Audiovisual Granular Synthesis
Creating Synergistic Relationships Between Sound and Image

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

Joshua Paris Batty
BMus Improvisation
BA (Hons) Creative Media

School of Media and Communication
College of Design and Social Context
RMIT University

August 2014
Declaration

I certify that except where due acknowledgement has been made, the work is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program; and, any editorial work, paid or unpaid, carried out by a third party is acknowledged; and ethics procedures and guidelines have been followed.

Joshua Batty

26.08.2014
Acknowledgements

I would like to thank my senior supervisors, Kipps Horn and Stefan Greuter, who were always there to provide advice, read countless drafts and suggest many more edits – especially during peak moments of frustration over the last four years. Without their encouragement and guidance this exegesis would not be what it is today.

I would also like to thank John Power for his helpful insights during my first year and, more recently, Adam Nash for agreeing to supervise my final year of study. Adam offered exceptional advice and insight relating to my work and helped me see the finish line when I was starting to lose hope.

Pierre Proske deserves a special thank you for helping me understand countless C++ and openFrameworks problems, especially during the first two years. I would also like to thank the openFrameworks and stack overflow communities for saving my sanity on several occasions.

I am eternally grateful to have formed MindBuffer with Mitchell Nordine and for his brilliant insights relating to music/art and code. Furthermore, I would like to thank Jake Rose for signing us to Enig’matik records and to all the promoters that have given us the opportunity to perform nationally and internationally over the past four years.

For funding, I sincerely thank the Australian Government and RMIT for granting me an Australian Postgraduate Award (APA) for the duration of my studies. Without the financial support of this scholarship I would not be writing these words today.

Dr Campbell Aitken of Express Editing Writing and Research provided professional editing services in accordance with the Institute of Professional Editors’ Guidelines for editing research theses.

My amazing fiancée Alice Hobday deserves an extra special thank you for both dealing with my daily ups and downs during this time and for patiently waiting four years for my PhD to end so that we can get married. Thanks to all my friends who have good-naturedly been waiting for me to return to life and have a drink with them again. Finally, a sincere thank you to my parents, Clinton and Julie Batty, not only for reading every single draft and offering feedback, but also for supporting my musical education and career for as far back I can remember. I am truly blessed to have such an amazing family and friends around me.
Without doubt this exegesis will contain errors, omissions and over-simplifications, for which I take absolute responsibility, while hoping that the rest of the material will be enough to stimulate insights and new trains of thought.
# Table of Contents

Declaration .................................................................................. i
Acknowledgements ....................................................................... ii
Table of Contents ........................................................................ iv
List of Figures ................................................................................ xi
List of Tables ................................................................................ xv
Abstract ....................................................................................... 1

1 Introduction .............................................................................. 3
   1.1 Overview ............................................................................ 3
   1.2 Research Components ....................................................... 3
   1.3 Definition of Terms ............................................................ 3
      1.3.1 Audiovisual Granular Synthesis (AVGS) ..................... 3
      1.3.2 Digital Signal Processing (DSP) ................................. 4
      1.3.3 Real-Time Rendering ............................................... 4
      1.3.4 Kortex ....................................................................... 4
      1.3.5 Perceptual Characteristics ......................................... 5
   1.4 Background ......................................................................... 5
   1.5 Audiovisual Art .................................................................... 6
   1.6 Audio and Visual Granular Synthesis ................................. 6
      1.6.1 Visual Granular Synthesis ........................................... 8
   1.7 Research Aims and Significance ......................................... 10
   1.8 Scope and Limitations ....................................................... 11
   1.9 Research Methodology ...................................................... 12
   1.10 Exegesis Outline ............................................................. 12

2 Theoretical, Technological and Artistic Influences ...................... 13
Table of Contents

2.1 Overview ......................................................................................................................... 13

2.2 Visual Music ....................................................................................................................... 13
  2.2.1 Historical Background ................................................................................................. 13
  2.2.2 Early Visual Music Technology .................................................................................. 14
  2.2.3 Identifying the Complementary .................................................................................. 17
  2.2.4 Synchresis .................................................................................................................... 18
  2.2.5 Parameter Mapping ...................................................................................................... 19
  2.2.6 Modern Computation ................................................................................................. 19
  2.2.7 Transcoding ................................................................................................................ 21
  2.2.8 Perception .................................................................................................................... 23
  2.2.9 Compositional Strategies ........................................................................................... 24
  2.2.10 Subjectiveness ........................................................................................................... 24

2.3 Live Cinema ....................................................................................................................... 25
  2.3.1 Live Cinema in Relation to DJs & VJs ......................................................................... 25
  2.3.2 ‘Liveness’ of Live Cinema .......................................................................................... 26
  2.3.3 Meaning and the Performer/Audience Relationship .................................................. 26
  2.3.4 Cinema vs Live Cinema .............................................................................................. 26
  2.3.5 Build your own Tools .................................................................................................. 27

2.4 Granular Synthesis ............................................................................................................ 28
  2.4.1 Quantum Sound ........................................................................................................... 28
  2.4.2 Pioneers of Quantum Sound ....................................................................................... 29
  2.4.3 Technology .................................................................................................................. 31
  2.4.4 Composition ................................................................................................................ 33
  2.4.5 Software ...................................................................................................................... 33
  2.4.6 Timescales .................................................................................................................... 35
  2.4.7 Control Strategies ....................................................................................................... 36

2.5 Illusionary Art ................................................................................................................... 36
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.5.1</td>
<td>Human Sensory Perception</td>
<td>37</td>
</tr>
<tr>
<td>2.5.2</td>
<td>Rhythmic Photic Stimulation</td>
<td>38</td>
</tr>
<tr>
<td>2.6</td>
<td>Community of Practice</td>
<td>41</td>
</tr>
<tr>
<td>2.6.1</td>
<td>My Personal Backstory</td>
<td>41</td>
</tr>
<tr>
<td>2.6.2</td>
<td>Chris Cunningham</td>
<td>43</td>
</tr>
<tr>
<td>2.6.3</td>
<td>Aphex Twin / Weirdcore</td>
<td>44</td>
</tr>
<tr>
<td>2.6.4</td>
<td>Squarepusher</td>
<td>45</td>
</tr>
<tr>
<td>2.6.5</td>
<td>Reflexus</td>
<td>46</td>
</tr>
<tr>
<td>2.6.6</td>
<td>Modell5</td>
<td>47</td>
</tr>
<tr>
<td>2.6.7</td>
<td>Varp9</td>
<td>48</td>
</tr>
<tr>
<td>2.6.8</td>
<td>Autechre – Gantz Graf</td>
<td>49</td>
</tr>
<tr>
<td>2.7</td>
<td>Summary</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>Methodology</td>
<td>51</td>
</tr>
<tr>
<td>3.1</td>
<td>Overview</td>
<td>51</td>
</tr>
<tr>
<td>3.2</td>
<td>Ontology and Epistemology</td>
<td>51</td>
</tr>
<tr>
<td>3.3</td>
<td>Practice-Based Research</td>
<td>51</td>
</tr>
<tr>
<td>3.4</td>
<td>Field of Practice</td>
<td>52</td>
</tr>
<tr>
<td>3.5</td>
<td>Research Questions</td>
<td>52</td>
</tr>
<tr>
<td>3.6</td>
<td>Software Prototypes</td>
<td>53</td>
</tr>
<tr>
<td>3.6.1</td>
<td>Hacking</td>
<td>53</td>
</tr>
<tr>
<td>3.6.2</td>
<td>Development and Test Platform</td>
<td>53</td>
</tr>
<tr>
<td>3.7</td>
<td>Evaluation</td>
<td>54</td>
</tr>
<tr>
<td>3.7.1</td>
<td>Qualitative Data</td>
<td>55</td>
</tr>
<tr>
<td>3.7.2</td>
<td>Quantitative Data</td>
<td>56</td>
</tr>
<tr>
<td>3.8</td>
<td>Validation of Key Research Outcomes</td>
<td>58</td>
</tr>
<tr>
<td>3.9</td>
<td>Summary</td>
<td>58</td>
</tr>
<tr>
<td>4</td>
<td>Intention and Aesthetics</td>
<td>59</td>
</tr>
</tbody>
</table>
4.1 Overview .................................................................................................................. 59
4.2 Intentions ..................................................................................................................... 59
  4.2.1 Why Real-Time is Important ............................................................................... 59
  4.2.2 Immersive Environments ..................................................................................... 60
  4.2.3 Musical Context .................................................................................................... 60
4.3 Aesthetics .................................................................................................................... 61
  4.3.1 Audio Visual Granular Synthesis (AVGS) in Detail ................................................ 61
  4.3.2 Visual Synths ........................................................................................................ 63
4.4 Summary .................................................................................................................... 65
5 Performance Iterations ................................................................................................... 66
  5.1 Overview ................................................................................................................... 66
  5.2 Iteration 1 – Brown Alley (Melbourne) 2011 ............................................................. 66
      5.2.1 Performance Strategy ....................................................................................... 67
      5.2.2 Reflection ........................................................................................................... 69
  5.3 Iteration 2 – Eclipse Festival (Cairns) 2012 ................................................................. 70
      5.3.1 Performance Strategy ....................................................................................... 71
      5.3.2 Reflection ........................................................................................................... 72
  5.4 Iteration 3 – Burning Man (Nevada) 2013 ................................................................. 73
      5.4.1 Performance Strategy ....................................................................................... 74
      5.4.2 Reflection ........................................................................................................... 75
  5.5 Iteration 4 – UFO Club (Sydney) 2014 ..................................................................... 77
      5.5.1 Performance Strategy ....................................................................................... 78
      5.5.2 Reflection ........................................................................................................... 79
  5.6 Introduction to the Final Iteration – Examination (Melbourne) 2014 ....................... 81
      5.6.1 Performance Strategy ....................................................................................... 81
  5.7 Summary .................................................................................................................... 83
6 Instrument Design .......................................................................................................... 85
6.1 Instrument Design .................................................................................................................. 85
  6.1.1 Programming Framework ................................................................................................. 86
  6.1.2 Audio Synthesis Libraries ................................................................................................. 87
Synthesis ToolKit ............................................................................................................................ 88
Maximillian ...................................................................................................................................... 88
Tonic ............................................................................................................................................... 88
Gamma ............................................................................................................................................ 89
  6.1.3 Technology ......................................................................................................................... 90
Video Frame Loading ...................................................................................................................... 90
Granular Audio Video Synchronisation .......................................................................................... 92
  6.1.4 Visual Shader Synths ......................................................................................................... 95
  6.1.5 FM Synthesis Engine ......................................................................................................... 96
  6.1.6 Visual Effects ..................................................................................................................... 98
Slit Scanning .................................................................................................................................. 99
Triangulation .................................................................................................................................. 100
Optical Flow ................................................................................................................................... 101
Box2d Physics ............................................................................................................................... 102
Pixel Physics .................................................................................................................................. 103
  6.1.7 Triple Screen AVGS .......................................................................................................... 104
6.2 Audiovisual Relationships ....................................................................................................... 105
  6.2.1 AVGS Relationships ......................................................................................................... 105
Start Time and Duration ................................................................................................................ 106
Jitter and Jitter Quantise ................................................................................................................ 106
Speed .......................................................................................................................................... 106
Pitch ............................................................................................................................................ 106
Grain Duration .............................................................................................................................. 107
Density ......................................................................................................................................... 107
Grain Envelope ........................................................................................................... 107

6.2.2 Post Processing Visual and DSP Relationships ............................................. 108

6.3 Performance Strategies....................................................................................... 110

6.3.1 Simple Harmonic Motion ............................................................................... 110

6.3.2 Presets ............................................................................................................. 113

6.3.3 Tweens ........................................................................................................... 113

6.3.4 Timeline .......................................................................................................... 114

6.3.5 JenOSC ............................................................................................................ 118

6.4 Interface and Mappings ...................................................................................... 120

6.4.1 Controller Considerations ............................................................................. 120

6.4.2 Mapping the Controller ................................................................................ 122

6.4.3 Controller Mappings ....................................................................................... 122

Timeline .................................................................................................................... 124

Harmonic Motion ..................................................................................................... 124

Tweening .................................................................................................................. 124

Presets ....................................................................................................................... 125

Pad Modes ................................................................................................................ 125

FX ............................................................................................................................... 125

Post-Processing & DSP ........................................................................................... 126

Audiovisual Granular Synthesis ............................................................................. 126

6.4.4 Touch Screen .................................................................................................. 126

6.5 Summary .............................................................................................................. 127

7 Discussion .............................................................................................................. 128

7.1 Overview ............................................................................................................. 128

7.2 Visual Short Term Memory ................................................................................ 128

7.3 Performance Interfaces ....................................................................................... 130

7.3.1 MIDI Controllers ............................................................................................ 130
7.3.2 Touch Screens ........................................................................................................131
7.4 Audiovisual Relationships .........................................................................................132
7.5 Performance and Control Strategies .........................................................................133
7.6 Technology ................................................................................................................135
7.6.1 Hardware Performance Boost ................................................................................135
7.6.2 Audiovisual Ring Buffer .........................................................................................135
7.7 Summary ....................................................................................................................136
8 Conclusion ......................................................................................................................137
8.1 Overview ....................................................................................................................137
8.2 Research Outcomes .................................................................................................137
8.2.1 Adapting Granular Synthesis for Visual Processing ..............................................137
8.2.2 Fusing the Audiovisual Construct .........................................................................138
8.2.3 Real-Time Performance of Audiovisual Art .........................................................139
8.2.4 Contributions ........................................................................................................140
8.3 Limitations ................................................................................................................141
8.4 Areas and Directions of Future Research .................................................................141
8.5 Potential Applications .............................................................................................142
8.6 Final Conclusions .....................................................................................................143
References .......................................................................................................................144
Appendix ..........................................................................................................................156
List of Figures

Figure 1: Borderlands iOS granular synthesiser interface for manipulating the main parameters of granular synthesis (Carlson and Wang, 2012) ................................................................. 8

Figure 2: 100 ms audiovisual grain (Batty, Horn, Greuter, 2013) ...................................................... 9

Figure 3: Schemas for matching pitch to colour scales, 1704–2004 (Collopy, 2009) ......................... 15

Figure 4: Some of the experimental patterns Fischinger created to generate synthetic waveforms (Fischinger, 1932) ......................................................................................... 16

Figure 5: Whitney’s setup at IBM for filming graphics from a computer screen (Moritz, 2012) .......... 17

Figure 6: The Atari Video Music unit (Cruiser, 2012) ........................................................................ 20

Figure 7: MilkDrop music visualiser plugin for the Winamp media player (MilkDrop, 2003) ......... 21

Figure 8: VOSIS interface showing three visual regions that have each been visually processed in different ways to achieve various sonic results (McGee, 2013) ......................................................... 22

Figure 9: TVestroy’s immersive performance setup with all visuals and sound generated from the same data (Stec, 2008) .......................................................................................... 23

Figure 10: A diagram of Gabor’s method for summing previously separated sound segments (Gabor, 1946, p. 447) ........................................................................................................... 30

Figure 11: Kinematical frequency converter with sound film (Gabor, 1946, p. 446) ......................... 31

Figure 12: Burroughs B6700 mainframe computer (B6700 Display Panel n.d.) ............................... 32

Figure 13: Modern granular synthesis programming environments ................................................. 34

Figure 14: Duchamp’s original optical machine, the Rotary Demisphere (Duchamp, 1925) ............ 39

Figure 15: Spiral patterns Duchamp created for the Rotorelief to induce illusionary optical effects (Duchamp, 1935) ................................................................................................................. 39

Figure 16: Brion Gysin and William S. Burroughs experiencing the stimulating effects of a Dreamachine (Gysin & Summerville, 1974) .................................................................................. 40

Figure 17: Jitterbugs audiovisual instrument created for my honours project .................................. 43

Figure 18: Chris Cunningham performing his triple-screen laser show live at Sonar, Barcelona, 2011 (Chris Cunningham Live n.d.) ................................................................................................. 44

Figure 19: Aphex Twin and Weirdcores’ live performance at the Lowlands festival, Netherlands, 2011 (Aphex Twin @ Lowland 2011) ......................................................................................... 45
Figure 20: Squarepusher performing his live audiovisual Ufabulum show at Simple Things Festival, Bristol, 2012 (Dolan, 2012) ................................................................. 46

Figure 21: Live Reflexus performance featuring an actor interacting with his digital self (Maduell, 2010) .................................................................................................................. 47

Figure 22: Live performance of Modell5 by Granular Synthesis, MAK Museum of Fine Arts, Vienna, 1995 (Sengmueller, 1995) ................................................................................. 48

Figure 23: Interface of Varp9 custom made for the group ‘Granular Synthesis’ (Langheinrich, 1999) 49

Figure 24: Screenshot from the Autechre Gantz Graf music video clip (Autechre - Gantz Graf, 2002) ................................................................................................................................. 50

Figure 25: Mixed method research design ............................................................................................................. 54

Figure 26: The cyclical process of action and reflection used to evaluate and evolve the Kortex prototype for public performance ................................................................................. 55

Figure 27: Performance analysis tools used during the development of Kortex to ensure consistent performance of 60 fps ............................................................................................... 56

Figure 28: OpenGL Profiler statistics used to identify expensive CPU and GPU calculations in Kortex ................................................................................................................................. 57

Figure 29: Three grain reordering patterns that display how the audiovisual grains are re-ordered over time ........................................................................................................................................ 62

Figure 30: Photograph of MindBuffer’s first gig (Brown Alley, Melbourne, Australia, June 2011) .... 67

Figure 31: MindBuffer Ableton set showing each track divided into Drums, Bass, Melody and Atmosphere stems with a Max4Live device on each track on the bottom left .................. 68

Figure 32: Initial performance framework for synchronising audio and visuals between two laptops .69

Figure 33: Mindbuffer set at Eclipse Festival (Cairns, Australia, November 2012) ........................................ 71

Figure 34: Custom Max/MSP patch that enabled coloured visual feedback on the midi controller ..... 72

Figure 35: Burning Man laptop protection ........................................................................................................... 74

Figure 36: Gypsy Nebula stage plan displaying the three non-rectangular projection surfaces, dotted lines represent stage enclosure ...................................................................................... 76

Figure 37: MindBuffer Burning Man set 2, photo taken from my performance perspective, (30th August, Nevada, 2013) ......................................................................................................... 77

Figure 38: Seventy metre wide stage during the MindBuffer set at Burning Man (30th August, Nevada, 2013) ................................................................................................................................ 77

Figure 39: PadMode MIDI controller mappings to enable rhythmic triggering functionality ................ 79

Figure 40: Tri-directional feedback loop of influence between the performer, instrument and audience ................................................................................................................................. 83
Figure 41: Kortex user interface featuring GUI controls, video and shader thumbnails, timeline automation and visual preview window ................................................................. 86

Figure 42: Side by side comparison of QuickTime averaging 20 fps while my custom algorithm performs at a consistent 60 fps ........................................................................... 94

Figure 43: An allocated section of memory in a circular ring and how data is inserted, read and deleted from the memory ...................................................................................... 95

Figure 44: Screenshot of AVGS burning into the visual shader synth layer rendering in the background ................................................................. 96

Figure 45: FM synthesis engine in Kortex ................................................................................................................................. 97

Figure 46: FM synthesis modulation routing interface ............................................................................................................. 98

Figure 47: Grey-scale time displacement maps used by the slit scanning effect ................................................................................ 99

Figure 48: Screenshot of slit scanning being applied to Kortex using a square pixel displacement map .................................................. 99

Figure 49: Original triangulation algorithm. Triangle size is randomly generated......................................................... 100

Figure 50: Updated triangulation algorithm featuring better blob detection and uniform triangle packing ................................................................. 101

Figure 51: The movement of the dancers ribbon triggers the optical flow ................................................................. 102

Figure 52: Randomly distributed rectangles provide a physics pixilation effect to the image underneath .................................................................................. 103

Figure 53: A screenshot showing a particle explosion occurring to a meditating girl’s head .................... 104

Figure 54: Picture of Kortex running three AVGS players simultaneously over three monitors in real-time ................................................................................................................. 104

Figure 55: Grain windowing shapes (a) Gaussian, (b) Cosine, (c) Triangle, (d) Rectangle, (e) Blackman-Harris................................................................. 108

Figure 56: Modular assignment interface to define audiovisual relationships ......................................................... 109

Figure 57: Position slider sets the same play position for each player ................................................................................... 111

Figure 58: Play position of each player out of phase with each other ................................................................................... 111

Figure 59: The phases of players 1 and 3 are locked with offset and period set to 0 .................................................... 112

Figure 60: The clamp function squared off the motion values when minimum range is exceeded .......... 113

Figure 61: Available easing types for transitions between the instrument’s states ........................................... 114

Figure 62: Max4Live device containing Kortex’s controllable parameters ......................................................... 115
Figure 63: Ableton Live set using four automation tracks to control Kortex via OSC messages sent from the Max4Live devices .......................................................... 116

Figure 64: Available tweening curves for key frame automation within Kortex ....................... 117

Figure 65: Adjustable timeline start and end time parameters loop the play back of automation data in real-time .......................................................... 118

Figure 66: JenOSC modular routing interface for creating relationships between musical information and visual parameters in Kortex ........................................... 119

Figure 67: Initial design prototype of the custom controller ................................................... 121

Figure 68: Livid CNTRL:R midi controller that was selected as the performance interface for Kortex .................................................................................. 121

Figure 69: Kortex and CNTRL:R interface mappings ................................................................. 123

Figure 70: Custom iPad app developed to trigger audiovisual content .................................. 127

Figure 71: Overlapping visual grains ...................................................................................... 129

Figure 72: Illusionary overlapping visual grains ...................................................................... 130

Figure 73: Touch screen interface for manipulating the grain reordering patterns ................. 132

Figure 74: AVGS demo software user interface ...................................................................... 156

Figure 75: Presenting my first audiovisual software at Overlaps, Melbourne 2009 .............. 158

Figure 75: Kortex alpha screenshot ....................................................................................... 159

Figure 76: GUI controls for the three independent AVGS players ......................................... 160

Figure 77: Kortex Version 1 – content switcher user interface ............................................. 160

Figure 78: Early Kortex Version 2 – screenshot featuring brand new GUI and DSP functionality .... 161

Figure 79: Early Kortex Version 2 – featuring visual post processing and timeline support ....... 162

Figure 80: Final Kortex Version 2 user interface ................................................................. 162

Figure 81: MAX patch developed for Iteration 1 ................................................................. 164

Figure 82: Generative audio MAX patch and Lemur touch screen used during Iteration 1 ......... 164

Figure 83: OpenFrameworks live visual software developed during Iteration 2 and 3 .......... 165

Figure 84: MindBuffer performing live at Brown Alley, Melbourne July 2013 .................... 166

Figure 85: AVGS texture wrapped around 3D GLSL shader geometry ................................ 167

Figure 86: AVGS texture wrapped around 2D GLSL shader geometry .............................. 167
List of Tables

Table 1: Comparison of audio synthesis programming environments 34
Table 2: Potential correlations between audio and visual characteristics, adapted from Olofsson (2009) 64
Table 3: Comparison of open source audio synthesis libraries 90
Table 4: Default audiovisual relationships in Kortex 109
Abstract

The aims of this research were to investigate how an audio processing technique known as granular synthesis can be translated to a visual processing equivalent, and to develop software that fuses audiovisual relationships for the creation of real-time audiovisual art. In order to carry out this project, two main research questions were posed. The first question was: how can audio processing techniques such as granular synthesis be adapted and applied to influence new visual performance techniques, and the second question was: how can computer software synergistically integrate audio and visuals to enable the real-time creation and performance of audiovisual art.

The project at the centre of my research was the creation of a real-time audiovisual granular synthesis instrument named Kortex. The research project involved a practice-based methodology and used an iterative performance cycle to evaluate and develop the Kortex prototype. These included performing iterations of the Kortex prototype at a number of local, interstate and international events.

Kortex facilitates the identification of shared characteristics found between sound and image at the micro and macro level. The micro level addresses individual audiovisual segments, or grains, while the macro level addresses post-processing effects applied to the stream of audiovisual grains. Audiovisual characteristics are paired together by the user\textsuperscript{1} at each level, enabling composition with both media simultaneously. This provides the audiovisual artist with a dynamic approach for the creation of new works.

Creating relationships between image and sound is highly subjective, yet an artist may use a mathematical, metaphorical/intuitive or intrinsic approach to create a convincing correlation between the two media. The mathematical approach expresses the relationship between sound and image as an equation. Metaphorical/intuitive relationships are formed when the two media share similar emotional or perceptual characteristics, while intrinsic relationships occur when audio and visual media are synthesised from the same source.

\textsuperscript{1} I use the term ‘user’ throughout this exegesis as a generic description of persons using Kortex as a performer, composer or artist.
Performers need powerful control strategies to manipulate large collections of variables in real-time. I found that pattern-generating modulation sources created overlapping phrases that evolved the behaviour of audiovisual relationships. Furthermore, saving interesting aesthetics that emerged into banks of presets, along with the ability to slide from one to the next, facilitated powerful transformations during a performance.

The project has contributed to the field of audiovisual art, specifically to the performance work of DJs and VJs. Kortex provides a single audiovisual composition and performance environment that can be used by DJs and VJs for creative collaboration. Kortex has enormous potential for adoption by the DJ/VJ community to assist in the production of tightly synchronised real-time audiovisual performances.
1 Introduction

1.1 Overview
This chapter outlines both the context and motivation for my research project. I describe the components that are submitted as part of this doctoral project, define the key terms used within the exegesis, and present a brief historical background to situate my project. This is followed by a discussion of the research project’s significance and the specific aims. I then outline the scope and limitations of my research and briefly summarise my research methodology. This chapter concludes with a section summarising the structure of the exegesis.

1.2 Research Components
In accordance with RMIT University guidelines (2002), my research project consists of two parts. The first part is the written exegesis, which describes the purpose, context and theoretical background of the research project as well as the process of knowledge production. In essence, the exegesis answers the questions of how the project was developed and what was achieved. The second part is the observable and durable record of the completed project, presented on a USB stick that accompanies this exegesis. To demonstrate the features of Kortex in an easily accessible format, the USB stick contains a basic version of Kortex demonstrating live AVGS, as well as a portfolio of several movies and screen recordings.

1.3 Definition of Terms
In the following section I describe the key terms audiovisual granular synthesis (AVGS), digital signal processing (DSP), real-time rendering, perceptual characteristics and Kortex as they were understood in this project.

1.3.1 Audiovisual Granular Synthesis (AVGS)
Granular synthesis is an audio processing technique that uses tiny segments of sound (grains) to create animated sonic textures or to facilitate time stretching/freezing/smearing and pitch shifting/smearing effects (Bencina, 2006). The duration of each grain typically extends from the threshold of timbre perception up to the duration of short sound objects (1 ms – 100 ms). Using granular synthesis, each individual grain can be processed and redistributed to form new sounds. Visual granular synthesis
follows a similar deconstruction process by which visuals are broken down into micro-segments, manipulated and rearranged to form new visual material. Applying the process of granular synthesis to both audio and visual material reduces that material to its smallest perceivable properties.

1.3.2 Digital Signal Processing (DSP)
Digital signal processing is the process of mathematically manipulating digital streams of data. DSP is referred to in this exegesis as digital effect processing applied to incoming audio data.

1.3.3 Real-Time Rendering
Real-time rendering refers to the computer’s ability to calculate and display images rapidly on a computer. The rate at which images are displayed on screen is measured in frames per second (fps) or in hertz (Hz). While Hz is commonly used to describe the refresh rate of a television set or computer monitor, the term fps is used when measuring software performance in games and other real-time graphics software. Akenine-Moller and Haines stated that an application displaying at 15 fps can be considered real-time up until 72 fps, when the update rate becomes effectively undetectable by the human visual system (Akenine-Moller & Haines, 2002, p. 1). Although the standardised video frame rates for Phase Alternating Line\(^2\) (PAL) is 25 fps, it is quite common for modern games to perform at 60 fps. However, while frame rates exceeding 60 fps are desirable for AVGS, the liquid crystal display (LCD) monitors, laptop and projectors that were used during development and performance had an upper refresh rate limit of 60 Hz. As a result, my research used 60 fps as the benchmark against which real-time performance of the software instrument was evaluated.

1.3.4 Kortex
Throughout this exegesis, I refer to the prototype software instrument created for my project as “Kortex”. Kortex is a unified audiovisual composition and performance environment. By ‘unified’ I mean that no secondary tools or software are required to compose and perform audiovisual art; everything that is required is embedded within the software instrument. Kortex may interfaced with using the graphical user interface (GUI), an external musical instrument digital interface (MIDI)

\(^2\) PAL is a video signal format used in various parts of the world including Australia and the UK. PAL uses amplitude modulation for the video information, and frequency modulation for the audio information. The Free On-line Dictionary of Computing. (http://dictionary.reference.com/browse/phase alternating line)
controller and touch screen interface. The development of Kortex began in January 2011 and continues to the present day. The instrument was developed on Mac OS X using openFrameworks and a collection of open-source C++ libraries. Kortex and its development process are described in more detail in Chapter 6.

1.3.5 Perceptual Characteristics
In this exegesis, I refer to both the nature and significance of aesthetic characteristics resulting from the synthesis of audio and visual elements. These characteristics can be experienced through auditory and visual senses and, as such, can be referred to as perceptual characteristics. The perceived audiovisual characteristics can be combined or paired through a process of subjective decision-making to form perceptual relationships. The characteristics of the relationships can be perceived in whole audiovisual works and in the individual auditory and visual elements that make up the finished work. A detailed discussion of the characteristic relationships can be found in Chapters 4 and 6.

1.4 Background
My project is situated within a long-standing historical fascination with the integration of the phenomena of sound and image. In, the following paragraphs I summarise the research of Edmonds and Pauletto (2004), Kotz (2009), and Daniels and Naumann (2010, 2011) to provide a brief historical background relating to audiovisual integration.

From pre-antiquity to the present day, humans have sought an overarching formula that would unite the phenomena of sound and vision. These endeavours can be traced back to the time of ancient Greek thinkers, particularly Pythagoras (570 BC – 495 BC), who searched for a mathematical expression that would unify sound, art, nature and the planets in an overarching cosmic harmony. However, Isaac Newton (1642 - 1727) discovered that light and sound are two completely separate phenomena (1704). Because sound occurs as fluctuations in air pressure and light exists as a small part of the electromagnetic spectrum, it is not possible to integrate sound and vision using an overarching mathematical expression. The one place where sound and image actually meet is in human perception (Foley & Matlin, 2009). When the brain receives auditory and visual stimuli, synthesis occurs and the two phenomena are perceived as a whole.

Twentieth century technological developments facilitated the integration of light and sound. In the 1920s, when film and the phonograph began to be combined, light and sound were reproduced onto a physical medium for the first time. Audio and visual technological developments continued in the 1960s with analogue electronics. With the introduction of digital technology during the 1980s, all
forms of media were able to be broken down into bits and bytes. Suddenly, media such as text, sound and image could be disconnected from their generating devices and run as emulations on a computer. Although sound and images’ differences prevent their physical unification, digitisation of all media into binary code technically united sound and image through computation. As a result of digitisation, sound and image can be manipulated and translated at will, dissolving the technical and aesthetic boundaries that once separated the two phenomena. This created what Golan Levin described as “inexhaustible, infinitely variable, time-based, audiovisual ‘substance’ that can be created, manipulated and deleted in real-time” (Levin 2000, p. 3).

1.5 Audiovisual Art
Audiovisual experiences abound today, ranging from synchronised music video clips and music visualisations to the laser/light shows playing nightly in clubs and festivals around the world (McGinness, 2007). The umbrella term ‘audiovisual art’ encompasses a multitude of vastly differing genres and audiovisual aesthetic experiences. However, there are sub-genres of audiovisual art, such as live cinema and VJ/DJ\(^3\) performance that focus on creating non-narrative, abstract, audiovisual experiences. Artists creating experimental and abstract forms of audiovisual art seek to develop their own unique voices by building custom tools. Jones (2008) suggested that these artists seek to elude the imposed aesthetics that come with off-the-shelf software, and instead “spend just as much time building tools as making works” (p. 4). My project is situated within this non-narrative/abstract/experimental side of audiovisual art.

1.6 Audio and Visual Granular Synthesis
In the 1940s, physicist Dennis Gabor (1946) proposed that any sound could be decomposed into discrete acoustic quanta or grains with independent time, amplitude and frequencies. As Roads (2001) observed, at this atomic level of detail, the concept of micro-sound is able to be explored through granular synthesis – a form of particle synthesis – by allowing musicians access to these quantum acoustical spectra for composition so as to probe the various modes of micro-temporal perception. This implies that the micro-acoustic events spawned through granular synthesis present new compositional elements at, as it were, the level beneath a note or tone. Thus, through deconstructing

\(^3\) Originally meaning ‘disc jockeys and video jockeys’ – people who mix records or visual imagery for an audience.
elements of an audio waveform and applying microprocessing to individual grains, these elements can be re-arranged and reconstructed in a compositional process. In this way a composer’s tool set and sound palette is extended to include micro-electronic acoustic elements as well as the more traditional tones and micro-tones available through conventional performance and compositional techniques. The same deconstruction methodology can be applied to visual material. Once a single element has been deconstructed into the smallest possible unit, it can then be mapped to the audio grain and re-composed into larger compositional works.

My research extends the idea of micro-sound to visual material by approaching individual frames of video as visual particles. As a single frame is the smallest time duration unit into which my software can deconstruct a video file, each frame is treated as a visual grain. The micro-sound compositional techniques suggested by Roads (2001) are then able to be translated across to the visual domain when composing with ‘micro-image’.

In order to compose with audiovisual particles, Kortex allows for manipulation of the following common parameters found in a traditional audio granular synthesiser such as the iOS application Borderlands seen below in Figure 1:

- Windowing function/envelope or waveform shape
- Grain size / Duration
- Position
- Volume
- Pitch
- Grain Density / Overlaps.

Each of the above parameters of audio granular synthesis is mapped with a visual process (discussed below) that matches a perceived relationship. This will be explained in more detail in section 6.2.1
1.6.1 Visual Granular Synthesis

When working with digital audio synthesis, variables are usually defined in milliseconds. Thus the resolution of which each variable is capable is extremely high when considered in the context of what the human ear can perceive. Video, on the other hand, is measured and defined in frames per second (fps), with 60 fps regarded as a common benchmark for real-time application in games (Akenine-Moller & Haines, 2002). At 60 fps, the resolution is considerably less than working with audio, with each frame of video taking 16.66 ms. Therefore the smallest temporal resolution of an interlinked audiovisual grain is 16.66 ms.

My research defines a visual grain as being a selection of frames that fall within the length of the grain envelope. Thus, as noted earlier, a single frame is the smallest time duration unit into which my software can deconstruct a video file into, with either a single frame or collection of frames treated as a visual grain.

The visual grain is presented as a rise and fall in opacity according to the envelope shape. For example, (see Figure 2), a Gaussian window function shapes the volume of the audio grain to rise and fall over a bell-shaped curve whilst the opacity or contrast of the image follows the same shape.
This effect is most noticeable when only a single grain is playing at any one time. As grain polyphony increases, i.e. several audiovisual grains are playing at once, the windowing functions overlap, leading to a more continuous perception of auditory volume and visual contrast. The audiovisual grains contain a variety of differing perceptual effects depending on the shape of the windowing function employed. The curved rise and fall of a Gaussian window provides a smoother modulation effect than the sharp modulation effect of a Triangle shaped window.

My definition of an audiovisual grain takes account of the perceptual commonality between the opacity of an image and volume of a sound. Mapping the windowing envelope of an auditory grain to the opacity of a selection of video frames is just one of a seemingly endless combination of audiovisual mappings. For example, the user may choose to map the grain’s window shape to a visual post-processing variable, such as the blur or pixilation amount.

To deconstruct the visual image further, one could treat a single pixel or particle as a visual grain. Particle engines are often used to model visual effects such as clouds, smoke, fire and other natural phenomena (Reeves, 1983). Massive collections of these visual grains would then be representational of the grain cloud terminology used in audio granular synthesis but could actually be perceived as a visual cloud (Roads, 2001, Callear, 2009).

Whether defining a single pixel or frame of video as the single visual unit that can then be treated as a visual grain, the main parameters that are defined in audio granular synthesis (grain size, pitch, position, overlaps, speed, windowing function) should be conceptualised in an appropriate visual relationship.
1.7 Research Aims and Significance

A multitude of commercial software applications exist aimed exclusively at either music composers or visual artists. Multimedia programs such as Max/MSP (Cycling 74, California) enable audiovisual artists to work with both sound and image within one environment. However, many artists lack the programming skills needed to work with sound and image, and as a result, many choose to use two different software environments, one dedicated to audio and one to visuals, and connect them using a protocol such as MIDI or open sound control (OSC). A gap exists within the field for a unified audiovisual composition and performance environment that needs no programming experience from the end user. Such an environment requires an intuitive user interface that can help artists to create their own audiovisual relationships from which to construct works.

My research project addresses this gap through the development of a prototype capable of real-time AVGS. To achieve this goal, my research project sought to answer the following two research questions.

1. How can audio processing techniques such as granular synthesis be adapted and applied to influence new visual performance techniques?

2. How can computer software synergistically fuse audio and visuals to enable the real-time creation and performance of audiovisual art?

In exploring these questions, I examined four cross-disciplinary areas. The first involved the various strategies and techniques used for creating visual music. The second investigated creative live cinema as a performative practice consisting of interrelated and interactive sounds and images. The third focused on implementing the technique of granular synthesis as a tool to explore micro-compositional relationships between sound and image; while the fourth area explored the perceptual limitations of the human sensory input system.

In summary, the research aimed to investigate:

- Sensory, perceptual and technical commonalities between sound and image
- Strategies for adapting audio processing techniques to visual media
- Complementary relationships between sound and image at the micro and macro compositional levels
- Current multimedia programming environments and libraries that facilitate real-time audiovisual art
• Techniques and strategies that enable powerful parameter manipulations in real-time.

To achieve these aims, my project required the development of:

• An iterative performance event cycle enabling the testing and refinement of performance strategies and techniques

• A multitude of audiovisual prototypes that facilitate experimentation with sound and image relationships

• A collection of control strategies that provide a performer with powerful manipulations over audiovisual parameters during a real-time performance

• A modular framework that assists the user in creating custom relationships between audio and visual processing techniques that facilitates emergence of a wide range of potential audiovisual aesthetics

• A visual granular synthesis processing technique

• New software techniques that facilitate real-time manipulation of audiovisual material

• A stable software performance instrument.

1.8 Scope and Limitations

The cross-disciplinary nature of my research required an investigation of granular synthesis, visual music, live cinema, human sensory perception, and audiovisual art. Each of these fields contains a substantial body of knowledge, some of which is beyond the scope of my research project.

I chose to limit the investigation into adapting audio processing techniques for visual processing to granular synthesis, and to the following common DSP audio effects: downsampling, filters, delays, flange and reverb. At 60 fps, a single frame of video with a duration of 16.66 ms is the smallest visual unit or grain that Kortex uses for visual granular synthesis. Continuing to deconstruct the image down to the individual pixels was outside of the scope of my project and is an area for future research.

This research focused primarily on the creation of a real-time audiovisual performance instrument. As such, I consciously avoided topics such as perception theory to describe the relationship between an audience and myself as a performer.
1.9 Research Methodology

This research project commenced in January 2011, and was an extension of my previous honours research into interactive audiovisual performance software. My primary research method was practice-based methodology, which is detailed in Chapter 3.

Practice-based research considers the creative artefact resulting from the research to be the contribution to knowledge. My practice-based research generated a software instrument prototype that will be used in a public performance in December 2014.

An iterative process of action and critical reflection guided prototype development and evaluation throughout this research. In this way, my investigation produced a detailed understanding of each prototype’s potential.

1.10 Exegesis Outline

The structure of the remainder of the exegesis is as follows.

- Chapter 2 provides background and an overview of relevant research in the areas of visual music, live cinema, granular synthesis and illusionary art, and presents influential artists, their works and software within my community of practice.

- Chapter 3 describes the research methodology and defines the methods and evaluation criteria.

- Chapter 4 describes my performance and artistic intentions for creating real-time synchronised audiovisual art, the principles that define my aesthetic practice and the relationship between the instrument, audience and myself as the performer.

- Chapter 5 describes the performance iteration cycle used to test and evaluate subsequent iterations of the Kortex prototype.

- Chapter 6 describes the software instrument prototype developed as part of this research project, and contains a presentation of selected results and a discussion.

- Chapter 7 presents the findings that emerged from designing and using Kortex as a live audiovisual instrument.

- Chapter 8 summarises the outcomes of this research project and suggests directions for future research.
# Theoretical, Technological and Artistic Influences

## 2.1 Overview
This chapter documents the literature review and referencing of technological and theoretical influences pertinent to my research project. The review is presented in five sections in order to evaluate and situate the cross-disciplinary nature of my research project. Firstly, I outline the development of visual music from pre-digital through to the current age, and discuss perceptual and compositional techniques for integrating sound and image. Secondly, I discuss the emerging live cinema practice with its relationship to DJ/VJ performance and traditional cinema. Thirdly, I outline granular synthesis as having evolved from quantum sound theories to a real-time digital synthesis technique and discuss the ramifications for composers working with micro-sound. Fourthly, I discuss illusionary art through presenting various 20th-century art works and neurological research in order to understand the limitations of the human sensory input system and their potential for creative applications. This chapter concludes with a description of my relationship to a community of practice by presenting a selection of artists and their creative works in the field of audiovisual art.

## 2.2 Visual Music
This section contains an historical overview of ideologies related to the integration of visual and audio elements in creative arts practices. The overview provides a foundational context for my research by situating my project within the field of real-time audiovisual art.

### 2.2.1 Historical Background
Roger Fry⁴ coined the term ‘visual music’ in 1912 when discussing the work of Kandinsky (1866 - 1944), using it to describe a pictorial form of music (Ward, 2006). Kandinsky, who was believed to have the cross-sensory condition called synesthesia, essentially translated sounds into visual forms.

---

⁴ An English artist and art critic best known as the champion of the movement he termed *Post-Impressionism*. *Encyclopedia Britannica* online (2014)
However, Kandinsky, although pivotal in a review of material concerning visual music, was by no means the first artist to explore ideas about the visualisation of music. Moritz (1986) noted that “Since ancient times artists have longed to create with moving lights a music for the eye comparable to the effects of sound for the ear” (p. 1). This longing took on a more substantial form from the 18th century, when artists began to link artistic imaginings with emerging technologies.

In 1720, the artist Louis Bertrand Castel (1688 - 1757) was preoccupied with imagining devices that would visibly render musical performance to “enable the eyes to partake of all the pleasures which music gives to the ears” (Conrad 1999, p. 393). Referring to Isaac Newton’s optical lectures of 1672, Alves (2012) and Fuxjager (2012) credit Newton as the first person to quantify a correspondence between colour and pitch when he described wrapping the musical octave around the colour wheel. The direct mapping of pitch to colour described by Newton inspired many inventors to develop the earliest instruments designed to visualise music, known as colour organs.

### 2.2.2 Early Visual Music Technology

Composers using the organs mapped colours to sound, according to their pitch, amplitude and timbre (see Figure 3 below for a brief chronological overview of attempts to associate colour with pitch). The organs were played using a conventional piano keyboard. When the performer struck a key, it would lift a specific curtain to reveal candle light shining through a coloured stained-glass pane. Various theories of colour through the eighteenth and nineteenth century influenced a profusion of different colour organs (McDonnell, 2007).
Theoretical, Technological and Artistic Influences

Figure 3: Schemas for matching pitch to colour scales, 1704–2004 (Collopy, 2009)

Following the colour organ, 20th-century artists such as Len Lye, Norman McLaren and Stan Brakhage experimented with painting animations directly onto blank rolls of film stock (Jordan & McLaren, 1953). This arduous process resulted in the production of tightly synchronised audiovisual works long before the age of digital technology. These artists managed to portray their visions by bypassing the conventional apparatus of the camera altogether and working with technology usually restricted to film technicians (Le Grice, 2001).

Other pioneers such as Paul Klee, and Oskar Fischinger experimented with abstract forms of visualising music that, even in relation to today’s audiovisual works, seem ahead of their time. Fischinger was fascinated by contemporary research relating to visual and auditory phenomena and was inspired to carry out his own experiments. In 1931, he drew shapes directly onto the film soundtrack (as seen in Figure 4), allowing the synthetic waveforms to be transformed into sound by a projector’s photocell (Kotz, 2009).
These pioneers demonstrated the effects that sound and image gave to each other when combined through various technologies. Pellegrino (1983) described the complex interactions between sound and light as the “principle of synergy” (p. 208), with the resulting form being greater than the sum of its parts.

DeWitt (1987) claimed the most impressive body of work during visual music’s early development came from the Whitney brothers, John and James, two influential animators working in abstract and experimental cinema. They sought to build devices that produced a synthesis of sound and image for the compositional process and, influenced by Fischinger’s work, aspired to bridge the gap they saw between the actual realisation of combining sound and animation (McDonnell, 2007). Alves (2012) reported that Whitney recognised that adding music to abstract animation involved multiple technical and aesthetic complexities, especially in maintaining equality between the two. During an interview in 1968, John Whitney observed:

Of course, even these actions on the screen might work better if there were music of some sort
related to them. I think of sound as a kind of partner to this visual experience, but I do not want the graphics to play the role of the lesser, subservient partner. *(Experiments in Motion Graphics: The John Whitney Collection 2009)*

The brothers got the chance to explore the relationship between consonance and dissonance in visual music during an artist in residence position at the International Business Machines Corporation (IBM) in 1966 (Alves, 2012). Inspired by the Baroque composer and harmony theorist Jean-Phillippe Rameau, the brothers used differential dynamics software programmed by IBM engineer Jack Citron to create their film ‘Homage to Rameau’. John Whitney Snr was invited to undertake another residency at IBM during the late 1980s, where he sought to create a real-time visual music piece named *Spirals*, seen below in Figure 5.

![Figure 5: Whitney's setup at IBM for filming graphics from a computer screen (Moritz, 2012)](image)

The Whitney brothers went on to develop visual equivalents to Schoenberg’s 12-tone system (McDonnell, 2007). That is, just as Schoenberg reduced music composition to a serial row before intuitively re-assembling these pieces into a more complex form, the brothers reduced the image “down to its most fundamental state – essentially a point of light, which could be ordered like a tone row” (Brougher et al. 2005, p. 125).

### 2.2.3 Identifying the Complementary

Arguably, the Whitney brothers’ greatest legacy to visual music was in describing the ‘complementary’ (Whitney, 1994). The complementary refers to a perceived associated bond that intertwines tone and colour, allowing artists to design temporal unions between the relationship of sound and image (McDonnell, 2007). In establishing a complementary, Whitney resisted automatically mapping characteristics from one medium to the other, insisting that composers do this mapping by ‘matching’ similarities between the audio and visual domains (Alves, 2012). My research follows the Whitney brothers’ notion of the complementary when, as we will see later on, I refer to ‘perceived
relationships’. Abbado (1988) suggested a method for establishing links between four complementary perceptual categories: dynamics, timbre/shape-surface attributes, spatial localisation and intensity. For example, Abbado stated a potential correspondence exists between the perceived intensity of a sound’s loudness and an image’s brightness. The complementary relationship of brightness to loudness results from identifying the perceived effect of intensity. Although this correspondence may not be universally true for all people, it provides the artist with a method for establishing relationships through subjective identification of audiovisual characteristics.

The musical characteristics of rhythm, harmony, form, time, duration and orchestration contain sonic qualities that the composer arranges into music. Similarly, the elements and principles of visual art, including shape, colour, form and contrast, allow an artist to communicate a variety of visual qualities in a related way. Music and visual art share many perceptual similarities; for example, both artists and composers use the quality of dissonance in a work to offset a resolution. The composer and artist use this technique to create a feeling of suspense and expectation (Kamien, 2010). The techniques visual artists and composers use to achieve this can also be used together to enhance each other, thus heightening the perceptual quality of dissonance/resolution in the audience. The potential correspondences across both music and visual arts present opportunities in visual music to form connections that share a similar emotional basis (Alves, 2012).

The subjective nature of perception allows each visual-music artist to form differing associations between characteristics. There are many strategies and techniques for mapping sound and image. These include cultural, psychologically inherent and novel associations for helping define meaning within visual music (Jones, 2008).

2.2.4 Synchresis

Synchresis is a term defined by Michel Chion as, “the spontaneous and irresistible weld produced between a particular sound phenomenon and visual phenomenon when they occur at exactly the same time” (Chion 1994, p. 63). The concept of synchresis contains obvious parallels to Whitney’s definition of the complementary. The main difference occurs from the viewpoint in which Chion and Whitney described the perceived relationship of sound and vision. Chion considered the fused audiovisual construct from the standpoint of film, while Whitney was coming from the viewpoint of non-representational abstract art.

Moody et al. (2007) suggested that a concrete example of synchresis in film occurs when a violent punch is accompanied by a loud audible thump. The exaggerated sound is perceived as connected by the audience even though the image and sound have nothing to do with each other. When sound is
used to enhance the image, the audience’s perception of the image gains what Chion (1994) terms “added value” (p. 5). That is, the effect of the two media experienced together is greater than if they were to be experienced in isolation from each other.

### 2.2.5 Parameter Mapping

Composers, theorists and practitioners of visual music have long been preoccupied with the fundamental issue of audiovisual mappings (Garro, 2012). That is, rather than merely making arbitrary associations between elemental characteristics of the senses, Alves (2012) suggested there is an opportunity for visual music artists to define relationships that share a common emotional basis. For example, dark colours and minor chords trigger a different emotional response than major chords and bright colour palettes.

Callear argued that the transposition process of values between the audio and visual domain provides the essential “compositional framework for the creation of audiovisual art” Callear (2013, p. 1). Understanding the correspondences that exist between the two media is thus a vital prerequisite for the audiovisual artist. Similarly, Momeni and Henry (2006) affirmed that the goal of the audiovisual artist lies in generating sound-image correspondences that contain perceivable links.

McDonnell (2007) noted that within the practice of visual music, visual elements are presented with strategies similar to those employed in the composition or performance of non-visual music. New visual patterns and relationships become accessible when an artist applies musical thinking to visual elements. For example, the visual music artist may employ musical compositional techniques such as consonance and dissonance, repetition and motives, orchestration, and harmony (McDonnell, 2007). By exploring the correspondences that exist between musical and visual characteristics, the visual music artist can compose animations akin to musical composition (Alves, 2012).

To achieve connections between images and music, Fuxjager (2012) suggested an artist translates music into moving images through applying a strategy of transcoding certain characteristics of music into certain visual characteristics. The artist in this sense would decide which musical parameters (volume, pitch, timbre) should manipulate which visual parameters (colour, position, movement, etc.) (Frank and Lia, 2010).

### 2.2.6 Modern Computation

During the past 100 years, artists have used different means to generate visuals from sound and to sonify sounds from images using digital technology. The oldest and most common way of visualising sound involves an oscilloscope – a cathode ray tube used to display the frequency and amplitude of an incoming audio signal. Karl Braun invented the cathode ray tube in 1897 and until the invention of
LCD screens, a cathode ray tube was inside every television (Atherton, 1990). This technology is still used in recording studios today to provide the engineer with a visual representation of the sounds’ frequency content.

One of the earlier video music units, the Atari Video Music Unit (1976) (Figure 6), facilitated the process of visualising sound and music more than ever before. With the advent of the personal computer, it became the instrument of choice for forging music and image connections (McDonnell, 2007). The computer’s ability to represent all media as bits and bytes enabled visual musicians to translate and link one medium to the other seemingly effortlessly (Daniels & Naumann, 2011).

Figure 6: The Atari Video Music unit (Cruiser, 2012)

Media players on computers offer real-time visualisations of music that are generated and synchronised from changes in the music’s loudness and frequency information. Popular media players such as Winamp (2014) used a plugin called MilkDrop (2003), seen below in Figure 7, a 3D music visualisation add-on that displays various 3D graphics that ‘dance’ to the users’ music. Fuxjager (2012) noted that the absence of user-defined parameter mappings excludes the user from the creative process because they are confined to the set parameter mappings of the programmer. Today, almost all media players that ship with Mac, Windows and Linux come with audio visualisers built in, allowing users to witness both 2D and 3D visual representation of their favourite music.
2.2.7 Transcoding

Sonification is the process of reproducing data (such as images or text) as sound events (Schneider, 2011). The process can be creatively used as an alternative method for the generation of new sounds. Sonification can also be used to encode visual data within sound to aid people with visual impairments (O’Neill & Ng, 2008).

A recent example of sonification being used in a creative application can be seen in the interactive software VOSIS (McGee, 2013). The application, seen below in Figure 8, allows the user to specify the position of multiple small rectangular areas within an image. Greyscale pixel data contained within each rectangle is then raster scanned to produce complex auditory wavetables. DSP effects are achieved through the process of post-processing the image. For example, a sonic bit reduction effect can be achieved through applying a masking image filter. Approaching sound from the visual medium...
essentially dissolves traditional approaches for sound generation and opens a myriad of creative potential.

Figure 8: VOSIS interface showing three visual regions that have each been visually processed in different ways to achieve various sonic results (McGee, 2013)

The commercial software package Metasynth (2013) is capable of converting images into sounds by converting the pixels’ location and brightness to frequency and amplitude. The frequency and amplitude data generated from analysing the image is then used to synthesise a corresponding sound. The process is also reversible, allowing an image to be generated from a snapshot of a sound’s frequencies and amplitudes by performing a fast Fourier transform on the sound.

Audio and images are recreated from a string of numbers in a computer. In this way it is possible to sonify and visualise mathematical algorithms that, through altering the algorithm’s variables, enable the image to become animated and produce musical structures from the audio. The audiovisual band Tvestroy (2009) make exclusive use of this technical approach to audiovisual synthesis in their work, as seen below in Figure 9. By synthesising images and audio from the same source, the media appear to have a closer bond or relationship, i.e. the image is the sound and the sound is the image.
2.2.8 Perception

Understanding how humans perceive sound and image provides another opportunity for identifying commonalities between the two. Haverkamp (2009) stated that the physical phenomenon of sound is perceived by humans in three main dimensions: temporal, spectral and spatial. Both the temporal and spectral qualities exist as specific instances that are shaped and evolved within time, while the spatial qualities are perceived in relation to the location and distance of the sound source. Fuxjager (2012) argued that, as with any time-based art, the central aesthetic feature of the artwork is in the structuring of the reception of time. The temporal structure of the resulting artwork contains elements that evolve over time (McDonnell, 2007). The moving images are structured in the same way that the composer uses time to evolve musical elements. The interaction and interplay between visual and musical elements establishes a temporal architecture similar to that of absolute music5 (Evens, 2005).

---

5 Chua (2004) describes absolute music as abstract, non-representational, instrumental music that, according to composers such as Richard Wagner, is void of concept of purpose.
2.2.9 Compositional Strategies

The composer of visual music needs to consider compositional strategies that complement a united audiovisual structure. Furthermore, different compositional techniques lend themselves to portraying a variety of emotional and psychological states. DeWitt (1987) proposed that rhythm can be used to induce a feeling of excitement through fast rapid motion of objects or colours. McDonnell (2007) noted that when visual elements are informed by the tempo, dynamics and rhythm of the music, the two media seem to fuse into a unity and togetherness. Cutting or transitioning to a different scene during a performance is another powerful compositional technique, shifting the audiences’ relationship with the audiovisual construct (Barreiro, 2010).

The aesthetics and compositional techniques of both music and cinema provide fertile grounds for experimentation within visual music practice. Garro (2012) asserted the visual music composer is thus:

compositing the ramifications of two artistic domains to arrive at a harmonious synthesis of the two, a locus where variables explain one another, not solving the equation, but making it, if nothing else, somewhat more transparent. (Garro, 2012, p. 106)

Using a comparative approach to visual composition lends itself to abstract non-narrative art similar to that of music. The non-narrative nature of visual music results from imagery and sounds that are often generated and manipulated using a computer. Because of this, Piche (2003) considered visual music “an experience of pure time, without the burden of a story. Where cinema is audiovisual novel, visual music is audiovisual poetry.” As a result of Piche’s statement, Garro (2012) proposed that the audience may find the resulting sound and imagery does not easily suggest a familiar storyline with which the audience may connect.

2.2.10 Subjectiveness

Visual musicians, whether performing with computers, colour organs or with any other specialised instrument, aim to create non-representational visuals that are as abstract as music (Fuxjager, 2012). Smith (1982) maintained that it is this ambiguity that helps give the artwork meaning. The mind of the viewer is assigned to contemplate the work’s abstractness, filling in the blanks to reveal personal subjective meaning. Mekela (2008) suggested the role of the audience moves from a passive one (as in traditional cinema) to an active participant in decoding meaning from the work. Garro (2012) asserted that during this bi-directional process between the audience and performer, audiences should not engage in intellectual validation of the medium but rather enjoy and make sense of the artistic messages being transmitted and received.
2.3 Live Cinema

Live cinema is an arts practice emerging from an intersection between experimental film and video, computer music, and Video Jockey (VJ) performance (Jones, 2008). While artists and scholars use the term live cinema to describe a wide range of live audiovisual performances, Fodel (2010) noted live cinema artist Costabile’s frustration with the term. She interpreted the term merely as “just humans operating machines in real-time” (Fodel, 2010, p. 9). Costabile’s statement probably comes from scholars’ attempts to academicise the practice. I base my project on Makela’s (2008) definition of the live cinema practice as “real-time simultaneous creation of sound and image in a performance event” (p. 1).

Although live cinema is relatively new, it shares an idea expressed as early as the 18th century with the invention of the magic lantern, namely live performance with interrelated sounds and images (Jones, 2008). The practice also shares similarities with the early silent cinema era, when non-verbal dialogues were utilised to communicate ideas and were traditionally accompanied by an orchestra playing music (Makela, 2008).

2.3.1 Live Cinema in Relation to DJs & VJs

Contemporary similarities exist between live cinema and the VJ scene. VJs may be defined as ‘visual artists who use digital media to express themselves to an audience during a live audiovisual performance’ (Hook, Green, McCarthy, Taylor, Wright, Olivier, 2011, p. 1). Both practices share an eagerness to explore new visual technologies in order to portray unique artistic visions (Makela, 2008). Fodel (2010) and D’Escrivan (2012) contend that within a club environment, an audience generally sees VJing as accompaniment to music or merely decorative rhythmic visuals. In this sense, the performance context differs from the theatre environment with a seated audience attentively watching a performance.

VJs resemble DJs in that they generally mix pre-existing material created by another artist. The online market for purchasing clips implies that many VJs use the same material (Makela, 2008). Typically, the VJ is required to mix appropriate visuals alongside DJs they may not have heard before. This excludes VJs from the compositional process, as they usually work ‘after the fact’ of music creation (D’Escrivan, 2012). In contrast, the artistic essence of live cinema seeks to develop audiovisual objects that D’Escrivan (2012) described as unified compositional/performance units. Live cinema artists construct these indivisible audiovisual objects so as to develop their own unique creative voices.
2.3.2 ‘Liveness’ of Live Cinema
The notion of liveness within the live cinema practice proposes another dilemma for both the artist and audience. How can the artist acknowledge an audience whilst engaged with a computer? The computer generally requires the full attention of the artist and often the audience cannot see their performative actions. Whilst the physical actions of turning a knob on an interface may be uninteresting to an audience, the artist may feel the need to prove liveness to an audience by becoming the featured content of the show, redefining their role away from an integrated part of the performance (Makela, 2008).

Fodel (2010) maintained each live audiovisual performance lies on a continuum between two extremes. On one hand, a performer is clearly acting out embodied motion that can be directly associated with the audiovisual sequence, with active participation from the audience. On the other hand, an audience watches a presentation of sound and visuals in a passive disembodied state, without any evident actions from the performer. Jones (2008) and Fodel (2010) asserted that within this hierarchy of liveness, the context of a performance can help dictate the audiences’ expectations and the strategies of the artist.

It should be noted that an audiovisual performance involves audio and visual perception on the part of an audience. The audience can choose to hear the performance separately from the visuals or see the performance separately from the sound. In hearing and seeing the performance simultaneously the work is experienced more wholly. In this way the visual strengthens the audio and the audio strengthens the visual.

2.3.3 Meaning and the Performer/Audience Relationship
Performing the results of my research will enable the transmission and communication of ideas for the public to interpret and from which it can derive meaning. Stone (1982) described the audience’s role as auditing interpretation when witnessing a performance based on each individual’s past, personal and cultural experience. Through this process of interpretation, meaning may be extracted and shared with other event participants. The performance allows a cultural exchange to occur between the performer and an audience. This exchange can involve emotional, perceptual and social engagement that creates the potential for the performer and audience to make meaning from a performance and bring about personal, social and cultural change.

2.3.4 Cinema vs Live Cinema
There is an inherent contradiction in the idea of live cinema – namely that the real-time situated nature of live performance seems incompatible with traditional cinema. D’Escrivan (2012) argued that
because film is edited and the final artifact is fixed, cinema is traditionally a ‘plastic art’. Makela (2008) contended that the absence of actors and dialogue in live cinema claims freedom from the linear story telling of traditional cinema. The context and goals of cinema and live cinema in the cinematographic space are fundamentally different.

While at first the term ‘live cinema’ may seem contradictory, in my research I viewed ‘live’ as a situated performance in a particular place and time to an audience invited to witness the performer’s actions. In contrast, ‘cinema’ is an engaging, immersive experience that asks the viewer to look past the screen into another place.

Live cinema is still a fairly new genre and is often classified as a form of avant-garde cinema. This has led practitioners of this genre of cinema to perceive its cultural role to be continually marking out the terrain of new underground movements (Arthur, 2005). Live cinema artists are therefore afforded a degree of freedom when developing new ways of communicating subjective visions.

### 2.3.5 Build your own Tools

Live cinema artists face practical and conceptual difficulties due to the many layers of technologies that are required for a performance. These include the performance interface and application software through to the computer hardware, projector and speakers. Jones (2008) stated that the individual practitioner must carefully consider each layer to avoid conveying unanticipated or distorted meanings. Though the risks of technological failure are increased due to the amount of layers, mastery over the various technological layers establishes a gateway for the artist to express a variety of radical personal visions.

For artists to manifest their visions, as in any art form, an understanding of how to mediate meaning from their tools (software, custom controllers and interactive interfaces) is required. Consequently, many artists working in live cinema spend just as much time building custom tools as they do making works (Jones, 2008). For example, Coldcut and Hexstatic created the custom software program, VJamm, to produce the now famous synced audiovisual work, *Timber* (Coldcut and Hexstatic, 1998). By writing custom software, artists are able to develop personal technological and creative languages that facilitate the communication of internal audiovisual experiences during the creation and performance of a work.

Building custom tools is not limited to the live cinema practice but is also common practice for visual musicians. McDonnell (2007) stated that visual music exists either as a fixed media video or film projection, or as non-fixed media within an improvisational performance event. She further stated that the latter involves building interfaces and systems that require considerable technical skill to realise
the delivery of visual music in a real-time setting. The resulting interface enables direct manipulation of the sound and image and may be performed in the same manner as traditional musical instruments. Aesthetic and technical connections between the sound and image parameters require a degree of artistic sensibility when designing a successful performance interface.

2.4 Granular Synthesis

The following discussion examines the main concepts and developments of granular synthesis and refers to key technological and compositional breakthroughs. This is followed by an examination of the micro and macro musical timescales, and the various compositional and control strategies used with granular synthesis.

2.4.1 Quantum Sound

Quantum sound can be defined as microacoustic phenomena lasting less than one-tenth a second (Roads, 2001). From a psychoacoustic point of view, the indivisible unit of a quantum sound is the building block on which all macroacoustic events are based (Truax, 1990). Therefore, the quantum level of microsound inextricably links the various timescales of music, prompting Truax to call it “the final frontier of acoustic and musical research” (Truax, 2003). Granular synthesis provides a means to manipulate and construct microsonic textures that form the basis from which all acoustic phenomena emerge (Roads, 2001).

The notion that all matter can be broken down and reduced into indivisible units can be traced back to the early atomist Greek philosophers. These theorists believed atoms to be the building blocks of the universe, including all matter, energy and sound (Opie, 2003). In this view, apparently continuous phenomena consisted of clusters of atoms or particles with unique properties that could be measured and studied. Much later, scientists hotly debated the ‘wave’ and ‘particle’ views of acoustics and optics and gave rise to the formation of modern science (Roads, 2001).

In 1905, Albert Einstein discovered Planck’s constant\(^6\) could be used to determine the emission of light at the quantum level (Einstein, 1965). Through analogy with the concept of photons (light particles), Einstein postulated that sound consisted of phonons, and eventually proved that light and sound are made up of discrete particles (Roads, 2001). This discovery led Einstein to define the

---

\(^6\) Planck’s constant is a fundamental physical constant characteristic of the mathematical formulations of quantum mechanics, which describes the behavior of particles and waves on the atomic scale, including the particle aspect of light (Encyclopedia Britannica, 2014).
smallest indivisible unit required for auditory discernment as the quantum of sound (Roads et al. 1997). In other words, from a psychoacoustic point of view, a “quantum of sound is the smallest duration of sound required to activate the auditory system” (Truax, 1990, p. 129).

2.4.2 Pioneers of Quantum Sound

The contemporary concept of quantum sound is usually credited to the physicist Dennis Gabor (1900-1979) (Carlson, 2012, Truax, 1990). In the 1940s, Gabor proposed that any sound could be decomposed into discrete acoustic quanta or grains with independent time, amplitude and frequencies.

Gabor was fascinated with the ‘timeless’ Fourier theorem\(^7\) used in mathematics to describe the quantum principle of sound (Truax, 2003). In 1946 and 1947 he published papers outlining a windowing system used to analyse and reproduce sound through articulating the quantum as a rectangular area which encompassed both the time and frequency domain (Gabor, 1946, Risset, 1991, Truax, 2003). Gabor’s system (seen below in Figure 10) involved the application of a bell-shaped Gaussian curve to the amplitude envelope of each grain, thereby combining the two previously separated dimensions of time and frequency (Roads, 2001).

\(^7\) The analysis of a complex periodic wave into its spectral components was theoretically established early in the 19th century by Jean-Baptiste-Joseph Fourier of France and is now commonly referred to as the Fourier theorem (Encyclopedia Britannica, 2014).
Gabor (1946) constructed several kinematic frequency converter machines to test his analytical conclusions concerning data reduction and time/pitch shifting. The design seen in Figure 11 utilised light initially shining through slits of an optical film projector, and then through the film and onto a photocell. As the film moved at a constant velocity past the slit, the photocell recorded light that was later converted into sound (Opie, 2003). This early method enabled the pitch to be altered without affecting time (pitch shifting) and vice versa (time stretching). Although Gabor (1946) was able to demonstrate the success of his theory, more research was needed before it could be considered viable for musical applications.
Iannis Xenakis was the first musician to utilise Gabor’s research and in 1960 coined the term ‘grains of sound’ (Roads, 2001). Xenakis was keenly interested in scientific and mathematical discoveries, and translated the knowledge into formalised compositional theories. Xenakis (1971, p. 43) observed “the attack, body and decline of a complex sound contained thousands of pure elementary sound particles”, each existing in three-dimensional space bound by frequency, amplitude and time.

In 1958, Xenakis produced the first recognised demonstration of musical granulation in the composition entitled Concret PH. The piece employed a technique of splicing tiny segments of analogue tape lasting one second each. The duration of segments exceeded the strict terms outlined by Gabor, who stated that a single grain’s duration should be 20–50 ms. Opie (2003) and Roads (2001) argued that because of this, the technique of arranging small pieces of sound was more analogous to sound sculpture than that of true granular synthesis. It would be several decades, until the advent of the computer, before Gabor’s theory of granular synthesis could be practically exploited in a music context (Opie, 2003).

2.4.3 Technology

Inspired by Xenakis’ ‘grains of sound’ theory, composer Curtis Roads spent much of his life
researching and writing about granular synthesis. He also wrote some of the first granular synthesis computer programs (Roads, 2001, Carlson, 2012).

Roads enrolled at the University of California, San Diego, where he had access to an advanced mainframe, the dual-processor Burroughs B6700. The Burroughs machine (Figure 12) was exceptionally powerful for its time, albeit extremely laborious and time-consuming to operate.

Figure 12: Burroughs B6700 mainframe computer (B6700 Display Panel n.d.)
Installed on the machine was the Music V program written in 1968 by Max Mathews (Mathews et al. 1969). The software was programmed onto punched paper cards using the programming language Algol. Due to the mainframe’s storage limitations, Roads was only able to make a one-minute study of monaural sound that took several days to process. The mainframe was unable to produce audio directly; instead, the mainframe processed data from the punch cards and transferred it onto a large digital tape. The digital tape then needed to be transferred to a disk cartridge using equipment at the Scripps Institute of Oceanography. Finally, once the data had been transferred, the cartridge was converted to audio on the DEC PDP-11/20 machine housed on campus at the Centre for Music Experiment (Roads, 2001, Opie, 2003).

2.4.4 Composition

In 1975 Roads used the arduous and lengthy process described above to produce the first-ever computer-generated granular synthesis piece, entitled *Klang-1*, which consisted of 766 individual grains of sound with each grain’s frequency, amplitude and duration programmed onto a separate punch card. Roads’ compositional process was to approach each grain as though it were an individual note. According to Roads (2001), the grains were all generated from sine waves using a simple Gaussian envelope with frequencies ranging from 16 Hz to 9937 Hz. This primitive approach to realising non-real-time computer-generated granular synthesis continued until the advent of the personal computer in the mid-1980s.

In 1986, Canadian composer Barry Truax (the pioneer of real-time granular synthesis) wrote a real-time application on the DMX-1000 signal processor to produce the piece *Riverrun* – the first-ever recording to be realised entirely with real-time granular synthesis (Truax, 1988, Carlson, 2012). Truax’s real-time application was capable of generating an impressive 2,375 grains per second, and one year later was able to granulate a brief sampled sound. By 1990 he had developed a technique to perform real-time granulation on a live incoming sound source from instrumentalists. This innovative technique led Truax to create several pioneering compositions (Roads, 2001).

2.4.5 Software

In the ensuing decade sound engineers developed multiple software environments for building real-time granular synthesis applications able to run as virtual studio technology (VST) plugins, commercial standalone software instruments and most recently as fully functional apps on mobile devices (Carlson, 2012). The most popular and widely used frameworks include Max/MSP (Cycling 74, California), AudioMulch (Sonic Fritter, Melbourne), Csound (open source), SuperCollider (open source), Kyma (Symbolic Sound, Champaign) and Reaktor (Native Instruments, Berlin) (Figure 13).
Each of the six frameworks shown in Figure 13 enables a unique approach to the creation of granular synthesis software; offering either text-based coding workflows or graphical modular patching environments, with Kyma offering a combination of the two. When choosing a framework for granular synthesis, it is important to consider the end potential of the application. For example, Reaktor facilitates the creation of granular synthesizers that can run either as standalone applications or in popular digital audio workstations (DAW), but users must have Reaktor installed on their computers. Csound would be the ideal choice of framework if the objective were to build a VST/audio unit (AU) plugin that can run without the need for secondary software. Table 1 highlights some of these functional differences between the previously mentioned frameworks.

<table>
<thead>
<tr>
<th>Granular Synthesis Capabilities</th>
<th>AudioMulch</th>
<th>KYMA</th>
<th>MAX/MSP</th>
<th>SuperCollider</th>
<th>Reaktor</th>
<th>Csound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requires External Hardware</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Build Standalone Apps</td>
<td>✗</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Build VST/AU Plugins</td>
<td>✗</td>
<td>✗</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Price</td>
<td>Free</td>
<td>$2970+</td>
<td>$399+</td>
<td>Free</td>
<td>$399</td>
<td>Free</td>
</tr>
</tbody>
</table>

Table 1: Comparison of audio synthesis programming environments

8 Subsequent source-less images, diagrams and tables were produced by myself.
From my own experience of working extensively with these frameworks, I have found that each imprints a unique sonic characteristic onto the final granular texture – so much so I can easily identify if a granular sound was produced using (for example) Reaktor or Kyma. Being able to identify the sonic characteristics of a framework helps me make aesthetic choices about whether or not I use existing frameworks or create my own. This granular texture is a result of the programming language used to build the framework, the internal read/write granular scheduling into the audio buffer and, in the case of Kyma, whether or not external hardware is required to execute the program. Whichever framework composers choose to generate granular sound depends on their familiarity with either text-based or graphical programming, and their personal preference towards the intrinsic sonic characteristics of the framework.

2.4.6 Timescales

The human perception of a sound differs as sound passes through one time scale to another. Because timescales are interlinked, to operate on the microsound level is to affect all other levels. Roads (2001) outlined the nine various timescales of music as:

1. **Infinite**
   The ideal time span of mathematical durations such as the infinite sine wave of classical Fourier analysis.

2. **Supra**
   Extending beyond the duration of an individual composition into days, months, years and centuries.

3. **Macro**
   The overall form of an entire musical composition.

4. **Meso**
   Division of form into hierarchies of phrase structures.

5. **Sound Object**
   A single note or evolving sound lasting between a fraction of a second to several seconds.

6. **Micro**
   Sound particles on the threshold of auditory perception measured in thousandths of a second.

7. **Sample**
   Atomic level of digital audio: individual binary samples measured in millionths of a second.

8. **Subsample**
   Fluctuations too brief to be recorded or perceived.

9. **Infinitesimal**
   The ideal time span of mathematical durations such as the infinitely small brief delta functions.
Looking at the above timescales, it seems logical to assume that the Infinite, Supra, Sample, Subsample and Infinitesimal regions exist on a scale too big or too small for most compositional applications. In contrast, the micro level provides the composer access to transient phenomena that directly affect the perception of sound as it crosses the remaining higher timescales. In everyday life, we are surrounded by collections of microsonic events: leaves crunching under our feet, the spray of water crashing against rocks, rain droplets on a tin roof, the crackling of burning embers and the humming sound from a swarm of bees are perceived by humans on the timescale of the sound object, yet result from thousands of independent microsonic events. When seen from this vantage point, the seemingly trivial grain provides the composer with the building block to create any sound imaginable. The once inaccessible micro time domain now stands at the forefront of compositional interest (Roads, 2001).

For a composer, interest in composing with microsound stems from its ability to blur the levels of musical structure. Two interesting compositional strategies exist for working with microsound: the bottom-up and multiscale approach. In the bottom-up approach, global morphological structures emerge from interactions between microsonic events specified by the composer; however, the multiscale approach allows the composer to intervene on every time scale. Arguably, this approach provides composers with a wider zone for creativity as they can mould, shape, insert and rearrange sound freely across all timescale boundaries (Roads, 2001).

2.4.7 Control Strategies

The enormous amount of data involved when specifying thousands of microsonic events per second demands powerful control strategies in order to make granular synthesis a viable synthesis technique for a composer (Truax, 1990). One such technique involves the use of intelligent algorithms that are able to translate high-level compositional directives into thousands of lower-level grain specifications (Roads, 2001). This technique helps to liberate the composer from the arduous process of specifying individual grains required in the early days of granular synthesis. Publishing high-level functions such as envelopes and presets provides the composer with detailed and elaborate transformations in which to shape the evolution of individual sound objects.

2.5 Illusionary Art

This section presents research literature about the flicker effect (a light stimulation technique used to induce various emotional states) by researchers W. Grey Walter, Michael Hutchinson and John Smythies. Their research is presented in order to understand how the illusionary afterimages caused by AVGS can be harnessed for artistic applications. For example, Walter (1965) reported from his studies...
into flickering light that “strange patterns, new and significant, emerged” (p. 92). Smythies (1960), as a result of undertaking the largest study on stroboscopic effects, concluded that the participants’ fascination with the illusionary patterns evoked by stroboscopic light was the result of the brain failing to adequately manage the sensory input.

2.5.1 Human Sensory Perception

We face a multitude of sensory experiences on a daily basis. My research aimed to examine the audiences’ subjective perceptual response to specific visual and auditory stimuli. To achieve this, human beings’ perceptual limitations and thresholds for sensory information needs to be understood.

One such perceptual limitation of interest is the phenomenon of the retinal afterimage. Retinal afterimages are caused when the eye’s photoreceptors, primarily those known as cone cells, adapt to overstimulation and lose sensitivity (Shimojo et al. 2001, p. 1677). The bright glow that fades in one’s vision immediately after a photoflash is an example of an afterimage that most people have experienced. This visual effect can be experienced through visual perceptual phenomena such as object substitution and feature inheritance (Sigman et al. 2008), visual motion perception (Ullman, 1979, Kohn & Movshon, 2003) and monocular rivalry or pattern rivalry (Maier et al. 2005). Maier et al. (2005) described how pattern rivalry elicits a wavering percept when superimposed moving visual patterns enjoy temporary periods of exclusive visibility. The perceived visual composition is shaped by the brain’s global interpretive assumptions of the stimuli.

The literature on human perception suggests afterimages are enabled by the brain’s visual short-term memory capacity. Consequently, afterimages can be accessed from iconic memory up to four seconds after the stimulus has disappeared from view. Visual short-term memory is then overwritten with new stimuli presented to the eyes, with a limit of storing about four objects at a time (Sligte et al. 2008). As

---

9 Visual replacement of the target image by the mask is referred to as object substitution (Sigman et al. 2008).

10 When two stimuli are presented in short temporal succession, features of the two different stimuli may be combined or inherited, by the other stimulus, a phenomenon referred to as feature inheritance (Sigman et al. 2008).

11 How the visual system constructs descriptions of the environment in terms of objects, their 3-dimensional shape, and their motion through space, on the basis of the changing image that reaches the eye (Ullman, 1979).
shown later, the illusionary images experienced from AVGS are a direct result of the brain’s capacity limit of visual short-term memory.

2.5.2 Rhythmic Photic Stimulation

The knowledge that a flickering light can cause visual hallucinations is something humans have known since the discovery of fire. (Hutchinson, 1986, p. 225)

Hutchinson (1986), in his book *Mega Brain*, describes many examples of ancient scientists’ fascination with the phenomenon of flickering light. In 125 AD, Apuleius experimented with the flickering light produced from the rotation of a potter’s wheel, noting that it induced a type of epilepsy. Similarly, in 200 AD Ptolemy studied the flickering light phenomena generated by sunlight emanating through the spokes of a spinning wheel. Ptolemy noted a feeling of euphoria that followed from the strange patterns and colour that appeared in the eyes of the observer (Budzynski, 2006, p. 2).

Charles Benham (1895), an English toy maker, invented the artificial spectrum top, also known as Benham’s top. When the disk is spun, pattern-induced flicker colours, called Fechner colours, are perceived by the brain. Benham’s top was the first manufactured consumer object to produce pattern-induced flicker colours (Geiger, 2003).

Marcel Duchamp constructed one of the first optical machines, the Rotary Demisphere, in 1925 (Clair, 1978). The rotary demisphere (seen below in Figure 14) was made from a disc containing a spiral pattern rotated at various speeds to create three-dimensional illusions of spiral forms. In 1935, Duchamp expanded on his earlier work with the invention of the Rotoreliefs. Although the device was similar in construction to the Rotary Demisphere, it allowed Duchamp to experiment with a wider variety of spiral patterns in order to explore optical effects. Some of the spiral patterns Duchamp invented for the Rotoreliefs can be seen in Figure 15.

12 The Fechner colour effect is an illusion of colour seen while looking at rapidly changing or moving black and white patterns.
Figure 14: Duchamp’s original optical machine, the Rotary Demisphere (Duchamp, 1925)

Figure 15: Spiral patterns Duchamp created for the Rotorelief to induce illusionary optical effects (Duchamp, 1935)
This illusory technique has visual similarities to the flicker effect produced by strobe machines and the ‘Dreamachines’ constructed in the early 1960s by artists Brion Gysin and Ian Sommerville (Hoptman, 2010). A Dreamachine is made from a cylinder with slits cut out and a light bulb suspended in the centre. The cylinder is placed on a turntable that rotates at either 45 or 78 revolutions per minute; Viewers sit in front of the Dreamachine with their eyes closed (Figure 16) and experience pulsating light behind their eyelids at a frequency of 8 to 13 pulses per second. The optic nerve, when stimulated at this rate, alters the brain’s electrical oscillations to the frequency of alpha waves, normally associated with relaxation. Although one can achieve the same alpha state through focused meditation, devices such as this allow quick access to stilling brainwave activity (Geiger, 2003).

Figure 16: Brion Gysin and William S. Burroughs experiencing the stimulating effects of a Dreamachine (Gysin & Sommerville, 1974)
While Duchamp’s inventions were the first to consider art as a pure retinal experience, the Dreamachine inventors and Op-Artists\(^{13}\) such as Bridget Riley and Victor Vasarely also adopted this same approach. San Francisco artist Nate Boyce described how, by strobing multiple different colours in sequence, colours seem to fuse into a composite retinal image and create various afterimages, comparable to hearing multiple different notes on a piano played at once. This enables one to see multiple colours in time simultaneously (Boyce, 2010).

In 1946 neurologist W. Gray Walter, after using an electronic stroboscope for physiological experimentation, presented the results of the first systematic study of the flicker effect, suggesting photic stimulation produced results comparable to hallucinogenic drugs (Geiger, 2003, p. 21). A later study measured the EEG responses of transcendental meditation practitioners during exposure to flickering light. Despite the fact participants were not actively meditating, all reported entering into an altered state of consciousness and reported, at the conclusion of the experiment, that they had indeed been meditating (Williams & West, 1975).

Flickering light phenomena used to induce a non-ordinary state of consciousness are produced by Jack Schwarz’s Integrating Stimulating Intensity Stroboscope (ISIS) device. The device uses repetitive rhythmic sounds and a variable light frequency in goggles that enable the wearer to achieve certain mental states (Budzynski, 2006, p. 2).

### 2.6 Community of Practice

I now provide a brief background on how I came to this research and discuss influential artists, software and works within my community of practice.

#### 2.6.1 My Personal Backstory

At the age of four I began studying trumpet, and began piano tuition the week I started primary school. It was at this point that I became fully immersed in music. Throughout primary and high school, I divided my non-academic time between practice for exams on trumpet, piano and musical theory, and rehearsing in classical and jazz ensembles.

---

\(^{13}\) The Op Art movement emerged in the mid 1960s and introduced an emphasis on the immersive experience. Op Art shifted the focus of pictorial art from narrative to the nature of vision, engaging the process of perception as both a tool and a subject (Houston, 2007).
During my secondary school years I received intensive tuition from renowned jazz instrumentalists and was subsequently invited to study jazz improvisation at the Western Australian Academy of Performing Arts (WAAPA). It was during this time that I developed a serious interest in creating and performing electronic music – so much so, that after two years in the jazz course at WAAPA my musical passion had shifted and I returned to Melbourne.

I then studied sound production and audio engineering and after graduating, was accepted to study jazz improvisation at the Victorian College of the Arts (VCA). At the VCA I studied improvisation using traditional and electronic instruments and was encouraged to integrate my passion for jazz and electronic music. By the time I graduated from the VCA my major interest was electronic sound production. I felt I could make a greater contribution to the field of electronic music than to the established world of jazz.

The next 18 months were spent composing electronic dance music using Live (Ableton, Berlin). During this period I became frustrated that my sound design skills were not up to par with those of my composition and music theory skills. To rectify this I purchased a Symbolic sound Kyma system, and temporarily shifted my attention to understanding and learning Max/MSP (see section 2.4.5). Two months later I travelled to San Francisco to attend the first ever Cycling 74 conference on Max/MSP/Jitter. Whilst attending the conference I met the creators of Max and received personal instruction on how to program with it. At the same time I was introduced to Jitter (the real-time visuals side of Max). It was at this conference that I not only acquired the skills to start creating my own programs, but saw the potential of using Jitter and Max/MSP to create real-time synchronised audiovisual software.

My honours research at RMIT in 2009 was on Interactive Multimedia Installations and designing interactivity with gestural interfaces. The culmination of the research was the creation of custom audiovisual manipulation software called Jitterbugs (Figure 17). Jitterbugs enabled synchronised audiovisual manipulation via a single touch-screen slider, and was demonstrated at the Apple Create World conference in Brisbane (2009) and at the New Instruments for Musical Expression conference in Sydney (2010).
Through creating Jitterbugs, I found my creative and aesthetic ideas for audiovisual manipulation had exceeded Max’s capabilities. Although the prototype had facilitated simultaneous manipulation of audio and visual relationships, in order to continue my research, I needed to learn to write code. As a result, Java and especially C++ became my tools of choice for composing sound, music and image during my PhD candidature.

2.6.2 Chris Cunningham

Chris Cunningham worked as an engineer and sculptor alongside Stanley Kubrick in the mid-90s and subsequently made a name for himself as a music video director. The next decade saw Cunningham produce music video clips for artists such as Aphex Twin, Squarepusher, Autechre, Bjork and Madonna, earning him a cult following of loyal fans. Throughout his work, Cunningham portrays the visual aesthetics in a rhythmical manner analogous to that of a musical composition. The video clips *Come my Selector* (*The Work of Chris Cunningham* 2003) and *Rubber Johnny* (2005) are prime examples of Cunningham’s intelligent use of scene cuts and post-production editing to emulate the syncopated rhythmical elements of the music. However, Cunningham’s recent triple-screen laser live show (Figure 18) produced the greatest impact on me as an artist when I witnessed his performance at the Sydney Opera House in 2011. During the immersive performance, the spatial and temporal interplay between the laser beams and multiple projection screens conveyed the impression of a visual orchestra. At times, the music and accompanying synchronised visual content was either so confronting or fast-paced that audience members around me were unable to look. In contrast, I found the synchronised audiovisual sensory overload experience Chris’s performance delivered to be one of my most satisfying, profound and influential experiences.
2.6.3 Aphex Twin / Weirdcore

Richard D. James, also known as Aphex Twin, has had a very strong influence on myself as a musician and performer. His highly detailed production and frenetic beats did not sound like any genre of music I had heard before. Aphex Twin named his own style of music ‘braindance’. Listening to braindance (i.e., music so fast and complicated only your brain can keep up) was a similar experience to appreciating a good jazz record. I resonated with not only with the sound and its relationship to jazz but with Aphex Twin’s complete disregard for existing musical styles, instead producing his own unique sound. Similarly, Aphex has a reputation of hiding his persona from the crowd when performing, preferring the music and visual experience to speak for itself. As a performer, this attitude has rubbed off on me immensely and is part of what motivated me to start creating visual experiences to accompany my music. Recently, Aphex’s live performances have featured visuals by the artist Weirdcore (seen below in Figure 19). Weirdcore’s digital glitch aesthetic is very similar to mine in that he utilises stroboscopic patterns to produce retinal afterimages.
Theoretical, Technological and Artistic Influences

2.6.4 Squarepusher

The musician who has influenced me the most is Tom Jenkinson, also known as Squarepusher. Over two decades, Squarepusher has transcended the boundaries of traditional jazz harmony and electronic music production to produce some of the most innovative music of our time. During a 2001 interview, Jenkinson described his compositional approach to dissolving the traditional roles of instrumentation as follows:

I like to take something and make it do what it’s not supposed to do… for instance taking the traditional roles of drums, bass and melody and making it so they actually appear to swap places, almost like the drums become the melody, or the bass becomes the melody, or the melody becomes the drums. (Squarepusher techtv interview, 2001)

I approach electronic music production in a similar way to Jenkinson, particularly in the ways I visualise music. Consequently, I have applied this philosophy to creating audiovisual relationships in order to challenge popular conceptions, and to dissolve the traditional aesthetic boundaries between audio and visual media.

In 2012, Squarepusher toured his live show for the album *Ufabulum*, featuring a wall-sized screen and a helmet made from LEDs that reacted to the music in real-time (Figure 20). Auditory volume and LED brightness were the central audiovisual relationship employed for the show. The brightness of the
LEDs reacting in conjunction with musical transients facilitated a highly immersive environment for the audience.

![Image: Squarepusher performing his live audiovisual Ufabulum show at Simple Things Festival, Bristol, 2012 (Dolan, 2012)](image)

**Figure 20:** Squarepusher performing his live audiovisual Ufabulum show at Simple Things Festival, Bristol, 2012 (Dolan, 2012)

### 2.6.5 Reflexus

Reflexus is a real-time audiovisual performance (Figure 21) that explores the relationship between gestures, spoken word and digital audiovisual sampling. The project was developed by Play Modes utilising openFrameworks (open source code) for visual capture and processing, and Reaktor for audio processing. Their software captures and processes audio and video in real-time and is what motivated me to utilise a ring buffer for AVGS (see section 6.1.3). During the performance Reflexus english (2009), multiple captured audiovisual copies become distorted in time, allowing the actor to strike up a conversation with his past digital self.
2.6.6 Modell5

Modell5 is a live audiovisual performance by the Austrian artist duo Granular Synthesis (1996). Personally, the aesthetics of Modell5 was a key inspirational force that motivated my decision to undertake this research. The immersive performance features loud sub-frequency sound and violent digital manipulation of Japanese performance artist Akemi Takeya’s face. The work comprises video content manipulated frame-by-frame and projected onto four wall-sized screens (seen below in Figure 22) to create a 44-minute work from the granular recombination of six seconds of material. The highly detailed manipulation demands the viewers’ attention and challenges the audiences’ perception of the audiovisual material.
Varp9 is audiovisual re-synthesis software developed by Dirk Langheimrich in 1997 for the group Granular Synthesis. The software streams frames at 25 fps from random access memory (RAM) and was one of the first software examples of real-time visual granular synthesis. Varp9’s specialty is its flicker engine that can mix two discrete video streams together by splicing them together frame by frame. This generates a visual effect of the two streams seemingly intersecting each other. Varp9 can be interacted with in real-time using either a MIDI controller or through interacting with the software’s user interface. Figure 23 shows the Varp9 user interface.
2.6.8 Autechre – Gantz Graf

In 2002 Alex Rutterford produced a music video clip for the electronic duo Autechre entitled *Gantz Graf*. For over a decade, many have regarded Gantz Graf (Figure 24) as one of the greatest examples of audiovisual synchronisation within the fields of computer graphics and computer music (Lamb, 2011). Although some may not connect with the cold, emotionless digital aesthetic of the work, I still find the challenging complexity of the work very significant. To produce the work, Rutterford spent six months manually creating and synchronising the visual object with the audio. During an interview in which he talks about his creative process, Rutterford observed:

> Everyone says ‘How long did it take you?’ , ‘How did you do it?’ …. I’d really love to be able to say to them, ‘I just wrote a computer algorithm and the computer did it all. I wrote a program and it just intelligently works it out’. But it doesn’t exist. It’s fools’ gold thinking that someone can sit there writing a piece of software that can make intelligent decisions about pace and animation. The closest I have seen is perhaps iTunes. (Rutterford, 2002)

Despite Rutterford’s apparent disbelief that a computer program is capable of creating intelligent synchronised audiovisual aesthetics, many audiovisual artists (including myself) have used the work as an aesthetic benchmark for creating real-time synchronised audiovisuals.
2.7 Summary

In this chapter I have referred to literature, theories and technologies that describe or were attempts to integrate audio and visual phenomena for aesthetic purposes through identifying subjective similarities between the two media using pre-digital and digital technology. I have summarised notions of performative strategies used by live cinema artists in comparison to cinema and DJ/VJ performance, specifically the tendency of live cinema artists to build their own custom tools as opposed to using off-the-shelf commercial software aimed at DJs and VJs. I then outlined the development of granular synthesis into a real-time audio processing technique in order to identify technical and compositional conventions. Examples of illusionary art and research into visual sensory phenomena were presented to understand the reason retinal afterimages are produced by AVGS. Finally, I discussed my personal back story and my relation to a community of practice in order to situate my practice within the field of audiovisual art by providing a brief review of related audiovisual software and performance practices which relate to my research project.
3 Methodology

3.1 Overview
This chapter provides a description of and a rationale for the methods and materials used to develop my research project. Firstly, I outline the nature of practice-based research and how this methodology supports my research. Secondly, I consider the three key questions explored in this study in relation to practice-based methodology. Thirdly, I present a description of the software used in my research practice: in particular how ‘hacking’ was used to evolve and develop the project prototypes. Fourthly, I briefly discuss the hardware used for my test platform, the programming environments considered for the creation of the project prototypes; the process undertaken to select appropriate audio synthesis libraries; and the cyclical process of action and reflection that guided the evaluation of qualitative and quantitative elements of my research.

3.2 Ontology and Epistemology
In my research, I adopted an interpretivist worldview in order to produce new knowledge. Carson et al. (2001, p. 5) defined interpretivism as an approach that allows the focus of research to be on understanding what is happening in a given context through interpreting the phenomenon under study. Furthermore, interpretivist research emphasises the importance of the researcher in the phenomenon, the existence of the researcher’s pre-understanding and the experiential learning which influences both the focus and progression of the research and the development of interpretive analysis (Carson et al. 2001). Weber (2004) stated that interpretivist research may be considered valid if a peer is willing to concede that the researcher’s claims are reasonable and plausible, based on the evidence collected and the context in which the research was undertaken.

3.3 Practice-Based Research
Practice-based research was described by Candy (2006) and Edmonds et al. (2005) as an investigation undertaken to gain new knowledge through the means of practice and to transfer knowledge through an evaluation of the outcomes of practice. Edmonds et al. (2005, p. 460) stated that while the aims of practice result in a creative artefact, the aims of practice-based research also generate new understandings or knowledge. In this way the co-evolution of two interdependent yet complementary processes give rise to two main outcomes of practice-based research. In my research these outcomes are the creation of an innovative, audiovisual software prototype that will be used synergistically
during a public performance in December 2014 and an evaluation and analysis of the processes leading to these outcomes which will contribute to new knowledge in my field.

3.4 Field of Practice

Within my field of practice it is quite common for artists to develop custom tools to portray their own unique creative voice as opposed to using off-the-shelf commercial software packages. This fact is exemplified by the artists I discussed in section 2.6, such as Granular Synthesis, Play Modes and Squarepusher, who have all been compelled to develop custom tools in order to create and perform their works. My field of practice is also closely aligned with the open-source creative coding community, specifically tools, languages or programs such as Processing (open source code) and openFrameworks (open source code). The Processing and openFrameworks communities design creative tools using cutting-edge software libraries that eventually make their way into commercial software (Kirn, 2014). In this sense, it is common that the first glimpses of new media processing techniques and art emerge from within these communities. This means artists and hackers have the potential to influence the direction of new aesthetic digital vocabularies for other fields of digital arts practice. For myself, as an artist and researcher, I chose to be a part of this field as it stands on the forefront of aesthetic and technical innovation.

3.5 Research Questions

In order to carry out my project, I posed two main research questions:

1. How can audio processing techniques such as granular synthesis be adapted and applied to influence new visual performance techniques?

2. How can computer software synergistically integrate audio and visuals to enable the real-time creation and performance of audiovisual art?

The first question required an assessment of how to adapt audio processing techniques to a visual medium. The second question necessitated evaluating how to effect real-time creation of audiovisual art through defining complementary audiovisual relationships. Three research goals were related to these questions:

(i) to establish a visual granular synthesis processing technique
(ii) to create a synchronised audiovisual software environment
(iii) to realise the creation and performance of audiovisual art in real-time.
3.6 Software Prototypes

The software prototypes for Kortex (the project instrument) evolved through an iterative process of action and reflection relating to the design and performative function of the instrument. Each prototype was designed to address specific requirements such as functionality, code optimisation, integration of audiovisual elements and artistic potential. These requirements were evaluated during each of the developmental stages. Functionality for the prototypes emerged from referencing research in the field of audiovisual art, graphics programming and sound processing, and through consultation with peers and online forums regarding the implementation of specific graphics and sound processing techniques. In addition, Github’s (2014) online code repository provided useful analytics regarding the development of the prototypes. At the time of writing, the Kortex prototype is currently at version three. Each version involved a significant enough change to warrant either a full rewrite of the prototype’s code or redesign of core functionality. Please refer to the Appendix to see the main features of and differences between each version.

3.6.1 Hacking

The practice of ‘hacking’ played a useful role in the development of Kortex. Hacking combines strategies from art, technology and design to find interesting and optimal solutions to technical problems. The hacker ethic first originated in the 1950s through the work of students of the Tech Model Railroad Club at the Massachusetts Institute of Technology (MIT). According to Levy (2010), in his definitive book on the concept of hacking, *Hacking: Heroes of the Computer Revolution*, these students lacked any training and instructions about MIT’s mainframe computer but were motivated to create “art and beauty on a computer” (p. 31). The original hacker ethic differs from the common misuse of the term as a cyber-criminal act. By adopting the hacking approach to bring together strategies from art, technology and design, I produced software prototypes in an iterative process to achieve the goals of my research.

3.6.2 Development and Test Platform

The test platform consisted of the following hardware: early 2011 17-inch Macbook Pro featuring a 2.3GHz Intel Core i7 CPU; 16GB 1333MHz DDR3 RAM; AMD Radeon HD 6750M graphics processing unit (GPU) with 1024MB of RAM and a SanDisk 480GB Solid State SATA Hard Drive (SSD) and an extra 16GB of RAM.
To allow for a higher resolution to run at 60 fps on a triple-screen setup, I built a hackintosh\(^\text{14}\) consisting of the following hardware: 3.9GHz Intel Core i7 4770k Quad Core CPU; Corsair Vengeance Pro 16GB 1333MHz DDR3 RAM; Z87E-ITX Mini ITX Motherboard; SanDisk 480GB SSD; Galaxy GeForce Nvidia GTX Titan GPU with 6GB of RAM enclosed in a mini-ITX case. The mini-ITX form was selected for portability rather than performance reasons; it can be taken as carry-on luggage for interstate and international performances. At the time of purchase, the CPU and GPU were the most powerful on the market, enabling my prototypes to run at 60 fps on a triple-screen setup rendering at a resolution of 1024x1024 pixels each.

### 3.7 Evaluation

The methods and processes discussed above describe a cyclical process of action and reflection. The iterative process of reflective practice led to the creation and evaluation of the project prototypes and associated new understandings and knowledge in my field of audiovisual arts practice. New insights generated through the process of evaluation were fed back into the development of subsequent software iterations. My research involved the collection and analysis of both qualitative and quantitative data. As a result, I used a mixed method research design approach (seen below in Figure 25) in order to evaluate the quantitative and qualitative aspects of my research.

![Figure 25: Mixed method research design](image)

During reflection, unexpected ideas often arose that differed from the objectives being pursued. For example, through reflecting on the grain reordering technique developed to manipulate AVGS (described in section 4.3.1), I realised the technique had further potential to manipulate the parameters

\(^\text{14}\) A hackintosh is a non-Apple computer that runs the Mac operating system. Hackintosh’s can be built and customised using a variety of PC parts not officially supported by Apple.
of the frequency modulation (FM) synthesis engine\textsuperscript{15} (described in detail in section 6.1.5). Although I had not intended the grain reordering technique to control FM synthesis during development, further applications became evident during use and reflection. These side effects were welcomed and explored for further developmental possibilities.

The evaluative performance cycle process is illustrated in Figure 26 and referred to in detail in Chapters 5 and 6.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure26.png}
\caption{The cyclical process of action and reflection used to evaluate and evolve the Kortex prototype for public performance}
\end{figure}

### 3.7.1 Qualitative Data

I investigated how granular synthesis audio processing techniques could be adapted and applied to visual material through the development of a prototype that combined audio and visual parameters as complementary constructs. While the real-time performance of the prototype can be measured through software analysis tools, it is difficult to measure the experience of complementary audiovisual

\textsuperscript{15} In Kortex, the FM synthesis engine can be driven from a number of control sources and is used to create audiovisual correlations in conjunction with the visual shader synths.
constructs. The qualitative data referred to below was analysed and the results fed back into the development of subsequent prototypes.

Qualitative data in the form of notes, images, screen recordings and audience feedback were collected at various stages throughout this research. I received feedback at the completion of each performance and made observations during the development and use of the prototypes. During development of each software prototype, I would often take screen shots and utilise screen capture software when I stumbled on unique parameter settings that created pleasing audiovisual aesthetics. The collected recordings and images were then used to remind me of unique states Kortex was capable of generating when combining specific techniques and processes. I was able to refer to these aesthetics when preparing material for public performances. In order to illuminate the various functionalities of the prototype, a selection of the visual data collected is presented in this exegesis. A selection of screen recordings is presented on the USB stick submitted with this exegesis that demonstrate selected audiovisual functionalities of the final prototype.

### 3.7.2 Quantitative Data

The development of the software prototype for this research project required real-time performance. Performance was therefore constantly tested and measured to determine the efficacy of critical functionality. This was achieved using the Quartz Debug application seen in Figure 27(a) to provide real-time visual feedback on the current speed of Kortex, and also through using the industry standard OS X OpenGL code profiler developed by Apple (OpenGL, 2014) seen in Figure 27(b) that measured the speed of code execution and calculated average fps while Kortex was running.

![Quartz Debug Frame Meter](image1)

![OpenGL Profiler](image2)

**Figure 27**: Performance analysis tools used during the development of Kortex to ensure consistent performance of 60 fps
During development I referred to the statistics window of the OpenGL code profiler (seen below in Figure 28). The statistics window provided data about functions that consumed the most CPU and GPU cycles that helped me to debug potential bottlenecks.

**Figure 28**: OpenGL Profiler statistics used to identify expensive CPU and GPU calculations in Kortex
3.8 Validation of Key Research Outcomes

The development of my research project led to the publication of a regular paper (Batty et al. 2013), demo paper (Batty, 2013) and demo performance at the Interactive Entertainment Conference 2013 (RMIT, Melbourne). Both papers and the demo performance were subject to peer review and feedback. The papers presented key findings relating to the real-time granular synthesis of audiovisual media in order to obtain feedback and confirmation from academic peers and industry experts. Feedback received from this process included framing AVGS within a musical context and the potential for AVGS to process live inputs. The Kortex prototype discussed in this exegesis evolved in part from audience feedback and personal experience performing at local and international events. Early prototypes of Kortex were performed and demonstrated at the 2012 Solar Eclipse festival in Cairns, Rainbow Serpent and Earthcore festivals in Victoria and the Burning Man festival in the USA in 2013.

3.9 Summary

In this chapter I outlined the design of the investigation. I explained the process of the investigation and creation of audio and visual processing techniques, and my approach to embedding the techniques in software that integrates audiovisual relationships in real-time audiovisual art performance. To progress knowledge I adopted two research methods: hacking and a cyclical process of action and reflection. As this research project included the development of software, I collected qualitative and quantitative data to evaluate and progress the artefact. The iterative evolution of the artefact took into account qualitative data, in the form of audience feedback, notes and screen shots, and quantitative data, in the form of software performance analysis. The research validation followed the accepted method of peer review through publication.
4 Intention and Aesthetics

4.1 Overview
This chapter describes the aesthetics of my audiovisual arts’ practice. Firstly, I present how my aesthetic practice (the way my audiovisual art is constructed, gives expression, and offers ways of knowing) is conjoined with my compositional and performance intentions; secondly, how my aesthetic practice is interpreted through an audience’s perception of my artistic intentions; and thirdly, how my aesthetic practice relates to the way my work affects both the audience and myself through the medium of performance technologies. In my project performance, performance effects are derived through the audiovisual aesthetic relationships made possible through Kortex.

4.2 Intentions

4.2.1 Why Real-Time is Important
From being a trumpet- and piano-playing jazz musician to an electronic performance artist, my creative practice has always involved aspects of the spontaneous and impulsive. At first, this may seem at odds with the algorithmic world of digital technology. However, I was drawn to computation because of the opportunities it offered for the creation of new (or at least expanded) sonic and visual environments, including the creation of non-traditional instruments. The computer provided me with the potential to create any sound imaginable. I accepted the idea that all digital media are a means for creating new sounds and images in creative performances; however, the creative expansion that computation offered my artistic vocabulary would have proved meaningless if it also was not able to be approached as an instrument – a tool for the performance of audio and visual materials. As a result, my project has involved the creation of, and performance with, a new digital instrument which facilitates the spontaneous and impulsive facets of my performance practice.

My familiarity with these modes of expression enables me to interact with the instrument to create audiovisual experiences, while reacting in real-time to the event and the audience. This is important to me because it creates the performance conditions under which impromptu events can unfold and emerge. Consequently, my performance strategy is to enable the audience, instruments and myself as the performer to directly influence the audiovisual experience throughout the performance.
This performance strategy contrasts with those of local and international acts I have supported over the past four years, in which performers pre-prepare their sets and merely stand behind their laptops for the duration of the performance. Essentially, the performance strategy of these artists is to write themselves out of the performance. In doing so, their chance of failure is decreased, yet the potential for an audience and the performer to influence the sensory experience is also minimised. I would argue that the risk of potential failure or success is what makes witnessing a live performance so captivating and is ultimately the difference between listening to a recorded performance and experiencing the real thing. As an artist, having the ability to influence unique creative outcomes from one performance to the next helps to maintain the personal satisfaction I achieve through performing, as each performance is unique to its time and place.

4.2.2 Immersive Environments

One of my main composition and performance intentions is to provide an audience with an immersive audiovisual sensory overload experience. The immersive environment is facilitated by employing a large sound system and three large projection screens that serve to engulf the audience’s field of view with visual imagery. The experience of sensory overload is produced through combining different audiovisual techniques in various combinations. One technique is to manipulate visual imagery close to the threshold of human perception so as to create composites images in the viewers’ mind that do not actually exist in reality. The rate of audiovisual stimuli presented to the brain renders any attempt to make sense of the content futile; therefore, an audience is encouraged to abandon their natural instinct to understand what they are witnessing and instead succumb to the ineffable nature of the sensory experience.

My reason for wanting to create such an experience is to instil a sense of wonder and awe in the audience. This is similar to the performance intention of a magician: asking an audience to suspend belief while they perform an illusion that cannot be immediately understood. Indeed, it is for this very reason that most people attend magic shows, as they derive pleasure from witnessing the unexplainable (Bishop, 2013). Personally, my favourite musicians and artists are those who are able to perform their craft at such an aesthetic and technical level that I abandon my natural instinct to analyse; instead, I become directly connected to the sensory experience. The pleasure I derive from such experiences is what motivated my intention to create immersive audiovisual sensory overload environments.

4.2.3 Musical Context

While creating instruments that successfully integrate audiovisual media for artistic expression is of
interest in itself, of primary significance is the context in which they are presented and performed to an audience. In my project, the visual synths are set up to only react to musical data. Thus without musical context they are unable to become animated. In contrast, AVGS, as defined in section 1.3.1, was created as an open-ended technique in and of itself. When I was creating the technique, I was conscious to not impose any biases on how it should be used or applied. Consequently, the AVGS technique may be experienced on its own or in the context of musical information. It is up to each individual artist to choose how they would like to work with the technique, based on the specific application and their own aesthetic practice.

My aesthetic practice involves featuring AVGS with and without musical context. Firstly, musical context provides a perceptual framework for an audience to experience the abstract nature of AVGS. As a result, the familiar experience of musical momentum is used to present a sequence of digestible states to an audience. However, there are times when it is preferable to feature AVGS without musical context, using it to invoke a feeling of timelessness in an audience. The contrast created by juxtaposing the two states has the potential to amplify an audience’s emotional response to the technique.

4.3 Aesthetics

Aesthetics can be defined as a set of principles underlying the work of a particular artist or artistic movement for example, the Cubist aesthetic (Oxford Dictionary online, 2014).

4.3.1 Audio Visual Granular Synthesis (AVGS) in Detail

The primary aesthetic quality of AVGS lies in the structuring and placement of audiovisual grains over time. Although different audiovisual content fed through AVGS will trigger a unique perceptual relationship to the work, of primary importance are the underlying patterns that define the temporal re-orderings of the audiovisual grains. In this way, the AVGS technique is not biased to specific content and encourages playful experimentation using a wide range of source material. I have experimented with source material ranging from computer-generated audiovisuals, pop star music videos clips, martial arts fight scenes to streaming live television channels.

The fascinating qualities that emerge using any of this material is the perceivably impossible forms and behaviours that can occur as a result of digitisation. For example, AVGS applied to a martial arts scene can produce an illusion of the attacker performing physically impossible strikes on an opponent. Additionally, AVGS applied to musical source material featuring a vocalist will facilitate the emergence of unnatural melodic patterns. Essentially, the effect of observing the digitally manipulated
human form short-circuits our hardwired expectations on how the form should behave.

Figure 29 shows a collection of strategies that may be employed to modulate and shift the perceived temporal structure. Each control strategy facilitates a unique perceptual relationship between the viewer and the resulting audiovisual aesthetic. Within each pattern, the user can control the speed at which the pattern is read through, the direction of the play head through the pattern, and the pattern’s amplitude. Subtle manipulations of these variables during a performance can have a dramatic effect on how the audiovisual construct is perceived.

![Figure 29: Three grain reordering patterns that display how the audiovisual grains are re-ordered over time](image)

The grain reordering patterns presented below help the viewer to experience the illusion of seeing multiple points in time simultaneously via harnessing persistence of vision. Although the shapes reorder both the auditory and visual grains, the brain perceives the auditory and visual stimulus in different ways. As the rate of reordering increases from 20 to 60 times a second, the auditory grains begin to fuse into a more continuous tone. Likewise, as the rate increases the brain perceives a greater overlapping visual montage, as a result of the visual short-term memory phenomena discussed in section 2.5.1. Primarily, the retinal afterimages that AVGS produces facilitate the perceived breakdown of logical time and assist the performer to create a sensory overload environment for the audience.
Grain Reordering Patterns:

- Loop Mode Forward (Ramp) = Machinegun-like effects (very rhythmical)
- Loop Mode Reverse (Saw) = Time reversed machinegun-like effects
- Loop Mode Palindrome (Triangle) = Similar to the previous mode except the temporal momentum of forward moving time is blurred
- Shift Pause (Square) = Time freezing. Play head jumps to positive and negative affecting the audiovisual grain to be frozen
- Random (S&H) = No perceivable pattern to the re-ordering
- Exotic (Additive) = Blending two oscillators. Manipulating the two oscillators’ amplitude and frequency parameters can create a near infinite amount of patterns.

Please refer to ‘AVGS Grain Reordering Patterns.mov’ in my portfolio to see an example of how these patterns reorder audiovisual grains throughout time.

4.3.2 Visual Synths

The creation of audiovisual aesthetic relationships is a highly subjective process and will differ between artists. Although the quality of the resulting aesthetic relationships is also subjective, there are combinations I find generally satisfying. These usually occur between the strong musical features, for example, kick drum, snare drum and loud musical transients, and visual parameters that have the greatest perceived impact on the visual scene, such as brightness, shape size and virtual camera position. Frequently, these types of perceptual relationships reinforce one another and help to direct an audience’s attention to the strongest musical features.

An example of this is mapping a snare drum playing a backbeat (occurring on the down beat of beats 2 and 4 of each bar) to trigger a random camera position in the 3D scene. At the same time other rhythmical elements, such as high hats and ghost notes, will usually be assigned to manipulate the 3D scene’s geometry. The viewer is then able to appreciate the rhythmical relationships between the music and the geometry from a unique perspective every two beats. This relationship works aesthetically by giving the audience enough time to adjust to the new visual perspective and to appreciate the other audiovisual relationships before changing again. If the camera position was to change each time a hi-hat was played, then all other music-to-visual relationships would be unable to be perceived by the audience. Because of this, perceptual relationships are more convincing when forefront musical elements are assigned to dramatic visual scene parameters, while 3D geometry
parameters work best when assigned to short rhythmic musical fills. Please refer to ‘Jennifer Max/Visual Synth Relationships.mov’ to witness the above audiovisual mapping.

The above is only one specific example that I use to produce a convincing audiovisual aesthetic correlation. Obviously, the success of the audiovisual correlations will depend on the type of audio and visuals that are to be integrated. Nevertheless, there are several direct one-to-one cross-domain mappings that an audiovisual artist can employ to achieve convincing audiovisual correlations. Table 2 contains an adapted collection of specific strategies outlined by Fredrik Olofsson (2009) that illustrates this point.

Table 2: Potential correlations between audio and visual characteristics, adapted from Olofsson (2009)

As the Table shows, each audiovisual correlation can be inverted. The perceived effectiveness of the inverse correlations is dependent on the sound and image elements to be mapped. I personally find the inverse correlations feel less natural. However, from a compositional standpoint, the inverse correlations can be used during a composition to provide ‘tension’; switching back to the normal correlation then provides the necessary ‘release’. The technique of tension and release is well known to music composers and can be utilised to function like an audiovisual cadence.

Finally, each unique correlation may become lost on the audience if too many are occurring at once. To address this, Olofsson (2009) suggested that an audiovisual artist employ only a few obvious correlations that are subsequently evolved throughout a composition. In my own work, I enjoy giving
the audience periods in which they can easily understand specific audiovisual relationships and periods of sensory overload, in which so many audiovisual correlations occur at once that the brain is unable to make sense of them all. Both methods can be utilised throughout a composition to enable dramatic contrasts from one section to the next.

4.4 Summary

In this chapter I outlined the performance intentions and effects of my aesthetic practice. As an artist my intention is to create and perform audiovisual sensory experiences in real-time and to explore how musical context can provide a conceptual framework to deliver these experiences to an audience. I showed that, while grain reordering techniques are the primary aesthetic feature of AVGS, different grain reordering patterns can shift an audience’s sensory perception of the audiovisual construct. Finally, I described my strategy for creating audiovisual aesthetic relationships between musical and visual characteristics, and how the subjective nature of these correlations may be successfully portrayed to an audience.
5 Performance Iterations

5.1 Overview

The following section outlines the performance iterations undertaken during my research into real-time creation and performance of audiovisual art. Early in my candidature I worked with electronic musician and C++ programmer Mitchell Nordine to create MindBuffer (our performance duo). After writing our first audiovisual track, we were approached and signed to Enig’matik records as a live audiovisual intelligent dance music (IDM) duo. Our aim was to use MindBuffer and Enig’matik records as a platform to produce and perform cutting-edge and tightly synchronised audiovisual musical experiences. Each iteration below describes the technological, aesthetic and performance strategies we used for notable national and international MindBuffer performances over the past four years, and concludes with an introduction to the final iteration that will be used for my examination performance in December 2014.

5.2 Iteration 1 – Brown Alley (Melbourne) 2011

The first MindBuffer show was performed at Brown Alley in Melbourne for the launch of Enig’matik Records, June 24, 2011 (Figure 30). A considerable amount of work went into composing music for the hour-long set and programming visuals and a framework that synchronised the music from one laptop with the visuals on a second laptop. Although the performance was not as polished as we would have liked, the initial aesthetic and performance framework created for this performance set the foundation from which the other iterations evolved.
5.2.1 Performance Strategy

Technically, the performance setup consisted of pre-composed music separated into one of the following four stems\(^{16}\): drums, bass, melody and atmosphere. These were exported out of Logic (Apple Computer, California) as either an eight or sixteen-bar phrase. Once all the tracks were exported, the stems were imported into Ableton Live and arranged to create appropriate mixes (see Figure 31 below). In order to synchronise the musical elements with the visuals, I created a Max4Live\(^{17}\) device to analyse the volume data in Ableton and send it over a network using OSC. The custom Max4Live device was then inserted onto the drums, bass, melody and atmosphere tracks.

---

\(^{16}\) Stems are groups of instruments mixed together. For example, the original composition may feature four independent drum tracks, rather than exporting each part as a separate file, the four tracks are summed and exported as one file.

\(^{17}\) Max4Live is an extension to Ableton Live that enables users to program devices using MAX/MSP/Jitter from within the Live environment.
Figure 31: MindBuffer Ableton set showing each track divided into Drums, Bass, Melody and Atmosphere stems with a Max4Live device on each track on the bottom left

Figure 32 below illustrates the framework that was constructed to handle real-time visual sketch triggering and audio synchronisation. At the time, I was unfamiliar with openFrameworks and decided to use Processing to create real-time visual environments that evolved and reacted to auditory data. To maintain visual interest, I programmed seven unique Processing sketches to be triggered as the set evolved. Within each Processing sketch, I identified between four and six visual parameters, such as
shape colour, shape speed, shape size etc., to be controlled either by a touch screen interface or by the volume data coming in as OSC from one of the four audio tracks. These parameter assignments were capable of being redefined in real-time during a performance using a custom interface I created for the Lemur (JazzMutant, Bordeaux) touch screen controller.

![Initial performance framework for synchronising audio and visuals between two laptops](image)

After playing one sketch, to load the next, I triggered the name on the touch screen to send OSC data to a Max/MSP patch. The Max patch identified the incoming message which was then sent into a message box containing a shell script to find and load the corresponding processing sketch. I then manually compiled the code onstage and instructed Syphon\(^{18}\) to switch its visual output to the new sketch.

### 5.2.2 Reflection

**What Worked**

The technique of using Max4Live to send independent volume data for the drums, bass, melody and atmosphere enabled flexible synchronisation of audiovisual relationships. During the performance I

---

\(^{18}\) Syphon is an open source Mac OS X technology that allows applications to share visual frames with one another at full frame rate in real-time.
was able to reassign specific auditory control to a variety of visual parameters in real-time from my controller. This enabled me to emphasise musical phrases and transitions by assigning visual aesthetics to specific instrumentation as the composition evolved.

**What Didn’t Work**

Although the Max4Live synchronisation technique generated interesting visual results, the lack of preparation and rehearsal time before this performance meant there was a great deal of trial and error when assigning a musical element to an appropriate visual aesthetic.

The Jazz Mutant Lemur touch screen interface programmed for this performance did not meet my expectations as a performance interface. In order to manipulate a visual parameter I needed to focus on positioning my finger on the correct slider. This diverted my attention from the media I was controlling. The inability to engage simultaneously with the performance interface and the visual media led me to conclude that non-tactile performance interfaces do not facilitate intuitive control of visual media in a performance setting. As a result, I decided that a MIDI controller interface featuring physical knobs and sliders would alleviate this limitation for future performances.

Compiling code live onstage did not help the fluidity of the performance. Furthermore, the framework’s reliance on a multitude of different software environments not only led to potential confusion but also increased the risk of technical failure. If one part of the signal chain crashed, the whole framework could fail to work correctly, requiring a quick restart at best, or at worst, cease to perform for the remainder of the set. The technological risk associated with such a hacked bleeding edge framework highlighted the need for a unified visual performance environment.

Please refer to ‘MindBuffer – Iteration 1 Demo.mov’ in my portfolio for a short audiovisual demo of the framework and to ‘Enigmatik Records, Melbourne Launch – MindBuffer AV Show.mov’ to witness an excerpt from the performance.

### 5.3 Iteration 2 – Eclipse Festival (Cairns) 2012

MindBuffer was invited to perform at the weeklong total solar eclipse festival held outside of Cairns, Australia, 10–16th November 2012 (Figure 33). We were booked to play a one-hour live audiovisual set and to open for Richard Devine, one of my most significant creative influences in the field of IDM. I spent three months redesigning the framework used in the earlier performance, aiming not only to overcome the limitations of the previous iteration but to expand the range of visual aesthetics capable of being assigned to audio synchronisation.
5.3.1 Performance Strategy

I wanted to alleviate the previous iteration’s dependence on multiple pieces of software needed to handle loading, manipulating and displaying visual imagery throughout the set. To achieve this I created a unified framework using openFrameworks. The design of the framework was heavily inspired by my experience with a popular VJ’ing application called VDMX. In VDMX, the user is able to assign and play back visual content on two independent channels that can then be mixed or blended together to create visual collages, or to assist with transitions between visual media. I adopted this design concept and programmed a collection of different blend modes inspired by those found in Photoshop (Adobe, California). This enabled me to blend video files that contained interesting visual textures, such as complex fractal patterns, into the visual audio reactive synthesisers I had programmed for the set.

In addition to creating a unified environment, my goal was to maximise the potential for visual complexity whilst still achieving the highest possible frame rate. To achieve this, I programmed the audio reactive visual synthesisers and effects using a low-level programming language that ran on the GPU as shaders. Implementing shaders to handle the real-time visual synthesis generation and visual post-processing effects enabled a higher-quality and better-performing visual output than the previous iteration. This process also facilitated my creation of a complex visual signal processing effects chain.
that enabled further deconstruction of the image. Visual effects were controlled independently using my controller or synchronised by routing one of the four incoming audio tracks.

Finally, for this performance, I exchanged the touch screen controller for a physical MIDI controller. This was done in recognition of the need for a tactile performance interface that became evident during reflection on the previous iteration. Using Max/MSP, I programmed a variety of states that enabled changing the colour of specific pads on the controller in order to obtain visual feedback regarding the current visual synthesiser and audiovisual assignments. Figure 34 below shows the complex interconnectedness between various functions which enabled this visual feedback to occur. Although my aim was to eradicate the reliance on secondary software in order to reduce the complexity of the performance framework, I did not at this point possess the required C++ knowledge to implement this specific functionality.

![Custom Max/MSP patch](image)

Figure 34: Custom Max/MSP patch that enabled coloured visual feedback on the midi controller.

### 5.3.2 Reflection

**What Worked**

I found the tactile feedback of the MIDI controller offered a greater level of expressiveness and playability with the visual media than the touch screen controller. This tactility was developed from
my experience as a trumpet and piano player, and also from the ability to engage with visual media while manipulating the physical interface. The visual feedback I programmed into the pads on the controller enabled me to quickly see the current audiovisual assignments. Consequently, I was able to perform with the audiovisual assignments in a more expressive manner.

A more responsive and convincing synchronisation of audio and visuals was achieved through the process of eliminating multiple software dependency by creating a unified environment in openFrameworks. The integration of implementing visual synths and effects as shaders that ran on the GPU not only resulted in a huge performance boost, but increased visual detail and expanded the audio synchronisation possibilities to include visual post-processing effects.

**What Didn’t Work**

During the performance I found myself wanting an even greater level of sensitivity with the controller. I could only perform audiovisual re-assignments at the end of musical phrases which usually meant every 8 or 32 bars of music. Although I could perform manipulations over various visual aesthetics using the physical sliders, the linear manipulations offered by the sliders did not allow me to synchronise and engage with the highly rhythmic parts of our music. For this to occur in a future iteration, I needed to rhythmically perform with the music by making better use of the middle 16 pads.

Finally, I was not satisfied with the range of visual material available throughout the one-hour performance. It took too long to program the new framework from scratch and the visual complexity of the shader synths was limited as a result of my basic understanding of the OpenGL Shading Language (GLSL). For this performance I created eight visual synths to trigger as the set evolved. This equated roughly to one visual synth per one and a half musical compositions. Although I was able to reassign audiovisual relationships during pivotal moments in a composition, there were instances when the composition dramatically changed its direction and intent, specifically at breakdowns and drops. As a result, I was unable to match the same emotional and aesthetic range visually within each composition.

5.4 **Iteration 3 – Burning Man (Nevada) 2013**

MindBuffer was invited to perform three live audiovisual sets at the annual Burning Man festival held in Black Rock, Nevada on 26th August – 2nd September. The festival is one of the most prestigious art and electronic music festivals in the world and we were very pleased to be booked to play there for our first-ever international gig.
Each performance invitation came from a unique sound camp located within the festival. As a result, our audiovisual set had to be flexible enough to adapt easily to the varying stage setups that each camp had at their disposal. Furthermore, the Burning Man site is held on a very inhospitable geographic location. The environment is made up of fine corrosive alkaline dust particles that tend to clog delicate electronic performance instruments. After hearing stories about the corrosive dust wrecking the technical gear of past performers, we needed to ensure that we protected our equipment from sudden dust storms during a performance. This was achieved by using painters’ tape on the laptop speakers and any unused ports, and by covering our gear in bubble wrap, as shown in Figure 35.

![Figure 35: Burning Man laptop protection](image)

5.4.1 Performance Strategy

We were booked to perform one-and-a-half-hour sets for each of the three events at Burning Man – thirty minutes longer than any of our previous sets. Up until then, I had only used a collection of visual shader synths for the performances because I did not feel the visual granular synthesis technique was stable enough in a live performance setting. However I managed to swap content in and out of the visual granular synthesis engine and, due to the extra 30 minutes of content needed to fill the set, I decided to trial a combination of both visual shader synths and visual granular synthesis for these
In order to tighten the synchronisation between the music and visuals, I programmed another Max4Live device that sent a pulse over OSC at the beginning of each musical phrase. During the past two iterations, I could only receive information about the drums, bass, melody and atmosphere’s current volume. Expanding the assignable data to include information about the beginning of musical phrases allowed the visual aesthetics to react not only to a stream of volume data but to macro compositional information.

The framework was further developed to allow saving audiovisual assignments as presets to be recalled and triggered during a performance. When switching from one visual synth to another, appropriate audiovisual assignments for a previous scene would not work aesthetically with the new scene. To address this I triggered a new visual synth at pivotal moments in the set, such as the beginning of a new track. The contrast from one scene to another helped to reinforce the new direction of the music. Utilising presets helped to maximise this effect by automatically switching the audiovisual assignments to the most appropriate relationships for the newly triggered visual synth.

5.4.2 Reflection

What Worked

Aesthetically, several of the refinements produced noticeable improvements over previous iterations. The ability to instantly trigger predefined audiovisual assignment presets facilitated powerful high-level interactions during the performances. Additionally, utilising the musical phrase data provided an ability to achieve dramatic synchronised changes to the visual aesthetic. In doing so, the visuals evolved from merely reacting to the instrument’s volume data (as is the case with popular media player visualisers) to achieving a more convincing perceivable relationship with the musical composition itself. The juxtaposition of visual granular synthesis alongside the visual shader synths enabled a wider visual aesthetic and proved to be a useful performance tool for exaggerating sudden musical changes at pivotal moments during the set.

What Didn’t Work

The harsh natural environment, along with the different technical setups each stage provided, meant our performances at Burning Man were very challenging. The first stage setup (depicted in Figure 36) featured multiple non-rectangular small projection surfaces. Because I became aware of this only a few hours before the show, I had to re-program the framework to enable multiple visual outputs for the show later that night.
In contrast, the stage for our second performance (seen below in Figures 37 and 38) featured an enormous 70 m wide by 30 m high concave projection arena. Unfortunately, the HDMI input for the projections could only be reached at the front of house while the audio input was on top of the projection wall. This meant my colleague and I were separated during the performance by a distance of roughly 70 m. As a result, not only were we unable to communicate during the performance, but more significantly, the OSC messages we relied on for audiovisual synchronisation could not be sent over an ethernet connection. Although this was not ideal, we created an ad-hoc network connection that was capable of sending 90% of the OSC data wirelessly from his laptop to mine, enabling a convincing synchronisation to occur. Please refer to the ‘MindBuffer Burning Man.mov’ video in my portfolio to witness an excerpt from this performance.
5.5 Iteration 4 – UFO Club (Sydney) 2014

We were invited back by the UFO Club to perform our live audiovisual set in Sydney in January 2014. The performance was our second night in a row opening for the London musician Vaetxh on his East Coast tour. The promoters told us that they were going to construct three large projection surfaces, engulfing the performers and audience in a “cave of screens”. As a result, I decided to use this
performance as an opportunity to debut not only the triple-screen capabilities of my research project instrument Kortex but to debut my new powerful mini-ITX desktop computer.

5.5.1 Performance Strategy

Up until this iteration, I had been developing Kortex as a software environment separate to the one I had been developing for the live MindBuffer performances. Although the two shared similarities, such as their visual post-processing effects, the live MindBuffer software was responsible for creating audio reactive visual shader synths, while Kortex’s sole purpose was the creation of AVGS. The previous iteration proved that visual granular synthesis and visual shader synths could work harmoniously together. Consequently, I upgraded Kortex’s functionality by merging in the previous iteration’s visual shader synth framework.

Several new technological and aesthetic features became possible as a result of merging the visual shader synths into Kortex. Kortex was already capable of rendering to three unique visual outputs. This meant I was able to now render the visual synths in either monophonic mode (one synth stretched over three screens) or in polyphonic mode (three unique synths per screen). At the same time, the huge performance advantage my modern desktop computer had over my 2011 MacBook Pro laptop allowed me to upgrade the resolution from a single 512x512 render canvas to three 1024x1024 render canvasses.

For this iteration, I reprogrammed all of the pre-set triggering and visual feedback for the MIDI controller to reach my goal of creating a unified environment. Consequently I was able to remove the dependency on Max/MSP and simplify the process of sending and receiving controller data with Kortex. As a result, I developed a feature called PadModes to expand the functionality of the middle bank of pads on the controller. PadModes (see Figure 39 below) was developed to address my desire to rhythmically interact with the music and visuals during a performance. In doing so, I was able to rectify the performance flaw that became evident after reflecting on the performance of Iteration 2.
Figure 39: PadMode MIDI controller mappings to enable rhythmic triggering functionality

5.5.2 Reflection

What Worked
The opportunity to output audio reactive visuals over multiple screens, along with doubling the visual resolution, contributed towards the audience experiencing a high degree of sensory immersion. Throughout the set, I switched between monovisual and polyvisual output rendering modes. For example, at times I had three unique visual shader synths reacting to the music during a high-energy musical phrase. When I knew a breakdown was approaching at the end of the musical phrase, I
assigned the phrase OSC trigger to instantly switch the rendering mode to immerse the audience within a single widescreen visual shader synth. The contrast in visual perception matched the contrast in musical direction, further reinforcing the perceived integration between the music and visuals.

From a performance standpoint, my familiarity with the controller mappings, the musical set and the inclusion of PadMode functionality enabled me to interact with Kortex just as any musician interacts with their instrument. I was able to approach manipulating the visuals in much the same way a drummer improvises rhythmically in a musical performance. At certain times during the performance I was able to switch between creating different visual polyrhythms in relation to the rhythmical activity of the composition. For example, when the music contained fast rhythmical phrases I was able to trigger visual elements using a half-time feel, and then resynchronise with the original tempo during the last bar of the phrase. This approach felt very natural to me as a musician even though the media I was improvising with was visual. The rhythmical interplay between the musical and visual media seemed to enhance and reinforce the stylistic and emotional intent of the compositions.

**What Didn’t Work**

Despite the many positive audiovisual synchronisation enhancements that occurred during the four iterations, I was still not completely convinced the visuals were a true representation of the musical compositions. Even though the volume data from the four music stems and the phrase data were producing pleasing results, I was not satisfied knowing that an even larger amount of micro and macro musical events were occurring in the track and could be used for visual synchronisation. For example, a wide range of rhythmical visual aesthetics is not possible using only the summed volume of the entire drum track. Far more interesting results could emerge by sending unique volume data for the kick drum, snare drum and hi-hats individually. In doing so, the visual aesthetics could be more intricately connected and reinforce the audience’s perception of the audiovisual relationships.

Finally, for this performance I was unable to manually select the polyvisual output to contain a combination of both visual granular synthesis and visual shader synths simultaneously. I needed to address this in order to heighten the technique’s ability to induce a feeling of sensory immersion for the final iteration.
5.6 Introduction to the Final Iteration – Examination (Melbourne) 2014

My examination performance will be held in Melbourne during December 2014 and will be the final iteration to take place within the time frame of my candidature. The final technological design and compositional practice iteration evolved from a four-year cyclical process of action and reflection regarding the real-time performance and creation of audiovisual art. The performance strategy outlined below addresses the required enhancements that became evident during reflection on the previous iterations. Furthermore, it describes the final performance framework developed to achieve the goals and aims of my research.

5.6.1 Performance Strategy

The musical compositions for the previous four iterations were first composed in Logic and Ableton Live and then arranged in Ableton to achieve smooth transitions from one song to the next during a performance. This approach guaranteed the ideal compositional arrangement of our music and ensured a steady momentum during the one-hour performances. However, two undesirable factors were inherent within this approach. Firstly, the personal satisfaction gained from performing the set decayed exponentially with each performance; this was especially so during our three sets at Burning Man within the space of four days. Secondly, the approach used in the previous iterations restricted the range of musical information that could be analysed by Max4Live and used for visual synchronisation; the micro and macro compositional structure, and each instrumental voice’s digital signal processing data, were unable to synchronise with the visual imagery. To alleviate the repetition of replaying the same set, and to facilitate complete integration of sound and image at the micro and macro compositional level, I needed to implement a dramatic shift in approach.

This was achieved through using a custom generative musical composition environment created by Nordine (2014) in C++ using openFrameworks. Named Jennifer Max, it was affectionately referred to simply as Jen. Jen can be used to create an infinite range of musical compositions based on high-level user directives. The user specifies musical parameters such as intricacy, complexity, time signature and tempo that are then used to generate a unique musical composition. As everything is being synthesised and generated in real-time, there is no need to prepare stems or .wav files, eradicating any of the previous iteration’s reliance on external commercial software packages. Most importantly, to enable audiovisual synchronisation, Jen sends the entire hierarchy of track information ranging from bar subdivisions to overall song position as well as the volume, pitch and position information for each musical voice via OSC.

In order to synchronise Kortex with Jen, I developed a separate openFrameworks add-on called JenOSC (see section 6.3.5 for a detailed technical description). In short, JenOSC is a modular patching
environment for creating relationships between incoming music data from Jen and specific visual aesthetics in Kortex. Combinations of user defined audiovisual relationships are then stored into a bank of presets that can be recalled during a performance. The central philosophy of this approach is that every single musical event on both the micro and macro scale has a direct potential influence over a visual aesthetic. Consequently this approach redefines my performance role from the position of a single instrumentalist to a higher (macro) level, similar to that of an orchestral conductor. More specifically, my performance role is to create the ideal conditions throughout the performance for audiovisual synchresis to occur.

The above approach will be used for the first time to synchronise AVGS with musical events. Although previous iterations partially implemented visual granular synthesis, this iteration will debut the accompanying audio of the AVGS technique developed for my research. In doing so, the AVGS synchronised with Jen will take on the foreground performance role, similar to that of the soloist in a band. Synchronising the AVGS technique with meaningful musical events provides the necessary context that had been missing up until this iteration. Without musical context, the audiovisual aesthetics slide in and out of timeless states and, although interesting results emerge which would suffice for an interactive installation, alone it is not enough to hold an audience’s attention for the duration of a one-hour performance. Clearly this is a result of the specific context in which AVGS is applied. In a sit-down performance the performer is required to hold the audience’s attention. However, if AVGS was used for an installation the audience could decide how long they want to engage with the content.

The novelty of experiencing AVGS in the context of musical events will decay in the audience unless contrasting aesthetics are provided. To overcome this, AVGS will have time to feature in parallel with the visual shader synths and by utilising the triple projection surfaces. Interesting visual aesthetics will emerge as a result of the complex interplay between simultaneous visualisation of the two techniques. For example, both AVGS and the visual shader synths will have periods of mutual visibility and periods of exclusive visibility over the range of the three projection surfaces. In this way, through creatively assigning spatial combinations of monovisual and polyvisual imagery, the audience will be able to engage more broadly throughout the duration of the performance and experience a greater degree of sensory immersion.

As a result of the generative framework outlined above, this performance will be a unique unrepeatable experience much in the same way a jazz ensemble never replicates a performance. The framework facilitates the conditions for spontaneous musical and visual aesthetics to emerge from the software that will then influence new ideas in myself as the performer. This setup creates a tri-directional improvisatory feedback loop between the computer, myself and the audience (seen below
in Figure 40). Most significantly, the framework developed for the final iteration enables the creation and performance of audiovisual art in real-time.

![Tri-directional feedback loop of influence between the performer, instrument and audience](image)

**Figure 40: Tri-directional feedback loop of influence between the performer, instrument and audience**

### 5.7 Summary

In this chapter I discussed my action and reflection performance cycle through highlighting four national and international performances that took place during my candidature. For each performance iteration, I described the performance strategy I employed to evolve the Kortex framework and provide a critical self-reflection of the performance outcomes. Some of the most important highlights that emerged were:

- The tactile nature of MIDI controllers facilitate more intuitive control as a performance interface than touch screen interfaces
- Integrating audio and visual functionality within a single performance framework reduces technological risk and complexity
- Both micro and macro musical data are required to produce a convincing synchronous visual representation of a musical composition
- Synchronising AVGS with musical events frames the technique and provides necessary context
- Utilising contrasting audiovisual aesthetics helps to maintain the audience’s interest throughout a performance.
These reflections were used to refine subsequent software prototypes and the performance framework in order to meet the aims and objectives of my research. Finally, I provided an introductory description of the final performance iteration that will take place for my examination.
6 Instrument Design

This chapter describes the evolution and final design of the project prototype instrument named Kortex. Firstly, I outline how I adapted audio granular synthesis to visual media to enable AVGS. Secondly, I define the nature of the micro and macro audiovisual relationships involved in Kortex. Thirdly, I discuss the technology and design decision-making processes and performance strategies employed to facilitate powerful manipulations of the instrument during a performance. Finally, I reflect on the external interface and the mapping relationships between the controller and the Kortex prototype. Throughout I discuss and illustrate the design of the instrument, its performance potential and its innovative application in the field of audiovisual art, particularly in the performance work of DJs and VJs.

6.1 Instrument Design

Kortex is a real-time audiovisual instrument that enables a unified environment for composition and performance through AVGS. The instrument enables the user (or performer) to reconstruct grains into new audiovisual elements and makes possible real-time improvisation of the material at the granular level.

The design motivation behind the development of Kortex was to unify the roles of the DJ and VJ. I coined the term AVJ (audiovisual jockey) to describe the unified performance role. In an AVJ performance the audience experiences a tighter synchronisation and perceived relationship between audio and visuals.

Currently available commercial software is aimed exclusively at producers and VJs. To create audiovisual works, the artist is required to send control data from one application to another. This requires the artist to be knowledgeable about two different software environments and does not enable a unified audiovisual compositional environment. Kortex is designed to facilitate micro and macro level audiovisual relationships, thus enabling a unified audiovisual composition and performance tool.

Perceptual relationships created through my instrument are assigned to a single control that links audio and visual characteristics together. The instrument enables the user to compose audiovisual works via parameter automation. Through interacting with either the on-screen user interface or a physical interface, it can be used to create improvised material in real-time, much as a musician would approach any other musical instrument (see Figure 41).
DSP effects can be added and rearranged in an intuitive modular fashion to provide the user with greater expressive capabilities. To work with the software, content must be pre-prepared as standard file formats. Audio files are loaded as .wav files and visual content as QuickTime movie files.

The instrument is designed to run at 60 fps at a resolution of 1920x1200. To allow the software to perform at this frame rate, the internal framework allocates memory when loading content and de-allocates the memory once complete. This allows the audiovisual artist to perform with different content in a live situation, much in the same way a DJ or VJ evolves content throughout the duration of a performance.

The following section describes the technology that was developed to enable AVGS, the visual effects that are implemented in Kortex, and the polyphonic AVGS performance setup developed to enable audience immersion.

### 6.1.1 Programming Framework

I selected multimedia programming environments in preference to rendering software packages as it was essential that the prototypes were built to allow maximum performance at real-time frame rates. The programming environments of interest as candidates for the creation of the prototypes were Reaktor, Ableton Live, Super Collider and Kyma for audio synthesis; Quartz Composer (Apple
Computer, California) and VVVV (VVVV Group, Berlin) for visual synthesis; and Max/MSP, Processing and openFrameworks, which feature both audio and visual synthesis capabilities.

It is common practice for audiovisual artists to utilise separate environments for audio and visual synthesis while synchronising the two, either via the MIDI or OSC protocol. However, based on my own experience with these setups, I decided that a unified audiovisual environment would be preferable for the creation of the prototypes. This decision was based on factors such as minimising the complexity of setup for performance, easier integration of synchronising audio and visual parameters during development, and the speed of software execution without the unnecessary overheads of two different environments running side by side. It was also important to leave open the option to distribute the prototypes as self-contained compiled software, eliminating the requirement for the end user to obtain 3rd-party software.

Previously I used Max/MSP during my honours studies to create an audiovisual prototype but quickly reached a performance limit based on the complexity of the instrument. As a result, I narrowed down the environment I needed in order to create the project prototypes to either Processing or openFrameworks. Processing uses the Java programming language while openFrameworks uses C++. Both Processing and openFrameworks use high-level syntax and have the ability to utilise 3rd-party add-ons for importing extra functionality. Although Java is more user-friendly for an inexperienced programmer, the critical difference is in how programs are compiled. Java code when compiled is converted to byte code, whereas C++ converts to machine language, which is ultimately closer to the hardware and therefore leads to faster code execution. As performance was of ultimate concern, openFrameworks, with its foundation in C++, became the environment that was used for the creation of the software prototypes for this research project.

### 6.1.2 Audio Synthesis Libraries

Audio synthesis libraries are a set of signal processing and algorithmic synthesis classes. I needed to select appropriate audio synthesis libraries in order to facilitate granular and FM synthesis functionality within Kortex. The selection was narrowed to synthesis libraries coded using the C++ programming language and released as open source. Those synthesis libraries that contained well-maintained code and thorough application programming interface (API) documentation were evaluated as more appropriate. I narrowed the analysis down to four audio synthesis libraries: Synthesis ToolKit (STK), Gamma, Maximilian and Tonic. Each contained example code and documentation to assist in the analysis of the framework’s usability and sound quality.
Synthesis ToolKit

The STK is developed by the Center for Computer Research in Music and Acoustics and has been used in the development of audio processing software for approximately 15 years. The toolkit requires no external library support, making it highly flexible and portable between platforms. The STK comes with a wide variety of code examples based on its popularity for use in education and development. Because of this, I decided that I would analyse the granular synthesis and DSP processing capabilities of the STK first. Using the STK granular synthesis example, I hacked together a working prototype in openFrameworks that enabled me to tweak parameters using a GUI during run-time. I then experimented with combining the output from the granular synthesis with the ToolKits reverb, delay and flange DSP code. Although the resulting prototype contained the required granular synthesis and signal processing, I found the signal processing of the STK lacked the crisp level of sonic detail I was searching for.

Maximillian

The next framework I tested was a recently released audio synthesis library called Maximilian. Maximilian came packaged with a similar collection of signal processing functionality to that of the STK. The library contains a C++ wrapper that integrates with openFrameworks, making it easy to immediately start developing prototypes. Granular synthesis is implemented with multiple methods for utilising various functionalities such as time stretching and pitch shifting. Methods that handle sample loading, playback and looping are also well implemented, making it easy to develop the initial version of the granular synthesis engine for the prototypes. The granular implementation of Maximilian was superior to the STK with respect to the sound produced and the ease of integration within openFrameworks. For this reason I decided to use Maximilian to begin granular synthesis integration within the prototypes. I needed to make my own additions and customisations to the Maximilian code during development to synchronise the granular windowing envelopes with the visual grains. This involved converting local floating-point values to pointers that could be retrieved by reference higher up in the application.

Tonic

Tonic was released in 2013 as a C++ alternative to popular synthesis environments such as Max/MSP, SuperCollider and Pure Data. Tonic lacks sample playback and spectral granular processing, but features an impressive set of synthesis functionalities. I had been using the synthesis functionality of Maximilian for the development of my FM synthesis engine (described later in section 6.1.5), but decided to implement the Tonic-based one instead because of the quality of its sound oscillators. During tests, Tonic seemed to run more efficiently than Maximilian during run-time, making it an
even more attractive option for the FM synthesis component.

After a month of developing an early FM synthesis prototype using Tonic, I intended to start synchronising the audio oscillators with real-time graphics. However, I discovered there were no ‘getter’ methods to retrieve the current value of an oscillator. I contacted the primary developer of Tonic, Morgan Packard, to discuss this issue. He gave the following response to my inquiry:

Tonic by design doesn't output single values. It always operates on buffers of floats. This is much more efficient (though a little more complicated) than sample-at-a-time calculation, which is how Maximilian does it. Eventually, the audio hardware needs buffers of values, not individual values anyway. So somewhere those individual values being output by Maximilian are getting rolled in to a buffer. This stage might be the most logical spot to mix your signals. (Packard, 2013, pers.com)

Although I had developed a decent-sounding FM synthesis prototype using Tonic, I abandoned it and continued searching for an alternative synthesis library based on Morgan’s confirmation that I would be unable to retrieve the values from Tonic to use for tight graphics synchronisation.

**Gamma**

The final framework I tested was Gamma. Gamma is a lightweight generic synthesis library that excels in the spectral processing and filtering of audio signals. After downloading Gamma, the user is required to build the source using the ‘make’ command via terminal. Initially, I was unable to progress past this stage and subsequently contacted Lance Putnam, the developer of Gamma, for assistance to compile the source. With Lance’s help I was able to compile Gamma and to test and run the examples via the command line. Of the four synthesis libraries, Gamma was the hardest to integrate with openFrameworks. I was reluctant to use Gamma as the synthesis engine for the prototypes due to the amount of work involved in wrapping the library to be compatible with openFrameworks. Nevertheless, the sound quality of Gamma far surpassed that of Maximilian, STK or Tonic, and Gamma allows one to retrieve the current phase and frequency values of its oscillators. Because of its superior sound quality and ability to replace the missing value retrieval functionality of Tonic, I decided that Gamma would be the ideal library with which to build my FM synthesis engine, while Maximilian would be used to integrate granular synthesis functionality.

Table 3 below highlights the main strengths and weaknesses of each of the four audio synthesis libraries discussed above.
6.1.3 Technology

As previously noted, Kortex was developed on OSX using openFrameworks (2014). OpenFrameworks is a collection of functional components that help the user achieve specific tasks. It has components that are used to draw 2D and 3D graphics, display images and video, synthesise and play back audio, and interface with low-level APIs. Several open-source libraries were used for the various functionalities of the instrument. They included the Maximilian and Gamma audio synthesis libraries, ofxTimeline for timeline parameter automation, ofxUI for GUI support, and ofxMidi for MIDI controller interfacing.

### Video Frame Loading

During the initial Kortex prototype development stage, I investigated various libraries, APIs and techniques that would allow for the fastest speed of code execution. I sensed it was critical that visual granular synthesis would require the fastest and most efficient methods for selecting, processing and displaying video frames. The first Kortex prototype used the QuickTime API for the loading and playback of video files. Random access functions would then set the current frame of video that should be displayed. Video footage was encoded separately with the h.264, photo-jpeg, animation and mpeg-4 video codecs for testing purposes. Regardless of the video codec used, the QuickTime API’s update speed was very slow when applying random access functions. I suspected that this poor performance occurred for two reasons. Firstly, the QuickTime API is very large and contains a lot of functionality that was not needed for my purposes. Secondly, the video player was accessing the video by referencing its location stored on the computer’s hard drive. It seemed that the computer was not able to return a reference to the various frames stored within the video fast enough per frame without noticeable update delays occurring.

Real-time frame rates, however, are extremely important. Preliminary measurements using an fps counter to assess the speed of the QuickTime API were not convincing in a real-time context and failed to provide a convincing perceivable link between the access speeds of visual frames compared to that of the audio. Frame rates would fluctuate between 20-35 fps, well below the steady real-time...
performance of 60 fps needed for visual granular synthesis.

To overcome this issue, I developed a way of working with frames of video in order to allow instantaneous random access to frames stored as textures in RAM. The algorithm uploads all the frames of video in RAM then discards the original video file. A computer with 16GB of RAM allows the user to upload video content with a maximum total duration of 40 seconds. The basic algorithm behind the storing of video textures is as follows:

1. Create a pointer to a temporary QuickTime video player object.
   
   ```cpp
   video = new ofVideoPlayer();
   ```

2. Load a movie file into the video player object.
   
   ```cpp
   video->loadMovie(fileName);
   ```

3. Get the total number of frames of the video and store the result in a temporary variable.
   
   ```cpp
   int totalFrames = video->getTotalNumFrames();
   ```

4. Create a variable ‘i’ that counts upwards from 0 until it equals the total number of frames.
   
   ```cpp
   for (int i=0; i<totalFrames; i++){
   ```

5. Assign the current value of ‘i’ divided by the total number of frames to a variable named progress. This formula remaps the duration into a percentage between 0 and 1.
   
   ```cpp
   progress = i / (float) ( totalFrames - 1);
   ```

6. Set the play position of the video object based on the current value of the progress variable.
   
   ```cpp
   video->setPosition( progress );
   ```

7. Create a pointer to a new openGL texture buffer for each frame.
   
   ```cpp
   ofTexture* tex;
   tex = new ofTexture();
   ```

8. Allocate the appropriate amount of memory on the graphics card to hold the texture buffer based on the resolution of the video.
   
   ```cpp
   tex->allocate( video->getWidth(), video->getHeight(), GL_RGB );
   ```

9. Fill the texture buffer with the pixels of the current frame in the video object.
   
   ```cpp
   tex->loadData( video->getPixels(), video->getWidth(), video->getHeight(), GL_RGB );
   ```

10. Push the stored texture buffer containing a frame of video into a vector container.
    
    ```cpp
    frames.push_back ( tex );
    ```

11. Iterate over points 5 to 10 until each frame in our video object has been transferred into an openGL texture buffer object stored in memory on the GPU.
    
    ```cpp
    for (int i=0; i<totalFrames; i++){
        progress = i / (float) ( totalFrames - 1);
        video->setPosition( progress );
        ofTexture* tex;
        tex = new ofTexture();
        tex->allocate(video->getWidth(), video->getHeight(), GL_RGB );
    ```
Once all frames in the movie have been converted to textures and stored in memory, the QuickTime API is deleted. From here, in order to use our stored textures, we need another algorithm to take care of calling and displaying the textures stored in memory 60 times a second.

**Granular Audio Video Synchronisation**

The next challenge was developing an algorithm that would synchronise the playback of video frames stored on the GPU with the granular audio. The algorithm I developed enabled the user, via interaction with a GUI, to set the start position, loop size and the speed at which the playhead would move through the loop. At this stage I got my first glimpse into the potential for achieving AVGS, and could move my attention to implementing other higher-level relationships between audio and visual processors. The algorithm for calculating which texture should be displayed is as follows:

1. Define the audio buffer size, sample rate, frame rate of original video and the frame rate of our application in the setup function.
2. Pass in the current value of start frame, loop size and speed defined by the user.
3. Set the speed for texture playback.
4. Check if speed is greater than 0.
5. Add the value of the jitter variable to the current frame for time scrambling.
6. Define the loop size by assigning the value of start frame + loop size to the end frame variable.
7. Check if end frame is greater than the total number of frames in the video.
8. Check if the current frame is greater than the total number of frames, if true….
9. Set current frame to wrap back to the beginning by subtracting the total number of frames.

```cpp
frames.push_back( tex );
}
12. Delete the QuickTime video player object from memory.
    delete video;
```
10. Check if the current frame is smaller than the start frame, if true….
   
   ```c
   if (currentFrame < startFrame)
   ```

11. Set end frame to wrap back around to the beginning by subtracting the total number of frames.

   ```c
   endFrame = endFrame - totalFrames;
   ```

12. Enable looping to occur once frames have been wrapped back to the beginning.

13. Else if the current frame and end frame do not exceed the total number of frames no warping needs to occur. Steps 7 – 11 are skipped.

14. Set current frame to the value of start frame if current frame is greater than end frame.

   ```c
   if (currentFrame > endFrame) {
       currentFrame = startFrame;
   }
   ```

15. Set current frame to the value of start frame if current frame is less than start frame.

   ```c
   if (currentFrame < startFrame) {
       currentFrame = startFrame;
   }
   ```

16. For safety, clamp the value of current frame to never exceed 0 and total number of frames.

   ```c
   currentFrame = ofClamp(currentFrame, 0, totalFrames);
   ```

17. Remap the value to a percentage between 0 and 1 by dividing current frame by the total number of frames. Assign result to the play position variable.

   ```c
   playPosition = currentFrame / totalFrames;
   ```

18. Return the value of play position

   ```c
   return playPosition;
   ```

19. Continue to iterate over steps 2 – 18 until program terminates.

The value returned by the above algorithm is then used to display the correct frame stored in our texture buffer and sets the play position of the granular audio. This algorithm is critical for synchronising the audio and video, and thus enabling AVGS.

In Kortex, the textures can be accessed and displayed in any order by calling the specific memory address of the individual texture stored in RAM. This technique consistently produces real-time frame rates of 60 fps as seen below in Figure 42. Please refer to ‘AVGS Algorithm Comparison.mov’ in my portfolio to witness a side-by-side comparison between the QuickTime API and the above custom algorithm.
I modified both my audio buffer and texture handling algorithm to a ring buffer technique¹⁹ (seen below in Figure 43) to enable the potential for live input (such as webcam, microphone data or live television) as well as real-time computer-generated content to be manipulated instantly using AVGS. Within the ring buffer function I was still able to use the grain reordering functions as well as set loop position, speed and direction parameters. Consequently, the ability to utilise real-time generated graphics, or any other live audiovisual stream, significantly expanded the possible applications for the AVGS technique.

¹⁹ The ring buffer, otherwise referred to as the circular buffer, refers to an area in memory of fixed length that is used to store incoming data. When the memory buffer is filled new data is written at the beginning of the buffer, overwriting the old. (http://www.boost.org/doc/libs/1_54_0/libs/circular_buffer/doc/circular_buffer.html)
Figure 43: An allocated section of memory in a circular ring and how data is inserted, read and deleted from the memory

6.1.4 Visual Shader Synths

Along with AVGS, Kortex supports manipulating a collection of visual shader synths in real-time. A visual shader synth is a fragment shader program created using the GLSL programming language. The main benefit stems from GLSL programs being executed in parallel on the graphics card. This results in creating highly detailed 3D geometric scenes in real-time without the need for off-line rendering. Once a visual shader synth has been triggered, the visual scene is sent through the post-processing effects pipeline for further deconstruction.

A visual shader synth can be displayed on its own or used as a texture to burn into the AVGS layer. Figure 44 below shows a woman’s face manipulated using AVGS burning into a visual shader synth, creating a 3D tunnel environment. A wide range of visual aesthetics can be produced using the technique of burning the AVGS and visual shader synth layers into each other. This is especially true when both layers are reacting to unique musical parameters or control strategies. In future research, I intend to extend this technique by merging AVGS into the 3D environment.
Please refer to ‘MindBuffer – CODE – The Grid.mov’ to witness how visual shader synths react to incoming audio data during a performance.

Figure 44: Screenshot of AVGS burning into the visual shader synth layer rendering in the background

6.1.5 FM Synthesis Engine

Kortex features a custom FM synthesis engine that is capable of creating a wide spectrum of sonic timbres. The FM synthesis engine seen below in Figure 45 was developed to facilitate the potential for creating audiovisual relationships in conjunction with the visual shader synths. From a design perspective the internal framework was inspired by traditional FM synthesisers such as Native Instruments FM8 (2014), however the sonic characteristics it produces are unique. Technically, the engine comprises carrier and modulator oscillators, switchable low pass, band pass and high pass filters and three low frequency oscillators (LFO) that can be assigned as modulation sources for any of the synthesis parameters. The user can choose between sine, triangle, square, ramp or sawtooth waveforms for both the carrier and modulator oscillators and the three LFO oscillators.
At the heart of the engine is the modulation routing interface seen below in Figure 46. The interface enables the user to route each of the five synthesis parameters (pitch, modulation frequency, modulation index, filter cutoff and amplitude modulation) to a modulation source that will drive its behaviour. The five modulation targets are bypass, LFO1, LFO2, LFO3 and AVGS. If the modulation source is set to bypass then no modulation occurs and the user drives the parameter manually. If the modulation source is set to one of the three LFO targets then the parameter will behave according to the oscillator shape, speed and depth of the specific LFO. The last modulation target is AVGS, which drives the synthesis parameter according to the grain reordering patterns described in section 4.3.1.
The five synthesis parameters described above can be assigned to a visual shader synth parameter to create audiovisual relationships. In this way, visual parameters directly respond to the state and behaviour of linked audio synthesis parameters. The modulation targets thus create time-varying rhythmical patterns for collections of interlinked audiovisual relationships. Consequently this enables a direct visual representation of the FM synthesis engine.

### 6.1.6 Visual Effects

Kortex contains several post-processing visual effects aside from the micro and macro audiovisual relationships that have been discussed already. The visual processing FX (effects) described in this section were included to enable further image deconstruction during a performance. They can be used to exaggerate musical cues and phrases or to construct abstract visual spaces that can evolve during an improvisation. At present, the visual FX listed below feature no auditory processing analogue. My future research will apply a method for adapting audio processing techniques to visual media outlined in this exegesis but in reverse – adapting the visual processing techniques to audio DSP effects. Section 8.3 provides a discussion of some limitations relating to implementing the slit scanning, optical flow and triangulation effects in real-time.
Please refer to ‘Triangulation Image Deconstruction.mov’ in my portfolio to see a combination of the effects described in this section, ‘AVGS’ Memory Corruption Effect.mov’ to observe the result of the deconstruction effects applied to AVGS, and to ‘MindBuffer – CODE – PanFM.mov’ to witness an example of how they are used in live MindBuffer performances.

**Slit Scanning**

Slit scanning is a time displacement and video delay method that produces a wide variety of time-based visual effects. It works by distorting the image by shifting pixels across time according to a displacement map. Figure 47 shows the selectable displacement maps in Kortex, while Figure 48 shows slit scanning being applied in Kortex using a square pixel displacement map.

![Figure 47: Grey-scale time displacement maps used by the slit scanning effect](image)

![Figure 48: Screenshot of slit scanning being applied to Kortex using a square pixel displacement map](image)
Triangulation

The triangulation effect generates a mosaic of triangular tiling based on the image. In order to generate triangles, the image is uploaded from the GPU to the CPU where the openCV library analyses the image using blob and feature detection. Triangles are then generated based on the result of the analysis. The colour of each triangle is derived from the pixel colour of the original image at that location. The controllable variables are density of triangles, shape contour, image threshold, fill opacity and wire frame opacity. Figure 49 shows the original triangulation prototype; while Figure 50 shows the triangulation algorithm implemented in Kortex in its current stage.

Figure 49: Original triangulation algorithm. Triangle size is randomly generated
Optical Flow

The optical flow effect draws a directional vector based on the difference and direction of movement between consecutive frames. The optical flow technique requires that images are uploaded from the GPU to the CPU for analysis using the openCV library. If an object passes from right to left then the optical flow effect will draw coloured lines (based on the speed and direction of movement) over the moving object. This can be seen in Figure 51, in which a black and white video is used to highlight the optical flow effect that is drawn in colour.
Figure 51: The movement of the dancers ribbon triggers the optical flow

Box2d Physics
The Box2d effect works by packing the canvas with randomly sized rectangles. The rectangles’ colour is generated in the same way the triangulation algorithm generates colours for the triangles. The Box2d physics library was implemented to achieve collision detection between the rectangles. The user can select between primitive shapes such as rectangles, triangles and circles or combinations of the three. Because Box2d is a complete physics library, interesting image deconstruction effects can be achieved by manipulating gravity, force and friction parameters. Figure 52 shows the Box2d effect in a resting state.
Figure 52: Randomly distributed rectangles provide a physics pixilation effect to the image underneath

**Pixel Physics**

The pixel physics effect transforms the image’s pixels into visual particles. The particles react to forces found in physics such as force, force radius, attraction, repulsion, friction and spring. Particles can be drawn as points, lines, line strips, triangle fans or quads. Image explosion type effects, shown below in Figure 53, are achieved according to the position and radius of the force to be applied. This is controlled in Kortex by analysing the audio. A random force position, force radius, friction and spring value is generated each time the audio volume exceeds a certain threshold.
6.1.7 Triple Screen AVGS

Kortex can be compiled to run on a single screen or as a triple-screen performance setup. The motivation for expanding Kortex to use three independent AVGS players was to maximise the feeling of immersion in the audience. During development I used three 27-inch monitors running at a resolution of 1920x1200, as seen below in Figure 54. The final performance (in December 2014) will output at the same resolution but will utilise three powerful projectors to display the image on large projection screens.

The process of porting Kortex to run on a triple-screen setup enabled polyphonic audiovisual granular synthesis. Each player features its own independent AVGS engine, post-processing and DSP and FX controls. A side effect of this expansion was the number of control variables in Kortex tripled.
Intelligent performance strategies (described in section 6.3) were required to enable high-level control over the instrument’s parameters.

Please refer to ‘Kortex Triple Screen.mov’ in my portfolio to witness a demo of Kortex running three AVGS players simultaneously, and to ‘Kortex Single Screen.mov’ to compare the output of Kortex when utilising a single AVGS player.

6.2 Audiovisual Relationships

Users of Kortex can form audiovisual relationships through interacting with and observing perceptual similarities between sound and image at the micro and macro level. The micro level addresses granular particle relationships between sound and image, while the macro level relates to relationships between DSP audio and visual effects. These relationships provide the user with ‘unified parameters’ for audiovisual manipulation.

At each of the micro and macro stages, perceivable characteristics for sound and image are identified. That is, each artist-user of Kortex perceives audio and visual characteristics through their visual and auditory senses. Once a set of perceivable audio and visual characteristics are identified, similarities between the characteristics of the two media at that level are connected to form a relationship. This exegesis suggests forming relationships between similar sonic and visual characteristics, although an artist may choose to form relationships from dissimilar characteristics. An example of a similar relationship would be mapping a sound’s bit resolution to an image’s pixel resolution. An example of a dissimilar relationship would be mapping a static effect, such as the image’s brightness, to a time-based effect, such as a sound’s delay time. The juxtaposition of the interlinked sonic and visual characteristics provides a mutual enhancing effect.

The following section describes the AVGS’ relationships as well as the visual post-processing and auditory DSP’s relationships.

6.2.1 AVGS Relationships

The audiovisual relationships between the main parameters of granular synthesis are associated with the audiovisual cloud’s start time and duration, the jitter and jitter quantise, the play head speed of the cloud, the density of grains overlapping within the cloud, and the pitch, duration and grain envelope of individual audiovisual grains within the cloud. The start time, duration and speed parameters also define the behaviour of the grain reordering patterns outlined in section 4.3.1.
In order to demonstrate the below relationships, I have included a basic demo version of Kortex on the USB stick. Please refer to the appendix for instructions on how to operate the software or refer to ‘AVGS Demo Software.mov’ in my portfolio to witness a screen recording of the demo software in action.

**Start Time and Duration**

The start time and duration parameters set the boundaries of the audiovisual cloud, and are defined in video frames. Duration is limited to the specific length of the loaded file. Setting the duration to one frame allows the user to scrub through the file in any direction using the Start Time parameter. Setting the duration between two and 15 frames creates a looping segment between four and 30 times a second, producing stuttering machinegun-like effects.

**Jitter and Jitter Quantise**

The Jitter parameter introduces a percentage of randomness to the play head position within the upper and lower boundaries of the audiovisual cloud. The parameter’s value range is mapped to a value between 0.0 and 1.0. With a setting of 0.25, the play head moves randomly within 25% of its current position inside the cloud. Jitter Quantise parameter sets the rate at which the randomness amount set by the Jitter parameter is applied to the play head position. These two parameters allow the user to scatter or time-scramble the otherwise linear playback of individual grains.

**Speed**

The speed parameter sets the rate at which the play head moves through boundaries of the audiovisual cloud. The parameter’s value range is mapped to a scale from 0.0 to 1.0. A value of 1.0 will play the file back at its original speed; a value of 0.0 will freeze the play head position, while a negative value will reverse the direction of the play head. The speed may be set to play up to 8x faster than the rate of the original file in both directions. This limitation is imposed on the speed parameter to maximise the slider resolution.

**Pitch**

The Pitch parameter is scaled from 0.0 to 8.0 and sets the pitch of the each individual audio grain. The quality of a low sonic pitch is muddy and dark, while a high sonic pitch is bright, shimmering and brilliant. Because of the effect this parameter has on the sonic qualities, Pitch is mapped to a bloom saturation visual effect for the visual component of the audiovisual grain. The parameter’s value range is mapped to a percentage. With a setting of 2.0, the grains pitch is one octave higher and the bright
areas of the visual grain are intensified. This parameter may also have a percentage of randomness applied. Pitch randomness is calculated and applied per audiovisual grain.

**Grain Duration**

Grain duration is defined in seconds; thus a value of 0.1 sec equals a grain length of 100 ms. The minimum size of an audiovisual grain is 16.66 ms, while the maximum size is capped at 500 ms. These constraints on grain duration were imposed as the maximum duration allocated memory can hold per grain is 500 ms, and so the minimum value equals the time duration of one frame at 60 fps. This parameter has a strong influence on the perceived timbre and luminosity of the audiovisual cloud.

Grain duration sets the length of the grain’s envelope function. Randomness may also be applied to scatter the duration of each grain. Low amounts of randomness applied to grain duration portray very digital, unnatural qualities, while higher values mask this effect, leading to a more organic quality. For example, in nature it is common for clusters of organic sound events, such as the crackling of a fire, to have varying lengths of duration. In contrast, sound events that feature evenly distributed durations can only be produced via digital means.

**Density**

Density sets the polyphony or the amount of audiovisual grains overlapping at any one time. A density of one grain triggers the audiovisual grain envelope sequentially. At this density, single audiovisual grains are triggered one after the other allowing for visual stroboscopic and auditory vibrato effects. At density values other than one grain, the summed values of the grain envelopes overlap, allowing for a more continuous perception of audiovisual grains volume and contrast. Higher density values also contribute to the fullness and complexity of the audiovisual cloud when grain duration and pitch are set to generate random values.

**Grain Envelope**

The grain envelope determines the waveform shape applied to each audiovisual grain. Each waveform imposes various perceptual characteristics on all other the audiovisual grain parameters. The envelope size is set by the grain duration parameter. The shape of the waveform used controls the auditory volume and visual contrast of each grain. When the value of the envelope is at zero, the grain’s volume is silent and it sets the contrast of the image to black. The volume and contrast of the audiovisual grain are directly proportional to the current value of the grain envelope’s waveform.

The waveforms that can be selected are Gaussian, cosine, triangle, rectangle and Blackman-Harris. See Figure 55 below.
6.2.2 Post Processing Visual and DSP Relationships

In addition to micro relationships, Kortex implements macro-level relationships between audio and visual effects. I analysed several post-processing effects for both audio and visuals in order to find perceivable similarities between their characteristics. To establish these similarities, I followed Olofsson’s (2009) technique for creating audiovisual correlations described in section 4.3.2, as follows. Blurring an image and inserting a reverb on a sound established one particular relationship. Blurring smears the pixels of the image by offsetting multiple copies of itself, just as reverb smears the sound through layering multiple offset copies. The blur and reverb seemed to have a similar effect of washing and un-focusing away the clarity of the sound and image.

A delay exists in the temporal time domain. The feedback parameter defines how many audiovisual copies are layered back, while the time parameter sets the duration between the copies. A similarity between the auditory down-sampling and visual pixilation effect was also observed. Down-sampling reduces the frequency resolution of a sound, while pixilation reduces the resolution of an image. I paired this association in the software, allowing the user to reduce the audiovisual resolution from a single parameter.

Table 4 shows the default relationships that I determined between the micro and macro audiovisual characteristics of the instrument. These relationships were chosen to reflect the subjective similarity I found between their processing characteristics. For example, an audio processing time-based effect, such as delay or flange, is associated with corresponding time-based visual effects. These recommended default relationships can be redefined by the user to reflect their own interpretations. Please refer to ‘Kortex AV Relationships.mov’ in my portfolio for an audiovisual demonstration of these relationships.

Figure 55: Grain windowing shapes (a) Gaussian, (b) Cosine, (c) Triangle, (d) Rectangle, (e) Blackman-Harris
Audiovisual Relationships

<table>
<thead>
<tr>
<th>Audio</th>
<th>Visual</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio Play Position</td>
<td>Visual Play Position</td>
</tr>
<tr>
<td>Amplitude</td>
<td>Contrast</td>
</tr>
<tr>
<td>Pitch</td>
<td>Bloom Saturation</td>
</tr>
<tr>
<td>Downsampling</td>
<td>Pixelation</td>
</tr>
<tr>
<td>Reverb</td>
<td>Blur</td>
</tr>
<tr>
<td>Audio Delay Time + Feedback</td>
<td>Frame Delay Time + Feedback</td>
</tr>
<tr>
<td>Filter Cutoff</td>
<td>Contrast</td>
</tr>
<tr>
<td>Flange</td>
<td>Texture Offset</td>
</tr>
</tbody>
</table>

Table 4: Default audiovisual relationships in Kortex

Each auditory and visual parameter usually has upper and lower bounds within which it moves. In MIDI terms, that parameter is within the positive integer range of 0 to 127. To achieve a higher precision range of values than MIDI, any parameter can easily be re-mapped as a scale value to output between 0.0 and 1.0. By exposing/publishing the value of the current state of each parameter coming from a visual or auditory control, audiovisual relationships can then be assigned by the user in the software in real-time. These mappings can then be saved and stored by the software as presets that the user can recall during a performance (see Figure 56).

Figure 56: Modular assignment interface to define audiovisual relationships
6.3 Performance Strategies

The huge number of variables involved with granular synthesis requires interfaces that facilitate powerful control strategies to offer the user elaborate transformations during a performance (Truax, 1990). The following section describes how simple harmonic motion, presets, tweening, timeline automation and JenOSC were implemented in Kortex to facilitate powerful transformations.

6.3.1 Simple Harmonic Motion

The technique of harmonic motion was initially implemented to overcome the limiting number of assignable sliders and knobs on the controller. Although harmonic motion was implemented to overcome a limitation, the technique enabled an even greater sphere of creative possibilities than I originally envisaged. In essence, the harmonic motion function gives the user control over three AVGS objects from a single controller, enabling powerful manipulations to occur during a performance.

Simple harmonic motion works by utilising the sine function to enable the periodic movement between two points, known as oscillation. The output of a sine function generates a smooth curve between the values $-1.0$ and $1.0$. By re-scaling the values, the sine function can be assigned to control the behaviour of either individual variables or collections of variables. This behaviour is known as simple harmonic motion or “the periodic sinusoidal oscillation of an object” (Shiffman, 2012, p. 140).

Shiffman stated that simple harmonic motion can be expressed as a function of time with the following two elements: amplitude – the distance from the centre of motion to either extreme, and period – the amount of time it takes to complete one full cycle of motion. I implemented a harmonic motion function calculated using four variables: position, amplitude, period and offset.

- Position defines the centre point of oscillation and sets the same value for each player if amplitude is at zero.
- Amplitude refers to the range of oscillation from the centre position and allows the players to fall out of phase with each other.
- Period relates to the time the value takes to complete one full cycle of motion.
- Offset adds small increments to the period of oscillation for each player, allowing the players’ values to phase in and out of sync with each other.
Figure 57 shows harmonic motion having no effect over the start position of the three players with amplitude, period and offset set to 0. Figure 58 shows how the start position of each player will begin to go out of phase with each other as amplitude is set to greater than 0.

Figure 57: Position slider sets the same play position for each player

Figure 58: Play position of each player out of phase with each other

Setting the period and offset to 0 while the values of object 1 and object 3 are in phase with each other allows the relative phase to become locked. In this state, the position variable is able to control the state of two independent values, as shown in Figure 59.
Figure 59: The phases of players 1 and 3 are locked with offset and period set to 0

The minimum and maximum value returned from the function is clamped to never go below 0.0 and to never exceed 1.0. I restricted the values to this range so the function does not have a chance to exceed the boundaries of the parameter assigned to it. This has a similar effect to how a limiter works in recording studios by never allowing a signal to exceed a specified range. As a result, if position is set at 0%, and with amplitude set to 50%, the harmonic motion’s range is between +50% and -50%. While the value is between 0% and +50% the motion behaves as expected. When the value is between 0% and -50% then the value returned is stuck at 0% until which time the motion becomes greater than 0%. This behaviour can be seen in the third image of Figure 60.

Setting the value of position close to its minimum or maximum range with a high enough amplitude changes the behaviour of the harmonic motion from smooth oscillation to mimic a bouncing behaviour. Figure 60 shows how harmonic motion changes from smooth oscillation to bouncing by moving the position value closer to its minimum value.
The control parameters start frame, loop size, speed, strobe rate, grain size, pitch/bloom, downsample/pixelate and reverb/blur are each routed through their own unique harmonic motion function. Each independent harmonic motion function can be manipulated during a performance through interfacing with the controller described in section 6.4. Please refer to ‘Simple Harmonic Motion.mov’ in my portfolio for a visual demo of simple harmonic motion controlling the frame position of three videos.

6.3.2 Presets
To perform with Kortex in an intuitive fashion, the user interface provides access to low-level individual control of grain and effects parameters, and high-level control of collections of parameters by performing with presets. The preset engine allows the user to save all the instrument’s variable settings as states; that is, the states relating to the positions of every control in the instrument. Because of the complexity of control levels in the instrument, triggering states during a performance allows the user to manipulate all controls at once, which would not otherwise be possible.

6.3.3 Tweens
To enable another level of expressive capability in the instrument, the user can choose to slide or ‘tween’ between one state and another. The word ‘tween’, derived from ‘between’, can be defined as an interpolation from one position to another (Penner, 2002). By default the shortest path between two points is a straight-line or linear tween. A linear tween has a constant velocity throughout without any acceleration. This lends itself to a mechanical or artificial movement.
Organic processes react to forces such as gravity, tension and friction. When these forces are not in balance, the result is acceleration and deceleration (Penner, 2002). To model this phenomenon, Kortex employs various easing types that facilitate more organic movement from one state to the next. Easing can be defined as acceleration in speed during a tween (Penner, 2012). The user can control the speed of the tween between states and also its easing type. The various easing types available to the user in Kortex are linear, exponential, circular, bounce and elastic (see Figure 61).

Figure 61: Available easing types for transitions between the instrument’s states

These easing shapes provide the user with flexible transformations during a performance. The exponential and circular shapes facilitate more organic changes than the linear shape, while the bounce and elastic shapes offer multi directional transformations. Rich visual and sonic results are achieved during a tween as the entire instrument is in a state of flux.

The pioneer of real-time granular synthesis, Barry Truax, implemented a similar instrument design by allowing the user access to low-level grain parameters, presets and tendency masks (tweens) with his GSX and GSAMX programs (Truax, 1988). During a performance the user was able to override stored parameters through intermingling spontaneous gestures with pre-planned functions (Roads, 2012).

6.3.4 Timeline

During the development of Kortex, there were occasions when I felt the urge to compose rather than improvise with AVGS. Creating compositional phrases and patterns required a method to enable parameter automation over time. A popular solution to parameter automation can be seen in commercial software such as After Effects (Adobe, California). After Effects allows users to control the value of variables by inputting ‘keyframes’ on a timeline. In 2011, no such timeline functionality for automating variables existed within the openFrameworks community. Instead, I took the approach of chaining Ableton, Max4Live, OSC and Kortex together as a temporary work-around hack to enable timeline functionality. This approach required a custom Max4Live device, shown in Figure 62, which contained the main AVGS controls of Kortex.
Each control was formatted to send the correct range of values expected by Kortex. The values were formatted further with a unique OSC address to then be sent over a local network into Kortex. Placing the Max4Live device on a spare channel allowed me to automate each parameter using the Ableton timeline. After making the necessary automations to specific parameters in the timeline, pressing play in Ableton would drive the behaviour of Kortex in real-time. Figure 63 below shows an Ableton Live project that contains four automation tracks to independently control the start frame, loop size, LFO speed and jitter parameters in Kortex. Each track contains a separate Kortex Max4Live device.
I pursued this compositional approach to parameter automation using Ableton Live from April 2011 – February 2012, but became frustrated with the fact the timeline was not directly accessible and integrated from within Kortex. However, a few months after creating the above approach, James George released an openFrameworks timeline alternative for parameter automation called ofxTimeline. Integrating ofxTimeline enabled Kortex to become a self-contained improvisational and compositional environment. Each parameter in Kortex contains a toggle switch that selects if the parameter is controlled by the GUI or automated using the timeline feature. For example, this allows the user to automate a sequence of start positions and loop sizes while improvising with the speed and FX controls.

The user can insert tweens using the timeline to facilitate non-linear automation of parameters. The available transition shapes using timeline automation can be seen in Figure 64.
Figure 64: Available tweening curves for key frame automation within Kortex

Collections of automated parameters can be saved out into banks that can be recalled during a performance or during composition. One method for approaching this feature is to treat a timeline bank as the material used for a chorus, while another timeline bank would contain the material used for a verse.

The play head that reads automation data can be limited to a specific start position and loop size within the timeline. The position and loop size may be set and manipulated live in real-time (see Figure 65), offering further potential for creative applications over the timeline.
Interestingly, the timeline feature in Kortex has proven to be a powerful technique for real-time improvisation. Future versions of Kortex will extend the creative potential of the timeline through implementing a unique timeline per parameter. Rules and interactive behaviours between the timelines could be set, opening up the potential for generative composition via parameter automation.

6.3.5 JenOSC

JenOSC is a modular routing extension for creating audiovisual relationships between musical information sent from Jennifer Max and both visual synth and AVGS parameters in Kortex. The musical information that is received is divided into four parts: compositional timescales, drums, bass and melody. Compositional timescales receive the current position of the composition, form, phrase, bar, beat, quaver and semiquaver. The musical elements, drums, bass and melody, send the current volume of each part and their current position through an actively playing note. Figure 66 below shows the modular routing interface for creating music-to-visual relationships. In this example, the compositional timescales and individual drum elements are able to be routed to the visual parameters growth, rubber, highlight, glow, intensity, camera X position and offset. Each musical element sends a number between 0.0 and 1.0 that can then be re-scaled by the user for each visual element to animate within an appropriate aesthetic range. Consequently, once an assignment has been made between a musical and visual parameter, the visual parameter becomes animated and directly reflects the musical element to which it is assigned. Collections of music-to-visual assignments may be stored as presets that can be re-called during a live performance.

---

20 Variable names describing their associated function in the shader algorithm seen in Figure 71.
Figure 66: JenOSC modular routing interface for creating relationships between musical information and visual parameters in Kortex
This approach has multiple aesthetic benefits. Access to the micro and macro compositional timescales and information about each musical voice’s volume and position enable a convincing visual representation of a composition to occur. However, careful consideration is required when creating collections of relationships. Because the range of potential correlations is increased, certain audiovisual correlations can be lost on an audience if either too many are occurring at once or a musical element has been mapped to an inappropriate visual parameter. For example, assigning the semiquaver position to the cameras’ X position will shift the visual perspective so fast that any other correlations will be unable to be perceived by an audience. However, if bar position was assigned to the cameras’ X position, then the audience would have enough time to make sense of the relationship and to also appreciate the other audiovisual correlations occurring in the scene. Finally, the addition of JenOSC provided the framework for AVGS to be experienced in relation to musical context.

6.4 Interface and Mappings

6.4.1 Controller Considerations

From the outset, Kortex was developed with the end goal of interfacing with and performing in real-time, in much the same way an instrumentalist would approach performing with a traditional instrument. From my past experience with mastering jazz trumpet and piano, I wanted to build a controller that would let me interface with the prototypes in a way that would encourage long hours of practice, play and ultimately mastery over the instrument. The challenges involved in designing the interface were balancing complexity, immediacy and flexibility.

I wanted to abandon the imposed limitations that came with off-the-shelf MIDI controllers and decided to build a custom controller. Although designing and building my own controller involved a substantial learning curve, I was convinced at the start that the uniqueness and close relationship between the controller and the prototypes would provide the most satisfying results.

I began by sketching out how many pots, sliders and pads I would need based on the controllable variables I had published for the user interface of the prototypes. Once I had an idea of how many components were needed, I considered various brain boards such as the Brain (Livid, Austin), Mega 2560 (Arduino, Ivrea) and MIDI CPU (Highly Liquid, Oregon) circuit boards that could connect the components together and allow for USB connectivity with a computer. Depending on the quality of the components, brain board and controller casing, the total cost was already exceeding $1000, even without the cost of a soldering iron and other expenses required for construction. From my initial research into what was required for the ideal controller, I decided to visualise one by sketching a design in Photoshop (seen below in Figure 67).
After reconsidering all the factors involved in building my own custom controller, I decided to perform another thorough search of available commercial controllers to see if any came close to my design prototype. In the third week of this searching process, Livid Instruments announced the release of a MIDI controller called CNTRL:R. The design of the CNTRL:R, seen in Figure 68, closely matches the design of my initial prototype. The great similarities between my design and the CNTRL:R encouraged me to abandon the idea of building a custom controller for this research project in the interests of time.
Having found the perfect controller, I then considered the practicalities of interfacing the controller and the prototypes.

6.4.2 Mapping the Controller

The CNTRL:R has the ability to send MIDI OUT and MIDI IN. It features 2x16 banks of knobs, 1x16 bank of endless rotary encoders with toggle functionality when pushed, eight sliders, 16 medium and 2x16 small pads that can be programmed to behave as pads or toggles. Because of the MIDI IN functionality, the pads can be programmed to display one of seven colours depending on a control change value sent to the specified pad. This feature provides a helpful visual cue to the user regarding the current state of an individual or bank of pads.

I then programmed a C++ add-on for openFrameworks in order to quickly start experimenting with various mapping setups with the prototypes. The code receives the MIDI data coming in and formats the messages into human readable types according to their type (pad, knob, encoder, slider) and bank (knobs bank1, knobs bank2 etc.). Methods that allowed the pads to behave like either a momentary pad or a Boolean toggle can be set, as well as a method that determines the pad’s colour. The C++ code was designed to be generic and non-biased to various implementations so I could prototype mappings with future projects. As such, I have published the code as open source through github so other CNTRL:R users can apply it within their own projects.21

6.4.3 Controller Mappings

Figure 69 shows the final mappings that interface the controller with the Kortex prototype. In the following section I describe the interface functionality and mapping designs.

21 https://github.com/JoshuaBatty/ofLividCNTRLR
Figure 69: Kortex and CNTRL:R interface mappings
**Timeline**

The loop bracket of the timeline can be controlled using the first two knobs in the top row of the left hand bank of knobs. Timeline loop position controls the start position in the timeline from which pre-programmed automation is read. Timeline loop size controls the length of automation data that is read in relation to the position knob. If timeline loop size is turned all the way to the left, timeline control is bypassed. Setting timeline loop size to any value greater than 0 activates timeline control and sets the size of automation data that is read. Stuttering and glitch effects are achieved when setting a small loop size and scrubbing the loop position knob. Accelerando and Ritardando effects can occur when manipulating the loop size in small increments.

**Harmonic Motion**

Harmonic motion is assigned to eight parameters: start frame, grain size, loop size, pitch, speed, crush, strobe rate and blur. The first four pads of the bottom two rows are assigned to switch if the sliders above should control one of two variables. For example, slider 1 can control either the harmonic motion’s start frame or grain size amplitude, or both if the two toggles are activated. The second row in the first bank of knobs controls the offset of the harmonic motion relating to the slider beneath it, while the third row controls the speed of the harmonic motion. If one of the eight harmonic motion parameters is toggled, then the related main parameter control on the right hand side of the controller sets the harmonic motion’s centre oscillation position. For example, if the start frame harmonic motion toggle is activated the start frame slider (1\textsuperscript{st} slider on the right hand side of sliders) would control the oscillation position. While complex manipulations and control can be easily achieved using these mappings, they also inspire an explorative and improvisational approach when interacting with the controls.

**Tweening**

The tween’s duration and easing type can be set using the last two knobs in the top row of the left-hand bank of knobs. These two parameters set the tween controls relating to the global audiovisual presets as well as the FM and granular synthesis presets in Kortex. Initially I wanted control over the global, FM and granular presets tween states separately but, due to the limited number of controls available, decided that combining them would encourage improvising with the tweening functions. Manipulating the duration while triggering presets feels similar to turntable scratching or changing the portamento on a synthesiser. As a result, in conjunction with the easing type, manipulations can range from staccato to legato.
Preset

Three preset banks can be controlled simultaneously on the controller:

- Global presets that change the state of every variable within Kortex can be triggered using the first three rows of the middle pads
- Granular synthesis presets change the state of every granular synthesis variable in Kortex by triggering the middle eight pads found on row one of the bottom pads
- FM synthesis presets control the FM engine in Kortex by triggering the middle eight pads, found on row two of the bottom pads. Assigning the preset selection to the pads over a slider or knob encourages non-linear rhythmic approaches when improvising. Powerful transformations can occur from the ability to instantly and simultaneously change the individual FM and granular synthesis states, as well as manipulating the entire global state of Kortex

Pad Modes

Pad modes (introduced earlier in section 5.5.1) were designed to overcome the limiting number of triggered presets available with 12 pads. Selecting the first pad in the bottom row of the centre pads increments pad mode by 1, and selecting the next pad to the right decreases pad mode by 1. For example, if pad mode is incremented from 1 to 2 then middle preset pads change from triggering presets 1–12 to trigger presets 13–24. Five available pad mode states enable the middle 12 preset pads to trigger up to 60 presets. The colour of the first pad mode pad represents the current preset bank, while the second pad mode pad colour represents the colour of the next preset available bank. Changing preset banks also changes the colour of the preset pads to give the user a visual cue as to which preset bank is currently active.

FX

A combination of GPU and CPU visual effects are assigned to the 12 rotary encoders. Assigning the FX section to the encoders allows for the user to push the encoder when an effect is desired and to push it again to switch it off. This allows for the user to manage optimal frame rates during a performance as the optical flow, slit scanning and triangulation effects require heavy CPU calculations. The slit scan delay maps can be selected via the last four pads of the bottom two rows, enabling non-linear rhythmic interaction. Manipulating the FX controls provide useful ways of deconstructing the image to achieve a sense of musical phrasing when improvising.
**Post-Processing & DSP**

Post-processing and DSP relationships are assigned to the top row of the upper right bank of knobs. These knobs control the default audiovisual relationships defined in Kortex. They offer a fine tuning alternative to using the other control methods and are generally interacted with in a subtle improvisational approach.

**Audiovisual Granular Synthesis**

Audiovisual granular synthesis parameters are assigned to the right-hand sliders and knobs on the controller. The parameters that produce the most obvious effects over the audiovisual grain when interacted with are start frame, loop size, speed and strobe rate. Interacting with a slider can be approached in a more immediate and aggressive fashion as opposed to interacting with a knob. Consequently, these fundamental parameters are assigned to the four available sliders to encourage immediate and frequent manipulation. The remaining AVGS parameters are assigned to knobs directly above these sliders. These parameters are best used sparingly to introduce subtle variations to the audiovisual construct.

**6.4.4 Touch Screen**

Although I had switched from using a touch screen interface for the tactile response of a MIDI controller after iteration 1 (see section 5.3), for my final examination performance I will utilise both the MIDI controller mappings outlined above and a touch screen interface. The touch screen application (seen below in Figure 70) was developed to provide instant visual feedback regarding the available audiovisual content that can be triggered during a performance.

Two factors led to the adoption of a touch screen interface for my examination performance. Firstly, all of the available pads on the MIDI controller were already assigned to other control parameters in Kortex. Furthermore, the MIDI controller is unable to provide visual feedback to the user regarding the specific audiovisual clip to be triggered, requiring the user to memorise clip-to-pad assignments at best, or at worst to simply guess. Secondly, the three visual outputs on my graphics card will be used to send visuals to the three projectors. As a consequence of the triple-screen setup, I am unable to use an onscreen graphical user interface to browse and trigger new content. In this scenario, touch screens facilitate an external GUI, without having to sacrifice an output on the graphics card needed to drive the three projectors.
6.5 Summary

In this chapter I outlined how I adapted the audio processing technique granular synthesis to the visual medium to enable AVGS, and described the relationships between micro audio and visual characteristics when they are synthesised into an audiovisual grain through a software instrument. The technology I developed to facilitate the instrument to run AVGS and visual shader synths in real-time was presented, as well as the creative image deconstruction potential of the effects I developed. I have implemented strategies and techniques in the instrument that provides the user with powerful transformations during a performance. Designing intelligent mapping relationships between an external MIDI controller and the Kortex prototype enables the user to approach the instrument much in the same way as an improvising musician.
7 Discussion

7.1 Overview
In this chapter I discuss the findings that emerged from designing and using the Kortex prototype as a real-time audiovisual instrument. Firstly, I consider how AVGS can be used to produce, intentionally, visual short-term memory phenomena for artistic applications. Secondly, I present my findings regarding performance interfaces from using both MIDI controllers and touch screens for controlling audiovisual media. Thirdly, I review the techniques and strategies for creating audiovisual relationships that emerged from this research. Fourthly, I examine how an artist can employ various control strategies to manipulate large collections of variables in a real-time performance situation. Fifthly, I discuss the real-time performance of Kortex running on specific hardware, before finally discussing the ways in which a live audiovisual ring buffer improved on the AVGS technique.

7.2 Visual Short Term Memory
I have observed that an interesting perceptual effect occurs when manipulating the length of the grain envelope. (To reiterate, the grain envelope defines a rise and fall in volume and opacity over a selection of audiovisual frames.) For example, if the grain envelope's length is five frames, and grain density is greater than one, the perceived aesthetic effect is of multiple overlapping audiovisual grains gradually phasing in and out of each other. Obviously, while the shape of the grain window will change how the grains overlap and intersect one another, the viewer can correctly perceive multiple audiovisual grains occurring simultaneously. This can be observed in Figure 71 below, which shows three audiovisual grains captured with a unique Gaussian window applied over a duration of five frames.
In contrast, setting grain density to one and defining the length of the grain envelope to a single frame has the potential to produce a similar perceived aesthetic as a result of the visual short-term memory phenomena. Although the viewer similarly perceives multiple overlapping audiovisual grains, the overlaps are constructed as an illusion in the mind. This is a result of the brain failing to adequately receive and process visual stimuli fast enough before the eyes are presented with new visual stimuli. Consequently, imagery is stored in visual short-term memory and the mind believes it is viewing multiple overlapping images. This effect is demonstrated in Figure 72, which presents the result of using a camera with its exposure set to 500 ms in order to convey the illusionary overlapping grains the mind perceives due to the persistence of vision effect.
The speed at which the playhead iterates over a selection of frames determines the visual phase pattern that the mind perceives. This is especially evident when the playhead speed is set anywhere between 20x and 150x the normal speed of playback. Once the speed exceeds 150x, the perception of momentum comes to a standstill and is similar to viewing the wheels on a car rotating at high speeds. For me, the audiovisual perceptual artefacts that occur via this approach are the most fascinating aesthetics of AVGS.

### 7.3 Performance Interfaces

In this section I discuss the observations I made through experimenting and performing with both touch screen and MIDI controller performance interfaces throughout the performance iteration cycle discussed in Chapter 5.

#### 7.3.1 MIDI Controllers

As a result of my background as a jazz instrumentalist, the tactile response of a MIDI controller facilitated an intuitive and expressive performance interface. Careful consideration went into creating the ideal mappings between control elements and specific functionality in Kortex. Once this stage was complete I was able to practise forming a relationship with the interface as if it were a traditional instrument. In doing so, I discovered which combinations of controls enabled the most expressive control over specific types of functionality in Kortex. For example, although both pots and sliders provide subtle and fine manipulations over parameters, sliders can be interacted with in a more aggressive fashion. Likewise, while the tactility of pads facilitates an ideal medium to rhythmically trigger banks of presets, they lack the visual feedback needed for triggering audiovisual content.
A delicate approach is required to maintain a balance between immediacy, expressibility and control as the number of parameters increases in the software instrument. A program like Kortex has upwards of one hundred controllable parameters that need to find space on the performance interface. PadModes was developed to circumvent this problem, but there is a limit to how many alternate states a set of controls can have before functionalities are buried so deep that immediacy suffers. Furthermore, the complexity of accessing these states limits the real-time creative potential of the instrument. Nevertheless, I believe the MIDI mappings outlined in section 6.4.3 balance these factors successfully and provide the most immediate, expressive and appropriate mappings between Kortex’s functionality and the interface.

7.3.2 Touch Screens

Initially in the first iteration I made exclusive use of a Jazz Mutant touch screen for my performance interface. As previously mentioned, fine parameter control over variables required that I diverted my attention from engaging with the visuals to correctly position my finger over the desired slider on the interface. Consequently I was not able to interact and engage simultaneously with the content I was producing, leading me to abandon touch screens as a performance interface.

Up until recently Kortex has been performed on single-screen setups. This meant I could display visual content via the HDMI output while using the laptop screen to display the grain reordering patterns and audiovisual thumbnails on the GUI. However, the triple-screen performance in Sydney for iteration 4 meant all the available outputs were used to send visuals to the projectors. This removed the ability to utilise a GUI to provide visual feedback and impacted my creativity during the performance. As a result I reconsidered utilising a touch screen interface to facilitate the required visual feedback. In doing so, I have found triggering audiovisual content and manipulating grain reordering patterns to be much more expressive and immediate than using a mouse to interact with an onscreen GUI. For example, I am able to easily select and punch in new audiovisual content in a rhythmical manner enabling me to play the interface as if it were a drum machine. Furthermore the interface seen below in Figure 73 provides instant visual feedback regarding the next 60 frames of audiovisual grain rearrangements. The visual feedback provides a highly intuitive approach for rearranging grains and encourages play and exploration.
Figure 73: Touch screen interface for manipulating the grain reordering patterns

The need to set up a local area network for communicating with the computer is the one downside to using a wireless touch screen device. Although most of the time this is not an issue, it does add extra stress when only having a few minutes to set up before performing. This can be alleviated slightly by using an ethernet cable, yet still requires configuration of the TCP/IP port, address and subnet mask for the OSC messages to be correctly received by the computer. I use an ethernet cable to connect the touch screen interface to my computer for two reasons. Firstly, it enables a more secure and reliable method to transfer messages, especially when playing in venues where nearly every audience member has a smartphone connected to a wireless network. Secondly, the latency of sending OSC messages over an ethernet connection is as low as 2 ms (Schmeder et al. 2010), so touch screen gestures and their associated control are perceived as immediate by the performer.

7.4 Audiovisual Relationships

The essence of my research was my investigation of how to create connections between image and sound so that the resulting construct could be perceived as a whole. During this process I discovered three different approaches that facilitate the formation of connections between the two media. These were defined by Edmonds et al. (2005) as being mathematical, metaphorical/intuitive or intrinsic. A mathematical relationship exists when the connection between audio and visual parameters is
expressed by an equation. Metaphorical and intuitive relationships are commonly found in film and are what I am referring to when describing perceptual characteristics. Finally, a relationship is said to be intrinsic when audio and visual media are synthesised from the same source.

Obviously genuine integration is very complex and nuanced and can be approached from many angles. I chose to take the metaphorical and intuitive approach to exploring relationships between image and sound. However, these relationships are bound by mathematical mappings that enable them to be perceived as emerging from the same source. Regardless of the approach taken, it is crucial that the resulting correlation can be perceived by an audience as an integrated audiovisual whole. Although the two media can be experienced separately from one another, the correlation seeks to strengthen and reinforce the emotional and aesthetic perception of the sensory object.

I have found modular routing interfaces facilitate an ideal approach for exploring various correlations between sound and image characteristics. The audiovisual routing interface described in section 6.2.2, and the JenOSC routing interface described in section 6.3.5, have become indispensable tools for instantly forging connections between the two media. I would recommend other audiovisual artists adopt a similar framework as it allows connections to be trialled and reformed at will according to the specific artistic goal in mind.

As previously noted, the creation of audiovisual relationships is highly subjective and, to a large degree, is influenced by each individual artist’s interpretation of the two media. This is in direct contrast to the ideas of the early Greek philosophers who sought to find an overarching equation that would describe a single fundamental relationship between sound and image. Although these early attempts appear to have been in vain, the subjective nature creates a near-infinite landscape of potential for artists to express their own unique voices based on their subjective interpretations of audio and visual characteristics. The digitisation of all media that occurred with modern computation has in the past few decades provided audiovisual artists with an extremely powerful tool to probe and explore the creation of new relationships. As new technology emerges and more of the world becomes digitised, the field of audiovisual art will continue to evolve and offer up new suggestions for linking and combining sound and image.

### 7.5 Performance and Control Strategies

Within the practice of audiovisual art, forming interesting collections of audiovisual relationships is fundamental, yet I believe the most important creative process exists in how those relationships behave and evolve over time. One analogy is to see the creation of audiovisual relationships like a composer selecting instrumentation for an orchestral work. The interplay between the instrumentation
emerges from collections of overlapping and intersecting musical patterns which the composer uses to create the composition. Likewise, an audiovisual artist can utilise pattern-generating modulation sources to animate the behaviour of the relationships. In doing so, evolving phrases are generated which the composer can use to transfer meaning to an audience.

The grain reordering patterns described in section 4.3.1 and the various control strategies described in section 6.3 were developed to facilitate the above approach. Of course, while a single parameter that manipulates the audiovisual construct can be directly controlled, it becomes unfeasible for a single performer to manipulate more than two parameters simultaneously in a live performance. Powerful control strategies are thus necessary to help the performer control huge collections of variables simultaneously.

A great example is the amount of control the grain reordering patterns enable to manipulate not only AVGS but the FM synthesis engine and the visual shader synths. Through manipulation of the grain reordering patterns, the user is defining the reordering of audiovisual grains throughout time, and the value of FM synthesis parameters that are in turn directly driving the parameters of the visual shader synths. Consequently, the performer can focus on improvising with various rhythmical patterns that affect the global audiovisual state of the instrument.

Finally, creating banks of presets in the studio that can be later recalled during a performance captures the best aesthetic qualities of Kortex and creates confidence when performing. Playful experimentation in the studio allows the user to trial and tweak combinations of variable states until a desired aesthetic state emerges. Creating a preset then captures the global state of the instrument which the performer can then instantly recall and present to an audience. In this way, the user captures the best of the aesthetic states that emerge during the studio exploration process.

I believe it is important for this process to happen in the studio instead of in front of an audience, for two reasons. Firstly, it can take a long time to reach the ideal combination of variable states. A slight change in the value of a single parameter can dramatically change the resulting aesthetic. Secondly, an audience can be exposed to a larger number of aesthetic states that the performer has deemed interesting through recalling presets. Although the performer can explore variable combinations in front of an audience, they can at any time recall a preset to instantly provide contrast at pivotal moments in the set. The ability to slide between these states over time using tweening functions provides the performer with an extremely powerful and versatile strategy when performing live audiovisual art.
7.6 Technology

7.6.1 Hardware Performance Boost
Kortex was developed with, and performed on, an early 2011 MacBook Pro laptop for the first three years of my candidature. As the prototypes’ complexity increased I decided to change the internal hard drive to a 480GB SSD and to increase the amount of available RAM from 4GB to 16GB. The change from a traditional hard drive to an SSD contributed to a 20 fps increase in speed, while the extra 12GB of RAM meant I was able to store up to one minute of video frames in memory when using visual granular synthesis. The upgrade in speed and memory enabled me to add functionality into the prototype that would not have been possible otherwise. However, even with the upgraded performance, the MacBook Pro could only run the prototype at 60 fps with a resolution of 512x512 pixels on a single screen.

To enable Kortex to run at 60 fps using a triple-screen setup I built myself a custom hackintosh mini-ITX desktop computer (see section 3.6.2). With the powerful CPU and GTX Titan graphics card, I was able to develop Kortex while outputting to three monitors with a resolution of 1024x1024 at 60 fps. Furthermore, the size of the mini-ITX case allowed me to transport the computer as carry-on luggage when performing interstate and internationally. Considering the ease with which I am able to upgrade the machine with faster technology in the future, in comparison to a laptop, while retaining the convenience of portability, has led me to conclude that custom mini-ITX computers provide the best overall value for touring audiovisual artists.

7.6.2 Audiovisual Ring Buffer
The decision to utilise a circular ring buffer to store, read and overwrite audio and visual media into their respective buffers led to a several positive AVGS enhancements. Without a ring buffer, new content had to be uploaded into a temporary memory buffer frame by frame. Once all content was uploaded, the current audiovisual buffer was swapped with the temporary buffer and new content was then able to be manipulated. The process of uploading new content into the temporary buffer while the current buffer was being used required a considerable amount of CPU cycles, halving the frame rate of the playback of the current buffer. Furthermore, uploading new content with a length of 30 seconds would take on average 15 seconds to complete. This meant I suffered an unacceptable performance cost, and had to plan well in advance to evolve the audiovisual material. In contrast, the adoption of a ring buffer means only one buffer is required for reading and writing audiovisual content. This alleviates any performance costs when writing into memory and enables new content to be immediately accessible for AVGS processing.
Another advantage to this technique is the ability to stream live input from multiple sources in real-time. For example, capturing microphone and camera input, live television or streaming from YouTube expands the creative potential use of AVGS as a real-time processing technique. Finally, the ability to immediately select and process audiovisual content enabled a similar creative approach to real-time DJ and VJ performance software.

7.7 Summary

In this chapter I discussed the results and findings relating to key aspects of my research that emerged through using Kortex as a live audiovisual instrument. I showed that manipulating the size of the grain envelope produces different perceptual effects, specifically how the visual short-term memory phenomena produces illusionary overlaps when grain size is defined as a single frame. This was followed by examining the practical benefits of using both MIDI controllers and touch screens as performance interfaces to control live audiovisuals. I concluded that MIDI controllers excel at fine parameter control because of their tactical feedback, while touch screens excel at controlling functionality that requires visual feedback. Next, the creation of audiovisual relationships was shown to be mathematical, metaphorical/intuitive or intrinsic. I presented my personal observations about the subjective nature of creating relationships between image and sound and suggested that audiovisual artists use a modular assignment matrix to easily trial and form connections. Once an audiovisual relationship has been created, powerful pattern-generating control strategies are vital to define the relationship’s behaviour over time and to assist the performer in manipulating large collections of audiovisual parameters in real-time. I then discussed the real-time performance boosts gained from building a custom hackintosh mini-ITX computer. Finally, utilising a live audiovisual ring buffer technique was shown to expand the potential application of AVGS to include live streaming sources and strengthened the capabilities of AVGS as a real-time performance technique.
8 Conclusion

8.1 Overview
This chapter outlines how my research project has contributed to knowledge in my field of study and summarises my research outcomes in relation to the research objectives. My objectives included: (i) to establish a visual granular synthesis processing technique; (ii) to create a synchronised audiovisual software environment; (iii) to realise the creation and performance of audiovisual art in real-time. The significance of the research findings are presented, as well as their limitations. I give recommendations for future research and potential applications of my project instrument that emerged as a consequence of my research.

8.2 Research Outcomes
The goal of my research was to develop an AVGS software instrument that would enable synchronous manipulation over pre-defined audiovisual relationships in order to create and perform audiovisual art in real-time. To achieve this, the investigation was guided by two research questions.

The first question was how audio processing techniques, such as granular synthesis, can be adapted and applied to influence new visual performance techniques. The second question was how computer software can synergistically fuse audio and visuals together to enable the real-time creation and performance of audiovisual art.

In order to answer the first research question, I investigated and described historical and more recent approaches to granular synthesis in Chapter 2. Through this investigation I uncovered approaches for deconstructing sound at the micro level, which were similarly used to deconstruct images in order to adapt granular synthesis for visual processing. The investigation established technical approaches and control strategies that informed the development of the project instrument, referred to as the Kortex prototype.

8.2.1 Adapting Granular Synthesis for Visual Processing
The findings of my project-led research into how audio granular synthesis can be adapted for visual processing are as follows:
• Visual media needs to be deconstructed down to the smallest indivisible unit, which within the scope of my research was a single frame of video

• Common parameters found in an audio granular synthesiser must be analysed individually to determine a parameter’s perceptual characteristics

• Micro relationships can then be defined based on similarities between the two media’s processing characteristics.

Visual granular synthesis can be achieved by following the same deconstruction methods that are applied to audio. That is, the main parameters of audio granular synthesis are window shape, grain length, grain pitch, and speed. These can be applied to the playback of a visual grain in an identical manner to audio granular synthesis. However, the grain pitch variable is related exclusively to audio processing and has no obvious direct relationship with visual processing.

In cases where an audio parameter has no direct visual equivalent, visual parameters must be chosen based on the artist’s subjective interpretation of the parameters’ processing effect. Raising the pitch has a perceptual effect of making the signal more brilliant and bright, while lowering the pitch has a perceptual effect of making the signal darker and less defined. When looking for a visual counterpart to map to the pitch parameter, the artist must search for a visual processing technique that affects the visual signal throughout its range from dark and undefined, through to normal, and on to brilliant and bright. These are characteristics that both audio and visual processing share. For example, the visual processing effect called ‘bloom’ contains all of these perceptual characteristics within its processing range. Although such judgements are subjective, I found that convincing audiovisual relationships occurred when I combined audio and visual processing techniques that shared as many similar perceptual characteristics as possible.

With regard to the second research question relating to using computer software to synergistically fuse audio and visuals to enable the real-time creation and performance of audiovisual art, my investigation uncovered strategies and techniques for identifying, defining and controlling the behaviour of complementary audiovisual relationships, as well as the capacity for the prototype to create and perform audiovisual art in real-time.

8.2.2  Fusing the Audiovisual Construct

The findings of my investigation into the process of defining complementary audiovisual relationships indicated that:
• Genuine integration of sound and image is very complex and is based to a large degree on the subjective interpretation of the artist

• A successful correlation seeks to create a sensory experience greater than would occur if the two media were experienced in isolation from one another

• Relationships between sound and image can be based either on their perceptual properties or their technical/mathematical implementation. Implementing a technical process used on one medium in the other can transfer similar perceptual characteristics

• Although the process is highly subjective, by identifying the aesthetic characteristics of an audio processing technique, one can then survey a range of visual processing techniques to look for a visual equivalent that shares similar aesthetic characteristics. The visual and audio processing techniques can then be said to share perceptual similarities that justify a relationship between the two

• Audiovisual artists can use this process as a starting point for creating synergistic relationships between the two media

• Once audio and visual media are collapsed into a digital soup of binary data, it is of key importance that the data is thoughtfully re-modulated back into the physical realm so that the correlations between the two media can be understood by an audience

• Transcoding processes, such as sonification, enable audiovisual relationships to emerge from the same data source. Despite the fact that sound and image are generated from the same binary data, the process of manipulating the data to achieve an auditory or visual result is also subjective.

8.2.3 Real-Time Performance of Audiovisual Art

The findings of the investigation into how software can assist in the creation and performance of audiovisual art in real-time indicated that:

• To manage the large number of control parameters in a real-time performance setting requires high-level control strategies. Thoughtful consideration when creating preset states, parameter automation banks, and designing intelligent controller mappings enables rich and complex transformations during a performance
• Pattern-generating modulation sources create various timescales of phrases similarly to how music is constructed

• Modular audiovisual assignment matrices enable the user to instantly trial and form connections between characteristics according to the specific artistic goal in mind

• Audiovisual software that features quick video access, post-processing visual effects and audio granular synthesis with effects processing signal chains is computationally expensive. It is recommended that audiovisual applications are programmed using a low-level programming language, such as C++, and a dedicated graphics programming language such as GLSL or High-level Shader Language

• Visual granular synthesis requires consistent and immediate random access to frames of video. APIs, such as QuickTime, stream the media file from its location on the CPU, producing an inconstant update speed. Transferring the frames of video as textures onto the GPU alleviates this issue and provides a consistent, real-time frame rate of 60 fps

• Visual effects that require textures to be transferred from the GPU to the CPU for analysis should be used sparingly. Visual processing techniques that function exclusively on the GPU are recommended if high frame rates are critically important.

8.2.4 Contributions
My research by project investigated how to create a synchronised AVGS software environment that facilitates the creation of audiovisual art in real-time. The main contributions of this research project were the design and creation of:

• A new software instrument that enables the real-time composition and performance of audiovisual art

• An audiovisual granular synthesis engine that enables artists to manipulate and reconstruct material into new audiovisual forms

• A modular framework that assists in creating custom relationships between visual post-processing and auditory DSP effects

• A modular framework that assists in creating custom relationships between visual shader synth parameters and micro/macro musical elements
• Powerful control strategies for manipulating large collections of parameters during the compositional process or while performing

• A novel technique for handling frames of video that allow instantaneous random access to individual frames stored as textures of RAM

• A collection of visual effects that further facilitate the potential for image deconstruction

• An overview of current multimedia programming environments and libraries that facilitate real-time audiovisual art.

8.3 Limitations
Specific limitations of my project outcome (Kortex) include a functional limitation of the instrument when transferring graphics’ calculations from the GPU to the CPU. Post-processing visual techniques, such as optical flow and triangulation, require computer vision analysis of each frame to check whether an object has moved or to find contours. Graphics textures need to be uploaded from the GPU to the CPU in order to perform computer vision analysis. Video frames are uploaded and processed on the GPU, before being transferred for analysis to the CPU, and then moved back to the GPU for final display for each player object. The frame rate halves each time this transfer from the GPU to the CPU occurs because the CPU is exponentially slower at graphics calculations than the GPU.

Kortex consistently runs three granular audiovisual players at 60 fps but drops to 5 fps when slit scanning, optical flow and triangulation are turned on for each player. Obviously, transferring textures from the GPU to the CPU for analysis renders these visual techniques unusable for real-time applications. In my future research I will attempt to eliminate this step through implementing GPU functionality for computer vision analysis. It is hoped that a GPU computer vision library would enable processing techniques, such as optical flow and triangulation, viable for real-time applications.

8.4 Areas and Directions of Future Research
This section offers recommendations for researchers who are interested in further investigating AVGS and the relationships between audio and visual processing techniques.

My research project investigated how low-level audio processing techniques such as granular synthesis, and high-level DSP, could be adapted to influence new visual processing techniques. The outcomes of my investigation open up other areas for extending this research, such as:
• Continuing to adapt audio synthesis techniques to create visual equivalents

• Adapting visual processing techniques to create audio equivalents

• Developing generative algorithms for calculating data that can be sonified and visualised from the same source.

A goldmine of creative applications can be tapped through the development of new audio DSP effects via translating specific visual processing techniques. Of particular interest is the capacity for an audio phase vocoder to be controlled by optical flow and slit scan visual effects. The ability of a phase vocoder to shift, phase, blur and average spectral bins makes the technique a promising tool for implementing time/frequency dissolving audio processing effects. I am currently working on implementing an audio phase vocoder in Kortex in an effort to create audio processing equivalents to the visual effects described in section 6.1.6.

Another area for extending my research is to integrate the visual post-processing and audio DSP effects into the granular synthesis engine. For example, instead of post-processing the audiovisual grain cloud using a macro reverb/blur relationship, the effect could be translated to be a microgranular relationship. Each audiovisual grain could then apply an individual reverb/blur process with unique parameter settings per grain. Other macro relationships defined in Kortex, such as audiovisual delays, pan/spatial location, and downsampling/pixelation, would benefit from integration at the granular level within the AVGS engine and release even greater creative potential.

This research has also highlighted a necessity for developing computer vision analysis that can perform in the graphics processing pipeline. This would alleviate real-time performance issues with visual effects that require calculations on the central processing unit (CPU).

8.5 Potential Applications

The technical and aesthetic capabilities of Kortex can be used by artists in different ways. One exciting direction would be to use Kortex for the production of a tightly synched music video clip, not only after the fact of music creation but during the music composition process. Gaming, advertising, TV and cinema could also use the complementary audiovisual manipulation in Kortex to strengthen and reinforce certain products, scenes, plots and characters.

DJs and VJs could use Kortex for many practical performance applications. The popularity of DJs and VJs in performance entertainment has conditioned audiences to expect simultaneous audiovisual experiences. Kortex is designed to provide a unified creative environment for audiovisual composition
and performance. Consequently, the instrument can help DJs and VJs to produce a tighter synchronisation between sound and visual elements, thus unifying them into a single audiovisual performance element. It is hoped that the development of Kortex will lead to more immersive audiovisual experiences for audiences, and provide DJs and VJs with an expressive platform for creative collaboration.

Kortex can be performed as a live instrument through interfacing with a MIDI controller, or as a compositional tool by automating parameters using the inbuilt linear timeline. In each case, the micro- and macro-level associations that define audiovisual relationships enable a unified variable for synchronous audiovisual manipulation. To provide a richer audiovisual performance experience, the user can approach performing audiovisual material in a similar way to that of an instrumentalist in a band. By performing on top of pre-composed music or alongside other musicians, the user takes the part of a live audiovisual musician, improvising in relation to other musical cues. However, when mastering any instrument, the best results require long periods of experimentation and practice.

### 8.6 Final Conclusions

This exegesis presents audiovisual mapping techniques used by visual artists for the creation of audiovisual art and describes my investigation of audio granular synthesis to uncover the aesthetic, technical and performance requirements of the technique. My investigation drove the development of a software prototype that is capable of audiovisual granular synthesis, visual shader synthesis, frequency modulation synthesis, and creating audiovisual art in real-time. Software development was guided by a cyclical process of action and reflection in order to evaluate and further develop subsequent prototypes.

With the advent of the computer, all media such as text, sound, and image are represented by a string of binary numbers. As a result, the boundaries that separate one medium from the other have become permeable. This development has many implications that have only recently started to be explored. It opens up an exciting new field and provides enormous creative potential for defining relationships between media and generating audiovisual art. The subjective nature of defining audiovisual relationships should encourage artists to explore and play with forming unique associations.

I hope that the results of my research project and exegesis serve as a starting point for future researchers, developers and artists interested in exploring the enhancing effects sound and image give to one another when approached in a synergistic way.
Due to the nature of this project, many of the resources I drew on were only available online. When publicly available web pages were used, their URLs are provided. The primary locations of such resources can disappear, but it might still be possible to retrieve them through the Internet Archive (http://www.archive.org).

Abbado, A 1988, ‘Perceptual Correspondences of Abstract Animation and Synthetic Sound’, 


*AudioMulch* for OSX 2014, software, version 2.2.4, Sonic Fritter, Melbourne.


*B6700 Display Panel*, n.d. photograph, viewed 20 June 2012, 


Csound for OSX 2014, open source code, version 6.0.


Daniels, D & Naumann, S 2011, *See This Sound: Audiovisualogy 2*, 1st edn, Walther König, Cologne.


*Experiments in Motion Graphics: The John Whitney Collection* 2009, dvd, Pyramid Media, Santa Monica, California.


*FM8 for OSX* 2014, software, version 1.0, Native Instruments, Berlin.


*Granular Synthesis: Modell 5. Motion Control* 1996, DVD, HTBA Hull, Germany, 4 February.


Kyma X for OSX 2014, software, version X.82, Symbolic Sound, Champaign.


Max/MSP/Jitter for OSX 2014, software, version 6.1, Cycling 74, California.


MetaSynth 5 for OSX 2013, software, version 5.01, U&I Software, Paris.

References

MilkDrop 2003, accessed 17 February 2011,


Moritz, W 2012, Digital Harmony: The Life of John Whitney, Computer Animation Pioneer, Digital Media Assistant, viewed 13 May 2014,


Nordine, M 2014, Jen's Making Beats!, Mitchell Nordine, viewed 10 March 2014,

Olofsson, F 2009, Audiovisuals with SC, Fredrik Olofsson, viewed 17 April 2014,
<http://www.fredrikolofsson.com/f0blog/?q=node/316>.


openFrameworks 008 for OSX 2014, open source code, version 0.8.0.

OpenGL 2014, Debugging Tools, viewed 14 June 2014,


*Processing for OSX 2014*, coding environment, version 2.0.

*Reaktor* for OSX 2014, software, version 5.0, Native Instruments, Berlin.


*SuperCollider* for OSX 2014, open source software, version 3.6.5.


Truax, B 1990, ‘Composing with Real-Time Granular Sound’, *Perspectives of New Music*, vol. 28, no. 2, pp. 120-134.


Glossary of Acronyms and Abbreviations

API........................................ Application Programming Interface
AVGS ..................................... Audiovisual Granular Synthesis
AVJ....................................... Audiovisual Jockey
CPU ....................................... Central Processing Unit
DAW ....................................... Digital Audio Workstation
DJ .......................................... Disc Jockey
DSP ....................................... Digital Signal Processing
FM ......................................... Frequency Modulation
FPS ....................................... Frames Per Second
GB ......................................... Gigabyte
GLSL.................................... OpenGL Shading Language
GPU ....................................... Graphics Processing Unit
GUI ....................................... Graphical User Interface
Hz .......................................... Hertz
IBM....................................... International Business Machine Corporation
IDM ....................................... Intelligent Dance Music
LCD ....................................... Liquid Crystal Display
LFO ....................................... Low Frequency Oscillator
MIDI ...................................... Musical Instrument Digital Interface
MIT ....................................... Massachusetts Institute of Technology
ms .......................................... Milliseconds
OSC ...................................... Open Sound Control
PAL ....................................... Phase Alternating Line
RAM .......................... Random Access Memory
SSD ............................. Solid State Drive
STK ............................... Synthesis ToolKit
VCA ............................... Victorian College of the Arts
VJ ................................. Video Jockey
VST ............................... Virtual Studio Technology
WAAPA ........................... Western Australian Academy of Performing Arts
Appendix

AVGS Demo Software

The AVGS demo software that accompanies this exegesis can be found on the USB memory stick and demonstrates the AVGS engine at the heart of Kortex. Note, the application will only run on a computer running Mac OS 10.8 or higher and requires a web cam and microphone. If using the inbuilt microphone, it is highly recommended that you use headphones to avoid feedback.

Figure 74 shows a screen-shot of the user interface. The available parameters provide the user with the ability to manipulate the audiovisual relationships described in section 6.2.

![Figure 74: AVGS demo software user interface](image)

Properties

Record Input ......................... Defines if new audiovisual content should be recorded into the buffer.
By default this is set to true

Volume ............................. Sets the overall volume level sent to the sound card
Set Position......................... If turned on, position is set using the position slider and the speed parameter becomes inactive. Turning set position to off reactivates the speed parameter to drive the position of the play head.

Position............................. If the set position toggle is activated, the position slider controls the play head position within the buffer.

Randomisers....................... The position, pitch, and grain size parameters have a rotary encoder to the left of the slider that introduces a percentage of randomness to each grain.

Speed............................... Defines the speed and direction at which the play head will read through the audiovisual buffer. The overlaps of the audiovisual grains become most noticeable when the speed parameter is set between -1.0 and 1.0.

Pitch............................... Sets the grains pitch.

Grain Size......................... Defines the size of the audiovisual grains in milliseconds.

Overlaps............................ Defines how many active audiovisual grains or voices are playing.

Grain Window...................... Selects a grain envelope shape to apply to the audiovisual grains. Available shapes are, Gaussian, Cosine, Triangle, Rectangle, and Blackman-Harris.

Blend Mode....................... Defines the blending mode used to mix the visual grains. Available modes are, Alpha, Add, Screen, Multiply, and Subtract.

Presets............................. Toggles 1 of 25 pre-defined presets. Triggering a preset will change all of the parameters listed above. Exploring the presets provides a quick way to explore the different aesthetics of the application.
First Audiovisual Experiment

The first piece of software I ever developed was a real-time application built in MAX/MSP/JITTER that enabled me to experiment with synchronising audio and visual parameters. The application was both my first attempt at creating software and experimenting with visuals. Looking back, a lot of ideas and features of Kortex can be seen to have germinated within this application, and it marks the point at which I became fascinated with combining audio and visuals. Figure 75 shows a picture of me presenting the application at an Overlaps meet up in 2009.

Figure 75: Presenting my first audiovisual software at Overlaps, Melbourne 2009
Kortex Version Alpha

The very first proof-of-concept Kortex prototype was developed between January and July 2011. The prototype focused on developing a technique to synchronise independent streams of audio and visual media as well as implementing real-time audio granular synthesis. Figure 76 shows a screenshot of Kortex in the early alpha stage.

![Kortex alpha screenshot](image_url)

Figure 76: Kortex alpha screenshot
Kortex Version 1

Kortex achieved Version 1 status in August 2011. The prototype enabled AVGS control over three audiovisual players. Each player could be controlled independently or simultaneously via a GUI. Along with independent AVGS control, Version 1 enabled new media content to be uploaded to RAM in run time. This provided a way to swap new content in and out of the players’ audiovisual buffers while the application was running. However, this technique strained the CPU as the application would drop from 60 fps to 30 fps while new content was being uploaded to a temporary buffer. Figure 77 shows a screen shot of the AVGS controls, while Figure 78 shows a screenshot of the content uploading user interface.

Figure 77: GUI controls for the three independent AVGS players

Figure 78: Kortex Version 1 – content switcher user interface
Kortex Version 2

Kortex upgraded to Version 2 in January 2012. Notable new features included a complete redesign of the GUI, the addition of a timeline for key frame animations, as well as auditory DSP and visual post-processing relationships. Figures 79 and 80 below show screenshots of Version 2 in the development stage; while Figure 81 shows the completed version with additional timeline support, preset saving with tween functionality, and a graphical modulation matrix for redefining audio and visual relationships.

Figure 79: Early Kortex Version 2 – screenshot featuring brand new GUI and DSP functionality
Figure 80: Early Kortex Version 2 – featuring visual post processing and timeline support

Figure 81: Final Kortex Version 2 user interface
Kortex Version 3

Kortex is currently at Version 3, the last version that will be developed during my candidature. Version 3 features visual thumbnails for triggering both AVGS content and visual shader synths; an FM synthesis engine; preset and project saving; a live audiovisual ring buffer; JenOSC support; simple harmonic motion and tweening; timeline control and audiovisual assignment modulation matrices. Consequently, Version 3 is the most comprehensive and feature-packed version to date, and unifies the aims and objectives of this research within a single environment.
MindBuffer Live Shader Synth Software

The visual shader synth functionality in Kortex evolved separately from AVGS until Iteration 4. Figure 82 below is a screenshot of the MAX patch that was developed to synchronise Ableton, VDMX, Syphon, Processing sketches and a touch screen interface for Iteration 1. The visuals were primarily programmed using the Processing language and routed through VDMX for mixing and post-processing. Figure 83 shows the Lemur touch screen interface and the generative MAX patch that was used for live performances during this period.

Figure 82: MAX patch developed for Iteration 1

Figure 83: Generative audio MAX patch and Lemur touch screen used during Iteration 1
Following Iteration 1, I decided to abandon my reliance on multiple pieces of software and to instead program all the required functionality using openFrameworks and GLSL. The GLSL shaders were used to program a variety of 2D and 3D audio reactive scenes and to handle all post-processing effects. OpenFrameworks was used to pipe OSC data, handle MIDI routing, and to output the visual canvas for display. Figure 84 shows a screenshot of the visual software that was used for live MindBuffer performances during Iteration 2 and 3. Figure 85 displays the software in action during a live MindBuffer performance at Brown Alley in Melbourne, July 4th 2013.

Figure 84: OpenFrameworks live visual software developed during Iteration 2 and 3
Figure 85: MindBuffer performing live at Brown Alley, Melbourne July 2013
GLSL Texture Mapping Experiments

Texture mapping experiments were created to explore interesting ways of combining the visual output of AVGS with the geometry of the visual shader synths. To achieve this, the visual output of AVGS was rendered into a frame buffer object, which was then passed into a texture2D object in GLSL. The algorithm that was responsible for creating the geometry in the shader script was then used to displace the coordinates of the texture2D object. Figure 86 below shows the result of an AVGS texture being wrapped around 3D geometry, while Figure 87 shows the texture wrapped around 2D geometry.

![Figure 86: AVGS texture wrapped around 3D GLSL shader geometry](image1)

![Figure 87: AVGS texture wrapped around 2D GLSL shader geometry](image2)