Creative process and performance practice of interactive computer music: a performer's tale

MARI KIMURA

Faculty, The Juilliard School, New York, USA
E-mail: mkimura@juilliard.edu, mari.kimura@nyu.edu
URL: http://homepages.nyu.edu/~mk4, http://www.marikimura.com

I have had a major interest in the performance practice issues in electronic and interactive systems over the years (see, for example, Kimura 1996). As a performer/composer often presenting pieces from the classical and other contemporary acoustic violin literature in traditional settings along with electronic works, and also as a teacher of interactive computer music performance at a conservatory where my students include highly trained performers, performance practice issues in computer music come up very frequently in association with the creative process. I tend to focus on creating MaxMSP patches that address a particular musical context or situation, rather than creating an elaborate versatile and reusable MaxMSP patch and then using that patch in a particular way to make music. This paper describes a few examples of my interest in this area: (i) System Aspects: Performance Practice Issues and Room Acoustics; (ii) 'Pragmatic' Programming and Performance of Interactive Music; and (iii) Creative Process and Interactive Computer Music.

1. SYSTEM ASPECTS: PERFORMANCE PRACTICE ISSUES AND ROOM ACOUSTICS

1.1.1. Creating a Max/MSP 'one-touch' system for performance

My musical interest and classical background led me towards a system that does not require, during the performance, any operation of the computer, use of a foot pedal, or technical assistance. In order to convey to the audience that using a computer is just one of many means to create music, I wanted to appear to use the computer as seamlessly as possible on stage. For this, I spent some time minimising the numbers of keys on the computer that I have to touch in front of the audience before the performance starts. In a perfect world, I would love to just walk out on stage and start playing, just like in any other acoustic concert situation (we still do tune our instruments); however, using tools in interactive music and often touring by myself with no technical support, I must also prevent unexpected accidents (loose cables, etc.) that would be severely disruptive if I did not correct them just before playing.

Since I am typically holding my instrument (the violin), I have only one hand free. Unless there is an emergency (such as a computer crash) I would not face the computer with both hands, shifting my gestural focus completely away from the audience. I typically keep my computer on during 'non-Max' pieces on the programme; then while 'bowing' (start Max using an assigned function key: one touch), I start the top-level patch (alias would be located on the desktop: one touch); and then finally I start the piece (on/off button: one touch). There is one little secret: I typically have a simple little patch generating a soft sine wave of 'A' (440Hz) before starting the piece (of course, one touch). This is for tuning my instrument, and it usually amuses the audience since I appear to be tuning the computer as well. But this is in fact a small sound check in disguise; I get to turn the audio on, check if audio-out is working, as well as checking my audio-in and pitch tracking.

1.1.2. Foot pedal

For my own composition, I am against using foot pedals in interactive performance. I must quickly add that this is entirely based on personal principles, and is to be applied only to certain situations and compositions. There are instruments and musical situations where using foot pedals makes natural sense, since the pedals are already built in as a part of the instrument (such as with certain percussion instruments or the piano), or a particular form of music calls for it.

There are at least two reasons for me not to use foot pedals. First, there is something inherently unnatural for a standing violinist to try while playing to locate a pedal on the ground and step on it. I often see electric violinists who step on foot pedals, physically turning their bodies or heads towards the ground. From the viewpoint of performance practice, the string and bow positions change and this affects the sound. But also, this is visually distracting for the audience; it brings the focus away from the music. One might think that this is a minor detail, but I see such distraction as unnecessary. But there is a more musical reason for my arguing against using foot pedals. In terms of timing and perception in performance, when one steps on a foot pedal in front of the audience, one creates a 'give-away'. If there is an effect change, which is triggered by a pedal, the audience first notices the pedal, then

Organised Sound **8**(3): 289–296 © 2003 Cambridge University Press. Printed in the United Kingdom. DOI: 10.1017/S1355771803000268

https://doi.org/10.1017/S1355771803000268 Published online by Cambridge University Press

the music that follows it. Again, following the principle that I want to be seamless, I would rather avoid the audience expecting to hear something before they hear it. It might also spoil any sense of surprise the change of scene is supposed to bring.

In an issue of *Contemporary Music Review* entitled, '*Aesthetics of Live Electronic Music*', George Lewis, one of the foremost performers and interactive artists of our time, comments about the use of foot pedals, or what he calls 'magic buttons', with his own highly sophisticated interactive computer system, Voyager: 'There is no built-in hierarchy of human leader / computer follower, no 'veto' buttons, pedals or cues. All communication between the system and the improviser takes place sonically. A performance of Voyager is in a very real sense the result of a process of negotiation between the computer and the improviser' (Lewis 1999). He also adds: 'In an interactive musical situation with a sophisticated structural analyser, magic buttons reduce both (a) the level of communication taking place between the performers and the program, and (b) the flexibility of the improvising performer. Usually devices of this kind force the performer to think about structuring the program's output as well as his/her own playing, yet providing tools for doing so of a quality markedly inferior to those possessed by the program itself' (Lewis 1986).

1.1.3. Technical assistance on (or off) stage

In order to be independent (and again, seamless) during performance, I am also against having a second person controlling the computer while I am playing. The piece could be thought of as a 'duo' performance between the performer and a person controlling the computer, but in almost all cases it is not: the second person simply fills in a technical function whose only *raison d'être* is the lack of built-in automatic coordination between the live performer and the computer. In these performances, in order to execute the written score and for the live performer to interact with a computer, the assistant keeps clicking (at the appropriate moments) on the computer, typically located at the P.A. console, on or off stage (and in most cases, that person is the composer). Note that here I am not discussing works of a free improvisation or 'open score' nature, but that are more traditional in the sense that one reads the score from beginning to end. Naturally, to create a 'stand-alone' interactive performance system for a written score could be difficult, without any external devices or persons.

Since I would rather have the audience hear the sound rather than noticing anything (or anybody) else, and since making such a system reliable is difficult, especially so for a pressing concert date or performance, I have had to make some accommodations. However, I do not want a recorded tape part to synchronise with, or a straight jacket to confine me in timing either, so here is one reliable solution I have used often and with good results.

1.1.4. A solution: using Flexible Timewindow

I have been my own 'tech person' and performer/interpreter at the same time in many venues, and after several accidents in my interactive performances over the years (some might remember my most catastrophic crash at ICMC in San Jose, 1992), I have become almost obsessive about the robustness of my pieces. For my works using Max/MSP, I have been using a Flexible Timewindow, one similar to the 'timeline' object; I set the predetermined timing of each section of the piece as an interval rather than a fixed time, and I make these intervals overlap: 'section 1' starting from the beginning to 20 seconds into the piece, 'section 2' from 15–30 seconds, 'section 3' from 25–50 seconds, and so on. Within these overlapping portions between the timeframes, I set some kind of trigger from the violin so that each section has some flexibility in timing for the interaction to take place in order to advance to the next section. When the piece is more or less predetermined or written, I find that this is a reliable way to change scenes; for example, I typically use a trigger note, or measuring a silence (rest) between notes. Just in case the trigger does not go through (which often happens in a performance in a concert hall due to the acoustics of the space), I do not forget to set a 'safety net' in each section saying: 'if the trigger did not work, move onto the next section by such and such time'. Sometimes, I find myself so afraid of any mishap that I bring myself to sacrifice an interactive patch and make the sections move onto the next automatically, or have the automatic version (or what I call a 'cheat' version) ready until I am able to debug the 'dangerous' one. This may be close to a 'straight jacket', but still it is better than stepping on the pedal to advertise my defeat. At the performance, I am the only one who knows what is interactive or not anyway, and in these emergency cases I disguise my performance in such a way, trying to sound as natural and genuinely interactive as possible, that the audience rarely catches my cheat version! Figure 1 shows an example of one of my first pieces using a Flexible Timewindow.

The timing between the first G and the F at [2a] must be at least three seconds. Any time between 60–90 seconds into the piece, the F will activate patch No. 2 and send program change information to the signal processor. Eleven seconds after patch No. 2 is activated, the program in the signal processor will change again, this time automatically at [2b]; the computer screen displays the timing to let the violinist synchronise with the F♯.

https://doi.org/10.1017/S1355771803000268 Published online by Cambridge University Press

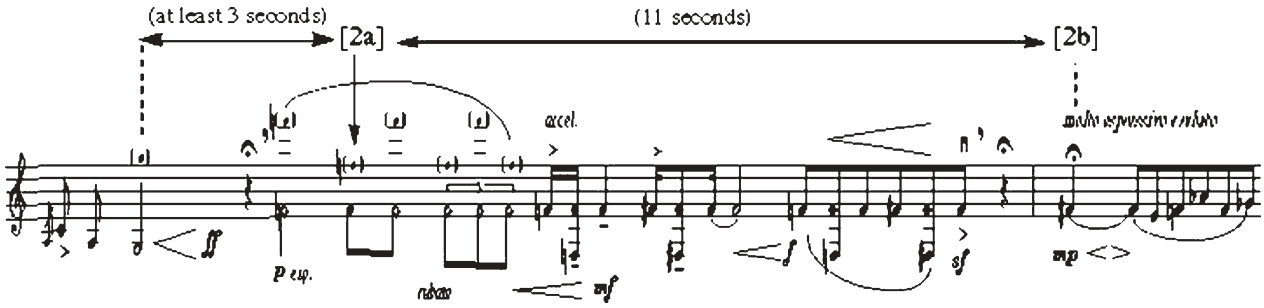


Figure 1. Excerpt from *Gemini* for violin and interactive computer.

1.2. Room acoustics and performance of electronic and interactive computer music

In concert situations, I often encounter various performance spaces that are not suited acoustically for electronic and interactive computer music, especially in solo recital settings. I have always been interested in the musical effect of room acoustics (see Kimura 1995), and sometimes surprised by the major effects the room acoustics have on electronic music in general, and on the acoustic pitch tracking for interactive music in particular. Needless to say, an electronic work can sound very different in a concert hall filled with a live audience compared to the studio environment where it was created.

1.2.2. Velocity tables

In *Izquierda y Derecha* ('Left and Right', 1998) for MIDI piano and violin, I play the Yamaha Disklavier interactively from the Zeta violin. There is a section where chords played by the piano are generated by Max, jumping around while covering five octaves. I made a velocity curve for the Disklavier, 'table.studio' (see figure 2), which I had set while rehearsing in a Yamaha studio in Tokyo, using the same piano to be used for the recital. Notice that the higher sounds were suppressed since I felt it was too loud and bright in the room. On the contrary, the velocity curve 'table.hall' (see figure 3) shows a last-minute adjustment I made at the dress rehearsal in the concert hall, using the same piano. I spent a good twenty minutes on stage, redrawing this velocity curve to fit the acoustical characteristics of the hall. Notice that, around 800 Hz and up (MIDI note number 80, or 'A5') in the 'table.hall', I needed to hear more of the upper register due to diffractions. The difference between the two curves played in the hall was astounding. An acoustic player adjusts his/her own playing instinctively to the performance space, but obviously in this case my system did not see (or hear) the space! Of course I took into account that this hall would be filled with the audience, creating some damping effect, thus I made the upper sounds a bit brighter than I was actually hearing at the dress rehearsal. In addition, I also made some tempo adjustments in certain sections; I made the fastest section a tiny bit slower, in order to accommodate the room acoustics and bring out the musical figures. This kind of last-minute adjustment takes precious dress rehearsal time away, thus adding

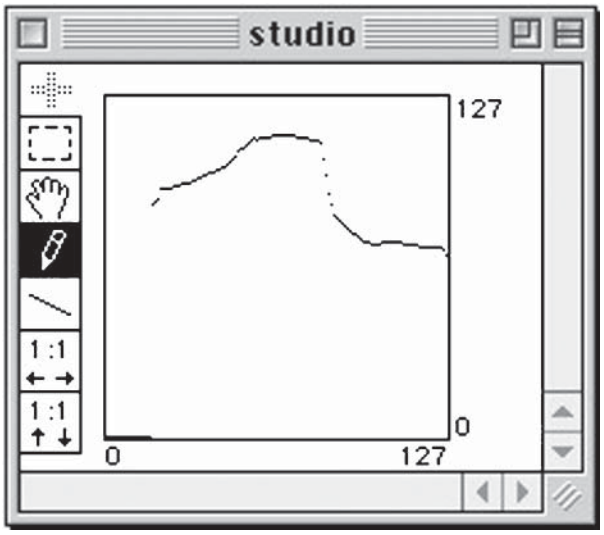


Figure 2. Velocity table: studio.

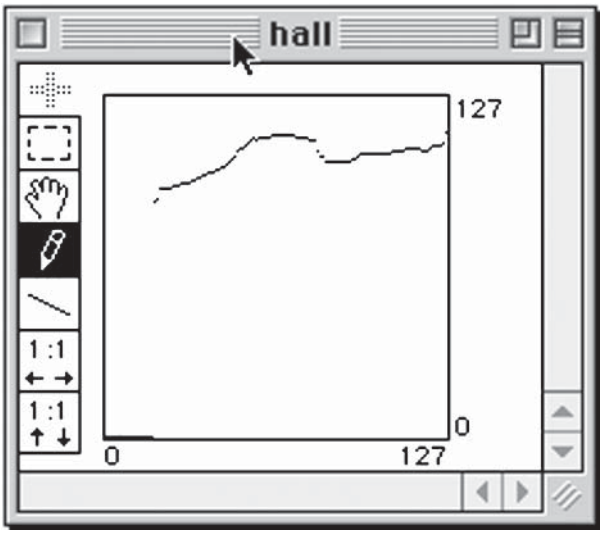


Figure 3. Velocity table: hall.

https://doi.org/10.1017/S1355771803000268 Published online by Cambridge University Press

some unwanted nervousness just before the concert; nevertheless, it is a necessary part of adjusting to a performance space.

A short excerpt of *Izquierda y Derecha* can be heard on the website: http://www.compaq.co.jp/info/qmagazine/qm/qm990708/gallery/gallery.html (The site is in Japanese but you can click on the speaker icon to hear the soundfile.)

2. 'PRAGMATIC' PROGRAMMING AND PERFORMANCE OF INTERACTIVE MUSIC

I find that there are at least two groups of issues I always consider when incorporating an interactive computer music system and acoustic instruments using Max/MSP. One is to create a system that is not only musically coherent but also provocative and fun, that could musically inspire both the performer and the audience. Another is more basic, but more difficult and important: how to teach the computer musicianship, or musical 'common sense'. Whether or not our goal is to make a computer a good ensemble player or not, the immediate performance issues confront me daily and I must deal with them. I believe that the development and the combination of these two elements, 'fun' and 'common sense' (in another words, developing the musical 'rights' and 'responsibilities' of the computer), is one of the most crucial issues in the performance practice of computer music. In the creative process of such musically 'intelligent' systems, one must involve some analysis of musical context and the mental process in performance and composition.

2.1. 'Pragmatic' programming

The first 'fun' part to making the computer musically interesting usually involves creating an interactive system such that the audience can ALMOST guess what the interaction is, but not quite. If the interaction is too complicated so that it starts to sound too random to the audience, I think there is always a danger that the excess of randomness could become very monotonous, very quickly. Or, on the other hand, if the interaction is too obvious, it also gets uninteresting very quickly. Jean-Claude Risset, in the above-mentioned issue of *Contemporary Music Review*, comments that 'the enthusiasm for real-time has gone too far', and adds: 'Patterns which are easy to produce through live manipulations tend to become clichés' (Risset 1999).

When working with improvisation and interactive computer music, I also ask myself this question: When do we interact with what we hear in improvisation? Do we listen to every single note that passes? The answer is: I am not reacting immediately to every note, in order to interact with it. All sounds are going into my brain, but that does not mean that I react to all of it. Since I need to be pragmatic in realising musical performances, and in order to mimic my own auditory behaviour, I often use a kind of random pitch tracking: 'Marifiddle'.

2.1.1. 'Marifiddle'

'Marifiddle' (see figure 4) is a patch using the MAX/MSP fiddle~ object, which is combined with

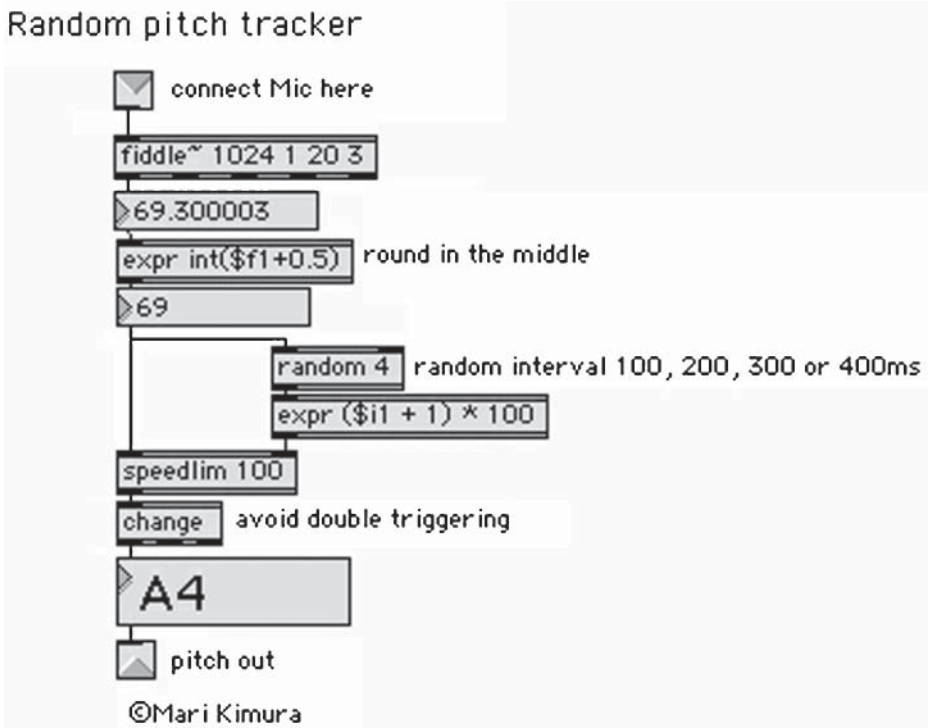


Figure 4. Marifiddle: a random pitch tracker.

https://doi.org/10.1017/S1355771803000268 Published online by Cambridge University Press

several functions to fit one of my particular needs. First, the tracking produces incoming notes, expressed as float numbers, which I round to the nearest integers (expr object). This usually works well, filtering out most of the pitch tracking mistakes. Then, the notes pass through a speedlim object, which receives a random timing between 100 to 400 ms at every note input. Finally, in order to avoid double triggering, the data goes through the change object. Note that this 'Marifiddle' is not meant for all occasions; it will not output repeated notes, for example, because of this change object. I find that 'Marifiddle' works well when I want to track just enough notes. The tracking is slightly unpredictable with four slightly different (100–400 ms) time intervals, while playing fast passages; I would not exactly know which note would be tracked. However, it is usually predictable that the pitch will be tracked, if the notes were played more than 400 ms apart; this works well if I want to trigger something. 'Marifiddle' illustrates the 'fine line' which I described above; neither the audience nor myself would know exactly what is being tracked, but the benefit for the performer is the assurance that the tracking will detect a trigger reliably without distraction for the performer.

2.1.2. 'Random trigger'

Similar to 'Marifiddle', one of the schemes I use often to generate quasi-improvisatory rhythm is 'Random trigger' (see figure 5). In this scheme there are three metro objects connected one to the other, sending

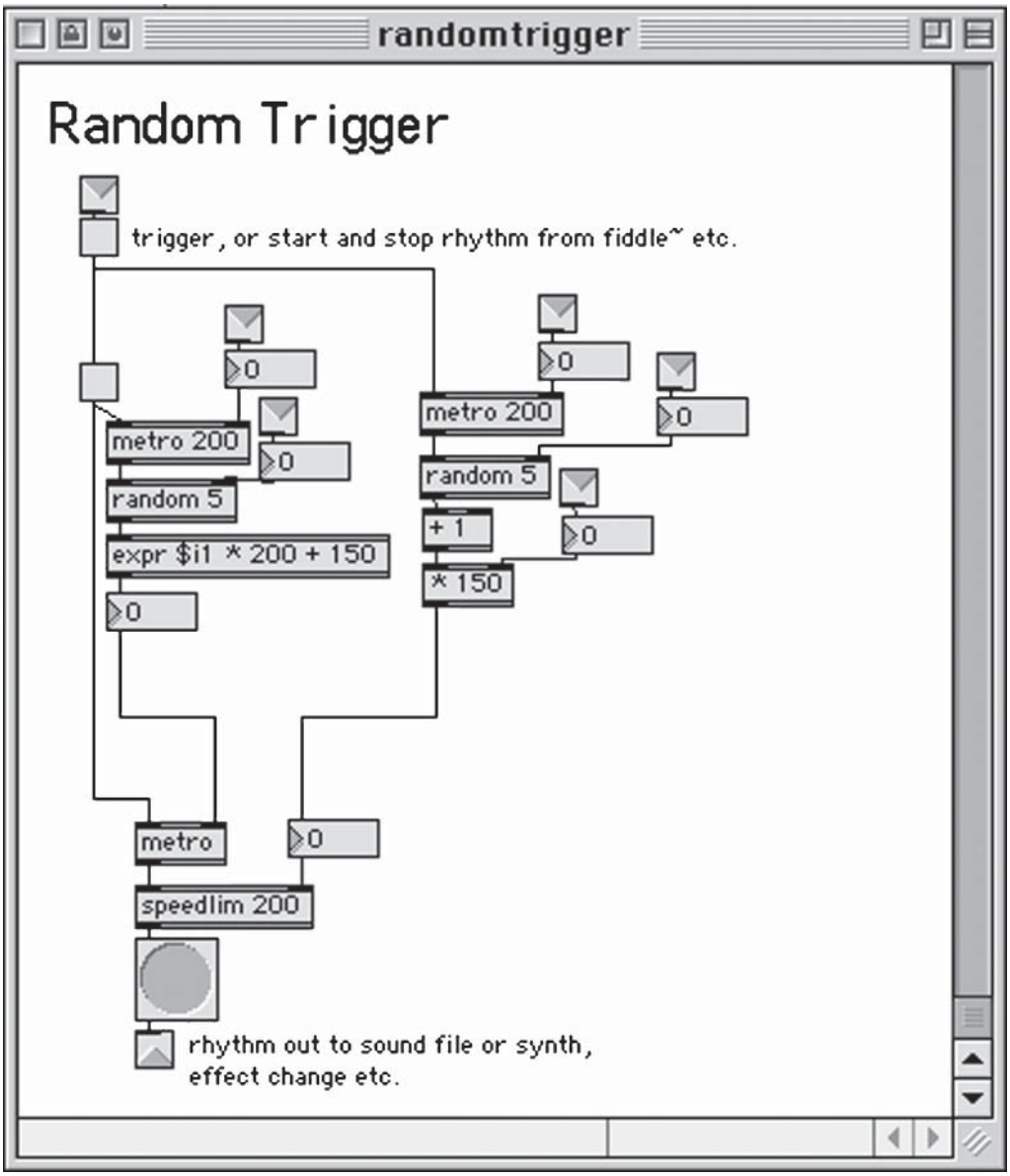


Figure 5. Random trigger: generates configurable random rhythm.

https://doi.org/10.1017/S1355771803000268 Published online by Cambridge University Press

random values into the time in milliseconds. Random trigger is activated and de-activated by a certain pitch, so the interaction is somewhat predictable to the performer but not too obvious to the audience; the listener can tell there is an interaction, but it is difficult to pinpoint exactly what the interaction is. All the random ranges (pitch ranges, speed ranges etc.) involved in this patch are configurable according to the musical context.

2.2. 'Low-tech interaction': adjusting computer for musical needs

Now I would like to discuss the second issue, the basic musical 'common sense' or 'responsibilities' of computers. As a performer constantly facing an audience in real life, I need to achieve the desired musical results pragmatically, even if I have not created the most sophisticated MaxMSP patch in time. My computer system must sound 'as if' it has high-level musicianship, so that my performance does not come across as sounding amateurish. Over the years while performing and composing with the violin and computer, I have come to create MaxMSP patches that are not meant to work for all occasions, but often work only in a 'case-specific' way in terms of musical context. The following two examples illustrate such instances, using rather 'low-tech' methods to serve specific musical needs: (i) making a crescendo, and (ii) intonation issues in computer music.

2.2.1. Making a crescendo in MaxMSP

When I want to make a crescendo in interactive music, I have to ask myself, 'How exactly does one makes a crescendo, and how can I translate it into MaxMSP?' A simple linear increase in amplitude over time, using line~ object, for example, never seems to make a satisfactory, or a 'musical' crescendo. In an acoustic performance, we make a crescendo according to many different aspects of music, including the context, tempo, dynamics, etc. and it is never a uniform, linear increase in amplitude which line~ in MaxMSP would generate. After some trials and errors, I came to use methods that are more 'analogue' in nature; I would draw a rough curve in a table (or function object) by hand, generating a particular curve to increase the amplitude, in order to send to the line~ object. In this way I get the desirable crescendo in the desirable context. Figure 6 illustrates a simple example MaxMSP patch of the variety and the effects of making a crescendo in MaxMSP. The curve (a) is a linear

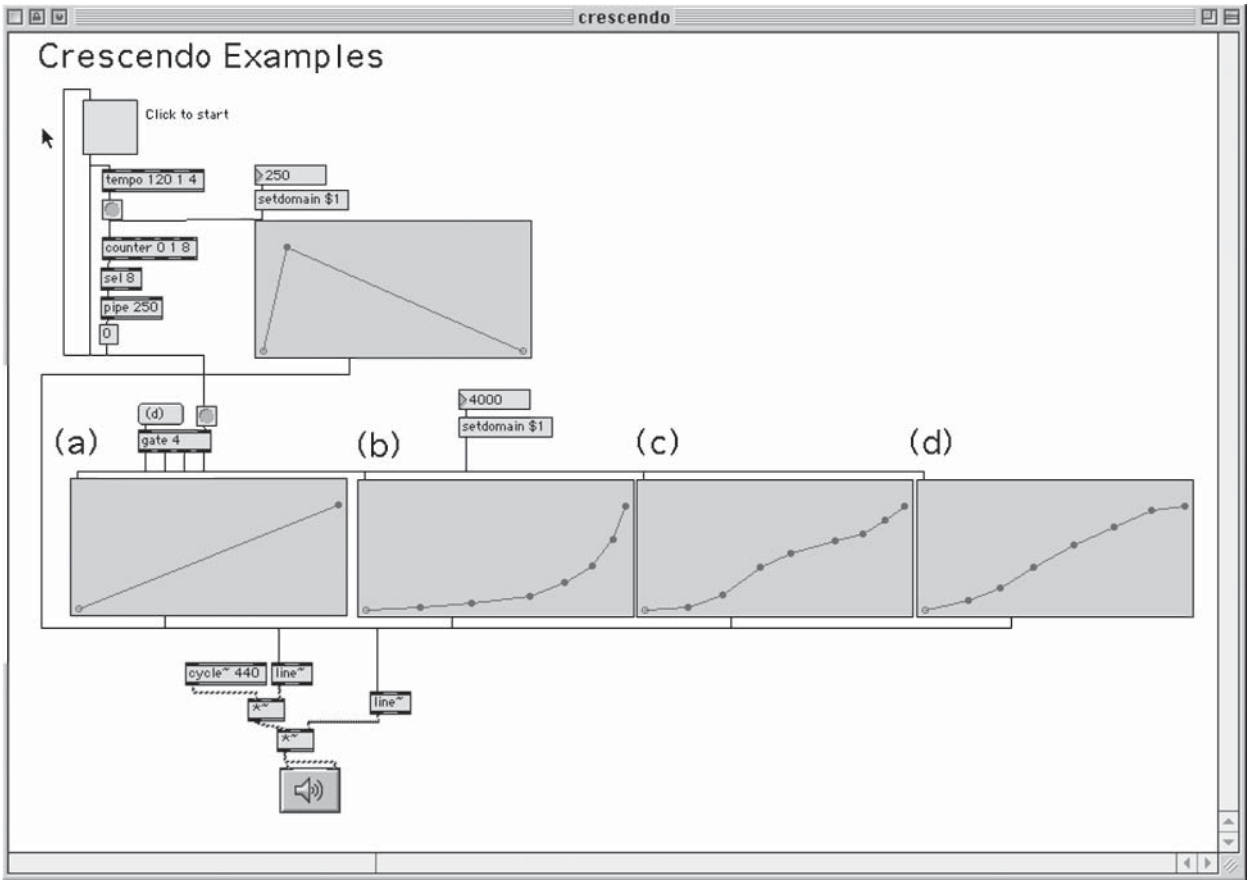


Figure 6. Crescendo curves: 4 different varieties.

https://doi.org/10.1017/S1355771803000268 Published online by Cambridge University Press

increase in amplitude, 0 to 1, over 4 seconds; the curve (b) is a rough model for an exponential (logarithmic) curve, which makes a big amplitude increase towards the end of the crescendo; the curve (c) starts out somewhat exponentially, but then gives a healthy amplitude increase in the middle, then after a small plateau, gives a rather acute increase at the end. I use this kind of curve often myself, which seems much more effective than a linear increase (curve (a)); finally, the curve (d) is a somewhat softer curve, closer to a linear curve, but the increase in amplitude occurs a little faster than curve (a) and then gives a small plateau at the end. This curve is also useful in some cases musically, giving a sense of enough satisfaction towards the end of the crescendo.

2.2.2. Intonation issues in electronic and computer music

When we string players perform with a piano tuned in equal temperament, we adjust our tuning somewhat to it, and the complexity of the piano timbre and its articulations also tend to dampen the intonation discrepancy in an ensemble with the piano. Now whether or not it is because we grew accustomed to hearing ourselves playing 'out of tune' with the piano over the years, being 'out of tune' with synthesised sounds bothers me more. It may be because the synthesised sounds are considerably simpler than any acoustic instrument, thus highlighting the pitch differences. It is often very bothersome listening to pieces for tape and live instruments out of tune with each other, although this issue has not been raised often. I think that some works for tape and acoustic instrument sound amateurish, partly because of this lack of fundamental musicianship: to be in tune. Naturally, even if the human performer tries to adjust to the intonation of the tape part, the tape part is a 'challenged' performer unable to hear. One could argue that the tape part is never meant to be a performer in the first place; however, since all sounds are coming from the stage to the audience, it is all part of a 'performance' if the composer wishes his/her work to be performed in front of a live audience. I think that one of the advantages of using an interactive system and real-time pitch tracking is that the computer has the capability to adjust to the acoustic instrument, and to be in tune on the spot. I simply let the fiddle~ object translate MIDI numbers into frequencies in floating numbers, so now we will not have to be a slave to equal temperament.

3. CREATIVE PROCESS OF INTERACTIVE COMPUTER MUSIC

3.1. 'Sound palette' in computer music

There are probably as many creative processes as the numbers of interactive computer works being created. However, I think that there clearly seems to be a crucial difference in the creative process between writing an acoustic work and writing for interactive computer music. We have learned about the traditional acoustic instruments, at least over the last 300 years. We more or less know what has been done with the particular instrument, and what can be done; in another words, we already have a good 'palette' of sounds as a base to put them together: to create compositions. What we then have to do is only to find our own voice, and 'paint' the picture using the 'palette' of sounds. However, ever-changing (or advancing) computer technology has so dramatically increased the possible size of this 'sound palette' that, although the sheer number of possibilities alone is very exciting, at the same time it can be a little bewildering and hard to keep up. I can offer a few examples of where one could go from here: (i) make intricate, sophisticated MaxMSP effects or functions, and play and compose using the patch; (ii) have a strong musical idea before even touching the computer, and create MaxMSP patches to suit the musical needs; and (iii) compose using a combination of the previous two methods.

The first possibility, to make music using a sophisticated patch, has been typically done by those who are excellent programmers able to come up with such a theory and practice to make an interesting and complicated patch. The advantage is clearly the interesting effect, and one could be musically inspired by the patch to create something that has never been heard before. The disadvantage, as far as I have observed, sometimes seems to be a curiously monotonous musical result. The work would start sounding like a collection or glossary of things one could do using this 'master patch': an experiment rather than a musical work. Also, no matter how complicated or sophisticated the effects or patch are, the audience will be able to 'catch on' at some point, and when they do, that particular effect becomes quickly boring.

The second possibility of having strong musical ideas and making patches to suit the musical needs, is clearly very appealing; usually the results are satisfying since the musical ideas come through to the audience clearly. The downside, or disadvantage as I see it, is that one tends to end up using a rather limited 'palette', since the musical ideas dictate the MaxMSP patch; however, there can always be more musical possibilities at one's disposal, if one experiments with different effects and functions, unrelated to the musical ideas. My experience working in collaboration with a primarily acoustic composer who had not worked much with electronics before, falls into this category. It became apparent that whatever the composer imagined was limited only by what I use, since the composer would tell me the kind of effect he wanted, but his 'sound palette' would be limited to my own 'palette'. I think this also applies to those 'master

https://doi.org/10.1017/S1355771803000268 Published online by Cambridge University Press

composers' who do not or cannot program by themselves, relying on the engineers or programmers they use; these works created by the same programmers and machines tend to sound as if they have the stamp of that particular institution or laboratory; or worse, they sound alike. For some of these composers, this tendency might be a positive 'stamp', showing where they belong and their cultural identity. To the audience outside of that particular cultural frame, however, the composer's musical individuality is not so apparent, but rather where he/she studies or works is at the forefront.

The third method, the combination of the two above, seems obvious to me for producing the most fruitful result. One gets to have a clear musical direction by having a strong musical design, but at the same time, one can expand the 'sound palette' and make his/her musical language in interactive computer music richer. The obvious problem to me in real life using this method is the time taken. If there is a pressing concert date or performance, and the deadline is approaching, there would be some sacrifice to be made, either musically or by limiting the expansion of one's 'palette'. The best-case scenario seems to be to have ample time before the important date, to finish the piece with what you have, then to improve or enrich the effects or details at leisure. This way, one can work within the framework of the musical context, and if one has enough time, work and tweak on the details.

There are a few problems that I have encountered using this third method. First, since the musical ideas and the 'sound palette' go hand in hand, when the 'palette' is expanded one might see different musical ideas emerging. This would create a desire to change the musical structure unexpectedly, and so appears a serious time management problem. Second, I find that it is important not to be too frugal about your 'sound palette'. It is natural to want to use a great patch or effect one has just written, but that particular effect might turn out to be totally musically irrelevant at that time. In order to keep a musical focus, it is sometimes better to be courageous and to put that 'super patch' aside somewhere and maybe use it for the future.

3.2. How to start building an interactive work

As I mentioned above, I work with acoustic musicians who usually have very strong musical ideas even before they come into the electronic studio. It is very interesting for me to watch their creative process evolve as they learn more and more about electronics and interactive systems.

The following questions are the typical ones I would ask my students while building a new work. I use these to give them some starting points, and the answers to these questions usually come from them quickly, since they already know what they want musically.

(1) When and how do you want to start playing or interacting?
(2) When and how do you want to end the piece or the interaction?
(3) When and how do you know you have had 'enough' of one section, and how do you measure this? By numbers of notes, repetitions, or increasing amplitude, etc.
(4) How do you define which section should be the climax or the new material?

4. CONCLUSION

The performance practice issues in interactive computer music performance are so diverse that I could not possibly have covered all kinds of interactive music; I did not include here my observations on ensembles, collaborations in interactive computer music and performance practice issues. As Robert Rowe writes, interactive computer music 'cannot be described as being "in its infancy" or passing through an "experimental phase"' (Rowe 1999). I believe that the performance practice issues must be discussed more in order to benefit both performers and composers in the field. I tried to illustrate in this article some of my own observations from works and performances by others, as well as from my own experiences on stage and from my teaching. I believe it is important at any rate for a performing artist to reflect on his/her 'real life tale', which is extremely important for the creative process in interactive computer music.

REFERENCES

Kimura, M. 1996. Computers for performers, *Computer Music Journal* **20**(4): 25–6. MIT Press.

Lewis, G. 1999. Interacting with latter-day musical automata. *Aesthetics of Live Electronic Music: Contemporary Music Review* **18**(3): 99–122.

Lewis, G. 1986. An excerpt from an unpublished article while working at STEIM.

Kimura, M. 1995. Performance practice in computer music. *Computer Music Journal* **19**(1): 64–75. MIT Press.

Risset, J.-C. 1999. Composing in real-time? *Aesthetics of Live Electronic Music: Contemporary Music Review* **18**(3): 31–9.

Rowe, R. 1999. Aesthetics of interactive music systems. *Aesthetics of Live Electronic Music: Contemporary Music Review* **18**(3): 83–7.

https://doi.org/10.1017/S1355771803000268 Published online by Cambridge University Press