Future Intelligence in Live Electroacoustic Music

Arne Eigenfeldt
Associate Professor
School for the Contemporary Arts,
Simon Fraser University,
Burnaby, BC Canada
arne_e@sfu.ca

1. Introduction

At the EMS 2007 symposium at Leicester, I presented a paper in which I discussed the history of live EA, and what I perceived to be some of the problems facing the genre [Eigenfeldt]. In this paper, I'd like to suggest what I believe to be some possible avenues of exploration that live EA might pursue in the coming years.

2. Defining Live EA

Firstly, what am I talking about when I use the term "live electroacoustic music"? Clearly, there are many different genres within live EA, but my approach will be somewhat specific. Many people involved in live EA have pursued it as a result of their own background in performance, and bring performance aspects to electroacoustic music. My own involvement has been from the perspective of a composer: this may seem a bit contradictory. Suffice to say, I am interested in creating situations in which live performers can interact with a computer, whose actions and responses are controlled, in part, by my compositional thinking.

This, therefore, limits what I will discuss regarding the potential future of live EA: I'm not going to discuss networked performance, EA instruments, hardware hacking, score-following, or live DSP, since my interest centres on works that use the computer, in performance, as an instrument, or ensemble.

But even the idea that the computer can be an instrument is somewhat misleading, since "instrument" usually implies single gestures: the computer is capable of multiple gestures at once, and so requires a different kind of thinking - compositional, if you like.

This specific form of live electroacoustic music used to be called "interactive computer music", which I think still serves my purpose. Firstly, it implies the use of the computer, rather than any new EA instruments or acoustic instruments played across networks. Secondly, it implies the notion of interactivity, which precludes the use of sequencer-like software - such as Ableton Live - for direct playback of material. Although one could stretch the notion of interactivity to include the direct signal processing of instruments playing predefined scores, for this paper, my definition of interactivity requires creative interaction by both computer and performer.

Implicitly, "interactive computer music" includes those pieces in which the computer is making decisions within performance. Pioneering works in this field included those by David Behrman, Salvatore Martirano, Joel Chadabe, Michael Waisvisz, Martin Barlett, Robert Rowe, and George Lewis. And while those earlier examples are quite clear in the use of the computer as a realtime performer/composer, each of these composers had to build their software from scratch. More recent works have continued this evolution, but use existing software, at least for starting points, such as MaxMSP, SuperCollider, and ChucK.

Due to the readily available software, more composers are pursuing this particular form of live EA. However, unlike those composers mentioned earlier, who are known for their near-exclusive focus on interactivity, contemporary composers, not just those in EA (let alone those focused upon interactive computer music) are creating works that use the computer instrumentally. These composers can be forgiven for not thinking about the future of this genre. I, however, spend almost all of my time designing and coding interactive software, so I've tended to think a lot about its future.

One caveat, however. A lot of what I'm about to describe is not my area of specialty: I'm a composer, not a computer scientist. I have been collaborating with a very patient CS researcher, who patiently explains the current research to me, so what I'm presenting is new research from the point of view of a composer in the field.

3. Why is there a need for intelligence?

The potential multiplicity of gestures available to the live laptopper has created a new problem: how to control the gestures, and more importantly, their evolution, in performance? I outlined in my earlier paper at EMS 2007, that constrained randomness has been one popular method of controlling complexity, and I also pointed out its limitations - the most important being a lack of awareness *by the procedures* over the material that they are modulating.

Some of the most impressive live EA that I've heard in recent years rivals the complexity of acousmatic music, which, clearly, is not composed using methods of constrained randomness. Now, I realize that not all composers of live EA are interested in this type of organized complexity - as opposed to the disorganized complexity that constrained randomness

will produce [Weaver] - some will happily purse a Cageian approach to improvisation and interaction. Joel Chadabe has suggested his own approach is akin to sailing a boat on a stormy lake, reacting to the immediate environmental changes. Thus, his software will present situations, generated through constrained randomness, to which he must react; the software will, in turn, react to these decisions, thus making the relationship interactive [Chadabe].

However, if we are attempting to emulate human decision-making - which is my specific desire - one cannot look to randomness; instead, we should attempt to emulate some notion of intelligence within our realtime software.

4. So what are we trying to do?

Musical intelligence is obviously a huge topic, one that is, and has been, actively researched [Wiggins and Smaill]. In order to simply our task somewhat, We can isolate some decisions that we might need to make as composers in a realtime performance situation.

Let's start with a simple case, of a laptop musician selecting soundfiles for performance. Let's assume that our laptopper has a series of folders that are filled with soundfiles, each related in some way; for example, "beats", "soundscapes", "synthetic noises", etc. After selecting some combination of these files for playback, at some point she will need to change sounds. What is the appropriate sound for that specific musical situation?

Looking at the menu of fifty sounds in "synthetic noises" might prove daunting in performance: for example, this requires her to remember exactly how "scratch 5" differed from "scratches 9".

5. MIR and sample organization

For this task, we can borrow methods of timbral organization used in the MIR (Music Information Retrieval) community. While the focus of this community is not on electroacoustic music - it seems to be focused at the moment on music recognition and recommendation - we can, as our community tends to do, dip into this commercial research and apply it to our own artistic needs.

One basic premise behind MIR is classification, and that's what our current task amounts to: classifying our timbal database. We probably already do this manually, organizing our samples into folders of "beats", "soundscapes", "synthetic noises", etc. However, this requires us to listen to each of the samples, and make subjective decisions based upon what we hear.

In order to use any sample in performance, it also requires us to remember each sample, and its particular characteristics. Instead, we can use software to make these decisions for us; for example, *Marsyas*, a popular analysis package in the MIR community [Tzanetakis and Cook], can be used to extract features from within our timbres (such as spectral centroid, spread, rolloff, and flux, as well as amplitude features, such as zero-crossing data and RMS). Once our sample library has been analysed, individual sounds can then be classified for similarity based upon a single feature, or combinations, of these features.

6. Self-Organizing Maps

One interesting method for organizing the feature analysis is the Self-Organizing Map [Kohonen] - also known as a SOM - which allows us to visualize our data in 2 dimensions. SOMs are a type of artificial neural network that use a neighborhood function so as to reflect feature proximity in a topological manner. In other words, similar features are located close to each other. The map itself is often a grid, with each node of the grid assigned three features values that are displayed as RGB colour values. For example, we could look for spectral evolution within our samples, and choose the standard deviation over time of the spectral centroid (brightness), energy (loudness), and flatness (noisiness): these would be translated into red, green, and blue values.

The example below (Fig. 1) displays a soundscape database using such a feature representation. The nodes that contain more red are associated with samples that have more dynamic frequency (spectral centroid) change; those squares that display more green have more variation in their spectral energy (loudness); those squares that display more blue have more dynamic change in their spectral flatness (noisiness). Notice how the different shades of colours, and therefore the related features, have organized themselves into close proximity.

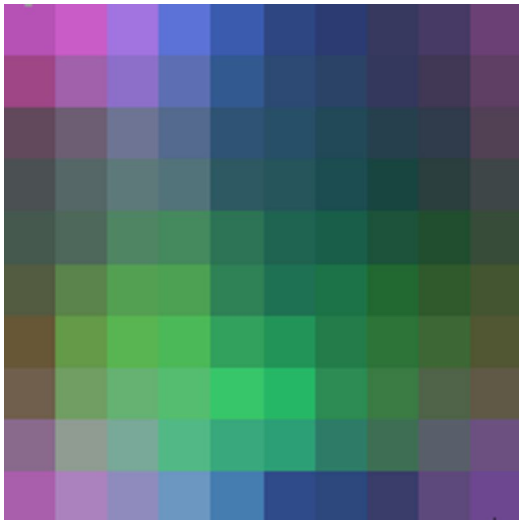

Figure 1. Resulting map from a soundscape database of 87 audiofiles.

The database would be mapped behind this image, and selecting a square by clicking on it would select members from the database that correspond to that classification.

Another example would be in selecting samples through frequency band similarity. For example, a large database of percussion samples can be analysed for energy in discrete frequency bands using a Bark analysis [Zwicker]. Bark analysis provides energy values for 24 frequency bands, which closely match our own critical bands; it thus can be thought of as a perceptual model.

In this example, the top three Bark bands of each sample, drawn from a database of over 1500 samples, were used as features for the SOM:

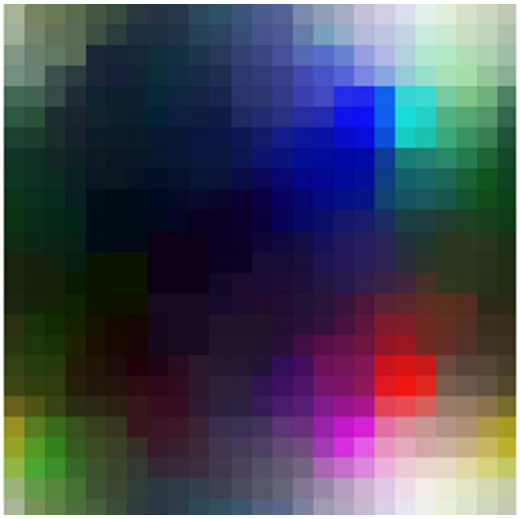

Figure 2. Resulting map from a percussion database of 1581 samples.

Dark colours represent those samples in which the dominant energy is low frequency; light colours represent those samples in which the dominant energies are high frequency. Thus, clicking on dark colours will select samples whose dominant frequency bands are all low frequency. Those that are closer to one primary colour represent samples whose Bark bands are spread out: for example, a sample with primarily low frequencies, but a single high component (i.e a drum with a metal rattle) will be bright red, blue, or green.

The power of the system is clicking *around* a given area, which will return samples that are similar to one another based upon their features. Thus, in performance, the user can make a meaningful musical decisions in terms of sample selection: does the situation requires more of the same (i.e. similarity) or dramatic change (i.e. dissimilarity)?

SOMs require training, which amounts to repeatedly feeding the network examples from the database: this takes time, so SOMs cannot be created in performance, although they can certainly be used in realtime.

7. Understanding performance data

Let's return to our laptop musician, who now wants to interact with a saxophonist, and both improvise together. This model has been a rather consistent one, perhaps because the notion of improvisation is so exciting, or because it allows a composer, such as myself, to be interacting in performance with a musician in ways that are not otherwise possible (my own limitation would be my lack of expertise on a musical instrument).

In the early decades of interactive computer music that utilized MIDI for input and output, the system's responses tended to focus upon the data that was readily available: melody (MIDI note data), harmony (collections of MIDI note data), and rhythm (inter-onset times between MIDI note data). Many systems were based around capturing MIDI data, and playing it back with some sort of alterations, such as transposition, modal change, time stretching/compression, etc.

Many of today's systems use similar strategies, albeit capturing audio data and playing it back with signal processing, such as filtering, delay, signal degradation, etc. In contemporary DSP-based systems, pitch and onsets are two of many features available for analysis.

Furthermore, communities such as NIME (New Instruments for Musical Expression) have focused a great deal of energy upon designing performance sensors: the analysis of this data can be used in addition to spectral, pitch, and loudness analysis. We can, again, look to the MIR community, for some strategies on how to understand this data.

8. Machine Learning

We have now entered into the field of machine learning. Machine learning has many, many applications, and has already been used in interactive computer music. Robert Rowe pioneered machine learning in interactive systems in the 1990s with Cypher [Rowe].

Two contemporary systems should also be mentioned. Francois's Pachet's Continuator is a realtime system that learns the improvisatory style of musicians as they play MIDI instruments [Pachet]. Continuator can continue the improvisation by performing in the style of the analyzed performer. It is a musical Markov system that first analyses musical input, produces a variable order Markov model [Ames] of this data, then uses it to search for probable continuations to new incoming musical information.

Similar processes are in use by Gerald Assayag's OMax [Assayag et al.]. Developed at IRCAM, this is a software environment that learns, in realtime, feature's of a musician's style, and plays along interactively. It is based, on imitation (using a variation of Markov models) and transformations, which are preset, and not learned. The system analyses the input it receives, and looks for transitions (where the fragments might be, and how they are combined) within it, then chooses randomly from these fragments.

Both of these systems began as MIDI incarnations, but now support direct audio input. The guts, however, use evolutions of the same principles explored by earlier interactive composers: record a live musician's performance, and play it back in an altered format; in both these systems, the playback is facilitated by Markov analysis so as to produce related, rather than direct, playback.

A limit to Markov-based systems, as pointed out by Chomsky in the 1960s and further explained in Ames, is their inability to deal with time, or hierarchy; longer order Markov chains will only reproduce the original, without context. And while their use produces initially interesting surface variations, they will eventually fall into an endless meandering that requires performer intervention.

9. Database Searching

One solution to this problem has been to use database beyond the immediate performance, something more akin to how musicians might interact with one another: extended memory. In such a case, rather than operating directly upon recorded material, the system could compare recent input to its existing database, and search for similar or related responses.

Of course, many algorithms for database searching exist, and the MIR community has successfully applied these to music. Furthermore, comparing musical features for similarity is something the MIR community excels at, since this is, of course, what drives it commercially. Many of its techniques are beyond the scope of this discussion, and, frankly, beyond my own understanding. Perusing any recent conference proceedings of the International Society for Music Information Retrieval [http://www.ismir.net/], the yearly conference for MIR, will produce a host of comparison algorithms.

There are many methods that show a great deal of promise for composers of interactive music, especially since these methods have now been optimized for music retrieval (which suits our own needs); they do require, however, a high level understanding of computer science.

10. Genetic Algorithms

One method that researchers are exploring is a combination of several methods, using the best of what each offers. For example, Genetic Algorithms offer the ability to generate many different solutions to a problem. GAs have been around for a while, and were first used in realtime performance in the 1990s in GenJam [Biles].

The problem with GAs and music has always been how to evaluate their evolution: what makes one individual musically better than another? One way to get around this is to avoid the subjective analysis of "better", and search for the objective goal of "closer".

In Gil Weinberg's Haile [Weinberg], a robot marimba-player that preceded his more recent Shimon, the GA's population are transcriptions of a jazz pianist's improvisations. In response to a live performer, the system chooses those individuals in its population that are deemed "similar" to the input. These individuals are then used as responses to the live performer. The system keeps track of which phrases are performed, and those used most often are selected for mating: the result is an evolving population of phrases that we've heard. This is a variation of Spector and Alpern's proposal, from 1994, that the GA use a case-based system of fitness; in other words, rather than beginning with a population of randomly generated individuals (as GA usually does), begin with a population drawn from a database that the artists find musically useful [Spector and Alpern].

11. Multi-agent systems

Current trends in AI research also suggest exciting new potential approaches to musical interactivity. Rather than being based in a hierarchical system, in which the computer is an extension of the composer (or acts as a single performer), multi-agent systems suggest a more evolutionary approach that could model groups of improvisors.

Software agents have been defined as being autonomous, social, reactive, and proactive, traits which they share with improvising musicians [Wooldridge and Jennings]. In such systems, the agents themselves are given intelligence - their interactions are not defined; instead, they are given rules on how to interact, and the interactions develop in evolutionary ways.

In the BDI model, agents have a set of Beliefs, Desires, and Intentions. The beliefs could be musical knowledge, such as tonality, melodic similarities, timbral transformation. However, one important point is that using the term belief, rather than knowledge, recognizes that what an agent believes may not actually be true, or could change. Desires are goals, or motivations for the agents, and intentions refer to the current plan of the agent. An interesting aspect of plans are that they can be only partially conceived, with details being filled in as they progress; this seems to parallel how musicians behave in improvisational settings, in which they are working towards something that hasn't been formalized.

One could easily imagine an ensemble of agents who beliefs include spatialization, and whose desires are to distribute a layered EA work through a multi-channel sound system. Agents would communicate their content in terms of spectral evolution, being careful not to mask one another's gestures.

12. Reinforcement Learning

Another model for musical interactivity that can be borrowed from computer science is the notion of reinforcement learning (RL). This is concerned with how agents ought to take actions in an environment, so as to maximize some notion of long-term reward. Again, like the BDI model, this suggests musical applications, particularly in improvisational settings. In our case, the environment to be explored is the improvisation; reward being some notion of successful interaction, or perhaps arrival at a final, agreed upon conclusion.

Agents learn to navigate through the space; one successful policy is to choose actions which have been successful in the past (they have been previously reinforced through rewards), but still occasionally taking random actions in order to explore the space. The effective improvisor similarly may rely upon trusted strategies, but exclusive reliance upon these would make her predictable.

RL has proven to be particularly well suited to problems which involve long-term versus short-term trade-offs: in a musical model, this might involve reacting directly to incoming data (i.e. imitating the musician) versus ignoring the musicians in order to move toward the next section (i.e. introducing motives from the next section).

13. Conclusion: Considering the Future

I'm going to conclude with a brief description of a system that proposes a new model for realtime human-computer interaction, which combines many of the above aspects. Peter Beyls, a Belgian composer, has documented his as-yet unnamed system that is based upon the idea of mutual influence, in which motivations evolve in performance that either integrate with a human, or express a native (machine) character. It includes a genetic algorithm to evolve musical processing functions that offer musical expertise to fulfill these contradictory goals. The shifting distance between human and machine musical statements is traced, and used to derive the fitness measure of the processing functions. The paper describes how man and machine can develop interesting musical interactions that have developed over rehearsal-time, something that seems to emulate how actual musicians may interact. Although we will probably have to wait to hear it in performance, it seems to suggest exciting new directions for the future of live electroacoustic music.

14. Bibliography

AMES, Charles, "The Markov Process as a Compositional Model: A Survey and Tutorial", pp. 175-187, in: Leonardo 22:2, 1989.

ASSAYAG, Gérard, Georges BLOCH, Marc CHEMILLIER, Arshia CONT, and Shlomo DUBNOV, "OMax brothers: a dynamic topology of agents for improvisation learning", pp. 125-132, in: Proceedings of the 1st ACM workshop on Audio and music computing multimedia, Santa Barbara, ACM, 2006.

BEYLS, Peter, “Interactive Composing as the Expression of Autonomous Machine Motivations”, pp. 267-274, in: Proceedings of the International Computer Music Conference, San Francisco, 2009.
BILES, John, “GenJam: A Genetic Algorithm for Generating Jazz Solos”, pp. 131-137, in: Proceedings of the International Computer Music Conference, San Francisco, 1994.
CHADABE, Joel, “Electronic Music: Unsung Revolutions of the 20th Century”, http://www.percontra.net/6music2.htm, Accessed 6 December 2009.
EIGENFELDT, Arne, “Realtime Composition or Computer Improvisation - A Composer's Search for Intelligent Tools in Interactive Computer Music", EMS07 Leicester, http://www.ems-network.org/spip.php?article264, accessed 6 December 2009.
KOHONEN, Teuvo, “The Self-organizing map” pp. 1-6, in: Neurocomputing, 1-3:6, 1998.
PACHET, Francois, “The Continuator: Musical Interaction With Style”, pp. 333-341, in: Journal of New Music Research, 32:3, 2003.
ROWE, Robert, Machine Musicianship, Cambridge, MIT Press, 2001.
SPECTOR, Lee, and Adam ALPERN, “Criticism, Culture, and the Automatic Generation of Artworks”, pp. 3-8, in: National Conference on Artificial Intelligence, Menlo Park, 1994.
TZANETAKIS, George, and Perry COOK, “MARSYAS: A Framework for Audio Analysis”, pp.169-175, in: Organized Sound, Cambridge University Press 4:3, 2000.
WEAVER, Warren, “Science and Complexity”, pp. 536-544, in: American Scientist, 36, 1948.
WEINBERG, Gil, Mark GODFREY, Alex RAE, and John RHOADS, “A Real-Time Genetic Algorithm in Human-Robot Musical Improvisation”, pp. 351-359, in: Computer Music Modeling and Retrieval, Sense of Sounds, Springer, Berlin, 2008.
WIGGINS, Geraint. and Alan SMAILL, “Musical Knowledge: What can Artificial Intelligence bring to the Musician?” in: E.R. Miranda, (ed.), Readings in Music and Artificial Intelligence, Amsterdam, Harwood Academic Publishers, 2000, pp. 29-46.
WOOLDRIDGE, Michael, and Nicholas JENNINGS, “Intelligent agents: theory and practice”, pp. 115-152, in: Knowledge Engineering Review, 10:2, 1995.
ZWICKER, Eberhard, “Subdivision of the Audible Frequency Range into Critical Bands”, pp. 248-248, in: Journal of the Acoustic Society of America, 33:2, 1961.