BEGINNER'S MIND: AN ENVIRONMENT FOR SONIC IMPROVISATION

Thomas Ciufo

Arts, Media, and Engineering
Arizona State University
tc@ciufo.org

ABSTRACT

Beginner's Mind is the newest work in the ongoing Sonic Improvisation Series. This performance system is a combination of software and hardware designed for real-time sonic exploration. This paper will describe the primary design strategies employed in this system, and discuss key aesthetic concerns.

1. INTRODUCTION

Unlike some of my other performance systems, [1, 2] *Beginner's Mind* is not based on any particular instrument or sound source. A primary design goal was to build a system that would be pliable enough to accommodate a wide range of unpredictable input sources, and work in a variety of performance contexts, including solo or ensemble improvisations. To meet these goals, the software design is highly modular and can be extended or reconfigured easily.

Several specifically iterative processes are used, but the signal routing design allows all processing nodes to connect to each other, making the entire system capable of unlimited signal routing and recursion. There is a system-wide buffering structure capable of recording any sound that comes into or out of the system during a performance. This sonic history can be accessed for playback at any point in time, and can be routed to any or all processing modules for additional real-time manipulation.

An extensive real-time audio stream analysis environment is implemented that analyzes the sound coming into and out of the system, and tracks more than twenty different features. The audio stream analysis system is also used to calculate a composite perceptual identity over the course of any phrase or segment. The perceptual identity measures are stored, and can be used for real-time, untrained timbre matching.

The software system is driven by a combination of audio analysis, generative data, and explicit control from foot pedals and a hardware control surface. The software makes use of specially designed high-level control structures, such as a signal routing/mixing matrix that can interpolate between two 1024-point mixing templates using a single continuous control value.

I am especially interested in balancing this complex signal routing and processing potential with a playable, intuitive control interface that supports multiple interaction/performance modes.

2. SIGNAL ROUTING / MIXING

Instead of building the signal processing network as a fixed configuration, this system uses a modular matrix mixing technique that is extremely flexible and powerful. All signal processing modules are connected to a two-dimensional signal matrix that can route any input to any output at any level. This enables dynamic and continuously variable signal routing and mixing, including serial, parallel, and tree structures with nesting, feedback, and adjustable delay times available at each node. With these delays, it is possible to achieve sequential signal transformation by setting progressively longer delay times at each connection point. In addition to enabling the processing network to be reconfigured on the fly, this modular matrix approach facilitates long-term system expansion and modification, as modules can be added or substituted without disrupting the signal architecture.

The current signal network supports thirty-two inputs and outputs, for a total of 1024 possible connection points, each with adjustable connection level. This is too much data to manage directly, so high-level control or navigation structures are necessary. Routing/mixing templates that describe the connection strength of each point can be designed and stored. In performance, these templates can be recalled in any order, with a single continuous control used to interpolate between two different states. It is also possible to use noise or other randomly generated data sets to fill the templates, creating complex and unexpected networks. This technique would normally cause the system to self-destruct because of the built-in feedback, but the interpolation feature makes it possible to delicately move between stable and unstable states, thereby exploring unknown territory without completely crashing the system. This ability to reconfigure the entire signal architecture either abruptly or through gradual interpolation is a unique and powerful feature of this system.

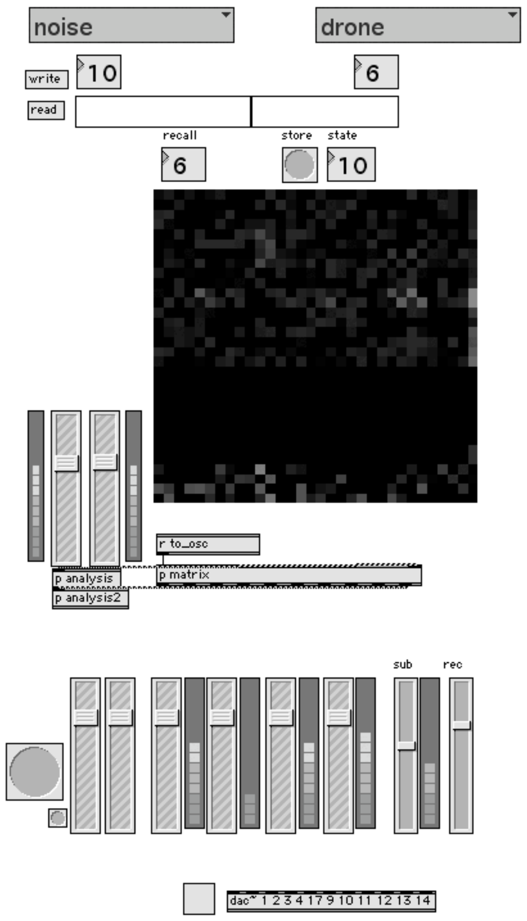

Figure 1. Matrix routing / mixing (pixel brightness equals signal connection strength)

3. PROCESSING MODULES

Within this dynamic signal network, a wide range of timbral and temporal processing modules are available. Describing the functionality of individual modules is somewhat misleading because the system output is completely dependent on the interconnections between processing modules, and the status of their control parameters at any given moment. Current modules include two types of timbre/frequency shifters, one based on ring modulation, and one that offsets or shifts FFT bins between the analysis and resynthesis stage. Signal decimation is available through bit depth and sampling rate reduction, as well as simple signal clipping. Two granular-based time stretchers are available with adjustable rate, window and location controls. Two short-term delay line scrubbers and a stereo granulator, both based on PeRColate objects [3] are implemented with adjustable delay times, scrub rates, and a variety of granulation controls.

One of the more playable modules is a resynthesis instrument based on resonant filtering. The current input signal is analyzed, with the frequencies and amplitudes of the eight most prominent spectral peaks used to control the amplitude and frequency of a bank of resonant band-pass filters. This filter bank is fed a mix of the input signal and a noise generator. The filter frequencies are only updated when an attack is sensed, creating a kind of time-displaced resonant shadow of the input sound. The bandwidth of the filters is continuously variable, transforming the sound from a noisy, broadband spectrum to a more peaky, narrowband timbre.

4. MEMORY MODULES / ARCHITECTURE

There are several processing modules that are designed specifically with recursion or memory functions in mind. This distinction is somewhat vague because all modules have adjustable output delays that can be combined with signal routing feedback to make any module recursive. Modules that incorporate internal storage memory include spectral loopers, feedback drones, and a system wide recording and playback environment. The spectral loopers are similar to those used in the *Eighth Nerve* guitar system [2] except the 512 band filters are located after the buffering structure, which means that the filters can continue to transform the spectral loops as they playback. Controls for each of the two independent loopers include record start/stop, playback rate, loop size, frame smoothing, buffer clearing, and dynamic filter interpolation. My interest in simple circular recording and playback functions surfaces again in the form of a constant feedback drone module. This processor is always in record mode and continues to additively recycle any sound that it receives. There are two output voices available for this module with independently adjustable feedback and transposition/playback rate controls.

In addition to the many short and medium length buffer functions available, this system also implements a high-level memory or sonic history function that can record any signal for any length of time. This is accomplished with a recording module that is given enough memory to record for the estimated maximum duration of a given improvisation. Recording can be started and stopped at any time, either autonomously using event segmentation analysis, or under manual control using a foot switch. The signal to be recorded can be accessed at any point in the mixing/routing matrix so signal input, output, or any point in between is available. It is interesting to capture the unprocessed input signal, which can be played back later, routed through a different processing network than when it was first heard. This allows a specific sound element to be transformed or reconstructed each time it is reintroduced. By capturing and reprocessing the output stream, a remarkable level of structural recursion and transformation is possible. New recordings do not overwrite previous recordings, so any sonic element may be recalled at any point during the performance, with variable rate four-voice polyphonic playback available. These carefully designed memory structures, combined with dynamic signal routing and mixing makes this performance system capable of behaviors that are complex yet pliable, unpredictable but formally cohesive.

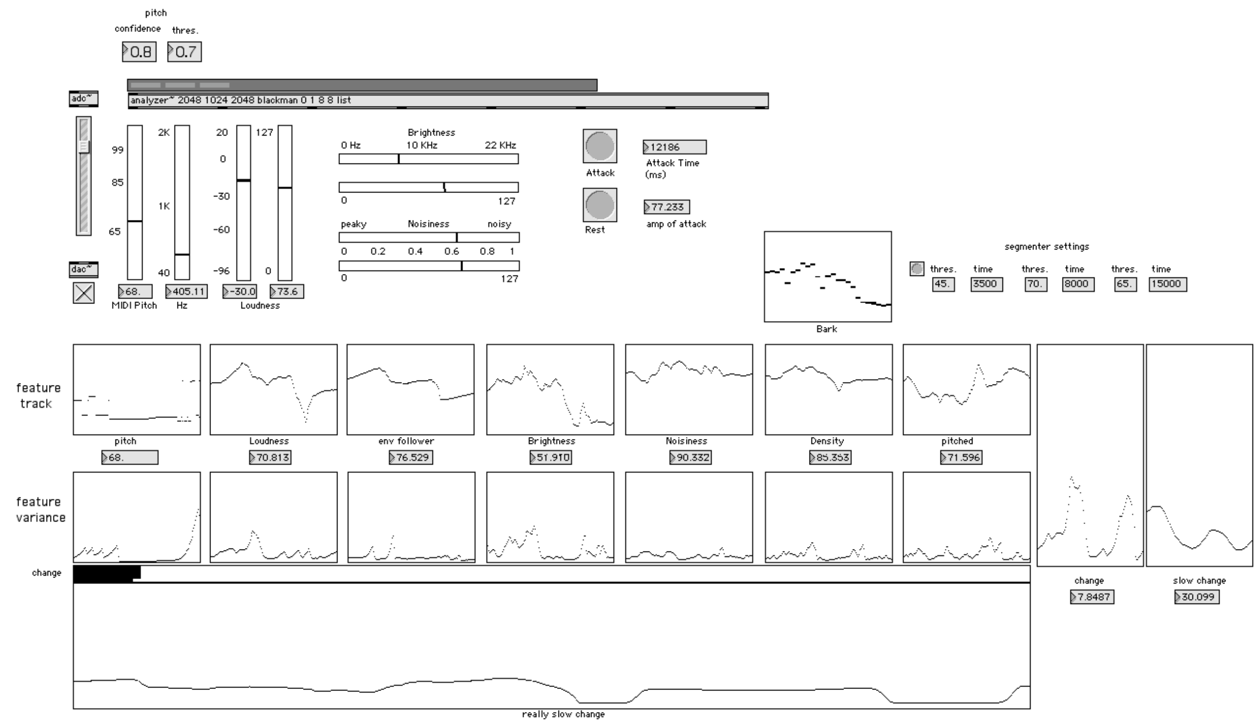

Figure 2. Audio stream analysis display

5. AUDIO STREAM ANALYSIS

This software features the most comprehensive and developed audio analysis implementation of any of my performance systems. This system uses (in addition to other techniques) the analyzer~ object [4] and the yin~ object [5] running in Max/MSP. Two complete analysis modules are utilized; one that listens to the input, and one that listens to the system output. This is especially beneficial because a single discrete sound element fed to the input may continue to resonate through the system for a long period of time. By placing an analysis module at both the input and output, a more comprehensive understanding of the total system behavior is available. Each analysis module tracks a number of features including pitch, (in both note numbers and Hz) loudness, amplitude envelope (with adjustable attack and decay), brightness, noisiness, Bark scale density, and pitch analysis confidence. Attack transients are analyzed along with the time between attacks, and the instantaneous loudness at the time that each attack occurs. Rest detection is based on a logic relationship between current loudness and time since the last attack. These parameters are adjustable, but an example would read; if loudness is below –50 and the time since the last attack is more than four seconds, then report a rest. Event segmentation and feature variance measurements are implemented, and operate on multiple time scales to reflect short, medium, and long-term activity. The frequency and amplitude of the eight most prominent spectral peaks is output, and is used to drive a resonant filter-based noise resynthesis module.

6. PERCEPTUAL IDENTITY

An exciting breakthrough with this software is the implementation of a phrase analysis and perceptual feature matching system, inspired by Wold [6] and others. As performed sound elements are recorded into a master storage module, they are simultaneously analyzed to create a ten-value perceptual identity. This identity consists of the calculated mean and standard deviation of the pitch, loudness, brightness, noisiness, and pitch stability analyses, taken over the duration of the phrase. This ten-value identity is calculated and stored each time a new phrase is performed and recorded. As an improvisation progresses, this growing collection of performed audio elements can become quite large, totaling fifty or more separate phrases. By running the same mean and standard deviation calculations on the perceptual features of the audio input, it is possible to make comparison between the current sound, and the collection of previously performed sounds. Stored sounds can be recalled based on their perceptual relationship to the current sound. An example would be to search the stored sounds for the one that is closest to, or furthest away from the current sound, across all perceptual features. This system is working reasonably well, and can usually match the instrument or sound type, as well as differentiate performance styles within a given sound subcategory.

7. SYSTEM INTEGRATION

Because of the modularity, complexity, and high number of controllable parameters available with this system, developing navigation and interaction strategies has been quite challenging. In an attempt to address the concerns of a variety of performance settings, I am designing this system with multiple control/interaction modes in mind. As discussed elsewhere [2] the relationship between autonomous, manual, automated, and analysis-based controls will greatly influence the behavior of a performance system, with certain relationships proving more appropriate to certain types of interaction.

For solo improvisations, I am more interested in autonomous and unpredictable system behavior, while in a group setting, a higher degree of playability is usually advantageous. If I use a system to process another performers sound, a high level of direct control may be most useful. In the past, I have designed instruments and performance systems for use in one primary context with the potential for overlap into adjacent settings. This software system appears to have the depth and flexibility to make it usable in a variety of settings, so multiple control modes seem to be worthwhile. To facilitate these multiple interaction strategies, I have centralized all control functions into one location within the complex software environment. Several different control modules can be connected to each parameter, and it is possible to switch between these control options with global mode messages. The mode setting can be decided at the start of a performance, or changed spontaneously while the system is in use.

The solo improvisation mode relies primarily on a combination of audio analysis based data, random and generative functions, and high-level meta-controls. Because this system is designed for use with a variety of sound sources or instruments, sensor-based manual controls are impractical. The primary physical control interface is a simple set of foot switches and two continuous foot pedals. The switches can be used to initiate events, such as recording or playback start/stop, or select specific mixing/routing templates.

In the current setup, one continuous pedal is used to control the interpolation between selected mixing/routing templates, and the other controls a specific continuous parameter that changes based on the status of the foot switches or other system conditions. For example, when the foot switch for the master recording module is pressed, the secondary continuous pedal is mapped to a mixing module that crossfades between the input and output signal. This type of interdependent controller remapping makes it possible to access many different parameters with only a few simple hardware controls.

High-level audio analysis based controls are used extensively in the solo performance mode. The activity analysis and event segmenters can be used to autonomously initiate various control functions including starting and stopping recording modules, triggering randomly selected buffer playback, and clearing system memory at major event junctions. The lower-level audio analysis data is used primarily to animate micro-controls on various modules, such as frequency shift offset, delay scrub speed, time stretch window size, and multiple granulation parameters. This creates responsive and ever changing sound transformations, even if only a single process is being utilized.

In addition to the solo performance mode, other control models are under development, including a fully autonomous mode where the only input to the system is the microphone. In this mode, all control functions are driven by input analysis, generative data, and logic routines without manual or explicit user control. On the other extreme, I am designing a control interface in which nearly every parameter is brought up on a hardware control surface. This design will be useful for group settings when I am processing the sounds made by other performers and need a more direct interface. Perhaps the most compelling aspect of this complex design is the possibility of moving into and out of the various modes in real-time.

8. CONCLUSION

It is still too early to make final pronouncements regarding this system. Many more performances across a range of contexts will be necessary before I can understand this complex instrument/system. I am excited by the possibility of exploring a range of interaction strategies and sonic spaces from within a highly flexible, modular framework. My hope is that this approach will allow more time to be directed towards exploring interaction strategies, instrument design, and performance practice, with less time devoted to low-level system construction.

9. REFERENCES

[1] Ciufo, T. Three Meditations for Prepared Piano and Computer. http://www.ciufo.org/research.html

[2] Ciufo, T. “Design Concepts and Control Strategies for Interactive Improvisational Music Systems”. In *Proceedings of the MAXIS International Festival / Symposium of Sound and Experimental Music*, Leeds, UK, 2003.

[3] Trueman, D. and L. DuBois. Percolate Software. http://www.music.columbia.edu/PeRColate

[4] Jehan, T. Perceptual Synthesis Engine: An Audio-Driven Timbre Generator. Masters Thesis. Massachusetts Institute of Technology, 2001.

[5] Cheveigne´, A. d. and H. Kawahara. “Yin, a Fundamental Frequency Estimator for Speech and Music”. *Journal of the Acoustical Society of America*, 2002.

[6] Wold, E., T. Blum, et al. “Content-Based Classification, Search, and Retrieval of Audio”. IEEE Multimedia, 1996.