A SYSTEM FOR MUSICAL IMPROVISATION COMBINING SONIC GESTURE RECOGNITION AND GENETIC ALGORITHMS

Doug Van Nort, Jonas Braasch, Pauline Oliveros
Rensselaer Polytechnic Institute
{vannod2,braasj,olivep}@rpi.edu

ABSTRACT

This paper describes a novel system that combines machine listening with evolutionary algorithms. The focus is on free improvisation, wherein the interaction between player, sound recognition and the evolutionary process provides an overall framework that guides the improvisation. The project is also distinguished by the close attention paid to the nature of the sound features, and the influence of their dynamics on the resultant sound output. The particular features for sound analysis were chosen in order to focus on timbral and textural sound elements, while the notion of "sonic gesture" is used as a framework for the note-level recognition of performer's sound output, using a Hidden Markov Model based approach. The paper discusses the design of the system, the underlying musical philosophy that led to its construction as well as the boundary between system and composition, citing a recent composition as an example application.

1 INTRODUCTION

In the context of free improvisation, the language that performers speak to one another and to the audience is developed throughout the course of a performance as well as rehearsal, listening to all facets of the sound that each player produces. Timbral and textural sound features become strong indicators of the musical form, and further it is the *shape* and direction of these qualities through which performer's speak to one another, expressing their *intention* for the future as much as their creation of the present moment or reaction to the past.

With this in mind, we have developed an interactive system for musical improvisation that analyzes the sonic content of performers, recognizes the nature of the sonic contours being produced in real time, and uses this information to drive a genetic algorithm. The output of this algorithm may be mapped to sonic or visual processes, creating a feedback loop and interplay between performer, machine recognition and a directed evolutionary process.

This particular choice of system design has arisen from our personal experience as improvisers performing together and with various other musicians, and observations made on this mode of musical creation in general. In terms of design constraints or what one might call "demands" of the system, we built around the following features

1. *A focus on timbral as well as textural information*.
The former is clearly a strong building block for improvisers defining their own performance language in concert. The latter is more separable in time than timbre, and in more sound-focused musics such as free improvisation becomes a strong structural element that interplays with larger sonic contours.

2. *Sonic gestural undestanding*.
We define note-to-phrase level sonic shapes as "sonic gestures", and feel that these convey the fundamental sense of musical intention and direction in improvised music. In this way, the refinement of the system focuses on the interplay between the system's response to this "immediate" information and the nature of the output, rather than building recognition of large-scale structural information that is less important in this context. In other words, the focus is shifted in our system to the immediacy of sound awareness.

3. *Continuous, on-line recognition with measure of certainty/uncertainty*.
While many systems exists for recognition of musical timbre, often the interest lies in the out-of-time acts of classifying musical notes, excerpts, passages or pieces in a way that is categorical. In contrast, our work builds an understanding or likely scenario of the type of sonic gesture that is being played, with a continuous degree of certainty about this understanding. In a sense, we are less interested in a musical retrieval than using the *process* of musical retrieval in a way that an improvising performer does, continually updating their expectation.

4. *Novel output from the system that is continuously influenced by performer's sonic gestures*.
Our goal was a system that would continuously produce spontaneous, novel, and what one might call "creative" events at the same relatively low level on which we focus for analysis and recognition. Therefore we explored the use of pro-

THIS WORK WAS SUPPORTED BY GRANT NUMBER 0757454 FROM THE NATIONAL SCIENCE FOUNDATION (NSF).

SMC 2009, July 23-25, Porto, Portugal

cesses that were directed while being under the influence of randomness.

With these design constraints in mind, we have arrived at a system in which continuous recognition of textural and timbrally-focused sonic gestures are recognized with varying degree of probability and confidence. This understanding then directs an evolutionary process that is finally mapped to an appropriately-defined system output. The sound analysis, gestural recognition and search process together are considered as an *agent* that reacts to the musical situation, influencing it by some output which is treated as an application of the agent rather than an inherent part of it.

2 RELATED WORK

There are several well-known examples of interactive systems for improvisation. One of the most prominent and musically successful is George Lewis' Voyager system [8], which converts pitch to MIDI data in order for various internal processes to make decisions depending on the harmonic, melodic and rhythmic content before producing symbolic output to interact with the human performer. Similar work can be see in Robert Rowe's Cypher system [11], which explores musical cognition and theory to form structuring principles based again on analysis of MIDI data. There exist other examples of MIDI/symbolic music content analysis systems, and the reader is directed to [11] as one point of reference.

As noted, our current system differs in that we examine continuous signal-level timbral/textural features to drive system output so that the system adapts to the changing audio content – as in [7] or [9] – with added layers of complexity from sonic gesture recognition and evolutionary processes. Another approach to analysis of sonic gestures was taken in [6] in which parameter curves for pitch, loudness, noisiness, roughness, etc. were extracted, with captured sequences being stored in a database in order to drive synthesized gestures having similar timbral contours. In this way the system is similar to the musical gesture-driven processing of [10], with the added layer that out-of-time gestural inflection drives the system rather than direct online parameters. Our system shares the conviction that sonic gestures are important in human-machine interaction for improvisation. However we differ in that the recognition itself is on-line with our system, and continual adaptation to the *anticipated* gesture is used as an element of the machine intelligence.

Finally, while the evolutionary paradigm has been widely used for algorithmic composition and sound design, there have also been several approaches to using evolutionary algorithms in an improvisational context. Biles' Genjam system [2] used an interactive genetic algorithm (GA) to evolve a system that learns to play jazz music along with a solo human player. The goal is to use the GA as a means to evolve the final state, while our interest is in the GA *process itself* as engaging with performer in improvisation. In a similar spirit is the work of [3], in which pitch values become centers of attraction for a swarm intelligence algorithm, producing a melodic stream that moves about these input values. Our system shares the interest of mutual influence between evolutionary/biological process and performer's sound output, while we focus on dynamics of recognition as an added layer to help guide this process.

3 SYSTEM OVERVIEW

The overall system, depicted in Figure 1, was written in Max/MSP utilizing several custom externals as well as the FTM, Gabor and MnM packages from IRCAM. In the first step, the system continually extracts spectral and temporal sound features. At the same time, onsets and offsets are tracked on a filtered version of the signal, which act as discrete cues for the system to begin recognizing sonic gestures. When such a cue is received, a set of parallel Hidden Markov Model (HMM) based gesture recognizers follow the audio, with the specific number of these being chosen as a product of needed resolution as well as processing power. The recognition continually provides a vector of probabilities relative to a "dictionary" of reference gestures. Processing on this vector extracts features related to maximum likelihood and confidence, and this information drives the fitness, crossover, mutation and evolution rate of a GA process acting on the parameter output space.

3.1 Sound Feature Analysis

The goal of the system is not to recognize a given sound quality absolutely, but rather to differentiate between sounds made by a performer along a continuum in several dimensions. In particular we believe that in the free improvisation context that is our focus, that global spectral features related to timbre are important for an immediate parsing of sound, with further qualitative differences coming from textures that are more separable in time and acting over a larger time scale (e.g. less than 20ms vs. 20-1000ms). Similarly, rather than the specificity of pitch values, the relative strength or "pitchness" of a note becomes important, as well as its register. In light of this we employed features that can be broken down into *global spectral, pitch strength and textural*.

For the first group we extract *Spectral Centroid* and *Spectral Deviation*. The first is a commonly-used feature that has proven to be a strong perceptual correlate of timbral brightness in distinguishing between sounds, while the second provides a useful means of differentiating between spectrally dense or sparse sounds. Deviation is calculated from the second order central moment of the power spectrum.

For the pitch strength features we use the robust Yin model [4], extracting *Frequency*, *Energy*, *Periodicity* and *AC Ratio*.

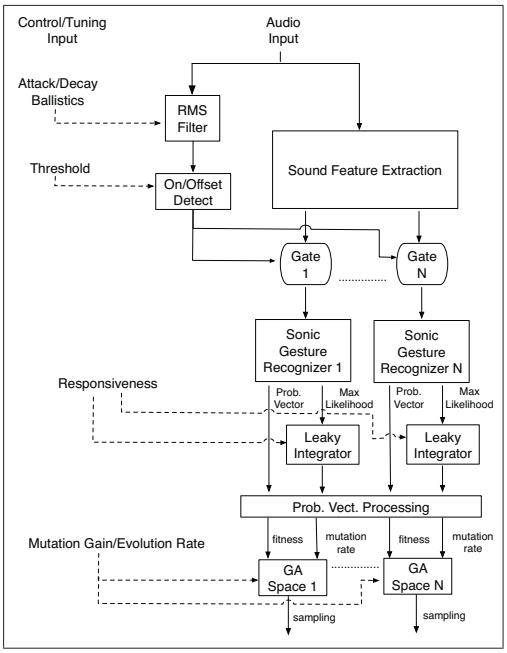

Figure 1. System Overview including feature extraction, recognition and mapping to GA process. The input parameters are used to tune the temporal response of system, and can be used to update this in real-time.

Periodicity provides a degree of "pitchiness", used as a measure of the confidence in the pitch estimate, while AC ratio is a qualitatively different measure of regularity, coming from the ratio of the first two autocorrelation coefficients.

The textural quality of the gestures is examined using the *3rd and 4th Order Moments of LPC Residual*. As was discussed in [5], higher order moments from the excitation describe jitter properties of this signal, which relate to nonlinear frequency modulations of sustained partials and so to textural phenomena. Therefore we extract the residual value by way of LPC analysis, and compute 3rd and 4th order moments on these values in order to differentiate between disparate musical textures. We have found that this measure is very useful in separating voiced and unvoiced content.

3.2 Onset/Offset Detection

Our system uses onsets and offsets as cues to indicate that a relevant event may have begun/ended, with the final decision of whether an event is relevant being determined in the recognition stage. Rather than model onset detection for particular types of events, we employ a straightforward approach that uses tuning parameters for thresholds and response time. Specifically, we utilize the levelmeter object from Max/MSP - which models a VU-style meter - to produce an RMS value of the input sound. The purpose of using this object is that allows for tuning the ballistics of attack/decay times, which strongly influences the onset detection. After extracting the smoothed value, the difference of successive values is taken, and if this difference is greater than a given threshold an onset is considered to have occurred. The same is done in the opposite direction with offset detection. This is represented in Figure 1 as the first two stages of the left-most signal path. When an onset has been detected, it opens up a gate which causes the system to begin searching for sonic gestures that it may recognize from the audio stream. The way that the individual gates close depends either on an offset cue or on other factors related to the recognition as we will discuss in section 3.3.4.

3.3 Sonic Gesture Recognition

Our sonic gesture recognizer is built on the efficient gesture-follower [1] modules from the MnM library developed at IRCAM. This implementation uses an HMM and dynamic time warping to follow as well as recognize gestures in real-time. While there are some trade-offs made with this implementation for efficiency and to allow use with a low number of training examples, we have found it to work well in light of our requirements 2 and 3 as stated in the introduction. These modules work on any data that one can represent in matrix form, leading us to adapt them for our sonic recognition stage. We use all eight of the employed sound features as a singular multidimensional gestural representation, producing a vector that will represent one state in the underlying HMM model, defined using a left-to-right state topology as is standard in applications such as speech recognition.

3.3.1 Gestural Dictionary

The gestural follower requires training examples as a basis for future comparisons. Our interest is not to provide exemplary gestures that performers must later try to recreate - and in this sense perhaps our approach is an outlier for HMM-based recognition. Rather, our goal is to populate a space of gestures that represent a general playing style, in disparate parts of "gesture space". That is, these sonic gestures should be orthogonal in some musical sense, and this is regarded as part of the composition for the system. From experience of using the follower implementation, however, two important considerations arise: the gestures should be roughly the same length and there is a complexity limit (total of number of states in the database) beyond which adding new gestures makes recognition impossible.

3.3.2 Continuous, Dynamic Attention

After the recognizer is trained on a set of gestures, it is ready to accept vectors of the same type for comparison, providing a probability for each member of the gestural dictio-

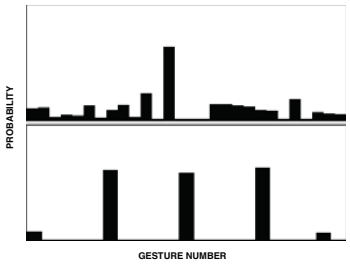

Figure 2. Output probability vectors. Top value shows strong certainty for one gesture while bottom shows confusion over three possible gestures.

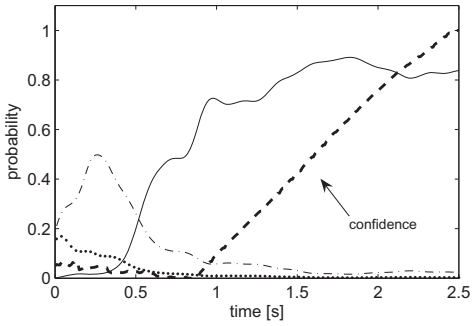

Figure 3. Probability dynamics for three most-likely gestures and related confidence level.

nary given the current input. In order to define the temporal boundaries of a gesture from the current input, the recognizer must be explicitly started and stopped, which we initiate with onsets. Most importantly, once a start message has been given the probabilities are updated in real-time for each member of the gesture space, providing a form of *dynamic attention*. This is important in the context of improvisation, where *one's expectation of what a sonic gesture is at any given moment is continually being updated*. We use this information to drive system output, thereby mapping this dynamic attention into action as an engaged improviser would.

3.3.3 Probability Dynamics Processing

From the raw probability values for each gesture, we extract the normalized probability, the maximally-likely gesture and the deviation between the maximum and the few highest values. These latter values each give some indication of how strongly the system believes that the performed gesture is one from the system. Both values are needed to know uniqueness as well as strength of recognition, as illustrated by Figure 2.

At the same time, instantaneous recognition values are not enough in order to usefully map the dynamics of recognition, as the HMM produces sudden changes in the probability vector. For stable gestures this is normally a slow oscillation between perceived values, but occasionally the recognizer will change course abruptly. Therefore, a leaky integrator is applied to the extracted maximum $m_{k,i}$ and deviation values $d_{k,i}$ to create a confidence value defined as

$$C_{n,i} = \delta(m_{n,i} - m_{n-1,i}) \sum_{k=0}^{n} 2^{\frac{-1}{\lambda_{k,i}}} (m_{k,i} d_{k,i}).$$

This represents a building of confidence in a given gesture's likelihood over time n for gesture space i. If the maximum probability value changes abruptly between two members (i.e. if a strong "change of mind occurs") then the integrator is cleared by the binary δ function. Otherwise, the value decays smoothly as determined by the response time $\lambda_{n,i}$. Figure 3 illustrates a situation in which uncertain movement keeps the confidence value low until this subsides, when the confidence begins to build.

3.3.4 Parallel Gestural Spaces

Although the follower does use dynamic time warping in order to provide a best guess of the gestural scale, the implementation is limited by the need to have similarly-sized gestures in a given dictionary. Further, it is not trivial to track the beginning and ending of gestures along differing temporal scales in one analysis, as well as to make decisions on what is considered a meaningful gesture as exceptional players often embed one type of gesture within another. In order to examine these different levels of granularity, we create gestural spaces that act on different time scales in parallel, as noted in the diagram of Figure 1. The generality of the diagram reflects the fact that the number may vary depending on computing power and musical context, while we have found that using three different temporal scales has been adequate for our own purposes thus far.

As noted we use onsets as a way to cue the recognition process. When an onset is detected, recognition is triggered using the shortest database of sonic gestures. If the smoothed maximum value stays below a given threshold, then the recognition stop after μ_i seconds, which represents half of the average length of gestures from the ith set. Otherwise, recognition ends after M_i, the maximum time over all gestures in i. If no offset is detected, then the next $N-1$ levels of recognizer immediately begin searching their databases. If the accumulated confidence value $C_{n,i}$ for space i is not above a given threshold by μ_i, then the recognizer is re-set to the beginning. Otherwise it is reset when M_i is reached. For example, Figure 3 represents gestures from a database where the average gestures length is 5 seconds, and $\mu_i = 2.5$. In the initial portion from 0-1 seconds there is devia-

tion between the three main gestures so that confidence $C_{n,i}$ remains low. However it increases rapidly after one gesture asserts itself, easily passing any threshold the user may set. If μ_i were instead 1 second, then the system would likely be reset as confidence would be below any threshold value.

3.4 Gesture-Driven Evolutionary Process

While sonic gesture recognition is an important part of our system, as noted it is ultimately the *process* of recognition and understanding that is central to its musical behavior. The goal is to have a continuous interplay between this process and an output that guides performers in a feedback loop as they in-turn guide this system. We have utilized genetic algorithms as a goal-directed process that moves in a globally predictable direction while maintaining random elements on a local scale. Rather than set the goal a priori as one commonly would in a GA used for optimization purposes, the "goal" changes as a product of the system's gestural recognition and confidence.

The underlying parameter space for our GA implementation is tied to the size of the gestural spaces employed. As noted there is a limit to the size of each gestural space, which we have found to be between 20-30 depending on the time scale of the gestures. At the same time, the required population size for a GA implementation is a product of the problem complexity for optimization purposes. For our application to real-time improvisation, we have found spaces as small as 20 members to be effective in moving towards a perceptible goal.

The reason that we constrain the population size to that of the gestural space is that each member of the gene pool is treated as an ideal output that should arise when a given sonic gesture is believed to be present. Therefore, if a performer "plays into" a certain known gestural type, the system will strongly recognize this and move the GA towards an output that is intended for this type of playing. The way that we achieve this is by mapping the probability for each gesture in a dictionary into the fitness for the corresponding member of the GA population. Thus, for example, in figure 2 if the top probability vector were in steady-state, then the GA would converge towards the member associated with the highly-probable member located in the center of the image.

While belief in a particular gesture causes output to converge towards a particular parameter set, the dynamics of this convergence are determined by the confidence value. As the confidence raises, the probability of mutation (randomization of output parameters at crossover step) as well as the depth of mutation (degree of randomization) decrease. Taking the example from figure 3, the fitness value would oscillate as a function of the three probability curves while the mutation rate would be high due to the low confidence. This would cause the members selected for breeding to move into new areas of the parameter space. After the inflection point – when one gesture begins to dominate and the confidence level starts to rise – the space would move towards the highly-probable level, with less mutation applied at each new generation. The rate of each generation ("rate of evolution"") is context sensitive, being controlled by the confidence level for large-scale gestures or by onsets for small-scale gestures. The relative nature and dividing line of "small" vs. "large" gestures is a product of musical context. The reason for making this distinction is that we have found that short, attack-focused gestures that occur with higher frequency can evolve the space at a reasonable rate, and appear more musical as the change in output is tied to musical events. Longer-scale gestures do not drive the space at a fast enough rate. Further, improvisation and other sound-focused music tends to listen for internal developments "inside" a given sound gesture as it unfolds, so that evolution of system output should not be tied to the initial moment of attack.

As with the gesture-follower, the GA implementation is built up from abstractions written in Max/MSP, while the core GA itself is a C external programmed with operators that are unique to our application. This external is instantiated with messages for population and member size. Each member is a string of float values in the range 0-1. A list of fitness values may be input – one for each member – and a bang message causes a random sampling of members of the population (with selection probability proportional to fitness) for mating/crossover. A simple one-point crossover occurs between members at a random location in the parameter list. This GA implementation is categorical in that each member is tied to a particular member in the corresponding gestural dictionary. Therefore, rather than using a random replacement operator, care must be taken in order to replace the proper parent from a previous generation with its children, where the children inherit the fitness of the parent until the recognition assigns a new value.as

4 CASE STUDY: ACOUSTICS/ELECTRONICS TRIO

The premiere of our system in concert was in the context of a new piece written by the first author for the New York City Electroacoustic Music Festival (NYCEMF) [1] . The piece was written for saxophone, accordion and laptop performer. The electronics capture the sound of the acoustic performers in real-time and transform them in order to define new sonic gestures having their own timbre and texture. The software system is a granular feedback-delay system written by the first author as a performance tool, where input sound may be scrubbed (via gestural control), time-stretched and novel transformations applied through per-grain processing and feedback-delay coupled with larger-scale (e.g. 5-15 sec-

[1] http://www.nycemf.org/ - last accessed on June 8th, 2009.

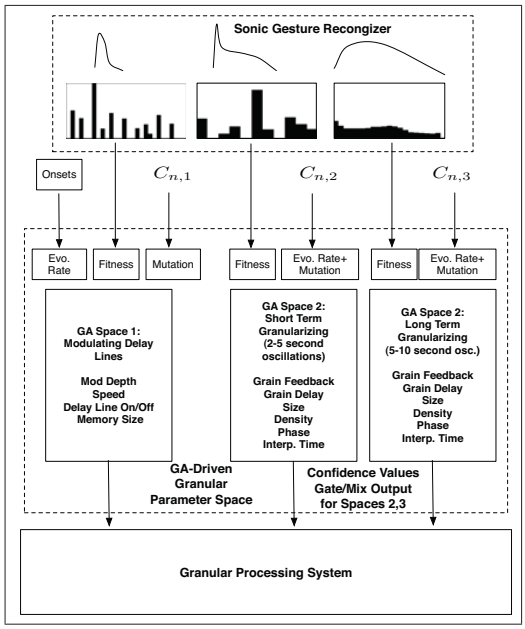

Figure 4. Gesture recognition to GA mapping. Compositional choices are reflected, such as onsets to drive space 1, and choice to mix or gate (depending on section) space 2 and 3 output depending on confidence values.

onds) delay modulations that return as independent gestures rather than transformations on previous material. The structure of the piece was centered around how the sonic gestures and sound processing should co-evolve over time, and how much influence the agent exerted over the human electronics performer. As such, "composing" took on several meanings.

The choice of sonic gestures with which to train the agent was one of the strongest compositional choices. This defined the central gestural types that the performer could then choose to "play into"" or to "play around". The system used 1, 5 and 10-second gestural spaces in parallel. The short gestures focused on a variety in regards to brightness and pitch material, while the 5-second gestures defined different textural values in terms of voiced vs. unvoiced qualities and rough vs. smooth tones. The 10-second gestures defined forms that one might call phrases: differentiating between fast, stunted patterns and slower drones, having different timbral qualities.

A second compositional choice was made in terms of the type of processing to map to each gesture space. The desire was for shorter gestures having sudden attack to lead to a variable number of output gestures that related timbrally to the input while having their own unique gestural character. This was achieved by mapping this smallest gestural space to sound parameters that controlled an array of modulating delay lines each with a unique modulation function, wherein the number of active delay lines, their modulation rate and depth, and memory size (i.e. how far back in time to look for input) were controlled by the agent. Meanwhile, medium and large-scale gestures were mapped into the granular parameters related to grain size, rate, inter-grain phasing, per-grain feedback gain and delay time as well as interpolation time between parameter changes. In this way, the extended gestures with slower attack could be scrubbed by the laptop performer or time-stretched automatically depending on the section, while the internal characteristics of the granular processing evolved at a rate that depended on the anticipated length of the gesture (i.e. whether driven from the 5 or 10 second gesture spaces). This application illustrates one of the great strengths of our system: that a particular gestural type can be mapped into a sound processing parameter set that is tailored to its dynamic sonic character. The nature of the sound processing can then be changed for different temporal scales of sonic gesture. Therefore, rather than content-based processing where the audio quality directly determines the transformation type (as in e.g. [10]), we have added the layer in which the processing type is determined by the audio feature content (indirectly) and type of gestural dynamics (directly) that the system *believes* is occurring, creating an appropriate interplay for improvisation.

5 REFERENCES

[1] F. Bevilacqua, F. Guédy, N. Schnell, E. Fléty, and N. Leroy. Wireless sensor interface and gesture-follower for music pedagogy. In *Proceedings of the 7th Int. Conf. on New interfaces for musical expression*, pages 124–129, 2007.

[2] J. A. Biles. Improvising with genetic algorithms: Genjam. In E. R. Miranda and J. A. Biles, editors, *Evolutionary Computer Music*, pages 137–169. Springer, 2007.

[3] T. Blackwell and M. Young. Self-organised music. *Organised Sound*, 9(2):123–136, 2004.

[4] A. de Cheveigné and H. Kawahara. Yin, a fundamental frequency estimator for speech and music. *J. Acoust. Soc. Am.*, 111:1917–1930, 2002.

[5] S. Dubnov, N. Tishby, and D. Cohen. Influence of frequency modulating jitter on higher order moments of sound residual with applications to synthesis and classification. In *Proc. 1996 Int. Comp. Music Conference (ICMC 96)*, pages 378–385, 1996.

[6] W. Hsu. Managing gesture and timbre for analysis and instrument control in an interactive environment. In *Proceedings of the 2006 Int. Conf. on New Interfaces for Musical Expression*, pages 376–379, 2006.

[7] T. Jehan and B. Schoner. An audio-driven perceptually meaningful timbre synthesizer. In *Proceedings of the 2001 International Computer Music Conference*, 2001.

[8] G. Lewis. Too many notes: Computers, complexity and culture in voyage. *Leonardo Music Journal*, 10:33–39, 2000.

[9] C. Lippe. A composition for clarinet and real-time signal processing: Using max on the ircam signal processing workstation. In *Proceedings of the 10th Italian Colloquium on Computer Music*, pages 428–432, 1993.

[10] E. Metois. Musical gestures and audio effects processing. In *Proc. of 1998 Int. Conf. on Digital Audio Efects (DAFx 98)*, 1998.

[11] R. Rowe. *Interactive Music Systems: Machine Listening and Composing*. The MIT Press, 1992.