The Frequencyliator – Distributing Structures for Networked Laptop Improvisation

Pedro Rebelo
Sonic Arts Research Centre (SARC)
Queen's University Belfast
Belfast BT7 1NN, Northern Ireland
+44 (0)28 90974829
p.rebelo@qub.ac.uk

Alain B. Renaud
Sonic Arts Research Centre (SARC)
Queen's University Belfast
Belfast BT7 1NN, Northern Ireland
+44 (0)28 90974829
alain.renaud@qub.ac.uk

ABSTRACT
The culture of laptop improvisation has grown tremendously in recent years. The development of personalized software instruments presents interesting issues in the context of improvised group performances. This paper examines an approach that is aimed at increasing the modes of interactivity between laptop performers and at the same time suggests ways in which audiences can better discern and identify the sonic characteristics of each laptop performer. We refer to software implementation that was developed for the BLISS networked laptop ensemble with view to designing a shared format for the exchange of messages within local and internet based networks.

Keywords
Networked audio technologies, laptop ensemble, centralized audio server, improvisation

1. INTRODUCTION
The desire to use machines to share musical data during a performance has been present since the earliest developments of computer technology. It not only suggests a reflection on the exchange of musical ideas between human performers but also an exploration of something computers became very good at: data transfer. The development of reliable high-speed networks and ever increasing computer power have made truly effective networked music events possible. We investigate the potential emergence of new forms for musical interaction that are a direct result of networked structures. BLISS (Belfast Legion for Improvised Sights and Sounds) [1]– a laptop improvisation ensemble together with other collaborators act as a "testbed" for the development of technologies for networked music ensembles (NMEs).

2. BRIEF HISTORY OF NMEs
John Cage's "Imaginary Landscape No. 4 for twelve radios" can be seen as an early NME experiment. The piece, "used radio transistors as a musical instrument. The transistors were interconnected thus influencing each other." [2] Although the levels of interactivity were limited to the dialing of radio-stations, gain and tone-colour, the desire to investigate the possibilities of cross-influence in networked instruments is evident in the piece.

NIME 06, June 4-8, 2006, Paris, France.

It is really with the development of microcomputers that more interactive NMEs began to emerge. From 1978 to 1982, The League of Automatic Music Composers originally composed of Jim Horton, Tim Perkis, and John Bischoff started using networked computers to exchange messaging data between each other with the goal of influencing their playing. The group, which used Commodore KIM-1 computers, developed ways to increase the level of interdependency between players by, for example, using frequencies in one computer to generate notes in another.

The technologies and playing techniques pioneered by the League of Automatic Music Composers were important since it was the first time that a group of electronic musicians attempted to include computers in a live music environment. The League represents a singular project in a period when computer music research was almost solely focusing on non real-time applications.

In 1987, the League of Automatic Music Composers became The Hub [3]. The ensemble included Chris Brown, Scot Gresham-Lancaster, Mark Trayle, Tim Perkis, Phil Stone, and John Bischoff. The principle of The Hub was also based on interdependency with a more elaborated system of communication such as a form of audience participation and the enabling of remote collaboration. Later on, the ensemble started using MIDI as the main communication protocol. The Hub, was initially the name of the central computer used to store and distribute messages amongst the performers. The group quickly developed a standardized interface with the aim of allowing any type of computer available at the time to connect to the network. In 1987, The Hub had its first networked performance through a collaboration with composers Nic Collins and Phil Niblock. A live performance took place between two spaces in New York which were linked by a basic modem connection through which messages were exchanged.

In 1997, the group started experimenting with remote performances over the internet although these presented significant technical challenges. The Hub still performs today on various occasions.

The existence of pioneering NMEs such as The Hub led to a wider spread practice in networked performance. The SoundWire project, led by Chris Chafe at Stanford's CCRMA [4] has developed a set of applications, which allow high quality networked performances over high speed internet. It is one of the first applications to take advantage of the bandwidth offered by next generation IP networks such as Internet2. The project has also developed ways to evaluate network reliably through the use of audio signals.

Standard internet infrastructure is increasingly being used for collaboration and interaction. This is the case of Prométhée

Numérique (2002), a radio art piece structured around a web-site which is accessible by remote web users [5].

The development of new message protocols such as Open Sound Control (OSC) allows much more flexibility for exchanging control messages between players. The flexibility provided by OSC as a "transport-independent, high-level application protocol" [6] is very suitable as a basis for the development of present and future NMEs.

Such technologies and practices has led to a new classification of various types of networked setups as illustrated by Barbosa [7]. These include: Local Interconnected Musical Networks, Musical Composition Support Systems, Remote Music Performance Systems and Shared Sonic Environments.

3. IMPROVISATION AND THE NETWORK

3.1 Performing Roles

In order to develop musical interactions within the context of networked improvisation, we have explored the dynamics between free or non-idiomatic performance [8] and pre-determined temporal structures. Whilst rejecting specific musical languages or idioms such as those characteristic of Jazz-based improvisation, free improvisation relies nevertheless on certain structures and roles. Amongst these are the constraints dictated by each musical instrument. A quartet consisting of drums, bass, piano and saxophone suggests certain relationships between the performers (which can of course be challenged and transgressed). These relationships are less apparent in a quartet composed solely of computer musicians. The desire to turn the computer into a universal instrument which is capable of all sounds at all times has produced a performance practice in which performative roles are not clearly identifiable. Although one can see this as a reflection of a wider social trend that has replaced the guitar hero with the cooler and certainly less visible DJ, the relative anonymity and dispersed responsibility that characterizes many computer-based ensembles could be seen as musically restrictive. While a solo laptop performer is capable or articulating dramatic shifts in texture, loudness or timbre, at literally the press of a button, once another performer is introduced, certain types of shifts become increasingly difficult to articulate. While instrumental performers often recur to visual cues and gestures for communication in a group context, the laptop performer rarely counts on this mode of communication to coordinate events. The posture of most laptop performers favors the intense gaze towards the computer screen at the expense of interpersonal cues. In contrast, instrument-based improvising ensembles have developed a performance practice which is very much based on the identification of roles (even though each musician will have a number of different roles during a performance), and in the communication of form through gesture and visual cues.

3.2 Musical Structures

We have attempted to develop an environment in which an ensemble performing solely on software instruments can resource to musical structures that are very much taken for granted in an instrumental ensemble. A non-exhaustive list would include the following:

- Potential for common pulse to emerge through rhythmic interaction.
- Synchronization of events that require negotiation and agreement from two or more performers.
- Balanced spectral structure with possibilities for both masking and extreme spectral separation.
- Identification of performative roles which suggest performer-to-performer as well as audience-performer interactions.
- Up beats or the ability to anticipate and precede events across the ensemble.

These structures can be identified in a number of improvisatory contexts (idiomatic or not) and often provide a collective platform for the development of musical form. The generic nature of the computer as a musical instrument often conflicts with the specificity that characterizes most musical performance situations. While maintaining a free improvisation approach to musical materials we have attempted to design a framework within which this "freedom" is constrained in order to facilitate certain types of performance interactions.

3.3 Networked Interactions

The musical structures identified above are difficult to recreate intuitively with NMEs due to the lack of common performance practice. Although the instrumental paradigm has been used to describe the software used in performance by a computer musician, it is rare that this software behaves like an instrument (unless it is designed to imitate an acoustic instrument). The constraints that define acoustic musical instruments are often unobserved in the development of a "software instrument". While this might offer the potential for novel instrumental roles, the desire for technological sophistication often obscures consideration for musical contexts. The limits in range, amplitude and articulation that characterizes acoustic instruments are replaced by flexibility, unlimited access to all soundfiles in a hard drive, and open-ended spectral behavior. Without wanting to disregard the possibilities that these software instruments suggest, it is worth considering the role of constraints in a situation that is often characterized by notional freedom.

The system has been used on a local network with four performers in concert. The first application is based on a score that has been written "offline". The score, which describes elements such as tempo variations, filter banks and bandwidth separation has been interpreted and entered into the system. This approach provides a central point of reference to the ensemble, which is not computerized.

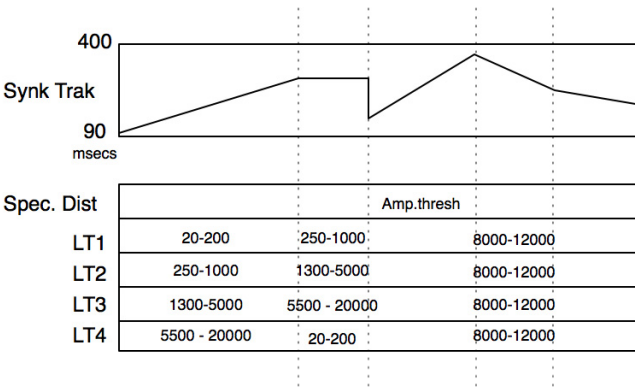

Figure 1: Excerpt from the score for bliss.net

Figure 2: The BLISS Ensemble Setup

4. THE FREQUENCYLIATOR

4.1 Concept

The Frequencyliator is a software tool being developed as the outcome of the initial experiments made with the BLISS ensemble. The concept is to create a framework for laptop improvisers to easily collaborate and exchange musical ideas over a local or remote networked setting. The primary goal of the Frequencyliator is to recreate such interactions through the implementation of a basic structure that is imposed to the ensemble. Basically, a server issues various messages to the ensemble, whilst each audio output is routed through the server for processing.

4.2 Design

The system is being developed with Max/MSP [9]. The application will include a series of message and/or DSP based modules:

- Shared timeline (Message)
- Bandwidth allocation mechanism (DSP)
- Countdown (Message)
- Sync event (Message)
- Spectrum analyzer (DSP)

All the modules will be connectable with each other on a local and global level. The messaging part of the system will be based on the Open Sound Control (OSC) protocol while the audio signals will use a low latency audio to UDP adapter such as the one developed by the SoundWire project [10]. Even though it is crucial to keep a server/client model to allow a master clock to act as a central point of synchronization for the ensemble as well as for connectivity reasons, the system will allow performers to send messages and audio content to each other. The software is designed around modules which implement a particular type of musical interaction. This interaction is then articulated through a function which can be pre-determined or generated in real-time, by an algorithm or manually by any of the performers. Each performer can create a new function which is then automatically suggested to the ensemble and approved or denied based on the percentage of positive and negative votes within a specific time limit. This promotes a basic yet dynamic way of exploring the notion of negotiation within an ensemble. Below is a more detailed description of the modules currently implemented.

4.2.1 Shared timeline

The Shared timeline is a central structure within the system. The timeline resembles a proportionate score that defines the form of a piece. Its generation can be pre-determined and implemented manually or generated in real-time by an algorithm. For example, a function can define variations in tempo whilst a secondary envelope can send frequency interpolation messages to a filter bank. Timeline messages can be sent to the entire ensemble (global) or to a selected member of the ensemble (local). Variations in timeline parameters can be suggested on an ad-hoc basis locally. The potential for common pulse is achieved through a common rhythmic structure, which is sent as messages to each performer, not unlike a conductor's beat. The timeline is implemented as a *qlist* MAX object. Each section of the qlist sends internal messages that influence the server's behavior (e.g. centre frequency and bandwidth for frequency distribution). External messages such as pulse and general section change messages are also included in the list.

4.2.2 Bandwidth allocation mechanism

This module allocates a given frequency bandwidth for each laptop plugged into the system. This is implemented with a range of filters that process the audio signal from each performer. Allocated bandwidth parameters are sent to each player as messages. The variation in bandwidth allocation will generally be controlled by the timeline but can be driven by interactive behaviors such as swapping bandwidths as triggered by amplitude thresholds. The filtering is achieved through the use of MSP objects *reson~* for bandwidth separation and *fffb~* for filterbanks. This mechanism allows for spectral structures across the ensemble (e.g. distribution of a complex spectrum through a series of filterbanks for each performer) as well as the identification of individual roles.

4.2.3 Countdown

The countdown mechanism is purely based on messages sent to each performer to allow for anticipation of events. A typical message would be a 10 second warning that a section of the piece is about to finish. This mechanism provides a reliable way to automatically warn each member of the ensemble when a new section is about to begin, usually meaning each performer will be assigned a new role within the ensemble, not unlike the transition from tune to solos in Jazz.

4.2.4 Sync event

This type of event is accessible by each performer and involves sending trigger events ("bang messages") either to the entire ensemble or to one or several carefully chosen performers. Each performer can choose to activate or de-activate the reception of such messages as well as the mapping to a local event.

4.2.5 Spectrum analyzer

This module detects and distributes characteristics of the audio signal of each performer. An FFT analysis on each signal captures partial distribution, spectral centroid, onsets as well as perceptual parameters such as brightness and noisiness through the use of Tristan Jehan's *analyzer~* MSP object [11]. The analysis data is formatted and shared amongst the ensemble, allowing for the synchronization of events on onsets, or "spectral grabbing" which consists of one performer grabbing an FFT frame from another signal in order to create similar

spectral content. Information from this module can be used to influence the behavior of other modules (e.g. timeline). BLISS has used this module to detect amplitude thresholds as analyzed after bandwidth allocations. The crossing of these thresholds causes a change in bandwidth allocations, effectively resulting in a "stealing" of bandwidths. This type mechanism adds a level of interactivity within the ensemble that resembles game structures present in works such as John Zorn's Cobra [12].

4.3 Interface

A standard interface rendering the events being broadcast by the server displays each parameter to the performers. Each performer uses the Frequencyliator as a subpatch window that is added to their software instrument. The suggestions are sent on an ad-hoc basis by any given member to modify elements as the piece develops can be "dragged and dropped" into the structure of the piece via this interface.

5. CONCLUSION AND FUTURE WORK

This server-based approach to structured improvisation provides a rich platform for networked ensembles.

At the moment, the interface only allows a unidirectional communication of messages from the server to the clients, whilst the clients send the output of their signals back to the server. This limitation will soon be overcome by introducing bi-directional communication for both messages and audio data. This approach will most certainly lead to an increased level of interaction since it will allow, for a performer to send messages to another performer or even to the server.

The system will be optimized for remote participation. The frequency separation mechanism introduced in this context provides an excellent way to identify distinctively the signal output of each performer. We can therefore envisage the participation of other laptop performers from remote locations. This approach is technically possible as initial audio tests as part of a separate project between SARC and CCRMA have proven to be very conclusive.

Finally, another goal is to standardize the interface of the Frequencyliator and make it available to anyone who wishes to participate in such a performance.

6. ACKNOWLEDGMENTS

We would like to thank Chris Chafe for his dedication to making the network heard, and to the other members of BLISS involved in this project: Tom Davis, Jason Dixon and Chris McClelland.

7. REFERENCES

[1] Bliss Web www.sarc.qub.ac.uk/~bliss Accessed on January 23, 2006.

[2] Pritchett, J. The Music Of John Cage. Cambridge University Press, Cambridge, UK, 1993.

[3] Brown, C and Bischoff, J, Hub Origins http://crossfade.walkerart.org/brownbischoff/. Accessed on January 23, 2006.

[4] Chafe, C, Wilson, S, Leistikow, R, Chisholm, D and Scavone, G, A simplified approach to high quality music and sound over IP. In Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFX-00), (Verona, Italy, December 7-9,2000).

[5] Tanaka, A. *Composing as a Function of Infrastructure.* Errant Bodies Press, 2003

[6] Wright, M, Freed,A and Momeni, A, OpenSoundControl: State of the Art 2003. In *Proceedings of the 2003 Conference on New Interfaces for Musical Expression (NIME-03)*, (Montreal, Canada, May 22-24, 2003

[7] Barbosa, A, Displaced Soundscapes: A Survey of NetworkSystems for Music and Sonic Art Creation. *LEONARDO MUSIC JOURNAL, Vol. 13, pp. 53–59, 2003.*

[8] Bailey, D. Improvisation: Its nature and practice in music. Da Capo Press, 1993

[9] *Cycling 74' MAX/MSP* http://www.cycling74.com/ Accessed on January 29, 2006.

[10] *SoundWire Project page* http://ccrma.stanford.edu/groups/soundwire/ Accessed on April 12, 2006.

[11] *Tristan Jehan's Max/MSP object page* http://web.media.mit.edu/~tristan/maxmsp.html Accessed on January 29, 2006.

[12] Zorn, J. *Cobra* Tzadik Cat. # 7335, 2002