Experimental Approaches to the Composition of Interactive Video Game Music

Joshua Rayman

MMus Research In Electroacoustic Music and Sonic Arts
University of East Anglia
School of Music

June 2014

This copy of the thesis has been supplied on condition that anyone who consults it is understood to recognise that its copyright rests with the author and that use of any information derived there from must be in accordance with current UK Copyright Law. In addition, any quotation or extract must include full attribution.

Abstract

This project explores experimental approaches and strategies to the composition of interactive music for the medium of video games. Whilst music in video games has not enjoyed the technological progress that other aspects of the software have received, budgets expand and incomes from releases grow. Music is now arguably less interactive than it was in the 1990's, and whilst graphics occupy large amounts of resources and development time, audio does not garner the same attention. This portfolio develops strategies and audio engines, creating music using the techniques of aleatoric composition, real-time remixing of existing work, and generative synthesisers.

The project created music for three 'open-form' games : an example of the racing genre (Kart Racing Pro); an arena-based first-person shooter (Counter-Strike : Source); and a real-time strategy title (0 A.D.). These games represent a cross-section of 'sandbox'-type games on the market, as well as all being examples of games with open-ended or open-source code.

Table of Contents

Figure List

Documentation

Video 1. KRP Composition Extract
Video 2. Rez demonstration
Video 3. KRP Composition Extract
Video 4. KRP MIDI Rendition Extract
Video 5. 0 A.D Composition Extract
Video 6. 0 A.D Second Composition Extract
Video 7. 0 A.D Unedited Footage (First Composition)

Introduction

Computing is continually advancing at a rapid pace, offering opportunities to both content creators and users that would have been hard to comprehend fifty years ago. Entirely virtual worlds allow users to explore and interact with a great variety of scenarios, environments and time periods - to exist beyond their own means. This creates a whole new world of gesture and interaction that would have been alien as recently as twenty-five years ago. Through video games, recent generations have been growing up with the access to a wide variety of stimuli, albeit virtually, with minimal interaction on the part of their physical body. Reflecting the interactive and vast nature of the possibilities at hand for users should be a natural goal of composers working in the medium, but despite the ever-changing technology in video games, aspects such as music are left behind. Most video games employ pre-recorded music, usually composed in small blocks or loops prior to the completion of the game. This method of working results in material that is of limited adaptability once the recording is complete.

The lack of progress is noted by the developers/composers for LucasArts Games title *Monkey Island 2 : LeChuck's Revenge*:

> *"Although music and sound effects are an important part of the game's 'feel', the technological progress which has been made in this area has been relatively limited... there has been little technological advancement in the intelligent control needed to provide automated music composition which changes gracefully and naturally in response to dynamic and unpredictable actions or the 'plot' of the game"*[1]

Music in video games can serve several functions, such as providing an ambient backdrop to the environment the player is experiencing or communicating information about the game play. One reason this is important to the overall experience of the game is the additional information music can provide to help the player, who lacks the tactile and spatial stimuli that they might experience in the real world. Depending on genre, audio can encourage certain actions, indicate danger, show change of location, and aid in creating tension. This is seen in games with minimal narrative, such as Flower, which

1 Land and McConnell, from Collins (2008), p51-52

uses music extensively to communicate the emotional context of a scenario[2], and is otherwise relatively abstract with no sentient objects to relate to. This is similar to how music can be used in film, an environment where the audience is also intended to live vicariously through the characters and situations on-screen.

Despite these important and useful functions of music in video games it is often marginalised and given few resources[3]. This is improving with the advent of larger storage capacity, as games exceed several gigabytes of data in size; expanding the library of music, stored in increasingly efficient compression algorithms, is becoming more assailable. Whilst the expansion of music content allows composers to write more music cues to cover a wider variety of situations, the audio is still triggered in a linear way, relying on pre-cued execution points and often embracing interactivity only through manipulation of basic digital signal processing (DSP) effects such as volume and equalisation. Games such as Dead Space are starting to incorporate aleatoric elements[4] and real-time DSP to their scores, as composer Jason Greaves composed and recorded with a sixty piece orchestra by recording the individual sections with "multiple dynamic levels and variations". These are then mixed at run-time by the game code[5]. Whilst this allows some aleatoric influence to the original score, with the game adapting to gameplay and input; it is an incredibly resource intensive process, requiring the orchestra to run through and record each variation and dynamic level – effective but prohibitively expensive in most commercial applications[6].

This could be due to an expectation of replication rather than simulation for music. Whilst video games graphics engines strive for realism, this is very rarely achieved by literally rendering a real-world photograph of an environment into a game. This would

2 Yuen, I. (October 31, 2013). Sound and Music as Narrative in Flower, *Ekostories.com*. Retrieved December 12, 2013 from http://ekostories.com/2013/10/31/sound-music-flower-narrative/

3 Collins (2008), p. 82

4 Sines, S. (August 10, 2008). Jason Graves Crafts Spine-Tingling Score for Dead Space. 1up.com. Retrieved December 11, 2013 from http://www.1up.com/news/jason-graves-crafts-spine-tingling-score

5 Electronic Arts Inc. (October 6, 2008). "Dead Space Reveals Spine-Tingling Score Composed and Conducted by Jason Graves", investor.ea.com. Retrieved May 15, 2014 from http://investor.ea.com/releasedetail.cfm?ReleaseID=338525

6 Hannigan, J. (April 10, 2014) *Ludomusicology Conference 2014 Composer Roundtable.* Ludomusicology Conference. University of Chichester. Address

look out of place on screen alongside the 3D objects and textures that make up most video games. A suspension of disbelief is enacted so that the player immerses themselves into the simulation of the virtual world presented. On the other hand, video game music is very rarely 'virtual'[7], which would be more malleable in real-time – instead mostly using actual sound recordings of real instruments, performers and groups. Borrowing from the film world, a symphonic feel is expected in some games (particularly action games such as the Battlefield series), which in my opinion creates the expectation in audiences of replication rather than simulation.

The idea of the virtual in aspects of video game creation even extends to sound effects - where once a game used pre-recorded sound effects, more advanced methods are being introduced to render sounds such as gun-shots and car engines in recent years, such as using granular synthesis to manipulate a pre-recorded sound effect; or even using physics to model the sound behaviour of an object in an environment. This requirement of game audio, both in terms of sound effects and music, is reflected upon by Marty O'Donnell, composer of titles such as *Halo* and *Myth* :

> *"The most important feature… is that it contains enough permutations and the proper randomization so that players do not feel like they're hearing the same thing repeated over and over"*[8]

It is curious that so few games try to simulate or synthesise music in real-time, which would be much more flexible than using pre-recorded tracks and would therefore be able to be manipulated in ways to create many more permutations. This is what fuelled my interest in exploring new approaches and strategies to video game music creation, which eventually focused on real-time sound generation**.** By creating 'audio engines' that depended on the game running to create sound, gameplay is closely tracked in the resulting music – creating dynamic sound worlds and new ways to deal with interactive media.

Working with video games presents a challenge to the composer, due to the restrictions set by the demands of the industry, both financially and due to limitations of the

7 Winkler p. 5

8 Collins (2013) p.33

transmission medium[9]. Music written for the present generation of video games is mostly pre-recorded and, as noted in several academic articles, tending towards film music : "[c]urrent trends indicate that a video game's musical accompaniment can now match the orchestral scores present in the film industry" (Villarreal, 2009); "there has been a trend in narrative-based console games towards cinematization" (Bridgett, 2007).

"Game platforms are able to change the playback rate (sometimes referred to as the pitch) and the volume of a sample on playback, but little else."[10] Currently dynamic music is very limited, despite the opportunity for a more interactive musical environment if pre-recorded music was less prevalent. This was the case in the past when MIDI-driven music was used. As with many constraints on the video game composer this was down to convenience. With early storage media being very limited, MIDI based text instructions for the hardware synthesisers that early consoles and computers used were more space efficient than recorded audio[11].

Using the definition of dynamic music set out by Jesper Kaae - "music which is in some way able to react to game-play, and/or is in some way composed or put together in real time by the computer"[12], the music created in this project consists of directions and parameters that are triggered as the game is played, and would be best represented on paper by some variant of graphic scoring – or some combination of traditional notation and additional instructions/directions.

Linear music, in contrast, is music that can stand alone independent of the game, as is currently seen in the majority of the video game music canon, with titles such as Final Fantasy and Metal Gear Solid which feature heavily orchestrated and cinematic music, sometimes even commissioning composers from the field of film and TV music[13]. Whilst this can achieve impressive results in linear games, which can be depended upon to deliver a specific scenario, or a set of trigger points, this becomes more difficult in

9 Villarreal (2009), p.8

10 Collins (2007), Loc 1835

11 Collins (2008), p. 50

12 Collins (2007), Loc 1024

13 Hyde-Smith, A. (2002) Interview with Harry Gregson- Williams. Music4Games. Retrieved December 6 2013 from http://www.metalgearsolid.net/features/interview-composer-harry-gregson-williams

free-form (also known as ‘open-world’, or ‘sandbox’) games, where games can only depend on the physical location of the player – the interactions and events in the game world are indeterminate.

One of the most important reasons for developing new approaches to creating music for video games is that the ratio of music to game play is currently, “no less than 1 to 40”,[14] and whilst storage mediums are allowing more music to be created, composers are still required to work in the same way, that is to “to create looping music that would remain interesting for the entire game”.[15] One way to make looped music remain interesting is to create responses to dynamically manipulate pre-composed music, using live data to change the sound mix, instrumental direction (dynamics, texture, timbre) or even the content (pitch, rhythm, tempo).

This both creates interactivity and also a sense of spontaneity – although pre-prescribed, the music creates the illusion of reaction to the player actions. This approach is the closest composers have for creating reactive music currently, and even still it is limited, as there is a small time frame between the player action and the expected musical response if the music is to be able to effectively track the player. The idea of variability is important to dynamic music as it is, if not able to create a completely unique music experience, efficient: “variability plays a significant role in dynamic game music both as a way to save computer memory and as a way to keep the music sound new and “fresh”. After all, computer games are most often designed to last for many hours, and so should the music”[16]

Analysis of video game music often borrows approaches and lexicon from film music discourse. Crossover between the two fields is demonstrated by the regularity of composers working between the two mediums. As an interactive medium, the theories of film music are inefficient for working with video games - simply transferring film-scoring theory, with it's inherent linearity will not work. For a composer, spotting a film is a concept that will not transfer into video game projects. However, approaching a

14 Collins (2007), Loc 1438

15 Villarreal (2009), p.8

16 Collins (2007), Loc 1170

video game as an interactive film score does present some potential – films make use of motifs, repeated material and other musical devices to express emotion and expand the depth of the visual image – which if combined with interactive audio technology, could transform the video game score into a real-time scoring environment. This is not usually achieved presently due to the reliance on pre-recorded audio, which loops audio identically each time. Repeating audio identically (re-triggering a static wave audio file) is problematic as the response to repetition forms an inverted-U curve. As familiarity with section of music grows, the relative preference of it follows until it peaks and begins to decline[17]. This makes the argument for using some repetition and reference within musical design (so the player does not encounter entirely new music on every occasion), but with a need for fluidity to create new and derived musical experiences and prevent the fatiguing of repeated exposure to the same audio.

Whilst for the non-interactive/narrative 'cut-scenes' a linear score poses no problems, as soon as the player has control of their character there is the potential of non-linearity. Some games eliminate this by creating a fixed path through an environment with a procedural series of tasks, but in open-world games, a player may explore the environment in any way their choose, encountering and executing tasks as they find them. This is described musically as 'multi-linearity'[18], and can be accommodated in the music by creating several 'branches' – audio options – that anticipate the different options on offer to the player.

However unless the music is created in real-time, they are still pre-composed cues and as such a composer would not be able to score the game in the way one would score a film in order to deal with this. Aleatoric principles are one way to solve this problem, but to execute it in detail would require the music and game-code to work together closer (currently music tracks are loaded in as a fixed sound file, to be triggered at certain points). Monkey Island 2 is an example of implementing a type of aleatoric composition into a video game, using the iMUSE system[19].

17 Hargreaves, p.34

18 Collins (2007), Loc 1273

19 Mackey, B. (June, 2012). iMUSE and the Secret of Organic Music. *1up.com*. Retrieved December 16, 2013 from http://www.1up.com/features/imuse-secret-organic-music

There are some parallels to the way that performing musicians approached early silent film music, when piano players in Nickelodeon's relied upon their repertory of music to sound-track films with no prior preparation and high turnover. However, players in silent film theatres often did not have the time or resources to prepare for each film, which is not the case for video game music composition as many parameters are known prior during the development process. The similarities are noted in Kaae's paper *Theoretical approaches to composing dynamic music for video games* "When Hollywood composers in the first half of the twentieth century laid the foundation for what could be called the classical film aesthetic, they seemed to struggle with the same problems as today's composers of dynamic music, particularly the problem of not knowing when things happen."[20]

From these observations, the conclusion of the early research was that the project should look to the post-modern composers that explored non-linearity in music, as this would offer the responsiveness to the player actions.

Project Notes

The two methods of composing explored are generative systems and aleatoric composition, triggered by real-time data taken from the games. Using Kart Racing Pro to demonstrate both systems, the first piece demonstrates a ambient-drone sound-scape inspired by the generative audio that is seen in *Spore* (noted during it's development for the music system which used Pure Data[21] and featured input from Brian Eno[22]). The other piece uses an aleatoric system to trigger stems, similar to the iMUSE system first seen in Monkey Island 2.

The first work for 0 A.D uses a modified version of the ambient-drone and a drum synthesiser modified in real-time by player and opponent data in a similar way. The

20 Collins (2007), Loc 1170.

21 Farnell, A. (November 12, 2007). Pd in video game Spore. *Pd-list archive* Retrieved December 16, 2013 from http://lists.puredata.info/pipermail/pd-list/2007-11/056307.html.

22 Wired.co.uk (September 28, 2012). Briano Eno on music that thinks for itself. Retrieved December 16, 2013 from http://www.wired.co.uk/news/archive/2012-09/28/brian-eno-peter-chilvers-scape.

second creates a type of synthesiser pad that is triggered by the players on screen actions.

All of the data is taken from these games using the same basic concept, I wrote pieces of code to export real-time data into a shared memory cell, which was offered by the Win32 API[23], and then accessed by Pure Data using a custom-written 'external' for the program. Access points for each game were the DLL extension function of Kart Racing Pro; a custom MetaMod : Source plugin for Counter-Strike : Source; and writing directly into the open-source Pyrogenesis engine (0AD). The data is transported into Pure Data patches and interpreted, eventually being fed into the synthesisers and other aspects of the patches written to create the music.

The shared memory cell solution was reached after exploring other techniques such as memory pipes or writing data out into a text file – which was my starting point, as I worked with Kart Racing Pro initially and this had the source code of a telemetry plug-in in it's development tool set. Using a text file proved too slow however, with both the game and Pure Data having to perform I/O access to the hard drive, so I moved onto handling the data in real-time – using the computers memory to handle the movement of the data rather than the hard disk.

In Kart Racing Pro, the original telemetry plug in presented an ideal template with which I could experiment with, and after some failure, I had a basic command line tool that could read data out of the shared memory cell in real-time. This was the most exposed game, as the developer PiBoSo had provided functionality for external DLL files to be loaded in by the main game engine, so my access was non-intrusive.

When working with the Half Life Source engine, I had more problems with the code which I approached in several ways, including a lot of work with the Source SDK code provided by Valve. Whilst this gave me a greater understanding to the inner workings of the code, it did not yield usable results in terms of moving data out of the game so eventually I settled on using an external scripting program, MetaMod : Source, to write a C++ plug-in that could access the game data and the push it into the shared memory.

23 Windows Dev Center, Retrieved December 30, 2013 from http://msdn.microsoft.com/enus/library/windows/desktop/aa366551(v=vs.85).aspx

In 0 A.D, an open-source video game currently under development by Wildfire Games, the code is built in two layers. The base engine, Pyrogenesis is written in C++, and most of the game logic is built upon that in JavaScript which allows the scripting to be worked on by a wider audience – including content creators. The separation created some problems for me, as the statistics are largely handled by the game in the JavaScript layer.

The project creates code for both the JavaScript scripting files and the C++ Pyrogenesis engine, to pull the statistics data and push it into a custom built message type, which is how the game passes information within it's different components. This is then accessed by a custom C++ component within the game code which establishes a shared memory cell and pushes the data into this.

"The thing that's hard about music for games is imagining how it's going to work in the game."[24] Michael Land

Whilst creating music for this project, this statement resonated. As the musical content was closely tied to the data in-game, often the best way to determine the success of a score or parameter modification was to run the game, leading to a lot of trial-and-error. However, once the code was written and the basic Pure Data patches were created, it became easier to work with the medium, and start to envisage the resulting end product.

Simple actions in game, such as the character location, are not dictated by a linear timeline, as one would expect in a film-composing situation. Instead these are placed into the hands of many thousands of players, whose actions are unpredictable. As such, by writing music that secedes to the player actions, these decisions of how to begin and end musical actions also defer to the in-game actions. Transitions become one of the hardest aspects to deal with in this type of interactive music. The way this is handled is similar to how the transitions in Monkey Island 2 were created - allowing the fragment to end, rather than creating hard cuts between pieces of music.

24 Collins (2008), p.51

How the project differentiates from other types of video game music is that the compositions are audio engines, built on a collection of synthesisers, rather than audio files. In Kart Racing Pro, drone synthesisers can be manipulated at every level in real-time, with particular focus on textural details. This is built on random number simulations, which are scaled by the game data passed into the Pure Data patch. Instrument dynamic levels, tempo and presence are also changeable. In 0 A.D the drone synthesiser is augmented by a drum machine which is also built out of basic audio objects in Pure Data. The pattern, texture, tempo and dynamics of this element is manipulated by Pure Data.

Example 1 - Kart Racing Pro (Ambient Sound-scape)

The first game included in the portfolio is 'Kart Racing Pro', a racing simulation designed to be a realistic tool to aid kart drivers training for driving in real life[25]. Whilst this is not a usual candidate for application of music, as fans of the genre want to emulate real life as closely as possible, the game offers parameters that react to both direct and indirect player actions. It is designed with an accurate physical model, and simulates parameters such as tyre temperature, engine temperature and also chassis dynamics such as kart pitch, roll and yaw. These data feeds, which are indirectly related to the players actions in the game (the inputs the player has are throttle, brake and steering values) can be manipulated to create interactive music that adapts to the player and each specific session. This could be used in to contribute to the training of the driver by presenting the user with more information. Where in real-life, the driver receives more sensory information; the game is limited to displaying external data (temperatures, for example) as visual data on an in-game dashboard. Manipulating the music to communicate some of this data is one way that it could be used to enhance the driver's awareness.

Arcade racing games often feature music, usually a popular music sound-track which has little interaction with the game other than beginning and ending with the session. In contrast to these games, racing simulations present more accurate input and environmental data. The user approach is also different as in a simulation game time is

25 Piboso. (2013) Kart Racing Pro Game Description, *kartracing-pro.com.* Retrieved December 17, 2013 from http://www.kartracing-pro.com

often spent 'learning' the nuances of the car and tracks, then utilising the skills that are learnt on circuit to improve lap times. In an arcade racing game such as Mario Kart, the emphasis is not on the drivers direct inputs in these games, rather how successfully they utilise the environment of the ‘circuit’, which features cartoon graphics on circuits laid out with ‘power-up’ items to either speed up the player or slow down their competitors. Additionally, the control inputs (steering, throttle, brake) in 'arcade' games are less precise, as simulators are designed with steering wheel peripherals and hardware in mind, whereas arcade racing games are designed to be played on game pads and keyboards - less precise controllers which require the game to aid the player, which changes the skills required to succeed.

Creating music that is reactive or dynamic is a common problem in the racing game genre that is avoided by developers using non-interactive, popular music-based soundtracks. Sometimes this linear pre-recorded music is manipulated at a basic level by DSP functions, such as in WipEout HD[26]. The first piece of music made for Kart Racing Pro consists of ambient/drone sound-scape that is shaped by the actions of the player. Taking cues from the real-time data that is fed into the Pure Data music patch (throttle, brake, steering values) as well as a group of environmental and derivative data (radiator/engine temperature, chassis loading – pitch/yaw/roll, world position, speed, fuel load) that can be affected by the player but not directly controlled.

26 “Music integration in Wipeout HD is achieved by applying audio effects that connect with the racetrack and gameplay. These include reverb when flying through tunnels (an effect that was already present in the original Wipeout), a hi-pass filter when jumping and characteristic changes in volume and sound texture when hit, in critical condition and upon enabling a shield.” Kayali, F. (2009) Pure Hardcore? wipEout HD and current game design. *Eludamos Journal for Computer Game Culture*. Retrieved December 17, 2013 from http://www.eludamos.org/index.php/eludamos/article/viewArticle/vol3no1-11/117

Figure 1 – Data fed into Pure Data

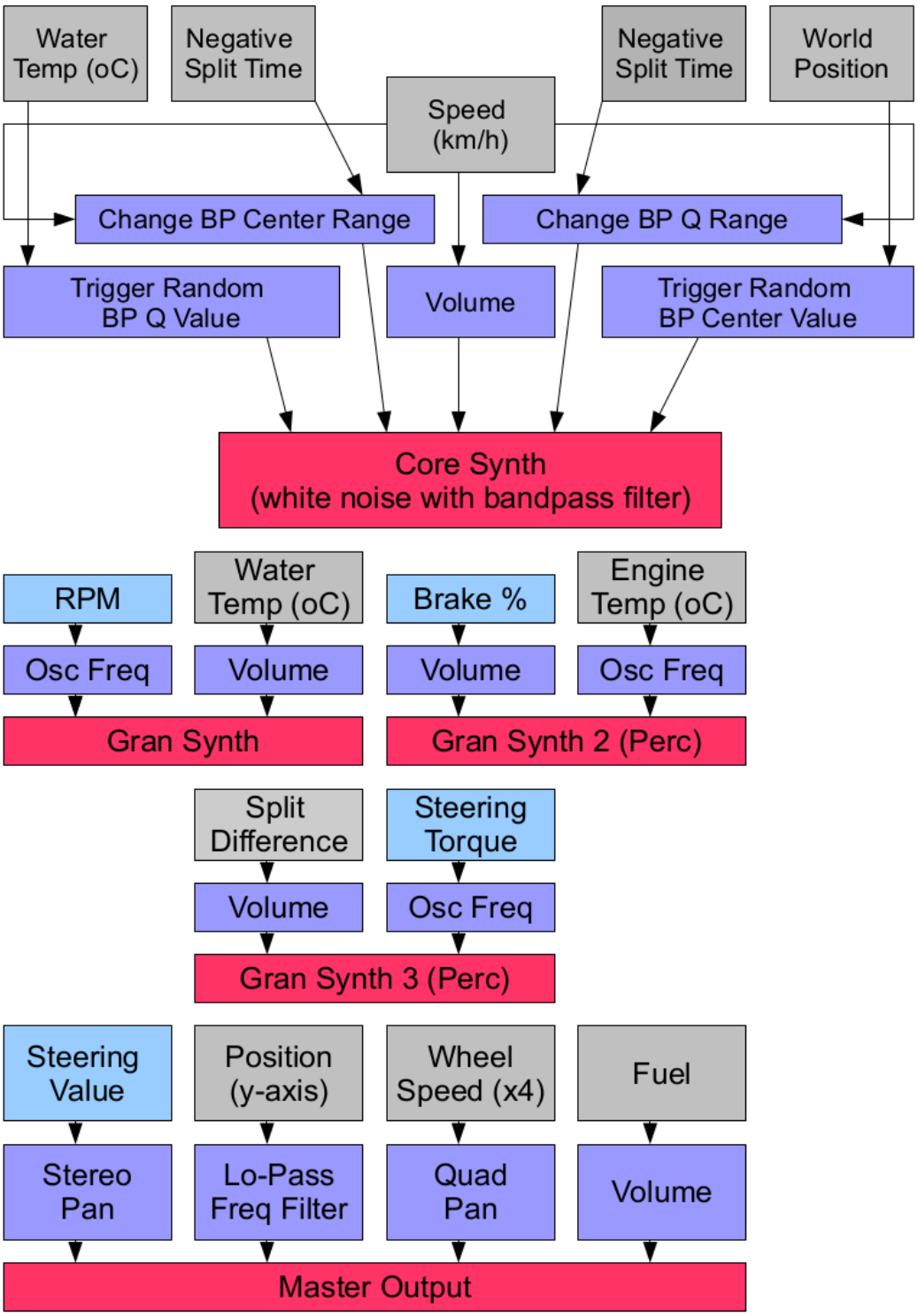
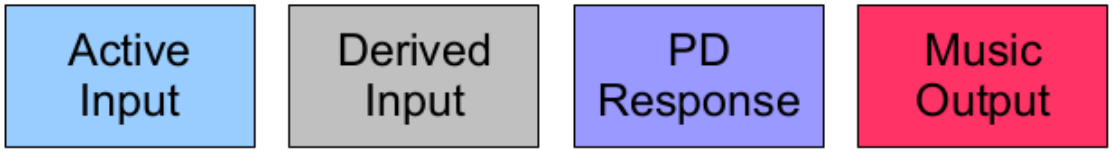


Figure 1 (Legend)



The patch for this piece uses the environmental data principally to manipulate the parameters in which the synthesisers operate. The temperatures the kart reports rise at low speeds and whilst coasting, and also depend on the weather conditions – which can either be chosen by the player or be dynamically generated every session. The reason that this data was used to change the patch rather than the direct inputs was to avoid turning the composition/patch into a pseudo-instrument, where the players game actions primarily have a direct effect on the music generated. This conclusion was reached by observing games that utilise direct input to execute music, games such as Rez and Flower.

Rez is a music genre game which depends on the player's interactions with the software to trigger musical fragments or samples. This is type of interaction between music and game play is suited to games that are abstract experiences – like the minimalist nature of Rez, which is set inside a super-computer and tells the story of defeating a virus within the system. It can utilise senses in a way that would be unsuitable if they were applied to a game that was more grounded in simulating real-life (for example, the street crime in America of Grand Theft Auto, or warfare in the many FPS games on the market). This is seen in Flower by the correlation between petal colour and the music triggered, where the game attempts to associate two senses together – sight and hearing.

In Flower, the player controls the environment to direct a flower petal through the level, interacting with other flowers that reactively add musical fragments to the audio backdrop. The game manipulates the volume dynamics and textural density of the audio, which are elements of interest to this project, and the composer was involved with the level design to make it work musically[27]. It relies on direct player interaction to trigger pre-recorded audio. Whilst this works for some games and genres, and is often used in music-orientated video games[28], with this approach the music is forced to the front of the player attention, which can become a distraction in other games.

27 "I spent a lot of time at ThatGameCompany so I had the opportunity to talk with the artists about the arrangement of flowers: tightening up lines of flowers or stretching them out, replacing red flowers with white flowers, and so forth so that the sound would work. " Interview with Vincent Diamante. Retrieved December 18, 2013 from http://www.gamesetwatch.com/2009/02/column_sound_current_audio_in_flow.php

28 Collins (2007), Loc 1299

Rez is a shooting game that takes place 'on rails', with the player having no direct control of their movement, as if being transported through the level environments by train. It takes the idea of music-orientation in video games further than most. Where most musical interaction is based on the idea of playing music samples depending on when the player hits a particular 'trigger', in Rez when the player shoots their gun, the gunfire is delayed and synced with the beat of the music. The player's interaction is subjugated to the beat of the music, and as such the game challenges the idea of a video game being played in real-time.

This is in contrast to other video games of the 'rhythm-action' genre, which Rez is also in a way part of. In other examples of the genre the player is expected to press buttons in time with the music. If the response is executed slightly out of time, the on-screen action reflects this, creating a failure/success model where the player is incentivized to improve, as it affects the flow of the music. Most video games are concerned with the real-time simulation of an environment (with varying degrees of realism). Rez challenges this accepted practice, and re-defines the purpose of music in a video game. The game play mechanic is unique, and is more akin to an interactive film than a normal video game.

These concepts are brought across in the project, but in a less direct way. The music created takes parameters generated by the player into consideration to avoid repetition, but by utilising derivative data it attempts to not create a pseudo-instrument music system. This is a benefit of rendering musical audio in real-time that is unlike Rez and Flower, which both rely on pre-recorded audio that is triggered in a particular sequence as the player interacts with the game. The musical material in the project is re-interpreted each session through the frequency filters of the synthesiser and volume fluctuations. In the game if the player executes an identical lap, the music is still not a perfect repetition. In contrast, if a player executed identical game actions in Rez or Flower, they would hear the same sound files being triggered.

Example Two – Counter-Strike : Source

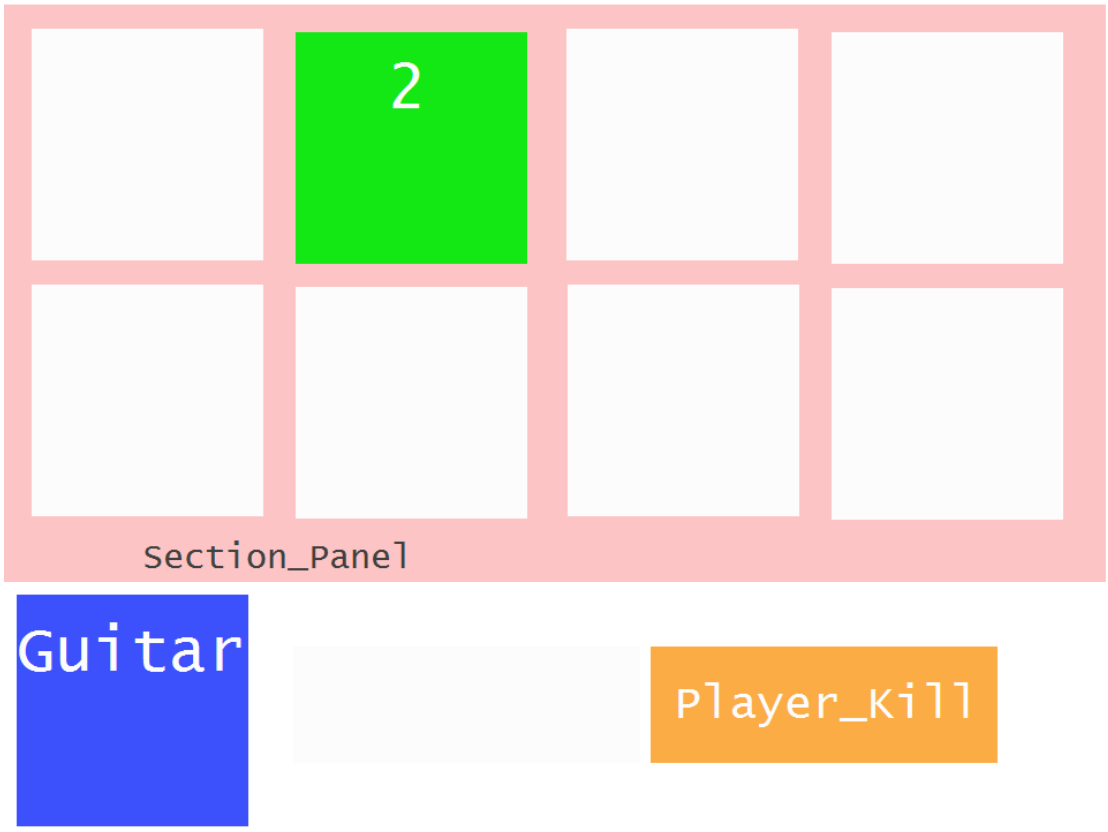


Figure 2 - The conductor patch, which tells the (guitar) player what section is active, the guitar is an active part and that the player_kill accent should be triggered.

Counter-Strike : Source was another game used for prototype compositions during the research. The music for this game was based on a system that could be used in a live performance, a method of demonstrating open-form composition. Whilst MIDI manipulation still relies upon sample-based instrument sound banks that are large and inefficient in terms of computer resources, in the future, this method of composition could be combined with virtual instruments (such as the PianoTeq software which uses mathematical modelling to create a virtual piano[29]). PianoTeq occupies 20 megabytes of disk space, whereas a sample library would require several gigabytes to store the amount of recordings that would be required to deliver a comparable fidelity of sound. There is also a difference in other system requirements between a modelled virtual instrument and a sample-based instrument – the minimum requirements for PianoTeq

29 About Pianoteq. *Pianoteq.com.* Retrieved December 7, 2013 from http://www.pianoteq.com/pianoteq.

are 256mb RAM and 1.8GHz CPU[30] (although users have managed to configure the software to run on a Netbook, which often have sub-1.0GHz clock speeds[31]). In comparison to this, the sample-based piano library "Synthogy Ivory II" requires 1.5GB RAM and 2.0GHz CPU[32]. Despite the virtual instrument's heavy loading on the CPU to compute the audio in real time, it appears from the comparison that swapping audio samples in and out also requires CPU load, in addition to the need for RAM and hard disk performance.

Influenced by the approach of games such as Monkey Island 2, open-form compositional methods used by Boulez, Stockhausen and improvisation systems such as John Zorn's game pieces, the work for this game is built out of fragments of music with indeterminate performance directions, and a conductor Pure Data patch which makes the decisions of what and how the players should perform, depending on what data is fed to it by the game. Information is taken about the game play such as player health, location, velocity, and the 'score' – which consists of the kill and death count of the user.

Open-form or indeterminate composition methods create no fixed end product and this mirrors the interactive medium of the video game itself, which is why it was of interest when exploring approaches to composing from the medium. Similar to the rational of Zorn's game pieces - "the exciting thing about that [open-form] music was its flexibility in terms of performance. It could be different every time."[33] Part of what is exciting as a performer are the nuances and minor differences that live performance allows for, and abandoning pre-recorded sound files allows the possibility of this aspect of live performance to be introduced to the video game score – the technology for creating music in real-time on a computer is available, but it hasn't been integrated fully with video games yet.

30 Modartt. (2014) PianoTeq FAQ, *pianoteq.com*. Retrieved May 25th, 2014 from http://www.pianoteq.com/faq?category=pianoteq_setting

31 PianoTeq Forum (2014) "Tips for Windows netbook users", *forum-pianoteq.com*. Retrieved May 25th, 2014 from http://www.forum-pianoteq.com/viewtopic.php?id=2501

32 Synthogy (2014) Ivory II Grand Pianos Description, *synthogy.com*. Retrieved May 25th, 2014 from http://www.synthogy.com/products/ivorygrand.html

33 Zorn, J. *Audio Culture.* (2008), p.197

Observation of Monkey Island 2 led to the design of the percussion track, which acts as the 'glue' holding together the other instruments in the piece. As the tempo is variable dependent on the game actions, a strong and simple percussion foundation anchors the other fragments and pieces of music. Monkey Island 2 also uses complementary music fragments in this area, at least rhythmically – which allows the melodic content to fit over the percussion loop, which underscores the entire area. The melodic content changes from room to room, but does not abruptly stop upon exiting the associated room. The use of percussion to dynamically underpin the music was something that first appeared in the Counter-Strike : Source music and was eventually carried over into the 0 A.D work, which uses a drum synthesiser as a central part of it's audio content.

The music for Counter-Strike was written with the idea of using live players to perform the eventual piece. The intention was to create a space that virtual instrumentation could occupy in the future of video game music production, utilising the relative flexibility of actual performers who can play more expressively than MIDI-based music can achieve presently[34], and be manipulated in real-time more efficiently than pre-recorded music. The music also embraced indeterminate or aleatoric principles for the very reasons that they were rejected by cinematic composers: “This new Boulezian model of musical structure has not generally been taken up by Hollywood film composers precisely because of its incompatibility with linear narrative, but its usefulness to game programmers struggling with non-linear forms seems self-evident”[35].

This concept introduced the need for player practicality and player-computer interaction, but was ultimately unsuccessful during the course of this project. Issues that eventually curtailed work on the piece were the difficulties of underscoring a first-person shooter - whereas the other games had relatively stable environments during gameplay, the FPS genre uses frantic and fast-paced mechanics. The original intention was also to use data from all the players in the server to build a picture of how many players were left on the map, using this data to create more tension in the music as the player lost team-mates. This was data which I did not successfully extract, which resulted in a limited amount of material that could be presented (compounded musically

34 Rowe, p. 110

35 Bessell, D. “What's that funny noise?” from Collins (2007), Loc 1193

by the decision to use traditional scoring). With time and more exploration, these aspects could be improved, but as the project continued other examples emerged as more promising from a compositional viewpoint.

The work on this game certainly informed other examples in this paper, the early experimentation on tempo mapping and percussive orientation are evident in the 0 A.D music presented below. Additionally, the time spent exploring the code, which was a larger task than in Kart Racing Pro, provided me with the experience I would need to work on the third game in a more efficient manner.

Example Three - Kart Racing Pro (Real-Time Remix)

Games in the sporting genre - football, hockey, extreme sports (skateboarding, snowboarding) – are generally sound-tracked with popular music that is licensed from the record label or artist. This provides some familiarity and cultural reference point for the audience and, particularly in extreme sports genre, aids in projecting a specific image which can be tailored by the music included. For example, the Tony Hawk's Pro Skater series of games is referenced for it's early use of licensed music, featuring a collection of punk rock tracks in it's first incarnation[36].

The disadvantage of this type of music in video games is that it is a finite length, and cannot be manipulated, as it will be provided to the game developers fully formed and as a complete recording. Games that use this type of sound track tend to present it as either non-diegetic music, or through some form of in-game diegetic sound-source like a radio, as seen in the Grand Theft Auto series. Even in games that feature a large amount of licensed music, such as Grand Theft Auto V, which includes 17 radio stations of around one hour of content each, cannot match the time the user may spend engaging with it. Trying to come up with a solution for this, whilst retaining the idea of using licensed music in a different way, the project uses Kart Racing Pro to re-interpret a composition by 'The Shadow Committee', an instrumental rock project.

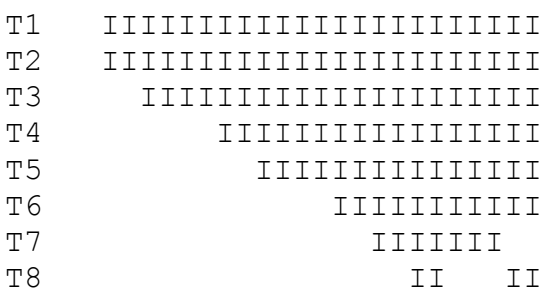
36 Elston, B. (2012, June 23) Game Music of the Day : Tony Hawk’s Pro Skater 2. *Gamesradar.com*. Retrieved December 10, 2013, from http://www.gamesradar.com/game-music-of-the-day-tony-hawks-pro-skater-2/

The idea of using licensed stems, or 'splits' is discussed in Collin's *Game Sound*, but suggests barriers to label and artist co-operation, “you may be able to obtain splits of tracks for developing adaptive audio, but the actual process may prove quite difficult”. However this quote comes from EMI back in 2004 - and the music industry has seen large changes in the last decade. Whilst the issues of creative control mentioned are still relevant, the increasing acceptance of remix culture by the music industry indicates that fears of piracy through remix stems have receded in recent years[37].

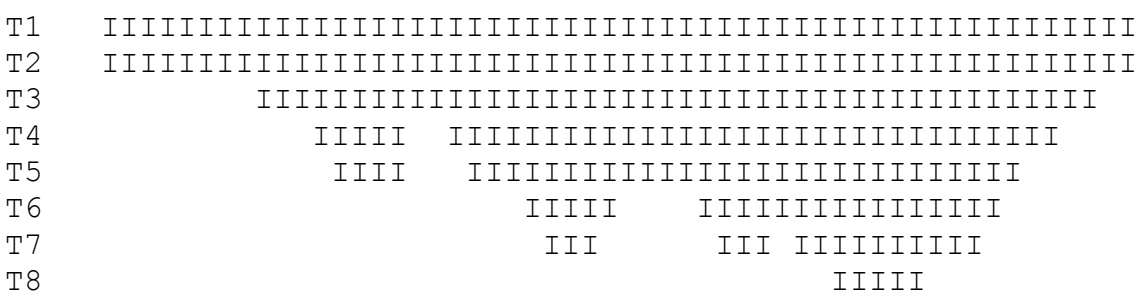
By working with the idea of creating a real-time remix, rearranging the music content another piece of work, the Pure Data patch for this piece re-works individual components in real-time, adding and removing textures and audio loops depending on the game actions. Whilst the linear version of the song runs for 4mins 42s and cannot be manipulated ‘vertically’ (adding and removing instrument tracks) mid-playback, the interactive version is built up of the melodic and percussive fragments and can play for an indeterminate amount of time.

Figure 3. Comparison of the structure between linear and dynamic versions of Deauville Grand Prix.

<u>Recorded Version</u>



<u>Dynamic Version (Early Version)</u>



37 Collins (2008), p.119

Dynamic Version (Video Version)

```
T1 IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
T2 IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
T3         IIIII   II
T4         IIIIIIIIIIIIII
T5                         IIIIIIIIIIIIII-21-IIIIIIIIIIIII
T6                           IIIIIIIIIIII-21-IIIIIIIIIIIII
T7                             IIIIIIIIII-21-IIIIIIIII
T8                                   IIII-21-IIII
```

As the three versions of the track show, the dynamic versions are longer, and also take longer to build up with peaks and troughs (particularly in the early version structure) – with the first two tracks being triggered alone. There is also a sense of momentum being built up in the structure of the dynamic piece, which is caused by the driver oscillating between setting quicker sector times (adding tracks to the mix) and slower times (losing tracks).

This approach allows the game to use recognisable content whilst re-interpreting it each time the game is played. In the Kart Racing Pro piece, the game sends information to the audio patch to tell the music which sector the driver is in, whether they have gone faster or slower than the previous lap, and the data that was sent to the original Kart Racing Pro patch (throttle, brake, steering values; environmental data). The 'live' data (throttle, brake, steering, environmental) is used to manipulate frequency filters[38] and dynamic controls of the individual stems, which also alters the sound each time it is played.

The idea of equalisation manipulation of a pop music sound track is used in snowboard game *SSX 3* (2003). The snowboard rider either races or performs tricks and jumps, and the sound-track is closely tied to the player actions – performing a large jump results in a hi-pass filter being added to the music.[39] The game is restrained by the amount of music it shipped with, and with limited adaptability it means despite introducing novel ways of treating audio, soon the listener fatigues due to the many repetitions heard

38 Video 3, filter application can be seen at 2:10; 2:30; 3:13

39 Kavasin, G. (2003, October 20) SSX3 Review. *Gamespot.com*. Retrieved 9th December 2013, from http://www.gamespot.com/reviews/ssx-3-review/1900-6077068/

during the game play[40]. SSX 3 attempts to address this too, by cutting up the songs into verses and choruses and repeating sections until the player completes their run (one track for one 'event'). The drawback of this approach, however, is that the loops are repetitive, the joins are audible and the game has to force some transitions in order to meet the rules of the music engine. For example, the end of the level triggers a hard-coded transition that doesn't wait for the currently playing fragment to end before cutting to the 'end' music.

Also included in the portfolio is a working example of a MIDI version of the same idea - which had some problems during it's creation but demonstrates how material can be re-imagined in a more adaptable way - as the Pure Data patch stores arrays of MIDI data which can be played forwards, backwards, frozen and also be manipulated at a sound level by changing the parameters of the synthesiser that is playing the notes in real-time too.

Example Four - 0 A.D.

The final pieces of the portfolio take on real-time audio synthesis with game of a different genre, and different viewpoint. Whereas the other two games in the portfolio are conducted in the first person, 0 A.D. is a real-time strategy with an omniscient viewpoint, where players build armies and fight. For this game, the music uses game data from both the player and the opponent to create a music landscape that evolves with the pace of the game.

This piece originally intended to synthesise in-game sounds in a granular synthesiser, to create a percussion section that would move with the progress of the session, as a response to the challenge of "the repetitive nature of sample-based playback"[41]. Whilst creating the Pure Data patch, however, creating a suitable solution appeared overly complex for the resulting sound, which was achieved more efficiently using white noise and filters. Once again, real-time data is utilised by the Pure Data patch, serving as both

40 Collins (2007), Loc 1835

41 Collins (2007) Loc 1835

a way of controlling the music created by the noise synthesisers, and also creating new musical accompaniment for each session.

The player resource data affects the ADSR profile of the individual synth drum sounds[42] (bass drum, snare and cymbal), and also the overall volume of the drums. The units the player creates act as a signpost for the drum kit to change it's sequencer pattern[43], and other events such as buildings created and lost trigger other temporary effects in the patch music. To compliment the player data powering the drum synthesiser in the piece, an ambient drone is also included, which uses resource and unit data from the computer-controlled opponent. By using a slowly developing volume swell to indicate the progress of the other player, an ominous presence is created, as the opponent grows stronger.

The close ties to each player collecting resources (the materials wood, food, stone, metal) was decided because it is one of the most important aspects of the game play, needed in order to build 'units' (soldiers and villagers) and buildings through which the player can engage with the game. To collect resources, the player must send a unit to collect them. To kill enemy units, the player has to send soldiers. The music is closely tied with this collection of resources and the creation of units. There does not appear to be another game that ties this type of statistical approach into it's music creation, but of the games that feature generative music, shaping musical output in real-time rather than simply triggering musical fragments at certain points, Spore uses Pure Data in the game engine to power a piece of software called 'The Shuffler' which "procedurally generates fragments for the soundtrack".[44]

The music created for this example still reflects the way that a composer approaches granular music, as noted by Barry Truax 'the composer functions not as an omniscient arbiter, but as the source of control messages that guide the overall process without directly determining it.""[45]. This has been a recurrent theme through the portfolio,

42 Video 5, 0:15 & 0:25 show the change in the sustain of the drum sounds.

43 Video 5, comparing the drum pattern in 1:15 and 2:55 shows the difference triggered by the population increase.

44 Sweetser, P. (2007). *Emergence in Games*, p.327

45 Truax, B. (1990). *Perspectives of New Music,* p.120-34.

repositioning the composer less as a direct creator of music, but more as a curator of sound, using the video game stimulus to shape the music to both the demands of the game and the interests of the composer.

So many real-time strategies use pre-recorded music that has little relevance to the action unfolding on screen, usually falling back on period music or music from the civilisations depicted, with some triggered audio warning when the player is under attack. This piece instead eschews cultural ties to the on-screen graphics completely, instead embracing the statistical layer of the game which can communicate what is happening.

An observation from working with the game code was the idea of utilising the data from the 'entities' of 0 A.D, which are the visual building blocks of the game - identifying units, buildings and other objects in game. Whilst the first piece uses player data that is already presented on the screen numerically, another way to work with the data the game presents would be to assess the entities within the view port of the game to generate a sound-scape which could change as the player moved around the map, exploring the use of consonance and dissonance depending on whether the objects the player sees on screen are friendly or their opposition[46].

Working from the original premise of using the on-screen data to explore music, the initial task in 0 A.D was to find how to move this data out of the game. Whilst my first idea was to return all the objects on the viewport, whilst looking through the code this seemed data intensive and also possibly impractical as rendering many sounds efficiently is resource intensive in Pure Data. The compromise from this was to work with the GetEntityState function, which returns the selected entity and any entity the mouse hovers over in game. This data is a reasonable approximation of the player focus, with the benefit of reducing the volume of the data returned in real-time.
From here, I intended to attach a unique sound to each entity ID using Pure Data, but found that the data flow was too fast. In addition, this required dynamic patching from Pure Data, as the number of entities in the game is unknown and changing. The

46 This can be seen in Video 6, with a player triggered consonant fragment at 0:40, and a dissonant fragment at 3:10.

compromise was to use entity ID class and distribute the sounds randomly. Using the entity ID, visibility, health and alert level (buildings only); I created a sixteen voice synthesiser in Pure Data that would be manipulated as the game was played.

In the Pure Data patch, a Fourier analysis was conducted of two notes, creating a consonant pad (of eight voices, based on the note C) and a dissonant pad (of four voices, based on the note C#). Ambient sounds were added (for the remaining four voices, based on bandpass filtered noise generators), which are triggered by the non-player objects (resources, wild animals)[47]. The result is an approximation of the initial idea, and could be extended to provide more permutations relatively easily.

Conclusions

The compositions presented in this project are basic in their musical nature, but intend to demonstrate the interactive potential of using real-time data flow to curate musical content. The work conducted provides a base which can be built upon, moving the data into Pure Data allows for non-coders to work with the same information and data to create more complex and intricate musical works, which I will continue experimenting with in different games.

The music demonstrated in these pieces sets out to communicate more information to the player, information to augment the visual image that is presented on the screen. In Kart Racing Pro, one of the most successful ways that this was achieved was in the real-time remix – which added and removed stems to the mix as the player achieved faster and slower sector times. Whilst play-testing this particular composition, it did create a flow to the session, building up and creating tension as the piece progressed, and stripping back the audio towards the end of a run. In 0 A.D, the game alluded indirectly to the pace of the opponent player's progress, without directly telling the player in a way that could be compared to their own efforts. This, in a statistics-orientated game was effective and important – giving the player a source of information without compromising the strategy element of the genre by revealing too much.

47 Video 6, 0:50

Although academic discussion of video game music is still relatively in it's infancy, the fact that Monkey Island 2 is still one of the first references is made, in terms of adaptability and dynamic music – something that was acknowledged when it was first released in 1991 - "[m]any game magazine reviews of the era mention the music and the iMUSE engine as selling factors for these games"[48]

During the research of this project, working with the code of the games presented future avenues such as connecting the audio engine into AI systems and other simulations that the game performs to render the main content, in order to give it some more intuitive flexibility. This could be achieved through the use of neural networks. First person games like shooting and adventure games rely on these types of AI/code systems to create states of alert/awareness in non-player characters. An AI controlled 'bot' sees the world through code rather than visually, seeing 'playable' areas and being able to scan it's field of view for friendly/enemy objects. Being able to tap into this code would create the potential for a powerfully adaptive music system, which could react to the world around it as fast as an AI character could and, in theory, therefore the player themselves.

Another way data could be used through music to help the player in racing games would be to change the audio when the kart is on the correct racing line – currently games are limited to painting big virtual lines/arrows on the circuit to indicate direction/racing lines. Using audio to present this information is a more subtle way of helping the driver, without potentially spoiling the visual environment, and would tie in with the idea of multi-modality[49] by reducing the amount of visual load on the player who has to process a lot of data very quickly whilst playing such games.

An opportunity for the future would be more creative uses of licensed popular music tracks – similar to how the second Kart Racing Pro piece manipulated the stems of a previously linear composition; engaging with remix culture could be a way of using popular music hooks, refrains and melodies. This current inflexibility is noted by some -

48 Collins (2008), p. 57

49 Collins (2013), p. 30

“there is limited adaptability inherent in most popular music”[50], however the music industry has started to embrace new modes of transmission, such as mash-up videos and other derivative work that is prevalent on YouTube and online, by releasing pop music stems for users to remix on their own. This has created an opportunity that could be used by video game producers to incorporate popular music in an interactive manner, with a 'live' remix that is created in real-time whilst the game is in motion.

“In the early days of video games, most composers of music were in fact programmers working on other aspects of the game”[51]. Originally, this was because they were the people with the skills to be able to create the code needed to make the music happen with limited tools and resources on hand. Whilst budgets and technology has improved over the years and tools have allowed external composers to become part of the process, detached from the actual game making process, the skills of interacting with the game code directly are valuable for the composer, as breaking down the barriers will ultimately lead to the possibility of a medium that is as expressive as their own creativity.

50 Collins (2008) p.119

51 Collins (2008) p.114

Glossary

Adaptive Music

Music which can react in some way, in real time to the the game whilst in progress. Also referred to as *Dynamic Music* or *Interactive Music*.

Aleatoric Music

Music which is composed to some extent, with some elements being left to chance (either during the creation or during the performance of the piece).

Diegesis

Diegesis in terms of music is the process of classifying whether a piece of music played back is diegetic (with a source within the depicted world) or non-diegetic (placed 'above' the world for the audience's benefit).

Game play

Game play refers to the way in which the user interacts with a game - the relationship between the player and the virtual world, it's rules and objectives.

Rail Shooter

A type of action-based video game where the player only has control over directing where to fire their virtual weapon.

Sandbox

A 'free play' game mode where the player has no specified objectives, and often significantly higher/unlimited resources.

Bibliography

Books

Collins, K. (2007). *From Pac-Man to Pop Music : Interactive Audio in Games and New Media.* [Kindle Version]. Aldershot, UK: Ashgate Publishing Limited.

Collins, K. (2008). *Game Sound : An Introduction to the History, Theory, and Practice of Video Game Music and Sound Design.* London: MIT Press.

Collins, K. (2013). *Playing with Sound : A Theory of Interacting with Sound and Music in Video Games.* Cambridge, MA : MIT Press.

Cox, C. & Warner, D. (2008). *Audio Culture : Readings in Modern Music* London, UK: Continuum.

Hoffert, P. (2007). *Music for New Media : Composing for Videogames,Websites, Presentations and other Interactive Media.* Boston, MA : Berklee Press.

Marks, A. (2009). *The Complete Guide to Game Audio*. Burlington, MA: Focal Press.

Rowe, R. (1994). *Interactive Music Systems : Machine Listening and Composing.* Cambridge, MA : MIT Press.

Winkler, T. (2001). *Composing Interactive Music : Techniques and Ideas Using Max*. Cambridge, MA : MIT Press.

Articles (Online)

Bencina, R. (2001). "*Implementing Real-Time Granular Synthesis*" (Accessed 02 Dec 2013)

Brodsky, W. (2002). "*The effect of music tempo on simulated driving performance and vehicular control*" Transportation Research Part F , (Accessed 02 Dec 2013)

Hargreaves, J. D. (1984) "*The Effects of Repetition on Liking for Music.*" Journal of Research in Musical Education, Vol. 32, No. 1. pp. 35-47 (Accessed 22 May 2014)

Villarreal, A. (2009). "*The Effects of Music on Gameplay and Immersion*" Masters Thesis, MIT , (Accessed 09 Jan 2013)

Video Games

EA Games. (2008). *Spore* [PC]

EA Sports. (2003). *SSX3* [PS2]

LucasArts Games. (1991). *Monkey Island 2 : LeChuck's Revenge* [PC]

PiBoSo. (2013). *Kart Racing Pro* [PC, Beta 10b edition, Nov 25 2013]

ThatGameCompany. (2009). *Flower* [PS3]

United Game Artists. (2001). *Rez* [PS2]

Valve. (2004). *Counter-Strike : Source* [PC]

Wildfire Games. (2013). *0 A.D.* [PC, Alpha, Build used December 2013]

Music

Zorn, J. (1984). *Cobra*

Stockhausen, K. (1957). *Klavierstuck XI*

Xenakis, I. (1954). *Metastasis*