accurate description of the ongoing performance. Some examples are a more precise beat tracker, the analysis of spectral features to identify different musical gestures, and the development of algorithms capable of classifying rhythmic patterns and to generate new ones. There's also the need of new effects and sound processing modules that, combined with expressive control signals, will enrich the electronic output. Some ideas are dynamic filtering, ring modulation, granulation and spatialization of the electronic sounds. The last addition to the software is a cue-based interaction system. Cues may be useful for a performer to organise long improvisations, or could be used to store presets and recall them whenever some specific sound environments are needed.

Figure 3: *Me vs Me? - Interactive drum performance*
https://youtu.be/taSawYaJ9YU

5. REFERENCES

[1] J. Cage, "The future of music: Credo," 1937.

[2] E. Varése and C. Wen-chung, "The liberation of sound," in *Perspective of New Music*, vol. 5, no. 1, 1966, pp. 11–19.

[3] K. Stockhausen and J. Kohl, "Electroacoustic performance practice," in *Perspectives of New Music*, vol. 34, no. 1, 1996, pp. 75–105.

[4] J. Chadabe, "Interactive composing: an overview," in *Computer Music Journal*, vol. 8, no. 1, 1984, pp. 22–27.

[5] G. E. Lewis, "Why do we want our computers to improvise?" https://www.youtube.com/watch?v=wDP8FsjyCaA, 2018.

[6] ——, "Too many notes: Computers, complexity and culture in *voyager*," in *Leonardo Music Journal*, vol. 10, 2000, pp. 33–39.

[7] P. F. Baisnee, J. B. Barrière, O. Koechin, and R. Rowe, "Real-time interaction between musicians and computer: Live performance utilisations of the 4x musical workstation," in *ICMC*, 1986, pp. 238–239.

[8] R. Rowe, "Feature classification and related response in a real-time interactive music system," in *ICMC*, 1990, pp. 202–204.

[9] P. Steinbeck, "George lewis's voyager," in *The Routledge Companion to Jazz Studies*. Routledge, 2019, pp. 271–270.

[10] A. Robertson and M. Plumbley, "B-keeper: a beat-tracker for live performance," in *NIME*, 2007, pp. 234–237.

[11] J. Gregorio, P. English, and K. E. Youngmoo, "Sound and interaction design of an augmented drum system," in *Audio Mostly*, no. 2, 2017.

[12] A. Guerri, "Agumenta - an interactive drumming experience," https://www.youtube.com/watch?time_continue=6&v=oQtuLBVuv64, 2015.

[13] C. Michalakos, "The augmented drum kit: an intuitive approach to live electronic percussion performance," in *ICMC*, Ljubljana, 2012, pp. 257–260.