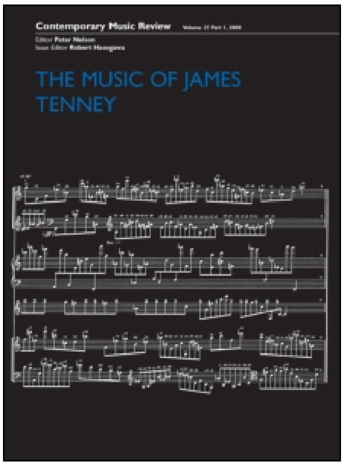

Contemporary Music Review
Publication details, including instructions for authors and subscription information:
http://www.informaworld.com/smpp/title~content=t713455393

Generative Music and Laptop Performance

Nick Collins

Online publication date: 03 June 2010

To cite this Article Collins, Nick(2003) 'Generative Music and Laptop Performance', Contemporary Music Review, 22: 4, 67 — 79
To link to this Article: DOI: 10.1080/0749446032000156919
URL: http://dx.doi.org/10.1080/0749446032000156919

Contemporary Music Review, 2003, Vol. 22, No. 4, 67–79

Routledge
Taylor & Francis Group

Generative Music and Laptop Performance

Nick Collins

Live computer music is the perfect medium for generative music systems, for non-linear compositional constructions and for interactive manipulation of sound processing. Unfortunately, much of the complexity of these real-time systems is lost on a potential audience, excepting those few connoisseurs who sneak round the back to check the laptop screen. An artist using powerful software like SuperCollider or PD cannot be readily distinguished from someone checking their e-mail whilst DJ-ing with iTunes. Without a culture of understanding of both the laptop performer and current generation graphical and text-programming languages for audio, audiences tend to respond most to often gimmicky controllers, or to the tools they have had more exposure to – the (yawn) superstar DJs and their decks. This article attempts to convey the exciting things that are being explored with algorithmic composition and interactive synthesis techniques in live performance. The reasons for building generative music systems and the forms of control attainable over algorithmic processes are investigated. Direct manual control is set against the use of autonomous software agents. In line with this, four techniques for software control during live performance are introduced, namely presets, previewing, autopilot, and the powerful method of live coding. Finally, audio-visual collaboration is discussed.

KEYWORDS: laptop music, algorithmic composition, generative music

Proof of Concept

Advanced generative music has the same public face as the simple cross-fades and pre-composed track library of an mp3 DJ, typically, the glowing apple on the back of a Macintosh Powerbook. Yet many composers are freeing themselves from the representational assumptions of ready-made software like Traktor and Ableton Live, and stunning and innovative projects are undertaken with graphical and textual programming languages for audio like PD, Max/MSP, SuperCollider (McCartney 1998) and custom software entirely written by the performers themselves. This article highlights adventurous programmer/composers rather than off-the-shelf software users, since they do not always receive the credit their experimentation deserves.

Those worried about the paucity of gestural content at the heart of laptop performances, might very much agree with Miller Puckette:

> There must be a direct and comprehensible relationship between the controls we use and the sounds we hear. (This would not be a bad thing from the audience's point of view either.) A performer who pushes a button to start a sequence is not showing us how the music was really made; all we learn about the music is what our ears can tell us. (Puckette 1991)

Contemporary Music Review
ISSN 0749-4467 print/ISSN 1477-2256 online © 2003 Taylor & Francis Ltd
http://www.tandf.co.uk/journals
DOI: 10.1080/0749446032000156919

Figure 1
Fabrice Mogini and Nick Collins perform live in a London public toilet.
They are synced up and both doing complex manipulations with SuperCollider – can you tell from this photo?

Miller was writing at a time when electro-acoustic tape music was the dominant avenue of serious electronica. Kim Cascone (2003) notes that the problems of acousmatic music have been transplanted without resolution to the field of laptop performance.

Is it a healthy state of affairs if all the focus of the performers is directed at their laptops rather than the audience? Concerns about the lack of visual spectacle in laptop music could be a distraction. After all, we are used to seeing DJs ignore us, though, admittedly, (with a proviso for turntable virtuosi like Q-Bert) their simple actions are easily deciphered. Any difficulties may pass in time: since audiences stare at DJs happily enough, why not the backs of laptops? If we have faith in the eventual education of audiences, from a transitional period now, within five years a superstar laptopist may appear on *Top of the Pops*.

The danger is that audiences are acclimatised to laptops, and understand that some music-making program is underway, but assume the use of a playback sequencer like Logic when the real performance is a live intergalactic jam with a being from the planet Sirius.[1] Good program notes might help if artists could be trusted to explain themselves objectively, and admit the restrictions of software like Ableton Live. This information flow also depends on wandering around informal venues making sure the audience is informed, interrupting the continuity of a club night to make announcements, and believing market forces will select the most profound performances worth visiting in the first place.

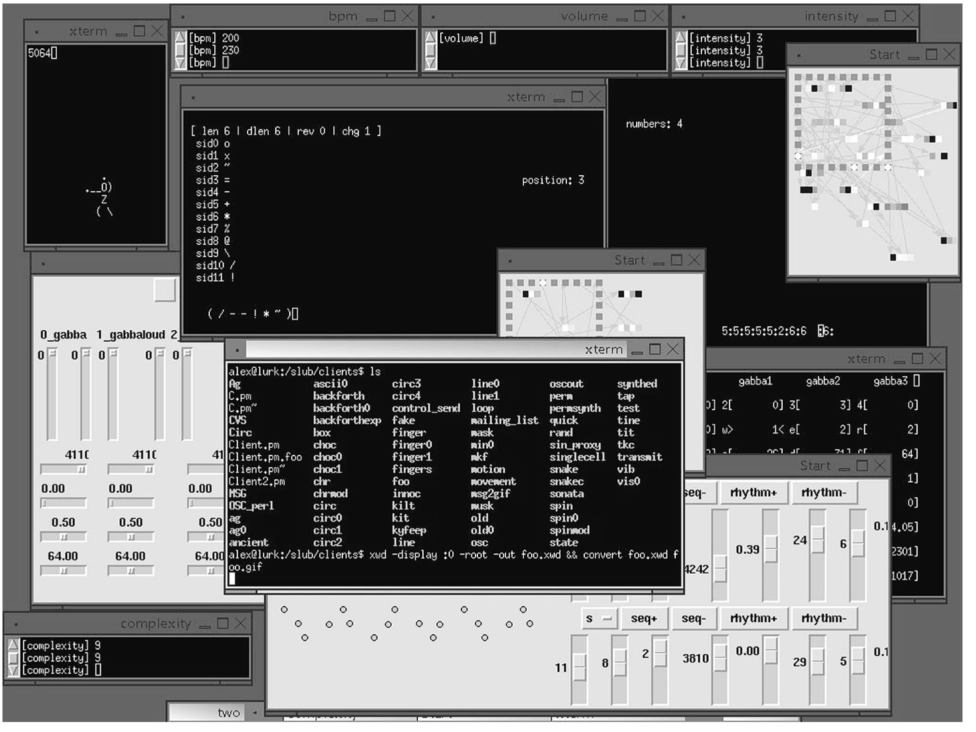

Figure 2
A screenshot from Alex McLean's performance setup (courtesy of Alex McLean).

Alex McLean (2003) has proselytised for the simple solution to this quandary – let the audience see what the performer sees! Reveal the instrument! Project all performer's laptop displays! He has a vested interest in revealing the shallow fakers of laptop music, since his own Linux command line antics for the London duo slub[2] are esoteric and hypnotically beautiful, and allied to deeply thought-out generative music. But, as we shall discuss, many programs, especially where live coding is being used, cannot be interpreted by a lay audience, and, in some cases, not even the author follows everything going on in their chaotic dynamical systems.

Roger Dean stands in opposition to McLean's position when he writes that "some audiences are already adjusted to the idea of focusing on sound rather than action; others will simply have to catch up" (Dean 2003). His book on improvisation includes antithetical opinions as well though; for instance, Jeff Pressing is quoted arguing that "physicality is essential to get engagement".

The story of the museum cleaner who dismantled a Damian Hirst exhibit, thinking it rubbish (McCormack and Dorin 2001, note 3), could be interpreted as evidence of insider knowledge adding importance and pomposity, often converting what seems stupid and pointless to what seems profound and well thought out. I suspect we take the most pleasure in the confounding of our assumptions, but that the damage is already done to the art work. Audiences judge on the information communicated, and if only all artefacts could communicate themselves self evidently. Apposite to this, Iannis Xenakis (1992: ix) provides a typically fierce quote on musical communication: "The quantity of intelligence carried by the sounds must be the true criterion of the validity of a particular music." This implies that the communication of the intelligence is critical. Listeners observe imperceptible intellect as stupidity, and the deeper insight gained by critics with access to the score/code/plan is a touch suspicious. Xenakis (1992: 8) corroborates this position, since his argument from the famous "The crisis of serial music" article depends on judging clearly the perceptual effect of a music, rather than totemising its internal system. We have returned to Miller Puckette's position.

In general circumstances, what hope is there for the audience to tell who the real innovators are? Ultimately, innovation is its own reward, and the pale imitators making millions from experimental research a decade hence must be stomached. I think it most appropriate to side-step issues of fame and credit and opacity, and instead to concentrate on education. Allied to issues of laptop performance are issues of the forms and twists of new generative music. I shall discuss why non-linear music has such fantastic rewards.

Why Make Generative Music in Live Performance?

Variously termed generative, non-linear, adaptive or algorithmic music, the reader is referred elsewhere (Ames 1987; Eno 1996; Roads 1996; Eacott 2000) for an introduction to algorithmic composition and generative music. I shall make no attempt to explain the vast arena of generative systems, of chaotic dynamics, the game of life and other emergent phenomena, artificial intelligence research into neural nets and expert systems, good old generate and test and the history of automated music. What I wish to dwell on is the reasoning that supports running such systems live, on a laptop.

I shall begin (as often happens in this field) with Iannis Xenakis. He discusses stochastic music pieces that vary with each performance:

> . . . music which can be distorted in the course of time, giving the same observer the n results apparently due to chance for n performances. In the long run the music will follow the laws of probability and the performances will be statistically identical with each other. (Xenakis 1992: 37)

In this situation, the nature of the distribution is heard over many auditions rather than one possibly aberrant sampling of it.[3] Xenakis makes the criticism that when a fixed run of the stochastic process is taken as the artwork, human memory will begin to work on its perception over multiple hearings: "Will it not change into a set of foreseeable phenomena through the existence of memory, despite the fact that the law of frequencies has been derived from the laws of chance" (Xenakis 1992: 37).

This is why generative pieces which re-compute the actual events, but remain overall in accord with a given recipe, must be run once per performance to avoid a foothold for memory. Freshness is hampered by memory. If the audience hears only one performance, the argument is still not entirely invalid since the composer hears it many times. A critic might insist that all generative music performers repeat the pieces in their set a number of times – in fact, they can play a piece once but get more mileage from the material. The freshness keeps it alive, in that the distribution is manifest, but there are no exactly repeating fine details to get hung up on. In this vein, generative music is perfect for interactive games, for as game composers well know (Harland 2000), fixed songs, no matter how good, when repeated twenty times over in one sitting, inevitably lead to the gamer disabling the music! In nightmares I find myself doomed in some unexpected afterlife to select a piece of music to hear on a loop for eternity. I know that no matter how complex and artful a fixed piece I choose, it must wear thin in time. I think this is why I design pieces that never repeat within the age of the universe (Collins 2002a).

Generative music has a capacity for greater subversion of memory and greater thrills of unknowability. In fact, our culture already has generative aspects enshrined in live music, for we expect repeatable behaviour but within given cultural and perceivable constraints, the leeway identified as nuance and interpretation. Human expression is expected, and we are very sensitive to "errors" of contextual aesthetic. Yet we can adapt: deliberate consistent errors that confound us can grow to form a new aesthetic. I am becoming increasingly enamoured with Autechre's awkward beats, which, with familiarity, become more vital as rhythms than the standard metronomic dance. Still, I expect to become bored with it soon: whilst playback of fixed media has no expressive deviation, generative music can submit any parameter to modification over time.

The storage requirements for generative music are small (which is why generative music and the Internet go hand in hand), for it is compact to store a routine that works in the style rather than a complete audio example. A list of fixed synthesis instructions could be passed, but in general the analysis data for a piece can exceed the storage space of the time amplitude samples. As for the storage problems of infinite pieces, the benefits of an infinite generative grammar can only be reaped.

As a thought experiment, one can imagine a critic working from what happened at a concert, not from what was potentiate. They applied the auditioning test – "you have ten minutes to impress me!" This emphasises that generative music is best appreciated when studied closely, when run many times, and that true appreciation can place you in the role of understanding everything the composer created. It is not trivial, for the design of generative music is of an order of magnitude harder than making fixed products. But this is the sort of responsibility for the audience that is unlikely to gain mass adherence. Further, we are idealistically assuming that composers understand the forces they unleash – in programming, the shortcut of using third-party code without analysis is a constant temptation.

If we consider a generative piece as a machine for creating products to a hazy mould or blurry specification, there is a definite justification for its existence as an exploratory engine. Instead of blind probing, it is helpful to have sampling procedures that guarantee some directed coverage of ground towards a final goal. These are search tools like the interactive genetic algorithm that find preferred spots in the output space (Dahlstedt 2001). Alternatively, it must be claimed that every possible output of a generative composition is of value. To do this with conviction, the designer must have predicted every possible contingency, which can be extremely difficult for the combinatorial explosions of music spaces. Software testing procedures become relevant to music composition.

Are we getting any closer to the point of live generative music? At a live concert, is generative music a music that says *this time* is special, *now* is privileged? If the decision making could *only* have been carried out at this point. In the limiting case, I assume we would be unwilling to take the seeding of a random number generator with the current time as evidence of any awareness of the present. If a live run for the evening show may be qualitatively equivalent to another made in the afternoon, a generative take stored earlier and now replayed seems just as valid (and far safer to run) than the real thing. In some cases, the generative system cannot play in real time (perhaps due to processing cost), so fixed products are the only solution.

Three situations can be envisaged where generative music is essential. First, there was not time or storage space to prepare a whole set in fixed form, which could happen where the piece is very long, is an installation work, or where there are a massive number of generative works to curate. Second, and rather weakly, perhaps a tenuous hidden conceptual thrill that the system is running during listening is desired. Third, there is some sort of interface that allows human engagement, or some time-specific dependency on the environment, that makes the generative system react to the location of the performance. It is the latter option that is assumed as the basis of the next section.

Control Possibilities, or Can I Convince Anybody This is Live?

Timeblind, a SuperCollider artist signed to the Orthlorng Musork label, writes provocatively:

> Laptops are just a phase. Right now it looks like we're watching TV or checking our email. The mouse sucks . . . people can't tell what I'm doing unless they can get where they can see me. I want to get these huge levers, like 6 feet tall. So you can see them in the back of the stadium. I'll paint my face white and hurl myself at these levers. Musically it will be slower and less interactive, but people will be sucked in by the physical display. (Chris Sattinger, personal communication)

It is not my aim to go through every mapping that might be attempted, every strange (and often gimmicky) control device on the market or constructible in the garden shed.[4] I would prefer, however, to emphasise some issues that arise particularly in the software control of interactive music systems. Rather than cover the issue of interactivity and computer-assisted improvisation in a general setting (Rowe 1993, 2001; Roads 1996; Dean 2003), I would like to home in on the situation for the laptop programmer/performer.

As a practical grounding for this discussion, so as not to wallow too much in theory, figure 3 shows the aforementioned Timeblind's SuperCollider-2-based laptop screen. This is just the top sheet of his multi-layered user interface: Timeblind is the author of the Crucial Library[5] extension set for SuperCollider, and a leading practitioner of laptop music black arts. Jesting that he might rename himself DJ Spacebar because of his predilection to key commands rather than mouse movements, he explains:

> SuperCollider has never been known for its fancy graphics. Visually my live setup is really just a display of selections. The objects that are selected vary in behaviour from fixed to wildly unpredictable. There isn't anything to look at visually that would indicate this, and it's entirely based on my own familiarity with each of the little creatures and what it might get up to. All changes are achieved with keystrokes which is vastly faster than the mouse, which I do not touch. The objects are bred in a database environment. My role is to find new ways to breed them and to kill the ugly ones. I play in a variety of situations, and need to vary the amount of risk I expose myself to in a live situation. I always need to hear things that surprise me, but have to keep plenty of rabbits in the hat. (Chris Sattinger, personal communication)

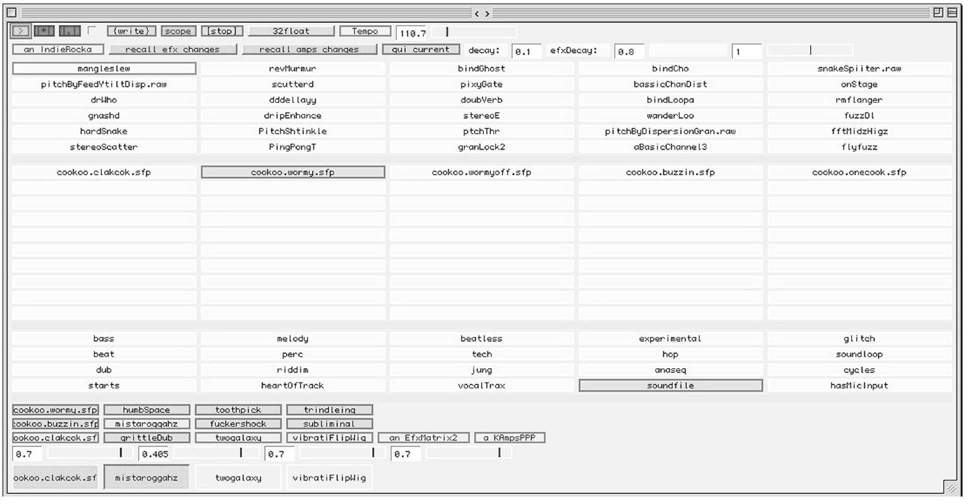

Figure 3
A live performance patch (screenshot courtesy of Chris Sattinger).

To break down the laptop performance situation, imagine you have some generative program hidden behind the back of your laptop. To justify running that generative system live, you will control it with a user interface constructed in your software, but your minimal mouse movements and keystrokes will not be revealing your manipulations to the audience, and you are too shy to project your screen because slub are playing after you.

In the design of interactive system for performance the trade-off between improvisation/adaptability and preparation/quality/seamlessness must be confronted. Risk taking can be made an important part of the work, or the work will demonstrate measured pre-composition. The performance could be rehearsed to the point where the sequence of actions is so predetermined that it cannot adapt to the audience or venue in any way. Should the performance then be pre-recorded? There remains the comfort, well known to the fan of concerti, that it could go wrong. If the model is instrumental performance practice, perhaps some expressive minutiae in the music will be under live control. The likely route is a safe middle way, and, whilst a seat-of-the-pants atmosphere can be established (for the performer, not the ignorant concert goers), if the performer did not debug their code properly, nothing too radical is likely to happen unless there is some form of back-up, or little pressure, as will be discussed.

Yet, assuming you wanted to make things difficult for yourself, how much could you control live? The solo laptop performer is not the most obvious one-man-band virtuoso. The poor mouse interface only allows clicking one slider at a time, so for polyphonic activity a host of MIDI controllers and brainwave trackers and armpit-sweat monitors must be hooked up. Regardless of such preparations, the comparative slowness of human anatomy cannot be avoided and the computer must be granted autonomy for inhuman speeds.

Machine music allows perfect synchronisation, rhythmic virtuosity that is unplayable even by splendid tabla giants, but is still perceivable as an effective musical flow (Collins 2002b[6]). Whilst one would hope to get some handles on the control of such material, the controls must remain within human haptic abilities, i.e. no faster than the 15-Hz barrier of physiology, and tasks suited to cognition.[7] A general energy level may be tracked, conducting cues can establish some level of the hierarchy of rhythmic representation, but it is impossible physically to play for the audience the machine-gun-fire buffer stutters or micro-polyphonic granular swarming or Gaussian distribution counterpoint. If the ability of machines to exceed human capabilities is exploited, the chance to produce that music as a direct instrumentalist is lost.

This situation necessitates conferring on the computer some decision making, some artificial intelligence, but the programmer is at least in charge of the nature of that autonomy. How can the polyphony of voices in the music be tackled, the complex nest of decisions be made? Programming could continue until virtually everything has been automated, leaving a few parameters mapped to banks of sliders or functions triggered by buttons on the onscreen user interface. In designing the interface for the live performance, rather than designing something tough to use, and rather than spending months practising with the system as if it justified the training for an established acoustic instrument, as likely as not the easy solution is selected.

That might have seemed a bleak temporary conclusion, but there is some hope, from a number of quarters. Using separated components of computer music on different machines, or, importantly, using a network protocol like Open Sound Control (Wright and Freed 1997) the control interface can be broadened to allow multiple performers to tackle the intricacies. The man-hours of control for a half-hour performance are given to more than one operator: the laptop orchestra is born.

There is a qualitative leap in the risks that a performer will allow himself to take when performing with back-up in the form of a secondary laptop, or a duet partner. A typical situation is playing a set in ping-pong mode, passing the active buck between laptopists, side by side. If one crashes (the ever-present danger of risk taking, and, computers being what they are, a mild one even for well-tested code), the other is there to step in and provide cover. The improvising ensemble provides an extension of this that is not necessarily conducive to a continuing improvement in fortunes if handled unsympathetically – too many cooks. . . . Careful on the fly distribution of responsibility founded in an understanding of each other's playing styles is a difficult proposition in a music where conventions are hardly as set in playing practice as traditional jazz. The situation is haunted by the same trade-off of pre-planning and extemporisation as we saw in set design. For sensitive performers, used to working with other musicians, this is an exciting opportunity. To egoists, the laptop jam night might sound fun, but in reality will turn into an aural clash.[8]

For an ensemble of laptop performers working within a standardised rhythmic representation, the performers can sync up to a common clock. Typically, one laptop or hardware sync source will be the master, the others slaves, but bi-directional tempo control is certainly plausible. The laptop performers might communicate to each other primarily by leaning and shouting, but a network chat client is useful. On a hopeful note, a report was recently posted on the Super-Collider mailing list about an intercontinental video-conferencing performance linking two performers in Japan with one in America. Using the new network-based SC Server[9], the participants sent instrument definitions to each other with only the time lag through a few communication satellites.

Four Programming Techniques for Laptop Performance Software

As a correlate of this discussion of software control of electronic music, the opportunity is taken to present four techniques for laptop music performance software design that have proved immensely powerful: presets, previewing, autopilot, and live coding.

Presets are not just for recalling states set up during rehearsal, but also states established during the live performance, where one may wish to engineer a return to some effective material to which the audience responded. It is obviously impractical to reset all sliders in a complex user interface manually to some set of values, time consuming to deal with them one by one, and would require interpolation of the values that the sliders control through intermediates from the current to the desired state. It makes great sense to have the ability to take snapshots, to save and load the parameter sets for the control user interface.

Previewing is what a DJ does when (s)he cues up the next record. With multiple laptops, or multiple outs on a soundcard, musical material can be auditioned before releasing it to the mix. A new rhythmic pattern can be lined up in the desired manner so that it kicks in immediately at the right place. Tempo matching to another laptop performer is completed without having to carry out the process over the PA-system! Or as Palle Dahstedt (2001) suggested, the musician can run genetic algorithm searches through the parameter spaces of synthesis and compositional algorithms, and interactively evolve the next pleasing texture live.

In autopilot mode, all responsibility for a process is relinquished to the compositional algorithm running in the machine. At any moment, however, the performer can take manual control of this process, to whatever practical degree. In the ideal, a performance system for generative music would allow zooming in on any component whatsoever to take over control. In practice, there is only a finite amount of time to code, and priorities must be marked out; though, if a system is built with the goal of autopilot in mind from the start, a lot can be achieved. Not to be over zealous, we have already seen how some algorithms are intractable, and some rhythms too difficult for the poor human physique. The human can at least retain some ability to nudge the fruit machine or rock the pinball machine, to force the output down certain avenues. Without this veto, generative music is happy to wander into territory that displeases both performer and audience but must be suffered before the next transition. User interface elements (like sliders, dials and 2-dimensional X-Y planes) can be spawned only for when the user requires them, rather than having an unwieldy screen of all possible components. Ultimately, computer agents can assist the human conductor through a highly intricate dance of cues and autonomous reactions. Whilst the level of artificial intelligence in current performances is rarely significant, one can at least imagine many reduced scenarios where automation provides great dividends, and is the only solution to certain compositional problems.[10]

Live coding (Collins *et al.* 2003) is seen in programs like Max/MSP, PD and Reaktor, which allow dynamic re-patching, the editing of a graphical network of audio modules on the fly. It is at its most powerful where an interpreted programming language for audio synthesis exists, as in SuperCollider, and new commands can be typed in to restructure the existing unit generator graph during performance (figure 4 gives a static glimpse into this).

We asked earlier how much one could expect to control at once, and gave a

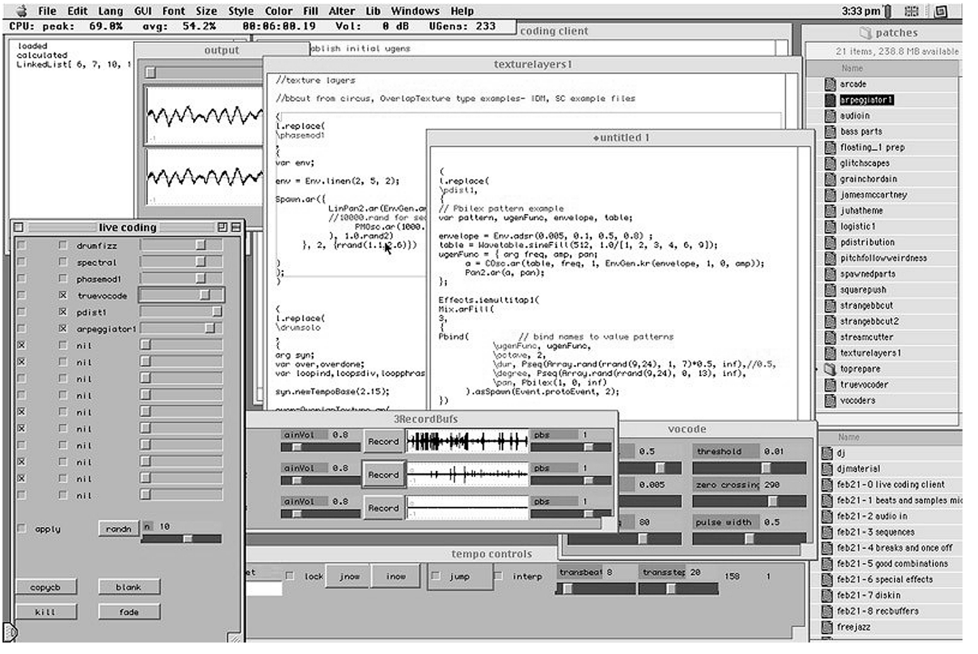

Figure 4
A live coding performance situation in SuperCollider – difficult for anyone but the coder to control, and hard enough for them, but certainly adaptable.

muted response. A similar question is to ask on what conceptual level audio programs can be controlled in the sense of accessing and perturbing the musical generation itself. With a real-time interpreter, the answer is that virtually everything is modifiable. This powerful technique has many ramifications, but, in particular, live rewiring allows the diversion of control and generation to whatever pathway is desired. To cope with autopilot mode in live coding, controls would just be patched in on the fly!

Rather than end on an unqualified propaganda for live coding, there are disadvantages to relate. The most profound for this paper is that live coding is probably the antithesis of revealing work to the audience. Dealings with arcane code mean that the investment of time and concentration at a gig can be taken up entirely by the coding itself. Nevertheless, especially when used as an available technique in an arsenal, live coding has significant advantages for performance control, and is a very interesting take if we wish to speak of the computer as an instrument.

Audiovisual Link-ups

Up to this point, one important avenue for highlighting the mechanisms of a laptop music performance has been avoided. The issue has been deferred because the act of mapping audio to graphics, to live video manipulations and the bouncing of OpenGL objects, is an area very much open to abuse: visuals are an overpowering medium and can easily detract from a musical performance. Yet, given sensitive handling, there are definite benefits here. The interested reader is referred to systems like nato+0.55 and jitter for Max/MSP[11], GEM for PD[12], or the DIPS Library for jMax[13]. Whilst combined audiovisual acts are rarer, independent video-jockeys (VJs) are becoming a more frequent sight. Though a VJ works from some tracking of the audio output of the artists they accompany, there is no guarantee that the artistic agenda of the laptop musician and laptop visual artist will coincide.

In collaboration with Swedish artist Fredrik Olofsson, I have been experimenting in linking automated audio-cutting algorithms (Collins 2002c) to video processing that reflects the nature of those algorithms.[14] The audio-cutting routines run at rates unattainable by human performers, though there are ways to control them indirectly through (sometimes esoteric) parameters of the generative processes, and relatively direct mappings, as from a notion of energy or liveliness. In order to demonstrate that these algorithms really are running live and under human control, live video of the audience or of ourselves dancing about in front of that audience is captured, and live audio collected as the target of the cut procedures. The video is then cut up exactly as the audio buffers are being cut. This has proved an effective demonstration technique for educating audiences. Whilst the mappings are not always so strait laced, for artistic depth often requires more abstraction, the capacity to show a 1–1 correlation is essential to getting the audience involved in the process. These tricks of involvement are as far as I trust going without an inside man in the crowd: initial attempts at interactive clubs have demonstrated that the average Joe Clubber should never be given anything musically important to play with.[15]

To round off with an anecdote, at a recent performance in Helsinki's Jumo Jazz Club gig, as part of the Pixelache VJ festival[16], our audio-visual duo achieved an intensity of sound and display that actually got the audience screaming.[17] Because I was live coding, I was unfortunately unable to take in most of the gig but for my SuperCollider text . . . though occasionally I would spawn some temporary event and show its duration was under control by taking my hands off the laptop. Next up was an Ableton Live user. As if to thwart the dreams of this very article, he got an absolute standing ovation.

Conclusion

It is illuminating to hope that laptop performers really are Xenakis's "composer pilot(s)" (Xenakis 1992: 144). Computers are justified as the natural tools for complex compositional systems, and their use in live performance brings many fascinating challenges, which have no general solution in the mould of traditional instrumental work, for the computer musician. Instead, the laptop performer embodies a new breed, the performer/composer/programmer who delights in anticipating their actions and who designs systems that provide the optimum balance of control and freedom of expression for their performance needs – whether the audience appreciates it or not.

Acknowledgements

Many thanks to Kim Cascone, John Eacott, Alex McLean, Fabrice Mogini, Fredrik Olofsson and Chris Sattinger.

References

Ames, C. (1987) "Automated composition in retrospect: 1956–1986". *Leonardo* 20(2), 169–185.

Cascone, K. (2003) "Grain, sequence, system (three levels of reception in the performance of laptop music)". In *Soundcultures*, ed. M. S. Kleiner and A. Szepanski. Germany: Suhrkamp.

Clarke, E. (1998) "Rhythm and timing in music". In *The Psychology of Music*, 2nd edn, ed. D. Deutsch. London: Academic Press.

Collins, N. (2002a) "Infinite length pieces: a user's guide". *Proceedings of MAXIS*, Sheffield, April. Available online: http://www.sicklincoln.org.

Collins, N. (2002b) "Relating superhuman virtuosity to human performance". *Proceedings of MAXIS*, Sheffield, April.

Collins, N. (2002c) "The BBCut library". *Proceedings of the International Computer Music Conference*, Goteborg, Sweden, September.

Collins, N. and Olofsson, F. (2003) "A protocol for audiovisual cutting". *Proceedings of the International Computer Music Conference*, Singapore, September.

Collins, N., McLean, A., Rohrhuber, J. and Ward, A. (2003) "Live coding techniques for laptop performance". *Organised Sound* 8(3), forthcoming.

Dahlstedt, P. (2001) "Creating and exploring huge parameter spaces: interactive evolution as a tool for sound generation". *Proceedings of the International Computer Music Conference*, Habana, Cuba.

Dean, R. (2003) *Hyperimprovisation: Computer-Interactive Sound Improvisation*. Middleton, WI: A-R Editions.

Eacott, J. (2000) "Form and transience – generative music composition in practice". *Proceedings of Generative Art 2000*, Milan, Italy. Available online: http://www.informal.org/research.htm.

Eno, B. (1996) "Generative music". Transcript of talk from Imagination Conference, San Francisco, 8 June 1996. *In Motion Magazine*. Available online: http://www.inmotionmagazine.com/eno1.html.

Harland, K. (2000) "Composing for interactive music". *Gamasutra*, 17 February. Available online: http://www.gamasutra.com/features/20000217/harland01.htm.

McCartney, J. (1998) "Continued evolution of the SuperCollider real time synthesis environment". *Proceedings of the International Computer Music Conference*, Ann Arbor, MI, USA.

McCormack, J. and Dorin, A. (2001) "Art, emergence and the computational sublime". *Proceedings of Second Iteration: Emergence*, Melbourne, Australia, 5–7 December.

McLean, A. (2003) "ANGRY – usr/bin/bash as a performance tool". In *Cream 12*, from the generative.net mailing list, ed. S. Albert. Available online: http://twenteenthcentury.com/saul/cream12.html.

Puckette, M. (1991) "Something digital". *Computer Music Journal* 15(4), 65–69.

Roads, C. (1996) *The Computer Music Tutorial*. Cambridge, MA: MIT Press.

Rowe, R. (1993) *Interactive Music Systems*. Cambridge, MA: MIT Press.

Rowe, R. (2001) *Machine Musicianship*. Cambridge, MA: MIT Press.

Ulyate, R. and Bianciardi, D. (2002) "The interactive dance club: avoiding chaos in a multi participant environment". *Computer Music Journal* 26(3).

Wright, M. and Freed, A. (1997) "Open sound control: a new protocol for communicating with sound synthesiers". *Proceedings of the International Computer Music Conference*, Thessaloniki, Hellas, pp. 101–104.

Xenakis, I. (1992) *Formalized Music*. Stuyvesant, NY: Pendragon Press.

Notes

1. Yourself and Stockhausen collaborating via SuperCollider3 patches?
2. See http://slub.org.
3. I often think Xenakis should have written more overtly generative music, rather than leaving us so many fixed products in the old style – if he was alive and well with a copy of SuperCollider, we might be passing around and preserving an oeuvre of stochastic music generators.
4. Much as Playstation game controllers for Frankie the Robot DJ, Martin Robinson's steering wheel, Jem Finer's collection of soundtoys for live audio capture, Nicolas Collins's "trombone propelled electronics", Concentric Rectangles' "custom-built sensor helmets, and working blender that sends midi data" and Kia Ng's motion-tracking system are exploring unconventional or subverted interfaces for audio.
5. Available from http://www.crucial-systems.com or as part of the SC3 download.

6. The paper discusses these issues in the search for smooth transitions from human to inhuman, involving notaries like Nancarrow, Ferneyhough and Jaffe.
7. Have you ever tried Paul Fraisse's difficult challenge of clapping a truly irregular rhythm, avoiding the coarse classification into fast and slow pulses? (Clarke 1998).
8. Incidentally, it is a well known phenomenon of laptop duetting that one can lose track of who is providing what. Since every performer has the capacity to flood the whole frequency range with noise, differentiation of material is increasingly difficult the more performers get involved.
9. SC Server is available from http://www.audiosynth.com.
10. The "Nancarrow problem" of writing music beyond human performance ability for one.
11. Available from http://www.cycling74.com/index.html.
12. Available from http://gem.iem.at.
13. Available from http://www.dacreation/dips.html.
14. http://klippav.org has media demonstrations, or, for a technical description, see Collins and Olofsson (2003).
15. The immature technique of a clubber bashing at any old interface is not the most pleasant artistic activity for an impartial observer. This is related in Ulyate and Bianciardi (2002).
16. See http://www.pixelache.ac.
17. There does seems to be a critical load at which an audience is provoked to react.