Towards a design of a versatile instrument for live electronic music

Younes Riad

Bachelor's Thesis

Institute of Sonology

The Hague

May 2013

Abstract

This thesis is divided in three parts all related to the design of instruments for live electronic music. In the first part three concepts I have found to be important in this context will be discussed: transparency, physicality and virtuosity. The second part deals with some works of others in the field of live electronic music that have inspired me in my own work. In the third section, I will discuss the work I have done myself in this field that is aimed at both creating an instrument for performance in ensembles as well as playing multilayered music solo.

Acknowledgements

I would like to thank my mentor Paul Berg, for his great support in both writing this thesis, as well as guiding me in creating my instruments and performing them. I would have never be able to come this far without him. I would like to thank Richard Barrett for confronting me with interesting insights about my work. And also I would like to thank all teachers of the Institue of Sonology for creating a very inspirational environment in which I have been able to develop myself as a musician, in which every teacher had his specific complementary role.

Furthermore I would like to thank my parents and my girlfriend for their great support and love during my studies. And Xavier De Wannemaeker and all friends I have met during my studies at Sonology for the inspirational exchange of ideas.

Contents

Introduction

Throughout my life as a musician I have always been interested in performing music live. When I started making electronic music this opened up a whole new set of questions. Performing electronic music requires the design of an instrument. This instrument not only defines how the music is performed but also defines what music is performed.

I have been searching for an instrument that is versatile in the sense that it allows me to perform different kinds of music with it, an instrument that can be used in wide spectrum of musical situations. I have found improvisation to be a solution for an instrument to be versatile. Being able to improvise allows one to quickly adapt to different kinds of musical situations without the need for preparing new material for every new musical situation. If an instrument is designed “correctly” it will allow the performer to play composed pieces as well as improvise in a wide spectrum of musical situations.

In the design of instruments for live electronic music there are a number of aspects that are important to me. These aspects are often automatically present in acoustic instruments but often seem to get too little attention in the design of instruments for live electronic music. There are three aspects that seem to be very important in the design of an instrument and which I will discuss in the context of instruments for live electronic music in this thesis. These aspects are: transparency, physicality and virtuosity.

A transparent instrument is one that allows the audience and possible co-performers to have some kind of understanding of the relationship between a performers' actions and the resulting sound.

With physicality I aim at the use of the body to control an instrument. In my point of view physicality does not imply any amount of use of the body, physicality describes the use of the body where their can be differences in *amounts* of physicality. Physicality is a way to achieve this transparency.

I understand virtuosity in the context of music to be the ability for a musician to control his instrument in an extraordinary way. A virtuosic musician is one that understands his instrument completely and is able to play with all possible sounds and techniques of his instrument with ease.

Another desire for me has been to create an instrument that allows me to perform multilayered music solo. Electronic music allows one person to compose music with a multitude of layers without problems and I have always been interested in doing this live. Considering my desire for a transparent and physical instrument this may be a very ambitious desire or problem. But I have made different instruments in which I have tried several solutions for this problem. My latest solution to date is to create an interactive music system (Rowe, 1993). My realization of such an interactive music system is an instrument that consist of two parts, one controlled by the computer and one that is a kind of “virtual musician” listening to the output of the performer's part and reacting upon that. The intelligence of this “virtual musician” in my implementation might not be fully conform to what the term implies though, since it is merely a solution for the problem instead of an extensive realization of that concept.

A personal view on live electronic music

Transparency

Since the computer is the performer in the case of fixed media compositions, there is no inherent necessity for it to be performed by human performers. But there are numerous of reasons why a piece of electronic music should be performed live. A composition in which elements of improvisation and especially one in which several musicians improvise together, is one of the most obvious examples. But also when a piece of electronic music is part of a theatrical performance that should be synchronized with the theatrical piece an obvious solution would be to choose for live performance of the piece.

When an electronic piece of electronic music is performed solo, on a stage, in a setting that would be similar to one in the case of a performance of an acoustic piece of music, questions about the relationship between the visual performance and the resulting music arise.

A problem with a lot of live computer music is that a relationship between the actions of the performer and the resulting audio can be unclear. In other words, there is a lack of transparency. There is no way of seeing how the performers movements (if any) contribute to the resulting music. This raises the question if there is an actual necessity and thus relevance for a presentation of a piece of music to be a live performance.

Live (computer) music has certain possibilities that do not exist in other ways of presenting a piece of music. Improvisation is probably the most clear example of such a possibility. Without the presence of a certain amount of transparency it becomes quite impossible to tell whether use has been made of the possibilities that live performance has to offer. Furthermore, especially in the cases where together with a lack of transparency there is a lack of physicality in a performance as well – which is often the case – a performance can be a very uninteresting event to witness.

Of course one can think of reasons why one would perform a piece of music without a presence of transparency between the performers actions and the resulting audio. But without a certain amount of transparency this cannot be made clear by the performance itself.

> “Laptop performance—performance on a standard computer system without novel controllers, usually by a solo musician—is an increasingly common mode of live computer music. Its novelty is diminishing, however, and it is frequently described as being uninteresting. Laptop performance often lacks the sense of effort and active creation that we typically expect from live music, and exhibits little perceivable connection between the performer’s actions and the resulting sound. Performance software designs tend to constrict the flow of control between the performer and the music, often leading artists to rely on prepared control sequences. Laptop performers become “pilots” of largely computerized processes, detracting from the live experience.” (Zadel, 2006, p. 3)

As pointed out in Zadel's Master's thesis quoted above, we find a lot of performances where the performer controls his (computer) instrument only by means of a computer keyboard and mouse in more popular electronic music, where in most cases the performer triggers prepared musical material which can have any possible level of complexity. This is maybe one of the best examples of a problematic situation in terms of transparency. Since the keyboard and mouse are very generic controllers which are used for many different tasks within the computer, they do not reveal what is happening on the computer. Using these controllers in the way just described also gives no clue about the relationship between the performers actions and the resulting music.

Popular electronic music, is in most cases created by one person. Pieces of popular electronic music mostly contain musical material for more than one instrument (which in the case of an electronic piece often is a synthesis process). The creators of these pieces often want to perform there music live by themselves. But because the pieces contain musical material for multiple instruments, it is impossible for one person to play such a piece of music completely live (in which case one would have to play all separate instruments in real time). To still be able to perform this music, often the choice is made to trigger prepared musical material. In my opinion this kind of performance should not be called live performance but rather semi-live performance. Because the material is not actually performed live, it is prepared and what is left for the performer to influence live is only the time at which such a piece of prepared material is played back. A practice that has more similarities with DJ-ing or remixing, rather than actual live performance. Therefore, the I'd like to classify it as semi-live performance. Still this kind of performance is generally referred to as actual live performance. Which makes it serve well to be used as an example of a problematic performance situation in terms of transparency.

> While the shift toward software tools is appealing for computer music production, it leads to frequently unsatisfying laptop performance for the following reasons. First, complex, layered pieces are generally hard for a single performer to play. Performance compromises must be made to execute such pieces since our cognitive and physical capabilities limit the amount we can do and keep track of at once. Second, the tools that performers use to play their music live are not necessarily well suited to real-time use as they draw on a heritage of studio-centric tools. (Zadel, 2006, p. 7)

I would not say studio-centric tools could not serve a live performance of electronic music. There are ways in which they could be used as real-time instruments, although they are often not the most convenient choice. As Zadel points out in the quote above, in order to play a complex, layered piece of electronic music solo, compromises have to be made. From this could also be concluded that if one wants to play an electronic piece of music solo and *live* without compromises, one should not desire to play (in particular) a layered piece of music. Since this tends to be physically impossible, one should rather focus on one layer.

Making a performance more interesting to watch, or clarifying why one chooses to perform a piece of electronic music live on stage are not the only reasons for a desire towards a transparent performance. When playing with others transparency between actions and audio can serve as a means of communication, of course in addition to the sonic communication which is already established by producing audio. Transparency, often established by a physical relationship between actions and audio, can expose something a performer is about to do. Which gives a co-performer the possibility to anticipate. This process can also serve as a way to make the communication between performers on stage more transparent which might make it more interesting to witness. Especially in the context of free improvisation this can be the case, being able to see performers on stage communicating might make an audience more involved. Where in composed music the decisions made by performers are obvious (they are written in a score) it might also be interesting to get more insight on this in the context of free improvisation (especially for the untrained spectator).

Angel Faraldo discusses transparency in terms of parametric understanding of a performance system.

> Though «explanations» might be useful for analytical purposes, I believe that the act of listening to music should in no way be parametric. Such an approach, though functional in that it may open a path to «think» music differently for the computer performer or instrument designer, should never be perceived as a distinctive quality of the music.
> (Faraldo, 2009, p. 24)

Assuming transparency is about giving the audience an understanding of the technicalities involved in the instrument performed, I would agree with Faraldo, but I do not agree with that assumption itself. I do not think transparency is about technical understanding per se. I think clarifying the technicalities of an instrument can definitely help in making it transparent, but it is not necessary for an audience to have complete understanding of these technicalities in order to have an understanding of how the performers actions relate to the resulting music. Taking for example, an acoustic instrument I think is transparent but of which the bigger audience does not exactly know how it technically works: a saxophone. There is no need for an audience to understand which different blowing techniques cause what audible result to have a good understanding of how the visible actions of the performer relate to the resulting sound. For a transparent performance it is enough to see a physical action and being able to connect that to an audible result, there is no need to understand the exact technicalities involved in this relationship between action and audible result.

Transparently boring

I would say the most transparent way of using controllers in general is by mapping them in a 1 on 1 manner, using them for low level control for instance. This can be done in many ways. For example by using a musical keyboard for triggering percussive samples. But also a certain gesture made with an accelerometer-based controller mapped to a particular musical event can be very effective. Technically this is not a very one-to-one way of controlling musical parameters but from a musical point of view it might be.

> An instrument which is too simple may seem toy-like and may only have a limited musical potential. Therefore, effort must be made to ensure that these instruments offer 'No ceiling on virtuosity' (Wessel & Wright, 2002). One method of doing this could be the use of complex mappings such as those discussed by Hunt (2000). By having some simple musical functions (such as pitch selection and modulation) controlled by gestures determined using the methods described here and more complex musical functions (such as timbral modifications) controlled by the interaction of various gestural parameters it may be possible to create an instrument which is both easy to use and also offers mucmh potential for musical creation and exploration. (Marshall, Hartshorn, Wanderley, and Levitin, 2009, p. 252)

The quote above illustrates the danger of making an instrument too transparent. If a certain gesture from the performer always result in the same sound it will become obvious after seeing the performance a couple of times. This can make a performance predictable. Changing simple one-to-one relationships between the performer's actions and the resulting audio during a performance can solve the predictability but this would make it very hard to maintain a minimal amount of transparency.
The use of complex mappings as discussed in the quote above and by Wanderley (2001) on the other hand seems to be a rather effective way to overcome predictability while still being able to maintain a substantial amount of transparency. A balance has to be found between a more complex mapping strategy on one hand, and a transparent an intuitive mapping strategy on the other in order to create an instrument which allows the performer to perform in a transparent but non-predictive way.

Physicality

Physicality – in the way I would like to use the term – does not necessarily imply big gestural movements. I would like to use it in a very basic way, to describe the body is used to achieve a certain goal, in contradiction to the mind or a computer. In this text I will use the term in relation to any kind of "body use", to describe little physical movements as well as big gestural ones, both extremes having their own *amount* of physicality.

As has been pointed out by Bahn, Hahn and Trueman (2001) in their article about physicality and feedback, one of the great distinctions between electronic and non-electronic musical performance is the little role that the body has played in the creation and performance of electronic music.

> Musical performance in a cultural context has always been inextricably linked to the human body, yet, the body has played only a minor role in the creation and performance of electronic music. (Bahn, Hahn, Trueman, 2001, p. 44)

Since the publication of the article quoted above, a lot has in changed in the performance of computer music. Namely that technology has become widely available and studio-equipment has become more affordable. This may have contributed to the fact that since 2001 the use of the body in the performance of electronic music has become more common. But still today in 2013, it is anything but certain that a performance of electronic music will contain a substantial amount of physicality to produce the music heard. The amount of performances without a necessity for physicality in order to produce music outnumber the performances that do have this necessity, especially in the field of popular electronic music.

Physicality can be a cause of, but does not necessarily result in, transparency. But as I have been pointing out earlier in this text, I think physicality in the performance of an electronic music is the best (if not the only) way to achieve transparency. Physicality in live electronic music can best be described as having physical control over the music (using the performers' body), where the amount of physicality depends on the amount of entanglement between the performers physical movements and the music.

This can be done in many ways, often with a different result in the amount transparency.

> Instead of musing on whether a computer is precise, reliable, or capable of invoking the human touch in music, you should focus on the notion of effort. Effort is something abstract in a computer but a human performer radiates the psychological and physical effort through every move. The creation of an electronic music instrument should not just be the quest for ergonomic efficiency. You can go on making things technically easier, faster, and more logical, but over the years I have come to the conclusion that this does not improve the musical quality of the instrument. I'm afraid it's true one has to suffer a bit while playing; the physical effort you make is what is perceived by listeners as the cause and manifestation of the musical tension of the work. (Waisvisz in Krefeld, 1990, p. 29)

A physical action automatically involves effort. Differences can be made in the amount of effort as well as the transparency of it. The more physicality involved in a certain action the more effort needed. Therefore I will write about amounts of physicality only as this implies effort.

I will give a few examples of different controllers which offer a certain degree of physical control and the transformation of this physicality to audio.

First of all, there is the previously discussed computer keyboard and mouse control setup. Although I would classify this setup as one of the least physical control, clearly there is a certain amount of physicality offered by these controllers which can not be neglected. But the amount of detail in the movement of the performer which can be conceived which these controllers is extremely low. In particular in the case of the computer keyboard. Besides the latency it might induce, it is not capable to track anything besides which key is pressed and at what time. Nor the velocity at which a key is pressed or the pressure used when held down can be tracked. The computer mouse can be used in a more physical way, although this is not much done because better solutions exist. From the movements made with the mouse one could extract different physical parameters such as speed, velocity, and the visual pattern 'drawn'.

Secondly, there are the common MIDI controllers containing sliders and knobs. These controllers imply some physicality by themselves. When sliders are sitting next to each other on a controller, more of them can be controlled at once, this is an advantage over for instance the control of parameters on a screen with a mouse. Knobs are harder to be controlled simultaneously. But also imply more physicality than controlling parameters on a screen with a mouse. By clever use of sliders and knobs some substantial amount of physicality can be achieved. Especially deriving higher level parameters from these controllers can be interesting. For instance measuring speed and velocity from fader movement.

An even more physical control of live electronic music can be achieved by the use of buttons, in particular musical keyboards and pad controllers who often have the ability to provide the user with parameters for the velocity at which a knob is pushed and the pressure with which it is held down. (Regarding musical keyboards versus pad controllers, I myself prefer the latter, since it implies no particular relationship between the buttons while a musical keyboard does: the twelve pitches of the western twelve-tone equal temperament.) When these controllers are used in a direct way, for instance by making them trigger an event when pushed, a physical relationship between action and audio becomes clear.

And then there are the controllers that require a substantial amount of physicality to be operated. I am aiming mainly at joysticks and accelerometer-based controllers. These controllers serve best for an achievement of a transparent relationship between a performers actions and the resulting audio. Of course there is still the question how these controllers should be used to reach a certain amount of transparency. While the controllers require physicality to be operated, clearly there are many ways in which they can be used without achieving transparency.

Virtuosity

> One way to define instrumental virtuosity would be to say that it is the ability to instantly access any technique, sound, note, fingering, or timbre available on one's instrument. It is the facility to move between loud and soft, high and low, noisy and clear, rich and thin sounds – at will and at any moment. A trumpet virtuoso, for instance, can move between a crystal clear note in the highest register to a subtone to a split tone in an instant and with total ease.
> (Pluta, 2012, p. 3)

Of course virtuosity is not restricted to the field of music. Every artistic and cultural discipline has its virtuosi, as well as every profession does. And every human activity that is done in an extraordinary manner is in a certain way virtuosic.

The circus is a phenomena which I understand to be all about virtuosity. People pay to see other people performing extraordinary actions or performing less special actions in an extraordinary way. And also in sports we see this voyeuristic behavior. For some reason still unknown to me, people like to watch other people doing extraordinary things.

Although it definitely is related to those practices and shares characteristics with them, a musical performance is not, and in my opinion should not be a circus-act, nor a sport. Although there are numerous musicians who's performance mostly consists of showing off there skills, this kind of fetishism is rarely musically interesting. Often a virtuosic action is understood to be a physical action. But especially in the case of music, this is not necessarily true. Timing for instance, which maybe mostly related to musicality, is something that allows a great amount of virtuosity as well.

Musicality is in my opinion the most important skill a musician has – not to neglect the possibility for one to be virtuosically musical of course. But musicality in itself does not make a musician. A means to express ones musicality is required, an instrument which allows one to produce sound (which could be ones voice) and the skill to play this instrument. The better the skill, the better one is able to express his musicality. But I think it is of great importance that virtuosity is only used where musically necessary. A guitarist playing notes as fast as he can just for the sake of it is most often a very uninteresting matter.

Especially in the field of live electronic music virtuosity is something which is often lacking in the performance of this music. There can be multiple causes responsible for this problem, but mostly, they are related to the design of the instrument that is used to perform the electronic music live, or the process of designing that instrument.

One of the most frequent causes is the instrument designer that keeps on altering the functionality of the instrument. This makes it hard – if not impossible – to practice with the instrument. Another cause of a lack of virtuosity in the performance of live electronic music can be the complexity of the instrument used. Especially in the design of electronic instruments one easily falls into the trap of making an instrument too complex. This often is the result of the desire to have an instrument which is capable of “everything”. A desire which is not very realistic in the first place but moreover, almost inevitably results in an instrument which is so complex it is too hard to control in an intuitive and thus virtuosic way. Not so say that intuition is virtuosity or automatically results in virtuosity, but I think without intuition, virtuosity is impossible.

In order to make a successful electronic instrument it is necessary to have a clear and realistic goal for the capabilities of the instrument in terms of sound. And maybe the most important factor to create an instrument which allows one to develop virtuosity in playing it is the way the instrument is controlled. For instance making use of several virtual “pages” on a MIDI controller (a feature which is found in most MIDI controllers nowadays) can make it very hard to intuitively navigate through the instrument. Instead, one better chooses to limit himself to only one page of a controller, which would result in having less parameters that can be controlled, or choose to add more controllers to the instrument.

Intuition is an important factor for virtuosity. I think a virtuosic performer is one that is able to intuitively apply techniques on his instrument in order to express himself. Thus without having to consciously think about those techniques.

In order for an instrument to allow intuition requires two important aspects:

- *The instrument should be controlled in a musical way:*

  An instrument in which control 'musical' aspects are controlled, by use of one-to-many and many-to-one mapping techniques for instance – a subject I will return to later in this text. Controlling the intensity of a sound with one parameter that controls different technical parameters like loudness and distortion, instead of controlling those technical parameters separately. This would be a way to avoid controlling pure technical aspects and achieve what I understand to be more 'musical' control.
- *The instrument should able it's player to practice it:*

  An instrument that is constantly changing can not be practiced. To able it's player to practice an instrument should not be developed further for substantial periods of time. After sufficient practice, relationships between actions and resulting sounds will become part of a performers muscle memory, which is one of the most important factors in intuitive performance.

The choice of controllers for an instrument is very important. Not only in terms of physicality and transparency but also in terms of virtuosity. A physical way of controlling an instrument allows one to develop virtuosity by making cognitive actions part of ones muscle memory. By physical control of an instrument I am pointing at the use of the body in controlling the instrument. Not to confuse physicality with the qualities of a sound, as sounds are sometimes described in terms of physicality.

Wessel and Wright (2002) suggest in their discussion of intimate musical control of computer instruments that an instrument with no ceiling on virtuosity and a low “entry fee” is possible.

> Is a low ‘‘entry fee’’ with no ceiling on virtuosity an impossible dream? We think not and argue that a high degree of control intimacy can be attained with compelling control metaphors, reactive low latency variance systems, and proper treatment of gestures that are continuous functions of time. With the potential for control intimacy assured by the instrument, musicians will be much more inclined to develop their performance skills and personal styles.(Wessel, D. and (Wessel & Wright, 2002, p. 12)

By avoiding one-to-one control mapping strategies and making use of many-to-one or one-to-many mapping techniques one is able to control musical parameters instead of technical ones. Hunt, Wanderley and Paradis (2002) discuss this topic extensively. In this article the authors show how changing the mapping for an electronic instrument can completely change the possibilities of it's outcome and influence the ease at which a certain musical gesture can be made. They also show, that with more complex mappings, learning to play the instrument does not come as easily as with one-to-one mappings but there is a more or less constant increase in skill. While learning to play an instrument with one-to-one mappings comes almost instantly but increasing skills on the instrument is very hard and more complex musical tasks are almost impossible.

Although a lot of emphasis in performance of western music is on virtuosity, I do not think it is the most important aspect of a performance. Musicality for example – which is not necessarily connected to virtuosity – plays a much bigger role. But in most cases, without skill, one is not capable of expressing that musicality. What I'd like to make clear is that virtuosity is by no means a replacement or a way to musicality and virtuosity without musicality seems completely pointless to me.

> Concert music is still a vital part of our musical life. To ensure that it remains so in the decades to come, despite drastic changes in style and technological resources, we must reflect on why people go to concerts at all. We suggest that one of the significant aspects of live performance (from the point of view of the audience) is virtuosity, and that there is a danger that perception of virtuosity may erode as new technologies are applied to musical performance (Schloss & Jaffe 1993).
> (Jordà, 2005, p. 96)

The fear for virtuosity to "erode" due to new technologies applied to musical performance seems not realistic. I can understand though there was reason to believe so in 1993, since at the time real-time computer music was still quite new and performances might have been striking only because of the technology involved. But today, in 2013, this is not the case. Since technology is widely available it is relatively easy to setup a simple computer instrument – a sampler for with some dynamic control over pitch and volume for instance – it becomes more important what one does with an instrument. We can see an example of this "sampler-virtuosity" in recent popular music.

We live in an age of finger drumming virtuosos, where drum pads are instruments. And so, while much of production is anything but real-time, here it makes perfect sense for three tracks to have accompanying live videos. The songs are each performances, something to be seen as well as heard.
(Kirn, 2012, p. 1)

Improvisation

Improvisation is a method of acting which is very widespread in all kinds of human activity. In his book *Improvisation: Its nature and practice in music*, Derek Bailey mentions Einstein's idea that improvisation is an emotional and intellectual necessity (Bailey, 1992). Improvisation is part of everybody's daily life, often unconsciously, but almost inevitably. A conversation between two people is a perfect example. Often both participants in a conversation have not prepared the conversation, moreover mostly they even did not know the conversation would take place. Therefore there is no other choice but improvising. Acting and reacting in the moment, in real time.

As well as improvisation is a very common method for doing everyday things, it also is a very common method for making music. The first music made by man could not have been anything but freely improvised (Bailey, 1992).

George Lewis describes the field called improvised music which has become prominent since 1970. He identifies improvised music "as a social location inhabited by a considerable number of present-day musicians coming from diverse cultural backgrounds and musical practices, who have chosen to make improvisation a central part of their musical discourse" (Lewis, 1950, pp. 234).
According to Lewis, this enables individual improvisers to "reference an intercultural establishment of techniques, styles, attitudes, antecedents, and networks of cultural and social practice".

Derek Bailey makes the distinction between idiomatic and non-idiomatic improvisation (Bailey, 1992). Idiomatic improvisation being mainly concerned with the expression of an idiom – such as jazz, flamenco or baroque – while non idiomatic improvisation is not, it is often referred to as free improvisation.

In the following text, I will be discussing non-idiomatic improvisation, unless noted otherwise. Richard Barrett (2012) explains the difference between free improvisation and other strands of improvisation to be that the framework in free improvisation does not exist before the music is created but rather comes along with the creation of the music. It is a kind of relativistic framework instead of a fixed one.

Improvisation and composition

Improvisation – idiomatic as well as non-idiomatic - and composition are very closely related things. In fact, as Richard Barrett (2012) states, improvisation is a method of making music, not a style, and thus a method of composition. He defines composition as the act of creating music regardless of the method used.

There are events characteristic to many non-realtime composed pieces of music that are very unlikely to happen in a free improvised piece of music, clearly. Without the use a common meter the chance that two improvising musicians play a note at the same time is quite small for example. The chance they both play a different note of the same chord at the same time is even smaller. But improvisation has its own unique features. As Barrett states, it is very unlikely for someone to think of the improvised piece of music when thinking it out in a more traditional sense of composing. Improvisation can give one the possibility to come up with a music which otherwise would have never come to the composers mind. Furthermore, playing together with others can create more than the sum of its parts. A successful attempt of a collaborative improvisations is one where the improvisation brings the musician to a place he could not have reached on his own.

To make further discussion about two methods of composition – real time and non-real time composition – more easy, I will stick with the most common terminology for these two methods, improvisation and composition respectively. A good way to distinguish one method from another is to define them in terms of their difference, which was beautifully described by Steve Lacy:

> In 1968 I ran into Steve Lacy on the street in Rome. I took out my pocket tape recorder and asked him to describe in fifteen seconds the difference between composition and improvisation. He answered: 'In fifteen seconds the difference between composition and improvisation is that in composition you have all the time you want to decide what to say in fifteen seconds, while in improvisation you have fifteen seconds.
> (Cited by Bailey, 1992, p. 141)

Of course the differences and similarities of improvisation and composition are far more nuanced than what is described in the quote above. Not to forget the occasions where improvisational aspects are used in composed pieces, and compositional elements in improvisational pieces. With the evolution of time more often musicians are asked to improvise by a composer, in a particular section of a composition for instance. Rules and guides are used in both idiomatic and non-idiomatic improvisation.

Improvisation may be one of the most obvious reasons to choose to perform an electronic piece of music. If human improvisation is required, or human improvisation is a goal in a piece of electronic music, one has no choice but performing the piece live.

In the case of a composed piece of electronic music as opposed to an improvised one, the choice to perform the piece is not always a necessary choice. For the obvious and maybe cliché reason that a human can never reproduce a composition as intended by the composer more accurately than a computer. Which is something that can be done perfectly in electronic music, composing a piece to be exactly performed as it was composed. Not to state that live performance of a composed piece of music is pointless. As mentioned earlier I think there are numerous reasons why one could choose to do so. But when there is a need for human improvisation in a piece of music, one has no other choice but performing the piece live.

Improvised electronic music

In electronic music, the meaning of improvisation and its contrast to composition becomes even more ambiguous. In most cases the improviser plays either a self-invented or specially-tailored instrument. In the creation of such an electronic instrument, one is absolutely free to pre-determine certain musical aspects of an instrument.

Because the amount of pre-determination is up to the designer of the musical instrument, the amount of actual control a performer has on the output of an electronic instrument may vary and not always allow an instrument to be able to improvise. As Richard Barrett (2012) states, an improvised performance that works is an improvised performance in which the musicians not only make nice sounds but also nice relations between those sounds. Relations between sounds are pretty easily made by means of timing, but so much more aspects of a sound can relate in some way ore another that only being able to create them by means of timing would be a great miss.

In the case of a sampler, where one has for example control over which sound to play, when to play it, at what playback speed and with what volume, one has already a pretty capable instrument. The choice of what sounds one has to play with is very important in this case, since the options for shaping or transforming the sound are pretty limited. The sounds stored into the instrument determine what the instrument will sound like. The instrument is not able to change the range of possible sounds during the performance of the instrument, but with the right set of sounds and the ability to access them easily one is able to create themselves a very versatile instrument.

In the case of the example just given, the amount of improvisation may be arguable, because the improviser playing on such an instrument is not able to produce musical material outside of the scope of what was prepared. But maybe this situation is comparable to performing with an acoustic instrument. Performing with a guitar for instance. A guitar also has a certain sound, the sound of the strings, which will always sound like strings in one way or another. As well as he is restricted to a range of possibilities, usually the twelve notes of the western well-tempered scale. Of course the boundaries in both mentioned restrictions can be stretched quite extensively, as guitar players have clearly proven, but only until a certain extent.

A look at a context

Solo performance

Introduction

There are several artists who play electronic music live and solo, that have inspired me to make my work, and influenced me in the choices made in the process. I would like to take this opportunity to discuss a few.

Tarek Atoui

Tarek Atoui is an artist that struck me with his performance, especially the physicality and the transparency it causes is what makes his performances very appealing to me.

From what I have been able to see in movies of his performances found on the internet, I have been able to get some insight in the design of his instrument. My findings are based upon two pieces of film, found on the internet. The first (Atoui, 2009a) shows what seems to be a rather early version of his instrument. Only two commercial controllers are part of it. In the second video (Atoui, 2009b), self-made controllers consisting of different kinds of sensors are mostly responsible for the hardware part of his the instrument. Since the second video is a more recent one, it will be the video I am referring to when writing about Atoui's instrument.

Atoui uses a multitude of self-made controllers, basically (groupings of) sensors connected to the computer. These sensors are in themselves not very complex, in fact, they seem to be quite simple. It is the fact that there are quite a few of them, widely spread out on a table, that makes his performance very physical and interesting to watch. Atoui makes use of faders, rotary knobs and sensors that imply a more intense physical way of interaction like pressure and light sensors.

The way Atoui has placed his sensors on the table he uses to perform, is very important for the physicality of his performance. But especially the fashion in which he plays those sensors, very rapidly changing from one to another, is what makes the performance such a physical one. This necessity for the quick and reasonably intense movement to interact with the sensors is a result of his esthetic preferences in conjunction with the fact that the sensors do not require complex interaction and are placed widely on the table. Aiming towards a physical performance, it seems to be an effective choice to have simple interactivity with a controller, by deliberately not making use of complex mappings.

The amount of physicality and effort involved when Atoui performs his instrument, is exceptional in the context of other performances of live electronic music. His body is almost constantly in motion, rapidly moving from one side of the table to another. But it also seems a decent amount of theater is involved, movements seem to be exaggerated, in the same way an enthusiastic guitar player in an energetic rock concert would do. Something extra-musical, and a valuable contribution to the performance. Which I think is interesting, also for the fact it results in a more transparent performance, since the clarity of the movement makes it more easy it relate it to a particular event in the music.

By studying the chronologically first of the two videos and comparing it with Tarek Atoui's recorded work, which is publicly released on CD (Atoui, 2008), it has appeared to me Atoui is playing live sound on top of what seem to be fairly long pieces of static pre-recorded material. A substantial part of the music in the first 40 seconds of the video is completely identical to the part between 0:07 and 0:47 minutes in the second composition on the CD. At this moment it can be seen in the video, Atoui does something with the mouse of his laptop after which the correlation with the CD disappears. Which is a sign he has chosen a new large pre-recorded piece to play with.

It was a disappointment for me to find out the music in Atoui's performance was not completely produced in real time. It does not contribute to the “live-ness” of the performance, simply because the inner composition of pre-recorded material is not in any way related to any event happening in real time during the performance. And in general, I think to play back long pieces of pre-composed material (longer than 20 seconds) that are left uninfluenced by the performer, has a negative result on a performance in terms of its transparency. Unless an audience is informed beforehand, it will be presented a performance in which a part of the music does not have a clear cause, and only a guess can be made about its origin. But because of the clarity of the relationship between Atoui's movements and the sounds they cause, it also quite clear what part of the music he is controlling directly, and what is related in another way.

The solution for playing multilayered music solo is one that for me, is not that interesting in itself. But I think that is exactly why it inspired me in my own work. Since the performance as a whole and the esthetic qualities it has is very appealing to me, but on the other hand the methods used are not conform to my ideas of electronic music, it triggered me to think about other solutions to the problem.

Jeff Carey

Jeff Carey is an artist that has performs his music in a very physical way as well, but mostly, plays single layered music.

Carey's music has a very specific sound world, one that I would describe as intense both in terms of spectral and rhythmical content as as well as loudness. The sounds in Carey's music are generally dense and have a broad spectrum, a sound world that causes certain expectations for the way it could be created. Since the sound is generally very intense, one might expect a cause for that sound that is intense as well, controlling the instrument in a way that is physically demanding for instance. And the way Carey performs his music does match this expectation to a certain extent. Carey uses a joystick, a pad controller and an ergonomic keypad normally used for gaming for the hardware part of his instrument. Recently he has added a small controller with faders and rotary knobs to this setup.

I have attended a short lecture he gave about his instrument at STEIM in Amsterdam (Carey, 2011). There, I understood he uses the keypad to switch between different synthesis processes, which he controls with his joystick and pad controller. Both the joystick and pad controllers require physical action for them to be controlled, but the mapping is ultimately responsible for the amount of physicality necessary to control the instrument. What matters is how physical properties of an action are measured and mapped to audio processes, measuring the speed at which the position of a joystick is changed and using that data to control the instrument for instance.

Looking at the video (Carey, 2012) it seems to appear Carey has chosen to use a reasonably direct way of mapping his controllers to the software part of the instrument. Not that he necessarily uses a one-to-one way of mapping (Wessel & Wright, 2002), but it is for instance clear a movement with the joystick causes a movement in the sonic outcome of the instrument. The intensity of the movements made by Carey, and the intensity of the sonic result, are on the other hand not very strongly related. A calm movement seems to be able to produce a calm, just as easily as an intense, sonic result. This non-correlation between intensity of movement and intensity of sonic result is most probably caused by the fact that Carey chooses to control higher level parameters of dynamic synthesis processes that can have any arbitrary sonic intensity.

The choice of two of three of the controllers used in Carey's instrument is very similar to mine, a joystick and a pad controller. I had however already put together these controllers before I came to know about Carey's work. But the similarities in choice of controllers is maybe not that accidental, they are maybe objectively just effective controllers for live electronic music.

Michel Waisvisz

Michel Waisvisz was well known as a performer of electronic music. For these performances several instruments for electronic music were made. These instruments were controlled in a physical and/or gestural way in most cases. Amongst other notable instruments – the well known Cracklebox for instance (Waiswisz, n.d) – he made an instrument with the help of several technicians and Frank Baldé, The Hands, which maybe could be seen as his “masterpiece”. With The Hands, Waiswisz gave striking performances in terms of both physicality and transparency, both solo and as part of ensembles.

The Hands made use of two custom controllers, connected to the hands in an ergonomic way, almost like gloves, in order to control the software (or in earlier versions, synthesis) part of is instrument. The Hands (Waisvisz, 2006), described the instrument to be a result of a search towards an instrument that allowed one to 'touch' sound.

> I wanted to operate, navigate, compose, mold and play sound in a sensible, refined and even sensual and groovy way! Something that was, at that time, not associated with electronic music at all!
> (Waisvisz, 2006)

The first version of the instrument consisted of two 'hand controllers' controlling a Yamaha DX7 by means of MIDI. It was premiered at the Amsterdam Concertgebouw in June 1984 after which several new versions of the instrument came into existence (Waisvisz, 2006). In a later version, the 'hand controllers' consisted of buttons underneath the fingers and ultrasonic sensors to measure the position of the hands, from which gestures could be extracted. The software part used for the instrument was a commercially available piece of software called LiSa (Live Sampling), developed at STEIM, by Frank Baldé. LiSa is basically an advanced sampler that can be controlled with MIDI.

By making use of live-sampling, the technique of recording material in real time and using that recorded material as source material to be transformed, Waisvisz was able to demonstrate the technical functionality of his instrument. In the way he used live-sampling, he was able to show the audience how the physical movements made related to resulting sound and sometimes even, the technical process used to obtain this resulting sound.

The use of an ultrasonic transmitter and receiver in order to measure the position of the hands in space, seems a rather effective one. I have experimented with Wii controllers, which allow similar control strategies as The Hands do. A big problem with using Wii controllers, or generally controllers that can be moved freely in space and from which there position is tracked, is that there is no physical point of reference. In comparison: a fader has an absolute position for it's minimum value, as well as for it's maximum and it's middle value. With Wii controllers, the problem is that this point of reference is relative to a point at which the Wii controller is calibrated, since the position is measured by doing calculations with the output of accelerometers. This problem is solved by using ultrasonic transmitters and receivers: if a receiver is statically positioned at a point in space, the distance between it and a transmitter will always be absolute. A problem that remains unsolved is that the point of reference is not physical but rather a point in air that is not easily marked (one could think of a wire hanging on a ceiling but this does is not ideal), which makes it hard to keep track of this reference point.

Matthew Ostrowski

Matthew Ostrowski's live performances are appealing to me because of their physicality and the transparency that it results in, which is maybe a more abstract kind of transparency than is the case with the artists discussed earlier in this text. Ostrowski makes use of a glove that he wears on his right hand which is able to track the position of his hand in space as well as the movements he makes with his fingers. The software part of his instrument is a MaxMSP patch. In his left hand, he holds an iPod which he uses to navigate through his software environment.

Different from Waisvisz, who also uses equipment to measure movement and position of his hands, is the use of only one hand instead of two, and especially the lack of relationship of physical properties of the hands. Something that makes Waisvisz use big gestural movements to control his instrument. The gestures Ostrowski makes in order to control his instrument are smaller and a little less theatrical, but both Waisvisz' and Ostrowski's performances are in a way very similar due to the use of gloves.

The software of Ostrowski's instrument consists of a multitude of sub-instruments that he plays one at a time. These instruments all have there own synthesis process and way in which Ostrowski's controllers are mapped. The use of multiple sub-instruments could reduce the amount of transparency electronic music performances. Because of the sub-instruments, more than one relation between the performers actions and the resulting sound exist. But Ostrowski has created quite a clear relationship between the movements he makes with his glove, and resulting sound. He has mapped his glove in such a way, that the amount of intensity of resulting sound relates to the amount of intensity of his physical movements. A relationship that seen from a technical point of view is more abstract, since there is no direct relationship between a movement and a specific parameter in a synthesis process for instance. But in my opinion a very convincing relationship in terms of “live-ness”.

In an interview with Cycling '74 (Ostrowski, 2012) Ostrowski explains he has a desire for having a way of controlling his instrument in a way in which he does not directly control parameters of the synthesis process but rather controls subjective aspects of the resulting sound, like intensity, stress and relaxation. He discusses the problematic situation in which a performer of electronic music sits in front of a large panel of knobs, buttons and sliders, giving the performer control over every single aspect of the resulting sound, but losing control over the sound as a total.

In order to be able to create more complex relationships between control data coming from his glove and the synthesis process, Ostrowski is making use of an explicit mapping layer between incoming control data and the synthesis process to be controlled. For this mapping layer he often makes use of physical models. The output of these physical models is then mapped to the different input parameters of the synthesis process. This most probably explains how Ostrowski is able to create strong relationship between physical movement and resulting sound. The use of physical models allows him to interconnect parameters of a synthesis process in a meaningful way and reduce the number of parameters.

Interactive performance

Introduction

Interactive computer music systems are those whose behavior changes in response to musical input. Such responsiveness allows these systems to participate in live performances, of both notated an improvised music.
(Rowe, 1993, p. 1)

It is the task of an interactive system to do interpretation of the musical input which is often a stream of MIDI data or an audio signal. In order for the computer to interpret the input data in a musical meaningful way, it has to do analysis on the data it receives. This analysis will make the machine able to “listen” to a certain extent to the musical input it receives, which often is implemented by emulating human musical understanding. Rowe argues this is a goal with implications for a much broader range of applications. He writes almost any computer music program would benefit from some degree of musical understanding.

At least as important is how the computer reacts to this analysis of input. Rowe calls this the response stage of the interactive music system. In this stage, the system generates musical material as a function of the analyzed and interpreted input. This is where the interactivity takes its form. By assigning different methods of composition to different classifications of analysis of input, the interactive music system is able to generate musical material that is musically related to the musical input coming from the user of the system.

Rowe proposes a rough classification system for interactive music systems. This classification system is built on three dimensions that consist of points describing the possibilities for a dimension. Those points should not be considered to be distinct classes, since any particular interactive music system may show a combination of these attributes.

The first dimensions makes a distinction between *score-driven* and *performance-driven* systems.

The second dimension groups methods of response as being *transformative, generative,* or *sequenced.*

The third dimension distinguishes *instrument* and *player* paradigms. Where systems in the instrument paradigm are concerned with extending musical instruments and systems in the player paradigm with constructing an artificial player.

Robert Rowe's Cypher

Cypher is a practical application of what Rowe describes to be a “real-time interactive music system” (Rowe, 1992, p. 43).

Cypher has two main parts: a listener and a player. Each part has interconnected agents that work on various levels of a hierarchy. The listener is responsible for listening to, and interpreting incoming MIDI data, the player for producing new musical output that can relate to that data.

The listener extracts features like rhythm and harmony, and detects phrases and some of their properties in incoming MIDI data. It consists of two levels. The first level extracts the following features of single events: vertical density, attack speed, loudness, register, duration and harmony. The second level processes the first level data in order to group events into phrases and to decide whether the group behavior is regular or irregular. The results of this analysis are sent to the player which responds to this data.

A group could be seen as a musical phrase. A group is determined by discontinuities in the level one feature agents. Every detected discontinuity in a feature is associated with a weight. All weights of detected discontinuities are summed and the result triggers a signal for a detected group boundary if it exceeds a certain threshold.

It is up to the user of Cypher to decide in which ways the player responds to this data. A graphical interface allows the user to make connections between certain extracted features and methods of responding. Responses used are using precomposed material, generative algorithms, or transforming incoming events are used to respond to the listener.

Two listeners exist in Cypher. One analyzing the incoming MIDI data, and the other analyzing the player's output. This second listener represents a “critic”. It analyzes the control data generated by the player and modifies its material before it is heard.

The player consist of two levels of composition. Level one consists of several compositional methods, level two calls processes based on the listener's analysis.

There are three types of available methods in the first composition level of Cypher's player:

- *Transformation of MIDI input:*
  A chain of many small modules that make simple changes in the material they receive as input.

- *Generative algorithms that create material:*
  Algorithms often making use of random operations using specific input parameters in order to generate distinctive material.

- *Playback of precomposed material:*
  A sequence of MIDI data previously stored on the computer that can range in duration from a few seconds to multiple minutes.

Level two composition processes are influencing the behavior of the composition section over phrase-length durations. They are called by messages coming from the level two listener and influence various kinds of musical flow. Rowe gives the example of a common strategy for the level two composition processes in which connections between extracted features and transformations on level one are broken.

Cypher, the article in which it is discussed and the book that Rowe wrote on the subject of interactive music systems have substantially inspired and influenced me. Namely in my latest work, Tw.2, in which I implemented a few of of his methods. Having two layers of analysis and multiple behaviors for the computer player are concepts I have taken from Rowe's writings about his work. And also my method for detecting phrases is greatly inspired by the way Rowe did it in Cypher.

George Lewis' Voyager

Voyager is another computer program that can be understood to be an interactive music system. It is discussed by George Lewis himself in an article about Voyager (Lewis, 2000).

The first version of Voyager was programmed between 1986 and 1988 at STEIM after which it was refined in the following years. It makes use of Robert Rowe's taxonomy of “player” and “instrument” paradigms. Lewis defines it to be an extreme example of a “player” where the program is not an instrument to be controlled by a performer. Lewis writes he creates a performance of Voyager as multiple parallel streams of music generation, rather than a call and response setup. No hierarchical differences exist between the performer and the “player”. The program can listen to a maximum of two performers that play MIDI-keyboards or acoustic instruments through pitch followers that attempt to convert the performer's sound into MIDI.

Unlike Rowe's Cypher, the performer of Voyager is not able to set up the program, choices about the behavior of the program and its relationship to the analyzed input are generated completely internally.

The player consists of 64 “players” that operate asynchronously and are controlled by MIDI. A function running at intervals between 5 and 7 seconds that recombines the MIDI “players” into smaller ensembles with their own behaviors. This function also chooses and assigns one of 15 melody algorithms, one of the 150 microtonally specified pitch sets, a volume range, microtonal transposition, beat, tempo, probability of playing a note, spacing between notes, interval width range, chorusing, reverb, portamento and how some of these parameters change over time. It also decides to which input it responds, one, both improvisers, or none. In the case of the latter, the “player's” behavior is generated internally without being influenced by any input. Lewis notes that this does not mean there is no interaction, since the “player” makes decisions that have consequences for the music that have to be taken into account by the performer.

> … Voyager asks us where our own creativity and intelligence might lie – not 'How do we create intelligence?' but 'How do we find it?'. Ultimately, the subject of Voyager is not technology or computers at all, but musicality itself.
> (Lewis, 2000, p. 38)

What remains unclear is how the input of Voyager is exactly related to its output, how compositional methods for the generation of output are influenced by Voyager's input.

Karlheinz Essl's m@zeº2

Karlheinz Essl has developed multiple computer programs that are concerned with generative and algorithmic aspects.

m@zeº2 Is an instrument that uses a substantial amount of stochastic operations in order to create musical output.

On his website Essl describes the instrument to be an attempt for an old idea he had for many years (Essl, n.d.):

> the vision of a music:
>
> - which composes itself at the moment of its sounding
> - which is not merely a reproduction of a pre-fabricated notation
> - which is capable of reacting on exterior influences
> - which exhibits infinite temporal expansion
> - which never repeats itself.
>
> (Essl, n.d.)

A system that would not fall into the category of interactive music systems as Robert Rowe describes, since its input is not necessarily musical input that is being analyzed in order for the computer to get some understanding of the musical content of the input, rather it uses MIDI input directly to control parameters of the instrument. Nonetheless I would like to classify this instrument as an interactive music system. This because the computer has a great amount of influence in the musical result that is not directly controlled by its performer but rather is up to the instrument to decide. In this way an interaction between performer and computer emerges in which both the computer and performer have a shared responsibility of the musical result the instrument outputs.

m@zeº2 Is written entirely in Max/MSP and makes use of Essl's Real Time Composition Library (RTC-lib) which is a collection of patches and externals for MaxMSP that offer possibilities for experimentation with several composition techniques such as serial procedures, permutations and controlled randomness. The instrument is based on an ever-growing collection of samples that are processed by “structure generators” which are controlled by the computer keyboard and mouse, several MIDI controllers and a webcam. The user is able to process the resulting sound of these “structure generators” with several DSP processes.

What interests me mostly about Essl's instrument is the balance between chance calculations that are able to surprise its user and the ability for the user to have manual control over the instruments output. A balance which I have attempted to find as well, with a reasonable amount of success, as a result from studying Essl's work, amongst others'.

As an instrument for live electronic music in terms of physicality, transparency and virtuosity Essl's instrument is in my opinion not the most successful one. In fact, it is maybe a good illustration of the more problematic instruments in terms of physicality and transparency.

The controllers used for the instrument do not imply much physicality in themselves. Which does not mean there is no possibility for mapping these controllers in such a way that they require a substantial amount of physicality. But video documentation of performances of Essl with this instrument, do not show much physicality in the way Essl controls his instrument. This, together with the use of chance operations has consequences for the transparency of the performance. Although Essl makes effort to theatrically emphasize the little physical movements made in order to control his instrument, not much insight in the workings of the instrument is gained by looking and listening to the performance.

> The MIDI controllers are utilized to control the various parameters of the compositional algorithms which generates the sound stream in realtime. As this is comprised of several nested and superimposed layers, the control is always "polyphonic". This, however, requires a specific playing technique which enables one to move several sliders and knobs independently at the same time – always controlled by the ear.
> (Essl, 2006)

This seems to be proof of a specific skill one has to obtain to be able to play the instrument which is something that could be developed to a virtuosic extent. But virtuosity is not only to be found in technical skills one develops on instrument, other ways of developing virtuosity are automatically present in most kinds of digital music instruments. Timing for instance allows a great amount of virtuosity, just being able to put a sound on and off at will, allows one to practice this possibly virtuosic skill.

Cort Lippe

Cort Lippe is a composer of electronic music that has composed numerous pieces dealing with the concept of interactivity between a performer – mostly an acoustic instrumentalist – and a computer program (Lippe, 2002). Often the computer program Lippe designed for his compositions transforms the sound of the performer with several DSP processes. The parameters of these transformations are influenced by analysis of the same sound that is used as input for the program.

> My compositions for instruments and computer have an underlying structure that cannot be altered. Most pieces do not include improvisation. Electronic and instrumental events are ordered and their unfolding is teleological. The computer "score" has events that are analogous to the pitch/rhythm parameters of traditional musical notation, in that these events are essentially the same for each performance. But, overlaid on this skeletal computer score, a great diversity of potential for variety exists, so that while the computer part is pre-composed conceptually, the performer shapes the final version during a concert. The computer score functions in much the same way as the musician's score.
> (Lippe, 2002, p. 4)

Lippe's way of creating interaction between performer and computer differs much from both Rowe's as well as Lewis' work in the sense that it does not allow free improvisation, which in my personal opinion is the most interesting area for performer-computer interaction. A system that is actually able to interactively improvise with a certain amount of freedom is in a sense more interactive than one in which interactivity is mostly deterministic. This because the relationship between performer and computer is less hierarchical, in contrast to what is mostly the case in Lippe's work, where the performer is only responsible for the shape of the final composition.

Lippe argues he is not much interested in the quantity or quality of performer-computer interactivity, rather, his interest lays in the quality of musical interactivity. He is interested in modeling the relationship between different musicians performing a piece of music together. I would like to argue that quantity and quality of performer-computer interactivity can not be disconnected from the quality of musical interaction the performer has with the computer. Not to say that a big amount of quantity and quality in performer-computer interaction will automatically result in a high degree of quality in musical interaction. But, if the goal is to have a system that is capable of interacting with the performer in interesting ways, there is a definite dependance on the quantity and quality of the system's listening and responding algorithms.

Using the input sound as source material for transformations for the computers' response allows an interesting type of transparency that can show the technical relationship between performer a computer. I like to hear the way improvising musicians are interacting with each other, and the same holds for interactive music systems. But there is a risk that a performance becomes too transparent, it can cause a performance to become predictable which in most cases is not very desirable.

Personal solutions for live electronic music

Introduction

In the past and up until today. I have been working on several instruments to realize my ideas about live electronic music. Those instruments were always a combination of a particular piece of software and particular controllers. The new instrument was always a follow up of the previous. Although sometimes quite radical changes were made, they were all based on the same idea and good features of the old instrument were taken as a starting point for the new.

There were two strong ideas that all instruments had in common, namely:

- The instrument should allow the performer to improvise freely, being able to play a completely new piece without having to change anything to the instrument.
- The instrument should allow the performer to play multilayered music solo.

Besides those conceptual ideas there were some criteria for the instruments that they had all in common, namely:

- The instrument should be stable.
- The instrument should be controlled in a physical way.
- The instrument should allow the performer to develop a certain amount of virtuosity.
- The instrument should create a certain amount of transparency between the performers actions and the resulting audio.

Up until now there have been four notable instruments of which I will discuss three shortly and the fourth, which is the last instrument to date, extensively.

JS

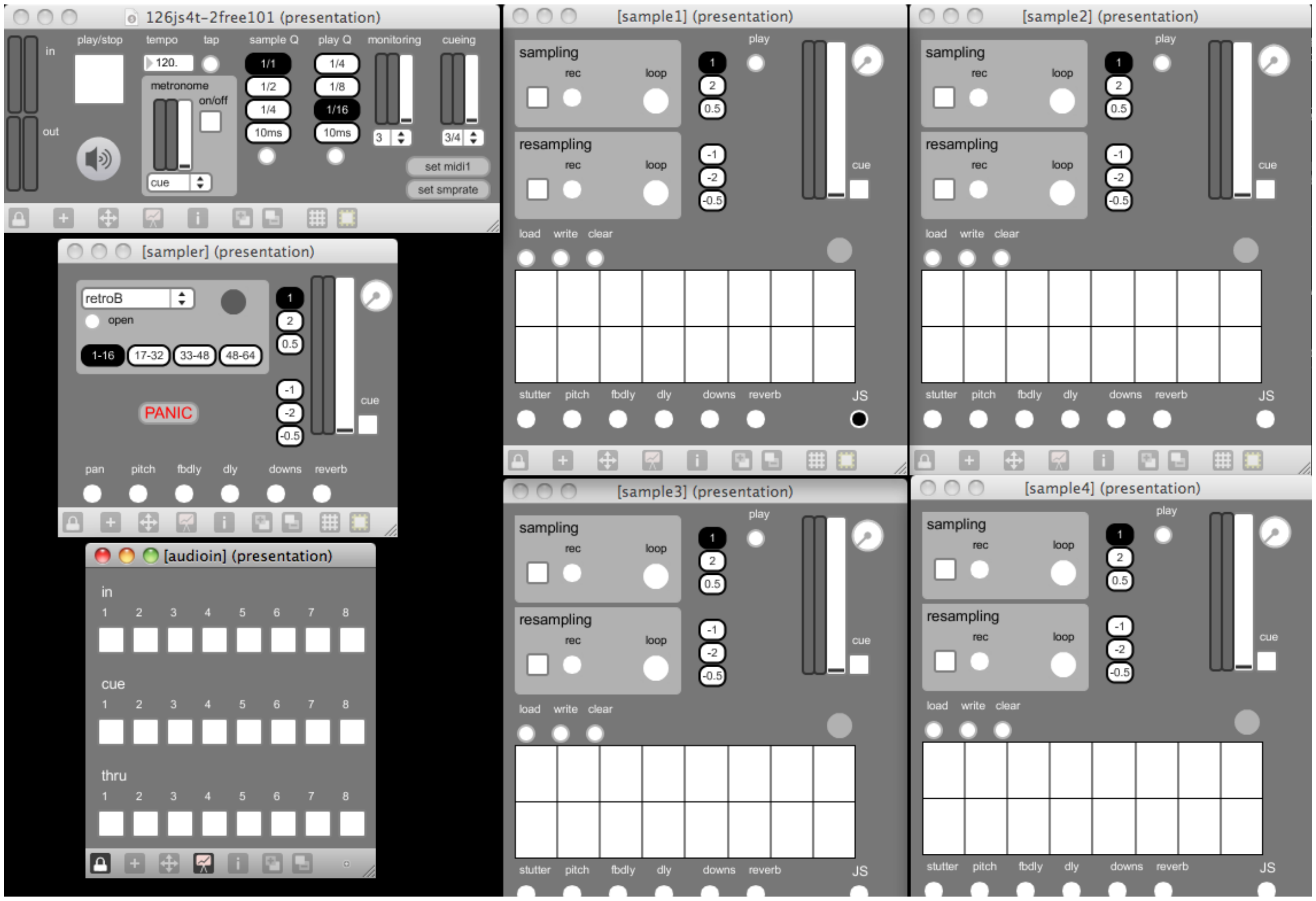

Figure 1, The GUI of JS

The first instrument that addressed both main aims of my instrument design was called JS. The software used for this instrument was a Max patch focussed around the idea of live sampling. The patch consisted of four “buffer players”, sub-patches that where able to play back a sampled buffer, in various ways. The buffer players were controlled by an Akai MPD24 – a MIDI controller mainly consisting of pads that can be used in a percussive manner - and a set of Saitek X52 joysticks – the kind normally used to play flight simulation games. The MPD was used to trigger the playback of the buffer at a specific starting point in that buffer. The joysticks were used to trigger and interact with parameters of different kinds of transformations. Transformations used were a stutter effect, a feedback delay, a downsampling transformation and reverb.

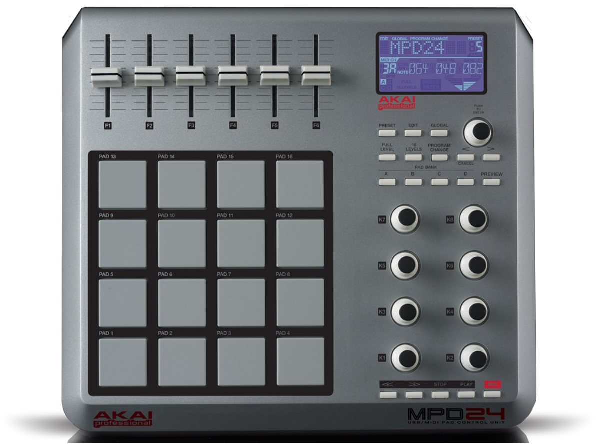

Figure 2, Akai MPD24

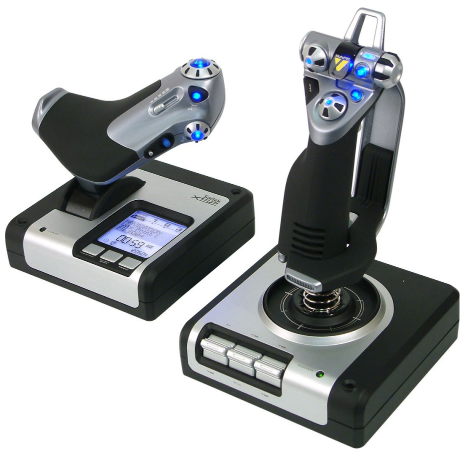

Figure 3, Saitek X52

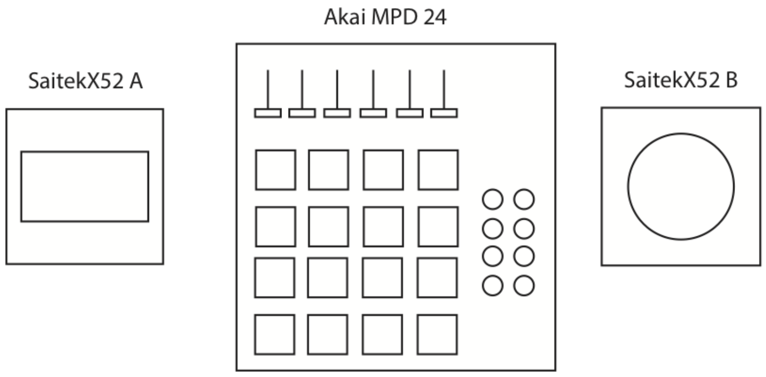

Figure 4, Schematic representation of JS's hardware set-up

The buffers loaded into the buffer players could be buffers which were stored on the computer, or live recorded buffers coming from either connections on the sound card or other parts of the patch, for instance another sampler. The fact that the buffer could be recorded live was a way to be able to improvise and be able to play a new piece without having to change the inner workings of the instrument. The ability to record arbitrary sounds as source material for further transformation gave me new input and thus new output every performance, within the contains of my own esthetics of course, defined by the character of the transformations, and the way I could interact with them.

Often I would take my source material from other musicians with whom I was playing. But to not be completely relying on the output of other musicians I made use of a more classical sampler. Effectively, an instrument in itself. This sampler was another part of the Max patch, also controlled by the same MPD24 and set of joysticks. This sampler was more classic in the sense that it worked in a pretty similar way to the well known Akai MPC series of samplers. It held different sets of sounds (collections of buffers) which could be chosen through the GUI of the Max patch. Whenever one of those sets was chosen I would be able to trigger the sounds by hitting the buttons on the MPD. Just like the MPC, every button was mapped to a specific sound. Unlike the MPC some parameters of the playback algorithm were controlled by the set of joysticks giving me the possibility to shape the sounds while triggering them. Modulating the playback speed of the sound, determining its playback direction and transforming it by means of bit reduction, distortion, and feedback delay were methods used to shape the triggered sounds.

Figure 5, Akai MPC2000XL

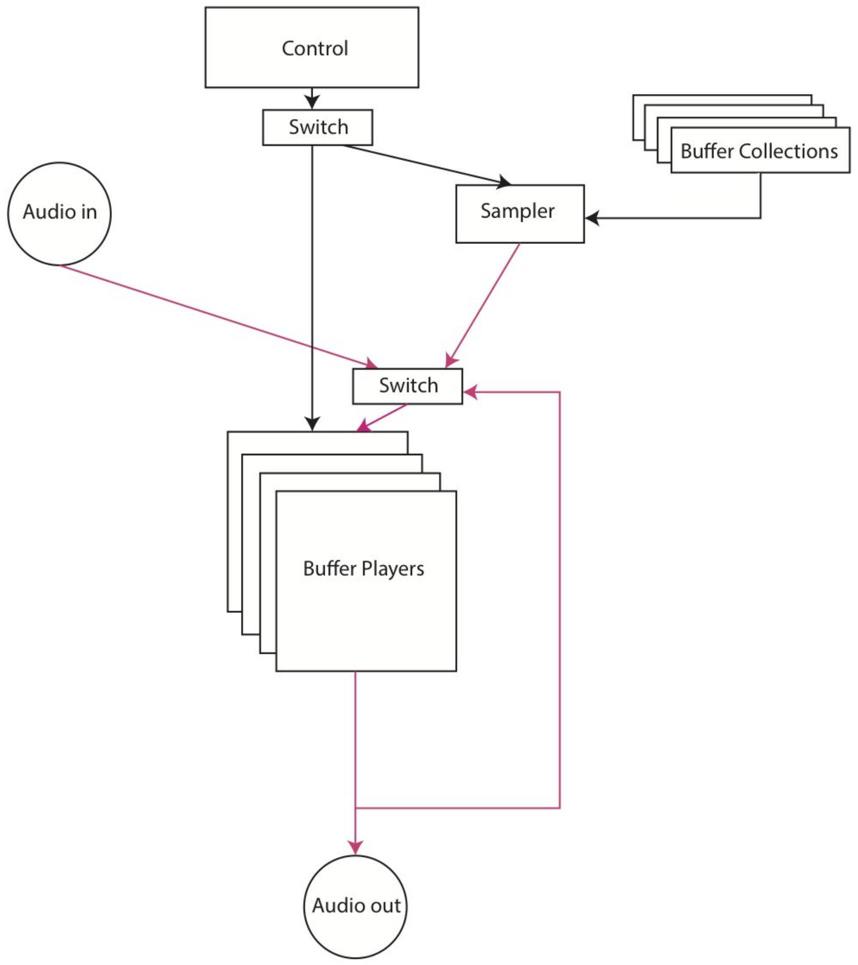

Figure 6, Schematic representation of JS's software

Evaluation

One way for me as the performer to play multilayered music with the instrument was to record sequences played with the sampler which would be looped. Being able to record four loops in total. Then I would go back and forth between transforming the recorded loops, playing sequences and recording new sequences.

A big downside of this way of working was that there was always a certain part of the music left simply looping audio. In terms of transparency I think this way of playing multilayered music was rather successful because it was clearly audible a just played sequence of material was being looped. On the other hand the parts of the patch that where to be played directly, allowed output without the performer touching the instrument, which caused possible confusion.

The joysticks and the pad controller both required a substantial amount of physicality to control the instrument, although there was no effort done on mapping these controllers in the most physical way and therefore there was still a lot to gain in terms of physicality.

In terms of allowance for virtuosity I think it was rather successful. Since the instrument was a rather simple one – a sampler with extended functionality – the musical quality of the instrument depended mostly on the quality of interaction with the instrument rather then the transformations or techniques used programming it.

PD

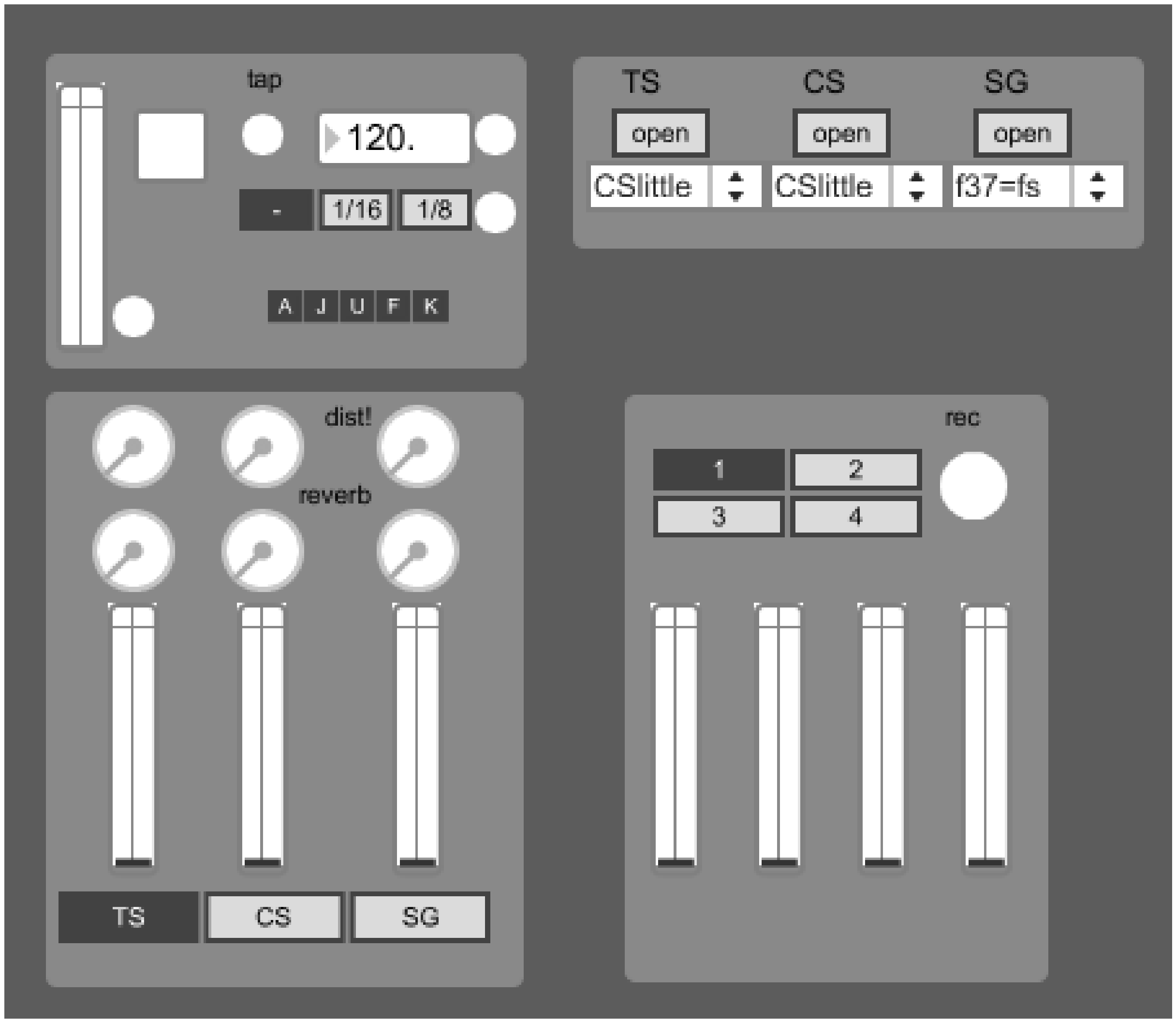

Figure 7, The GUI of PD

PD was the name of the first instrument I started working on when starting to study Sonology. It was based on the same ideas as my first instrument with an emphasis on the problem of multilayered solo performance.

I found it to be problematic that a part of the music was always looping in *JS.* I came up with the idea to record control signals instead of audio. This control data could control for instance a synthesis process based on stochastic algorithms that would, when looped, still be a stochastic process recalculated every loop. In this way I could avoid loops being static.

I made use of the possibility to create external objects in Max with Java in order to create objects that where able to efficiently record and store large streams of numbers and play them back. I would record phrases played with the sub-instruments of the instrument. These phrases would then be stored in one of the four control-data recorders. The playback of these stored phrases of recorded control signals could then be started and stopped by myself and some parameters of the playback itself could be controlled.

PD consisted of 3 sub-instruments. A “classic sampler”, a granular buffer player and an algorithmic buffer player. And also in this instrument, as well as in *JS,* the two main controllers where an Akai MPD24 and a set of joysticks. An expression pedal was added for level control and an UC-33 – a controller with faders and rotaries – to control the control signal players.

Figure 8, Roland EV-5 Expression Pedal

Figure 9, UC-33

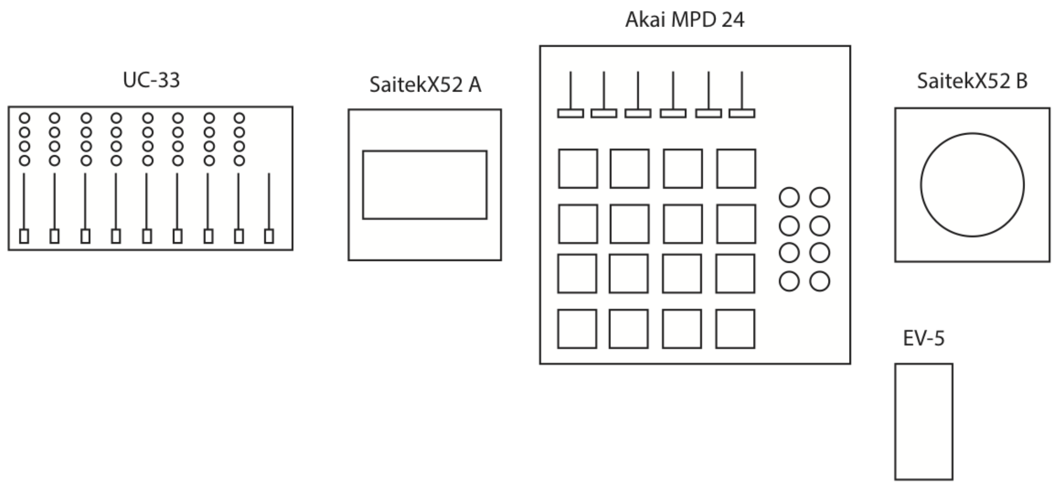

Figure 10, Schematic representation of PD's hardware set-up

The “classic sampler” was a classic sampler in the sense that it is very close in functionality to commercially available hardware samplers like the AKAI MPC: samples were mapped to pads on the MPD 24, every pad a different sample. I extended this functionality with the ability to modify the playback speed and direction.

The granular buffer player had one sample mapped to every pad on the MPD24. A granular process would be triggered whenever one of these pads was hit, using the buffer mapped to that particular pad as source material. The joystick controlled parameters of the granular synthesis process like the rate at which the grains where triggered, the size of the grains, the playback speed of the grains, and several “jitter” parameters to add randomness to the process. This randomness would be maintained during playback of the recorded streams of control signals.

The algorithmic buffer player played buffers in a rhythmic manner. Several stochastic calculations were used to calculate different parameters of the algorithmic process. Examples of these parameters are for instance the next buffer to be played and the playback speed at which a buffer was played. Most of these stochastic calculations were in fact tendency-mask-like generators of which the boundaries were controlled with a “center” and “span” parameter. With a center of 5 and a span of 2 the lower boundary of the tendency-mask-like generator would be at 5-2 = 3, and the upper boundary at 5+2 = 7 and thus, values would be chosen randomly within the range of 3 and 7. The center and span parameters would be controlled by the joystick.

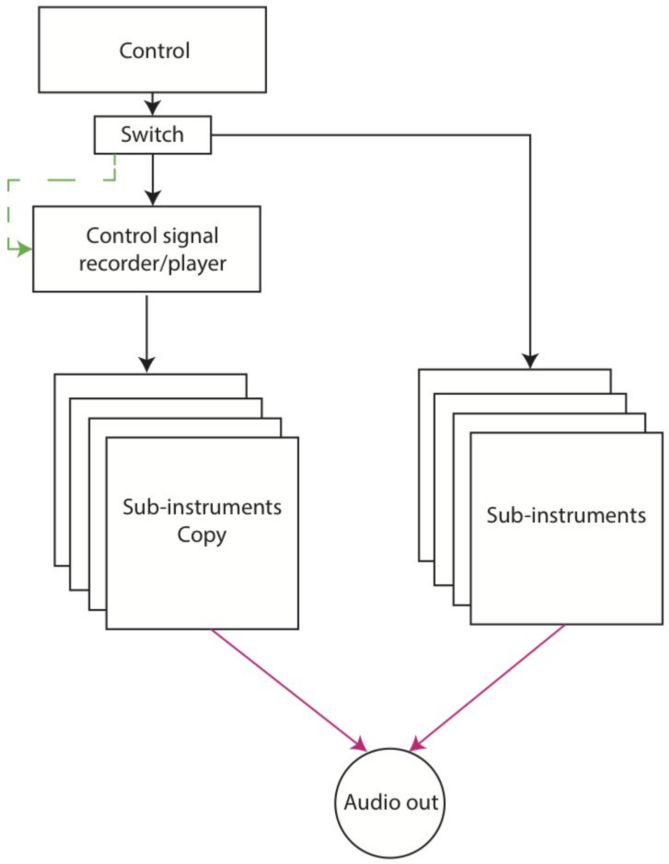

Figure 11, Schematic representation of PD's software

The rhythmical structure of the algorithmic process would be controlled by the MPD24. Every pad had a different kind of stochastic distribution mapped to it. These stochastic distributions would then calculate the durations between playback of the buffers, creating a particular rhythmical structure. After being calculated the durations were multiplied with a value controlled by one of the joysticks to give me, the performer, control over the speed at which these rhythmic structures would be playing.

A short list of commonly used distributions and there characteristics follows:

Beta	- A distribution with a higher chance for values on the borders of the distribution.
Gaussian	- A distribution with a high chance for values in the middle of the distribution.
Uniform	- A distribution with equal chance for all values within the range of the distribution.
Exponential	- A distribution with an exponential decay in chance for occurrence of the higher values in the range of possible values.
Manually curved	- A distribution that has a higher chance for lower or higher values, depending on the curvature. Often done by choosing a uniform distributed value between 0 and 1, and raising it to a power. Raising it to the power of 2 would give a higher chance for lower values in the range, raising it to 0.5 would give a higher chance for high values in the range.

Evaluation

After playing for a while with the instrument I came to the conclusion that having to record control signals in order for the computer to make sound on itself was not the most convenient solution to create multilayered music. It caused the necessity for the performer to play something first. The layers created by the computer were always following the performer in the sense that a kind of material was first played by the performer after which variations on that material where played by the computer. It caused every performance to have the same structural tendencies.

This way of playing multilayered music was also not quite a step forward in transparency. Since the loops often contained material that was different from their recorded sources due to stochastic distributions, it was less clear that a just recorded phrase or sequence was being looped.

The amount of physicality and transparency of the directly controlled sub-instruments in PD has grown in comparison to JS. Mainly because the sub-instruments could not output anything without the performer actually touching and interacting with them, which created a more clear visual relationship between the musical output of the instrument and the movements made to control it.

The amount of virtuosity the instrument allowed, was not necessarily more than in JS. The instruments got a little more complex, but still the quality of musical output was determined by the quality of interaction, not by the instruments themselves.

Tw.0

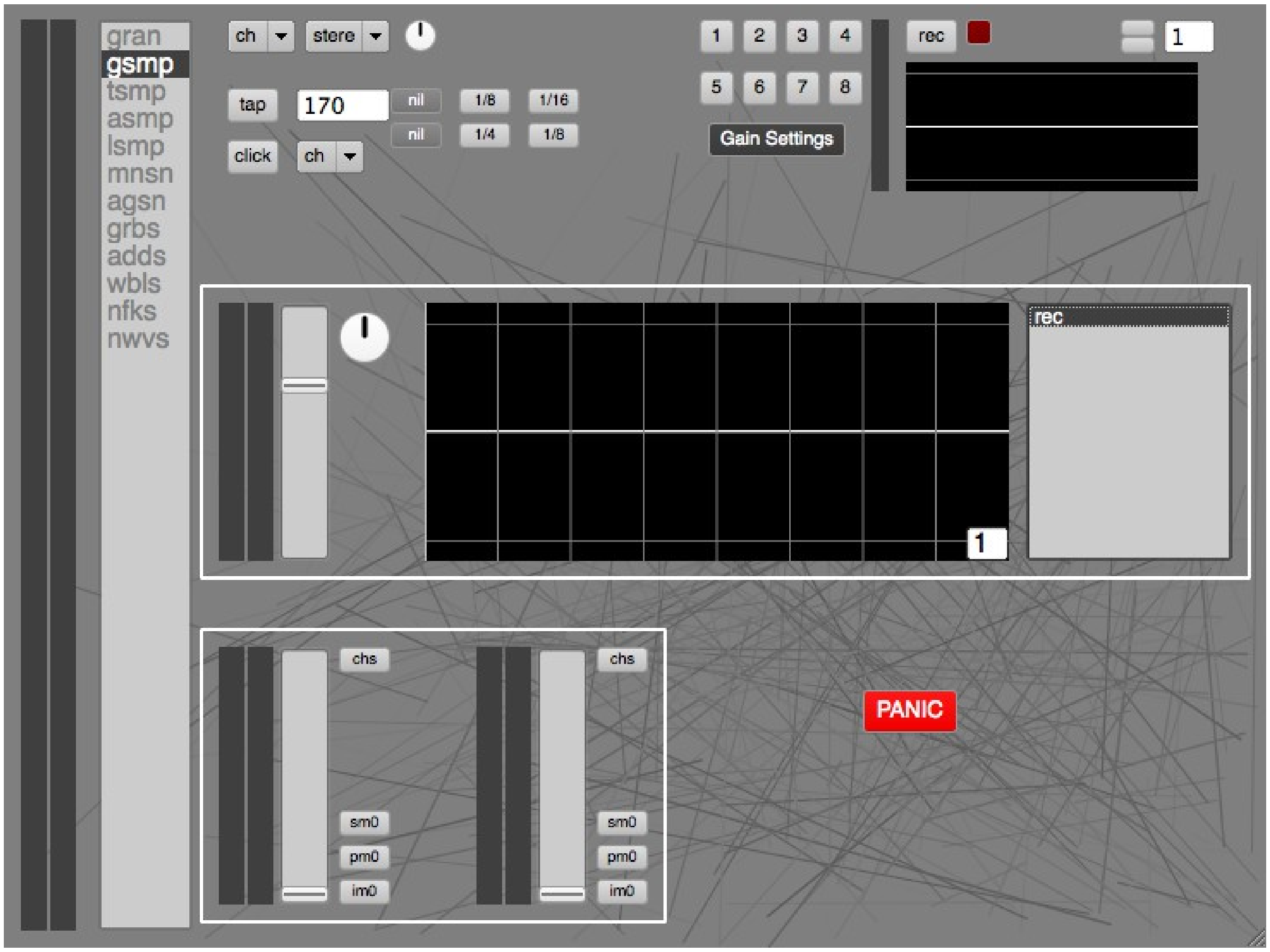

Figure 12, GUI of Tw.0

Tw.0 was the second instrument I have created while studying Sonology. Still my main aims for the instrument had not changed very much. Although having reached the point at which the instrument made improvisation possible in the previous two instruments already, the way the instrument allowed me to play multilayered music solo was still not optimal.

Tw.0 was my first instrument of which the software was programmed in SuperCollider. This, for me, made programming easier. Especially when there is a desire for an instrument which is very versatile and has different ways of playing it, by making use of sub-instruments for instance, keeping code organized can become a very uncomfortable pain in the neck in MaxMSP 5. SuperCollider allows one to easily create a file structure for a patch or program. Allowing one to have a folder with subfolders in which all code is organized according to classifications. In MaxMSP 5, there were possibilities for creating a patch that would allow a similar file structure though, but spending time programming such non-musical code is not quite a hobby for me.

Two.0 used the same hardware set-up except for the UC-33, that was replaced with a similar controller with less rotary knobs but more convenient dimensions: the Korg nanKONTROL.

Figure 13, Korg nanoKONTROL

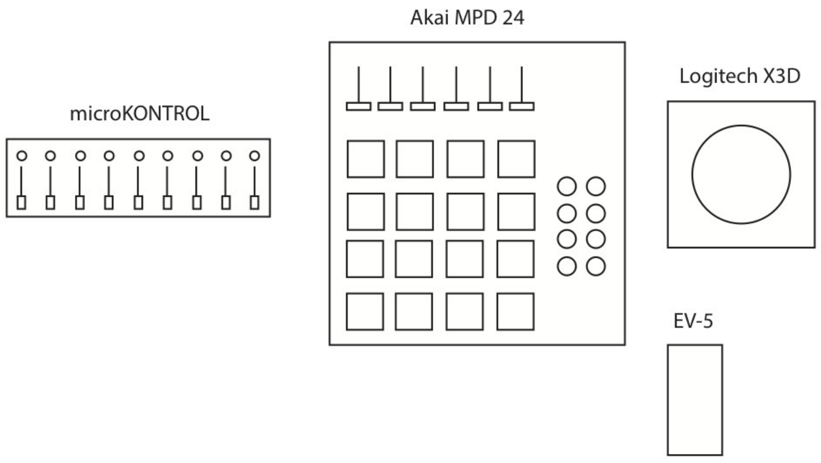

Figure 14, Schematic representation of Tw.0's hardware set-up

The biggest difference, and improvement, in *Tw.0* compared to *PD* was the idea of performer and computer instruments. I came up with the idea to create an instrument which consisted of two kinds of sub-instruments: performer instruments and computer instruments. The performer instruments would be the instruments played live by the performer, and the computer instrument would be an algorithmic process making use of stochastic calculations in order to be able to “improvise”.

In order to derive two parameters from the music played with the performer instruments, pitch and speed, musical control signals were read from the controllers used to control the performer instruments. Speed in this case was the reciprocal of the duration between onsets of buttons that triggered events. Those parameters were read by the algorithms that represented the computer instruments who used those analysis parameters as input parameters and thus “reacting” upon them. This, together with the use of stochastic calculations, made the computer instruments into something which could be seen as an artificial co-performer. Looking at it from this point of view the result of the combination of those two kinds of instruments could be seen as a very rudimentary duet between performer and computer.

Instead of having only two sub-instruments *Tw.0* had a multitude of performer instruments: a granular buffer player with manual control of the reading point, a granular buffer player with automatic control of the reading point of which speed was variable, a simple MPC-like sampler, an algorithmic buffer player similar to the one in *PD*, a looping buffer player similar to the ones in *JS*, an additive synthesizer, a feedback based synthesizer, and an algorithmic, digital-sounding rhythmic instrument. As in my previous instruments, changing between performer instruments caused the mapping of almost all controllers to completely change but, keeping basic ideas about functionality for some of the controllers to be consistent throughout all instruments. One of these ideas was for example that the MPD24 pad controller always was used to triggered sounds, and the joystick to control continuous parameters of the instrument.

The computer instruments were quite simple in terms of their ability to analyze and react upon the performer. Another way of looking at the computer instrument could be to describe as what Richard Barrett calls a reactive instrument. The performer had control over certain aspects of the computer instruments. The performer had the ability to start and stop the computer instruments, control the volume at which the instruments where playing and change the material an instrument was playing to other randomly chosen material. And quite importantly, the performer had the ability to change the way the computer instrument reacts upon the performer's control signals. For both the pitch and the speed analysis the performer could choose if the computer would oppose or follow.

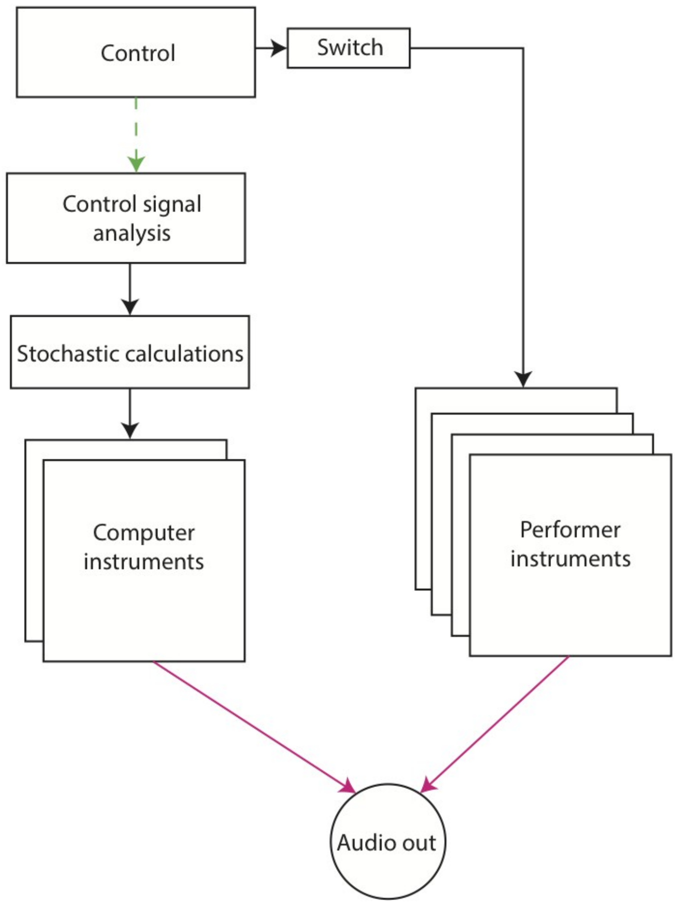

Figure 15, Schematic representation of Tw.0's software

The concept of having a human and a virtual musician playing a duet together was merely a result of my search for a convenient way to perform multilayered music solo. My work was therefore, not about embodying this concept, like in the case of Rowe's Cypher for instance. Although it might have been interesting to exploit this concept, I had no intension in doing so because I did not feel there was a musical necessity for it.

Not as a reason but more as a cause of this relationship between the instrument and this concept, the computer instruments were kept quite simple. There was simply no need for the computer instruments to be complex since there was the possibility for the performer to influence their behavior.

Evaluation

An obvious downside of the way I choose to obtain the analysis parameters from the performer's controllers, was that they did not always represent the actual resulting sound well. Because the same controllers were used for different instruments with different synthesis processes and algorithms – although making sure the right part of the right controller was tracked according to the parameter to be read – the sounding result of a movement from a tracked controller could be very different from one performer instrument to another.

This way of playing multilayered music was possibly quite transparent as well as extremely untransparent, depending on the way the audience was informed. If the audience had no idea about the distinction between performer and computer instruments, there would be a substantial amount of musical output without a clear cause. On the other hand, if the audience was to be informed about this distinction, they would be able to understand where the musical output that had no direct relationship to my movements would originate from. That, would make the relationship between the directly controlled sub-instruments and their output more clear as well.

Tw.2

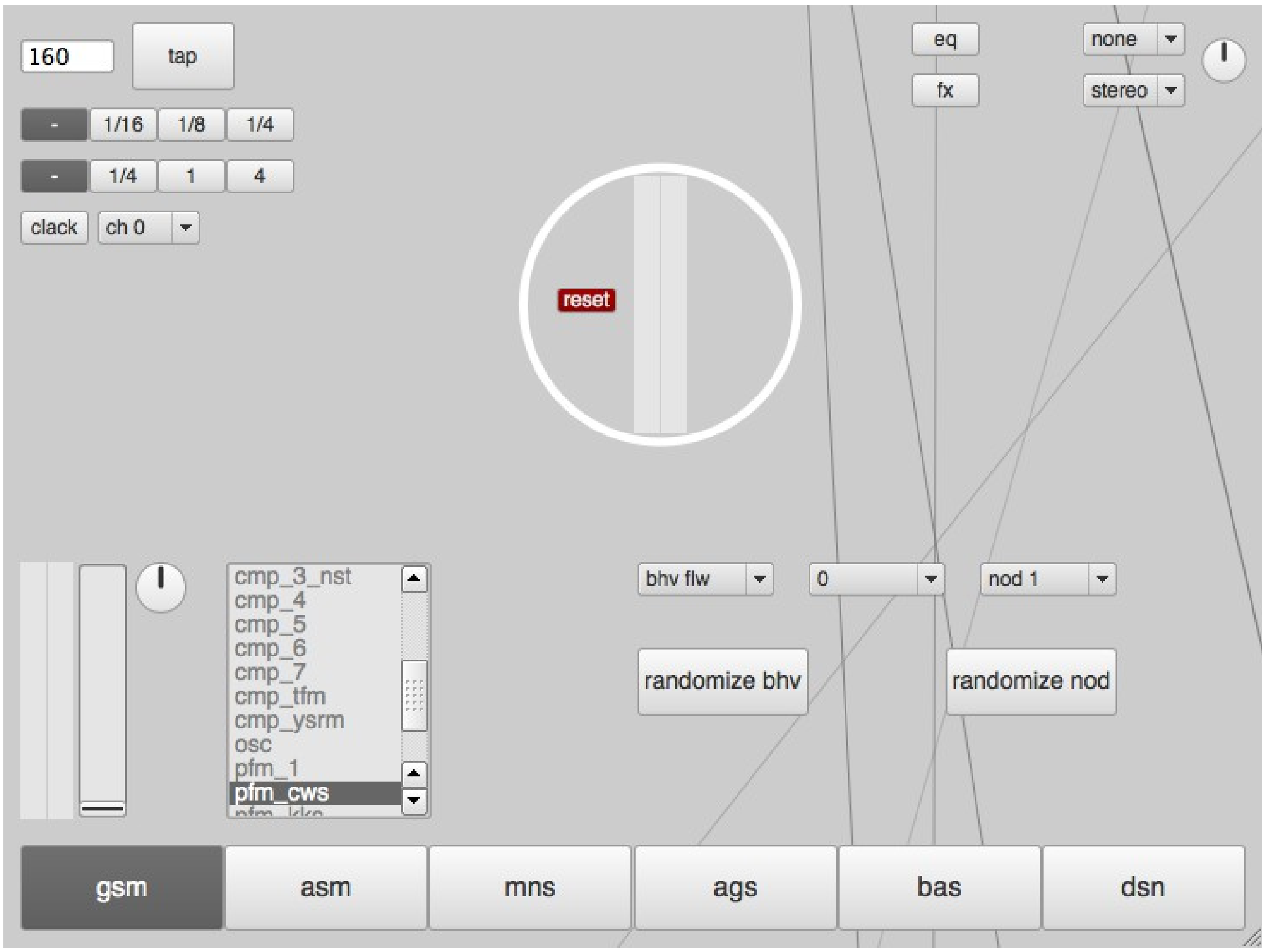

Figure 16, The GUI of Tw.2

Introduction

Tw.2 is a complete rewritten version of *Tw.0* and could be seen as the second version. It is my most recent instrument to date. It was written using a Model-View-Controller[1] inspired software architecture pattern allowing easy extension of the application. Already while playing with *Tw.0* all controllers started to get replaced one by one with new, better controllers were used to control the software. In particular the use of the iPad app Lemur was important since it allowed me to close my computer screen while performing and take away the mystery that it causes.

After having played solo with *Tw.0* for a while, the idea of the actual realization of the concept of a more intelligent virtual player, with which a performer would form a duet, became more appealing. Therefore I have chosen to completely rethink the computer instruments, making them a lot more capable of reacting and interacting with the performer and vice versa.

On the contrary, almost not much has been changed to the performer instruments besides the controllers used to control them and some details in how they are mapped.

1 Model-View-Controller is a software architecture pattern which separates code defining the functionality of a program from code that defines the programs' current state and code used to for interaction with the program (a GUI for instance).

Hardware

The change of hardware was not at all a drastic change. Very similar new controllers were used to replace the old ones.

The two joysticks were replaced with one joystick and an iPad. Since the left joystick of the set was never really used intensively in previous instruments, and when it was used, it was used more as a set of faders than as a physical controller. The left joystick was mainly used to control the dry/wet parameters of global transformations for the performer instruments. These transformations were changed to be controlled with faders on the MPD24 pad controller. The left joystick and the Korg nanoKONTROL made place for an iPad with the Lemur App. Not only the functionality of the joystick and the nanoKONTROL was replaced, but the iPad also enabled me to replace my onscreen GUI allowing me to close my laptop.

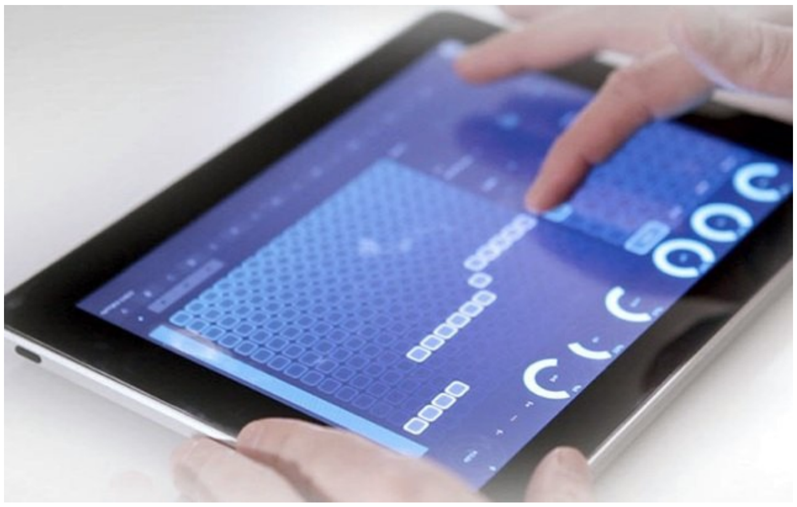
Figure 17, iPad with the Lemur App

I replaced the right joystick with a new joystick, the Logitech Extreme 3D Pro, which had very similar features, with the most different feature being its price, which was a factor four less than the set of joysticks. A big pre, because these “toys” are not made for very intensive use, and do not last very long. Furthermore the new joystick used is a pretty common one, being easily found in computer shops, in contrary to the set of joysticks, which had to be ordered online. As with all previous instruments, the right joystick was used to control continuous parameters of the instrument selected to play by the performer.

Figure 18, Logitech Extreme 3D Pro

The Akai MPD24 was replaced by the fairly new Keith McMillen QuNeo. As well as the Akai, it has sixteen velocity and pressure sensitive pads, and although different from the Akai's, it has faders as well. The big difference is that everything on the QueNeo is pressure sensitive which makes the controller a very expressive one. Furthermore it has the exact size of an iPad, a lot smaller than the MPD24, resulting in a big improvement of the mobility of my instrument. Like the MPD24, the QuNeo was mainly used to trigger events and articulate them with the pads' pressure sensitivity.

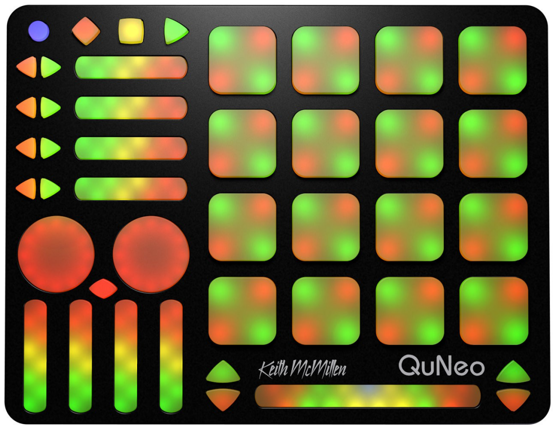

Figure 19, Keith McMillen QuNeo

The expression pedal and it's use did not change since I first incorporated it into my instrument. I have found the use of the foot to control the overall volume of the performer instruments to be very effective. Only the device used to convert the control voltage coming from the expression pedal into MIDI was not very satisfying. Three of the four inputs of the device died in the first year, which in terms of stability of the instrument was worrisome. Opening up the device made clear it was not to be fixed and with a price just above 100 euro, it was not possible to accept I would have to replace the device regularly and therefore another solution should be found. Commercial alternatives to the device were rare so I choose to build a similar device with an Arduino interface. This made the total cost of the device about 30 euro and gave me the possibility to easily fix the device whenever needed, since the necessary circuitry used was fairly simple.

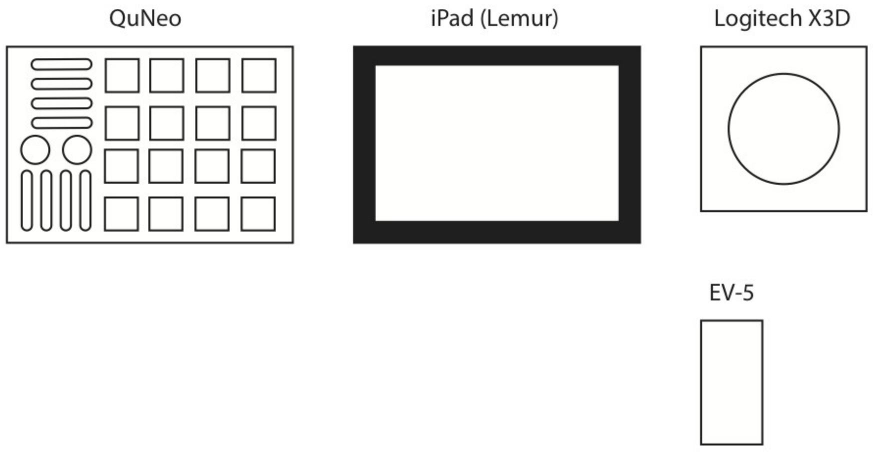

Figure 20, Schematic representation of Tw.2's hardware set-up

I would say the hardware of the instrument and the mapping to the software is what defines the amount of possible physicality, transparency and allowance for virtuosity. The joystick is clearly a very physical controller, which holds for the pad controller and expression pedal as well. They all allow the instrument to extract several features from the performer's body movements. Using these controllers, allowing physicality, virtuosity and transparency to take place is now mostly a matter of mapping.

Software

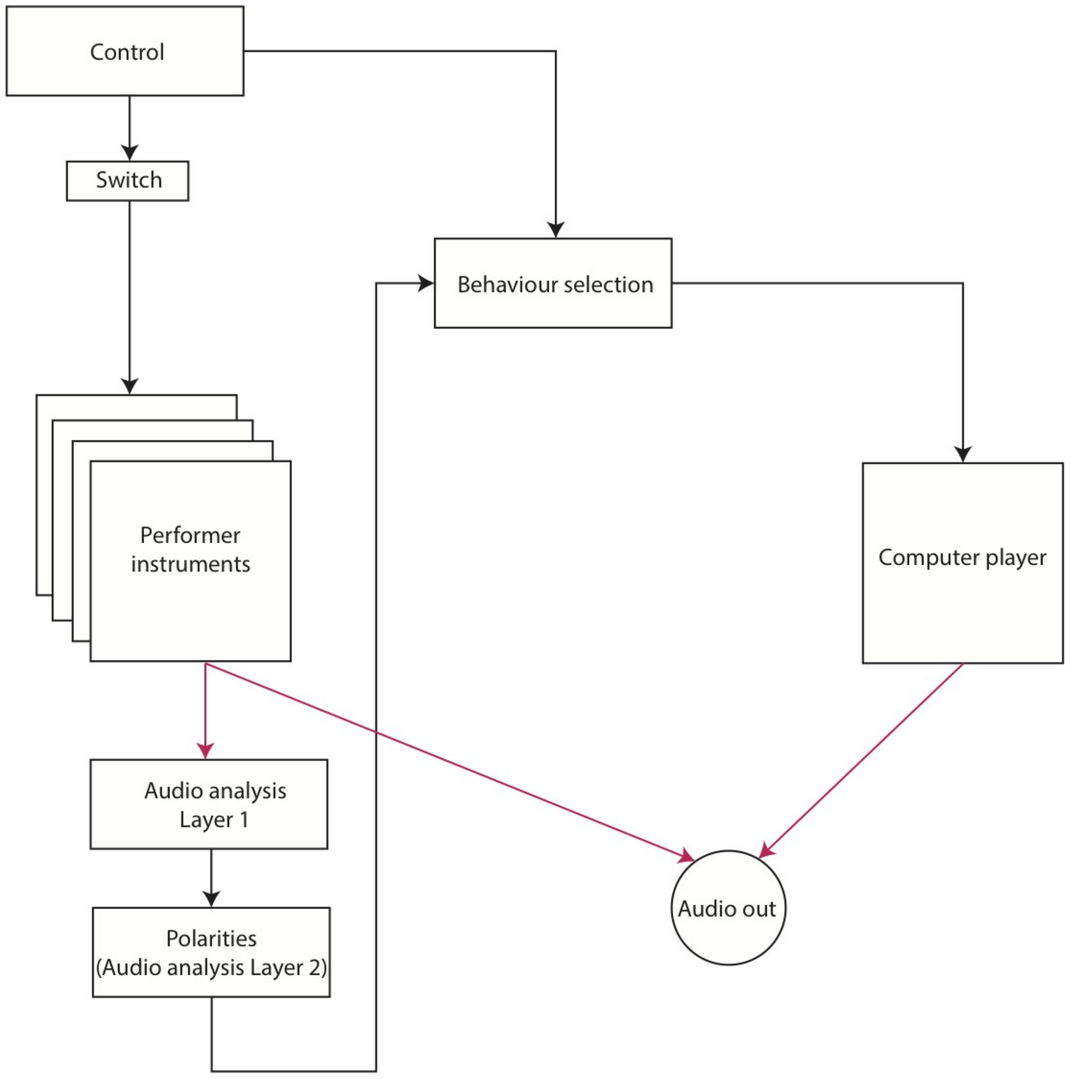

Figure 21, Schematic representation of Tw.2's software

Performer instruments

The conceptual software design of the performer instruments have not changed much since Tw.0, although code was completely rewritten and the structure of it changed quite radically. The instrument currently contains four performer instruments:

- GSM, a granular based buffer player.
- ASM, an algorithmic buffer player.
- MNS, a combination of synthesizers based on AC multiplication and/or feedback.
- AGS, an algorithmic process producing real time synthesized percussive material.

GSM

GSM is an instrument that allows the performer to trigger and hold one, or multiple, granular synthesis processes, which use a buffer as source material for its grains. The buffer used is selected with a pad on the QuNeo, which also turns the process on and off. The instrument is polyphonic; multiple synthesis processes can be played simultaneously all using different buffers as source material. The amount of pressure applied to the pad is used for articulation, in this case controlling the rate at which grains at triggered and the size of the grains.

The joystick is used to control parameters of the granular process, the playback speed of the grains, the position in the buffer at which the grains are reading or the speed at which the process reads through the buffer depending on the current mode of the instrument. For all of the mentioned parameters a jitter parameter defining the amount of jitter (randomness) is used to determine the value for the next grain.

Whenever polyphony is created by holding more than one pad at the same time, all voices will listen and receive the same values for their parameters, the buffer used as source material is what differentiates one voice from another.

GSM has two modes in which the way the position in the buffer for the grains is read differs from mode to the other. In the first mode, the position changes continuously in a linear manner. It reads from the beginning to end of the buffer at a variable speed determined by the Y-axis of the joystick. In the second mode, the position at which the granular process reads is not automatically changing – unless jitter is applied. Instead, the position is manually determined by the position of the Y-axis of the joystick.

ASM

ASM is an algorithmic buffer player. The instrument allows the performer to trigger and hold algorithmic processes that play back buffers in a rhythmic manner. Just as in GSM, it is a polyphonic instrument, it allows the performer to trigger and hold multiple processes at once. With the pads the performer triggers and holds a process with a specific algorithm mapped to the pad which is responsible for calculating the durations in between playback of buffers, resulting in a specific rhythmical structure. Several stochastic as well as deterministic algorithms are used. The durations are then multiplied with a multiplication factor controlled by the Y-axis of the joystick. Furthermore the pressure of the pad is used to determine the duration of the playback of the buffers, applying a lot of pressure results in long sounds, applying only a little results in very short sounds with the shortest possible sound being not much more than a click.

The right joystick is used to control parameters of the algorithmic process, the buffer to be read, the playback speed at which its read and the multiplication factor for the durations between sounds. As well as jitter parameters for the buffer to be played back, the duration of the sounds, their playback speed and their position in the space.

As in GSM, whenever multiple processes are held at once, all processes read the same values coming form the joystick for their parameters. What differentiates them is the algorithm used for their rhythmical structures, as well as the independently controlled duration of the sounds.

And also as GSM, ASM has two different modes. In the first mode, a stochastic algorithm is used to determine the next buffer to be played back. In the second mode, the buffer is not algorithmically changing but instead the algorithm is used to determine the position in one buffer that is read. This results in a process which is referred to as brassage.

MNS

In terms of control, MNS is quite similar to GSM. It consists of four sub-instruments of which one is chosen with a hat-switch on the joystick. Those sub-instruments are defined in SuperCollider as SynthDefs[2]. The four sub-instruments are:

- *fbm*: An instrument that feeds back its output to control the frequency of another oscillator which is used to control the phase of the first oscillator.
- *cfb*: An instrument that feeds back the output of one oscillator into the frequency of another oscillator which on its turn feeds back its output into the frequency of the first oscillator.
- *ofb*: An instrument that reads a buffer at a frequency which is modulated by a sine wave.
- *ffm*: An instrument with two sine wave oscillators of which the outputs are fed back into the frequency control of two other oscillators which are used to control the phase of the first sine wave oscillators.

All of these sub-instruments' SynthDefs have the same amount of arguments with the same names, all scaled inside the SynthDef to the right range for the specific SynhtDefs. This makes them interchangeable without changing any code but the Synths name.

The instrument is triggered by the pads on the QuNeo. Each pad is mapped to a specific frequency, when the pad is pushed, a Synth will be triggered according to the selected the sub-instrument, with for it's pitch argument the pitch mapped to that specific pad. The pressure applied to the pad is measured by the QuNeo and used to control the amount of modulation, which can be either the amount of feedback or the range of the frequency modulation depending on the sub-instrument chosen.

2 A SynthDef is a class in SuperCollider that defines a synthesis process. Multiple independent instances of such a SynthDef can be created with another SuperCollider class called Synth.

The X-axis of the joystick is used to control another the frequency oscillator, since every sub-instrument consists of an oscillator which is controlled by another, every SynthDef has a second frequency parameter, which generally can be seen, as a modulation frequency.

Furthermore, the joystick is used to control a looping envelope that controls the amplitude of the instrument. A button on the joystick turns the looping envelope on and off, and the Y-axis controls the frequency at which it is looping.

AGS

AGS is quite similar to ASM in terms of control, and also in terms of it's programming. The main difference between the two instruments are the Synths the instrument uses. Instead of playing back buffers, it plays back digital sounding Synths.

As ASM, AGS is an instrument that allows its performer to trigger and hold rhythmical algorithmic processes. These processes are triggered with the pads on the QuNeo. Each pad has a specific deterministic or stochastic rhythmical distribution mapped to it.

AGS is also polyphonic. Multiple rhythmical processes can be triggered and held at once. All parameters controlled with the joystick will be common amongst those processes.

As well as MNS, AGS has four sub-instruments. Which are:

- *asd:* An instrument outputting a saw wave oscillator which is down sampled at a modulated rate.
- *acm:* An instrument outputting a square wave which multiplied by noise with a variable frequency.
- *esn:* An instrument outputting an impulse oscillator with a percussive envelope on the impulse's frequency parameter.
- *sin:* An instrument outputting a simple sine wave at variable frequency.

And also like MNS, AGS' sub-instruments are selected with the hat-switch on the joystick. And in order to make the Synths interchangeable in terms of code, all SynthDefs representing the sub-instruments have the same amount of parameters with the same name.

The joystick is used for controlling tendency-mask-like generators for frequency and modulation parameters of the sub-instruments. The X and Y axis are responsible for controlling the center of the tendency-mask-like generators for pitch and modulation respectively. The Z-axis of the joystick controls the spans of the tendency-mask-like generators masks for both parameters. In order to control either the pitch or modulation generator, a button on the joystick has to be pushed and held. In this way, the performer can choose for instance to control only one of the two, or both generators.

Pushing and holding another button on the joystick, allows the performer to control a parameter which is used as a multiplier for the durations calculated by the algorithmic processes with the Y-axes of the joystick. Allowing the performer to change the speed of the rhythmical structures produced by the instrument.

Analysis

In order for the computer instruments to be able to react on the performers output, it has to have a notion of what that output is. Therefore, analysis of the audio output by the performer is done. I have split up the analysis into two layers. The first layer is a SynthDef analyzing technical aspects of the performers output, the second consists of a piece of code in the SuperCollider language that converts this technical data into more perceptual parameters.

The SynthDef representing layer one of the analysis analyzes a number of aspects, namely:

Pitch

I have chosen to analyze pitch by analyzing the cumulative distribution of the frequency spectrum. This is realized by using SuperCollider's “SpecPcile” UGen.

An unconventional method, but trial and error has proven it gives me the best results. Since the output of the performer can vary widely and is definitely not always pitched sound, analyzing pitch with conventional pitch trackers do not give me very satisfying results, since the data is useless whenever the sound is a little too noisy. I am not so much interested in exact pitches of sounds since I am not working with melodies in the classic sense of the word. Un-pitched sounds often have a spectral tendency around a certain frequency which is what interests me. Maybe measuring the spectral centroid of the frequency spectrum would be a more obvious choice in order to obtain this information, but the cumulative distribution of the frequency spectrum has proven to give me more pleasing results.

Loudness

Loudness is measured by analyzing perceptual loudness. SuperCollider's “Loudness” UGen is used in order to retrieve this data.

Speed

The speed is extracted by measuring the rate at which events appear. An event is understood to be one sound. The start of events are found by measuring onsets in the signal which is done with

SuperCollider's “Onsets” UGen. The time in between those onsets is measured, the reciprocal of that result, converts it into speed.

A problem with this method is that the duration between two onsets is only given when the last onset is detected. So if the performer starts playing slowly after playing fast it takes a while for the method to recognize this. Therefore, I measure the time elapsed since the last onset continuously and compare it to the last measured duration. If the elapsed time since the last onset is longer then the last duration measured between two onsets, the speed analysis is determined by the reciprocal of this new duration, which will keep on growing until a new onset is detected.

Duration

The duration is meant to represent the duration of events. It is extracted by measuring the duration in between silences, by using SuperCollider's “DetectSilence” UGen. It does give results which are a little rudimentary, but is effective.

Like in the measurement of speed, the same technique is used to see if a duration of the current event is longer than the last measured duration. If the duration since the last change from silence to non-silence is longer than the last measured duration, the output of the duration analysis will be determined by this new duration which keeps on growing until a new silence is detected.

Duration should maybe compared to the measured duration for speed. Since there might be no silence in between two events, by just detecting silences, multiple events might be understood by the analysis as one, and output one long duration instead of multiple shorter ones. If I would just compare the measured durations for speed and duration, and take the shortest for being the duration of the event, less errors would be made.

Spectral flatness

The spectral flatness outputs a value between 0 and 1. An output of 0 is measured for a single sine wave, and an output of 1 for white noise. SuperCollider's UGen “SpectralFlatness” takes care of this.

For each of these measured parameters the derivative is measured in order to determine the amount of change, which is an important musical aspect.

The second layer of analysis consists of a piece of code in the SuperCollider language that combines and scales some of the technical data retrieved by layer one, in order to extract more perceptual parameters. I have named these perceptual parameters “polarities”, inspired by the polarities used by Bjarni Gunnarsson in his compositional environment EPOC, which is described in his Master's thesis (Gunnarsson, 2012).

The extracted polarities are:

Pitch
Pitch is extracted simply by copying the first layer's pitch analysis.

Speed
Speed is also extracted by simply copying its first layer's analysis.

Density
Density is extracted by copying the mean value of the first layer's speed and duration analysis.

Loudness
Loudness is again simply extracted by copying the first layer's loudness analysis.

Intensity
Intensity is extracted by summing the first layer's loudness, spectral flatness and speed, divided by three, with weights applied to speed and loudness, 1.5 and 0.5 respectively.

Entropy
Entropy is extracted by summing the pitch, amp and spectral flatness derivatives, dividing the result by two and clipping that result between zero and one.

Phrase detection

Besides extracting polarities form the first layer of analysis, the second layer extracts phrases from the data retrieved by the first. All values of the polarities during a phrase are stored and the mean is calculated at the end of the phrase. The results are stored as phrase-polarities, which together give a general overview of the last played phrase. Also, the duration of the phrase is measured and stored.

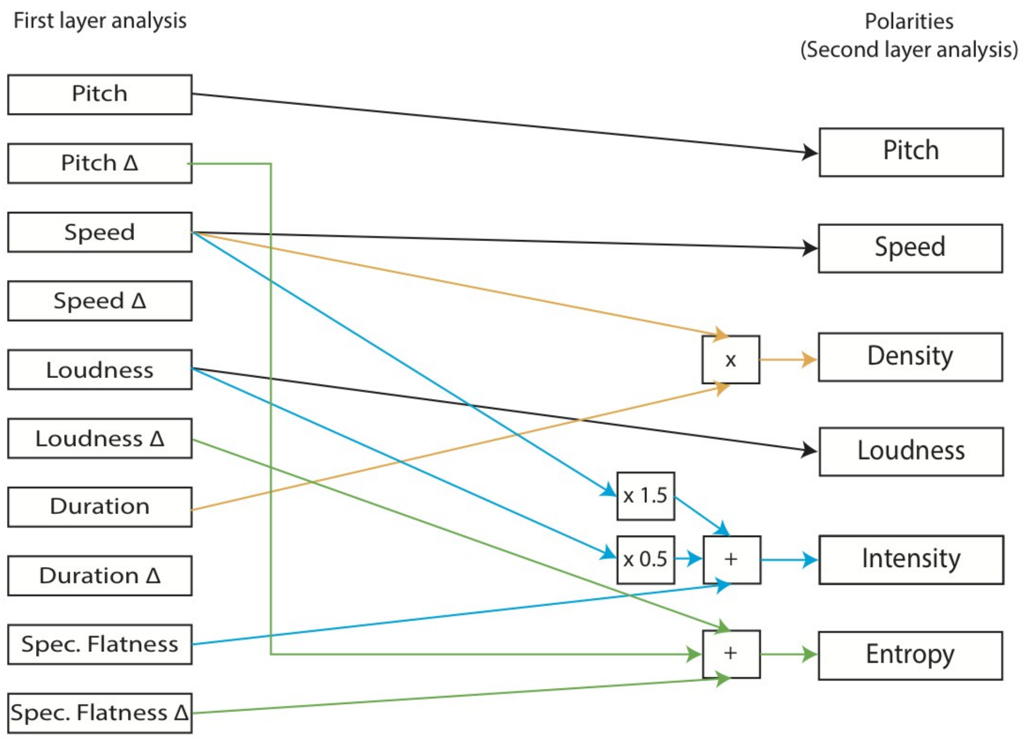

Figure 22, Schematic representation of relationship between layer 1 and layer 2 analysis

Phrase detection is done by detection the start and end of a phrase. The start of a phrase is detected if loudness is greater than 0. At that moment a variable holding the current state of the phrase detection is set to 1. While a phrase is being detected, the start of a new phrase can be detected, which implies the end of the current phrase. This happens if specific thresholds for loudness and loudness, pitch, speed and spectral flatness derivatives are crossed. The end of a phrase is also detected if there is silence for longer than 0.25 seconds.

Computer player

The computer is playing with the performer by use of “objects” and “nodes”. Objects are in essence SuperCollider's Patterns[3] which are encapsulated in a function, which adds functionality to the Pattern in order to allow it to be influenced by the polarities. Patterns represent a sequence of musical events that can be generated in real-time in wide variety of ways. Nodes are hierarchically one layer higher than objects. They are functions for making phrases by combining different objects, taking into account the phrase-polarities.

How objects and nodes respond on polarities and phrase-polarities depends on the setting of two variables, polarity-behavior and phrase-behavior. Polarity-behavior determines for every polarity if it should be followed or opposed by the computer. It is a list of values for every polarity. Different lists are stored into the software acting as presets which can be chosen from during performance. Phrase-behavior determines how the nodes, or the creation of phrases, relate to the detection of phrases. Three different phrase behaviors exist from which the performer can instruct the computer to choose. The choices that determine polarity and phrase-behaviors are random non repeating choices made by the computer, whenever it is instructed to do so by the performer. An instruction that is done by pressing a specific button on the iPad.

Objects

Objects are functions with an argument to start and stop them, an argument for the duration for the object to sound and an argument for every polarity.

Inside this function a Pattern is created, either a Pbind[4] or Pmono[5]. Both a Pbind and a Pmono will trigger a Synth and allows other patterns to be supplied for the Synth's arguments. Often stochastic patterns are used in combination with the polarities.

3 Patterns represent a sequence of musical events that can be generated in real-time in a wide variety of ways.
4 A Pbind is a Pattern that generates a sequence of Synths using other Patterns for the calculation of their parameters.
5 A Pmono is similair to a Pbind but creates only one Synth of which the parameters are changed over time.

Objects exist in four different categories: *rtm*, *dyn*, *pad* and *txt*.

Rtm stands for rhythm. The *rtm* objects produce rhythmical structures.

Dyn stands for dynamic.

Pad simply stands for pad. Objects in this category are slow sequences of long sounds that are rather static and are pitched to a certain extent.

Txt stands for texture.

Nodes

Nodes are SuperCollider functions that construct phrases by combining objects. One node has several objects assigned to it, which the node can use in order to make phrases. A phrase is implemented to be a sequence of objects within a predefined duration, in which the objects are scheduled to occur each at their own moment. The construction of phrases is done by making use of stochastic calculations as well. Phrase-polarities are often used as percentages for the chance wether a certain object will become part of a phrase or not. For instance, an *intensity* phrase-polarity of 0.7 would be a 70 percent change that a certain object will occur in the phrase to be generated.

The node that will be used is usually constant for a few minutes. What node will be used is determined by the computer with a non-repeating random choice. When this choice is made is determined by the performer, by pressing a dedicated button on the iPad.

Phrase-behavior determines how the creation of nodes is related to phrase-detection. In the current state of the software, there are three different phrase-behaviors, “oppose”, “follow” and “ignore”.
The setting “oppose” triggers a node, and thus a phrase, at the end of every detected phrase. The duration for this phrase is determined by multiplying the duration of the last detected phrase with a stochastically generated number. The “follow” setting triggers a node whenever a start of a phrase is detected. Also when the phrase-behavior is set to its “follow” setting, the duration for the generated phrase is determined by a multiplication of the duration of the last detected phrase with a stochastically

generated number. And the “ignore” setting ignores the phrase detection and determines when to start phrases based on it's own “clock”, of which the speed is determined by a stochastic function that uses the length of the last detected phrase as input. The output of this stochastic function is then also used, to determine the length of the phrase a node will create, after which it is multiplied with a number that is often stochastically obtained as well.

The reaction to phrases by the computer player is what makes it – in my opinion – convincingly interacting with the performer. It causes an interaction on a structural level, which is a very notable aspect of the music. Also the interaction on smaller time scales is of course important, equally I would say, but not in terms of clarifying the relationship between the computer player and performer, it causes more complex relationships that play a big role in the quality of the music.

As a node has only a certain amount of objects from which it can make a phrase – to be specific, about four to six – a very specific soundworld belongs to a node. This causes a certain amount of consistency throughout a passage of the performance. A very valuable consistency in my opinion, since it can cause structural movement over a larger timescale. That is controlled by he performer, by making the computer choose a new node whenever he wishes.

The amount of possibilities in terms of relationships between behaviors, polarities, phrases, nodes and objects is maybe endless. And most certainly I have not tried all. But I am not on a quest to find the best solution, rather, I am searching for an effective solution. The fact that so much unexplored possibilities still exist, gives me as a developer of the instrument the possibility to explore a wide range of musical directions. Since these technical relationships between pieces of code and functionality of the software, directly represent relationships in structural, timbral, rhythmical and other kinds of musical aspects.

Conclusion

I think an instrument for live electronic music is best designed when a balance is found in between transparency, physicality and virtuosity. All three of these concepts are equally important, and I think if they are balanced correctly, they will be strongly connected in the sense that one will cause another and vice versa. In my work as it is currently I am not yet at the point where I have found that perfect balance. I think I can still gain quite a lot because I have the feeling my ideas about designing an instrument for live electronic music have only recently become completely clear to me, something for which writing this thesis has greatly helped. But on the other hand I do not expect to find a completely perfect balance since I suppose that might only be possible in a utopian world.

In terms of playing multilayered music solo I think working with some some sort of interactive music system is an effective solution. It allows a performer to focus and play one layer completely live, and play, or cause, a multilayered piece of music at the same time. The difficulty and challenge lies in finding the balance between controlling the end result of the piece of music, and letting the system surprise the performer.

I am now at the point where I have created a program that allows me to easily add material without having to change the existing code. I think I will work with this system for some time without changing the inner workings so I can focus on making more music without having to deal with writing the less musical parts of the program. Although I think there are still much things in my system that can be improved I think it is at a state where it can produce new and interesting musical results by means of adding material.

References

Atoui, T. (2008). *Mort Aux Vaches* [CD]. Amsterdam: Staalplaat.

Atoui, T. [STEIM Amsterdam]. (2009a). *Tarek Atoui* [video file]. Retrieved from http://vimeo.com/1658451

Atoui, T. [TarekAtoui]. (2009b, July 2). *Un-drum / strategies of surviving noise 1/1* [video file]. Retrieved from http://www.youtube.com/watch?v=PaARgYKJm-o

Bahn, C. Hahn, T. Trueman, D. (2001). Physicality and Feedback: A Focus on the Body in the Performance of Electronic Music. In *Proceedings of the International Computer Music Conference* (pp. 44-51).

Bailey, D. (1992). *Improvisation: Its Nature And Practice In Music.* Da Capo Press, Inc.

Barrett, R. (2012). *Improvisation 1.* Presentation at the Institute of Sonology, December 21, The Hague, the Netherlands.

Carey, J. (2011). Presentation at a SuperCollider users group meeting, May 11, Amsterdam, the Netherlands.

Carey, J. [BaltimoreSDIYGroup]. (2012, November 18). *Jeff Carey's performance on 11/03/2012 (video 10 of 14)* [video file]. Retrieved from http://www.youtube.com/watch?v=yQ1jUh-2rpM

Essl, K. (n.d.) m@zeº2. Retrieved from http://www.essl.at/works/maze.html

Essl, K. (2006). Improvisation with Computers. Retrieved from http://www.essl.at/bibliogr/computer-improv.html

Hunt, A. Wanderley, M. M. & Paradis, M. (2002). The importance of parameter mapping in electronic instrument design. *Proceedings of the 2002 Conference on New Instruments for Musical Expression* (pp. 1 – 6).

Jordà P. S. (2005). Crafting musical computers for new musics' performance and improvisation (Doctoral thesis, Universitat Pompeu Fabra), Retrieved from http://mtg.upf.edu/files/publications/PhD2005-sjorda.pdf

Kirn, P. (2012). Finger-Drumming Video EP: Three Tracks, Played Live on MPD24, Zynewave Podium. Retrieved from http://createdigitalmusic.com/2012/12/finger-drumming-video-ep-three-tracks-played-live-on-mpd24-zynewave-podium/

Krefeld, V. (1990). The Hand in The Web: An Interview with Michel Waisvisz. *Computer Music Journal, 14(2)* (pp. 28 – 33).

Lewis, G. (1950). Improvised Music after 1950: Afrological and Eurological Perspectives. *Black Music Research Journal, 22 (*pp. 215 – 246).

Lewis, G. (2000). Too Many Notes: Computers, Complexity and Culture in Voyager. *Leonardo Music Jounal, 10 (*pp. 33-39).

Lippe, C. (2002). Real-Time Interaction Among Composers, Performers, and Computer Systems. *Information Processing Society of Japan SIG Notes, 2002(123) (*pp. 1 – 6)

Marshall, M. Hartshorn, M. Wanderley, M. M. & Levitin, D. (2009). Sensor Choice for Parameter Modulations in Digital Musical Instruments: Empirical Evidence from Pitch Modulation. *Journal of New Music Research, 38(3) (*pp 241 – 253).

Ostrowski, M. [cycling74com] (2012). *An Interview with Matthew Ostrowski, Part 2: The Glove* [video file]. Retrieved from http://www.youtube.com/watch?feature=player_embedded&v=8k-UcvGQbko

Pluta, S. (2012). Laptop Improvisation in an Multi-Dimensional Space (Doctoral Thesis, Columbia University). Retrieved from http://academiccommons.columbia.edu/download/fedora_content/download/ac:146917/CONTENT/Pluta_columbia_0054D_10721.pdf

Rowe, R. (1992). Machine Listening and Composing with Cypher. *Computer Music Journal, 16(1) (*pp. 43 – 63).

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

Waiswisz, M. (n.d). Bio. Retrieved from http://www.crackle.org/whoami.php

Waiswisz, M. (2006). The Hands. Retrieved from http://www.crackle.org/TheHands.htm

Wanderley, M. M. (2001). Gestural control of music. *International Workshop Human Supervision and Control in Engineering and Music* (pp. 632-644).

Wessel, D. & Wright, M. (2002). Problems and Prospects for Intimate Musical Control of Computers. *Computer Music Journal, 26(3) (*pp. 11–22).

Zadel, M. (2006). A Software System for Laptop Performance and Improvisation (Master's thesis, McGill University), Retrieved from http://www.music.mcgill.ca/~zadel/research/publications/zadelmastersthesis.pdf