Research Report

The Quintessence Project: Re-envisioning Mediæval Music through the Lens of Electroacoustic Technology

Helen Thomson

As a performer, my interest in musicology has its origin in its application to performance practice. The project discussed here was originally conceived as a way of solving a set of performance problems, but the process has taken me in various unexpected directions, from the investigation of the history of Gregorian Chant performance practice, to an examination of video tracking hardware and software.

I have long wished to integrate my various divergent (and arguably abstruse) areas of interest and specialisation: I have long been a performer of Gregorian Chant and other very early music; I have also been a solo performer of contemporary classical music since the age of ten; and I have an abiding and as yet largely unexplored interest in improvisation. The Quintessence Project represents an attempt to hybridise all of these interests. It is also an attempt to address various issues—both real and perceived—that exists for both contemporary and early music performance practice as they currently stand.

Early music, and in particular mediæval music, is often perceived as emotionally flat or musically naïve. Rightly or wrongly, the classical music norm is orchestral or operatic, which tends to make it difficult to entice an audience to attend a performance with unfamiliar qualities, particularly where these qualities are understated (the spare, meditative nature of a Chant performance is a good example). It is similarly difficult to arouse audience interest in contemporary, avant-garde musical programmes. Finally, early and contemporary music programmes frequently lack visual and dramatic interest.

Gregorian Chant in particular suffers from various pragmatic performance difficulties. Because it is rhythmically free and monodic, it is reasonably difficult for a group of singers to arrive at a good ensemble sound without a great deal of rehearsal. Chant is also notated differently from modern musical notation, so that specialist knowledge is required in order for the singers to perform untranscribed music. Where the notation incorporates semiological markings, the notation can become extremely complex, and its interpretation contentious. All these factors mean that Chant, despite being an extraordinarily beautiful form, is not very often performed in a concert setting.

My agenda was to rehabilitate Chant, or adaptations of Chant, as concert repertoire; to recast Mediæval music using modern techniques; and, if possible, to do these things as a solo performer. I also wished to incorporate various dramatic, visual and improvised elements which are not normally seen in this kind of repertoire.

The solution lay in the use of technology. With recent advances, it is now possible to manipulate found sound in real time using domestic, off-the-shelf hardware. In order to create polyphonic textures, my initial aim was to find, or have built, a programme which would allow two types of real-time looping. First, there is a canon looper, whereby the audio input is begun, and playback is triggered to begin before the original audio input is finished, and second, there is a looper involving the input of layer upon layer of different sounds, known as the live looper. The two looping mechanisms are shown in Figures 1 and 2.

Figure 1. Canon Looper Schematic

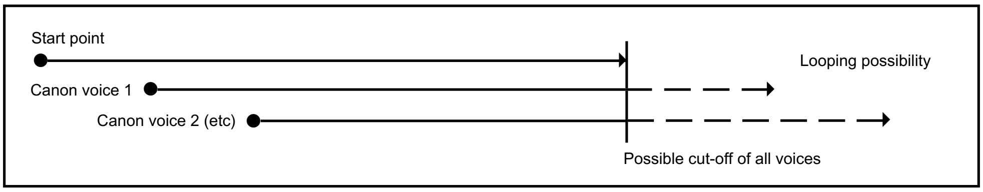
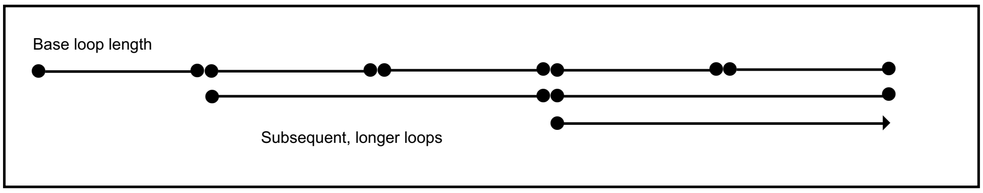


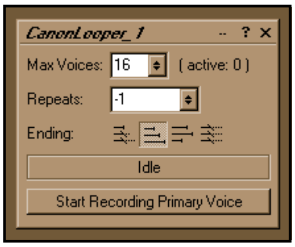
Figure 2. Live Looper Schematic



On the advice of Melbourne-based composer Ros Bandt, I engaged Ross Bencina, creater of the AudioMulch real-time audio manipulation environment, to build these looping contraptions. The process of evolving a specification for these contraptions was one of the harder intellectual puzzles I have ever tried to grapple with.

First, I had to conceptualise the result, and Figures 1 and 2 represent a maximally pared-down guide to what I had in mind. The next step was to think of all the parameters I might wish to modify. In the case of the canon looper (Figures 1 and 3), this was relatively straightforward: Ross and I thought that I may want to specify ahead of time how many voices I would be using, but we also wanted to build in the possibility of deciding on the number of voices 'on the fly'. We also wished to be able to specify the number of times the canon would repeat, or to have the performer decide that at the time of performance as well by setting the repeat parameter to -1, as shown in Figure 3. Finally, we wanted to be able to specify how the looper finished. The three modes we implemented were: first, all voices end

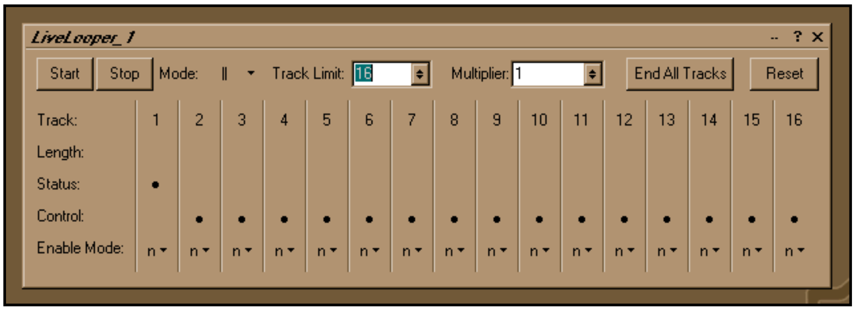
Figure 3. Canon Looper Contraption User Interface



in sequence after the first voice ends; second, all voices end immediately; and third, all voices end in sequence after the next voice ends, whether it is the first voice or not. We established that the remainder of the control parameters could be controlled by a single, multifunctional button. The text on the button initially reads 'Start Recording Primary Voice'. It then cycles through 'Start First Cannon Voice' (which also effectively sets the length of time that elapses between each voice beginning and the next beginning), 'Stop Adding Cannon Voices' (this step is skipped if you have specified a maximum number of voices, and already reached that maximum), 'Stop Recording Primary Voice' (which begins the a loop from the beginning of the piece unless you have set the number of repeats to 0), and 'End', which stops the canon according to the ending mode selected (unless it has cycled through the number of repeats specified, in which case the canon automatically ends according to the specified ending mode).

Specifying the parameters required for the canon looper was relatively straightforward. Working out every parameter that I might wish to control in the other looper was a much trickier proposition.

Figure 4. Live Looper User Interface



The jumping off point, as it were, for the 'Live Looper,' is Loop 1, which sets the primary loop length. Ross has implemented the system such that subsequently recorded loops may be either the same length as, or a multiple or fraction of loop 1, using the 'multiplier' parameter. There are three modes in which the loops may be laid in. The manual (II) mode, which in Figure 4 you see selected, records a loop when the start button is pressed and stops recording that loop when the stop button is pressed. It remains idle until the start button is pressed again, at which point the next track begins recording. The 'chained' recording mode begins recording when the start button is pressed, but the difference is that the stop button both stops recording the current track and begins recording the next track. The 'continuous' mode assumes that all loops are going to be the same length as the immediately previous track, and automatically stops recording a given track at that length, before moving on to the next track and continuing recording. These latter recording modes were implemented in order to facilitate the quick, easy and, in the case of continuous mode, actually untethered building of polyphonic textures.

One of the most complicated processes was solving the stopping/starting problem. I specified that I wished to be able to stop all the loops together or independently, according to various different stopping modes: immediate, end of track, end of quantization cycle, end of primary cycle (these two things are different if the multiplier is a number other than 1), and end of composite cycle. I also wished to be able to restart the loops according to the same parameters.

Getting this to happen presented a serious conceptual challenge: in particular, we needed to decide exactly how we were to define 'stopping'. This is a much more complex task than one might imagine. If we define 'stopping' as 'muting the track at a certain defined point', then there is little difficulty, but consider the following: were I to have recorded a primary loop of length t, and have laid in two subsequent loops of length 2t and 3t, respectively, then loops 2 and 3 are going to phase in and out with one another. If I stop loops 2 and 3, and subsequently wish to restart them strictly in phase, then 'stop' defined as 'mute' is not adequate. While simultaneously unmuting them at a 1t boundary might mean that they start playing back simultaneously, there is no guarantee that they will do so in phase.

However, there are stopping and starting scenarios where stop and start *must* mean mute and unmute. If I wish to start a stopped loop immediately (that is, not at the next 1t boundary, but straight away), in order for the loops not to drift out of synchronisation with one another, this kind of restarting must be an unmute. The solution we evolved in the end involved the q (end of quantization cycle) and 1 (end of primary cycle) loops resetting the phase of the track to the start as playback began, but all the other enable modes being straightforward mute/ unmute.

The preceding discussion gives an idea of the challenges that present themselves when developing a new music-making tool. After a lot of discussion and experimentation, the loopers are complete and functional, and using this technology I have already been able to adapt various early pieces (including the 'Sanctus' from the Gregorian Mass XIII, Hildegard's *Kyrie* and *O Virgo Splendens* from the *Red Book of Montserrat*) for performance at my final Master's degree recital at the University of Melbourne. From a performance practice perspective, however, the bulk of the work still lies ahead.

At this stage, in order to control the loopers and other contraptions, the performer still needs to be in front of the computer. The next step, and the one which will, I hope, set the Quintessence Project apart, is to untether the performer from the instrument: my aim is that audio output will be controlled by movement through the performance space. I have considered various options for a gestural control interface. I have thought about ultrasound, infrared, laser and accelerometer/gyroscope-based systems. In most of these cases, in order for the performer to be untethered, some kind of electronic equipment would have to be carried or worn by the performer. This equipment is likely to be reasonably complex and possibly also quite fragile, and it would need to be custom built. Another area to explore is the possibility of audience participation, which means that any trackable device would have to be fairly robust.

For this reason, I began to look into a solution which incorporates video tracking. This method has its own challenges, the largest of which is that the tracked object might become occluded from the tracking mechanism. The amount of computational power required for a truly real-time control system (I anticipate that audio manipulation and video tracking will need to be done on two different machines) is also problematic. As no specialised equipment is needed, and given the fact that almost anything can be tracked—from hands or other body parts, to Nerf balls, for example (which allows for the abovementioned audience participation possibility, as well as opening up various throwing or bouncing possibilities)—I was convinced to continue along this path. I am currently experimenting with a freeware program called Eyesweb, which, among other types of video processing, is capable of of tracking objects of a certain colour and outputting MIDI data, which would then be used as a control mechanism for AudioMulch.

Once this work is done, I plan to return to the development of various effects that are not yet easily achievable in real time. In particular, I wish to implement four things: first, a time-stretch algorithm, so that a sound can be played back more slowly than it was initially recorded, but at the same pitch; second, a polyphony algorithm, where a midi keyboard can be used to provide desired pitches for the real-time pitch-shifting of an incoming sound; third, a harmonic picker, where a pitch tracker follows the pitch of an input sound, filters the sound for a given formant, and outputs that formant; and fourth, a spatialisation contraption, which would allow the performer, via the gestural control interface, to control the sound emerging from a surround speaker system, so that it might emerge from any given speaker, or rotate through each speaker, or emerge randomly from each speaker.

In the longer term, I am interested in developing an electroacoustic instrument that will evolve into a particular form based on the preferences of the user, a footpedal, and the gestural control interface alluded to above. In the 'learning' phase of the process, once an audio loop is entered, the computer would analyse the sound (percussive or melodic, changing slowly or quickly, the timbre of the sound), and apply, at random, an audio manipulation to that sound. The user then gestures to alter various parameters of that sound, and hits a 'yes' or a 'no' pedal, depending on whether the modification meets with the approval of the user. Progressively, the user and the computer then evolve a repertoire of modifications to a number of different types of sound. Once the learning process is complete, the user can then input different types of sound, and use a pedal board to choose between possible modifications, and control the parameters of those modifications by moving through the performance space.

The electroacoustic field presents a wealth of musical, visual and dramatic opportunities. My hope is that the very sophistication of the technology will allow it to form so seamless a part of the performance that it becomes transparent, and the performance assumes a fluid, organic dimension, which is simultaneously facilitated by, and yet almost at odds with, its technological origins.