Sight, Sound, the Chicken, and the Egg

Audio-Visual Co-dependency in Music

A folio of audio-visual compositions, written commentary, and accompanying materials, submitted in fulfilment of the requirements of the degree of Doctorate of Philosophy

by

Simon Katan

School of Arts
Brunel University

September 2012

Abstract

Amongst the modern day abundance of audio-visual media, where sounds represent everything from the swooping of virtual cameras through 3D spaces to the pressing of buttons and receiving of emails, and conversely where VJs routinely accompany live musical performance with an increasingly sophisticated language of abstract computer animation, the notion of music as a necessarily exclusively aural medium seems somewhat out of place. Psychological theories relating to the cognition of sound, in particular physical schema, accounting for the ubiquity of vertical plane pitch metaphors in most musical cultures, provide evidence of a deep-rooted spatially informed understanding of sound thus providing a common ground for both sound and vision in music. Furthermore, Western Classical composition is rife with examples of visually conceived forms from Bach's *Crab Canon* (1747) to Xenakis' architecturally inspired *Metastasis* (1954). However, in practice the gap between the listener's auditory experience and the composer's visual concept is often insurmountable.

Rising to Schaeffer's call for "Primacy to the ear!" (Schaeffer, 1967, pp. 28-30), acousmatic composers have sought to derive music exclusively from experientially verifiable criteria. However, in its pervasiveness of other musical genres, no doubt aided by technologically and commercially driven domination of the pre-recorded over the live listening experience in the latter half of the twentieth century, such an approach has lead to the neglect of visual aspects in the live performance of much art-music. This research aims to begin to redress this balance through the composition of, largely computer realised, audio-visual works whose conception arises not from a superimposition of one medium upon another, but through the very relations between the media themselves. Utilising modern computers' ability to synchronise physical and virtual visual events with synthesised sound in real time not only affords composers an invaluable tool for enhancing listener's perception of formal structures but also implies causal relationships between the sonic and the visual which can provide a base of intuitive understanding on which more complex formal ideas can be built.

Acknowledgements

I would like to thank Professor Christopher Fox for his expertise, encouragement, and generosity with his time, and also Carl Faia – I left every supervision feeling more enthused, focussed and determined to reach my goals. Thanks to Reynaldo Young, for inspiring me to be a good composer, the many incites over the years, and for proof reading this thesis. Other good friends who have generously devoted their time to give me invaluable advice about C++, Computer Vision, 3D geometry, SuperCollider, Linux and the rest are Evan Raskob, Pete Todd, Mauritius Seeger, Paul Prudence, Dan Stowell, and Ryo Ikeshiro.

None of this work would have been possible without the selfless hard work of the open source communities who develop and maintain the resources I have used, especially the SuperCollider and OpenFrameworks contributors.

I would also like to thank the other members of Brainer, Cimeon Ellerton and Luke Fraser for their blind dedication to an unknown cause and endless Sunday afternoons in my living room. Further thanks goes to Filipos Kanikaris for his choreographic direction and magnificent patience, Tim Trimingham Lee for his directorial comments and camera work, and Richard Thomas for his sound recording skills. Thanks to the Composition Students of Trinity Laban 2010 who performed in *Les Escaliers Méchaniques,* to Neil Luck my dedicated apprentice who worked on the Spitalfields installations, and to Carolein Teunisse for her computer mapping skills. I'd especially like to thank David Austin for rendering my computer graphics.

I'd like to thank Brunel University for their financial assistance through the Isambard Scholarship, facilities and support. I also thank the commissioning bodies, Borealis Festival, Spitalfields Festival, MoTA, SoundWaves, Lewisham Arts, and The Portman Gallery.

Finally thank you to my friends many of whom are already listed above for their support, interest, patience, and concern over the past three years and of course to my parents and sister for their love and support.

Table of Contents

Accessing Online Resources

Each project is accompanied by video documentation and other related materials including SuperCollider implementations, C++ source code, and descriptive and performer scores where relevant. This information is stored online and can be accessed via the hyperlink displayed at the top of each section. In order to ensure that this material is viewed only in the context of this thesis these pages have been password protected. To access them simply log in with the username "**examiner**" and the password "**chickenegg**" (case sensitive) when the dialogue box appears in your browser. Source code is contained in zip files, video footage can be viewed directly in the browser or downloaded for later viewing by contextual clicking and selecting "Save target as…"

Class Diagrams

In order to improve readability for the lay reader, class diagrams are idealised using a simplified form of Unified Modelling Language whereby a single arrow type denotes ownership. Towards these ends, some class names have been changed from the source code files in order that naming conventions remain consistent throughout the written text. Where this has happened, the details of name changes are written into the README file in the source code folder. In addition to this, very small classes and structs of limited significance have been omitted. For non-programmers, it's important to note that rather than indicate order of process such arrows simply show whether one class has knowledge of another class and all its subclasses.

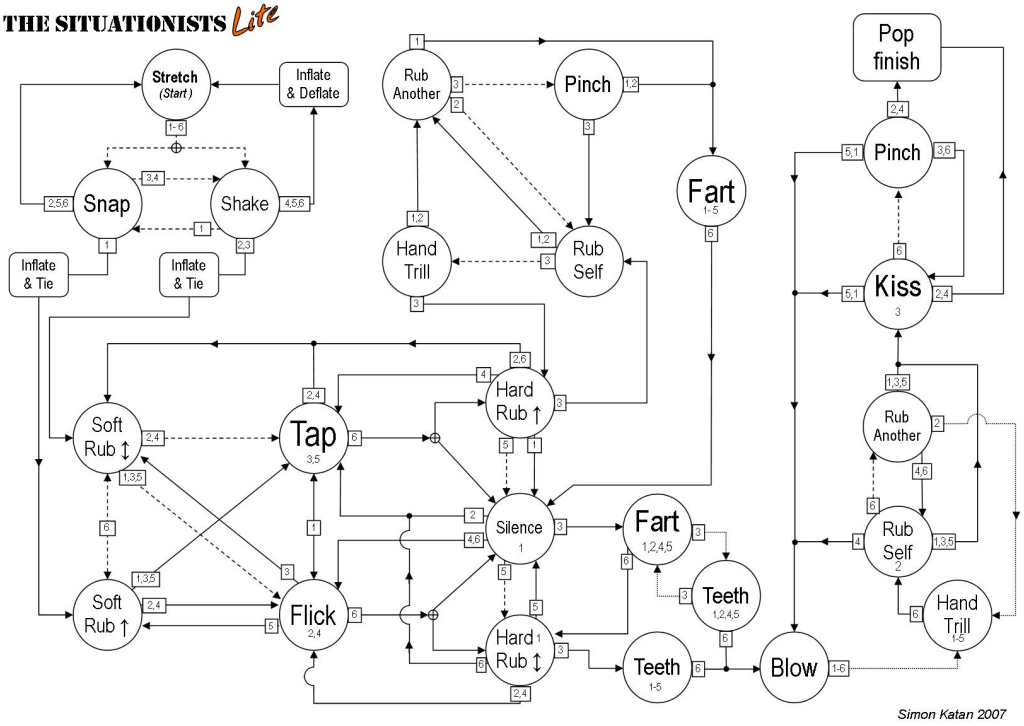
List of Figures

Introduction

Shortly before embarking on this doctoral thesis, I attended a lecture at the Guildhall School of Music and Drama, delivered by Tom Johnson whose music I admire greatly. Aside from explaining his ingenious and mathematically complex treatments of pitch and rhythmic material, he also performed a number of his works for solo piano including a piece from his piano cycle *Counting Keys* (1986). I single this particular piece out not because of the complexity or beauty of its form. Indeed it was of a variety Johnson referred to as 'dumb music', meaning that the formal processes were deliberately primitive and on the surface level – you'd have to be dumb not hear them. Rather I was struck by the failure of the audience comprising students, composers, and performers, when questioned by Johnson, to describe precisely the process that had been heard. Somewhat dismayed, Johnson patiently delivered a clear verbal description before proceeding with a repeat performance. "Do you get it now? " he asked. We all responded with a sheepish nodding of heads, our intellectual prowess forever tainted in his eyes, although in retrospect, it seems that Johnson's notion of 'dumb music' came off worse than the audience from the incident. After all, how can one be too dumb to get 'dumb music' ?

Such disparity between composers' conception and audience's perception of formal structures was the starting point for this thesis. The problem was familiar enough to me through my own experiences with my performance ensemble 'Halal Kebab Hut.' This group of six improvisers had been formed to perform my algorithmic compositions for humans, where the musicians' scores consisted of interlinking sets of rules that, in combination, created complex formal structures (Figure 0.1). Despite the success of the compositions in terms of their contribution towards the improvisational context, the musicians expressed frustration that the audience had no way of appreciating the algorithmic structures. They wanted the audience to share in their experience of being nodes in a complex rule-based system.

Figure 0.1 Halal Kebab Hut - an algorithmic score



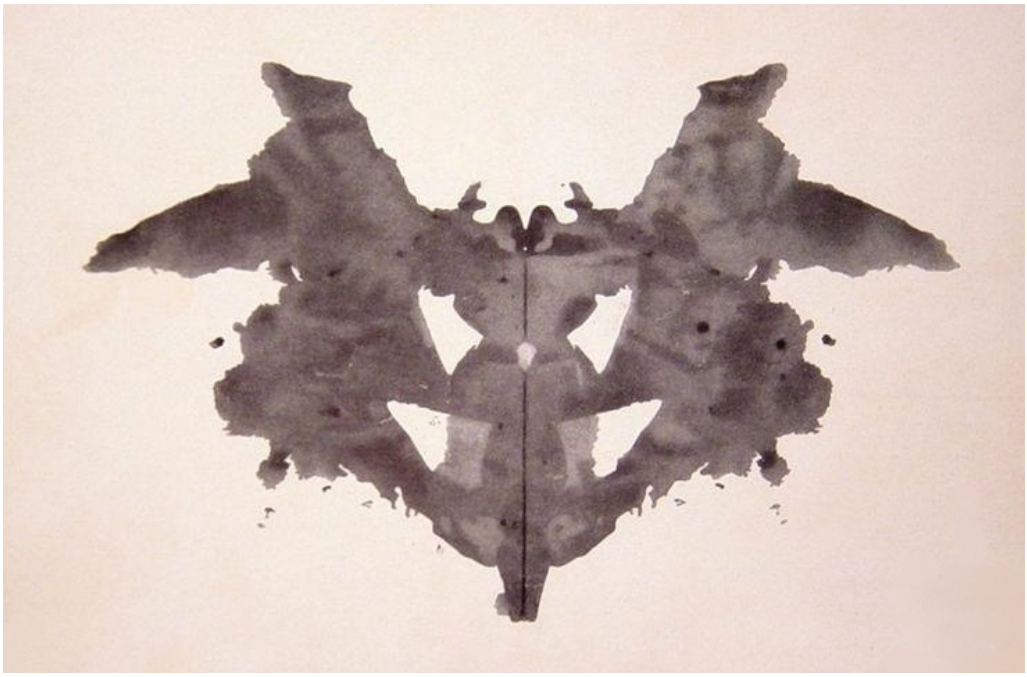
However, the problem extends beyond contemporary and experimental music. Psychologists Tillman and Bigand (Tillmann & Bigand, 2004, p. 216) cite various experimental studies demonstrating our low capacity for perceiving global musical structures within tonal music. For example, one such experiment by West-Marvin and Brinkman asks specialist musician participants to identify whether an excerpt, shorter than two minutes, starts and ends in the same tonality. In total, sixty-four percent succeeded in correctly identifying that the tonality had changed, though there were differences across the various specialisms – seventy-one percent for music theorists, sixty-one percent for performers and somewhat amusingly forty-eight percent for composers. Tillmann and Bigand's findings lead them to support Levinson's view of music's form as, "continuational and successional, not spatial and architectonic" (Levinson, 1997, p. 161). However, Nick Collins points out that their case is "not watertight, particularly with regard to memory for motifs, recognition of development and recurrence of sections" (Collins, 2009, p. 105). Nevertheless, one can't help but agree with his view that "the composer's disregard for perceptual criteria … pervades contemporary music" (Collins, 2009, p. 111).

It is a flaw of which Trevor Wishart is particularly critical, attributing the growth of the gap between conception and perception of form to the hegemony of scribe orientated culture which manifests itself in music through the notational system.

> "…with the increasing domination of notation, there has been a move towards Platonic idealism in our conception of what music is. In the most extreme cases, music is viewed as an essentially abstract phenomenon and the sound experience of essentially secondary importance" (Wishart, 1996, p. 35).

By way of example, Wishart cites the concept of retrograde as found in serial music such as the perfect arch-form of *Der Mondfleck* from Schoenberg's *Pierrot Lunaire* (1912), but also in Bach's so-called *Crab Canon* (1747). Of particular interest to me is the marked difference between the ease with which the relationship between original version and retrograde is seen on the score, and the "considerable aural retentivity and the performance of a rapid feat of mental inversion" necessary to grasp the same relationship rendered in the sonic sphere (Wishart, 1996, p. 39). Indeed, as my own informal experiments with playing audiences of musicians and non-musicians *Crab Canon* and asking them to identify the formal relationships confirm, whilst the task of observing inversion and retrograde relationships of even complex and abstract images is trivial for most (Figure 0.2) hearing such relationships is a highly specialised skill available to only a fraction of trained musicians.

Figure 0.2 Symmetry in a Rorschach blot



As Wishart mentions, the key to this disparity is memory. The permanence of the objects in the visual world acts as a memorial prop, allowing us to repeatedly scan objects in order to discern the various relationships between them. Sounds, on the other hand, are fleeting; recognising relationships between sections of music requires the accurate recall of all the sounds and their temporal order. In this regard notation can act an extremely useful tool for

the musically literate listener, not only providing a reminder of past events, but also allowing its readers to peer into the future as the piece is progressing, thus significantly improving their capacity for pattern recognition. All this tallies with a feeling I often have when listening to formally complex music akin to scrabbling around in a darkened room with a dim torch. I try to understand the geography of the room, but fail to remember enough details and the relationships between them to do so. Following a score, in effect turns on the lights, allowing us see the whole and absorb all of its intricate patterns and interrelations.

Objecting to the social exclusivity of notation, and insisting on the primacy of sound in music, Wishart advocates the exclusive use of experientially verifiable criteria for composition. However, I would argue that given the increasing prevalence of formal constructions in our contemporary digital world, it would be a strange thing indeed to banish such forms from the sphere of music. The capability of modern computers to synchronise sound synthesis with generative graphics in real time offers an opportunity for the development of a music that is both aural and visual in its conceptualisation and realisation. By these means this thesis envisions a form of music where the visual amplifies perception of formal structures, allowing audiences a real time appreciation of form, hitherto inaccessible or solely reserved for those able to study a score.

Such aims warrant some discussion as to what is meant by appreciation of form. In discussion of Levinson, McAdams refers to a distinction between “implicit and explicit apprehension of form” (McAdams, 2004, p. 299). Whilst the model of perception that Tom Johnson envisaged for his ‘dumb music’ is an example of explicit apprehension, experiments such as Reber’s work with artificially constructed grammars demonstrate implicit learning (Reber, 1967, p. 855). In this case, a group of participants tasked with memorising series of sentences manufactured via an artificial grammar and subsequently asked to distinguish grammatically correct sentences from incorrect ones, demonstrated a high level of efficiency in learning despite being unable to articulate the rule based structure. In the sphere of music, Tillmann and Bigand note that “despite the complexity of the tonal system, experimental studies in music cognition have shown that sensitivity to musical structure does not require explicit learning: nonmusician listeners tacitly understand the context dependency of events’ musical functions and, more generally the complex relations between tones, chords and keys” (Tillmann & Bigand, 2004, p. 212). Such findings reveal notions of transparent form to be somewhat reductive. Was Johnson’s music really rendered invalid by the inability of the audience to articulate its formal structure? Were

there not concurrent relationships, perhaps more amorphous or harder to verbalise, that the audience implicitly understood? Indeed, what would have been the value had the transfer of form from score to audience been successful?

In computer music, the hierarchically tiered nature of software, renders impossible Steve Reich's conception of "a compositional process and a sounding music that are one and the same thing" (Reich, 2002, p. 34). No matter how rigorously the composer renders their processes, there will always be lower-level formal constructions such as libraries, compilers, and ultimately processors, whirring away beneath the musical surface. T.O.P.L.A.P's draft manifesto attempts to lay down some dogma on the issue in the field of live coding. Its fifth demand states "Code should be seen as well as heard, underlying algorithms viewed as well as their visual outcome," although quite how underlying the algorithms need to be remains unspecified (T.O.P.L.A.P, 2010). In practice, live coders simply project their working screen, which depending on the language and how many variables and routines have been preloaded into the environment reveals more or less about the underlying process. For example, Thor Magnusson performs with his own language Ixi Lang, (Magnusson, 2011) which is specifically designed to present extremely simple code for lay audiences. The language sits on top of SuperCollider controlling its synth definitions and samples. As the audience only get to see Ixi Lang one could argue that Magnusson is violating T.O.P.L.A.P's demand and obscuring the underlying algorithm. However, one could equally argue that SuperCollider's language does the same for its server and so on. Collins, himself a founder member of T.O.P.L.A.P, comments on the practice of beginning from blank screens that "Depending on the level and standard libraries of the programming language itself, and the nature of any user-prefabricated libraries and facilities in their live coding system, the actual challenge can very much vary." (Collins, Live Coding Practice, 2007)

With this in mind, the visibility or audibility of formal processes becomes, not a binary tenet of any centralised dogma, but a matter of degree to be decided at least at the level of individual artist, if not within the work itself. Indeed a single piece of music may simultaneously foster implicit, and explicit parsing. Perhaps then a more useful approach might be to view forms in the context of their intended effects on the listener, including formal transparency itself amongst the structural elements. For example, for a hypothetical listener hearing a recording of a Bach fugue, the initial presentation of subject and answer lay bare the material and a concept for its organisation. This explicit introduction serves to encourage them into a rational interpretation of the piece, which is reaffirmed by the

repetition of the process with each subsequent voice entry. However, as the treatment of the material becomes increasingly elaborate, this rational interpretation eventually becomes impossible to maintain. Rather than suddenly being confronted with a frustrating and incoherent noise, the listener falls back to a more localised interpretation based on their implicit understanding of tonal phrase construction. However, the recognition of fragments of the subject in the texture combined with the conceptual understanding of Bach's compositional method and reputation as a master of polyphony, leads the listener to assume that the various lines must still be logically related even if they are unable discern the relations. Awed by Bach's skill in having outwitted them, the listener resolves to listen to the recording again with even greater focus.

Viewed from this angle, the Bach fugue balances transparency and obfuscation to effect a powerful manipulation of the listener with the intention of submerging them in a labyrinthine world. Interestingly, Bob Snyder's descriptions of memory sabotage through musical form by experimental composers such as Feldman, Cage, and La Monte Young (Snyder, 2000, p. 254) imply a similar kind of listener manipulation albeit with different motivations. Such a view of music reception is similar to McAdams' concept of "the online 'in-time' listener experience solicited during the course of a work, founded in human cognitive capability" (Collins, 2009, p. 105), and it is this direction in which the practical work of this thesis is aimed. To clarify, as opposed to utilising visual media simply as a means of illuminating idealised musical formulations, the aim is to realise compositions that, though using perceptual advantages of the visual sphere, nonetheless takes a heterogeneous approach to the perception of form.

Despite the predominance of the visual influence of the score in contemporary classical music, the latter half of the twentieth century saw a gradual negation of visual aspects in the performance of the genre. One only need think of the highly precise conducting style exemplified by Pierre Boulez, or the common practice of cladding all performers in black. In free improvisation, similar practice can be found in the restriction of gestural body movements, in the pseudo-scientific performance style of the lower case scene. In some cases, such disregard for the visual yields unintended negative consequences for the audience. For example, I've often experienced solo performances where, so as to avoid awkward page turns, the musician reads off of multiple music stands lined up across the stage. As the performer moves onto the second stand after what seems like an eternity, they unwittingly signal just how long the piece has left to go, turning themselves and the music

stands into a physicalized form of progress bar. It's no wonder that so many classical audiences choose to shut their eyes.

Such oversights appear to stem from an obsession with media specificity in music. Seth-Kim-Cohen points out that music is the only medium which includes as part of its discursive vocabulary a term for foreign matter – the so-called 'extra-musical' (Kim-Cohen, 2009, p. 39). All that is given this label is considered at best a superficial decoration, and at worst a distraction to be ignored. Nowhere is this attitude applied more fundamentally than in acousmatic composition. Indeed, the term itself refers to the practices of Pythagoras, who would lecture to his students from behind a curtain to encourage them to focus only on his words without the distractions of his gestures and facial expressions. In the context of music acousmatic implies a form of listening where sounds are disassociated from their context in order to focus attention towards acoustic qualities of the sounds in and of themselves.

It is not surprising then that the most extreme negations of the visual are found in this genre. For example, in performances by Francisco Lopez who objects to making the performer the visual focal point of an electronic music performance, audiences are seated in concentric circles facing away from the stage and towards the speakers. Just to be sure there's no accidental interference from the LED lights on his equipment or the glow of the computer screen, he covers it with a dark cloth. However, these measures are somewhat redundant as he also strongly encourages his audience to wear provided blindfolds. Interestingly such practice creates quite a spectacle although it is not clear whether this is intentional on the part of Lopez, who according to Kim-Cohen is "blissfully (if problematically) naïve regarding connotations of his extended text" (Kim-Cohen, 2009, p. 124).

Figure 0.3 acousmatic listening in a Fransisco Lopez performance

I'm not unsympathetic to the idea of audio-only music. The dislocation of sound from source engages the imagination in a different way. Denis Smalley advocates Luc Ferraris' mode of reduced listening stating that "to find out what happens in the life of a sound or sound structure, or what attracts us about a sound quality or shape, we must temporarily ignore how sound was made or what caused it" (Smalley, 1986, p. 63). However, it's important to realise that the acousmatic view of music, as pervasive as it is, is a relatively recent one, which gained influence over the course of the 20th century as recorded sound gradually came to replace performance as the predominant mode of dissemination. Historically, the splitting of sound from sight was considered something quite unusual. Musicologist Richard Leppert cites the long tradition of positioning musicians offstage in late medieval mystery plays, Italian renaissance pastorals, through to Wagnerian Music Dramas at Bayreuth to conjure up magical and mysterious effects. Leppert asserts, "For much of Western History, at the most fundamental levels of human perception, the sound is the sight and the sight is the sound" (Leppert, 1993, p. xx). Rather than concern himself with the communication of formal structures, he approaches the visual in musical performance in terms of its wider social role.

> "Precisely because musical sound is abstract, intangible, and ethereal – lost as soon as it is gained – the visual experience of its production is crucial to both musicians and audience alike for locating and communicating the place of music and musical sound within society and culture." (Leppert, 1993, p. xx)

Support for such a view can be found in Kim Cascone's critique of laptop performance in 2003 where he complains that "during laptop performances, the standard visual codes disappear into the micro-movements of the performer's hand and wrist motions, leaving the mainstream audience's expectations unfulfilled" (Cascone, 2003). Here Cascone flips acousmatic values on their head by stating "Spectacle is the guarantor of presence and authenticity, whereas laptop performance represents artifice and absence, the alienation and deferment of presence." In other words, in its performance, music, like all the other performance arts, has a stake in the visual as well as aural.

Such views coincide with psychological research, from Thompson, Graham, and Russo, which provides empirical evidence of visual aspects of performance, such as facial expressions and bodily gestures, readily influencing musical perception on multiple levels (Thompson, Graham, & Russo, 2005, pp. 203-227). At a basic level, visual information is able to signal the timing of musical events, focusing listeners' attention towards critical acoustic information at specific moments in time and enhancing musical intelligibility. For example, in analysis of footage of an instrumental performance by B.B. King, Thompson *et.al* note how King's facial expressions closely track his guitar sounds and not the accompanying instruments, serving to draw attention to his nuanced treatment of individual notes and away from larger-scale structure. At a perceptual level, facial and bodily gestures can signal important melodic, harmonic, and rhythmic events. In one experiment, using the same footage of B.B. King, it is shown how King's facial expressions heighten perceptions of consonance and dissonance, another experiment demonstrated that a trained singer was able to convey melodic interval size through facial expression alone. Thompson *et al.* comment that "facial expressions may reflect the additional concentration that is needed to perform notes or passages that are unexpected or tonally unstable", or "convey points of closure, intervallic information and points of expectancy fulfilment violation". In this way performers present themselves not only as producers of sound but also as fellow listeners, "highlighting the musical activity as a shared experience" (Thompson, Graham, & Russo, 2005, p. 204).

Aside from performance, the visual also plays a large role in music's composition often through notation's two-dimensional spatialisation. Indeed it is this system and its prioritisation of the discrete and quantifiable parameters of pitch, harmony and rhythm over the phenomenological ones of timbre and texture which Wishart and the world of acousmatic composition has sought to overturn. Wishart blames notation for a reorientation of our conception of music away from the aural and towards the visual product of the score

which he sees as a facet of elitism -– part of a scribe oriented hegemony in society. Indeed, as has already been discussed, not all ideas conceived within notational systems translate well to aural perception, and consequent strategies to achieve transparency of form such as Milton Babbit's infamous advocation of the development of expert listeners ring true with Wishart's analysis (Babbit, 1958, pp. 38-40). Nevertheless, despite its rejection of the priorities of the notational system, acousmatic music still uses rather a lot of spatial concepts in its composition, albeit with much closer attention paid to their aural perceptibility. Certainly one can still find instances, for example in the pitch-space of Denis Smalley, of a reliance on pitch-height metaphors (Smalley, 1986, p. 79).

This is perhaps to be expected as cognitive psychology of music indicates that our physical relationship to the world is key to our understanding of music. Bob Snyder posits that this relationship might be mediated via preverbal preconceptual structures known as image schemas formed in very early childhood. "Image schematic metaphors are based on recurring aspects of relation with the world, and the experiences that generate music come from that same world." (Snyder, 2000, p. 110). One such schema is our gravity-based experience of spatial orientation, or up and down. This is developed in early childhood through activities such as falling down, watching other things fall down, and lifting things. We often refer to this schema in everyday language to make abstract concepts comprehensible. For example when we talk about taxes going up, we are subconsciously using the shared spatial experience of piles of things getting physically higher as the number of items in them gets larger.

Snyder uses the same schema to explain the ubiquitous pitch-height metaphor in music. In this case the key component is gravity, the tension that results from it, say when lifting a toy brick, and the sense of release once it has been dropped. This makes sense if you think about how we refer to being in a state of "heightened tension." Snyder creates the connection by viewing pitch in the context of melody, pointing to the way in which melodies rise to create tension and fall to release it. He doesn't comment further but I would speculate that the final piece in the puzzle comes from the increased vocal tension required to produce higher notes and the subsequent release when dropping back to lower notes.

With this in mind the reasons for many musical conventions and practices, for example why rock guitarists so often pull tense faces and bodily poses when playing extended high passages, become clear. One also comes to a realisation about Western notation. Its

mapping of sounds to symbols isn't arbitrary, but originates from the very same metaphorical relationships that we have for listening and comprehending music. This explains for me how Cardew's *Treatise* (1967), which I always regarded as a failure in terms of an improvisatory notation, is successful as a pictorial representation of music. As Cardew himself states "*Treatise* had been an elaborate attempt at the graphic notation of music; after that time it became simply graphic music (which I can only define as a graphic score that produces in the reader, without any sound, something analogous to the experience of music)" (Cardew, 1971, p. xi). *Treatise* not only abstracts musical notation from its discrete, analytical qualities, but also expands its metaphorical world, tapping into the very same image schema that music relies on. One is left with the strange sense of pre-aural music – a notation of ideas as Cardew might have called it (Cardew, 1971, p. iii). By tapping into the implicit metaphorical worlds that underlie music and language, my research aims in its symbiosis of aural and visual media, to emulate this same quality in a real time context.

Of course it should be pointed out that our contemporary world is already teeming with composed audio-visual relationships many of which pervade our everyday lives on a largely subconscious level. An example can be found in the so-called earcons of the auditory feedback of our computers, mobile phones and tablet devices. Amongst these one finds strategies for relating sounds to events ranging from the imitative clicks that accompany virtual button presses, to the iconic, near consecratory, chord that accompanies the switching on of an apple computer. In computer games, audio-visual relationships are more pervasive, and although the hyperrealism of many AAA games leaves little room anything other than imitative sound, there are still numerous releases in the tradition of early computer games such as *Pac-Man*(1980) and *Super Mario Bros*(1985) which allow for more abstract relationships between sonic and visual events. Of course, these examples at least in part stem from the older field of foley art where the metaphorical relationships between sound and event can often take on a psychological quality. For example, in Warner Brothers' *Roadrunner(1949)* cartoons where the sounds of Second World War dogfights represent the high-speed footfall of *Roadrunner*, or conversely in the space age dogfights of *Star Wars(1977)* where Ben Burtt fashions the sound of a TIE fighter from a drastically altered elephant roar (Carlsson).

In music, the most recent and wholehearted adoption of audio-visual performance is in electronic dance music, where DJs are now routinely accompanied by VJs. Although VJing has antecedents dating back as far as Norman McLaren, Oscar Fischinger and John Whitney,

the discipline only truly congealed in the 1990s through the rave scene via groups such as The Light Surgeons and Hex, and massively proliferated in the 2000s as a result of the increasing processing power of laptop computers and a sudden drop in price of digital projectors after the dot-com crash. Often the relationship between audio and visual is loose, with the steady beat being articulated through devices such as the switching of images or colour flashes, and only the largest macro structural elements such as breaks being observed. Furthermore, in the live improvised context, the continual selection by both DJ and VJ of fixed material from large stockpiles perhaps places an upper limit on the degree of synchronicity between media.

However, increased processing power coupled with graphical programming environments such as vvvv, Jitter, and Isadora, has also seen the growth of live algorithmic generative visuals as opposed to the mixing and applying of effects to pre-rendered footage. Here combinations of audio signal analysis and control data from the audio artist are used to produce a more refined visual response. For example, in Davide Quayola's *PTA* (2007) each individual sound is accompanied by a dedicated synchronised animated visual symbol running down the centre of the screen. The highly sympathetic pairing of animations with sounds achieves an ambiguity as to whether audio is triggering video or vice versa. However, after a short while it becomes apparent that the various animations are fixed units. Their incapacity for supporting sonic development combined with the explicit relationship between audio and visual causes the visual to restrict the aural, resulting in the unsatisfying stasis of both media forms. In his live set for his album *Test Pattern* (2008) Ryoji Ikeda uses a more fluid approach of strobing black and white horizontal lines, reminiscent of barcodes. However, this is at a cost of specificity, with the visuals having a more superficial, decorative function – they provide a generalised rhythmical response to the sound whilst the bulk of the development occurs in the audio.

The audio responsive tessellating patterns of Paul Prudence's *Son Lattice* (2007*)* demonstrate significantly greater scope. Not only does the system of recursively patched renderers – an imitation of analogue video feedback – generate a seemingly endless amount of patterns, but the patterns themselves are manipulable and can fluidly transform into one to another. This crucially allows Prudence enough flexibility to visually articulate musical patterns, manually tweaking parameters during performance to achieve a meaningful response to the audio signal. Nevertheless, most likely as a result of the unrelated origins of

the visual and sonic media, *Son Lattice* leaves an impression of concurrent and inter-related audio-visual streams rather than a single event stream with audio-visual consequences.

Interestingly, I find the current practice to be outstripped in terms of sophistication and fluidity of audio-visual relationships by the work of the field's antecedents such as Oscar Fischinger and Norman McLaren. I suspect that is partially as result of the fixed as opposed to real time nature of the work as well as the analogue means which more easily allow for flexible and even inconsistent approaches to mapping. One example, is Norman McLaren's short film *Dots* (1940). McLaren produces both the visual and sonic material by a process of drawing directly onto the film. The piece exhibits the same one to one quality and sympathy between animation and sound that Quayola achieves but also has a flexible approach to mapping resulting in a rich set of audio-visual relationships. Initially attacks are represented by the sudden appearance of symbols which shrink away after the sound's termination. However, later symbols combine to create new sounds, and other sounds are created via the movement of symbols with exhaust like emission of squiggles and dots. All this is rapidly and humorously developed over the one minute thirty seconds duration of the piece, the visual presenting multiple hypotheses for what might be causing the sound.

Karl Kliem aka Deinststelle's visualisation of Carsten Nicolai aka Alva Noto's *Neue Stadt (Skizze 8)* (2001) – it seems VJs are fond of stage names – manages to achieve a similar effect. Here Noto's glitch audio work, consisting of spliced fragments of sample oscillators and noisy electronic equipment in a rhythmical arrangement, is accompanied by minimal visuals, comprising series of arrangements of horizontal and vertical lines. Small and precise transformations, such as shifts in position and changes in the thickness of line happen in synchronisation with various attacks, highlighting repeating patterns within the musical texture. The selective visual rendering of audio attacks and the adaptation of rules for visualisation with each new arrangement of lines, temper the almost mechanical effect. Despite their economy of means, Deinststelle's visuals even manage to make subtle references, at one point using the console green colouring to indicate a flashing cursor, and at another mimicking the led graphic display of a hi fi equaliser. As with McLaren, the success of the work appears to lie in an approach allowing flexibility whilst maintaining a strong causal link between sonic and visual material.

One, perhaps unlikely, practice that exhibits both these qualities is live coding. The nascent field developed in response to criticisms of laptop performance typified by those of Cascone.

I would argue that the key function of the visual output in live coding is not the illumination of form, but rather the establishment of a causal relationship between the actions of the performer and the sonic output. Comparisons are often made with viewing regular instrumentalists such as guitarists – we may not explicitly understand how the instrument functions but our experience and appreciation of the music is nonetheless enhanced. Indeed in the live context, even when the code is in my preferred language of SuperCollider, I find the task of explicitly relating the code to the sonic output to be beyond my capacities. It may be that my code parsing abilities are still too novice, but the requirement of expert skills for the appreciation of form would put the music in the same category as Milton Babbit from which, with its pop references and humour, live coding feels far removed. In any case the success of networking ensembles such as Benoit and the Mandlebrots, Bile, and Slub, where the presentation of multiple screens in cyclical rotation prevents any possibility of complete code parsing indicates that such an activity is not at the centre of the music. Furthermore, unlike the reading of a score, reading code seems at odds with the temporal flow of the sonic events.

Aware of such issues McLean, Griffiths, Collins and Wiggins offer an alternative for how code might be parsed, proposing a "codeomorphology" where the "elaborate dance of spatial change to code is evident over time" (McLean, Griffiths, Collins, & Wiggins, 2010, p. 2). Nevertheless, my experience in watching live coding performances is that such an effect is minimal in comparison to the sonic development. For example, miniscule alterations to the code such as the replacement of a single parameter can result in significantly more noticeable sonic changes. With the exception of Magnusson's *Ixi Lang*, it seems that the visual presentation of code is determined by the performer's requirements and not the audience's. In such a context, expecting the code to elucidate "abstract thinking gestures" (McLean, Griffiths, Collins, & Wiggins, 2010, p. 1) seems unreasonable, though perhaps one shouldn't discount the possibility that with continual practice, and if code one day becomes a lingua-franca, that a gestural language akin to the bodily and facial expressions of the strutting guitarist might develop in the visual world of the integrated development environment.

The graphical live coding systems of Dave Griffiths, inspired by computer games such as *Core Wars* (1984) and *Carnage Heart* (1995), attempt to frame programming environments in the visual language of computer games in order to appeal to wider audiences. The first of these *Betablocker* involves the visual rendering of a fictional CPU with 256 bytes of memory,

where multiple threads of execution are able to run simultaneously and even modify and delete each other. As one might imagine, though the system is entirely deterministic performances quickly become quite chaotic. The dialectic between unpredictability and control is seemingly a significant motivation Griffiths' interface design. However, as the visual rendering concerns itself with the underlying process and not the sonification, the relation of audio to video remains something of a mystery to the audience who barring a fair amount of computer knowledge will be most unlikely to discern what is going on, although they may nevertheless enjoy the spectacle.

Al Jazari is intentionally more sympathetic to lay audiences and features robotic agents triggering sounds by moving around a 3D grid. The user programs the robots' movements using a simple language into sequences which hover in thought bubbles above their heads. Despite the simplicity of the language and interface, complexity is facilitated by the robots' capacity to follow, avoid and message each other. This seems to achieve the perfect balance between transparency and complexity for the audience making the premise explicit whilst allowing detail to emerge as a consequence of the systems rules. Nevertheless, as in *Beta Blocker* both sound and vision in *Al Jazari* serve to elucidate the conceptual space of the code rather than develop meaningful relations between themselves. One consequence of this is that the visual space is as fixed as the system it represents and though in *Al Jazari* the movement of the robots is able to sustain interest for quite some time, eventually one becomes frustrated by the unchanging nature of the space.

An alternative approach with similar aims to live coding is the development of new digital musical instruments for live performance. Rather than conveying underlying processes, the aim here is to make electronic instruments sensitive to the nuanced movements of performers, allowing them to become gesturally expressive in a similar fashion to traditional instrumental performance. The field dates back at least as far as Max Mathews' 1970 Radio Baton. Since then many such instruments have been produced through institutions such as Steim and IRCAM, as well as through commercial and amateur development. One contemporary version is Marco Donnarumma's *Xth Sense*, a biophysical interactive system that captures "sonic matter created by the performer's limbs" and simultaneously uses it as both control data and sonic material (Donnarumma, 2012). Watching Donnarruma in performance, the movement certainly has an expressive quality, although whether it serves to elucidate the form in the same way that traditional instrumental gesture does is questionable. One problem is that given the wide timbral variety of the music, the gestural

world seems less defined with similar movements triggering various types of sound at different points within the piece. There is a resulting lack of development in the gestural language. Furthermore, despite the sound's derivation from the movement the link between the two is often unconvincing. Technology is a poor surrogate for the performance traditions that developed the gestural languages of instrumental genres. It is perhaps this that composer Johannes Kreidler is commenting on in his *Kinect Studies* (2011), which comprises arbitrary uses of the advanced sensor technology, including realisations of Fluxus, La Monte Young and even Abramovic pieces. Indeed, some live coders see the use of sensors in computer music as a superimposition of instrumental performance values onto the world of computer music. For example, Matthew Yee King claims live coding as the "first performance art form to come out of computer music that's truly computer orientated" (McCallum & Smith, 2011)

Tangible interfaces such as Sergi Jorda's *Reactable* indicate a third approach which appears to combine the best of both worlds, allowing for the conceptual manipulation of computer processes through a graphical interface and fine parametric manipulation through mechanical control by the performer. One problem with the *Reactable,* which became apparent to me when watching a performance by Carles Lopez at Beam Festival in 2012 is that its table top design means that, without using a overhead camera and projection, very little of the interface is visible to the audience. This is exacerbated by the round form which eventually leads the performer to turn their back to them, obscuring the view of the interface altogether.

When performing with his Monome the artist Daedalus uses the simple solution of mounting the interface angled away from him and towards the audience. Interestingly his highly flamboyant performance style of sudden dips and twisting hand gestures as he hits the Monome's buttons has no functional relation to the production of sound yet clearly enhances the articulation of musical form. The increasing accuracy of sensor technology combined with continual advances in digital projection bodes well for more performance orientated interfaces. The *V Motion Project* by Assembly, gives some hint of what this future may look like. A Kinect based system is used to respond to choreographed movements with visual feedback via a projected interface with VJ accompaniment. Though the interface clearly sacrifices flexibility in order to achieve a commercial Dubstep performance, it's striking how well the choreographed gestures and projection serve to amplify the effect of

the musical form. Both examples demonstrate a synergy between physical and sonic gesture in electronic dance music which simply isn't present in the electronic avant-garde.

It seems that this problem cannot be solved through the development of new technology alone. Rather new gestural languages have to be developed that suit and articulate electronic music's sonic world. Such a task might seem daunting if not insurmountable to non-dancing, experimental musicians such as myself. After all, the physical gesturing of musicians appears to operate on a subconscious level, most likely absorbed through imitating other performers. Jonathan Burrows and Matteo Fargion's *Both Sitting Duet* (2002) perhaps give some indication of how a new gestural language might be synthesised. This forty minute, silent composition consists entirely of hand and arm gestures performed from seated positions. The piece is categorised as dance, yet something about the rigorous organisation of material, rhythmic phrasing, and juxtaposition of tempi, give one the sense that they are viewing music without sounds. Indeed, the temporal and motivic organisation of the piece is derived from the late Feldman violin piano duet, *For John Cage* (1982). The gestural language draws from the everyday including actions such as reaching down to touch the floor, brushing one's trousers, clapping, counting, and making a circle by touching the tip one finger against the tip of the thumb. Crucial to the work's humour and success is the relationship between the gestures and the two white middle-aged male individuals who perform them. Ramsay Burt comments "Somehow their age and gender, together with the easy intimacy that was established through their interdependency was endearing" (Burt, 2006).

Putting gesture at the centre of musical practice, such as the design of movement based interfaces does, requires the kind of detailed consideration of physical gestures and their context dependent semantics exhibited by Burrows and Fargion. The failure of sensor based electronic music to do so, leaves me with a sneaking suspicion that, despite its inclusion of visual elements, the old acousmatic attitude that the visual is more superficial than the aural still pervades the genre. At this point it would be wise to clarify my own intentions as regards to the physicality of performance. Whilst this research will concern itself with the projection of the presence of the performer through visual and sometimes physical means, such as the movement of the mouse pointer on a screen, a tangible interface, or indeed physical gesture, the development of a gestural language for electronic or contemporary music remains outside the scope of this research. Indeed in some projects, particularly those involving solo-laptop performance, I would argue that the physicality of performance is

reflected more convincingly through the movements of the mouse pointer on the screen in combination with the presence of a nearby performer at laptop than by any artificially-superimposed schema of physical gestures.

Where physical gesture has been employed it has been in the service of my final and most crucial desideratum for the creation of audio-visual music, namely that the relationship between the two media must not only be equal, but co-dependent. In other words, as opposed to one medium serving as a decorative or elucidatory device for the other, it should be the audio-visual relationships themselves that constitute the argument of the work. It is conspicuous at this point that although the relationship between sound and vision has already been discussed at some length, aesthetic approaches as to the precise nature of the sounds that will be employed have not yet been mentioned. Once again the governing factor in the selection and deployment of sounds in this research is the development of co-dependent relationships with visual counterparts. This does not by any means that the function of sound is arbitrary. The intention is to foster a great deal of empathy between related sonic and visual events. This will most likely be achieved by an iterative process of modifying both visual and sonic elements in negotiation with each other with the aim of creating audio-visual gestalts.

When operating as the sole artist, the endeavour described in the preceding paragraphs will necessitate straying into foreign disciplines such as computer graphics, computer vision, visual arts, and choreography, where my experience and knowledge will undoubtedly be inferior to practitioners in those fields. Such disadvantages are perhaps balanced out by the fact that interdisciplinarity itself is the subject of the work, thereby allowing me to assemble a personal practice based on the intersections of between the various disciplines. If the goal is met then neither medium will function as an independent layer – discussions as to the primacy of the ear will become as meaningless as those over chicken and eggs. Instead audio and video will be locked into a seemingly inevitable relationship as manifest as it is equal.

1. Tecken 6.99

Video and code: http://www.simonkatan.co.uk/phd/tecken699.html

Overview:

Tecken, the Swedish word for symbol, is also an IKEA tea pot retailing at £6.99. According to Wikipedia, the store employs an elaborate system for the naming of its house hold goods; Bookcases are occupations, bed covers are plants and precious stones, carpets are Danish place names and so on. The word seemed an apt title for a piece in which sugar cubes are used to signify sounds which mysteriously morph in relation to the position and movement of the cubes according to a hidden system.

Tecken 6.99 (27.10.2009) came about as a result of initial research into using basic computer vision techniques to create tangible interfaces for live synthesis. I was particularly struck by how readily our perception creates causal relations between the objects we manipulate and the sounds we hear. It seemed to me that such interfaces offered not only the possibility of creating instruments accessible to all, irrespective of prior musical knowledge, but also of realising new hybrid mediums consisting of objects, actions, and the sounds that symbolise them. Inspired by the title of the show for which piece had been commissioned 'The Dissolving Cube,' I chose a large number of sugar cubes as my tangible interface. These not only provided a modular system of identical elements that could be combined in numerous ways, but the organic quality of the cubes added an extraordinary quality to the interface encouraging some rather wild speculation about how the installation might work.

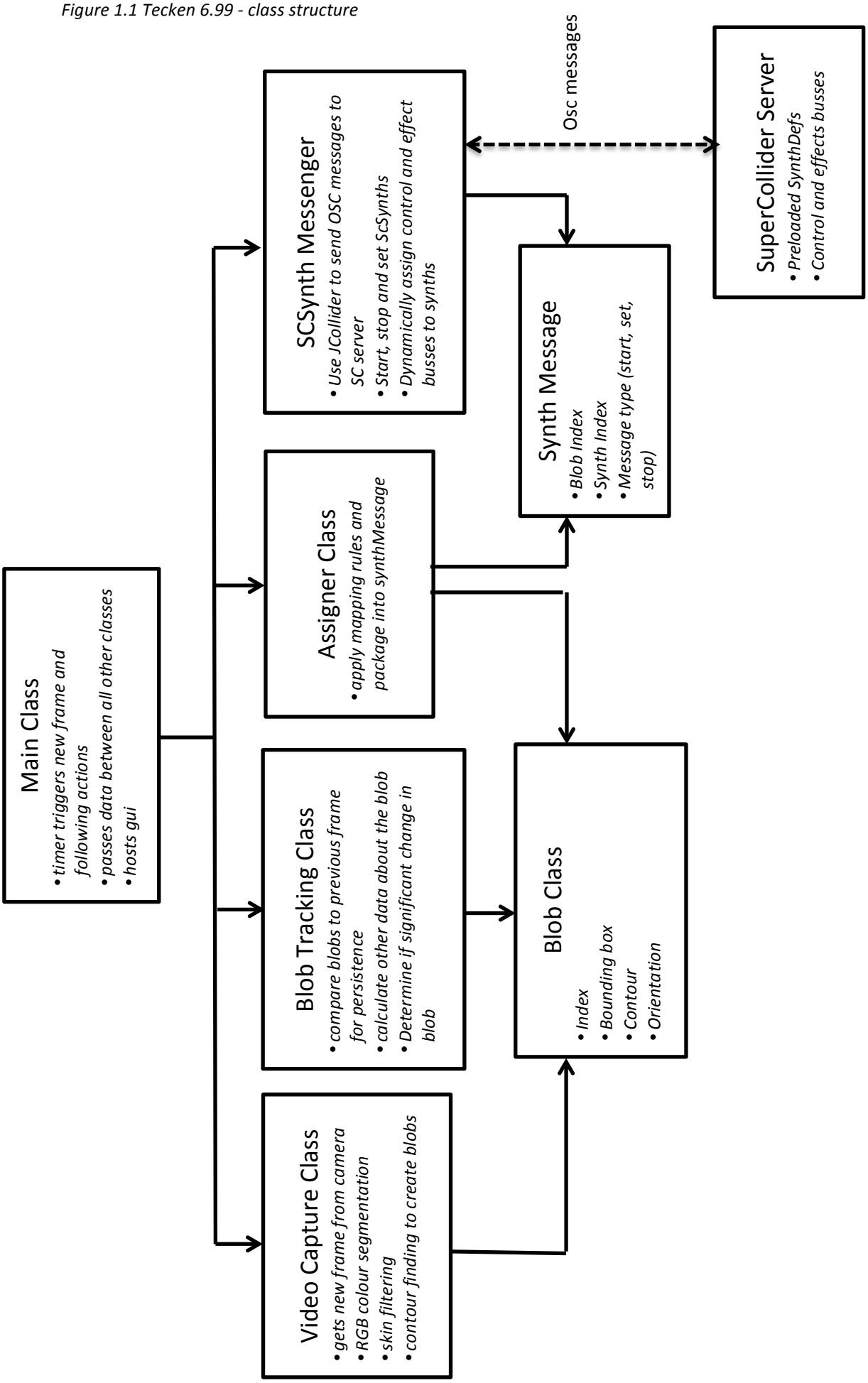
Implementation:

A primary concern was to address the question of how to make the interaction interesting. More specifically I wanted to avoid the double bind that arises with symbolic rule-based interfaces whereby the mapping of input to output is either so complex as to be completely incomprehensible to the user or is so literal as to become banal. Contrary to using a single mapping system, simple or complex, my solution was to use a gradated series of mapping systems requiring increasing amounts of effort to access as they become more complex and

sonically rewarding. The aim was that this simple game-like mechanic would create a semi-structured linear experience for the user whereby the sonic possibilities of the installation would gradually open up before them.

The tracking was done via a standard webcam using a bespoke tracking program built in Java, hosting a Processing applet. This program in turn used Hanns Holger's JCollider library to send and receive OSC messages directly to and from SuperCollider's server on which I loaded the SynthDefs for the sound output (Figure 1.1).

Figure 1.1 Tecken 6.99 - class structure



The system responded to different configurations of sugar cubes according to a predefined rule-set which is described in Figure 1.2 - Figure 1.5.

Figure 1.2 Tecken 6.99 - general system behaviours

Behaviour	Description
1	The sound output remains stable whist the user is adjusting the cubes.
2	Once the user has finished there will be a sonic response.
3	There are three possible cube formations - isolated cubes, small grouping, and large groupings.
4	Each cube formation is responsible for its own sonic output whose parameters are determined by various spatial mappings
5	There is no co-dependency between cube formations.

Figure 1.3 Tecken 6.99 - isolated cube behaviours

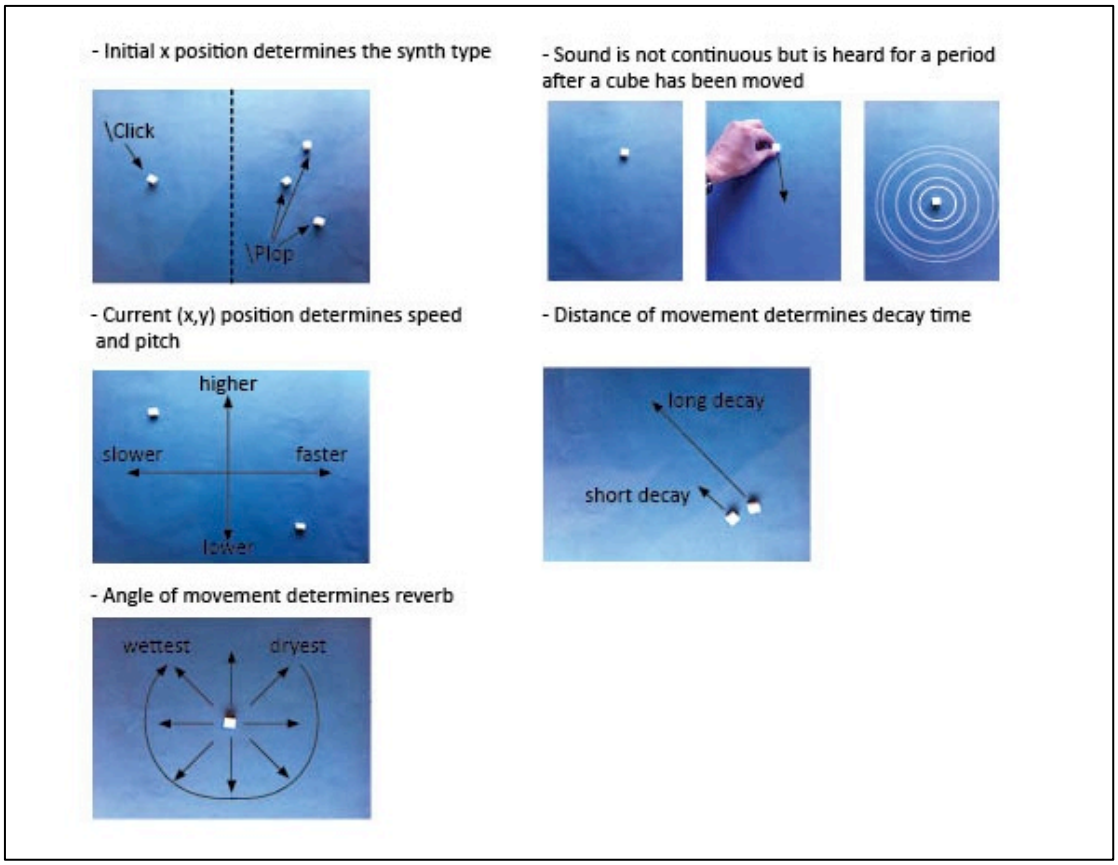


Figure 1.4 Tecken 6.99 - small cube group behaviours

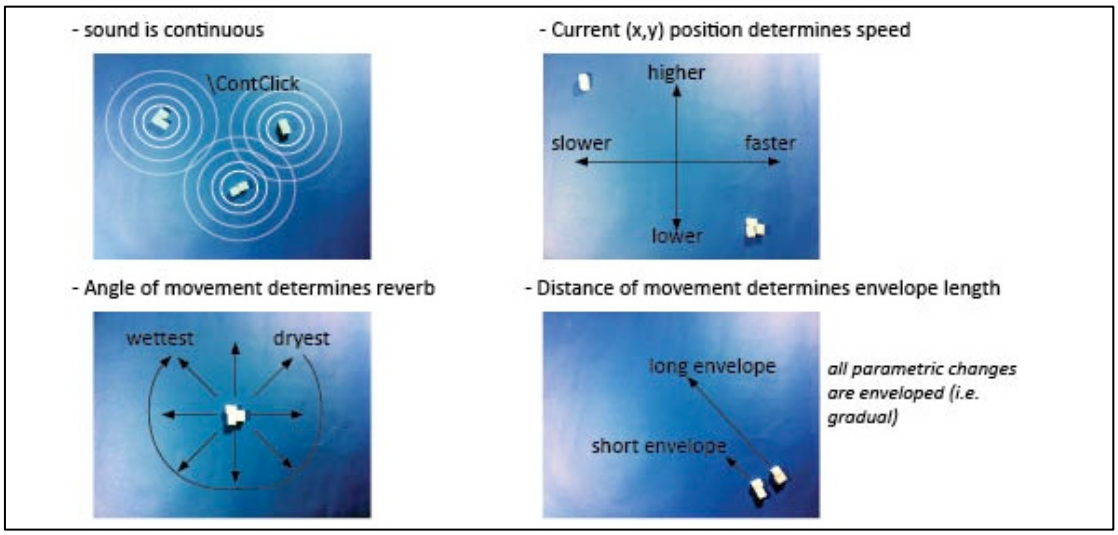
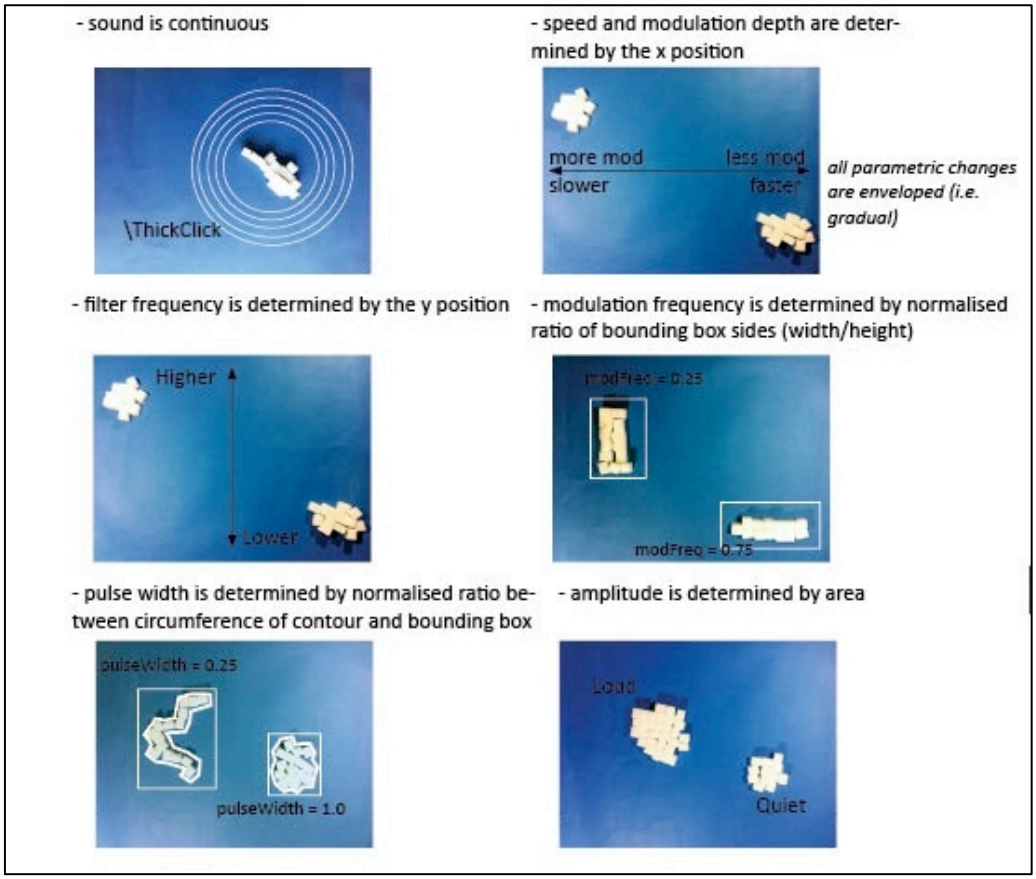


Figure 1.5 Tecken 6.99 - large cube group behaviours



Conclusions:

The project ran in the Portman Gallery for a one week period. At a glance, the audience at the opening comprised invited artists and critics as well as students and teachers from the attached school. The Gallery space was typically noisy but the installation was housed in an

enclosed booth increasing the audibility of sounds and allowing for more contemplative interaction. Although the project ran smoothly and was well received, there were a number of technical and conceptual design flaws from which I was able to draw lessons for future work. The tracking program performed effectively and robustly within the context it was designed for, but as a first attempt at computer vision, it was somewhat naive and its structure unnecessarily complex. Much of this complexity resulted from using Processing, which is primarily a sketching tool, to build a relatively large application, but also from implementing my own esoteric computer vision from the ground up rather than using one of the established libraries such as OpenCV. A further inefficiency came from messaging the server directly from the tracking program, which dynamically managed synths, control busses and audio busses. A much cleaner approach would have been to message SCLang with lower level data about the blobs, thus using its more flexible and appropriate tools to manage the synths.

Some observation of users' reactions to the installation also revealed a number of issues with the interaction design. In testing the project with various groups and individuals, I'd had strongly positive responses with participants spending much time enjoying and exploring the system in the way that I had imagined. However, I found the engagement in the gallery to be more superficial and in some cases, for example where participants used the cubes to make patterns and figurative shapes, it was clearly demonstrated that my underlying structure was not coming across.

I've identified two key differences between the tests and my final presentation in the gallery that could explain the difference in outcomes. Firstly, each test started with just a few sugar cubes in front of the camera with me gradually providing more cubes, whereas in the gallery the participants approached the installation in whatever state the previous user had left it. This disrupted the intended semi-structured linear experience as the users in the gallery mostly encountered the installation in its final state first. Secondly, during the tests, the output from the tracking program was visible to the participants whereas in the gallery it was concealed. This lack of visual feedback led to a degree of scepticism as to whether the installation was really responding to the cubes position and also a lack of clarity about the nature of the tracking with some users trying to pile the cubes vertically to change the sound and some not understanding that the installation would only respond after they had adjusted the cubes.

Whilst providing visual feedback certainly would have aided more intuitive understanding of the work, the action followed by response aspect of the design added an unwanted conceptual barrier for the user to overcome. In general, I felt that the approach had been too static and notational to really utilise what computer vision has to offer as a sound controller. Nonetheless, the project not only confirmed that computer vision could be used to build accessible and intriguing tangible interfaces, but that given the correct presentation, mapping rules themselves could be used to form the argument of a work.

2. Les Escaliers Méchaniques

Video, code and score: http://www.simonkatan.co.uk/phd/escaliers.html

Overview:

Les Escaliers Mécaniques (26.01.2010) is a piece for twelve performers and two escalators – one up, one down. The performers simply travel up and down the escalators in various combinations, but a camera tracks their position, direction, and speed, which in turn triggers computer synthesised sounds. There are five possible modes of movement, stand on the up escalator, walk up the up escalator, stand on the down escalator, walk down the down escalator, and walk in the wrong direction on either escalator, i.e. stationary. Each performer controls a distinct set of sounds responding in a particular way to their movements. The result is a musical work where a world of sonic visual relations is explored through site-specific choreography.

A commission from Borealis Festival for a site-specific piece at Kings Place in London, somehow involving its long and visually striking escalators, provided me with a second opportunity to explore computer vision interfaces. A further stipulation of the commission was to use a number of composition students from Trinity College of Music in some guise and so I opted to write a performance as described in the program notes above. As with the sugar cubes in *Tecken 6.99*, the escalators provided a somewhat fantastical and unlikely interface, though in this case, I aimed to use computer vision to generate real time responses to movement as opposed to subsequent reactions to static scenes. Far from being a blank canvas, the escalators were rich with implications for the speed and pitch of the sonic material and also imposed structural complexities through their physical properties that would inform the composition of the piece.

Implementation:

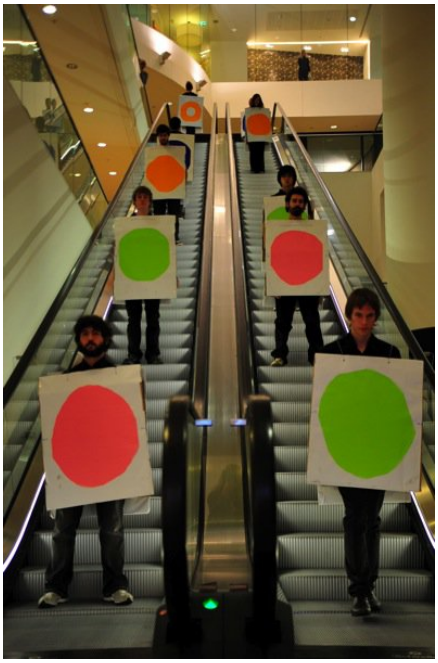
Following from my previous work, much of the argument of the piece was formed by the mappings themselves, and my concern was with creating a symbolic rule-based system that trod the line between literality and incomprehensibility. My strategy was to use multiple

simple mappings that could be easily and intuitively understood. For example, for one performer a pulse is adjusted according to their speed on the escalator, whilst for a different performer the pitch of the sound is controlled by their vertical position on the escalator. In the name of comprehensibility I ruled out the options of multiple performers controlling a single synth or a single performer controlling multiple synths. Furthermore, the sound of each synth should be perceivable as a single audio stream, which was largely a case of making sure that all sonic output of an individual synth was mapped to the control data in the same way. One could draw an analogy with a chamber ensemble where there is an emphasis towards the audibility of individual instrumentalists within the texture. In the composition, which is mostly deterministic, the mappings are carefully introduced with the intention of encouraging the listener into a highly rational interpretation of the piece that, via the sheer number of mappings and their combination into complex textures, becomes increasingly challenged as the piece progresses.

To aid the computer tracking, the performers wore sandwich boards with large coloured disks on them. Each sandwich board was one of four colours and had a solid disk on the front and a hollow disk on the back. This allowed for sounds to not only be triggered by speed and vertical position but also by the direction in which the performer was facing (Figure 2.1). A camera was placed on the first floor balcony facing down onto the escalator and the audience mostly stood on the ground floor at the foot of the escalators where there was stereo sound. In response to my experiences with *Tecken 6.99*, I also provided a digital video feedback, which clearly demonstrated how the computer vision was working for the audience. This was projected onto the wall running down the side of the escalator allowing simultaneous views of the performance and the digital output.

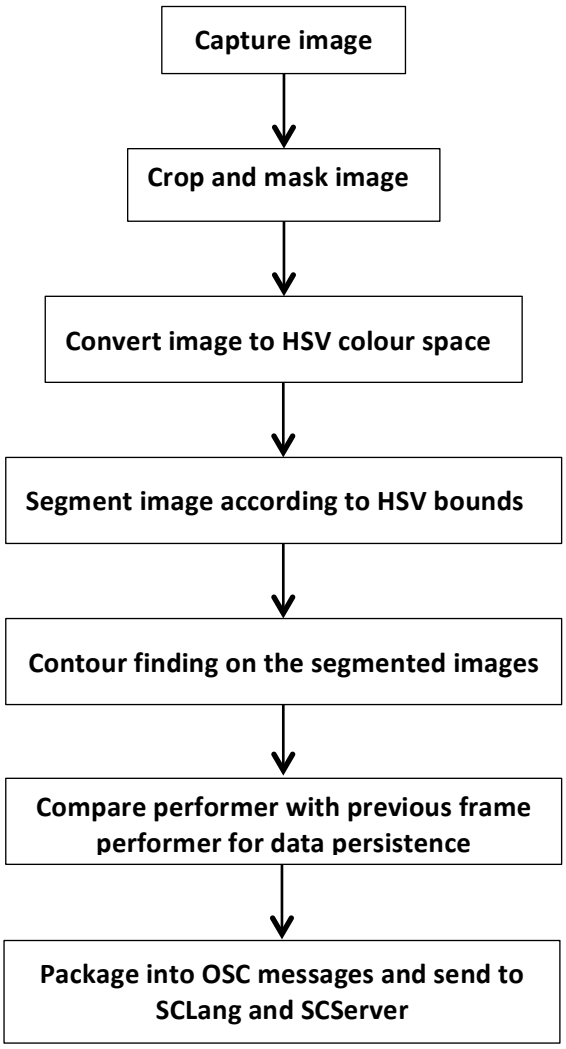
Figure 2.1 Les Escaliers - performers on the escalators

Figure 2.2 Les Escaliers - performers on the escalators



To improve on the technical implementation from the previous project, I created a new tracking application using Open Frameworks, a C++ framework wrapping many libraries including OpenGL and OpenCV. I was able to take advantage of the better memory management and speed of C++ as well as the highly optimised functions of OpenCV functions to obtain large improvements in terms of frame rate, resolution and robustness. Furthermore I simplified OSC messaging by using a fixed array of performer indexes in the tracking program that directly corresponded to an array of control busses in SuperCollider. The faster frame rate meant that I was now able to usefully use the OSC data on a frame-by-frame basis, using the speed and direction of the movement over periods of multiple frames to capture the quality of movement (i.e. smooth, jerky, bouncy). A summary of the processes and class structure are shown below (Figure 2.3, Figure 2.4).

Figure 2.3 Les Escaliers - Computer vision processes



- 640x480 pixels
- 25 fps
- quality webcam with drivers to disable auto-gain and adjust shutter speed

- Only the tracked escalator area remains
- Reduces CPU load
- Improves reliability.

- Behaves more linearly than RGB allows
- Brightness (V) can be disregarded improving tracking where light conditions are variable

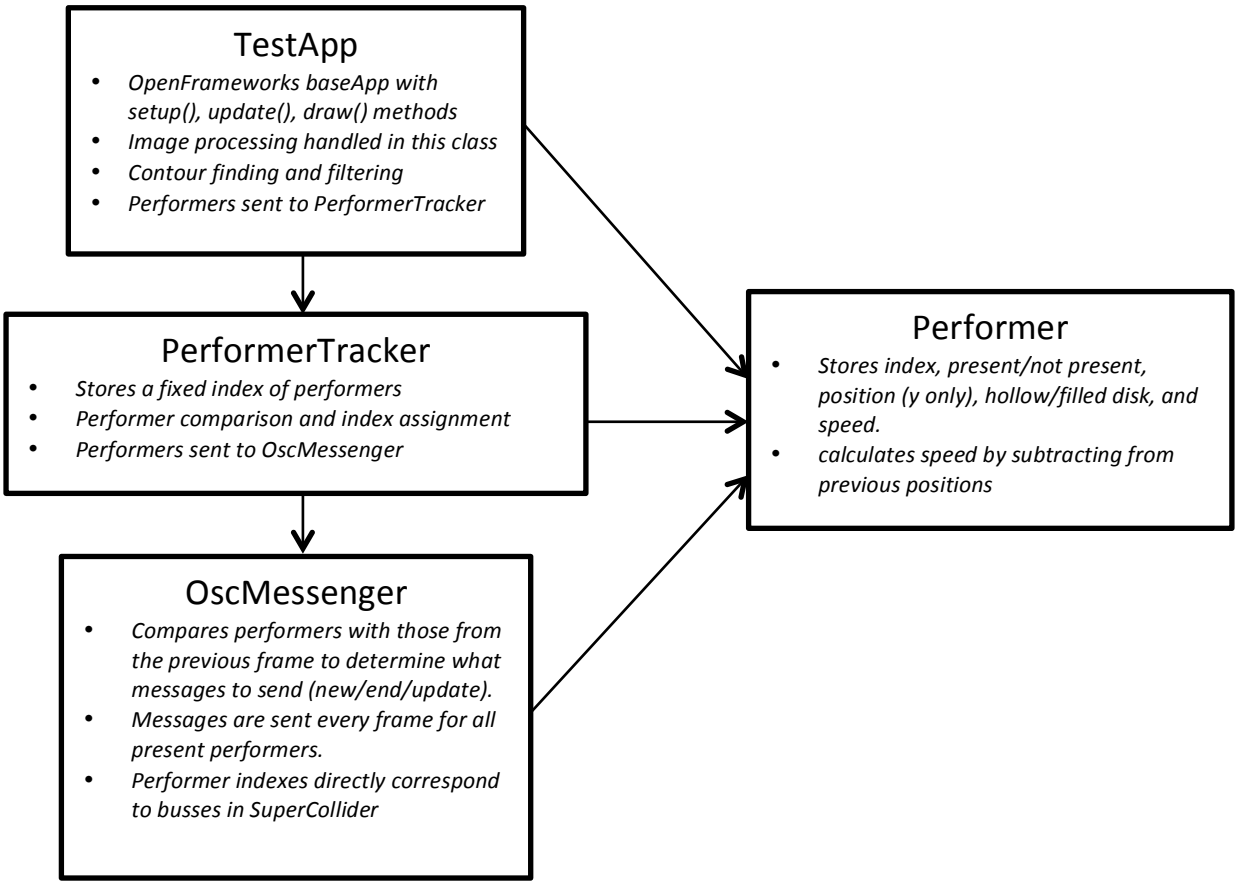
- Involves iterating through every pixel and colouring white or black according to whether it is within the predefined colour bounds.
- Process is repeated for each different colour that is being tracked

- Use ofxOpenCV contourfinder for fast performance
- Rogue contours are filtered according to size criterion
- A performer instance is created for each contour

- Uses distances between performers in the present and previous frames

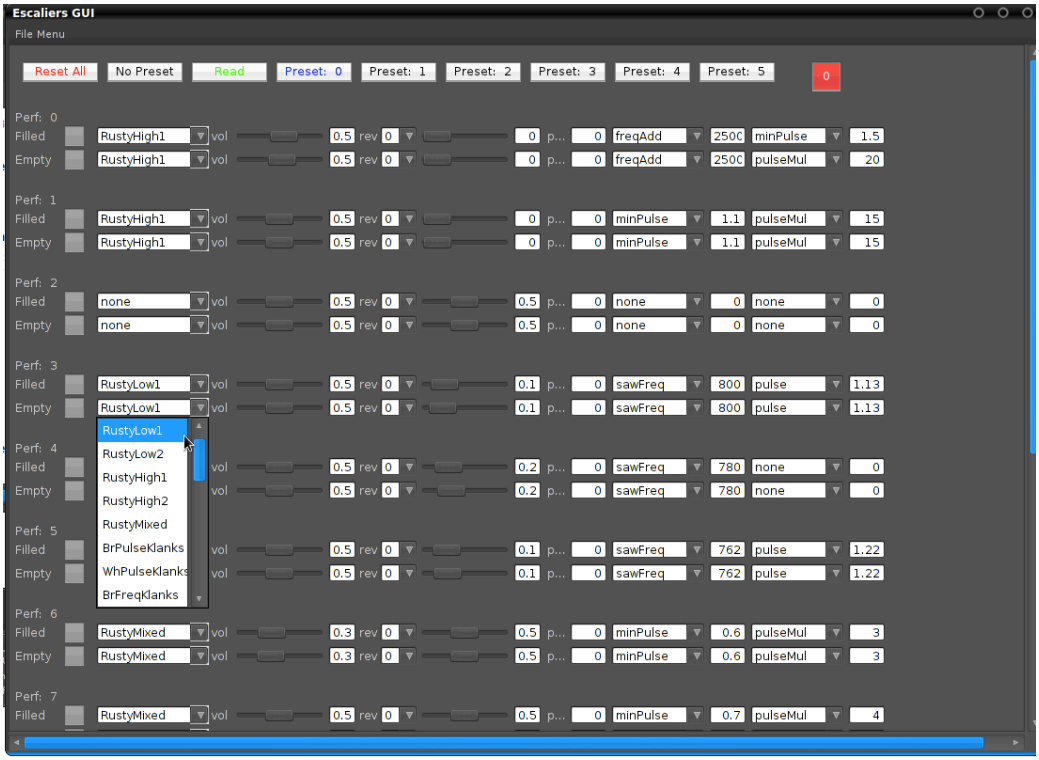
- Performer indexes are fixed and control busses created in SCLang
- Create and destroy synth messages handled by SCLang
- Control Busses updated by SCServer

Figure 2.4 Les Escaliers - class structure



It was desirable that the sounds triggered by individual performers would change during the course of the piece, and this required the creation of a user interface to switch quickly between presets in SuperCollider. I opted for a system of banked presets, similar to digital multi-effects units for electric guitars, which could hold up to 25 saveable presets with independent variable setting for the 12 performers on each preset. For each performer in the preset a synthDef for the filled symbol and a synthDef for the hollow symbol could be selected. For each of these synthDefs, volume, reverb, pan, and two further variables could be independently set (Figure 2.5). This system allowed me to compose complex changes of sounds that could be quickly and easily executed from my computer during the performance.

Figure 2.5 Les Escaliers - SuperCollider GUI



The initial sounds of the composition were inspired by the mechanical sounds of very old escalators such as one finds in certain London underground stations. However, as the piece progresses the sonic material becomes more abstract. In total the piece used twenty-eight synthDefs with a variety of mapping strategies (

Figure 2.6). In most cases panning is mapped to performers' height on the escalator as this also correlates with their horizontal position in the performance space. Where height is used to control other parameters, it is mostly positively correlated with parameters that have magnitudinal qualities such as frequency, pulse frequency and amplitude as is the natural inclination of a musician, though in some cases the relationship is inverted. Where performers' speed is used, it is often used as a trigger, allowing performers to start and stop their sound, switch between sounds, or trigger enveloped parametric changes by adjusting their walking pace. There are a number of synths that output short sonic events as opposed to continuous sound requiring the performers to turn backwards and forwards in order to reset the synth and trigger more sounds.

Figure 2.6 Les Escaliers – synthDef parameter mapping

SynthDef	Features	Height =>	Speed =>	Adjustable Parameters
RustyLow1	intermittent pulse	pan (+)	stop/start (+)	frequency pulseFrequency
RustyLow2	intermittent pulse	pan (+)	stop/start (+)	frequency pulseFrequency filterFrequency
RustyHigh1	continuous pulse	pan (+)	pulseFrequency (+) detune (-)	baseFrequency pulseFreqMul minPulseFrequency legato
RustyHigh2	continuous pulse	pan (+) amp (+)	frequency (+)	pulseFrequency legato
RustyMixed	continuous pulse	pan (+)	frequency (+) pulseFrequency (+)	legato pulseFreqMul minPulseFrequency
WhPulseKlanks	continuous paired pulses	pan(+) pulseFrequency(+)		baseFrequency secondaryPulseAmp
BrPulseKlanks	continuous paired pulses	pan(+) pulseFrequency(+)		baseFrequency secondaryPulseAmp
WhFreqKlanks	continuous paired pulses	pan(+) baseFrequency(+)		secondaryPulseAmp
BrFreqKlanks	continuous paired pulses	pan(+) baseFrequency(+)		secondaryPulseAmp
HighSteam	continuous	pan(+) filterFrequency(+)		
LowSteam	continuous	pan(+) filterFrequency(-)		
SoftSteam	continuous	pan(+) filterFrequency(+)		
SpaceShip1	switch between two continuous pulsed sounds	pan(+) soundA freq(+)	soundA/soundB (+)	soundA amp soundB amp
SpaceShip2	switch between two continuous pulsed sounds	pan(+) soundA freq(+)	soundA/soundB (+)	soundA amp soundB amp
SpaceShip3	switch between two continuous pulsed sounds	pan(+) soundA freq(+)	soundA/soundB (+)	soundA amp soundB amp
Sin1	continuous	pan(+) frequency(+)		
Conveyor1	continuous	signal clip (+)		
Conveyor2	continuous	signal smoothing (+)		
WhiteSweep	continuous	amp(+) filterFrequency(+)		
WhiteHit	short sound	pan(+) filterFrequency(+)		
BrownHit	short sound	pan(+) filterFrequency(+)		
WhKlankHit	short sound	pan(+)		
BrKlankHit	short sound	pan(+)		
KlanknWhistle	short sound	pan(+)		
Boing	short sound	pan(+)		
ClockWork	continuous pulsed sound, intermittent accel	pan(+)	trigger accel	frequency
WhKlankWork	continuous pulsed sound intermittent accel	pan(+)	trigger accel	baseFrequency
WhKlankInt	continuous pulsed sound intermittent cresc	pan(+) pulseFrequency(+)	trigger cresc	baseFrequency

NB. (-) means inverted mapping, (+) means normal polarity

I initially scored out the choreography according to my musical requirements. However, the limitations of using non-dancers and the physical constraints of working on the escalators meant that significant modifications had to be made to the piece in terms of movement and sound. Such constraints included the performers not being able to easily pass each other on the same escalator, the logistics of who is at the top and who is at the bottom, and the escalators being too long to be completely covered by the camera shot which meant that at the very top and bottom of the escalator performers did not trigger sound. This final issue lead to problems in maintaining continuity and density of sound for certain sections of the piece. The final choreography consisted of simple, easy to remember routines, often performed in small groups by the performers as they descended or ascended the escalators (view score online). These routines were carefully cued and synchronised with changes of sounds, which I controlled from the first floor balcony facing the escalators.

Conclusions:

As has already been described, the performance took place on the escalators in the lobby of the Kings Place Concert Halls. The audience not only comprised the Borealis concert goers, who were most likely well versed in contemporary music, but also general public who had been attending a Martin Amis talk and book signing. Both groups were talking and buying interval drinks as the performance started. However, they became more focussed once the music got underway.

The piece received a positive reception with the most enthusiastic referring directly to the mind-boggling aspect of the numerous audio-visual relationships. In this regard my strategy of inter-combining multiple simple and intuitively understandable mappings to create a more complex whole appeared to pay off. Although any individual relation, when focussed upon in isolation, was easy enough to comprehend, the continual changing of relationships and their build up into dense textures was able to create a series of on going perceptual challenges for the audience, which prevented the material from becoming banal. Furthermore, the purposefully explicit opening of the piece, which featured solo entries, actively encouraged the audience into a mode of cognition where they would attempt to decode the mappings at work. The visual feedback also fulfilled its function of assuring the audience that it was the performers' movements who were triggering the sounds and not vice versa. A notable exception was my parents who chose to ignore the screen throughout

the performance, their subsequent comments making clear that they had not apprehended the relationship between performers and sound.

However, despite its enjoyable novelty aspect, the site-specific aspect of the commission had lead to a form which was slightly unsatisfying. The sheer length of the escalators, the inability of performers to pass each other on them as a result of the large sandwich boards, and the slow movement required by my still rudimentary computer vision implementation, all combined to enforce numerous pauses in which the performers could regroup at the top and bottom landings after each passage. Whilst this effect was tolerable at the beginning of the piece it prevented me from achieving the kind of climax that the work called for. Two possible solutions would be to have more performers or shorter escalators. A third would be to implement a more robust computer vision tracking, perhaps using a mean shift algorithm, which would remove the necessity of such large markers and hence allow the performers to comfortably pass each other on the escalators. This would not only improve the continuity but also allow more rapid movement by the performers.

3. Random Walk

Video and code: http://www.simonkatan.co.uk/phd/randomwalk.html

Overview:

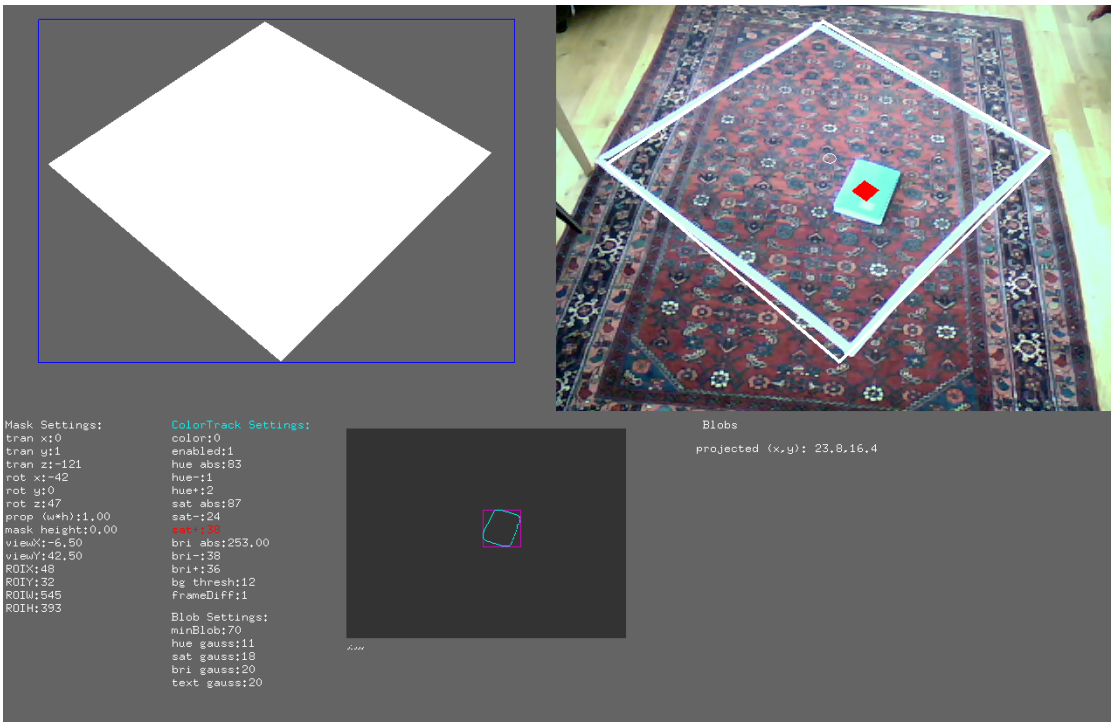
A random walk is a mathematical model of a journey consisting of successive random steps. The results of random walk analysis have been applied to computer science, physics, ecology and economics. For example, the search path of a foraging animal, the price of a fluctuating stock and the financial status of a gambler can all be modelled as random walks. This game-installation is an invisible maze. A single player, whose position is tracked by a computer, has to respond to morphing sounds and images to discover as many hidden targets as possible. The closer you are to a target, the clearer and louder the sound becomes. The sounds are all sampled from Les Percussions de Strasbourg's recording of Xenakis' *Pleiades*(1987) and treated with granular synthesis.

Random Walk (19.06.2010) was developed in response to a commission from Spitalfields Festival for Xenakis Unbound in Spitalfields Market. The brief was for a sonically based game-installation that would appeal to children and adults alike and utilised samples from a chosen recording of Xenakis' *Pleiades.* I decided to construct the installation as an invisible sonic maze with the aim of using a game mechanic to encourage players to engage with the sonic-spatial relationships of the work. Players would be rewarded for correctly decoding a relationship with a new set of sounds and relationships to be decoded. This would happen as they tried to find a series of hidden spots within a marked area using only sounds, which would change as they approached spots to guide them. For each spot there would be different sounds, changing in different ways; for example speeding up, slowing down, becoming clearer, getting higher, becoming more staccato, or getting louder. In trying to solve the maze, players would unwittingly find themselves on a random walk creating a strange dance-like spectacle for onlookers. Given the almost granular approach to development that occurs in some sections of *Pleiades* and Xenakis' pioneering use of the technique, granular synthesis seemed an ideal way to approach the treatment of samples from the piece.

Implementation:

The concept required tracking a user's position on a 2D plane, and having investigated various other technologies, I settled once again on computer vision as the most practical and affordable method. As with the previous work, the maze was powered by an application, which was custom built in OpenFrameworks. However, for this project I implemented a number of technical improvements allowing for robust tracking in a more challenging environment. Players were tracked by brightly coloured markers, which they wore to play the game. These were tracked through a combination of colour segmentation, as used in *Les Escaliers Méchniques*, but also frame differencing which greatly increased the robustness of the tracking, especially given the changing light conditions in which the installation had to operate. On top of this, erosion, dilation, and gaussian blurring all reduced noise thus increasing the reliability of the contour finding (Figure 3.1).

Figure 3.1 Random Walk - colour segmentation, contour finding and tracking



A further development was the use of OpenGL's depth buffer to obtain 2D coordinates for the player's position in relation to the floor plane given their screen coordinates. This worked by manually positioning a virtual frame to match a marked out square on the floor in relation to the view from the camera. By taking the screen coordinates of the tracked objects and using OpenGL to project them onto the virtual frame, reasonably accurate 2D

coordinates for the player's floor position can be achieved without the complication and CPU expense of pre-distorting the camera image (Figure 3.2, Figure 3.3).

Figure 3.2 Random Walk - video capture before frame calibration

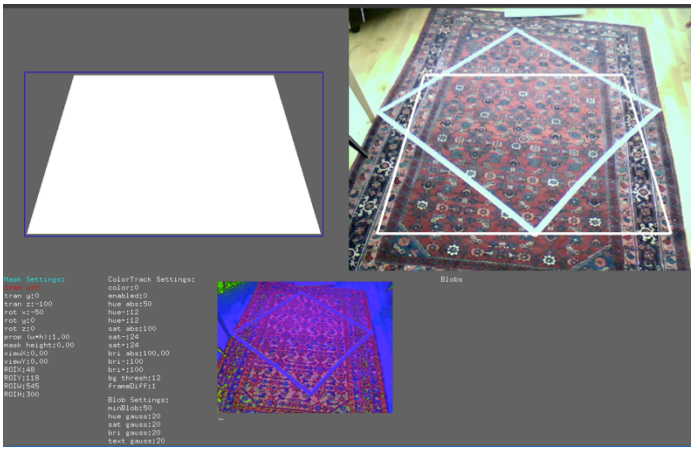
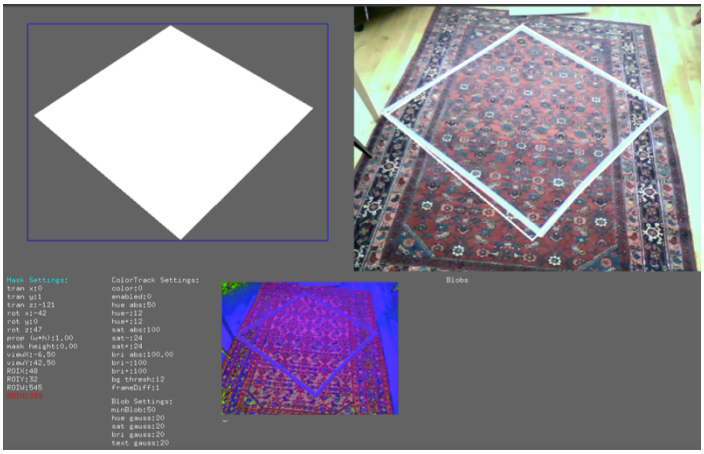
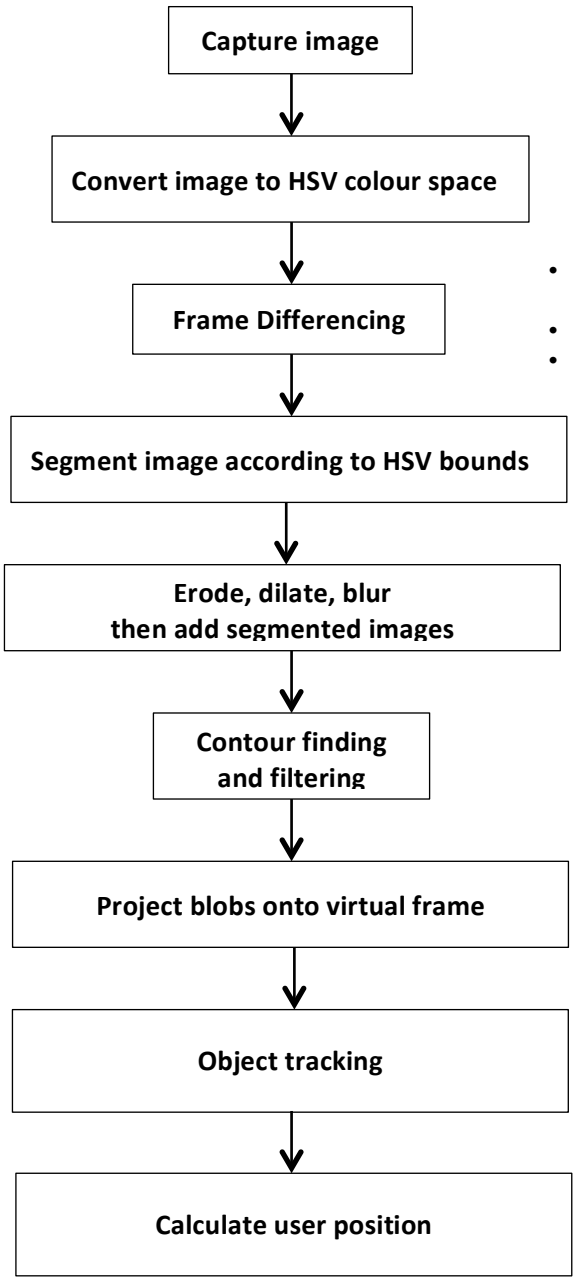


Figure 3.3 Random Walk - video capture after frame calibration



A flow diagram of the computer vision process, and class tree of the entire tracking program can be found below (Figure 3.4, Figure 3.5).

Figure 3.4 Random Walk - computer vision process



- 640x480 pixels
- 25 fps
- quality webcam with drivers to disable auto-gain and adjust shutter speed

- Take the absolute difference between the current frame and the previous frame (usually on H channel)
- Threshold to produce a binary mask
- Has the result of excluding all non moving objects in the scene

- Has the combined the effect of reducing noise in the binary image to be tracked.

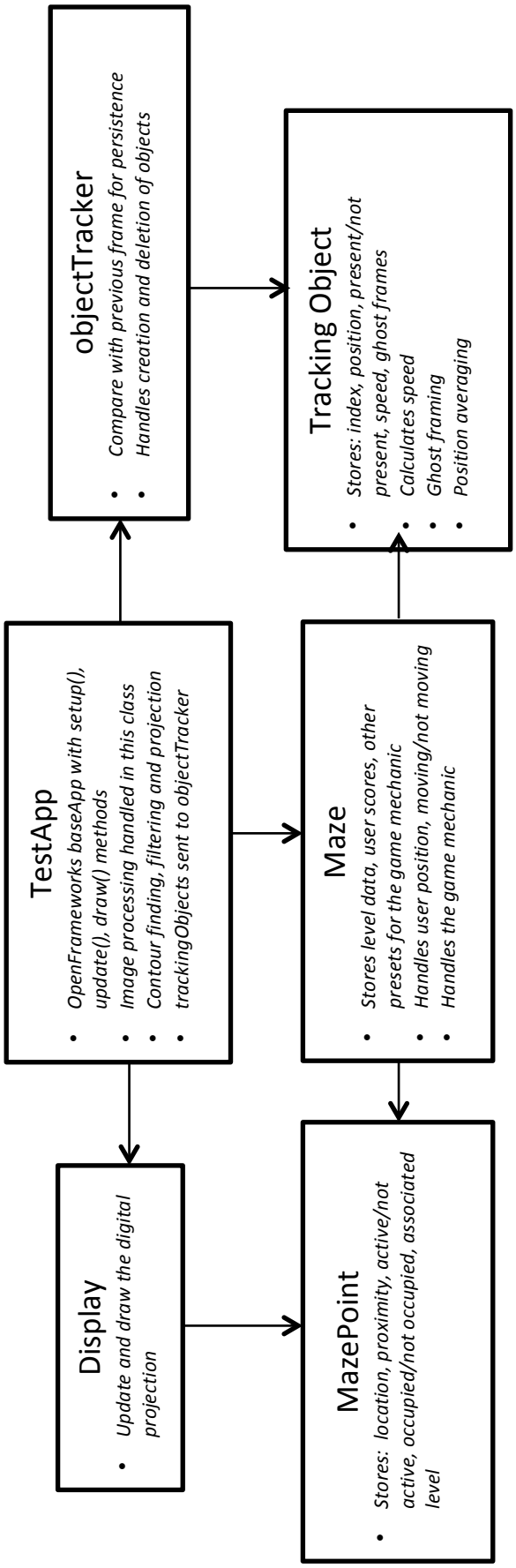
- Only two largest blobs representing the player's feet are selected.

- Virtual frame has been manually positioned in relation to the floor from the perspective of the camera and so effectively describes the floor plane.

- The projected trackingObjects are tracked for persistence and ghost framing is applied for better reliability.

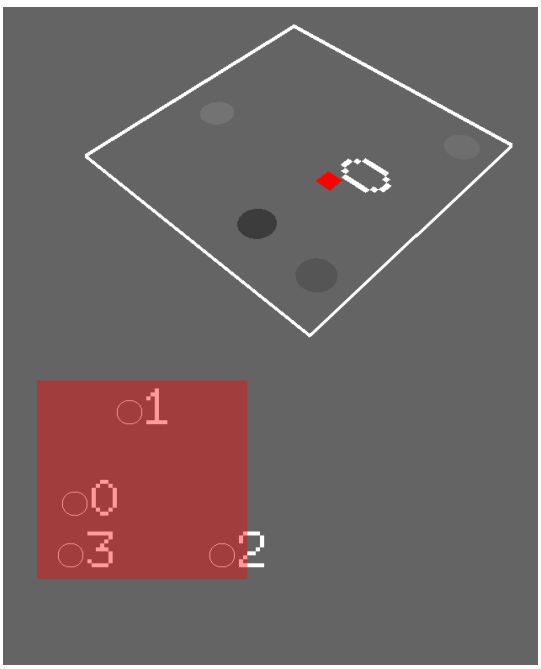
- The user position is the average of the two trackingObjects

Figure 3.5 Random Walk - class structure



As is often the case with games and installations designed for public spaces, the game mechanic had to respond to a number of conflicting criteria as well as the physical limitations that the chosen environment and technology imposed. The original concept was to have a complex maze, requiring not only specific directions, but also specific speeds of movement. The play area would be a white stretched canvas onto which players could write instructions to help following players, allowing each player to learn from the previous one's experience – a metaphor to scientific progress – whilst simultaneously producing a choreographic score of the random walk over the course of a day. However, complications with attaching the canvas to the marketplace floor and concerns about comprehensibility and playability lead to an alternative mechanic, which was perhaps conceptually poorer, but more immediate in terms of players' engagement with sound. In this version, for a single player with no floor canvas, the hidden spots were based on location only and required no set order of movement between them. When players believed they had found a spot, they simply stopped moving; if they were correct the tracking program would provide sonic and visual confirmation. The maze was split into five levels of contrasting sound worlds each with varying numbers of spots to be found (Figure 3.6). Players were given a fixed amount of time to complete the maze and awarded points according to how quickly they managed to solve it. A digital projection provided information about their progress and instructions as to what to do next.

Figure 3.6 Random Walk - maze visualisation



The small red square in the top frame represents player position. The four spots are shaded according to their proximity to the player

As with previous computer vision work, some kind of immediate feedback that would reassure the user that the technology was indeed responding to their movement was desirable, and the obvious solution in this case was to output sound only as long as the player was moving. This provided an intuitive relationship with the interface that could be played by the user as if it was an instrument whilst creating a somewhat balletic impression of movement to music for any onlookers.

As has previously been mentioned, when the player was moving the sound responded in accordance to player's proximity to the hidden spots in a variety of ways depending on which level the player had reached. The samples where chosen for each level in order to form distinct subsets in terms of texture and timbre for each level. Such an approach fitted well with the structure of *Pleiades*, given Xenakis' categorisation of the movements by the materials of the instruments themselves. To achieve this, the SuperCollider implementation created a dedicated synth for each spot in the current level, taking a single parameter that represented the user's proximity to that spot. Each level had its own synthDef from which the synths were instantiated, specifying a variant of granular synthesis to be performed on the selected samples for that level. As in previous works, the motivation for taking only a single parameter was to maintain transparency in the installation's sonic-spatial relationships, as it was these upon which the game mechanic was based. Nevertheless, through statistical randomisations of other parameters a pleasing degree of variation was achieved. A description of the samples used in each level and the corresponding sound/proximity relationship follows (Figure 3.7). The positive and negative markings in the proximity mappings column refers to the polarity of mapping, with a positive polarity indicating that the value increases as the player approaches. The track numbers and times in the samples column refer to the recording as referred to in the introduction.

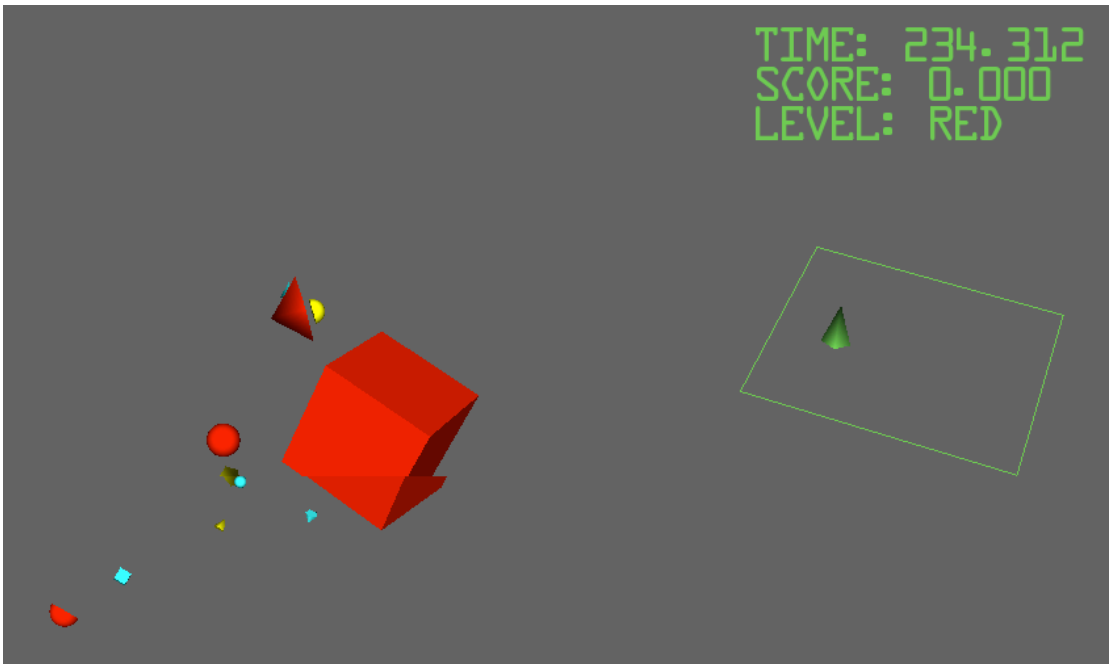
Figure 3.7 Random Walk - table of mappings

Level	Samples	Processing Description	Proximity Mappings
1	Kettledrum Track 1 6m17 Track1 6m21 Track 1 5m57	Stochastic playback rates including reversed grains. As the proximity reaches its maximum the sounds become forward playing only and the playback rate becomes stable.	trigger speed (+) rate (+) rate detune amt (-) prob reversed (-) grain position (+) grain duration (+) pan fluctuation (-)
2	Exotic bells Track 1 2m10 Track 2 3m15 Track 2 6m41 Track 2 7m15 Track 2 7m28	No reverse grains, but with playback rates extending below the sample rate. As the proximity reaches its maximum the grains become more prominent in the texture by virtue of increased pitch, duration and amplitude.	rate (+) grain duration (+) trigger speed (+) pan fluctuation (+) amplitude (+)
3	Tuned mallet percussion Track 2 5m22 Track 1 6m32 Track 2 4m56	Stochastic playback rates including reversed grains. This time a non-metrical trigger is used. As the proximity reaches its maximum the sounds become forward playing only and the playback rate becomes stable.	density (+) prob reversed (-) rate detune amt (-) grain duration (+) pan fluctuation(+) amplitude (+)
4	Tubular bell process Track 2 0m	No fluctuation in playback rate or range. Instead the position scrubs along the file , its processual treatment of the material creating the variation.	pos(+) pos fluctuation(-) pan fluctuation(-)
5	Pulse driven Track 1 4m05 Track 1 4m49 Track 2 9m35 Track 2 10m23 Track 2 12m19	Uses multiple samples with forward and reverse grains. As proximity reaches its maximum, the player converges on a single sample with forward playback only.	trigger rate(+) grain dur (+) amplitude (+) prob of other sample(-) pan (+)

Despite the limited parametric input of the synths, multiple types of osc message were required from the tracking program in order to correctly administer the audio in response to gameplay; for example, changing level, playing confirmation music, and resetting the game. The confirmation music consisted of a texture comprising all the untreated samples that had been heard thus far in the game. In this way as the player progressed further through the game they were rewarded with an increasingly developed confirmation texture.

In addition to the aural feedback I also added a visual feedback in which the various spots were represented by floating three-dimensional objects in space. As the player approached a spot the object representing that spot moved towards the viewer (Figure 3.8).

Figure 3.8 Random Walk – screenshot of visual feedback



Conclusions:

The installation was situated in the Spitalfields Market, a covered market place, on a Saturday during the Summer. The festival had organised a number of simultaneously running events which were spread out across the market place. The installation attracted shoppers and tourists as they passed by and so the audience was very diverse. A large proportion of active participants were families and children.

Though in terms of technology the piece performed well, and the game mechanic and sonic results were effective, retrospective analysis nevertheless reveals some design flaws in the work. In the first place, the visual feedback quickly overwhelmed the sonic interaction. I found that many participants faced the screen for the entire duration of gameplay using the sonic output as a secondary guide. A couple of subsequent tests showed that players were not only able to find the spots without the visual guide, but also became more focussed towards the sonic output and its relationship to their movement through the space. One option would have been to replace the projection entirely with a pre-recorded narrator to instruct the player.

I was also dissatisfied with the type of game-mechanic employed given the social environment in which onlookers weren't necessarily expecting to participate in an activity. A softer version without the arcade paraphernalia of levels, scores, and time limits would have been more appropriate and inclusive. In such a version one could envisage users casually approaching the installation and choosing whether to follow the game mechanic to some extent or simply enjoy the sounds that they were triggering. However, such an approach would have required tracking of multiple users without markers, a difficult goal to achieve given the technology available at the time. The subsequent release of the X-box Kinect has greatly eased the implementation of multiple user tracking, though this technology also brings its own restrictions for interaction design.

4. Soundpit

Video and code: http://www.simonkatan.co.uk/phd/soundpit.html

Overview:

Soundpit (19.06.2010) is an interactive sound installation involving the motion tracking of multiple coloured balls. In a strange meeting of Ping pong and *Plunderphonics* (Oswald, 1985), samples ranging from avant-garde compositions to pop hits are triggered and manipulated according to each ball's colour, speed, direction, and location. Numerous hidden rules governing aspects such the selection of samples, how sounds are processed, and what happens when balls collide, create an intricate labyrinth of audio-visual relationships for the curious participant to explore.

Soundpit was initially developed in response to a commission from Spitalfields Festival in 2010 for Xenakis Unbound in Spitalfields Market and exhibited under the name *Brownian Motion*. The piece has subsequently been retitled *Soundpit* and exhibited at Music Orbit, Borealis Festival 2011, Net Audio 2011, and SuperCollider Symposium 2012. With each showing the piece has undergone modifications to improve and extend the work, which are summarised below (Figure 4.1). As with *Random Walk*, the brief was for a sonically based game-installation that would appeal to children and adults alike and utilise samples from a chosen recording of Xenakis' *Pleiades.* However, the commissioning body requested that this second installation appeal to children as young as two years, and should therefore emphasise immediate response to interaction over a more rule orientated approach. The idea of creating a ball pit in which every ball produced its own sound, both suited young children and permitted me to make oblique reference to the physical model of Brownian Motion which Xenakis had employed in his music.

Implementation:

Using plastic balls as a tangible interface offered a distinct advantage over the sugar cubes of *Tecken 6.99* in that the objects themselves implied how they were supposed to be used (i.e. rolled along the floor). Furthermore, due to their lightness and uneven proportions, the

particular plastic balls that are used in children's ball pits roll with a strange irregular movement, which becomes increasingly apparent as they decelerate. So, in contrast to the more or less symbolic representation of movement in *Les Escaliers Mécaniques,* I was keen that *Soundpit* should capture this quality of movement in its sonification. The installation used four colours of ball with each colour triggering a distinct set of sounds. A range of different physical setups has been used across the various presentations of the installation with adaptations being made according to the environment. These have included the use of a containing wooden frame, mounting the camera on the ceiling, and even creating a tabletop version with Ping pong balls.

Figure 4.1 Soundpit – history of modifications

Event	Date	Modifications
Spitalfields Festival	06.2010	- Wooden frame setup - Xenakis samples only - Abstract visual output
Music Orbit	11.2010	- Multi sampling implemented - Live camera visuals
Borealis Festival	03.2011	- Retitled Soundpit - Ping pong balls on table top - Varese and Stockhausen samples - Rectangle triggered switching rules - Overlaid face visuals
Net Audio	05.2011	- Full room setup - Ceiling mounted camera
Supercollider Symposium	04.2012	- Wooden frame setup - Michael Jackson added to samples - Rectangle switching rule replaced with time triggered variant

Figure 4.2 Brownian Motion - Spitalfields Festival in 2010

Figure 4.3 Soundpit - Net Audio Festival in 2011

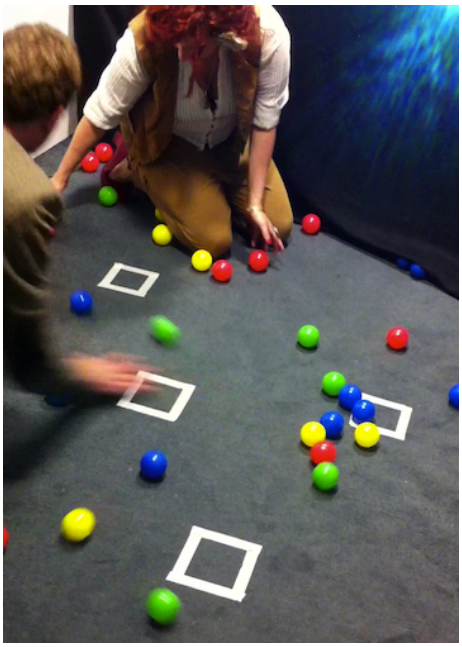
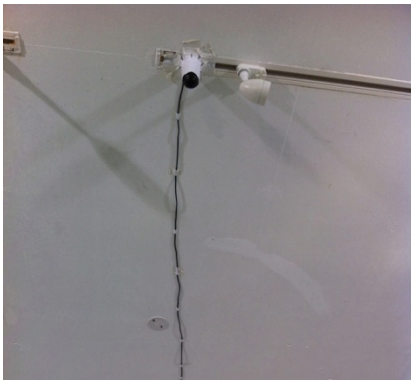


Figure 4.4 Soundpit - SuperCollider Symposium 2012

Figure 4.5 Soundpit - ceiling mounted camera



The tracking application was built in OpenFrameworks, though in order to accurately capture the movement of a large number of small objects, moving at speed, a variety of more sophisticated computer vision techniques were required. In the first place, I opted for the more elaborate background segmentation method of background averaging which builds a model of the background based on the average and standard deviation of brightness over multiple frames. As part of my implementation I was able to independently set thresholds for values above and below the average in terms of standard deviations. This allowed me greater control in excluding shadows from my foreground mask.

Rather than base colour segmentation on single pixel samples with manual adjustment of thresholds, a 2D histogram of hue and saturation values was built from a sample of pixels from an image of the object to be tracked. This technique allowed the creation of unique sets of data for each of the tracked colours, which captured the varying hues and saturations of the balls arising from their curved surface. Background segmentation was then carried out via OpenCV's back project function, which sets each pixel in the image to be segmented to the corresponding value in the 2D histogram based on its hue and saturation values. This yields an image in effect representing the probability of each pixel being a member of the type characterised by the histogram. Brightness thresholding, blurring, erosion, and dilation were then performed on the resultant image to produce a clean segmentation mask. This same process was repeated for each of the four colours. For increased robustness I added a further function allowing the use of multiple samples for the same colour and the resultant back projections to be summed and averaged. This proved particularly useful when working with variable or fluctuating lighting conditions.

For object tracking, I found contour finding and distance comparison to be too coarse a tool and so utilised the mean-shift algorithm. This works by using a search window and computing the centre of mass for that window (not necessarily the centre). The window is then centred on that point and the process repeated, potentially yielding a new centre of mass as the window has now taken in new data. This continues until the window stops moving or a maximum number of iterations are completed. For my implementation, I used contour finding to detect the appearance of balls. Search windows are then set in accordance with the size of the bounding box of the contour. When the ball disappears from view, the contour area reduces to zero triggering the termination of the mean-shift search window.

Figure 4.6 Soundpit - computer vision process

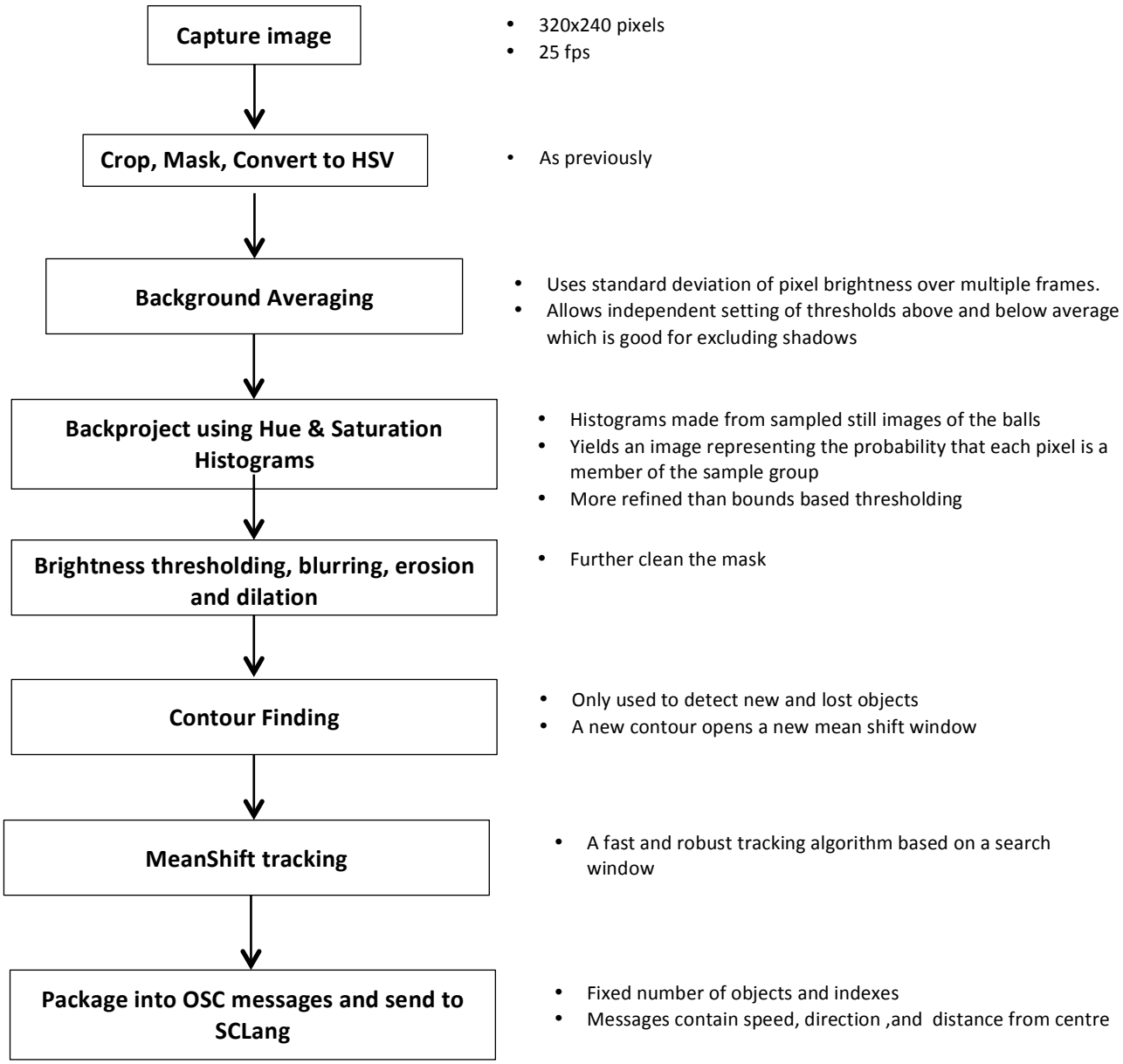
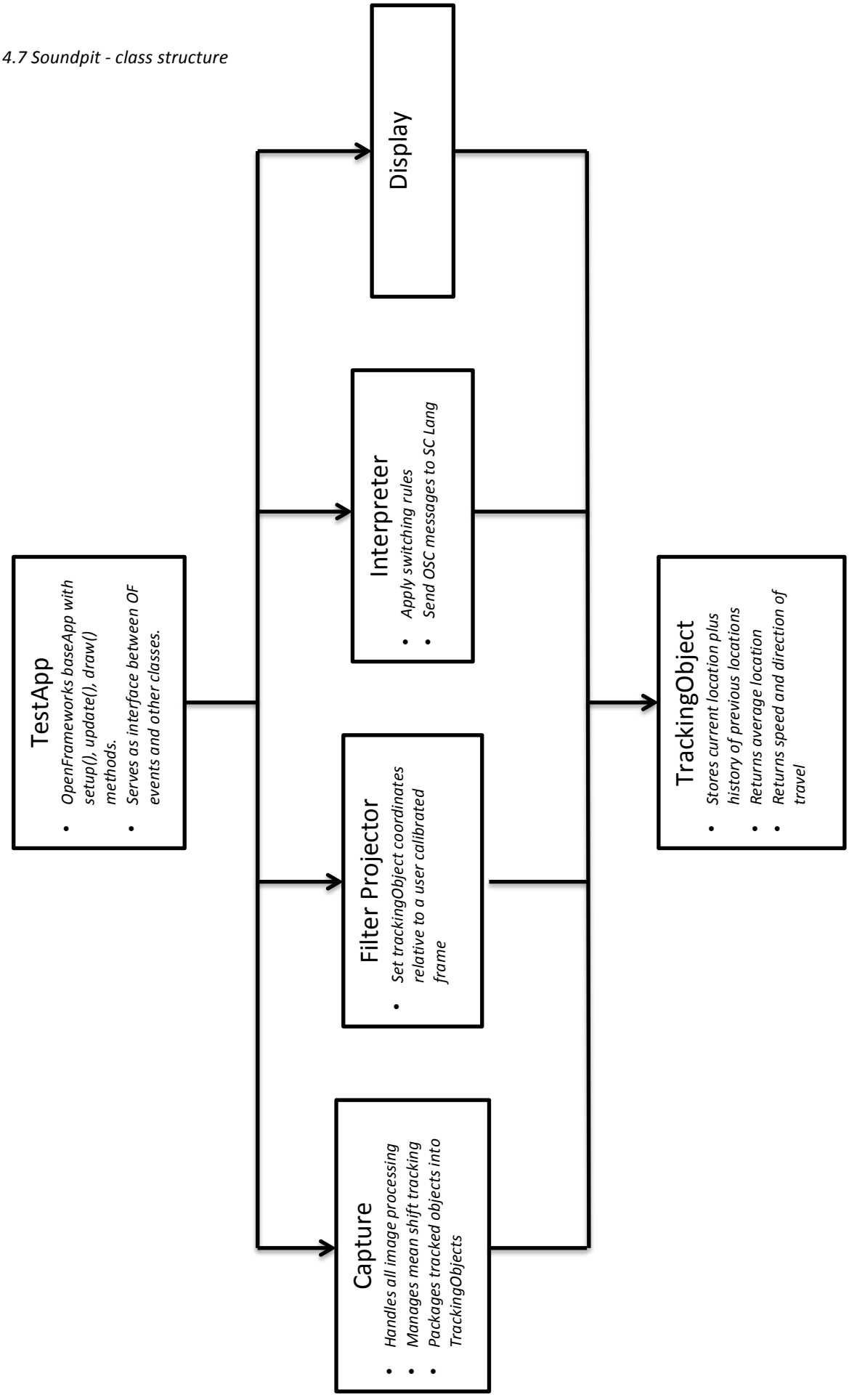


Figure 4.7 Soundpit - class structure



In the interests of clarity and conciseness in the OSC messaging between my tracking application and SuperCollider, I opted for fixed arrays of ten objects per colour with identical indexing in both programs. If there were more than ten balls of a particular colour, the surplus balls were simply ignored. In practice this aspect went unnoticed by the users, as by the time there were that many balls in view of the camera, the sonic texture was already very dense. Tracking messages sent to SuperCollider contained normalised values for the speed of the ball, the direction of travel, the distance from the centre point of the frame. A separate set of messages were sent to indicate when balls collided at speed, containing similar information as well as details of the colours involved in the collision.

As with *Random Walk* all the sound was produced via granular synthesis of samples. However, in this case, mostly smaller grain sizes were used in order to more accurately reflect the movement of the balls. Towards this end, custom envelopes with short attacks were used. Once again, the strategy was to use a multitude of simple, transparent and intuitive mappings as opposed to more complex ones. In some cases it was found that the simplest mapping techniques produced the most interesting results, for example, pitch shifting of the example according to direction of travel and adjusting the trigger speed according to the speed of the ball.

Figure 4.8 Soundpit - synthDefs and Mappings

SynthDef	Features	Speed =>	Direction =>	OffCentre=>
ShortSingles	• Loops a sample. • Variable playback rate with added random variation on each iteration	trigger speed	playback rate	pan width reverb mix
LongSingles	• Loops a sample. • Variable playback rate throughout loop	playback rate	-	pan width reverb mix
Portamento	• Granular synth. • Plays whole sample • Variable rate.	trigger speed	playback rate	pan width reverb mix
BandWidth	• Granular synth • Plays whole sample. • Playback rate randomly varied around a fundamental within a variable bandwidth	bandwidth of rate variations. trigger speed.	fundamental	pan width reverb mix
VarGrain	• Granular synth • Fixed start position • Variable grain durations.	playback rate	grain duration trigger speed	pan width reverb mix
ClassicGran	• Granular synth • Variable start positions. • Fixed playback rate. • Variable grain duration.	grain duration trigger speed	start position	pan width reverb mix
SeriesGrain	• Granular synth • Plays whole samples from a series of samples in a fixed order.	trigger speed	playback rate	pan width reverb mix
PSRGrain	• Granular synth • Variable grain duration • Variable start positions. • Always plays to the end of the sample.	playback rate trigger rate	start position	pan width reverb mix

Figure 4.9 Soundpit – colour synthDef and sample allocations

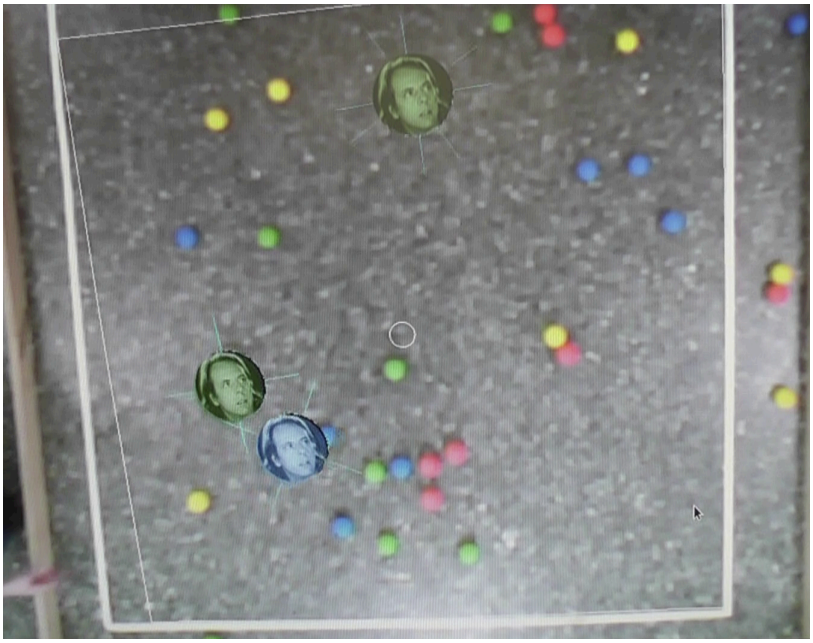
Ball Colour	Sample	SynthDef
Red	Xenakis - vibe_fade.wav	ShortSingles
	Stockhausen - elem2.wav	VarGrain
	Varese – mouse2_1m30.wav	LongSingles
	Michael Jackson – andIWont.aiff	LongSingles
Blue	Xenakis – highdrum1.wav	ShortSingles
	Stockhausen (multiple samples)	SeriesGrain
	Varese - demonicOrg_7m12.wav	VarGrain
	Michael Jackson – yaKnow.aiff	VarGrain
Yellow	Xenakis – tube_long.wav	Portamento
	Stockhausen - crazymix_short_15m_22s.wav	LongSingles
	Varese - fuzz1.wav	ShortSingles
	Michael Jackson – riff.aiff	ClassicGran
Green	Xenakis – T1_4m17_cuteQuiet.wav	BandWidth
	Stockhausen – weirdcollection_11m49s.wav	BandWidth
	Varese – nadaluu_4m30.wav	PSRGrain
	Michael Jackson – whosBad.aiff	BandWidth

Figure 4.10 Soundpit - collision rules

Colour A	Colour B	Sample	Delay effect
Red	Red	Xenakis	Y
Red	Blue	Xenakis	Y
Red	Yellow	Xenakis	Y
Red	Green	Xenakis	Y
Blue	Red	Stockhausen	Y
Blue	Blue	Stockhausen	Y
Blue	Yellow	Stockhausen	Y
Blue	Green	Stockhausen	Y
Yellow	Red	Varese	Y
Yellow	Blue	Varese	Y
Yellow	Yellow	Varese	Y
Yellow	Green	Varese	Y
Green	Red	Michael Jackson	N
Green	Blue	Michael Jackson	Y
Green	Yellow	Michael Jackson	N
Green	Green	Michael Jackson	N

The initial version of the installation used samples from Xenakis' *Pleiades* with different granular treatments for each colour. There were also several synths for collisions between balls. Subsequent versions saw the addition of samples from Stockhausen's *Kontakte* (1960), Varese's *Poème Électronique* (1958), and in homage to John Oswald's *Dab*(1989), Michael Jackson's single *Bad* (1987). Each colour had an associated sample from each of the chosen recordings and corresponding granular treatment. In order to provide clarity for the users in this more complex situation, a supplementary visual feedback was added. This comprised a projection of the camera view with images of the composers' of the sampled material faces overlaid on each ball as it moved across the screen. As the samples changed for each colour so did the faces.

Figure 4.11 Soundpit - overlaid graphics onto moving balls



The changing of samples was controlled independently for each colour according to a switching rule, the design of which was in part motivated by my observations of the initial incarnation of the installation. I had been dissatisfied with the constant density of texture, and the frenetic and generalised interaction of the users. In order to elicit more contemplative engagement, in which the properties of individual balls could be explored, the switching rule required brief moments of stillness in order to trigger changes of sample. The required duration of stillness was set at a length, determined through trial and error, which would produce a noticeable pause in sonic output but also allow occasional accidental triggering. In this way it was hoped that rather than having to be informed of the rule, users would discover it by chance. Each change of sample was accompanied by a sonic cue

consisting of an untreated fragment of the recording that the new sample was taken from. In earlier versions, the switching rule involved the marking of four squares in the capture area. The population of these squares by different coloured balls informed how the samples would change. However, in subsequent versions I determined that this aspect of the rule added an unnecessary level of complexity, and replaced it with a simple mechanism whereby the colour with the largest number of balls present would be the one to change samples. In order to reduce the sonic texture, samples triggered by collisions were limited to a dedicated collision mode for each colour, in which the ball would not trigger any sounds unless colliding with another ball.

Conclusions:

In its initial showing, the installation was placed in the covered market at Spitalfields running simultaneously alongside *Random Walk*. It was of particular appeal to very small children. This to a large degree can be attributed to the floor level proximity of the interaction. Subsequent showings used many different types of spaces from small rooms without natural light to bright gallery spaces. The audiences varied from those comprising predominantly academics and artists to ones made up of families and children. In all cases observing the interaction was enjoyable as the installation elicited playful responses irrespective of participants' age.

In comparison to the previous installations, *Soundpit* achieved a more intuitive form of interaction leading to observably deeper levels of engagement from greater numbers of users. This was in part due to a looser rule base, although perhaps more significant was the implementation of robust and fast tracking of the balls in motion which allowed their quirky movement to be convincingly rendered in sound. The resultant causal association between the movement of the ball and the sonic output created a sonic persona for each of the balls in the mind of the user, which was further amplified by the projection of the sampled composer's face onto the balls. Essential in making these aspects clear was the rejection of more complex sonification strategies such as mapping multiple balls to single synths, or asynchronous aural feedbacks. Though the later inclusion of multiple sample categories was intended to compensate for the simplicity of the sonic mappings, it was clear from observation that users were amply engaged by exploring the sonic-spatial relations of the different balls, even when the samples didn't change. This lead to a lack of motivation to make the sample group change and hence the switching rule less effective at initiating

stillness than I had originally hoped. Whilst the projection was enjoyable and made the concept of sample groups clear, it also created a degree of tension between the visual aspect of the ball pit and the screen, with the screen invariably drawing users' attention away from the physicality of the ball. Ideally the sites of visual feedback and interaction would be combined, for example by flying a projector over the ball pit, though this would raise new difficulties with regards to the computer vision implementation.

5. Nautical But Noise

Video, code, and score: http://www.simonkatan.co.uk/phd/nautical.html

Overview:

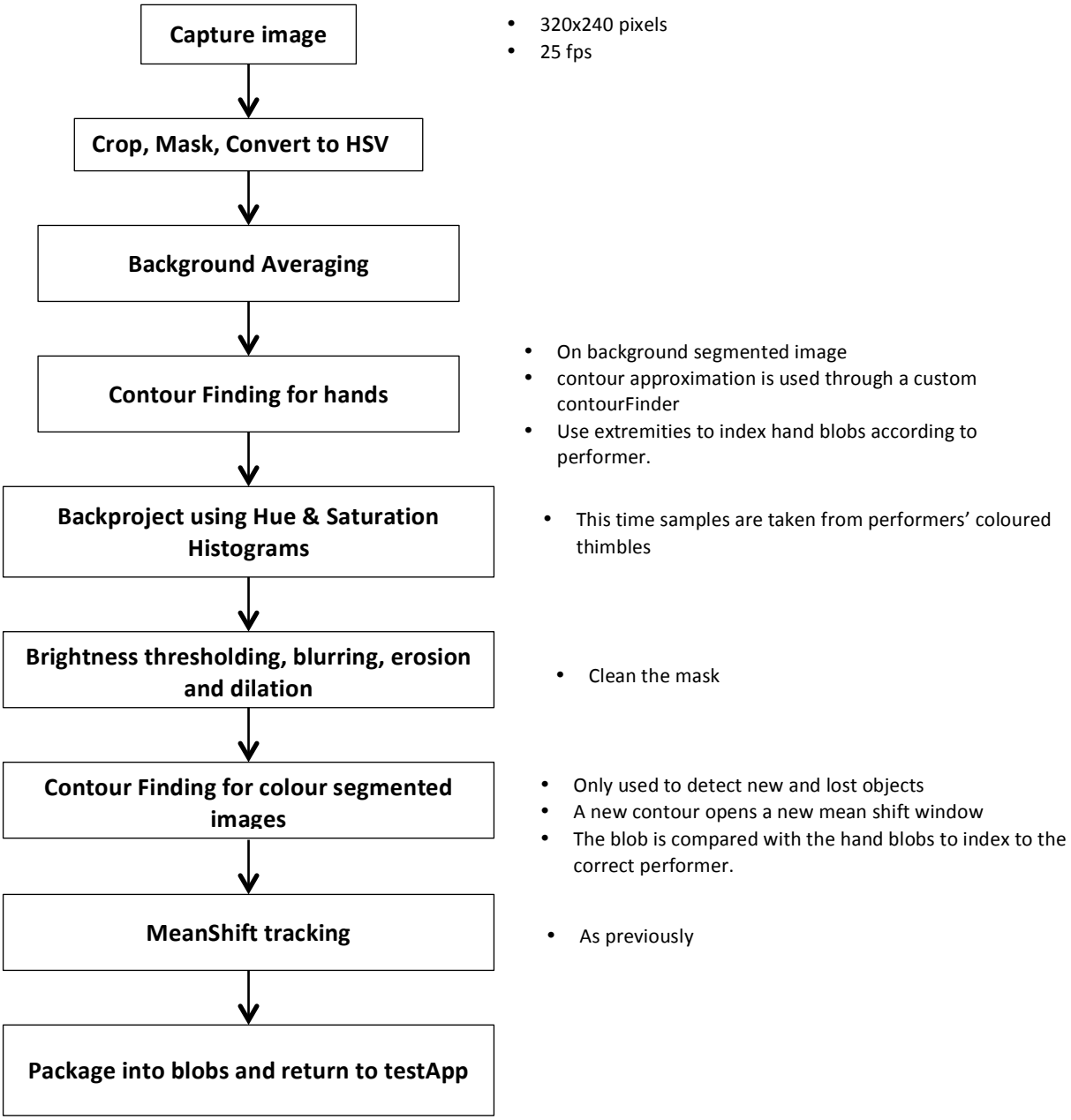
Nautical but Noise (11.11.10) is an audio-visual composition combining live-hand tracking with overlaid graphics, and granular synthesis. The project, commissioned by the Sampler Festival Lewisham, was a collaboration between Luke Fraser and myself. Having gained a good understanding of colour tracking through *Les Escaliers Méchaniques* and *Soundpit*, I was keen to build an interface using the same techniques that would allow fine motor control with fingers and could be used to make a linear composition. In particular I wanted this piece to combine the symbolic complexity of the multiple mapping systems of *Les Escaliers Méchaniques* with the sonification of quality of movement achieved in *Soundpit*. I proposed that for the collaboration I should produce a control interface outputting OSC with normalised data, Luke should produce a sound interface, and that we should then co-devise the final piece.

Initially the commissioning body suggested that we would be doing a situational performance in a local café or pub. The location prompted thoughts of the type of music often played on radios in cafés and pubs and so arose an idea of using 80s pop samples through granular synthesis with the artists' heads floating above the performers' fingertips. It seemed a logical step to connect fingertips together with graphical lines, the connections resulting in granular 'mash-ups' between the respective artists. After some time the commissioning body informed us that they were unable to find a café or pub willing to host us and that we would instead be performing at Synthesis, the festival's electronic dance music night. Luke deemed that the theme needed an extra tweak for the situation and so decided on the nautical setting opting to work with Enya's, *Orinoco Flow* (1988), Christopher Cross', *Sailing* (1980), Sissels' and James Horner's *Titanic: Music from the Motion Picture*(1997), and Roger Payne's, *Songs of the Humpback Whale* (1970).

Implementation:

Initially conceived for a café style table, the interface uses a camera mounted on a mike stand looking down onto the table with performer's seated on either side. Each performer wears four differently coloured thimbles (red, yellow, blue and green), two on each hand. Each colour is assigned to a different artist and, on placing their hands under the camera, the button-like graphic of the relevant artist will be overlaid onto the respective thimbles. As this happens, OSC messages are sent to a dedicated audio computer to trigger the corresponding synth, which uses samples from the portrayed artist. Included in the messages are the normalised coordinates of the button graphic, which are used to control parametric variables in the synth. When both performers place their hands on the table, various connections between fingers are formed according to an algorithm and are rendered as lines on the graphical output. These connections result in OSC messages containing the normalised coordinates of both connected button graphics, which in turn causes the triggering of other synths combining samples from the two corresponding artists.

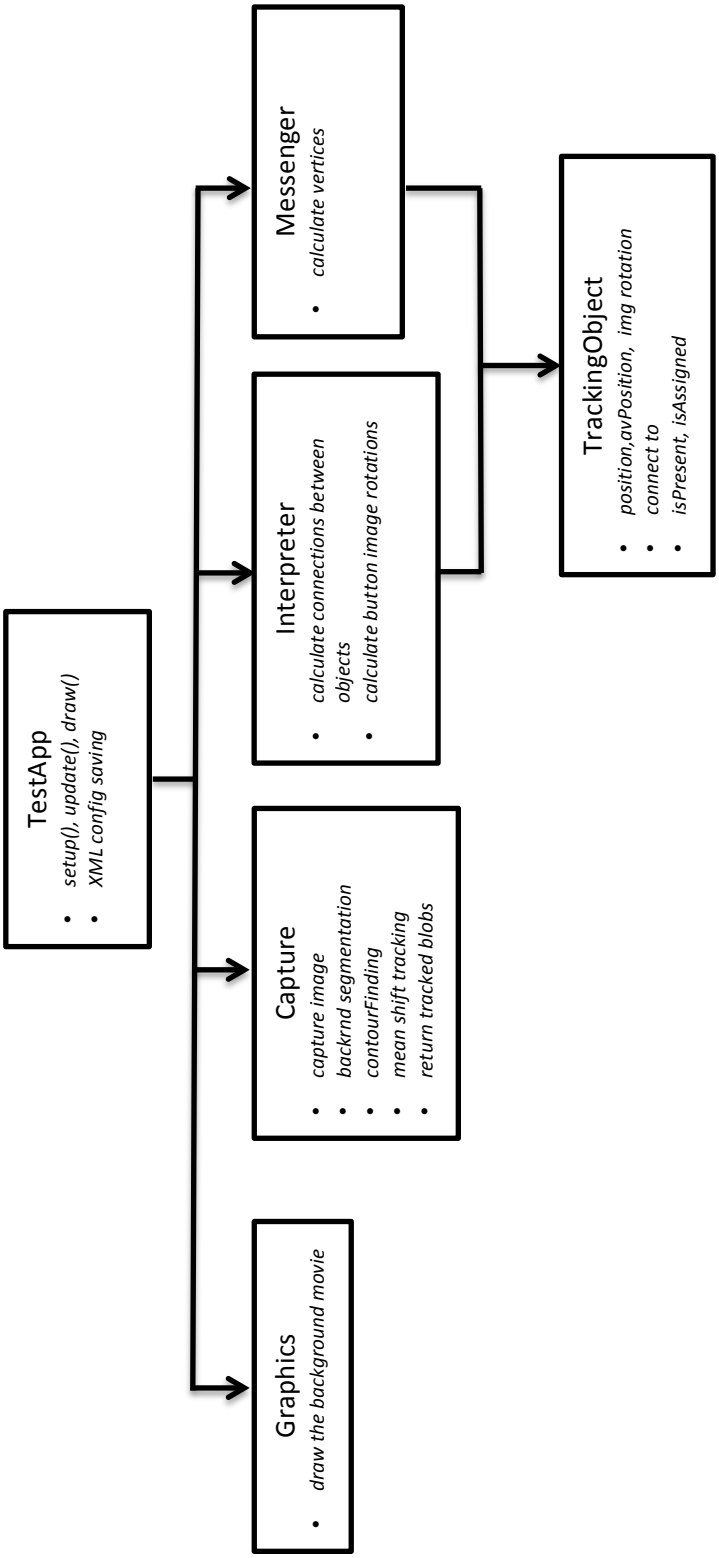
Figure 5.1 FingerTracker - Computer Vision process



The OpenFrameworks interface FingerTracker, uses much the same computer vision techniques as *Soundpit* with a couple of minor additions relating to the tracking of fingers (Figure 5.1). The background segmented mask is used for the identifying of hands and arms. For this a custom contourFinder, which uses OpenCV's approximation functions to produce simplified contours, is used. As the blobs for performers' hands and arms will necessarily

extend to the edges of the screen, a simple algorithm using the x coordinates and widths of the blobs is able to identify which arm belongs to which performer. The coloured blob tracking for the fingers works in the same way as in *Soundpit* with the exception that on initialising new mean shift windows, the newly discovered contours are compared with arm blobs for intersections which are used to assign the window to the relevant performer. One issue with this method is a potential reassignment of mean shift windows when the performers place fingers of the same colour close enough to each other that they can be interpreted as a single blob. This is overcome by continually checking mean shift windows against arm blob contours whilst fingers of the same colour are in close proximity and reassigning the mean shift window to the opposite arm if necessary. Similar methods are used to prevent misallocations on blob initialisation though this is less likely to occur. A final complication arose from the varying colour of skin tones. It was found that the interface often mistook patches of skin for the various colours to be tracked, and so the careful adjustment of lighting and thresholds to exclude the skin was required.

Figure 5.2 FingerTracker - class diagram



The algorithm assigning connections between performers fingers which is contained in the interpreter class excludes connections between fingers from the same performer and only allows one connection per finger. With four tracked nodes for each performer there are a total of twenty-four possible combinations (Figure 5.3). Combinations 1 to 8 only occur when no connections are possible, otherwise the connection assigns a score for each colour which is shown in brackets and calculates the absolute difference of all the possible connections in the given frame. Precedence is then given to connections of a lower score, meaning that connections between fingers of the same colour are ranked first. Where equal rankings occur, such as between combinations 13 and 15, or 22 and 24, precedence is given to the combination with the lower sum.

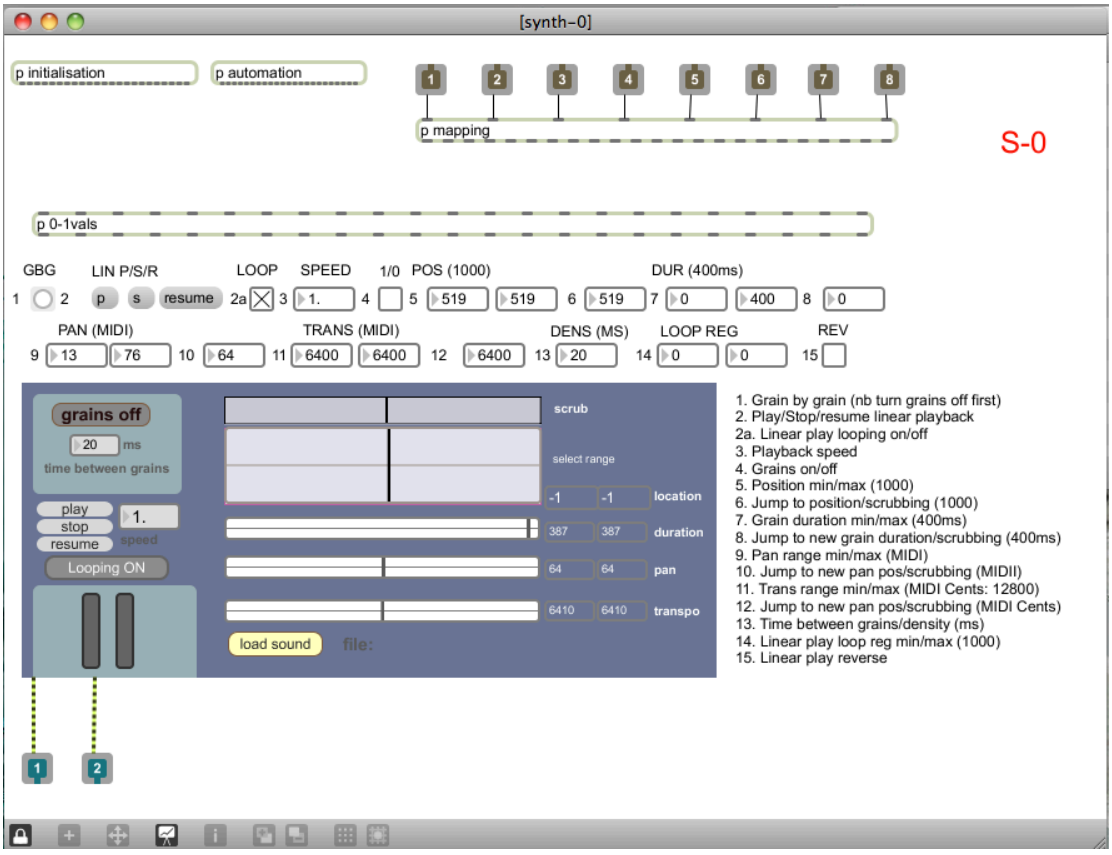
Figure 5.3 FingerTracker - Performer colour combinations

Combination	Synth	Performer 1 Colour	Performer 2 Colour	Abs Difference	Artist Combination
1	16	Red (0)	-	-	Enya
2	17	Yellow (1)	-	-	Sissel
3	18	Green (2)	-	-	Christopher Cross
4	19	Blue (3)	-	-	Roger Payne
5	20	-	Red (0)	-	Enya
6	21	-	Yellow (1)	-	Sissel
7	22	-	Green (2)	-	Christopher Cross
8	23	-	Blue (3)	-	Roger Payne
9	0	Red (0)	Red (0)	0	Enya
10	1	Red (0)	Yellow (1)	1	Enya/Sissel
11	2	Red (0)	Green (2)	2	Enya/Cross
12	3	Red (0)	Blue (3)	3	Enya/Payne
13	4	Yellow (1)	Red (0)	1	Sissel/Enya
14	5	Yellow (1)	Yellow (1)	0	Sissel/Sissel
15	6	Yellow (1)	Green (2)	1	Sissel/Cross
16	7	Yellow (1)	Blue (3)	2	Sissel/Payne
17	8	Blue (3)	Red (0)	3	Payne/Enya
18	9	Blue (3)	Yellow (1)	2	Payne/Sissel
19	10	Blue (3)	Green (2)	1	Payne/Cross
20	11	Blue (3)	Blue (3)	0	Payne/Payne
21	12	Green (2)	Red (0)	2	Cross/Enya
22	13	Green (2)	Yellow (1)	1	Cross/Sissel
23	14	Green (2)	Green (2)	0	Cross/Cross
24	15	Green (2)	Blue (3)	1	Cross/Payne

In addition to the indexes and normalised coordinate positions of the respective fingers, the data in the OSC data also includes the coordinates of the mid-point and the distance between the two fingers. In combination this data controls 24 synths in Luke's machine. All

the synths are derived from a patch based around a modified granular synth realised in Max MSP (Figure 5.4).

Figure 5.4 Nautical But Noise - modified granular synth



This synth offers various treatments of the given sample ranging from simple looping playback to classic granular synthesis with OSC parameters freely assignable to different sonic parameters. Rather than give an exhaustive breakdown of all the synths I've given a description of a few contrasting ones as examples. A simple example is synth 16 which appears in section A2 (Appendix A or online score) and is triggered by my solo Enya. This synth simply loops the whole sample mapping the play speed to the x coordinate and a pitch shift to the y coordinate. An additional feature is that the particular sample is chosen according to what quartile of the screen it's in. This is clearly apparent from the change of sample in section A3. Synth 5 which is triggered by a duo of Sissels provides a more complex treatment and mapping. Here the processing is granular with the x coordinate of the connecting line between the duo controlling the grains position in the sample, the y coordinate controlling the grain length, and the length of the line controlling the trigger tempo with shorter lines resulting in faster tempos. This set of mappings offers a fair degree of flexibility which can be seen in the contrasting uses of the synth in sections B1 and C1. A

third example is synth 4 which is triggered by a combination of Sissel and Enya. This is a dense granular treatment using random grain sizes where the search region is mapped to the minimum and maximum x values and the range of transposition of grains is mapped to the minimum and maximum y values.

A final point to note in the construction of the interface is the animated graphic backdrop which was built by graphic designer and programmer Paul Martin. In the original concept the waves were supposed to respond to the amount of ongoing activity in the screen. However, GPU limitations necessitated the use of a lower resolution pre-rendered video. Nevertheless, with the ghostly hands hovering beneath the waves, the overall image is still effective.

Once the synths and the interface had been built the piece itself was created collaboratively between Luke and myself through a process of devising. This involved improvising with the various synths and making extensive notes and comments. Through this process sections of material gradually emerged from which the final piece was constructed. During this process an esoteric form of notation was derived from which the score of the piece was written. Though the performers' actions are clear from the video, I have nevertheless included a descriptive score which uses this notation in Appendix A and in the online resource.

Conclusion:

Nautical But Noise has been performed twice in strongly contrasting environments. The first was at Sampler Festival Lewisham. The venue was large and had large screen and powerful sound system. Luke and I operated the interface to one side of the screen dressed in nautically related apparel in view of the audience which at a glance were mostly in their twenties and thirties. The second performance took place in a ferry terminal in Bergen using a small projection onto the wall above a luggage conveyor belt. Luke and I were in the same attire in a more discrete position under the escalators which faced the conveyor. We began the performance just as a group of around four-hundred elderly ferry passengers disembarked from a long and apparently rough trip. In retrospect it was unsurprising that after their ordeal at sea they were unreceptive to sight of yet more rolling waves. Without exception they collected their luggage and left as soon as they could manage.

Adverse weather conditions aside, *Nautical But Noise* received a warm response in performance, and indeed there were many positive aspects of the piece. The configurations of connecting lines between fingertips were pleasing to eye and captured the imagination of the audience. As with my previous computer vision works there was some wild speculation as to how the effect was achieved. A second pleasing aspect was the ambiguous narrative that the use of characters faces tied in with matching samples and the nautical backdrop suggested. One might interpret periods of rapid movement accompanied by noisy granular treatments such as in section D1 as conflicts, and interpret the coming together of all the characters in F6 as a final resolution. This narrative effect is reinforced by the initial introduction of all the characters using solo synths, in which the maintenance of the integrity of the sample is the greatest, and the gradual arrival at more obscure combinations as the piece progresses. The addition of the ghostly projection of our hands to this scenario adds dual levels of agency to the performance. The pretence of the autonomy of the various finger puppets is a shallow one. Nevertheless, as the video footage demonstrates, the hands themselves take on a disembodied quality quite apart from the spectacle of Luke and myself sitting at the table nearby.

In terms of the communication of mappings *Nautical but Noise* switches freely between transparency and obscurantism. Mappings such as shown by the solo Enya synth in section A3 or the Enya and Sissel duo synth in section E4 have clearly discernable parameter mappings, although the use of longer length samples prevents any communication of the quality of movement. On the other hand the solo humpback whale synths that appear in section A1 and subsequently in B2 don't exhibit any linear behaviour to the audience. In this case the unpredictability of the behaviour arises from the use of audio samples with long silences in them. Another obstacle to the correct interpretation of mappings is the complex interplay between parameters in some of duo synths caused by the independent movement of the two performers, combined with occasionally counter intuitive mapping combinations, for example in using minimum and maximum coordinate values to control granular search ranges. This lack of transparency is further exacerbated by the simultaneous presentation of multiple synths for example in sections E3, F3, and F6. Here the audio streams are impossible to separate and the audio-visual relations can only be understood as a single gestalt. Of course some of the synths have already been explored in earlier sections, although due to the permutative nature of the symbolic system it's not easy to recall earlier synths. In retrospect it may have been better to see fewer synths more thoroughly explored.

This occasional lack of connection between audio and video is perhaps symptomatic of the delineated collaborative approach which we adopted. We both worked independently on our aspects of the project, communicating ideas via phone and email, and then came together to devise the final work when the majority of creative decisions about the interface had already been made. This left us with something of a straight jacket within which to work. For one thing the method for switching between synths had an arbitrary quality to it and excluded many combinations that we might have found desirable. One option might have been to have a foot pedal system which would change the rules governing how connections were made. With regards to the computer vision, the difficulty in segmenting skin from the coloured markers indicated that my particular approach to problem of tracking fingertips was not ideal. A more robust method might have used contour analysis to find fingertips instead.

Nevertheless such self-criticisms don't consign *Nautical but Noise* to failure. The final form simply has different priorities from the initial concept. Rather than progress through an intricate build up of audio-visual relationships, the piece operates on a narrative level via an ever changing array of sonic treatments of material personified through the graphical overlay, the final result taking on a quality of ersatz opera come finger puppet show.

6. God Over Djinn (SoundNest)

Video, code, and score: http://www.simonkatan.co.uk/phd/god.html

Overview:

> '*Genie:* Oh aren't you acquainted with recursive acronyms? I thought everybody knew about them. You see, "GOD" stands for "GOD Over Djinn" – which can be expanded as "GOD Over Djinn, Over Djinn" – and that can, in turn be expanded to "GOD Over Djinn, Over Djinn, Over Djinn" – which can, in its turn, be further expanded ... You can go as far as you like.
> *Achillies:* But I'll never finish!
> *Genie:* Of course not. You can never totally expand GOD.' (Hofstadter, 1999, p. 113)

God Over Djinn (09.03.2011) is a computer visual and synthesised sound composition that draws on our intuitive understanding of the physical world, playing with ambiguities between scale and perspective in two-dimensional representation to create seemingly infinitely nested worlds of colliding objects. Whilst the triggering of sounds is subservient to the Newtonian physics that guide the objects, the apparent one to one nature of its mappings belie a more complex relationship between the visual and the aural in which the hierarchical nesting of objects has varying aural consequences and the sonic results of visual events continually shift throughout the piece.

As the title suggests, *God Over Djinn* was in part inspired by Douglas Hoftsadter's, Gödel, Escher, Bach, and in particular draws from its references to recursion, which performs a pivotal role in the composition. In the first place I wanted to use recursion as a tool for creating hierarchical structure within a visual world, which could then be used towards musical ends. Through the process of composition I discovered many more possibilities which recursion offered, not least of which was the ability to imply a limitless visual and hence compositional space. Hoftstadter's book soon lead me to the drawings of M.C.Escher, himself fond of recursive devices. However, the aspect of his work that most interested me was the bending of implicit rules of perspective to create paradoxical worlds and the use of commonly recognisable forms to imply multiple orientations and light sources, forcing viewers into simultaneous contradictory interpretations. A further and interrelated source of inspiration came from research into the function of implicit knowledge in music cognition. Tilman and Bigand's description of tacit understanding of musical structures within short

time windows by non-trained listeners, as well as Bob Snyder's conjecture that our conceptualisation of music might rest on pre-conceptual image schema, lead me to wonder whether, just as Escher used familiar objects to create a base of implicit understanding, I could tap into a stream of implicit knowledge within the audio visual world to do the same. The final inspiration came from very early 2D computer games such as *Pong* (1972) and *Asteroids*(1981) which due to the near abstract quality of their limited graphics, rely heavily on implicit knowledge about the physical worlds they are representing in order to be playable.

Implementation:

As with the previous works the composition of the piece involved the design and creation of a bespoke software interface using OpenFrameworks with sound from SuperCollider. However, in this case the software package, called Sound Nest, is intended to ultimately become a composition and performance tool in its own right. For this reason I will describe the compositional process in two parts, the design of the interface and the subsequent composition of the piece.

SoundNest Implementation:

My starting point was to intuitively consider areas of implicit knowledge that 2D computer games rely upon and might be useful for my interface design.

Figure 6.1 SoundNest - Implicit knowledge used by 2D computer games

Area of knowledge	**Assumptions**
Gravity/ Zero Gravity	Objects have mass fall to the ground (we equally understand zero-gravity environments)
Restitution	Objects rebound when they hit surfaces
Containment	Objects can be contained by other objects
Motion	Objects can be propelled or can be self-propelling
Momentum	Objects can lose energy and slow down
Attraction	Objects can attract other objects
Size	Larger objects require more force to move than smaller objects
2D perspective	A 2D space can imply a 3D scene from a viewpoint in perfect alignment with the Z-axis (eg. a perfect overhead shot)
Size/Distance	When objects get smaller they are moving away from us. When all objects get smaller equally we are moving away from the objects.
Restitution/Depth	Objects which don't rebound in 2D space imply depth (i.e. one object is in front of the other)
Screen as viewport	The screen can be a window on a larger world

It was clear that containment would be essential to the environment in order to facilitate the desired recursions – containers can be contained by containers and so on. However, quite how the objects should behave in other regards, for example, whether they should respond to gravity or be self-propelling was less clear. Nonetheless I opted to use Box2D, a popular physics engine, to build my interface as it encapsulated most of the physical behaviours that I would likely require.

Secondly I considered possible audio-visual relationships for my system with a similar emphasis on what could be implicitly understood.

Figure 6.2 SoundNest -Potential audio-visual relationships

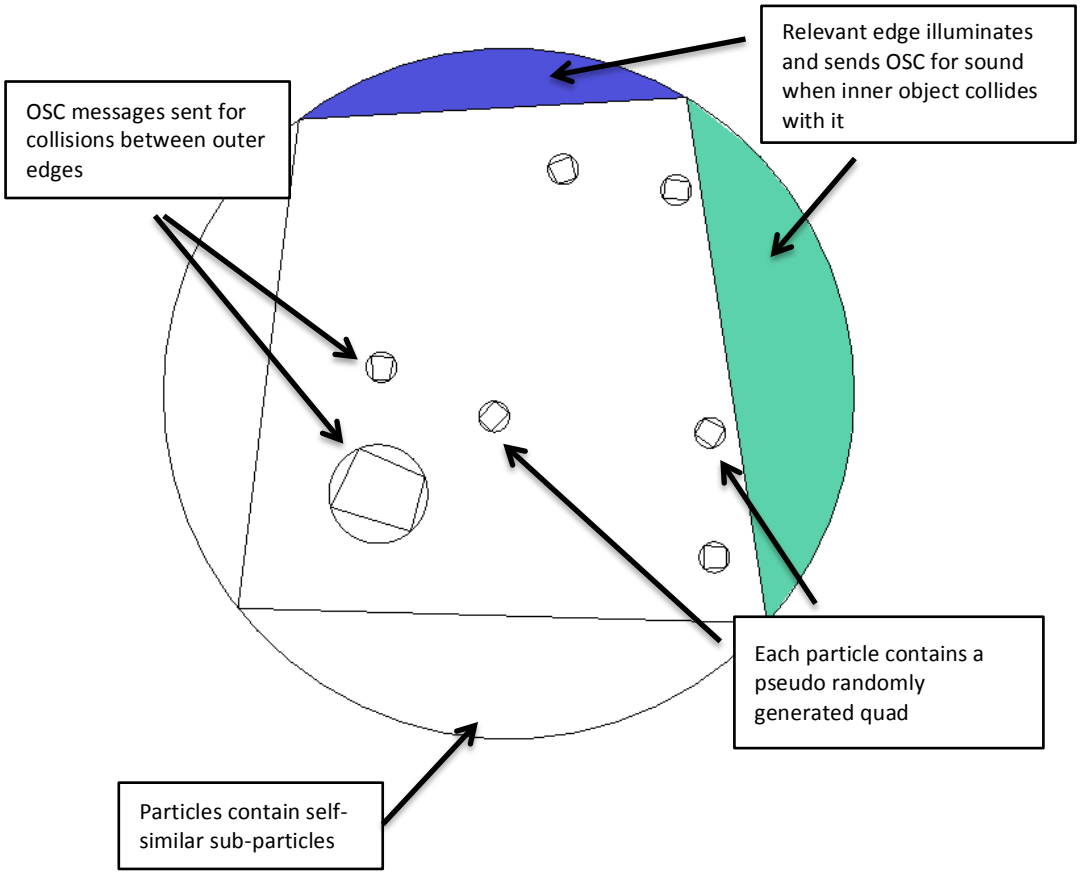
Physical parameter	**Audio result**	**Real world relation**
Collisions between objects	A sound results	Direct correspondence
Speed of collision	Changes volume/freq/timbre	Faster collisions have more force
Size of objects colliding	Larger objects produce lower sounds	Common experience of resonating objects (eg. drums)
Particular surfaces involved in the collision	Different sounds result	Implies surfaces are made of different materials, or are like buttons on a console
Self propelling objects	Produce a continuous sound, varying according to speed	Implies an internal process causing the movement
Distance of object from the viewer implied through size	Louder or quieter	Direct correspondence
2D position of objects or collisions	Spatialisation	Direct correspondence

A notable exclusion from the above table was the mapping of vertical position to frequency. Indeed, though the metaphorical relationship between pitch and height is well known, there is no physical phenomenon to justify its inclusion here. Although pitch to height mappings did appear in the final composition, I was keen to exclude them at this stage in order to ensure that the piece did not become overly reliant on such devices.

After extensive prototyping with Box2D and SuperCollider I arrived at system of near identical particles in a zero gravity environment with sounds resulting from collisions between the various particle surfaces (Figure 6.3). These particles are not self-propelling, but instead are created with an initial force and direction. Each particle has five discreet surfaces –one outer edge to be triggered by collisions with other particles, and four inner edges to be triggered by collisions with the sub-particles it contains. The two types of

surface send contrasting OSC messages reflecting their different intended uses. Collisions between outer edges are intended for sounds of indefinite pitch. The OSC message contains normalised data about the speed of the collision, the sizes of the two colliding particle, the x plane position, the impulse of the collision as well as a value in seconds for the decay of the sound, to be elaborated on shortly. Collisions between particles and inner edges are intended for pitched sounds. In this case the OSC message contains an index reflecting which of the four edges has been struck, normalised values for a fundamental frequency, range, decay, and pan, as well as an unspecified extra command that can be set as a string. This setup allows a highly flexible approach to sonification. By sending a fundamental frequency to SuperCollider, various mappings of physical to sonic parameters can be manipulated and performed in SoundNest, whilst the inclusion of the non-specific range value and an extra command allows SoundNest use of SuperCollider's many tools for manipulating frequency values.

Figure 6.3 SoundNest – annotated image of the particle system



Initial experiments involving nesting containers within containers in Box2D had produced unsatisfying results. I found that inner particle forces tended to cancel out outer particle

forces meaning that it was quite hard to keep particles moving. Furthermore collision detection in Box2D was not reliable enough to ensure that fast moving sub-particles would never escape their containers. This led me to adopt the rather unconventional solution of using multiple Box2D worlds. With this solution, each particle has its own world populated by the inner edges and whatever sub-particles that particle contains. In addition to this each particle also holds a reference to a body existing in the Box2D world of its containing super-particle. The only exception to this is the outermost particle that is not contained by any particle.

Aside from more predictable physical behaviour, this system offers a number of other advantages. Despite the implications of its name, the Box2D world is actually lightweight in terms of CPU and so having multiple worlds adds little extra load. However, reducing the number of bodies in any single world saves a significant amount load as the demands of collision detection grow exponentially with the number of bodies. It also removes the need for a large particle size range, which would have conflicted with Box2D's intolerance of very large or very small bodies. Furthermore, it also means that the body data for each sub-particle is local to its super-particle rather than global to the entire system, which gives more reasonable data ranges for sonification. Finally, as each particle holds all of its containing particles and all particles are of the same class, a method called on a particle can recursively call the same method on all the particles it contains, with those particles calling the same method on all of the particles which they contain and so on until empty particles are reached. These recursive function calls greatly reduce the complexity of drawing the particles. The draw function simply scales, translates and rotates the coordinate geometry according to the particle's current attributes and draws the relevant shapes. Then, without altering the current geometry, the program calls the draw function for all the sub-particles and so on. Not only does this greatly reduce the complexity of the code, but it's also pleasingly analogous to Hofstadter's GOD which inspired the piece.

Figure 6.4 SoundNest - class structure

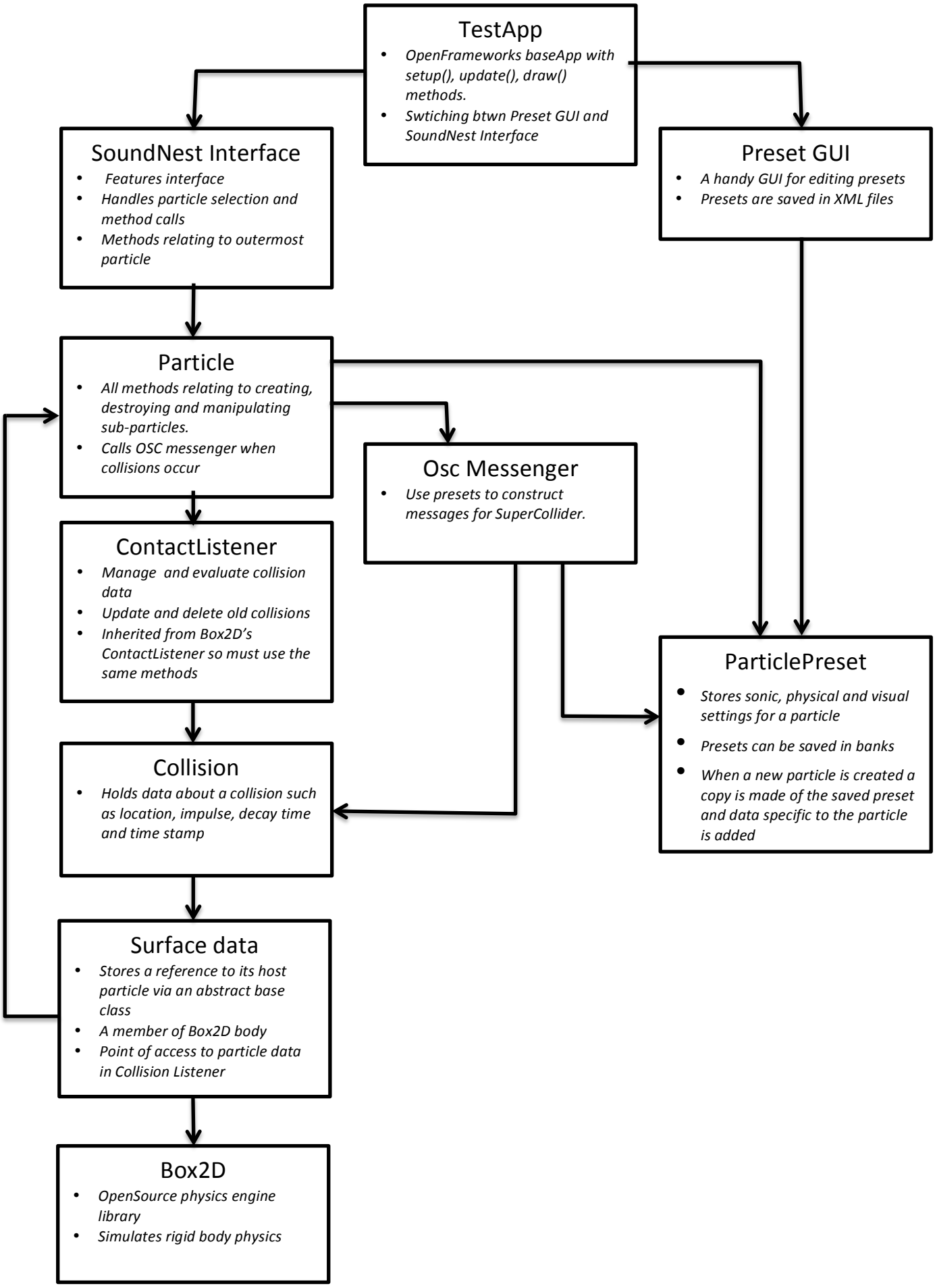
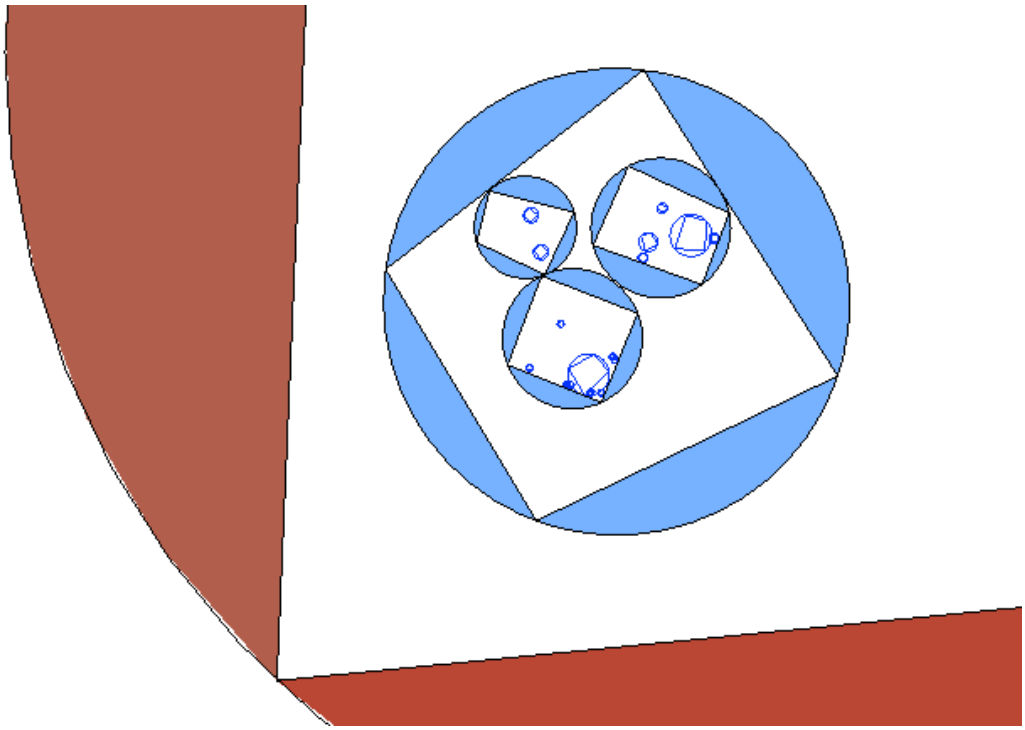


Figure 6.5 SoundNest – Particles and sub-particles



The features and layout of the control interface were developed through iterative design with a focus on usability in the live context. Towards this end, economy of key commands was of prime importance so as to avoid the use of complex menu systems. Seeing as there was no separation between control surface and visual output, a further requirement was for any graphics relating to user controls to intrude as little as possible onto the visual output. Figure 6.6 lists of all the features in the final program though many other features were developed in the design process and subsequently rejected. There then follows some description and commentary on key features. Though it will most likely be difficult for the reader to discern the potential for variety and nuance in SoundNest from this description alone, it is hoped that the documentation, descriptive score and commentary on *God Over Djinn* that follows will serve as an exposition.

Figure 6.6 SoundNest - feature list

Feature	Key Action	Mouse Action	Notes
ZoomIn	'z'	Optional left click for custom zoom target	Normally self centering
ZoomOut	'x'		
Drag	'1'	Left for outermost/ sub-particle, Right for sub-sub particles and drag	Left on a point which is not a sub-particle moves the outermost particle
AdjustSpeed	'2'	Left for outermost, right for subs. Drag and release to adjust.	
Transform	'3'	Left for outermost/ sub-particle, Right for sub-sub particles	changes to current preset
AddToInner	'4'	Left adds to outermost/ right to sub-particles	adds current preset
AddOuter	space bar		
StepInto	'5'	Left only	
RemoveOuter	'6'	Left sub-particles, Right for sub-sub particles	
Destroy	'7'	Left sub-particles, Right for sub-sub particles	
Clear	'8'	Left sub-particles, Right for sub-sub particles	
Select Single	Shift + 's'	Left sub-particles, Right for sub-sub particles	hold keys and drag for selecting multiple
SelectByType	Shift + 'd'	Left sub-particles, Right for sub-sub particles	
Change Preset	left/right arrow		
Change Bank	'+' , '-'		
Open Preset Editor	Shift + 'P'		

Particles can be created in two ways. A feature called 'addToInner', applies to the outermost particle and its sub-particles, and allows particles to be created in a specific location using the mouse. A complication arising during implementation was the translation of mouse coordinates into world coordinates for sub-particles. The problem was again solved by recursive calling of translation functions. A second feature called 'addOuter' instantiates a new outermost particle. This particle is of a fixed size and contains the old outermost particle as a sub-particle. This happens by resizing the old outermost particle to match the super-particle's geometry and instantiating a body in its world. The new outermost particle is large enough so that its edges are drawn off screen, and the resizing of sub-particle is such that the screen rendering remains unchanged when this feature is called. This means that the viewer remains unaware of this process until the user to zooms out, dramatically revealing the new containing particle and implying that it has been there all along.

Furthermore by zooming out before calling ‘addOuter’ the user is able to control the new sub-particle size of the old outermost particle.

This functional role played by the zoom in and out feature leads to some further unconventionality in the implementation of drawing. Whilst it appears to the user and viewer that zooming in and out translates the camera along the z-axis, the program is actually scaling outermost particle and hence all the sub particles. This prevents the need for complex back calculation in the ‘addOuter’ method and is also reflective of the contradiction between scale and perspective in the visual output of the interface.

There are several methods of destroying particles. ‘Destroy’ applies to sub-particles and sub-sub-particles of the outermost particle to be destroyed through mouse selection, whilst ‘clear’ destroys all sub-particles of a selected particle. In a similar fashion to other functions, both features recursively trigger the deletion of all the sub-particles of any particles to be destroyed. However, a further feature, ‘removeOuter’ allows a particle to be destroyed whilst leaving all its sub-particles intact. This in effect transfers them up one level of recursion. The implementation of this required not only copying the sub-particles to the super-super-particle, but also performing the necessary transformations to their dimensions and forces to match its coordinate geometry. The final method ‘stepInto,’, involves making the selected particle the outermost particle, thus deleting the old outermost particle. The method, which is essentially a retrograde of ‘createOuter’, only works for sub-particles of the outermost particle.

There are three features, each applying to the outermost particle, its sub-particles and sub-sub-particles, and relating to the manipulation of particles once they have been created. ‘Drag’ allows particles to be moved via the mouse pointer. In practice this feature is as a way of increasing particle energy rather than for accurately positioning particles, though when applied to the outermost particle, it is in effect a method of moving the viewport. The ‘adjustSpeed’ feature scales the speed of all sub-particles within the selected particle evenly. Finally ‘transform’ allows a particle’s sonic and physical properties to be changed to a different preset leaving its sub-particles unchanged.

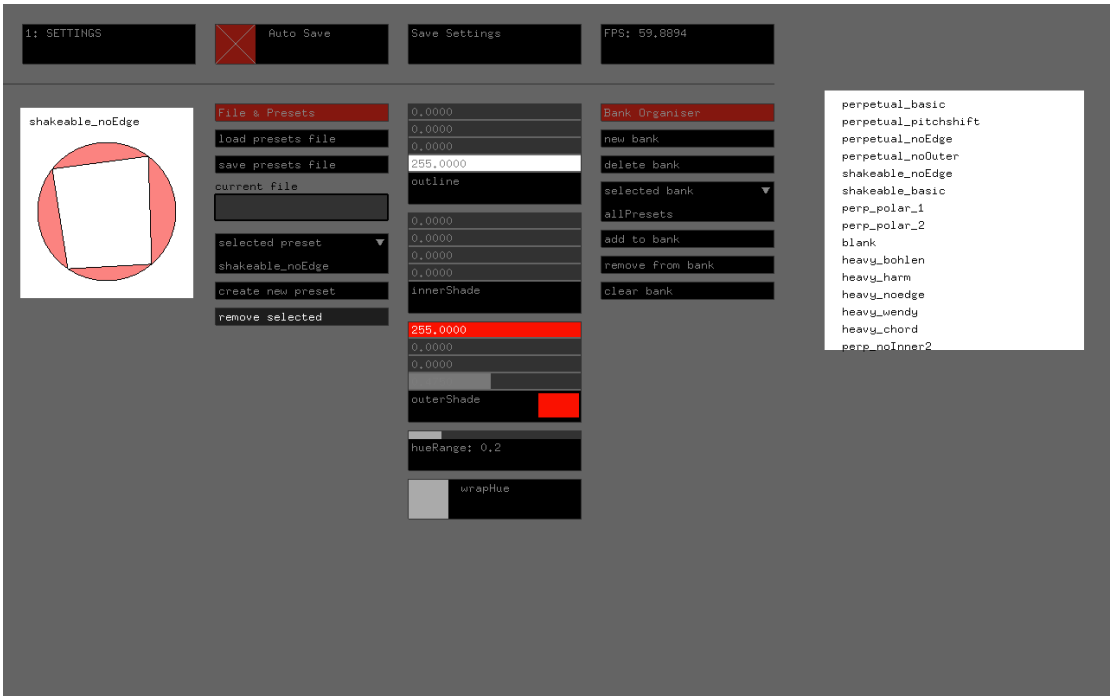
A key part of the interface design was the inclusion of presets for the creation of particles. This allowed a great deal of sophistication and flexibility in the behaviour of particles whilst allowing SoundNest to be usable as a live performance tool. As with *Les Escaliers*

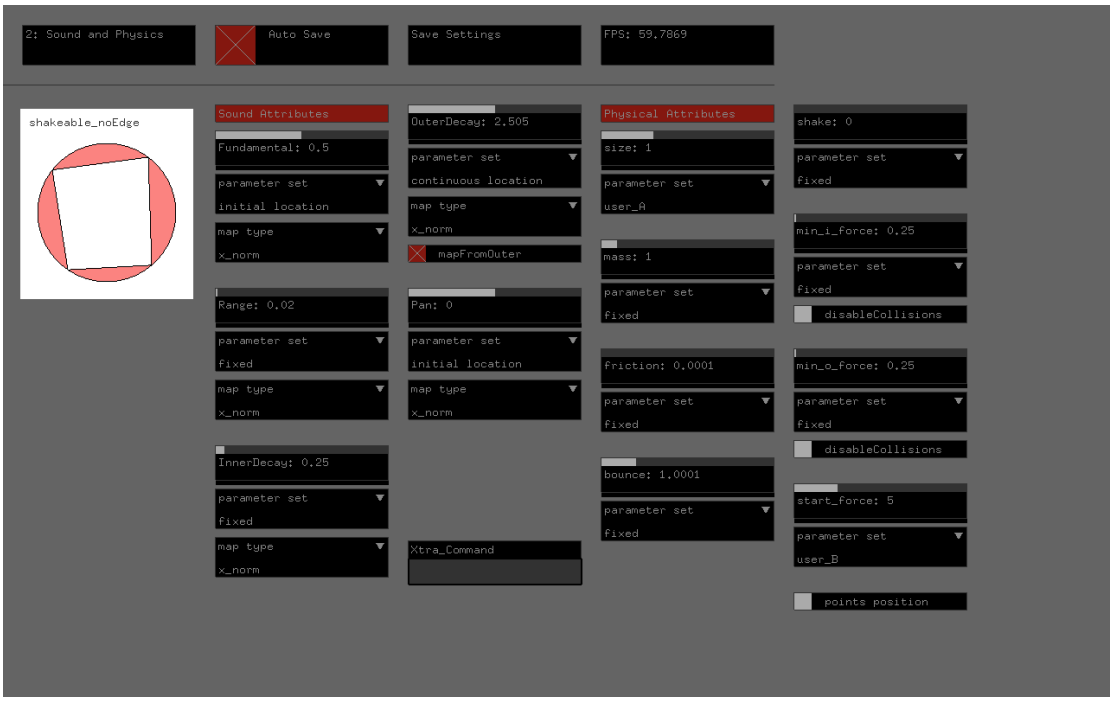
Méchaniques the preset interface was influenced by the design of modern multi-effects units for electric guitars. In this case users can create and save their own presets, determining attributes relating to appearance, sound, and physical display. These presets can then be arranged in banks in order to facilitate easy access during live performance. In addition to this, the preset interface allows a number of ways for the live setting of parameters before and during instantiation. There are two mouse-controlled parameters that are set from the point where the user clicks to create a new particle until they release to confirm the particle creation; 'userA' is a normalised value representing the distance between the point where the user initially clicked and the current mouse location; 'userB' is a normalised value representing the angle between the vector formed by the initial click position and the current mouse position and the y-axis. Attributes can also be set by various mappings, which either by using the initial location where a particle is created, or by continuously updating the attribute based on the current location. The full range attributes and options are shown in Figure 6.7.

Figure 6.7 SoundNest - preset attributes

Attribute	Range	Set Types	Description
innerShade	0 to 1 RGBA	fixed	shading of inner quad
outerShade	0 to 1 RGBA	fixed	non-active shading of outer segments
hueRange	0 to 1	fixed	amount to vary each of the four inner edge colours
fundamental	0 to 1	fixed, userA, userB, initial location, continuous location	fundamental frequency for sub-particle/edge collisions
range	0 to 1	fixed, userA, userB, initial location, continuous location	controls relative pitches of particle edges (dependent on SC implementation)
innerDecay	0 to 4	fixed, userA, userB, initial location, continuous location	decay from inner edges in seconds
outerDecay	0 to 4	fixed, userA, userB, initial location, continuous location, mapFromOuter	decay from outer edges in seconds
pan	-1 to 1	fixed, userA, userB, initial location, continuous location,	pan
size	0.15 to 3	fixed, userA, userB	size of particle
mass	0.1 to 10	fixed, userA, userB	actually the density of the particle
friction	0 to 5	fixed, userA, userB	how much the particle will lose momentum
bounce	0 to 5	fixed, userA, userB	how much momentum the particle will gain when rebounding off of surfaces
shake	0 to 10	fixed, userA, userB	how much force to apply to a particle when its super-particle is hit by another particle
min_i_force	0 to 25	fixed, userA, userB	minimum force required for an inner collision to be registered (also has a disable option)
min_o_force	0 to 25	fixed, userA, userB	minimum force required for an outer collision to be registered (also has a disable option)
start_force	0 to 20	fixed, userA, userB	force added to the particle when it's created

Figure 6.8 SoundNest - preset interface





God Over Djinn commentary:

God Over Djinn is a largely determinate, linear composition that is intended as an exposition of the features of SoundNest. In this regard the form of the composition was very much determined by the logic of the interface. The form and interface commands used to achieve it are set out in a descriptive score (Appendix B and online). The following paragraphs make reference to this in order to comment on some key aspects of the work.

The pitched sounds in *God Over Djinn* are all created from a straightforward sine oscillator. My reasoning behind such a decision was twofold. In the first place, such a simple sound was reminiscent of early computer game sounds and so would match my retro-graphics and pong-like movement of particles. Secondly, sine tones would work well in dense textures created by large numbers of particles, yielding additive synthesis type timbres. In contrast, the edge collisions use SuperCollider's Klank Ugen which, when provided with enharmonic partials, produces sounds of a metallic quality. The combined size of the colliding particles is used to determine a base frequency from which the other partials are calculated, meaning that, as one might expect, collisions between smaller particles produce higher sounds than collisions between larger particles.

This choice of sounds reinforces the contrasting audio-visual relationships implied by the two types of collision. The response to inner edge collisions is somewhat symbolic. There is nothing physical about either the sine tone beep that is heard or the lighting up of the segment. Instead, one might imagine that there is some kind of sensor or button that is being triggered by the collision. Primarily our attention is drawn to linear and discrete relationships such as the one between the resultant pitches and the hues of the lighting segments, or to comparisons between the pitches of hues of different particles. Responses to edge collisions tend towards the opposite end of the spectrum. The metallic sound implies a materiality, to which the variation in pitch according to particle size conforms. This physicality is accompanied by a lack of specificity. Aside from the enharmonic nature of the sound, the use of combined particle size and force as sonic determinants prevents the precise repetition of events. Furthermore, the visual feedback relies entirely on the contrary movement of the collided particles, excluding the conveyance of discrete audio-visual relationships.

A point to note is the different configurations of physical presets. One example is the perpetual configuration, which is used extensively throughout the piece (Figure 6.9). Here the physical parameters are carefully balanced so that a moving particle will not come to rest. One has to be careful when doing this as when the forces of bounce exceed density one can easily end up with positive feedback loops in which particles take on ever more force. A contrasting example is provided by the heavy configuration, examples of which can be found in the latter sections of the piece (Figure 6.10). Here the mass and friction values are set high meaning that there is a poor conservation of momentum and the particles are difficult to move.

Figure 6.9 God Over Djinn - Perpetual physics configuration

Attribute	**Parameter**
Mass/Density	1.0
Friction	0.0001
Bounce	1.0001

Figure 6.10 God Over Djinn - Heavy physics configuration

Attribute	**Parameter**
Mass/Density	5.0
Friction	15.0
Bounce	0.05

Despite the early avoidance of mappings such as vertical axis to pitch, in the conception of the interface, they still play a vital role in the composition. However, in many cases they are primarily used as a practical means for controlling and organising pitch rather than as an explicit formal device. One example is in section A1 where the pitch to y-axis mapping of the 'perpetualBasic' preset allows the creation of a varied pitch texture and an orderly transformation in section A2 to 'perpetualPitchShift' creating a filter sweep like effect. A more dynamic example can be found in sections A4 and A5 where the y-axis pitch mappings reflect the sudden changes in particle geometries.

Nevertheless, where such mappings are used, their combination with other aspects prevents them from becoming tiresome. In the first place, though the fundamental frequency of a particle is determined by its vertical position, the arrangement of its four inner pitches is not. As can be seen by the multiple tunings adopted in section D5, the flexibility offered by

the construction of the OSC system, allows for any kind of arrangement that can be thought of. For example, the 'heavyChord' preset employs two static pitches and two mapped ones.

Mappings are also used to create subtle variations in texture, for example in section D1 where the 'perpetualNoInner 2' preset maps outer decay to x-axis position. A further subtle contrast is that of continuous mapping and initial position mapping. The former being used to create dynamic effects, such as in sections A2 to A5, and the latter being used to create static effects such as in section B.

A final but crucial aspect to highlight is the composition's overarching strategy of revelation as development. At almost every stage in the composition aspects of the interface's capabilities are purposefully withheld in order to allow for paradigmatic expansion. This not only happens through the mechanism of revealing recursions which could easily become tiresome, but also through other parameters. For example, section A initially encourages the audience to use the screen to frame the space but then reveals a larger space, and implies only a single type of movement through static sub-particles but exposes a more complex movement through dragging them across the space. The strategy is still present in section C where the edge collision and shake properties are revealed, and in section D, which reveals that hundreds of particles can exist simultaneously.

Conclusions:

God Over Djinn has been performed numerous times to small and large audiences, at festivals and conferences but also as casual one to one performances given without use of a projector. In public performances I have opted to sit to one side of the screen at a small table with my laptop. The intention is that the audience will initially focus on me but soon shift their gaze to the screen, my role as the real time manipulator of events having been comfortably established.

The enthusiastic response to performances of *God Over Djinn* to some degree confirmed my intuition that visual media could be used to amplify and reinforce music's communications of its formal properties. The key difference between this and previous compositions, however, was the use of a mutually held base of implicit knowledge about the physical world. Many audience members, of all different creeds, spoke of simultaneous experiences of complexity and transparency in viewing the piece. The implicit aspects of the work appear

to operate in a similar way to the reference points historically provided by the tonal system, serving as a base from which audiences can build an understanding of new and complex works in real time.

Nevertheless, this isn't to suggest that every auditory aspect has a visual complement or vice versa. Despite first appearances, the relation between sound and vision is more varied than a simple one to one mapping. For example, the visual hierarchy of nested particles is not necessarily reflected in the audio output. No specific mappings have been made to depth of recursion. Instead the hierarchical arrangement of layers manifests itself in sound through local and sometimes causal relations between layers. For example, when a sub-particle maps pitch to the y-axis of its super-particle or when a sub-particle has a shake value.

In response to possible suggestion that visual might be subservient to audio or vice versa, I would emphatically argue to the contrary – both media are absolutely co-dependent here, clearly poorer without the presence of the other. Even where it has no direct effect on the audio, the visual recursive aspect nonetheless plays a functional role in the work. For the audience it serves to create logical connections between sections of the piece giving a sense of unfolding space, dramatic revelation, and gradual transformation. Likewise the audio is not simply a decorative sonification of the moving image, it is its very justification - extra particles are added in order to thicken the sonic texture and so on. A better view might be that the interface has become the composer and audiences' shared conceptual space.

Undoubtedly, compositional priorities have been shaped by the esotericisms of the interface; for example, its prioritisation of texture over melody or its organisation of pitch into tetra-chords. Though normally problematic for a composer, in the case of a bespoke interface, its tacit influence becomes an advantage. Nevertheless, the point brings into question to what degree SoundNest is valid as a compositional tool separate from *God Over Djinn*. The answer it seems lies in whether the interface is too strongly imbued with the composer's own aesthetic choices to allow others to use it. This will only become apparent when further compositions are attempted with the interface.

It might be that extra features such as the ability to manipulate shapes and graphics for particles might be required. Other features that I can envisage include the integration of tuning systems directly into the interface to allow greater control over pitches during performance and extra behaviours for particles such as the ability to self-replicate or self-

destruct. Another area of potential would be the inclusion of non-colliding layers of particles, implying a depth along the z-axis. Such layering could act as a control for amplitude for inner collisions, which remains a constant in the current version, and would certainly add an extra dimension of visual dissonance between scale and perspective. At the time of writing the software still remains closed. Although it is perfectly functional for the composer, a portable release would require significant extra development and maintenance.

7. Musical Matryoshka

Video and code: http://www.simonkatan.co.uk/phd/matryoshka.html

Overview:

Musical Matryoshka (16.06.2011) is a performance lecture on the topic of recursion and its relation to art and music. The work was in part inspired by my studies of Douglas Hofstadter's Gödel, Escher, Bach (Hofstadter, 1999). Aside from his musings on recursion, Bach and Cage, I was also interested in Hofstadter's use of homeomorphism to link topics ranging from Number theory to ant colonies, with each subject area enhancing the meaning of the others. I was also interested in his novel and holistic approach to the presentation of his material, not only through his use of dialogues which were themselves exemplar of the point under discussion, but also through his innovative use of type-facing and illustration – Hofstadter was a pioneer in this regard producing the book on word-processor and typesetting himself. In homage to Hofstadter, my lecture attempts to emulate these facets through a power point presentation, where not only the script but also the power point presentation itself exemplifies the topic under discussion. The work has been performed twice firstly at the Brunel Researching the Arts conference 2011 and secondly at the ArtistsTalk.eu series in Ljubljana Slovenia. In both cases the audience comprised a combination of musician and non-musician artists and academics.

The conceptual starting point was the interaction between recursion and memory. Hofstadter points this out in a number of ways referring to computer languages, narrative devices, tonal modulations in music and also by creating his own recursive dialogue in the little harmonic labyrinth which also uses typesetting to emphasise the recursive form. I

imagined my own presentation as a deeply nested narrative presented through a series of slides whose spatial arrangement would reflect the narrative hierarchy. At a later point the same technique could be used to demonstrate Hofstadter's notion of mental stacks in tonal music. Above all I wanted to use the spatialisation of slides as a way of allowing the audience to consider how their own memories were operating during the lecture. The challenge of maintaining a mental stack would become analogous to the challenge of navigating a geographical space.

A second aspect I wanted the presentation to highlight was the amorphous relationship between recursion and the defining and redefining of paradigms, extensively commented on in the lecture itself. Initially the presentation should masquerade as a standard PowerPoint, but then gradually reveal its strange nature as the lecture continues. With each revelation, an implicit rule would be broken and a new paradigm implied. Though a simple spatialisation of slides may have been possible using a commercial package, it was clear that the full range of features that the presentation demanded (Figure 7.1) would require the design of a bespoke software package.

As the notion of commenting further on the presentation itself seems somewhat redundant, the rest of this chapter consists of a discussion on the implementation of the PowerPoint interface called Recursor followed by the text of paper.

Figure 7.1 Musical Matryoshka - features occuring in sections

Section	**New Features**
A Self Similar Introduction	- X-axis spatialisation - Horizontal Camera Movement - Voices speak back to the presenter
Defining Recursion	- Y-axis spatialisation - Vertical camera movement - Text floats above slides whilst the camera moves
A Self-Referential Example of Quining	- Writing XML code in XML code
Paradoxes and Metaphors	
This Section Breaks Implicit Rules	
Recursion and Meta-narratives	- Camera Zoom Out - Slide Droste Effect
Recursion in Tonal Music	- Rotation of camera angle - New slides placed relative to camera angle resulting in slides at multiple angles. - Musical extracts with slides timed to music - Floating Captions
Recursion in Tonal Music	- floating image above slides
Deliberately Recursive Music	

Recursive Music Via Computer Animation	- Inclusion of SoundNest interface within a single slide automated functions - Physicalisation of typeface with collision triggered audio -Physicalisation of slides with collision triggered audio - Addition of extra physical objects derived from earlier occuring images
Conclusion	

Recursor Implementation:

In order to make Recursor useable in a presentation, the interface needed to allow the advance programming of sequences of actions and the subsequent stepping through them using a single button. Seeing as my program was only for personal use I opted to use XML as the interface by which I would program the sequences. In the XML interface, sequences consist of stages, which can hold multiple commands of different action types (Figure 7.2). All commands within a stage will be executed simultaneously although the 'delay' action, used as the final command of a stage, can be used to link stages together. The combination of these commands allows the user a great deal of flexibility in how actions are executed, for instance allowing overlapping commands to be executed. Finally, to allow easier editing, multiple XML files can be linked together by using the 'loadXml' action as the final command of the final stage (Figure 7.3).

Figure 7.2 Recursor - XML interface

Action	Parameters	
addSlide	ref	index for reference slide
	pole	(N,S,E,W) direction from reference slide
	offset	distance from reference slide
	con	(0,1) make a connector ?
	title	text for title
addConnector	home	reference to home slide
	target	reference to target slide
	h_pole, t_pole	(N,S,E,W) home pole
	h_off, t_off	home offset
	mode	straight line or corner brace
addBullet	x, y	position on slide
	font, size	
	text	display text
	quine	(0,1) allow use of < and > in the text
addImage	filepath	
	x,y,w,h	position and dimensions
	z_rot	angle degrees
	fade_in	seconds
removeImage	filepath	
	fade_out	
addMovie	filepath	
	x,y,w,h	
	speed	fps
	loop	(0,1) repeat or not
addBrace	x,y,w,h	
	z_rot	
	fade_in	
loadSounds	section	
playSound	section	
	bufnum	
flash	filepath	

	in, hold, out	envelope for flash
pushAmbig	w,h	dimensions of ambig statement
	zoom	distance from slide
popAmbig	slide	slide ref to drop on
	x,y	position to drop
nest	x,y,w,h	for nested image
	z_rot	new camera rotation (nested slide will appear with straight orientation)
	s_mul	speed of zoom
zoomTo	zoom	new camera position
	s_mul	speed (0 – 1)
moveTo	slide	
	mode	1 = straight line, 2 = XY, 3 = YX
	s_mul	
	decel	
	pole, offset	for non-centred final position
god	godcomm	add, select, transform, zoom,
simple	scomm	addWord, split, addForce, addRandomForce, addIndexForce, addSqImage, addSqSld, clear
addFrontWorld	title	name for OSC messaging
clearFrontWorld		
bindToFrontWorld	slide	
	frontworld	index of frontworld to add to
	x,y,w,h	
	fixed	(0,1)
front world	scomm	
loadXml	file path	
delay	time	seconds

Figure 7.3 Recursor - XML stages

```
<RECURSOR>


<STAGE>
<COMM><ACTION>anAction</ACTION></COMM>
<COMM><ACTION>anotherAction</ACTION></COMM>
</STAGE>

//wait for user input

<STAGE>
<COMM><ACTION>anAction</ACTION></COMM>
<COMM><ACTION>delay</ACTION><TIME>1.5</TIME></COMM>
</STAGE>

//next stage happens automatically after 1.5 seconds

<STAGE>
<COMM><ACTION>anAction</ACTION></COMM>
<COMM><ACTION>loadXml</ACTION><FILEPATH>xmls/theNextFile.xml</FILEPATH></COMM>
</STAGE>

</RECURSOR>
```

With the exception of the initial slide, which is created in the centre of the screen, slides are created with a position relative to a reference slide. Extensive use is made of defaults

throughout the interface to make the XML code more concise. For example, a new slide added with no parameters set will be added to the east of the current slide with a straight connector. Visible objects such as images, movies, and bullet points can only be added to the currently visible slide. Objects' positions are set as an absolute value relative to the top left hand corner of the screen. The slide class holds various functions including drawing functions and those relating to adding and removing objects.

Slides are viewed through a camera, which moves through the 'moveTo' action, gliding from one slide to the next with variable speeds and can also zoom in and out with the 'zoomTo' action. For both of these, the movement is enveloped to make a smooth transition, and a parameter called 's_mul' is used to adjust the speed as a proportion of the default. Though these functions are situated in the testApp, it is the slides themselves who determine whether they are visible and hence need to be drawn. Figure 7.4 shows a small sequence from the final presentation where slides and bullet points are added. Note that the 'moveTo' command must be called before adding the objects. However, as all of the commands are executed simultaneously, the viewer sees the slide appear with the objects already on it. Sound samples are grouped into sections in SuperCollider and can played back simply by calling the correct index.

Figure 7.4 Recursor - Adding of new slides and bullet points

```
<STAGE>

    <COMM><ACTION>loadSounds</ACTION><SECTION>0</SECTION></COMM>

    <COMM>
    <ACTION>addSlide</ACTION><BG_COLOR>0xFFFFFF</BG_COLOR><FG_COLOR>0x333333</FG_COLOR>
    <TITLE>Postgraduate Seminar Series 02.11.2011</TITLE>
    </COMM>

    <COMM>
    <ACTION>addBullet</ACTION><X>20</X><Y>130</Y><SIZE>24</SIZE><FADE_IN>1</FADE_IN>
    <TEXT>Musical Matryoshka: Recursive structures through sonic and visual media</TEXT>
    </COMM>
</STAGE>

  <STAGE><COMM>
    <ACTION>addBullet</ACTION><X>40</X><Y>250</Y><SIZE>20</SIZE><FADE_IN>1</FADE_IN>
    <TEXT>How can we express recursive forms in music ?</TEXT>
  </COMM></STAGE>


  //2010

<STAGE>
    <COMM>
    <ACTION>addSlide</ACTION><BG_COLOR>0xFFFFFF</BG_COLOR><FG_COLOR>0x333333</FG_COLOR>
    <TITLE>ArtistTalk.eu  15.10.2011</TITLE>
    </COMM>

    <COMM>
    <ACTION>moveTo</ACTION><SLIDE>1</SLIDE>
    </COMM>

    <COMM>
    <ACTION>addBullet</ACTION><X>20</X><Y>130</Y><SIZE>24</SIZE><FADE_IN>0</FADE_IN>
    <TEXT>Musical Matryoshka: Recursive structures through sonic and visual media</TEXT>
    </COMM>


    <COMM>
    <ACTION>addBullet</ACTION><X>40</X><Y>250</Y><SIZE>20</SIZE><FADE_IN>0</FADE_IN>
    <TEXT>How can we express recursive forms in music ?</TEXT>
    </COMM>
</STAGE>
```

There are two different features involving floating static objects over the slides. The more simple version called ‘flash’, is used in the ‘Recursion in Tonal Music’ section, and simply draws an image on top of the slides for a given period of time. ‘PushAmbig’ and ‘popAmbig’, as used in the ‘Defining Recursion’ section, are a little more complex. These methods are intended for when a statement can’t be attributed to any of the characters within the narrative. With ‘pushAmbig’ the text is drawn above the slides on a small slide in the centre of the screen, the rest of the slides are shaded darker. Once the source of the statement is attributed through the narrative, the user can call ‘moveTo’ to go to the correct slide, followed by ‘popAmbig’ which returns the shade to normal and drops the small slide onto the current slide.

One area to note is the writing of XML code in XML, as it occurs in the ‘Self-referential example of Quining’ section. Ironically, unlike most languages that manipulate strings, the XML language doesn’t allow the use of ‘<’ and ‘>’ characters as text and instead attempts to

read them as code. In other words, the XML language is incapable of Quining. The solution required a simple work-around was applied with some degree of remorse about the deception, whereby '£' and '$' were used to represent '<' and '>' in the XML file.

Though quite dramatic, the application of the Droste effect used at the end of 'Recursion and Meta-narratives' was relatively straightforward to implement. The nest action simply causes the calling of a function within the referenced slide, which takes a screen shot and then resizes and adds that image to the slide. The illusion of nesting is created, firstly by centering the camera on the image and setting the zoom so that it is same size as the screen, and then subsequently zooming out to imply that what was captured by the screen shot was in fact contained within the slide. The 'z_rot' parameter not only causes the rotation of the image on slide, but also causes the rotation of the camera such that image is still shown as normally oriented. Now all the slides appear to be at an angle, but any new slides will be added at the new orientation and so will appear straight. At the end of 'Recursion and Meta-narratives' and start of 'Recursion in Tonal Music', this feature serves as a metaphorical reflection of the consequential change in perspective as a result of the fourth wall being broken.

'Recursive Music via Visual Animation' required the entire inclusion of SoundNest within Recursor's code. The automation of its functions required a wrapper class in Recursor simply called 'god'. Commands intended for god in the XML file are then directed using the 'god' action, followed by a 'godComm'. The same SoundNest OSC responder nodes are loaded in SuperCollider to produce sound. In order that every slide doesn't unnecessarily waste CPU by running SoundNest, a 'setup' 'godComm' is required to initialise SoundNest.

Figure 7.5 Recursor - use of SoundNest

```
41
42 <STAGE>
43 <COMM><ACTION>god</ACTION><GODCOMM>setup</GODCOMM></COMM>
44
45     <COMM><ACTION>god</ACTION><GODCOMM>add</GODCOMM>
46     <PRESET>14</PRESET><X>0</X><Y>0</Y><A>0.2</A><B>0</B></COMM>
47
48     //add inner
49  <COMM><ACTION>god</ACTION><GODCOMM>select</GODCOMM>
50     <MODE>0</MODE><LEVEL>1</LEVEL><INDEX>0</INDEX></COMM>
51
52 <COMM><ACTION>god</ACTION><GODCOMM>add</GODCOMM>
53     <PRESET>0</PRESET><X>0</X><Y>0</Y><A>0.25</A><B>1</B></COMM>
54 </STAGE>
55
56
57 //transform to sound
58 <STAGE>
59     <COMM><ACTION>god</ACTION><GODCOMM>transform</GODCOMM>
60     <PRESET>15</PRESET><A>0.25</A><B>3</B></COMM>
61 </STAGE>
62
63 //transform to discrete pitch
64 <STAGE>
65     <COMM><ACTION>god</ACTION><GODCOMM>transform</GODCOMM>
66         <PRESET>0</PRESET><A>0.25</A><B>3</B></COMM>
67 </STAGE>
68
```

The physicalisation of typeface, as seen in the same section, is called by the 'addWord' 'scomm'. The implementation was significantly more involved and required a new class called simpleWorld. This is a wrapper around a single Box2D world, constrained by adjustable edges. CompoundShapes, consisting of bodies with multiple triangular fixtures, can be inserted into the world. The addWordToWorld method in the simpleInterface class uses the character contours supplied by ofTrueTypeFonts to perform a Delunay Triangulation. The resultant triangle vertices are then supplied to the new compoundShape and the create method called. This creates a series of polygon shaped fixtures in Box2D whose combined contour approximately matches that of the word, which is rendered on top. Another method allows forces to be applied, moving the contour shapes around the world, and a final method allows the splitting of compound shapes into substrings. Here the concerned body is destroyed and the list of vertices in the compound shape used to create two new compound shapes with corresponding bodies.

In order for sound to result from collisions, the simpleWorld class also requires listener and messenger classes. The simpleWorld's listener is inherited from the same base class as SoundNest's listener and uses the same method of a continually updated list of collision objects. However, much of the filtering from SoundNest is omitted in this case. The simpleMessenger uses data from the collision objects to send OSC messages to

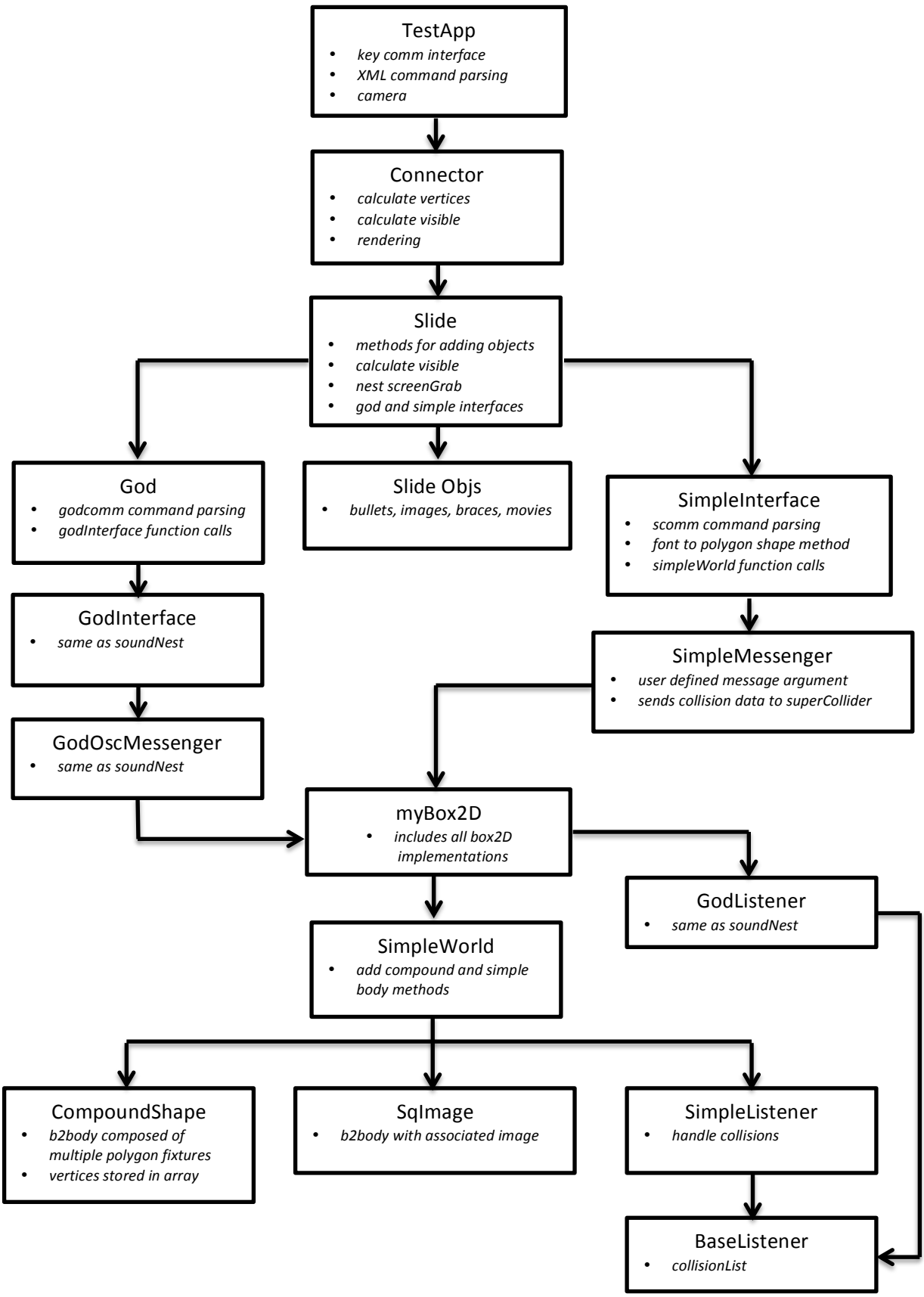
SuperCollider with arguments of the two object types, normalised coordinates and the impulse. In SuperCollider the messages from the simpleMessenger are identified via a dedicated message address. Object types are used to select the correct pre-recorded samples of the words to be heard, and the normalised coordinates are mapped to pan and pitch shift, and the impulse to amplitude.

The physicalisation of the slides and other objects is similarly achieved by using the simpleWorld classes. However, in this case, multiple simpleInterfaces are encapsulated in an array of frontWorlds. A dedicated message address for SuperCollider is set by the title parameter in the XML files, thus allowing the triggering of unique sounds for each of the frontWorlds. The presence of frontWorlds prevents the rendering of all other objects such as slides and connectors. However, slides can be bound to front world objects, allowing them to be rendered to an offscreen texture and subsequently drawn onto moving bodies. This method is used once in the presentation when towards the end of 'Recursive Music via Computer Animation', the slide in view begins to drift off the screen. This is done through the careful setting of the front world object size and scaling to exactly match the scale and position of the previously rendered slide. The SimpleWorld class allows not only the addition of compoundShapes to form words but also more simple ones such as rectangles using addSqImage. This technique is used to render the rest of the moving slides as by the time they appear they are much smaller than normal, and the rendering from the previous method would slow the frame rate significantly. The same method is used to add extra frontWorlds for the rest of the section referencing sounds and images from earlier in the presentation.

Figure 7.6 Recursor - adding of front world

```
<STAGE>

    <COMM><ACTION>addFrontWorld</ACTION><TITLE>slides</TITLE></COMM>

    <COMM>
    <ACTION>frontWorld</ACTION><INDEX>0</INDEX><SCOMM>scaleTo</SCOMM>
    <SCALE>800</SCALE><S_MUL>0</S_MUL>
    </COMM>


    <COMM>
    <ACTION>bindToFrontWorld</ACTION>
    <SLIDE>28</SLIDE>
    <FRONTWORLD>0</FRONTWORLD>
    <X>640</X>
    <Y>400</Y>
    <W>51.2</W><H>32</H>
    <FIXED>0</FIXED>
    </COMM>

    <COMM><ACTION>delay</ACTION><TIME>20</TIME></COMM>
</STAGE>
```

Figure 7.7 Recursor - classStructure



Musical Matryoshka Text:

Abstract:

This is an abstract, which is intended to describe the paper that I shall be presenting at the Researching the Arts conference in a few weeks time on the topic of recursion in music. Actually, that's not quite correct - I meant to say that this is a sentence intended to describe that this is an abstract, which is intended to describe the paper that I shall be presenting at the Researching the Arts conference in a few weeks time on the topic of recursion in music. To be more precise, the this from the previous sentence which appears to refer to the sentence in which it's contained is actually intended to refer to the sentence preceding that sentence in order to clarify what this refers to in that sentence. In anticipation that this and my attempts to define it may have already created some confusion in you the reader, I shall put to one side its definition for the time being and instead attempt define what this is about. This is an abstract, which is intended to describe the paper that I shall be presenting at the Researching the Arts conference in a few weeks time on the topic of recursion in music.

A Self-similar introduction

Today I'd like to speculate on possibilities of expressing recursive forms in music, but in the first place I'd like to explain a little about recursion as it's a somewhat nebulous term. Actually, I was presenting this very same paper at this very same conference last year when, at this exact point, a member of the audience whom to save embarrassment I'll call H, rudely interrupted "So just what is recursion?" "That's strange," I replied, "actually I was presenting this very same paper at this very same conference last year when, at this exact point, a member of the audience whom to save embarrassment I'll call O, rudely interrupted 'So just what is recursion?' 'What a coincidence!' I replied, 'actually I was presenting this paper at this very same conference last year when, at this exact point, a member of the audience whom to save embarrassment I'll call F, rudely interrupted "So just what is recursion?" "How fortuitous!" I replied, "actually, I was presenting this paper at this very same conference last year when at this exact point, a member of the audience whom to save embarrassment I'll call S, rudely interrupted 'So just what is recursion?' 'Well it's funny you should say that,' I replied, 'actually I was presenting this paper at this very same

conference last year when, at this exact point, a member of the audience whom to save embarrassment I'll call T, rudely interrupted, "So just what is recursion? " ' " ' "

Defining Recursion

"By now you're probably wondering if this will ever end," I said to F who sheepishly nodded his head wishing that he'd never asked the question in the first place. "Don't worry" I reassured him, "Recursive forms aren't necessarily infinite. For example, I named this talk after Matryoshka and, although they have a recursive physical form in that they constitute a set of hierarchically nested, self-similar instances…" "Oh it's like your recursive story but with a spatial instead of temporal ordering," interrupted H. "Yes, exactly, " I replied.

"Anyway, where was I? Oh yes, although they have a recursive physical form in that they constitute a set of hierarchically nested, self-similar instances…" "Oh it's like your recursive story but with a spatial instead of temporal ordering," interrupted O who by virtue of being one nesting down was unaware that the point had already been covered by H. "Sorry, if I'd known that O was going to mention it, I wouldn't have brought it up," said H. "That's okay," I replied.

"Anyway, where was I? Oh yes, although they have a recursive physical form in that they constitute a set of hierarchically nested, self-similar instances, the dolls don't actually get infinitely small. Incidentally, the procedure by which the dolls are revealed can also be defined recursively by including a stage which calls new instance of itself in order to reveal the next doll."

A Self-referential example of Quining

"But it's always so disappointing when you get to the last doll," said F. "Somehow all that self-similarity seems to imply that I should be able to reveal an infinite number of dolls. Aren't there any infinitely recursive processes?" "Well, you've changed your tune," I replied. "Actually there's many, but I have a very good example for the purposes of this lecture. You might have noticed by now that this PowerPoint presentation isn't exactly normal. In fact in order to achieve the particular effects that I want, I've had to create my own version of PowerPoint." "Ooh that's very clever," said O. "Yes I agree, very clever," said H. "Thanks, although my PowerPoint isn't as easy to use. In order to add words and images, I have to type instructions in code. So to add the statement 'recursion = infinite?' on F's slide …" "so

this is my slide?" "Yes it's the 4th of the total set of slides and represents when I presented this paper to you at the conference of 2009." "But it was the first slide back then?" "Yes that's right." "Okay just checking."

I continued. "Anyway, to add the statement 'recursion = infinite?' on F's slide, I have to write the following code.

<ACTION>addBullet</ACTION><TEXT>recursion = infinite ?</TEXT>

The action bit tells the program that I'm adding a bullet point and the text bit says what words the bullet point has to contain. Now in order to add the bullet point I've just written, I had to use the following piece of code.

<ACTION>addBullet</ACTION>
<TEXT> <ACTION>addBullet</ACTION><TEXT> recursion = infinite ? </TEXT>
</TEXT>

Now a section of code itself has been quoted inside a new piece of code. Of course in order to add that bullet point, I had to write the following piece of code.

<ACTION>addBullet</ACTION><TEXT>
<ACTION>addBullet</ACTION><TEXT>
<ACTION>addBullet</ACTION><TEXT>
recursion = infinite ?
</TEXT>
</TEXT>
</TEXT>

Now a section of code quoting another piece of code is quoted within a new piece of code. One can see how this process might continue producing ever-longer lines of code."

Paradoxes and Metaphors

"Okay I can see how that works," said F, "but something else strikes me about your example." "Oh what's that?" I replied. "Well there seems to be a paradoxical situation in which the harder you try to describe the code within the bullet points, the further you appear to get from doing so." "Yes that's why I like it," I replied. "One could understand it as a comment on the allusiveness of objectivity. Each time we try to reach the core of something we find yet another layer beneath." "Or conversely, each time we try to get

outside of something, in order to get a proper look at it, we discover we've simply entered a further outer world."

All this reminds me of that M.C. Escher print of the Dragon trying to eat its own tail." "Don't you mean the ancient Greek symbol, Ouroboros which represents eternal return?" "Well it's similar but, according to Escher, this dragon, despite being a two dimensional depiction, stubbornly tries to be in three dimensions but is doomed to failure with every attempt." (Hofstadter, 1999, pp. 473-474) "A fitting metaphor!" "Actually, it's mentioned in Douglas Hofstadter's, 'Gödel, Escher, Bach' which contains many other fascinating examples of recursive situations. One that comes to mind is a quotation of Oxford philosopher J.R.Lucas on the subject of consciousness. Lucas says, "In saying that a conscious being knows something, we are saying, not only that he knows it, but that he, knows that he knows it, and that he knows that he knows that he knows it and so on (Hofstadter, 1999, pp. 388-389). However, the main focus of Gödel Escher Bach is the mathematician Kurt Gödel, who managed the paradoxical feat of deriving a theorem of number theory claiming not to be a theorem of number theory - the equivalent of making the statement 'This statement is not true.' " "Didn't that destroy number theorem?" "Actually, no, it proved that number theorem was incomplete which lead to a whole new class of numbers called Supernatural Numbers, which were larger than infinity." "So Gödel managed to get number theorem to reference itself and thereby discovered a larger world in which number theorem was nested." "Precisely." (Hofstadter, 1999, pp. 438-460).

This Section Breaks Implicit Rules

"Wow, I had no idea that recursion was such a profound topic," said F. "I'm glad to have had this conversation. I feel sorry for S and T - they really missed out." "Actually, they didn't" "How is that possible? " "I had exactly the same conversation with them." "What! Even this bit?" "Yes even this bit." "I don't believe you." "It's true." "Alright then, tell me what T says next in the conversation, if you're telling the truth then it should be what I say next in this conversation." "Well, at this exact point in the conversation T interrupted me with 'Hang on, if you were telling A an anecdote last year about D at the previous year's conference and now your telling me an anecdote this year about A at last year's conference, then if you present this paper next year…' at which point S interrupted me with 'Hang on, if you were telling T an anecdote last year about A at the previous year's conference and now your telling me an anecdote this year about T at last year's conference, then if you present this

paper next year…' at which point F interrupted me with 'Hang on, if you were telling S an anecdote last year about T at the previous year's conference and now your telling me an anecdote this year about S at last year's conference, then if you present this paper next year…' "

"Ha so you were telling the truth" said O, "I'd never be so stupid …" "What an idiot!" interrupted H as I was telling him the story. "Why can't O work it out like the others?" "Oi!" replied O. "It's not so easy to realise that you're part of a nested structure from within your own layer. " "Now stop right there," complained H. "O is breaking the rules. He can't be aware of what I'm saying because he's part of the anecdote that you're…" "I can hear what you're saying," interrupted O. "Stop it at once!" H called back. "Well, you're breaking rules now too, as you shouldn't be able to directly address me either," O retorted.

Recursion and Metanarratives

I tried to calm them both down. "Listen it's not your fault," I said, "you're only breaking the rules as a result of me breaking the fourth wall." "Why would you do that ?" they replied. "In the first place, I wanted to draw an analogue between Gödelian self-reference which breaks number theory and dramaturgical self-reference which implicitly breaks the fourth wall. I could make a further comparison with Lyotard's characterisation of the postmodern condition as skepticism towards metanarratives (Lyotard, 1984), and the resultant suspicion that such a characterisation might itself be a metanarrative. As Escher's picture seems to indicate, recursion is intimately entwined with our attempts to escape the paradigms within which we operate. However, aside from all this, I also wanted to share the truth with the audience. The truth is that you are all fictional characters in my paper."

Up until this moment, this whole presentation has been designed with the aim of highlighting the inherently recursive nature of language and the relative ease with which our subconscious is able parse its hierarchical constructions. Naturally when a certain degree of complexity is reached things can get a little confusing, but this is all part of the fun. Personally, I find that the haziness that arises from not being able to quite keep track of what level is being referred to, or the sudden realisation that the layer you thought to be the outermost is actually contained within yet another layer to have a somewhat fantastical effect.

Recursion in Tonal Music

musical excerpt: Bach *Prelude No.15 G major BWV 860*

So given all this, it seems inevitable that the topic should be of interest to artists and indeed musicians. Hofstadter claims that we hear music recursively (Hofstadter, 1999, p. 129). Borrowing push, pop terminology from computer science, he describes the way in which we maintain a mental stack of keys in a piece of tonal music. Modulating away from the home key is the same as pushing. In computing this means suspend the current task and take up a new lower level task. In both computing and tonal music, multiple consecutive pushes can happen in a row before a pop occurs. A pop means end the current task and resume a higher-level task from where you last left it. The stack is where a computer or human stores the details of all the key changes. So the more consecutive pushes in a row, the deeper the stack. A deeper stack means our brain has to work harder to remember where it all began thus resulting in a greater degree of tension and subsequent release when we finally manage to get back to the tonic.

As attractive as this model may seem, I'd add a couple of notes of caution. In the first place, unlike computer language function calls, tonal music doesn't always pop via the same route that it pushed. Secondly, as Hofstadter himself admits and the work of experimental psychologists such as Tillmann and Bigand confirms (Tillmann & Bigand, 2004, p. 218), when it comes to music our mental stacks aren't actually very good (Hofstadter, 1999, p. 130). Nevertheless, there still remains a strong degree of hierarchical organisation in our cognition at the micro level, and, at a macro level, I would also argue that the cultural legacy of the Western Classical tradition ensures that even if we aren't correctly perceiving tonal recapitulations, we believe that we are and that this is often enough for the purposes of the composer.

Deliberately Recursive Music

musical excerpt: Tom Johnson *Rational Melodies: XV* (1982)

In any case, I'm more concerned with the conscious expression of recursion in music. A contemporary example would be the self-replicating loops of Tom Johnson. The principal is simple though the math involved in their construction is far from trivial. When certain notes in a specially calculated looping melody are picked out, they produce the same phrase but at

a slower tempo. In theory such processes could be repeated ad-infinitum though in practice, just as physical bounds limit the number of Matryoshka, so our perceptual bounds limit the number of concurrent self-similar melodies.

musical excerpt: Per Nørgård *Voyage Into the Golden Screen: 2nd Movement* (1968).

Per Nørgård's work with his infinity series presents a more developed example of recursive music. The series exhibits similar properties of self-similarity to Johnson's loops though the recursive means by which it's constructed yields an infinite string of notes, never quite repeating themselves. However, despite the mathematical uniqueness of Nørgård's series and the inventiveness with which he employs it, I still find that this music only gives a vague sense that something recursive might be happening.

Whilst Johnson's and Nørgård's music is most certainly recursive on paper, both feel a long way from expressing the paradoxical and profound world of recursion that I have been talking about. This is puzzling, as recursion seems ideally suited for expression in music. The problem seems to be a lack of sufficiently audible or familiar structures within contemporary musical language that are capable of constructing hierarchies on which recursive forms rely. To some degree this is a result of modernisms' and experimentalisms' efforts at negating pre-existent musical structures. The question then is how can composers achieve the hierarchies necessary to express recursive forms without resorting to the historical devices which we spent so much time avoiding.

Recursive Music Via Computer Animation

One possible solution is to have music use the formal properties of another medium. I've been doing this recently with real time computer animation, in my composition *God Over Djinn.* A key tool of the animated world is its ability to tap into our implicit knowledge of the immediate environment for use towards its own ends. For example, we implicitly understand that an object can be contained by another object. But we also know that when that inner object collides with the edge of the outer object it should make a sound. We're also pretty good at simple mappings. We can all understand that each edge of the containing object should produce a different pitch, like buttons on a console, and that each object might have its own discrete set of pitches. It's only a small leap of the imagination to understand that different types of collision, for example, a collision between the outer edges

of two objects would produce a different type of sound as if the edges were made of different materials. This is ample knowledge to produce a pretty complex texture that one could argue is universally understandable.

However, seeing as also we're perfectly happy with notions of scaling and perspective we can go a stage further. As we would expect, each containing world conforms to the rules of the inner worlds. At this point some interesting implications emerge. In the first place despite the obvious visual hierarchical relationship between the innermost and outermost objects, there's no consequent aural relationship. We think we're observing a one to one mapping but we're not. Of course I can introduce some causal relations between these nested objects which, depending on my purposes, I can make more or less obtuse. The other interesting thing is that there is now a visual paradox reminiscent of Escher. Are the objects smaller or further away? And what does it mean when I do this?

Most importantly the secure foundation of implicit knowledge upon which the composition is based seems to afford me a large number of hierarchical nestings without being limited by the necessity of aurally defining each layer. Furthermore, this visual world seems offer endless opportunities for surprising the audience by extending the scope of the composition exploiting our curious habit of framing whatever is in view.

Conclusion

"Ooh that was pretty!" "Ah F so you've come back. I take it you've finished sulking about not existing." "I've come to terms with it. Anyway, as I was saying, I liked all that visual recursive music but aren't you cheating a little?" "How so?" "You're letting the animation do all the work of creating the recursion instead of the sound. Wouldn't real recursive music be able to communicate its recursion solely through sound?" "Well, I think that depends on whether you regard music as an exclusively sonic medium. I would argue that the predominance of notation within the Western classical tradition and the ubiquity of conceptual frameworks based on physical schema throughout musical cultures indicates to the contrary." "Alright I'll leave that to one side. But still I'm curious is it possible express recursion in music through sound alone?"

"I'm not sure, but I could imagine various possible approaches. For example, we could exploit music's ability to construct what Bob Snyder refers to as secondary meta-parameters

(Snyder, 2000). Another way might be through building up musical along grammatical lines, thereby utilising languages' ability to recursively expand sentences indefinitely. In any case, I would contend that for the rich semantic world that recursion offers to really be communicated through music, music needs to not only create recursive structures within its own forms but also seek to engender the same cognitive effects that the presentation of recursion has in other mediums. A recursive music would need to entwine its listeners in a labyrinth of paradoxical self-references, playfully engage them into constructing untenable mental stacks and continually surprise them through the constant expansion or deepening of frames of reference."

8. DarkStar

Video and code: http://www.simonkatan.co.uk/phd/darkstar.html

Overview:

DarkStar (17.10.2011) is an installation exploring relationships between the permanent and the transient, and is a combination of public monument, interactive artwork and virtual gallery. The piece was commissioned by Martin Bricelj whilst I was artist in residence at Mota in Ljubljana, Slovenia, and premiered at Sonica Festival 2011. It takes the physical form of a large sphere, raised above the heads of its viewers with constellations of stars gently floating across its surface. Through the glowing of its stars in sequence to form a band of light rotating around the sphere once a minute, with the width of the band reflecting the current lunar phase, *DarkStar* functions as an esoteric, cosmically concerned, public clock.

However, onlookers in the vicinity of *DarkStar* are also able to interact with it by pointing towards an individual star. Initially they will notice that the star that they pointed at is no longer moving with the others. If they hold their pointing pose for longer, the star will begin to grow, revealing its individual character via real time generative sound and graphics. Gradually the star's sound and graphic will come to dominate *DarkStar*, though when there are multiple users, the stars will compete for dominance. As soon as a user drops their pointing pose the star will return to its normal state.

Currently *DarkStar* hosts five types of stars with contrasting sonic and visual qualities. However, it is my intention to add more stars to this collection with each showing, the ultimate goal being for every star to have a unique sound and graphic. With hundreds of stars and the possibility of their inter-combination, onlookers will find a seemingly limitless world to be explored. Furthermore, the existence of thriving communities of artists around the popular open source libraries upon which the software is based, offers the potential of a rolling 'open call for stars' thus turning the installation into a form of roaming virtual gallery.

Implementation:

The brief

This was a particularly challenging commission in a number of respects. Firstly, the commissioning body already had a concept for the work, which although rather confused, still had a number of fixed parameters. The piece was envisioned as a prototype for a larger public monument which appears as a mirrored geodesic sphere during the day, but through back projection comes alive with stars at night. Aside from the aforementioned time keeping functions of the installation, the commissioning body wanted the arrangement of the stars to be somehow be reflective of sociological data, thus providing continual variation throughout the month. However, they also wanted the installation to have an interactive element with instantaneous feedback. The simultaneous use of a variety of methods such as GPS, machine listening, and iPhone interaction were suggested. The final challenge of the commission was to produce all of this work within the six weeks of my residency, and despite my advice to contrary, I would be the sole artist working on the project taking responsibility for graphics, sound, and interaction.

Figure 8.1 DarkStar - concept drawing

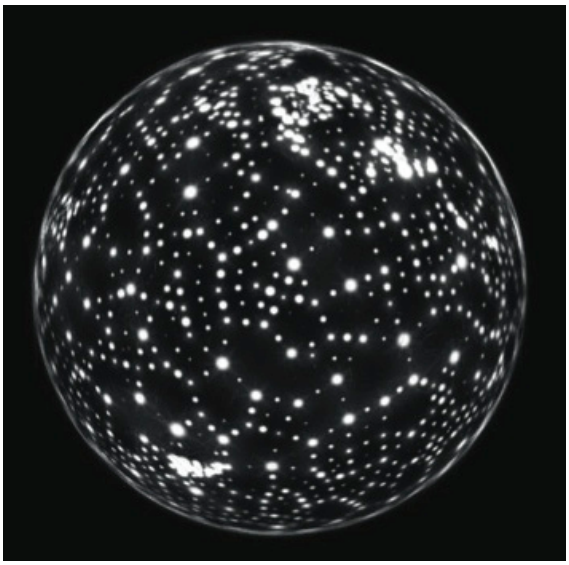


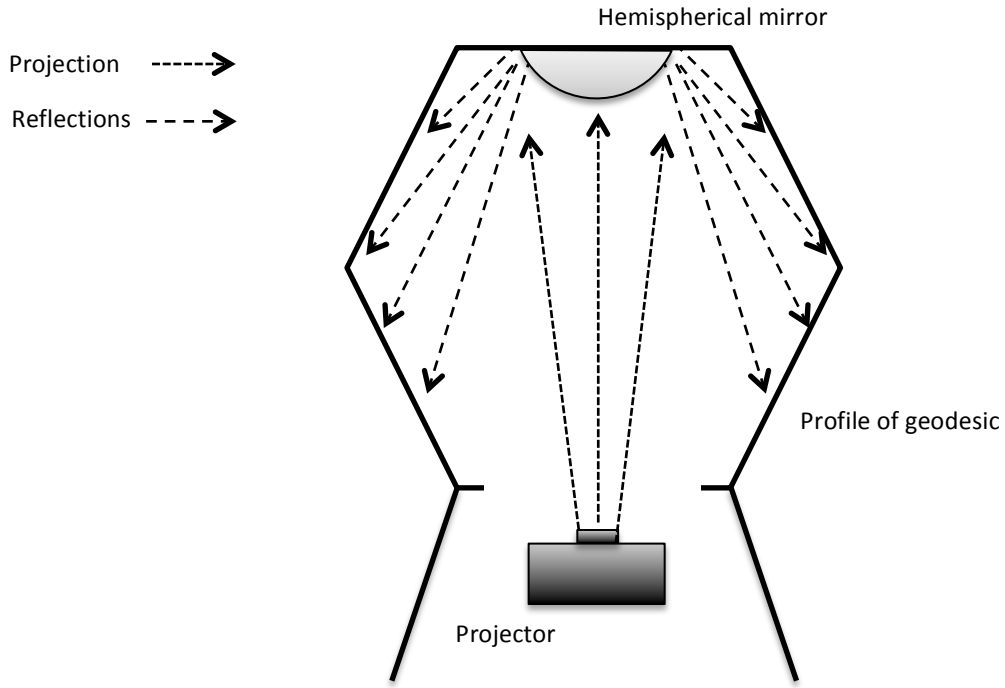
Figure 8.2 Darkstar - concept description

In view of the unfeasibility of time constraints I sought to reduce the scope of the brief. In the first place, I found the dual aims of representing sociological data and instantaneous feedback through the same media potentially contradictory. Though a solution was undoubtedly possible, given that the installation was only being shown for a two week period, and that most viewers would only visit once, I decided that the interactive element should take precedence. I was clear that I wanted the aural and visual feedback from individual user's interactions to be directly discernable. I envisaged two possible ways by which this could happen; users could either create and destroy connections between the stars, or bring different stars into focus. I also wanted to reduce the modes of interaction to a single medium. I considered that both versions might work with the audience interfacing through a mobile app via a wireless router, but that the latter would also be possible through gesture recognition with computer vision. Regarding the intention to upscale the project for a much larger sphere and many more users, my preferred option was to use the mobile interface. However, given the time required for an App Store release, the app would be have to be for the Android platform only. With this in mind, the commissioning body

preferred the universal accessibility of the computer vision interface as is described in the opening paragraphs.

A further problematic area was the physical construction of the sphere and back projection onto its surface. Rather than using a ready-made solution such as Pufferfish's PufferSphere, the commissioning body wanted to build their own geodesic dome and use a projection method similar to Paul Bourke's method for hemispherical domes (Bourke, 2005).

Figure 8.3 DarkStar - proposed projection method

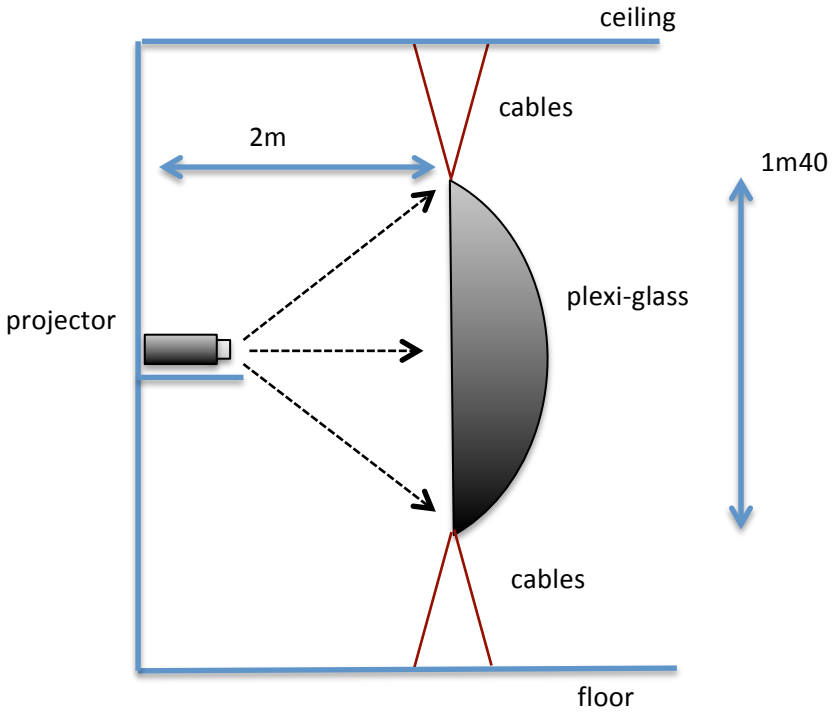


However, though the organisation had separately commissioned a design and purchased a mirror, many aspects had not been considered. As Paul Bourge's article suggests, type of projector, aspect ratio, size of sphere in relation to the mirror all need to be selected carefully before proceeding. However, in discussion with the commissioning body it was clear that this had not been done. They planned on using their own projectors, which were underpowered in terms of lumens, had too low a native resolution, and were LCD resulting in wide gaps between pixels and a poor contrast ratio. The projectors had not been tested to see whether they could focus at the necessary size for the mirror. Furthermore, I had no previous experience of projection mapping and at this stage it was unclear whether I would get any help with the implementation. However, perhaps the most significant problem was

that six weeks away from the launch of the project, the commissioning body had neither drawn up accurate plans for the sphere nor sourced a manufacturer.

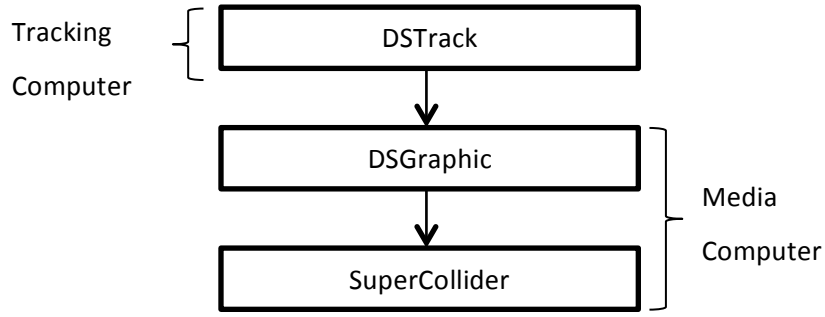
Thankfully after a meeting with a local designer, the commissioning body were convinced to opt for an alternative design. This involved back projection onto a suspended hemispherical surface. Rather than employ a complex geodesic construction involving the manufacture of a system of connecting struts, this version would use a vacuum moulded, opaque, plexi-glass surface which they were able to easily and cheaply source this from a local manufacturer. The removal of the mirror significantly reduced the complexity of the projection mapping although the short distance between the hemisphere and projector as it was to hang in the gallery space now required a short throw lens.

Figure 8.4 DarkStar – final projection method



Knowing that I needed more advanced graphics than in previous interaction projects and wanting graphics and interaction to run on different threads, I opted for dedicated tracking and graphics programs, DSTrack and DSGraphics, on dedicated machines with communication via OSC. As previously these application were realised in OpenFrameworks and sound was realised in SuperCollider.

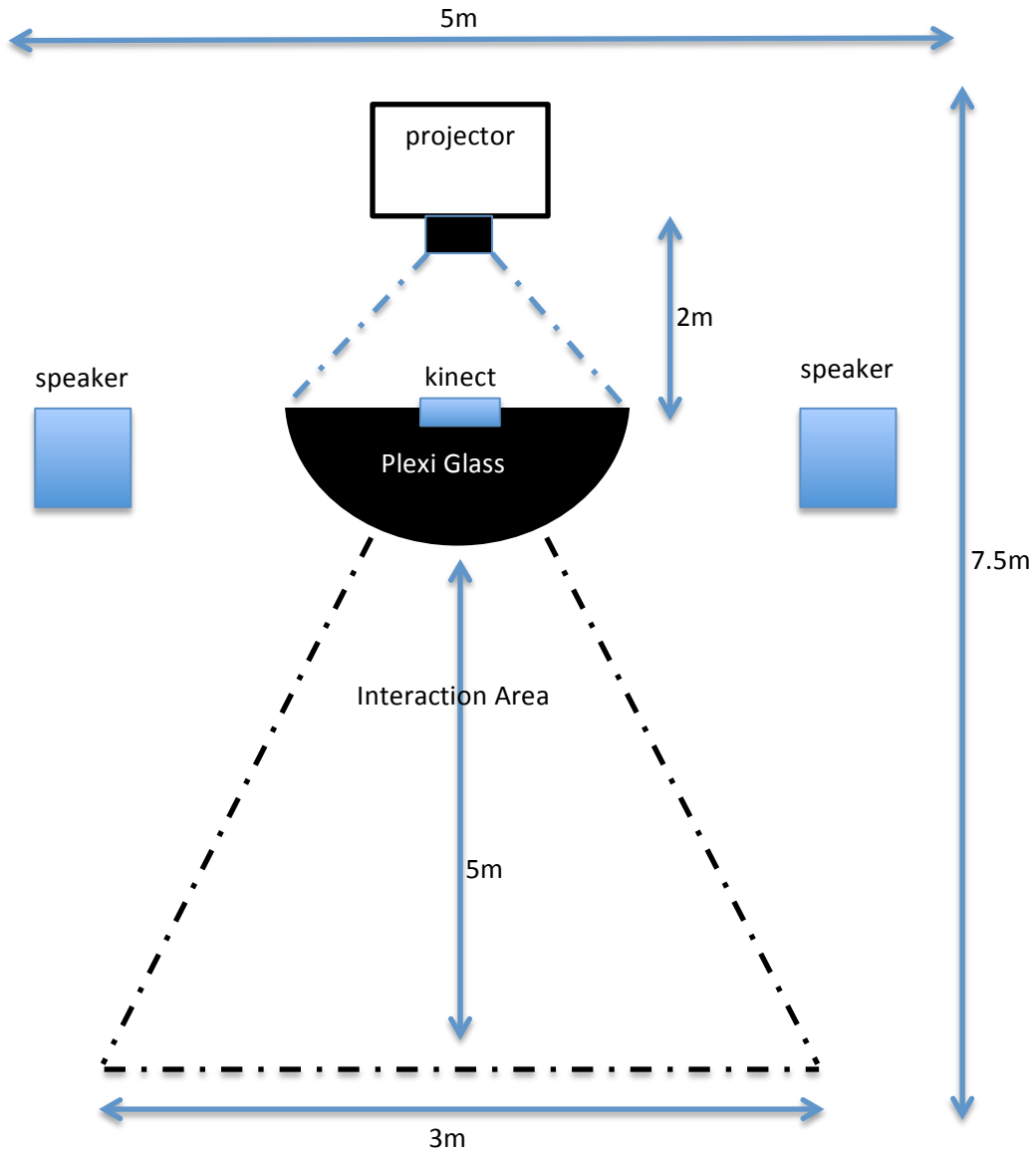
Figure 8.5 DarkStar -OSC communication



DSTrack Implementation:

With the brief clarified and the projection surface set, the problem of tracking could now be tackled. The interface I envisaged had to enable users to select and move individual points on the surface from a distance. Having considered and rejected a number of methods including the overhead placement of an IR camera and contourFinding, and the placement of an IR camera behind the projection surface with the audience using IR pointers, I settled on using a high mounted X-box Kinect as the most practical solution. Not only was this equipment inexpensive, but its hardwired depth perception and user tracking features would allow me to derive a three dimensional vector from users' point gestures to calculate an accurate intersection with the hemisphere. The Kinect was mounted just above the hemisphere with a downward tilt to track users in the vicinity.

Figure 8.6 DarkStar - overhead view showing tracking area

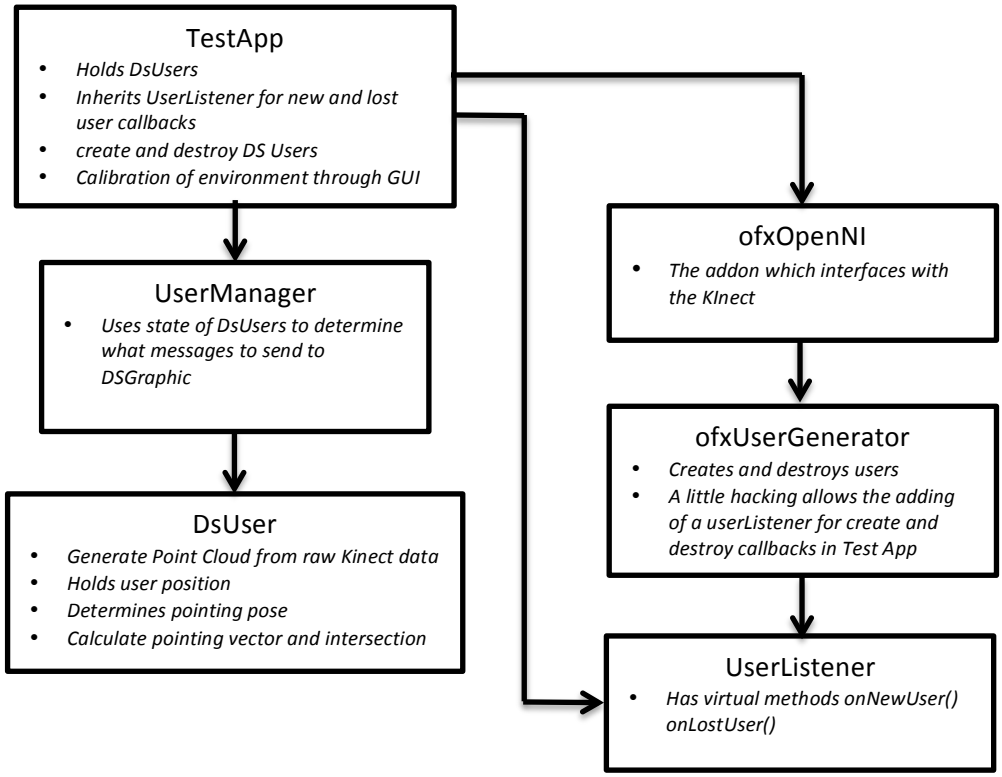


The basic functionality of DSTrack is best summarised by the messages it sends to DSGraphics. These comprise of messages for when a user begins and ends pointing at the sphere, when a user begins and ends moving their point, and messages to update the projected sphere intersections of current point vectors. The application uses OpenFramework's ofxOpenNI addon, which wraps the OpenNI library that interfaces with the X-box Kinect. OpenNI robustly handles the segmentation of users who are in view of the camera and returns a masked image, similar to those used in previous computer vision

projects, for each user. In addition to this it gives a 3D coordinate for the user's centre of mass in relation to the camera. Though ofxOpenNI has accessor methods to these variables, a certain amount of unreliability with regards to the deleting of old users and user indexing as well as the need for data persistence across a number of extra variables led me to create my own class of dsUser. I rejected the use of the Kinect's skeletal tracking features, as I did not want users to have to hold the calibration pose in order to be registered by the Kinect, especially as there was no way of feeding back to them when the pose had been registered. Since the project launch, various libraries have been updated to perform calibration without need of the pose which would have allowed use of the feature, improving accuracy and performance.

A small amount of hacking of the ofxUserGenerator class in ofxOpenNI allowed me to detect callbacks for new and lost listeners from my testApp, which are then used to create and destroy dsUsers with corresponding references to the ofxOpenNI user. When ofxOpenNI is slow to delete a user who has left the scene, the centre of mass will be set to the origin. When this happens a dsUser will simply flag itself as inactive and no processing will occur.

Figure 8.7 DSTrack - class structure



The data processing for dsUsers is set out in Figure 8.9. Note that in addition to the user mask dsUser also uses ofxOpenNI's depth mask, which allocates a depth value giving the z-distance from the camera, for every pixel. The angle for the correctional rotation which compensates for the tilt of the Kinect is the angle between the y-axis the normal to the floor plane provided by ofxOpenNI, and the axis of rotation is the cross product of the y-axis and the floor plane. All of the dsUser's points are then rotated using this angle and axis around the floor point provided by ofxOpenNI, thus orientating the point cloud to the floor instead of the camera. This allows meaningful measurements such as user height to be taken and for the users and sphere to be in the same coordinate space. In order to omit outliers, highest and furthest points are taken as mean averages from samples of multiple points. The proportions for sternum and eye line are taken from average body proportions based on a height of eight times the head. The point test requires the furthest point to exceed a threshold, set through trial and error, which is a proportion of the user's height. The final four stages of processing only happen if the point test has been passed.

Figure 8.8 DS Track - pointing test screen shot

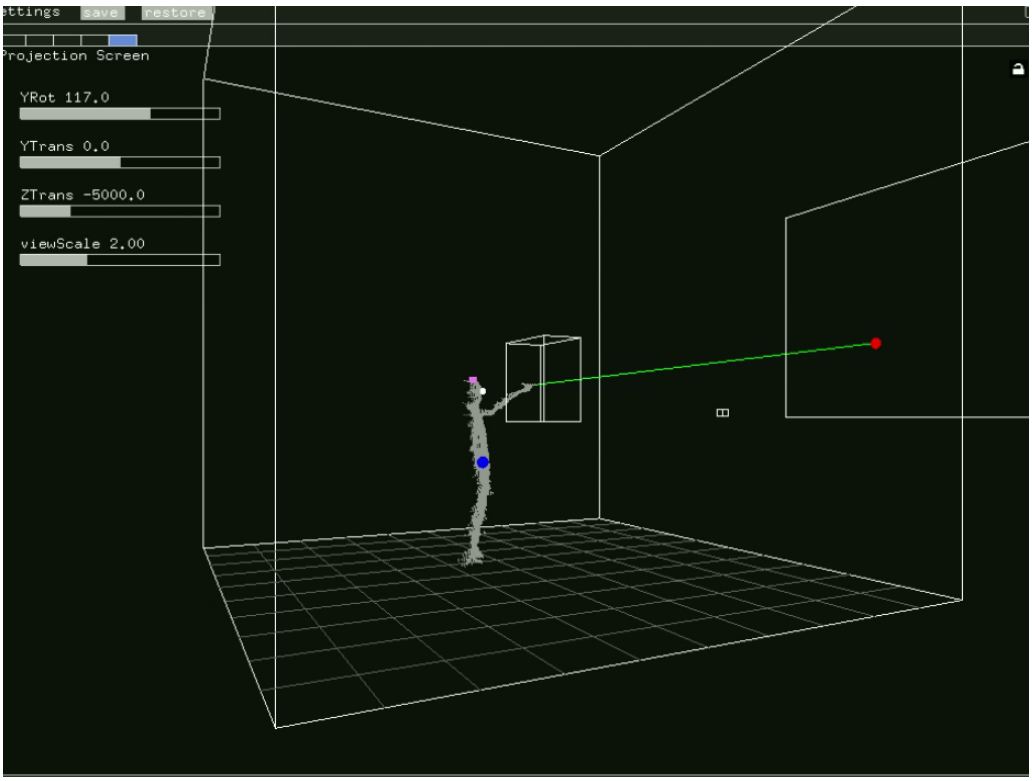
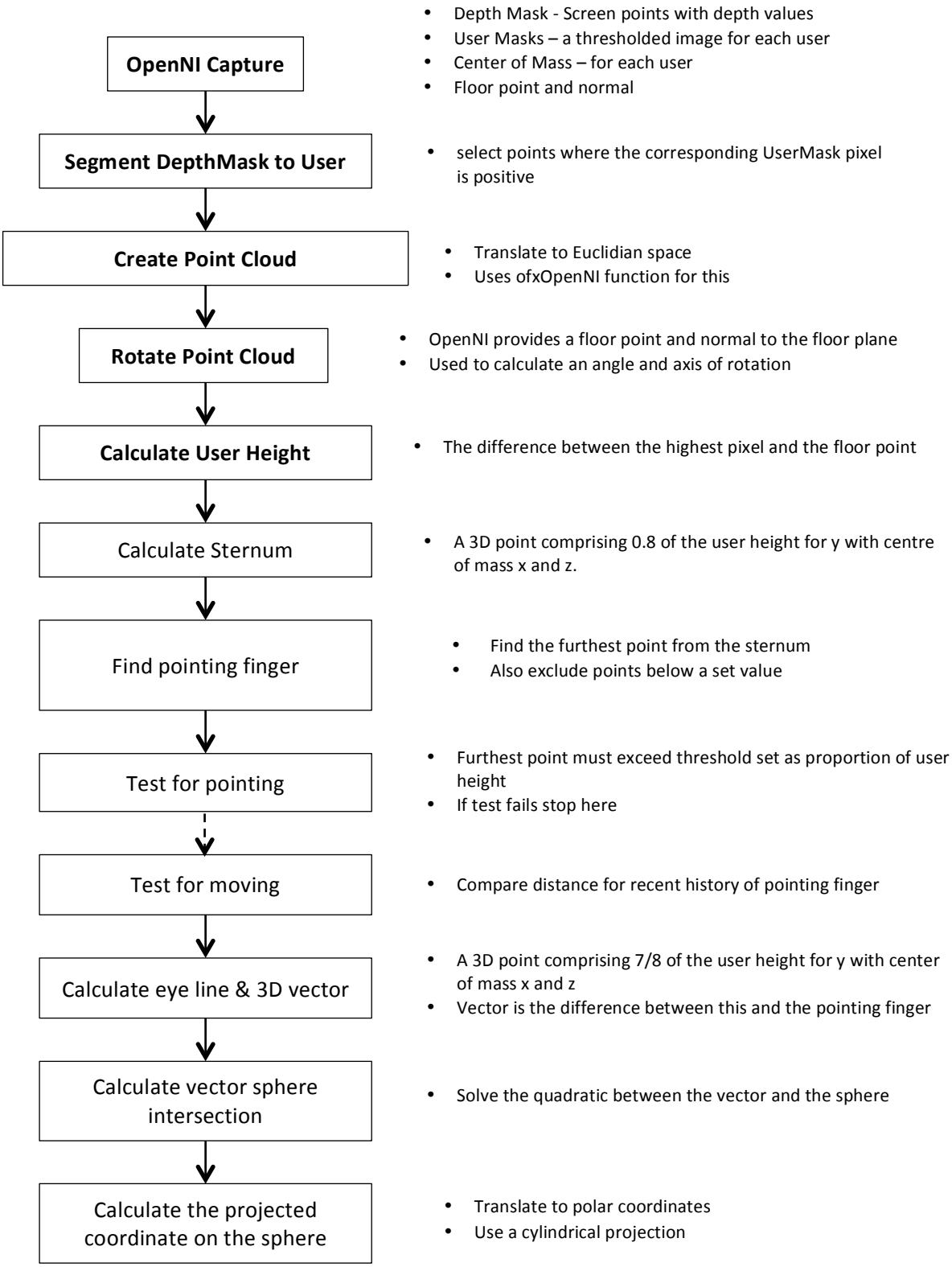


Figure 8.9 DSTrack - data processing



The calculation of the intersection between the pointing vector and the sphere is a little more complex. A point beyond the sphere *p2* is used to form a ray with the finger point which when substituted into the Cartesian equation for the sphere produces a quadratic equation. The solution to this equation provides the Cartesian coordinate of intersection. Where there are two intersections the intersection that is further from the user is rejected. A discriminant value of less than zero indicates that the ray has missed the sphere altogether. Where this happens a second test for a larger buffer sphere is performed if an intersection is found then this is used instead. This method allows for the program to allow wide points to still be registered without distorting the geography for accurate points. The resultant intersection is now translated from Cartesian into polar coordinates and then into 2D coordinates to be sent to DSGraphic using an azimuthal projection.

OSC messaging is handled by the userManager. It's important to note that only pointing gestures resulting in intersections with the sphere are registered with DSGraphic. The userManager keeps a list of the current users who are intersecting with the sphere and uses this determine what messages to send to DSGraphic. A final point of note is that the sphere size and position and other variables can be adjusted through a GUI to allow for easy calibration in response to different physical setups.

DSGraphic implementation:

The functionality of DSGraphic comprises the lunar clock functions of *DarkStar*, the handling of messages to facilitate interaction between users and stars, graphic rendering and the appropriate messaging to SuperCollider. In DSGraphic every rendered star is instantiated in an object of the same name with, amongst other variables, a 2D coordinate determining its projected position on the sphere. They are stored in a two dimensional array arranged in columns according to x position to facilitate quick searching, and are initially laid out in a hexagonal pattern, filling the sphere. Stars have an active and inactive state. In their inactive state they increment their x coordinates by a constant amount creating the impression of a slow rotation around the sphere. Each inactive star is rendered as a blurred white circle, and lit according to an intensity variable. For most stars, for most of the time this value is too low for the star to be visible. Just a few permanently lit stars are enough to give the impression of a single axis of rotation; occasionally stars chosen at random have their intensity values incremented and decremented to give the impression of twinkling via a method called twinkle.

Inactive stars are also responsible for *DarkStar's* time keeping function through a one minute cycle. This happens through the calling of twinkle on stars in sequential order counter to their direction of travel. The timing of this process is calculated to create a vertical band of light that travels across the sphere over a period of thirty seconds. As this happens a message is sent to SuperCollider to begin the corresponding audio sequence. This uses two clipped low frequency sine waves with amplitude modulation to create a low polyrhythmic rumbling texture. This is followed by thirty seconds of silence combined with regular star movement before the start of the next cycle.

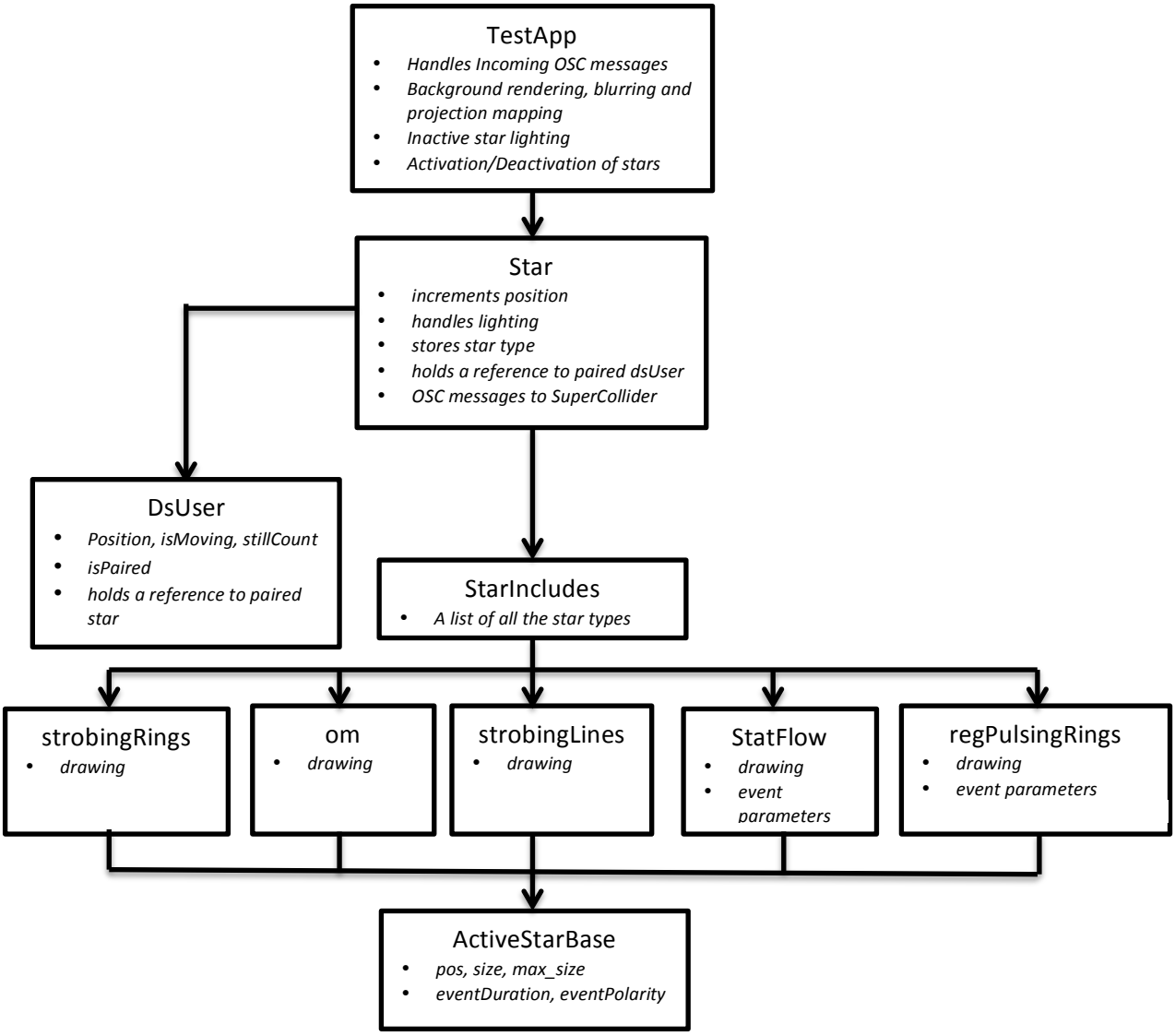
Stars are triggered into their active state via messages from DSTrack. When a user's pointing gesture first intersects with the sphere a "/newUser" message is sent to DSGraphic. This causes a dsUser of a given id to be flagged as active. For DSGraphic, a dsUser is not a person in view of the camera but a coordinate representing an intersection with the sphere. The dsUser class also holds a variable stillCount, which counts how many frames since the coordinate last changed. This is used by a method called pairPointsAndStars which looks for active dsUsers that aren't paired and also have a stillCount of over twenty. For these it finds the star currently nearest to the dsUser's coordinate and registers it as active. The respective ids of user and star are copied to each object, and a "/newStar" message with the dsUser id and type of star as arguments is sent to SuperCollider.

Star deactivation happens when a "/lostUser" message is received from DSTrack. In this case, after a certain number of ghost frames, the dsUser and paired star are both flagged as inactive. The manageStars method then uses the star's new location to place it in the correct column in the star array. In this way, the arrangement of stars gradually loses its hexagonal pattern through interaction.

When a star is active its position is no longer incremented with all the other stars. Rather it gradually increments its position towards the dsUser's current coordinates. From the moment it becomes active, the star also increments its size. If active for long enough and left unabated, the star will dominate the entire sphere. However, where there are neighbouring active stars and the edges of the two stars intersect, the larger star will continue to grow causing the smaller one to contract.

In contrast to the uniform blurred circles of inactive stars, active ones have individual rendering patterns. The ultimate aim is for every star to have its own active draw type, though for the moment there are only five types. With this in mind, a modular system has been employed in order to allow for the later addition of other star types by myself and others. Each star has its own object, inherited from activeStarBase, which is responsible for active mode rendering. The activeStarBase has update and draw methods as well as several variables governing an unspecified event that is triggered by changes in the paired dsUser 's isMoving variable. This event has a duration, which is set within the child class and a polarity contingent on whether the dsUser has switched from moving to still or vice versa. In this way slight movements from the pointing user cause intermittent events in the star. These shall be commented on in the descriptions of the stars themselves. Finally, in order produce its own sound, each active star sends frame by frame update messages to SuperCollider containing arguments for the dsUser id, a normalised value for the size of the star, and the x position of the star which is used for panning. In addition to this, there is also an argument to trigger new events with corresponding duration and polarity.

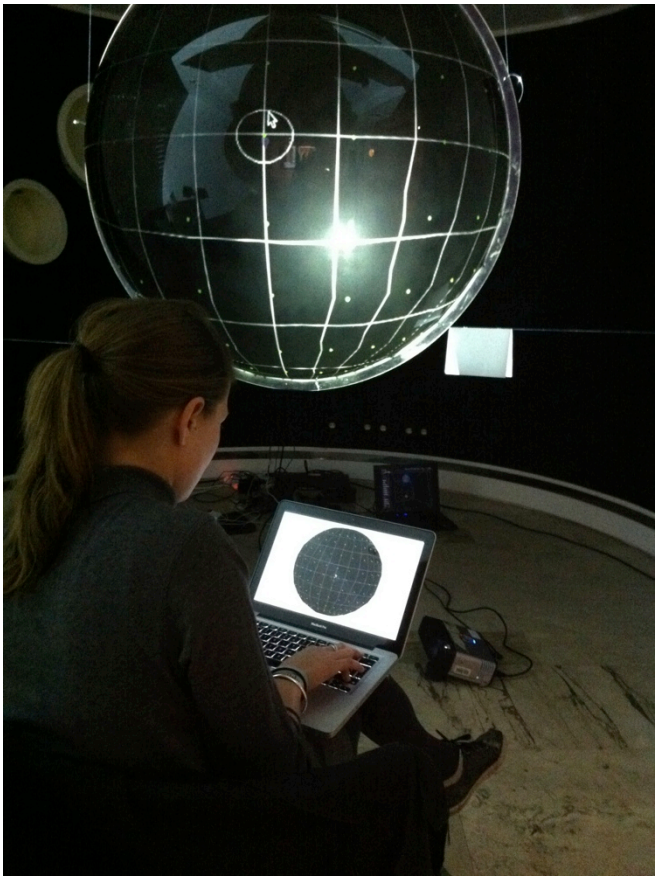
Figure 8.10 DSGraphic - class structure



Before coming to describe the qualities of the individual stars, some comments should be made on how the projection mapping was achieved. Though from the outset, the mapping of the graphic onto the sphere was supposed to be handled by another artist, the commissioning body was unable to confirm who this would be until most other aspects of the development had already been completed. In the end I was paired with Carolien Teunisse, an artist from Holland specialising in projection mapping. However, the late initiation of this partnership lead immediately to platform issues as Carolien works entirely with commercial software rather than coding environments. After some investigation we determined that we wouldn't be able to find a way to transfer the graphics from the GLUT environment of OpenFrameworks into the necessary software package. Through discussion with Carolien about how she usually maps projections we arrived at the solution of creating an interface for her to warp the image in DSGraphic. This interface renders the screen

image to an off screen texture and then redraws it binding it to a ten by ten grid of quads. The position of the quad corners can be adjusted through a basic key interface and saved in an XML file. This allowed Carolien to carefully adjust the points, warping the screen image into a sphere.

Figure 8.11 DSGraphic - Carolien Teunisse using projection mapping interface



Though the look of the inactive stars had been largely determined by the commissioning body's concept for the piece, I had designed the active star concept and so had a greater degree of freedom over how the stars should look. Nevertheless, there was a sizeable difference of opinion over to what degree the stars different characters should be reflected through their graphical forms with me tending towards variety and the commissioning body tending towards uniformity. Whilst I sympathised with the desire for a stylistically concise work, it nonetheless seemed that we had completely different thresholds for what this entailed. I would posit that the commissioning body's viewpoint, influenced by their background in graphic design, was inappropriate to the time-based animated world which would be far more tolerant of variation than static images in print. Nevertheless, a particularly satisfying compromise was found. This involved rendering all active stars using individual white pixels, randomly placed within bounds, resulting in an effect similar to

interference on analogue TV sets. Whilst providing a uniform element to the installation, this technique also proved surprisingly flexible, allowing for different types of shape, movement, and shading densities, all through stochastic means.

The StrobingRings class renders rings by randomly selecting points within certain bounds and rotating them around a central point by random amounts. By adding an offset to every point, which changes every few frames, a strobe effect is created whereby the viewer sees several rings simultaneously when only one is being rendered. The graphic is accompanied by synth producing a quasi-pitched sound by using LFNoise0 at high frequencies. The frequency changes on a pulse, complimenting the random movement of the rings. Interestingly it was found that the best tempo for the audio was slightly slower than the strobing. The class also has events for both polarities, one of which temporarily speeds the rate of strobing and audio, with the other having the opposite effect, which also serve to reinforce the sense of synchronisation.

StrobingLines uses a similar technique, this time employing multiple ranges, offsets and rotations to create series of overlapping lines. In this case, the audio accompaniment uses a clipped and filtered version of LFNoise0 creating pulsed unpitched attacks. The assignment of other parameters such as filter frequency and clip to other LFNoise ugens serves to disrupt the pulse whilst still leaving a vague sense of tempo. The class has events for each polarity, both of which interrupt the regularity of the strobe in different ways. The first which randomly blacks out frames is accompanied in the audio by an increase in the amount of clipping which, depending on the randomly fluctuation of the filter frequency, will cut out attacks. The second similarly blacks out frames but also leaves the line direction unchanged. This new constant is accompanied in the audio by a removal of the filtering and clipping which combined with some enveloping results in a drum roll like effect. Once again, of note is the lack of need of precise synchronisation to create the desired effects.

Statflow uses Gaussian distributions to draw nested circles with tapered edges. The more static animation is accompanied by a low resonant sound created by passing a saw wave through a resonant low pass filter. A small amount of filtered LFNoise0 is added to create a crackling sound which when combined with the graphic appears to emanate from the tapered edges. Unlike the previous classes Statflow has only one event for one polarity. The event itself involves the inner circle disappearing and a sudden expansion of the outer circle, creating a sense of the star exploding. The event is accompanied by a loud bang of LFNoise0.

This is accompanied with decaying crackling sounds as the outer circle expands further. Finally a white noise filter sweep accompanies the star's return to its original form through contracting the outer circle and the gradual reformation of the inner circle. The greater duration and impact of this event makes it necessary to pass triggers through a probability determined gate in order to reduce their frequency.

RegPulsingRings uses a similar technique as StrobingRings to create an evenly spaced series of concentric rings. Once again there is no strobe effect. Instead the radius of the rings expand in such a way as to maintain the comparative distance between rings as well as the size of the outermost ring. The effect is reminiscent of drips falling into a pool in water. The motion is accompanied by a soft pulsating sound, created using a clipped sine tone with amplitude modulation. Of its two events, one involves a long contraction and re-expansion of the rings, accompanied by a pause in the amplitude modulation and a decrease and increase in pitch corresponding to the expansion and contraction. The second event is essentially the inverse though happens over a shorter period of time

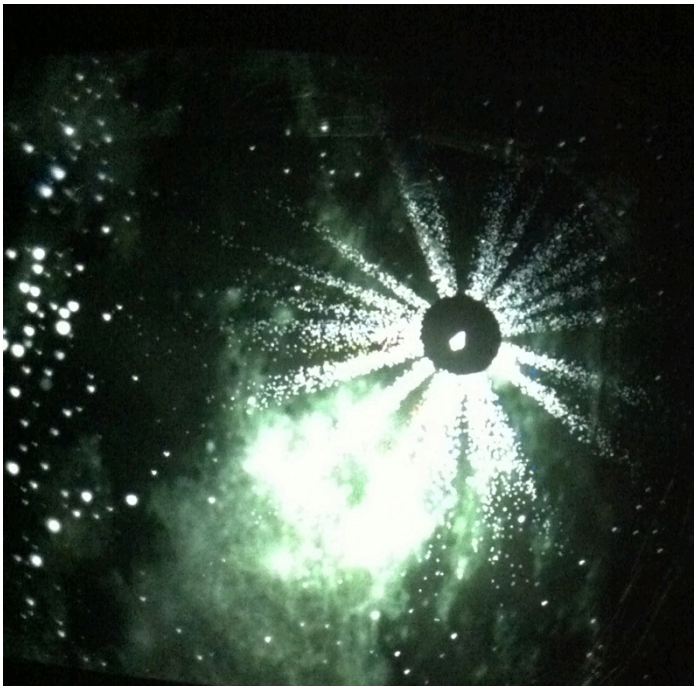
Om uses a circular arrangement of flaring lines, which are created with a Gaussian distribution. The lines randomly increment their rotation creating a shimmering effect. The image is much like a child's image of the sun with the exception that its centre is shaded black. This is accompanied by a gradually shifting chordal texture produced by overlaid synths which pass brown noise through the resonant Klank synth. The fundamentals are randomly selected from Bohlen-Pierce ratios, whilst the partials follow a Wendy Carlos Alpha tuning. The event which is triggered by both polarities randomly draws fine horizontal lines across the black centre of the star implying interference on a screen. The accompanying crackles continue with the effect.

Conclusion:

Despite the many frustrations encountered in working with MoTA in the building of *DarkStar*, the project was well received at its launch. The audience at the opening comprised the people from local arts-scene, journalists and funders. The physical object was aesthetically pleasing and worked well within the space. The plexi-glass hemisphere was hung with nylon fishing wire with the curved wall behind it painted black, creating an impression that the installation was floating. The gallery space was intimate but just large enough to support this effect. The use of back projection from such a short distance meant

that the audience looked directly into the light of the projector lending the white stars a luminous quality. Furthermore, the imperfections of the sanded plexi-glass gave a natural textured feel to the projection.

Figure 8.12 DarkStar - projection through plexi glass



Also the visual aesthetic of the graphics was suitably coherent and sophisticated. Whereas graphics in previous projects had been of a largely functional nature and built around known techniques, the graphics for DarkStar had been created in order to match a brief and had with some difficulty taken me into unfamiliar territories. The technique of drawing with random pixels proved more than flexible enough for the purposes of the installation. One could imagine an almost endless amount of stars, which could be created. The technique also showed potential for use in other pieces.

Judging by the amount of time spent with the installation, the audience also seemed to enjoy the interaction. The pointing gesture had a dramatic quality to it and I enjoyed the way that the installation manipulated its audience to become part of the spectacle. Some reported feeling a sense of power as their star engulfed the globe. The method of determining point gesture and direction was also robust. However, one area of regret was that the interaction lacked some of the intuitive qualities of *Soundpit*. The audience had to be given instructions, albeit simple ones, in order to use the installation. Furthermore, the intentionally sluggish movement of the stars with the intersection acting as an attractor,

created some degree of confusion. This raises a question of whether interaction need always be directly responsive to the users actions. In this case, a direct correspondence between the point and the star would have been unsatisfactory both in terms of the graphical movement and the users' movements. Here the stars' sluggishness insists on a slow and steady form of interaction.

Figure 8.13 DarkStar - user interaction

I was also pleased with the coherence of the audio-visual world. The glitch-like techniques employed in the sound complimented the minimal black and white graphics. Furthermore, the coherence of the sounds meant that they inter-combined in a pleasing way. In particular, the timbre and frequency of the regularly occurring wave event was balanced to be impressive enough in isolation but blend into the background when there were other events happening. A hitherto uncommented feature of DarkStar is that its audio-visual relationships are impressionist, largely devoid of symbolic, parametric representational systems. A consequence of this is that with the exception of their intermittent events and the increase in amplitude with size, the stars are mostly static in quality with little extra to be discerned about the star after its first viewing.

The stasis of these audio-visual relationships needs to be compensated by multitude. *DarkStar* falls short in this sense by simply not having enough stars. One could imagine that for a user to have the sense of exploring a world, fifty stars might be required. An impression of exploring an infinite world would require it to be unlikely that the same star is seen twice. One could speculate that this would need stars at least in the order of hundreds. Though a

larger team and a better working relationship with the commissioning body would have allowed more detail, such work could not be achieved in a matter of weeks or months, but more likely years. In the current environment of rapidly changing technologies, devotion to a single project over numerous years seems foolhardy – the implementation will undoubtedly be outmoded by the time the project comes to fruition. The possibility of curating contributions from open source communities in an open call could speed up the process, though whether there would be sufficient uptake or the required coherence achieved remains to be seen.

Nevertheless, the framework offered by *DarkStar* would allow the flexibility for such a large amount of variations to be possible. One could also imagine more advanced behaviours for stars. For example, stars whose form is permanently changed by events, perhaps in a generative manner. Some stars could dominate when larger, whilst others could dominate when smaller. Stars could act as attractors for inactive stars. Some have suggested that stars could combine with each other, though I think that given the modularity of the star classes this would be hard to achieve. In conclusion I would argue that even in its incomplete form, *DarkStar* nonetheless demonstrates a viable method for the non-linear presentation of audio-visual material. With the required detail, one could envisage much deeper engagement from users, perhaps lasting for hours at a time. Certainly the form suggests similar, more achievable, interactive non-linear presentational modes, perhaps using touch devices. Nevertheless, in its fully realised spherical form, surrounded by groups of onlookers eerily frozen in a pointing pose, *DarkStar* would be a spectacle indeed.

9. Cube with Magic Ribbons (SoundCircuit)

Video, code and score: http://www.simonkatan.co.uk/phd/cube.html

Overview:

Cube with Magic Ribbons (24.03.2012) is a computer visual and synthesised sound composition for live performance. The piece takes its title from a drawing of M.C.Escher which is rich with contradictory perspectives but it is also inspired by the wrapped spaces found in the two dimensional graphics of early computer games such as *Asteroids* and *Pac-Man*. It was created using a custom visual sequencer SoundCircuit, which rather than employing a conventional DAW layout, allows multiple virtual tape-heads to travel through a two-dimensional wrapped space along tracks that can be freely inter-connected. As the tape-heads travel through the resultant network, the topological layout of the tracks comes to directly influence the macro form of the music. Furthermore, as the piece unfolds the nature of this already confusing space reveals itself to be increasingly elastic and complex, yet inexorably intertwined with the musical form.

Having been satisfied with the results from *God Over Djinn*, I was keen to produce a further interface and composition along similar lines. In this case I wanted to shift the focus away from texture and towards sequences of events. In particular, I was interested in the relationship of sequence to memory and in exploring an intuition that formal relations between streams of events such as, symmetries, transpositions, retrogrades and inversions, are considerably easier to parse visually than aurally. I also wanted to see whether it would be possible to find a visual paradigm that could equal the expansiveness and drama of SoundNest's visual recursions. Towards this end, the notion of a two-dimensional space wrapped across both horizontal and vertical axis seemed an appropriate context. Such a space is disorientating from the start – one might imagine that it is merely the projected surface of a sphere or cylinder when in actual fact it's a torus.

It struck me that an important quality of *God Over Djinn* was the notational look and feel of its thin black lines on white background. It seemed to me that this visual aesthetic had been useful in communicating the precise and symbolic relationship between audio and visual.

Cornelius Cardew's notation in *Treatise* creates a similar effect, but also playfully distorts its symbols towards expressive ends. I wondered whether SoundCircuit could produce something akin to an animated version of *Treatise* – a "graphic music" which actually produces sound (Cardew, 1971). Similarly the visual world of circuit diagrams seemed to fit my aims. Not only did circuitry provide its own catalogue of visual symbols, but also there was an analogue between the idea of current passing sequentially along tracks through various components and my notion of interconnected tracks with moving tape-heads. Furthermore, circuit diagrams, as sets of symbolic instructions for actions to be carried out, are also notations.

Implementation:

Once again the composition of the piece involved the design and creation of a bespoke software interface using OpenFrameworks with sound from SuperCollider. As with *God Over Djinn* I have chosen to split the work between interface SoundCircuit as a composition and performance tool, and the composition *Cube with Magic Ribbons* which acts as an exposition for the features and functionality of the software.

SoundCircuit Implementation:

As with SoundNest, SoundCircuit was designed using an iterative process, whereby features were gradually refined through on-going experimentation with the aim of achieving maximum usability within a live context. To some degree my previous work with SoundNest had prepared the ground, meaning that more parameters were defined from the outset. Nevertheless, many aspects of the interface were decided upon during the implementation, necessitating some considerable refactoring at key moments in the process. The most significant of these included how the size of the space would be changed and what would happen to objects after resizing. Another decision was how the tape heads should turn at junctions between tracks, and a further one was whether to have a single or multiple surfaces.

The final implementation uses a single two-dimensional surface, wrapped along the horizontal and vertical axis, and viewed through a camera hovering over it. The user is able to draw tracks onto the surface in lines parallel to the two axis only, and add readers to them, which move along the tracks at adjustable speeds. Nodes are found at junctions between tracks, and these can be programmed to restrict the direction of travel of the

readers. Finally an array of different objects called blips can be added to tracks. When a reader passes over a blip it triggers a sonic and visual response from the blip.

An important aspect is the omission of delete and adjust functions for tracks and blips. This was motivated by a desire for SoundCircuit to function as an analogue for memory, which after all has no delete. This constriction, placed on the performer, means that the audience will observe the construction of a visual world whose size and detail grows in correspondence with that of the music produced. However, this visual world can't be viewed in its entirety at any given moment. Instead, the audience catches glimpses of previous moments in the work as the camera floats over its surface, and perhaps attempts to piece together the geography of the world and hence the structure of the music.

Figure 9.1 SoundCircuit - annotated screen shot

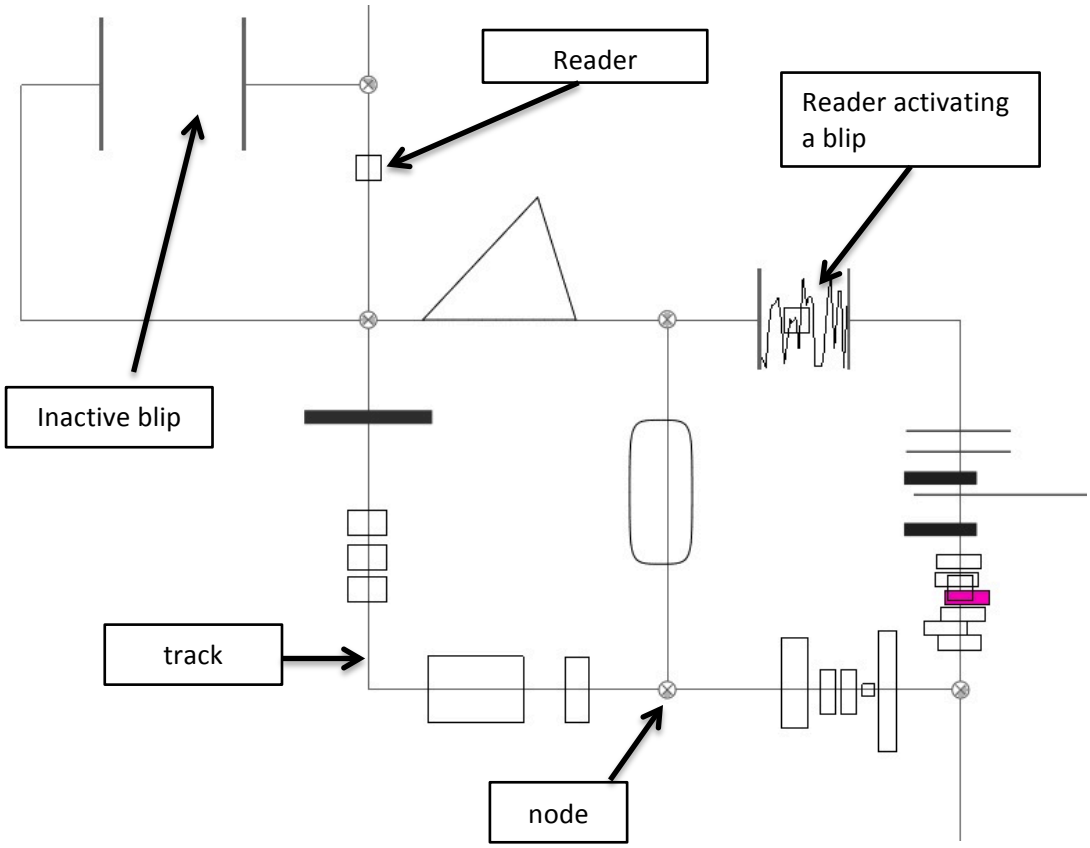


Figure 9.2 SoundCircuit - feature list

Feature	Key	Mode	Mouse	Notes
Add Mode	'1'	any		
Adjust Mode	'2'	any		
Destroy Mode	'3'	any		
Add Short Track	hold 't'	Add	Left and drag	track or node must be selected
Add Long Track	hold 't'	Add	Right and drag	when nothing is selected
Add Blip (1)		Add	Left and drag	track must be selected
Add Blip (2)		Add	Right and drag	track must be selected
Add Blip (3)	hold tab	Add	Left and drag	track must be selected
Add Blip (4)	hold tab	Add	Right and drag	track must be selected
Change Preset	up and down arrows	any		
Change Bank	left and right arrows	any		
Select Nearest Reader	hold 'r'	any	move the mouse pointer	
Follow Reader	hold 'r'	any	right click	
Add Reader	hold 'r'	Add	Left (optional drag to adjust speed)	
Adjust Reader Speed	hold 'r'	Adjust	Left and drag	
Destroy Reader	hold 'r'	Destroy	Left	
Add Space		any	Right and drag	when nothing is selected
Drag Camera		any	Left and drag	when nothing is selected
Toggle Camera Roll	'z'	any		
Toggle Camera Tilt	'x'	any		
Toggle follow reader mode	space	any		
Zoom In/Out	'c' , 'v'	any		
Toggle Node Sockets		add or adjust	Left and drag clockwise/counter-clockwise	when mouse is over a node

The wrapped space involved several stages of implementation. The most trivial of these was the wrapping of the moving readers. This simply involves modulo calculations on a reader's horizontal and vertical coordinates with the effect that when a reader reaches a top border it appears at the bottom border and so on. However, I also wanted the camera to move freely through the space, wrapping the space so that the user was unaware of any borders. The solution to this involved drawing the scene nine times in a grid (Figure 9.3). Then a similar modulo operation is carried out on the camera's central coordinate as it is projected

onto the plane below. As can be seen from comparing Figure 9.4 with Figure 9.5 the rendered screen image is identical. The result is a seamless transition from border to border, creating the illusion of an unbounded space.

Figure 9.3 SoundCircuit - camera In neutral position

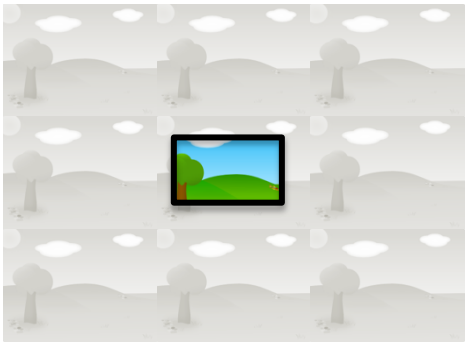


Figure 9.4 SoundCircuit - camera just before modulo

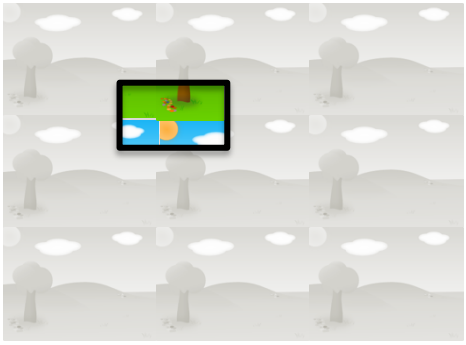
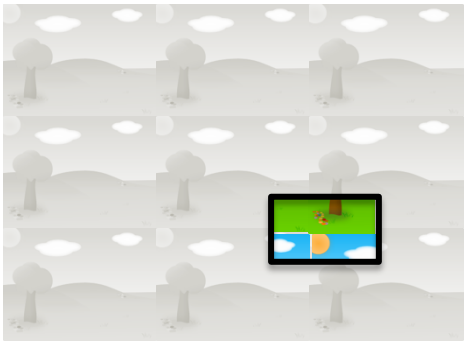


Figure 9.5 SoundCircuit -camera just after modulo



This implementation called into question whether I should operate in a coordinate space for all rendered scenes and make multiple copies of objects, or in a coordinate space of the central scene using translation functions for the purposes of rendering. Deliberating arguments for both types of coordinate systems, I chose the latter option. This resulted in a number of knock-on complications relating to the placement and interaction of tracks and

blips, which crossed over the borders of the coordinate space (Figure 9.6). The similarity between solutions for both tracks and blips justified the use of a base class called segment.

Figure 9.6 SoundCircuit - a wrapped segment

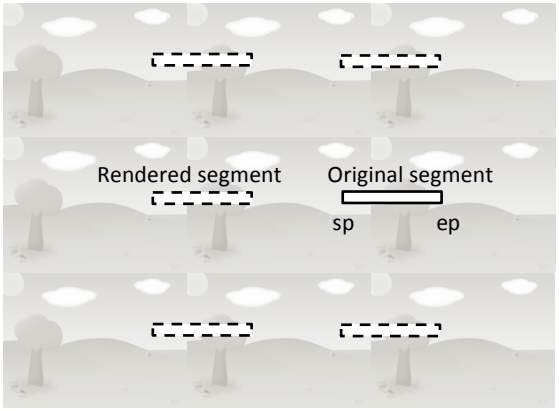
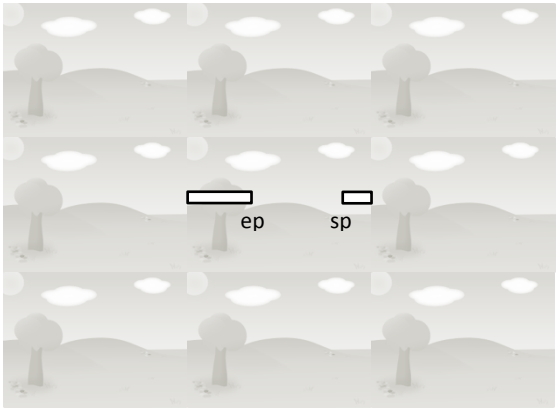


Figure 9.7 SoundCircuit - wrapped segment test bounds



A comparison between the start point and end point is used to determine whether a segment does indeed cross a boundary and needs to be wrapped. If the vector between the start point and end point, which should normally have a positive sum, is negative then the segment is wrapped. The multiple drawing of scenes makes sure that the rendering is correct irrespective of camera position. However, in order to facilitate correct detection of intersections with segments, appropriate bounds need to be established on the opposite side of the space. A further complication arises from segments running along their parallel border. In this circumstance intersecting points on one side of the boundary go undetected. The solution is translate, modulo and retest any points close to borders. There were also further complications specific to blip and track adding functions, which are discussed in the descriptions of those features.

The use of the floating camera required an algorithm for the translation of screen to world coordinates. The need for the use of this function on every frame meant that the standard gluProject and gluUnProject functions were too slow. However, given that I was only using a single plane perpendicular to the z-axis, the writing of my own light weight translation function with no need for costly calls to the depthBuffer was straightforward enough. In addition to translating mouse movements, this function is used to project the four corners of the screen onto the surface at every frame. The resulting quad is used to determine which objects are to be drawn by testing the bounding rectangles of each object for intersections. This prevents off screen drawing yielding significant efficiencies in GPU overhead.

Readers are the most simple of the interface's objects. They travel in straight lines, checking for intersections with nodes and blips on every frame. When an intersection with a blip occurs, the reader calls the react function of the blip and retrieves the relevant information from the blip in order to send an OSC message to SuperCollider. When the reader intersects with a node, it retrieves the possible directions from the node and, discounting the current direction of travel, randomly chooses a new direction. If there are no possible directions the reader stops until a new possible direction is found.

Nodes can't be added or removed directly but are created when tracks are drawn. There are always nodes at the beginnings and ends of tracks as well as at junctions between tracks, though they will not be created on top of already existing nodes. They have four parameters called sockets relating to the directions North, South, East and West, which can be opened or closed via a simple interface using the mouse. Only sockets with tracks adjoining them can be opened; other sockets will be registered as permanently closed. A node with two or more permanently closed sockets will not be rendered on screen, as there are no meaningful adjustments that can be made. Finally, a function to determine if a node is superfluous, checks if it joins only two tracks in the same direction. If this condition is met then the node flags itself for deletion.

It might be discernible from the above that tracks, whilst visually representing potential routes for readers, do not directly control their movement. Instead their function is to determine the position of nodes, the possible positions of blips, and constrain the positioning of other tracks. In order to make the interface useable in a live context, I was keen that the track-adding interface should assist the user as much as possible in creating

well-formed track configurations whilst preventing the formation of malformed ones (Figure 9.8).

Figure 9.8 SoundCircuit - rules for well formed tracks

Rule No.	Tracks …
1	must have a node at the start and end points otherwise they are rejected
2	can create nodes which have no intersection or proximity to other tracks or nodes
3	can create nodes which intersect with perpendicular tracks
4	can use existing nodes where there is intersection with start or end points
5	can use the same node as start and end point
6	can't create nodes which intersect with or are in proximity to existing nodes
7	can't be in proximity to or intersect with existing nodes (with the exception of rule 4)
8	can't create nodes which are in proximity to but don't intersect with perpendicular tracks
9	can't cross perpendicular tracks
10	can't create nodes that intersect with blips

This is achieved through two features 'add long track' and 'add short track.' The former attempts to create a single track spanning the entire surface in the chosen direction expanding outwards from an unoccupied point. If no intersections are found, the resultant track will have no nodes. If a single intersection is found it will have one node shared by start and end point. Otherwise the track will have start and end nodes adjoining the closest perpendicular intersecting tracks on either side of the selected unoccupied point. The wrapped space complicates the finding of neighbouring intersections by creating the possibility of a variety of arrangements of neighbouring intersections. For example, in Figure 9.9, the selected point, denoted by a circle, has a start point with a lower x value and an endpoint with a higher x value, whereas in Figure 9.10 both start point and end point have lower x values and

Figure 9.11 both start point and end point have higher x values. This required various functions to detect all the intersections, and use their distribution to determine how they should be sorted.

Figure 9.9 SoundCircuit – neighbouring intersections

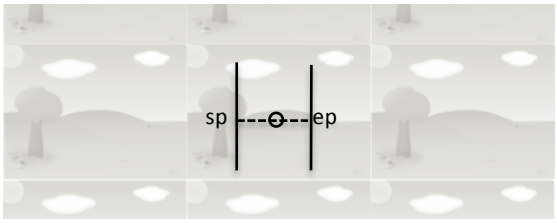
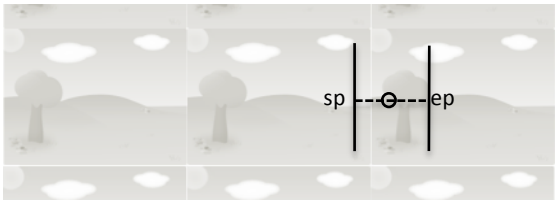


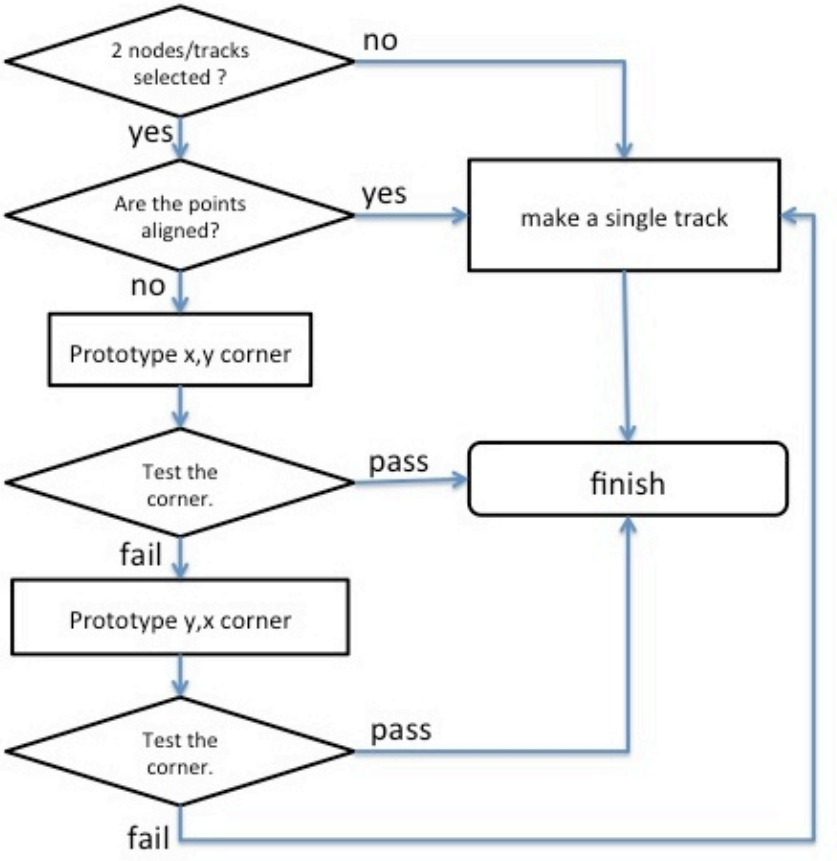
Figure 9.10 SoundCircuit - neighbouring intersections

Figure 9.11 SoundCircuit - neighbouring intersections



The 'add short track' feature uses the same functions to deal with the complications of wrapped space, but has a number of different applications. It is able to create single tracks, which are either free floating or connected to an existing node or point on a track, and of a user-determined length. It is also able to join nodes or points on tracks with single tracks or pairs of tracks in corner formations depending on their alignment. The feature discerns all this from how the user attempts to draw the track using a process summarised by Figure 9.12. The result is a feature that can be used intuitively by the user from a single click and drag motion, without needing to worry about rules determining well-formed tracks or being very precise in their drawing.

Figure 9.12 SoundCircuit - add short track process



The rules governing the well-formedness of blips are considerebly fewer than those of tracks. Nevertheless, the 'add blip' function uses the same functions to prevent unwanted intersections by constraining the blip lengths.

Figure 9.13 SoundCircuit - rules for well-formed blips

Rule	Blips…
1	Can only be added to tracks
2	Can't intersect with nodes
3	Can't intersect with other blips

Complexity in blips stems from a desired modularity in visual and audio integration, and the desire to create an interface that affords as much flexibility for the user as possible without altering the source code.

The blip class acts as a timer for handling the reaction envelope. This can be one of two types, an attack sustain release (ASR), or an attack release (AR). For the ASR type the sustain time is determined by the length of the blip divided by the speed of the reader reacting with it minus the length of the attack time, whilst for AR envelopes length has no bearing on the reaction time. There is also a post decay value, which extends the decay of the visual reaction beyond that of the audio reaction. Each of these parameters can be set either as a value in seconds or as a ratio of the duration of the blip, which is calculated according to the speed of the reader passing over it. This combination of features affords the user the flexibility of separating blip duration from length, allowing the possibility of arranging overlapping events on the same track.

A second class called blipPreset is used to store and set the time related parameters as well as unspecified lists of audio and visual parameters. This facilitates the creation of different types of blips with different numbers and varieties of audio and visual parameters. Each of these parameters is an instance of a class called paramAttribute, which holds options relating to the parameter, such as minimum and maximum values, and whether the value is fixed, set by the user, random, mapped to location or derived from another parameter. As has already been mentioned the reader accesses the audio parameters directly from a blip's preset and uses them to construct the OSC message for SuperCollider. The user is able to group audio parameters into synthDefs, to match the arguments needed by SuperCollider. This is done through an XML interface by editing a file called synthDefs.xml (Figure 9.14). Here the user simply defines the name of a synthDef and the names of the arguments for it. When constructing their presets the user is now able to use their own synthDefs, setting minimum and maximum values and other aspects using the labels they have defined.

Figure 9.14 SoundCircuit - an extract from synthDefs.xml

synthDefs.xml

```
<SOUND_CIRCUIT>
    <SYNTH_DEFS>
        <DEF>
            <NAME>basic</NAME>
            <PARAM><T>freq</T></PARAM>
            <PARAM><T>clip</T></PARAM>
            <PARAM><T>amp</T></PARAM>
            <PARAM><T>pan</T></PARAM>
        </DEF>
        <DEF>
            <NAME>soft</NAME>
            <PARAM><T>freq</T></PARAM>
            <PARAM><T>clip</T></PARAM>
            <PARAM><T>amp</T></PARAM>
            <PARAM><T>pan</T></PARAM>
        </DEF>
        <DEF>
            <NAME>clip</NAME>
            <PARAM><T>freq</T></PARAM>
            <PARAM><T>filFreqL</T></PARAM>
            <PARAM><T>filFreqH</T></PARAM>
            <PARAM><T>amp</T></PARAM>
            <PARAM><T>pan</T></PARAM>
        </DEF>
```

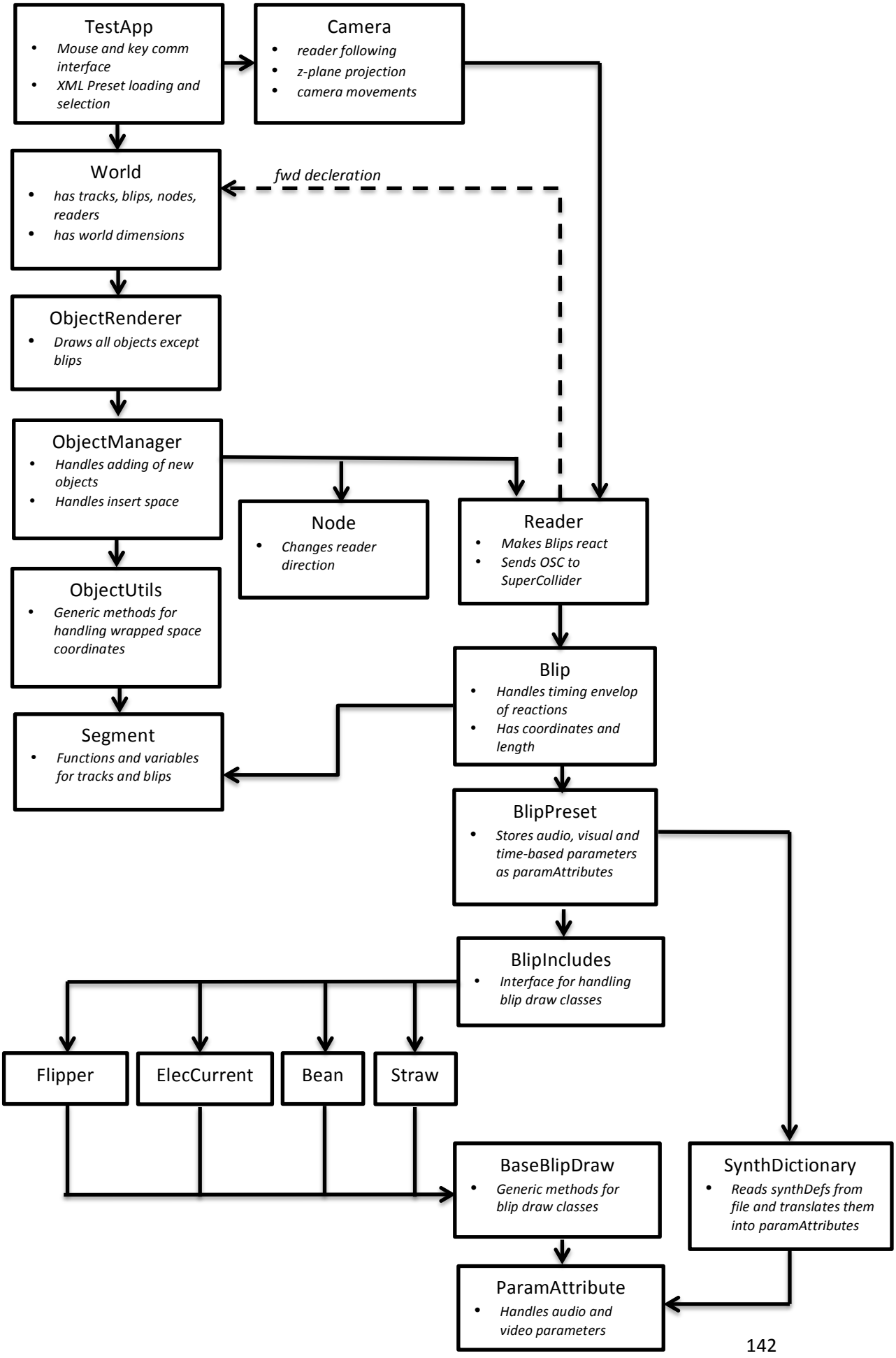
For the implementation of the graphic side, the inclusion of graphics within the program itself led to a different approach from sound. As with *DarkStar* the necessity of having a collection of animated objects that could be added to at a later date warranted the use of deriving drawing classes from a base class. Whilst this class held a few generic functions relating to time parameters and constructing a bounding rectangle, most of the work is done in the derived class. In contrast to the audio where time parameters are sent to SuperCollider once, at the start of the attack and SuperCollider does the rest of the calculation, the draw classes are updated every frame with normalised data for the envelope and post decay. The draw class handles the post decay value in case it needs to distinguish between the main envelope and the post decay. When both values in a blip are zero it is drawn in its inactive state. In contrast to *DarkStar*, blip draw classes define their own parameters and default values in a static method. This is called when a blipPreset is constructed to add the necessary parameters for the particular draw class.

To date I have designed four draw classes, “flipper”, “bean”, “straw”, and “elecCurrent.” I put a great deal of focus on flexibility in their design meaning that, through careful adjustment of presets, each are able to exhibit a wide range of behaviour. With some tweaking of parameters, two instances of the same draw class can have an almost completely different appearance. The function of each class shall be demonstrated via the following description of *Cube With Magic Ribbons*.

Given the greater flexibility and complexity of the preset system in comparison to SoundNest, and the present unlikelihood of anyone other than the author using the software, the design of a preset GUI was left to one side for the time being. Instead presets are defined by editing an XML file called "presets.xml." Here the user is able to freely combine their own synthDefs with draw classes and package them into a preset. As with SoundNest these presets can subsequently be organised into banks for speedy access during performance. Furthermore, the XML loading functions allows the overloading of individual parameters with up to three alternative versions. These can be accessed during performance via combinations of left and right mouse clicks and the tab key allowing the performer to access variant presets without any menu searching. As shall be demonstrated through the discussion of *Cube With Magic Ribbons*, this infrastructure serves to create a highly flexible and practical environment for the creation and management of blips.

A final feature, which might impress the code-literate viewer though is comparatively simple in terms of implementation, is the 'add Space' feature. This feature allows the user to extend the space along whichever plane is currently viewed as horizontal. The space is inserted extending in either direction from the point where the user clicks. Tracks running parallel to the plane and intersecting the perpendicular plane from the selection point are extended in line with the amount of inserted space. All other objects have their coordinates shifted away from the selection point by half the amount of inserted space. This feature is not only vital in preventing the performer's space from becoming over-crowed, but is also particularly useful in disturbing the viewer's conception of the visual space and creating a sense of dynamism.

Figure 9.15 SoundCircuit - class structure



Cube with Magic Ribbons Commentary:

Cube with Magic Ribbons is a largely determinate composition with some room for improvisation at the meso-level of structure. As with *God Over Djinn*, the form is largely determined by the logic of the interface and is intended to act in part as an exposition of its features. The form and interface commands used to achieve it are described in a descriptive score (Appendix C and online). The following paragraphs make reference to this in order to comment on some key aspects of the work.

The sympathetic pairing of sound with animation is vital to the success of the work. In some cases such pairings are inspired by external reference points. One example can be found in the 'elec' preset which first appears in section A3. The graphic is clearly representative of static electricity jumping between two contacts. The accompanying synthDef uses filtered low frequency noise to replicate the sound one might expect from the electricity. The 'needle Glitch' preset in section A2, provides a subtler example. Its graphic is a single line that reacts by pivoting in a flicking movement that is reminiscent of the way that a needle on an analogue peak-meter jumps when the input signal clips. This graphic event is paired with the 'brown Glitch' synthDef, which imitates a clipping signal by hard enveloping brown noise through a resonant low pass filter. There are also more abstract pairings, which nevertheless imply causal relationships between sound and graphic. For example, the 'basicRing' preset, found in section D uses a sound, which in respect of its hard attack and long decay has a bell-like quality. This is paired with a rectangular graphic, which reacts by glowing a hue derived from the pitch and spinning on an axis parallel to the track. Though the image makes no direct reference, it implies that the spinning of the rectangle is in someway, perhaps mechanically, the cause of the sound.

Such pairings are sometimes reinforced through the derivation of visual parameters from audio ones. For example, the 'elec' preset, derives 'density' which varies the number of points in the moving jagged line of the electricity with 'frequency', which varies the speed of the low frequency noise oscillator. Presented variations in frequency reinforce the agreement between sound and graphic – when we see more electricity it means that we hear more too. In the case of the 'needleGlitch' preset, multiple parameters such as line thickness, colour, offset and height are derived from sound parameters. By the time section A2 is completed, and there are many varieties of 'needleGlitch' blip, these derived parameters work together to imply an underlying symbolic system. The 'basicRing' preset

adopts both approaches simultaneously with some degree of internal contradiction. Speed of rotation is derived from frequency and the height of the rectangle from amplitude. When the frequency is high and the amplitude low, the fast rotation might imply that the smaller object rotates faster as a result of its lesser mass. However, a low frequency coupled with low amplitude encourages an alternative interpretation – the reduced energy of the object's rotation produces less sound.

Figure 9.16 Cube - needleGlitch visual parameter derivations

```
<VISUAL><NAME>rotation</NAME><ABS_VAL>0</ABS_VAL></VISUAL>
<VISUAL><NAME>height</NAME><SET_TYPE>slave</SET_TYPE><SLAVE_TO>amp</SLAVE_TO></VISUAL>
<VISUAL><NAME>grayCol</NAME><SET_TYPE>slave</SET_TYPE><SLAVE_TO>filFreq</SLAVE_TO></VISUAL>
<VISUAL><NAME>thickness</NAME><SET_TYPE>slave</SET_TYPE><SLAVE_TO>attack</SLAVE_TO></VISUAL>
<VISUAL><NAME>trackOffset</NAME><SET_TYPE>slave</SET_TYPE><SLAVE_TO>freq</SLAVE_TO></VISUAL>
```

Another point to note is the flexibility of the draw objects. This is perhaps best demonstrated through comparison between the 'sawTooth' and 'softBasic' presets. Despite quite different appearances, both use the 'bean' draw object, which draws and manipulates bezier curves between multiple vertices. In the case of 'sawTooth' just three vertices are specified, and their control points are set using 'a_add' and 'b_add' to make a tooth-like triangle with two points on the track and one off the track. When the blip reacts, the negative setting of 'a_swell' and 'v_swell' causes the outer point of the tooth to fold towards the track, bending the bezier curves and giving an impression of depth as it moves. For 'softBasic' however, 'a_add' and 'b_add' are configured so as to create a curved configuration of four convex beziers, looking a little like an old TV set. Here the 'b_swell' is used to make the blip inflate as the reader passes through it.

Figure 9.17 Cube - swellTooth visual presets

```
<VISUAL><NAME>numV</NAME><ABS_VAL>3</ABS_VAL></VISUAL>
<VISUAL><NAME>displace_l</NAME><ABS_VAL>0</ABS_VAL></VISUAL>
<VISUAL><NAME>displace_r</NAME><ABS_VAL>0</ABS_VAL></VISUAL>

<VISUAL><NAME>height</NAME>
<SET_TYPE>slave</SET_TYPE><SLAVE_TO>amp</SLAVE_TO><MIN_VAL>0</MIN_VAL><MAX_VAL>-200</MAX_VAL>
</VISUAL>

<VISUAL><NAME>a_add</NAME><ABS_VAL>0</ABS_VAL></VISUAL>
<VISUAL><NAME>b_add</NAME><ABS_VAL>-1</ABS_VAL></VISUAL>
<VISUAL><NAME>o_swell</NAME><ABS_VAL>0</ABS_VAL></VISUAL>
<VISUAL><NAME>b_swell</NAME><ABS_VAL>0</ABS_VAL></VISUAL>
<VISUAL><NAME>a_swell</NAME><ABS_VAL>-1</ABS_VAL></VISUAL>
<VISUAL><NAME>v_swell</NAME><ABS_VAL>-1</ABS_VAL></VISUAL>
```

Figure 9.18 Cube - basicSoft visual presets

```
<VISUAL><NAME>numV</NAME><ABS_VAL>4</ABS_VAL></VISUAL>
<VISUAL><NAME>displace_l</NAME><ABS_VAL>0</ABS_VAL></VISUAL>
<VISUAL><NAME>displace_r</NAME><ABS_VAL>0</ABS_VAL></VISUAL>
<VISUAL><NAME>height</NAME><SET_TYPE>slave</SET_TYPE><SLAVE_TO>amp</SLAVE_TO>
<MIN_VAL>10</MIN_VAL><MAX_VAL>30</MAX_VAL></VISUAL>
<VISUAL><NAME>a_add</NAME><ABS_VAL>-1</ABS_VAL></VISUAL>
<VISUAL><NAME>b_add</NAME><ABS_VAL>0</ABS_VAL></VISUAL>
<VISUAL><NAME>o_swell</NAME><SET_TYPE>slave</SET_TYPE><SLAVE_TO>amp</SLAVE_TO>
<MIN_VAL>0.5</MIN_VAL><MAX_VAL>1.5</MAX_VAL></VISUAL>
<VISUAL><NAME>b_swell</NAME><ABS_VAL>-0.25</ABS_VAL></VISUAL>
<VISUAL><NAME>a_swell</NAME><ABS_VAL>0</ABS_VAL></VISUAL>
<VISUAL><NAME>v_swell</NAME><ABS_VAL>0</ABS_VAL></VISUAL>
<VISUAL><NAME>blankTrack</NAME><ABS_VAL>0</ABS_VAL></VISUAL>
<VISUAL><NAME>trackOffset</NAME>
<SET_TYPE>slave</SET_TYPE><SLAVE_TO>freq</SLAVE_TO><MIN_VAL>0.8</MIN_VAL><MAX_VAL>-0.8</MAX_VAL>

</VISUAL>
```

Nevertheless, both presets make similar use of the envelope functions. In both cases an ASR envelope is used meaning that the visual length of the blip determines the duration of the sustain portion of the sound. In both cases the proportional setting option is used to ensure that the attack forms the majority of the envelope. Both presets use the post decay funtion to create a spring-like return to form after the sound has ceased. 'BasicRing' demonstrates a quite different approach to envelope functions. Here an AR envelope is used meaning that there is no relation between the length of the blip and duration of event, and hence no need for proportional setting. Post decay is not used meaning that graphic and sound ends together.

Another point of note is the use variant blips to achive finer control over parameters. One place where this occurs is in section B where the 'fineSoft' preset is used. Unlike the similar 'basicSoft' preset which assigns a large frequency range to the user B parameter, this preset splits a smaller frequency range equally between the four possible variants. This allows for a much finer control of pitch through the mouse, facilitating subtle melodic inflections.

Figure 9.19 Cube - frequency ranges in fineSoft

```
<SOUND><NAME>freq</NAME>
<SET_TYPE>userB</SET_TYPE><MIN_VAL>0.38</MIN_VAL><MAX_VAL>0.44</MAX_VAL>
<ALT><SET_TYPE>userB</SET_TYPE><MIN_VAL>0.32</MIN_VAL><MAX_VAL>0.38</MAX_VAL></ALT>
<ALT><SET_TYPE>userB</SET_TYPE><MIN_VAL>0.44</MIN_VAL><MAX_VAL>0.5</MAX_VAL></ALT>
<ALT><SET_TYPE>userB</SET_TYPE><MIN_VAL>0.26</MIN_VAL><MAX_VAL>0.32</MAX_VAL></ALT>
</SOUND>
```

Many blips also take advantage of the random and mapping functions of the paramAttribute class. For example, the use of 'hardMappedRandom' in section C3 allows the quick creation of pointillist melodic fragments via stochastic means – all the perfromer need do is click

where they want the blips. The clearest use of the mapping function can be found in section E3. Here the user is able to click in scattered locations in no particular order yet the varying mappings are performed to ensure that the filtersweep effect gradually emerges.

The composition also demonstrates different applications of track and node configurations. Whilst, section E demonstrates a more conventional use of multiple looped tracks , section B demonstrates how the random direction changes of readers at nodes can create alrgorithmically generated melodies of irregularly repeating fragments in orginal and retrograde form. Section C6 shows a statistical control of readers through nodes to effect a gradual transformation. Sections D5 and D6 shows the track and node configurations as a means of visualising process. A key determinant in the arrangement of the tracks and nodes is the need to preserve the integrity of track from section A for the recapitulation in section F6. This means that space can only be inserted on the axis paralell to the track resulting an oblong space.

A final comment should be made about the manipulation of the camera throughout the piece. Here concerns are balanced between the practicalities of adding blips accurately and inserting space, allowing the audience to view the world, and also to draw their attention to particular blip reactions. Whilst for most of the piece the follow mode and camera orientation is toggled freely, certain sections such as section D require a static view throughout as repetitive circular camera movements can be annoying to watch. As has already been mentioned in the implementation description the limited camera view works as a vital component for engaging the viewer's memory via their attempts to build a geography of the space. In this way the changing camera views are instrumental in altering the viewer's conception of the space. The use of follow mode in sections A1 and A2 give the initial false impression of a static image and then that the track might be moving. The change of orientation at section B1 not only establishes the moving camera but reveals the possiblity of track layouts in two dimensions. The camera tilt at section F1 once again reconceptualises the audience understanding of the visual space by implying the possibility of further construction in three dimensions.

Conclusions:

As with *God Over Djinn, Cube with Magic Ribbons* has been widely performed in a wide variety of public and private contexts with an extremely good reception in all circumstances. In public performance an identical setup of laptop at small table to one side of the screen was used. In a similar fashion to *God Over Djinn*, listeners reported an absolute understanding of process combined with a pleasurable disorientation as they tried to comprehend the visual space. In one performance the audience gasped at the initial camera roll in section B1, only to burst into laughter immediately afterwards. Later one of them posited that the spontaneous laughter had been caused by the realisation that they had been so affected by the simple rotation of a bunch of lines. Such a reaction makes perfect sense to me, for even though the rotation itself is simple, the expansion of possibilities it represents is more profound. As with *God Over Djinn*, the successive revelation and breaking of implicit constants are the fundamental means of formal progression.

However, the variety of blips in *Cube with Magic Ribbons* creates a richer world than in *God Over Djinn*. As has been shown, blip types are able to imply their own symbolic systems, physical systems and external referents. The cumulative effect of their presentation over the course of the piece in various combinations and permutations, could be described as the construction of something akin to an abstract language in the way that Cardew might have intended for *Treatise*. Furthermore the geographical layout allows us to view the history of its construction – a moment where a particular blip was first encountered, or loop where there was a dense textural build up. Nevertheless, this construction in no way feels complete. As of yet, blips remain fixed components of the music whose development progresses through their inter-combination. One could imagine a situation where series of presets are used to create sequences where blips' behaviours and appearance themselves are developed in the same way that Cardew develops his graphic symbols in *Treatise*.

However, although the current draw objects provide a significant amount of flexibility in allowing the user to create individual blips, they have not yet reached a sufficient degree of functionality to facilitate such development. Nevertheless, the flexible system of SoundCircuit's programmable presets and modular draw objects should allow this to be worked towards without major structural changes. Indeed, unlike with SoundNest, these same features evidently allow for potential further compositions by others and myself.

Towards this end, the first action would be to create many more draw objects and explore their potential audio-visual relationships through the preset system.

There are many other possibilities for the extension of SoundCircuit. Readers could also react when they intersect with blips. This could range from graphical effects such as shaking or spinning to more functional behaviours such changing direction, slowing down, or jumping to a different position. Nodes could similarly be extended, perhaps allowing readers to rebound but also to allow a single node to simultaneously act as corner for two pairs of tracks. A practical addition would be to allow SoundCircuit to deal with midi values rather than normalised ones for pitch. This would require implementing pitch and tuning functions within the interface and would allow for advanced versions of mapped and random distributions of pitch parameters as well as more intuitive setting of preset parameters. A more ambitious development might be the expansion of the surface beyond the z plane. This could happen in a number of ways. One might be to allow the surface to become contoured which could act as a way of adding space without disturbing other tracks. Another way might be to have multiple surfaces, perhaps animating in such a way as to imply the surface of a cube. Many have suggested that SoundCircuit could be released as a commercial iOS app. This would require a significant amount of extra development in terms of achieving an intuitive GUI.

Towards these ends it has been commented that I should consider implementing SoundCircuit as a touch interface. Although such an approach would certainly be more fashionable, I'm not convinced that in this case it would be to the interface's advantage as a live performance tool. In the first place, touch would be considerably less accurate when it came to the placement of blips and tracks. Most probably the user would be required to zoom in and out in order to achieve accurate placement adding an unnecessary stage to the user interaction, which also is unpleasant to the viewer. There would be little to gain from the gestural and fine control that such interfaces offer - the audience is focussed on the screen not the gestural movements of the performer. Furthermore, any apparent advantage gained through not having the mouse pointer would be lost through the need for graphical buttons, sliders, and menus to replace key commands. Indeed, in the current version, the mouse pointer plays the key role of concisely embodying the agency of the performer in a near universally understood form. It is reasonably safe to assume that an audience will have experience of modern computer software hence an implicit understanding of mouse

interaction, and so, just as command line operates in live-coding, the mouse acts as the guarantor of performer presence.

But there are also other levels of agency within *Cube with Magic Ribbons* and its performance. Audiences with even a limited degree of computer literacy will deduce from the projection that some kind of software preparation has had to occur prior to performance and whilst watching many will consider how the work has been realised. Just as an audience might want to know whether a singer writes their own songs so as to gain an understanding of the balance of authorship, so my audiences often want to know whether I designed the software. On a lower level than the live performer is the relationship between the rectangular readers and the objects they pass over. It becomes pretty clear that the readers are responsible for causing those objects to react. The agency of the readers is further reinforced through the camera following them, a gesture which by reference to FPS gaming and similar cinema techniques implies their autonomy, personifying them to some degree. Finally there are the blips, of which the varied directions of causality between sound and visual have been discussed previously.

In using black on white lines I had tried to imply a notational quality reminiscent of Cardew's *Treatise.* However, SoundCircuit's notational qualities extend beyond the aesthetic and into the functional. Its graphical world is not merely a decorative way of representing pre-existent sound worlds, but it is the framework in which the music is conceived. The placement of blips serves as a way to order events in time, their graphical form allows their identification and communicates their parametrical properties. The interconnection of tracks allows for macro organisation. As is the case with staff notation, the graphical instructions for the sounds to be made also functions as their representation, thus allowing the conceptualisation of events outside of time. Interestingly, although the breaking of proportional representation might be novel in terms of DAWs, it is hardly the case in terms of notation. The same can be said of many other aspects. In experimental notations, one can certainly find cases of indeterminate form and even topological organisation. Where SoundCircuit differs significantly is in its liveness – the creation of the notation is also the performance. In this way the notation manages to be simultaneously past, future, and present.

10. What Is Life ? (SoundLens)

Video, code, and score: http://www.simonkatan.co.uk/phd/whatislife.html

Overview:

What Is Life ? (2012) is a computer visual and synthesised sound composition for live performance. The piece uses a custom visual sequencer called SoundLens which comprises an OpenFrameworks visual interface and an OSC controlled SuperCollider patch. These combine to imply an analogue between a simulated visual focal effect, reminiscent of a camera lens or microscope, and auditory depth of field. Into this world are placed multiple copies of a sound producing mechanical object. Inspired by a coincidental similarity between certain arrangements of these objects and the double helix structure of DNA, the piece takes its title from the name of Erwin Shrodinger's book in which he develops the concept of a complex molecule with the genetic code for living organisms. Nevertheless, there is a deeper connection between title and piece in that it progresses solely through the copying and mutation of its own material.

Aside from the aforementioned, focal analogues, the construction of this interface, was also motivated by a desire, following my work with SoundNest and SoundCircuit, to develop software orientated around metrically relating attacks. Although SoundCircuit was undoubtedly modifiable towards these ends, I was interested in finding an alternative paradigm to the mapping of temporal to linear position. The solution that presented itself was a gravitational system, using rotating objects constrained by prismatic joints. In such a system the time of attack becomes dependent on multiple parameters the length of the joint, and the phase and speed of rotation. The combination of parameters allows for a seemingly irrational visual relationship between rotational phases of objects to result in a metrical sonic relationship. I considered that such a dissonance between audio and visual worlds might be a fruitful area for exploration.

A final consideration was that I wanted to find an approach for formal development other than the gradual build up of material that happens in SoundCircuit. Initially I considered a large, randomly generated, database of objects on which various searches could be

performed according to the parametric data. Rather than adding or deleting objects, the performer would slowly progress the piece by bringing the results of such searches in and out of focus, creating kaleidoscopic transformations of material. However, such an interface left few functions for the mouse to control, and this being my main means of communicating the agency of the performer, I rejected the database approach and opted for the addition of various copy functions which transform the objects in the process.

Implementation:

As with the previous two visual sequencer works, I have divided the implementation into interface design under the title SoundLens, and composition realisation under the title *What Is Life ?* However, in this case, the SuperCollider implementation is considered part of the interface as it has specialised synthDefs and osc responder nodes for realising the aural focus effect.

SoundLens Implementation:

Mindful of the proliferation of code that had occurred in SoundCircuit and the consequential difficulties involved in refactoring when changes were made according to compositional priorities, I opted for a modular design approach involving the concurrent development of multiple programs to test out ideas and solve problems. These were eventually combined into a single package once the most significant details had been determined. Such a technique undoubtedly improved decoupling in the final package, allowing for greater flexibility when further changes needed to be made. However, it also led to a degree of redundancy in the class structure. For example, in generic classes such as mappingEngine and distributionEngine which are only used for a single function (Figure 10.1). Nevertheless, such classes may come in useful in future expansions of the interface. A second difference in the design process was the omission of a user interface for the design of presets which instead are hardcoded within a function in the OpenFrameworks testApp. This is partly due to a continuing lack of certainty as to whether the current configuration is ideal, and the pragmatic observation that the application will not be released within the timeframe of this research.

Figure 10.1 SoundLens - class structure

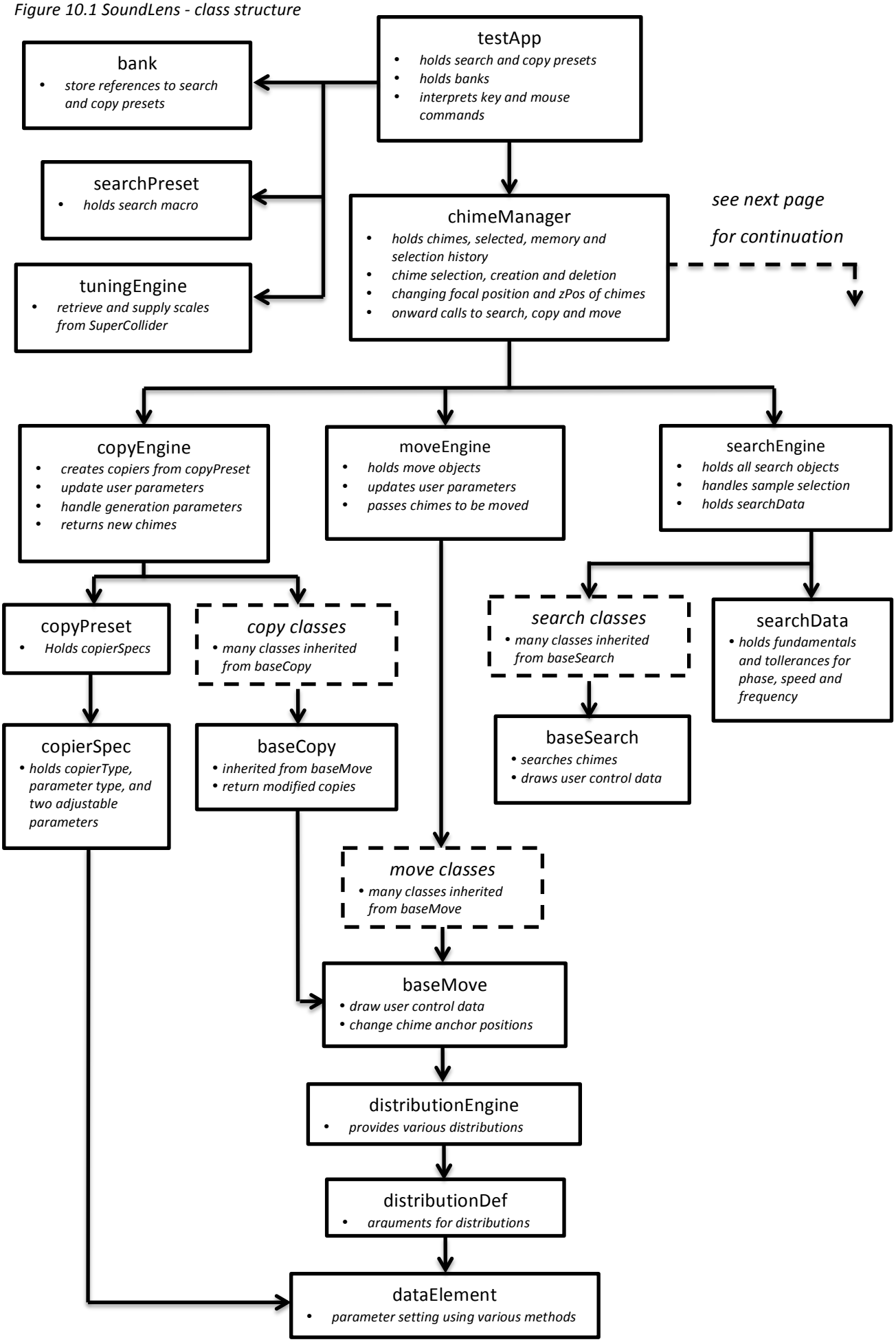
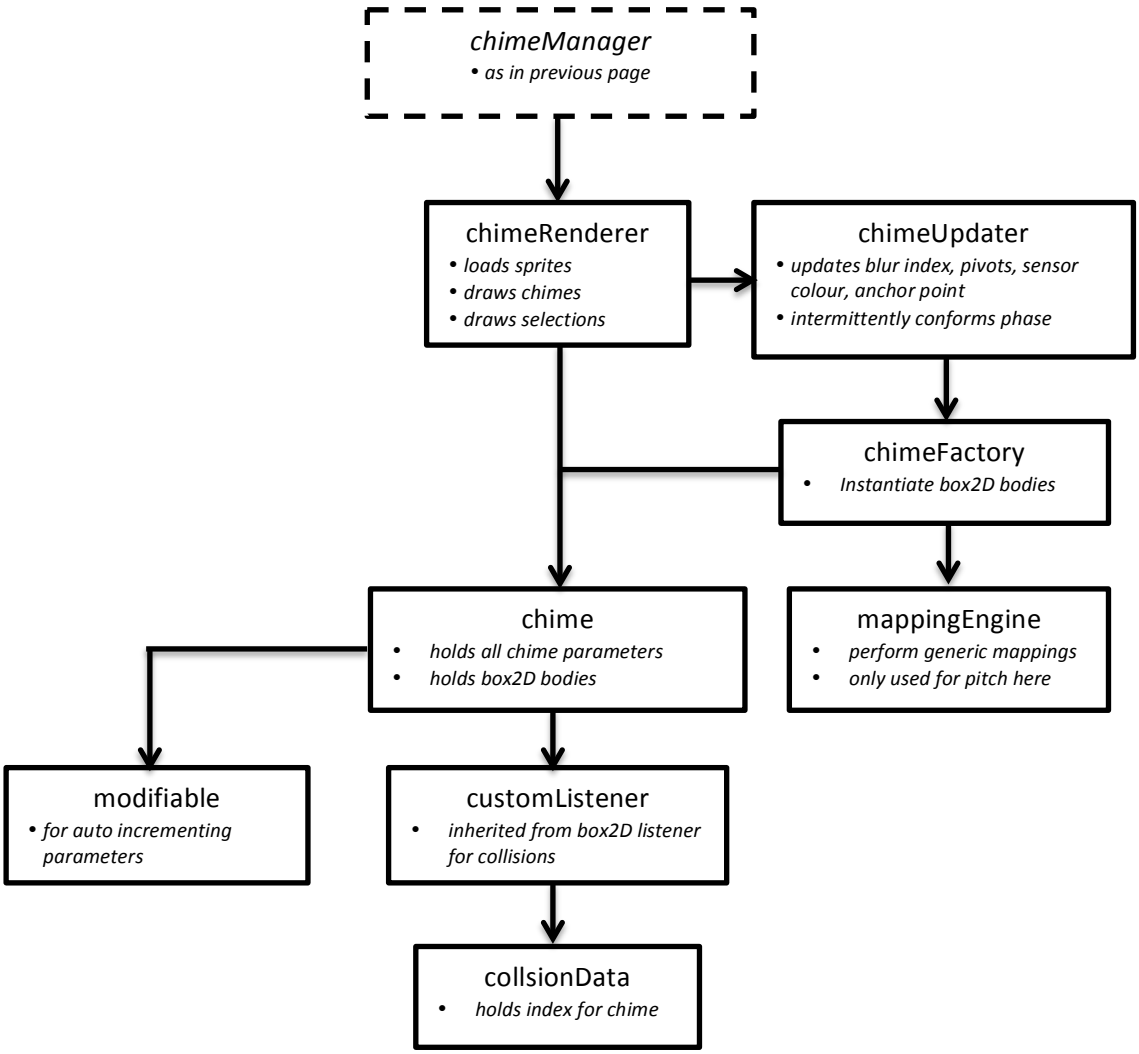


Figure 10.2 SoundLens - class structure continued ...



Once again the interface is largely mouse based with a certain number of commands assigned to keystrokes. As in other projects, the UserA variable referring to the drag distance of the mouse, and the UserB variable referring to the drag angle in relation to the vertical axis, are extensively used. Similarly, the organisation of keystrokes and click commands was developed empirically through iterative practice. However, this time the involved nature of the tasks to be carried out require more complex keystroke and mouse combinations, for example in operating search macros where the function of the left mouse click changes sequentially. Some of the more subtle features are those that involve the saving and combining of groups of chimes, or starting a search from a set of previously selected chimes. Although it might have been easier to arrange these more complex features on a second screen such as an iPad, I preferred the discipline of maintaining a single screen so that the actions of the performer are less obscured from the audience.

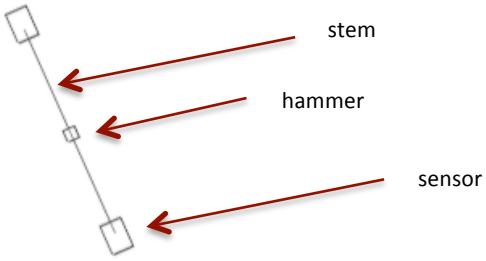
Figure 10.3 SoundLens - feature list

Feature	Key	Mode	Mouse	Notes
Move focal point	'z' , 'x'	copy		
Move towards focal point	'a'	copy		chimes must be selected
Move away from focal point	's'	copy		chimes must be selected
Equalise z positions	'z'	copy		multiple chimes must be selected. Moves z positions of a group of chimes towards an average point
Move mode	hold 'm'	copy		
Adjust Pivots	hold 'n'	copy	left click and drag (userB for param type, userA for value)	all four pivot parameters set this way
Search mode (fresh set)	hold space	copy		
Search mode (existing set)	hold ctrl + space	copy		multiple chimes must be selected
Save to memory	shift + '1','2' … '0'	copy		chimes must be selected
Recall memory	'1','2' … '0'	copy		
Recall history	'[', ']'	copy		cycles backwards and forwards through previous selections of chimes
Combine with memory	TAB + '1', '2' …'0'	copy		chimes must be selected. Only makes a current selection, save function must be called again to assign to memory
Invert selection	'i'	any		
Delete invisible chimes	Backspace	copy		
Change copy preset	'up', 'down'	copy		
Change search preset	'up', 'down'	search		
Change move preset	'up', 'down'	search		
Change bank	'left' ,'right'	copy		banks limit number of copy and search options for more navigable menus
Change sieve	'r', 't'	any		
Move chimes		move	right click and drag to adjust parameters	depends on which move type is selected
Copy chimes (fixed position)		copy	left click and drag to adjust parameters	new chimes are in the same position as the old ones
Copy chimes (new position)		copy	right click and drag to adjust parameters	new chimes drift to where the mouse was clicked
Inclusive search		search	left click and drag to adjust parameters (each click is the next search in the macro)	chimes matching the search criteria will be included in the final results
Exclusive search		search	right click and drag to adjust parameters (each click is the next search in the macro)	chimes matching the search criteria will be excluded from the final results

The starting point for the interface design was a collection of Box2D bodies called a chime comprising a rotating kinematic body known as a stem with rectangular dynamic bodies known as sensors attached by weld joints at either end. A dynamic body called a hammer is attached to the centre stem with a prismatic joint constraining movement to the vector

between the two sensors. Collisions between hammer and sensors cause the sensor to activate, lighting with a colour and triggering a note from SuperCollider. For the purposes of simplifying search procedures, the properties of both sensors are matched. As with SoundNest, the decay of the light happens in accordance with the decay of the triggered note, although pitch is mapped to the height of the sensor. Once again for reasons of reducing CPU load, each chime is contained within in its own Box2D world.

Figure 10.4 SoundLens - annotated chime

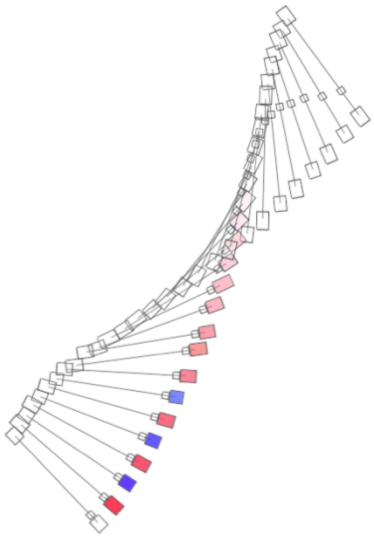


In addition to this, each chime holds a number of accessible variables comprising phase, speed, length, frequency, colour, and decay. Phase describes the rotational phase of the stem and hence determines the timing of the attack points, and is set as a fraction of 180 degrees – the matching of sensors means that greater rotations have equivalents within this range. Provided that their lengths and rotational speeds are the same, two chimes of identical phase will trigger coincidental attack points whilst two chimes whose phases create an absolute difference of 90 degrees will make a regular pulse. Of course, providing that the common interval is a divisor of 180, faster pulses can be achieved using multiple chimes spread by smaller phase intervals. However, in order that the user can accurately set such intervals the phase of chimes has to be conformed to the others on instantiation. This happens by back calculating what the current rotation of the new chime would be if it had been created at the start of the program. The same method has to be called intermittently on each chime in order to compensate for accumulative inaccuracies resulting from discrepancies between my method of back calculation and Box2D's method.

Although the physical behaviour of chimes is unrelated to their global position, panning is mapped to the horizontal axis. The global position is described by its anchor point and adjusted via a series of features which for the most part operate on groups of chimes, arranging them in various ways according to their parameters. The most simple of these are 'gather' and 'shift', the headings of which should amply describe their behaviour. However,

there are a number of other features which spread a group of chimes linearly. These firstly sort a group of chimes by a given parameter and then arrange the chimes along an axis selected by the user. A particularly attractive application of this is when a group of chimes of regularly incrementing phases is spread by that value. It is this arrangement, reminiscent of DNA's structure, which is mentioned in the introduction (Figure 10.5). However, aside from pure visual aesthetical concerns, these features also aid the user in selecting chimes to copy and highlighting particular threads of a texture.

Figure 10.5 SoundLens- spreadByPhase



In addition to this kind of adjustment, position can also be manipulated by a number of variables that describe a set of connected rotating struts that displace the chime from the anchor point. Although the struts are displayed in Figure 10.6 they are neither rendered on screen nor instantiated in Box2D. Instead, the positions of the pivots between struts are sequentially calculated by a function taking the current rotation and phase as arguments. This function expresses the current rotation of the chime across the various pivots whilst maintaining the eventual orientation of the chime. The net result is to allow for different spatial configurations of chimes which can be used as a way of drawing attention to speed, phase or pitch relationships between groups of chimes. For example, Figure 10.7 shows three groups of chimes of matching phase and pitch values with speeds related by the ratio 6:3:2. By spreading them out via an identical pivot configuration the sonic phasing of the attack points is emphasised in the visual sphere.

Figure 10.6 SoundLens - pivots

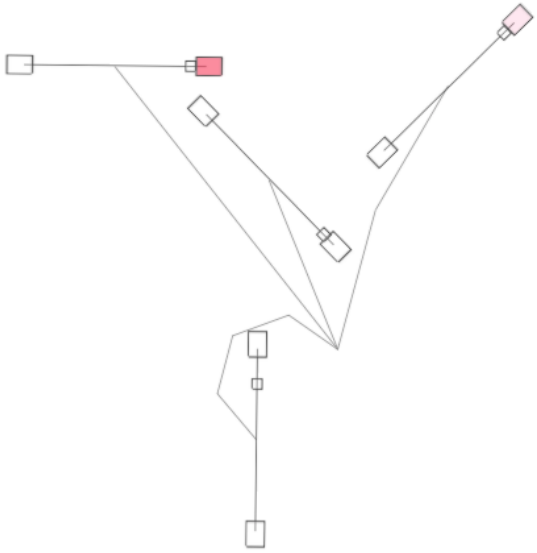
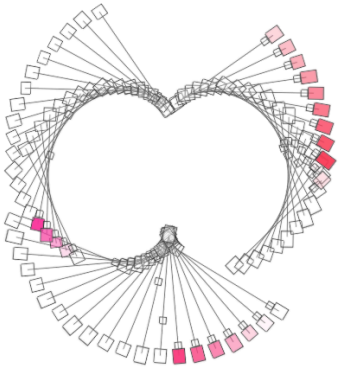
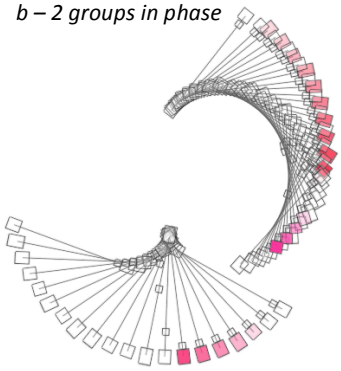


Figure 10.7 SoundLens - pivots expressing speed ratios

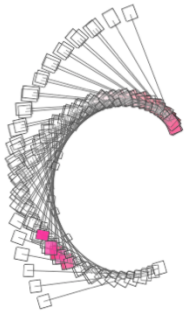
a – 0 groups in phase



b – 2 groups in phase



c – 3 groups come into phase



As has already been mentioned, chimes can only be created through copying other ones. However, truly identical copies of chimes are rarely made. Instead some parametrical mutation, adjustable by the user, occurs during the copying process. The nature of these mutations is determined by the selected copyPreset. These presets will eventually be user defined through an XML interface or GUI but are currently hard coded within a function in

the testApp. A preset combines various parametric mutations into a single copying process. These mutations are contained within a set of classes deriving from a base copier. As can be seen from Figure 10.8, these are organised by the type of mutation and not the parameter to be manipulated which is instead taken as an argument in the class' constructor. Such a structure means that, with just a few generic copier classes, an extensive range of copyPresets manipulating various parameters in different ways is available to the user. A final feature to note is that the parameters of the individual copiers to be adjusted by user controls are specified in the preset and not the class. This allows a flexible, context dependent, approach to user input. For example, in two presets using the mutate class one might set deviation via the userA parameter and the other might set it via a vertical mapping. A final point to note is the effect of copying on the colour of the lit sensors. These reflect the generation of a chime as for each copy the hue is incremented. When groups of chimes are copied, the oldest chime in the group is taken as the generation for all the copies. In this way one observes gradual shifts in colour as a performance progresses.

Figure 10.8 SoundLens - copy classes

Class Name	Arguments	Descritption
Transpose	amount	Add the amount to the parameter. Wrap/constrain if necessary.
Mutate	range, deviation	Add or subtract a randomly chosen amount using a normal distribution.
Arrange	type (rotate, shuffle, reverse, sort - asc) num	Perform the permutation on the parameter values leaving the others intact
Invert		based on the highest and lowest values in the given selection
Resize	proportion	expand the range equally in either direction
Sieve	offset	snap the new values to the nearest scale value. (only for freq)

An equally significant area of functionality is that of searching through and selecting groups of chimes. The structure here is similar to that of copiers; search classes are generic and refer to type of search as opposed to particular parameters, and different types of searches can be combined into presets. A key difference is that whereas combined copy functions are all executed simultaneously, search functions are performed sequentially in a macro with the results from the previous function being fed into the current one. In the first place, this method allows the user to predefine commonly used sequences of searches, preventing cumbersome switching between search functions. As the user may discontinue the search at any stage in the macro, larger search sequences can encapsulate shorter ones, further reducing the number of presets for the user to look through. The sequential aspect also

affords greater control over parameter adjustments set for the relevant search at each stage of the macro by allowing the user to respond to the results from the previous stage. The various search classes and their properties can be viewed in Figure 10.9.

Figure 10.9 SoundLens - search classes

Class Name	Parameters	Arguments	Description
Position	position	search rectangle	exclude outside rectangle
Unique	phase, speed, freq,	minDist	exclude duplicate values within the minDist
BiPassFilter	phase, speed, length, freq, decay	band, bandwidth	exclude outside the band
SinglePoleFilter	phase, speed, length, freq, decay	highPass/lowPass cutOff	exclude all above or below cutOff
Match	phase, speed, length, freq, decay	sample chime tolerance	exclude all that don't match the sample within the tolerance
Fundamental	phase, speed, freq	sample chime fundamental tolerance	exclude all that when divided by the fundamental return a remainder which is greater than the tolerance
Quant	phase, speed, freq	fundamental tolerance offset mul	exclude certain multiples of the fundamental
Sieve	freq	tolerance offset	exclude all that don't match any degree of the scale within the tolerance

One extra feature relevant to copying and searching is the pitch sieve. This uses a scale as a grid by which to search and copy notes. The feature taps into SuperCollider's extensive library of tempered and microtonal scales by obtaining them through OSC requests. When used in the context of searches, a tolerance parameter determines how closely the notes have to match. When used in the context of copying, the pitch parameter simply snaps to the nearest value.

A significant area that has not been commented on yet is the focal functioning of the interface. In the visual sphere this is achieved through Gaussian blurring – a process which blends the colour of a pixel with its neighbours in proportion with their distance from the pixel. Each chime has a parameter called zPos governing an imagined z coordinate which remains unrealised in terms of chime's distance from the camera. Instead the camera has a focal point in the same coordinate system, and the distance between the chime's zPos and the camera's focal point is used to determine the amount of blurring to apply to the graphic. One way of changing the focus of chimes is through adjusting the focal point, thereby simultaneously adjusting the focus of all chimes in the manner of a camera lens. A second method is to move the zPos of an individual or group of chimes towards or away from the focal point, thereby only changing the focus for that group. On reaching the maximum blur

amount, a chime, though still searchable, is no longer rendered on screen and becomes inaudible. Chimes in such a position can be permanently deleted using the backspace key. Although SoundLens' focal space is bounded by minimum and maximum focal points of minus one and plus one, a wrapping function makes it always possible to move chimes far enough away from the current focal point such that it reaches maximum blur and can be deleted.

As both the focal point and each chime's zPos are independently adjustable, it is quite possible to have multiple chimes with different amounts of blurring. With the requirement of having hundreds of simultaneously visible chimes, such a situation caused some difficulty with regards to GPU overheads. Initial attempts involved drawing each chime into its own off screen buffer and then rendering it onscreen with the required amount of blur. However, this method resulted in unsustainable drops in frame rate, due to a significant amount of overhead incurred by opening and closing buffers for each object. After investigating some complex implementations of variable depth shaders in GLSL which would use a single off screen buffer for the whole scene, I settled on an altogether more simple and optimal solution of using sprites. These are collections of pre-rendered images indexed according to the amount of blur. To prevent over extensive image resizing there are small, medium, and large sprites for each type of object. The sprites themselves are made in a separate program which incrementally blurs and labels the images and organises them into folders. For each object SoundLens simply selects the relevant image for rendering according to the type, blur amount, and size and then renders it to the screen with the appropriate amount of rotation and resizing. This method is able to work as there are only a few types of objects and all the objects are rigid bodies.

As with SoundNest, the chosen sound is a simple sine tone. Once again the reason for choosing such a simple sound is that more complex spectra will result via the inter combination of sine tones. However, in this case there is also an aural blurring to match the Gaussian blurring in the visuals. The technique, which uses Supercollider's PV_MagSmooth object top smooth out the magnitude of FFT bins over time using recursive averaging, was suggested by Dan Stowell after I showed him a prototype of the graphical interface. Using this, the sound implementation continually writes to ten buffers which are smoothed to varying degrees by a synth using the PV_MagSmooth object. Attacks are then assigned to buffers according to the blur amount of the associated chime in the graphics application. Amplitude is also mapped to the focal distance which reinforces the sense of foreground

and background and is also useful in preventing distortions when too much signal is put through the heavily smoothed buffers. The overall effect is a subtle and incremental softening and sharpening as chimes move in and out of focus.

What is Life ? Implementation:

What is Life ? has a broadly determinate macro-level structure but allows significantly more scope for real time decision making than *God Over Djinn* or *Cube With Magic Ribbons*. A second difference to note is that where as these other two compositions served more or less as an exposition of their respective interfaces, this composition only explores a fraction of *SoundLens'* potential. This is in part due to the sheer number of possibilities offered through inter combination of copy and search modules, but also due to the slower, contemplative mode of development that the interface implies. The descriptive score of the piece (Appendix D and online) is idealised in terms of feature use with incidental use of memory functions and searches being ignored for the sake of clarity. Once again the following paragraphs make reference to this in order to comment on some key aspects of the work, but also suggest other aspects that might have been explored.

The work demonstrates several uses of focal features. The 'move towards focal point' feature is the most extensively used and largely acts as a means of fading in new material. Indeed, as the interface always creates new chimes at a maximum distance from the focal point, the use of this feature is requisite to the copying of any chimes. The slow rate at which chimes can be brought into focus, allows for subtle additions to the texture, as occurs in section B2, but also places a ceiling on the rate at which development can occur through incremental textural build up. Nevertheless, the varying degrees to which objects are brought into focus adds an extra dimension to the texture, for example, allowing the accenting of notes in section A or creation of echo like effects when copies of groups of chimes are made in sections C2 to C4.

Furthermore, the ceiling can be circumvented in couple of different ways. Firstly by fading in larger numbers of chimes at any given time as happens in the textural build up in section D1, and secondly by using the 'move focal point' feature to effect a rapid shift between textures as happens in section C1. Such a section implies a further area of exploration. One could imagine developing material through the cyclic manipulation of multiple clearly defined layers. This would work by performing operations on the least focussed layer, moving the

focal point to reveal the changes, subsequently working on the new least focussed layer, and so on.

A second aspect that is demonstrated by the composition is the interface's capacity precise metrical control. This is most apparent in section A where the pattern of entries is created by repeatedly copy transposing by a regular a phase interval which is not a divisor of a single phase. As the transpositions wrap smaller fundamental phases are gradually implied. I feel the effect to be something similar to the metrically ambiguous introductions one sometimes finds in salsa and electronic dance music. A different manipulation is evident in sections C1 and D2 where the phaseFund search is used to bring different meters into focus. A somewhat confusing aspect of the interface here is that where as in the phase transposition of section A phase is referred to as a decimal fraction of a single phase, the phaseFund search uses integer divisions of the phase. Such divisions are easier to read in the search features – compare 1/17 to 0.0588235 – but in the copy features, they would exclude the possibility of using transpositions which aren't divisors.

A further manipulation is the varying of rotational speed that occurs in section E4. When applied to subgroups of chimes these effect rhythmic diminutions and augmentations. However, in section E4 the effect is somewhat absorbed into the ongoing texture and it is doubtful that relationships such as posited in Figure 10.7 will be realised by the user, neither is the use of pivot features able to aid the listener in discerning them. One could imagine a different composition developed from the starting point of rationally related speed relationships which might be more effective in this regard. However, variable speeds also change the meaning of the phase parameter as chimes of the same phase now have differing attack points. It is for this reason that chimes of varying stem lengths are avoided. Nevertheless, longer stems could be used as a different way of transposing phases, increasing the complexity of the audio-visual relationships. The addition of a timestamp feature that records the system clock time of a chimes last attack might be useful here. A further aspect that was not explored was the use of filter search classes on phase. One way in which these might be used is in the construction of new sequences from old ones by the complementary intersection of phase segments.

The composition only makes apparent the precise control of tuning from section D3 when the minor pentatonic tonality emerges. From that point on, tonality is enforced through the use of the sieve class in copyPresets. In earlier sections of the piece, where the sieve isn't

used, the tuning is equally tempered quartertones. In section A, the difficulty in controlling sequential frequency transpositions through the vertical axis mapping is deliberately imposed. To some degree, the lack of precise control is fruitful in developing rich and unexpected melodic textures. However, one could also imagine building a texture in a more controlled manner using the same sieve copy classes but changing the scale or offset as the texture is built. In this way one could imagine strands of tonalities threaded through the texture which could then be explored using the 'move focal point' feature. Indeed, that only one of the one hundred and eight available scales is used gives some indication of how much further this area could be explored. Another pitch control feature which is used is the 'arrange copy' feature, for example, in section C3 to create variations and in sections C4 and E2 where it is used to create scalic passages. Nevertheless, there are plenty of uses of the feature that remain unexplored, for example the gradual transformation of a sequence of sounds by gradually arranging segments of a shuffled sequence into scalic patterns eventually arriving at a single descending passage, or expositions based on the manipulations of a short root sequence.

The features controlling movement are evident throughout the work, their use motivated by a combination of functional, perceptual, and visual aesthetical concerns. For example, the use of the 'spread preserve' feature in section B1 lays bare the phase and colour relationships to the audience, but also allows for the selection and copying of subgroups based on their order of creation. The use of 'shift' in C2 not only allows for more easy selection of subgroups later on in the piece but also changes the audience's cognition of the sequence by bringing to their attention the particular chimes that have been moved. Nevertheless, the transformations effected by the 'move chimes' feature have little aural effect other than some subtle changes to panning. However, the dramatic visual changes serve to articulate the macro form of the piece, with visual transformations normally preceding new types of sonic transformation and each section having a different visual as well as aural composition. One area of development is in the use of pivots which is certainly underrepresented in light of the formulations that had been discovered during pre-composition. When they are eventually used in section F1, their use is more decorative than functional. Aside from the aforementioned time constraints, their underuse can also be accounted for by the cumbersome nature of the four parameter interface for adjusting the pivots, and the difficulty in selecting chimes by position once they are moving by pivot. Nevertheless, these features offer some great potential in representing speed and phase relationships which could perhaps be explored in a dedicated composition.

Conclusions:

What Is Life ? is as of yet unperformed in public so it is not possible to comment as to audience reception. However, from my own observations, a number of differences in what effects the piece and the interface might produce are apparent. The increased emphasis on parametric manipulation seems to lead to a reduction in the variety and depth of audio-visual relations. Essentially the fundamental hammer sensor, and focal relationships are all there is. Furthermore, despite the extent of visual activity and contrast within the work, the enclosing space is essentially static. The revelatory aspect present in *God Over Djinn* and *Cube With Magic Ribbons* is therefore less strongly articulated. Indeed, one might conclude that, with the use of visual features of no consequent aural effec, that the audio-visual coupling is looser than in the other pieces.

However, I would argue that this is not the case; the piece is simply more orientated away from the progressive and towards the contemplative, and needs to be viewed from this perspective. As in many minimal works, the audience is given longer time frames in which to attune themselves to the complex patterns and subsequently begin to perceive their own sub-patterns within them. In this case however, there is the added challenge of attributing individual sonic attacks to corresponding hammer strikes in the visual sphere. As the dense textures build, one finds that if one focuses on an individual or small group of chimes a connection can be made, but the perception of the whole is impossible.

Such an approach chimes well with SoundLens' analogue between aural and visual focus. One could view the visual focal and spatial aspects of the piece as my intervention into the perceptual processes of the audience. By repositioning the chimes or shifting the focus I am able to direct their attention towards certain details, or disturb previously stable configurations. In retrospect, perhaps the most fruitful areas are where this happens with little or no sonic change. *What Is Life?* uses devices such as textural build up and well ordered visual composition to balance slowly evolving forms with more progressive elements. However, I could imagine a different composition which disposes entirely with attempts at linear progression, instead consisting of series of sonically static scenes which are visually manipulated in order to draw the audience to and from different details within the texture.

Such speculation leads me to reserve judgement as to whether the current configuration of SoundLens is the most appropriate. For example, the exclusive use of copying functions imply a certain type of development which relies on build up of elements rather than their transformation. In this regard the extension of the interface to include features which gradually transform parameters over time, in the way that position is already altered would be desirable. The inclusion of such parameters might call into question the necessity of the copy function itself, one might perhaps opt for a fixed number of chimes to be brought in and out of focus and manipulated. The second questionable aspect is whether the constrictions imposed by the use of the mouse as primary interface are warranted by its role in reflecting the agency of the performer. Many of the most significant actions such as moving the chimes and performing searches have a more obtuse relationship to the mouse than the creation of blips in *Cube With Magic Ribbons.* One could imagine an alternate interface where these actions are transferred to ancillary controls, allowing for more sophisticated behaviour. The focal features themselves might be controlled by large manual jog dials thus providing an alternative non-screen based projection of performer agency. Luckily the modular design of the interface would allow such changes to be made without necessitating extensive code modification. I nevertheless, reserve judgement on such matters until the composition has had a good public airing.

11. Brainer

Overview:

Brainer is a composer/performer collective between Cimeon Ellerton, Luke Fraser, and myself. Inspired by artists such as Jonothan Burrows and Matteo Fargion, Xavier Le Roy, and The Bohman Brothers, the group concerns itself with a form of performance drawing equally from the fields of music, dance, and theatre. The result is a form of abstract comic theatre dealing with symbolic and gestural relations between sounds, movements, words, and images. As such, the group's work seems wholly pertinent to this thesis, and has undoubtedly influenced the previously described works. Brainer collaborates in a variety of ways, according to the particular demands of the piece. These can range from communal blank slate devising to an individual composing a fully scored composition, which the rest of the group performs. Each of the four compositions presented here have a different mode of collaboration as well as contrasting uses of media and approaches to creating relationships between them.

Do It Again:

video: http://www.simonkatan.co.uk/phd/doitagain.html

Do It Again (16.07.2010) was commissioned by SoundWaves festival in 2010 and has subsequently been performed at Borealis Festival Norway, and the Deptford X Festival Lewisham. For the composition of the piece I suggested a three way devising process around a few basic rules. Firstly we should not only perform the piece without recourse to notation, but also attempt to compose it in the same manner. This rule was in response to our dissatisfaction with previous works where scored composition had produced overly complex movement and vocal patterns in which the results achieved did not necessarily warrant the effort expended on realising the notation. Secondly the piece was to be performed by the three of us sitting at a table approximately 1m80 by 1m with no other instruments or props. Thirdly, each member was to come to the first rehearsal with a single gesture, which didn't necessarily have to involve movement or sound, but should somehow relate to the table. Finally the member who brought the gesture shouldn't make any suggestions about its use until both of the other members had significantly developed the idea.

My gesture was to slowly and repeatedly bang the table with my hand. Cimeon's was to let his head fall to the table and groan. Luke characteristically rejected my imperative of relating gestures to the table, opting instead for silent and slow conducting of downbeats. The piece was developed in regular rehearsals over a period of months through experimentation, observation, and extensive discourse. Videoing and watching back rehearsals was an essential part of the process. Luke Fraser's short documentaries *Do It Again* and *Do It Again Again* collate the resultant footage giving some idea of how this process worked. Through development of the three gestures, others such as rhythmic clapping, applauding, circular arm movements, and lunging across the table, suggested themselves. These served to form inter-connections between the initially disparate material, resulting in a quirky form that nonetheless has an organic coherence to it - a quality that I have noticed in a lot of devised theatre - the work of Simon McBurney springs to mind.

Though movement clearly takes precedence over sound in the piece, all the gestures relate to musical performance. *Do It Again* could equally be viewed as a movement piece about the performance of music, or, in a similar way to Burrows and Farignon's *Both Sitting Duets,* as a piece of music where sound has been replaced with movements and gestures. Perhaps, fairer would be to define it as a hybrid form as opposed to either of the above. In any case the band is quite comfortable with any of these definitions as the ambiguity of media is quite deliberate.

The piece begins by using the ambiguity of the performance situation to toy with audience expectations - what might three men sitting at a table in a musical performance be about to do? The over-repetition of slow and deliberate table-banging gestures combined with extended periods of silence, only amplifies the effect. The intention is to create the impression, and perhaps fear, in the mind of the audience that this gesture will form the entirety of the work. However, as the piece continues, the pace gradually increases, as does the teleological development of gestures.

In retrospect, I feel that this development could have gone further, for example by experimenting with shortening some of the movement gestures to achieve greater density of material and faster and more complex interplay, or by developing the relation between sound producing gestures and the sonic outcomes. That this didn't happen is partly a reflection of the fifteen-minute constraint of the commission, but also reflective of the process which we used to create the piece. Whilst the slow gesture by gesture construction

that the devising process encourages creates strong and pragmatic connections with the material, it appears to constrain the formal detail that can be included in a work; perhaps with further practice such constraints could be mitigated.

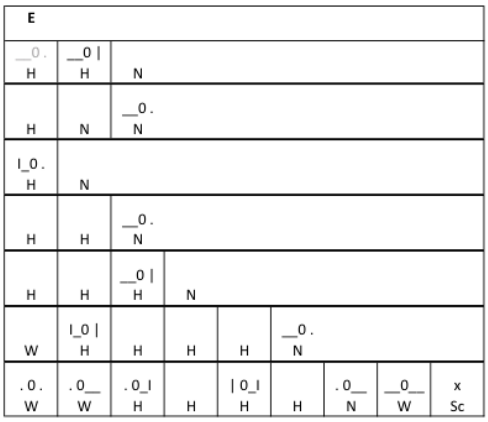
Tool Box Song

video and score: http://www.simonkatan.co.uk/phd/toolboxsong.html

Tool Box Song (14.03.2012) is a vocal and movement piece based on five words 'hammer', 'spanner', 'wrench', 'nail', and 'screw.' The piece was composed by Cimeon Ellerton and myself, and using a more conventional score-based compositional process. The original score was composed by Cimeon in 2008 and performed at Kings Place as part of SPNM's Sound Source series. However, the group was dissatisfied with the formal development of the piece, especially the use of a broken canon form in the second half which was not only felt to be unsuitable to the material, but also imposed a level of difficulty in performance which was not justified by the audio visual results. Having long had the intention to rework the piece, a commissioned set of vignettes from Borealis Festival in 2012 provided a good opportunity for me to revisit the material as part of the commission. Aside from wanting to improve the formal development of the work, the group also wanted to perform it from memory, as it was felt that the presence of sheet music on stands diminished the visual effect of the corresponding arm movements. The final requirement on the reworking was that, like *Do It Again*, the piece should be performed from a seated position at a table in order to match the rest of the set.

The original piece had been scored using standard notation with symbols for movement drawn above the notes. However, given the repetitiveness of the material and that each of the five words operates as a fixed unit with no variation in pitch or rhythm throughout the piece, I determined that the notation could be condensed into a single symbol for each unit with a separate system of notation for movement. This eased the task of memorisation by reducing many pages of music into just two sheets and allowing the performer to focus only on the varying parameters.

Figure 11.1 Tool Box Song - notation example



The final piece can be divided into two halves. The first half maintains the mechanical, ambience of original version, which is created through the laborious, almost arbitrary, processual treatment of the five units, based on their exhaustive permutation and reinforced through unison vocals with the omission of rests throughout. Nevertheless, I made extensive modifications to the vocals in order to make greater use of the cadential potential of the melodic material and rewrote the movements in order to take full advantage of the table setup. The former is achieved, despite the materials' melodic limitations, by repeatedly ending phrases with 'screw' which is the most suitable unit for giving a sense of closure by virtue of being one beat longer than the others as well as being the highest tonic note. In later sections such as C and E the 'screw' cadence is prolonged with the effect of creating sub-cadences using different units.

Like the original, the movements themselves consist of stylised imitations of hammering and wrenching in a series of interconnected angular arm motions. The movement not only reinforces the vocal material, but also has a dialectical relationship with it through creating its own forms. The clearest example of reinforcement is coincidence of cadences of movement and vocal phrases. The subtle build attempted in the vocals through prolongation of cadences is complemented by a more obvious build up in movement which is initially intermittent, with performers using a single arm at a time and holding positions for several beats in hocket-like patterns, but ramps so that by section F all three performers are moving their arms on every beat. Finally there is the correspondence between turning motions for 'wench' and 'spanner' and lifting and falling motions for 'hammer' and 'nail' which is made particularly clear in section E which contrasts 'hammer' and 'wench.'

The dialectical part of the relationship is most clearly seen the contrast between the unison of the vocals and the heterogeny of the arm movements. Here the movement uses the symmetrical layout of performers to superimpose its own form onto the vocals. For example, in section B where the movement for the left and right hand performers switches between symmetrical and oppositional movements, in section C where the central performers' solo movements are set in opposition to the left and right performers, or in section F in which the symmetrical pairings between arms changes with each phrase.

The second half begins with a short bridging section in which the audience are given some brief respite from the otherwise uninterrupted vocals. Once the vocals have reappeared, the rest of the section reuses the material from the first half but combines it with new features such as non-unison vocals, quaver displacement of vocals, clicking fingers, and movements extended over several beats. In contrast to the first half, unison movements make brief appearances, for example in section I, although the pragmatics of matching the movements to the differing vocal parts means that they soon breakdown. The last two sections act together as a final cadence, with section J, a non-unison reiteration of section E, building up tension through its omission of 'screw', followed by section K which uses it repeatedly by way of ersatz finale.

The contrast between *Do It Again* and *Tool Box Song* in terms of method clearly produces quite different results. Despite its highly constrained material, *Tool Box Song* achieves a far greater level of formal detail which is tightly packed into a five-minute composition. Here the combination of aural and visual information, not only allows for multiple complimentary streams of information but also creates the sense of a logical grammar of audio-visual relations for the audience to discern. However, compositional detail is achieved at the expense of nuance and improvisation in performance, the focus instead moving towards the virtuosity of enacting such a dense and relentless set of permutations. Perhaps, both qualities could have been achieved in a more extended composition - at the end of the process one still had the feeling that the material offered further possibilities to be explored. In many cases it was only discovered that sections were particularly challenging or could have been developed further, after they had already been committed to memory and it was too late to make further changes. In such a context, one wonders what forms might have been achieved in a devising context.

Birdie Songs

video and scores: http://www.simonkatan.co.uk/phd/birdiesongs.html

Birdie Songs (14.03.2012) are an ongoing series of vignettes involving mimes based on the obscene finger gesture, often referred to as 'the bird', accompanied by acousmatic sound. The series forms part of Brainer's Borealis Festival 2012 commission and was inspired by the festival's theme of protest. Of the two Birdie songs composed to date, *Birdie Song #1* is composed entirely by myself, and *Birdie Song #2* by Luke Fraser. For both pieces, the movement is fully scored and performed from memory and the sound tracks composed from samples.

The initial inspiration for *Birdie Song #1* came from the playground version of making the bird where the performer of the gesture pretends to wind their finger up as though it was a geared mechanical device. Carrying this pre-existing mimetic element a stage further, I envisaged that different weights and ratios of gearing could be implied through sound, and that the contrast between these would be just enough material to form a short piece. After some experimentation with various household objects, I finally settled on samples from two different sized ratcheting screw drivers, a dial from a plastic toy tape player, and a stainless steel mechanical cheese grater. The notion of weight is reinforced by the manner in which the performers grip the imaginary handle. Heavier weights require a full-handed grip, whereas lighter weights only require index finger and thumb. Finally the strange ability of one of the performers to exert control over the other performers' fingers, adds a narrative twist to the composition.

In contrast, *Birdie Song #2* combines a more emphatic version of the gesture with series of orchestral hits, sampled from the endings of all of Beethoven's Symphonies. The combination of media turns the performers into a troop of psychotic conductors with the audience as orchestra. Though the movement is more abstract than in *Birdie Song #1*, performers nuance gestures through subtle arm movements and facial expressions. Nevertheless, the main argument of the piece progresses in a similar vein to *Tool Box Song* through use of symmetries and hocketing patterns.

Despite their use of scored notation for movement, both *Birdie Songs* allow a greater degree of freedom for the performers than *Tool Box Song*. However, whilst *Birdie Song #1* tends

more towards the mimetic – imitating real world phenomena, *Birdie Song #2* tends towards the symbolic. In the former the intention for the movement to be construed as the cause of the sound is clear, whilst in the latter the relationship could be interpreted in either direction. Future *Birdie Songs* are envisaged where new semantic contexts will be implied through varying sound sources and further development of movement. Another area of development is the portrayal of causal relations between performers through movement and contact. As with *Do It Again*, the extent to which either composition can be categorised as music is debatable. In this case, it is not because of a lack of sound media, but because the performance does not involve the production of the sound heard. One might be tempted to define the sound component as music and the movement as dance, but I would argue that given the interdependence of both media, not only in terms of performance but also conception, such a definition would be more misleading than simply calling the pieces music.

Set Filler

video and score: http://www.simonkatan.co.uk/phd/setfiller.html

SetFiller (14.03.12) is a piece for voice and interactive projection, and was also developed for Brainer's set at Borealis 2012. For this piece, a devising model, similar to *Do It Again* was adopted, though the interface and visual design was implemented and composed solely by myself. Fillers – words or sounds filling an utterance or pause in a conversation - were taken as the starting point for the composition, though as the composition progressed the scope was expanded to include monosyllabic words and utterances exhibiting a high degree of semantic flexibility through intonation. This limited material is subjected to an exhaustive treatment whereby the performers hold abstract conversations, attempting to convey meanings purely via the moderation of their inflection. The difficulty of such a task is exacerbated by the fact that the performers faces are covered for the duration of the performance with a combination of cartoon mask, balaclava, and brown paper bag. Instead, the audience sees synchronised and rapidly changing images of the still performers' faces in different expressive poses, creating a fruitful counterpoint between voice and image reminiscent of the photomontage style of Marker's *Jetée* (1962).

The compositional process began with an attempt at an exhaustive classification of fillers with corresponding intonations and associated semantic interpretations. However, we soon

decided that more scope and fluidity could be achieved through improvisational means. The final score (Appendix 6) uses an esoteric combination of notation reflecting the different approaches taken to the material, ranging from the conversational, for example in sections A1 and D, to the semantically abstract for example in section H where conventional musical syntax takes precedence. Perhaps the most fruitful parts of the composition are those, such as section C, which manage to occupy a space somewhere in the hinterland.

The visuals were subsequently added on top of this structure. The process involved the identification of common expressive areas in the composition under titles such as negativity, indecision, disbelief, scorn, and confusion, and then developing strong visual gestures using face and hands to match these. We then carried out a photo shoot and subsequently cropped and sorted the images into folders according to performer and what section they would appear in. These are displayed by a relatively straightforward OpenFrameworks interface which assigns each of the performers a trigger via their own mouse which they hold during the performance. The performer simply presses the button each time they say a word which displays an image of their face with a suitable expression on the screen. After a short amount of experimentation it was decided that, with our backs turned to the screen and faces covered, the risk of the pieces sections going out of sync was too high, and I adopted the role of lead performer, triggering changes of section with my right mouse button. The downward pointing arrows on the crib sheet indicate where I trigger new sections. A final point to note about the interface is that aside from the organisation of images according to mood, its functionality extends to different ways of displaying the images including single shots, slow fades, referential split screen, and tiling effects. Such techniques are used to further articulate the form and become an additional source of humour in the work.

Though like *Birdie Songs, Set Filler* manages to achieve a balance between formal organisation and nuanced improvisation, the staged compositional process in which the aural precedes the visual, nonetheless limits the degree of nuance that might be achieved in terms of the relation between utterance and facial expression. One could imagine a much finer exploration of relations between the two media, perhaps including contradictory as well as complimentary combinations. Such a process would involve a more iterative process where the aural and visual material is repeatedly revisited and modified before a final version is settled upon. Nevertheless, a high degree of empathy between the aural and visual experience exists, the latter greatly enhancing the effects of the former. The

combination of frozen facial expression and utterance is strangely forceful, we are compelled it seems to attribute one to the other.

Conclusions:

Brainer's performances have taken place in a wide variety of contexts to both musically and non-musically orientated audiences. We've often found the latter to be more accommodating, perhaps as a result of having fewer prior expectations. Occasionally performances have been site-specific or in unorthodox contexts, but our preferred environment is most certainly in front of a seated and focussed audience.

In light of the presented works, the project of Brainer might best be viewed as an ongoing investigation as how best to co-compose audio-visual relations. Each of the projects has its successes and failures in this regard. The key questions seem to be firstly how to get the formal construction to emerge equally from both media and secondly how to strike a satisfying balance between the nuance that can be achieved through improvisation in live performance, and the architectonic intricacies that formal construction allow. This second issue is all the more pertinent here as, in contrast to the other works in this thesis, Brainer's work places the body and its expressive qualities at the centre of the work. The interaction between our faces, voices and physiques, all play an important role in the meaning of the work and have to be considered in its composition. Nevertheless, one can find many parallels between this work and the rest of the research, for example, the concern with the mimetic and the implication of causality through audio-visual relation, and also the attempt to instil symbolic language systems within the course of the single work. Perhaps the most fundamental common ground that unites Brainer's work and the rest of this thesis is a dialectical relationship between mimesis and abstraction.

Conclusion

In terms of meeting the criteria for creating audio-visual co-dependent music set out in the introduction, the practical element of this research has undoubtedly been successful. In all of the works, equal though not necessarily identical formal development takes place in sonic and visual realms. Furthermore, the two media are mutually reinforcing – their hypothetical separation would result in significant losses of meaning on either side. Such an observation is backed up by the impossibility in most cases of describing the works without equally referring to audio and visual elements and the relationship between them from the outset. Finally the relationship between audio and visual is not only functional but exhibits a self-evident, unforced cohesion. As opposed to parsing parallel streams of complementary or juxtaposed media, one experiences a unified sensory object.

I would argue that a significant contributory factor to the realisation of such an effect was my own interdisciplinary practice. Although such a working situation was necessitated by external factors such as low commission budgets and my inability to find collaborators with complementary aims and skills, it nonetheless had its advantages. Firstly, it fostered an intimate and reflexive understanding of the various media involved, greatly facilitating the development of relevant and practical compositional concepts. Furthermore, assuming complete control of all elements within a project allows for a more flexible and responsive relationship between them during the compositional process. For example, relationships between sound and graphics in projects such as *Cube With Magic Ribbons* could be iteratively adjusted until both sides coincided, whereas in a collaborative project such as *Nautical but Noise* where each artist worked individually on their own area, the number of possible iterations was severely limited. It seems that in most collaborative digital projects, integration is achieved by a staged production process, for example music first and visuals second, and that this order of precedence imposes its own hierarchy on the end result. In light of such observations, I remain doubtful that a collaborative process would have achieved a similar level of integration of media.

However, as mentioned in the introduction, interdisciplinarity carries with it the risk of a division of skill, attention, and experience across the various subject areas. Indeed, I've often heard artists from a variety of disciplines express reservations, particularly in relation to meeting the qualitative requirements of the alien field. I would argue that this

misunderstands the nature of interdisciplinary practice which develops expertise precisely at the intersections between disciplines. Furthermore, one finds an increasing amount of interdisciplinary activity in the arts, not only under categories such as digital performance and sound art, but also amongst computer musicians such as Alo Allik, Fredrik Oloffson, Takeko Akamatsu, and Andre Bartetzki where one finds the use of projected visuals, interaction, sculpture, dance, or performance art. Nevertheless, the acquisition of the relevant knowledge and skills to cover the diverse areas of open source programming, computer vision, computer graphics, choreography, visual arts and digital synthesis presented a significant research challenge which, in certain projects, undoubtedly prevented more detailed realisations.

Another process related aspect that deserves comment is the use of bespoke software design in all of the computer-facilitated projects. On several occasions it has been suggested to me that the work would be more easily realised through patching software such as Max/MSP/Jitter or Isadora instead of OpenFrameworks and SuperCollider. Like many live coders, I prefer using code and well-documented open source libraries as a means of interfacing with the machine. For the main part I regard the decision as whether to type lines or draw them as a matter of personal preference, but I would argue these points in favour of my selection of the particular tools used for these projects. Firstly, Max MSP has higher CPU overheads than SuperCollider and when image processing takes place on the same machine this can limit what can be achieved. Secondly, I personally find the process of connecting high level objects found in patching languages to be more normative than wrapping open source libraries in my own code. Working in the OpenFrameworks environment encourages an approach of reading and adapting source code, and freely inter-combining functions from various other libraries – there is a well trodden path from high level entry points down to lower level techniques which is supported by the generosity of forum communities. Such a development environment has encouraged an esoteric use of libraries in many of the projects, for example, using OpenCV to sum multiple histograms in *Soundpit,* or instantiating multiple Box2D worlds using a nested class structure in *Sound Nest*. Finally, for the works where the user interface itself is part of the artistic endeavour, I would argue that code is undoubtedly correct tool for the job. To mediate the building of such software through a patching language would have been a strange approach indeed. It should be noted that the relevant knowledge, underlying concepts and indeed the very decision to build such software stems directly from previous coding experiences – one should never underestimate the influence of tools on their masters.

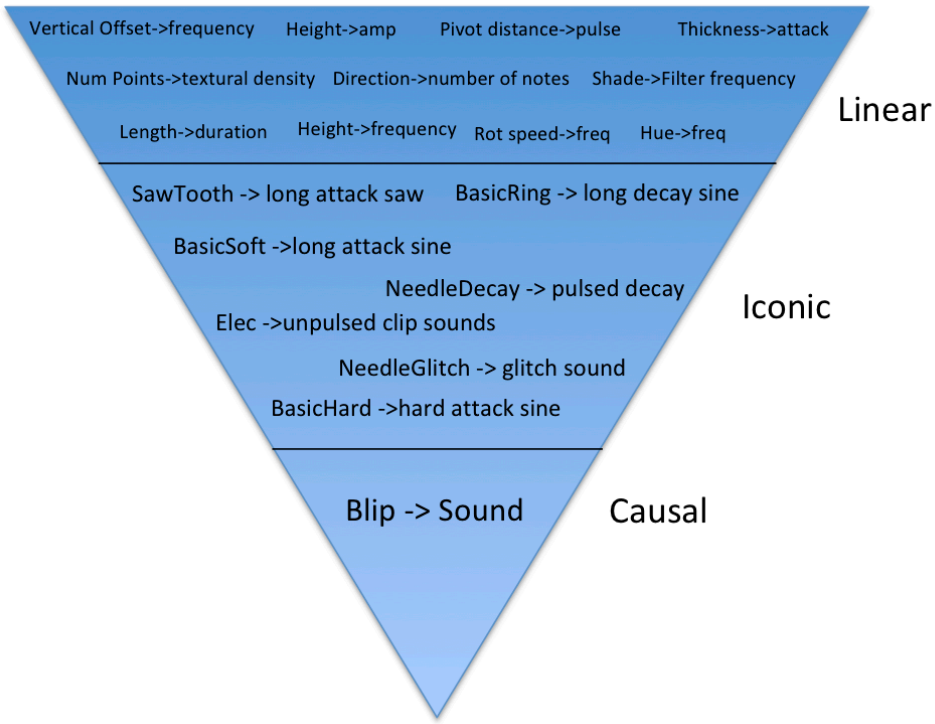
Retrospective analysis of the material itself reveals a number of recurrent features via which co-dependency is achieved. The primary of these is the bottom up conception of the works from primitive audio-visual relationships. Each work is premised on a base causal relation between a tangible or graphical object or event, and a reciprocal sonic event or stream of events, creating an illusory sound source. In the first place these are established via the temporal coincidence of sonic and visual phenomena. In most cases the coincidence of onsets is enough to establish the relation, although in circumstances where there is a high degree of ongoing activity, the coincidence of terminations has an increased significance.

Figure 12.1 implied sound sources

Composition	**Illusory Sound Sources**
Tecken 6.99	Individual and groups of sugar cubes
Les Escaliers	Movement of each performer
Random Walk	User movement
Soundpit	Movement of each ball
Nautical But Noise	Characters and connections between them
God Over Djinn	Collisions between animated objects
DarkStar	Active stars
Cube With Magic Ribbons	Activated blips
SoundLens	Chimes when hit with a hammer
Birdie Song #1	The mechanism raising the performer's finger
Birdie Song #2	An imaginary orchestra conducted by the performer's finger
Set Filler	The projected faces

Resting on top of the base causal relationships are numerous higher-level relationships which fall into two broad categories. One category consists of linear parametrical mappings such as vertical position to pitch, horizontal position to pan, and speed of movement to tempo. They are particularly useful in establishing causality where the sound is more continuous, such as in *Soundpit* or *Nautical But Noise*. The second category consists of iconic relationships in which parameters are non-scalable. Some examples would be the different types of blips in *Cube with Magic Ribbons,* the different composer's heads in *Soundpit*, or the different coloured sandwich boards in *Les Escaliers Méchaniques.* Through their repetition, iconic relationships establish arbitrary categories that reinforce causality through creating predictable variations, and also allow the differentiation of continuous event streams. By these means, rather than dilute the base causal relation, the variation and complexity created by higher-level relations reinforces it.

Figure 12.2 classification of relationships in Cube With Magic Ribbons



Another feature to mention is the varying degrees of mimesis across the total set of relationships. Amongst of the various base causal relationships in Figure 12.1, the relationships in *God Over Djinn, What Is Life?,* and *Soundpit*, which have unambiguous correspondences to real world phenomena, might be considered the most mimetic. Via their familiarity, such relationships are implicitly understood and strongly infer causal links between visual and sonic events. At the other end of the spectrum are relationships such as those in *Tecken 6.99* and *Nautical but Noise.* Given their abstract nature one might refer to such relationships as symbolic. Indeed, some symbolic relationships are reminiscent of notational systems. For example, the use of pitch to vertical height mapping in *Les Escaliers Méchaniques*, or the correspondence of different colouring of particles in *God Over Djinn* to variances in sonic response and physical behaviour. I would tentatively speculate that these types of relationship foster more explicit modes of cognition, encouraging the memorisation and categorisation of symbols and relationships for a rationalistic interpretation. Perhaps the most stimulating are those relationships that fall somewhere in the middle, exhibiting a mimetic quality that is nonetheless ambiguous about what it is representing. For example, one could form multiple hypothesis about what the SawTooth blip represents in *Cube with Magic Ribbons.* In reviewing the works it's apparent that where I have been most successful in achieving a heterogeneous approach to the parsing of form, there has also been a balance

between mimetic and symbolic relationships. Indeed, such an observation is supportive of my suppositions about the cognitive effects of mimetic and symbolic relationships.

A further point to be noted is the avoidance of any relationships that would obfuscate causality. It is for this reason that mimetic aspects aren't subverted or inverted, and in a similar vein, complex and obtuse uses of linear relationships are also excluded. Although multiple audio parameters might derived from a single visual parameter, the integrity of visual objects and events are always maintained in their sonification. There are no cases of multiple visual events contributing towards a single sonic event, for example, multiple animated objects controlling various parameters of a single sound stream. The result, which is a little counterintuitive given the area of research, is that for the majority of works, the audio-visual relationships themselves are static elements with no development occurring by means of their permutation, inversion or abstraction.

Instead progression occurs through the accumulation of numerous variations of linear and iconic relationships under the gestalt of the visual object or event. As has been described, via their hierarchical arrangement, all audio-visual relationships in the work serve to reinforce the illusion that sound is heard as a direct consequence of the witnessed events. A further advantage of this model is its tolerance with regards to the inaccurate parsing of higher-level relationships. For example, viewers will not initially be able to determine that a parameter such as the shading of a NeedleGlitch line in *Cube with Magic Ribbons* relates to filter frequency. As the piece progresses and they see more NeedleGlitch objects they might develop a vague sense that there is a consistent relationship. Those so-minded will consciously attempt to discern the relationship, yet a failure to do so or even an incorrect interpretation will not disrupt parsing elsewhere. All this has clear implications for the accessibility of the form, which via its use of implicit and everyday knowledge about the audio-visual world allows the work a large degree of independence from specialised cultural capital, and via its hierarchical construction allows for a flexible approach to parsing.

A possible objection to this emergent model might be that, although from the point of view of the audience there is an ambiguity about whether audio represents video or vice-versa, the actual process of realisation has a clear order of precedence with the unidirectional flow of control data from video to audio. In response I would point out that the relationship of audio to video is far from slavish. In the first place, not all video elements are rendered in audio, for example, the recursion in *God Over Djinn,* the camera position in *Cube with Magic*

Ribbons, the arrangement of chimes in *What Is Life?*. To some degree one can make the same statement about audio. For example, many of the sonic textural details in *Soundpit* and *Les Escaliers Méchaniques,* are superimpositions on rather than sonfications of the control data. Furthermore, many of the visual decisions are made as a result of their sonic implications, although this doesn't exclude occasional ones made purely on visual grounds. Finally, as previously discussed, the combined compositional process means that both media are developed iteratively in tandem.

A second objection that might be raised is that the initial concept of elucidating formal structures has been largely neglected. Indeed, none of the work focuses directly on algorithmic generative or permutational forms. In the case of the former, as hinted at through review of Dave Griffiths' work in the introduction, I am not clear that the aims of elucidating such structures and producing satisfying audio-visual relations are reconcilable. Ultimately both audio and video get sublimated to the communication of the underlying process. This certainly doesn't render the idea invalid but makes it a project aside from this research. Although permutational forms feature in some compositions such as *Cube with Magic Ribbons* and *What Is Life? ,* little attention is drawn to them. It seems that once visually rendered, such processes, which were after all trivial in the first place, lose much of their mystique. For such renderings to become interesting, far more sophisticated or complex processes might need to be employed, though such a project nevertheless faces the same issues as rendering algorithmic forms.

The apparent banality of transparently rendering permutational forms, raises the question that given the degree of transparency in my own renderings, how interest is maintained. Though the mimetic quality of the audio-visual relations may have some charm for the viewer, without some form of macro development it would soon have little left to offer. Much of this is mitigated by the aforementioned accumulation of variant relationships. However, the effect varies from piece to piece. For example, in *Les Escaliers Méchaniques*, the effect is to eventually disorientate the audience via the sheer number of audio-visual relationships, whilst in *Cube With Magic Ribbons*, where iconic relationships more clearly define categories, the accumulation of audio-visual relationships is non-disruptive and creates an effect akin to observing the real time construction of a language. A regret here is that the accumulation of variant relationships has too often coincided with textural build-ups of sonic and visual material. This is certainly not a prerequisite of the model described but rather a manifestation of using sequencers for live-performance which focus on the

addition and subtraction of objects rather than their transformation. For example, one could easily imagine works where a single stream of audio-visual events becomes increasingly divergent.

Amongst the installation works the accumulation takes place via the user's personal linear experience of the work. As they explore the possibilities of the installation they uncover evermore variants, eventually arriving at a sense of boundlessness via the sheer number of audio-visual relationships. However, although the more open-ended forms of *Soundpit* and *Dark Star* are better suited for the installation setting than *Tecken 6.99* and *Random Walk* which attempt to impose macro linear forms, they nevertheless fail to achieve a sense of boundlessness simply as a result of a paucity of variants. In *Dark Star* where the audio-visual objects are at their most static, the demand for variants is greatest with perhaps hundreds of stars being needed to facilitate a convincing effect. For an individual artist such an endeavour would involve an undertaking of years. I wonder whether a touch application might provide a more suitable and financially viable platform for open forms such as these. One could imagine labyrinthine worlds of audio-visual relations arranged in computer game structures or generative audio-visual objects whose composition morphed via user interaction to produce endless variations.

Apparent in both the realised and speculative projects is the recurrence of an inhibitive element preventing the total comprehension of the presented system. In *God Over Djinn* it is the disorientating effects of the recursive graphics, in *Cube with Magic Ribbons* it is the equally disorienting wrapped space and camera setup, and in *What is Life?* it's the perceptual impossibility of tracking all the chime phases simultaneously . In *Musical Matryoshka* continual expansion of the frame of reference inhibits total comprehension whilst in *Les Escaliers Méchaniques, DarkStar* and *Soundpit* the overwhelming number of audio-visual relationships has the same effect. This recurrent conjuring of the intangible is the same as occurs in the earlier described reception of the Bach fugue but is also present in Feldman's "enveloping environments, in which listeners experience music from the 'inside' a composition" (Burt, 2006). Indeed, one might speculate that the intangible is requisite in pointing towards the ethereal, and that this is perhaps the strongest argument against 'dumb music' such as Johnson supposed he had created.

However, the presentation of the intangible will have little meaning unless it arises from or results in tangible constituents, and it is here that the attention to the apprehension of form

is crucial. Just as in the Bach fugue, where the explicitness of the opening material encourages the listener into a particular interpretation, lending crucial meaning to their eventual disorientation, so the works here begin by deliberately presenting the audience with basic, almost naïve premises. These could be viewed as exemplar of Hofstadter's "frame message" (Hofstadter, 1999, p. 166) which says, "I am a message; decode me if you can!" In applying his model to the music of Bach, Hofstadter's assumes the "inner message" – the extracted meaning intended by the messenger – to be the rational interrelation of notes, and the "outer message" – the means of decoding the inner message – to be tonal system which realises those interrelations (Hofstadter, 1999, p. 175). However, I would argue for a more complex situation based on our perceptual experience, rather than what is evident from the score. The "outer message" is the same but leads us towards an inevitable failure, revealing an alternative "inner message" which tells us that despite knowing the notes must be rationally related we will never be able to completely grasp the complete nature of their interrelation. In other words the "inner message" is that we will never be able to understand the "inner message."

I sense an analogy in Hofstadter's description of Gödel's Incompleteness Theory, which shows how the assumption of consistency in Number Theory, or TNT in the case of Hofstadter, forces one to conclude that Number Theory (or any similar system) is 'essentially incomplete' – we can never formally encapsulate all possible truths about Number Theory (Hofstadter, 1999, pp. 438-451). The logical derivations of Number Theory – perhaps analogous to the "outer message" – lead us towards failure. However, this in turn expands our world of true statements, for if not all true statements can be surmised by our formal systems, then it stands to reason that there are true statements that we cannot yet articulate. Indeed, as discussed in *Musical Matryoshka,* Gödel's proof announces to us the new class of "Supernatural Numbers" (Hofstadter, 1999, p. 452) This happens by adding the negation of Gödel's theorem as a new axiom of Number Theory. The resultant inconsistency can only be resolved by a new interpretation of the Number Theory's symbols, which supposes the existence of the new number class.

Bringing matters a little closer to earth, one can perhaps use such a description to create a model for linear progression focussing directly on the formal relations of the interface. Inconsistent behaviour could be introduced as a way of creating tension by implying that the current interpretation is incomplete, forcing the listener into a search for a new more valid one. The discovery of the resolving interpretation would be equivalent to an expansion of

the formal system. In this way one could imagine music, where development occurs not through expansion of the sonic or visual fields but through using a process of tension and release to effect a progressive widening of the formal system itself. Though the revelations of works such as *Cube With Magic Ribbons* and *God Over Djinn* create the sense of an expanding frame of reference, there is still an absence of tension due to the mutually reinforcing nature of all the audio-visual relationships. Ironically whilst fixed forms such as Deinststelle's visualisation of *Neue Stadt* and Norman McClaren's *Dots* achieve inconsistency so effortlessly, such a task is significantly more challenging for real time audio-visual computer music. Nevertheless, one could imagine the introduction of variant objects, which exhibit more complex behaviour, perhaps through contingency on other events, or by varying responses cyclically. The cognitive tensions arising from these seemingly inconsistent behaviours might be resolved simply through observation of repetition.

To achieve these results, such works would have to pay careful heed to the linear presentation of events. Indeed, as set out in the introduction, a concern with exerting a macro linear influence on audience experience has an involvement with the conception of all the works including even the installations. Of course such an approach stands in contrast to Nyman's conception of "mobile perception," though nevertheless in my work, such direction is not totalitarian, and directionality is only attempted at certain levels, leaving other perceptual aspects to be more freely experienced by the audience (Nyman, 1999, p. 28). Indeed, there is a correspondence between those aspects that have no stake in directing the macro linear experience and those that are left for improvisation during performance – revelatory effects require some predetermination of architecture. In any case, rather than adopt a polarised stance on the issue, I prefer to see such approaches as non-mutually exclusive, to be chosen and inter-combined according to the composer's needs.

A final aspect to be discussed is the relationship between the audience and the performer, who has the additional roles of programmer and composer. Although, despite Brainer's gestural explorations, this research has not focussed on the development of a physical gestural language for computer music, I would argue that, in terms of the live dynamic, the solo performances offer some advantages over live coding performance practice. In the first place, the temporal flows of visual and sonic events are in agreement. The visible screen actions of the performer tend to match the sonic output in terms of impact, and the time between action and reaction is often immediate and usually short. Where a lengthier visible

preparation precedes a sonic event, for example the drawing of looped tracks in *Cube with Magic Ribbons,* the equal weighting of aural and visual plays to the performer's advantage by creating an effect of expectation.

Although perhaps unfashionable, the use of the mouse as the fundamental interaction device proves more advantageous for my particular performance needs than any other currently available control surfaces. Firstly, the mouse interface is so commonplace that even the least computer literate members of the audience will correctly interpret the connection between the ubiquitous, unobtrusive pointer icon and its connection to the performer's physical actions. Secondly, in a spatial interface the mouse pointer neatly surmises the agency of the performer. It shows their interventions with the system, for example, when they drag a particle across the screen in *God Over Djinn,* and to some degree their thought process, for example as the pointer wanders whilst the performer decides on what area to manipulate next. Of course key commands, menu systems, and the use of right and left mouse buttons means that, unlike in live coding, not all actions are determinable from viewing the screen. Nevertheless, I would argue that the performer's role is still adequately conveyed. The T.O.P.L.A.P restriction of disallowing unprojected ancillary control screens seems a good way of maintaining a certain degree of transparency. Although, a control surface allowing for the direct manual manipulation of graphical objects would be more desirable in terms relating the actions of the performer, I find the notion of filming a touch screen unattractive. A better solution might be to use overlaid graphics as happens in *Nautical But Noise*. As technologies improve one could imagine such interfaces using gesture recognition for rapid manual and perhaps virtuosic shifting between modes, whilst also allowing fine motor control for nuanced gestures.

Aside from the differences from live coding in the rendering of performance actions, there is also a subtle shift in the performance's extended text, particularly with regards to authorship. Composer, Reynaldo Young points out that when we watch an acoustic performance of a Mozart sonata, "we are certainly processing, comparing, expecting, patterns; and we do let those patterns satisfy, shock, surprise us, and so on", "but, above all, we are listening to *Mozart* – a fellow being who is in charge (has the power) at that moment, by mastering the appropriate skills, of articulating the space-time sharing experience we are so much in need of as a species" (Young, p. 14) To this I would add the agency of the performer, for even less experienced listeners are likely to have sufficient knowledge of the classical tradition and its workings to form a crude distinction between Mozart's conceptual

offering and the realisation of the performer. In my performances there is yet another layer of authorship as, in an increasingly computer literate world, the esoteric functionality and stripped down graphics imply the interface's bespoke design. The frequency with which I'm asked whether I built the interface following my performances supports such a presumption. As the audience watches my performance, they consider not only the interaction of visual and audio and its gestural connection to the actions of the performer, but also the design and functionality of the software. Despite my authorship from software design to final performance, the determination of boundaries between my real time input and the input of the software becomes significant to the audience.

Considering the base causal relationships in the context of the performer/composer results in extended hierarchical chains of causality. For example in *Cube with Magic Ribbons*, the performer operates the software; the software controls the movements of the reader; the reader causes the blips to react and the reaction causes the sound. One future area of research might be to explore possibilities of how far such chains can be extended, for example through an interface involving the interconnection of graphically rendered, sound producing mechanisms such as pistons, hammers, and gears. In this case, temporal ordering and not temporal coincidence would play a role in establishing causality. Temporal coincidence at the base level has been a prerequisite for this research, but one could envisage fruitful explorations involving the temporal separation of sonic and visual events, for example using a gesturally controlled interface incorporating varying degrees of latency, to create overlapping streams of audio-visual pairings. Alternatively one might use live performers in combination with a form of projected graphic notation. In this case, temporal ordering would change the implied function of the notation – when preceding a performance action one sees it as instructional and when following a performance action one sees it as representational. The use of animated notation suggests further possibilities involving the use of machine listening to create feedback loops between performers and a projected score. The performers respond to the instructions of the score and the score in turn changes in response to the sounds of the performers. Rather than using standard notation, such scores would consist of animated symbols with corresponding interpretive rules for the musicians. Such worlds offer yet a further possibility in which the audience is able to directly influence events in real time via their mobile devices.

The audio-visual world offers extensive uncharted territories, of which this research has explored only a fraction. Technological developments call for this exploration to begin in

earnest. Given the modern media context, the exclusion of the visual as a norm in modern computer music has become increasingly untenable. However, I also sense that audiences, well weathered by the visual pyrotechnics of recent years, are becoming aware and intolerant of superficial combinations of audio and video. To fully explore the potential of the audio-visual world, the production processes must be unified. This requires computer musicians to not only acquire the relevant skills but also allow their compositional priorities to be shaped by the new medium. Their work should be informed not simply by the respective aesthetic qualities of the separate mediums, but should emerge from the thought through investigation of the relations between them. My experience is that when this approach is taken, one achieves a result greater than the sum of its parts. In its ability to speak directly to the cultural experience of modern audiences without recourse to elitist cultural capital, patronising simplification, or cynical use of popular references, such an approach offers exciting opportunities for computer and acoustic musicians alike.

Bibliography

Burt, R. (2006, November). Both Sitting Duet. *Ballet-dance magazine* from http://www.ballet-dance.com/200611/articles/BurrowsFargion20061018.html accessed on 09.05.2012

Babbit, M. (1958, Feb). Who Cares if You Listen. *High Fidelity* VIII/2, 38-40, 126-127.

Bourke, P. (2005). Using a spherical mirror for projection into immersive environments (Mirrordome). *Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia* , 281-284.

Cascone, K. (2003). Grain, Sequence, System: Three levels of reception in the performance of laptop music. (M. S. Kleiner, & A. Szepanski, Eds.) *Soundcultures* .

Cardew, C. (1971). *Treatise Handbook.* Peters Edition.

Carlsson, S. E. (n.d.). *Sound Design of StarWars*. from http://www.filmsound.org/starwars accessed on 24.09.2012

Collins, N. (2007). Live Coding Practice. *Proceedings of the 2007 Conference on New Interfaces for Musical Expression* .

Collins, N. (2009). Musical form and Algorithmic Composition. *Contemporary Music Review , 28* (1).

Donnarumma, M. (n.d.). *Xth Sense*. from http://marcodonnarumma.com/works/xth-sense/ accessed on 09.24.2012

Hofstadter, D. R. (1999). *Godel Escher Bach: an Eternal Golden Braid* (20th-anniversary ed.). London: Penguin.

Kim-Cohen, S. (2009). *In the Blink of an Ear: Toward a Non-Cochlear Art.* NYC: Continuum.

Lyotard, F. (1984). *The Postmodern Condition: A Report on Knowledge.* Minneapolis: University of Minnesota Press.

Levinson, J. (1997). *Music in the Moment.* NYC: Cornell University Press.

Leppert, R. (1993). *The Sight of Sound.* University of California Press.

Nyman, M. (1999). *Experimental Music: Cage and Beyond* (2nd Edition ed.). Cambridge: Cambridge University Press.

Magnusson, T. (2011). The ixi lang: A SuperCollider Parasite for Live Coding. *ICMC 2011 Proceedings* .

McAdams, S. (2004). Influences of Large-Scale Form on Continuous Ratings in Response to a Contemporary Piece in a Live Concert Setting. *Music Perception , 22* (2), 297-350.

McCallum, L., & Smith, D. (Directors). (2011). *Show Us Your Screens* [Motion Picture].

McLean, A., Griffiths, D., Collins, N., & Wiggins, G. (2010). Visualisation of Live Code. *Electronic Visualisation in the Arts Proceedings* .

Oswald, J. (1985). *Plunderphonics, or Audio Piracy as a Compositional Prerogative*. from http://www.plunderphonics.com/xhtml/xplunder.html accessed on 24.09.2012

Schaeffer, P. (1967). *La musique concrete.* Paris: Universitaires de France .

Snyder, B. (2000). *Music and Memory: an introduction.* Massachusetts: MIT Press.

Smalley, D. (1986). Spectro Morpohology. In S. Emmerson, *The Language of Electro Acoustic Music* (pp. 61-93). London: Macmillan Press.

Reber, A. S. (1967). Implicit Learning of Artificial Grammars. *Journal of Verbal Learning and Verbal Behaviour , 6*, (pp. 855-863).

Reich, S. (2002). *Writings on Music 1965-2000.* NYC: Oxford University Press.

Tillmann, B., & Bigand, E. (2004). The Relative Importance of Local and Global Structures in Music Perception. *The Journal of Aesthetics and Art Criticism , 62* (2).

Thompson, W. F., Graham, P., & Russo, F. A. (2005). Seeing music performamce: Visual influences on perception and experience. *Semiotica , 156*, 203-227.

T.O.P.L.A.P. (2010, November 14). *draft manifesto*. from http://T.O.P.L.A.P.org/ accessed on 26.08.2012

Wishart, T. (1996). *On Sonic Art.* Amsterdam: Overseas Publishers Association.

Young, R. (n.d.). *writings.* from http://www.reynaldoyoung.com/writings/socialAspectsInMusicalProcessing.pdf accessed on 24.09.2012

Media References

Scores:

Bach, Johann, Sebastian. (1747), *Crab Canon,* BWV 1079. Frankfurt: Peters Edition.

Cardew, Cornelius (1967) *Treatise*. London: Peters Edition

Feldman, M (1982) *For John Cage.* London: Peters Edition

Johnson, Tom (1986) *Counting Keys*. Paris: Editions 75

Johnson, Tom (1982) *Rational Melodies: XV*. Paris: Editions 75

Nørgård, Per (1968) *Voyage Into the Golden Screen.* Edition Wilhelm Hansen

Xenakis, Iannis (1954) Metastasis. London: Boosey & Hawkes

Schoenberg, Arnold (1912) *Pierrot Lunaire* . Universal Edition

Recordings:

Gould, Glenn (1965) *J. S. Bach The Well Tempered Clavier Book I*. Sony

Cross, Christopher (1980) *Sailing.* Warner Bros.

Enya (1988) *Orinoco Flow.* Reprise/EMI

Jackson, M (1987) *Bad.* Epic

Horner, James and Sissel (1997) *Titanic: Music from the motion picture.* Sony Classical

Knussen, Oliver & Danish National Radio Symphony Orchestra (1992) *Nørgård: Gilgamesh - Voyage Into the Golden Screen: 2nd Movement.* da capo.

Les Percussions de Strasbourg (1987) *Iannis Xenakis Pleiades.* Musique d'abord

Oswald, John (1985) *Dab,* Mystery Laboratory

Payne, Roger (1970) *Songs of the Humpback Whale.* Living Music

Stockhausen, Karlheinz (1960) *Kontakte.* Wergo

Varese, Edgar (1960) *Poeme Electronique.* Phillips

Live Performances:

Fargion, M and Burrows, J (2002) Both Sitting Duet,
http://www.youtube.com/view_play_list?p=A37337B6A22DFE0E accessed on 07.05.2013

Ikeda, Ryoji (2008) *Test Pattern,*
http://www.ryojiikeda.com/project/testpattern/ accessed on 07.05.2013

Prudence, Paul (2007) *Son Lattice*
http://www.transphormetic.com/11_sonLattice/sonLattice01.htm accessed on 07.05.2013

Quayola, Davide (2007) *PTA,*
http://www.quayola.com/soundvisualisation/pta/ , accessed on 07.05.2013

Films:

Deinstelle (2001) *visuals for Neue Stadt (Skizze 8)*. Mille Plateux

Jones, Chuck (1949) *Road Runner*. Warner Bros.

Kreidler, Johannes, (2011) *Kinect Studies*.

http://www.youtube.com/watch?v=UAlcTnvbBS0 accessed on 07.05.2013

Lucas, George (1977) *Star Wars*. 20th Century Fox

Marker, Chris (1962) *La Jetée.* Argos Films

McLaren, Norman (1940) *Dots,* commissioned by Solomon Guggenheim Foundation.

Musical Interfaces:

Donnarumma, Marco (2011) *Xth Sense*

url: http://marcodonnarumma.com/works/xth-sense/ accessed on 07.05.2013

Jorda, Serge (2006) *Reactatable*

url: http://www.reactable.com/ accessed on 07.05.2013

Crabtree, Brian & Cain, Kelli (2006) *Monome*

url: http://monome.org/ accessed on 07.05.2013

Griffiths, Dave (2008) *Al Jazari*

url: http://www.pawfal.org/dave/index.cgi?Projects/Al%20Jazari accessed on 07.05.2013

Griffiths, Dave (2006) *Beta Blocker*

url: http://www.pawfal.org/dave/index.cgi?Projects/Betablocker accessed on 07.05.2013

Matt, Jonny & Assembly ltd, (2012) *The V Motion Project*

url: http://www.assemblyltd.com/work/v-motion-project accessed on 07.05.2013

Matthews, Max (1970) *Radio Baton*

Computer Games:

Iwatani, Toru & Funaki, Shigeo (1980) *Pac-Man*. Namco

Miyamoto, Shigeru & Tezuka Takashi (1985) *Super Mario Bros*. Nintendo

Jones, D.G. & Dewdney A. K. (1984) *Core Wars*

Artdink, (1995) *Carnage Heart*. Sony Computer Entertainment

Logg, Ed & Rains, Lyle(1981) *Asteroids.* Atari Inc.

Alcorn, Allan (1972) *Pong*. Atari Inc.

Open Source Libraries and Resources

OpenFrameWorks url: http://www.openframeworks.cc/ accessed on 07.05.2013

SuperCollider url: http://supercollider.sourceforge.net/ accessed on 07.05.2013

OpenCV url: http://opencv.org/ accessed on 07.05.2013

Box2D url: http://box2d.org/ accessed on 07.05.2013

Processing url: http://processing.org/ accessed on 07.05.2013

JCollider url: http://www.sciss.de/jcollider/ accessed on 07.05.2013

List of Works & Performances

Tecken 6.99:

26.11.2009, *The Dissolving Cube*, The Portman Gallery, Bethnal Green, London.

Les Escaliers Mécanqiues:

25.01.2010, *Borealis Festival – London Launch*, Kings Place, London.

Random Walk:

19.05.2010, *Researching the Art Conference*, Brunel University, Uxbridge.

19.06.2010, *Xenakis Unbound,* Spitalfields Festival, London.

Soundpit:

19.06.2010, *Xenakis Unbound,* Spitalfields Festival, London.

18.11.2010, *Music Orbit,* Chelsea Theatre, London.

23.03.2011, *Borealis Festival,* Odda, Norway.

13.05.2011, *Sonic Maze –Net Audio Festival, Camden* Roundhouse, London.

12.04.2012, *SuperCollider Symposium 2012,* Mile End Arts Pavilion, London.

Nautical But Noise:

11.11.2010, *Sampler Festival - Synthesis,* The Albany, Deptford, London.

25.03.2011, *Borealis Festival,* Bergen, Norway.

God Over Djiyeahnn:

09.03.2011, *Student Composers' Concert,* Brunel University, London.

25.06.2011, *Beam Festival,* Brunel University, London

02.08.2011, *ICMC 2011,* Huddersfield University, London.

24.03.2012, *Agony Art,* Chisenhale Studios, London.

Musical Matryoshka:

16.06.2011, *Researching the Art Conference*, Brunel University, Uxbridge

11.10.2011, *Artisttalk.eu,* Ljubljana, Slovenia.

DarkStar:

12.10.2011, *Sonica 2011*, Ljubljana, Slovenia.

30.03.2012, *The Artaud Forum,* Brunel University, Uxbridge

Cube with Magic Ribbons:

24.03.2012, *Agony Art,* Chisenhale Studios, London.

18.06.2012, *Researching The Arts,* Brunel University, London

23.06.2012, *Beam Festival,* Brunel University, London

06.09.2012, *The Shag, …*Studios, Lambeth, London.

09.09.2012, *Annex East Olympic Closing Party,* Annex East, Stratford, London.

12.09.2012, *ICMC 2012,* Metelkova Menza, Ljubljana, Slovenia.

21.09.2012, *Sho-Zyg,* Goldsmiths University, London.

What Is Life?:

unperformed

Do It Again:

16.07.2010, *Soundwaves Festival,* Brighton University, Brighton.

23.09.2010, *Depford X - Launch,* The Albany, Depford, London.

29.10.2010, *Scaledown,* Euston, London.

23.03.2011, *Borealis Festival*, Odda, Norway.

Tool Box Song:

17.02.2012, *Scaledown,* Euston, London.

15.03.2012, *Borealis Festival,* Bergen, Norway.

Birdie Songs:

15.03.2012, *Borealis Festival*, Kings Place, London.

Set Filler:

15.03.2012, *Borealis Festival*, Kings Place, London

Appendix A *Nautical But Noise* – Descriptive Score

Also available in www.simonkatan.co.uk/phd/nautical.html

Key
Lu = Luke
Si = Simon

EN = Enya
SS = Sissel
CC = Christopher Cross
HB = HumpBack Whale

VS = alternate between these states
AND = both showing at the same time (will form a duo with a connecting line)
+ = simultaneous duos, or solo and duos (the interface excludes simultaneous solos from opposing performers)

(0,0 -> 0.5,0.5) = prescribed direction of travel according to screen coordinates where (0,0) is the bottom left hand corner and (1,1) is the top right hand corner.

NB. This score is intended only as a reference to accompany video footage and not as a complete set of instructions for realising the work.

Section	Description	Actions
A1	Hands move gently across the sea making gentle wave sounds. Humpback whales make intermittent appearances.	No fingers showing all hands making slow circular motions Lu[HB] VS Si[HB] x2 *adlib*
A2	Only Luke's hand is left. Duets between Enya and Sissle as the sea sound fades away.	All hands withdraw except Lu (0) Lu[SS] VS *Si[En] x2 (0,0.5 -> 0.2,0) (0,0 -> 1,0) (0,0.75 -> 0.5, 0) (0.25, 0 -> 1, 0.25)
A3	A duet between Enya singing "sail away" and Christopher Cross singing "sailing"	Lu[CC] VS *Si[En] x2.5 (0,0 -> 0.25,0.5 -> 0,1) (1,0 -> 1,1) *accel each time*
A4	Enya and Sissle descend on one hand. The sound slows and descends accordingly.	Si[EN, SS] (1,1 -> 0.25,0)
B1	A second hand enters. There is a sudden switch to two Sissles joined by a line. The sound is static except when the hands move.	Lu[SS] AND Si[SS] (0.25,0 -> 0.5,0.5) *then adlib on y = 0.5 keeping short even distance* *come to rest in the center*
B2	Humpback whales intermittently join in	Lu[SS] AND Si[SS] *adlib as before* + Lu[HB] * VS Si[HB] x2
B3	Christopher Cross moves across the top of the screen with some granulated guitar arpeggios	Lu[SS] AND Si[SS] *adlib as before* + Lu[CC] (1,1 -> 0,1)
B4	Christopher Cross reappears and descends to the bottom of the screen Sissels with the Sissels who form two Mickey Mouse-like ears over his head. The sound descends into a low rumble.	Lu[SS] AND Si[SS] (0.5,0.5 -> 0.2,0) + Si[CC] (1,1 -> 0.2,0)
C1	Christopher Cross disappears and is replaced by two joined Enyas creating a slowly morphing chordal texture	Lu[SS] AND Si[SS] *ad-lib* + Lu[EN] AND Si[EN] *ad-lib*
C2	The Sissels disappear	Lu[EN] AND Si[EN] *ad-lib*

Section	Description	Actions
C3	Two joined Christopher Crosses enter bringing in some defined attacks.	Lu[EN] AND Si[EN] *ad-lib* + Lu[CC] AND Si[CC] *ad-lib*
C4	The four characters move to the outer corners of the screen. The connecting lines form a cross in the centre of the screen, the mappings of the two synths combined with screen resolution creating a 4:3 polyrhythm.	Lu[EN] AND Si[EN] (->0,0) (->1,1) + Lu[CC] AND Si[CC] (->0,1) (->1,0)
C5	The four characters converge on the centre creating an accelerando	Lu[EN] AND Si[EN] (->0.5,0.5) + Lu[CC] AND Si[CC] (->0.5,0.5)
C6	A succession of fast transformations.	Lu[SS] AND Si[SS] + Lu[HB] AND Si[HB]
		Lu[EN] AND Si[SS] + Lu[HB] AND Si[CC]
		Lu[SS] AND Si[EN] + Lu[HB] AND Si[CC]
		Lu[SS] AND Si[EN] + Lu[CC] AND Si[HB]
		Lu[EN] AND Si[SS] + Lu[CC] AND Si[HB]
		Lu[EN] AND Si[SS]

Section	Description	Actions
D1	The Sissel and Enya duo move wildly around the screen eventually settling on opposite corners. The sound is a dense and noisy granulation	Lu[EN] AND Si[SS] *ad-lib rapid movement* (->0,0) (->1,1)
D2	Enya descends onto Sissel. The granulation contracts in frequency range.	Lu[EN] AND Si[SS] (0,0) (1,1->0,0)
D3	Christopher Cross appears again singing the words “wind is right sail away” in a demonic voice. As he descends Enya and Sissel ascend creating a rising climactic passage.	Lu[EN] AND Si[SS] (0,0 -> 0.3,1) + Si[CC] (1,0.3 -> 0.25,0)
E1	A sudden switch to a duo of honking humpback whales which swim around each other.	Lu[HB] AND Si[HB]
E2	Eventually one humpback swims off	Si[HB]
E3	Enya intermittently makes appearances performing strange duos with the lone whale	Lu[EN] AND Si[HB] *adlib* (0.75,0.25) *adlib* (0.25,0.25) (->0.5,0.5) (0.5,0.5) VS x3 *SI[HB] *adlib*
E4	A sudden switch to a gentle Sissel and Enya duo to the words “turn it up” that rises and falls with pitch shifting and playback rate modulation	Lu[SS] AND Si[EN] (0.5,0.5 - > 0.5,1) (- > 0,0)
E5	A further sudden switch to a Sissel and Humpback duo effecting a bright rising chord as the two characters spread out across the screen.	Lu[HB] AND Si[SS] (0,0->0,1) (0,0 -> 1,1)
F1	A brief silence followed by a noisy alternations of all four characters from each player	Lu [ALL] * VS Si[ALL] x 2 *ad-lib* *ad-lib*

Section	Description	Actions
F2	The characters emerge one final time but this time characters emerge from the other hand creating connections between them, gradually calming the music.	Lu [ALL]
		Lu [HB,CC,SS] + Lu[EN] AND Si[EN]
		Lu [HB,CC] + Lu[EN] AND Si[EN] + Lu[SS] AND Si[SS]
		Lu [HB] + Lu[CC] AND Si[EN] + Lu[SS] AND Si[SS]
		Lu [HB] AND Si[HB] + Lu[CC] AND Si[EN] + Lu[SS] AND Si[SS]
F3	The hands move rapidly around the screen effecting glissandi. They pause intermittently creating different geometrical configurations.	Lu [HB] AND Si[HB] + Lu[CC] AND Si[EN] + Lu[SS] AND Si[SS] *ad-lib (intermittent pausing)*
F4	The missing characters are introduced resulting in four duos of paired characters, drawing four parallel lines across the screen. The resulting texture is static and morphs slowly as the hands rearrange themselves eventually forming a double cross in the centre of the screen.	Lu [HB] AND Si[HB] + Lu[SS] AND Si[SS] + Lu[CC] AND Si[CC] + Lu[EN] AND Si[EN]

Section	Description	Actions
F5	The hands move to the corners of the screen once again creating a polyrhythmic texture. Before moving back to the centre of the screen bringing all the characters together.	Lu [HB] AND Si[HB] (->0,1 ->0.5,0.5) (->1,0->0.5,0.5) + Lu[SS] AND Si[SS] (->0,0->0.5,0.5) (->1,1->0.5,0.5) + Lu[CC] AND Si[CC] (->0,1->0.5,0.5) (->1,0->0.5,0.5) + Lu[EN] AND Si[EN] (->0,0->0.5,0.5) (->1,1->0.5,0.5)
F6	The characters move slowly across the screen subtly changing the texture and taking on the appearance of a Ouija board. They eventually settle in the bottom left corner.	
G1	Gradually characters disappear, eventually only leaving a duet of a Humpback and Sissel.	Lu [SS] AND Si[HB] + Lu[EN] AND Si[EN] + Lu[CC] AND Si[CC]
		Lu [SS] AND Si[HB] + Lu[EN] AND Si[EN]
		Lu [SS] AND Si[HB]
G2	The duo ascends to the top corners of the screen gradually a chord comes to dominate the texture	Lu [SS] AND Si[HB] (0,0->0,1) (0,0 -> 1,1)
G3	There is a switch to a duo between a humpback and Christopher Cross and a staged descent is made with reciprocal discrete downward modulations in pitch.	Lu [HB] AND Si[CC] (0,1->0,0.9) (1,1 -> 1,0.9) (0,0.9->0,0.75) (1,0.9 -> 1,0.75) (0,0.75->0,0.5) (1,0.75 -> 1,0.5)
G4	A final switch to a duo between Enya and a humpback whale as the characters continue to descend the grain size becomes increasing small until only percussive clicks are heard. The characters disappear, bringing the piece to a close.	Lu [HB] AND Si[EN] (0,0.5->0,0) (1,0.5 -> 1,0)

Appendix B *God Over Djinn* – Descriptive Score

Also available in www.simonkatan.co.uk/phd/god.html

Section	Description	Feature Use		Presets	
A1	Starts from blank screen. Incremental addition of sub-particles, each containing one particle. Performer drags to reveal more space, which is eventually completely filled.	addToInner	Lft click to add sub-particle Rgt click to add sub-sub	Perpetual Basic	Physics balanced for conservation of momentum. Short decay. Wendy Carlos Alpha tuning. Pitch mapped to y-axis.
		drag	to move the outermost particle revealing more of the space		
A2	Zoom out to reveal the outermost particle. Textural change through gradual transformation of sub-particles starting with lowest and proceeding to highest creating a filter sweep effect	Zoom	zoom out to reveal outermost particle	Perpetual PitchShift	Long decay Close linear tuning Continual pitch mapping to y-axis
		Transform	sub-particles transformed to perpetual pitchShift		
A3	Sub-particles are dragged to collide with each other revealing a more complex movement, making a subtly shifting texture due to continual mapping of pan and y-axis. Then increase in speed and zoom out.	Drag	lft click move sub-particles at first slowly then faster	Perpetual PitchShift	
		Zoom	zoom out		
		AddOuter	to tacitly add new outermost particle for next section		
A4	sub-sub-particles released into outermost spread out into a space that has not yet been defined. Transforming of geometry resulting in a sudden narrowing of frequencies and subsequent widening as particle spread out	RemoveOuter	lft click on single sub-particle to release into outermost	Perpetual PitchShift	
		AdjustSpeed	lft click on outermost to increase speed of spread		
		Zoom	zoom out to reveal previously hidden outermost		

Section	Description	Feature Use		Presets	
A5	A further outermost is added, the sub particle to the bottom of the screen and its sub-particles released. They create a rising sequence as they drift up from the bottom	AddOuter	tacitly add new outermost	Perpetual PitchShift	
		Drag	Lft click to pull sub-particle towards bottom of outermost		
		RemoveOuter	lft click on single sub-particle to release into outermost		
A6	Zoom in on a single sub-particle. Call of 'stepInto' suddenly reduces the dense texture to a single particle.	Zoom	zoom out to reveal outermost	Perpetual PitchShift	
		AdjustSpeed	lft click to slow sub-particles to a stand still		
		Zoom	lft click and zoom in on single sub-particle		
		StepInto	lft click on zoomed in sub-particle		
		Zoom	further zoom in till particle fills the screen		
B1	The particle is transformed to a PerpetualPolar changing the frqeuencies.	Transform	lft click on outermost	Perpetual Polar	Long Decay with slightly wider pitch range. Initial location sets pitch fundamental through polar coordinates
		Zoom	slight zoom out to prepare for next section		
		Drag	lft click on outermost – position will set pitch in the next section		
B2	'addOuter' feature is repeatedly used to build up a deeply nested image, reminiscent of many Droste effect images.	AddOuter	tacitly add new outermost pitched by position of old outermost	Perpetual Polar	
		Drag	drag the new sub-particle so that it hits the edges of the outermost		
		Zoom	zoom out to reveal outermost		
		Drag	lft click on outermost – position will set pitch for next		

Section	Description	Feature Use		Presets	
C1	Transition section using 'perpetual noInner' preset to enclose the entire nested series. Process is repeated twice to nest it as a sub-sub particle.	AddOuter	tacitly add new outermost	Perpetual noInner	inner edge collisions are disabled but outer edge collisions enabled with adjustable decays. Size of particle effects pitch on collision
		Drag	drag sub-particle to maintain movement		
		Zoom	zoom out to reveal outermost		
C2	Two additional sub-particles are added, collisions between outer edges occur and the metallic outer edge collision sound is heard for the first time.	AddToInner	Lft click to add sub-particles of varying sizes and long outer decays.	Perpetual noInner	
C3	The sub-particle containing the 'perpetual Polar' series is cleared leaving only the outer edge collision sound remaining	clear	rgt click on sub-particle	Perpetual noInner	
C4	'shakeable Basic' particles of different sizes are added to the sub-particles. Process is repeated for sometime building up the texture	AddToInner	Rgt click to add sub-sub-particles of varying sizes and short outer decays.	Shakeable Basic	A particle with momentum losing physics. Has shake enabled meaning a force in a random direction is applied when the super-particle collides
C5	Select by type feature is used to select sub-sub particles and add 'Shakeable Basic' particles to them	Select by type	lft click on sub-particle select all the 'shakeable Basic' sub-sub-particles	Shakeable Basic	
		AddToInner	lft click to add particles inside of selected particles		

Section	Description	Feature Use		Presets	
D1	Fifty identically sized ‘perpetual noInner 2’ particles surround the previous configuration.	AddOuter	tacitly add new outermost	Perpetual noInner 2	Variant on ‘Perpetual noInner’ allows adding of a start force
		AddToInner	lft click to add sub-particle with start force		
D2	Old sub-particle containing the shakeable particles is then destroyed leaving just the shimmering high frequency texture of the ‘perpetual noInner 2’s	Destroy	lft click on sub-particle		
D3	Five larger sub-particles are added. Results in a Brownian motion caused by collisions with the faster moving sub-particles. The larger particle sizes means that lower frequencies are added to the texture.	AddToInner	lft click to add to outermost	Heavy noEdge	Particles have very high mass and friction values.
D4	Fifteen sub-particles are added with a start force to each of the ‘heavy noEdge’ sub-particles.	selectByType	select the heavy noInner sub-particles	perpetual noOuter	outer collisions disabled
		addToInner	lft click to add to selected particles		
D5	‘heavy noEdge’ particles are transformed one by one into a variety of other heavy particles with complementary tunings, mapping frequency along the y-axis. Results in a subtly shifting drone-like texture	Transform	lft click on ‘heavy noEdge’	heavy Harm	Tuned as partials from fundamental.
				heavy Wendy	Wendy Carlos Alpha tuning.
				heavy Bohlen	Bohlen-Pierce tuning
				heavy Chord	A chord using tempered tuning. Two notes are static and two are mapped to y-axis.

Section	Description	Feature Use		Presets	
E1	The 'perpetual noInner 2' particles have lost all of their momentum through collisions with heavy particles meaning that the high pitched texture they were generating is no longer audible. They are now destroyed.	SelectByType	select all the perpetual noInner 2 particles		
		Destroy	destroy them		
E2	One by one, the contents of the remaining heavy sub-particles are slowed down, and their sub-particles released into the outermost particle where they move but make no noise	Adjust Speed	rgt click on sub-particle to slow down contents		
		RemoveOuter	on sub-particle to release contents		
E3	After the disappearance of the final heavy sub-particle there is no more sound, and the performer zooms out to the point where there is only a white screen.	Zoom Out			

Appendix C *Cube With Magic Ribbons* – Descriptive Score

Also available in www.simonkatan.co.uk/phd/cube.html

Section	Description	Feature Use		Blip Presets	
A1	There is a single track across the middle of the screen with a single reader. The camera is in follow mode and as there are no points of reference to observe the movement, the rendering has the appearance of a static image	Add Long Track			
		Add Reader			
A2	Needle Glitch blips are incrementally added to the track to build up a sequence of high and low filtered noise.	Add Blip	Lft and rgt click and tab combinations used for variations	needle Glitch	**envelope:** AR **draw object:** Straw Line pivots to make flicking motion. Thickness and colour derived from sound parameters **synthDef:** Brown Glitch Uses brown noise passed through a resonant low pass filter with hard attack and decay.
		Toggle follow	occasional dropping out of follow mode for more accurate placement of blips		
A3	Several blips from the 'elec' preset are added at varying points in the sequence	Add Blip	Durations kept short to compliment needle Glitch blips	elec	**envelope:** ASR (length determines duration) **draw object:** Elec moving jagged lines in active state imply an electric current passing between contacts. Number of line points derived from 'frequency' of the synthDef. **synthDef:** elec filtered low frequency noise continues electricity imitations.

Section	Description	Feature Use		Blip Presets	
A4	Several blips from the 'sawTooth' preset are added at varying points in the sequence	Add Blip	Durations kept short	sawTooth	**envelope:** ASR Proportional setting makes attack 0.99 of duration PostDecay used **draw object:** Bean Variable bezier curves between multiple vertices. Three vertices used with control points set to make a tooth-like triangle. For reaction the outer point of the tooth to folds towards the track. In post decay tooth slowly regains its position. **synthDef:** Swell combines a saw wave oscillator with a slow swelling envelope
A5	Several blips from the 'softBasic' preset are added at varying points in the sequence	Add Blip	Durations kept short	softBasic	**envelope:** ASR Proportional setting makes attack 0.99 of duration PostDecay used **draw object:** Bean Variable bezier curves between multiple vertices. Uses configuration of four convex beziers. Inflating motion for reaction. Deflation for Post decay. **synthDef:** Soft Uses a sine oscillator with a soft attack
A6	Several blips from the 'hardBasic' preset are added at varying points in the sequence	Add Blip	Durations kept short	hardBasic	**envelope:** ASR **draw object:** Flipper rectangle of variable offset and height illuminates with a with specified colour and rotates on x,y, or z-axis for reaction. rotation not used here. hue derived from frequency **synthDef:** Basic A sine tone. Hard attack and decay used here.

Section	Description	Feature Use		Blip Presets	
B1	A long track is added perpendicular to the original track. The reader moves onto the new track suddenly ending the previous sequence and the camera rotates to align with the new track.	Add Long Track	perpendicular to first track		
		Adjust Node	close sockets to first track and open sockets to new track		
		Toggle Roll	just as reader changes direction		
B2	A more melodic sequence is built using blips of longer durations from the 'fineSoft' preset.	Add Blip		FineSoft	same as BasicSoft but with fine control of pitch through alternative settings
B3	A third long track is added, and the melody building continues. The reader randomly changes direction at the node between 2nd and 3rd tracks leading to melodic variations, and retrogrades.	Add Long Track		FineSoft	
		Adjust Node	fully open the node between 2nd and 3rd tracks		
C1	First extra reader is added, creating multiple melodic lines. Throughout the rest of section C the reader incrementally continues adding readers.	Add Reader			
C2	Two track loops created with nodes adjoining 2nd and 3rd tracks. 'basicHard' and 'elec' presets are used to make melodic events here.	Add Short Track	to make loops	basicHard	Low frequency, high amplitude version is used to make bass sound
		Add Blip		Elec	
		Adjust Node	open nodes to loops		

Section	Description	Feature Use		Blip Presets	
C3	Two loops extending towards each other stem from the corners of the previous loops. They are populated with short duration 'hardMappedRandom' blips in rising sequences and pointillist–style sequences	Add Short Track	to make loops	hard Mapped Random	A variant of hardBasic which uses paramAttributes' map and random distribution functions to control pitch
		Add Blip	different variants for map and random functions		
		Adjust Node	open nodes to loops		
C4	Other previously used blips are added to the loops	Add Blip		Elec, Sawtooth	
C5	The two chains of loops are joined with a bridging loop. A series of evenly spaced, short 'hardBasic' blips at the same frequency are added.	Add Short Track	to make loops	hardBasic	
		Add Blip			
		Adjust Node	open nodes to loop		
C6	By this point there are many readers making multiple, irregularly repeating cycles of events. This is gradually transformed as all the readers are directed to the final loop, culminating in a single repeated note. Follow mode is switched off for the entirety of section D	Adjust Node	close nodes to point towards final loop		
		Toggle followMode	Switch off in preparation for section D		
D1	A loop is connected to the final loop, making sure that the unconnected side borders the 1st track of section A. Whilst the connecting node is still closed a single blip using a preset 'basicRing' is added Finally the readers are directed into the new loop creating a texture of bell-like sounds	Add Short Track		basicRing	**envelope:** AR **draw object:** Flipper Hue mapped to pitch Rotation on x-axis Speed of rotation derived from frequency **synthDef:** Basic Hard attack and Long decay create a bell-like sound
		Add Blip			
		Adjust Node	point the node towards the final loop		

Section	Description	Feature Use		Blip Presets	
D2	The readers are destroyed one by one, exposing different rhythmic patterns Finally there is only a single reader left.	Destroy Reader			
D3	More 'BasicRing' blips of close frequencies are added on either side of the initial blip. As blips are incrementally the frequency range is expanded with higher notes on the top side of the loop and lower notes on the bottom side Space is intermittently added across the horizontal plane to make room for more blips, subtly changing the rhythm of the melodic sequence	Add Blip		Basic Ring	
		Add Space	Intermittently drag to create small amounts of space in order to add blips		
D4	After the sequence is around twenty blips long, several very low 'Basic Ring' blips with long decays are added in the far bottom corner of the loop	Add Blip		Basic Ring	
D5	A vertical track is added towards the opposite end of the loop, which joins the top and bottom tracks, thus shortening the sequence.	Add Space	Make a little space between the blips to add the track		
		Add Long Track			
		Adjust Node	point both the nodes towards the new track to change the reader path		
D6	Repeat D5 several times until the sequence consists only of the lowest blips				

Section	Description	Feature Use		Blip Presets	
E1	With the low blips still sounding, space is inserted between them and the original track from section A. A new long track and reader are added into the newly formed space.	Add Space			
		Add Long Track	runs parallel to track from section A		
		Adjust node	The resultant node which intersects with the long track from section B must be pointed towards the new track		
		Add Reader			
		Toggle Follow mode	Start following the new reader		
E2	Blips using the 'needleDecay' preset are added to the new track.	Toggle Roll		NeedleDecay	**envelope:** AR **draw object:** Straw With long decay there is random pivoting which reduces with envelope. Thickness and colour are derived from filter frequency and speed **synthDef:** clipDecay Decaying pulsed sound from LFSaw passed though Low and High Pass Filters
		Add Blip	Filter-Frequency and speed varied. Blips are scattered sparsely and evenly across the whole track.		
E3	The preset is switched to 'needleMapDecay' As blips continue to be added in scattered locations, an orderly sequence of blips with gradually increasing filter frequencies and speeds emerges.	Add Blip		NeedleMapDecay	As the name implies this is almost the same as the previous preset with the difference that frequencies and filter frequencies are mapped using paramAttribute's map function.

Section	Description	Feature Use		Blip Presets	
F1	The camera position is changed to render the surface as a plane extending into the distance in three-dimensional space	Toggle Roll			
		Toggle Tilt			
		Zoom Out			
F2	The space is extended once again and a further track is added between section E's track and the track from section A	Add Space			
		Add Long Track	runs parallel to track from section A		
		Adjust node	The resultant node which intersects with the long track from section B must be pointed towards the new track		
		Add Reader			
F3	A variant of 'Needle MapDecay' is used to create an orderly sequence with filter frequencies moving in the opposite direction	Add Blip		NeedleMapDecay	
F4	A further track is added using the same method as F2	same as F2			
F5	A melodic sequence is created on this track using the 'Saw Tooth' preset	Add Blip	Using different lengths and variants for different frequencies	SawTooth	
F6	A reader is added to the original track from section A, creating a recapitulation.	Adjust Node	The intersecting node with the long track from section B must be pointed back towards the section A track		
		Add Reader			
G1	Extra readers moving at different speeds are added to all the tracks, making the texture even denser	Add Reader	Hold mouse button and drag to set speed		

Section	Description	Feature Use		Blip Presets	
G2	A joining track is created between the section A track and the one containing the melodic sequence of 'SawTooth' blips The readers are gathered on the 'SawTooth' track	Add long track	Makes a joining track perpendicular to the two long tracks		
		Adjust Node	Open the node on the section A track . Leave the other node un touched		
G3	The same process is used to move those readers to opposing 'Needle Map Decay' track	same as G2			
G4	A joining track is created between the final loop of section D and the long track from section E The reader travels across ending the very low blip sequence which is by now barely audible.	Add Long Track			
		Adjust Node	open the node on the section D loop		
G5	The same process as G2 is used to move all the readers on the opposing 'Needle Map Decay' track onto the other 'Needle Map Decay' track from section E. Now all the readers are on the same track	same as G2			
G6	A short track is drawn stemming off of the final track at the point where the 'Needle Map Decay' blips are at their tallest (the climax of the sequence) All the readers head up the tracking and stop at its end, creating an abrupt end to the piece.	Add Short Track	Readers will stop at the end as the node has only a single connection		
		Adjust Node	Point the node towards the end of the short track.		

Appendix D *What is Life ?* – Descriptive Score

Also available in www.simonkatan.co.uk/phd/whatislife.html

Section	Description	Feature Use		Presets
A1	The initial chime comes into focus	Move towards focal point	intial chime is automatically selected	
A2	Other chimes appear and spread across the top half of the screen building a regular pulse of climbing frequencies until ten chimes have been added.	Copy chimes (new position)	Freq transpose is positive but varied. Set phase to 0.3	**copyPreset:** fp trans 1 **copy objects:** transpose(freq), amt = y map transpose(phase), amt = userB
		Move towards focal point	Bring the chime towards the focal point but not necessarily the whole way to create a varied texture	
A3	Chimes appear at the bottom half of the screen eventually filling out the lower frequency range until there are ten more chimes.	Copy chimes (new position)	Freq transpose is negative but varied. Set intial phase to 0.25 to displace then to 0.3	**copyPreset:** fp trans 1
		Move towards focal point	as earlier	
A4	Five more chimes appear filling out the texture even further.	Copy chimes (new position)	Freq transpose is negative but varied. Set intial phase to 0.27 to displace then to 0.3	**copyPreset:** fp trans 1
		Move towards focal point	as earlier	
B1	The chimes drift to form a diagonal line spanning the screen	Search mode (fresh set)	selects all the chimes	**movePreset:** spread preserve
		Move chimes	It's implied that move mode must be on and the correct preset selected. These sort of details will be omitted from hereon.	

Section	Description	Feature Use		Presets
B2	Small sections of the sequence are copied in mutated form, subtly thickening the overall texture.	Search mode (fresh set) Inclusive search (1)	select a few chimes with rectangle	**searchPreset:** position **macroStages:** position, filter_blur
		Copy chimes (fixed position)	small changes to freq and phase amts	**copyPreset:** fp transMut **copy objects:** transpose(freq), amt = userA transpose(phase), amt = userB mutate(phase), range = 0.5, dev = 0.25
		Move twds focal point	varying focus	
B3	The chimes all drift to the opposing diagonal with a steeper angle. They take on the appearance similar to a strand of DNA.	Search mode (fresh set)	select all the chimes	**movePreset:** spread phase
		Move chimes		
B4	Sections of the sequence are copied but their phase and frequencies are transposed creating intersecting groups of chimes that create a percussive effect and lend the image a three dimensional quality.	Search mode (fresh set) Inclusive search (1)	select a few chimes with rectangle	**searchPreset:** position
		Copy chimes (fixed position)	Set phase to 0.5 Set freq to wide positive and negative intervals	**copyPreset:** fp trans 2 **copy objects:** transpose(freq) amt = userA transpose(phase) amt = userB
		Move twds focal point	varying focus	
B5	The chimes all drift into a new configuration in a horizontal line across the centre of the screen	Search mode (fresh set)	select all the chimes	**movePreset:** spread freq
		Move chimes		
B6	More groups of chimes are copied in mutated form until the texture becomes so dense that it is no longer possible to pick out individual attacks in any of the frequency range.	Search mode (fresh set) Inclusive search (1)	select a section of chimes with rectangle	**searchPreset:** position
		Copy chimes (fixed position)	Set freq to 0 Set phase around 0.25	**copyPreset:** fp transMut
		Move twds focal point	varying focus	

Section	Description	Feature Use		Presets
C1	Suddenly the focus changes bringing a smaller number of chimes, which are articulating a regular pulse, into sharp focus.	Search mode (fresh set) Inclusive search (2)	1st stage - fundamental = 20, adjust tolerance by eye 2nd stage – adjust tolerance by eye	**searchPreset:** phaseFund **macroStages:** phaseFund, unique, quant
		Equalize z-positions		
		Move away from focal point	until selected group is no longer visible	
		Move focal point	until the selected group is in sharp focus; the other group will be out of focus	
		Invert selection	selects all the out of focus chimes	
		Move away from focal point	until selected group is no longer visible	
		Delete invisible chimes		
C2	Small sections of the resultant sequence are moved to other parts of the screen and copied with changing phases so that they make their own sequences.	Search mode (fresh set) Inclusive search (1)	select a section of chimes with rectangle	**searchPreset:** position
		Move chimes	to an empty space	**movePreset:** shift
		Copy chimes (new position)	set phase amt to a multiple of 0.05 repeat copy enough times for complete filling of phase (eg. 0.25 copy four times)	**copyPreset:** p trans **copy objects:** transpose(phase), amt = userB
		Move twds focal point	varying focus	
C3	Other small sections are copied in the same way but this time with shuffling of the phases and frequencies creating more varied effects	Search mode (fresh set) Inclusive search (1)	select a section of chimes with rectangle	**copyPreset:** fp arrTrans **copy objects:** arrange(freq), arrType = userB, num = 1 arrange(phase), arrType = userB, num = 1 transpose(phase), amt = userA
		Move chimes	to an empty space	
		Copy chimes (new position)	set arrType to shuffle or rotate set phase amt to a multiple of 0.05 repeat copy enough times for complete filling of phase (eg. 0.25 copy four times)	
		Move twds focal point	varying focus	

Section	Description	Feature Use		Presets
C4	A new group is created of regular ascending tones that is arranged in a similar way to the DNA like structure of B3	Search mode (fresh set) Inclusive search (3)	1st stage – fundamental phase =20 2nd stage – set tolerance by eye 3rd stage – mul = 2, offset = 0	**searchPreset:** phaseFund **macroStages:** phaseFund, unique, quant
		Copy chimes (fixed position)	set arrange type to sort -asc	**copyPreset:** fp arrTrans
		Move chimes	center around an empty space	**movePreset:** spread phase
		Move twds focal point	sharp focus	
D1	The sub-groups of chimes spread around the screen are organised into wheel like structures and copied and transposed, gradually creating a dense mechanical looking texture.	Search mode (fresh set) Inclusive search (1)	select a sub group with rectangle	**searchPreset:** position
		Move chimes	gather into wheel like formation	**movePreset:** gather
		Copy chimes (new position)	vary frequency to achieve a good overall balance phase is less important here	**copyPreset:** fp trans 2
		Move towards focal point	varying focus	
D2	All the texture suddenly moves out of focus. Then new rhythms emerge as certain chimes are brought forward	Move focal point	so that everything is out of focus but still visible	**searchPreset:** phaseFund
		Search mode (fresh set) Inclusive search (1)	low fundamental values for slow tempi	
		Equalize z-positions		
		Move twds focal point	varying focus	

Section	Description	Feature Use		Presets
D3	Whilst this process is occurring other chimes disappear. Gradually a minor pentatonic tonality emerges	Search mode (fresh set) Inclusive search (1) Exclusive search (1)	1st stage – start with low fundamental values and gradually increment 2nd stage – sieve = minor pentatonic, offset = 0	**searchPreset:** fundSieve **macroStages:** phaseFund, sieve
		Move away from focal point	until not visible	
		Delete invisible chimes		
E1	When only the pentatonic tonality remains, the chimes are once again organised into DNA like strands in vertical positions across the screen	Search mode (fresh set) Inclusive search (1)	use minDistances between 15 and 30	**searchPreset:** unique2 **macroStages:** unique, unique
		Move chimes	in an empty space	**movePreset:** spread phase
		Equalize z-positions		
		Move twds focal point	varying focus	
		Save to memory	save each of these for quick selection later	
E2	Now the strands are copied into descending sequences. The tonality remains the same but DNA strands exhibit more complex movements	Recall memory	for the strand to be copied this time	**copyPreset:** fpArrSieve **copy objects:** arrange(freq), arrType = userB, num = userA arrange(phase), arrType = userB, num = userA sieve, offset = 0
		Copy chimes	use sort - asc	
		Adjust pivots	add at least one pivot, vary other parameters	
		Move twds focal point	varying focus	
		Save to memory	for quick selection later	
E4	Now faster and slower moving strands appear.	Recall memory		**copyPreset:** fsTransSieve **copy objects:** transpose (freq), amt = userA transpose(speed), amt = userB sieve, offset = 0
		Copy chimes	spread frequencies evenly	
		Adjust pivots		
		Move twds focal point	varying focus	
E5	Finally the tonality begins to break down	Recall memory		**copyPreset:** ftransSieve **copy objects:** transpose (freq), amt = userA sieve, offset = userB
		Copy chimes	use offsets of greater than 1	
		Adjust pivots		
		Move twds focal point	varying focus	

Section	Description	Feature Use		Presets
F1	Suddenly all the chimes begin to gather around a central point, and then burst into different formations	Search mode (fresh set)	Selects all chimes	**movePreset:** gather
		Move chimes		
		Recall memory	select some groups	
		Adjust pivots	make phase mul negative for outward facing shapes	
F2	Gradually the chimes begin to fade into the background until none are left.	Search mode (fresh set) Inclusive search(1)	gradually increase tolerance as process is repeated	**searchPreset:** matchUnique **macroStages:** matchUnique, unique
		Move away from focal point	until not visible	
		Delete invisible chimes		