Sound processors for live performance

Eduardo da Silva Afonso[1], Ruy Borges da Cunha Junior[1], Regis Rossi A. Faria[1]

[1]Laboratório de Acústica e Tecnologia Musical – (LATM),
Departamento de Música, Universidade de São Paulo (USP)
Ribeirão Preto –SP – Brazil

eduardo.afonso@usp.br; ruy.cunha@usp.br; regis@usp.br

Abstract. *We introduce a set of sound processing tools under development for live music applications using the Pure Data (Pd) patch-oriented framework. The development addresses aspects such as ease of interaction with the user and ease of tool integration in real-time applications. Three tools are presented on the current stage: a progressive filter, combining stop-band and band-pass types with time-varying partial tuning capability; a time-stretcher and pitch transposition tool based on granular synthesis; and an augmented interface to integrate a MIDI control surface with the patches in Pd. Several differential features are addressed in the current design, such as an individual tagging scheme to easy address control metadata all over the patches.*

1. Introduction

Filters and time-frequency modifiers are not novelties in the computer music scene, but means to explore their creation, their application to a sound, and its operation are always a typical arena for innovations.

Live composition and performance employing sound processors are often limited by exhausting procedures to install software, to apply and integrate them into a patch of processes, and to operate all the parameters so to adjust the behavior as desired. It is not rare to find difficulties in achieving the correct expected reaction following a calibration action. The need of adjustments on levels and ranges and the lack of resources to monitor the result of partial operations add more complexity and insecurity.

Aiming to approach interaction and integration issues of applying sound processors in live applications, we are exploring new avenues to design and to use sound processors modules on a patch-oriented framework. A first generation of tools produced in the current stage of this research project is presented in the following sections.

2. The sound processors

Three tools – two sound processors and an augmented control interface – were developed in the form of abstractions in the Pd (Pure Data) patch-oriented framework [1].

The first tool designed was so called a *progressive filter*, i.e., an implementation of band-pass and stop-band filters that can be connected in parallel or in series and can have their central frequency (*fc*) and tuning parameter (*Q*) vary in time and tuned to specific harmonic partials of a tone. What makes this implementation differ from others lies in two features: first, it is possible to immediately tune the filter over the fundamental frequency of a harmonic tone, access its first 10 partials on-the-fly and then sweep it, progressively transforming its spectral density; second, it is possible to address the filter parameters using a tagged message to it.

The progressive filter module combines in one block three types of filters which can be selected one at a time per instantiated object: a 1st order band-pass filter; a 2nd order band-pass filter; or a 2nd order stop-band filter. A slider controls the central (or stop) frequency (f_c) and another controls the quality factor level (Q). It also includes a fundamental (f_0) and partial frequency selector (up to 10 partials) to alternatively determine a frequency around which the filter shall operate, a RMS level indicator (VU) and a gain controller, plus a toggle button to turn on/off. These auxiliary resources add monitoring capability to the module, extremely useful for live situations.

Figure 1 shows the processor band-pass-filtering a white noise signal centered on the 4th partial (800 Hz) of a tone of fundamental frequency 200 Hz with a Q=10. A log-spectrum window at right shows the filtering profile resulting.

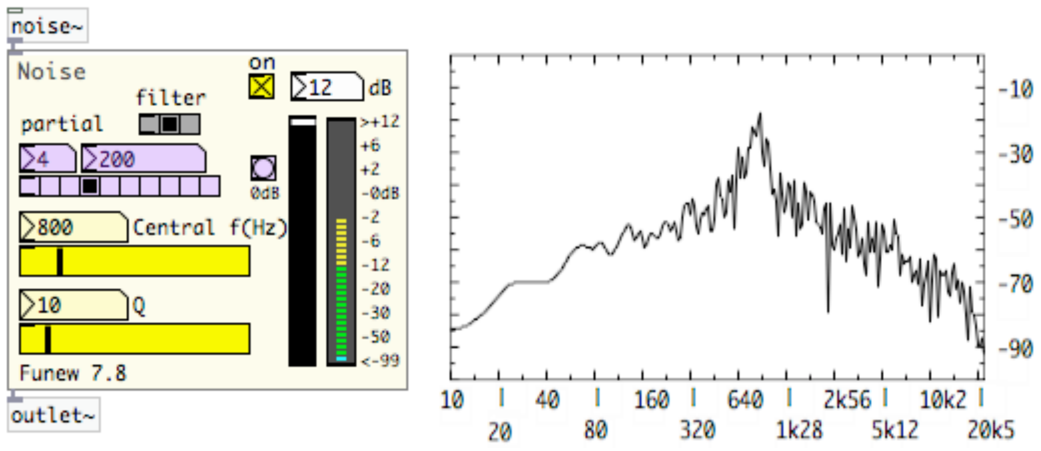

Figure 1: Progressive filter over a white noise, an a log-spectrum example

A differential feature in this tool is the possibility of having metadata generators control the filter behavior by sending commands to modify its parameters, either directly seeding its inlet ports or by sending messages to one of its parameter. For example one can have a pre-programmed table driving the fundamental frequency f_0 and another changing the Q factor of an individual instantiation filtering a specific sound.

The second tool developed is a *time-pitch processor* for changing the duration and/or the pitch of sound objects, a case of time-frequency stretching tool. This type of processor is already available in various programs and devices for sound processing [3] [4], but particular operating requirements and features are very frequent, demanding customized implementations.

The original concept can be extended in many ways and there are several forms to implement the effect. In this research, the theory of granular synthesis was the theoretical basis for its construction [5]. According to this theory, the signal can be enveloped in grains, which define portions of the signal to be sequenced in time in a determined speed, without changing its pitch.

The current abstraction was built inspired by reference implementations found in the literature and on the web [6]. Major additions and modifications were done however, including operations in the grain formation algorithm, modularity, controllability and pitch transposition capability.

Figure 2 shows the time-pitch processor interface. One can stretch or shorten the duration of the signal and independently alter its pitch. Two sliders, number boxes and preset bars are provided to define levels for stretching and for pitch transposition in standard tempered scale intervals. The sound is loaded into an internal array, shown on left up. The grain size, shape and generative parameters can be customized, including through manipulation of the grain envelope, by direct editing its curve over the array.

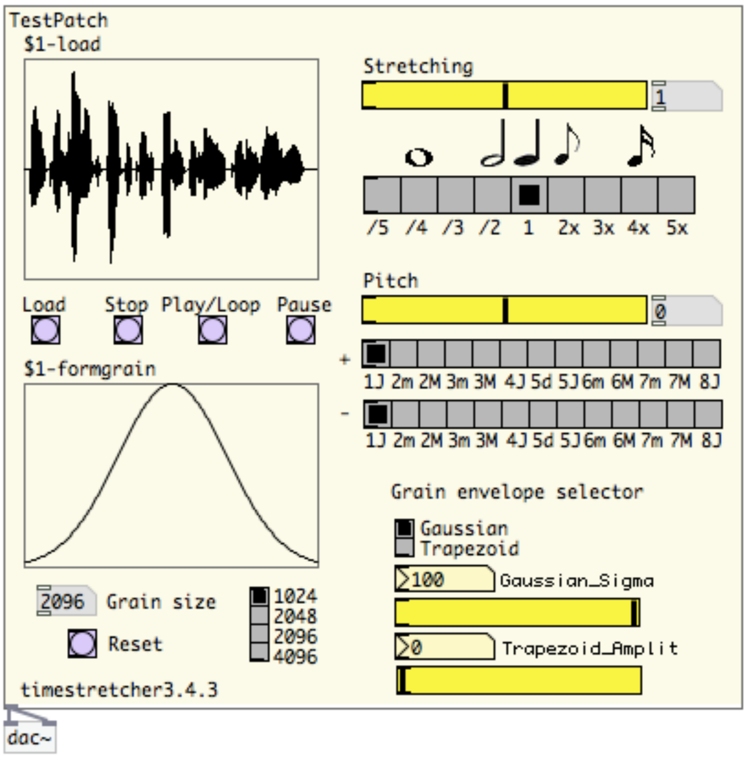

Figure 2: The time-stretcher and pitch transposition tool

The third tool developed aims to augment the addressing capabilities of an existing MIDI controller: a KORG nanoKontrol2 [2]. Its interface, shown on figure 3, permits not only to monitor and alter the values of buttons and sliders but it also extends the functions available in the original hardware, by permitting to assign each column (or channel) buttons and sliders to control addressable parameters of any processor module existing in the patches.

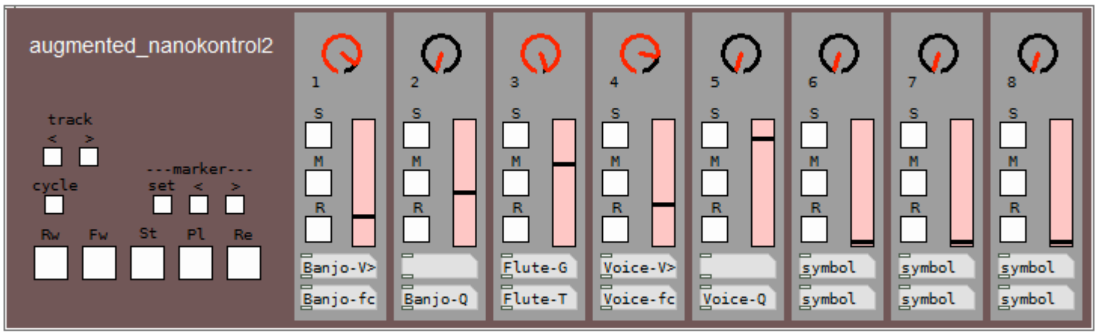

Figure 3: Augmented nanoKontrol2 GUI

Knobs and sliders are easily addressable through *tags* named by the user. For example, one can assign the gain control of a progressive filter called *Banjo* ("Banjo-Vol") to the column 1 knob, and its central frequency ("Banjo-fc") to the column 1 slider by just typing the tags in appropriate symbol fields in the GUI.

3. Conclusions

The spectral and time-pitch processors plus the augmented controller interface are the initial building blocks towards a more comprehensive library of tools for live music processing under development. We are particularly interested in investigating the processes of producing and adapting software to specific demands, mostly related to the creation and shaping of musical sounds in real time interactive applications, with a focus on usability and controllability.

Several features presented throughout this article make this development flexible enough to meet the needs of a composer building a real-time generative music application, and controlling it using a patch-oriented framework. These include a standard design to permit individual processors instantiation, integration, and a unique tag assignment scheme for addressing their individual parameters all over the patches.

In our tests we also found that many issues can affect the quality of the time-stretcher, such as the grain shape, size, inter-grain distance, and phase synchrony, which can for example introduce graininess in extreme stretching conditions. These issues are under investigation now and are to drive improvements for a second generation of tools.

4. Acknowledgments

The authors thank CNPq Brazilian National Research Council (process no. 134887/2012-3) and Dean’s Office for Undergraduate Studies of the University of São Paulo for supporting this research through their respective institutional programs of Undergraduate Research (PIBIC) and Teaching with Research.

5. References

[1] Puckette, M. Pd documentation, 2012. In: http://crca.ucsd.edu/~msp/. Access on 31 Dec. 2012.

[2] KORG NanoKontrol2. In: http://www.korg. com/nanoseries2/. Access on 31 Dec. 2012.

[3] Jaroszewicz, M. Spectral Tools. In: http://dl.dropbox.com/u/25820520/Spectral ToolsSource.zip. Access on 31 Dec. 2012.

[4] Charles, Jean-François. A tutorial on Spectral Sound Processing using MAX/MSP and Jitter. Computer Music Journal, 32:3, p.87-102, Fall 2008.

[5] Roads, C. Introduction to Granular Synthesis. Computer Music Journal, Vol.12, No.2, p.11-13, 1988.

[6] PURE DATA forum~ / Time Stretching. “timestretch_mmb_bd”. In: http://puredata.hurleur. com/sujet-2779-1.html. Access on 31 Dec. 2012.